OpenJDK 11

Release notes for OpenJDK 11.0.12
Abstract

This document provides an overview of new features in OpenJDK 11, as well as a list of potential known issues and possible workarounds.
# Table of Contents

PREFACE ................................................................. 3  
MAKING OPEN SOURCE MORE INCLUSIVE ........................................ 4  
CHAPTER 1. SUPPORT POLICY FOR OPENJDK ........................................ 5  
CHAPTER 2. DIFFERENCES FROM UPSTREAM OPENJDK 11 ........................................ 6  
CHAPTER 3. OPENJDK FEATURES .................................................. 7  
  3.1. NEW FEATURES AND ENHANCEMENTS ........................................ 7  
    3.1.1. Added customize PKCS12 keystore generation ................................ 7  
    3.1.2. Removed root certificates with 1024-bit keys ................................ 7  
    3.1.3. Removed Telia company’s Sonera Class2 CA certificate ......................... 8  
    3.1.4. Upgraded the default PKCS12 encryption and MAC algorithms ............... 8  
    3.1.5. Improved encoding of TLS Application-Layer Protocol Negotiation (ALPN) values ..... 8  
    3.1.6. Added support for certificate_authorities extension ........................... 8  
CHAPTER 4. ADVISORIES RELATED TO THIS RELEASE .......... 10
OpenJDK (Open Java Development Kit) is a free and open source implementation of the Java Platform, Standard Edition (Java SE). The Red Hat build of OpenJDK is available in two versions, OpenJDK 8u and OpenJDK 11u.

Packages for the Red Hat build of OpenJDK are made available on Red Hat Enterprise Linux and Microsoft Windows and shipped as a JDK and JRE in the Red Hat Ecosystem Catalog.
MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.
CHAPTER 1. SUPPORT POLICY FOR OPENJDK

Red Hat will support select major versions of OpenJDK in its products. For consistency, these versions will be the same ones that Oracle designates 'LTS' for the Oracle JDK.

A major version of OpenJDK will be supported for a minimum of six years from the time it is first introduced.

OpenJDK 11 is supported on Microsoft Windows and Red Hat Enterprise Linux until October 2024.

NOTE

RHEL 6 has reached the end of life in November 2020. Due to this, OpenJDK is not supporting RHEL 6 as a supporting configuration.

For more information, see the OpenJDK Life Cycle and Support Policy.
CHAPTER 2. DIFFERENCES FROM UPSTREAM OPENJDK 11

OpenJDK in Red Hat Enterprise Linux contains a number of structural changes from the upstream distribution of OpenJDK. The Windows version of OpenJDK tries to follow Red Hat Enterprise Linux as closely as possible.

The most notable changes are the following:

- On Red Hat Enterprise Linux, we dynamically link against native libraries such as zlib for archive format support and libjpeg-turbo, libpng, and giflib for image support. Likewise, we dynamically link against HarfBuzz and Freetype for font rendering and management.

- On Red Hat Enterprise Linux, system-wide timezone data files are used as a source for timezone information. On Microsoft Windows, the latest available timezone data from Red Hat Enterprise Linux is included.

- On Red Hat Enterprise Linux, system-wide CA certificates are used. On Microsoft Windows, the latest available CA certificate from Red Hat Enterprise Linux is used.

- The src.zip file includes the source for all of the JAR libraries shipped with OpenJDK.
CHAPTER 3. OPENJDK FEATURES

3.1. NEW FEATURES AND ENHANCEMENTS

This section describes the new features introduced in this release. It also contains information about changes in the existing features.

NOTE
For all the other changes and security fixes, see https://mail.openjdk.java.net/pipermail/jdk-updates-dev/2021-July/006954.html.

3.1.1. Added customize PKCS12 keystore generation

Added new system and security properties for enabling users to customize the generation of PKCS #12 keystores. This includes algorithms and parameters for key protection, certificate protection, and MacData. Find the detailed explanation and possible values for these properties in the “PKCS12 KeyStore properties” section of the java.security file.

Also, added support for the following SHA-2 based HmacPBE algorithms to the SunJCE provider:

- HmacPBESHA224
- HmacPBESHA256
- HmacPBESHA384
- HmacPBESHA512
- HmacPBESHA512/224
- HmacPBESHA512/256

For more information, see JDK-8215293.

3.1.2. Removed root certificates with 1024-bit keys

The following root certificates with weak 1024-bit RSA public keys have been removed from the cacerts keystore:

- Alias name: thawtepremiumserverca [jdk]
  Distinguished name: EMAILADDRESS=premium-server@thawte.com, CN=Thawte Premium Server CA, OU=Certification Services Division, O=Thawte Consulting cc, L=Cape Town, ST=Western Cape, C=ZA

- Alias name: verisignclass2g2ca [jdk]
  Distinguished name: OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized use only", OU=Class 2 Public Primary Certification Authority - G2, O="VeriSign, Inc.", C=US

- Alias name: verisignclass3ca [jdk]
  Distinguished name: OU=Class 3 Public Primary Certification Authority, O="VeriSign, Inc.", C=US

- Alias name: verisignclass3g2ca [jdk]
3.1.3. Removed Telia company’s Sonera Class2 CA certificate

The following root certificate have been removed from the cacerts truststore:

- Alias Name: soneraclass2ca
  Distinguished Name: CN=Sonera Class2 CA, O=Sonera, C=FI

For more information, see JDK-8261361.

3.1.4. Upgraded the default PKCS12 encryption and MAC algorithms

Updated default encryption and MAC algorithms used in a PKCS #12 keystore. The new algorithms based on AES-256 and SHA-256 are stronger than the old algorithms that were based on RC2, DESede, and SHA-1. See the security properties starting with keystore.pkcs12 in the java.security file for more details.

Defined a new system property named keystore.pkcs12.legacy for compatibility. It will revert the algorithms to use the older, weaker algorithms. There is no value defined for this property.

For more information, see JDK-8242069.

3.1.5. Improved encoding of TLS Application-Layer Protocol Negotiation (ALPN) values

SunJSSE providers cannot read or write certain TLS ALPN values. This is due to the choice of Strings as the API interface and the undocumented internal use of the UTF-8 character set that converts characters larger than U+00007F (7-bit ASCII) into multi-byte arrays.

ALPN values are now represented using the network byte representation expected by the peer, which should require no modification for standard 7-bit ASCII-based character Strings. However, SunJSSE now encodes/decodes string characters as 8-bit ISO_8859_1/LATIN-1 characters. his means the applications that are using characters above U+000007F encoded with UTF-8 may need to be modified to perform the UTF-8 conversion, or you can set the Java security property jdk.tls.alpnCharset to "UTF-8" to revert the behavior.

For more information, see JDK-8257548.

3.1.6. Added support for certificate_authorities extension

The certificate_authorities extension is an optional extension introduced in TLS 1.3. It indicates certificate authorities (CAs), the endpoint support and used by the receiving endpoint to guide certificate selection.

This OpenJDK release supports the certificate_authorities extension for TLS 1.3 on both the client and the server sides. This extension is always present for client certificate selection, while it is optional for server certificate selection.
Applications can enable this extension for server certificate selection by setting the `jdk.tls.client.enableCAExtension` system property to `true`. The default value of the property is `false`.

**NOTE**

If the client trusts more CAs than the size limit of the extension (less than $2^{16}$ bytes), the extension is not enabled. Also, some server implementations do not allow handshake messages to exceed $2^{14}$ bytes. Consequently, there may be interoperability issues when `jdk.tls.client.enableCAExtension` is set to `true` and the client trusts more CAs than the server implementation limit.

For more information, see JDK-8244460.
CHAPTER 4. ADVISORIES RELATED TO THIS RELEASE

The following advisories have been issued to bugfixes and CVE fixes included in this release.

- RHEA-2021:2761-04
- RHEA-2021:2762-03
- RHEA-2021:2753-02
- RHSA-2021:2784-02
- RHSA-2021:2783-02
- RHSA-2021:2782-02
- RHSA-2021:2781-02
- RHSA-2021:2780-01
- RHSA-2021:2779-01

Revised on 2021-07-21 15:07:33 UTC