Red Hat build of OpenJDK 21

使用 FIPS 在 RHEL 上配置红帽构建的 OpenJDK 21

Last Updated: 2024-05-29

Red Hat build of OpenJDK 21 使用 FIPS 在 RHEL 上配置红帽构建的 OpenJDK 21

法律通告

摘要

Red Hat build of OpenJDK 是 Red Hat Enterprise Linux 平台上的红帽产品。在带有 FIPS 的 RHEL 中配置红帽构建的 OpenJDK 21 指南提供了 FIPS 概述，并解释了如何使用 FIPS 启用和配置红帽构建的 OpenJDK。

目录

提供有关红帽构建的 OPENJDK 文档的反馈

要报告错误或改进文档，请登录到 Red Hat JIRA 帐户并提交问题。如果您没有 Red Hat Jira 帐户，则会提示您创建一个帐户。

流程

1. 单击以下链接 **以创建 ticket**。
2. 在 **Summary** 中输入问题的简短描述。
3. 在 **Description** 中提供问题或功能增强的详细描述。包括一个指向文档中问题的 URL。
4. 点 **Submit** 创建问题，并将问题路由到适当的文档团队。

使开源包含更多

红帽致力于替换我们的代码、文档和 Web 属性中存在问题的语言。我们从这四个术语开始：master、slave、黑名单和白名单。由于此项工作十分艰巨，这些更改将在即将推出的几个发行版本中逐步实施。详情请查看 CTO Chris Wright 的信息。

第 1 章 联邦信息处理标准(FIPS)简介

联邦信息处理标准(FIPS)提供了改进跨计算机系统和网络的安全与互操作性的指南和要求。FIPS 140-2 和 140-3 系列适用于硬件和软件级别的加密模块。美国国家标准与技术研究院使用可搜索处理和批准的加密模块列表实施加密模块验证计划。

Red Hat Enterprise Linux (RHEL)提供了一种集成框架，可在系统范围内启用 FIPS 140-2 合规性。当在 FIPS 模式下运行时，使用加密库的软件包会根据全局策略自行配置。大多数软件包都提供了一种更改默认对齐行为的方法，以满足兼容性或其他需求。

Red Hat build of OpenJDK 21 是一个 FIPS 策略感知型软件包。

其他资源

- 有关加密模块验证计划的更多信息，请参阅 *国家标准和技术* 网站上的加密模块 验证计划 CMVP。
- 有关如何启用 FIPS 模式安装 RHEL 的更多信息，请参阅 安装启用了 FIPS 模式的 RHEL 8 系统 。
- 有关如何在安装 RHEL 后如何启用 FIPS 模式的更多信息，请参阅 将系统切换到 FIPS 模式。
- 有关如何在 RHEL 上以 FIPS 模式运行红帽构建的 OpenJDK 的更多信息。请参阅在 RHEL 上以 FIPS 模式运行 OpenJDK。
- 有关红帽遵守政府标准的更多信息，请参阅 政府标准。

第 2 章 红帽构建的 OPENJDK 21 中的 FIPS 设置

在启动时，红帽构建的 OpenJDK 21 会检查系统 FIPS 策略是否已启用。如果启用了此策略，红帽构建的 OpenJDK 21 会执行一系列自动配置，用于帮助 Java 应用程序遵守 FIPS 要求。

这些自动配置包括以下操作：

- 为加密操作安装包含 FIPS 认证的网络安全服务(NSS)软件令牌模块的受限安全供应商列表
- 为 Java 强制 Red Hat Enterprise Linux (RHEL) FIPS 加密策略，该策略限制可用的算法和参数

注意

如果在 JVM 实例运行时在系统中启用了 FIPS 模式，则必须重启 JVM 实例，以允许更改生效。

您可以配置红帽构建的 OpenJDK 21，以绕过上述 FIPS 自动化。例如，您可能希望通过硬件安全模块(HSM)而不是 NSS 软件令牌模块实现 FIPS 合规性。

您可以使用 *system* 或 *安全属性* 指定 FIPS 配置。

要更好地了解 FIPS 属性，您必须了解以下 JDK 属性类：

- 系统属性是前缀为 **-D** 的 JVM 参数，它通常采用 **-Dproperty.name=property.value** 的形式。不需要特权访问权限来传递这些值。只有启动的 JVM 会受到配置的影响，持久性取决于存在启动程序脚本。UTF-8 编码的值对系统属性有效。
- 安全属性在 **$JRE_HOME/conf/security/java.security** 或 **java.security.properties** 系统属性指向的文件中提供。修改 **$JRE_HOME/conf/security/java.security** 文件中的值需要特权访问权限。对此文件的任何修改都会保留，并影响到同一红帽构建 OpenJDK 21 部署的所有实例。非基本拉丁 Unicode 字符必须使用 **\uXXXX** 编码。

当系统和安全属性的名称相同并且设置为不同的值时，系统属性会优先使用。根据配置，属性可能会影响具有不同名称的其他属性。

有关安全属性及其默认值的更多信息，请参阅 **java.security** 文件。

以下列表详细介绍了影响红帽构建的 OpenJDK 21 的 FIPS 配置的属性：

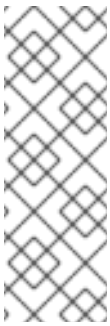
属性	类型	默认值	描述
security.useSystemPropertiesFile	安全性	**true**	当设置为 **false** 时，此属性禁用 FIPS 自动化，其中包括全局 crypto-policies 对齐。
java.security.disableSystemPropertiesFile	System	**false**	当设置为 **true** 时，此属性禁用 FIPS 自动化，其中包括全局 crypto-policies 校准。这与 **security.useSystemPropertiesFile=false** 安全属性的作用相同。如果两个属性都被设置为不同的行为，则 **java.security.disableSystemPropertiesFile** 将具有优先权。

属性	类型	默认值	描述
com.redhat.fips	System	**true**	当设置为 **false** 时，此属性在仍然强制执行 FIPS crypto-policy 时禁用 FIPS 自动化。如果将上述任何属性设置为禁用 FIPS 自动化，则此属性无效。crypto-policies 是 FIPS 自动化的先决条件。
fips.keystore.type	安全性	**PKCS12**	当红帽构建的 OpenJDK 21 处于 FIPS 模式时，此属性设置默认密钥存储类型。支持的值有 **PKCS12** 和 **PKCS11**。

除了前面描述的设置外，也可以将特定配置应用到在 FIPS 模式下使用 NSS DB 密钥存储。这些密钥存储由 **SunPKCS11** 安全提供程序和 NSS 软件令牌处理，后者是安全供应商的 **PKCS facilities** 后端。

以下列表详细介绍了红帽构建的 OpenJDK 21 的 NSS DB FIPS 属性：

属性	类型	默认值	描述
fips.nssdb.path	系统或安全性	**sql:/etc/pki/nssdb**	指向 NSS DB 位置的文件系统路径。 此属性的语法与 **SunPKCS11** NSS 配置文件中的 **nssSecmodDirectory** 属性相同。属性允许 **sql:** 前缀来指示引用的 NSS DB 是 SQLite 类型。

属性	类型	默认值	描述
fips.nssdb.pin	系统或安全性	**pin:** (empty PIN)	**fips.nssdb.path** 指向的 NSS DB 的 PIN（密码）。 您可以使用此属性以以下形式传递 NSS DB PIN： • **pin:<value>** 在这种情况下，&lt **;value&** gt; 是一个明文 PIN 值（例如：**pin:1234abc**）。 • **env:<value>** 在这种情况下，&lt **;value** > 是一个包含 PIN 值的环境变量（例如：**env:NSSDB_PIN_VAR**）。 • **file:<value>** 在这种情况下，& **lt;value** > 是 UTF-8 编码文件的路径，其中包含其第一行中的 PIN 值（例如 **file:/path/to/pin.txt**）。 **pin:<value** > 选项适合两个情况下，PIN 值作为 JVM 参数传递，或者通过系统属性以编程方式传递。PIN 值的编程设置为应用程序提供了灵活性，以决定如何获取 PIN。 **file:<value** > 选项与 NSS **modutil -pwfile** 和 **-newpwfile** 参数兼容，该参数用于 NSS DB PIN 更改。  **注意** 如果加密操作需要 NSS DB 身份验证，且状态没有被验证，Red Hat build of OpenJDK 21 会执行带有这个 PIN 值的隐式登录。应用可以通过在任何加密操作前调用 **KeyStore::load** 来执行显式登录。

重要

执行安全评估，以便您可以决定保护所存储密钥和证书的完整性和机密性的配置。此评估应考虑威胁、上下文信息和其他安全措施，如操作系统用户隔离和文件系统权限。例如，默认配置值可能不适用于存储键的应用程序，并在多用户环境中运行。使用 RHEL 中的 **modutil** 工具创建和管理 NSS DB 密钥存储，并使用 **certutil** 或 **keytool** 导入证书和密钥。

其他资源

- 有关启用 FIPS 模式的更多信息，请参阅 将系统切换到 FIPS 模式。

第 3 章 RED HAT BUILD OF OPENJDK 21 中的 FIPS 自动化

本章论述了如何在红帽构建的 OpenJDK 21 中实施 FIPS 自动化，以及 FIPS 自动化如何影响您的应用程序。

3.1. 安全供应商

启用 FIPS 模式后，红帽构建的 OpenJDK 21 将安装的安全供应商替换为受限列表。有些安全服务和算法可能会被丢弃，以便只有 FIPS 认证模块执行加密操作。以下列表描述了已安装的安全供应商、服务、算法和启用的配置：

SunPKCS11-NSS-FIPS

根据 **$JRE_HOME/conf/security/nss.fips.cfg** 处找到的配置，使用 NSS 软件令牌初始化的 NSS 软件令牌（这是服务提供商的 **PKCS** BACKEND 后端）：

- **name = NSS-FIPS**
- **nssLibraryDirectory = /usr/lib64**
- **nssSecmodDirectory = ${fips.nssdb.path}**
- **nssDbMode = readWrite**
- **nssModule = fips**
- **attributes (*,CKO_SECRET_KEY,CKK_GENERIC_SECRET)={ CKA_SIGN=true }**

注意

不鼓励更改此配置。

所有加密服务都已启用。这包括 **AlgorithmParameters**,**Cipher**,**KeyAgreement**,**KeyFactory**,**KeyGenerator**,**KeyPairGenerator**,**KeyStore**,**Mac**,**MessageDigest**,**SecretKeyFactory**,**SecureRandom**, 和 **Signature**。

SUN

仅与 X.509 证书相关(**CertificateFactory**,**CertPathBuilder**,**CertPathValidator**,**CertStore**), **AlgorithmParameterGenerator**,**AlgorithmParameters**, 和 **KeyStore** (**JKS**,**PKCS12**)服务被启用。

SunEC

仅启用 **AlgorithmParameters** 和 **KeyFactory** 服务。

SunJSSE

仅启用与 TLS 相关的服务(**KeyManagerFactory**,**SSLContext**,**TrustManagerFactory**)和 **KeyStore** (**PKCS12**)。

SunJCE

仅启用 **AlgorithmParameterGenerator**、**AlgorithmParameters**、**KeyFactory** 和 **SecretKeyFactory** （除 **BKDF2** 算法外）服务。

SunRsaSign

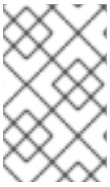
仅启用 **AlgorithmParameters** 和 **KeyFactory** 服务。

XMLDSig

所有服务都已启用。这包括 **TransformService**、**KeyInfoFactory** 和 **XMLSignatureFactory**。

3.2. CRYPTO-POLICIES

在 FIPS 模式中，红帽构建的 OpenJDK 21 会从 RHEL 中的全局 FIPS 加密策略获取禁用的加密算法和其他配置列表。您可以在 **/etc/crypto-policies/back-ends/java.config** 中找到这些值。您可以使用 RHEL 中的 **update-crypto-policies** 工具来一致管理 crypto-policies。



注意

当在 FIPS 模式下配置红帽构建的 OpenJDK 时，加密策略批准算法可能无法使用。当 NSS 软件令牌中没有 FIPS 认证的实现，或者 **SunPKCS11** 安全供应商不支持它时，会出现这种情况。

3.3. 信任 ANCHOR 证书

在 FIPS 模式中，红帽构建的 OpenJDK 21 默认使用全局信任 Anchor 证书存储库。这个行为等同于非 FIPS 模式。此软件仓库位于 **/etc/pki/java/cacerts**。使用 RHEL 中的 **update-ca-trust** 工具来一致管理证书。另外，您还可以将 Trust Anchor 证书存储在您自己的 **PKCS12** 和 **PKCS11** 密钥存储中，并使用它们进行 TLS 通信。如需更多信息，请参阅 **TrustManagerFactory::init** 文档。

当未设置 **javax.net.ssl.trustStoreType** 系统属性并且启用了 FIPS 模式时，红帽构建的 OpenJDK 21 会自动将此系统属性设置为 **keystore.type** 安全属性值。这个行为等同于非FIPS 模式。

3.4. KEYSTORES

在 FIPS 模式中，红帽构建的 OpenJDK 21 允许使用 **PKCS12** 和 **PKCS11** 密钥存储类型。默认使用 **PKCS12**。您可以使用 **fips.keystore.type** 安全属性更改默认密钥存储类型。应用程序也可以选择在调用 **KeyStore.getInstance (<type>)时要使用的** 密钥存储类型。

打开 **PKCS11** 密钥存储时，红帽构建的 OpenJDK 21 使用位于 **/etc/pki/nssdb** 的 SQLite NSS DB。这个 NSS 数据库可能不适合存储密钥。您可以通过为 **fips.nssdb.path** 属性设置值来指定不同的数据库。如需更多信息和安全注意事项，请参阅 红帽构建的 OpenJDK 21 中的 FIPS 设置 。

当您将 **fips.keystore.type** 安全属性设置为 **PKCS11** 和 FIPS 模式时，红帽构建的 OpenJDK 21 会自动将 **javax.net.ssl.keyStore** 系统属性分配给 **NONE** 的值。此行为可通过保存手动配置步骤 **来促进 PKCS** BLS 密钥存储的使用。如需更多信息，请参阅 JDK-8238264。

更新于 2024-05-29