Oracle® AutoVue Integration Software Development Toolkit (ISDK)

Security Guide

Release 21.1.0

F86929-01

September 2023

Oracle AutoVue Integration SDK Security Guide, Release 21.1.0

F86929-01

Primary Author:

Contributing Author:

Contributor:

Contents

Preface

This documentation provides guidelines on how to create secure integrations with the AutoVue Integration Software Development Toolkit (ISDK), all of its components, and the communication between the different components. For the most up-to-date version of this document, go to the AutoVue Documentation Web site on the Oracle Technology Network (OTN) at
`https://www.oracle.com/technetwork/documentation/autovue-091442.html`

Audience

This document is intended for Oracle partners and third-party developers (such as integrators) who want to create an integration between AutoVue and a content repository.

Related Documents

For more information, see the following documents:

- *Oracle AutoVue Integration SDK Overview and Installation Guide*
- *Oracle AutoVue Integration SDK Technical Guide*
- *Oracle AutoVue Client/Server Deployment Security Guide*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Overview

AutoVue Integration Software Development Toolkit (ISDK) is a software package designed for Oracle partners and third-party integrators to develop new integrations between AutoVue server and enterprise systems[1]. The purpose of the integration is to enable communication between AutoVue and the enterprise systems as well as to integrate AutoVue's capabilities into their environments.

1.1 General Security Principles

This section outlines the general security principles of the ISDK.

1.1.1 Keep Up to Date on Software

One of the principles of good security practice is to keep all software versions and patches up-to-date. Oracle continually improves its software and documentation. Make sure you install the latest version of ISDK. Throughout this document, an ISDK maintenance level of 21.1.0 is assumed. Refer to the following link for updates on Oracle patches, security alerts and third-party bulletins: `https://www.oracle.com/technetwork/topics/security/alerts-086861.html`.

1.1.2 Authentication

Ensure that all users accessing data through an ISDK integration are authenticated and verified to have access to the data. The user must log in to filesys or other DMSes before viewing a file and/or creating markups.

[1] Also referred to as Document Management Systems (DMS).

2

Security Considerations

Since the ISDK is intended for developers/integrators, security considerations vary between development (design time) and deployment (run time) environments. Typically, a development environment consists of a desktop machine that has development tools (such as JDeveloper or Eclipse) and an application server configured to talk to a backend DMS server. Where as, a deployment environment is more complex and distributed (that is, it consists of several machines/servers). Note that administrators may want to include additional hardware/software (such as load balancer, proxy, web servers, and so on) into the environment.

2.1 Security Consideration for Development Environment (Developers/Integrators)

Before building a secure integration based on ISDK, you must ensure that the setup is in a secure development environment. The security considerations for setting up a secure environment are different from building a secure integration.

2.1.1 Installing ISDK

For installation and configuration steps, refer to the *Oracle AutoVue Integration SDK Design, Installation and Configuration Guide*.

Take note of the following security considerations when installing the ISDK:

- You should only deploy and run the required ISDK projects.
- The ISDK samples (ISDK Skeleton, and Sample Integration for FileSys DMS) are included in the zipped media package. The samples are meant for education and demonstration purposes only and are not meant for production. You must ensure that your integration module is developed, tested, and secured before deploying in production environment.
- Default users for the sample integration for FileSys are not enabled.
- File permissions for ISDK applications (such as ISDK Skeleton) in the ISDK installation folder do not allow global access on Linux operating systems.
- The ISDK VueServlet is configured by default to use secure socket layers (SSL) to communicate with the AutoVue server.

2.1.2 Post-Installation Configurations

This section discusses the following security-related configurations that can be made after installing the ISDK.

2.1.2.1 Adding User Credentials to the Sample Integration for FileSys Project

User credentials for the Sample Integration for FileSys project are controlled by the credential.txt file. For more information refer to *Configuring User Control* section of the *ISDK Design and Installation Guide*.

2.1.2.2 Enterprise Security API Resource Files

This release of the ISDK includes the Open Web Application Security Project (OWASP) Enterprise Security API (ESAPI) Java Edition and related ESAPI to provide enhanced security. The latest version is available for download from `https://github.com/esapi/esapi-java-legacy`.

The ISDK customizes the following default resource files provided by the ESAPI:

- *AvESAPI.properties*
- *Validation.properties*

The sample integration for FileSys customizes the following in the *Validation.properties* file in order to validate URLs:

- *Validator.URL*
- *Validator.URL1*

The sample integration for FileSys customizes the following in the AvESAPI.properties file:

- *HttpUtilities.ApprovedUploadExtensions*
  Allows files extensions for the FileSys repository sample files. All new file formats must have their extension added to this parameter.
- *Validator.FileName*
  Validates file names.

  The default value is as follows:

  Validator.FileName=^[a-zA-Z0-9!@#$%^&{}\\[\\]()_+\\-=,.~'`
  \\p{L}&&[^*/:?|<>\\\"\\x00-\\x1F]]{1,255}$
- *Validator.DirectoryName*
  Validates directory path.

  The default value is as follows:

  Validator.DirectoryName=^[a-zA-Z0-9:/\\\\!@#$%^&{}\\[\\]()_+\\-=,.~'`
  \\p{L}&&[^*?|<>\"\\x00-\\x1F]]{1,255}$

The ISDK Skeleton project only customizes the *AvESAPI.properties* file.

After installing the ISDK, the customized resource files for the ISDK projects can be accessed from the ISDK installation folder. For example, for the sample integration for FileSys project the location is the FileSys/ESAPI_resources directory.

Since ESAPI requires that a file path must match the canonical path exactly, and all file paths defined in the web.xml descriptor file of the ISDK/ISDK-based projects should be case-sensitive.

If it is required to add new files to the FileSys repository folder, make sure that these files meet the following requirements of the rules defined in the ESAPI resource files:

- The filename must match the regular expression defined by Validator.FileName.
- The directory must match the regular expression defined by Validator.DirectoryName.

- The file extension must be included in the allowed list as defined by HttpUtilities.ApprovedUploadExtensions.

ESAPI has a default search order to find and load its resource files. That is, the application server searches specific folder locations for the resource files and then loads these resources before loading applications. It is possible to change the location of the resource files by using -Dorg.owasp.espai.resources JAVA_OPTIONS in the WebLogic application servers' startup or in the setDomainEnv script.

Example 2–1 Changing the location of the resource files

```
Step 1: Copy the contents from inside the ESAPI_resources folder for the ISDK
project to a secure directory.
For example: C:\mysafe_esapi_resources_locations
Step 2: Edit startWebLog.cmd and add a new JAVA_OPTIONS.
For example: Set JAVA_OPTIONS-...-Dorg.owasp.esapi.resources-C:\mysafe_esapi_
resources_location
Step 3: Start WebLogic Server. For the sample integration for FileSys project, the
WebLogic Server console should state that the C:\mysafe_easpi_resources_
location\EASPI.properties directory is found in org.owasp.esapi.resources. For the
ISDK Skeleton project, the WebLogic Server only displays information if there is
an error locating the resources.
```

Note: You must safe-guard your ESAPI resources directory to avoid unauthorized access.

2.1.2.3 Securing the FileSys Content Repository Folder

The Sample Integration for the FileSys project allows viewing of files located inside the FileSys content repository root (RootDir). The RootDir is defined in the web.xml descriptor file. It is important to note that you should not expose any other local files inside RootDir except for the sample content repository data shipped with FileSys (filesysRepository.zip) or any data that is to be viewed in AutoVue through the ISDK-based integration.

2.1.2.4 Configuring SSL Communication

A SSL-enabled development environment for an ISDK-based integration includes the following components: ISDK, AutoVue server, Oracle WebLogic Server, and IDE.

2.1.2.4.1 ISDK User default SSL configuration Ensure that the web.xml descriptor file for the ISDK project uses the following VueServlet init-param:

```
<init-param>
<param-name>EnableSSL</param-name>
<param-value>true</param-value>
</init-param>
```

2.1.2.4.2 Enabling SSL between Oracle WebLogic Server and IDE For information on how to set SSL communication between WebLogic Server and a client (such as Eclipse), refer to the "Configuring Demo Certificates for Clients" section of the *Oracle Fusion Middleware Securing Oracle WebLogic Server* document.

2.1.2.4.3 Oracle WebLogic Server For information on how to set SSL refer to the "Setting Up SSL: Main Steps" section of the *Oracle Fusion Middleware Securing Oracle WebLogic Server* document.

2.1.2.4.4 AutoVue Server For information on how to set SSL for Oracle AutoVue with a valid CA-issued certificate, refer to the *Oracle AutoVue Client/Server Security Guide*.

> **Note:** During development you may create a self-signed certificate to test. Alternately, you may use the two default demo keystores provided by the WebLogic Server to setup SSL communication between WebLogic Server and the AutoVue server. For more information, refer to Security Considerations for ISDK Integrators.

2.1.2.4.5 Disabling SSL Communication Depending on your needs, the default SSL configuration of the ISDK may be disabled by updating the web.xml descriptor file for the ISDK project with the following VueServlet init-param:

```
<init-param>
<param-name>EnableSSL</param-name>
<param-value>false</param-value>
</init-param>
```

2.2 Security Considerations for ISDK Integrators

This section provides security-specific information for developers producing ISDK-based integrations.

2.2.1 Security Enhancement for Integrations with AutoVue

If you are developing your own integration between the AutoVue server and a document repository (typically, through the use of the ISDK that ships with AutoVue), there are a few points that have to be considered to enhance the system's overall security. For example, you must ensure that the original URL to a file does not contain sensitive information such as user or server information. For more information, refer to the "Integrations with AutoVue" section of the *Oracle AutoVue Client/Server Deployment Security Guide*.

2.2.2 Vuelink Cryptography Algorithms

For security reasons, the Vuelink cryptography algorithms have been upgraded. The communication channel between AutoVue and Vuelink is encrypted providing an additional level of security. All information transferred from AutoVue Server to Vuelink is encrypted using the Diffie-Hellman protocol. The 21.1.0 AutoVue installer uses Elliptic-Curve (EC) algorithm for Key-Agreement and Advanced Encryption Standard (AES) for the data encryption. However, both the algorithms are configurable without having to make any new changes to the AutoVue codebase. In order to maintain the highest possible level of encryption security, the algorithms have to be updated in the future according to the state of art of cryptography.

New configuration parameters have been added on both AutoVue Server side and Vuelink side to specify the cryptography algorithms to use.

On AutoVue side, the administrator has to specify these algorithms in jvueserver.properties configuration file, in the variables:

- jvueserver.encryption.algorithm
- jvueserver.encryption.keyagreement.algorithm

Example 2–2 Examples of Algorithms

```
jvueserver.encryption.algorithm=AES
jvueserver.encryption.keyagreement.algorithm=EC
```

On the Vuelink side, you have to specify the two following initialization parameters in the Vuelink servlet descriptor within the web configuration file (web.xml):

- DecryptionAlgorithm
- KeyAgreementAlgorithm

Example 2–3 Examples of Initialization Parameters

```
<servlet>
    <servlet-name>FilesysVuelink</servlet-name>
    <servlet-class>com.cimmetry.vuelink.filesys.FilesysVuelink</servlet-class>

    <!-- Encryption Algtorithms -->
    <init-param>
       <param-name>DecryptionAlgorithm</param-name>
       <param-value>AES</param-value>
    </init-param>
    <init-param>
       <param-name>KeyAgreementAlgorithm</param-name>
       <param-value>EC</param-value>
    </init-param>
…
</servlet>
```

All the parameters are required by an integration solution because AutoVue does not define default values for them. So, the integration solution will not work if they are not provided. AutoVue installer add them currently to jvueserver.properties and the Vuelink web configuration file. It uses the following values for them:

- Encryption / Decryption Algorithm: AES
- Key Agreement algorithm: EC

Note: AutoVue 21.1.0 is backward compatible with the previous versions of Vuelinks.

- If you update AutoVue and run it with an old version of Vuelink, then ensure to remove these parameters from the jvueserver.properties configuration file to force AutoVue using the old encryption support.
- The encryption algorithms used to pass authentication information between the AutoVue server and document repositories have been made configurable. Both the AutoVue server and ISDK framework were modified to enable this enhancement. The goal of this enhancement is to allow more modern algorithms to be used without requiring code changes. Note that the cryptographic algorithms configured for AutoVue and the VueLink must match. Otherwise, it will not be possible to pass authentication information.
- In addition, you have to verify which encryption algorithms are supported by the Java version used by your servers to select them accordingly. It is highly recommended to use the same versions of Java on both AutoVue and Vuelink sides to ensure having the same encryption support on both sides. Using for example EC/AES on Java 8 update 144 will not match itself on Java 8 update 172 because Oracle introduced a change on the cryptosystem strength when it released Java 8 update 151.

2.2.3 Handling User Credentials after Authorization Exception Being Thrown

Since user credentials captured by AutoVue when an authorization exception is thrown are sent only once, it is important for the ISDK to cache the authentication information and retrieve them in subsequent requests. This is done by caching the connection object inside the DMSSessionImp (subclass of DMSSession). For more information, refer to the *AutoVue Integration SDK Technical Guide*.

2.2.4 Setting SSL Communication

The following steps describe how to set up SSL communication.

> **Note:** The certificates used in the following steps are self-signed and cannot be used in a production environment.

1. Create a client certificate.

   Use the CerGen utility provided with WebLogic to create a client certificate. WebLogic Server trusts the certificate as it is already configured to trust the demo CA.

   - Run C:\Oracle\Middleware\user_projects\domains\mydomain\bin\setDomainEnv.bat
   - Select a directory to store the client certificate. For example, c:\temp
   - Run the following command to create the following files: avcertfile.cer.pem, avcertfile.cer.der, avkeyfile.key.pem and avkeyfile.key.der.

     ```
     java utils.CertGen -certfile avcertfile.cer -keyfile avkeyfile.key
     -keyfilepass password -cn autovue
     ```

   - Run the following command to create keystore c:\temp\avkeystore.jks.

     ```
     java utils.ImportPrivateKey -keystore avkeystore.jks -storepass password
     -storetype jks -keypass password -alias av -certfile avcertfile.cer.pem
     -keyfile avkeyfile.key.pem -keyfilepass password
     ```

2. Import the WebLogic demo certificate to the web browser.
3. Export the certificate from Internet Explorer as a base-64 encoded format and then save the certificate as c:\temp\wlsdemo.cer.
4. Import the certificate to the AutoVue server's JRE using the following command:

   ```
   keytool -import -alias democert  -v -file c:\temp\wlsdemo.cer -keystore AV_
   INSTALL\jre\lib\security\cacerts
   ```

5. Update the jvueserver.properties file with the following:

   ```
   jvueserver.ssl.enable=true
   jvueserver.cmdline=… -Djavax.net.ssl.keyStore=" c:\temp\avkeystore.jks"
   -Djavax.net.ssl.keyStorePassword="password" ...
   ```

6. Restart the AutoVue server.

2.2.5 Security Recommendations

- The installer of a third-party integration with DMS/PLM systems based on ISDK framework should offer the lockdown mode listed by the following:

  - Avoid default accounts.
  - Remove test scripts, protect Web administration pages if exiting, and do not install demonstration code by defaults.
  - Avoid inappropriate file permissions. That is, avoid giving more permissions than necessary.
  - Debugging scripts must be well documented, optional, and protected from unauthorized use.
  - Remove any unnecessary files.
  - Avoid dumping sensitive data to installation logs and protect them from unauthorized access.

- Analyze your source code using security analyzing tools.
- Familiarize yourself with security risks. For example, the Open Web Application Security Project (OWASP) offers security related information at `https://www.owasp.org`.

3
Security Features

This section outlines specific security mechanisms offered by the ISDK.

3.1 The Security Model

The ISDK security features arise from the need to protect data from deliberate unauthorized attempts to access the file stored in backend repository.

The critical security features that provide this protection are:

- Restrict IP Access - Restrict access to the backend DMS/PLM systems and ISDK-based integration.
- Authentication - Ensure that only authenticated individuals get access to DMS/PLM systems through ISDK-based integration.
- Authorization - This is only for documents stored inside a DMS.

3.2 Configuring and Using Restrict IP Access

It is recommended to tighten the deployment and limit access to the ISDK through a filtering mechanism provided by the WebLogic application server.

The following steps describe how to configure the filtering mechanism.

1. Log onto WebLogic Admin console.
2. From the left panel, select the domain that you want to configure (the domain that AutoVue Web Services is deployed on).
3. Select **Security** and then **Filter**.
4. Select the **Connection Logger Enabled** checkbox to enable the logging of accepted messages. The Connection Logger logs successful connections and connection data in the server. This information can be used to debug problems relating to server connections.
5. In the **Connection Filter** field, specify the connection filter class to be used in the domain.

   To configure the default connection, specify

   ```
   weblogic.security.net.ConnectionFilterImpl
   ```

6. In the **Connection Filter Rules** field, enter the syntax for the connection filter rules. The syntax is as follows:

   ```
   Target localAddress localPort action protocols
   ```

   The following is the recommended rule set:

```
# Allow access from the Weblogic application server machine
<Weblogic IP or hostname> * * allow
# Allow access from the AutoVue machine
<autovue IP or hostname> * * allow
# Allow access from IP address range
<IP range to be permitted> * * allow
# Refuse the other access for all other machines
0.0.0.0/0 * * deny
```

Replace the <*Weblogic IP or hostname*> and <*autovue IP or hostname*> with the actual hostname or IP address of the machines and <*IP range to be permitted*> with the real IP address range.

For information on connection filter rules and syntax, refer to the "Using Network Connection Filters" section in the *Oracle Fusion Middleware Oracle WebLogic Server Administration Console Help*.

7. Click **Save**.
8. Restart the WebLogic Server so that your changes can take effect.

Note: If you accidentally enter rules that completely block access to the WebLogic server, and are no longer able to access the administration console, you must locate the config.xml file inside the WebLogic server machine (under the domain directory) and remove the <connection-filter-rule> parameters that deny access to the server from legitimate machines.

3.3 Configuring and Using Authentication, Encryption and Signature

To facilitate the integration with the backend system, ISDK provides the following authentication mechanism: Single Sign On (SSO) with Cookies.

The following section is a high-level description about security and authentication mechanisms provided in this release of Integration SDK. Refer to *the AutoVue Integration SDK Technical Guide* for detailed information about how to implement the security and authentication mechanisms in your integration.

3.3.1 Single Sign-On (SSO)/Cookies

The ISDK can leverage cookie information sent by the Web browser when AutoVue is launched to achieve the single sign-on (SSO) authentication process. When using the WebStart deployment, the administrator can specify the names of cookies whose values should be passed to the AutoVue Client through the VueJNLPServlet configuration settings. This is similar to the DMS_PRESERVE_COOKIES parameter facility that was provided with the Applet deployment of the client in previous releases. The AutoVue server will pass the names and values of the specified cookies to the integration.

For many integrations, the authentication information will be associated with the standard JSESSION cookie. In such cases, it is recommended to set the AutoVue DMS_PRESERVE_ COOKIES parameter (if the applet is being used), or the "Cookies" configuration parameter of the VueJNLPServlet to "JSESSIONID".

Caution: Setting the DMS_PRESERVE_COOKIES to TRUE directs AutoVue to pass all cookies to the integration. For security reasons, it is not recommended to set this parameter to TRUE.

3.3.2 Encryption of User Credentials

The authorization block is encrypted by AutoVue. To support the authorization encryption:

- ISDK handles a getProperty request to return a value for the ISDK public key.
- The AutoVue server uses the ISDK public key to encrypt the entire authorization block.
- The AutoVue server includes its public key in requests sent to the ISDK. The ISDK then uses AutoVue's public key to decrypt the content of the authorization block.
- All the above handling is done at the ISDK framework level and is transparent to your ISDK-based integrations. Existing interfaces for returning authorization elements remain unchanged (that is, the authorization element from the existing interface has been decrypted by the ISDK framework).

> **Note:** In addition to the encryption of the authorization block, the password inside the authorization block has another level of encryption.

3.3.3 Logging and User Info

For security reasons, the ISDK does not dump to the logs any sensitive user/password information. It is recommended that any components that you add to the integration should do the same.

3.4 Configuring and Using Java Security Manager for Oracle WebLogic

The Java Security Manager is used with WebLogic Server in order to provide additional protection for resources running on a Java Virtual Machine (JVM).

For detailed information on setting up the Java Security Manager, refer to the *Oracle Fusion Middleware Programming Security for Oracle WebLogic Server*.

WebLogic Server provides a sample Java security policy file, weblogic.policy, which you can edit and use. For instance, if WebLogic Server is installed in the default installation directory, you should customize WL_HOME\server\lib\weblogic.policy for the ISDK.

> **Note:** This policy file must be configured and the permissions based on your configuration must be added in order for WebLogic Server and associated applications to work properly.

If the Sample Integration for FileSys is deployed on WebLogic Server outside of the IDE, then you must add the lines of code provided in Example 3–1 to weblogic.policy.

Example 3–1 Code Sample

```
grant codeBase "file:/C:/Oracle/Middleware_10.3.5/user_
  projects/domains/mydomain/servers/AdminServer/tmp/_WL_user/filesys/-"
{
   permission java.io.FilePermission "<<ALL FILES>>", "read";
   permission java.io.FilePermission "C:\\temp\\-", "read,write,delete";
   //folder for log4j log files: filesys.log, filesys.log.1., ...
   permission java.io.FilePermission "C:\\Users\\AppData\\Local\\-",
     "read,write,delete";
   //On Windows, user's LOCALAPPDATA, in vuelink.java, File.createTempFile() needs
   //this folder
   permission java.io.FilePermission "C:\\Oracle\\filesysRepository\\-",
     "read,write,delete";

   permission java.util.PropertyPermission "*", "read,write";
   //Needed for vuelink.java, if keeping current System.getProperties()
   //Comment the above line and use the following one if using
   //System.setProperty("java.protocol.handler.pkgs",..) instead of
   //System.getProperties()
   //permission java.util.PropertyPermission "java.protocol.handler.pkgs",
   // "read,write";

   permission java.util.PropertyPermission "org.owasp.esapi.*", "read";
   permission java.util.PropertyPermission "user.home", "read";
   permission java.util.PropertyPermission "user.dir", "read";
   permission java.util.logging.LoggingPermission "control";

   permission java.security.SecurityPermission "insertProvider.SunJSSE";
   permission java.lang.RuntimePermission "createClassLoader";
   permission java.lang.RuntimePermission "setContextClassLoader";
};
```

If the ISDK Skeleton is deployed on WebLogic Server outside of the IDE, then you must add the lines of code provided in Example 3–2 to weblogic.policy.

Example 3–2 Code Sample

```
grant codeBase "file:/C:/Oracle/Middleware_10.3.5/user_
  projects/domains/mydomain/servers/AdminServer/tmp/_WL_user/skeleton/-"
{
   permission java.io.FilePermission "C:\\temp\\mysafe_esapi_resources_
location\\-"
  "read";   //esapi resource file location
   permission java.io.FilePermission "C:\\temp\\autovueconnector.log",
"read,write";
   permission java.util.PropertyPermission "org.owasp.esapi.*", "read";
   permission java.util.PropertyPermission "*", "read,write";
   permission java.util.logging.LoggingPermission "control";
   permission java.security.SecurityPermission "insertProvider.SunJSSE";
};
```

If the ISDK project (FileSys, or ISDK Skeleton) is deployed on WebLogic Server from inside the IDE, then you must add the lines of code provided in Example 3–3 to weblogic.policy.

Example 3–3 Code Sample

```
grant codeBase "file:/C:/Oracle/Middleware_10.3.5/user_
  projects/domains/mydomain/servers/AdminServer/tmp/_WL_user/_auto_generated_ear_
  /-"
{
   permission java.io.FilePermission "<<ALL FILES>>", "read";
   //need to validate path
   permission java.io.FilePermission "c:\\temp\\-", "read,write,delete";
   //folder for log4j log files
   permission java.io.FilePermission "C:\\Oracle\\filesysRepository\\-",
    "read,write,delete";
   permission java.io.FilePermission "C:\\Users\\AppData\\Local\\-",
    "read,write,delete";
   //On Windows, user's LOCALAPPDATA, in vuelink.java, File.createTempFile() needs
   //this folder
   permission java.util.PropertyPermission "org.owasp.esapi.*", "read";
   permission java.util.PropertyPermission "user.home", "read";
   //required when deploying from eclipse
   permission java.util.PropertyPermission "user.dir", "read";
   //required when deploying from eclipse

   permission java.util.PropertyPermission "*", "read,write";
   //Required when vuelink.java does not get rid of System.getProperties() API
   //If removing System.getProperties() API from vuelink.java in the future, then
   //the above policy line can be removed and the following three used instead:
   //permission java.util.PropertyPermission
   // "com.sun.xml.ws.api.streaming.XMLStreamWriterFactory.woodstox", "write";
   //permission java.util.PropertyPermission
   // "com.sun.xml.ws.api.streaming.XMLStreamReaderFactory.woodstox", "write";
   //permission java.util.PropertyPermission "java.protocol.handler.pkgs",
   // "read,write";

   permission java.security.SecurityPermission "insertProvider.SunJSSE";

   permission java.lang.RuntimePermission "createClassLoader";
   permission java.net.SocketPermission "localhost:49702", "connect";
   //for wsclient only
   permission java.net.SocketPermission "stestvl:5099", "connect,resolve";
   //required to connect to AutoVue when deploying from eclipse
};

 // The following grant permission to eclipse. Not desalinized because of not
 //necessary during development phase.
 grant codeBase "file:/C:/eclipse/eclipse-workspace-helios/-"
{
   permission java.io.FilePermission "<<ALL FILES>>", "read,write,execute,delete";
   permission java.util.PropertyPermission "*", "read,write";
   permission java.security.SecurityPermission "insertProvider.SunJSSE";
};
```

A
Feedback

If you have any questions or require support for AutoVue please contact your system administrator.

If at any time you have questions or concerns regarding AutoVue, please contact us.

A.1 General AutoVue Information

Web Site	`https://www.oracle.com/applications/autovue/`

A.2 Oracle Customer Support

Web Site	`https://www.oracle.com/support/`

A.3 My Oracle Support AutoVue Community

Web Site	`https://community.oracle.com/hub/`

A.4 Sales Inquiries

E-mail	`https://www.oracle.com/corporate/contact/global.html`