**RMJavaSDK User guide**

In order to use RMJavaSDK, you must configure your Java Runtime Environment (on the machine running RMJavaSDK) to support 256-bit keys. Due to import limits on cryptographic algorithms in some countries, the Oracle implementation of the JDK/JRE provides a default cryptographic jurisdiction policy file that limits the strength of cryptographic algorithms to a 128-bit key size, as detailed on the Oracle website on [this web link](http://docs.oracle.com/javase/7/docs/technotes/guides/security/SunProviders.html).

*Note:* As the SDK consumer, it is your responsibility to verify that you are permitted to use a 256-bit key in accordance with local regulations. Install the [Java Cryptography Extensions (JCE) Unlimited Strength JurisdictionPolicy Files](http://www.oracle.com/technetwork/java/javase/downloads/index.html) corresponding to the version of JDK/JRE you have for your machine running RMJavaSDK.

The minimum sourceCompatibility for the consumer project is Java 1.7. It can be deployed on that and later versions of Java.

If you are deploying on Java 7, there is a bug about Diffie-Hellman prime length restriction <http://bugs.java.com/bugdatabase/view_bug.do?bug_id=6521495> (Skip this if you are using Java 8). Please follow the instructions in this answer to solve the problem: <https://stackoverflow.com/questions/6851461/java-why-does-ssl-handshake-give-could-not-generate-dh-keypair-exception/29176597#29176597> (but use the version of the 2 bouncy castle jars from lib folder instead of jars in the link)

**Steps for using RMJavaSDK**:

1. Register an app in RMS.

Use AccessKeyMgmt.sh (AccessKeyMgmt.bat in Windows command prompt) file to register an app in RMS, you can skip this step if you have already registered an app previously. In order to run the AccessKeyMgmt tool, make sure RMS\_INSTALL\_DIR is set in your environment variable and have tomcat and jre presented in RMS\_INSTALL\_DIR. AccessKeyMgmt.sh --help shows you how to use this tool.

1. Write your program to consume RMJavaSDK.

In your program, you need to use appId and appKey (obtained from step 1) together with routerURL to construct a RightsManager object. With this RightsManager object, you’ll be able to encrypt/decrypt files.

1. Encrypt/decrypt files.

With the encrypt and decrypt methods of RightsManager, you can enrypt/decrypt files. Refer to the sample codes we provided. Note that the tenantName parameter is optional, if tenantName is null, the default public tenant will be used internally.

1. Run your code.

All the libraries required to run the RMJavaSDK are located in the lib folder. Assuming your program (Runner.java for example) is in the same directory with the lib folder, then you can compile and run the program with following steps:

* 1. javac –cp ;lib\\* Runner.java
  2. java –cp ;lib\\* Runner

1. Javadocs and references.

Refer to sample code classes in the zipped file to see how to use the SDK. The SDK has thorough Javadocs also. It is highly recommended to read the Javadocs if there is any uncertainty about usage of the SDK.