Oracle® AutoVue

Security Guide

Release 21.1.0.4

F86913-02

April 2025

Security Guide, Release 21.1.0.4

F86913-02

Primary Author:

Contributing Author:

Contributor:

Contents

4 Java Web Start Client Deployment

A Feedback

Preface

This document provides guidelines on how to securely install and configure the AutoVue server and its associated components.

For the most up-to-date version of this document, go to the AutoVue Documentation Web site on the Oracle Technology Network 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, and system administrators) whose task is to ensure the secure installation and configuration of the AutoVue server.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Related Documents

For more information, see the following documents:

- *Oracle AutoVue, Client/Server Deployment Installation and Configuration Guide*
- *Oracle AutoVue, Client/Server Deployment Planning 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.

Convention	Meaning
`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 of Oracle AutoVue Security

AutoVue is Oracle's suite of Enterprise Visualization solutions, which are designed to view, digitally annotate and collaborate on any digital information in an organization. AutoVue delivers visualization capabilities for many document types, including business documents such as Office and Graphics, as well as technical document types such as 2-D/3-D Computer Aided Design (CAD) and Electronic Design Automation (EDA).

This section provides an overview of Oracle AutoVue, Client/Server Deployment and discusses the objectives and security principles of AutoVue.

1.1 Oracle AutoVue, Client/Server Deployment Overview

The Client/Server deployment has AutoVue installed on a server, to which client machines connect to access and view documents. The Client/Server deployment provides a complete, open and standards-based set of integration tools that allows customers to tie AutoVue to any enterprise applications. This deployment provides users with a consistent view of data and business objects, and expands work flow automation to document-based processes.

1.2 Security Objectives of Oracle AutoVue

The security objectives of Oracle AutoVue are based on the operational environments and risk scenarios in which AutoVue may be deployed. The security objectives are:

- Providing Basic Security Services
- Ensuring Deployment and Configuration Flexibility
- Ensuring Scalability and Predictability

1.2.1 Providing Basic Security Services

Oracle AutoVue integrates with the following security services required in a multi-user, networked environment:

- Authentication

  This service enables a system to verify the identity of users who request access to the AutoVue server.

- Authorization

  Authorization ensures a system grants access to resources in compliance with the security policies defined for those resources. Access decisions are based on the authenticated identity and the privileges given to the requesting user.

- Accountability

Accountability ensures that users who access the system can be held accountable for their usage of the system and system resources. This enables you to monitor system usage to identify unauthorized users.

- Data Protection

  This service prevents unauthorized users from accessing sensitive data. Use encryption to protect the confidentiality of data sent through a public network. Encryption can also be used to protect highly sensitive data from users who bypass access control mechanisms of a system.

1.2.2 Ensuring Deployment and Configuration Flexibility

Oracle AutoVue security services are designed to support the full range of AutoVue deployment scenarios. Security mechanisms in AutoVue are aimed at ensuring that practical, real-world constraints on deployment can be met. The constraints include the need to deploy certain components of AutoVue in the Demilitarized Zone (DMZ), to deploy it in the corporate intranet, and enable those components to communicate across a firewall.

1.2.3 Ensuring Scalability and Predictability

As systems grow in size, there will be a breaking point where a new server is required because of the overall workload or deployment requirements. You can scale your AutoVue deployment to meet your needs while ensuring a secure environment.

1.3 General Security Principles

This section describes fundamental principles to using Oracle AutoVue securely.

1.3.1 Keep Software Up to Date

It is good security practice to keep all software versions and patches up-to-date. Throughout this document, an AutoVue maintenance level of 21.1.0.4 is assumed. For updates on critical patches and other security alerts, refer to the `Oracle Critical Patch Updates, Security Alerts and Third Party Bulletin`.

1.3.2 Restrict Network Access to Oracle AutoVue

Keep both the AutoVue server and any document repository behind a firewall. In addition, you may want to place a firewall between AutoVue servers when deployed as a server farm. Firewalls provide an assurance that access to these systems is restricted to a known network route which can be monitored and, if necessary, restricted.

1.3.3 Keep Up to Date on Latest Security Information

Oracle continually improves its software and documentation. Make sure to check the Oracle AutoVue Documentation Library on the `Oracle Technology Network (OTN)` for updates to this document.

1.3.4 Authentication

Whenever possible, use the authentication facilities of the install environment to verify the identity of a user that requests access to AutoVue.

2

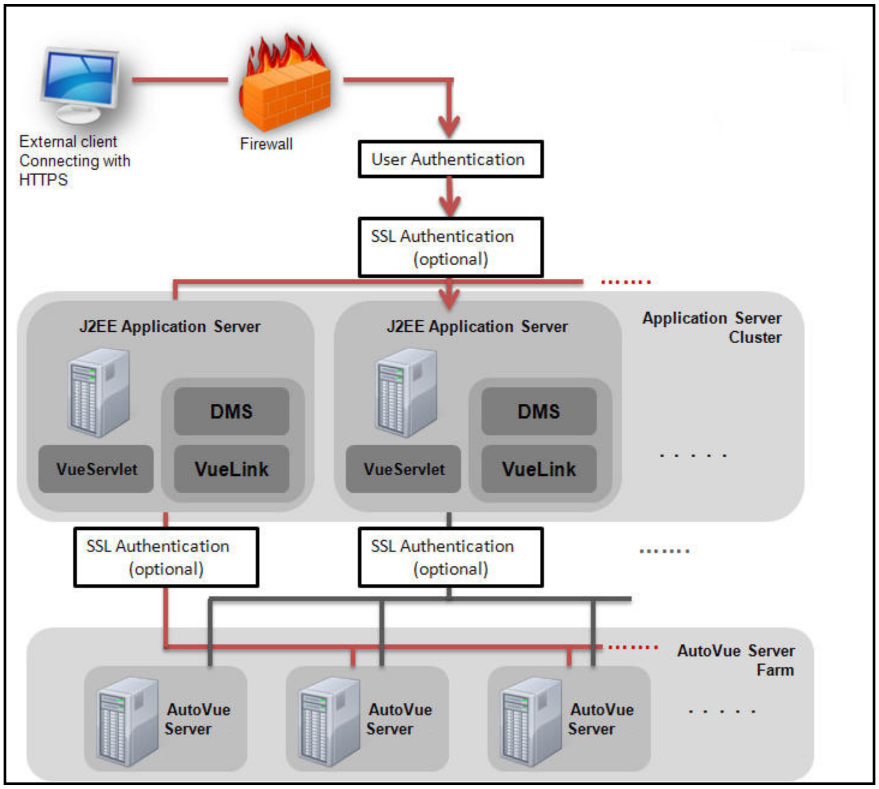
Determining Your Security Needs

Before deploying AutoVue, you must determine your security needs and make sure that you take the appropriate security measures.

2.1 Security Architecture of Oracle AutoVue

Figure 2–1, "Security Architecture of Oracle AutoVue" illustrates the elements of the Oracle AutoVue security architecture.

Figure 2–1 Security Architecture of Oracle AutoVue



2.1.1 User Authentication

A user authentication facility has been added between the client and the AutoVue server to allow integrators to connect AutoVue to Identity Management Systems. There are two valid authentication mechanisms: a DMS authentication and an authentication plug-in (for example, Kerberos)

> **Note:** If AutoVue is installed as a stand-alone server (that is, without an authentication plug-in or DMS authentication), then users cannot connect to the server. In this scenario, in jvueserver.properites, the system administrator must set jvueserver.authentication.enable to FALSE so that users can connect. Oracle recommends to prohibit all unauthenticated connections to the AutoVue server.

2.1.1.1 DMS Authentication

DMS authentication is implemented when AutoVue is integrated with a DMS backend system. A VueLink typically authenticates users through a session cookie, or by using a username/password prompt. Once the VueLink has authenticated the user, it returns the username to AutoVue in the GetProperty Action for CSI_UserName.

2.1.1.2 Authentication Plug-In

The implementation of the authentication mechanism (for example, Kerberos) makes use of a plug-in on the AutoVue client and another plug-in on the AutoVue server. The client uses its plug-in to obtain user credentials as part of the process of connecting to the server. The client encrypts the credentials and sends them to the server which uses its plug-in to authenticate the user who is trying to connect. If the server does not recognize the credentials, it refuses the connection.

A pair of these plug-ins are supplied with AutoVue: the *UsernamePasswordObtainer* class and the JAASAuthenticator class. The *UsernamePasswordObtainer* class is supplied so that the client can prompt the user for login information (username and password). The *JAASAuthenticator* class is supplied so that the server can use the Java Authentication and Authorization Service to authenticate using the authentication mechanisms specified in the configuration text file, jaas_authen.conf. The default version of this file is configured to authenticate using the Kerberos protocol which is supported by Windows Active Directory and many other standard identity repository solutions.

> **Note:** If you do not select the Default installation option, user authentication between the AutoVue client and server can be configured by following the procedure provided in Section 2.1.1.3, "Configuring Server to use JAAS Authentication Plug-In."
>
> Perform the procedure provided in Section 2.1.1.3, "Configuring Server to use JAAS Authentication Plug-In" only if you choose the **Configure Later** option when specifying the authentication mechanism between the AutoVue server and the client during AutoVue installation.

2.1.1.3 Configuring Server to use JAAS Authentication Plug-In

To configure the server to use the JAAS authentication plug-in supplied with AutoVue, perform the following:

1. Edit jvueserver.properties to specify the plug-in by removing the comment in the following line:

   ```
   jvueserver.authenticator=com.cimmetry.jvueserver.JAASAuthenticator
   ```

2. Create a text file called jaas_authen.conf in the <AutoVue install root>\bin directory. Add the following text in the file:

```
/**
** Example JAAS Login Configuration for the AutoVue server
**/
AVServer
{
com.sun.security.auth.module.Krb5LoginModule required storeKey=true;
};
```

3. Edit jvueserver.properties and add the following highlighted lines after the -Djava.security.policy parameter of jvueserver.cmdline:

```
jvueserver.cmdline=-Xmx128M -
Djava.security.policy="C:\Oracle\AutoVue\bin\policy"
-Djava.security.krb5.realm=<realm> -Djava.security.krb5.kdc=<kdc>
-Djava.security.auth.login.config=<full path to jaas_authen.conf>
```

Replace *<realm>* with your security realm.

Replace *<kdc>* with your key distribution center.

4. Startup the AutoVue server.
5. Launch the AutoVue client.

An authentication dialog appears and prompts for login information. On logging in successfully, the AutoVue client launches.

2.1.2 Enabling SSL Communication

Secure Socket Layer (SSL) is an industry-standard protocol for securing network connections. It is recommended that all communication between the AutoVue client and the VueServlet and also between the VueServlet and the AutoVue server is encrypted.

2.1.2.1 SSL Between the AutoVue Client and the VueServlet

The VueServlet component is implemented as a standard Java servlet which executes within the context of an application or servlet engine. The application engine handles the communications configuration for all servlets and applications, including the provision of secure socket layer services. Secure sockets are implemented through the use of signed digital certificates and a secure handshaking procedure. Although the details of importing a digital certificate into an application server is implementation-specific, the basic process is described in the following steps.

In order to enable SSL between AutoVue client and the VueServlet, you must ensure that SSL is enabled for the application server and that you have a CA-issued certificate installed with your application server/Web server. For example, for the WebLogic application server. The certificate is configured with two keystores:

- *DemoIdentity.jks:* Contains a demonstration private key for WebLogic Server. This keystore contains the identity for WebLogic Server.
- *DemoTrust.jks:* Contains the trusted certificate authorities from the WL_ HOME\server\lib\DemoTrust.jks and the JDK cacerts keystores. This keystore establishes trust for WebLogic Server.

If a recognized commercial certificate authority (CA) has provided the SSL certificate installed on the application server, then no additional configuration is required to enable the link between AutoVue client and VueServlet.

In some environments (usually non-production testing or development situations), self-signed certificates or certificates generated by a local certificate authority may be used. In these situations, the certificate will have to be available in the keystore of the JRE that is used to run

the AutoVue client, or a truststore that is provided in the JRE runtime parameters. Documentation related to creating certificates, keystores, and truststores is available on Oracle's Java documentation site at
https://docs.oracle.com/javase/8/docs/technotes/tools/unix/keytool.html

2.1.2.2 SSL Between the VueServlet and the AutoVue Server

Perform the following steps to enable SSL between the VueServlet and the AutoVue server.

1. In the web.xml descriptor file for the VueServlet, add the following init-param:

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

2. Make the following modification to the AutoVue server's jvueserver.properties file:

   ```
   jvueserver.ssl.enable=true
   ```
3. Set the following in jvueserver.cmdline entry in jvueserver.properties:

   ```
   -Djavax.net.ssl.keyStore=<path to the keystore>
   -Djavax.net.ssl.keyStorePassword=<password for the keystore>
   ```

If the AutoVue Server was installed with a recognized certificate authority signed certificate, then no further configuration is required.

If a self-signed, or local certificate authority based certificate was used, then the AutoVue Servers' certificate will need to be installed into the truststore of the application server that is running the VueLink.

4. Use the instructions in the next section to export the AutoVue Servers' certificate.
5. Copy the certificate to the machine running the application server and import the certificate into the application server's truststore.
6. Restart the AutoVue server and the application server hosting the VueServlet.

SSL is now configured between the VueServlet and the AutoVue server.

2.1.2.3 Exporting AutoVue Certificate

When installing AutoVue with a self-signed certificate, the cacerts that needs to be updated is the one associated with the jvueserver.jre specified in the jvueserver.properties file. For example, if the setting is:

jvueserver.jre=C:\Progra~1\Java\jre1.8.0_441\bin\javaw.exe

Then the related cacerts is in C:\Program Files\Java\jre1.8.0_441\lib\security\cacerts

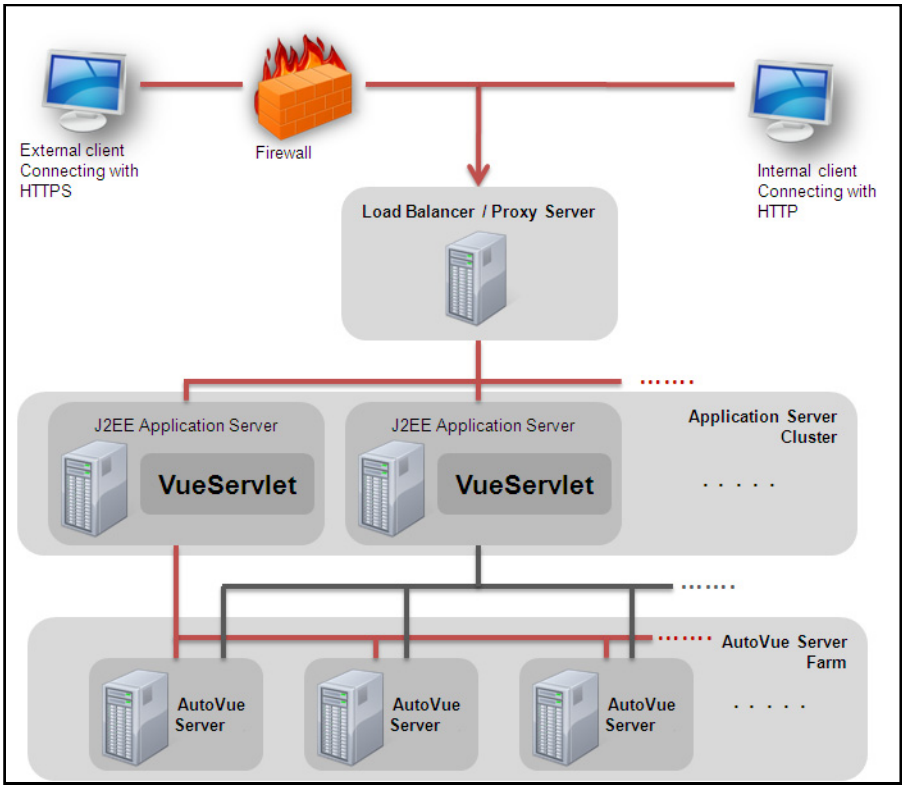
The AutoVue certificate can now be imported to other trust stores.

2.2 Recommended Deployment Topologies

The section describes standard architectures for deploying Oracle AutoVue to secure internet access. For a more complete discussion of AutoVue deployment architectures, refer to the *Oracle AutoVue, Client/Server Deployment Planning Guide.*

Figure 2–2, "Deployment Architecture - Standalone" illustrates the most basic deployment architecture. This standalone/non-integrated deployment provides failover as the AutoVue servers are setup in a server farm.

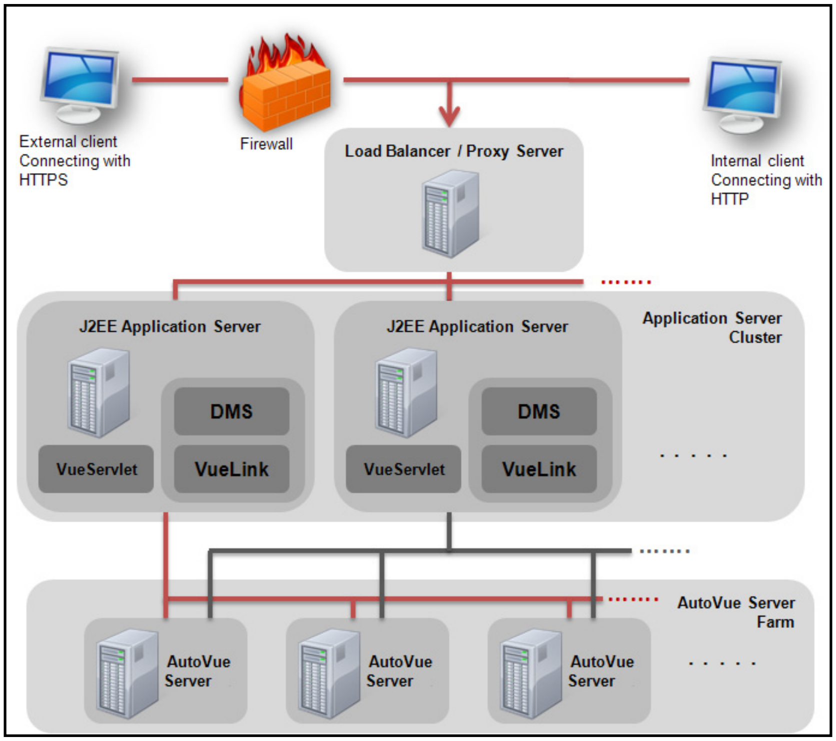
Figure 2–2 Deployment Architecture - Standalone



Note: All servers in the server farm must set the same RMI host (jvueserver.rmi.host.*) for each server in the farm. For example, if a new server is added to an existing server farm without updating the jvueserver.properties file for each server in the farm, then the new server will not be able to connect to the farm. If this happens, a security warning is logged in the server logs.

Another standard deployment architecture is AutoVue with a content repository integration as shown in the Figure 2–3, "Deployment Architecture - Content Repository Integration". This deployment architecture also provides failover for the AutoVue servers.

Figure 2–3 Deployment Architecture - Content Repository Integration



2.3 Clustered Deployments

The DMS_PRESERVE_COOKIES parameter is used when AutoVue is deployed in a clustered environment or when an integration with AutoVue relies on setting cookies and having the client pass them back as part of the request.

In AutoVue version 20.0, the DMS_PRESERVE_COOKIES parameter was updated to allow integrators to specify the exact list of cookies that the AutoVue client should pass on to the AutoVue server and/or the integration/VueLink servlet. If an AutoVue installation is being used in an integrated environment, Oracle recommends that you make use of this enhancement to restrict the use of cookies that are necessary for your deployment/integration to work.

2.4 VueServlet

Oracle recommends that the ServerInfo parameter of the VueServlet is set to FALSE. When set to TRUE, the server IP address is displayed if a user accesses the VueServlet page. The default value for this parameter is FALSE.

You can configure secure socket connection between the VueServlet and the AutoVue server. Refer to section Section 2.1.2, "Enabling SSL Communication" for information on how to set the EnableSSL parameter for the VueServlet.

2.5 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), the following points should be considered to enhance the system's overall security:

- Ensure that the original URL to a file does not contain sensitive information such as user or server information. Setting sensitive information in URLs is a potential security risk since URLs may be accessed by other users.
- The DMSARGS parameter should not contain a Session ID or other session or user-sensitive information. It is recommended to use cookies for Session ID. Additionally, integrations should use the CSI_UserName to query the user name instead of passing the username through DMSARGS.
- Pass user information through the CSI_UserName property that is queried by the AutoVue server at the beginning of a session. Ensure that you follow a consistent approach to passing user information.

> **Note:** For AutoVue integrations, save the username in the HTTP session on the application server and return it using the GetProperty Action for CSI_UserName.

- Always ensure that you pass valid user information to AutoVue. Handle incorrect authentication properly and enforce encryption of sensitive information.
- Security enhancements are added to AutoVue on an ongoing basis. The ISDK for AutoVue also leverages these security enhancements. Integrators are encouraged to upgrade their integrations to use the latest AutoVue and ISDK to benefit from these enhancements.

3

Secure Installation and Configuration

This chapter describes the secure installation and configuration steps of the AutoVue server.

3.1 Installation Overview

Oracle recommends that the AutoVue server is run as an unprivileged named user to ensure that direct access to the server and files on the server is restricted. Users connecting to the AutoVue server through the client can still view files and generate streaming files.

All the components included in the AutoVue deployment should be installed in a secure manner. The following sections cover the following steps:

- Installing the AutoVue Server
- Deploying the VueServlet
- Running the AutoVue Server as a Service

When AutoVue is installed as a component to an existing application, the security requirements of the application must be applied to AutoVue. Take note of the following recommendations when security measures should be applied:

- Use the HTTPS protocol when accessing the VueServlet from the AutoVue client.

  ```
  For Example: https://<AutoVue server hostname>:8443/servlet/VueServlet
  ```

- Use SSL communication between the VueServlet and AutoVue Server. For more information, refer to Section 2.1.2, "Enabling SSL Communication."
- Use the HTTPS protocol between the AutoVue server and Oracle VueLink.

This section discusses security considerations when installing the AutoVue server and its components.

3.2 Installing the AutoVue Server

By default, the AutoVue installer provides a secure installation of the AutoVue server. That is, only essential AutoVue features are installed. For complete instructions on installing AutoVue, refer to the "Installing AutoVue" section of the *Oracle AutoVue Client/Server Deployment Installation and Configuration Guide*.

3.3 Deploying the VueServlet

The VueServlet should be deployed on a secure installation of WebLogic. For information on deploying VueServlet, refer to the *Oracle AutoVue Client/Server Deployment Installation and Configuration Guide*. Refer to WebLogic documentation for more information on secure installations. If you are using an application server other than WebLogic, please refer to its respective security guidelines.

3.4 Running the AutoVue Server as a Service

When running the AutoVue server as a service on either Windows or Linux operating systems, it is recommended that you run it as a named user and not as Local System Account as the local system account has more privileges than a named account. For more information on running the AutoVue server as service, refer to the "Running the AutoVue Server as a Service" section of the *Oracle AutoVue Client/Server Deployment Installation and Configuration Guide*.

4

Java Web Start Client Deployment

This chapter provides details of the security features that are new to the Java Web Start deployment of the AutoVue Client.

4.1 Client Overview

Prior to 21.0.1 releases of AutoVue Client-Server, the AutoVue client was implemented using Java Applet technology. This technology allowed viewing windows to be embedded inside HTML documents which were displayed by web browser applications. The Java Applet technology depends on the Java Plug-In and a browser integration API known as Netscape Plugin Application Programming Interface (NPAPI). Due to various security and technical issues related to the NPAPI interface, web browsers are in the process of deprecating and removing its support.

From AutoVue 21.0.1, an alternative deployment option was offered in the form of a Java Web Start implementation. Java Web Start is a technology that builds on the file association facilities of browsers through the use of Java Native Launch Protocol (JNLP) files. JNLP files contain the specifications of the applications runtime requirements, its code location, and execution parameters. Based on these specifications, the Java Web Start launcher downloads the necessary resources and launches the application as a separate process on the users' machine.

4.2 Security and the Launch Process

Java Web Start is a looser integration of web browser and Java technology than the prior Applet technology. It implements a "launch and forget" strategy; once the Java Web Start Launcher application has been invoked with the JNLP file there is no relationship between the browser and the launched application. A useful capability in the AutoVue Client applet was the ability of the embedding web application to affect the viewers' behavior - selecting files to view, automatically switching to specific modes, etc. Providing similar functionality required implementing a new communication channel for the browser.

AutoVue provides two solutions that a custom HTML client can choose between to achieve this control channel. AutoVue supports a loopback connection where its client starts an embedded JSON-RPC server opening a socket on the loopback network and listens for commands from the browser. In addition, AutoVue supports a Rendezvous communication protocol where the HTML client and AutoVue client exchange messages through a Rendezvous servlet deployed on server side. The sequence of operations required from initiating a viewing session to establishing the connection between the users' browser and the AutoVue client are designed to allow the operation to be as secure as the applet implementation.

A loopback connection offers a higher level of security since it completely prevents the traffic between the client and the server. The loopback connection also requires more configuration under SSL environment as described in the section *Integrating in an SSL Environment*.

Since the loopback connection requires more configuration in SSL mode, it was deprecated in the AutoVue Release 21.1.0. It will be removed in future release. Use the Rendezvous approach that requires much less configuration under SSL mode.

JNLP files are handled by web browsers the same as any other file. Most browsers will store them in a temporary directory. After the download is complete, the browser looks up the application associated with the MIME type and launches it with the file as a command line parameter. The downloaded files are not persistent and are automatically deleted by Java Web Start as soon at they are loaded by the launcher. Based on experience from the applet implementation, browser cookies were the most sensitive pieces of information. An additional level of security is provided in the case of the loopback connection to secure more cookie information under this protocol. The launch protocol for AutoVue provides a public key facility that allows cookies to be passed to the client in an encrypted form. In case of the loopback connection protocol, the private encryption key can be safely passed to AutoVue through the loopback communication channel. In this case, the launch sequence proceeds as follows:

- The browser creates/obtains a cryptographic key pair.
- The browser invokes a servlet that generates the JNLP required to launch the AutoVue client, passing it the public key from the key pair.
- The servlet builds the JNLP file, where administrator selected cookies are encrypted with the public key.
- The downloaded JNLP file is launched by the Java Web Start launcher.
- The AutoVue client opens its JSON-RPC socket.
- The browser delivers the private key to the AutoVue client, which uses it to decrypt the cookie data that was included in the JNLP parameters.

Two servlets are provided with AutoVue to provide a reference example of this launch process implementation. The VueKeyPairServlet uses the standard Java runtime library to generate a 2048 bit RSA key pair. Its results are returned to the client in the form of a Javascript function that integrates with the provided launch code. For good security, this solution should use secure links to keep the information private. VueJNLPServlet provides the implementation of a JNLP generator. It takes a template JNLP file and customizes it with the codebase and cookie information based on the server configuration and client information. The servlet configuration allows administrators to select which cookies will be delivered to the AutoVue client.

One additional feature in the web browser to AutoVue client connection is the restriction of the JSON-RPC socket through a dynamically generated "ticket". Once a browser has connected to the client, the connection will be dedicated to the browser/client pair.

In the case of Rendezvous communication, the ticket is also used as a "Rendezvous ID" to match the browser/client pair. The Rendezvous session is also authenticated using the authentication cookies passed within the JNLP file. A mailbox stored within the authenticated session is dedicated to the communication between the browser/client pair.

The JavaScript layer (autovue.js) provided with AutoVue supports both protocols (loopback connection and rendezvous communication). The HTML sample shipped with AutoVue provides both Rendezvous servlet host for the Rendezvous communication and a list of client port ranges for the loopback connection. The JavaScript layer will automatically use the loopback connection under HTTP connection to optimize the security level and use the Rendezvous communication under HTTPS connection to optimize the usability of the solution and prevent the client configuration required under SSL environment, as mentioned in the next section.

4.3 Integrating in an SSL Environment

As mentioned previously, the LiveConnect interface that was used by web browsers to control the AutoVue client applet has been replaced by a local JSON-RPC server in the AutoVue Client application. The browser passes commands to AutoVue through JSON encoded function calls passed through the XMLHttpRequest API in Javascript.

When the browser displays secure pages (i.e. retrieved from https:// URLs), it activates an additional security policy. This policy reviews the sources of content for sub-content to detect "mixed content". Some tags retrieve content that is display only, which is considered "passive" and can be tolerated. For more powerful tags, the security risk is higher and browsers will block their execution. This includes XMLHttpRequest requests. In order for the browser to connect to AutoVue Client channel to function with a loopback connection, when the launch page is served over HTTPS, AutoVue's JSON-RPC server must also open a secure interface.

Having AutoVue open a secure interface requires that the client be able to provide a server certificate, which provides the servers identification information as well as a key pair which is used to encrypt data. In the AutoVue Client use case, the security aspects of the secure connection are unimportant as all the traffic that will be sent on the link is on the loopback network, which normally should not be exposed externally to the users' machine. This allows the implementation to use a self-signed certificate for "localhost". This certificate is normally flagged as suspect by browsers, but can be accepted by users as a "certificate exception". Adding this exception allows the mixed active content restriction to be avoided.

Note that the self-signed localhost certificate offers limited possibility of misuse, even for intentional attackers. Common browsers will flag the certificate as non-trustable in the address bar by default, and attempts to impersonate another server would need the users' machine to be improperly configured, or having the attacking application run on the users' system.

In the case of Rendezvous communication, the command messages go to the server through the existing secured connection. No localhost SSL certificate is required in this case.

4.3.1 Setup for SSL

If a deployment will be using a loopback connection approach and running with HTTPS based pages, the administrator should look at performing the following setup steps before users actively use the system:

- Generate a local self-signed certificate for their user community. This can be done with the MakeAvCert utility provided with AutoVue, or an administrator may use their own tools to generate the certificate (OpenSSL)
- Deploy the localhost certificate as an exception to their users' machines.
- Install the localhost certificate on their server site and configure their JNLP template with the URL that will allow AutoVue Clients to retrieve the certificate when needed.

A Feedback

If you have any questions or require support for AutoVue, please contact your system administrator. If the administrator is unable to resolve your issue, please contact us using the links below.

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/index.html`

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`