Overview

Java Remote Method Invocation (Java RMI), lets you, the developer, to create
distributed applications in Java.

RMI allows an object to invoke methods of remote Java objects running on another
Java Virtual Machine (JVM), possibly on different hosts. RMI uses object serialization
to marshal and unmarshal parameters, and doesn't truncate types, supporting true
object-oriented polymorphism.

RMI Security Recommendations

Follow these recommendations to improve the security of your RMI applications:

- Follow Secure Coding Guidelines for Java SE.
- Ensure that you run a security manager when using RMI, either on a client or
  server. See The Security Manager.
- Establish a reasonable security policy. For example, grant SocketPermission and
  allow listen, accept, connect, and resolve actions only among hosts
  communicating with RMI. Don't have the security policy grant AllPermission. See
  Permissions in the Java Development Kit and Default Policy Implementation and
  Policy File Syntax.
- Restrict the communication to be local if RMI is being used only for communication
  among JVMs on the local host. To accomplish this task, specify the appropriate
  socket permissions in the security policy file. Alternatively, you can use RMI APIs
directly to restrict connections only to the local host. See the RMISocketFactory
  class.
- Ensure that the value of the java.rmi.server.useCodebaseOnly property is True. By
default, the java.rmi.server.useCodebaseOnly property is set to True. If you set this
property to False, then remote code loading is enabled, which increases the level
of security risk to the system.
- Run RMI over Secure Sockets Layer (SSL)/Transport Layer Security (TLS) and
  request authentication for both server and client. This is possible using custom
socket factories. An application can export a remote object to use custom socket factories that create sockets of a desired type (for example, SSL sockets). Using this technique, an application can use SSL socket communication instead of the default socket communication. See the following:

- `SslRMIClientSocketFactory` class
- `SslRMIServerSocketFactory` class
- Java Secure Socket Extension (JSSE) Reference

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 that have purchased support 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.