PRODUCT RELEASE

ObjectStore

ObjectStore Standard 2024.0

September 2024

IGNITETECH

ObjectStore

TABLE OF CONTENTS

ObjectStore

CHANGE LOG

ObjectStore Standard 2024.0

RESOLVED ISSUES

- **Incorrect file version for the executable binaries**

  File and product versions for DLLs and executables should now correctly be “2024.0”

- **[IBUENG-1924] Vulnerability in libexpat library**

  Libexpat was upgraded to version 2.6.2

- **[IBUENG-1701] Compilation error when building with C++ 20**

  It should no longer break with the error *C2760: syntax error: ‘register’ was unexpected here; expected ‘statement_seq’”* when compiling. Note that C++20 support is still a work in progress.

ObjectStore Standard 2023.1 (2013.0 update 25)

NEW FEATURES

- **Visual Studio 2022 (versions up to 17.2) is now supported.**

  All code that is linked with ObjectStore must be compiled with the **/d2FH4-** command line switch when 64-bit MSVC2019/MSVC2022 is used. ObjectStore does not support the FrameHandler4 format for exception handling. The **/Zc:implicitNoexcept-** and **/Zc:sizedDealloc-** command line switches are also mandatory.

- **Windows Server 2022 is now supported**

RESOLVED ISSUES

- **[OSCID-38179] ObjectStore assertion in objectstore::shutdown()**

  Assertion was removed as obsolete (it was used in 1999 to catch memory leak that was fixed in 2000).

- **[OSCID-38184] osverifydb bug**

  The bug in osverifydb was fixed. Osverifydb started to work quicker in fast mode if the database has a big number of bad pointers.

- **[OSCID-38154] Example dllschema lnk error on Windows**

  Example was fixed for MSVC2017.

ObjectStore Standard 2022.1 (2013.0 update 24)

RESOLVED ISSUES

- **[OSCID-38055] Resolved Jetty and Restlet vulnerabilities**

  Jetty and restlet have been updated to versions 9.4.41 and 2.4.3 respectively.

- **[OSCID-38155] os_schema_evolution::augment_new_member_initialisation parses namespaces incorrectly**

  os_schema_evolution::augment_new_member_initialisation was not namespace-aware and treated the namespace as a class name. This has been corrected.

- **[OSCID-38156] External architecture files cause an exception for known architectures**

Previously, loading an external architecture specification would cause an exception if the architecture was one already built into ObjectStore. Now an exception is raised only if the loaded architecture differs from the built-in specification.

- **[OSCID-38163] Emergency memory reserve for out of memory conditions**

  Previous versions of the ObjectStore server exited immediately if an out of memory exception was raised. It is now possible to reserve a block of memory when the server starts and use that memory as an emergency pool so the server can attempt to shut down cleanly. See below for details.

- **[IPCST-49316] Access violation with ossg and Visual Studio 2019**

  A change to some of the include files distributed with Visual Studio 2019 caused a crash in ossg. This has been fixed.

NEW FEATURE

- **New environment variable OS_SERVER_RESERVE_MEMORY**

  There is a new environment variable OS_SERVER_RESERVE_MEMORY, which contains the number of bytes of memory to be reserved when the server starts. The value must be greater than zero and is specified in decimal or in hexadecimal preceded by “0x”. In the event of an out of memory condition the server will use this memory as an emergency pool and attempt to perform a clean shutdown.

  The value required is heavily dependent on what the server is doing but 2 MiB is a reasonable amount. This is particularly useful for heavily loaded 32-bit systems where the limited address space may cause out of memory exceptions to be raised.

  If no reserve memory is allocated, or if a second out of memory condition occurs when the server is trying to shut down, then the server reverts to the previous behavior and exits immediately.

ObjectStore Standard 2020.2 (2013.0 update 23)

RESOLVED ISSUES

- **OSCID-37909] - Vulnerabilities detected on objectstore_20130u20**

  Data Services Administrator component contains Jetty version 9.2.28 that has a vulnerability CVE-2017-9735. Jetty and Restlet were updated to versions 9.4.28 and 2.4.0 respectively.

ObjectStore Standard 2020.1 (2013.0 update 22)

NEW FEATURES

- **Visual Studio 2019 (version 16.4.5 at the time of writing) is now supported.**

  New architectures were introduced and architecture files have been updated.

  All code that is linked with ObjectStore must be compiled with the **/d2FH4-** command line switch when 64-bit MSVC2019 is used. ObjectStore does not support the FrameHandler4 format for exception handling. The **/Zc:implicitNoexcept-** and **/Zc:sizedDealloc-** command line switches are also mandatory.

- **Windows Server 2019 is now supported**

RESOLVED ISSUES

- **[OSCID-37764] ossg with -smf flag throws err-0001-0126**

  On one of the previous releases the list of C++ operators was not updated. The C++20 standard adds the “spaceship” operator <=> and the ObjectStore C++ parser was not updated to reflect the change, causing an operator list overflow.

NOTES

CERTIFIED PLATFORMS

Platform	Version
Java	8
Visual Studio	2017, 2019, 2022*
Windows	2012 R2, 2016, 2019, 2022
GCC	4.4.7, 4.8.5
Red Hat Enterprise Linux	6.10, 7.9

*Up to VS 2022 17.2

SUPPORT DISCONTINUANCE

- Consumer-grade Windows versions (7, 8, 8.1, 10, 11) are no longer supported.
- For RHEL 6 and 7, only the latest minor versions are supported (6.10 / 7.9).
- 32-bit builds are no longer supported.

KNOWN ISSUES

- When updating to ObjectStore 2024.0 or later, please update the registry keys from “ObjectStore 2013” to the latest major version (e.g. “ObjectStore 2024” for 2024.x, “ObjectStore 2025” for 2025.x, etc.)
- New architectures were introduced in ObjectStore 2019.1 (2013.0 update 15). The Vector Header handling code was also changed.

  New architectures were added for MSVC2015 and MSVC2017, both x32 and x64. This leads to problems in interwork setting during the interaction between existing and newly added architectures when previous clients try to read data created or modified by the client with the new architecture. Existing clients could use architecture definition files to work around this issue.

  The error shown is like: *ObjectStore internal error <maint-0025-0131>Unknown architecture code 53. (err_internal)*

Vector Header handling code was changed for MSVC2015 and MSVC2017 x64 platforms due to the changes coming from a bugfix by Microsoft in its x64 compiler code. Unfortunately, this parameter is not alterable via architectural definition files because the behavior is complicated. Therefore it is worked around by setting an environment variable OS_FIX_VECTOR_HEADERS to 1 or solved by upgrading ObjectStore Client installations to an ObjectStore 2019.1 (2013.0 update 15) or later.

The error shown is like: *An incorrect vector header was found during inbound relocation at 0X000000003000 3D80. The source architecture is an Intel platform with 64-bit Windows and Visual C++ 141 or higher. The address corresponds to offset 0x3d80 within cluster # 0, segment #0 of database DATABASE_NAME.*

There is a **workaround** for these issues:

1. Providing an architecture definition file for use with existing client installations. The architecture definition file is shipped in the release package.
2. Setting the environment variable OS_FIX_VECTOR_HEADERS to 1 - this makes sure that vector headers created by MSVC2015/2017 x64 could be read by the existing clients.

There is a **solution** to these issues:

1. Rebase the existing ObjectStore Client installations to ObjectStore Standard 2019.1 (2013.0 update 15) or later for the corresponding platform.

Those issues affect only the ObjectStore Client; the ObjectStore Server and the Cache manager are unaffected, so any existing ObjectStore Server installation could continue to be used without any modifications.

- New architectures were introduced in ObjectStore 2020.1 (2013.0 update 22).

  New architectures were added for MSVC2019 x32 and x64.

  The Vector Header handling code was NOT changed. MSVC2019 x64 vector headers have the same format as MSVC2015 and MSVC2017.

- New architectures were introduced in ObjectStore 2023.1 (2013.0 update 25).

  New architectures were added for MSVC2022 x32 and x64.

  The Vector Header handling code was NOT changed. MSVC2022 x64 vector headers have the same format as MSVC2015, MSVC2017, MSVC2019.

- Databases growing above 1 TB can cause server crashes and data loss.

  The ObjectStore server can't handle the databases reaching its theoretical size limitations correctly. The servers which reach these constraints will crash, and this crash will likely result in the loss or corruption of the latest data submitted to it.

  For the file-based databases, the file size can be monitored, and when this size approaches 1 TB it should be compacted, or separated into multiple databases.

  Note that the 1 TB limit is related to the internal ObjectStore structures - it's possible that the server reaches this database size way before the logical database size reaches this threshold. If the stored data is significantly smaller, the oscompact utility should be used to reduce the physical database size.

- The indexed cursor iteration order might be not correct when multiple virtual inheritance is used.

  When the member that is the subject of indexing is in one of the virtually inherited structs the sorting function fails. The workaround for this issue is to move the member being indexed and sorted on to the leaf node.

- C++11 and higher (Modern C++) features are not supported in ObjectStore contexts.

  Microsoft Visual Studio 2015 introduced advanced support of C++11/C++14 standard, and Microsoft Visual Studio 2017 improved the standard compliance further. All newly introduced features (e.g. lambdas) are not supported in ObjectStore contexts - e.g. in ObjectStore transactions, TIX exception capture blocks, etc. It is still possible to use Modern C++ in other parts of the application.

- /Zc:implicitNoexcept- /Zc:sizedDealloc- compiler command-line switches are mandatory for Visual Studio 2015 and later.

  Microsoft Visual Studio 2015 introduced the support for the C++11 standard.

  The C++11 standard includes the noexcept specifier, which is implicit for destructors. ObjectStore does throw from destructors, so using the /Zc:implicitNoexcept- compiler command-line switch is mandatory for the compilation of the code that is linked to ObjectStore.

  The C++11 standard includes sized deallocation. ObjectStore defines overloads of new and delete operators that conflict with sized deallocation, so using the /Zc:sizedDealloc- compiler command-line switch is mandatory for the compilation of the code that is linked to ObjectStore.

HOW TO EMBED OBJECTSTORE COMPONENTS INTO ANOTHER INSTALLER

1. Install the required components of ObjectStore on the development machine.
2. Zip up all files in the installation directory.
3. Bundle the zip into your installer package.
4. Add instructions into the installer to:
    4.1. Extract the bundled zip
    4.2. Create below registry settings.
    4.3. Start the ObjectStore services.

Here is the list of registry entries and system environment variables that should be applied to the target machine. They are shared by the ObjectStore components and named according to the legacy installer. The "${OS_INSTALL_DIR}" should be replaced in runtime to the path of unzipped ObjectStore components.

C++ Interface - DBMS Client and Server

```
setx OS_ROOTDIR = ${OS_INSTALL_DIR}\OStore
setx OS_TMPDIR = ${OS_INSTALL_DIR}\temp
setx LIB = ${OS_INSTALL_DIR}\OStore\lib;%LIB%
setx INCLUDE = ${OS_INSTALL_DIR}\OStore\include;%INCLUDE%
```

```
setx PATH = ${OS_INSTALL_DIR}\OStore\bin;%PATH%

[HKEY_LOCAL_MACHINE\SOFTWARE\ObjectStore Inc.\ObjectStore 2024\Registration\]
"ObjectStore ExamplesDirectory"=""
"ObjectStore HTML Documentation Directory"=""
"ObjectStore Install Directory"="${OS_INSTALL_DIR}\OStore"
"ObjectStore PDF Book Files Directory"=""
"ObjectStore Single Directory"=""
"ObjectStore Suite Top Directory"=${OS_INSTALL_DIR}"
"Program Folder Name"="ObjectStore Win64 2024.0"
"Update"="0"

[HKEY_LOCAL_MACHINE\SOFTWARE\ObjectStore Inc.\ObjectStore 2024\Server\]
"Auto Start Server"="1"

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\services\ObjectStore Cache Manager R7.0\]
"DelayedAutostart"=dword:00000001
"DisplayName"="ObjectStore Cache Manager R7.0"
"ErrorControl"=dword:00000001
"ImagePath"="${OS_INSTALL_DIR}\OStore\BIN\OSCMGR6.EXE"
"ObjectName"="LocalSystem"
"Start"=dword:00000002
"Type"=dword:00000010
"WOW64"=dword:00000001

[HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\services\ObjectStore Server R7.0\]
"DelayedAutostart"=dword:00000001
"DisplayName"="ObjectStore Server R7.0"
"ErrorControl"=dword:00000001
"ImagePath"="${OS_INSTALL_DIR}\OStore\BIN\OSSERVER.EXE"
"ObjectName"="LocalSystem"
"Start"=dword:00000002
"Type"=dword:00000010
"WOW64"=dword:00000001
```

DSA as Server

Same as in C++ Interface plus:

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\services\dsa\]
"Description"="ObjectStore System Administration and Monitoring"
"DisplayName"="Data Services Administrator"
"ErrorControl"=dword:00000001
```

```
"ImagePath"="${OS_INSTALL_DIR}\sysadmin\bin\wrapper.exe -s
${OS_INSTALL_DIR}\sysadmin\bin\conf\wrapper.conf"
"ObjectName"="LocalSystem"
"Start"=dword:00000002
"Type"=dword:00000010

[HKEY_LOCAL_MACHINE\SOFTWARE\ObjectStore Inc.\ObjectStore 2024\DataServices\]
"UseType"="DSA Server"
```

DSA as Process Manager

Same as in DSA as Server except:

```
[HKEY_LOCAL_MACHINE\SOFTWARE\ObjectStore Inc.\ObjectStore 2024\DataServices\]
"UseType"="DSA Process Manager"
```

DDML

Same as in C++ Interface except:

```
setx PATH = ${OS_INSTALL_DIR}\DDML\bin;${OS_INSTALL_DIR}\OStore\bin;%PATH%
```

Java Interface

```
setx OS_ROOTDIR = ${OS_INSTALL_DIR}\OStore
setx OS_TMPDIR = ${OS_INSTALL_DIR}\temp
setx LIB = ${OS_INSTALL_DIR}\OStore\lib;%LIB%
setx INCLUDE = ${OS_INSTALL_DIR}\OStore\include;%INCLUDE%
setx PATH = ${OS_INSTALL_DIR}\OSJI\bin;${OS_INSTALL_DIR}\OStore\bin;%PATH%
setx OSJI_ROOTDIR = ${OS_INSTALL_DIR}\OSJI
```

JMTL

Same as in Java Interface except:

```
setx PATH =
${OS_INSTALL_DIR}\JMTL\bin;${OS_INSTALL_DIR}\OSJI\bin;${OS_INSTALL_DIR}\OStore\bin;%PATH%
```

IGNITETECH UNLIMITED

If you would like to upgrade your support plan to take advantage of the **Unlimited Program features**, please contact us for more information. If you have any questions about this release, please open a **support ticket** and our support team will be happy to assist you.

To ensure all of the appropriate staff within your organization are informed about important product updates, please notify **success@ignitetech.com** with emails for individuals who should receive these announcements.