Red Hat build of OpenJDK 17

Red Hat build of OpenJDK 17.0.5 のリリースノート

Last Updated: 2024-05-04

Red Hat build of OpenJDK 17 Red Hat build of OpenJDK 17.0.5 のリリースノート

法律上の通知

概要

Red Hat build of OpenJDK 17.0.5 ドキュメントのリリースノート には、Red Hat build of OpenJDK 17 の新機能の概要と、潜在的な既知の問題と考えられる回避策のリストが記載されています。

目次

はじめに

Open Java Development Kit (OpenJDK) は、Java Platform Standard Edition (Java SE) のオープンソース実装です。Red Hat build of OpenJDK には、8u、11u、17u の 3 つのバージョンがあります。

Red Hat build of OpenJDK 向けパッケージは、Red Hat Enterprise Linux および Microsoft Windows で利用でき、Red Hat Ecosystem Catalog の JDK および JRE として同梱されています。

RED HAT BUILD OF OPENJDK ドキュメントへのフィードバック

エラーを報告したり、ドキュメントを改善したりするには、Red Hat Jira アカウントにログインし、課題を送信してください。Red Hat Jira アカウントをお持ちでない場合は、アカウントを作成するように求められます。

手順

1. 次のリンクをクリックして **チケットを作成します**。
2. **Summary** に課題の簡単な説明を入力します。
3. **Description** に課題や機能拡張の詳細な説明を入力します。問題があるドキュメントのセクションへの URL を含めてください。
4. **Submit** をクリックすると、課題が作成され、適切なドキュメントチームに転送されます。

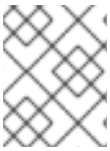
多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、今後の複数のリリースで段階的に用語の置き換えを実施して参ります。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

第1章 RED HAT BUILD OF OPENJDK のサポートポリシー

Red Hat は、Red Hat build of OpenJDK の一部のメジャーバージョンを製品でサポートします。一貫性を保つために、これらのバージョンは長期サポート (LTS) として指定されている Oracle JDK バージョンと同様のままとなります。

Red Hat build of OpenJDK のメジャーバージョンは、最初に導入された時点から少なくとも 6 年間サポートされます。詳細は、OpenJDK のライフサイクルおよびサポートポリシー を参照してください。



注記

RHEL 6 のライフサイクルは 2020 年 11 月に終了します。このため、Red Hat build of OpenJDK はサポート対象設定として RHEL 6 をサポートしません。

第2章 アップストリームの OPENJDK 17 との相違点

Red Hat Enterprise Linux の OpenJDK には、Red Hat build of OpenJDK のアップストリームディストリビューションの構造上の変更が数多く含まれています。Microsoft Windows バージョンの Red Hat build of OpenJDK は、Red Hat Enterprise Linux の更新にできる限り従います。

以下は、Red Hat build of OpenJDK 17 における最も注目すべき変更のリストです。

- FIPS のサポート。Red Hat build of OpenJDK 17 は、RHEL が FIPS モードであるかどうかを自動的に検出し、Red Hat build of OpenJDK 17 がそのモードで動作するように自動的に設定します。この変更は、Microsoft Windows 向けの Red Hat build of OpenJDK ビルドには適用されません。

- 暗号化ポリシーのサポート。Red Hat build of OpenJDK 17 は、有効な暗号化アルゴリズムとキーサイズ制約のリストを RHEL システム設定から取得します。これらの設定コンポーネントは、トランスポート層セキュリティー (TLS) 暗号化プロトコル、証明書パス検証、および署名された JAR によって使用されます。さまざまなセキュリティープロファイルを設定して、安全性と互換性のバランスをとることができます。この変更は、Microsoft Windows 向けの Red Hat build of OpenJDK ビルドには適用されません。

- RHEL の Red Hat build of OpenJDK は、アーカイブ形式のサポート用の **zlib**、イメージのサポート用の **libjpeg-turbo**、**libpng**、**giflib** などのネイティブライブラリーと動的にリンクします。また、RHEL はフォントのレンダリングと管理のために、**Harfbuzz** および **Freetype** に対して動的にリンクします。この変更は、Microsoft Windows 向けの Red Hat build of OpenJDK ビルドには適用されません。

- **src.zip** ファイルには、Red Hat build of OpenJDK に同梱されるすべての JAR ライブラリーのソースが含まれます。

- RHEL の Red Hat build of OpenJDK は、タイムゾーン情報のソースとして、システム全体のタイムゾーンデータファイルを使用します。

- RHEL の Red Hat build of OpenJDK は、システム全体の CA 証明書を使用します。

- Microsoft Windows の Red Hat build of OpenJDK には、RHEL で利用可能な最新のタイムゾーンデータが含まれています。

- Microsoft Windows の Red Hat build of OpenJDK は、RHEL から入手可能な最新の CA 証明書を使用します。

関連情報

- Improve system FIPS detection (RHEL Planning Jira) を参照してください。

- システム全体の暗号化ポリシーの使用 (RHEL ドキュメンテーション) を参照してください。

第3章 RED HAT BUILD OF OPENJDK の機能

最新の Red Hat build of OpenJDK 17 には、新機能が含まれている可能性があります。さらに、最新リリースは、以前の Red Hat build of OpenJDK 17 リリースに由来する機能を強化、非推奨、または削除する可能性があります。

注記

その他の変更点やセキュリティー修正については、OpenJDK 17.0.5 Released を参照してください。

Red Hat build of OpenJDK の機能強化

Red Hat build of OpenJDK 17 では、以前のリリースの Red Hat build of OpenJDK で作成された機能に拡張が行われました。

cpu.shares パラメーターが無効になっている

Red Hat build of OpenJDK 17.0.5 リリースより前は、Red Hat build of OpenJDK は、**cgroups** とも呼ばれる Linux コントロールグループに属する **cpu.shares** パラメーターの誤った解釈を使用していました。このパラメーターにより、Java 仮想マシン (JVM) が使用可能な CPU よりも少ない CPU を使用する可能性があり、コンテナー内で動作するときの JVM の CPU リソースとパフォーマンスに影響を与える可能性があります。

Red Hat build of OpenJDK 17.0.5 リリースでは、スレッドプールのスレッド数を決定するときに **cpu.shares** パラメーターを使用しないように JVM が設定されます。この設定を元に戻したい場合は、JVM の起動時に **-XX:+UseContainerCpuShares** 引数を渡します。

注記

-XX:+UseContainerCpuShares 引数は非推奨の機能であり、将来の Red Hat build of OpenJDK リリースで削除される可能性があります。

JDK-8281181 (JDK バグシステム) を参照してください。

SHA-1 署名 JAR

Red Hat build of OpenJDK 17.0.5 リリースでは、**SHA-1** アルゴリズムで署名された JAR はデフォルトで制限され、署名されていないかのように扱われます。これらの制限は、次のアルゴリズムに適用されます。

- ダイジェスト、署名、およびオプションで JAR のタイムスタンプに使用されるアルゴリズム。
- コード署名者とタイムスタンプ機関の証明書チェーン内の証明書の署名アルゴリズムとダイジェストアルゴリズム、およびそれらの証明書が失効しているかどうかを確認するために使用される証明書失効リスト (CRL) またはオンライン証明書ステータスプロトコル (OCSP) 応答。

さらに、制限は署名済みの Java Cryptography Extension (JCE) プロバイダーにも適用されます。

以前にタイムスタンプが付けられた JAR の互換性リスクを軽減するために、この制限は、**SHA-1** アルゴリズムで署名され、**January 01, 2019** より前にタイムスタンプが付けられた JAR には適用されません。この例外は、将来の Red Hat build of OpenJDK リリースで削除される可能性があります。

JAR ファイルが制限の影響を受けるかどうかを判断するには、CLI で次のコマンドを発行します。

```
$ jarsigner -verify -verbose -certs
```

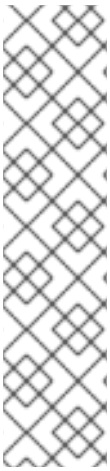
前のコマンドの出力から、**SHA1**、**SHA-1**、または **disabled** のインスタンスを検索します。さらに、JAR が署名なしとして扱われることを示す警告メッセージを検索します。以下に例を示します。

```
Signed by "CN="Signer""
Digest algorithm: SHA-1 (disabled)
Signature algorithm: SHA1withRSA (disabled), 2048-bit key

WARNING: The jar will be treated as unsigned, because it is signed with a weak algorithm that is now disabled by the security property:

jdk.jar.disabledAlgorithms=MD2, MD5, RSA keySize < 1024, DSA keySize < 1024, SHA1 denyAfter 2019-01-01
```

新しい制限の影響を受けるすべての JAR をより強力なアルゴリズムに置き換えるか、再署名することを検討してください。

JAR ファイルがこの制限の影響を受ける場合は、アルゴリズムを削除して、**SHA-256** などのより強力なアルゴリズムでファイルに再署名できます。Red Hat build of OpenJDK 17.0.5 の **SHA-1** 署名付き JAR に対する制限を削除する必要があり、セキュリティーリスクを受け入れる場合は、次のアクションを実行できます。

1. **java.security** 設定ファイルを変更します。または、このファイルを保存して、必要な設定で別のファイルを作成することもできます。
2. **SHA1 usage SignedJAR & denyAfter 2019 01 011** エントリーを **jdk.certpath.disabledAlgorithms** セキュリティープロパティーから削除します。
3. **jdk.jar.disabledAlgorithms** セキュリティープロパティーから **SHA1 denyAfter 2019-01-01** エントリーを削除します。



注記

java.security ファイルの **jdk.certpath.disabledAlgorithms** の値は、RHEL 8 および 9 のシステムセキュリティーポリシーによって上書きされる場合があります。システムセキュリティーポリシーで使用される値は、ファイル **/etc/crypto-policies/back-ends/java.config** で確認でき、java.security ファイルで **security.useSystemPropertiesFile** を false に設定するか、**-Djava.security.disableSystemPropertiesFile=true** を JVM 渡すことで無効にします。これらの値はこのリリースでは変更されていないため、値は Red Hat build of OpenJDK の以前のリリースと同じままです。

java.security ファイルの設定例については、JBoss EAP for OpenShift の **java.security** プロパティーのオーバーライド (Red Hat カスタマーポータル) を参照してください。

JDK-8269039 (JDK バグシステム) を参照してください。

SunMSCAPI プロバイダーは、新しい Microsoft Windows キーストアタイプをサポートします

SunMSCAPI プロバイダーは、ローカル名前空間を **Windows-** に追加する必要がある次の Microsoft Windows キーストアタイプをサポートしています。

- **Windows-<local_computer_name>**
- **Windows-<root_local_computer_name>**

- **Windows-<current_username>**
- **Windows-<root_username>**

これらのタイプのいずれかを指定することにより、ローカルコンピューターの Microsoft Windows キーストアの場所へのアクセスを提供できます。これにより、ローカルシステムに保存されている証明書へのキーストアアクセスが提供されます。

JDK-6782021 (JDK バグシステム) を参照してください。

HTTPURLConnection の keep-alive 動作を制御するためのシステムプロパティー

Red Hat build of OpenJDK 17.0.5 リリースには、**HTTPURLConnection** の **キープアライブ** 動作を制御するために使用できる次の新しいシステムプロパティーが含まれています。

- サーバーへの接続を制御する **http.keepAlive.time.server**。
- プロキシーへの接続を制御する **http.keepAlive.time.proxy**。

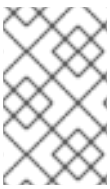
Red Hat build of OpenJDK 17.0.5 リリースより前では、**keep-alive** 時間が指定されていないサーバーまたはプロキシーが原因で、ハードコードされたデフォルト値によって定義された期間、アイドル接続が開いたままになる場合がありました。

Red Hat build of OpenJDK 17.0.5 では、システムプロパティーを使用して **keep-alive** の時間のデフォルト値を変更できます。**keep-alive** プロパティーは、サーバーまたはプロキシーのいずれかの HTTP **keep-alive** 時間を変更することでこの動作を制御します。これにより、Red Hat build of OpenJDK の HTTP プロトコルハンドラーは、指定された秒数が経過した後にアイドル状態の接続を閉じます。

Red Hat build of OpenJDK 17.0.5 リリースより前では、次のユースケースで、**HTTPURLConnection** の特定の **keep-alive** 動作が発生していました。

- サーバーが **Connection:keep-alive** ヘッダーを指定し、サーバーの応答に **Keep-alive:timeout=N** が含まれている場合、クライアントの Red Hat build of OpenJDK **keep-alive** キャッシュは **N** 秒のタイムアウトを使用します (**N** は整数値)。
- サーバーが **Connection:keep-alive** ヘッダーを指定しているが、サーバーの応答に **Keep-alive:timeout=N** のエントリーが含まれていない場合、クライアントの Red Hat build of OpenJDK **keep-alive** キャッシュはプロキシーに対して **60** 秒のタイムアウトを使用し、**5** サーバーの秒。
- サーバーが **Connection:keep-alive** ヘッダーを指定しない場合、クライアントの Red Hat build of OpenJDK **keep-alive** キャッシュは、すべての接続に対して 5 秒のタイムアウトを使用します。

Red Hat build of OpenJDK 17.0.5 リリースでは、前述の動作はそのままですが、2 番目と 3 番目に挙げた使用例におけるタイムアウトは、デフォルトの設定に依存するのではなく、**http.keepAlive.time.server** および **http.keepAlive.time.proxy** プロパティーを使用して指定できるようになっています。



注記

keep-alive プロパティーを設定し、サーバーが **Keep-Alive** 応答ヘッダーの **keep-alive** 時間を指定した場合、HTTP プロトコルハンドラーはサーバーによって指定された時間を使用します。この状況は、プロキシーと同じです。

JDK-8278067 (JDK バグシステム) を参照してください。

第4章 このリリースに関連するアドバイザリー

以下のアドバイザリーは、本リリースに含まれるバグ修正および CVE の修正に発行されています。

- RHSA-2022:6999
- RHSA-2022:7000
- RHSA-2022:7001
- RHSA-2022:7051
- RHSA-2022:7054

改訂日時: 2024-05-04