



UNIX®, Linux, and macOS Installation Details

Version 2025.1
2025-06-03

InterSystems®, HealthShare Care Community®, HealthShare Unified Care Record®, IntegratedML®, InterSystems Caché®, InterSystems Ensemble®, InterSystems HealthShare®, InterSystems IRIS®, and TrakCare are registered trademarks of InterSystems Corporation. HealthShare® CMS Solution Pack™, HealthShare® Health Connect Cloud™, InterSystems® Data Fabric Studio™, InterSystems IRIS for Health™, InterSystems Supply Chain Orchestrator™, and InterSystems TotalView™ For Asset Management are trademarks of InterSystems Corporation. TrakCare is a registered trademark in Australia and the European Union.

All other brand or product names used herein are trademarks or registered trademarks of their respective companies or organizations.

This document contains trade secret and confidential information which is the property of InterSystems Corporation, One Congress Street, Boston, MA 02114, or its affiliates, and is furnished for the sole purpose of the operation and maintenance of the products of InterSystems Corporation. No part of this publication is to be used for any other purpose, and this publication is not to be reproduced, copied, disclosed, transmitted, stored in a retrieval system or translated into any human or computer language, in any form, by any means, in whole or in part, without the express prior written consent of InterSystems Corporation.

The copying, use and disposition of this document and the software programs described herein is prohibited except to the limited extent set forth in the standard software license agreement(s) of InterSystems Corporation covering such programs and related documentation. InterSystems Corporation makes no representations and warranties concerning such software programs other than those set forth in such standard software license agreement(s). In addition, the liability of InterSystems Corporation for any losses or damages relating to or arising out of the use of such software programs is limited in the manner set forth in such standard software license agreement(s).

THE FOREGOING IS A GENERAL SUMMARY OF THE RESTRICTIONS AND LIMITATIONS IMPOSED BY INTERSYSTEMS CORPORATION ON THE USE OF, AND LIABILITY ARISING FROM, ITS COMPUTER SOFTWARE. FOR COMPLETE INFORMATION REFERENCE SHOULD BE MADE TO THE STANDARD SOFTWARE LICENSE AGREEMENT(S) OF INTERSYSTEMS CORPORATION, COPIES OF WHICH WILL BE MADE AVAILABLE UPON REQUEST.

InterSystems Corporation disclaims responsibility for errors which may appear in this document, and it reserves the right, in its sole discretion and without notice, to make substitutions and modifications in the products and practices described in this document.

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)

Tel: +1-617-621-0700

Tel: +44 (0) 844 854 2917

Email: support@InterSystems.com

Table of Contents

1 AIX® Platform Notes	1
1.1 I/O Pacing Parameters	1
1.2 File System Mount Option	1
1.3 Memory Management Parameters	2
1.4 AIX® Tunable Parameters	3
1.4.1 AIX® Interprocess Communication Tunable Parameters	3
1.4.2 maxuproc	4
1.4.3 Required C/C++ Runtime Libraries	4
1.4.4 Shared Library Environment Variable for InterSystems IRIS Shared Library Support	4
1.4.5 Use of Raw Ethernet	5
2 Red Hat Linux Platform Notes	7
2.1 Shared Memory Limits	7
2.2 Locked-in Memory	8
2.3 Using Kerberos	9
3 SUSE Linux Platform Notes	11
3.1 I/O Scheduler	11
3.2 Shared Memory Limits	11
3.3 Locked-in Memory	12
3.4 Using Kerberos	12
4 Ubuntu Platform Notes	13
4.1 Semaphore Deletion Setting	13
5 Installing a Web Server	15
6 Determining Owners and Groups	17
7 Installing as a Nonroot User	19
7.1 Why InterSystems IRIS Installation Uses Root	19
7.2 Nonroot Installation Limitations	19
7.3 Nonroot Installation Differences	20
8 Unattended Installation Parameters	21
9 Unattended Installation Packages	25
10 Starting InterSystems IRIS	27
11 Adjustments for Large Number of Concurrent Processes on macOS	29
12 Installing a Development Environment	31
13 Adding UNIX® Installation Packages to an InterSystems IRIS Distribution	33
13.1 UNIX® Installation Package Tutorial	33
13.2 Contents of the parameters.isc File	36

List of Tables

Table 13–1: InterSystems IRIS Installation Parameter File Variables 37

1

AIX® Platform Notes

The default settings of several AIX® parameters can adversely affect performance. The settings and recommendations are detailed for the following:

- [I/O Pacing Parameters](#)
- [File System Mount Option](#)
- [Memory Management Parameters](#)
- [AIX® Tunable Parameters](#)
- [Required C/C++ Runtime Libraries](#)
- [Shared Library Environment Variable for InterSystems IRIS Shared Library Support](#)
- [Use of Raw Ethernet](#)

1.1 I/O Pacing Parameters

AIX® implements an I/O pacing algorithm that may hinder InterSystems IRIS write daemons. In AIX® 5.2 and AIX® 5.3, I/O pacing is automatically enabled when using HACMP clustering; beginning in AIX® 6.1, however, I/O pacing is enabled on all systems and the default high-water mark is set higher than in earlier releases.

If write daemons are slowing or stalling, you may have to adjust the high-water mark; for information, see [Disk-I/O Pacing](#) in the IBM AIX documentation.

Important: Beginning in AIX® 6.1, you should not have to make any high-water mark adjustments.

If you have questions about the impact to your system, however, contact the [InterSystems Worldwide Response Center \(WRC\)](#) or your AIX® supplier before making any changes. These recommendations apply to both JFS and Enhanced JFS (JFS2) file systems.

1.2 File System Mount Option

Different mount options may improve performance for some workloads.

Note: Non-InterSystems IRIS workloads that benefit from file system caching (for example, operating system-level backups and/or file copies) are slowed by the `cio` mount option.

To improve recovery speed using the IRIS.WIJ file after a hard shutdown or system crash, InterSystems recommends a mount option that includes file system buffering (for example, `rw`) for the file system that contains the IRIS.WIJ file.

For information about **mount** options, see [mount Command](#) in the IBM AIX documentation.

1.3 Memory Management Parameters

The number of file systems and the amount of activity on them can limit the number of memory structures available to JFS or JFS2, and delay I/O operations waiting for those memory structures.

To monitor these metrics, issue a **vmstat -vs** command, wait two minutes, and issue another **vmstat -vs** command. The output looks similar to the following:

```
# vmstat -vs
1310720 memory pages
1217707 lruable pages
144217 free pages
1 memory pools
106158 pinned pages
80.0 maxpin percentage
20.0 minperm percentage
80.0 maxperm percentage
62.8 numperm percentage
764830 file pages
0.0 compressed percentage
0 compressed pages
32.1 numclient percentage
80.0 maxclient percentage
392036 client pages
0 remote pageouts scheduled
0 pending disk I/Os blocked with no pbuf
5060 paging space I/Os blocked with no psbuf
5512714 filesystem I/Os blocked with no fsbuf
194775 client filesystem I/Os blocked with no fsbuf
0 external pager filesystem I/Os blocked with no fsbuf
```

If you see an increase in the following parameters, increase the values for better InterSystems IRIS performance:

- *pending disk I/Os blocked with no pbuf*
- *paging space I/Os blocked with no psbuf*
- *filesystem I/Os blocked with no fsbuf*
- *client filesystem I/Os blocked with no fsbuf*
- *external pager filesystem I/Os blocked with no fsbuf*

When increasing these parameters from the default values:

1. Increase the current value by 50%.
2. Check the **vmstat** output.
3. Run **vmstat** twice, two minutes apart.
4. If the field is still increasing, increase again by the same amount; continue this step until the field stops increasing between **vmstat** reports.

Important: Change both the current and the reboot values, and check the **vmstat** output regularly because I/O patterns may change over time (hours, days, or weeks).

See the following IBM web pages for more detailed information:

- For a complete description of each of the fields reported by **vmstat**, see [vmstat Command](#) in the IBM AIX documentation.
- For instructions on how to increase these parameters, see [VMM page replacement tuning](#) in the IBM AIX documentation.
- For a complete description of managing I/O tunable parameters, see [ioo Command](#) in the IBM AIX documentation.

1.4 AIX® Tunable Parameters

1.4.1 AIX® Interprocess Communication Tunable Parameters

The following table lists the tunable parameters for the IBM pSeries AIX® 5.2 operating system. None of the following listed parameters require tuning because each is dynamically adjusted as needed by the kernel. For more information, see [Interprocess communication tunable parameters](#) in the AIX documentation.

Parameter	Purpose	Dynamic Values
msgmax	Specifies maximum message size.	Maximum value of 4 MB
msgmnb	Specifies maximum number of bytes on queue.	Maximum value of 4 MB
msgmni	Specifies maximum number of message queue IDs.	Maximum value of 4096
msgmnm	Specifies maximum number of messages per queue.	Maximum value of 524288
semaem	Specifies maximum value for adjustment on exit.	Maximum value of 16384
semmni	Specifies maximum number of semaphore IDs.	Maximum value of 4096
semmsl	Specifies maximum number of semaphores per ID.	Maximum value of 65535
semopm	Specifies maximum number of operations per semop() call.	Maximum value of 1024
semume	Specifies maximum number of undo entries per process.	Maximum value of 1024
semvmx	Specifies maximum value of a semaphore.	Maximum value of 32767
shmmax	Specifies maximum shared memory segment size.	Maximum value of 256 MB for 32-bit processes and 0x80000000u for 64-bit
shmmmin	Specifies minimum shared-memory-segment size.	Minimum value of 1
shmmni	Specifies maximum number of shared memory IDs.	Maximum value of 4096

1.4.2 maxuproc

`maxuproc`, which specifies the maximum number of processes than can be started by a single nonroot user, is a tunable parameter that can be adjusted as described in this subsection.

If this parameter is set too low then various components of the operating system can fail as more and more users attempt to start processes; these failures include loss of CSP pages, background tasks failing, etc. Therefore, you should set the `maxuproc` parameter to be higher than the maximum number of processes that might be started by a nonroot user (including interactive users, web server processes, and anything that might start a process).

Note: Do not set the value excessively high because this value protects a server from a runaway application that is creating new processes unnecessarily; however, setting it too low causes unexplained problems.

InterSystems suggests that you set `maxuproc` to be double your expected maximum process count which gives a margin of error but still provides protection from runaway processes. For example, if your system has 1000 interactive users and often runs 500 background processes, then a value of at least 3000 would be a good choice.

The `maxuproc` value can be examined and changed either from the command line or from the **smit/smitty** administrator utilities, both as root user, as follows:

- From the command line, view the current setting:

```
# lsattr -E -l sys0 -a maxuproc
```

then modify the value:

```
# chdev -l sys0 -a maxuproc=NNNNNN
```

where `NNNNNN` is the new value.

- From the administrator utility **smit** (or **smitty**) choose **System Environments > Change / Show Characteristics of Operating System > Maximum number of PROCESSES allowed per user**.

If you increase the value of `maxuproc`, the change is effective immediately. If you decrease the value of `maxuproc`, the change does not take effect until the next system reboot. In both cases the change persists over system reboots.

1.4.3 Required C/C++ Runtime Libraries

You must ensure that the required C/C++ runtime is installed on your IBM AIX® system before installing InterSystems IRIS.

InterSystems IRIS for AIX is compiled using the IBM XL C/C++ for AIX 16.1.x and 17.1.x compilers. If the system on which you are installing InterSystems IRIS does not have the corresponding version of the runtime already installed, you must install it. You can download the latest 16.1.x and 17.1.x runtimes from the [Fix list for XL C/C++ Runtime for AIX](#) on the IBM Support website.

1.4.4 Shared Library Environment Variable for InterSystems IRIS Shared Library Support

The InterSystems IRIS shared library support contain a batch file that references any installed C linker.

If you have either the standard UNIX® C libraries or any proprietary C libraries defined in the `LIBPATH` environment variable, then your environment is ready.

If not, append the paths for the standard UNIX® C libraries to `LIBPATH`; these paths are `/usr/lib` and `/lib`.

1.4.5 Use of Raw Ethernet

In order to use Raw Ethernet, an IBM AIX® machine must have the DLPI (Data Link Provider Interface) packages installed. If the machine does not have the DLPI packages, obtain them from your IBM provider and create DLPI devices through the following procedure:

1. Log in as `root`.
2. In the PSE drivers section of the `/etc/pse.conf` file, uncomment the four lines that refer to the DLPI drivers.
3. Save the file.
4. Restart the computer.

If the DLPI devices are not installed, the **EthernetAddress()** method of the `%SYSTEM.INetInfo` class returns a null string rather than information about the Ethernet device.

2

Red Hat Linux Platform Notes

This topic includes the information on the following adjustments:

- [Shared Memory Limit](#)
- [Locked-in Memory](#)
- [Using Kerberos](#)

2.1 Shared Memory Limits

Depending on your version of Red Hat Enterprise Linux you may need to set kernel parameters related to shared memory.

RHEL 9

shmmax (maximum size shared memory segment) — The default for this parameter is sufficient.

shmall (maximum size of total shared memory system wide) — This should be set to 18446744073692774399 (in bytes).

RHEL 8

shmmax (maximum size shared memory segment) — This should be set to 18446744073692774399 (in bytes).

shmall (maximum size of total shared memory system wide) — This should be set to 18446744073692774399 (in bytes).

Changing the Kernel Parameters

You can change these parameters in the `proc` file system without a restart. The new memory limits remain in effect until you restart the Red Hat Linux system.

For example, type the following command:

```
$ echo 18446744073692774399 >/proc/sys/kernel/shmmax
```

You can put this command into a startup script.

Alternatively, you can use **sysctl(8)**, if available, to set this parameter permanently. Add a line to the `/etc/sysctl.conf` similar to the following:

```
kernel.shmmax = 18446744073692774399
```

This file is usually processed at startup, but **sysctl** can also be called explicitly later.

For more information, reference [Adjusting kernel parameters for database servers](#) in the Red Hat Documentation.

msgmni

The *msgmni* parameter may also be set too low if you are running more than one instance of InterSystems IRIS on a machine. Set this value to three times the number of instances of InterSystems IRIS that run simultaneously on your system.

Other Parameters

Other parameters are sufficiently sized for an InterSystems IRIS application. To view the values of other parameters, look in the files `/usr/src/linux/include/asm-xxx/shmparam.h` and `/usr/src/linux/include/linux/sem.h`.

2.2 Locked-in Memory

On Linux platforms, if shared memory is allocated in huge pages, the pages are automatically locked in memory and no further action is required. See the [Configuring Huge Pages on Linux](#) section in this chapter for information about allocating huge pages.

If not using huge pages, you can configure InterSystems IRIS to lock the shared memory segment in memory to prevent paging. This is described in the [LockSharedMemory](#) section of the “memlock” entry in the *Configuration Parameter File Reference*.

Otherwise, you must increase the maximum size that may be locked into memory. View the current value using the **ulimit** command:

For example, to display all current limits:

```
bash$ ulimit -a
core file size (blocks, -c) unlimited
data seg size (KBytes, -d) unlimited
file size (blocks, -f) unlimited
pending signals (-i) 1024
max locked memory (KBytes, -l) 32 <----- CHANGE TO unlimited
max memory size (KBytes, -m) unlimited
open files (-n) 1024
pipe size (512 bytes, -p) 8
POSIX message queues (bytes, -q) 819200
stack size (KBytes, -s) 10240
cpu time (seconds, -t) unlimited
max user processes (-u) 49000
virtual memory (KBytes, -v) unlimited
file locks (-x) unlimited
```

To display only *max locked memory*, use the `-l` option:

```
bash$ ulimit -l
32
```

Set *max locked memory* to unlimited. If you have privileges, you can alter the value directly using the **ulimit** command; however, it is better to update the *memlock* parameter in the `/etc/security/limits.conf` file. If the *memlock* limit is too low, Linux reports a `ENOMEM` - “Not enough memory” error, which does not make the cause obvious. The actual memory is allocated; it is the lock that fails.

For more information, see [memlock](#) in the *Configuration Parameter File Reference*.

2.3 Using Kerberos

To use Kerberos on the Red Hat Linux platform, you must install the `krb5-devel` package in addition to the `krb5-libs` package. Installing `krb5-devel` establishes the required symbolic links for using Kerberos. The package is required for production environments, not only development environments. See the [Red Hat Network](#) web site for more information about these components.

3

SUSE Linux Platform Notes

This topic includes the information on the following adjustments:

- [I/O Scheduler](#)
- [Shared Memory Limits](#)
- [Locked-in Memory](#)
- [Using Kerberos](#)

3.1 I/O Scheduler

The I/O scheduler for SUSE Linux is responsible for ordering the I/O requests submitted to a storage device. On SUSE Linux 15, it may default to `BFQ` (Budget Fair Queueing) which is known to cause performance issues with InterSystems IRIS. InterSystems recommends changing this setting to `NONE`. For details on changing the I/O scheduler, see the SUSE documentation: [Tuning I/O Performance](#).

Note: Setting the I/O scheduler to `NONE` may not be optimal for all use cases. Users should test the system's application workload after making any changes.

3.2 Shared Memory Limits

The default shared memory limits (*shhmax* and *shmall*) on SUSE Linux 32-bit platforms are too small for InterSystems IRIS, and can be changed in the `proc` file system without a restart.

InterSystems IRIS uses shared memory for database buffers, global buffers, routine buffers, as well as license use. If the machine is being used only for InterSystems IRIS, InterSystems recommends setting the shared memory to approximately half the total memory. For more information, see [Memory Planning](#) in *System Resource Planning and Management*, as well as [Memory and Startup Settings](#) and [Determining License Capacity and Usage](#) in the *System Administration Guide*.

Note: The recommendations to change the shared memory limits do not apply to SUSE Linux 64-bit systems.

For example, to allow 512 MB, type the following commands:

```
#sets shmall and shmmx shared memory
echo 536870912 >/proc/sys/kernel/shmall      #Sets shmall to 512 MB
echo 536870912 >/proc/sys/kernel/shmmx      #Sets shmmx to 512 MB
```

You can put these commands into a script that is run at startup. The SUSE Linux product documentation recommends you put the commands in the `/etc/init.d/boot.local` script file.

You can change the settings for the system memory user limits by modifying a file called `/etc/profile.local`. Add lines similar to the following:

```
#sets user limits (ulimit) for system memory resources
ulimit -v 512000      #set virtual (swap) memory to 512 MB
ulimit -m 512000      #set physical memory to 512 MB
```

In this same file, you can permanently change the values for the `PATH` and `CLASSPATH` parameters by adding lines similar to the following:

```
#sets env values PATH and CLASSPATH
export PATH=$PATH:/usr/iris/bin:/path/to/j2sdk/bin:/.
export CLASSPATH=
        $CLASSPATH:/iris/dev/java/lib/JDK18/interSystems-jdbc-3.0.0.jar.
```

Important: To avoid the risk of losing your changes during system upgrades, do not change the `/etc/profile` file.

3.3 Locked-in Memory

On Linux platforms, if shared memory is allocated in huge pages, they are automatically locked in memory and no further action is required. See [Configuring Huge Pages on Linux](#) for information about allocating huge pages.

If not using huge pages, you can configure InterSystems IRIS to lock the shared memory segment in memory to prevent paging. This is described in the [LockSharedMemory](#) section of the “memlock” entry in the *Configuration Parameter File Reference*.

Otherwise, you must increase the maximum size that may be locked into memory. See the [Locked-in Memory](#) section of the *Red Hat Linux Platform Notes* in this chapter for instructions.

3.4 Using Kerberos

To use Kerberos on the SUSE Linux platform, you must install the `krb5-devel` package in addition to the `krb5-libs` package. Installing `krb5-devel` establishes the required symbolic links for using Kerberos. The package is required for production environments, not only development environments. See the [SUSE documentation](#) web site for more information about these components.

4

Ubuntu Platform Notes

This topic includes the information on the following adjustments:

- [Semaphore Deletion Setting](#)

4.1 Semaphore Deletion Setting

Under some circumstances, the OS may delete an instance's semaphores when the instance owner connects to an Ubuntu host, for example using SSH. To prevent this, edit the `/etc/systemd/logind.conf` file and change **RemoveIPC=yes** (the default) to **RemoveIPC=no**.

Updating to a newer version of Ubuntu may revert **RemoveIPC** to the default value of **yes**. After updating Ubuntu, be sure to change **RemoveIPC** to avoid unwanted semaphore deletion.

5

Installing a Web Server

Before installing InterSystems IRIS, you should install a supported web server in order to access web applications, including the Management Portal. The installation procedure will vary depending on the web server you choose. For details, you should refer to your web server's documentation.

In many cases, the InterSystems IRIS installer can automatically configure a new or upgraded instance to serve its built-in web applications using your web server. For details, see [Connect Your Web Server Automatically](#).

If you would like to manually set up your web server, you can do this after installing IRIS. For details, see [Connect Your Web Server Manually](#).

If you intend on configuring the web server manually (or not configuring a web server at all) and you are performing an unattended installation, make sure that you set the parameter `ISC_PACKAGE_WEB_CONFIGURE="N"`. Otherwise, the installation will attempt to automatically configure your web server and will fail if a web server is not detected.

Important: InterSystems recommends using the Apache httpd web server because it can be automatically configured during the installation process. Make sure it is installed and running before beginning the installation process. In most cases, it is not necessary to manually configure the Apache web server.

6

Determining Owners and Groups

The installation process prompts for the following user and group information:

- *Owner of the instance*
- *Effective user for the InterSystems IRIS superserver and its jobs*
- *Effective group for InterSystems IRIS processes*
- *Group allowed to start and stop the instance*

For more information about these categories, see the [UNIX® User and Group Identifications](#) section in the “Using InterSystems IRIS on UNIX®, Linux, and macOS” chapter of *System Administration Guide*.

Important: InterSystems IRIS must set user, group, and other permissions on files that it installs. To accomplish this, InterSystems IRIS sets **umask** to 022 for the installation process - do *not* modify the **umask** until the installation is complete.

The user account you identify as *Owner of the instance* and the group you identify as *Group allowed to start and stop the instance* must both exist before you begin installation. If an entry you provide at one of these prompts does not exist, the prompt is repeated, so verify that the user and group you intend to provide exist before you begin installation.

If your operating system contains the **useradd** and **groupadd** utilities (or **mkgroup** and **mkuser** on AIX®), the system creates the account for the *effective user for InterSystems IRIS superserver* and the *effective group for InterSystems IRIS processes*, if the entries you provide do not exist. However, if these utilities are not present and an entry you provide does not exist, the prompt is repeated. If you are not sure that your system has these utilities, verify that the user and group you intend to provide exist before you begin installation.

Note: If your operating system uses Network Information Services (NIS) or another form of network-based user/group database, the **groupadd** and **useradd** utilities (or **mkgroup** and **mkuser** on AIX®) may create a local user and/or group that could conflict with existing entries in the network database. To avoid this problem, it may be best to create the InterSystems IRIS effective group and effective user in your network database using the appropriate administration tools prior to beginning installation, rather than allowing the utilities to create them.

While performing a custom UNIX® reinstallation or upgrade, you will have the option to change the *Effective group for InterSystems IRIS processes* from the original installation. If you specify a different effective group, the system will update the ownership of all files in the IRIS installation directory that were previously owned by the original effective group to the new effective group. Please note that you are responsible for updating the ownership of any files created outside the IRIS installation directory to ensure continued access.

Tools used on UNIX® operating systems to display process ownership may or may not show effective versus real ownership. See the “[UNIX® Users, Groups and Permissions](#)” chapter of the *System Administration Guide* for details on how InterSystems IRIS assigns permissions.

7

Installing as a Nonroot User

When installing InterSystems IRIS in a production environment, InterSystems recommends using root privileges. It is possible to run an InterSystems IRIS installation without root privilege, but these installations have several limitations. The following sections describe these limitations and the differences from a standard InterSystems IRIS installation.

- [Why InterSystems IRIS Installation Uses Root](#)
- [Nonroot Installation Limitations](#)
- [Nonroot Installation Differences](#)

Important: Root privilege should only be used when *installing* InterSystems IRIS. Once the installation is complete, all users should interact with InterSystems IRIS using nonroot privileges.

7.1 Why InterSystems IRIS Installation Uses Root

InterSystems IRIS is typically installed using root and operated using nonroot privileges. Several features require root access, but the majority of processes run as a user or group that you specify during installation. The purpose of these users and groups, and how they use root, is described in [UNIX Users, Groups, and Permissions](#).

InterSystems IRIS processes that utilize root privileges include:

- The [Virtual IP](#) process, which has root as its effective user ID (UID) to modify network settings on the operating system.
- The Control Process, which has the instance owner as its effective UID and root as its real UID. The real UID is used to get [large pages](#) at startup and to communicate with other InterSystems IRIS processes.
- The startup executables, which have root as the effective UID.

Installing InterSystems IRIS as root also enhances security by ensuring that only users with root privileges can modify or replace the file structure.

7.2 Nonroot Installation Limitations

While nonroot installations of InterSystems IRIS are supported, there are several features that cannot be used in instances installed in this way:

- The installation mount point cannot be mounted with *nosuid* set, which may result in a 'Permission denied' error.

- The Web Gateway cannot be configured to use an external web server.
- A [mirror Virtual IP](#) cannot be used.

Note: For alternative methods of routing network traffic, such as using a network load balancer or the Web Gateway, see [Redirecting Application Connections Following Failover or Disaster Recovery](#).

- There is no option to specify the instance owner and group allowed to start and stop InterSystems IRIS during installation (as described in [Determining Owners and Groups](#)).
- There is no group access. All instance files, including the registry, are owned and can be read, written, and executed by the installing user only.

For example, where a standard instance might have:

```
-rws--x--- 5 root develop 43282 Aug 28 07:52 irismgr
-r-x--s---x 1 <nonroot-user> irisusr 23058 Aug 28 07:52 irisuxsession
```

a nonroot instance would have:

```
-rwx----- 5 <installing-user> develop 43282 Aug 28 07:52 irismgr
-r-x----- 1 <installing-user> develop 23058 Aug 28 07:52 irisuxsession
```

The registry is located in the directory specified by *IRISSYS*, and nonroot instances are found in that registry. (The **iris** executable is also in that directory.) Only nonroot instances may be in the nonroot registry. Any attempt to access a root-installed instance from a nonroot registry fails. Conversely, a nonroot instance may be defined in a root-registry, but an attempt to access the instance by any user other than the owner fails.

Note: InterSystems recommends that the registry be placed in a directory that is local to the machine on which the instance is installed, not an NFS directory. Note that the standard location `/usr/local/etc` is such a directory.

7.3 Nonroot Installation Differences

Along with the feature limitations described above, there are several apparent differences between root and nonroot InterSystems IRIS installations:

- The *IRISSYS* environment variable must be defined as an existing directory writable by the installing user, and must be present during installation and all instance operations.
- The [ISCAgent](#) is installed in the directory specified by *IRISSYS*.

Note: For information about starting the ISCAgent for a nonroot instance, see [Starting the ISCAgent for Nonroot Instances on UNIX®/Linux and macOS Systems](#) in the “Mirroring” chapter of the *High Availability Guide*.

- Only the installing user’s account can access and operate the InterSystems IRIS instance.
- All InterSystems IRIS executables and processes run as the installing user.

8

Unattended Installation Parameters

The following describes the parameters used with the `irisinstall_silent` script in unattended installation. Some parameters include a default value which the installation uses if the parameter is omitted.

ISC_CPF_MERGE_FILE="*<location>*"

Specifies the location of the configuration merge file when performing a configuration merge.

For more information, see [Automating Configuration of InterSystems IRIS with Configuration Merge](#).

ISC_PACKAGE_INSTANCENAME="*<instancename>*"

Required

Specifies the name of the instance to be installed or upgraded: if the instance does not exist, this is a new installation; if it does exist, this is an upgrade. For example:

```
ISC_PACKAGE_INSTANCENAME="MyIris"
```

ISC_PACKAGE_INSTALLDIR="*<installdir>*"

Required for new instances

Specifies the installation directory for the new instance to be installed; for example:

```
ISC_PACKAGE_INSTALLDIR="/opt/MyIris"
```

If the specified directory does not exist, the installation attempts to create it. This parameter is ignored if you are upgrading an installation.

Review the [Installation Directory](#) section of this book for information about choosing an installation directory.

ISC_PACKAGE_INSTALL_INTEGRATEDML="Y" | "N"

Default: Y

Specifies whether or not to install InterSystems IntegratedML.

For more information about IntegratedML, see [Using IntegratedML](#).

ISC_PACKAGE_UNICODE="Y" | "N"

Optional

Specifies whether or not this is a UNICODE installation; valid values are Y or N. See [Character Width Settings](#) for more information.

ISC_PACKAGE_INITIAL_SECURITY="Minimal" | "Normal" | "LockedDown"

Optional

Default: LockedDown

Specifies the initial security setting for the installation; valid values are: "Minimal", "Normal", or "LockedDown".

If it is set to "Normal" or "LockedDown", ISC_PACKAGE_USER_PASSWORD is required.

"Minimal" is only available for InterSystems IRIS installations.

ISC_PACKAGE_MGRUSER="<user>"

Optional

Default: <installer's-username>

Specifies the login name of the owner of the installation. For example:

ISC_PACKAGE_MGRUSER="jcsmith"

If the security level is "Minimal", this parameter is ignored, and ISC_PACKAGE_MGRUSER is set to "root".

ISC_PACKAGE_MGRGROUP="<group>"

Optional

Default: <installer's-group>

Specifies the group that is allowed to start and stop the instance. For example:

ISC_PACKAGE_INITIAL_MGRGROUP="irisusr"

ISC_PACKAGE_USER_PASSWORD="<password>"

Required for installation of secure instances

Specifies the required password for an instance with Normal or Locked Down security.

If the security level is "Minimal", this parameter is ignored.

If the security level is "Normal" or "LockedDown" and this parameter is not specified, the installation fails and an error is thrown.

ISC_PACKAGE_CSPSYSTEM_PASSWORD="<password>"

Specifies the password for the CSPSystem user.

If the security level is "Minimal", this parameter is ignored.

If the security level is "Normal" or "LockedDown" and this parameter is not specified, the value of ISC_PACKAGE_USER_PASSWORD is used.

ISC_PACKAGE_IRISUSER="<user>"

Optional

Default: irisusr

Specifies the effective user for the InterSystems IRIS Superserver.

If the security level is "Minimal", this parameter is ignored and set to the default.

ISC_PACKAGE_IRISGROUP="*<group>*"

Optional

Default: irisusr

Specifies the effective user for InterSystems IRIS processes.

If the security level is "Minimal", this parameter is ignored and set to the default.

ISC_PACKAGE_CLIENT_COMPONENTS="*<component1> <component2> ...*"

Optional

Default: <all-client-bindings-installed>

Specifies the client bindings to be installed from the client_components package (see [Unattended Installation Packages](#)).

Specified components (bindings) must be space-delimited. Available components are listed in package/client_components/manifest.isc. Installation validates the specified components and removes those that do not exist or are not supported on a particular system.

ISC_PACKAGE_WEB_CONFIGURE="Y" | "N"

Optional

Default: Y (automatically configure the gateway for an external web server)

Specifies whether or not to configure the Web Gateway for an external web server.

Set this option to N if you don't want the web server configured automatically.

ISC_PACKAGE_WEB_SERVERTYPE="Apache" | "None"

Optional

Default: Apache

Type of existing web server for the Web Gateway to use. For example:

ISC_PACKAGE_WEB_SERVERTYPE="Apache"

ISC_PACKAGE_WEB_APACHE_VERSION=2.4

Optional, with ISC_PACKAGE_WEB_SERVERTYPE="Apache"

Default: autodetected

Version of Apache web server.

ISC_PACKAGE_WEB_APACHE_USER="*<username>*"

Optional, with ISC_PACKAGE_WEB_SERVERTYPE="Apache"

Default: autodetected

Username for Apache web server.

ISC_PACKAGE_WEB_APACHE_CONF="*<path_to_httpd.conf>*"

Optional, with ISC_PACKAGE_WEB_SERVERTYPE="Apache"

Default: installation attempts to autodetect file location

Location of the Apache Web server configuration file, for example: /etc/httpd/conf/httpd.conf

By default, installation attempts to autodetect file in one of several standard locations. Installation exits with error if `ISC_PACKAGE_WEB_SERVERTYPE="Apache"` and `httpd.conf` location is undetermined.

`ISC_PACKAGE_WEB_GATEWAY_DIR=<web_gateway_directory>`

Optional, for new Web Gateway installations only

Default: `/opt/webgateway_`

Directory to contain the Web Gateway files.

`ISC_PACKAGE_STARTIRIS="Y" | "N"`

Optional

Default: `Y` (start the instance automatically after installation)

Specifies whether or not to start the installed InterSystems IRIS instance following installation.

`ISC_INSTALLER_MANIFEST=<location>`

When installing with an installation manifest, specifies the location of the exported manifest class.

See [Using the Manifest](#) for more information.

`ISC_INSTALLER_PARAMETERS=<var>=<value>,<var>=<value> ...`

When installing with an installation manifest, specifies variable name/value pairs.

`ISC_INSTALLER_LOGFILE=<filename>`

When installing with an installation manifest, specifies the log file name.

`ISC_INSTALLER_LOGLEVEL=<level>`

When installing with an installation manifest, specifies the [log level](#), from -1 (“none”) to 3 (“verbose”).

`ISC_PACKAGE_SUPERSERVER_PORT=<port_number>`

Optional

Default: 1972 (if available, otherwise 51773 or the first available subsequent number)

Specifies the Superserver port to be used by the instance being installed.

`ISC_PACKAGE_USER_DATABASE=Y | N`

Default: `Y`

Specifies whether or not to install the optional users database.

9

Unattended Installation Packages

The installation scripts for each component are contained in the packages directory below the directory containing the `irisinstall_silent` script. Each package is in its own directory, and each package directory contains a `manifest.isc` file defining prerequisite packages for the package in that directory.

The `standard_install` package is the starting point for a server install in which all packages are installed. To define a custom package, you can use the `manifest.isc` file for the `standard_install` package as a template, as follows:

1. Copy the `standard_install` directory to a new directory.

For example, copy it to a directory named `custom_install`; initially, the `manifest.isc` file in the new directory is similar to the following:

```
#This is the target for a standard (non-client-only) install
package: standard_install
prerequisite: install_mode
prerequisite: database_server
prerequisite: databases
prerequisite: gadget
prerequisite: fop
prerequisite: renderserver
prerequisite: printserver
prerequisite: excelexporter
prerequisite: callin_components
prerequisite: client_components
prerequisite: addenda
prerequisite: install_confirmation
prerequisite: copyright
```

2. In the new directory, modify the `manifest.isc` file as follows:

- Set the package key to the value of the directory name (**required**).
- Add and/or remove prerequisites for your custom installation.

For example, in the following `manifest.isc` file, the value of the package key has been changed to match the directory name (`custom_install`).

```
#This is the target for a custom (non-client-only) install
package: custom_install
prerequisite: install_mode
prerequisite: database_server
prerequisite: gadget
prerequisite: fop
prerequisite: renderserver
prerequisite: printserver
prerequisite: excelexporter
prerequisite: callin_components
prerequisite: client_components
prerequisite: addenda
prerequisite: install_confirmation
prerequisite: copyright
```

Then you can specify the new custom package when performing unattended installations; for example: `sudo ISC_PACKAGE_INSTANCENAME="MyIris" ./irisinstall_silent custom_install`.

Note: See the [Adding UNIX® Installation Packages to an InterSystems IRIS Distribution](#) appendix for information about creating your own UNIX® installation packages and adding them to an InterSystems IRIS distribution.

10

Starting InterSystems IRIS

For most installations, InterSystems IRIS starts automatically when the install completes. If you need to start InterSystems IRIS, first log in to your operating system, then start InterSystems IRIS using the **iris** command:

```
iris start <instance_name>
```

Use the **iris** command to start and stop InterSystems IRIS. It is described in greater detail in the [Controlling InterSystems IRIS Instances](#) section of the *System Administration Guide*.

Note: If the permissions on all elements of the path to the mgr subdirectory do not provide read access to the `irisusr` group (at a minimum), the instance fails to fully start and the following message is recorded in `messages.log`:
Element of path *manager_subdirectory* could not be read (errno 2).

Once InterSystems IRIS is started, initiate an InterSystems IRIS session using the **iris terminal** command, as described in [Connecting to an InterSystems IRIS Instance](#) in the “Using Multiple Instances of InterSystems IRIS” chapter of the *System Administration Guide*:

```
iris terminal <instname> [parameters]
```

Where *instname* is the instance name that you chose during the installation.

For more information, see the “[Using InterSystems IRIS on UNIX®, Linux, and macOS](#)” chapter of the *System Administration Guide*.

11

Adjustments for Large Number of Concurrent Processes on macOS

Make the following adjustments if you are running a system that requires a large number of processes or telnet logins:

1. *Remote connections* — The number of pty (pseudo terminal) connections is limited to 128 system-wide. If your applications count on telnet or other pty-using connections for users to access, keep this in mind.
2. *Number of processes* — If the pty limit is not a problem, but you need to run a larger number of processes, there are limits to that as well.
 - *System-wide process limits* — The `kern.maxproc`, `kern.maxprocperuid` parameters are set to 532 and 100 by default. You can change them using the following commands:

```
administrator$ sudo sysctl -w kern.maxproc=2500
kern.maxproc: 2065 -> 2500
administrator$ sudo sysctl -w kern.maxprocperuid=2500
kern.maxprocperuid: 2000 -> 2500

administrator$ sysctl -a | grep maxproc
kern.maxproc = 2500
kern.maxprocperuid = 2500
```

Note, however, that 2500 is the absolute unchangeable upper limit.

12

Installing a Development Environment

InterSystems IRIS does not include a development environment; therefore, you must install a [supported IDE](#).

13

Adding UNIX® Installation Packages to an InterSystems IRIS Distribution

This appendix describes how to add a new UNIX® installation package to an existing InterSystems IRIS® data platform distribution. It is presented in the form of a tutorial in which we create a simple package that copies additional files into the InterSystems IRIS instance directory.

Note: Because install packages are implemented through UNIX® shell scripts, you can also write packages that perform much more complex operations.

13.1 UNIX® Installation Package Tutorial

Suppose we have written an InterSystems Callout shared library (see the “Creating an InterSystems Callout Library” chapter in *Using the Callout Gateway*) to connect to an imaging device named Foo9000. We compile this library as `libfoo9000.so` and want to install it with InterSystems IRIS. In addition, we want the installation to prompt users to provide the network server name for the device (Foo9000) to which we want the library to connect. This information is stored in a configuration file in the InterSystems IRIS instance’s `install-dir\mgr` directory.

We start with an existing InterSystems IRIS kit:

```
~/kit:>ls
irisinstall  cplatname docs  lgpl.txt NOTICE
copyright.pdf dist  kitlist LICENSE package
```

... and our compiled library (`libfoo9000.so`):

```
~/lib:>ls
libfoo9000.so
```

First, we need to choose a location in the kit to store our library, then copy the library to that location. By convention, platform-specific libraries go in `dist/package/platform` directories (for example, `~/kit/dist/foo9000/lnxsusex64`):

```
~/kit:>cd dist
~/kit/dist:>mkdir foo9000
~/kit/dist:>cd foo9000
~/kit/dist/foo9000:>mkdir lnxsusex64
~/kit/dist/foo9000:>cd lnxsusex64
~/kit/dist/foo9000/lnxsusex64:>cp ~/lib/libfoo9000.so .
```

Next, we need to create the installation package directory and add the `manifest.isc` file (which describes the package) to it. In its simplest form, the `manifest.isc` file includes only the name of the package, which must be identical to the name of the package directory (`foo9000`).

```
~/kit/package:>mkdir foo9000
~/kit/package:>cd foo9000
~/kit/package/foo9000:>emacs manifest.isc
package: foo9000
```

Without any content the package does not do anything, but in this tutorial we want to do the following:

1. Prompt users for the name of the server hosting the Foo9000.
2. Save this information in a configuration file in the `install-dir\mgr` directory.
3. Copy the library (`libfoo9000.so`) into the instance binary directory.

The package installer performs actions in phases, the most important of which are the following:

- “parameters” phase
- “install” phase

Note: Packages can contain Bourne shell scripts, with the same name as the phase, for each phase. The package installer runs the script for each package at the appropriate time during the phase. If your package script successfully completes its given phase, it returns an error code of 0 explicitly or implicitly via its final command; otherwise it returns a non-zero error code.

The “parameters” phase collects information necessary for the package's installation, typically by prompting users, and should not make any permanent changes to the system. Users are typically given the opportunity to cancel the installation after the “parameters” phase; if they do so, the installation should have had no effect on their system.

The “install” phase modifies the system. During the install phase users should *not* be prompted for information because the install may be unattended or automated.

Some packages do not require information from users and, therefore, do not need a “parameters” script. If the script for a particular phase is not included in a package, no actions are performed for that package during the phase.

Here is our first attempt at a “parameters” script for the `foo9000` package:

```
~/kit/package/foo9000:>emacs parameters
#!/bin/sh
echo "Please enter host name of the Foo9000 imaging server: "
read host
echo "Host $host entered."
```

If we try running this script, as follows, we see that it does indeed prompt us for the host name, which it records in the `host` variable:

```
~/kit/package/foo9000:>sh parameters
Please enter host name of the Foo9000 imaging server:
host1
Host host1 entered.
```

However, what do we do with the `host` value once we've acquired it? When the script is finished running, it will be lost and unavailable when we need to write it to the configuration file during the “install” phase.

Note: Remember that we do not want to create the configuration file now because the “parameters” phase should have no effect on the user's system.

The package installer provides a convenient pair of functions – **Import** and **Export** – that let multiple phases and multiple packages share information. We can use these functions by including them in the `parameters.include` file through the usual shell script mechanism:

```
#!/bin/sh
. parameters.include
echo "Please enter host name of the Foo9000 imaging server: "
read host
echo "Host $host entered."
Export foo9000.host $host
```

The **Export** function takes the name of a parameter variable to export and its value, typically from a variable local to the script. The **Import** function works in reverse: the first argument is the local variable into which you want to import the previously exported value, and the second argument is the name of the parameter variable to which it was exported.

Note: By convention, parameter variables are given a name of *package name.local variable name* (for example, `foo9000.host`).

Since our “parameters” script now collects all the Foo9000 information needed to complete the installation, we can turn to writing the “install” script:

```
~/kit/package/foo9000:>emacs install
#!/bin/sh
. parameters.include
Import host foo9000.host
echo host=$host > ???/mgr/foo9000.cfg
cp ???/dist/foo9000/???/libfoo9000.so ???/bin
```

There are a few details (???? in the preceding script) we need to provide:

- Where is the instance directory in which the install is being created?
- Where is the kit we're installing from?
- Which platform is being installed?

Although we could include these questions in the “parameters” script, that may confuse users because they already entered that information earlier in the install. Instead, we import parameter variables from other packages that can provide the information we need. This is possible because each successful installation using the `irisinstall`, `irisinstall_client` or `irisinstall_silent` scripts (as described in [InterSystems IRIS Installation](#)) creates the `parameters.isc` file, which contains these variables and their values, in the installation directory. The variables in the `parameters.isc` file are listed in the [InterSystems IRIS Installation Parameter File Variables](#) table at the end of this appendix.

Note: For security reasons, the `parameters.isc` file is accessible only by the root user.

In order to use the parameter variables from a particular package, we must inform the package installer that our package (foo9000) depends on the other package and, therefore, our package must be processed later in each phase than the other package. We do this by adding “prerequisite” values to our package's `manifest.isc` file:

```
~/kit/package/foo9000:>emacs manifest.isc
package: foo9000
prerequisite: server_location
prerequisite: legacy_dist
prerequisite: platform_selection
```

Now we can import parameter variables from these packages and use them to complete our install script:

```
~/kit/package/foo9000:>emacs install
#!/bin/sh
. parameters.include
Import host foo9000.host
Import tgtdir "server_location.target_dir"
Import srcdir "legacy_dist.source_dir"
Import platform_family "platform_selection.platform_family"
echo host=$host > $tgtdir/mgr/foo9000.cfg
cp $srcdir/dist/foo9000/$platform_family/libfoo9000.so $tgtdir/bin
```

Our package (foo9000) is nearly complete. The final task is to add our package to the prerequisite list for an appropriate preexisting package. Then, to complete installation of that package, the package installer processes ours. In this case, we want our library to be installed and configured any time an InterSystems IRIS server is installed, so we add our new package to the “database_server” package's prerequisite list inside its manifest.isc file:

```
~/kit/package/database_server:>emacs manifest.isc
package: database_server
prerequisite: legacy_dist
prerequisite: platform_selection
prerequisite: server
prerequisite: server_location
prerequisite: upgrade
prerequisite: available_disk_space
prerequisite: posix_tools
...
prerequisite: isql
prerequisite: zlib
prerequisite: udp
prerequisite: bi
prerequisite: foo9000
```

As you can see, many packages are required to create a server installation, but now, when we run **irisinstall**, our package (foo9000) is configured and installed:

```
~/kit:>sudo ./irisinstall
Your system type is 'SuSE Linux Enterprise Server 10 (x64)'.
Currently defined instances:
IRIS instance 'INSTANCE1'
directory: /home/testUser/INSTANCE1
versionid: 2018.1.0.508.0
conf file: iris.cpf (SuperServer port = 1972)
status: crashed, last used Sat Sep 22 08:37:32 2018
Enter instance name: INSTANCEPACK1
Do you want to create IRIS instance 'INSTANCEPACK1' ? Y
...
Please enter host name of the Foo9000 imaging server:
host1
Host host1 entered.
...
Do you want to proceed with the installation ? Y
...
Installation completed successfully
~/INSTANCEPACK1/bin:>ls libfoo*
libfoo9000.so
~/INSTANCEPACK1/mgr:>cat foo9000.cfg
host=host1
```

13.2 Contents of the parameters.isc File

The following table lists the variables in the parameters.isc file with a description and an example value or a list of valid values.

Table 13–1: InterSystems IRIS Installation Parameter File Variables

Variable name	Description (Valid values) or Example
<i>dist.source_dir</i>	Source directory of the installation media. /iriskit
<i>legacy_dist.source_dir</i>	For legacy purposes, source directory of the installation media. /iriskit
<i>product_info.version</i>	InterSystems product version number. 2018.1.0.100.0
<i>product_info.name</i>	Name of InterSystems product. InterSystems IRIS
<i>platform_selection.platform</i>	InterSystems abbreviation for install platform. lnxrhx64
<i>platform_selection.platform_family</i>	InterSystems abbreviation for install platform family. lnxrhx64
<i>platform_selection.endianness</i>	Platform endian byte order. (big/little)
<i>platform_selection.os</i>	Platform operating system; value of uname command. Linux
<i>posix_tools.user_add</i>	Portable Operating System Interface (POSIX)-compliant user add tool. /usr/sbin/useradd
<i>posix_tools.group_add</i>	POSIX-compliant group add tool. /usr/sbin/groupadd
<i>posix_tools.grep</i>	POSIX-compliant grep utility. grep
<i>posix_tools.id</i>	POSIX-compliant id utility. id
<i>posix_tools.ps_opt</i>	Extend full options for process listing. -ef
<i>posix_tools.gzip</i>	Gnu-compatible zip utility. gzip
<i>posix_tools.shared_ext</i>	Extension for shared library files. so

Variable name	Description (Valid values) or Example
<i>posix_tools.symbolic_copy</i>	POSIX-compliant symbolic copy command. cp -Rfp
<i>posix_tools.tr</i>	POSIX-compliant translation utility. tr
<i>posix_tools.shared_ext1</i>	Alternate extension for shared library files. so
<i>posix_tools.permission</i>	POSIX-compliant permissions applied to selected files. 755
<i>posix_tools.dir_permission</i>	POSIX-compliant permissions applied to selected directories. 775
<i>server_location.target_dir</i>	Target directory of server installation. /test/IRIS
<i>server_location.is_server_install</i>	Indicates whether or not this is a server installation. (N/Y)
<i>server_location.is_nonroot_install</i>	Indicates whether or not this is a nonroot install. (N/Y)
<i>server_location.instance_name</i>	Instance name. IRIS
<i>server_location.is_new_install</i>	Indicates whether or not this is a new install. (N=upgrade/Y=new)
<i>server_location.is_new_directory</i>	Indicates whether or not to create a new directory. (N/Y)
<i>server_location.registry_dir</i>	Location of the InterSystems IRIS registry directory (must be on a local filesystem). /usr/local/etc/irissys
<i>server_location.iris</i>	Directory in which iris resides during installation. /iriskit/dist/lnxrhx64/bin/shared/iris
<i>server_location.is_dr_mirror</i>	Whether this instance is a disaster recovery (DR) mirror member. (N/Y)
<i>postinstall*</i>	Specifies packages to run after parameter file phase. upgrade

Variable name	Description (Valid values) or Example
<i>install_mode.setup_type</i>	Type of installation. (Development/Server/Custom)
<i>unicode_selection.binary_type</i>	Binary type of install. (unicode/eightbit)
<i>unicode_selection.install_unicode</i>	Indicates whether or not to install the Unicode version of the product. (N/Y)
<i>security_settings.iris_user</i>	Effective user for the InterSystems IRIS superserver irisusr
<i>security_settings.iris_group</i>	Effective group for InterSystems IRIS. irisusr
<i>security_settings.manager_user</i>	Owner of the instance. root
<i>security_settings.manager_group</i>	Group allowed to start and stop the instance. develop
<i>security_settings.dbencrypted</i>	Whether to enable an encryption key at startup (0/1)
<i>security_settings.dbenckeyfile</i>	The path of the encryption key. This parameter may be blank.
<i>security_settings.dbenckeyuser</i>	The name of an administrator who can activate the key. This parameter may be blank.
<i>security_settings.dbenckeypassword</i>	The password for the key administrator. This is cleared before the parameter file is stored. This parameter may be blank.
<i>security_settings.personal_database</i>	Indicates whether or not to use the Personal Database feature. (N/Y)
<i>security_settings.initial_level</i>	Initial security settings. (Minimal/Normal/LockedDown)
<i>security_settings.already_secured</i>	If this is an upgrade from a pre-5.1 instance, indicates the need for security settings. (N/Y)
<i>security_settings.password</i>	Password field cleared before the parameter file is stored if running from irisinstall .

Variable name	Description (Valid values) or Example
<i>installer.manifest</i>	Location of the <code>DefaultInstallerClass.xml</code> (the exported %Installer class); for example: <code>/home/user/Downloads/DefaultInstallerClass.xml</code>
<i>installer.manifest_parameters</i>	Location of installer manifest parameters. <code>SourceDir=/home/user/Downloads</code>
<i>installer.manifest_loglevel</i>	Specifies the log level of the manifest. (-1/0/1/2/3)
<i>installer.manifest_logfile</i>	Specifies the log file name. <code>/manifests/IRIS-installManifestLog.txt</code>
<i>port_selection.superserver_port</i>	Superserver port number. 1972
<i>port_selection.jdbcgateway_port</i>	Java Database Connectivity (jdbc) gateway port number. 62972
<i>csp_gateway.configure</i>	Indicates whether or not to configure the Web Gateway for an external web server. (N/Y)
<i>csp_gateway.web_server_type</i>	Type of existing web server for the Web Gateway to use. (Apache/None)
<i>csp_gateway.apache_version</i>	Version of Apache web server
<i>csp_gateway.apache_user</i>	Username for Apache web server
<i>csp_gateway.apache_conf_file</i>	Location of the Apache Web server configuration file. <code>/etc/httpd/conf/httpd.conf</code>
<i>csp_gateway.apache_pid_file</i>	File that records the process id of the Apache web server daemon. <code>/usr/local/apache/logs/httpd.pid</code>
<i>csp_gateway.apache_use32bit</i>	Indicates whether 32-bit architecture is used for the Apache web server. Y/N
<i>csp_gateway.directory</i>	Directory to contain the Web Gateway files.
<i>license_key.enter_key</i>	Indicates whether or not to install the key during installation. N/Y
<i>license_key.license_file</i>	Location of the key file information if the value of <i>enter_key</i> is Y.
<i>agent.user_account</i>	Username for ISCAgent. <code>iscagent</code>

Variable name	Description (Valid values) or Example
<i>agent.user_group</i>	Group name for ISCAgent. iscagent
<i>agent.install</i>	Indicates whether or not ISCAgent is installed. (N/Y)
<i>client_location.target_dir</i>	Target directory of a client-only installation. test/IRIS
<i>client_location.is_client_install</i>	Indicates whether or not it is a client install. (N/Y)
<i>install*</i>	database_server
<i>postinstall*</i>	database_server
<i>japanese_docs.install</i>	Indicates whether or not to install the Japanese documentation sources. (N/Y)
<i>install*</i>	Component name to install.

* The install variable appears several times in the parameter file, once for every component to install. A custom or client-only install conditionally generates any or all of the following:

- dev_kit
- odbc
- cpp_binding
- cpp_sdk
- engine_link_libraries
- light_cpp_binding
- addenda
- install_confirmation
- copyright

