(/)

How to Import a .cer Certificate Into a Java KeyStore

Last modified: October 22, 2020

by Donato Rimenti (https://www.baeldung.com/author/donato-rimenti/)

Java (https://www.baeldung.com/category/java/) +
Security (https://www.baeldung.com/category/security-2/)

1. Overview

A KeyStore, as the name suggests, is basically a repository of certificates, public and private keys. Moreover, **JDK distributions are shipped with an executable to help manage them, the *keytool* (/keytool-intro).**

On the other hand, certificates can have many extensions, but we need to keep in mind that a **.cer file contains public X.509 keys and thus it can be used only for identity verification.**

In this short article, we'll take a look at how to import a *.cer* file into a Java KeyStore.

2. Importing a Certificate

Without further ado, let's now import the Baeldung public certificate file inside a sample KeyStore.

The *keytool* has many options but the one we're interested in is *importcert* which is as straightforward as its name. Since there are usually different entries inside a KeyStore, we'll have to use the *alias* argument to assign it a unique name:

```
> keytool -importcert -alias baeldung_public_cert -file baeldung.cer -keystore sample_keystore
> Enter keystore password:
...
> Trust this certificate? [no]:  y
> Certificate was added to keystore
```

Although the command prompts for a password and a confirmation, we can bypass them by adding the *storepass* and *noprompt* arguments. This comes especially handy when running *keytool* from a script:

```
> keytool -importcert -alias baeldung_public_cert -file baeldung.cer -keystore sample_keystore -storepass pass123 -noprompt
> Certificate was added to keystore
```

Furthermore, if the KeyStore doesn't exist, it'll be automatically generated. In this case, **we can set the format through the *storetype* argument. If not specified, the KeyStore format defaults to *JKS* if we're using Java 8 or older. From Java 9 on it defaults to PKCS12**:

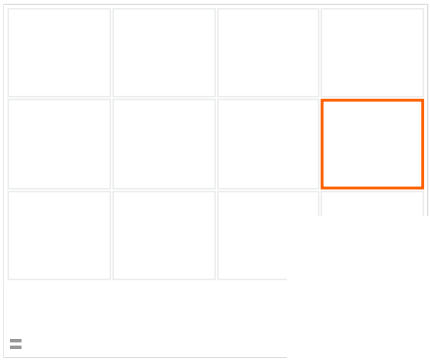
```
> keytool -importcert -alias baeldung_public_cert -file baeldung.cer -keystore sample_keystore -storetype PKCS12
> Enter keystore password:
> Re-enter new password:
...
> Trust this certificate? [no]: y
> Certificate was added to keystore
```

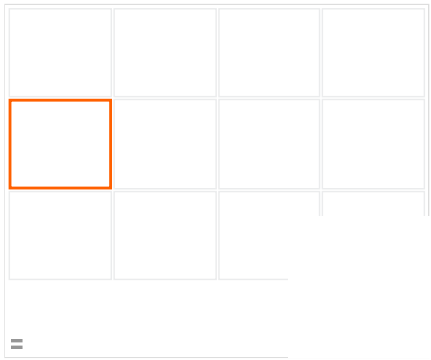
Here we've created a PKCS12 KeyStore. The main difference between JKS and PKCS12 is that JKS is a Java-specific format, while PKCS12 is a standardized way of storing keys and certificates

In case we need, we can also perform these operations programmatically (/java-keystore).

3. Conclusion

In this tutorial, we went through how to import a .cer file inside a KeyStore. In order to do that, we used the *keytool's importcert* option.





CATEGORIES

SPRING (HTTPS://WWW.BAELDUNG.COM/CATEGORY/SPRING/)
REST (HTTPS://WWW.BAELDUNG.COM/CATEGORY/REST/)
JAVA (HTTPS://WWW.BAELDUNG.COM/CATEGORY/JAVA/)

SECURITY (HTTPS://WWW.BAELDUNG.COM/CATEGORY/SECURITY-2/)
PERSISTENCE (HTTPS://WWW.BAELDUNG.COM/CATEGORY/PERSISTENCE/)
JACKSON (HTTPS://WWW.BAELDUNG.COM/CATEGORY/JSON/JACKSON/)
HTTP CLIENT-SIDE (HTTPS://WWW.BAELDUNG.COM/CATEGORY/HTTP/)
KOTLIN (HTTPS://WWW.BAELDUNG.COM/CATEGORY/KOTLIN/)

SERIES

JAVA "BACK TO BASICS" TUTORIAL (/JAVA-TUTORIAL)
JACKSON JSON TUTORIAL (/JACKSON)
HTTPCLIENT 4 TUTORIAL (/HTTPCLIENT-GUIDE)
REST WITH SPRING TUTORIAL (/REST-WITH-SPRING-SERIES)
SPRING PERSISTENCE TUTORIAL (/PERSISTENCE-WITH-SPRING-SERIES)
SECURITY WITH SPRING (/SECURITY-SPRING)

ABOUT

ABOUT BAELDUNG (/ABOUT)
THE COURSES (HTTPS://COURSES.BAELDUNG.COM)
JOBS (/TAG/ACTIVE-JOB/)
THE FULL ARCHIVE (/FULL_ARCHIVE)
EDITORS (/EDITORS)
OUR PARTNERS (/PARTNERS)
ADVERTISE ON BAELDUNG (/ADVERTISE)

TERMS OF SERVICE (/TERMS-OF-SERVICE)
PRIVACY POLICY (/PRIVACY-POLICY)
COMPANY INFO (/BAELDUNG-COMPANY-INFO)
CONTACT (/CONTACT)