IBM Security AppScan Source Utilities

V 9.0.3.7

用户指南

IBM Security AppScan Source Utilities

V 9.0.3.7

用户指南

目录

第 1 章 Ounce/Make 构建实用程序

Ounce/Make 是一种从使用 makefile 的构建环境自动将配置信息导入 AppScan® Source 的工具。使用 Ounce/Make 便无需手动从 makefiles 导入配置信息。

Ounce/Make 实用程序提供一个命令行界面，用来从 makefile（makefiles 提供从源文件构建可执行应用程序或库的方法）生成 AppScan Source 项目文件 (.ppf)。生成的 ppf 文件包含 AppScan Source 评估对应 makefile 负责编译的源代码所需的所有信息。Ounce/Make 生成 .ppf 文件后，您可以将项目文件导入或添加到 AppScan Source 中。

要启动此实用程序，请执行下列操作：

- 在 Windows 系统上，运行 <install_dir>\bin\ouncemake.exe（其中 <install_dir> 是 AppScan Source 安装位置），例如，在 Windows（32 位）上：

  ```
  C:\Program Files\IBM\AppScan Source\bin\ouncemake.exe
  ```

- 在 Linux 系统上，运行 <install_dir>/bin/ouncemake，例如：

  ```
  /opt/ibm/appscansource/bin/ouncemake
  ```

Make 是一种将程序的编译和链接（依此类推）自动化的工具（考虑模块的相互依赖性及其修改时间）。Make 从 makefile（此文件指定要构建的目标集、这些目标所依赖的文件以及生成这些目标要执行的命令）读取指令。

需求

要成功创建 AppScan Source 项目文件，必须在合适的环境中运行 Ounce/Make。以下列表详细列举了需求以使 Ounce/Make 成功运行。如果您不符合所有这些需求，那么 Ounce/Make 将失败。

- 运行 Ounce/Make 所通过的目录必须包含有效的 makefile。
- 构建环境必须能够发出将会成功的 make 命令。
- 运行 Ounce/Make 之前，运行 make clean 命令。可以在运行 Ounce/Make 之前显式运行 make clean，或者通过指定 - clean 选项来将其与 Ounce/Make 包含在一起。
- Ounce/Make 遇到的 makefile 在下列情况下不能包含硬编码绝对路径：
  - 调用其他 makefile 时的 make 可执行文件路径：

    例如，请不要引用路径 /usr/bin/make -f makefile.mk。在 makefile 中，通过 make 可执行文件或变量来引用 make。变量可以是 *make macro*、${MAKE} 或是在属性文件中指定的其他变量。
  - 编译源代码时的编译器可执行文件路径：

    例如 /usr/bin/gcc -I.. -DFOO -o myfile.o myfile.cpp
  - 链接对象文件时的链接程序可执行文件路径

    例如 /usr/bin/ld file1.o file2.o

- 在 #include 语句中。

  要使用 #include 语句，请将以下标志添加到项目文件作为配置选项：

  ```
  --remote_root <remote dir>
  ```

  其中，<remote dir> 定义远程目录的安装点。

  注：您只能指定单个 remote_root。#include 语句的所有硬编码路径都必须解析为单个安装点。
- 调用 make 时，请不要在命令行上为 make、编译器和链接程序可执行文件指定宏（例如 make CC=gcc LD=ld）。

Ounce/Make 支持内容

Ounce/Make 支持多个计算机平台、编译器和 make 版本。

平台

您可以在 AppScan Source 支持的所有平台上运行 Ounce/Make。

Make 版本

Ounce/Make 支持下列 make 版本：

- make：UNIX make 实用程序是一种用于管理和维护计算机程序的软件引擎工具
- gmake：GNU Make 是一种从程序的源文件来控制程序的可执行文件及其他非源文件的生成的工具
- nmake：NMAKE.exe（Microsoft 程序维护实用工具）是一种基于描述文件中的命令来构建项目的 32 位工具

编译器

Ounce/Make 支持下列编译器：

- GCC：GNU Compiler Collection。AppScan Source 支持 C 和 C++ 前端
- cl：cl.exe 是 Microsoft Visual C++ 的编译器

操作

Ounce/Make 通过 Make 进行操作来从 makefile 中抽取必要的配置信息和创建相应的 AppScan Source 项目文件 (ppf)。为此，Ounce/Make 要求您提供有关构建环境的特定信息，如使用的 make 版本。

使用属性文件描述构建环境（请参阅第 7 页的『Ounce/Make 属性文件』）。

运行 Ounce/Make

您可以从任何成功执行 make 命令的目录运行 Ounce/Make。在大多数情况下，ouncemake 可执行文件不在对其进行调用所在的目录中。因此，建议您将包含 ouncemake 的目录添加到 PATH 环境变量以启动调用 ouncemake。

例如，如果源代码和/或 makefile 驻留在目录 \development\srcdir 中，那么要运行 ouncemake，必须首先切换到该目录：

```
%cd \development\srcdir
```

然后运行 ouncemake：

```
%ouncemake <options>
```

注：为实现向后兼容性，仍然接受 pmake 命令。不过，在将来发行版中可能会除去 pmake。

命名项目文件

Ounce/Make 使用本主题中概述的约定来命名 AppScan Source 项目文件。

- 创建 AppScan Source 项目文件时，Ounce/Make 使用相对目录路径（从已对 ouncemake 进行调用的目录到 ouncemake 创建项目文件所在的目录）。
- 调用 ouncemake 所在的目录成为路径名中的第一个组件。
- 下划线替换所有路径分隔符，如反斜杠 (\)（在 Windows 上）和斜杠 (/)（在 Linux 上）。
- 当文件名长度超过操作系统限制时，Ounce/Make 从左边开始去除路径的组件，直至文件名长度符合系统命名约定为止。
- 在文件系统根目录（如 / 或 c:\）运行 ouncemake 时，ouncemake 会创建名为 root.ppf 的 AppScan Source 项目文件。

AppScan Source 将创建的 .ppf保存在 ppf 表示的 makefile 旁边的位置中。例如，如果运行 Ounce/Make 来创建单个项目文件，那么 AppScan Source 会将 ppf 保存在已对 Ounce/Make 进行调用的目录中。请参见第 15 页的『示例 2：带递归选项的 Ounce/Make』，以查看在多项目方式中创建的 ppf 文件。

注：如果目录中有多个 makefile，那么 Ounce/Make 在此目录中仅创建一个 .ppf 文件。

示例 1

此示例说明创建的 ppf 文件，其中路径分隔符替换为下划线。

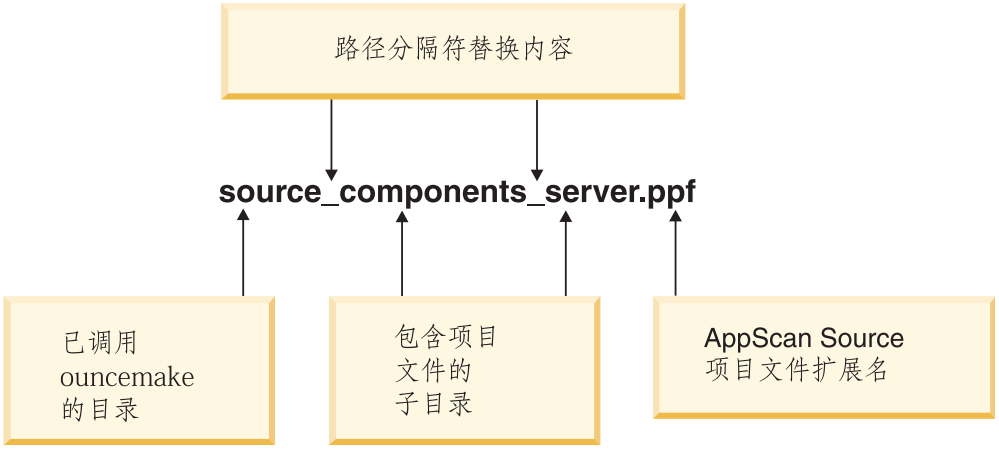
从以下目录调用 ouncemake：

```
C:\development\source
```

在执行期间，Ounce/Make 在以下目录中创建 AppScan Source 项目文件：

```
C:\development\source\components\server
```

ppf 的名称为 source_components_server.ppf：



示例 2

Microsoft Windows 和 Linux 会对路径和文件名进行限制。这些操作系统将字符数限制为 255。示例 2 显示文件名超过路径长度限制的情况。

用户从以下目录调用 Ounce/Make：

```
C:\path1\path2\path3\path4\path5\development\source
```

在执行期间，Ounce/Make 在以下目录中创建 AppScan Source 项目：

```
C:\path1\path2\path3\path4\path5\development\source\components\server
```

如果文件名最多可为 25 个字符，那么由于路径限制，生成的文件名为：

```
components_server.ppf
```

显式命名项目

您可以选择性地使用 OUNCE_PROJ_NAME 环境变量来显式指定新创建的项目的名称。与任何其他环境变量一样，定义 OUNCE_PROJ_NAME=value 环境变量。

如果未设置，那么按现有项目命名约定来生成名称。

例如，要从命令行将项目名称指定为 MyMakeProject，请使用以下命令：

```
$(MAKE) $(MAKE_ARGS) all OUNCE_PROJ_NAME=MyMakeProject
```

Ounce/Make 输出

运行时，Ounce/Make 会生成输出来通知您特定事件。

下列事件会触发输出消息：

- 项目创建
- 调用 make
- 错误

项目创建

当 Ounce/Make 创建 AppScan Source 项目文件时，它将输出包含 ppf 名称的消息。以下示例是一条 Ounce/Make 项目创建输出消息：

```
Created AppScan Source project <project_name>.ppf in directory <directory>.
```

调用 make

当 Ounce/Make 调用 make 可执行文件时，将会列显可执行文件名和选项。以下样本显示 Ounce/Make 在调用 make 时生成的输出。

```
/usr/bin/gmake -f makefile.mk release
```

错误

如果在执行期间发生错误，那么 Ounce/Make 将列显错误消息来描述错误。如果在 make 可执行文件运行时发生了错误，那么显示的错误消息是 make 可执行文件错误。如果错误特定于 Ounce/Make，那么会显示相应的错误消息来描述 Ounce/Make 错误条件。

Ounce/Make 命令语法和 make 选项

Ounce/Make 支持在其运行时可更改其行为方式的多个选项。

您可以在属性文件中设置这些选项（请参阅第 7 页的『Ounce/Make 属性文件』以了解更多详细信息）或在命令行上包含它们。如果在属性文件中设置这些选项，那么每次运行 Ounce/Make 时不必在命令行上指定它们。

概要

Ounce/Make 支持以下语法：

```
ouncemake [options] [-- make_options]
```

选项

连字符 (-) 必须在所有选项之前。您必须单独指定选项；不能将其并置在单个连字符后。例如，命令：

```
ouncemake -sr
```

不是受支持的语法，但是您可以运行：

```
ouncemake -s -r
```

注：各选项必须以空格分隔。

运行 Ounce/Make 时，可以使用简略选项或完整单词。

下表包含描述各选项的列。

- **选项**：标识在被调用时 Ounce/Make 将理解的选项。
- **缺省值**：如果适用，说明在不指定选项时 Ounce/Make 缺省情况下如何操作。
- **描述**：使用此选项时的 Ounce/Make 行为。

选项	未指定选项时的缺省值	描述
-a <application_name> -application <application_name>	关闭	当指定时，Ounce/Make 会创建名为 <application_name>.paf 且包含 Ounce/Make 所创建的所有项目的应用程序文件。此文件创建在 ouncemake 运行所在的目录中。
-b -build	关闭	在收集 make 选项时执行构建。 此选项与 Cygwin 不兼容。
-r -recursive	非递归	Ounce/Make 以递归方式跟踪对其他 makefile 的所有调用。例如，如果某个 makefile 存在于源代码树的根位置以调用子目录中的所有 makefile，那么在包含根 makefile 的目录中调用 ouncemake -r 将导致 Ounce/Make 跟踪对子目录 makefile 的调用。
-nr -non_recursive	非递归	Ounce/Make 不以递归方式跟踪对其他 makefile 的调用。
-s -single_project	多项目方式	单项目方式。在单项目方式中时，Ounce/Make 在对其进行调用的目录中仅生成单个项目文件。 如果未指定，那么表示 Ounce/Make 处在多项目方式中。
-ns -non_single_project -m -multiple_project	多项目方式	多项目方式。在此方式中，Ounce/Make 针对其遇到的每个 makefile 都会在各目录中生成 AppScan Source 项目文件。
-nv -non_verbose -q -quiet	非详细方式	非详细方式。Ounce/Make 仅输出其自己的输出消息。Ounce/Make 禁止从 make 进行输出。
-v -verbose	详细方式	详细方式。Ounce/Make 将 make 及其自己的输出输出到标准输出。
-l log_level	1（关闭）	1 到 10。10 提供最多记录。

选项	未指定选项时的缺省值	描述
-c <clean_command> -clean <clean_command>	关闭	当指定时，Ounce/Make 将 <clean_command> 解释为命令并执行此命令。 <clean_command> 应是用户通常会执行以进行清除的命令。例如，make clean 是执行清除的常用命令。请注意，必须使用引号将命令引起来。 由于 Ounce/Make 在运行之前需要进行清除，因此如果不指定此选项，那么会出现提示，询问是否要继续。
-nc -no_clean	关闭	告诉 Ounce/Make 不要运行清除，并且不显示提示来提醒将**不**运行清除。
-p <properties_file>	不适用	允许用户指定属性文件以供 Ounce/Make 使用。<properties_file> 必须是 Ounce/Make 应使用的属性文件的绝对路径。
-? -h -help	不适用	Ounce/Make 选项的帮助。

Ounce/Make 属性文件

Ounce/Make 属性文件 (ouncemake_properties.xml) 是 XML 文件，其中包含有关构建环境的详细信息。当 Ounce/Make 运行时，它基于搜索路径或 -p 选项查找属性文件。属性文件必须指定在构建环境中使用的 compiler 以及 linker 和 make 元素。

要了解与属性文件中指定的必需和可选元素相关的信息，请参阅第 8 页的『Ouncemake 属性文件元素』。

运行 Ounce/Make 时，如果在命令行上不指定 -p 选项，那么 Ounce/Make 会仔细检查其搜索路径来查找 ouncemake_properties.xml 文件（请参阅第 5 页的『Ounce/Make 命令语法和 make 选项』以了解关于在命令行上指定选项的信息）。Ounce/Make 搜索路径与平台有关。

Microsoft Windows 搜索路径为：

- 当前工作目录
- <data_dir>\config（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）

Linux 搜索路径为：

- 当前工作目录

- <data_dir>/config（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）
- 用户的主目录
- /etc

注:

- 在 Linux 上，如果不指定 -p 选项而且属性文件在搜索路径中不存在，那么 AppScan Source 将在当前工作目录中自动创建一个属性文件。
- 为实现向后兼容性，Ounce/Make 会识别 pmake_properties.xml。如果 pmake_properties.xml 和 ouncemake_properties.xml 均存在于同一目录中，那么 ouncemake_properties.xml 优先。

Ouncemake 属性文件元素

OuncemakeProperties（属性文件的根元素）包含本主题中所列的元素。

所需元素:

- Compiler
- Linker
- Make

可选元素:

- MakeOptions
- Options
- GlobalProjectOptions
- FileOptions
- Executable
- MountRoot

Compiler

Compiler 元素指定构建环境中使用的编译器可执行文件。此元素的值必须是可执行文件的绝对路径。此元素包含一个可选属性 macro，它指定当构建特定于编译器路径的值时存储编译器可执行文件的变量的名称。如果不指定 macro 属性，那么缺省情况下 Ounce/Make 将 macro 属性设置为 CC。属性文件可包含多个 Compiler 元素，但是必需至少一个 Compiler 元素。

macro 属性的值在所有 Compiler 元素中必须唯一。例如，不能多次将 CC 列为 macro 属性。

示例

```
<Compiler macro=CXX>/usr/bin/gcc</Compiler>
```

描述

CXX：makefile 通过 CXX 宏引用此编译器。

/usr/bin/gcc：告诉 Ounce/Make 您使用的是 /usr/bin/gcc 编译器。

Linker

Linker 元素指定构建环境中使用的链接程序可执行文件。此元素的值必须是可执行文件的绝对路径。属性文件可包含多个 Linker 元素，但是必需至少一个 Linker 元素。

此元素包含一个属性 macro，它指定当构建时存储链接程序可执行文件的变量的名称。如果不指定 macro 属性，那么 LD 为缺省 linker macro。

macro 属性的值在所有 Linker 元素中必须唯一。例如，不能多次将 LD 列为 Linker 属性。

示例

```
<Linker macro=LD>/usr/bin/ld</Linker>
```

描述

LD：告诉 Ounce/Make 您使用的是 LD 宏

/usr/bin/ld：告诉 Ounce/Make 您使用的是 /usr/bin/ld 链接程序

Make

Make 元素指定构建环境中使用的 make 可执行文件。此元素的值必须是可执行文件的绝对路径。属性文件可包含多个 Make 元素，但是必需至少一个 Make 元素。

此元素包含一个属性 macro，它指定当构建时存储 make 可执行文件的变量的名称。如果不指定 macro 属性，那么缺省情况下 Ounce/Make 将属性设置为 MAKE。

make macro 属性的值在 Make 元素中必须唯一。

示例

```
<Make macro=MAKE>/usr/bin/make</Make>
```

描述

- MAKE：告诉 Ounce/Make 您使用的是 MAKE 宏。
- /usr/bin/make：告诉 Ounce/Make 您使用的是 /usr/bin/make make

MakeOptions

MakeOptions 元素指定要传递给 make 可执行文件的选项。此元素是可选的，并且在属性文件中不能多次出现。

示例

```
<MakeOptions>-f makefile.mk release</MakeOptions>
```

描述

选项：

```
-f makefile.mk release
```

传递给 make 可执行文件。

Options

Options 元素指定调用时要传递给 Ounce/Make 的选项。

此元素提供特定 Ounce/Make 命令行选项的备用，如第 5 页的『Ounce/Make 命令语法和 make 选项』中所述。

这些属性使用以下语法：

```
<option> = <true | false>
```

Options 元素及其属性不是必需的。

Options 元素可能包含下列属性：

属性	描述
recursive	布尔值。true 或 false。 值为 true 时，暗指 -r 命令行选项。
single_project	布尔值。true 或 false。 值为 true 时，暗指 -s 命令行选项。
verbose	布尔值。true 或 false。 值为 true 时，暗指 -v 命令行选项。
clean	使用引号 (") 引起来的字符串值，如 "make clean"。 设置时，暗指 -c 命令行选项。值应是进行清除要执行的命令。例如 gmake clean。
build	布尔值。true 或 false。 在收集 make 选项时执行构建。 **注：**与 Cygwin 不兼容。
应用程序	字符串值。设置时，暗指 -a 选项。指定的值应是您需要的应用程序名称。
no_clean	布尔值。true 或 false。 告诉 Ounce/Make 不要运行清除，并且不显示提示来提醒将不运行清除。

注：Ounce/Make 使用属性文件中设置的选项。不过，如果运行带有命令行上包含的选项的 Ounce/Make，那么这些选项优先于属性文件中设置的选项。如果运行不带命令行上的选项的 Ounce/Make，那么 Ounce/Make 将应用属性文件选项。

示例

以下是属性文件中利用所有属性的示例行：

```
<Options recursive="true" single_project="false" verbose="false"
clean="nmake.exe clean" no_clean="false"></Options>
```

描述

- recursive="true"
- single_project="false" 指导 Ounce/Make 在多项目方式中操作。在多项目方式中操作时，recursive 属性也应设置为 true。
- verbose="false" 关闭详细程度。

- clean="nmake.exe clean" 清除构建环境。
- no_clean= "false" 告诉 Ounce/Make 运行清除命令，从而禁止消息声明将运行清除。

GlobalProjectOptions

GlobalProjectOptions 元素为项目中包含的所有文件指定项目级选项。GlobalProjectOptions 元素及其属性不是必需的。

以下列表描述 GlobalProjectOptions 元素的属性：

- include_paths：字符串值。要应用于项目中所有文件的包含路径列表（以分号分隔）。
- macros：字符串值。要应用于项目中所有文件的宏列表（以分号分隔）。
- compiler_options：字符串值。要应用于项目中所有文件的编译器选项列表（以空格分隔）。请不要在此处指定包含路径和宏。

示例

以下是属性文件中利用所有属性的示例行：

```
<GlobalProjectOptions include_paths="/usr/include;/usr/local/include"
macros="DEBUG;WIN32" compiler_options="-f non-const-strings"/>
```

描述

- include_paths="/usr/include;/usr/local/include" 将路径添加到项目配置的"包含"部分。
- macros= "DEBUG;WIN32" 将 DEBUG 和 WIN32 宏添加到项目配置的"宏"部分。
- compiler_options= 将编译器选项添加到项目配置的"选项/命令行"部分。

FileOptions

FileOptions 元素允许为具有特定扩展名的文件指定包含路径、宏及其他编译器选项。您可以多次使用 FileOptions 来为具有不同扩展名的文件指定不同选项。例如，如下所示，如果您具有同时包含 C 和 C++ 文件的项目，请创建两个 FileOptions 元素，每个文件类型一个元素。

以下列表描述 FileOptions 元素的属性：

- extensions：字符串值。分号分隔的文件扩展名列表。每个具有与此列表中扩展名匹配的扩展名的文件都会获取此属性指定的选项。如果文件扩展名应用于出现的多个 FileOptions 属性，那么 Ounce Make 属性文件中出现的第一个此属性优先。
- compiler_options：字符串值。要应用于所有具有指定扩展名的文件的编译器选项列表（以空格分隔）。请不要在此处指定包含路径和宏。
- include_paths：字符串值。要应用于所有具有指定扩展名的文件的包含路径列表（以分号分隔）。
- macros：字符串值。要应用于所有具有指定扩展名的文件的宏列表（以分号分隔）。

示例

下列 FileOptions 示例显示如何配置 Ounce Make 属性文件，以将正确的选项应用于 C 和 C++ 文件。

带有 extensions="c" 的 FileOptions 元素将其其他属性值仅应用于具有 c 扩展名 <filename.c> 的文件。带有 extensions="cpp;cxx" 的 FileOptions 元素会将其其他属性值仅应用于具有 cpp (<filename.cpp>) 或 cxx (<filename.cxx>) 扩展名的文件。

```
<!-- g++ options for C files -->
<FileOptions
  extensions="c"
  compiler_options="-gcc_linux_i386"
  include_paths="/usr/local/include;
 /usr/lib/gcc-lib/i386-redhat-linux/3.2.3/include;
 /usr/include"
  macros=""/>

<!-- g++ options for C++ files -->
 <FileOptions
  extensions="cpp;cxx"
  compiler_options="-g++_linux_i386"
  include_paths="/usr/include/c++/3.2.3;
 /usr/include/c++/3.2.3/i386-redhat-linux;
 /usr/include/c++/3.2.3/backward;/usr/local/include;
 /usr/lib/gcc-lib/i386-redhat-linux/3.2.3/include;
 /usr/include"
  macros="__GNUG__=3" />
```

描述

extensions="c" 和 extensions="cpp;cxx"

指定这些文件选项适用的文件扩展名。

Executable

Executable 元素允许在构建环境中使用任何可执行文件（compiler、linker 或 make 除外）。此元素的值必须是可执行文件的绝对路径。Executable 元素包含一个属性 macro，它指定当构建时用于存储可执行文件名的变量的名称。 macro 属性的值在所有 Executable 元素中必须唯一。

属性文件可包含任何数量的 Executable 元素。 Executable 元素是可选的，并且仅当您在构建环境中使用其他可能导致 Ounce/Make 失败的可执行文件时才必需。通过识别这些可执行文件，Ounce/Make 可防止失败。

示例

以下是属性文件中的示例行：

```
<Executable macro=ARCHIVE>/usr/bin/ar<Executable>
```

描述

此示例向 Ounce/Make 表明构建使用的是 /usr/bin/ar 可执行文件，并且 makefile 通过 ARCHIVE 宏引用此可执行文件。

MountRoot

如果您是从远程系统评估源代码，那么必需 MountRoot 元素。MountRoot 告诉 Ounce/Make 将指定的安装点追加到其遇到的所有绝对包含路径之前。

通过指定 MountRoot，Ounce/Make 将 --remote_root 选项隐式添加到生成的 ppf 中的编译器选项列表。

MountRoot 仅在 Linux 上有效。

示例

以下是属性文件中的示例行：

```
<MountRoot>/mnt/mount_point/usr/include</MountRoot>
```

描述

如果指定的包含路径之一为 /usr/include（如 gcc -I/usr/include ...），那么 Ounce/Make 会将安装点追加到 /usr/include 之前，以便生成的 ppf 文件中存储的路径包含正确的路径，如 /mnt/mount_point/usr/include。

样本属性文件

AppScan Source 的安装在 <data_dir>\config 目录（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）中包含以下样本属性文件：

- SampleOuncemakeProperties-Windows.xml
- SampleOuncemakeProperties-Linux.xml

您可以修改这些文件以创建定制属性文件。如果您不熟悉 XML，请使用这些样本作为模板来创建您的文件。

示例

此部分描述使用 Ounce/Make 的三种方法。

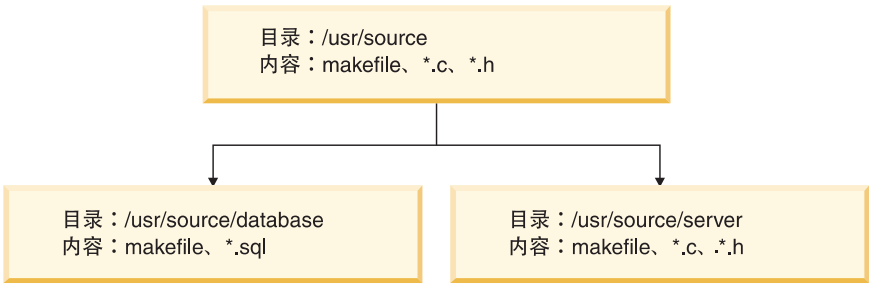
第 14 页的『示例 1：不带选项的 Ounce/Make』 说明不带选项的 Ounce/Make，仅根据要调用的 Ounce/Make 所在目录中的 makefile，创建一个 AppScan Source 项目文件。

第 15 页的『示例 2：带递归选项的 Ounce/Make』 使用带 -r（递归）选项的 Ounce/Make，指示 Ounce/Make 以递归方式操作并跟踪对其他 makefile 的所有调用。

在第 15 页的『示例 3：带单项目和递归选项的 Ounce/Make』 中，Ounce/Make 使用 -r（递归）和 -s（单项目）选项，以基于 Ounce/Make 遇到的所有 makefile 的递归处理来创建单个 AppScan Source 项目文件。

目录结构和文件

所有三个示例都使用相同的目录结构和文件：



此图显示包含 makefile 和源文件的根目录 (/usr/source)。/usr/source 目录包含两个子目录：/usr/source/database 和 /usr/source/server。/usr/source/database 目录包含一个 makefile 和多个 SQL 文件。/usr/source/server 目录包含一个 makefile 和多个源文件。

此示例对三个 makefile 进行下列假设：

- /usr/source 中的 makefile 构建 /usr/source 中的源文件并且调用 /usr/source/database 和 /usr/source/server 中的 makefile。
- /usr/source/database 中的 makefile 将 SQL 文件导入到数据库中。
- /usr/source/server 中的 makefile 构建 /usr/source/server 中的源文件。

示例 1：不带选项的 Ounce/Make

此示例说明以非递归方式使用 Ounce/Make。在非递归方式中，Ounce/Make 仅查看对其进行调用的目录中的 makefile。如果原始 makefile 包含对其他 makefile 的调用，那么 Ounce/Make 将忽略这些调用。

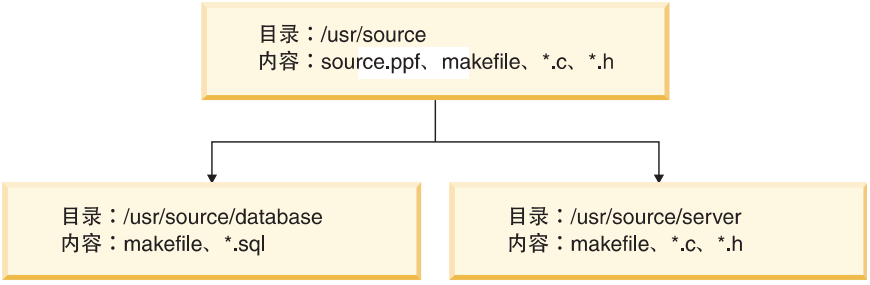
此示例从 /usr/source 运行 Ounce/Make。

请参阅『目录结构和文件』，以了解此示例使用的目录结构和文件的图形说明。

命令

```
ouncemake
```

下图显示 Ounce/Make 运行后的目录内容：



Ounce/Make 运行后，/usr/source 此时包含名为 source.ppf 的 AppScan Source 项目文件。此项目文件包含用以评估 /usr/source 中所有源文件的全部必要信息。在非递归方式中，Ounce/Make 会忽略 /usr/source 中 makefile 内的调用（调用 /usr/source/database 和 /usr/source/server 中的 makefile）。

示例 2：带递归选项的 Ounce/Make

通过 -r 选项，Ounce/Make 以递归方式操作（跟踪对 makefile 内包含的其他 makefile 的调用）。由于多项目文件方式为缺省方式，因此在与 -r 选项配合使用时，Ounce/Make 会为遇到的每个编译源代码的 makefile 创建 AppScan Source 项目文件。

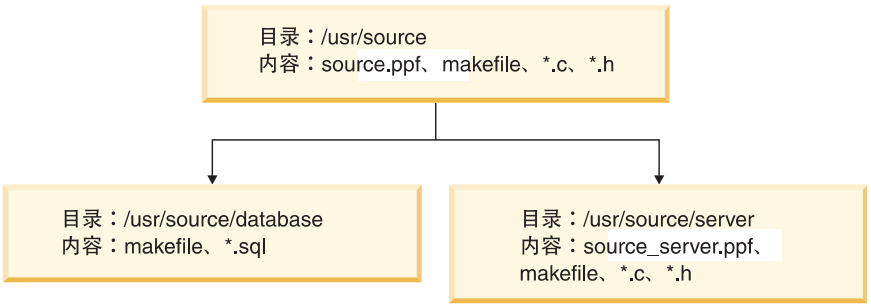
请参阅第 14 页的『目录结构和文件』，以了解此示例使用的目录结构和文件的图形说明。

命令

```
ouncemake -r
```

-r（递归）选项指示 Ounce/Make 跟踪对其他 makefile 的 makefile 调用。有关递归选项的更详细描述，请参见第 5 页的『Ounce/Make 命令语法和 make 选项』中的表。

下图显示 Ounce/Make 运行后的目录内容：



在此示例中，Ounce/Make 在 /usr/source 和 /usr/source/server 中创建 AppScan Source 项目文件。由于 /usr/source 中的 makefile 调用了 /usr/source/database 和 /usr/source/server 中的 makefile，因此 Ounce/Make 会查看这些 makefile 是否编译了源代码。

就 /usr/source/database 中的 makefile 而言，Ounce/Make 确定此 makefile 不编译源代码；因此，它未创建 AppScan Source 项目文件。不过，Ounce/Make 确定 /usr/source/server 中的 makefile 已编译该目录中的源文件，并且因此为 /usr/source/server 中的源文件生成了 AppScan Source 项目文件。

示例 3：带单项目和递归选项的 Ounce/Make

示例 3 说明在单项目方式中以递归方式使用 Ounce/Make。Ounce/Make 针对遇到的所有 makefile 编译的源代码组合生成单个 AppScan Source 项目文件。

请参阅第 14 页的『目录结构和文件』，以了解此示例使用的目录结构和文件的图形说明。

从 /usr/source directory 运行以下命令：

命令

```
ouncemake -r -s
```

-r（递归）选项指示 Ounce/Make 跟踪对其他 makefile 的 makefile 调用。有关递归选项的更详细描述，请参阅第 5 页的『Ounce/Make 命令语法和 make 选项』中的表。

-s 选项指示 Ounce/Make 在对其进行调用的目录中仅生成一个 AppScan Source 项目文件，而不是为遇到的每个 makefile 都创建新项目。

下图显示 Ounce/Make 运行后的目录内容。

/usr/source 中存在单个 AppScan Source 项目文件。此 AppScan Source 项目文件包含 /usr/source 和 /usr/source/server 中整个源代码的配置信息。

第 2 章 AppScan Source 命令行界面 (CLI)

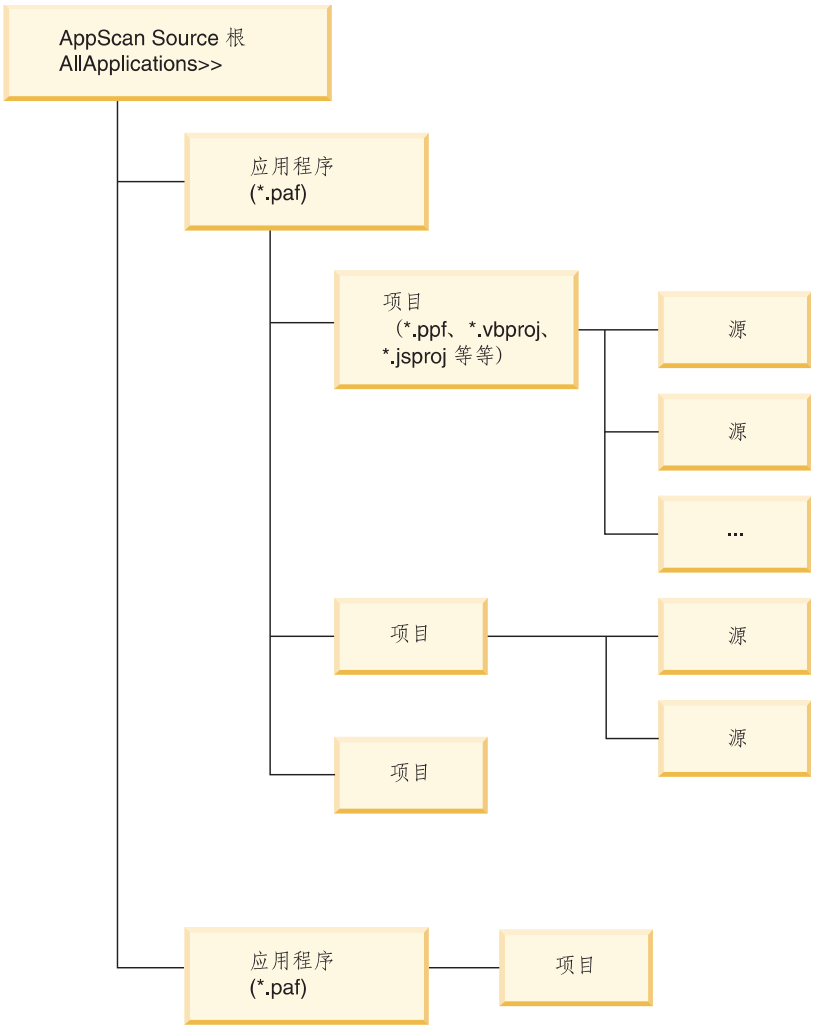
CLI 是核心 AppScan Source 功能的界面。

CLI 支持从命令行执行 AppScan Source 功能，或者运行脚本文件以记录在交互式会话期间发出的命令。在运行时，CLI 提供用户反馈作为向控制台或日志文件（可选）的输出。

对象和上下文

AppScan Source 对象树

对象树显示 AppScan Source 对象的基本结构。您可以扫描树结构内的任何对象（应用程序、项目或源文件）。



活动对象和当前上下文

许多 AppScan Source 命令行界面 (CLI) 命令仅在当前上下文中起作用；即在当前目录或在指定文件上起作用。调用任何需要当前上下文的命令之前，请使用 SetCurrentObject (SET, CD) 命令移至对象。

此外，一些 CLI 命令仅对活动对象起作用；此对象可以是带评估的当前选择的源文件、项目或应用程序。

AppScan Source 命令行界面 (CLI) 许可权

许多 CLI 命令都要求您拥有相应的 AppScan Source 许可权。

如果您不具有某项操作的许可权，那么会出现错误消息。请注意，Scan (SC)、Publish 和 Report (RPT) CLI 命令需要 AppScan Source for Automation 许可证。

有关更多信息，请咨询管理员。

启动 AppScan Source 命令行界面 (CLI)

- 在 Windows 系统上，运行 <install_dir>\bin\AppScanSrcCli.exe（其中 <install_dir> 是 AppScan Source 安装位置）。例如，在 Windows（32 位）上：

  C:\Program Files\IBM\AppScanSource\bin\AppScanSrcCli.exe
- 在 Linux 系统上，运行 <install_dir>/bin/appscansrccli，例如：

  /opt/ibm/appscansource/bin/appscansrccli

通过 CLI，可以在应用程序启动时在命令行上传递一个有效命令。

示例：

CLI 传递 script 命令以自动执行整个 CLI 会话。

```
<install_dir>\bin\AppScanSrcCli script myscript.txt
```

以本地语言显示 AppScan Source 命令行界面 (CLI)

本主题描述以本地语言显示 CLI 所需的步骤。

过程

- 在文本编辑器中打开 <data_dir>\config\cli.cp（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）。向文件中添加以下行：

  -Duser.language=<language>

  其中 <language> 是您要显示的语言。支持以下值：
  - de，表示德语
  - es，表示西班牙语
  - fr，表示法语
  - it，意大利语
  - ja，表示日语

- ko，表示朝鲜语
- ru 表示俄语
- pt_BR 或 pt-BR，表示巴西葡萄牙语
- zh_CN 或 zh-CN，表示简体中文
- zh_TW 或 zh-TW，表示繁体中文

在您添加了此行后，保存文件。

- 在文本编辑器中打开 <data_dir>\config\ounce.ozsettings。在文件中查找 locale 设置。此设置将与以下类似：

```
<Setting
      name="locale"
      value=""
      default_value=""
      description="The language (and optionally the Country/Region)
       in which UI and messages should be displayed. E.g. fr for
       French, or pt-BR for Portuguese (Brazil)"
      display_name="Locale"
      type="text"
      available_values=""
      read_only="false"
      hidden="false"
      force_upgrade="false"
 />
```

在此设置中，修改 value 属性，以便其设置为以上步骤中指定的相同语言设置。例如，如果您在以上步骤中添加了 -Duser.language=fr，那么请将 value 属性更改为 value="fr"。

在您修改了此设置后，保存文件。

- 启动 CLI（如果在 CLI 已在运行时做出了这些更改，那么您将需要重新启动 CLI）。

命令语法

AppScan Source 命令行界面 (CLI) 命令遵循带有必需和可选参数的用法模板（类似于命令 shell）。CLI 命令不区分大小写，并且不需要用于不同参数的 switch。

所有 CLI 命令的语法为：

command required_args [optional_args]

命令描述和用法遵循以下约定：

- **command**：实际命令，以 **bold Courier** 字体标识。
- 必需参数：以 courier 字体标识。
- 字面值：以 *italics* 标识。
- 可选参数：以方括号 [ ] 括起来。
- 具有选项的必需参数：以花括号 { } 括起来。
- 某些参数需要关联的值（通过尖括号 < > 进行标识）。如果使用这些参数而不包含值，那么将收到错误消息。
- 每个命令都具有长名称和缩写。可在命令标题中的圆括号内找到缩写。
- 所有命令名都是一个单词，其中不带任何空格或跳格。

- 参数无需命令开关，但是**与顺序相关**。

  要点：请务必记住参数与顺序相关。

参数可以包含空格和跳格。请使用双引号 (" ") 将包含空格或跳格的参数括起来，否则会将其解析为多个参数。

AppScan Source 命令行界面 (CLI) 命令

此部分描述了所有可用 CLI 命令、必需和可选参数以及示例。各命令与『AppScan Source 命令行界面 (CLI) 命令摘要』中标注的特定命令类别对应。

各命令由长名称和缩写组成。例如，用于扫描应用程序、项目或文件并创建评估的命令为 scan，缩写为 sc。

AppScan Source 命令行界面 (CLI) 命令摘要

下表列出了 CLI 命令、各命令的描述以及是否需要登录。

表 *1. CLI* 命令

命令和缩写	描述	需要登录
about (a)	显示 AppScan Source 命令行界面版本和版权信息。	
clearcache (cc)	除去漏洞分析高速缓存和定制规则签名数据。使用 openapplication (oa) 命令打开应用程序之后，可通过发出 clearcache 来清除其高速缓存。	
delete (del)	从当前对象中删除子对象。	是
deleteassess (da)	该命令已重命名。请参阅 removeassess (da)。	
deleteuser (du)	从 AppScan Source 数据库中删除用户。	是
delvar (dv)	删除单个变量。	是
details (det)	列出当前对象的评估详细信息。	是
echo	将所有输入和输出回传到屏幕。	
getaseinfo (gase)	打印 AppScan Enterprise Server 设置。	是
help (?)	列出所有命令或单个命令的帮助。	
import (im)	使用 import 命令可将项目（如 .ppf）添加到现有应用程序。	是
info (i)	列出有关当前对象的属性和值的信息。	是

表 *1. CLI* 命令 *(续)*

命令和缩写	描述	需要登录
`list (ls, dir)`	列出对象树中当前对象下的所有对象。树以对象标识、名称和类型的图形表示法显示。	是
`listassess (la)`	列出对象树中当前对象的对象标识和评估日期/时间。使用 `listassess` 可获取标识以和 `details` 命令配合使用。	是
`listgroups (lgrp)`	列出所有组、其许可权以及各组的描述。	是
`listusers (lu)`	列出所有 AppScan Source 用户。	是
`log`	打开或关闭消息记录。	
`login (in)`	登录到 AppScan Enterprise Server（替换 `login_local (local)`）。	
`login_file`	使用令牌文件（令牌文件是将 `-persist` 选项与 CLI 第 34 页的『`login (in)`』 命令配合使用创建而成，或在创建 AppScan Source for Automation 用户时创建而成）登录到 AppScan Enterprise Server。	
`logout (out)`	从 AppScan Source 注销并终止命令行界面 AppScan Source 命令行界面 (CLI) 会话。	是
`moduser (mu)`	为 AppScan Source 用户修改用户信息，如许可权、用户标识和名称。	是
`newuser (nu)`	创建新的 AppScan Source 用户（需要有效用户名、密码、全名）。 AppScan Source 用户可存在于 AppScan Enterprise Server 用户存储库和 AppScan Source 数据库中，或者如果您需要无法访问服务器的用户，那么可在本地将其创建为 AppScan Source 用户。还可新建 AppScan Enterprise Server 上已存在的 AppScan Source 用户。	是
`openapplication (oa)`	该命令还可以用于打开现有应用程序，或用于创建新的 AppScan Source 应用程序文件。	是
`openassessmentfile (oaf)`	打开 AppScan Source 评估文件 (`file_name.ozasmt`)。	是

表 1. CLI 命令 (续)

命令和缩写	描述	需要登录
password (passwd)	通过 password 命令可以更改您的密码，如果您具有管理员许可权，那么还可以更改其他用户的密码。	是
printuser (pu)	printuser (pu) 命令在屏幕上显示有关单个用户的信息。	是
publishassess (pa)	将当前评估或所选评估发布到 AppScan Source 数据库。使用该命令时，评估可用于 AppScan Source 客户机（例如 AppScan Source for Analysis），但不可用于 AppScan Enterprise Console（使用 第 44 页的『publishassessase (pase)』 命令可发布到 AppScan Enterprise Console）。	是
publishassessase (pase)	将当前评估或所选评估发布到 AppScan Enterprise Console。使用该命令时，评估不可用于 AppScan Source 客户机（例如 AppScan Source for Analysis）（使用 第 43 页的『publishassess (pa)』 命令可发布到 AppScan Source 客户机）。	是
quit	结束并关闭 AppScan Source 命令行界面会话。如果您已登录，将发出注销命令。	
record (rc)	打开或关闭命令记录。	
refresh (rf)	Refresh 从磁盘复原当前项目或应用程序。	是
register (reg)	向 AppScan Source 数据库注册项目和应用程序。	是
removeassess (da)	从内存中除去选定评估或当前评估。	是
report (rpt)	Report 生成指定类型的 AppScan Source 报告，包括结果报告和 AppScan Source 报告。使用此命令需要有效的 AppScan Source for Automation 许可证。	是

表 *1. CLI* 命令 *(续)*

命令和缩写	描述	需要登录
scan (sc)	扫描一个应用程序（或所有应用程序）、项目或文件。使用此命令需要有效的 AppScan Source for Automation 许可证。	是
script (scr)	运行命令脚本。	
setaseinfo (sase)	指定 Enterprise Console 设置。	是
setcurrentobject (set, cd)	使用 setcurrentobject 导航对象树。	是
setvar (sv)	创建新变量或修改现有变量。	是
unregister (unreg)	该命令用于从当前节点注销先前注册的应用程序或项目。	是

以下命令已从 CLI 中移除或不推荐使用：

- add：不推荐。请勿使用。
- listproducts (lprod)：不推荐。请勿使用。
- liststopobject (lstop)：不推荐。请勿使用。
- login_admin：已除去。请勿使用。
- login_local (local)：不推荐。使用login (in)。
- new：不推荐。使用openapplication (oa)。
- reset (r)：已除去。请勿使用。
- runassess (ra)：不推荐使用。使用scan (sc)。

about (a)

描述

显示 AppScan Source 命令行界面版本和版权信息。

语法

about

示例

```
AllApplications>> About
-------------------------------------------------------------------------------
-
|       Security AppScan Source 8.5.0.0 Build 175.
|
|       Licensed Materials - Property of IBM Corp. © Copyright IBM"
|       Corp. and its licensors 2003, 2011.  All Rights Reserved.  IBM,
|       the IBM logo, ibm.com, Rational, AppScan and ClearQuest are
|       trademarks or registered trademarks of International Business
|       Machines Corp., registered in many jurisdictions worldwide.
|       Other product and service names might be trademarks of IBM or
|       other companies. A current list of IBM trademarks is available
|       on the Web at "Copyright and trademark information" at
|       www.ibm.com/legal/copytrade.shtml. Linux is a registered
```

```
|       trademark of Linus Torvalds in the United States, other
|       countries, or both. Microsoft, Windows, Windows NT and the
|       Windows logo are trademarks of Microsoft Corporation in the
|       United States, other countries, or both. UNIX is a registered
|       trademark of The Open Group in the United States and other
|       countries.  Java and all Java-based trademarks and logos are
|       trademarks of Sun Microsystems, Inc. in the United States, other
|       countries, or both.
|
|       This program includes: Jacorb 2.3.0, Copyright 1997-2006 The
|       JacORB project; Jericho HTML Parser 2.1, Copyright 2005 Martin
|       Jericho; and XOM 1.0d22, Copyright 2003 Elliotte Rusty Harold,
|       each of which is available under the Gnu Library General Public
|       License (LGPL), a copy of which is available in the Notices file
|       that accompanied this program.
--------------------------------------------------------------------------------
```

clearcache (cc)

描述

除去漏洞分析高速缓存和定制规则签名数据。使用 openapplication (oa) 命令打开应用程序之后，可通过发出 clearcache 来清除其高速缓存。

发出该命令之后，如果启用了 Java 递增分析，那么为应用程序或项目启动的下一个扫描将是完整扫描。

语法

clearcache

示例

```
AllApplications>> openapplication SimpleIOT
AllApplications\SimpleIOT>> clearcache
AllApplications\SimpleIOT>>
```

Java 的递增分析

启用递增分析时，AppScan Source 将对分析数据进行高速缓存。然后，当您重新扫描项目或应用程序时，AppScan Source 使用该值来确定代码更改，并将仅再次分析受更改影响的代码部分。最终结果是代码的完整分析，但是一小部分时间内的完整分析。

关于此任务

在 Windows 和 Linux 上支持递增分析。启用后，将对 AppScan Source 项目或应用程序或者对 Eclispe 项目或工作空间执行递增分析。启用递增分析之后，对项目、应用程序或工作空间运行的第一个扫描将始终是完整扫描（完整扫描期间才会更新漏洞分析高速缓存）。这使 AppScan Source 能够对后续扫描的数据进行高速缓存。因此，只要漏洞分析高速缓存未被清除而且只要已更改文件的数量不超过您可以确定的阈值设置，那么项目、应用程序或工作空间的扫描是递增扫描。

要启用和使用递增分析，请执行以下步骤：

过程

1. 在文本编辑器（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）中打开 <data_dir>\config\scan.ozsettings。在文件中查找 incremental_analysis 设置。此设置将与以下类似：

```
<Setting
  name="incremental_analysis"
     read_only="false"
  default_value="false"
  description="Attempt to scan only changed files,
    instead of re-scanning everything."
  type="bool"
  value="false"
  display_name="Incremental Analysis"
  hidden="true"
/>
```

   在该设置中，修改 value 属性。如果属性设置为 true，该设置将打开。如果属性设置为 false，那么 AppScan Source 在扫描时将不会执行递增分析。

2. 在 <data_dir>\config\scan.ozsettings 中，找到 percentage_of_files_changed 设置：

```
<Setting
  name="percentage_of_files_changed"
     read_only="false"
  default_value="50"
  description="In incremental scanning, if percentage of files
  being changed since last scan exceeds the threshold, full
  scan will be initiated. The percentage ranges from 0 to 100.
  Default threshold is 50, which represents 50%."
  type="int"
  value="50"
  display_name="Percentage of files being changed"
  hidden="true"
/>
```

   该设置使您能够指定开始完整扫描之前需要更改的文件的百分百。缺省情况下，该阈值百分比为 50%，这意味着如果在项目、应用程序或工作空间中有 50% 或更多文件发生更改后重新扫描，那么将启动完整扫描而不是递增分析扫描。在该设置中，根据需要将 value 属性更改为您期望的阈值百分比。

3. 在修改所有相关设置之后保存 <data_dir>\config\scan.ozsettings，然后启动或重新启动支持递增分析的 AppScan Source 产品。例如，重新启动 AppScan Source for Analysis、AppScan Source for Development Eclipse 插件 或 AppScan Source 命令行界面 (CLI)，或者重新启动 AppScan Source for Automation 服务。
4. 现在，当您通过相同扫描配置重新扫描 Java 应用程序或项目时，如果更改的文件的数量未超过阈值而且如果漏洞分析高速缓存未清除，那么将执行递增分析。
5. **清除漏洞分析高速缓存：**如果递增扫描有问题，或者想要在启用递增分析时执行完整分析扫描，请在再次扫描之前清除漏洞高速缓存：
   - AppScan Source for Analysis:
     a. 打开 AppScan Source 项目的"属性"视图。如果要扫描应用程序，打开任何子项目的属性视图（删除项目的高速缓存也将删除其应用程序的高速缓存）。
     b. 在"概述"选项卡中，单击**清除高速缓存**。

- AppScan Source for Development Eclipse 插件：删除 <data_dir>\temp\<workspace>\<project>，其中：
  - <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述.
  - <workspace> 是在其中进行扫描的 Eclipse 工作空间的名称。要删除整个工作空间的高速缓存，请删除整个 <data_dir>\temp\<workspace> 目录。
  - <project> 是要扫描的 Eclipse 项目的名称。要删除项目的高速缓存，请删除 <data_dir>\temp\<workspace>\<project> 目录。
- AppScan Source 命令行界面 (CLI)：使用 clearcache 命令，如*IBM Security AppScan Source Utilities* 用户指南中所述。
- AppScan Source for Automation：使用 ScanApplication命令 -clearcache 参数，如*IBM Security AppScan Source Utilities* 用户指南中所述。

结果

在 AppScan Source for Analysis 中进行扫描之后，可使用**评估差异**功能在代码更改之前和之后比较评估。

提示：

- 要强制执行完全分析扫描，禁用递增分析或清除漏洞分析高速缓存。
- 执行递增分析时，应在进行任何修改之后运行完全分析扫描。
  - 安全性规则更改或适用于项目或应用程序的定制规则更改。
  - 扫描配置更改。
  - 影响扫描的 .ozsettings 文件更改。
  - 应用程序或项目属性的更改。例如，在所有应用程序或选定应用程序或项目的 AppScan Source for Analysis"属性"视图中进行的任何更改。
  - 将新项目添加到应用程序或删除现有项目。
  - 从扫描排除文件。例如，在 AppScan Source for Analysis 中，可通过在"资源管理器"视图中右键单击文件并选择**从扫描排除**以从扫描排除该文件。
- 有关递增分析的最新信息可在http://www.ibm.com/support/docview.wss?uid=swg21994390中找到。

注：

- 递增扫描之后，编辑器中的结果标记可能不再出现在正确的位置。
- 没有跟踪的已修复结果将出现在递增扫描结果中。
- 在递增分析期间不能同时具有多个 AppScan Source 产品或组件。此外，当您在扫描时，另一个用户无法同时在相同机器上扫描相同应用程序或项目。

delete (del)

描述

从当前对象中删除子对象。

使用 list (ls, dir) 命令可查看当前对象的子代。

识别子对象以按名称或标识进行删除。

语法

del {name|*id* object_id}

- name：子对象名。要查看当前对象的子代，请发出 list 命令。
- *id*：字面值。指示按对象标识进行删除。
- object_id：按标识进行删除时的必需参数。object_id 是要删除的子代的数字标识。

示例

- 删除名为 App1 的应用程序：

  ```
  AllApplications>> Delete App1
  ```

- 删除标识为 40 的项目：

  ```
  AllApplications\App1>> Delete id 40
  ```

deleteassess (da)

该命令已重命名。请参阅 removeassess (da)。

deleteuser (du)

描述

从 AppScan Source 数据库中删除用户。

语法

du username

username：要删除的 AppScan Source 用户。

示例

删除用户 agraham：

```
AllApplications>> deleteuser agraham
AllApplications>> Deleted user 'agraham'.
```

delvar (dv)

描述

删除单个变量。

语法

delvar <variable name>

variable name：要删除的变量的名称。如果存在具有该名称的变量，那么会直接将其删除，后面显示成功消息。

示例

删除 myvar 变量：

```
AllApplications>> delvar myvar
Deleted variable '%myvar%'.
```

details (det)

描述

列出当前对象的评估详细信息。

检索有关当前评估或按评估标识选择的评估的详细信息。使用 listassess (la) 可获取评估及其标识的列表。

提示：使用 details 命令之前，请使用 log 命令来打开记录。您可以生成 Microsoft Excel 或其他接受 CSV 数据的程序能够打开的逗号分隔文本 (.csv) 格式的日志文件。

语法

details [id]

id：当前会话中的评估的对象标识。如果未指定标识，那么会显示（或向日志文件发送）最新评估的详细信息。

示例

显示最新评估的详细信息：

```
AllApplications\Myapps>> details
File, line, col, name, type, severity, confidence
C:\MyApps\WebGoat\webgoat\src\lessons\CommandInjection.java
, 119
, 0
, java.lang.Throwable.printStackTrace
, Vulnerability.Info
, 3-SeverityType_info
, 3-ConfidenceType_low
  .
  .
  .
C:\MyApps\WebGoat\webgoat\src\session\ECSFactory.java
, 625
, 0
, java.lang.String.equalsIgnoreCase
, Vulnerability.Info
, 3-SeverityType_info
, 3-ConfidenceType_low

 -------------------
  Total Call Sites: 1283
  Total Definitive Security Findings with High Severity: 10
  Total Definitive Security Findings with Medium Severity: 15
  Total Definitive Security Findings with Low Severity: 53
  Total Suspect Security Findings with High Severity: 24
  Total Suspect Security Findings with Medium Severity: 25
  Total Suspect Security Findings with Low Severity: 6
  Total Scan Coverage Findings with High Severity: 123
  Total Scan Coverage Findings with Medium Severity: 69
  Total Scan Coverage Findings with Low Severity: 56
  Total Lines: 17197
  Max V-Density: 929.8482293423273
  Max V/kloc: 22.155027039599933
  V-Density: 929.8482293423273
  V/kloc: 22.155027039599933
```

echo

描述

将所有输入和输出回传到屏幕。

该命令通常在脚本内使用。

语法

echo

示例

输出注释：

```
3/11/08 2:15 PM  AllApplications>>
3/11/08 2:16 PM  echo "this is a test"
3/11/08 2:16 PM  this is a test
```

getaseinfo (gase)

描述

打印 AppScan Enterprise Server 设置。

此命令显示已通过 setaseinfo (sase) 命令设置的 AppScan Enterprise Server 配置。

语法

getaseinfo

注：

- 您必须登录到 AppScan Source 命令行界面 (CLI)，并具有管理 AppScan Enterprise 设置许可权才能设置 url。有关用户帐户和许可权的信息，请参阅《*IBM Security AppScan Source* 安装和管理指南》的"管理 *AppScan Source*"一节。
- userid 和 password 存储在运行 AppScan Source 客户机（例如，AppScan Source for Analysis）的机器上 - 而 url 则存储在 Enterprise Server（它可能位于远程机器上）中。您无法从远程机器（例如，通过从其发出 getaseinfo 命令）来访问 userid 和 password 信息。

示例

```
AllApplications>> getaseinfo

Username:        my_domain\my_username
URL:       http://my_aseserver/ase
```

help (?)

描述

列出所有命令或单个命令的帮助。

AppScan Source 命令行界面 (CLI) 帮助会显示各命令的命令语法、别名和样本用法。Help 可用于个别命令，您也可以请求获取所有命令的列表（带有各命令的简短概述）。

语法

help [command]

command：要显示其联机帮助的 CLI 命令。您必须处于树中的帮助可用于特定对象的位置。

示例

- 列出所有有效命令的帮助：

  ```
  AllApplications>> ?
  ```

- 列出 register (reg) 命令的帮助：

  ```
  AllApplications>> Help register
  ```

import (im)

描述

使用 import 命令可将项目（如 .ppf）添加到现有应用程序。

使用该命令添加项目时，支持以下项目文件：

- AppScan Source 项目文件 (.ppf)。
- Microsoft Visual Studio C/C++ (.vcproj)
- Microsoft Visual Studio C# (.csproj)
- Microsoft Visual Studio Visual Basic (.vbproj)

导入项目时，可使用通配符来替换实际文件名和/或文件扩展名，并且您可以指定其他参数来指示是否应导入指定目录下的所有项目（递归导入）。

语法

import path [true|false]

- 参数数量根据所导入对象的类型而异。
- path：所导入对象的完整路径。
- true 或 false：可选。包括或省略子目录。缺省值为 false（省略子目录）。

示例

```
AllApplications\testit>> import c:\testapps\java\webgoat\*.ppf
AllApplications\testit>> ls
23942: webgoat (Project [local])
```

info (i)

描述

列出有关当前对象的属性和值的信息。

语法

信息

示例

列出当前应用程序的标识和名称：

```
AllApplications\Myapp>> Info
```

输出类似于：

```
object: Myapp(Application)
id: 1

...
```

list (ls, dir)

描述

列出对象树中当前对象下的所有对象。树以对象标识、名称和类型的图形表示法显示。

语法

list [*all*]

- all：可选。显示对象树中的所有对象。
- 不传递任何参数将仅列出当前对象的直接子代。

示例

- 列出当前对象树中的对象：

```
AllApplications>> List
1: WebGoat_4_0 (Application [local] [registered])
2: test_application (Application [local] [registered])
```

- 列出对象树中的所有对象：

```
AllApplications>> LS all

1: WebGoat_4_0 (Application [local] [registered])
|----198: compile (Project [local] [registered])
|----|----470: WebContent\lp\JavaScriptValidation.html (Source)
|----|----475: WebContent\lp\RemoteAdminFlaw.html (Source)
|----|----483: WebContent\lp\WelcomeScreeen.html (Source)
|----|----474: WebContent\lp\ReflectedXSS.html (Source)
|----|----477: WebContent\lp\SqlInjection.html (Source)
. . .
```

listassess (la)

描述

列出对象树中当前对象的对象标识和评估日期/时间。使用 listassess 可获取标识以和 details 命令配合使用。

语法

listassess [*all*]

all：可选。列出内存中的所有评估。

示例

- 列出当前评估对象信息：

```
AllApplications\WebGoat_4_0>> listassess

AllApplications\WebGoat_4_0>> 4762: WebGoat_4_0 (Application, Tue Mar 11 14:20:23
EDT 2008)
AllApplications\WebGoat_4_0>> 9033: WebGoat_4_0 (Application, Tue Mar 11 14:43:31
EDT 2008)
```

- 列出所有评估对象信息：

```
AllApplications\Applications>> ListAssess all
542: putty (Application, Fri Aug 04 08:38:18 EDT 2008)
1804: JavaAny (Application, Tue Apr 01 13:17:33 EDT 2008)
2971: snare (Application, Tue Apr 01 08:42:53 EDT 2008)
4773: SimpleIOT (Project, Tue Apr 01 07:31:11 EDT 2008)
4874: JSPWiki (Application, Tue Apr 01 13:22:08 EDT 2008)
```

listgroups (lgrp)

描述

列出所有组、其许可权以及各组的描述。

语法

listgroups

示例

```
AllApplications>> ListGroups

----------------------------------------
| Group: APPS
| Description: Application and Project Management
| Permissions:
|  REGISTER (Register)
|  ATTRAPPLY (Apply Attributes)
|  ATTRMODIFY (Manage Attributes)
|  SCAN (Scan)
|  VIEWREGISTER (View Registered)
| Group: ASSESSMENTS
| Description: Assessment Management
| Permissions:
|  ASMNTDELETE (Delete Published Assessments)
|  ASMNTPUBLISH (Publish Assessments)
|  ASMNTSAVE (Save Assessments)
|  ASMNTVIEWPUBLISH (View Published Assessments)
| Group: KB
| Description: Knowledgebase Management
| Permissions:
|  CUSTOM (Manage Custom Rules)
|  PATTERN (Manage Patterns)
| Group: ADMIN
| Description: Administration
| Permissions:
|  ASE (Manage AppScan Enterprise Settings)
|  LDAP (Manage LDAP Settings)
|  USER (Manage Users)
----------------------------------------
```

listusers (lu)

描述

列出所有 AppScan Source 用户。

如果可用，那么输出将包含用户标识、名称和用户存储库。

语法

listusers

示例

列出所有 AppScan Source 用户：

```
AllApplications>> Listusers
All Users
  admin     (myASEServer\admin)
  jsmith    (John Smith) local
  joandarcy (Joan Darcy) local
AllApplications>>
```

log

描述

打开或关闭消息记录。

将会话记录到指定文件或查看当前记录状态。整个会话（包括输出）将记录到指定文件。

提示：要仅记录命令输入，请使用 record (rc) 命令。

语法

log {*on* file|*off*}

- 需要 on 或 off 参数才能打开或关闭记录。不带参数的 Log 识别当前状态。
- on：必需此参数以打开记录。On 需要 file 名称。
- file：日志文件的名称。您可以指定要生成的文件的类型，如 .txt 或 .csv。
- off：关闭当前记录会话。

示例

- 记录到名为 MyLogFile.txt 的文件：

  ```
  AllApplications>> Log on c:\MyLogFile.txt
  ```

- 关闭记录：

  ```
  AllApplications>> Log off
  ```

login (in)

描述

登录到 AppScan Enterprise Server（替换 login_local (local)）。

如果登录成功，那么提示符将更改为 AllApplications>> 以指示您已登录。

语法

login server user_id password [-persist] [-acceptssl]

- server：必需。指定 AppScan Enterprise Server 实例的 URL。该 URL 的格式为 http(s)://<hostname>:<port>/ase，其中 <hostname> 是已安装了 AppScan Enterprise Server 的机器的名称，<port> 是服务器运行所在的端口。该 URL 的示例为 https://myhost.mydomain.ibm.com:9443/ase。
- user_id：必需。
  - 如果 AppScan Enterprise Server 认证方法为用户标识和密码，输入用户用户标识。
  - 如果 AppScan Enterprise Server 配置为使用 Windows 认证，请输入用于连接到 Enterprise Console 的域和用户名（以 \ 分隔域和用户名，例如 my_domain\my_username）。
  - 如果 AppScan Enterprise Server 已配置 LDAP，请输入用于连接到 Enterprise Console 的用户名。
  - 在 Windows 上，如果支持 AppScan Enterprise Server进行通用访问卡 (CAC) 认证，输入 common_name<cn_certificate>，其中 common_name 是 CAC 通用名 (CN)，cn_certificate 是证书签发人的 CN。如果 common_name 或 cn_certificate 包含空格，在值的两边加上问号 ("common_name<cn_certificate>")。
- 密码：如果 AppScan Enterprise Server 认证方法为用户标识和密码，该字段为必填字段。指定 Enterprise Server 用户标识的密码。

  注：支持 AppScan Enterprise Server 进行 CAC 认证时，不使用该选项。
- -persist：可选。通过加密密钥使登录凭证持久。如果使用了 persist 参数，那么 login 会将用户名和密码保存到名为 cli.token 的加密密钥文件。cli.token 保存到用户 home 目录的 .ounce 文件夹。
- -acceptssl：可选。自动接受 SSL 证书。有关更多信息，请参阅第 35 页的『AppScan Enterprise Server SSL 证书』。

  要点：如果未包含此参数，在遇到无效证书的情况下，登录到或发布到 AppScan Enterprise Console 将失败，并具有错误（如果您在通过其他 AppScan Source 客户机产品登录时尚未永久接受证书）。

示例

- 用户 John Smith 登录到已由此管理员设置好的 AppScan Enterprise Server：

```
>> login https://myserver:9443/ase/ johnsmith mypassword -acceptssl
Login successful.
AllApplications>>
```

- 在 Windows 上，用户 Jane Smith 登录到支持进行 CAC 认证的 AppScan Enterprise Server。她的 CN 为 janesmith，其公司的证书 CN 为 mydomain-co-ABC123-CO：

  ```
  >> login https://myserver:9443/ase/ janesmith<mydomain-co-ABC123-CO>
  ```

  然后，将通过 Windows 安全性对话框来提示她输入其 CAC 卡密码。当她通过密码完成该对话框时，命令行将显示以下内容：

  ```
  Login successful.
  AllApplications>>
  ```

AppScan Enterprise Server SSL 证书

安装 AppScan Enterprise Server 时，应对其进行配置以使用有效 SSL 证书。如果未完成该操作，那么在 Windows 和 Linux 上通过 AppScan Source for Analysis、AppScan Source 命令行界面 (CLI) 或 AppScan Source for Development 登录到服务器时将接收不可信的连接消息。

SSL 证书存储位置

已被永久接受的证书存储在 <data_dir>\config\cacertspersonal 和 <data_dir>\config\cacertspersonal.pem（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述） 中。如果不需要永久地存储证书，请除去这两个文件。

AppScan Source for Automation 和 SSL 证书验证

缺省情况下，使用 AppScan Source for Automation 时将自动接受证书。此行为由 自动化服务器 配置文件 (<data_dir>\config\ounceautod.ozsettings（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）) 中的 ounceautod_accept_ssl 设置来确定。如果编辑了该设置以至于 value="true" 设置为 value="false"，那么将尝试 SSL 验证，而且当遇到无效证书时到 AppScan Enterprise Console 的登录或发布将失败，并会出现错误。

AppScan Source 命令行界面 (CLI) 和 SSL 证书验证

缺省情况下，当使用 CLI login 命令时，将尝试 SSL 验证，而如果遇到无效证书，那么登录到 AppScan Enterprise Console 或向其进行发布的操作将失败并出错（如果在通过其他 AppScan Source 产品登录时，您尚未永久接受证书）。可以通过在发出 login 命令时使用选项 -acceptssl 参数来修改此行为。使用此参数后，将自动接受 SSL 证书。

login_file

描述

使用令牌文件（令牌文件是将 -persist 选项与 CLI 第 34 页的『login (in)』 命令配合使用创建而成，或在创建 AppScan Source for Automation 用户时创建而成）登录到 AppScan Enterprise Server。

如果登录成功，那么提示符将更改为 AllApplications>> 以指示您已登录。

语法

login_file server token_file [-acceptssl]

- server：必需。指定 Enterprise Server 实例的 URL。
- token_file：必需。指示令牌文件的路径和文件名。如果令牌文件是通过 CLI 第 34 页的『login (in)』 命令创建而成，那么文件为 ouncecli.token 并驻留在用户 home 目录的 .ounce 文件夹中。如果令牌文件是针对 AppScan Source for Automation 而创建，那么文件为 <data_dir>/config/ounceautod.token（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）。
- -acceptssl：可选。自动接受 SSL 证书。有关更多信息，请参阅第 35 页的『AppScan Enterprise Server SSL 证书』。

  要点：如果未包含此参数，在遇到无效证书的情况下，登录到或发布到 AppScan Enterprise Console 将失败，并具有错误（如果您在通过其他 AppScan Source 客户机产品登录时尚未永久接受证书）。

示例

用户 John Smith 使用在 Windows 上从 CLI 创建的令牌文件登录到 AppScan Enterprise Server 中：

```
>> login_file https://myserver:9443/ase/
     C:\Users\Administrator\.ounce\ouncecli.token
Login successful.
AllApplications>>
```

login_local (local)

描述

在 AppScan Source V6 中不推荐使用。使用login (in)。

logout (out)

描述

从 AppScan Source 注销并终止命令行界面 AppScan Source 命令行界面 (CLI) 会话。

如果未登录，那么 logout 为无效命令。

语法

logout

示例

注销：

```
AllApplications>> Logout
>> Logged Out.
>> Exiting Security AppScan Source Command Line Interface...
```

moduser (mu)

描述

为 AppScan Source 用户修改用户信息，如许可权、用户标识和名称。

语法

```
moduser --userid|-u <user id>
[--fullname|-f <user first and last name>]
[--group [group[:permission[;permission...]
 [--group...]]
[--removegroup [group[:permission
 [;permission...] [--removegroup...]]
```

用户名和名称

- --username|-u：必需。有效的 AppScan Source 用户名。
- --fullname|-f：可选。用户的全名。如果此条目中包含空格，请使用 " 符号将其括起来（例如 -f "Joe Smith"）。

组和许可权

可选。

组和许可权标识用于标识该用户所允许执行的 AppScan Source 任务。未专门标识为许可权的一部分的任务可用于所有用户：

--group：要为此用户添加的组和组许可权。如果指定组时不带任何许可权，将向用户授予该组中的所有许可权。

或者

--removegroup：要从此用户中除去的组和组许可权。如果指定组时不带任何许可权，将除去该组中的所有许可权。

组和许可权包括：

- ASSESSMENTS：评估级别许可权。
  - ASMNTDELETE：删除已发布的评估。
  - ASMNTPUBLISH：发布评估。
  - ASMNTSAVE：保存评估。
  - ASMNTVIEWPUBLISH：查看已发布的评估。
- ADMIN：管理许可权。
  - ASE：管理 AppScan Enterprise 设置
  - USER：管理用户设置，包括添加和删除用户以及更改用户许可权。
- APPS：应用程序和项目级别许可权
  - ATTRAPPLY：将属性应用到应用程序。
  - ATTRMODIFY：创建、删除和修改属性。
  - VIEWREGISTER：查看已注册的应用程序和项目。
  - REGISTER：注册/注销应用程序和项目。默示 VIEWREGISTER 许可权。
  - SCAN：扫描应用程序和项目。

- KB：知识库管理许可权。
  - CUSTOM：管理定制规则。
  - PATTERN：创建、编辑或删除模式。
- FILTER：过滤器管理。
  - SHAREDFILTERS：管理共享过滤器。
- SCANCONFIG：扫描配置管理
  - SHAREDCONFIGS：管理共享扫描配置。

示例

创建 Joan Darcy 的用户凭证（通过 newuser (nu) 命令）之后，系统管理员确定该用户仅需要保存、发布和查看许可权；但是不需要删除许可权。此外，Joan 还需要知识库模式许可权：

```
moduser --userid joandarcy --removegroup
ASSESSMENTS:ASMNTDELETE --group KB:PATTERN
```

newuser (nu)

描述

创建新的 AppScan Source 用户（需要有效用户名、密码、全名）。 AppScan Source 用户可存在于 AppScan Enterprise Server 用户存储库和 AppScan Source 数据库中，或者如果您需要无法访问服务器的用户，那么可在本地将其创建为 AppScan Source 用户。还可新建 AppScan Enterprise Server 上已存在的 AppScan Source 用户。

注：如果 AppScan Enterprise Server 支持通用访问卡 (CAC) 认证，那么 newuser (nu) 命令不适用。

语法

```
newuser --userid|-u <user id>
--password|-p <password>
--fullname|-f <user first and last name>
[--group [group[:permission[;permission...]
 [--group...]]
```

识别信息

- --userid|-u：必需。用户标识。不允许空格。
- --password|-p：用户密码。
- --fullname|-f：用户的全名。如果此条目中包含空格，请使用 " 符号将其括起来（例如 -f "Joe Smith"）。

组和许可权

可选。组和许可权标识用于标识该用户所允许执行的 AppScan Source 任务。未专门标识为许可权的一部分的任务可用于所有用户：

--group：此用户的组和组许可权。如果指定组时不带任何许可权，将向用户授予该组中的所有许可权。组及其许可权包括：

- ASSESSMENTS：评估级别许可权。

  - ASMNTDELETE：删除已发布的评估。
  - ASMNTPUBLISH：发布评估。
  - ASMNTSAVE：保存评估。
  - ASMNTVIEWPUBLISH：查看已发布的评估。
- ADMIN：管理许可权。
  - ASE：管理 AppScan Enterprise 设置
  - USER：管理用户设置，包括添加和删除用户以及更改用户许可权。
- APPS：应用程序和项目级别许可权
  - ATTRAPPLY：将属性应用到应用程序。
  - ATTRMODIFY：创建、删除和修改属性。
  - VIEWREGISTER：查看已注册的应用程序和项目。
  - REGISTER：注册/注销应用程序和项目。默示 VIEWREGISTER 许可权。
  - SCAN：扫描应用程序和项目。
- KB：知识库管理许可权。
  - CUSTOM：管理定制规则。
  - PATTERN：创建、编辑或删除模式。
- FILTER：过滤器管理。
  - SHAREDFILTERS：管理共享过滤器。
- SCANCONFIG：扫描配置管理
  - SHAREDCONFIGS：管理共享扫描配置。

LDAP 认证

如果 AppScan Enterprise Server 用户存储库中没有 LDAP 用户，那么无法将其添加到 AppScan Source 用户存储库中。要添加将通过 LDAP 进行认证的 AppScan Source 用户，那么必须先将 AppScan Enterprise Server 用户存储库配置为使用 LDAP 存储库。有关这方面的信息，请参阅 AppScan Enterprise Server *Planning & Installation Guide*。

如果您在使用 LDAP 认证而且想要添加不是 LDAP 用户组的一部分的 AppScan Source 用户，发出 newuser 命令。

示例

在 AppScan Enterprise Server 上创建用户 Joan Darcy。她得用户名为 joandarcy，密码为 123456。Joan 可通过 APPS 和 ASSESSMENTS 组中的所有许可权以及 KB 组中的定制规则许可权来使用 AppScan Source：

```
AllApplications>> newuser --userid joandarcy --password 123456
--fullname "Joan Darcy" --group APPS --group ASSESSMENTS --group KB:CUSTOM
AllApplications>> Created user 'joandarcy'. User ID: 888
```

openapplication (oa)

描述

该命令还可以用于打开现有应用程序，或用于创建新的 AppScan Source 应用程序文件。

使用该命令打开应用程序时，支持以下应用程序文件：

- AppScan Source 应用程序文件 (.paf)。
- Eclipse 或 Rational® Application Developer for WebSphere® Software (RAD) 工作空间 (.ewf)

  注：当您使用 openapplication 来打开工作空间目录（通过指定其路径）时，将生成 .ewf 文件。
- WAR 文件 (.war)
- EAR 文件 (.ear)
- 仅 Windows：Microsoft Visual C++ 工作空间文件 (.dsw)
- 仅 Windows ：Microsoft Visual Studio.NET 解决方案文件 (.sln)

注：要了解 AppScan Source for Analysis、AppScan Source for Automation 和 AppScan Source 命令行界面 支持哪些版本的导入文件，请参阅http://www.ibm.com/support/docview.wss?uid=swg27027486。在此页面中，选择您在使用的 AppScan Source 版本所对应的选项卡，然后选择您在使用的 AppScan Source 组件。如果 AppScan Source 支持从其他开发环境打开和扫描文件，该支持将在**受支持软件**选项卡的**编译器和语言**部分中列出。

语法

```
openapplication file [-appserver_type]
[-include_all_lib_jars] [-include_lib_jars] [-no_ear_project]
```

- file：必需。
  - 如果正在使用命令打开应用程序，请指定现有应用程序的路径和文件名称。
  - 如果正在使用命令创建 AppScan Source 应用程序，请指定有效路径和新文件名称（确保新文件名称还未存在）。
  - 如果正在使用命令打开 Eclipse 或 Rational Application Developer for WebSphere Software (RAD) 工作空间，那么可仅指定工作空间路径。
- -appserver_type：可选。如果要打开的应用程序包含 JavaServer Pages（例如 WAR 或 EAR 文件），使用该设置来指定应用程序服务器以用于 JSP 编译。指定以下某项（用双引号括起）：
  - Tomcat 7
  - Tomcat 8
  - WebSphere 7.0
  - WebSphere 8.0
  - WebSphere 8.5
  - WebLogic 11g
  - WebLogic 12c

注：

- 指定应用程序之前，确保已在 AppScan Source for Analysis 首选项中对其进行正确的配置。
- 如果未使用 -appserver_type，在 AppScan Source for Analysis 中当前设置的缺省 JSP 编译器将用于 JSP 编译。现成可用的 Tomcat 7 是缺省 JSP 编译器。

- 对于 WAR 文件：
  - -include_all_lib_jars：使用该设置可在扫描期间在 WAR 文件中包含所有库。
  - -include_lib_jars：使用该设置可在 WAR 文件中指定扫描期间想要包含的库。使用该设置时，请勿包含库路径信息，并通过逗号分隔多个库。
- -no_ear_project：导入 EAR 文件时，将自动创建用于存储共享库的项目。 如果没有共享库，那么将创建项目，但项目将为空。使用该设置时，将不会为 EAR 文件创建项目。

示例

- 要在 Windows 上打开 Microsoft Visual Studio .NET 解决方案文件：

```
AllApplications>> openapplication c:\mysln.sln
```

- 要打开 Eclipse 工作空间：

```
AllApplications>> oa "C:\Users\myname\My Documents\myworkspace"
```

或

```
AllApplications>> oa "C:\Users\myname\My Documents\myworkspace\myworkspace.ewf"
```

- 要打开 WAR 文件并仅在扫描时包含其库中的某些库：

```
AllApplications>> oa c:\mywar.war -include_lib_jars lib1.jar,lib2.jar
```

openassessmentfile (oaf)

描述

打开 AppScan Source 评估文件 (file_name.ozasmt)。

已打开的评估成为当前评估。

语法

openassessmentfile path_file

path_file：现有评估文件的路径和文件名。

示例

打开先前保存的评估：

```
AllApplications>> OpenAssessmentFile c:\priority.ozasmt
```

password (passwd)

描述

通过 password 命令可以更改您的密码，如果您具有管理员许可权，那么还可以更改其他用户的密码。

要更改密码，请两次输入新密码。两次输入的新密码必须匹配才能使更改生效。更改您自己的密码时，必须首先输入当前密码。

语法

password [user name]

username：要更改其密码的用户的名称。如果不指定用户名，那么表示您更改的是您自己的密码。

示例

- 更改您自己的密码：

```
AllApplications>> Password
Type current password: ******
Type new password: *******
Verify new password: *******
Password changed.
AllApplications>>
```

- 更改其他用户的密码（如果您具有管理员许可权）：

```
AllApplications>> passwd johnsmith
Changing password for user johnsmith
Type new password: *******
Verify new password: *******
Password changed.
AllApplications>>
```

printuser (pu)

描述

printuser (pu) 命令在屏幕上显示有关单个用户的信息。

语法

printuser username

username：AppScan Source 用户名。

示例

显示用户 Joan Darcy 的原始用户数据：

```
printuser joandarcy
User: joandarcy
First Name: Joan
Last Name: Darcy
e-mail Address: jdarcy@example.com
Groups:
  - APPS     (Application and Project Management)
    REGISTER     (Register)
    ATTRAPPLY     (Apply Attributes)
    ATTRMODIFY     (Manage Attributes)
    SCAN       (Scan)
    VIEWREGISTER    (View Registered)
  -(ASSESSMENTS) (Assessment Management)
    ASMTDELETE     (Delete Published Assessments)
    ASMNTPUBLISH   (Publish Assessments)
    ASMNTSAVE    (Save Assessments)
```

```
    ASMNTVIEWPUBLISH (View Published Assessments)
  -[KB]     (Knowledgebase Management)
    CUSTOM      (Manage Custom Rules)
  -[ADMIN]    (Administration)
```

注：方括号 [ ] 指示此用户不具有该组中的所有许可权。

publishassess (pa)

描述

将当前评估或所选评估发布到 AppScan Source 数据库。使用该命令时，评估可用于 AppScan Source 客户机（例如 AppScan Source for Analysis），但不可用于 AppScan Enterprise Console（使用 第 44 页的『publishassessase (pase)』 命令可发布到 AppScan Enterprise Console）。

使用此命令需要有效的 AppScan Source for Automation 许可证。

使用此命令发布评估时，评估将自动注册到 AppScan Enterprise Server。使用该命令时无需手动注册评估。

语法

pa [*id*]

id：可选字面值。识别评估标识。如果未包含标识，那么此命令适用于最新扫描。您可以使用 listassess (la) 命令来查找评估标识。

示例

在以下示例中，我们将当前对象设置为应用程序 pebble，扫描 pebble 项目，然后发布评估（同时会自动将其注册）。

```
AllApplications>> cd pebble
AllApplications\pebble>> scan pebble
New Scan started

Scanning Project pebble (1 of 1)
Preparing project for scan...
  .
  .
Preparing for Vulnerability Analysis...
Performing Vulnerability Analysis...
Generating Findings...
Preparing project for scan...
  .
  .
Scanned Project pebble:
  Total files: 15
  Total findings: 167
  Total lines: 385
  vkloc: 0.44448395412925595
  v-Density: 22.446439683527426
Scanned Application pebble:
  Total files: 15
  Total findings: 167
  Total lines: 385
  vkloc: 0.44448395412925595
  v-Density: 22.446439683527426
Scan completed:
  Total files: 15
```

```
  Total findings: 167
  Total lines: 385
  vkloc: 0.44448395412925595
  v-Density: 22.446439683527426
Elapsed Time - 18 Seconds

AllApplications\pebble>> publishassess

Assessment Successfully Published.
```

publishassessase (pase)

描述

将当前评估或所选评估发布到 AppScan Enterprise Console。使用该命令时，评估不可用于 AppScan Source 客户机（例如 AppScan Source for Analysis）（使用 第 43 页的『publishassess (pa)』 命令可发布到 AppScan Source 客户机）。

语法

publishassessase
-aseapplication <ase_application> [*id*] [*path*]
[-folder <location>] [-name <published_assessment_name>] [-preventOverwrite]

- -aseapplication <ase_application>：当连接到 AppScan Enterprise Server V9.0.3 和更高版本时，需要该选项（除非禁用需求，如此处所示）。当连接到 AppScan Enterprise Server 的较低版本时，关联应用程序是可选的。使用该选项可指定要与评估关联的 Enterprise Console 应用程序。
- *id*：可选字面值。识别评估标识。您可以使用 listassess (la) 命令来查找评估标识。
- *path*：可选字面值。评估文件的路径和文件名。
- -folder <location>：可选。仅当连接到 V9.0.3 之前的 AppScan Enterprise Server 版本该选项才适用。指定要发布到的 Enterprise Console 文件夹。如果不使用该参数，那么评估将不会被发布到缺省 Enterprise Console 文件夹。
- -name <published_assessment_name>：可选。评估将在 Enterprise Console 中另存为的名称。如果未使用此参数，那么将根据为生成评估而已扫描的 AppScan Source 应用程序来生成名称（将在此名称前追加 AppScan Source:）。
- -preventOverwrite：可选。如果名称相同的评估已存在于服务器上，那么包含该参数可防止发布。

当该可选参数为整数时，命令假设它是评估标识。当它不为整数时，命令假设它是所保存评估文件的路径。

如果未使用 id 或 path 命令来指定评估，那么将采用由最新扫描所生成的评估。

要点：

升级到 AppScan Source V9.0.3.4 时，将注意到以下更改：

- 在将评估发布到 AppScan Enterprise Console 时，现在必须将评估与 AppScan Enterprise 中的应用程序关联（如果您正在运行 AppScan Enterprise Server V9.0.3 和更高版本）。因此，如果自动化脚本不包含应用程序关联，那么它们可能失败。在 AppScan Enterprise Server 中，如果想要利用 AppScan Enterprise Server 应

用程序安全性风险管理功能，那么需要应用程序关联。请参阅http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/c_overview.html。

- 此外，必须从 AppScan Enterprise URL 除去端口。
  1. 在 AppScan Source for Analysis 中，单击**编辑** > **首选项**。
  2. 在 AppScan Enterprise Console 设置中，从 **Enterprise Console URL** 字段除去端口。
- 发布评估之后，它仅可在 AppScan Enterprise"监视器"视图中可用（在先前发行版中，评估可在 AppScan Enterprise"扫描"视图中可用）。http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/t_workflow_for_applications.html中描述了如何迁移到该视图。

这是使用通用访问卡 (CAC) 认证时 AppScan Source 与发布到 AppScan Enterprise Server 所需的 AppScan Enterprise Server 之间的已更改通信协议的结果。

如果不想在启用了 CAC 认证时将评估发布到 AppScan Enterprise Server，或者如果不想利用 Enterprise Server 应用程序安全性风险管理功能，可还原至先前的通信协议，如下所示：

1. 打开 `<data_dir>\config\ounce.ozsettings`（其中 `<data_dir>` 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）。
2. 在此文件中，找到以下设置：

```
<Setting
  name="force_ase902_assessment_publish"
  value="false"
  default_value="false"
  description="Use ASE 9.0.2-style assessment publish"
  display_name="Use ASE 9.0.2-style assessment publish"
  type="boolean"
  read_only="true"
  hidden="true"
/>
```

3. 在此设置中，将 `value="false"` 更改为 `value="true"`，然后保存文件。
4. 重新启动将从其中发布评估的 AppScan Source 产品。

当该设置设置为 `value="true"` 时：

- 如果在发布时将评估与 AppScan Enterprise 中的应用程序关联，那么评估将在"监视器"和"扫描"视图中可用。
- 如果在发布时不将评估与应用程序关联，评估将在"扫描"视图中可用。
- 启用 CAC 认证后您将无法将评估发布到 AppScan Enterprise Server。

有关更多信息，请参阅http://www.ibm.com/support/docview.wss?uid=swg21993010。

quit

描述

结束并关闭 AppScan Source 命令行界面会话。如果您已登录，将发出注销命令。

语法

quit

示例

```
AllApplications>> Quit

>> Logged Out.
>> Exiting Security AppScan Source Command Line Interface...
```

record (rc)

描述

打开或关闭命令记录。

Record 仅捕获发出的命令和参数，但不捕获输出。

提示：要捕获输出，请使用 log 命令。

语法

record {*on* file|*off*}

- on 或 off：需要此参数才能打开或关闭记录。
- file：所记录文件的名称。Record On 必需包含有效文件路径的 file 参数。如果此文件尚不存在，那么 AppScan Source 命令行界面 (CLI) 会予以创建。如果文件已存在，那么日志数据会追加到文件末尾。
- 不带参数的 Record 识别记录是打开还是关闭。

示例

- 记录到名为 MyCommands.txt 的文件：

  ```
  AllApplications>> Record on C:\MyCommands.txt
  ```

- 关闭命令记录：

  ```
  AllApplications>> RC off
  ```

refresh (rf)

描述

Refresh 从磁盘复原当前项目或应用程序。

要重新装入可能已更改的项目或应用程序时，请使用 refresh。

语法

refresh

示例

刷新 MyProject：

```
AllApplications\MyApplication\MyProject>> Refresh
```

register (reg)

描述

向 AppScan Source 数据库注册项目和应用程序。

通过 Register，可以向 AppScan Source 注册当前节点上的应用程序或项目。注册使 AppScan Source 可以识别这些应用程序和项目。可以发布已注册的应用程序和项目（使用 publishassess (pa)）。

注册后，还可使用 unregister (unreg) 注销应用程序或项目。

语法

register

示例

```
AllApplications>> cd pebble
AllApplications\pebble>> register
AllApplications\pebble>> 'pebble' registered successfully.
```

removeassess (da)

描述

从内存中除去选定评估或当前评估。

语法

removeassess [*id*]

id：可选字面值。识别评估标识。如果未包含标识，那么此命令适用于最新评估。使用 listassess (la) 可查找评估标识。

示例

要除去评估，请首先获取评估标识：

```
AllApplications\Applications>> ListAssess
542: putty (Application, Tue Apr 01 08:38:18 EDT 2008)
180: JavaAny (Application, Tue Apr 01 13:17:33 EDT 2008)
```

然后，除去标识为 180 的评估：

```
AllApplications\Applications>> RemoveAssess 180
```

report (rpt)

描述

Report 生成指定类型的 AppScan Source 报告，包括结果报告和 AppScan Source 报告。使用此命令需要有效的 AppScan Source for Automation 许可证。

可用报告输出格式为 HTML、PDF 和 zip。

语法

```
report "<report type>" <output format> <output location>
[<assessment id>] [-includeSrcBefore:<n>] [-includeSrcAfter:<n>]
[-includeTrace:<definitive|suspect|coverage>]
```

- report type：要生成的添加了双引号的报告名称。请指定以下某项：
  - Findings 报告：
    - Findings by Bundle
    - Findings by API
    - Findings by Classification
    - Findings
    - DTS 活动
    - Findings by Type
    - Findings by CWE
    - Findings by File
  - AppScan Source 报告：
    - CWE SANS Top 25 2011
    - DISA Application Security and Development STIG V3R10
    - OWASP Mobile Top 10
    - OWASP Top 10 2013
    - PCI Data Security Standard V3.2
    - Software Security Profile
  - 定制报告（如果可用）。

  在输入添加了双引号的报告类型时，请完全按照以上列表中所指定进行输入 - 例如 Findings by Classification 或 Software Security Profile。
- output format：为此报告指定以下某种格式：
  - html：将报告生成为 HTML 并将其在线显示。
  - zip：创建包含所有 HTML 报告组件的 ZIP 文件
  - 对于 PDF 格式的报告，可以指定详细级别：
    - pdf-summary：包含每个定制报告组的计数
    - pdf-detailed：包含每个漏洞属性的每个 API 的计数
    - pdf-comprehensive：包含由每个 API 的每个结果组成的表
    - pdf-annotated：包含所有结果、结果附带的任何说明以及指定的代码片段
    - output location：用于编写报告的文件路径。
- output location：指定要将报告保存到的绝对路径和文件名。
- assessment id：可选。从 listassess (la) 命令获取的评估标识。如果省略评估标识，报告将从最新扫描生成。
- -includeSrcBefore:<n>：可选。要包含在报告中每个结果之前的源代码的行数。
- -includeSrcAfter:<n>：可选。要包含在报告中每个结果之后的源代码的行数。
- -includeTrace:<definitive|suspect|coverage>：可选。在明确、可疑或扫描覆盖范围结果的报告中包含跟踪信息（请参阅第 141 页的『分类』以了解关于结果分

类的信息）。要包含多个结果分类的跟踪信息，请多次指定该选项。例如，要包含明确和可疑结果的跟踪信息，指定 `-includeTrace:definitive -includeTrace:suspect`。

示例

- 请求将 *Findings by API* 报告写入到 HTML。在报告中，包含明确结果的跟踪信息：

```
AllApplications>> report "Findings by API" html
2C:\reports\findings.html -includeTrace:definitive
```

- 请求使用现有评估 542 将 *OWASP Top 10* AppScan Source 报告写入到包含综合详细信息的 PDF：

```
AllApplications>> report "OWASP Top 10 2013" pdf-comprehensive
/reports/webgoat_OWASP_13_comp.pdf 542
```

scan (sc)

描述

扫描一个应用程序（或所有应用程序）、项目或文件。使用此命令需要有效的 AppScan Source for Automation 许可证。

要点：如果您所处理的是在开发环境中具有依赖性的 AppScan Source 项目（例如 IBM® MobileFirst Platform 项目），请确保在导入该项目之前在开发环境中对其进行构建。导入该项目后，如果您修改其中的文件，请确保在 AppScan Source 中进行扫描之前在开发环境中重新构建该项目（如果不执行此操作，那么 AppScan Source 将忽略对文件做出的修改）。

语法

```
scan [path][config <proj_config>][-name <assessment_name>]
[-scanconfig <scan_configuration_name>]
```

- `path`：可选。所导出评估将另存为的完整路径和文件名 (`.ozasmt`)。

  注：

  - 如果指定有效目录但不指定文件名，那么将根据应用程序名称、项目名称以及创建评估时使用的扫描配置来为您创建评估文件 (`.ozasmt`)。
  - 如果指定有效目录但文件名不存在，那么将使用所指定的文件名在该位置创建评估文件。
  - 如果指定已存在的文件，那么将覆盖现有文件。
  - 如果在不存在的目录中指定文件名 (`.ozasmt`)，那么将不保存任何评估。
- `config <proj_config>`：可选。此参数仅对项目级别评估有效。如果您的项目具有配置文件，请使用此参数来指定配置文件。
- `-name <assessment_name>`：可选。提供评估的名称。该名称在 AppScan Source 客户机产品中用于将评估彼此区分开（例如，在 AppScan Source for Analysis 中，该名称将出现在"我的评估"视图的**名称**列中）。
- `-scanconfig <scan_configuration_name>`：可选。指定要用于扫描的扫描配置的名称。如果未指定扫描配置，那么缺省扫描配置将用于扫描。

示例

- 扫描所有应用程序中项目的缺省配置：

```
AllApplications>> Scan
```

结果显示为：

```
New Scan started
  .
  .
Preparing for Vulnerability Analysis...
Performing Vulnerability Analysis...
Generating Findings...
Preparing project for scan...
  .
  .
Scanned Project:
  Total files: 15
  Total findings: 167
  Total lines: 385
  vkloc: 0.44448395412925595
  v-Density: 22.446439683527426
Scanned Application:
  Total files: 15
  Total findings: 167
  Total lines: 385
  vkloc: 0.44448395412925595
  v-Density: 22.446439683527426
Scan completed:
  Total files: 15
  Total findings: 167
  Total lines: 385
  vkloc: 0.44448395412925595
  v-Density: 22.446439683527426
Elapsed Time - 18 Seconds

New Scan started. Please wait...
Assessment complete
-------------------
Total Call Sites: 75
Total Definitive Security Findings with High Severity: 25
Total Definitive Security Findings with Medium Severity: 37
Total Definitive Security Findings with Low Severity: 9
Total Suspect Security Findings with High Severity: 20
Total Suspect Security Findings with Medium Severity: 80
Total Suspect Security Findings with Low Severity: 60
Total Scan Coverage Findings with High Severity: 50
Total Scan Coverage Findings with Medium Severity: 33
Total Scan Coverage Findings with Low Severity: 17
Total Lines: 3000
...
```

- 扫描 Prj1 的调试配置：

```
AllApplications\Prj1>> SC config debug
```

script (scr)

描述

运行命令脚本。

此脚本可以包含带参数的任何有效 AppScan Source 命令行界面 (CLI) 命令。record (rc) 命令可生成脚本文件。

脚本可以使用 ${VAR} 符号表示法来引用环境变量（其中 VAR 是环境变量的名称）。

当 script 命令从命令行传递到 AppScanSrcCli 时，会在脚本末尾自动退出 AppScanSrcCli。

语法

script script_file

script_file：要运行的脚本文件的完整路径和文件名。

脚本中的注释

以井号 (#) 开头的行是注释。以交互方式或从脚本文件内运行时，CLI 会跳过注释。

以下示例脚本包含注释：

```
login localhost User 123456
#navigate to my workspace
cd MyApplication
#scan my application
scan
logout
```

示例

经常登录并以交互方式使用 CLI 的用户可能希望在从控制台运行的同时记录用于重复执行的脚本。您可以使用 record (rc) 命令对整个会话编制脚本，然后重复或按计划时间间隔运行此脚本。通过允许单个批处理文件或脚本登录、创建项目、扫描、复制文件等等，使用脚本将整个会话自动化。

- 运行名为 myscript.txt 的脚本：

  ```
  AllApplications>> Script c:\myscript.txt
  ```

- 以下示例脚本将登录，打开记录，创建应用程序，导入给定目录下的所有 ppf 项目，运行评估，然后退出：

  ```
  log on C:\mylogfile.log
  new MyApplication C:\myAppFolder
  cd MyApplication
  im c:\Projects\*.ppf
  scan
  logout
  ```

  请注意，如果脚本总是从 AppScanSrcCli 命令行来运行，那么 logout 不是必要的。

setaseinfo (sase)

如果 AppScan Enterprise Server 已与 AppScan Enterprise Console选件一起安装，那么您可以向其发布评估。Enterprise Console 提供各种用于处理评估的工具，例如报告功能、问题管理、趋势分析和仪表板。

描述

指定 Enterprise Console 设置。

如果您打算从 AppScan Source 向 Enterprise Console 发布评估，可使用 setaseinfo 命令指定 Enterprise Console连接设置。

语法

```
setaseinfo --userid|-u <user name>
--password|-p <password>
--url|-l <url>
```

- userid：必需。
  - 如果 AppScan Enterprise Server 认证方法为用户标识和密码，输入用户用户标识。
  - 如果 AppScan Enterprise Server 配置为使用 Windows 认证，请输入用于连接到 Enterprise Console 的域和用户名（以 \ 分隔域和用户名，例如 my_domain\my_username）。
  - 如果 AppScan Enterprise Server 已配置 LDAP，请输入用于连接到 Enterprise Console 的用户名。
  - 在 Windows 上，如果支持 AppScan Enterprise Server进行通用访问卡 (CAC) 认证，输入 common_name<cn_certificate>，其中 common_name 是 CAC 通用名 (CN)，cn_certificate 是证书签发人的 CN。如果 common_name 或 cn_certificate 包含空格，在值的两边加上问号 ("common_name<cn_certificate>")。

  至少您必须是 QuickScan 用户。如果您连接到版本低于 V9.0.3 的 AppScan Enterprise Server，必须在 Enterprise Server 有您自己的用户文件夹。
- 密码：如果 AppScan Enterprise Server 认证方法为用户标识和密码，该字段为必填字段。指定 Enterprise Server 用户标识的密码。

  注：支持 AppScan Enterprise Server 进行 CAC 认证时，不使用该选项。
- url：指定 AppScan Enterprise Server 实例的 URL。该 URL 的格式为 http(s)://<hostname>:<port>/ase，其中 <hostname> 是已安装了 AppScan Enterprise Server 的机器的名称，<port> 是服务器运行所在的端口。该 URL 的示例为 https://myhost.mydomain.ibm.com:9443/ase。

  注：如果已设置 Enterprise Console URL，那么该参数是可选的。

注：

- 您必须登录到 AppScan Source 命令行界面 (CLI)，并具有管理 AppScan Enterprise 设置许可权才能设置 url。有关用户帐户和许可权的信息，请参阅《*IBM Security AppScan Source* 安装和管理指南》的"管理 *AppScan Source*"一节。
- userid 和 password 存储在运行 AppScan Source 客户机（例如，AppScan Source for Analysis）的机器上 - 而 url 则存储在 Enterprise Server（它可能位于远程机器上）中。您无法从远程机器（例如，通过从其发出 getaseinfo 命令）来访问 userid 和 password 信息。

示例

```
AllApplications>> setaseinfo --userid my_username --password my_password
 --url https://my_aseserver/ase
AppScan Enterprise 集成信息已配置。
AllApplications>>
```

setcurrentobject (set, cd)

描述

使用 setcurrentobject 导航对象树。

该命令可设置对象树中当前选择的对象。此对象必须是当前对象的有效子代。要查看当前对象的子对象，请使用 list (ls, dir) 命令。按名称或标识选择对象。

语法

setcurrentobject {name|*id* object id}

- name：要设置的子代的名称。当按对象名浏览树时必需此参数。
- *id*：字面值。按对象标识进行选择。
- object id：要设置的子代的数字标识。按标识选择时必需此参数。

示例

- 从根浏览到标识等于 1 的应用程序：

```
AllApplications>> SetCurrentObject applications
AllApplications>> CD id 1
AllApplications\Applications>>
```

- 后退一级：

```
AllApplications\Applications>> CD ..
```

- 返回到对象树的顶层（根）：

```
AllApplications\Applications>> Set /
```

或

```
AllApplications\Applications>> Set \
```

setvar (sv)

描述

创建新变量或修改现有变量。

语法

setvar <name> <path>

- name：变量的名称。如果变量已存在，那么 AppScan Source 命令行界面 (CLI) 将更新路径。如果变量不存在，那么 CLI 将创建值为指定路径的变量。
- path：变量的现有路径。如果路径不存在，将不会创建变量。

示例

- 创建路径为 C:\Myapps 的 AppScan Source 变量 Myvar：

```
AllApplications>> SetVar Myvar C:\Myapp
AllApplications>> Created new variable '%Myvar%'
```

- 基于 SRC_ROOT 系统环境变量的值创建名为 SRC_ROOT 的 AppScan Source 变量：

```
# login
login localhost admin
# create the variable SRC_ROOT with the value
# of the environment variable SRC_ROOT
setvar SRC_ROOT ${SRC_ROOT}
```

unregister (unreg)

描述

该命令用于从当前节点注销先前注册的应用程序或项目。

注销后，该应用程序或项目将无法发布。有关更多信息，请参阅 register (reg) 命令。

语法

unregister

示例

```
AllApplications\Myapp>> Unregister
AllApplications\Myapp>> 'Myapp' unregistered successfully.
```

运行自动化评估

通过 AppScan Source 命令行界面 (CLI)，可以自动导入 AppScan Source 项目文件 (.ppf) 和扫描源代码。从命令行可以运行脚本，如以下样本：Run_Assessments.txt。

```
AppScanSrcCli scr c:\<install_dir>\bin\Run_Assessments.txt
```

样本 Run_Assessments.txt

```
# Log in.
Login <hostname> <username> <password>
# Turn on logging.
log on c:\myLogFile.log
# Create a new application named "testit."
new testit c:\AppTest
# Navigate to the newly created application.
cd testit
# Import the Project files (.ppfs) under c:\projects\joans.
im c:\projects\joans\*.ppf
# Refresh the Project.
refresh

# Run an assessment.
scan
# Register the assessment
register
# Publish the assessment
publishassess
# Log out and end the CLI session.
quit
```

输出

```
Logging to 'c:\mylogfile.log'...

AllApplications>> new testit c:\AppTest
AllApplications>> cd testit
AllApplications\testit>> import c:\TestApps\testproj\*.ppf
AllApplications\testit>> refresh
AllApplications\testit>> ls
```

```
214: testproj (Project [local])

AllApplications\testit>> la

testit has no current assessments.

scan
New Scan started at 15:41:55
Scanning Project testproj (1 of 1)
Preparing project for scan...
.
.
.

Searching File C:\TestApps\\testproj\src\se\bluefish\blueblog\metarepository\Meta
Category.java (21 of 33)
Searching File C:\TestApps\testproj\src\se\bluefish\blueblog\metarepository\Meta
Repository.java (22 of 33)

-------------------
Total Call Sites: 348
Total Definitive Security Findings with High Severity: 5
Total Definitive Security Findings with Medium Severity: 1
Total Definitive Security Findings with Low Severity: 4
Total Suspect Security Findings with High Severity: 0
Total Suspect Security Findings with Medium Severity: 8
Total Suspect Security Findings with Low Severity: 0
Total Scan Coverage Findings with High Severity: 16
Total Scan Coverage Findings with Medium Severity: 27
Total Scan Coverage Findings with Low Severity: 16
Total Lines: 7386
Max V-Density: 732.2772813430815
Max V/kloc: 10.42512862171676
V-Density: 732.2772813430815
V/kloc: 10.42512862171676

AllApplications\testit>> register
AllApplications\testit>> 'testit' registered successfully.
AllApplications\testit>> pa

Assessment Successfully Published.

AllApplications\testit>> la

AllApplications\testit>> 27001: testit (Application, Fri Mar 14 15:41:55 EDT 2008)

AllApplications\testit>>
```

第 3 章 Ounce/Ant 构建工具

此部分描述如何使用 Ounce/Ant，它是用于集成 AppScan Source 和 Apache Ant 的 AppScan Source 构建实用程序。将 Ounce/Ant 与 Ant 环境集成有助于将构建和代码评估自动化。

Ounce/Ant 是一种集成到 Apache Ant 构建环境中的 AppScan Source 构建工具。Ounce/Ant 自动生成包含所有必要信息（如源文件和类路径）的 AppScan Source 项目和应用程序，而不是在 AppScan Source for Analysis 中手动创建和配置 AppScan Source 项目和应用程序文件。然后，AppScan Source 可以扫描项目和应用程序文件。

Apache Ant V1.7 或更高版本需要与 Ounce/Ant 一起使用。

Ounce/Ant 和 Apache/Ant 集成

此任务主题中概述的步骤对于将 Ounce/Ant 集成到 Apache/Ant 构建环境中是必需的。

过程

1. 将 ounceant.jar 文件和 ant-contrib-1.0b3.jar（后者为可选）复制到 ant lib 目录。

   AppScan Source 安装将 ounceant.jar 和 ant-contrib-1.0b3.jar 放置在 <install_dir>\lib（其中 <install_dir> 是 AppScan Source 安装位置）中。
2. 可选： 覆盖 build.compiler 属性。

   请参阅第 59 页的『创建项目』，以了解有关覆盖 build.compiler 的更多信息。）

   使用以下某种方法覆盖此属性：

   - 在 build.xml 文件中使用 property 标记。

     ```
     <property name="build.compiler"
     value="com.ouncelabs.ounceant.OunceCompilerAdapter"/>
     ```
   - 使用 -D 选项调用 Ant 时，在命令行上指定 build.compiler 的值。

     ```
     ant -Dbuild.compiler=
     com.ouncelabs.ounceant.OunceCompilerAdapter
     ```
   - 在文本文件中包含 build.compiler 的值，并且指示 Ant 使用 properties 选项在该文件中装入属性，如 Ant 文档中所述。
3. 创建 taskdefs。

   要使用 Ounce/Ant 任务，必须在 taskdef 任务中引用 ounceant.jar。例如，

   ```
   <taskdef resource="com/ouncelabs/ounceant/task/ounce.xml"
   classpath="ounceant.jar"/>
   ```

   要使用 var 任务，ant-contrib-1.0b3.jar 必须按如下所示引用 taskdef 任务：

   ```
   <taskdef resource="net/sf/antcontrib/antlib.xml"/>
   ```

Ounce/Ant 属性

本主题中的表描述了可选的 Ounce/Ant 属性。

属性	有效值/缺省值	描述
ounce.default. configuration_name	<config_name>	AppScan Source 项目中创建的配置名称。 如果未设置，请使用 Configuration 1。
ounce.build.compiler	编译器名称	Ant build.compiler 属性配置为使用 OunceCompilerAdapter 时要使用的编译器可执行文件的名称。 如果未设置，请使用 javac。
ounce.project_name	name	项目名称。 如果未设置，Ounce/Ant 将生成名称。请参阅第 60 页的『命名项目』以了解更多信息。
ounce.project_dir （必需此属性以设置 ounce.project_name）	<project_dir>	项目工作目录。 如果未设置，请使用当前构建文件目录。
ounce.application_name	<app_name>	应用程序名称。请参阅第 61 页的『创建和命名应用程序』。
ounce.application_dir （必需此参数以设置 ounce.application_name）	<app_dir>	应用程序工作目录。 必须设置 ounce.application_name 和 ounce.application_dir 以向应用程序中添加项目。
ounce.projects. store_full_paths （可选）	true 或 false	存储应用程序和项目中创建的绝对路径。 如果未设置，请使用相对路径。

设置属性

您可以使用 var 任务来设置任意次数的属性。调用 ounceCreateProject 时（无论显式还是隐式），它将使用当前属性值。您可以通过此任务主题中概述的某种方法来设置属性。

过程

- 使用 Ounce/Ant 任务的属性（如果该属性存在）。

- 在相应的 AppScan Source 任务之前使用 var 任务。通过 var 设置的属性仅针对设置了这些属性的文件有效。要使用 var 任务，Ant 添加库必须在 Ant lib 目录中，并且 taskdef 任务必须对其进行引用。

  ```
  <taskdef resource="net/sf/antcontrib/antlib.xml"/>
  ```

- 在 Ant 命令行上使用 -D。
- 将属性放置在构建属性文件（通常称为 build.properties）中。

创建项目

ounceCreateProject 任务会生成 AppScan Source 项目。

显式或隐式使用 ounceCreateProject：

- 在 Ant 构建文件中显式调用 ounceCreateProject。
- 通过覆盖 build.compiler 属性，在每次调用 javac 任务时隐式调用 ounceCreateProject。

可以将项目分组到应用程序中。您可以使用可选的应用程序属性或相应的 Ounce/Ant 属性将创建的项目放置到应用程序中。

此外，也可以使用 Ounce/Ant 来扩充项目。可以将多个项目的项目属性合并到一个 AppScan Source 项目中。如果多个 javac 或 ounceCreateProject 要扩充同一项目，请保留相同的名称和目录。

注：如果覆盖 build.compiler 以创建项目，请首先运行 ant clean。

请参阅 第 60 页的『ounceCreateProject 任务示例』，以了解属性及其嵌套属性的用法。

ounceCreateProject

ounceCreateProject 接受以下参数和嵌套元素：

指定为属性的参数	描述
name	项目的名称。
workingDir	项目的工作目录。
classpath	项目的类路径。
sourcepath	项目源依赖关系的源路径。不会扫描这些文件中是否存在漏洞。
jdkName	扫描项目时要使用的 AppScan Source JDK 名称。（必须在 AppScan Source for Analysis 中创建）
appName	包含此项目的应用程序（可选）。
appDir	应用程序目录（可选）

指定为嵌套元素的参数	描述
ounceSourceRoot	指定项目源根
ounceWeb	指定项目 Web 内容

指定为嵌套元素的参数	描述
ounceExclude	允许从父元素中指定的文件集排除特定文件。

ounceCreateProject 任务示例

```
<ounceCreateProject
  name="myProjectName"
  workingDir="${sandbox}/myProject"
  classpath="${my.class.path}"
  sourcepath="${my.source.path}"
  jdkName="jdk15"
  appName="myApp"
  appDir="${sandbox}>
<ounceSourceRoot dir="src"/>
<ounceExclude dir="src/test"/>
<ounceExclude file="src/mydir/Bad.java"/>
<ounceSourceRoot dir="src2"/>
<ounceSourceRoot file="src3/mydir.java"/>
<ounceWeb webContextRoot="web/myProject.war"/>
<ounceExclude dir="web/test"/>
<ounceExclude file="web/partial.jsp"/>
</ounceCreateProject>
```

ounceSourceRoot

ounceSourceRoot 接受指定为属性的以下参数：

指定为属性的参数	描述
dir	有效 Java™ 源根的路径
file	个别文件的路径

ounceWeb

ounceWeb 接受指定为属性的以下参数。

指定为属性的参数	描述
webContextRoot	有效 Web 上下文根或 war 文件的路径。

ounceExclude

ounceExclude 接受指定为属性的以下参数：

指定为属性的参数	描述
dir	要排除的目录的路径
file	要排除的个别文件的路径

命名项目

您可以使用缺省命名约定或通过手动创建项目名称来命名项目。

缺省项目命名

Ounce/Ant 缺省命名方案允许生成唯一项目名称。

```
<directory>_<target_name>[_<number>]
```

- <directory>：项目目录的名称（不包括完整路径）
- <target name>：当前目标的名称
- <number>：从 1 开始的整数，并且在必要时递增以保证其唯一性。请注意，仅当存在冲突时，<number> 才会出现。

示例

通过以下参数，项目名称成为 test_compile。如果在编译目标中创建了两个项目，那么第二个项目名称成为 test_compile_1。

```
<working_directory> = C:\mydir\test
<directory> = test
<target_name> = compile
```

手动命名项目

如果通过 ounceCreateProject 创建项目，那么可以使用 name 属性来指定项目名称。如果未指定 name 属性，或者使用 javac 来创建项目，那么 AppScan Source 任务会查看 ounce.project_name 属性，如果设置了此属性，那么它使用其值作为项目名称。因此，在每次调用 javac 任务之前，可以使用 var 来设置 ounce.project_name 属性，从而为创建的每个项目提供唯一名称。

创建和命名应用程序

Ounce/Ant 可以在操作期间创建 AppScan Source 应用程序并将创建的 AppScan Source 项目分组到应用程序中。

您通过两个 Ant 属性来创建这些应用程序：

- ounce.application_name
- ounce.application_dir

要点：要创建应用程序，必须同时设置上述两个属性，或设置 ounceCreateProject 任务的 appName 和 AppDir 属性。

如果在项目创建期间设置了这些属性，那么会将项目添加到具有由 ounce.application_name 指定的名称和由 ounce.application_dir 指定的工作目录的应用程序。

您可以使用 var 任务多次设置这些属性，以将项目分组到您需要的应用程序中。相应的应用程序名称指示将多个项目合并成单个应用程序。

提示：如果使用 ounceCreateProject，那么还可以使用 appName 和 appDir 属性来指定要在其中包含项目的应用程序。

构建集成

ounceCli 任务在构建期间允许访问 AppScan Source 命令行界面 (CLI) 以进行扫描。ounceCli 任务可以从 Ant 调用 cli。

ounceCli 需要以下指定为属性的参数：

- dir：AppScan Source 安装目录的位置
- script：要运行的 cli 脚本文件的位置
- output：cli 输出文件的位置

示例

```
<ounceCli
dir="${installDir}"
  script="${scripts}/cli_script.txt"
  output="log.txt"/>
```

第 4 章 AppScan Source Data Access API

Data Access API 提供对 AppScan Source 所生成的评估结果（包括结果和结果详细信息）的访问权。它还提供对评估度量值如分析日期和时间、代码行、V-Density 和结果数的访问权。

Data Access API 包含在以下 AppScan Source 组件的安装中：

- AppScan Source for Analysis
- AppScan Source 命令行界面 (CLI)

Data Access API 安装为 `<install_dir>\sdk\ouncesdk.jar`（Windows 上）和 `<install_dir>/sdk/ouncesdk.jar`（Linux（其中 `<install_dir>` 是 AppScan Source 安装位置） 上）。

Data Access API 需要 JDK 1.5 或更高版本。

在 Windows 上：运行使用 SDK 的程序时，请在命令行上指定以下 Java 虚拟机 (JVM) 参数：

```
java -classpath
<install_dir>\lib\avalon-framework-4.1.5.jar;
<install_dir>\lib\commons-lang3-3.3.2.jar;
<install_dir>\lib\icu4j-52_1.jar;
<install_dir>\lib\jacorb.jar;
<install_dir>\lib\log4j-1.2.8.jar;
<install_dir>\lib\logkit-1.2.jar;
<install_dir>\sdk\ouncesdk.jar;
<install_dir>\lib\xml-apis.jar;
<install_dir>\lib\saxon9.jar
... com.company.product.ClassName
```

在 Linux 上：运行使用 SDK 的程序时，请在命令行上指定以下 Java 虚拟机 (JVM) 参数：

```
java -classpath
<install_dir>/lib/avalon-framework-4.1.5.jar:
<install_dir>/lib/commons-lang3-3.3.2.jar:
<install_dir>/lib/icu4j-52_1.jar:
<install_dir>/lib/jacorb.jar:
<install_dir>/lib/log4j-1.2.8.jar:
<install_dir>/lib/logkit-1.2.jar:
<install_dir>/sdk/ouncesdk.jar:
<install_dir>/lib/xml-apis.jar:
<install_dir>/lib/saxon9.jar
... com.company.product.ClassName
```

Data Access API 对象模型

图 1 - 详细描绘评估的对象模型的 UML（统一建模语言）图

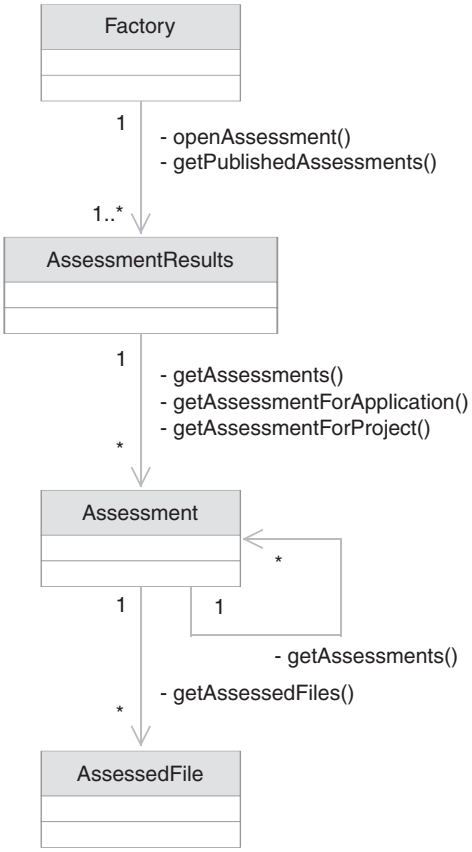


图 2 - 详细描绘结果的对象模型的 UML（统一建模语言）图

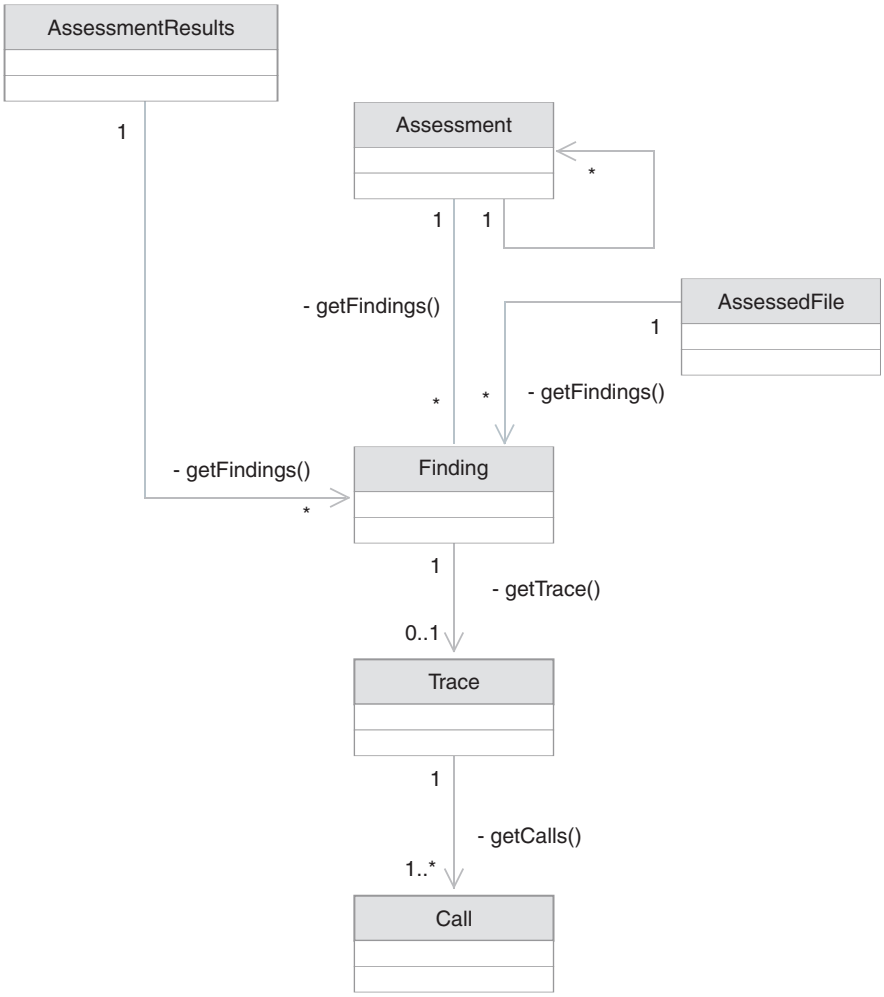


图 1 - 详细描绘评估的对象模型的 UML（统一建模语言）图

此图的顶部为 Factory 类。 Factory 类显示为通过 Factory 类中的两个方法 openAssessment() 和 getPublishedAssessments() 来与 AssessmentResults 类相关联。这两个类之间的关系是一个 Factory 类与一个或多个 AssessmentResults 类的关系。

此图在 Factory 类之下描绘了 AssessmentResults 类。 AssessmentResults 类显示为通过三个方法 getAssessments()、getAssessmentForApplication() 和 getAssessmentForProject() 来与 Assessment 类相关联。这两个类之间的关系是一个 AssessmentResults 类与零个或更多 Assessment 类的关系。

此图在 AssessmentResults 类之下描绘了 Assessment 类。一个 Assessment 类显示为通过 getAssessedFiles() 方法与零个或更多 AssessedFile 类相关联。Assessment 类还描绘了其通过 getAssessments() 方法与自身的关系。此关系显示为一个 Assessment 类与零个或更多子 Assessment 类的关系。

此图在 Assessment 类之下描绘了 AssessedFile 类。一个 Assessment 类通过 getAssessedFiles() 方法与零个或更多 AssessedFile 类相关联。

图 2 - 详细描绘结果的对象模型的 UML（统一建模语言）图

此图的顶部有三个类：AssessmentResults、Assessment 和 AssessFile。所有这三个类均描绘为通过 getFindings() 方法来与 Finding 类相关联。此关系显示为一个类与零个或更多 Finding 类。 Assessment 类还描绘了与其自身的关系，即一个 Assessment 类与零个或更多子 Assessment 类的关系。

此图在 Finding 类之下描绘了 Trace 类。一个 Finding 类通过 getTrace() 方法与零个或一个 Trace 类相关联。

此图在 Trace 类之下描绘了 Call 类。一个 Trace 类通过 getCalls() 方法与一个或许多个 Call 类相关联。

使用 Data Access API

您可以在 <install_dir>\sdk\sample\com\ouncelabs\sdk\sample（其中 <install_dir> 是 AppScan Source 安装位置）内包含的 SamplePublished.java 和 SampleSdk.java 文件中找到多个数据访问 API 方案的完整示例。

Data Access API 类和方法

此部分详细描述了 AppScan Source Data Access 类和方法。这些类包括：

- com.ouncelabs.sdk.Factory
- com.ouncelabs.sdk.assessment.AssessedFile
- com.ouncelabs.sdk.assessment.Assessment
- com.ouncelabs.sdk.assessment.AssessmentDiff
- com.ouncelabs.sdk.assessment.AssessmentResults
- com.ouncelabs.sdk.assessment.Finding
- com.ouncelabs.sdk.assessment.Trace
- com.ouncelabs.sdk.assessment.Call
- com.ouncelabs.sdk.assessment.SeverityType
- com.ouncelabs.sdk.assessment.ClassificationType
- com.ouncelabs.sdk.assessment.AssessmentFilter
- com.ouncelabs.sdk.assessment.OunceException

AssessedFile

com.ouncelabs.sdk.assessment.AssessedFile 类表示个别文件的评估。它提供对文件的评估数据的访问权，如其结果和统计信息。

AssessedFile.getFindings

签名

```
public Finding[] getFindings()
```

描述

提供对此 AssessedFile 的所有结果的访问权。

返回值

此评估文件的结果对象数组。

抛出内容

如果 Data Access API 无法检索结果，那么将抛出 OunceException。

AssessedFile.getStats

签名

```
public AssessmentStats getStats()
```

描述

提供对此文件的累积统计信息的访问权，如结果的总数、总行数等等。

返回值

包含此评估的统计信息的 AssessmentStats 对象。

抛出内容

如果 Data Access API 无法检索统计信息，那么将抛出 OunceException。

AssessedFile.getFilename

签名

```
public String getFilename()
```

描述

提供对此 AssessedFile 所表示文件的绝对路径的访问权。

返回值

包含此 AssessedFile 所表示文件的绝对路径的字符串。

Assessment

com.ouncelabs.sdk.assessment.Assessment 类表示应用程序或项目的评估。

Assessment.getFindings

签名

```
public Finding[] getFindings()
```

描述

提供对此评估的所有结果的访问权。

返回值

此评估的结果对象数组。

Assessment.getStats

签名

```
public AssessmentStats getStats()
```

描述

完整的统计信息列表，包括：已扫描的文件总数、已发现的漏洞总数、扫描时间和漏洞密度。

返回值

包含此评估的统计信息的 AssessmentStats 对象

Assessment.getAssessments

签名

```
public Assessment[] getAssessments()
```

描述

返回此评估的子 Assessment 对象。由于项目评估没有评估子代，因此仅对应用程序评估对象才会返回评估子对象。

返回值

- 对于应用程序评估，返回 Assessment 对象数组，应用程序中已扫描的每个项目都对应有一个对象。
- 对于项目评估，返回空数组。

抛出内容

如果 Data Access API 无法检索评估，那么将抛出 OunceException。

Assessment.getFiles

签名

```
public AssessedFile[] getFiles()
```

描述

提供对属于此评估的每个文件的访问权。

返回值

AssessedFile 对象数组。每个对象表示属于此评估的一个文件。

抛出内容

如果 Data Access API 无法检索评估文件，那么将抛出 OunceException。

Assessment.getFileByPath

签名

```
public AssessedFile getFileByPath(filePath)
```

描述

按路径提供对评估文件的访问权。

返回值

与指定文件路径对应的 AssessmentFile 对象，而如果指定文件路径与评估文件不对应，那么将返回 null。

抛出内容

如果 Data Access API 无法检索评估文件，那么将抛出 OunceException。

Assessment.getAssesseeName

签名

```
public String getAssesseeName()
```

描述

提供生成此评估的来源应用程序或项目的名称。

返回值

应用程序或项目的名称。

AssessmentDiff

com.ouncelabs.sdk.assessment.AssessmentDiff 类包含两个评估之间的差异，从而提供两个评估之间的变化量。

AssessmentDiff.getCommonFindings

签名

```
public Finding[] getCommonFindings()
```

描述

获取两个评估的公共结果。

AssessmentDiff.getLostFindings

签名

```
public Finding[] getLostFindings()
```

描述

获取最旧评估中有而最新评估中没有的结果。

AssessmentDiff.getNewFindings

签名

```
public Finding[] getNewFindings()
```

描述

获取最新评估中有而最旧评估中没有的结果。

AssessmentFilter

使用 com.ouncelabs.sdk.assessment.AssessmentFilter 类可在检索已发布的评估时指定过滤条件。

AssessmentFilter.AssessmentFilter

签名

```
public AssessmentFilter(int maxResults)
```

描述

仅指定要检索的已发布评估最大数的构造函数。您可以通过在类中调用 set 方法来指定其他过滤选项（例如，setUserName() 用于按已发布的用户进行过滤）。

参数

- maxResults：要返回的最大结果数。

AssessmentFilter.AssessmentFilter

签名

```
public AssessmentFilter(String appName,
String userName,
Double dateProximityDuration,
int DateProximityUnit,
Long dateRangeStart, Long dateRangeEnd,
int maxResults)
```

描述

采用所有条件选项作为参数的构造函数。

参数

- appName：应用程序名称。（如果不是按 appName 进行过滤，那么值为 null。）
- userName：已发布应用程序的用户。（如果不是按用户名过滤，那么值为 null。）
- dateProximityDuration：当与 dateProximityUnit 配对时，指的是从当前日期开始要过滤的单位数（天、周等等）。（如果不是按日期距离进行过滤，那么值为 null。）
- dateProximityUnit：当与 dateProximityDuration 配对时，指的是用于计数的单位，如天、周等等。如果指定了 dateProximityDuration，那么必需此参数。请参阅 第 76 页的『DateProximityUnit.value』 以了解有效单位。
- dateRangeStart：日期范围的起始值。（如果不是按日期范围进行过滤，那么值为 null。）

- dateRangeEnd：日期范围的结束值（如果不是按日期范围进行过滤，那么值为 null。）
- maxResults：要返回的最大结果数。

AssessmentResults

com.ouncelabs.sdk.assessment.AssessmentResults 类表示整个评估。

AssessmentResults.getFindings

签名

```
public Finding[] getFindings()
```

描述

提供对此 AssessmentResults 的所有结果的访问权。

返回值

此评估的结果对象数组。

抛出内容

如果 Data Access API 无法检索结果，那么将抛出 OunceException。

AssessmentResults.getAssessments

签名

```
public Assessment[] getAssessments()
```

描述

针对评估为属于此 AssessmentResults 的每个应用程序，提供对其评估数据的访问权。

返回值

Assessment 对象数组，属于 AssessmentResults 的每个应用程序都对应有一个对象。

抛出内容

如果 Data Access API 无法检索评估，那么将抛出 OunceException。

AssessmentResults.getStats

签名

```
public AssessmentStats getStats()
```

描述

提供对此评估的累积统计信息的访问权，如已评估的文件总数、总行数等等。

返回值

包含此评估的统计信息的 AssessmentStats 对象。

抛出内容

如果 Data Access API 无法检索评估统计信息，那么将抛出 OunceException。

AssessmentResults.getAssessmentForApplication

签名

```
public Assessment[] getAssessmentForApplication(String applicationName)
```

描述

提供对指定应用程序的评估数据的访问权。

参数

applicationName：应用程序名称。

返回值

Assessment 对象数组，与指定名称匹配的每个应用程序都对应有一个对象。

抛出内容

如果 Data Access API 无法检索评估，那么将抛出 OunceException。

AssessmentResults.getAssessmentForProject

签名

```
public Assessment[]
getAssessmentForProject(String projectName, String applicationName)
```

描述

提供对指定应用程序中指定项目的评估数据的访问权。

参数

projectName：项目名称。

applicationName：项目驻留所在的应用程序的名称。

返回值

Assessment 对象数组，与指定名称匹配的每个项目都对应有一个对象。

抛出内容

如果 Data Access API 无法检索评估，那么将抛出 OunceException。

AssessmentResults.getName

签名

```
public String getName()
```

描述

返回 AssessmentResults 的名称。

返回值

包含 AssessmentResults 的名称的字符串。

AssessmentResults.getMessages

签名

```
public Message[] getMessages()
```

描述

提供对在 AppScan Source for Analysis 中运行评估时控制台中出现的所有状态消息的访问权。

返回值

Message 对象数组。

抛出内容

如果 Data Access API 无法检索消息，那么将抛出 OunceException。

Call

com.ouncelabs.sdk.assessment.Call 类表示 AppScan Source 跟踪 中的节点。

Call.getCalls

签名

```
public Call[] getCalls()
```

描述

提供对在此调用的上下文中所进行调用的访问权。

返回值

Call 对象数组（按照其调用顺序）

抛出内容

如果 Data Access API 无法检索调用，那么将抛出 OunceException。

Call.getFilename

签名

```
public String getFilename()
```

描述

提供对发生此调用所在的文件名的访问权。

返回值

此调用的文件名。

Call.getLineNumber

签名

```
public int getLineNumber()
```

描述

提供对发生调用所在的行号的访问权。

返回值

此调用的行号。

Call.getColumnNumber

签名

```
public int getColumnNumber()
```

描述

此调用提供对发生调用所在的列号的访问权。

返回值

发起调用的列号。

Call.getSignature

签名

```
public String getSignature()
```

描述

提供对已调用 API 的签名的访问权。

返回值

此调用的签名。

Call.getSrcContext

签名

```
public String getSrcContext()
```

描述

提供对调用的源上下文的访问权。

返回值

此调用的源上下文。

Call.getMethodName

签名

```
public String getMethodName()
```

描述

提供对已调用的方法名称的访问权。

返回值

此调用的方法名称。

Call.getClassName

签名

```
public String getClassName()
```

描述

提供对被调用方法所属类名的访问权。

返回值

此调用方法的类名。

Call.getTraceType

签名

```
public TraceType getTraceType()
```

描述

提供对此调用的跟踪类型的访问权。例如，跟踪类型可以指定调用是源还是接收器。

返回值

表示调用的跟踪类型的 TraceType 对象。

ClassificationType

com.ouncelabs.sdk.assessment.ClassificationType 类表示结果的分类。

静态成员

- ClassificationType.Vulnerability
- ClassificationType.Type1

- ClassificationType.Type2
- ClassificationType.Unknown

ClassificationType.value

签名

```
public int value()
```

描述

提供对此 ClassificationType 的值的访问权。返回的值对应于静态成员的某个值。

返回值

此 ClassificationType 对象的分类值。

DateProximityUnit

当 com.ouncelabs.sdk.assessment.DateProximityUnit 与 dateProximityDuration 配对时，指的是用于计数的单位，如天、周等等。如果指定了 dateProximityDuration，那么必需此参数。此部分中描述了有效单位。

静态成员

- DateProximityUnit.Hour
- DateProximityUnit.Day
- DateProximityUnit.Week
- DateProximityUnit.Month
- DateProximityUnit.Year

DateProximityUnit.value

签名

```
public EnumType value()
```

描述

提供对此 DatProximityUnit 的值的访问权。返回的值对应于静态成员的某个值。

返回值

此 DateProximityUnit 对象的日期值。

Factory

com.ouncelabs.sdk.Factory 提供用于初始化、登录和打开评估的方法。此类是 Data Access API 的入口点。

Factory.init

签名

```
public void init(String installDir, boolean acceptInvalidSSLCerts)
```

描述

初始化 Data Access API。必须先调用 Factory.init，再调用任何其他方法。

参数

- installDir：AppScan Source 安装目录（请参阅第 139 页的『缺省安装位置』）的路径。
- acceptInvalidSSLCerts：如果想要接受无效 SSL 证书（SSL 证书将不需要验证），请将布尔值设置为 true。

抛出内容

如果初始化失败，那么将抛出 OunceException。通常，仅在您指定错误目录时才会发生这种情况。

Factory.shutdown

签名

```
public void shutdown()
```

描述

退出 Data Access API。如果调用了 Factory.init，那么必须调用 Factory.shutdown，Data Access API 才能正确关闭。如果未调用，那么 OunceScanner.exe 可能会继续作为游离进程运行。

Factory.clearCache

签名

```
public void clearCache()
```

描述

此调用会清除 Data Access API 的高速缓存。处理大量结果导致内存使用量过多时，应定期使用此调用。

Factory.login

签名

```
public void login(String user name, String password, String host name[int :port])
```

描述

登录到在指定 URL 或指定主机名上运行的 AppScan Enterprise Server。您必须登录，然后才能将评估发布到 AppScan Source 数据库。

参数

- username：登录的用户的用户名。
- password：登录的用户的密码。
- hostname：要登录到的 AppScan Enterprise Server 的 URL 或主机名。hostname 可以是 IP 地址或域名。

如有必要，可以通过追加 :<port> 来指定 AppScan Enterprise Server 端口号。例如 public void login("hwall", "shhh2008", "MyHost:2880")。

抛出内容

如果登录失败，那么将抛出 OunceException。

Factory.openAssessment

签名

```
public AssessmentResults openAssessment(String pathname)
```

描述

打开通过路径名指定的评估。

参数

pathname：评估文件的文件路径。

返回值

表示通过 pathname 指定的评估的 AssessmentResults 对象。

抛出内容

- 如果指定的文件名不存在或无法打开，那么将抛出 FileNotFoundException。
- 如果 Data Access API 无法装入评估文件，那么将抛出 OunceException。

Factory.getPublishedAssessments

签名

```
public AssessmentResults[] getPublishedAssessments(AssessmentFilter filter)
```

描述

提供对已发布的评估的访问权。采用可用于指定要检索的已发布评估集的 AssessmentFilter 对象。

参数

filter：指定要检索的已发布评估集的 AssessmentFilter 对象。

返回值

与指定过滤器匹配的 AssessmentResults 对象数组。

抛出内容

如果 Data Access API 无法检索评估结果，那么将抛出 OunceException。

Factory.diffAssessments

签名

```
public AssessmentDiff diffAssessments
(AssessmentResults result1, AssessmentResults result2)
```

描述

查找两个评估结果之间的差异。

抛出内容

OunceException。

Finding

com.ouncelabs.sdk.assessment.Finding 类表示评估中的个别结果。它提供对结果的所有关联数据的访问权，如分类和严重性。

Finding.getFilename

签名

```
public String getFilename()
```

描述

提供对此结果所在文件名的访问权。

返回值

包含此结果所在文件的绝对路径的字符串。

Finding.getLineNumber

签名

```
public int getLineNumber()
```

描述

提供对结果所在行号的访问权。

返回值

此结果的行号。

Finding.getApiName

签名

```
public String getApiName()
```

描述

提供对此结果的 API 名称的访问权。

返回值

此结果的 API 名称。

Finding.getCallerName

签名

```
public String getCallerName()
```

描述

针对包含此结果所表示 API 的调用的方法，提供对其签名的访问权。

返回值

调用者的签名。

Finding.getClassification

签名

```
public ClassificationType getClassification()
```

描述

提供对结果的当前分类的访问权。

返回值

表示结果的分类的 ClassificationType 对象

Finding.getSeverity

签名

```
public SeverityType getSeverity()
```

描述

在结果中获取严重性的当前值。

返回值

表示结果的严重性的 SeverityType 对象。第 86 页的『SeverityType.value』 中对 SeverityType 进行了描述。

Finding.getVulnerabilityType

签名

```
public String getVulnerabilityType()
```

描述

提供对结果的当前漏洞类型的访问权。

返回值

此结果的漏洞类型。AppScan Source 安全知识库中对诸如缓冲区溢出或 SQL 注入之类的漏洞类型进行了描述。

Finding.getOriginalClassification

签名

```
public ClassificationType getOriginalClassification()
```

描述

提供对结果的原始分类的访问权。如果结果未修改，那么 getOriginal 与 getClassification 相同。

返回值

表示结果的原始分类（如果已修改分类）的 ClassificationType 对象。

Finding.getOriginalSeverity

签名

```
public SeverityType getOriginalSeverity()
```

描述

获取结果的严重性的原始值。如果结果未修改，那么 getOriginalSeverity 与 getSeverity 相同。

返回值

表示结果的原始严重性（其中严重性已修改）的 SeverityType 对象。

Finding.getOriginalVulnerabilityType

签名

```
public String getOriginalVulnerabilityType()
```

描述

提供对结果的原始漏洞类型的访问权。如果结果未修改，那么 getOriginalVulnerabilityType 与 getVulnerability 相同。

返回值

表示结果的原始漏洞类型（其中漏洞已修改）的 VulnerablilityType 对象。

Finding.getModifiedClassification

签名

```
public ClassificationType getModifiedClassification()
```

描述

提供对已修改的结果分类的访问权。

返回值

表示结果的分类（其中分类已修改）的 ClassificationType 对象。（如果不匹配，那么返回 null。）

Finding.getModifiedSeverity

签名

```
public SeverityType getModifiedSeverity()
```

描述

获取已修改的严重性值。如果已修改，那么与 getSeverity() 相同。

返回值

表示结果的严重性（其中严重性已修改）的 SeverityType 对象。（如果不匹配，那么返回 null。）

Finding.getModifiedVulnerabilityType

签名

```
public String getModifiedVulnerabilityType()
```

描述

提供对已修改的结果漏洞类型的访问权。

返回值

表示结果的漏洞类型（其中漏洞已修改）的 VulnerablilityType 对象。（如果不匹配，那么返回 null。）

Finding.getSrcContext

签名

```
public String getSrcContext()
```

描述

提供对此结果的源上下文的访问权。

返回值

此结果的源上下文。

Finding.getDefectSubmissionUser

签名

```
public String getDefectSubmissionUser()
```

描述

提供对上次将此结果提交到外部缺陷跟踪系统的用户名的访问权。请注意，它是缺陷跟踪系统用户名，可能不同于 AppScan Source 用户名。

返回值

上次将此结果提交到缺陷跟踪系统时的提交用户。

Finding.getDefectSubmissionDate

签名

```
public String getDefectSubmissionDate()
```

描述

提供对上次将此结果提交到外部缺陷跟踪系统时的日期字符串的访问权。

返回值

上次将此结果提交到缺陷跟踪系统时的提交日期字符串。

Finding.getDefectInfo

签名

```
public String getDefectInfo()
```

描述

提供对上次将此结果提交到外部缺陷跟踪系统时的外部缺陷跟踪系统标识的访问权。

返回值

带有与此结果关联的缺陷标识的字符串。

Finding.getProperties

签名

```
public String[] getProperties()
```

描述

此调用提供对与结果关联的属性的访问权。这些属性提供有关此结果的漏洞类型以及有关此结果的 API 的信息。

返回值

属性数组：

- `Vulnerability.value`：Findings 视图中报告的漏洞类型，例如：

  ```
  Vulnerability.BufferOverflow
  Vulnerability.Injection.SQL
  ```

- `Mechanism.value`：API 安全性机制；用于定义定制报告类别，例如：

  ```
  Mechanism.AccessControl
  Mechanism.Cryptography
  ```

- `Technology.value`：API 技术；用于定义定制报告类别，例如：

  ```
  Technology.Communications.HTTP
  Technology.Database
  ```

- `Attribute.value`：API 方法标记，例如：

  ```
  Attribute.Deprecated
  Attribute.Modifier.Protected
  ```

请参阅 AppScan Source 安全知识库，以了解有关属性值的详细信息。

抛出内容

如果 Data Access API 无法检索属性，那么将抛出 `OunceException`。

Finding.isExcluded

签名

```
public boolean isExcluded()
```

描述

指示是否从评估中排除了此结果。已排除的结果不添加到评估的任何统计信息中。

返回值

如果已排除，那么返回 `true`；如果未排除，那么返回 `false`。

Finding.getTrace

签名

```
public Trace getTrace()
```

描述

提供与此结果关联的 AppScan Source 跟踪（如果有）的访问权。

返回值

结果的跟踪对象（如果有）；如果没有，那么返回 null。

抛出内容

如果 Data Access API 无法检索跟踪，那么将抛出 `OunceException`。

Finding.getNotes

签名

```
public String getNotes()
```

描述

提供对结果说明的访问权。

返回值

结果的任何说明，如果没有注释，那么返回 null。

抛出内容

如果 Data Access API 无法检索说明，那么将抛出 OunceException。

Trace

com.ouncelabs.sdk.assessment.Trace 类表示结果的 AppScan Source 跟踪。

Trace.getCalls

签名

```
public Call[] getCalls()
```

描述

提供对 AppScan Source 跟踪的根调用的访问权。

返回值

Call 对象数组。当前总是包含单个对象。

抛出内容

如果 Data Access API 无法检索调用，那么将抛出 OunceException。

SeverityType

com.ouncelabs.sdk.assessment.SeverityType 类表示结果的严重性。

静态成员

严重性类型包括：

- SeverityType.High
- SeverityType.Medium
- SeverityType.Low
- SeverityType.Info
- SeverityType.Unknown

SeverityType.value

签名

```
public int value()
```

描述

提供对此 SeverityType 的值的访问权。返回的值对应于静态成员的某个值。

返回值

此 SeverityType 对象的严重性值。

OunceException

com.ouncelabs.sdk.assessment.OunceException 是 Data Access API 使用的异常类。要了解有关异常的更多信息，请在异常上调用 getMessage()。

第 5 章 Ounce/Maven 插件

此部分描述 Ounce/Maven 插件，它使用 Maven（一种 Apache 构建工具）将 AppScan Source 集成到 Maven 工作流程中。

通过 Ounce/Maven 插件，可以创建 AppScan Source 应用程序和项目文件。如果还安装了 AppScan Source for Automation，那么使用 Ounce/Maven 可运行源代码安全扫描并生成综合性安全报告。

有关 AppScan Source for Automation 的信息，请参阅 第 93 页的第 6 章, 『AppScan Source for Automation』。

有关 AppScan Source 系列产品的更多信息，请参阅 http://www.ibm.com/software/rational/products/appscan/source/。

安装 Ounce/Maven

开始之前

以下是安装和运行 Ounce/Maven 的必备软件：

- Apache Maven V2.x 或更高版本：有关部署和使用 Maven 插件的信息，请参阅 http://maven.apache.org/ 处的 Apache Maven 项目 Web 站点。
- 要使用 Maven 插件进行扫描，需要 AppScan Source for Automation：有关更多信息，请参阅第 93 页的第 6 章, 『AppScan Source for Automation』。

关于此任务

安装并配置 Maven 后，首次引用 Ounce/Maven 时它会进行下载。

Ounce/Maven 站点文档包含 Ounce/Maven 目标、其参数、示例、使用说明和详细示例的描述。

您可以在 http://mojo.codehaus.org/ounce-maven-plugin 处找到站点文档。

过程

1. 如果尚未安装 Maven，请从 http://maven.apache.org/ 安装 Maven。遵循 Maven Web 站点上的指示。
2. 执行以下任一操作：
   - 编辑 Maven pom 文件以包含一个或多个 Ounce/Maven 目标，如 Ounce/Maven 站点文档中所述
   - 从命令行调用一个或多个 Ounce/Maven 目标，如 Ounce/Maven 站点文档中所述

使用 Ounce/Maven

通过 Ounce/Maven 插件，您可以使用 Ounce/Maven 来创建 AppScan Source 项目和应用程序，扫描应用程序，发布所生成的评估，并生成 AppScan Source 报告。与对任何其他 Maven 插件一样指定 Ounce/Maven 目标和参数。

您可以通过两种方法调用 Ounce/Maven 命令：

- 使用 Maven pom（构建）文件：通过 pom 文件，您可以创建 AppScan Source 应用程序和项目文件作为构建的一部分。安装 Ounce/Maven 后，您可以修改 Maven pom 文件以根据 AppScan Source 任务的需要指定 ounce:application 和 ounce:project-only 目标。
- 从命令行：从命令行调用 ounce:project、ounce:scan 和 ounce:report 目标，以创建 AppScan Source 项目文件（或覆盖 pom 文件中的项目文件参数），启动 AppScan Source 扫描，发布评估，并生成 AppScan Source 报告。

每个 Ounce/Maven 目标包含多个参数。

- 有关 Ounce/Maven 目标的信息，请参阅 第 89 页的『Ounce/Maven 目标』。
- 有关每个目标的参数的信息，请参阅 http://mojo.codehaus.org/ounce-maven-plugin 处的 Ounce/Maven 站点文档。

Ounce/Maven 方案

通过 Ounce/Maven，您可以：

- 创建 AppScan Source 应用程序和项目文件
- 扫描应用程序并将结果（评估）发布到 AppScan Source 数据库
- 从评估结果生成报告
- 将 AppScan Source 报告集成到 site 目标中

此部分对这些任务进行了简短描述。有关特定于 AppScan Source 的概念和语言的详细信息，请参阅 *AppScan Source for Analysis* 用户指南。

创建应用程序和项目文件

AppScan Source 使用应用程序和项目模型来管理数据，如下所示：

- **应用程序**：应用程序包含配置数据和可定制信息，以及该应用程序中的项目列表。
- **项目**：项目由文件集（包括源代码）及其相关信息（如配置数据）组成。要扫描项目，该项目必须是应用程序的一部分。

ounce:application 目标创建 AppScan Source 应用程序 (.paf) 文件 - 而 ounce:project 和 ounce:project-only 目标则创建 AppScan Source 项目 (.ppf) 文件。

请参阅 *AppScan Source for Analysis* 用户指南，以了解有关其他应用程序和项目文件格式的信息。

扫描应用程序

AppScan Source 扫描会分析源代码中是否存在安全漏洞。扫描的结果是评估，它是 XML 文件。

注：有关 AppScan Source 功能的详细信息，请参阅 *IBM Security AppScan Source for Analysis* 用户指南。

从命令行使用 `ounce:scan` 目标可扫描应用程序及其项目，还可选择通过评估生成报告。

扫描完成后，通过 Ounce/Maven，您可以：

- 将评估发布到 AppScan Source 数据库。这使评估结果可供具有数据库访问权和必要特权的其他用户使用。
- 生成报告。

报告

对于许多用户而言，来自 AppScan Source 的首选输出是提供有关评估的数据的详细报告。通过 `ounce:report` 目标，可以从新的或先前的评估生成这些报告。如果需要，`ounce:report` 目标可以扫描应用程序，然后发布评估或生成有关评估的报告。不过，与 `ounce:scan` 不同，您可以使用 `ounce:report` 来生成现有评估的报告。

将报告与站点目标集成

要将 `ounce:report` 与站点目标集成，请将 Ounce/Maven 插件添加到 `pom` 的报告部分中。站点目标执行 `ounce:report`，并且报告输出成为应用程序的站点文档的一部分。

与站点目标集成时，无需指定任何其他参数。您可以将报告类型指定为 第 91 页的『reportType 值』中描述的某个值。如果不指定报告类型，那么缺省报告为 `Findings`。

Ounce/Maven 目标

Ounce/Maven 提供以下目标以执行 AppScan Source 功能：

- `ounce:application`：生成一个 `pom` 应用程序文件，其中包含对 AppScan Source 文件所定义全部子项目的引用。需要应用程序以进行扫描（并因此进行报告）。
- `ounce:project`：根据 Maven 子项目的数量创建一个或更多 AppScan Source 项目文件。 `ounce:project` 目标旨在从命令行运行并将 Maven 构建合并到该目标中。
- `ounce:project-only`：根据 Maven 子项目的数量创建一个或更多 AppScan Source 项目文件。 `ounce:project-only` 目标旨在将 AppScan Source 项目文件的创建集成到 Maven 构建生命周期中。
- `ounce:scan`：扫描应用程序。您可以选择发布评估或生成报告。从命令行运行 `ounce:scan` 目标。
- `ounce:report`：生成 AppScan Source 报告。如果刷新结果时需要，请在生成报告之前执行扫描。从命令行运行 `ounce:report` 目标。

注：

- 要使应用程序和项目文件可移植，请创建路径变量以将文件路径映射到其所在位置。

- 有关如何使用 Ounce/Maven 目标的示例，请参阅 http://mojo.codehaus.org/ounce-maven-plugin/ 处的 Ounce/Maven 站点文档。

ounce:application

描述

生成一个 AppScan Source 应用程序文件，其中包含对 pom 文件所定义全部子项目的引用。需要应用程序以进行扫描和报告。尽管 ounce:application 不创建 AppScan Source 项目文件，但您可以运行 ounce:project-only 以在同一构建文件内创建项目。

ounce:project

描述

从命令行使用 ounce:project。ounce:project 目标将根据 Maven 子项目的数量创建一个或多个 AppScan Source 项目文件。

使用 ounce:project

从命令行调用 ounce:project 目标时，它自动构建任何所需的依赖性。如果不存在应用程序文件，那么 ounce:project 会构建此文件。

ounce:project 的所有参数都是可选的。项目名称由 Maven artifactId 确定，不能将其覆盖。

从包含顶级 pom.xml 文件的目录（或者，如果仅需要构建的子集，从包含子 pom.xml 的子目录）运行带有 ounce:project 目标的 Maven。

ounce:project-only

描述

ounce:project-only 目标从 pom 文件内进行使用。ounce:project-only 目标将根据 Maven 子项目的数量创建一个或更多 AppScan Source 项目文件。

使用 ounce:project-only

ounce:project-only 的所有参数都是可选的。项目名称由 Maven artifactId 确定，不能将其覆盖。

ounce:scan

描述

ounce:scan 目标可生成给定应用程序的扫描操作，并且随后可选择为所生成的评估创建报告。从命令行运行 ounce:scan。

注： ounce:scan 目标不能用作 Maven 构建生命周期的一部分。

使用 ounce:scan

指定 ounce:scan 时，可以请求 Ounce/Maven 在扫描后立即从评估运行报告。在此情况下，请指定 『reportType 值』和 『reportOutputType 值』中描述的报告参数。如果指定 reportType，那么还必须指定 reportOutputType 和 reportOutputPath。

注：如果需要，ounce:scan 目标将创建或更新应用程序和项目文件。

ounce:report

描述

ounce:report 目标从评估生成报告。如果不指定现有评估，那么 ounce:report 在生成报告之前将运行 ounce:scan。从命令行运行 ounce:report。

指定 『reportType 值』和 『reportOutputType 值』中描述的报告参数。如果指定 reportType，那么还必须指定 reportOutputType 和 reportOutputPath。

reportType 值

- Findings 报告：
  - Findings by Bundle
  - Findings by API
  - Findings by Classification
  - Findings
  - DTS 活动
  - Findings by Type
  - Findings by CWE
  - Findings by File
- AppScan Source 报告：
  - CWE SANS Top 25 2011
  - DISA Application Security and Development STIG V3R10
  - OWASP Mobile Top 10
  - OWASP Top 10 2013
  - PCI Data Security Standard V3.2
  - Software Security Profile
- 定制报告（如果可用）。

reportOutputType 值

- 为此报告指定以下某种格式：
  - html：将报告生成为 HTML 并将其在线显示。
  - zip：创建包含所有 HTML 报告组件的 ZIP 文件。
- 对于 PDF 格式的报告，可以指定详细级别：
  - pdf-summary：包含每个定制报告组的计数
  - pdf-detailed：包含每个漏洞属性的每个 API 的计数

- pdf-comprehensive：包含由每个 API 的每个结果组成的表
- pdf-annotated：包含所有结果、结果附带的任何说明以及指定的代码片段
- pdf-annotated：将已注释的报告生成为 PDF 文件。

第 6 章 AppScan Source for Automation

通过 自动化服务器 (ounceautod)，您可以在软件开发生命周期 (SDLC) 过程中使 AppScan Source 工作流程的关键方面自动化，并且将安全性集成到构建环境中。通过自动化服务器，您可以将扫描和发布评估的请求进行排队，并生成有关应用程序代码安全性的报告。

通过 AppScan Source for Automation 客户机命令行可执行文件 (ounceauto) 将请求提交到服务器。处理请求时，自动化服务器 作为关联 AppScan Enterprise Server 的客户机来运行，并且只能连接到单个 AppScan Enterprise Server。它在 TCP 端口（缺省值为 13205）上侦听仅来自本地主机的连接。

- 在 Windows 系统上，自动化服务器 作为 **IBM Security AppScan Source Automation** 服务来运行。
- 在 Linux 系统上，它作为守护程序来运行：
  - 要停止此守护程序，请发出以下命令：`/etc/init.d/ounceautod stop`
  - 要启动此守护程序，请发出以下命令：`/etc/init.d/ounceautod start`

自动化服务器 以指定的 AppScan Source 用户身份处理请求并因此继承该用户的许可权。此用户标识必须具有其所需的所有许可权，具体取决于其所需要运行的命令。例如，如果该用户标识需要运行 `PublishAssessment` 命令，那么可以向此用户标识授予发布和注册许可权，而不需要授予扫描许可权（请参阅《*AppScan Source* 安装与管理指南》的"管理 *AppScan Source*"一节以获取更多详细信息）。将请求提交到 自动化服务器 无需用户凭证。

从命令行指定 自动化服务器 登录凭证

如果您未在安装过程中按《IBM Security AppScan Source 安装和管理员指南》中所述指定 自动化服务器 登录凭证，那么必须在安装后配置 自动化服务器 才能作为 AppScan Source 用户来运行。

关于此任务

在该任务主题中，

- `<install_dir>` 是 AppScan Source 安装位置.
- `user_id` 是 自动化服务器 在处理请求时用于认证的用户。必须在 AppScan Source 中通过所需的许可权对用户进行定义。

  注：如果用户尚不存在，那么您将需要在安装面板中或通过命令行来指定该用户 - 然后通过 AppScan Source for Analysis 或 AppScan Source 命令行界面 (CLI) 手动创建该用户（安装后）。要通过命令行创建新用户，请使用 第 38 页的『newuser (nu)』 命令。创建新用户时，请确保指定的用户名和密码与为 自动化服务器 登录指定的用户名和密码相同。其他设置（如许可权）可根据需要进行设置。
- `password` 是用户的密码。
- `--persist` 选项在磁盘上保留登录凭证，并通过指定用户名和密码创建加密秘钥文件。

- 仅 Windows：common_name 是您 CAC 常用名称 (CN)。
- 仅 Windows：cn_certificate 是证书签发者的 CN。

过程

- **在 Windows 系统上：**
  - 如果 AppScan Enterprise Server 认证方法为用户标识和密码，发出以下命令：

    ```
    <install_dir>\bin\ounceautod.exe -u <user name> -p <password> --persist
    ```

  - 如果 AppScan Enterprise Server 支持通用访问卡 (CAC) 认证，输入以下命令：

    ```
    <install_dir>\bin\ounceautod.exe -u <common_name<cn_certificate>> --persist
    ```

    注：如果 common_name 或 cn_certificate 包含空格，在值的两边加上问号 ("common_name<cn_certificate>")。
- **在 Linux 系统上：**
  1. 将 LD_LIBRARY_PATH 更改为包含 <install_dir>/bin（其中 <install_dir> 是 AppScan Source 安装位置），例如：

     ```
     export LD_LIBRARY_PATH=/opt/ibm/appscansource/
     bin:$LD_LIBRARY_PATH
     ```

  2. 发出以下命令：

     ```
     <install_dir>/ibm/appscansource/bin/ounceautod
                    -u <user_id> -p <password> --persist
     ```

结果

指定登录凭证后，它们将保存到磁盘。

自动化服务器 配置文件

ounceautod 配置文件 ounceautod.ozsettings 指定 自动化服务器 守护程序的属性。ounceautod.ozsettings 文件驻留在 <data_dir>\config（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）中。

ounceautod.ozsettings 文件包含以下属性：

属性	值	描述
ounceautod_max_ concurrent_requests	1（缺省值）	允许的并发请求数。
ounceautod_accept_ssl	true（缺省值）	自动接受 SSL 证书。更多相关信息，请参阅第 35 页的『AppScan Enterprise Server SSL 证书』。
ounceautod_port	13205（缺省值）	要在其上运行 ounceautod 的端口号。 **注：**此设置中的端口值必须与 <data_dir>\config 中 ounceauto.ozsettings 文件内的端口设置匹配。

属性	值	描述
ounceautod_server_hostname	<hostname>:port	已安装 AppScan Enterprise Server 的计算机的主机名及其在该计算机上运行所在的端口。

如果修改了该文件，那么必须重新启动 自动化服务器 服务（在 第 93 页的第 6 章, 『AppScan Source for Automation』 中进行了描述）。

自动化服务器 记录

自动化服务器 生成日志文件 ounceautod.log，此文件记录对 自动化服务器 的每个请求以及响应。日志条目驻留在 <data_dir>\logs\ounceautod.log（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述） 中。

自动化服务器 针对所有请求记录下列事件：

- 已收到的请求
- 已启动的请求
- 已完成的请求
- 已失败的请求

各日志条目包含：

- 日志条目时间戳记
- 事件类型
- 请求类型
- 请求标识
- 调用者（如果指定）
- 特定于请求的信息

从命令行使用 Ounceauto

AppScan Source for Automation 的接口 ounceauto 用于向命令行参数指定的服务器发出命令，打印输出（如果有），然后返回。每个请求分配有请求标识，并且与请求有关的所有日志条目都包含该请求标识以便识别。

从命令行运行 ounceauto，如下所示：

```
ounceauto <command name> <command arguments>
```

Ounceauto 命令包括：

- GenerateReport
- PublishAssessment
- PublishAssessmentASE
- ScanApplication
- Wait

GenerateReport

描述

从评估创建报告。

语法

```
ounceauto GenerateReport
-assessment <assessment path>
-type <report type>
-output <output format>
-file <output location>
[-caller <caller>]
[-includeSrcBefore <n>]
[-includeSrcAfter <n>][-includeTraceDefinitive]
[-includeTraceSuspect]
[-includeTraceCoverage]
```

- `-assessment <assessment path>`：要为其生成报告的评估文件的路径。
- `-type "<report type>"`：添加了双引号的报告类型名称。报告类型包含 Findings 报告、AppScan Source 报告和定制报告。

  AppScan Source 报告类型包括：

  – Findings 报告：
    - `Findings by Bundle`
    - `Findings by API`
    - `Findings by Classification`
    - `Findings`
    - `DTS 活动`
    - `Findings by Type`
    - `Findings by CWE`
    - `Findings by File`
  – AppScan Source 报告：
    - `CWE SANS Top 25 2011`
    - `DISA Application Security and Development STIG V3R10`
    - `OWASP Mobile Top 10`
    - `OWASP Top 10 2013`
    - `PCI Data Security Standard V3.2`
    - `Software Security Profile`
  – 定制报告（如果可用）。

  在输入添加了双引号的报告类型时，请完全按照以上列表中所指定进行输入 - 例如 `Findings by Classification` 或 `Software Security Profile`。
- `-output <output format>`：为此报告指定以下某种格式，
  – `html`：将报告生成为 HTML 并将其在线显示。
  – `zip`：创建包含所有 HTML 报告组件的 ZIP 文件
  – 对于 PDF 格式的报告，可以指定详细级别：

  - pdf-summary：包含每个定制报告组的计数
  - pdf-detailed：包含每个漏洞属性的每个 API 的计数
  - pdf-comprehensive：包含由每个 API 的每个结果组成的表
  - pdf-annotated：包含所有结果、结果附带的任何说明以及指定的代码片段
  - output location：用于编写报告的文件路径。
- -file <output location>：指定要将报告保存到的路径和文件名。
- -caller <caller>：可选。为报告生成操作指定调用者。调用者可以是实际用户的名称，但这不是必需的。调用者名称将写入 ounceauto 日志文件。
- -includeSrcBefore <n>：可选。要包含在每个结果之前的源代码的行数。
- -includeSrcAfter <n>：可选。要包含在每个结果之后的源代码的行数。
- -includeTraceDefinitive：可选。在明确结果的报告中包含跟踪信息（请参阅第 141 页的『分类』以了解关于结果分类的信息）。
- -includeTraceSuspect：可选。在可疑结果的报告中包含跟踪信息。
- -includeTraceCoverage：可选。在扫描覆盖范围结果的报告中包含跟踪信息。

返回值

如果成功，那么返回请求标识；如果请求提交不成功，那么返回 -1。

示例

- 将 Findings by API 报告生成为 HTML 文件。在报告中，包含明确结果的跟踪信息：

```
ounceauto generatereport -assessment C:\Ounce\Data\Webgoat.ozasmt
-type "Findings by API" -output html
-file C:\reports\Webgoat_Findings.html-includeTraceDefinitive
```

- 要将 OWASP Top 10 2013 AppScan Source 报告生成为 PDF：

```
ounceauto generatereport -assessment C:\Ounce\Data\Webgoat.ozasmt
-type "OWASP Top 10 2013" -output pdf-annotated
-file C:\Reports\Webgoat_OWASP.pdf
```

PublishAssessment

描述

将所选评估发布到 AppScan Source 数据库。

语法

```
ounceauto PublishAssessment - file <assessment.ozasmt>
[-caller <caller>]
```

- -file <assessment file>：评估文件的完整路径。
- -caller <caller>：可选。为报告生成操作指定调用者。调用者可以是实际用户的名称，但这不是必需的。调用者名称写入到 ounceauto 日志文件。

返回值

如果成功，那么返回请求标识；如果请求提交不成功，那么返回 -1。

示例

发布 WebGoat_Internal 评估：

```
ounceauto publishassessment -file C:\Ounce\Data\WebGoat_Internal.ozasmt
```

PublishAssessmentASE

描述

将所选评估发布到 AppScan Enterprise Console。

语法

```
ounceauto PublishAssessmentASE -file <assessment_file>
[-aseapplication <ase_application>] [-caller <caller>]
[-folder <location>] [-name <published_assessment_name>]
[-preventOverwrite]
```

- -file <assessment_file>：必需。评估文件的路径和文件名。
- -aseapplication <ase_application>：当连接到 AppScan Enterprise Server V9.0.3 和更高版本时，需要该选项（除非禁用需求，如此处所示）。当连接到 AppScan Enterprise Server 的较低版本时，关联应用程序是可选的。使用该选项可指定要与评估关联的 Enterprise Console 应用程序。
- -caller <caller>：可选。为报告生成操作指定调用者。调用者可以是实际用户的名称，但这不是必需的。调用者名称将写入 ounceauto 日志文件。
- -folder <location>：可选。仅当连接到 V9.0.3 之前的 AppScan Enterprise Server 版本该选项才适用。指定要发布到的 Enterprise Console 文件夹。如果不使用该参数，那么评估将不会被发布到缺省 Enterprise Console 文件夹。
- -name <published_assessment_name>：可选。评估将在 Enterprise Console 中另存为的名称。如果未使用此参数，那么将根据为生成评估而已扫描的 AppScan Source 应用程序来生成名称（将在此名称前追加 AppScan Source:）。
- -preventOverwrite：可选。如果名称相同的评估已存在于服务器上，那么包含该参数可防止发布。

返回值

如果成功，那么返回请求标识；如果请求提交不成功，那么返回 -1。

示例

要将 WebGoat_Internal 评估发布到 AppScan Enterprise Server V9.0.3 或更高版本：

```
ounceauto publishassessmentase -file C:\Ounce\Data\WebGoat_Internal.ozasmt
-aseapplication myapplication
```

要点：

升级到 AppScan Source V9.0.3.4 时，将注意到以下更改：

- 在将评估发布到 AppScan Enterprise Console 时，现在必须将评估与 AppScan Enterprise 中的应用程序关联（如果您正在运行 AppScan Enterprise Server V9.0.3 和更高版本）。因此，如果自动化脚本不包含应用程序关联，那么它们可能失败。在 AppScan Enterprise Server 中，如果想要利用 AppScan Enterprise Server 应

用程序安全性风险管理功能，那么需要应用程序关联。请参阅http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/c_overview.html。

- 此外，必须从 AppScan Enterprise URL 除去端口。
  1. 在 AppScan Source for Analysis 中，单击**编辑** > **首选项**。
  2. 在 AppScan Enterprise Console 设置中，从 **Enterprise Console URL** 字段除去端口。
- 发布评估之后，它仅可在 AppScan Enterprise"监视器"视图中可用（在先前发行版中，评估可在 AppScan Enterprise"扫描"视图中可用）。http://www.ibm.com/support/knowledgecenter/SSW2NF_9.0.3/com.ibm.ase.help.doc/topics/t_workflow_for_applications.html中描述了如何迁移到该视图。

这是使用通用访问卡 (CAC) 认证时 AppScan Source 与发布到 AppScan Enterprise Server 所需的 AppScan Enterprise Server 之间的已更改通信协议的结果。

如果不想在启用了 CAC 认证时将评估发布到 AppScan Enterprise Server，或者如果不想利用 Enterprise Server 应用程序安全性风险管理功能，可还原至先前的通信协议，如下所示：

1. 打开 <data_dir>\config\ounce.ozsettings（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）。
2. 在此文件中，找到以下设置：

```
<Setting
  name="force_ase902_assessment_publish"
  value="false"
  default_value="false"
  description="Use ASE 9.0.2-style assessment publish"
  display_name="Use ASE 9.0.2-style assessment publish"
  type="boolean"
  read_only="true"
  hidden="true"
/>
```

3. 在此设置中，将 value="false" 更改为 value="true"，然后保存文件。
4. 重新启动将从其中发布评估的 AppScan Source 产品。

当该设置设置为 value="true" 时：

- 如果在发布时将评估与 AppScan Enterprise 中的应用程序关联，那么评估将在"监视器"和"扫描"视图中可用。
- 如果在发布时不将评估与应用程序关联，评估将在"扫描"视图中可用。
- 启用 CAC 认证后您将无法将评估发布到 AppScan Enterprise Server。

有关更多信息，请参阅http://www.ibm.com/support/docview.wss?uid=swg21993010。

ScanApplication

描述

扫描指定应用程序并执行与扫描相关的其他操作。

语法

```
ounceauto ScanApplication
-application <name of application>|
  -application_file <path to application file>
[-name <assessment name>]
[-scanconfig <scan_configuration_name>]
[-save <filename>]
[-caller <caller>]
[-publish]
[-clearcache]
[-report <report type> <output format>
<output location>]
[-includeSrcBefore <n>]
[-includeSrcAfter <n>][-includeTraceDefinitive]
 on[-includeTraceSuspect]
[-includeTraceCoverage][-appserver_type][-include_all_lib_jars]
[-include_lib_jars]
[-no_ear_project]
```

- -application <name of application> **或** -application_file <path to application file>：需要其中一项。
  - 如果指定 -application <name of application>，请指定要扫描的应用程序的名称。
  - 如果指定 -application_file <path to application file>，请指定以下某个文件类型的完整路径和文件名：
    - AppScan Source 应用程序文件 (.paf)。
    - Eclipse 或 Rational Application Developer for WebSphere Software (RAD) 工作空间 (.ewf)

      注：当您使用 openapplication 来打开工作空间目录（通过指定其路径）时，将生成 .ewf 文件。
    - WAR 文件 (.war)
    - EAR 文件 (.ear)
    - 仅 Windows：Microsoft Visual C++ 工作空间文件 (.dsw)
    - 仅 Windows ：Microsoft Visual Studio.NET 解决方案文件 (.sln)

    注：要了解 AppScan Source for Analysis、AppScan Source for Automation 和 AppScan Source 命令行界面 支持哪些版本的导入文件，请参阅http://www.ibm.com/support/docview.wss?uid=swg27027486。在此页面中，选择您在使用的 AppScan Source 版本所对应的选项卡，然后选择您在使用的 AppScan Source 组件。如果 AppScan Source 支持从其他开发环境打开和扫描文件，该支持将在**受支持软件**选项卡的**编译器和语言**部分中列出。
- -name <assessment name>：可选。评估的名称。
- -scanconfig <scan_configuration_name>：可选。指定要用于扫描的扫描配置的名称。如果未指定扫描配置，那么缺省扫描配置将用于扫描。
- -save <filename>：可选。将评估结果保存到此文件。
- -caller <caller>：可选。生成操作指定调用者。调用者可以是实际用户的名称，但这不是必需的。调用者名称将写入 ounceauto 日志文件。
- -publish：可选。扫描后发布评估。
- -clearcache：可选。在扫描之前除去漏洞分析高速缓存和定制规则签名数据。如果已启用了 Java 递增分析，那么扫描将是完整扫描。

- -report：可选。扫描后生成报告。
  - 必需的 -report 命令选项：
    - <report type>：报告的类型。报告类型包含 Findings 报告、AppScan Source 报告和定制报告。请参见 第 96 页的『GenerateReport』 中的选项。
    - <output format>：指定报告格式。请参见 第 96 页的『GenerateReport』 中的选项。
    - <output location>：用以保存报告的位置。
  - 可选的 -report 命令选项：
    - -includeSrcBefore <n>：要包含在每个发现项目之前的源代码的行数。
    - -includeSrcAfter <n>：要包含在每个发现项目之后的源代码的行数。
    - -includeTraceDefinitive：在明确结果的报告中包含跟踪信息（请参阅第 141 页的『分类』以了解关于结果分类的信息）。
    - -includeTraceSuspect：在可疑结果的报告中包含跟踪信息。
    - -includeTraceCoverage：在扫描覆盖范围结果的报告中包含跟踪信息。
- -appserver_type：可选。如果要打开的应用程序包含 JavaServer Pages（例如 WAR 或 EAR 文件），使用该设置来指定应用程序服务器以用于 JSP 编译。指定以下某项（用双引号括起）：
  - Tomcat 7
  - Tomcat 8
  - WebSphere 7.0
  - WebSphere 8.0
  - WebSphere 8.5
  - WebLogic 11g
  - WebLogic 12c

  注：
  - 指定应用程序之前，确保已在 AppScan Source for Analysis 首选项中对其进行正确的配置。
  - 如果未使用 -appserver_type，在 AppScan Source for Analysis 中当前设置的缺省 JSP 编译器将用于 JSP 编译。现成可用的 Tomcat 7 是缺省 JSP 编译器。
- 对于 WAR 文件：
  - -include_all_lib_jars：使用该设置可在扫描期间在 WAR 文件中包含所有库。
  - -include_lib_jars：使用该设置可在 WAR 文件中指定扫描期间想要包含的库。使用该设置时，请勿包含库路径信息，并通过逗号分隔多个库。
- -no_ear_project：导入 EAR 文件时，将自动创建用于存储共享库的项目。 如果没有共享库，那么将创建项目，但项目将为空。使用该设置时，将不会为 EAR 文件创建项目。

返回值

如果成功，那么返回请求标识；如果请求提交不成功，那么返回 -1。

示例

- 扫描 WebGoat 应用程序，将其发布，然后以 John Smith 作为调用者来注释日志：

```
ounceauto scanapplication -application_file C:\WebGoat\WebGoat.paf
-publish -caller JohnSmith
```

- 扫描 WebGoat 应用程序并在 C:\WebGoat directory 中创建 Findings 报告。在报告中，包含明确结果的跟踪信息：

```
ounceauto scanapplication -application WebGoat
-report Findings html C:\WebGoat\MyReport.html-includeTraceDefinitive
```

- 扫描 WAR 文件并仅包含其库中的某些库：

```
ounceauto scanapplication -application_file c:\mywar.war
-include_lib_jars lib1.jar,lib2.jar
```

Wait

描述

阻塞直至指定请求完成为止。

语法

ounceauto wait -requestid <request_id>

-requestid <request_id>：要等待的请求的标识。

返回值

如果与 <request_id> 对应的请求成功完成，那么返回值为 0，否则为 -1。

示例

以下 Windows 示例说明应用程序的扫描，然后等待扫描完成。

```
ounceauto scanapplication -application_file WG0.paf
ounceauto wait -requestid %errorlevel%
```

在 Linux，此示例等同于：

```
ounceauto scanapplication -application_file WG0.paf
ounceauto wait -requestid $?
```

第 7 章 Framework for Frameworks 处理 API

AppScan Source 提供了一组 Java API，使您可以添加对您应用程序中使用的框架的支持。在这些 API 中提供的类和方法使您可以处理没有提供内置支持的框架。

注：AppScan Source 中具有针对以下框架的内建支持：

- Apache Struts 1 和 2
- Spring MVC 2.5 和 3
- ASP .NET MVC（仅限 Windows）
- Enterprise JavaBeans (EJB) 2
- ASP .NET（仅限 Windows）
- J2EE
- JavaServer Faces (JSF) 2
- .NET 4.5（仅限 Windows）
- Jax - RS（V1.0 和 V1.1）
- Jax - WS (V2.2)

在现代框架中，许多影响应用程序运行时行为的信息已从源代码移到配置文件或注释中。在过去，这可能会在进行静态分析时造成盲点。产品团队可以为单个应用程序创建定制规则，但是，没有框架可以灵活的自动描述这些框架的活动。

通过使用 Framework for Frameworks API，您可以迅速而又轻松地在 AppScan Source 中为新框架添加支持。实现方法是处理框架的关联配置信息，然后将该数据通过关联的 API 返回给 AppScan Source。

Framework for Frameworks API 包含在以下这些产品的安装中：

- AppScan Source for Automation
- AppScan Source for Analysis
- AppScan Source for Development

这些 API 将安装到 `<install_dir>\walalib`（其中 `<install_dir>` 是 AppScan Source 安装位置）。

示例项目归档安装在 `<data_dir>\samples\F4FEjbExample.zip`（其中 `<data_dir>` 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述） 中。

注：类名以 `Appscan.Synthetic`、`Appscan.Synthetic.Validator` 和 `AppScan.Synthetic.Replacement` 开始的跟踪节点对应于由 AppScan Source 合成的方法。

- `AppScan.Synthetic` 方法用于在使用框架的应用程序代码中将跟踪聚合到一起。
- `AppScan.Synthetic.Validator` 方法可对框架运行时执行的底层验证进行建模。如果需要，您可以选择一个 validator 方法并将其标记为**验证器**。

- AppScan.Synthetic.Replacement 方法指示应用程序代码中的某个方法已替换为 AppScan Source，以捕获框架的不相关组件（如控制器和视图）之间的数据流。

主要 Framework for Frameworks API 组件

F4FHandler

F4FHandler 是所有新框架处理程序都必须扩展的抽象类。它会通过必须覆盖的抽象方法实施框架处理程序工作流程。同时还会提供对其他两个 API 组件 F4FApp 和 F4FAction 的访问权。

F4FApp

提供有关所扫描应用程序的信息，包括其配置中的信息（如源根和 Web 根）。此组件还会提供对应用程序中每个类的访问权，以供确定必要信息，如注释和超类。

F4FAction

为 AppScan Source 提供有关所扫描应用程序的信息。通过调用此 API 中提供的方法，可向扫描引擎提供入口点、抽象方法和其实施之间的连接的信息，以及其他更多信息。

使用 Framework for Frameworks API

此部分重点说明使用 Framework for Frameworks API 时需要执行的步骤。在可行的情况下，这些步骤将与提供的示例结合，以供您参考学习。

在此示例中，我们将：

- 创建扩展 F4FHandler 的类
- 实施 F4FHandler 的两个抽象方法
- 创建包含指定您的处理程序类的清单文件的 Java 归档 (.jar)
- 将 .jar 导出到 AppScan Source waflgens 目录
- 『关于示例』
- 第 105 页的『将示例项目导入 Eclipse 或 Rational Application Developer for WebSphere Software (RAD)』
- 第 108 页的『创建实施 F4FHandler 的类』
- 第 109 页的『为处理程序创建清单文件』
- 第 109 页的『为处理程序创建 JAR』
- 第 112 页的『将 JAR 导出到 waflgens』

关于示例

AppScan Source 安装中包含了可以读取和提供有关企业 Java Bean (EJB) 2 应用程序（也称为企业 Java Bean，或简称为 Bean）的 Framework for Frameworks 信息的样本处理程序。

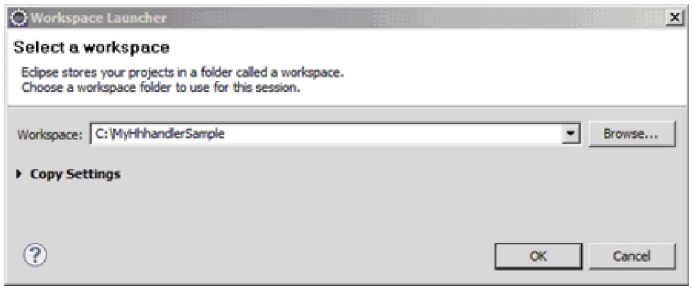
EJB 是一种旨在简化业务逻辑组件的使用和复用的框架。每个 bean 都表示一种业务组件，必须有其他 Bean 必须使用以与其交互的关联接口类。这些接口通过应用程序的

EJB 配置文件 (EJB-jar.xml) 与 Bean 相关联。这会创造一种盲点，因为静态分析引擎不会拾取源中的这些关联，因而使 EJB 框架特别适合通过 Framework for Frameworks 来处理。

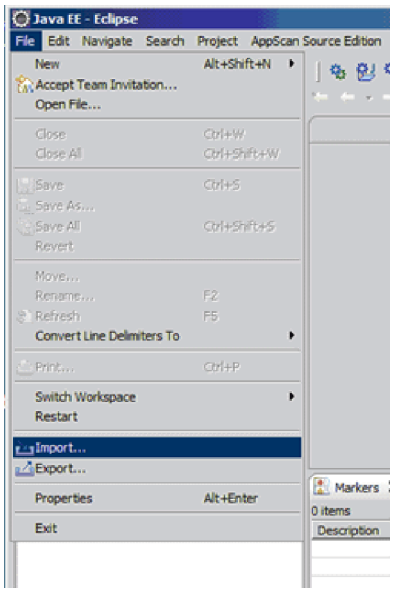
将示例项目导入 Eclipse 或 Rational Application Developer for WebSphere Software (RAD)

过程

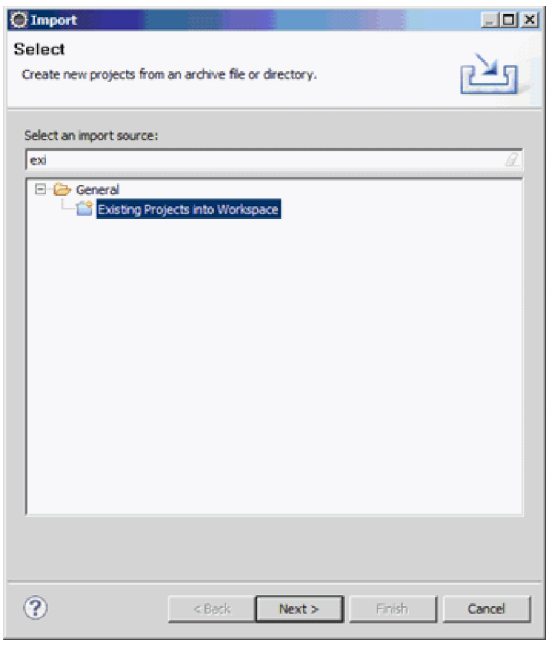
1. 启动 Eclipse。
2. 为您的处理程序项目创建新的工作空间。



3. 从主工作台菜单中选择**文件** > **导入**。



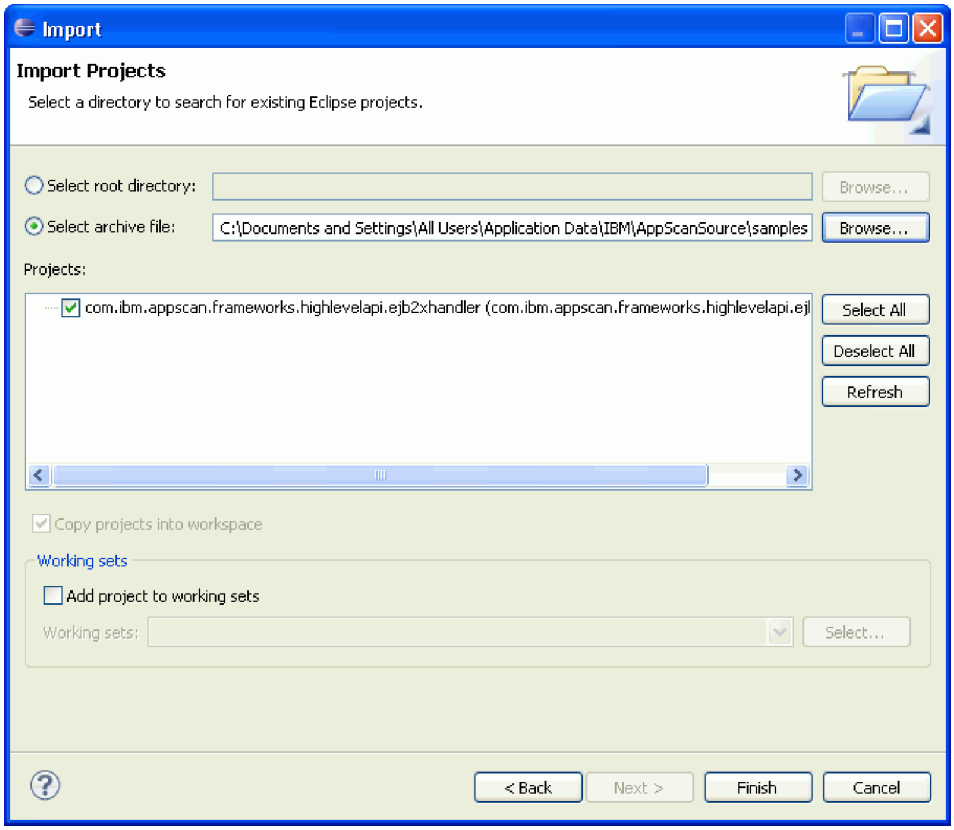
4. 在"导入"向导的**选择**页面中，选择**现有项目到工作空间**，然后单击**下一步**。



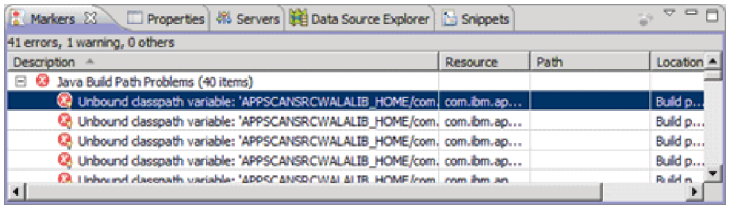
5. 选中**选择归档文件**单选按钮，然后单击**浏览**。在**选择包含了所要导入项目的归档**对话框中，找到 `<data_dir>\samples\F4FEjbExample.zip`（其中 `<data_dir>` 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章，『安装和用户数据文件位置』中所述），然后单击**打开**。

   此 zip 文件是包含 `Ejb2xHandler` 框架处理程序示例的配置和源代码的 Eclipse 归档。

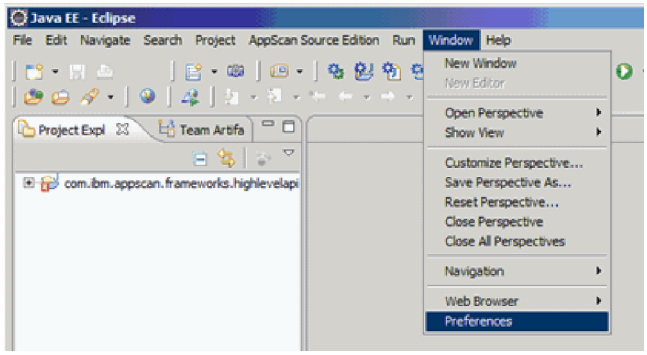
   单击**完成**导入此归档。



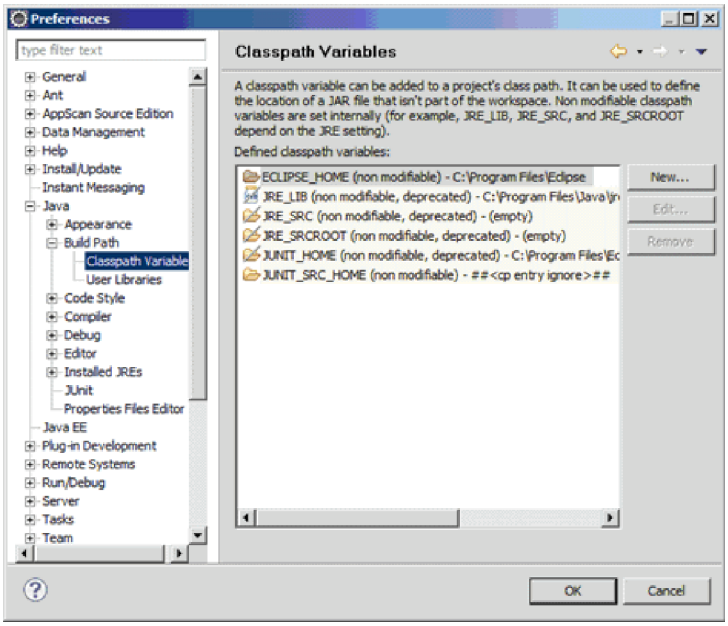
6. 导入归档后，您会发现有几个构建错误。此示例的类路径变量指向保存其库的目录。



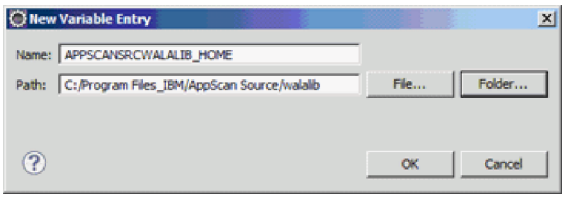
7. 下一步，您将定义变量。从主工作台菜单选择**窗口** > **首选项**。



8. 在"首选项"对话框中，选择 **Java** > **构建路径** > **类路径变量**打开"类路径变量"首选项页面。单击**新建**。



9. 创建名为 APPSCANSRCWALALIB_HOME 的新变量并将 **Path** 设置为 <install_dir>\walalib（其中 <install_dir> 是 AppScan Source 安装位置）。



单击所有对话框上的**确认**后，样本构建完成，没有错误。

创建实施 F4FHandler 的类

要创建新的框架处理程序，必须首先创建可以实施 F4FHandler 的类。有两个方法必须实施，以支持 Framework for Frameworks 的功能。

覆盖 isApplicable

用途： AppScan Source 调用 isApplicable 来确定是否应运行您的处理程序。如果您返回 True，它将调用 handleApp 来运行您的处理程序。如果您返回 False，将不再调用任何对象。

注： isApplicable 会在方法开头检查确保应用程序是 Java 应用程序，然后再继续。

示例观察： 在 Ejb2xHandler 和 isApplicable 中，首先检查以查看语言是否适合（因为 EJB 仅在 Java 应用程序中出现）。如果应用程序是基于 Java 的，isApplicable 会搜索 ejb-jar.xml 的任何实例，这对 EJB 2 应用程序的配置文件来说是必需的。如果发现配置文件，会将其读入处理程序并返回 True，以使 AppScan Source 知道它应该调用 handleApp 来处理这些配置文件中包含的信息。

覆盖 handleApp

用途：AppScan Source 调用 handleApp 以使您可以确定和设置有关当前所扫描的应用程序使用的框架的信息。参数 F4FApp 和 F4FAction 用于获取有关应用程序的信息，和设置有关如何处理存在的框架和处理程序所处理的框架的具体信息。

为处理程序创建清单文件

创建文件 Manifest.txt 来包含导出 JAR 清单时需要的信息。具体而言，在装入处理程序 JAR 时，AppScan Source 会查找 Framework-Handler 条目以确定哪个类中包含 F4FHandler 功能。

将以下两行加入 Manifest.txt 文件：

```
Manifest-Version: 1.0
Framework-Handler: package.HandlerClassName
```

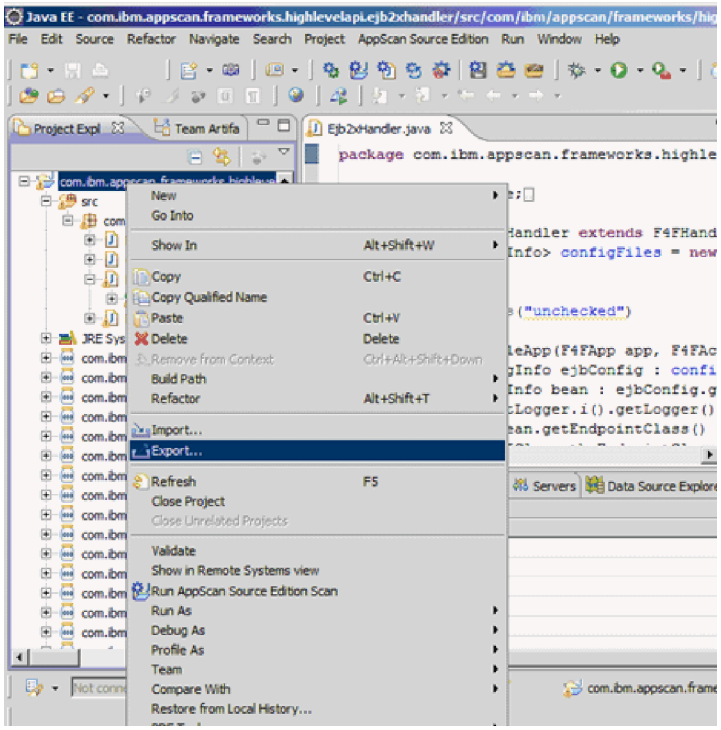
其中 package.HandlerClassName 是实施 F4FHandler 接口的类的完整软件包和类名称。

示例观察：Manifest.txt 中包含将在创建 Ejb2xHandler 的 JAR 文件清单时放入其中的信息。此信息只有两行，一行列出版本，另一行指定框架处理类，Framework-Handler: com.ibm.appscan.frameworks.highlevelapi.ejb2xhandler.Ejb2xHandler。

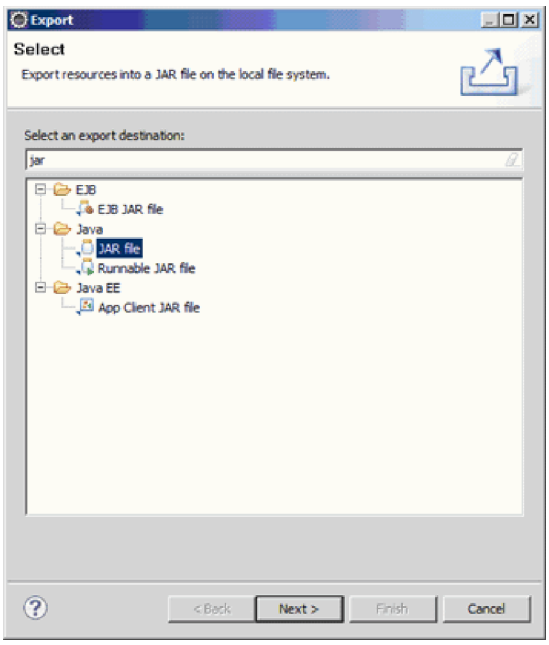
为处理程序创建 JAR

过程

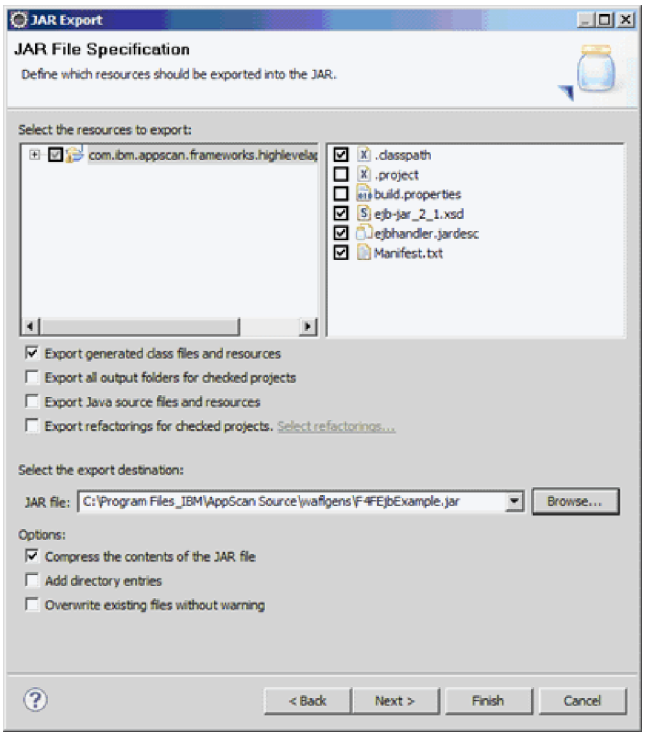
1. 在"Eclipse 项目浏览器"视图中右键单击项目，然后选择**导出**。



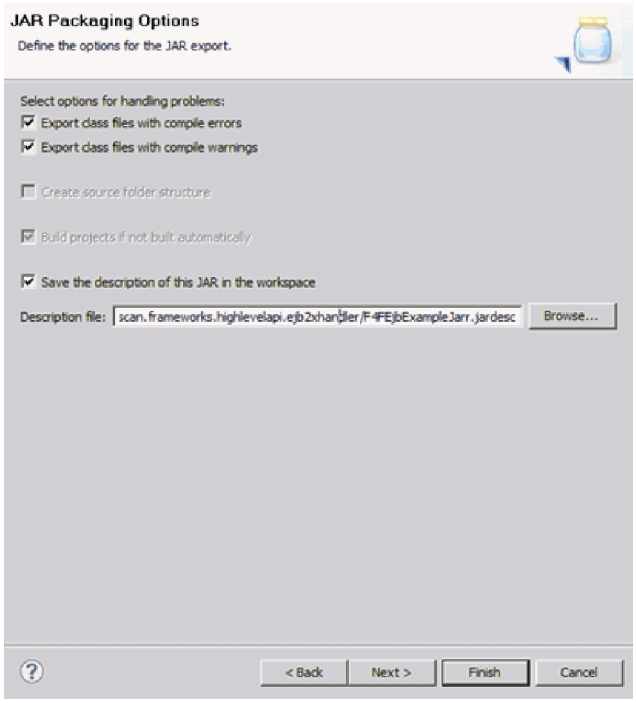
2. 在"导出"向导中，选择 **JAR 文件**，然后单击**下一步**。



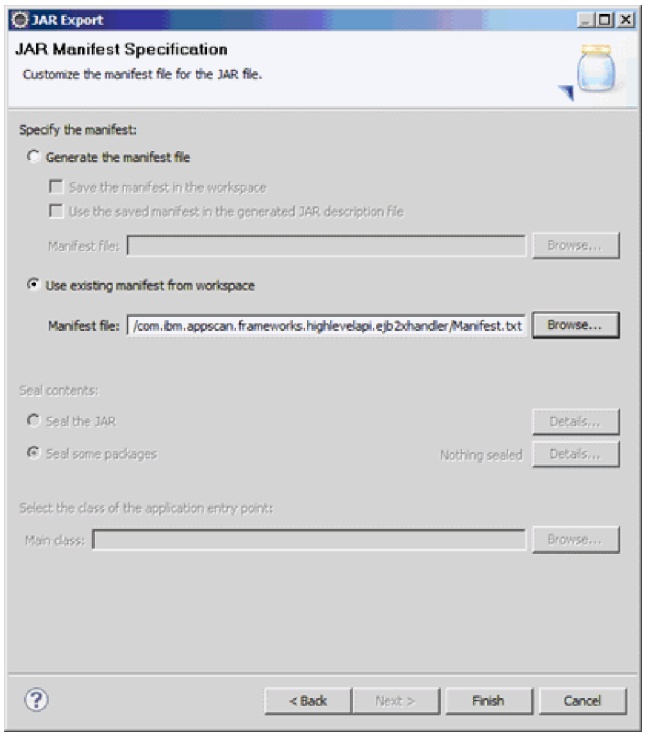
3. 在"JAR 导出"的**选择导出目的地**部分中，为 JAR 文件选择合适的位置，并提供名称。单击**下一步**。



4. 再次单击**下一步**。此外，还可以选择为描述文件设置位置和名称。此操作会将一个 `.jardesc` 文件保存到将包含此次导出的所有 JAR 导出设置的项目中。如此一来，您就可以重复导出而不必再次指定这些设置了。



5. 选中**使用工作空间中的现有清单**单选按钮，单击**浏览**。选择早些时候创建的 `Manifest.txt` 文件。



6. 单击**完成**

将 JAR 导出到 waflgens

如果没有在创建处理程序 JAR 时将其直接导出到 waflgens 目录，那么现在将其复制到此目录。完整路径为 <install_dir>\waflgens（其中 <install_dir> 是 AppScan Source 安装位置）。

处理程序执行的常见操作

常见 Web Service 入口点

需多框架都自己提供应用程序入口点。一个常见的示例是显示在配置文件或代码注释中指定的 Web Service。在应用程序配置文件或直接在字节码中搜索指定入口点后，可使用方法 F4FAction.addTaintedCallback 来在适当的方法中创建已感染数据入口点。

观察示例：在 EJB 2 中，web service 入口点通过在应用程序的配置文件（ejb-jar.xml）中定义 *endpoints* 来声明。然后，handleApp 会依次通过 ejb-jar.xml 中定义的 bean，只要定义了端点类，它会获取方法名称列表。然后，它会使用 addTaintedCallback 方法将其实施声明为 Web Service 入口点。

替换方法

现代的框架通常会利用虚拟函数和抽象来进行更为松散的业务组件耦合。虽然对于开发流程，这可以算作一项改进，但是，当虚拟函数和其实施之间的连接是在配置文件中处理的，或通过代码中的注释处理的时，也会带来静态分析方面的困难。F4FAction.replaceCalls 允许处理程序指定这些连接。

示例观察：在 EJB 2 中，每个 bean 都有一个声明其他 bean 如何与其交互的接口集合（本地和远程）。这意味着，只要调用 bean 的接口 class.method，它将被具有实际 ImplementationClass.method 的框架替代。

从第 62 行开始，我们的示例处理程序将依次通过每个 bean，并获取其远程和本地接口，然后使用其实际实施替代这些接口。

记录

处理程序可使用 com.ibm.wala.andromeda.util.logging.TaintLogger 类在执行期间记录参考消息，并使错误消息显示在 AppScan Source 用户界面中。TaintLogger 类使用 log4j 库。要记录消息，首先通过调用 TaintLogger.i().getLogger() 来获取 Logger 对象。然后，调用 Logger 上的记录方法（例如，Logger.warn）以记录您需要的消息。日志消息将显示在 <data_dir>\logs\scanner_exceptions.log（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述） 中。如果使用 Logger.error 或 Logger.fatal 来记录消息，那么错误消息也将显示在 AppScan Source 用户界面的"控制台"视图中。

Framework for Frameworks API 类和方法

This section contains the Framework for Frameworks API classes and methods, abbreviated for Adobe PDF. To access the complete set of API documentation, launch the AppScan Source for Analysis online help and navigate to Reference > Framework for Frameworks handling APIs > Framework for Frameworks API classes and methods.

The complete set of API documentation can also be found at http://www.ibm.com/support/knowledgecenter/SSS9LM/welcome.

F4FActions

```
java.lang.Object
  extended by com.ibm.appscan.frameworks.highlevelapi.F4FActions

public class F4FActions
extends java.lang.Object
```

Class for specifying how the application's framework constructs should be modeled. An `F4FHandler` mutates the `F4FAction` object passed to `F4FHandler.handleApp(F4FApp, F4FActions)` as it analyzes the application.

Constructor Detail

`F4FActions`

```
public F4FActions()
```

Create an empty `F4FActions` object. Should not be needed for implementing a new framework handler, as the relevant `F4FActions` object will be passed to `F4FHandler.handleApp(F4FApp, F4FActions)`.

addTaintedCallback

```
public void addTaintedCallback(IMethod method,
                               int numParams)
```

Same as 『addTaintedCallback』 `(String, int)`, but takes an `IMethod` directly rather than a VDB signature

addTaintedCallback

```
public void addTaintedCallback(java.lang.String vdbMethodSig,
                               int numParams)
```

Make a method a tainted callback, with all parameters tainted.

注： For .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`.

Parameters:

- `vdbMethodSig` - the signature of the `callback` method
- `numParams` - the number of parameters for the `callback` method, including the `this` parameter

replaceCalls

```
public void replaceCalls(java.lang.String oldVDBSig,
                         java.lang.String newVDBSig)
```

Replace all calls to one method with calls to another method. We require that the descriptors for the old and new method (i.e., the number of arguments, argument type, and return type) are identical.

注： replacement will only occur when `oldVDBSig` is the _declared_ target at a call site. So, if `oldVDBSig` is `Integer.toString()`, and we see a call to `Object.toString()`, we will _not_ perform a replacement at that call site, even though it might invoke `Integer.toString()`.

注： for .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

Parameters:

- `oldVDBSig` - signature of method whose calls should be replaced
- `newVDBSig` - signature of method to replace calls with

replaceCallsWithSyntheticExpr

```
public void replaceCallsWithSyntheticExpr(java.lang.String vdbSig,
    com.ibm.appscan.frameworks.specinfo.SyntheticExpr expr)
```

Replace all calls to a method with an arbitrary WAFL `SyntheticExpr`. For example, one could replace calls with an assignment via an `AssignmentExpr`.

注： replacement will only occur when `oldVDBSig` is the _declared_ target at a call site. So, if `oldVDBSig` is `Integer.toString()`, and we see a call to `Object.toString()`, we will _not_ perform a replacement at that call site, even though it might invoke `Integer.toString()`.

注： for .NET apps, we need fully-qualified VDB signatures. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`

Parameters:

- vdbSig - signature of method whose calls should be replaced
- expr - synthetic expression to replace calls with

replaceCallsWithParamPattern

```
public void replaceCallsWithParamPattern(java.lang.String oldVDBSig,
                                         java.util.Map<java.lang.String,
                                         java.util.Map<java.lang.Integer,
                                         java.util.regex.Pattern>>
                                         newSig2Pattern)
```

Replace calls to one method with calls to another method only if the parameters of String type are constants meeting specified patterns. We require that the descriptors for the old and new method (i.e., the number of arguments, argument type, and return type) are identical.

注：replacement will only occur when oldVDBSig is the _declared_ target at a call site. So, if oldVDBSig is Integer.toString(), and we see a call to Object.toString(), we will _not_ perform a replacement at that call site, even though it might invoke Integer.toString().

注：for .NET apps, we need fully-qualified VDB signatures. So, instead of int as a parameter type, we need System.Int32, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see DotNetVDBUtil.systemName2VDBShortName

Parameters:

- oldVDBSig - signature of method whose calls should be replaced
- newSig2Pattern - maps VDB signature of each possible replacement method m to a map M from integer parameter positions to Patterns. If the string constant parameters in the appropriate positions match the patterns in M at some call site, a replacement to m will be performed.

addFrameworkInfo

```
public void addFrameworkInfo
(com.ibm.appscan.frameworks.specinfo.IFrameworkInfo info)
```

Add arbitrary additional framework info. This method should only be needed for rare cases where the other APIs provided are insufficient.

addTaintedCallback

```
public void addTaintedCallback(java.lang.String vdbMethodSig,
                               java.util.Collection<TaintedParam>
                               taintedParams)
```

Make some method a tainted callback, with only certain parameter access paths being treated as tainted.

注：for .NET apps, we need fully-qualified VDB signatures. So, instead of int as a parameter type, we need System.Int32, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see DotNetVDBUtil.systemName2VDBShortName

Parameters:

- `vdbMethodSig` - the signature of the callback method, in VDB format
- `taintedParams` - information on which parameter access paths should be tainted

addHighLevelSyntheticMethod

```
public void addHighLevelSyntheticMethod(HighLevelSyntheticMethod m)
```

equivalent to `addHighLevelSyntheticMethod(m, true)`

addHighLevelSyntheticMethod

```
public void addHighLevelSyntheticMethod(HighLevelSyntheticMethod m,
                                        boolean isEntrypoint)
```

Add a high-level synthetic method. A corresponding WAFL synthetic method (possibly an entrypoint) will be generated.

Parameters:

- `m` - the method
- `isEntrypoint` - should the method be marked as an entrypoint in WAFL?

createGlobal

```
public Global createGlobal(java.lang.String name,
                           java.lang.String declaredVDBType,
                           boolean isEntrypointScoped)
```

Create a new global that can be accessed from `HighLevelSyntheticMethods`.

Parameters:

- `name` - name for the global
- `declaredVDBType` - the declared type of the global (e.g., `java.lang.String`).

  注：for .NET apps, we need a fully-qualified VDB type. So, instead of `int` as a parameter type, we need `System.Int32`, etc. To see the full mapping from fully-qualified names to the names usually used in VDB, see `DotNetVDBUtil.systemName2VDBShortName`
- `isEntrypointScoped` - if true, the global is scoped to a single entrypoint (i.e., it is request-scoped). Otherwise, the global is scoped across entrypoints (i.e., it is "session" or "application" scoped)

Returns:

- a Global object, which can be read/written inside a `HighLevelSyntheticMethod`

createGlobal

```
public Global createGlobal(java.lang.String name,
                           IClass declaredClass,
                           boolean isEntrypointScoped)
```

Just like 『createGlobal』(`String, String, boolean`), but takes an `IClass` for the declared type instead of a type name

getGlobals

```
public java.util.Collection<Global> getGlobals()
```

For internal usage.

getAdditionalFrameworkInfo

```
public java.util.Collection
<com.ibm.appscan.frameworks.specinfo.IFrameworkInfo>
getAdditionalFrameworkInfo()
```

For internal usage.

getCallReplacement2SigsInfo

```
public java.util.Map
<java.lang.String,java.util.Map
<java.lang.String,java.util.Map
<java.lang.Integer,java.util.regex.Pattern>>>
getCallReplacement2SigsInfo()
```

For internal usage.

getCallReplacement2ExprInfo

```
public java.util.Map
<java.lang.String,com.ibm.appscan.frameworks.specinfo.SyntheticExpr>
getCallReplacement2ExprInfo()
```

For internal usage.

getCallback2TaintedParams

```
public java.util.Map
<java.lang.String,java.util.Collection<TaintedParam>>
getCallback2TaintedParams()
```

For internal usage.

getHighLevelSyntheticMethods

```
public java.util.List
<com.ibm.wala.util.collections.Pair
<HighLevelSyntheticMethod,java.lang.Boolean>>
getHighLevelSyntheticMethods()
```

For internal usage.

toString

```
public java.lang.String toString()
```

Overrides:

- toString in class java.lang.Object

F4FApp

```
java.lang.Object
  extended by com.ibm.appscan.frameworks.highlevelapi.F4FApp

public class F4FApp
extends java.lang.Object
```

Representation of an application, with methods to query various properties of classes, methods, etc. Implemented mostly by delegating to methods from the T.J. Watson™ Libraries for Analysis (WALA); the goal is to consolidate the most useful WALA methods in a single type. See the WALA home page (http://wala.sourceforge.net) for full details on WALA APIs.

Constructor Detail

```
public F4FApp(IClassHierarchy cha)
```

Should not be needed to implement a new handler. The relevant `F4FApp` object will be passed as a parameter to `F4FHandler.handleApp(F4FApp, F4FActions)`

getAppClass

```
@Deprecated
public IClass getAppClass(java.lang.String vdbClassName)
```

Deprecated. Use `getIClass(String)` instead; this method simply delegates to that one.

getIClass

```
public IClass getIClass(java.lang.String vdbClassName)
```

get the `IClass` for some class in the application, including library jars/DLLs. If no class with the provided name is found, return `null`

Parameters:

- `vdbClassName` - class name in VDB format, e.g., `java.lang.String`

getClassAnnotations

```
public java.util.Collection<Annotation>
getClassAnnotations(IClass klass)
```

Get the annotations/attributes for a class. For .NET, the result will include inherited attributes.

Parameters:

- `klass` - the class whose annotations are desired

getMethodAnnotations

```
public java.util.Collection<Annotation>
getMethodAnnotations(IMethod method)
```

Get the annotations / attributes for a method. For .NET, these will include inherited attributes.

Parameters:

- `method` - the method whose annotations are desired

getFieldAnnotations

```
public java.util.Collection<Annotation>
getFieldAnnotations(IField field)
```

Get the annotations / attributes for a field.

Parameters:

- `field` - the field whose annotations are desired

getMethodParametersAnnotations

```
public java.util.Collection<Annotation>[]
getMethodParametersAnnotations(IMethod method)
```

Get annotations on parameters as an array of Collections, where each array element gives the annotations on the corresponding parameter. Note that the `this` parameter for an instance method cannot have annotations.

Parameters:

- `method` - the method whose parameter annotations are desired

getAllApplicationClasses

```
public java.util.Collection<IClass>
getAllApplicationClasses()
```

Get all the classes in the application (i.e., excluding those in library jars).

getClassHierarchy

```
public IClassHierarchy getClassHierarchy()
```

Get the WALA class hierarchy for the application. Most handlers should be able to work via the other methods in this class and shouldn't need to operate directly on the class hierarchy. But, access is provided for advanced use.

getMethodsDeclaredInClass

```
public java.util.Collection<IMethod>
getMethodsDeclaredInClass(IClass klass)
```

Get all the static and instance methods declared in `klass`

getClassMethods

```
public java.util.Collection<IMethod>
getClassMethods(java.lang.String className,
                java.lang.String methodName)
```

Get all the methods in an class with a particular name. If the class cannot be found, returns an empty Collection.

Parameters:

- `className` - the class name in VDB (i.e., source-level) format, e.g., `java.lang.String`
- `methodName` -

getClassMethods

```
public java.util.Collection<IMethod>
getClassMethods(IClass appClass,
                java.lang.String methodName)
```

Get all the methods in an class with a particular name.

Parameters:

- `appClass` - the class
- `methodName` -

getStringConstantsReturnedByMethod

```
public java.util.Collection<java.lang.String>
getStringConstantsReturnedByMethod(IMethod method)
```

Get the possible String constants returned by the method. E.g., if the method has a statement return "result";, then "result" will be in the returned Collection. Throws an `IllegalArgumentException` if the return type of method is not `String`.

F4FHandler

```
java.lang.Object
  com.ibm.appscan.frameworks.highlevelapi.F4FHandler

public abstract class F4FHandler
extends java.lang.Object
```

The abstract class that any new framework handler should extend.

注： Any sub-class of `F4FHandler` must have a no-argument constructor, for instantiation using reflection

Constructor Detail

`F4FHandler`

```
public F4FHandler()
```

handleApp

```
public abstract void handleApp(F4FApp app,
                               F4FActions actions)
```

Define what actions should be represented in the generated WAFL specification to handle the given application.

Parameters:

- `app` - the application to be analyzed
- `actions` - the actions to be taken; implementations of `handleApp(F4FApp, F4FActions)` should mutate this parameter and store the desired actions

isApplicable

```
public abstract boolean isApplicable()
```

Is the framework handler applicable for the target application? Implementations should check the language of the app, whether relevant configuration files are present, etc.

setFrameworksInput

```
public void setFrameworksInput(FrameworksInput input)
```

Should not be invoked by a framework handler.

getFrameworksInput

```
protected FrameworksInput getFrameworksInput()
```

Get the parsed representation of the input parameters to the frameworks code.

TaintedParam

```
java.lang.Object
  extended by com.ibm.appscan.frameworks.highlevelapi.TaintedParam

public class TaintedParam
extends java.lang.Object
```

A type used to represented how some method parameter may reference tainted data.

Constructor Detail

```
public TaintedParam(int paramPos,
                    java.lang.String accessPath)
```

Creates a new `TaintedParam` object for a particular parameter position and access path.

Parameters:

- `paramPos` - the position of the tainted parameter, numbered starting from 0 (where the `this` parameter of an instance method is parameter 0)
- `accessPath` - access path of fields from the parameter that should be tainted, e.g., "f.g". Use "" to indicate the parameter itself should be tainted.

getParamPos

```
public int getParamPos()
```

getAccessPath

```
public java.lang.String getAccessPath()
```

hashCode

```
public int hashCode()
```

Overrides:

- `hashCode` in class `java.lang.Object`

equals

```
public boolean equals(java.lang.Object obj)
```

Overrides:

- `equals` in class `java.lang.Object`

toString

```
public java.lang.String toString()
```

Overrides:

- toString in class java.lang.Object

高级别合成方法

要对框架中的高级数据流进行建模，合成方法很有用。例如，很多标准框架（如 Struts 和 Spring）鼓励对应用程序使用模型-视图-控制器 (MVC)。通过 MVC 结构，通常按以下方式来处理客户机表单提交：

1. 根据 URL，确定用于保存提交的表单数据的应用程序模型类 M 以及包含业务逻辑的控制器类 C。
2. 创建 M 模型对象，并根据 HTTP 请求中的（不可信）表单数据来设置其属性。这些属性通常通过 *setter* JavaBeans 来设置（例如，setName() 或 setAddress ()）。
3. 对 M 对象中的数据执行某些验证。
4. 创建 C 控制器对象，并将 M 对象传递到执行业务逻辑的方法 C.execute()。通常，execute() 将返回用于呈现结果的视图的名称。
5. 根据视图名称，确定要显示的相应视图文件（例如，JavaServer Pages）。通常，M 对象中的数据将通过请求或会话对象的属性来传递到视图。

可以通过 Framework for Frameworks 合成方法来对所有以上功能进行建模，因而揭示行为以供 AppScan Source 分析。 Framework for Frameworks API 提供高级别合成方法以简化合成方法的生成。

注：类名以 Appscan.Synthetic、Appscan.Synthetic.Validator 和 AppScan.Synthetic.Replacement 开始的跟踪节点对应于由 AppScan Source 合成的方法。

- AppScan.Synthetic 方法用于在使用框架的应用程序代码中将跟踪聚合到一起。
- AppScan.Synthetic.Validator 方法可对框架运行时执行的底层验证进行建模。如果需要，您可以选择一个 validator 方法并将其标记为**验证器**。
- AppScan.Synthetic.Replacement 方法指示应用程序代码中的某个方法已替换为 AppScan Source，以捕获框架的不相关组件（如控制器和视图）之间的数据流。

VDB 格式

Framework for Frameworks API 中的方法需要某些 String 以通过 VDB 格式来表示类型名称或方法特征符。VDB 类型名称就是源级别标准名称（例如，java.lang.String），或者，如果是内部类，那么使用美元符号 ($) 来表示（例如，javax.swing.text.DefaultEditorKit$DefaultKeyTypedAction）。对于 .NET，在使用 Framework for Frameworks API 时，应避免使用短名称（如 int 和 string）。请改为使用标准名称，如 System.Int32 和 System.String。

VDB 方法特征符包含以下两个部分：

1. 方法名称，包括封闭类的标准名称（例如，java.lang.String.substring）。

2. 方法的描述符，可提供参数类型和返回类型。参数类型括在括号中，并且用分号 (;) 来分隔。右括号后跟冒号 (:)，然后是返回类型。

VDB 方法特征符示例：

```
java.lang.String.substring(int;int):java.lang.String
```

内部类中的示例 VDB 方法特征符为：

```
javax.swing.text.DefaultEditorKit$DefaultKeyTypedAction.
    actionPerformed(ActionEvent):void
```

对于大部分需要 VDB 格式的 String 的 API 方法，经常会有一种改为使用 IClass 或 IMethod 对象的等效方法。例如，方法 F4FActions.addTaintedCallback(String,int) 要求其第一个参数为 VDB 格式的方法特征符 - 而 F4FActions.addTaintedCallback (IMethod,int) 则使用 IMethod 对象来标识此方法。因此，使用采用 IClass 和 IMethod 对象的 API 方法可能会更方便，原因如下：

1. 用户不需要担心这些方法的 VDB 格式化，因为这在内部处理。
2. 在获取 IClass 或 IMethod 对象时（例如，通过 F4FApp.getClassMethods()），请确保对应的类或方法在应用程序代码中实际存在。

使用高级别合成方法

此主题说明如何使用高级别合成方法的某些重要特性。有关完整详细信息，请参阅 Framework for Frameworks API 类和方法 Javadoc 文档。

- 『创建高级别合成方法』
- 『创建局部变量』
- 第 124 页的『添加调用』
- 第 124 页的『添加返回值』
- 第 125 页的『全局变量』

创建高级别合成方法

要创建高级别合成方法并将其添加到 F4FActions 对象 actions，请使用如下代码：

```
HighLevelSyntheticMethod m = HighLevelSyntheticMethod.make();
actions.addHighLevelSyntheticMethod(m);
```

您可以通过将 VDB 方法特征符传递到 HighLevelSyntheticMethod.make() 来指定合成方法的名称、参数类型和返回类型。

创建局部变量

可以通过以下方式来创建高级别合成方法 m 的新局部变量：

```
Local l = m.newLocal("java.lang.String");
```

构造方法不需要在合成方法中调用。考虑到以上代码，在向合成方法中添加更多语句时，我们可以假设 l 是指非空 String 对象。

添加调用

高级别合成方法中的大部分语句将为方法调用（通过 addCall() 方法添加）。addCall() 的参数表示要调用的方法、调用的文件位置信息以及在调用时要传递的参数。以下示例用于添加对 setter 方法 sample.BeanType.setName() 的调用（假定存在 F4FApp 对象 app）：

```
Collection<IMethod> setterMethods =
   app.getClassMethods("sample.BeanType", "setName");
// assume there is exactly one "setName" method in BeanType
IMethod setter = setterMethods.iterator().next();
HighLevelSyntheticMethod m = HighLevelSyntheticMethod.make();
Local l = m.newLocal("sample.BeanType");
m.addCall(setter, null, l, Taint.taint());
```

处理 MVC 体系结构中的客户机表单提交的步骤 2 - 5（如第 122 页的『高级别合成方法』中所概述）可以在某种程度上进行建模，方法是按照上述方式向合成方法中添加相应调用。

某个调用的参数由 Param 类型的对象来表示，这可以是以下任何一种：

- Local 对象，表示局部变量
- Taint 对象，表示不可信或受感染数据。
- EnclosingFormal 对象，表示高级别合成方法的形参。例如，如果您具有一个包含特征符 synthMethod(int):void 的合成方法，那么 EnclosingFormal.FIRST 是指方法的 int 参数。
- 代表全局变量的 Global 对象（已在第 125 页的『全局变量』中讨论）。

注：对于非静态实例方法，要以 this 传递的值必须提供给 addCall()。在以上示例中，l 中的值将以这种方式传递到 setName()。

添加返回值

合成方法可通过使用 setReturnedValue() 方法返回值。返回值对于生成标记方法来对复杂框架验证建模比较有用。例如，如果要在将已感染 HTTP 请求值传递到模型对象的 setter 方法之前使用对该值执行复杂验证的框架，那么可通过使用以下代码在 AppScan Source 发现的跟踪上暴露验证：

```
String validatorSig =
   "Synth.validatorModel(java.lang.String):
   java.lang.String";
HighLevelSyntheticMethod validator =
   HighLevelSyntheticMethod.make(validatorSig);
// just return the parameter passed in
validator.setReturnedValue(EnclosingFormal.FIRST);
HighLevelSyntheticMethod m = ...;
Local validated = m.addCall(validatorSig, null, Taint.taint());
// now, validated can be passed to a setter
```

合成方法 Synth.validatorModel() 仅返回作为参数传递到其自身的 String。然后，在另一个合成方法中添加对 Synth.validatorModel() 的调用，将已感染值作为自变量传递。该调用的结果存储在 validated 中，可在后续调用中将该结果传递到 setter 方法（如『添加调用』中的示例所示）。涉及该查找和关闭数据的跟踪将包括对 Synth.validatorModel() 的调用，如果认为验证已足够，那么 AppScan Source 用户可选择过滤跟踪。

全局变量

全局变量是一项用于揭示应用程序不同部分之间的流的高级功能。例如，全局变量可以用于针对从控制器到视图的数据流进行建模（通过请求或会话属性）。用于创建和访问全局变量的关键操作为：

- 要创建一个表示全局变量的 Global 对象，请使用 F4FActions.createGlobal()。
- 要写入到全局变量，请使用 HighLevelSyntheticMethod.addGlobalWrite()。
- 要从全局变量读取，请在对 HighLevelSyntheticMethod.addCall() 的调用中将 Global 对象作为 Param 参数来传递 - 或者，通过 HighLevelSyntheticMethod.setReturnedValue() 使其从合成方法中返回。

示例：控制器类将用户的名字写入到请求属性 firstName，然后视图读取此请求属性并将值呈现给响应。在高级别上，用户可以按如下所示将此流进行建模：

```
Global firstNameGlobal = actions.createGlobal
   ("firstName", "java.lang.String", true);
HighLevelSyntheticMethod controller = ...;
Local firstNameLocal = controller.newLocal("java.lang.String");
controller.addGlobalWrite(firstNameGlobal, firstNameLocal, null);
HighLevelSyntheticMethod view = ...;
view.addCall("Response.write(java.lang.String):void",
   null, firstNameGlobal);
```

通过将某个写入添加到控制器合成方法中的 firstNameGlobal，然后将 firstNameGlobal 传递到 view 合成方法中的 Response.write()，将揭示从控制器到视图的数据流。

注：该示例进行了大幅度的简化。其中一点是，完整版本将需要揭示数据到 firstNameLocal 的正确流。

示例：合成方法创建

本示例展示了合成方法的创建。在此合成方法中，我们首先感染一个类字段，然后向用户的的代码中添加一个方法，并最终将此方法的结果传递到接收器。

- 『应用场景』
- 『合成方法中的建模对象』
- 第 126 页的『步骤 1：创建空的合成方法』
- 第 126 页的『步骤 2：对类字段的感染进行建模』
- 第 127 页的『步骤 3：对 createUser() 方法的调用进行建模』
- 第 127 页的『步骤 4：对客户机的数据返回建模』

应用场景

在本示例中，createUser 是在 JAX-RS 框架中实施的 REST API。用可使用诸如 http://host:port/users/createUser 的 URL 调用该 API。框架运行时将该调用指派给 UserRestService.createUser()，因为路径匹配。此外，框架运行时在调用 createUser() 之前从客户机数据初始化 urlInfo 类变量。最后，运行时会将 createUser() 返回的数据发送回客户机。

合成方法中的建模对象

为使用合成方法来捕获 『应用场景』，我们具备以下元素：

- 感染 UserRestService 的 urlInfo 类变量。

- 对 UserRestService.createUser() 的调用。
- 客户机的数据返回。

以下是 createUser() 方法的代码：

```
import java.io.BufferedWriter;

@Path("users")
public class UserRestService {
  @Context
  UriInfo urlInfo;

  @Path("/createUser")
  public User createUser(){
    MultivaluedMap<String, String> queryParams =
      urlInfo.getQueryParameters();
    String name = queryParams.getFirst("name");
    User user = new User(name);
    return user;
  }
}
```

步骤 1：创建空的合成方法

```
22    public class JAXRSHandler extends F4FHandler{
23    @Override
        public void handleApp(F4FApp app, F4FActions actions) {
24    HighLevelSyntheticMethod synthMethod = HighLevelSyntheticMethod.make();
```

- 合成方法对象在第 24 行初始化。

步骤 2：对类字段的感染进行建模

以下代码段演示了如何能够感染特定类字段。在此示例中，我们希望感染使用 @Context 注释进行了注释的类字段。

```
27    // create a local variable of the appropriate type
28    Local userRestServiceClazzInstance =
        synthMethod.newLocal("com.ibm.appscan.UserRestService");
29
30    // get all the class fields
31    for(IField field: app.getIClass
        ("com.ibm.appscan.UserRestService").getDeclaredInstanceFields()){
32
33      //get all the annotations associated with the field
34      Collection<Annotation> fieldAnnotations = app.getFieldAnnotations(field);
35
36      //for each annotation of the field check if it is an @Context annotation
37      for (Annotation annotation : fieldAnnotations) {
38
39        if (annotation.getType().getName().toString().equals
            ("Ljavax/ws/rs/core/Context")) {
40
41          // call the F4F API to assign taint to the field
43          synthMethod.addInstanceVariableWrite(
                 userRestServiceClazzInstance /*Variable representing
                 class instance*/,
44               field /*field to taint  */,
                 Taint.taint() /* taint */,
                 null);
45        }
46      }
47    }
```

Framework for Frameworks API addInstanceVariableWrite 采用四个参数：

1. 第一个参数为我们想要感染其字段的类引用。在例子中，本地变量 userRestServiceClazzInstance 引用了该参数。
2. 第二个参数是我们想要感染的类字段。
3. 第三个参数为将分配给类变量字段的新值。在例子中，我们想要感染该变量，这样就会传递 Taint.taint()。
4. 最后一个参数是 FilePositionInfo，在此示例中为 null。

步骤 3：对 createUser() 方法的调用进行建模

在该步骤中，我们在我们自己的合成方法中模拟调用。

```
50    // call createUser() and store the returned value to a local variable
51    Local returnObj = synthMethod.addCall(
52      "com.ibm.appscan.UserRestService.createUser():com.ibm.appscan.User"
          /* signature of the method to be called */,
          null, /* FilePositionInfo */
53        userRestServiceClazzInstance /* Object on which the method
            will be called */);
```

addCall 高级别 API 采用三个参数：

- 第一个参数是我们要调用的方法的签名。在本例中，我们想要调用 User.createUser()，因此将使用其签名。
- 第二个参数为 FilePositionInfo。该参数将作为 null 传递，因为本例中不需要该参数。
- 最后一个参数代表调用 createUser() 方法所需的参数的列表。

由于 createUser() 不采用任何参数，因此传递给此调用的唯一参数是 this 对象，这是一个 User 类型的局部变量 (userRestServiceClazzInstance)。方法 createUser() 将返回新近创建的 User，其存储在名为 returnedObj 的新局部变量中（如第 51 行所示）。

步骤 4：对客户机的数据返回建模

在最后这一个步骤中，我们将创建一个接收器以对客户机的数据返回进行建模。

```
58    // create a PrintWriter object
59    Local printWriterObj = synthMethod.newLocal("java.io.PrintWriter");
60
61    // we want to call the print API on the PrintWriter object.
      // The print API takes a single argument of Object type returns void
62
63    // we create a param list of size 2. The first item in this list is always the
      //  'this' object and rest are actual method arguments.
64    Param[] printWriterObjParam =  new Param[2];
65    printWriterObjParam[0] = printWriterObj;  // The "this" object
      // the value returned by the call to the createUser method
66    printWriterObjParam[1] = returnObj;
67
68    // Now add a call to the print method
69    synthMethod.addCall("java.io.PrintWriter.print(java.lang.Object):void",
        null, printWriterObjParam);
70  }
```

- 在第 59 行，在合成方法中创建 PrintWriter 类的 Local 对象。
- 从第 64 行到 66 行，准备在 PrintWriter 类实例 (printWriterObj) 上调用 print 方法所需的参数列表。
  - 第一个参数为对象引用 (printWriterObj) 本身。

- 第二个参数为 print 方法的参数。我们想要打印返回值，该值存储在局部变量 returnObj 中。

- 在第 69 行，我们添加了对 PrintWriter.print 方法的调用，以对客户机的数据返回进行建模。

将新 Web service Framework for Frameworks 处理程序与现有 Web service 描述语言定制扫描程序集成

关于此任务

使用 AppScan Source 扫描 JAX-RS 或 JAX-WS 应用程序时，应确保可以捕获所有 Web service 入口点，从而不会遗漏应用程序中的任何漏洞。本文档描述了执行此操作的方法。它概述了要将用于分析不受支持 Web service 框架的新 Framework for Frameworks (F4F) 处理程序与 Web service 描述语言 (WSDL) 定制扫描程序集成而将执行的步骤。

有必要完成此工作，从而使 WSDL 定制扫描程序将能够区分那些已由此新 F4F 处理程序发现的 Web service 入口点和那些存在于 WSDL 文件中的 Web service 入口点。通过此信息，WSDL 定制扫描程序将清除已由此新 F4F 处理程序分析的配置结果。

- 『将 Web service F4F 处理程序连接到 WSDL 定制扫描程序』
- 第 129 页的『特征符映射』

将 Web service F4F 处理程序连接到 WSDL 定制扫描程序

开始之前

本部分假定已开发并测试主要 F4F 处理程序与合成方法生成。

过程

1. 添加从插件到 com.ibm.appscan.frameworks.wsdl.util 的依赖关系，其中包含用于报告所找到服务的助手 API。它还存在于 walalib 文件夹内。
2. 在主要处理程序（用于扩展 F4FHandler 类）中 handleApp() 方法的开头，添加以下行：

   ```
   String filePath = WSDLReportingUtil.getServicesFilePath
        (getFrameworksInput().getFileLocs());
   if (filePath!=null) {
   WSDLReportingUtil.initXmlDocument(filePath);
   }
   ```

   这些行用于创建将由定制 WSDL 扫描程序使用的临时文件。
3. 在 handleApp() 方法的结尾，添加以下行以确保在 F4F 处理程序退出之前保存所有已报告的服务：

   ```
   WSDLReportingUtil.saveXmlDocument();
   ```
4. 要报告服务，请从 WSDLReportingUtil API 添加对以下 API 之一的调用：
   - reportService(String targetNameSpace, String serviceName, String methodSignature)
   - reportService(String targetNamespace, String serviceName, String operationName, ArrayList<String> wsdlParams)

其中：

- targetNameSpace 是服务的目标名称空间。例如，在 JAX-WS 中，它位于注释中或与程序包名匹配。
- serviceName 是服务的名称。它位于注释中，或者可能与服务类名匹配。
- methodSignature 是 WSDL 文件中经过某些简化的方法特征符。
- operationName 是方法的名称，它链接到 WSDL 文件内一种端口类型中的操作名称。
- wsdlParams 是采用简单 WSDL 格式的参数类型列表。

5. 当前，定制 WSDL 扫描程序作为独立扫描程序或作为 Java/JSP 扫描的一部分来工作。如果新 F4F Web service 处理程序设计为使用 Java 以外的语言（例如 .NET），请确保通过执行以下步骤来使用适当扫描类型调用 WSDL 定制扫描程序：
   a. 转至 <data_dir>/ltd 目录（其中 <data_dir> 是 AppScan Source 程序数据的位置，如第 139 页的第 9 章, 『安装和用户数据文件位置』中所述）。
   b. 创建所有文件的备份副本以预防任何意外问题。
   c. 打开对应于适当扫描类型的 .ltd 文件。例如，对于 C++，打开 cpp.ltd 文件。
   d. 在结尾的 <LanguageTypeDefinition> 标记前添加以下行：

      ```
      <Scanner name="wsdl"/>
      ```

   e. 复制下一步将需要的 file_extention_set_name 值。
   f. 打开 file_extensions.xml 并查找具有上一步中所复制的名称的 FileExtensionSet。
   g. 在文件中该处添加以下行：

      ```
      <FileExtension extension="wsdl" assess="true"/>
      ```

   h. 保存所有文件并重新启动 AppScan Source。 WSDL 定制扫描程序现在将作为完全应用程序扫描的一部分来运行。

结果

此时，WSDL 定制扫描程序将仅显示未由框架实施的 WSDL 文件中的配置结果（在"结果"视图中）。已在应用程序中实施的 WSDL 文件中的任何服务都不会显示为配置结果。这些服务将由新 Web service F4F 处理程序进行分析。

特征符映射

关于此任务

WSDL 定制扫描程序会尝试将已报告的服务与对应的 WSDL 服务、WSDL 端口类型和 WSDL 操作进行匹配。这是以 WSDL 格式来报告 Web service 的原因。已添加一些简化来使内容更易于理解：

- 对于任何 XSD 基本类型，都无需添加目标名称空间。请改为仅使用名称（例如 string、int、float 和 double）。
- 对于集合和数组，将在类型后添加 []（例如 string[] 和 int[]）。
- 对于将映射到 XSD 复杂类型（使用 JAX-B 或任何其他技术）的类类型，类型将如下所示：

  ```
  http//my-types/myxsdnamespace/:Customer
  ```

名称空间应该作为 Java 注释的一部分来提供。如果它不是 Java 注释的一部分，那么应该与调用者服务名称空间或程序包名相匹配，具体取决于框架的实施。

注：映射错误将很可能阻止 WSDL 定制扫描程序将已报告的 Web service API 与关联 WSDL 操作进行匹配。如果发生此情况，那么将出现额外的一个配置结果。此额外结果应被视为错误肯定并表示映射问题。

如果您不确定预期的特征符应具有什么样的外观，请从 AppScan Source 对 WSDL 文件本身运行 WSDL 扫描。这将生成 WSDL 服务下的所有操作的结果。单击每个结果都会在"结果详细信息"视图中显示方法特征符。

第 8 章 AppScan Source 客户机组件错误消息

CRWSA0900E 无法对已排除的捆绑软件重命名

已排除的捆绑软件是存储所有已排除结果的内置捆绑软件。它不能被重命名。

CRWSA0907E 未能添加 <>，因为它已存在于数据库中且无法被覆盖。

定制规则已存在于 AppScan Source 数据库中且无法重新创建或覆盖。在以下其中一种情况下会发生此错误：

- 数据库中随附的规则标记为接收器，并且您在尝试创建的定制规则将接收器标记为感染传播。
- 数据库中随附的规则标记为传播器，并且您在尝试创建的定制规则通过添加感染传播而更改了该规则。

CRWSA0920E 将签名导入到项目时出错

请在导入签名之前验证项目的配置是否正确。

CRWSA0925E 为"{0}"输入的值无效。

输入的值的格式无效。请输入有效格式。

CRWSA0930E 为"{0}"输入的数据格式无效。

CRWSA0935E 为"{0}"输入的数字格式无效。

CRWSA0940E 未知用户

请确保用户在 Enterprise Server 上存在且已正确输入了用户信息。

CRWSA0945E 指定用户标识太短。用户标识的长度至少为一个字符。

CRWSA0950E 指定密码太短。密码的长度必须至少为 6 个字符。

CRWSA0955E 指定用户标识太长。用户标识的长度不能超过 255 个字符。

CRWSA0960E 指定用户标识包含无效字符。

CRWSA0965E 指定用户密码太长。密码的长度不能超过 16 个字符。

CRWSA0970E 指定密码不匹配。请重新输入密码。

CRWSA0975E 无法修改 Admin 用户。

CRWSA0980E 指定电子邮件地址无效。- 所需格式为 <user>@<domain>，其中 <domain> 不能超过 255 个字符。

CRWSA0985E"电子邮件"是必需的。

CRWSA0990E"名称"是必需的。

CRWSA0995E 指定"用户标识"已存在于存储库中，"名称"和"电子邮件"字段已更新。

CRWSA1010W 无法添加用户 {0}。

请参阅 AppScan Enterprise Server 帮助以获取相关信息。

CRWSA1050E 装入外部编辑器时出现意外错误。

请确保外部编辑器可用且工作正常。

CRWSA1055E 装入外部编辑器时出错。属性文件已损坏。

请确保外部编辑器可用且工作正常。

CRWSA1060E 装入外部编辑器时出错。属性文件缺失。

请确保外部编辑器可用且工作正常。

CRWSA1065E 装入外部编辑器时出错。无法装入首选编辑器

请确保外部编辑器可用且工作正常。

CRWSA1070E 装入外部编辑器时出错。编辑器插件返回了故障。

请确保外部编辑器可用且工作正常。

CRWSA1075E 装入外部编辑器时出错。无法创建可执行实例。

请确保外部编辑器可用且工作正常。

CRWSA1080E 装入 Eclipse 时出错。连接失败。

请确保已正确配置了要用作外部编辑器的 Eclipse（使用 plugins\ 目录中包含 com.ouncelabs.core.filelauncher.jar 的 Eclipse）。

CRWSA1085E 装入 Eclipse 时出错。服务器没有响应。

请确保已正确配置了要用作外部编辑器的 Eclipse（使用 plugins\ 目录中包含 com.ouncelabs.core.filelauncher.jar 的 Eclipse）。

CRWSA1090E 装入 Eclipse 时出错。协议失败。

CRWSA1095E 装入 Eclipse 时出错。Eclipse 的路径无效。

请确保已正确配置了要用作外部编辑器的 Eclipse（使用 plugins\ 目录中包含 com.ouncelabs.core.filelauncher.jar 的 Eclipse）。

CRWSA1100E 装入定制编辑器时出错。无法执行 {0}

CRWSA1120E 扫描失败

CRWSA1125E 检索扫描结果时发生错误

CRWSA1150W 警告：将"{0}"编辑为验证例程将对当前结果或跟踪图无效。确定要将根节点作为验证例程添加吗?

CRWSA1155E 需要至少一个源、接收器、源属性或接收器属性。

CRWSA1160E 评估正在运行时无法启动定制规则向导。

CRWSA1165E 添加定制规则时出错

CRWSA1170E 未能添加 {0}。

CRWSA1175E 未能添加 {0}，因为它已存在于数据库中且无法被覆盖。

CRWSA1185E 验证失败，因为编译过程中出错

CRWSA1190E 编译失败

CRWSA1195E 尝试打开编辑器时发生错误。

当编辑器未能打开时，请尝试重新启动 AppScan Source。

CRWSA1200E 在该系统上找不到选定文件。

CRWSA1205E 创建评估的源代码标记时发生错误。

CRWSA1210E 找不到源代码片段功能的一个或多个源文件。希望系统提示您提供缺失文件的位置吗?

CRWSA1215E 尝试连接到 AppScan Enterprise Console 时失败。- 请验证服务器是否在指定位置上运行。

CRWSA1220E {0,number} 秒后登录超时 - 建议重新启动 AppScan Enterprise Console 服务。

CRWSA1225E 存储未保存的过滤器时出错。将不保存对过滤器的更改。

CRWSA1230E 扫描时无法保存评估。

CRWSA1240E 找不到应用程序

CRWSA1245E 找不到 {0}

请验证指定 JRE 的位置。

CRWSA1250E 找不到 startup.jar 或 org.eclipse.equinox.launcher 插件

请验证指定基于 Eclipse 的安装。

CRWSA1290W 一个或多个项目缺省 JDK。已为这些项目设置了缺省 JDK。

AppScan Source 未能识别已导入项目的指定 JDK。要更正此问题，请使用 Java 和 JSP 首选项页面将其添加为已识别的 JDK。

CRWSA1295W 项目缺少 JDK。已为此项目设置了缺省 JDK。

AppScan Source 未能识别已导入项目的指定 JDK。要更正此问题，请使用 Java 和 JSP 首选项页面将其添加为已识别的 JDK。

CRWSA1300E 一个或多个项目缺少 JDK，且未指定缺省值。单击"确定"按钮后将提示您选择缺省 JDK。稍后，此 JDK 将用于未明确设置 JDK 的所有项目。

CRWSA1305E 应用程序是只读的。对评估所作的更改不会在以后的扫描中持续存在。

可能因为应用程序是远程应用程序或需要保存。无论何种情况，对评估所作的更改在以后的扫描中将不可用。

CRWSA1340E"{0}"未能注册

CRWSA1345E 应用程序文件不可写入。

CRWSA1350E 删除条目时出错

CRWSA1355E 评估正在运行时无法删除应用程序。

CRWSA1360E 评估正在运行时无法删除项目。

CRWSA1365W 您想要对其应用更改的任何项目都需要刷新。

CRWSA1370W 某些选定结果具有现有注释。确定要将其覆盖吗?

CRWSA1375W 未能导出到文件

CRWSA1380W 没有要导出的结果。评估结果中缺少所有选定的结果。

CRWSA1383E 没有要提交的结果。所有选定结果之前已提交，并且已设置了"仅提交缺陷一次"首选项。

CRWSA1385W 必须完成以下字段才能提交缺陷：

CRWSA1395W 捆绑软件已成功保存到 {0}。

CRWSA1400E 找不到项目文件"{0}"。

请验证 AppScan Source 项目文件是否存在。

CRWSA1405E 找不到 ASP.NET 目录"{0}"

请验证 ASP.NET 目录是否存在。

CRWSA1410E 导入项目文件"{0}"时出错

CRWSA1415E 导入 ASP.NET 目录"{0}"时出错

CRWSA1420E 创建项目时出错

CRWSA1425E 工作目录"{0}"不存在。请选择有效目录。

CRWSA1430E Web 上下文根"{0}"不存在。请选择有效的目录或 WAR 文件。

CRWSA1435E Web 上下文根"{0}"不能是父目录或是与项目目录"{1}"相同的目录。

CRWSA1440E 内容根"{0}"不存在。请选择有效目录。

CRWSA1445E 文件或目录"{0}"不存在。请选择有效的文件或目录。

CRWSA1450E 内容根"{0}"不能是父目录或是与项目目录"{1}"相同的目录。

CRWSA1455E 方法 {0} 已具有针对类型 {1} 的 ActionObject。

CRWSA1460E"不是"有效目录

CRWSA1465E 项目缺少 JDK

CRWSA1470E 该 Java/JSP 项目引用了无效的 JDK ({0})。将会提供缺省 JDK。

CRWSA1475E JDK 名称已存在

CRWSA1480E 名为"{0}"的 JDK 已存在。

请指定唯一 JDK 名称。

CRWSA1485E JDK 名称必须至少具有一个字符。

CRWSA1490E 必须为该操作选择 JDK

必须选择现有 JDK。

CRWSA1495E 无法保存

CRWSA1500E 评估正在运行时无法保存

CRWSA1505E 源根目录"{0}"已作为根目录存在。

CRWSA1510E 目录"{0}"不存在。请选择有效目录。

CRWSA1515E 预编译目录"{0}"不存在。请选择有效目录。

CRWSA1520E 类 {0} 存在于依赖关系 {1} 中和源路径上。

请验证指定的依赖关系。

CRWSA1525E 类路径 ({0}) 上出现意外文件类型。

请确保 Java 项目仅包含 .jar、.class 或 .java 文件。

CRWSA1530E 扫描正在运行时无法验证项目。

CRWSA1535E

CRWSA1540E 配置名称是空的

CRWSA1545E 源根目录无效

CRWSA1550E 源根目录"{0}"无效。- 其本身为文件的源根目录必须位于工作目录下。

CRWSA1555E 找不到有效的上下文根。

CRWSA1565E 已成功为 {0} 创建编码例程

CRWSA1568E 无法解析路径"{0}"中的变量

请使用"更改变量"首选项页面添加路径变量。

CRWSA1570E 属性名称 {0} 无效

CRWSA1575E 属性值 {0} 无效

CRWSA1580E 必须首先创建属性，然后才能添加值。

CRWSA1585E 应用程序必须具有非空名称。

CRWSA1590E 条目"{0}"重复。请指定唯一值。

CRWSA1595E 名称不能为空。

CRWSA1600E 必须选择包含结果的视图后才能生成报告。

CRWSA1625E 访问被拒绝。当前帐户缺少针对 {0} 的许可权。

CRWSA1630E 尝试导入工作空间时发生错误。请确保在"首选项 > Eclipse 工作空间导入器"中正确配置了导入器。

CRWSA1640E 无法添加项目。{0} 是引用或只读应用程序。

该应用程序是只读的，或者是对 Eclipse 工作空间 或 Microsoft Visual Studio 解决方案的引用。无法添加项目。

CRWSA1650E 找不到样式表 {0}。

CRWSA1653E 错误

该错误后跟分号 (:) 和描述错误的消息。这些文档提供了某些这些消息的可能解决方案。

- 尝试使用 AppScan Source Maven 集成生成报告的尝试导致了错误"报告选项的语法不正确"。
- 运行 ounceauto scanapplication 导致了"CRWSA1653E: 您必须首先登录"
- 扫描在 AppScan Source for Security 中导致了"正在初始化管理器 applicationmanager"
- 使用 AppScan Source CLI 将评估发布到 AppScan Enterprise 的操作失败，错误代码为 CRWSA1653E
- 从 CLI 发布评估导致了 CRWSA1653E 错误

CRWSA1655E 模型需要名称

CRWSA1660E 必须为模式指定至少一个条件

CRWSA1665E 条件不能为空。

CRWSA1670E 模式需要文件。

CRWSA1675E 名称"{0}"的扫描规则已存在。选择唯一名称。

CRWSA1700E 报告布局当前为空。请在"报告布局"选项卡中添加报告元素。

CRWSA1705E 没有要导出的捆绑软件

CRWSA1710E 没有要导出的打开的评估

CRWSA1715E 您在"新密码"和"确认新密码"字段中输入的密码不匹配

CRWSA1725E 密码太短。请输入长度在 **6** 到 **16** 个字符之间的密码。

CRWSA1730E 密码太长。请输入长度在 **6** 到 **16** 个字符之间的密码。

CRWSA1735E 设置文件"{0}"没有显示名称，将被跳过。

CRWSA1736E 无法打开文件 **<filename>**。确保运行 **Automation Server** 所使用的用户对此文件具有许可权

CRWSA1737E 无法打开文件 **<filename>**。确保运行 **Automation Server** 所使用的用户对此文件具有许可权，并确保 **Automation Server** 是本地应用程序

CRWSA1740E 名为"{0}"的设置文件已打开。

CRWSA1745E 设置文件"{0}"对于用户不可见。

CRWSA1750I 该选项卡包含了某些需要重新启动 **AppScan Source** 服务以使更改生效的设置。在重新启动 **AppScan Source** 服务之前，请关闭该计算机上正在运行或已连接到该计算机的所有 **AppScan Source** 客户机应用程序。要立即重新启动服务吗?

CRWSA1755E"{0} 具有缺少一个或多个所需属性（**name**、**display_name** 或 **type**）的设置。将不装入此文件。"

CRWSA1760E 发生许可错误：

CRWSA1775E AppScan Enterprise Console 连接未能初始化。请在外部浏览器中验证带有以下设置的 **AppScan Enterprise Console** 的可用性。

CRWSA1780E AppScan Enterprise Console 连接未能初始化。请检查日志和/或在外部浏览器中验证带有以下设置的 **AppScan Enterprise Console** 的可用性。**-** 错误详细信息：

CRWSA1790E 未能将评估发布到 AppScan Enterprise Console。检查日志和/或联系系统管理员。

CRWSA1795E 未能初始化发布服务。请验证 AppScan Enterprise Console 连接设置。

CRWSA1800E 未能初始化发布服务。请验证 AppScan Enterprise Console 的可用性和您登录的能力。

CRWSA1805E 无法下载文件。- 尝试连接到站点时发生错误。

请确保您的互联网连接能正常工作且站点可用。

CRWSA1810E 需要首先登录，然后才能打开在 V7.0.0 产品中创建的捆绑软件

CRWSA9999E AppScan Source for Analysis 遇到严重错误，且必须关闭。

第 9 章 安装和用户数据文件位置

安装 AppScan Source 时，用户数据和配置文件存储在安装目录外。

- 『缺省安装位置』
- 『缺省 AppScan Source 数据目录』
- 『AppScan Source 临时文件位置』

缺省安装位置

安装了 AppScan Source 后，该软件位于以下缺省位置之一：

- Microsoft Windows 32 位版本：

  `<SYSTEMDRIVE>:\Program Files\IBM\AppScanSource`
- Microsoft Windows 64 位版本：

  `<SYSTEMDRIVE>:\Program Files (x86)\IBM\AppScanSource`
- Linux：如果您是 root 用户，那么安装向导会将软件安装在 `/opt/ibm/appscansource` 中。如果您不是 root 用户，那么可以安装 AppScan Source for Development Eclipse 插件，它在缺省情况下安装到 `<home_directory>/AppScan_Source`。
- macOS：`/Applications/AppScanSource.app`

要点：

- 安装目录名只能包含英语字符。不允许文件夹名包含非英语字符。
- 如果您是在 Windows 上进行安装，那么必须拥有管理员特权才能安装 AppScan Source 组件。
- 如果您是在 Linux 上进行安装，那么必须拥有 root 用户特权才能安装 AppScan Source 服务器组件。

缺省 AppScan Source 数据目录

AppScan Source 数据由诸如配置、样本和目录文件的项组成。安装了 AppScan Source 后，数据文件在缺省情况下位于以下位置：

- Microsoft Windows：`<SYSTEMDRIVE>:\ProgramData\IBM\AppScanSource`

  注： `ProgramData\` 是隐藏的文件夹，要查看该文件夹，必须在**资源管理器**中修改您的查看首选项以显示隐藏的文件和文件夹。
- Linux：`/var/opt/ibm/appscansource`
- macOS：`/Users/Shared/AppScanSource`

要了解如何更改 AppScan Source 数据目录的位置，请参阅第 140 页的『更改 AppScan Source 数据目录』。

AppScan Source 临时文件位置

某些 AppScan Source 操作会导致创建临时文件，这些临时文件缺省情况下存储在以下位置：

- Microsoft Windows：<SYSTEMDRIVE>:\ProgramData\IBM\AppScanSource\temp

  注：ProgramData\ 是隐藏的文件夹，要查看该文件夹，必须在**资源管理器**中修改您的查看首选项以显示隐藏的文件和文件夹。
- Linux：/var/opt/ibm/appscansource/temp
- macOS：/Users/Shared/AppScanSource/temp

临时文件位置始终在 AppScan Source 数据目录内的 temp 目录中。您可以通过更改数据目录来更改临时文件位置，如『更改 AppScan Source 数据目录』中所述。这将使 temp 位于您已选的数据目录中。

更改 AppScan Source 数据目录

您可能想要更改 AppScan Source 数据目录的位置，以管理硬盘空间。您可以按照本主题中的步骤来在 AppScan Source 安装后更改此位置。

开始之前

完成此任务之前，请确保所有 AppScan Source 客户机应用程序都已退出或关闭。AppScan Source 客户机应用程序包括：

- AppScan Source for Analysis
- AppScan Source for Development（Eclipse 或 Visual Studio 插件）（仅在 Windows 和 Linux 上受支持）
- AppScan Source 命令行界面 (CLI)
- AppScan Source for Automation

此外，如果您已安装 AppScan Source for Automation，请确保 自动化服务器 已关闭：

- 在 Windows 上，请停止 **IBM Security AppScan Source Automation** 服务。
- 在 Linux 上，发出以下命令：/etc/init.d/ounceautod stop
- 在 macOS 上，发出以下命令：launchctl stop com.ibm.appscan.autod

过程

1. 定义 APPSCAN_SOURCE_SHARED_DATA=<data_dir> 环境变量，其中 <data_dir> 是要将 AppScan Source 数据存储在的位置。

   注：

   - <data_dir> 位置必须是 AppScan Source 所安装在的机器上已存在的完整绝对路径。
   - <data_dir> 目录名称只能包含英语字符。不允许文件夹名包含非英语字符。
2. 找到安装 AppScan Source 时创建的缺省数据目录（请参阅第 139 页的『缺省 AppScan Source 数据目录』以了解缺省数据目录位置）。
3. 将缺省数据目录的内容复制或移动到上述环境变量中所指定的 <data_dir> 位置。
4. **仅适用于 Linux 上安装的 AppScan Source for Automation**：
   a. 编辑 /etc/init.d/ounceautod 文件。
   b. 找到以下行：

```
su - ounce -c
'export LD_LIBRARY_PATH="/opt/IBM/AppScan_Source/bin":$LD_LIBRARY_PATH &&
cd "/opt/IBM/AppScan_Source/bin" &&
"/opt/IBM/AppScan_Source/bin/ounceautod" -s' >>
"/var/opt/ibm/appscansource/logs/ounceautod_output.log" 2>&1 &
```

并将其替换为以下内容：

```
su - ounce -c
'export APPSCAN_SOURCE_SHARED_DATA=<new data directory path here> &&
export LD_LIBRARY_PATH="/opt/IBM/AppScan_Source/bin":$LD_LIBRARY_PATH &&
cd "/opt/IBM/AppScan_Source/bin" &&
"/opt/IBM/AppScan_Source/bin/ounceautod" -s' >>
"<new data directory path here>/logs/ounceautod_output.log" 2>&1 &
```

注：以上命令为一行。

c. 保存 `/etc/init.d/ounceautod` 文件。

下一步做什么

如果您已安装 AppScan Source for Automation，请启动 自动化服务器：

- 在 Windows 上，请启动 **IBM Security AppScan Source Automation** 服务。
- 在 Linux 上，发出以下命令：`/etc/init.d/ounceautod start`
- 在 macOS 上，发出以下命令：`launchctl start com.ibm.appscan.autod`

分类

结果由 AppScan Source 分类以指示它们是安全性结果还是扫描覆盖范围结果。安全性结果表示实际或可能的安全漏洞 - 而扫描覆盖范围结果表示可改进配置以提供更好的扫描覆盖范围的区域。

每个结果都属于以下分类之一：

- **明确**安全性结果：一种结果，其中包含明确的设计、实施或策略违例，此违例为攻击者提供机会来使应用程序以一种非意愿方式运行。

  该攻击会导致对数据、系统或资源的未授权访问、偷窃或损坏。每个明确安全性结果都得到完整而清楚的表达，并且漏洞情况的特定底层模式已知并予以描述。
- **可疑**安全性结果：一种结果，指示可疑且可能存在漏洞的情况，该情况需要更多的信息或调查。不正确使用时可能会产生漏洞的代码元素或结构。

  可疑结果不同于明确结果，因为存在某种未知的情况妨碍对漏洞进行最终确定。此不确定性的示例可以是使用动态元素或者使用源代码不可用于的库函数。因此，需要多一级别的调查才能对可疑结果是否为明确结果进行确认或否定。
- **扫描覆盖范围**结果：表示可改进配置以提供更佳扫描覆盖范围的区域的结果（例如，丢失接收器结果）。

注：某些情况下，**无**分类可用于表示某个分类既不是安全性结果也不是扫描覆盖范围结果。

声明

本信息是为在美国国内供应的产品和服务而编写的。IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您所在区域当前可获得的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并不意味着授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往：

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

有关双字节字符集 (DBCS) 信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往：

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
19-21, Nihonbashi-Hakozakicho, Chuo-ku
Tokyo 103-8510, Japan

本条款不适用于英国或任何这样的条款与当地法律不一致的国家或地区：

INTERNATIONAL BUSINESS MACHINES CORPORATION"按现状"提供此出版物，不附有任何种类的（无论是明示的还是默示的）保证，包括但不限于默示的有关非侵权、适销或适用于某种特定用途的保证或条件。

某些国家或地区在某些交易中不允许免除明示或默示的保证。 因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。 IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对任何非 IBM Web 站点的引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的：(i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及 (ii) 允许相互使用已交换的信息，请与下列地址联系：

IBM Corporation
2Z4A/101
11400 Burnet Road
Austin, TX 78758 U.S.A.

只要遵守适当的条件和条款，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本文档中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际程序许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示目标和意愿。

显示的所有 IBM 价格皆为 IBM 建议的最新零售价格，且可随时更改，而不另行通知。经销价格可能会有差异。

本信息仅限规划目的使用。在提供所述的产品之前，此处的信息得随时更改。

本信息包含在日常业务操作中使用的数据和报告的示例。为了尽可能完整地说明这些示例，示例中可能会包括个人、公司、品牌和产品的名称。 所有这些名称都是虚构的，如与实际商业企业所使用的名称和地址有任何雷同，纯属巧合。

版权许可证：

本信息包含源语言形式的样本应用程序，用以阐明在不同操作平台上的编程技术。如果是为按照在编写样本程序的操作平台上的应用程序编程接口（API）进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。如果是为按照 IBM 的应用程序编程接口 (API) 进行应用程序的开发、使用、经销或分发为目的，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。

这些样本程序的每份拷贝/任何部分或任何衍生产品，都必须包括如下版权声明：

© (贵公司的名称) (年份)。此代码的某些部分是根据 IBM 公司的样本程序衍生出来的。

如果您正在查看本信息的软拷贝形式，图片和彩色图例可能无法显示。

商标

IBM、IBM 徽标和 ibm.com 是 International Business Machines Corp. 在全球许多管辖区域内的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。IBM 商标的当前列表可从 Web 站点 www.ibm.com/legal/copytrade.shtml 处获取。

Adobe、Acrobat、PostScript 以及所有基于 Adobe 的徽标是Adobe Systems Incorporated 在美国和/或其他国家或地区的注册商标或商标。

IT Infrastructure Library 是 Central Computer and Telecommunications Agency 的注册商标，该企业现已成为 Office of Government Commerce 的一部分。

Intel、Intel 徽标、Intel Inside、Intel Inside 徽标、Intel Centrino、Intel Centrino 徽标、Celeron、Intel Xeon、Intel SpeedStep、Itanium 和 Pentium 是 Intel Corporation 或其子公司在美国和其他国家或地区的商标或注册商标。

Linux 是 Linus Torvalds 在美国和/或其他国家或地区的商标。

Microsoft、Windows、Windows NT 以及 Windows 徽标是 Microsoft Corporation 在美国和/或其他国家或地区的商标。

ITIL 是英国政府商务部的注册商标和欧盟注册商标，且已在美国专利和商标局注册。

UNIX 是 The Open Group 在美国和其他国家或地区的注册商标。

Java 和所有基于 Java 的商标和徽标是 Oracle 和/或其子公司的商标或注册商标。

Cell Broadband Engine 是 Sony Computer Entertainment, Inc. 在美国和/或其他国家或地区的商标。

Tape-Open、LTO、LTO徽标、Ultrium 和 Ultrium 徽标是 HP、IBM Corp. 及 Quantum在美国和/或其他国家或地区的商标。

索引

C

D

F

L

O

W

Printed in China