Micro Focus VisiBroker 8.5 SP7

Release Notes

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK
http://www.microfocus.com

2021-03-16

Contents

Micro Focus VisiBroker 8.5.7 Release Notes

Installing VisiBroker

Before Installing SP7

This release updates VisiBroker 8.5. Before installing this Service Pack you must have VisiBroker 8.5 installed. If you do not have VisiBroker 8.5, you should download it and follow the instructions in the *Installation Guide*.

Installing SP7

To install this release on top of an existing VisiBroker 8.5 installation:

1. Download the release archive to your VBROKERDIR folder.
2. Unpack the archive in the same folder.
3. Restart the application.

Upgrading an earlier installation to JDK 11

VisiBroker 8.5 SP7 supports Java 11. VisiBroker product installers for 8.5 SP5 and older versions could only be used with a Java JDK 8 or below. If your VisiBroker product installation was performed with an installer for 8.5 SP5 or an older version, and you wish to migrate your installation to JDK 11 or higher, there are some additional steps required after the installation described above. This process is not necessary if you are upgrading from 8.5 SP6.

Note: Micro Focus strongly recommends that you create back-ups of all files before making any modifications.

1) Locate all the VisiBroker `.config` files in use by your installation and your applications. The default file names and their locations are:

   ```
   [VBROKERDIR]/bin/toolsjdk.config
   [VBROKERDIR]/bin/vbconsole.config
   [VBROKERDIR]/bin/vbj.config
   [VBROKERDIR]/bin/vbjc.config
   ```

   If you have modified or extended your application's VisiBroker configuration you may also have additional files. Check your start-up scripts for the argument `-VBJconfig`. Note that `.config` files can also include further `.config` files with the `'include'` directive.

2) For each `.config` file:

   a) Locate and update any `'javahome'` directive lines to point to your new JDK 11 location. For example, replace:

      ```
      javahome /usr/lib64/jvm/java-1.8.0
      ```

      with:

      ```
      javahome /usr/lib64/jvm/java-11
      ```

b) Remove any lines that specify the VM property `'vmprop java.endorsed.dirs'`. For example, remove:

```
vmprop java.endorsed.dirs=$var(installRoot)/lib/endorsed
```

This JVM option is no longer supported by JDK 11 and will generate an error if not removed.

c) Remove any directives that reference `.jar` files that are no longer present in the JDK. For example, remove all entries like:

```
addpath $var(JDK_HOME)/lib/tools.jar
addpath $var(JDK_HOME)/jre/lib/rt.jar
```

d) To each `.config` file add entries like those below. These specify the location of CORBA classes that have been removed from the JDK:

```
addendorsejar $var(installRoot)/lib/endorsed/vbjendorse.jar
addomgjar $var(installRoot)/lib/visi-omg.jar
```

If you do not make these additions you may see ClassNotFoundExceptions for classes in the `org.omg.*` packages.

e) Remove any entries using the directives "`addbootjars`" or "`addbootpath`". Such entries do not normally occur in default VisiBroker configurations but you might have added them. As their corresponding JVM options `-Xbootclasspath` and `-Xbootclasspath/p` are no longer supported by JDK 11, using either of these two directives will cause a JVM Loader error.

3) If the JDK 11 you wish to use is 64-bit and VisiBroker was installed with a 32-bit JDK version then apply the appropriate **VisiBroker 64-bit Service Overlay** so that Java services and tools will work with the new JDK. To accomplish this, run the 32-bit VisiBroker installer as normal, but select your 64-bit JDK selected during the installation process. Immediately after installation unpack the appropriate **VisiBroker 64-bit Service Overlay** patch into the installation directory. Contact Micro Focus SupportLine (https://supportline.microfocus.com/) for further information.

Platforms and Compilers

For a full list of platforms and compilers supported by VisiBroker 8.5 SP7, see the ***VisiBroker 8.5 SP7 Platform Support Notes*** (https://www.microfocus.com/documentation/visibroker/visibroker857/Micro_Focus_VisiBroker_Platform_Support_Notes_857.pdf).

Note:

1. Support for SUSE Linux Enterprise Server 15 and Red Hat Enterprise Linux 8 has been added.
2. Support for Windows Server 2019 has been added.
3. Support for Visual Studio 2019 has been added to the product at this release. A single version of VisiBroker for Windows (patch filename `08.05.00.p7_opt_vs2015_x64.ojdk8.zip`, product installer filename `vb85-08.05.00.P7-winvs2015-x64.exe`) can now be used for all applications built with Visual Studio 2015, 2017, and 2019. This patch and installer are binary compatible with all existing installations of the distinct Visual Studio 2017

targeted version of VisiBroker that was produced at Service Pack 6 and which has now been withdrawn.

4. VisiBroker 8.5 for MontaVista Linux Carrier Grade Edition 6 is no longer receiving updates. Please contact Technical Support if you need updates for this platform.

New Features

VisiBroker 8.5 SP7 provides enhancements in the following areas.

- OpenSSL
- Support for RedHat Enterprise Linux 8 and SuSE Linux Enterprise Server 15
- VisiSecure supports Transport Level Security (TLS) 1.3
- Configurable OpenSSL Security Levels
- Configurable Diffie-Hellman Groups for key exchange
- New security configuration options

OpenSSL

OpenSSL v1.1.1g is now supported.

Support for RedHat Enterprise Linux 8 and SuSE Linux Enterprise Server 15

VisiBroker 8.5 SP7 is supported on RedHat Enterprise Linux 8 and SuSE Linux Enterprise Server 15.

If applications are to be re-compiled on either of these platforms using the supplied GCC 8 then you must apply the following compiler macro definition:

```
-D_GLIBCXX_USE_CXX11_ABI=0
```

Information on the macro `_GLIBCXX_USE_CXX11_ABI` can be found in the GCC documentation at:
https://gcc.gnu.org/onlinedocs/libstdc++/manual/using_dual_abi.html

Compilation on these platforms exclusively with `_GLIBCXX_USE_CXX11_ABI=1` (that is, the platform compilation default) will be supported in a new alternative VisiBroker Linux binary release scheduled for early 2021. Contact Micro Focus technical support (https://supportline.microfocus.com/) for further details.

VisiSecure supports Transport Level Security (TLS) 1.3

VisiBroker 8.5.7 supports TLS version 1.3. All secured connections now default to use TLS1.3, provided both endpoints support this protocol.

- The C++ ORB moves from defaulting to TLS1.2 and now defaults to TLS1.3
- The Java ORB moves from defaulting to TLS1.0 and now defaults to the highest TLS level available from the JRE; see Issues resolved in this Service Pack.

Note that the newer TLSv1.3 cipher suites cannot be used with TLSv1.2 and earlier protocol versions; and vice versa, cipher suites from TLS versions up to TLSv1.2 cannot be used with TLSv1.3.

Where an existing Public Key Infrastructure could not support TLSv1.3, and it can be detected, VisiBroker automatically downgrades the TLS protocol version to the best usable version to suit the configured PKI. All automatic TLS protocol downgrades are recorded in the system log. For example, if you have configured a server without certificates, this requires a TLS_ECDH_anon cipher suite, which is not available at TLSv1.3, so that server will downgrade to TLSv1.2.

Configurable OpenSSL Security Levels

VisiBroker 8.5.7 for C++ implements a higher level of security than previous versions. OpenSSL defaults to disallowing the use of some weak and compromised security features such as small key sizes and weak cipher suites.

OpenSSL introduces a hierarchy of security levels, from 0 to 5, which become progressively more strict. Level 0 is provided to support legacy behaviors.

The new configuration options `vbroker.security.server.socket.TLSSecurityLevel` and `vbroker.security.client.socket.TLSSecurityLevel` enable you to specify this security level. Refer to https://www.openssl.org/docs/man1.1.0/man3/SSL_CTX_get_security_level.html for details of the PKI requirements for each security level.

Configurable Diffie-Hellman Groups for key exchange

With VisiBroker 8.5.7, Diffie-Hellman Key Groups (DH Groups) supersede the older Elliptic Curve configuration lists. (Note that Elliptic Curves are a sub-set of the possible range of Diffie-Hellman Groups.) You can optionally specify the groups to be used using the `vbroker.security.server.socket.TLSCipherGroups` and `vbroker.security.client.socket.TLSCipherGroups` configuration properties.

At different TLS levels, different DH Groups are supported. At TLSv1.3, the following five groups are supported:

- `X25519`
- `X448`
- `prime256v1`
- `secp384r1`
- `secp521r1`

At TLSv1.2 and below, only the following three groups are supported:

- `prime256v1`
- `secp384r1`
- `secp521r1`

When using ECDSA certificates, the identity key held within the certificates must lie in one of the DH Groups supported at the level of TLS being used.

A number of elliptic curves that were supported in previous releases are not supported in this release. This is due the insecure nature of the key sizes of those groups. The only DH Groups supported in this release are listed above.

New security configuration options

The following configuration options are new in VisiBroker 8.5.7 for C++:

`vbroker.security.server.socket.TLSSecurityLevel` `vbroker.security.client.socket.TLSSecurityLevel`	Specifies the security level from 0 (for legacy support) to 5. Defaults to 1.
`vbroker.security.server.socket.minTLSProtocol` `vbroker.security.client.socket.minTLSProtocol`	Specifies the minimum TLS protocol version required. Use this with the `maxTLSProtocol` options to create a range of supported protocol versions.
`vbroker.security.server.socket.maxTLSProtocol` `vbroker.security.client.socket.maxTLSProtocol`	Specifies the maximum TLS protocol version allowed.

`vbroker.security.TLS13CipherSuites`	A comma-separated list of cipher suites to be used if connecting at TLSv1.3. The existing `vbroker.security.cipherList` remains in use for TLSv1.2 and earlier. See the *Security Guide* for information on permitted cipher suites.
`vbroker.security.server.socket.TLSCipherGroups` `vbroker.security.client.socket.TLSCipherGroups`	Specifies the Diffie-Hellman key exchange groups to be used. The `server` property supersedes `vbroker.security.server.socket.ecdheCurve`, which is deprecated in this release.
`vbroker.security.server.socket.` `EnforceServerCipherPriority`	Enables the server to apply its own cipher suite preference order during the TLS handshake, overruling client preferences.
`vbroker.security.server.socket.MinDHGroupSize`	A minimum size of DH parameters required to feed into the Diffie-Hellman key exchange during TLS handshake. This only applies to fixed `_DH_anon` cipher suites with this release.

The following configuration option is amended in VisiBroker 8.5.7 for Java:

`vbroker.security.transport.protocol`	The default value has changed. It is now the highest TLS version supported by the underlying Java VM JSSE provider. Previously it was TLSv1.

See the VisiBroker 8.5.7 *Security Guide* for full details of these options.

Deprecated Features

Configuration options

The following configuration options are deprecated in VisiBroker 8.5.7 for C++:

`vbroker.security.server.socket.enabledProtocols` `vbroker.security.client.socket.enabledProtocols`	You are recommended to use the `maxTLSProtocol` and `minTLSProtocol` options instead, to specify a range of supported TLS protocol versions.
`vbroker.security.server.socket.ecdheCurve`	You are recommended to use `TLSCipherGroups` instead.

Unsupported Features

Unsupported cipher suites

Note that for security reasons:

- Cipher suites that include usage of the RC4 cipher are no longer supported in VisiBroker for C++.
- Other than the `TLS_DH_anon_` group of cipher suites, cipher suites using Fixed DH (that is, `TLS_DH_*` or `TLS_ECDH_*`) keys are no longer supported at all.

- Certificates signed with `MD5withRSA` are not supported at OpenSSL Security Levels above 0.

Unsupported Public Key Infrastructure

Note that for security reasons DSA certificates are no longer supported by VisiSecure.

Certicom Security Provider

The Certicom security provider was deprecated at VisiBroker 8.5 SP4. From VisiBroker 8.5.7 it is unsupported and only the OpenSSL `vbsec` library is supplied.

VisiBroker for .NET

VisiBroker for .NET has been discontinued, and references to it have been removed from the product and its documentation.

Removed Tools

The utilities `vbconsolew`, `lmadmw`, `JdsServerW`, `JdsExplorerW` have been identified as redundant and have therefore been removed from this release. The equivalent utilities, with the same names but without the trailing '`w`', are still included.

User Documentation

New documentation released with this Service Pack is available online, from https://www.microfocus.com/documentation/visibroker/visibroker857/.

Service Pack Archives do not contain the updated documentation, so the documentation accessed from within the product for these versions is the legacy documentation from the VisiBroker 8.5 GA version. Any platforms that have a new installation since 8.5 (such as Windows 10, introduced at 8.5 SP3) will contain the documentation that was current at the time of introduction.

Possible Problems when Upgrading to SP7

Maximum DH Group key size at Java 7

Most implementations of Java 7 have a fixed maximum DH Group key size of 768 bits. This conflicts with the OpenSSL `vbsec` supplied with VisiBroker for C++, which has a minimum allowable DH Group key size of 1024 bits.

Note that this also relates to the interaction between VisiSecure running on Java 7, and VisiSecure running on Java 8 and above. This means that VisiSecure running on Java 7 will be unable to establish an anonymous connection with more modern TLS implementations using any `TLS_DH_anon` cipher suites.

Java 7 is past End of Life. You are strongly recommended to upgrade to Java 8 or above. If this is not possible, however, possible workarounds are:

- Explicitly enable `anon` by removing it from the `jdk.tls.disabledAlgorithms` property configuration; AND also do one of the following:
    - Downgrade to a TLS protocol version that prefers a TLS_ECDH_anon cipher suite, that is, TLSv1.1 or TLSv1.0. This can be done through the use of the `vbroker.security.transport.protocol` property.
    - Explicitly disable `DH_anon` using the `jdk.certpath.disabledAlgorithms` and `jdk.tls.disabledAlgorithms` properties in the `java.security` configuration file.

- Explicitly enable one or more `TLS_ECDH_anon` cipher suites. This can be done through the use of the `vbroker.security.cipherlist` property.
- Remove `ECDH_anon` from the `jdk.tls.legacyAlgorithms` property configuration.

Note: Most of the the options listed above are only available in the later JDK 7 implementations.

Anonymous Connections at TLSv1.3

TLSv1.3 does not support the use of anonymous connections. If you wish to use TLSv1.3, you must configure certificates to enable these connections. Anonymous connections can still be used, but TLSv1.2 or below must be manually configured.

TLS_ECDH_RSA and TLS_ECDH_ECDSA cipher suites

The collection of `TLS_ECDH_RSA` and `TLS_ECDH_ECDSA` cipher suites are no longer available. If you are using these in your configuration, you should switch to using `TLS_ECDHE_ECDSA` cipher suites instead. This includes, but is not limited to, all configurations using EC identity certificates containing RSA signatures.

Use of Weak and Deprecated Hash Methods and Ciphers

In configurations that currently use weak hash methods and ciphers, in order to retain the current Public Key Infrastructure you must set your `TLSSecurityLevel` property to `0`. This includes, but is not limited to, RSA key sizes smaller than 1024, MD5 hashes and any other weak hash methods.

Solaris SPARC Unable to Connect Using TLSv1.3 when using NIOSSL

The OracleUcrypto security provider on Solaris SPARC will always fail at the TLS handshake when attempting to create TLSv1.3 connections. The workaround is to comment out this provider, which appears above the SunJSSE provider in the preference order list in the `java.security` configuration file. This will enable behavior parity with the other platforms.

Known Issues

Illegal Reflective Access warning with Java 11

Running VisiBroker applications with Java 11 may emit a warning like:

```
WARNING: An illegal reflective access operation has occurred
```

This issue is benign and will be fixed in a future VisiBroker release. Classes with this known issue are:

```
com.inprise.vbroker.rmi.CORBA.Java2FieldAccess
com.inprise.vbroker.util.TypeDescriptor
```

(RPI 636558, RPI 636559)

Use of deprecated constructors

Compiling generated code from IDL with Java 11 can result in warnings regarding the use of deprecated constructors for the following classes:

```
java.lang.Boolean
java.lang.Integer
java.lang.Long
java.lang.Short
java.lang.Float
java.lang.Double
java.lang.Character
java.lang.Byte
```

This issue is benign and will be fixed in a future VisiBroker release.

(RPI 636582)

Java NIOSSL Issues with TLSv1.3

The use of TLSv1.3 is not currently supported when using Java NIOSSL. When using NIOSSL, explicitly configuring the use of TLSv1.3 can fail at the TLS handshake. This behavior has been shown to lead to an infinite loop in the Java security layer. You will experience this issue only if you have explicitly enabled TLSv1.3 via the setting of properties `vbroker.security.server.socket.enabledProtocols` or `vbroker.security.client.socket.enabledProtocols`, which is unnecessary.

When NIOSSL is configured for an application, the NIOSSL connections will default to operating at a maximum level of TLSv1.2 until this issue is fixed.

(RPI 647741)

Dual IPv6/IPv4 Osagent

An alternate dual IPv6/IPv4 Smart Agent (OSAgent) was made available in previous releases as an optional patch that replaced the default IPv4 OSAgent implementation. This patch is required in the event that you need to use the OSAgent with your applications in network environments without any IPv4 capability. This functionality will be made available as a future HotFix to SP7. Please contact Micro Focus Technical Support if you need to receive this urgently.

Resolved Issues

The resolved issues that customers have reported are listed in this section. The numbers that follow each issue are the Reported Problem Incident number followed by the Customer Incident Numbers (in parentheses). RPIs that have numbers only (and no text) are included to confirm that the RPIs have been fixed, since no further information is required.

- The VisiBroker Java ORB property `vbroker.security.transport.protocol` had a default value of `TLSv1`, irrespective of the capabilities of the JDK being used. The value of this property controls the SSLContext Algorithm value used; see https://docs.oracle.com/en/java/javase/11/docs/specs/security/standard-names.html#sslcontext-algorithms for details.

  A default of `TLSv1` meant that, by default, VisiBroker for Java servers and clients would not attempt to use the more secure `TLSv1.1` (and later) protocol versions in their communication.

  VisiBroker for Java now establishes the highest TLS version supported by the underlying Java VM JSSE provider, and uses that by default as the preferred maximum TLS version bound.

  If required, you can restore the previous behavior by specifying:

  ```
  vbroker.security.transport.protocol=TLSv1
  ```

  647550

- A problem that led to `vbconsole` failing to start with JDK 11 has been fixed by adding a directory `tmp/vbconsole` to the VisiBroker product installation.

  A problem that led to `vbconsole` failing to start in a Windows x64 environment has been fixed by rebuilding `vbconsole` to pick up the right 64-bit artifacts from the JDK installation.

  1116791 (3187853)

- VisiBroker 8.5 SP7 is supported on RedHat Enterprise Linux 8 and SuSE Linux Enterprise Server 15. See Support for RedHat Enterprise Linux 8 and SuSE Linux Enterprise Server 15 for more information.

  1117962 (3202806) and 1118921 (3213726)

- A VisiBroker for Java server might hang indefinitely. This problem has been fixed at runtime by properly handling data saved in an intermediary buffer read from the secure connection.

  1116667 (3185838)

- A new property `vbroker.orb.serverRecvTimeoutDisconnects` is introduced. When an NIO connection times out because it has exceeded the value configured in the property `vbroker.orb.serverRecvTimeout`, this new property gives users the option to allow their server to continue waiting for incoming data on the connection, rather than closing the connection.

  If `vbroker.orb.serverRecvTimeoutDisconnects` is set to true, the NIO connection in question will be closed if it times out. This was the previous behavior, and is the default.

If set to false, the connection continues to wait for incoming data.

1119602 (3218752)

- On Windows systems, sending messages that are greater than 65473 bytes over a secure connection could result in a comms failure message. Logic that checks the result of WSAGetLastError now behaves as intended, and large secure messages are now sent correctly.

  1120330 (3226944)

- Unexpected MARSHAL exceptions could be encountered when GIOP message fragmentation was enabled in VisiBroker for Java. The problem was present in VisiBroker 8.5 SP5 and SP6. This issue has been fixed.

  1120490 (3229473)

Updates and SupportLine

Our Web site gives up-to-date details of contact numbers and addresses.

Further Information and Product Support

Additional technical information or advice is available from several sources.

The product support pages contain a considerable amount of additional information, such as:

- The WebSync service, where you can download fixes and documentation updates.
- The Knowledge Base, a large collection of product tips and workarounds.
- Examples and Utilities, including demos and additional product documentation.

To connect, enter https://www.microfocus.com in your browser to go to the Micro Focus home page.

Note: Some information may be available only to customers who have maintenance agreements.

If you obtained this product directly from Micro Focus, contact us as described on the Micro Focus Web site, https://www.microfocus.com. If you obtained the product from another source, such as an authorized distributor, contact them for help first. If they are unable to help, contact us.

Disclaimer

This software is provided "as is" without warranty of any kind. Micro Focus disclaims all warranties, either express or implied, including the warranties of merchantability and fitness for a particular purpose. In no event shall Micro Focus or its suppliers be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages, even if Micro Focus or its suppliers have been advised of the possibility of such damages. Some states do not allow the exclusion or limitation of liability for consequential or incidental damages so the foregoing limitation may not apply.