GemStone/S 64 Bit™ Installation Guide

For Linux on x86_64 Compatible Systems

Version 3.5

June 2019

INTELLECTUAL PROPERTY OWNERSHIP

This documentation is furnished for informational use only and is subject to change without notice. GemTalk Systems LLC assumes no responsibility or liability for any errors or inaccuracies that may appear in this documentation.

Warning: This computer program and its documentation are protected by copyright law and international treaties. Any unauthorized copying or distribution of this program, its documentation, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted under the maximum extent possible under the law.

The software installed in accordance with this documentation is copyrighted and licensed by GemTalk Systems under separate license agreement. This software may only be used pursuant to the terms and conditions of such license agreement. Any other use may be a violation of law.

Use, duplication, or disclosure by the Government is subject to restrictions set forth in the Commercial Software - Restricted Rights clause at 52.227-19 of the Federal Acquisitions Regulations (48 CFR 52.227-19) except that the government agency shall not have the right to disclose this software to support service contractors or their subcontractors without the prior written consent of GemTalk Systems.

This software is provided by GemTalk Systems LLC and contributors "as is" and any expressed or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall GemTalk Systems LLC or any contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

COPYRIGHTS

PATENTS

GemStone software is or has been covered by U.S. Patent Number 6,256,637 "Transactional virtual machine architecture" (1998-2018), Patent Number 6,360,219 "Object queues with concurrent updating" (1998-2018), Patent Number 6,567,905 "Generational garbage collector with persistent object cache" (2001-2021), and Patent Number 6,681,226 "Selective pessimistic locking for a concurrently updateable database" (2001-2021).

TRADEMARKS

GemTalk, **GemStone**, **GemBuilder**, **GemConnect**, and the GemTalk logo are trademarks of GemTalk Systems LLC, or of VMware, Inc., previously of GemStone Systems, Inc., in the United States and other countries.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Solaris, **Java**, and **Oracle** are trademarks or registered trademarks of Oracle and/or its affiliates. **SPARC** is a registered trademark of SPARC International, Inc.

Intel and **Pentium** are registered trademarks of Intel Corporation in the United States and other countries.

Microsoft, **Windows**, and **Windows Server** are registered trademarks of Microsoft Corporation in the United States and other countries.

Linux is a registered trademark of Linus Torvalds and others.

Red Hat and all Red Hat-based trademarks and logos are trademarks or registered trademarks of Red Hat, Inc. in the United States and other countries.

Ubuntu is a registered trademark of Canonical Ltd., Inc., in the U.S. and other countries.

SUSE is a registered trademark of Novell, Inc. in the United States and other countries.

AIX, **POWER6**, **POWER7**, and **POWER8** and **VisualAge** are trademarks or registered trademarks of International Business Machines Corporation.

Apple, **Mac**, **MacOS**, and **Macintosh** are trademarks of Apple Inc., in the United States and other countries.

CINCOM, **Cincom Smalltalk**, and **VisualWorks** are trademarks or registered trademarks of Cincom Systems, Inc.

Other company or product names mentioned herein may be trademarks or registered trademarks of their respective owners. Trademark specifications are subject to change without notice. GemTalk Systems cannot attest to the accuracy of all trademark information. Use of a term in this documentation should not be regarded as affecting the validity of any trademark or service mark.

GemTalk Systems LLC
15220 NW Greenbrier Parkway
Suite 240
Beaverton, OR 97006

Preface

About This Documentation

This document explains how to install GemStone/S 64 Bit version 3.5 on a workstation running Linux on x86_64 Compatible Systems, and how to upgrade from previous GemStone/S 64 Bit versions.

Terminology Conventions

The term "GemStone" is used to refer to the server products GemStone/S 64 Bit and GemStone/S, and the GemStone family of products; the GemStone Smalltalk programming language; and may also be used to refer to the company.

Technical Support

Support Website

gemtalksystems.com

GemTalk's website provides a variety of resources to help you use GemTalk products:

- **Documentation** for the current and for previous released versions of all GemTalk products, in PDF and HTML.
- **Product download** for the current and selected recent versions of GemTalk software.
- **Bugnotes**, identifying performance issues or error conditions that you may encounter when using a GemTalk product.
- **Supplemental Documentation** and **TechTips**, providing information and instructions that are not in the regular documentation.
- **Compatibility matrices**, listing supported platforms for GemTalk product versions.

We recommend checking this site on a regular basis for the latest updates.

Help Requests

GemTalk Technical Support is limited to customers with current support contracts. Requests for technical assistance may be submitted online (including by email), or by telephone. We recommend you use telephone contact only for urgent requests that require immediate evaluation, such as a production system down. The support website is the preferred way to contact Technical Support.

Website: techsupport.gemtalksystems.com

Email: techsupport@gemtalksystems.com

Telephone: (800) 243-4772 or (503) 766-4702

Please include the following, in addition to a description of the issue:

- The versions of GemStone/S 64 Bit and of all related GemTalk products, and of any other related products, such as client Smalltalk products, and the operating system and version you are using.
- Exact error message received, if any, including log files and statmonitor data if appropriate.

Technical Support is available from 8am to 5pm Pacific Time, Monday through Friday, excluding GemTalk holidays.

24x7 Emergency Technical Support

GemTalk offers, at an additional charge, 24x7 emergency technical support. This support entitles customers to contact us 24 hours a day, 7 days a week, 365 days a year, for issues impacting a production system. For more details, contact GemTalk Support Renewals.

Training and Consulting

GemTalk Professional Services provide consulting to help you succeed with GemStone products. Training for GemStone/S is available at your location, and training courses are offered periodically at our offices in Beaverton, Oregon. Contact GemTalk Professional Services for more details or to obtain consulting services.

Table of Contents

Chapter 1. Installing GemStone/S 64 Bit Version 3.5

Review the Installation Procedure . . . 7
Check the System Requirements . . . 8
    Configure the Operating System . . . 8
Prepare for Installation . . . 14
    Set the Environment . . . 15
    Set the GemStone Key File . . . 15
    Verify TCP/IP . . . 16
    Define the NetLDI Service . . . 16
Run the Installation Script . . . 16
    Decisions to Make During Installation . . . 17
Change Passwords for Administrative Accounts . . . 19
Add Users . . . 19
    Users must set environment . . . 19
Install the default TimeZone . . . 20
Further Configuration and Administration . . . 20

Chapter 2. Upgrading from GemStone/S 64 Bit 3.3.x or 3.4.x

Keyfiles . . . 21
Upgrade Procedure . . . 22
    Prior to Upgrade in existing application . . . 22
    Prepare for Upgrade . . . 22
    Perform the Upgrade . . . 24
    GsDevKit Upgrade . . . 25
    Post-upgrade Application Code Modifications . . . 25
    Backup repository and configure GBS . . . 26

Chapter 3. Upgrading from 3.2x

Keyfiles . . . 27
Upgrade Procedure . . . 28
Prior to Upgrade in existing application . . . 28
Prepare for Upgrade . . . 29
Perform the Upgrade . . . 30
GsDevKit Upgrade . . . 31
Post-upgrade Application Code Modifications . . . 31
Backup repository and configure GBS . . . 34

Chapter 4. Upgrading GLASS/GsDevKit Applications

Upgrade Procedure . . . 35
1. Ensure that GemStone 3.5 is installed and your repository upgraded . . . 36
2. If necessary, customize the upgrade instructions . . . 36
3. Perform the Upgrade . . . 37
4. Load your Application Code . . . 38
5. Complete the Upgrade Process . . . 38

Chapter 5. Configuring GBS for GemStone/S 64 Bit

Supported GBS client Smalltalk Platforms with Linux . . . 40
GBS Setup or Upgrade Procedure . . . 40
Upgrade GBS to a version that supports v3.5 . . . 41
Update GBS to reference v3.5 libraries . . . 41

Chapter

1 Installing GemStone/S 64 Bit Version 3.5

This chapter describes the procedure for installing GemStone/S 64 Bit™ version 3.5. We recommend that you set up GemStone initially on a single machine, to ensure that all the pieces work together. At the end of this chapter, we suggest refinements you might want to make. Further setup to run a distributed system are described in the *System Administration Guide*. You will need to adjust the installation process to meet your specific needs.

If you are upgrading to this release from a previous version of GemStone/S 64 Bit, follow the instructions in the appropriate later chapter of this Installation Guide. These upgrade instructions will provide details on steps that need to be taken before and after the installation as described in this chapter.

Review the Installation Procedure

The following list summarizes the steps to install GemStone/S 64 Bit.

- Check the System Requirements. 8
- Prepare for Installation 14
- Set the Environment 15
- Set the GemStone Key File 15
- Verify TCP/IP 16
- Define the NetLDI Service 16
- Run the Installation Script 16
- Change Passwords for Administrative Accounts 19
- Add Users 19
- Install the default TimeZone 20

Check the System Requirements

Before you install GemStone/S 64 Bit, ensure that the following system requirements are satisfied. Systems meeting these requirements are suitable for installing GemStone/S 64 Bit and beginning development, but additional system resources may be necessary to support large applications.

For Compiler and Debugger versions, refer to the GemBuilder for C manual.

Platform

- System with an x86_64-compatible processor.

RAM and Swap space

- While small installations can run on systems with only a few GB of physical RAM, increasing RAM is important for GemStone performance.

  Total swap space should be at least equal to the amount of RAM. Due to the way GemStone uses memory, systems with insufficient swap space allocated have a risk of memory errors even if there is available RAM.

Disk space

- Space for the installed distribution files—you need approximately 450 MB for GemStone/S 64 Bit, and additional space for other products.
- Additional disk space as required for your repository.

  The repository files should be located on a disk drive that does not contain swap space. Use of multiple disk drives is advisable for servers.

Operating system

- Red Hat Linux ES 6.9
  kernel version 2.6.32-696.18.7.el6.x86_64 and glibc-2.12-1.209.el6_9.2.x86_64
- Red Hat Linux ES 7.4
  kernel version 3.10.0-693.11.6.el7.x86_64 and glibc-2.17-196.el7_4.2.x86_64
- Ubuntu 16.04 LTS
  kernel version 4.4.0-97-generic and glibc 2.23-0ubuntu9
- Ubuntu 18.04 LTS
  kernel version 4.15.0-45-generic and glibc 2.27-0ubuntu1
- SUSE Linux 12
  kernel version 3.12.44-52.10-default and glibc-2.19-20.3.x86_64

Configure the Operating System

The kernel must be configured to support shared memory and semaphores. See your operating system documentation for further information. These requirements apply both to server nodes and to client nodes.

1. Shared memory

The upper limit for shared memory single segment size and total usage should be set to values larger than your desired Shared Page Cache size, and not more than 75% of your real memory size.

The single segment maximum size, `shmmax`, is set in bytes, and the total shared memory limit, `shmall`, is configured in pages, with a base page size of 4KB. Note that the results of ipcs may be reported in kbytes.

For example, if you have 8192 MB of real memory:

```
8192 MB * .75 = 6144 MB
6144 MB * 2^20 = 6442450944 bytes
6442450944 / 4K = 1572864
```

To set shared memory sizes, you would append the following text to the `/etc/sysctl.conf` file. The settings are read from this file during the boot process.

```
# Shared Memory setting for GemStone
kernel.shmall = 1572864
kernel.shmmax = 6442450944
```

For more details, consult your Linux operating system documentation.

2. Semaphores

You may need to increase the settings for semaphores. These settings are configured by setting `kernel.sem` to a 4-element array, with the equivalent to the old semmsl, semmns, semopm, and semmni. For example, append the following to the `/etc/sysctl.conf` file.

```
kernel.sem=1000 512000 64 2048
```

The first element sets the maximum number of semaphores per id (per semaphore set). This parameter limits the number of GemStone sessions that can log in to a particular Stone and connect to its shared page cache.

On the Stone's node, this parameter must provide **two** semaphores for each user who will log in to that Stone from any node plus an overhead of **four**. In distributed systems, nodes that have only user sessions must provide **two** semaphore for each user session on that node plus an overhead of **one**.

The number of semaphores actually requested for a particular shared page cache depends on the GemStone configuration file read by the process that starts the cache and is (SHR_PAGE_CACHE_NUM_PROCS * 2) + 1.

The second value sets the total number of semaphores in the system, which must be increased to along with the first.

3. File Descriptors

Each user session requires two file descriptors, and others are needed for extents, transaction logs, and other overhead. The default setting for `fs.file-max` setting is usually sufficient.

4. Locking the Shared Page Cache in memory

If you intend to lock the shared page cache into memory via the stone configuration option SHR_PAGE_CACHE_LOCKED, then the linux user starting the stone must either have the Linux capability CAP_IPC_LOCK, or have a RLIMIT_MEMLOCK resource limit set greater than the size of the SPC.

5. OOM Killer

If your system runs low on memory, the Linux OOM killer may select GemStone processes to terminate. To protect the shared page cache and other critical GemStone processes, each GemStone process's oom_score_adj, which is used to select processes to terminate, is adjusted. If the userid that will be running the server processes has the CAP_SYS_RESOURCE privilege, critical GemStone processes have their oom_score_adj reduced, making them safer; if the user does not have CAP_SYS_RESOURCE, then non-critical processes such as Gems have their score increased, so they will be selected rather than more critical processes.

To set CAP_SYS_RESOURCE on kernels v2.6.32 and later, set the capability on the executables:

a. Install `libcap2-bin`

b. `for i in pgsvrmain gem stoned shrpcmonitor; do sudo setcap cap_sys_resource=pe $GEMSTONE/sys/$i ; done`

c. `for i in startstone topaz; do sudo setcap cap_sys_resource=pe $GEMSTONE/bin/$i ; done`

6. PAM

If you are using UNIX authentication for GemStone logins, or if you run NetLDI as root with setuid (i.e. not in guest mode), you must have PAM (Pluggable Authentication Module) configured on the server. You may include a specific GemStone authorization service name, or allow the default "other" authentication definitions to be used.

PAM authentication definitions are in files under the directory `/etc/pam.d`. Alternatively, they can be lines in the configuration file `/etc/pam.conf`, but this usage is deprecated on many distributions. On these distributions, the presence of the `/etc/pam.d` directory will cause `/etc/pam.conf` to be ignored.

The specific GemStone service file names are `gemstone.gem` for user authentication, and `gemstone.netldi` for a NetLDI running with authentication.

The libraries that are specified in the stack depend on how you are configuring PAM to perform the authentication. The examples below are for PAM configured to invoke LDAP for authentication.

For example, to allow GemStone UNIX authentication, which uses PAM, to authenticate via LDAP, create a file named `/etc/pam.d/gemstone.gem` with the following contents:

```
auth                required            pam_ldap.so
```

For NetLDI authentication, again using LDAP, create a file named `/etc/pam.d/gemstone.netldi` with the following contents:

```
auth                required            pam_ldap.so
```

Red Hat, by default, installs a file `/etc/pam.d/other` which disables "other" authentication. On Ubuntu, it is enabled by default. You can allow the "other" authentication stack to be used for GemStone authentication by ensuring that the file `/etc/pam.d/other` has the following contents:

```
auth                required            pam_ldap.so
```

Consult your System Administrators for more information on how authentication is handled on your system.

7. Large Memory Pages

The default size for memory pages is 4KB. If you have a large repository, using large pages (in Linux, these are called Huge Pages) may improve performance. Linux supports 2MB and 1GB huge pages; a Linux installation may be configured to support one or both of these. To use large pages, you must determine the huge page size and the number of huge pages needed, configure Linux to allocate the required number of pages, and configure GemStone to use large pages.

To configure the use of large pages:

a. Determine pages sizes available on your Linux system, and decide on the size you intend to use. Smaller repositories may use 2MB pages, but repositories with multi-GB shared page cache sizes may benefit more from 1GB pages.

   To determine the default huge page size, execute:

   ```
   unix> grep -e '^Huge' /proc/meminfo
   ```

   which reports, for example,

   ```
   HugePages_Total:    5000
   HugePages_Free:     2756
   HugePages_Rsvd:        0
   HugePages_Surp:        0
   Hugepagesize:       2048 kB
   ```

   To determine if you have another page size available, execute:

   ```
   unix> ls /sys/kernel/mm/hugepages/
   ```

   If you have 2MB pages, it will return:

   ```
   hugepages-2048kB
   ```

   If you have both 2MB and 1GB pages, it will report:

   ```
   hugepages-1048576kB  hugepages-2048kB
   ```

b. Determine the number of huge pages that will be needed, based on your GemStone configuration. This is calculated by the utility **largememorypages**. This utility needs several details to compute the number of required pages: the shared page cache size, the maximum number of GemStone processes, and the number of shared counters. These can be provided by arguments or read from an existing configuration file.

   `largememorypages [-e` *path* `][-z` *path*`][-F` *cacheFrames* `| -M` *cacheKB*`]`
   `[ -P` *maxProcesses*`] [-C` *maxSharedCounters*`] [-p` *largeMemoryPageSize*`]`

   You may specify the required values using a valid configuration file. This is parsed as a Stone configuration file and the necessary values extracted.

   -e *path*
   specifies an executable config file (same as startstone -e).

   -z *path*
   specifies a system config file (same as startstone -z).

   If a config file is not specified, then -M or -F and -P and -C are required. These arguments can be used in addition to the configuration file, in which case they override values set or computed in the configuration file:

-F *cacheFrames*
shared cache size expressed in 16 KB frames
-M *cacheKB*
shared cache size with a optional units suffix.
-P *maxProcesses*
setting for SHR_PAGE_CACHE_NUM_PROCS
-C *maxSharedCounters*
setting for SHR_PAGE_CACHE_NUM_SHARED_COUNTERS.
-p *largeMemoryPageSize*
setting for SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_SIZE_MB. Optional; the default is 2MB, may also specify 1 GB (or equivalent using appropriate units).

The following example is for an (approximately) 20GB shared page cache, 200 processes, and 1900 shared counters, with 2 MB memory pages:

```
unix> largememorypages -p 2MB -M 20GB -C 1900 -P 200
Cache config is 1310720 pages = 20480 MB,  total is 21384 MB,
overhead 4% of configured size
Large page size requested is: 2 MB.
Large page overhead: 1.12 MB
For 1310720 pages, 200 processes and 1900 shared counters, 10
pusherThreads
  required cache size is 22422749184 bytes.
Number of 2 MB large pages required: 10692
<with further update commands>
```

c. Configure Linux to use large pages, according to the instructions for your Linux distribution. The following commands may be used. You must execute these as root.

To enable large/huge pages until the next system reboot:

```
echo numpages > /sys/kernel/mm/hugepages/hugepages-
hugePageSize/nr_hugepages
```

For example:

```
unix> echo 10692 > /sys/kernel/mm/hugepages/hugepages-
2048kB/nr_hugepages
```

To permanently enable large pages:

```
echo "vm.nr_hugepages=numpages" >> /etc/sysctl.conf
```

Note `nr_hugepages` may fail if insufficient contiguous memory is available. You can confirm the number of large memory pages that are available for use using `grep -e '^Huge' /proc/meminfo`.

d. Ensure that the executables can use large pages.

You can configure the executables to use large pages, by using `setcap`:

```
unix> setcap cap_ipc_lock=pe $GEMSTONE/sys/startshrpcmon
unix> setcap cap_ipc_lock=pe $GEMSTONE/sys/shrpcmonitor
```

Alternatively, the SPC monitor process can be run with an effective user ID of root:

```
chown root $GEMSTONE/sys/shrpcmonitor $GEMSTONE/sys/startshrpcmon
chmod u+s $GEMSTONE/sys/shrpcmonitor $GEMSTONE/sys/startshrpcmon
```

Configure GemStone to request large pages by set the configuration option SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_POLICY. This can be set to 1 or 2; with a setting of 1, the cache will be started anyway if the request for large pages is denied, while a setting of 2 indicates that startup should fail if large pages cannot be allocated.

If you are using 1GB pages, you will also need to update the setting for SHR_PAGE_CACHE_LARGE_MEMORY_PAGE_SIZE_MB. By default, this is set to 2; if you are using 1GB pages, set this to 1024.

8. Configuring GemStone's Shared Page Cache

The configuration option SHR_PAGE_CACHE_SIZE_KB defines the size (in KBytes) of extent page space in the shared page cache. The maximum acceptable value for this configuration option is limited by system memory, kernel configurations, cache space allocated by SHR_PAGE_CACHE_NUM_PROCS and space allocated for other GemStone caches.

For more general information about these and other configuration options, see Appendix A of the *System Administration Guide*.

9. System clock

The system clock must be set to the correct time. When GemStone opens the repository at startup, it compares the current system time with the recorded checkpoint times as part of a consistency check. A system time earlier than the time at which the last checkpoint was written may be taken as an indication of corrupted data and prevent GemStone from starting. The time comparisons use GMT.

10. TCP keepalive option

GemStone processes ordinarily use the TCP `keepalive` option to determine how long they will wait after communications activity ceases unexpectedly. This setting can be useful for reaping stale RPC Gems, but the operating system default may not be appropriate for this purpose. For further information, refer to your operating system documentation.

11. Unset LD_BIND_NOW

On some Linux distributions, setting the environment variable LD_BIND_NOW may result in process startup failures due to loading incorrect shared libraries.

Prepare for Installation

Installing GemStone can be done as a regular user, but in order to setup shared security, some portions of the installation should be done when logged in as the root user. Other steps of the installation are done as the unix user who will be the GemStone administrative account.

In addition to the installation directory, these are the portions of the system that are affected by the installation of GemStone:

`/dev/raw`
: Optional raw partitions for repository extents and transaction logs.

`/etc/services`
: Internet services database, for NetLDI name lookup.

`/opt/gemstone`
: Default location for server lock files, host name id file, and log files for GemStone network servers (NetLDIs). See the *System Administration Guide* for details.

1. Log in as the GemStone administrator to the machine on which you are installing GemStone. This part of the installation should **not** be done as root, to ensure all the files are not owned by root.
2. Determine that adequate swap space is available.:

   ```
   unix> cat /proc/swaps
   ```

3. Select the drive on which you will install the GemStone software, and the installation directory on this drive, *InstallDir*. Make this directory the current working directory.

   We recommend that you avoid choosing either an NFS-mounted partition or one containing UNIX swap space for the initial installation. Mounted partitions can result in executables running on the wrong machine and in file permission problems. Existence of swap space on the same drive can dramatically slow GemStone disk accesses.

4. GemStone/S 64 Bit is provided as a zipped archive file with a name similar to `GemStone64Bit3.5.0-x86_64.Linux.zip`. Move this distribution file to the directory location in which GemStone will be installed, *InstallDir*.
5. Unzip the distribution file using `unzip`.
6. The *InstallDir* now contains a GemStone directory with a name similar to `GemStone64Bit3.5.0-x86_64.Linux`.

   In addition to subdirectories, this directory also contains two text files: `PACKING`, which lists all of the GemStone files, and `version.txt`, which identifies this particular product and release of GemStone.

Set the Environment

Perform the following steps to properly configure the operating environment.

1. Set the environment variable GEMSTONE.
   a. If more than one installation of any GemStone/S product resides on this machine, check for existing GemStone environment variables:

      ```
      > env | grep GEM
      ```

      All GemStone environment variables are displayed.

   b. If any environment variables exist and are not appropriate for the new installation, you must specifically unset each one. For example:

      ```
      > unset GEMSTONE GEMSTONE_SYS_CONF
      ```

   c. Set the environment variable GEMSTONE to the *full pathname* (starting with a slash) of your new GemStone installation directory. For example:

      ```
      > export GEMSTONE=InstallDir/GemStone64Bit3.5.0-x86_64.Linux
      ```

Set the GemStone Key File

To run GemStone, you must have a key file for the correct version of GemStone/S 64 Bit and for the appropriate platform. The keyfile must be located where GemStone can find it on startup:

- A file specified by the KEYFILE configuration parameter in the configuration file used by the stone. This is not set by default, but may be defined to read a keyfile with any name in any location.
- `$GEMSTONE/sys/gemstone.key`
- `$GEMSTONE/sys/community.starter.key`

Licensed Customer key file

Licenced customers can email keyfiles@gemtalksystems.com or contact GemTalk Technical Support to request a keyfile for version 3.5 for their platform or platforms. Keyfiles for previous versions of GemStone are not valid with v3.5.

Community key file

The GemStone distribution includes a community key file, `community.starter.key`., with product and system limits per the Community and Web Edition License. See https://gemtalksystems.com/licensing for details on the license terms.

If you do not install a custom keyfile, this starter keyfile will be used instead.

Installing a keyfile

To specify the location and name of the keyfile using the KEYFILE configuration parameter, edit the configuration file that will be used by the v3.5 stone to include the location and name of the keyfile.

You may also put the keyfile in the default location, `$GEMSTONE/sys/gemstone.key`. This requires modifying the write permissions of the `$GEMSTONE/sys` directory; ensure you change this back to not writable, after this update.

Verify TCP/IP

To run GemStone, TCP/IP must be functioning, even if your machine is not connected to a network.

Verify that TCP/IP networking software is functioning:

› **`/bin/ping`** *hostname*

where *hostname* is the name of your machine. If **ping** responds with statistics, TCP/IP is functioning.

Define the NetLDI Service

The NetLDI service, by default `gs64ldi`, should be defined in your system services database. A NetLDI is required for certain kinds of local and remote sessions to log into GemStone, and if it cannot be resolved by name, you must refer to it by port number. For clients on remote machines, the same NetLDI service name and port number must be defined on the remote machines as well as the main host.

If you are upgrading from a previous version, you may need to keep the NetLDI for that version running. In this case, select a distinct name and port for the NetLDI for GemStone/S 64 Bit 3.5.

1. Determine whether the `gs64ldi` service is already defined. How to do this will depend on how your system is set up. The GemStone distribution includes an executable that will allow you to do this:

   ```
   unix› $GEMSTONE/install/getservbyname gs64ldi
   s_name=gs64ldi s_port = 50377 s_proto = tcp
   ```

   If `gs64ldi` is defined, skip the rest of this procedure and continue with the installation at "Run the Installation Script" on page 16.

   If it is not defined, continue performing this procedure.

2. Add an entry similar to the following to the system services database:

   ```
   gs64ldi 50377/tcp #GemStone/S 64 Bit 3.5
   ```

   Choose a port number that is not being used by another service. The port number should be in the range 49152 <= port <= 65535, to confirm to IANA standards (http://www.iana.org/assignments/port-numbers).

3. If several machines will be running GemStone, have the UNIX system administrator update the system services database for each machine. This includes Windows client machines as well as UNIX nodes. Note that the port number must be the same for every machine.

Run the Installation Script

1. Log in as root.

   Although you can complete the installation as a non-root user, we do not recommend this when installing a multi-user system. During installation, GemStone system security is established through file permissions and process attributes. To ensure that the installation is successful, you must install as root. See the *System Administration Guide* for details on setting up GemStone server file security.

2. Invoke the installation script from the `install` subdirectory:

```
unix> cd $GEMSTONE/install
unix> ./installgs
```

`installgs` is an interactive script that analyzes your system configuration and makes suggestions to guide you through installing GemStone on your machine.

NOTE

You can usually terminate execution of the installation script with Ctrl-C without risk to your files. When it is not safe to do so, the message `Please do not interrupt` *appears on the screen. If this happens, wait for the message* `now it is OK to interrupt` *before you interrupt the script. You can run the script again from the beginning as many times as necessary.*

3. Log out as root

   After running installgs, log out as user root. Further work, such as setting the TimeZone, is done as the GemStone administrative user.

Decisions to Make During Installation

During installation, you are asked several questions. The entire installation dialog is not reproduced here, but the main points are addressed. Some questions may not be asked, depending on answers to previous questions.

Whenever you are asked to answer "yes" or "no," answer with y or n. When the script offers a default answer in square brackets (such as "`[y]`"), press Enter to accept the default.

Do you want the installation script to set up directories for server lock files and NetLDI logs?

The default location for server lock files and NetLDI log files is `/opt/gemstone`, although for compatibility with earlier products `/usr/gemstone` is used only if it exists. If the environment variable GEMSTONE_GLOBAL_DIR is defined to point to a valid directory, this overrides the default server lock files and log file location; however, all Gemstone processes that will interact on this machine must have this environment variable set to the same directory.

If these directories do not exist, the installation script offers to create `/opt/gemstone` and the subdirectories `locks` and `log`. Then, the script offers to set access (770) to these directories.

If you answer no to creating the directories, you must create them (or provide a symbolic link) before starting the server.

Do you want the installation script to set the owner and group for all the files in the GemStone distribution?

If you answer **yes**, the script will prompt you for the owner and group you want to use. Refer to Chapter 1 of the *System Administration Guide* for more information about setting owner and group permissions.

If you answer **no**, the permissions will remain the same as when the files were extracted from the distribution media.

Do you want the installation script to protect the repository file?

The default, which we recommend, gives only the owner read and write access (600) through ordinary UNIX commands. Other users can read and write the repository through a GemStone session. If you choose not to protect the repository, the setuid bit is cleared from all executables, which causes them to run under ownership of the user who invokes them.

Default: Set the repository permission to 600, and leave the setuid bit applied.

Allow NetLDI to Run as Root?

Do you want the installation script to allow non-root users to start a NetLDI that runs as root?

The NetLDI is a network server that permits remote processes to interact with the repository. There are two ways to set up a NetLDI so that it can provide services to all GemStone users: it can run as root, or it can run in guest mode with a captive account.

- To run NetLDIs as root, accept the default "yes" response. Ownership of the NetLDI executable is changed to root, and the setuid bit is set. Any GemStone user will be able to start a NetLDI process that is accessible to all GemStone users because it will always run as root. For certain services, users will need to authenticate themselves by supplying a password. Alternatively, answer "no" but log in as root before starting the NetLDI.

  If the NetLDI uses a port number less than 1024, it must run as root.

- To run NetLDIs in guest mode with a captive account, answer "no" to the prompt, because those modes are not permitted if the NetLDI runs as root. "Guest mode" means that GemStone users do not have to supply a UNIX password to use NetLDI services. The "captive account" is an account that owns all processes the NetLDI starts; typically, it is the GemStone administrative account that owns the files. You must start the NetLDI while logged in as that account.

Default: Change ownership of the `netldi` executable to root, and set its setuid bit.

Set up an Extent?

Do you want the installation script to set up an extent now?

GemStone is distributed with a read-only copy of the initial repository in `$GEMSTONE/bin/extent0.dbf`. Before you can start GemStone, this file must be copied to a suitable location and made writable. The script offers to copy the file to its default location of `$GEMSTONE/data`.

If you are a new GemStone user, we recommend that you answer y. If you are an existing GemStone user, you might prefer to answer n, then copy the extent to a different location yourself. (If you choose a location other than the default, you must edit your configuration file before starting GemStone. For information, see the *System Administration Guide*.)

Default: Place a writable copy of `extent0.dbf` in `$GEMSTONE/data`.

Start a NetLDI?

Do you want the installation script to start a NetLDI?

If you prefer, you can start these processes manually at any time.

Almost every host needs a NetLDI. You must start a NetLDI when the Stone repository monitor or Gem session processes will run on this machine.

You can start a NetLDI that runs as root by answering yes to this prompt and the confirmation that follows. However, if you want to start the NetLDI in guest mode with a captive account, you must do that after completing the installation. For more information about guest mode with captive account, see Chapter 3 of the *System Administration Guide.*

Default: Do not start a NetLDI at this time.

Start an Object Server?

As root, you cannot start an object server, but the script offers to start one as another user. You will start the server later in the installation, so answer no.

Default: Do not start an object server at this time.

Change Passwords for Administrative Accounts

GemStone comes with a number of built-in System user accounts, which are needed to perform administrative operations (such as adding application user accounts).

- The **DataCurator** account is used to perform system administration tasks.
- The **SystemUser** account ordinarily is used only for performing GemStone system upgrades.
- The **GcUser** account is used by the garbage collection task, which runs automatically as a separate login.

The initial password for these administrative accounts is `swordfish`.

Access to each of these accounts should be restricted; you should always change the passwords for these accounts, to provide basic security for your application.

The chapter entitled User Accounts and Security in the *System Administration Guide* tells you how to change the passwords

Add Users

For each of the users in your system, you should establish GemStone accounts, which involves creating an individual UserProfile in GemStone.

The chapter entitled "User Accounts and Security" in the *System Administration Guide* provides information on how create accounts for GemStone users, and the options for authentication. This task can be done by executing Smalltalk code, or using GemBuilder for Smalltalk tools. See the *GemBuilder for Smalltalk Users's Guide* for information on the GUI tools in GemBuilder.

Users must set environment

After GemStone/S 64 Bit 3.5 has been installed, you should notify each person who will be using GemStone about the installation, and explain how to setup their environment.

Each user must:

- Set the environment variable GEMSTONE to the *full pathname* (starting with a slash) of the GemStone/S 64 Bit 3.5 directory.
- update their path to include the $GEMSTONE/bin directory.
- Optionally, update the man path (MANPATH variable) to include the $GEMSTONE/doc directory. GemStone provides man pages for utility functions.

These last two steps can be done using scripts that are part of the GemStone distribution. The directory $GEMSTONE/bin contains the files gemsetup.sh and gemsetup.csh, which define the GemStone environment for users by modifying the PATH and MANPATH variables to include $GEMSTONE/bin and $GEMSTONE/doc, respectively.

For example:

```
> export GEMSTONE=installdir
> export PATH=$GEMSTONE/bin:$PATH
> export MANPATH=$MANPATH:$GEMSTONE/doc
```

If the user will use GemStone frequently, consider adding these steps to the login shell initialization file.

Install the default TimeZone

GemStone/S 64 Bit is shipped with a default time zone of US Pacific. If you are in another Time Zone, edit the file installtimezone.txt in the GemStone upgrade directory, then file it in as SystemUser.

Further Configuration and Administration

This chapter has guided you through installation of GemStone/S 64 Bit 3.5, with the objective of getting a simple, default configuration up and running.

The next chapters explain the process of upgrading a previous version of GemStone/S 64 Bit to version 3.5; and Chapter 5 provides information on GemBuilder for Smalltalk.

For more information and details on customizing your GemStone object server, Gem client processes, and setting up distributed configurations, see the *System Administration Guide*.

Chapter

2 Upgrading from GemStone/S 64 Bit 3.3.x or 3.4.x

This chapter describes how to upgrade an existing GemStone/S 64 Bit 3.3.x or 3.4.x installation to GemStone/S 64 Bit version 3.5.

For upgrading from GemStone/S 64 Bit versions 3.2.x, which require extra steps such as recompilation, see Chapter 3, starting on page 27.

To upgrade from GemStone/S 64 Bit version 3.1.x and earlier, you must first upgrade to 3.3.x or 3.4.x version, and then upgrade to v3.5.

If you are using GemBuilder for Smalltalk (GBS), you also need to upgrade the client libraries that are used by GBS. You may also need to upgrade your version of GBS; versions of GBS earlier than 8.4 or 5.4.5 are not supported with v3.5. See Chapter 5 for details.

Keyfiles

New keyfiles are required with GemStone/S 64 Bit version 3.5; keyfiles for GemStone/S 64 Bit 3.4.x or earlier will not work with v3.5. To acquire a keyfile for version 3.5, email keyfiles@gemtalksystems.com, or contact GemStone Technical Support, preferably providing your existing keyfile.

Keyfiles also manage access to GemConnect and GemBuilder for Java, and the new X509-Secured GemStone feature. If you are using these add-on products, you must use a keyfile with the appropriate permissions.

Upgrade Procedure

The following list summarizes the steps necessary to perform the upgrade to GemStone/S 64 Bit v3.5.

- Prior to Upgrade in existing application. 22
- Prepare for Upgrade 22
- Perform the Upgrade 24
- Post-upgrade Application Code Modifications 25
- Backup repository and configure GBS 26

NOTE

*The following instructions use the version number **3.4.3** to refer to the version you are upgrading from, and version number **3.5** indicate the target version you are upgrading to.*

Prior to Upgrade in existing application

1. Check for use of deprecated methods

Verify that your application does not invoke any methods that were deprecated in previous releases, by enabling error or logging on deprecation in your existing repository. Deprecated methods are subject to removal in major releases; finding them before upgrading allows the deprecation messages to provide replacement instructions.

For details on finding deprecated methods, refer to the *Programming Guide for GemStone/S 64 Bit*.

2. File out modifications to GemStone classes

File out any modifications or additions you made to GemStone/S 64 Bit kernel class methods. For more information about fileout, see the *GemStone/S 64 Bit Topaz Programming Environment*.

You will need to carefully compare these changes with GemStone/S 64 Bit 3.5 kernel methods, and refer to the *Release Notes* for version 3.5 to determine whether your changes are still necessary or appropriate.

CAUTION

Any changes that you have made to the GemStone/S 64 Bit kernel classes will be lost during upgrade; you MUST file these out in order to preserve the changes in version 3.5.

Prepare for Upgrade

1. Install and configure GemStone/S 64 Bit 3.5

Install GemStone/S 64 Bit 3.5 to a new installation directory, separate from the installation directory for version 3.4.3, as described in Chapter 1, starting on page 7.

Configure GemStone/S 64 Bit 3.5 the way you expect to use it — that is, with the appropriate extent locations and sizes.

If you copy the configuration files from your previous version to the version 3.5, be sure to review any changes in configuration parameters to determine if changes are needed.

You should ensure that adequate space is available for extents, transaction logs, and a backup during the upgrade. You must provide space for the extents and transaction logs for both repositories, the old and the new.

2. Reset SystemUser password

Log in to the version 3.4.3 system as a user with OtherPassword privilege, such as DataCurator, and reset the SystemUser password to 'swordfish':

```
topaz 1> printit
(AllUsers userWithId: #SystemUser) password: 'swordfish' .
System commitTransaction.
%
```

The upgrade script logs in with the SystemUser account and the default password, and resets the password for DataCurator and GcUser.

3. Stop user activity

Log in to the version 3.4.3 system as a user with SessionAccess and SystemControl privileges, such as DataCurator, and halt all user activity on the repository.

```
topaz 1> printit
System stopUserSessions.
%
```

4. Shut down the repository

You may now shut down the Stone. At the UNIX command line:

```
> stopstone stone343
```

where *stone343* is the name of the version 3.4.3 stone on this machine. The repository must be cleanly shut down to avoid needing recovery when it is restarted with the new version's executables.

5. Set up the version 3.5 environment.

Set the environment variables required for the upgrade. For example:

```
> export GEMSTONE=InstallDir35
> export PATH=$GEMSTONE/bin:$PATH
> export upgradeLogDir=tempDir
```

where *InstallDir35* is the GemStone/S 64 Bit version 3.5 installation and *tempDir* is a temporary directory for which you have write permission.

6. Copy extent files

Copy your version 3.4.3 extent files into the location specified by the v3.5 configuration file option `DBF_EXTENT_NAMES`:

a. Identify the complete set of extent files that are used by your 3.4.3 stone. This can be found by examining the configuration file for the version 3.4.3 repository, looking for the last entry for `DBF_EXTENT_NAMES`.

b. The target location is the setting for `DBF_EXTENT_NAMES` in the version 3.5 installation. Copy each of these extent files to the target location.

For example:

```
> cp InstallDir343/data/extent0.dbf InstallDir35/data
> cp InstallDir343/data/extent1.dbf InstallDir35/data
> cp InstallDir343/data/extent2.dbf InstallDir35/data
```

Before upgrading, ensure that there are no transaction logs from a previous version of GemStone/S 64 Bit in any of the transaction log locations specified in the configuration file that will be used by version 3.5.

Perform the Upgrade

1. Start the Stone

Start the 3.5 Stone on the 3.4.3 extents you just copied:

```
> startstone stoneName35
```

2. Upgrade image

Ensure you are in a directory to which you have write permission, and run the upgrade script.

The upgrade is performed by the script `upgradeImage`. This script has optional switches to specify the stone name and to set to size of the GEM_TEMPOBJ_CACHE_SIZE used for the upgrade process.

`upgradeImage [-h] [-c` *cacheSize*`] [-s` *stoneName*`]`

`-h` prints this usage information.

`-c` *cacheSize* sets the size of the GEM_TEMPOBJ_CACHE_SIZE; if this is not used, the script will default to use a value of 100000.

`-s` *stoneName* sets the name of the running stone to upgrade; if this option is not used, the script will default to **gs64stone.**

For example,

```
> upgradeImage -s stoneName35
```

The script will prompt you to press the return key to begin.

The script invokes subordinate scripts to complete the upgrade. The upgrade process will take some time. You can examine the progress, if desired, by examining the file `$upgradeLogDir/upgradeImage.out`.

The script should complete with the message:

```
Upgrade completed. No errors detected.
```

If not, please preserve the Stone log file and the contents of `$upgradeLogDir`. Contact your internal GemStone support person or GemStone Technical Support.

3. Restore System Account passwords

Log in to GemStone/S 64 Bit version 3.5 as DataCurator or SystemUser, and change the password for SystemUser, DataCurator, and GcUser to a secure password, such as the passwords used for these accounts in v3.4.3. For example:

```
topaz 1> run
(AllUsers userWithId: 'SystemUser') password: '343Password'.
(AllUsers userWithId: 'GcUser') password: '343Password'.
(AllUsers userWithId: 'DataCurator') password: '343Password'.
System commitTransaction
%
```

where *343Password* is the account password used in version 3.4.3.

GsDevKit Upgrade

If you are using the open-source Development Kit for GemStone/S 64 Bit (GsDevKit, also referred to as Seaside or GLASS), you will need to perform another step to upgrade your GsDevKit image. This step upgrades the GsDevKit base code, and you will also need to reload your application code.

For details, see Chapter 4, starting on page 35.

When you have completed the GsDevKit upgrade, continue with the upgrade process and perform the following steps.

Post-upgrade Application Code Modifications

1. Reinstall any other GemStone products that modify kernel classes.

If you use GemConnect or GemBuilder for Java, you must reinstall the appropriate version of these products into your repository at this time.

To install, use the procedure in the *Installation Guide* for that product.

2. File in Kernel class changes

If you have modified any kernel class methods of the previous version or if you have added methods to kernel classes, carefully compare your changes with the changes in version 3.5 to see whether your changes are still necessary or appropriate. Carefully review the *Release Notes* for each intervening version, as well as examining code in the image.

If the kernel class changes are still applicable, file in the changes, verify that errorcount is 0, and commit.

3. Resort for ICU-based collation

If the repository has been progressively upgraded through any 3.1.x version, and contains data structures that were built in v3.1.x that depend on ICU collation order or encoding (that is, they involve Unicode strings, or traditional strings if the repository is in Unicode Comparison Mode, including GsDevKit application), then these collections need to be sorted and indexes rebuilt to avoid the (small) risk of lookup failures. If resort/rebuilt has been done in 3.2.x or 3.3.x it does not need to be done again.

The ICU libraries that provide Unicode based collation are versioned as the Unicode standard changes. Since most changes in the standard are not important for most

applications, by default applications continue to use their original version of ICU library (either the version in which the repository originated, or as determined during upgrade). However, 3.1.x applications used a much older ICU version and cannot be handled automatically.

The version of the ICU shared libraries that will be loaded by session is defined by the global IcuLibraryVersion. The IcuLibraryVersion after upgrade will be the same as an existing value of IcuLibraryVersion before upgrade, and if that key is not present, it is determined based on the originating GemStone version.

Backup repository and configure GBS

1. Make backup

At this point, you should create a full backup of the upgraded repository.

2. Configure GBS

If you are using GBS clients, ensure you are running a supported version of GBS and client Smalltalk. You must use GBS version 8.4 or later for VW, or GBS 5.4.5 or later for VA, to connect to a GemStone/S 64 Bit v3.5 repository.

Configure GBS to use the version 3.5 client libraries. Depending on the GBS version you are upgrading from, the required libraries, library naming conventions, and the process GBS uses to identify the correct library to load may have changed.

See Chapter 5, 'Configuring GBS for GemStone/S 64 Bit' for details. If your GBS clients run on a different platform than your GemStone server, refer to the *Installation Guide* for that platform.

Chapter

3 Upgrading from 3.2x

This chapter describes how to upgrade an existing GemStone/S 64 Bit 3.2.x installation to GemStone/S 64 Bit version 3.5. If you are upgrading from version 3.3.x or later, which do not require recompile, see Chapter 2 on page 21. Upgrading from earlier versions requires an intermediate upgrade to version 3.2.x or preferably a later version.

The compiled method bytecodes changed between v3.2 and v3.3, and the upgrade process requires that you file in all application source code so it can be recompiled, and recompile all persistent blocks. Alternatively, you may iterate and recompile all methods in your application.

If you are using GemBuilder for Smalltalk (GBS) or other clients-only nodes in your configuration, you also need to upgrade the client libraries on that node.

You also need to upgrade your version of GBS; versions of GBS earlier than 8.3 or 5.4.4 are not supported with v3.5. See Chapter 5 for supported versions of GBS for use with GemStone/S 64 Bit 3.5, and instructions on installing updated client libraries.

Keyfiles

New keyfiles are required with GemStone/S 64 Bit version 3.5; keyfiles for GemStone/S 64 Bit 3.4.x or earlier will not work with v3.5. To acquire a keyfile for version 3.5, email keyfiles@gemtalksystems.com, or contact GemStone Technical Support, preferably providing your existing keyfile.

Keyfiles also manage access to GemConnect and GemBuilder for Java, and the new X509-Secured GemStone feature. If you are using these add-on products, you must use a keyfile with the appropriate permissions.

Upgrade Procedure

The following list summarizes the steps necessary to perform the upgrade to GemStone/S 64 Bit version 3.5.

- Prior to Upgrade in existing application. 28
- Prepare for Upgrade 29
- Perform the Upgrade 30
- Post-upgrade Application Code Modifications 31
- Backup repository and configure GBS 34

NOTE

*The following instructions use the version number **3.2.16** to refer to the version you are upgrading from, and version number **3.5** indicate the target version you are upgrading to. The process is the same when upgrading from any 3.x repository, and upgrading to any 3.2.x versions for which this Installation Guide applies.*

Prior to Upgrade in existing application

1. Check for use of deprecated methods

Verify that your application does not invoke any methods that were deprecated in previous releases, by enabling error or logging on deprecation in your existing repository. Deprecated methods are subject to removal in major releases; finding them before upgrading allows the deprecation messages to provide replacement instructions.

For details on finding deprecated methods, refer to the *Programming Guide for GemStone/S 64 Bit*.

2. File out your application code

If you do not already have source code for your application stored externally to the GemStone repository in a code management system, it is recommended to file out all application source code. Filein of application code is used to recompile all methods. You may also write code to manually recompile methods in all classes; see "Recompile application code" on page 32 for details.

You should confirm that the format of your filed out code does not create new versions of your application classes on filein.

GemStone supports multiple versions of the same class, but tools operate on the most recent version of the classes. If you have instance of older versions of your applications classes that have not been migrated to the latest version, these class versions will not be upgraded by filein. We recommend that you migrate all instances to the most recent version of your application classes.

3. File out modifications to GemStone classes

File out any modifications or additions you made to GemStone/S 64 Bit kernel class methods. For more information about fileout, see the *GemStone/S 64 Bit Topaz Programming Environment*.

You will need to carefully compare these changes with GemStone/S 64 Bit 3.5 kernel methods, and refer to the *Release Notes* for version 3.3 and all release notes after the

version you are upgrading from, to determine whether your changes are still necessary or appropriate. For a listing of Release Notes, see GemStone/S 64 Bit Release History.

CAUTION

Any changes that you have made to the GemStone/S 64 Bit kernel classes will be lost during upgrade; you MUST file these out in order to preserve the changes in version 3.5.

Prepare for Upgrade

1. Install and configure GemStone/S 64 Bit 3.5

Install GemStone/S 64 Bit 3.5 to a new installation directory, separate from the installation directory for version 3.2.16, as described in Chapter 1, starting on page 7.

Configure GemStone/S 64 Bit 3.5 the way you expect to use it — that is, with the appropriate extent locations and sizes.

If you copy the configuration files from your previous version to the version 3.5, be sure to review any changes in configuration parameters to determine if changes are needed.

You should ensure that adequate space is available for extents, transaction logs, and a backup during the upgrade. You must provide space for the extents and transaction logs for both repositories, the old and the new.

2. Reset SystemUser password

Log in to the version 3.2.16 system as a user with OtherPassword privilege, such as DataCurator, and reset the SystemUser password to 'swordfish':

```
topaz 1> printit
(AllUsers userWithId: #SystemUser) password: 'swordfish'.
System commitTransaction.
%
```

The upgrade script logs in with the SystemUser account and the default password, and resets the password for DataCurator and GcUser.

3. Stop user activity

Log in to the version 3.2.16 system as a user with SessionAccess and SystemControl privileges, such as DataCurator, and halt all user activity on the repository.

```
topaz 1> printit
System stopUserSessions.
%
```

4. Shut down the repository

You may now shut down the Stone. The repository must be cleanly shut down to avoid needing recovery when it is restarted with the new version's executables. For example:

```
% stopstone stone3216
```

where *stone3216* is the name of the version 3.2.16 stone on this machine.

5. Set environment variables for version 3.5

Set the environment variables required for the upgrade.

```
> export GEMSTONE=InstallDir35
> export PATH=$GEMSTONE/bin:$PATH
> export upgradeLogDir=tempDir
```

where *InstallDir35* is the GemStone/S 64 Bit version 3.5 installation and *tempDir* is a temporary directory for which you have write permission.

NOTE

Use a separate log directory for each repository you convert.

6. Copy extent files

Copy your version 3.2.16 extent files into the location specified by the v3.5 configuration file option `DBF_EXTENT_NAMES`:

a. Identify the complete set of extent files that are used by your 3.2.16 stone. This can be found by examining the configuration file for the version 3.2.16 repository, looking for the last entry for `DBF_EXTENT_NAMES`.

b. The target location is the setting for `DBF_EXTENT_NAMES` in the version 3.5 installation. Copy each of these extent files to the target location.

For example:

```
> cp InstallDir3216/data/extent0.dbf InstallDir35/data
> cp InstallDir3216/data/extent1.dbf InstallDir35/data
> cp InstallDir3216/data/extent2.dbf InstallDir35/data
```

Before upgrading, ensure that there are no transaction logs from a previous version of GemStone/S 64 Bit in any of the transaction log locations specified in the configuration file that will be used by version 3.5.

Perform the Upgrade

1. Start the Stone

Start the 3.5 Stone on the 3.2.16 extents you just copied:

```
% startstone stoneName35
```

2. Upgrade image

Ensure you are in a directory to which you have write permission, and run the upgrade script.

The upgrade is performed by the script `upgradeImage`. This script has optional switches to specify the stone name and to set to size of the GEM_TEMPOBJ_CACHE_SIZE used for the upgrade process.

`upgradeImage [-h] [-c` *cacheSize*`] [-s` *stoneName*`]`

`-h` prints this usage information.

`-c` *cacheSize* sets the size of the GEM_TEMPOBJ_CACHE_SIZE; if this is not used, the script will default to use a value of 100000.

`-s` *stoneName* sets the name of the running stone to upgrade; if this option is not used, the script will default to **gs64stone.**

For example,

```
% upgradeImage -s stoneName35
```

The script will prompt you to press the return key to begin.

The script invokes subordinate scripts to complete the upgrade. The upgrade process will take some time. You can examine the progress, if desired, by examining the file `$upgradeLogDir/upgradeImage.out`.

The script should complete with the message:

```
Upgrade completed. No errors detected.
```

If not, please preserve the Stone log file and the contents of `$upgradeLogDir`. Contact your internal GemStone support person or GemStone Technical Support.

3. Restore System Account passwords

Log in to GemStone/S 64 Bit version 3.5 as DataCurator or SystemUser, and change the password for SystemUser, DataCurator, and GcUser to a secure password, such as the passwords used for these accounts in v3.2.16. For example:

```
topaz 1> printit
(AllUsers userWithId: 'SystemUser') password: '3216Password'.
(AllUsers userWithId: 'GcUser') password: '3216Password'.
(AllUsers userWithId: 'DataCurator') password: '3216Password'.
System commitTransaction
%
```

where *3216Password* is the account password used in version 3.2.16.

GsDevKit Upgrade

If you are using the open-source Development Kit for GemStone/S 64 Bit (GsDevKit, also referred to as Seaside or GLASS), you will need to perform another step to upgrade your GsDevKit image. This step upgrades the GsDevKit base code, and you will also need to reload your application code.

For details, see Chapter 4, starting on page 35.

When you have completed the GsDevKit upgrade, continue with the upgrade process and perform the following steps.

Post-upgrade Application Code Modifications

1. Reinstall any other GemStone products that modify kernel classes.

If you use GemConnect or GemBuilder for Java, you must reinstall the appropriate version of these products into your repository at this time.

To install, use the procedure in the *Installation Guide* for that product.

2. File in Kernel class changes

If you have modified any kernel class methods of the previous version or if you have added methods to kernel classes, compare your changes with the changes in version 3.5 to see whether your changes are still necessary or appropriate. Carefully review the *Release Notes* for each intervening version, as well as examining code in the image.

If the kernel class changes are still applicable, file in the changes, verify that errorcount is 0, and commit.

3. Recompile application code

The upgrade process requires all executable methods to be recompiled. You should have fileins available, from Step 2. on page 28. Alternatively, you may iterate classes in your image and recompile each one. If you are reusing scripts that managed recompile for the 2.x conversion, verify that the configuration parameter `GemConvertArrayBuilder` is not set.

GsDevKit environments will not need to perform this step; application code reload is done as part of the upgrade described in Chapter 4.

a. Recompile by filein

File in all development and application code. Verify that errorcount is 0, and commit.

If you have instances of previous versions of classes, these old class versions will not be recompiled by this process. You should ensure that all application instances are migrated to current class versions before conversion, or manually recompile instances of older class versions.

b. Recompile within image

To recompile classes without filein, for each class in your repository execute

Class `recompileAllMethods`

You should ensure that you do not miss any classes. If you have instance of older versions of your classes, you will need to recompile methods on these older versions. For example:

```
class classHistory do: [:aClassVersion |
      aClassVersion recompileAllMethods]
```

Recompile is required for 3.2.x methods to be executable in v3.5. Attempting to execute methods that have not been recompiled will result in errors.

4. Recompile source code in persistent blocks

The compiled code in persistent blocks also requires recompile before it can be executed. All persistent executable blocks will need to be recompiled as part of upgrading.

c. Application-specific stored blocks

If your application stores persistent blocks, you will have to locate and recompile all such blocks before they can be executed.

d. SortBlocks in SortedCollections

To recompile the sortBlocks in persistent SortedCollections in your application, you may run the `postconv` script. This script only converts simple blocks; if your

SortedCollection blocks are not simple (such as referring to method context), they cannot be automatically recompiled.

`postconv [-c <numCacheWamerGems>][-h][-s <stoneName>] [-r] [-n <numberOfSessions>] [-t <tempObjCacheSize>] [-u <userId>]`

`-c <numCacheWamerGems>` specifies the number of cache warmer threads in a single gem to load the object table into the shared cache before starting post-conversion. If not specified, no cache warming is done.
`-h` prints this usage information.
`-s <stoneName>` sets the name of the running stone to scan; if this option is not used, the script uses gs64stone**.**
`-n <numberOfSessions>` specifies the number of parallel sessions which will convert the instances of SortedCollection and its subclasses. By default, use one session.
`-r` specifies to reuse an existing version of `$upgradeLogDir/AllSortedCollections.bms`, if it exists. This file contains the OOPs of all instances of SortedCollections and its subclasses. By default, the existing file is deleted and a new one created.
`-t <tempObjCacheSize>` sets the size of GEM_TEMPOBJ_CACHE_SIZE in KB; by default, 20000.
`-u <userId>` is the UserId whose SymbolList includes all subclasses of SortedCollection, for which instance's sortBlocks will be converted. If not specified, defaults to SystemUser

For example,

`% postconv -s` *stoneName35*

The `postconv` script will write progress messages to stdout. When it completes, it will report:

`postconv[INFO]: Congratulations!` *NNN* `SortedCollections were successfully converted. No errors were detected.`

If postconv reports errors, the results include information on the specific sortBlocks that failed recompile, such as those for complex sort blocks. These will need manual repair. Contact GemTalk Technical Support for assistance.

5. Resort for ICU-based collation

If the repository has been progressively upgraded through any 3.1.x version, and contains data structures that were built in v3.1.x that depend on ICU collation order or encoding (that is, they involve Unicode strings, or traditional strings if the repository is in Unicode Comparison Mode, including GsDevKit application), then these collections need to be sorted and indexes rebuilt to avoid the (small) risk of lookup failures. If resort/rebuild has been done in 3.2.x or 3.3.x it does not need to be done again.

The ICU libraries that provide Unicode based collation are versioned as the Unicode standard changes. Since most changes in the standard are not important for most applications, by default applications continue to use their original version of ICU library (either the version in which the repository originated, or as determined during upgrade). However, 3.1.x applications used a much older ICU version and cannot be handled automatically.

The version of the ICU shared libraries that will be loaded by session is defined by the global IcuLibraryVersion. The IcuLibraryVersion after upgrade will be the same as an existing value of IcuLibraryVersion before upgrade, and if that key is not present, it is determined based on the originating GemStone version. Details may be in the log file for the SymbolGem for the upgrade.

6. Regenerate cached instances of PetitParser classes

Instance of PetitParser classes (classes with names that begin with PP) are not automatically converted to the new class versions. If you are upgrading from v3.2x and using PetitParser classes directly, and you have persistent instances, you should regenerate these instances.

Backup repository and configure GBS

1. Make backup

At this point, you should create a full backup of the upgraded repository.

2. Configure GBS

If you are using GBS clients, ensure you are running a supported version of GBS and client Smalltalk. You must use GBS version 8.3 or later for VW, or GBS 5.4.4 or later for VA, to connect to a GemStone/S 64 Bit v3.5 repository.

Configure GBS to use the version 3.5 client libraries. Depending on the GBS version you are upgrading from, the required libraries, library naming conventions, and the process GBS uses to identify the correct library to load may have changed.

See Chapter 5, 'Configuring GBS for GemStone/S 64 Bit' for details. If your GBS clients run on a different platform than your GemStone server, refer to the *Installation Guide* for that platform.

Chapter

4 Upgrading GLASS/GsDevKit Applications

This chapter describes the additional upgrade step that applies when upgrading an application that is using variants of the open-source Development Kit for GemStone/S 64 Bit (GsDevKit, referred to also as also as GLASS or Seaside) to GemStone/S 64 Bit v3.5. The term GsDevKit is used to collectively refer to any of these environments.

The process described here can be used to upgrade repositories using GLASS, GLASS1, the older GsDevKit environment, and tODE. There are a number of possible configurations and there may be additional setup required for some environments. For more background on these environments, see https://github.com/GsDevKit/GsDevKit_upgrade/blob/master/README.md#upgrading-glassgsdevkit-applications-to-gemstone-350

If you are using the most recent version, from github.com/GsDevKit/GsDevKit_home, then you may use the upgrade scripts provided there to perform the entire upgrade, rather than using the instructions in this *Installation Guide*.

The complete process for upgrading includes the GemStone standard upgrade, followed by additional GsDevKit upgrade. Upgrading GemStone is described in earlier chapters of this *Installation Guide*; Chapter 2 for version 3.3.x and 3.4.x, and Chapter 3 for upgrade from 3.2.x. You will need to follow the steps in that chapter, which will note the point at which the GsDevKit upgrade takes place.

Upgrade Procedure

The GsDevKit upgrade occurs partway through a standard GemStone upgrade.

To upgrade a GsDevKit/GLASS application:

- First, install the new version if GemStone, and upgrade your repository, according to the instructions in Chapter 2 or Chapter 3.
- After the **upgradeImage** step, you will upgrade the GsDevKit application as described in this chapter.
- After the GsDevKit upgrade completes, continue with the remaining steps of the GemStone upgrade in Chapter 2 or Chapter 3.

1. Ensure that GemStone 3.5 is installed and your repository upgraded

You must first follow install version 3.5 and follow the instructions in Chapter 2 or Chapter 3, before you can upgrade your GsDevKit application. These instructions will let you know at which point you perform the GsDevKit upgrade.

You should also have confirmed that your application code has been updated as required. We recommend that you install your application code in a test GemStone/S 64 Bit v3.5 repository, and verify that your code is working correctly, making changes as necessary. Do this prior to upgrading the data in your application, to ensure the upgrade process will go smoothly.

2. If necessary, customize the upgrade instructions

There are many ways a GsDevKit or GLASS application may be built, and a variety of packages that can be loaded. The **upgradeSeasideImage** script will upgrade a hypothetical standard installation, but there may be customizations required in specific cases.

The upgrade is performed by an upgrade script. By default, this is `$GEMSTONE/upgrade/createGsDevKit_upgrade.topaz`.

The default upgrade script is a file containing topaz commands for the upgrade.

Example 4.1 Default upgrade instructions in createGsDevKit_upgrade.topaz

```
run
  UserGlobals
   at: #GsDevKit_Image_Upgrade
       put: (GsuAbstractGsDevKitUpgrade
          upgradeUserName: SeasideUpgradeUser).
System commitTransaction
%
```

In this script, SeasideUpgradeUser is an internal global that by default resolves to DataCurator.

Customizing upgrade

To create customized upgrade instructions, make a copy of this file, edit the copy, and pass the path to your customized upgrade script file as an argument to the **upgradeSeasideImage** script.

The following example shows a customized file.

Example 4.2 Example customized upgrade

```
run
UserGlobals
   at: #GsDevKit_Image_Upgrade
   put: ((GsuAbstractGsDevKitUpgrade
            upgradeUserName: 'DataCurator'
            upgradeSymbolDictName: #UserGlobals)
      bootstrapApplicationLoadSpecs: {
      { 'Metacello' .
         'github://dalehenrich/metacello-work:master/repository' } .
      { 'GLASS1' .
         'github://glassdb/glass:master/repository' .
         #( 'default' 'Base' 'Announcements') } .
      { 'Seaside3' .
         'github://SeasideSt/Seaside:master/repository' .
         #( 'CI') }
      } ).
System commitTransaction
%
```

Further information about LoadSpecs is provided in the comment for `$GEMSTONE/upgrade/createGsDevKit_upgrade.topaz`.

You may also refer to the example scripts on github, at github.com/GsDevKit/GsDevKit_upgrade/tree/master/bin.

3. Perform the Upgrade

The GsDevKit upgrade is performed by the script **upgradeSeasideImage**, which is located in the `$GEMSTONE/seaside/bin` subdirectory.

Prior to executing **upgradeSeasideImage**, if you need to do a customized upgrade, you should have set up the upgrade script as described above. This is provided a argument to this script.

upgradeSeasideImage [-c *tempObjCacheSize* **] [-s** *stoneName* **] [-u** *gemstoneUser* **]**
[-p *password* **] [-P** *pathToUpgradeScript* **]**

-c *tempObjCacheSize*
set the size of the GEM_TEMPOBJ_CACHE_SIZE; if omitted, default is 100000.

-s *stoneName*
set the name of the running stone to upgrade; if omitted, default is **gs64stone.**

-u *gemstoneUser*
specify the GemStone user name, if seaside was installed as a user other than DataCurator. If omitted, defaults to DataCurator.

-p *password*
specify the password for *gemstoneUser*. If omitted, defaults to swordfish.

-P *pathToUpgradeScript*
path to customized to GsDevKit_upgrade instance creation script. If omitted, `$GEMSTONE/upgrade/createGsDevKit_upgrade.topaz`.

For example,

```
unix> $GEMSTONE/seaside/bin/upgradeSeasideImage -s stoneName35
```

The script will prompt you to press the return key to begin.

The script should complete with the message:

```
Seaside Upgrade completed. No errors detected.
```

4. Load your Application Code

After upgrade has successfully completed, reload your application code.

This should be done for upgrades from 3.3.x and 3.4.x as well as upgrades from 3.2.x.

5. Complete the Upgrade Process

To complete the upgrade, return to Chapter 2 or Chapter 3 and complete the remaining upgrade steps.

Chapter

5 Configuring GBS for GemStone/S 64 Bit

This chapter describes how to configure or update your client Smalltalk application using GemBuilder for Smalltalk (GBS) on Linux to run with GemStone/S 64 Bit version 3.5.

This chapter describes Linux clients only; for GBS clients running on Windows, see the *GemStone/S 64 Bit Windows Client Installation Guide*. GBS clients are not supported on Solaris, AIX, or Macintosh. For a table of all supported GBS and client Smalltalk platforms, see the *GemStone/S 64 Bit Release Notes* for v3.5.

The GemStone/S 64 Bit v3.5 server requires a compatible version of GBS; versions of GBS earlier than 8.4 (or GBS v5.4.5 for VA Smalltalk, which is supported on Windows only), are not supported with GemStone/S 64 Bit v3.5 or later.

In addition to using the appropriate version of GBS, you must use GemStone/S 64 Bit 3.5 client libraries with your GBS client application, to be able to log in to the v3.5 server. These libraries are specific to the GemStone/S 64 Bit server version and to the client platform.

GemStone/S 64 Bit provides both 64-bit libraries and 32-bit libraries.

- With 64-bit VisualWorks environments, you must use the 64-bit GemStone client libraries, and it is possible to login either RPC or linked.
- With 32-bit VisualWorks Smalltalk environments, you must use 32-bit libraries, and can only login in RPC. 32-bit processes cannot load 64-bit libraries.

For instructions on installing and configuring GBS, see the GemBuilder for Smalltalk Installation Guide for the appropriate version of GBS.

Supported GBS client Smalltalk Platforms with Linux

GBS version 8.4 or later is required for use with GemStone/S 64 Bit v3.5 or later servers.

The following VisualWorks client Smalltalk versions are supported:

GemBuilder for Smalltalk v8.4

- **VisualWorks 8.3.2 (32-bit and 64-bit)**
  - RedHat Linux 6.9 and 7.4; Ubuntu 16.04 and 18.04
- **VisualWorks 7.10.1 (32-bit)**
  - RedHat Linux 6.9 and 7.4; Ubuntu 16.04
- **VisualWorks 7.10.1 (64-bit)**
  - RedHat Linux ES 6.9 and 7.4

For supported client Smalltalk versions on Windows, refer to the *GemStone/S 64 Bit Windows Client Installation Guide*.

GBS Setup or Upgrade Procedure

Shared Libraries for GBS Client Node

The GBS client requires a set of shared libraries (.so files) that are provided as part of the GemStone server product distribution. When these are loaded into the VisualWorks image in which GBS code is installed, the GBS client can log into the GemStone server.

The shared libraries must be the same version as the GemStone server. Since they are loaded into the client smalltalk VM, they must the appropriate for the client platform and client executable bit size (32-bit or 64-bit).

If your GBS client is on a different platform than your GemStone/server, you will need to download the version-specific libraries for the platform that the GBS client is running on.

You can either install the full GemStone/S 64 Bit Server on your GBS client node, or copy just the specific shared libraries you need.

Install full GemStone/S 64 Bit on client

If your clients run on the same machine as the server, you have no need to do anything further – you can use the libraries in their locations in the existing server installation.

Otherwise, you may find it useful to install the full GemStone/S 64 Bit Server on the client.

If you will run linked sessions on the GBS client, or other configurations in which the gem is on the same node as the GBS client, you will need much of the GemStone server installation. The GemStone/S 64 Bit installation also includes tools such as topaz, gslist, and VSD, that may be useful to run on your client.

Install GemStone/S 64 Bit on the client machine following the instructions in the Installation Guide for the client platform. You do not need to configure an extent or perform similar server tasks.

Copy only specific client libraries

If you will only be running RPC sessions, and do not require tools such as gslist, vsd, or topaz on the client, you do not need to install the full GemStone/S 64 Bit Server on the client node. You may copy only the small set of library files that GBS requires. If you want to run linked sessions as well as RPC, you will need a full server installation on the client.

The files that are required depend on the bit size of the Client smalltalk application.

32-bit VisualWorks clients, RPC only

With 32-bit VisualWorks, the following files are needed:

```
$GEMSTONE/lib32/libgcirpc-3.5.0-32.so
$GEMSTONE/lib32/libssl-3.5.0-32.so
$GEMSTONE/lib32/libkrb5-3.5.0-32.so
```

64 bit VisualWorks clients, RPC only

With 64-bit VisualWorks, the following files are needed:

```
$GEMSTONE/lib/libgcirpc-3.5.0-64.so
$GEMSTONE/lib/libssl-3.5.0-64.so
$GEMSTONE/lib32/libkrb5-3.5.0-64.so
```

GBS provides a number of options as to where on the client machine to place the shared libraries. Refer to the *GemBuilder for Smalltalk Installation Guide* for details on these options

Upgrade GBS to a version that supports v3.5

Versions of GBS/VW earlier than 8.4 are not supported with GemStone/S 64 Bit v3.5.

If you are currently running with an older version of GBS, you should upgrade the GBS client, and possibly also upgrade VisualWorks, to a more recent version for v3.5.

See the *GemBuilder for Smalltalk Installation Guide* for installation instructions, and the *GemBuilder for Smalltalk Release Notes* for details on the changes in these versions.

Update GBS to reference v3.5 libraries

Once you have installed the GemStone server on the GBS client machine, or copied the appropriate shared libraries, you need to ensure that the client Smalltalk executable – a VisualWorks application – will load the v3.5 libraries.

Determining library name to specify

Whether or not you have a full server installation on the client or have copied a few libraries, there is a specific library name you will specify to have GBS load using the `libraryName:` parameter.

32-bit VisualWorks clients, RPC logins only:

```
libgcirpc-3.5.0-32.so
```

64-bit VisualWorks clients, RPC logins only:

```
libgcirpc-3.5.0-64.so
```

64-bit VisualWorks clients, Linked and RPC logins:

```
libgbslnk-3.5.0-64.so
```

Setup GBS to load the new libraries

- If you have set the GBS configuration parameter `libraryName:`, update this to the new library name, and save your image.
- If you have set the GBS configuration parameter `libraryName:` to an empty string, ensure that no other client libraries of the same name are in the current working directory or the bin directory or subdirectory of your VisualWorks image's VISUALWORKS directory.
- For a new GBS application, refer to the *GemBuilder for Smalltalk Installation Guide* for details on the library loading setup options.

Stop and restart the client VM

GBS loads the client libraries into the client Smalltalk VM the first time a GemStone server call is made after each startup of the VM.