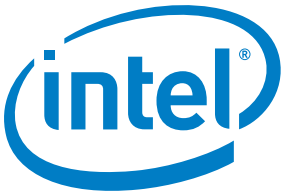


Intel® Cache Acceleration Software (Intel® CAS) Version 2.0 for Linux*

Administrator's Guide

February 2013

Order Number: 328496-001

Disclaimers

Contents

February 2013
Order Number: 328496-001

Figures

Tables

Revision History

Date	Revision	Description
January 2013	001B	Initial Beta release of document.
February 2013	001	Initial public release of document.

1 About This Guide

This guide offers the quickest way to install and begin using Intel® Cache Acceleration Software (Intel® CAS) Version 2.0 for Linux*. This guide assumes users have a basic knowledge of storage and application management, as well as basic Linux* system administration.

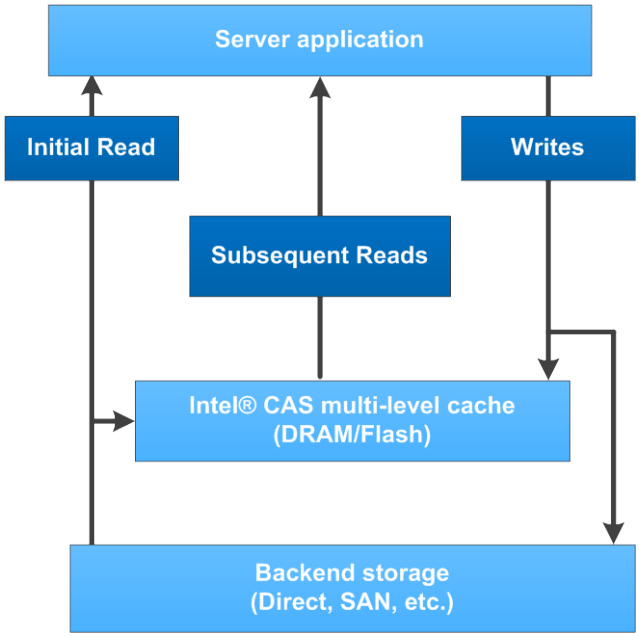
1.1 Intel® Cache Acceleration Software Overview

Intel® CAS accelerates Linux* applications by caching active (*hot*) data to a local flash device inside servers. Intel® CAS implements caching at the server level, utilizing local high-performance flash media as the cache drive media inside the application server as close as possible to the CPU, thus reducing storage latency as much as possible.

The Intel® Cache Acceleration Software installs into the Linux* operating system itself, as a kernel module. The nature of the integration provides a cache solution that is transparent to users and applications, and your existing storage infrastructure. No storage migration effort or application changes are required.

As shown in Figure 1, on an initial read, data is retrieved from backend storage and copied to the Intel® CAS cache. A second read promotes data to system memory. Subsequent reads are returned at high-performance RAM or flash speed. All data is written synchronously to both the backend storage and the cache. When the cache is full, newly identified active data evicts stale data from the cache, utilizing the Intel® CAS proprietary eviction algorithm.

Figure 1. Block Diagram



Intel® CAS for Linux* employs a block-based caching architecture that caches all of the activity on the selected device.

Intel® CAS for Linux* supports the 64-bit version of Red Hat* Enterprise Linux* (RHEL*) and SUSE* Linux* Enterprise Server (SLES*) operating systems running on a physical server (see Section 2.1 for details).

1.2 Reference Documents

The following resources are suggested for assisting with Intel® CAS testing and operations and/or learning more about caching and application I/O performance management.

Table 1. Reference Documents

Document	Location
DT generic data testing program	http://www.scsifaq.org/RMiller_Tools/dt.html
FIO	http://freecode.com/projects/fio

1.2.1 Documentation Conventions

The following conventions are used in this manual:

- `Courier font` - code examples, command line entries, filenames, directory paths, and executables
- **Bold text** - graphical user interface (GUI) entries and buttons

See Glossary for definitions of the terms and acronyms used in this document.

2 Product Specifications and System Requirements

2.1 Supported Platforms

Intel® CAS supports the platforms listed below for 64-bit processors.

Table 2. Supported Platforms

Platform	Notes
Red Hat* Enterprise Linux* (RHEL*) 5.6	Built and configured with x86_64
Red Hat* Enterprise Linux* (RHEL*) 6.1	Built and configured with x86_64
SUSE* Linux* Enterprise Server (SLES*) Version 11 Service Pack 1	Built and configured with x86_64

2.2 System Requirements

The table below lists system requirements for Intel® CAS.

Table 3. System Requirements

Memory	Minimum 4GB system RAM will support up to an initial 200GB of SSD cache. For each additional 200GB of SSD cache, an additional 2GB of system RAM will be required.
CPU Overhead	Intel® CAS only consumes approximately 2% of CPU resources. It is recommended that the CPU is not at maximum capacity.
Flash/SSD	Linux* supported Flash device (SAS, SATA, PCIe, Fiber Channel, RAID) 60GB or larger recommended. Validated and supported on the Intel® SSD Data Center Family. **NOTE:** Refer to Appendix B on the optimal configuration for the Intel® SSD 910 Series and Intel® CAS for Linux. 512B logical sector size is required.
Storage	Primary storage device including SAN, local disk, RAID, iSCSI, or Fiber Channel. Operating system must be on separate partition or drive than the Core storage device to be cached. 512B logical sector size is required.
File System	ext3 file systems are required for primary storage or core storage. **NOTE:** Each core device is limited to 16TB of capacity. Block size must be 4096 bytes; Sector size must be 4096 bytes.

3 Installing Intel® Cache Acceleration Software

This section describes how to install Intel® CAS.

The installation package consists of loadable kernel modules that provide the caching layer and management CLI that controls caching behavior.

3.1 Configuring the Flash Device

Prior to configuring Intel® CAS, you must have a Flash media device installed. The Flash device can be any SSD drive or any PCIe Flash Card supported by the Linux* operating system (see Table 3 for details).

Physically install the Flash device in the server, and make note of the device created corresponding to the drive, following the hardware product installation guide.

Assuming you have identified a core device (the device to be cached), make sure that it is formatted and it is **not (auto) mounted** when the OS boots up.

3.2 Installing the Intel® CAS Software

Note: You must have root privileges to install the software.

Note: If an earlier version of the software is already installed, you must uninstall it before continuing. See Section 3.3 for details on uninstalling.

Note: For a SLES platform, you must change a module before running the installation script. Edit the `/etc/modprobe.d/unsupported-modules` file, find the variable `allow_unsupported_modules` and set it to `1`.

1. Download or copy the Intel® CAS installer file to your home directory on the target Linux* server. The installation instructions use the example of `/~` (equivalent of `$HOME)` on the server file system.
   The installer file name is in the format:
   `Intel-CAS-X.XX.XX.XXXX-distro.sh` where:
   - `X.XX.XX.XXXX` is the version information
   - `distro` is one of the following: `rhel6`, `rhel5`, or `sles11sp1`
2. Change to home directory and switch to root:
   `$ cd ~ && su`
3. Make the installer file executable:
   `# chmod a+x ./Intel-CAS-X.XX.XX.XXXX.sh`
4. Launch the installation:
   `# ./Intel-CAS-X.XX.XX.XXXX.sh`

Note: For more information about the installer, call
`./Intel-CAS-X.XX.XX.XXXX.sh --help`

5. You will be prompted with a license which you will have to read and accept to proceed with the installation.

Note: To automatically accept the license, call:
`./Intel-CAS-X.XX.XX.XXXX.sh --accept-license.`
This flag is for convenience only; the flag should be used with caution as it is the equivalent to having read and accepted the license agreement.

3.3 Uninstalling the Software

Stop Intel® CAS as described in Section 5.1.2.

Note: All applications using Intel® CAS must be closed and file systems must be unmounted before uninstalling.

Assuming the Intel® CAS installer is in your home directory as described in Section 3.2:

1. Change directory to your home and switch to root:

```
$ cd ~ && su
```

2. Make the installer file executable, if it wasn't already:

```
# chmod a+x ./Intel-CAS-X.XX.XX.XXXX-distro.sh
```

3. Launch the uninstallation:

```
# ./Intel-CAS-X.XX.XX.XXXX-distro.sh --uninstall
```

4 Configuring Intel® Cache Acceleration Software

Note:

1. You must be logged in as root to start, stop, or configure the Intel® CAS software.
2. You cannot accelerate the partition that the operating system is on.

4.1 Using the Configuration Utility

Intel® CAS is configured using a command line utility called `intelcas`, which is installed by default in `/usr/sbin`.

The configuration command set is listed in Table 4. If you execute the configuration utility in a Linux* shell without any arguments, this list of commands is returned.

Note: Intel recommends using a partition instead of a drive for the primary storage/core device. This can prevent errors caused by accidently selecting the OS drive for caching.

Table 4. Configuration Commands

Command	Description

`add [-f] <cache_id> <state> <cache_device> <core_device>`
 	
Example:
`intelcas add 1 1 /dev/sdc /dev/sdb1`
	Adds a cache/core device mapping. The cache can be loaded with an old state (previous cache metadata will not be marked as invalid) or with a new state (previous cache metadata marked as invalid).
	where:
	`[-f] =` forces creation of a cache even if a filesystem exists on the cache device.
	WARNING: This will destroy the filesystem and should be used with caution.
	`<cache_id>` = Cache ID to create. Value between 1 and 16384
	`<state> =` Specifies whether to reinitialize the cache
	0 – Do not reinitialize the cache
	1 – Reinitialize the cache
	`<cache_device> =` Location of the Flash/SSD device.
	`<core_device> =` Location of the primary storage/core device.

`remove <cache_id>`
Example:
`intelcas remove 1`
	Deletes the cache/core device mapping, which is one way to disable caching.
	where:
	`<cache_id>` = Cache ID to delete. Value between 1 and 16384

Command	Description

`reset_counters <cache_id>`
where:
`<cache_id>`
	Resets the performance and status counters for a specific cache instance.
	where:
	`<cache_id>` = Cache ID to reset. Value between 1 and 16384.

`version`	Reports the Intel® CAS Kernel Module and command line utility version numbers.
`help`	Generates a list of supported commands with their respective arguments/configuration options.

`list`
Example:
`intelcas list`
 	
Returns:
`CacheID CacheDevice CoreDevice WritePolicy Status`
`1 sdb1 sdc1 WriteThrough Running`
`2 sdb2 sdc2 WriteThrough Running`
`3 sdb3 sdc3 WriteThrough Running`
`4 sdb4 sdc4 WriteThrough Running`
	Provides a list of every cache instance in the system currently running. It also lists:
	• Flash/SSD device used in the instance
	• Storage device used in the instance
	• writePolicy of the instance (WriteThrough by default)
	• Status of the instance

`stats <cache_id>`
Example:
`intelcas stats 1`
	Prints performance and status counters for a specific cache instance.
	Detailed output is listed in Section 6.1.

4.2 Configuration Example

In this example of Intel® CAS configuration, the following is assumed:

- The cache device (flash drive) is `/dev/sdc`
  The cache device is either a raw block device or previously set up member of a cache/core mapping. In the case of a new cache initialization, all data previously stored on the cache device is overwritten.
- The core device (primary storage) to be cached is `/dev/sdb1`
  The core device may contain old data and have an optional file system. Intel® CAS requires that the file system is of type EXT3 and uses a block size of 4096 and a sector size of 4096.

Note: Intel recommends that you backup all data on your core device before completing these steps.

1. Verify that the Intel® CAS module loaded properly:

   ```
   lsmod | grep intelcas
   ```

Note: If Intel® CAS is not loaded, contact customer support because your OS/kernel configuration may be different from what is officially supported by Intel® CAS.

2. If the Intel® CAS module is not already loaded, follow the installation instructions in Section 3.2.
3. Ensure that both the cache device (`/dev/sdc`) and the primary storage device (`/dev/sdb1`) are formatted but **not mounted**. Enter the following command to display all mount points:

   ```
   df –k
   ```

4. If needed, un-mount the device(s) by entering one or both of the following commands:

   ```
   umount /dev/sdc
   ```

   ```
   umount /dev/sdb1
   ```

   Also remove any entries from `/etc/fstab` or similar mechanisms that auto-mount the cache and core drives on OS boot-up.

5. To set up a write-through cache numbered 1 with new metadata, using `sdc` as the cache drive, and `sdb1` as the core drive, enter the command:

   ```
   intelcas add 1 1 /dev/sdc /dev/sdb1
   ```

   The add command creates a new device `/dev/intelcas1`

Note: You may notice a brief delay after entering the `intelcas add` command. Typically, this is less than 60 seconds.

To confirm the creation of this device, enter one of the following commands:

```
ls /dev/intelcas1
```
or
```
intelcas list
```
or
```
intelcas stats 1
```

If you wish to mount the new cache device to the Linux* file system, perform steps 6 through 8.

6. (Optional) The core device must contain a file system if you want to mount Intel® CAS. If you want to mount Intel® CAS and your core device does not contain a file system, you must create one, using the command:

   ```
   mkfs –b 4096 –t ext3 /dev/intelcas1
   ```

7. A directory is required to mount the new cache device. In this example, the mount point is `/mnt/cache/c1`. Enter the command:

   ```
   mount /dev/intelcas1 /mnt/cache/c1
   ```

Note: Intel® CAS ensures that no changes are required to the application; it can use the same file system mount point (for example, `/local/data`).

Note: If your application uses the raw device directly (for example, some installations of Oracle*), then you must change the application to use the cache device instead.

8. It may be necessary to change permissions for the mount point to allow applications access to the cache. The example below shows two commands that give user `bob` administrator privileges to the mount point:

   ```
   chown bob:adm /mnt/cache/c1 –R
   ```
   or
   ```
   chmod  g+w  /mnt/cache/c1
   ```

   This command provides privileges to all users:

   ```
   chmod 777 /mnt/cache/c1
   ```

Intel® CAS is now enabled on the system and files to be accelerated can be written to `/mnt/cache/c1`.

5 Running Intel® Cache Acceleration Software

5.1 Startup and Shutdown

5.1.1 Starting the Intel® CAS Software

Follow the instructions below to start Intel® CAS.

1. Start the Intel® CAS device(s) using the following command format for each device.

   Once the cache device and core device have been identified, use the following command to enable the caching pair. When starting after a fresh installation, you must initialize the cache device by entering a 1 in the state field.

   ```
   intelcas add <cache_id> <state> <cache_device> <core_device>
   ```

   Example: `intelcas add 1 1 /dev/sdc /dev/sdb1`

   If the system has crashed or had an unplanned shutdown, refer to Section 5.1.3.
2. Mount the Intel® CAS device as in the same manner used in previous mounts.

   ```
   mount /dev/intelcas1 /mnt/cache/c1
   ```
3. Your Intel® CAS device is ready for use.

To start the Intel® CAS software automatically on a reboot, add a script to run the `intelcas add` command followed by mounting of the `intelcas` device as documented in Section 4.2.

The following items should be considered:

- Intel® CAS must be started **before** any applications that depend on the Intel® CAS device or mount point.
- Depending on the size of the device, this may significantly increase the amount of time to boot. No progress indicators are displayed and it may appear that the system is in a hung state.

5.1.2 Stopping the Intel® CAS Software

To stop Intel® CAS cleanly on shutdown, add a script to un-mount the `intelcas` device followed by the `intelcas remove` command.

Intel® CAS must be stopped **after** any applications that may try to access the Intel® CAS device.

The following example uses `cacheID = 1`. To stop Intel® CAS, login as root and run:

```
# umount /dev/intelcas1
# intelcas remove 1
# rmmod intelcas
```

Depending on the size of the device, removal may take a significant amount of time (approximately 30 minutes).

5.1.3 Handling an unplanned shutdown

After an unclean shutdown using write-through, the cache must be reinitialized with new metadata. If you try to load an old state, you are prompted with a message stating an unclean shutdown has occurred and the cache must be reinitialized. To do this, follow the steps in Section 5.2.

5.2 Clearing the Cache

To clear the cache (invalidate the entire cache), perform the following:

1. Login as root.
2. Enter the add command and set the `<init>` argument to `1` as shown below:

   ```
   intelcas add 1 1 /dev/sdc /dev/sdb1
   ```

   This re-initializes the cache instead of loading old state. For details on the `add` command, see Table 4.

Caution: If the file system changes while caching is not running, it is important that cache be cleared using this method.

Intel® CAS maintains state across a reboot. If the system is rebooted without starting the caching subsystem, then the state of the cache data may become out of sync with data in primary storage. In this case, restarting the cache system without clearing the cache results in data corruption.

If the cache contains any dirty data, the data is automatically transferred to the disk during the cache removal procedure.

5.3 Rebooting and Power Cycling

Intel® CAS devices do not automatically become available for use after a reboot or power cycling of the system. It is necessary to recreate the cache pairing of the core storage device and the caching device. Use the following steps upon every system reboot or power cycling to continue to use Intel® CAS cache devices.

1. Ensure that the Intel® CAS kernel module has loaded properly using the following command:

   ```
   /sbin/lsmod | /bin/grep intelcas
   ```

   If the intelcas module is loaded, continue with step 2.

If the module did not load properly, check that all the components of Intel® CAS are installed. You can use the `–list` option on the install file, for example:

```
./Intel-CAS-X.XX.XX.XXXX.sh --list
```

If any of the components are missing, you will have to uninstall Intel® CAS and reinstall. See Section 3.3 and Section 3.2 for more details.

2. Ensure your devices are the same as prior to the reboot or power cycle of the system.

   Change directory to either `/dev/disk/by-uuid` or `/dev/disk/by-id`

   Use the command `ll` to view the association of physical devices and their identifiers.

   Verify that the identifiers for the device and/or partitions are correctly associated with the correct physical devices. It is critical to ensure that the identifier for the caching SSD is known. Using the wrong device will lead to undetermined system behavior and the possible loss of data.

3. Start the Intel® CAS device(s) using the following command format for each device.

   Once the cache device and core device have been identified, you may use the following command to enable the caching pair.

   ```
   intelcas add <cache_id> <state> <cache_device> <core_device>
   ```

   If the system has crashed or had an unplanned shutdown, then you must reinitialize the cache by entering `1` in the state field as shown in the example below.

   ```
   intelcas add 1 1 /dev/sdc /dev/sdb1
   ```

   If it was a regular planned shutdown, use `0` in the state field.

   ```
   intelcas add 1 0 /dev/sdc /dev/sdb1
   ```

4. Mount the Intel® CAS device as in the same manner used in previous mounts.

   ```
   mount /dev/intelcas1 /mnt/cache/c1
   ```

5. Your Intel® CAS device is ready for use.

6 Monitoring Intel® Cache Acceleration Software

6.1 Viewing Cache Statistics

Example:

```
# intelcas stats 1
```

Returns:

```
CacheID:                          1
CacheDevice:                      sdb1
CoreDevice:                       sdc1
WritePolicy:                      WriteThrough
Status:                           Running

Cache Size(4K Blocks):            7253025
Core Size(4K Blocks):             30523492
Cache Occupancy(4K Blocks):       7253025
Longest Queue Length:             54
Queue Length:                     22
Cache misses completed from core to be written to cache: 0

+---------------------------+----------------------+---------+
| Request statistics        |                Count |       % |
+---------------------------+----------------------+---------+
| Req Reads                 |              4070487 |     31% |
| Req Writes                |              8740017 |     68% |
| Req Cache hit             |               868616 |     21% |
| Req Sequential hits       |               756520 |     18% |
| Req Non sequential hits   |               112096 |      2% |
| Req Partial Misses        |               591103 |     14% |
| Req Full Misses           |              2610768 |     64% |
| Req Aborted               |                    0 |      0% |
| Req Yielded               |                59940 |      0% |
| Sequential requests (1p)  |              1482744 |     11% |
| Random requests     (1p)  |             11327760 |     88% |
| Sequential requests (16p) |              1595799 |     12% |
| Random requests     (16p) |             11214705 |     87% |
| Sequential requests (32p) |              1640355 |     12% |
| Random requests     (32p) |             11170149 |     87% |
| Sequential requests (64p) |              1640381 |     12% |
| Random requests     (64p) |             11170123 |     87% |
| Sequential requests (128p)|              1640441 |     12% |
| Random requests     (128p)|             11170063 |     87% |
| Average Seek Distance     |                    0 |      0% |
|---------------------------|----------------------|---------|
| Total Requests            |             12810504 |         |
|---------------------------|----------------------|---------|
```

```
+---------------------------+----------------------+--------+
| Block statistics          |                Count |      % |
+---------------------------+----------------------+--------+
| Blocks Reads              |             33785414 |    32% |
| Blocks Writes             |             68749962 |    67% |
| Blocks Sequential hits    |              5929318 |    17% |
| Blocks Non sequential hits|              1576526 |     4% |
| Blocks Compulsory misses  |              2174297 |     6% |
| Blocks Invalid misses     |                    0 |     0% |
| Blocks Conflict misses    |             21034499 |    62% |
| Blocks Updated            |                    0 |     0% |
| Blocks abort update       |                    0 |     0% |
|---------------------------|----------------------|--------|

|--------------------|-------|-------|-------|----------|
| Pages statistics   |   Min |   Max |   Avg | Variance |
|--------------------|-------|-------|-------|----------|
| Read               |     1 |    16 |     8 |       45 |
| Write              |     1 |    16 |     7 |       52 |
|--------------------|-------|-------|-------|----------|
```

6.2 Resetting the Performance Counters

The performance counters are automatically reset every time the cache is started.

To manually clear the performance counters, use the `reset_counters` command. See Table 4 for details.

7 Frequently Asked Questions

How do I contact support?

Contact support by phone at 800-538-3373 or at the following URL: http://www.intel.com/support/ssdc/cache/cas/

How do I test performance?

In addition to the statistics provided (see Section 6.1), third-party tools are available that can help you test I/O performance on your applications and system, including:

- FIO (http://freecode.com/projects/fio)
- dt (http://www.scsifaq.org/RMiller_Tools/dt.html) for disk access simulations

Where are the cached files located?

Intel® CAS for Linux* does not store files on disk; it uses a pattern of blocks on the SSD as its cache. As such, there is no way to look at the files it has cached.

How do I delete all the cache files?

Stop the Intel® CAS software as described in Section 5.1.2, then uninstall the software as described in Section 3.3.

Does Intel® Cache Acceleration Software support write-back caching?

No. The current release of Intel® CAS for Linux* supports write-through mode. In write-through mode, both read and write operations build the cache data, but only read operations are accelerated (writes are written into the cache and to backend storage concurrently). Data integrity is never a concern with write-through.

How do I change the cache drive used by the Intel® CAS Service?

Stop the caching system (Section 5.1.2) and use the `intelcas add` command for your new core drive to be cached.

Must I stop caching before adding a new pair of cache/core devices?

No, new `intelcas` instances can be created while other instances are running.

Can I assign more than one core device to a single cache?

No, only one core device may be used. If you wish to use multiple HDDs as a core device, they must appear as a single core device through the use of a system such as RAID.

Can I add more than one cache to a single core device?

No, if you wish to map multiple cache devices to a single core device, the cache devices must appear as a single device through the use of a system such as RAID. (However, this scenario has not been validated by Intel.)

Where is the log file located?

All events are logged in the standard Linux* system logs. You can use the `dmesg` command or inspect the `/var/log/messages` file. During testing or kernel debugging, use the command `echo 8 > /proc/sys/kernel/printk` so that all messages are logged.

Typical log sample of successful cache initialization:

```
[Intel® CAS] Inserting cache 1
[Intel® CAS] Parameters (policies) accepted:: 1 1 1
[Intel® CAS] Creating the device object (of size 32 bytes)...
[Intel® CAS] Reducing cache space by 317145088 bytes for allocation
of persistent metadata.
[Intel® CAS] Disk subxts = 122096384, Cache subxts = 7734668, Cache
lines = 7734668
[Intel® CAS] Allocating collision table :: ==> 7734668 entries, 40
bytes each for a total size of 295 MBytes
[Intel® CAS] Allocating hash table :: ==> 1933667 entries, 4 bytes
each for a total size of 7 MBytes
[Intel® CAS] Allocating concurrency control table :: ==> 699050
entries, 24 bytes each for a total size of 15 MBytes
[Intel® CAS] Allocating persistent metadata lock table :: ==> 75535
entries, 24 bytes each for a total size of 1 MBytes
[Intel® CAS] Allocation of metadata complete
[Intel® CAS] Saving cache state...
[Intel® CAS] Done saving cache state!
[Intel® CAS] Starting 1 threads for cache fill
[Intel® CAS] Starting threads for I/O requests
[Intel® CAS] Cache 1 successfully added
```

Typical log sample of successful cache removal:

```
[Intel® CAS] Removing cache 1
[Intel® CAS] Stopping request thread 1...
[Intel® CAS] Thread 1 stopped
[Intel® CAS] Stopping fill thread 1...
[Intel® CAS] Thread 1 stopped
[Intel® CAS] Flushing dirty data to disk.
[Intel® CAS] Saving cache state...
[Intel® CAS] Done saving cache state!
[Intel® CAS] Cache 1 successfully removed
```

Why do tools occasionally report data corruption with Intel® CAS?

Some applications, especially micro-benchmarks like dt and FIO, may use a device to perform direct or raw accesses. Some of those applications are written to not request from the device its alignment and block size restrictions but rather configure these values explicitly, for instance via user parameters. Intel® CAS supports a 4K block size and alignment. Thus in order for these programs to work, a 4K value both for block size and alignment for the cache device must be used.

Do I need to partition the cache device?

No. If you do not specify a partition, Intel® CAS uses the entire device as the cache device.

Can I use a partition on a SSD as a cache device?

Yes, however, using the entire SSD device as the cache is recommended for best performance.

Do I need to format the partition or the device configured as the cache device?

No; the cache device has no format requirement. If any formatting is used, it is removed by the caching software.

What is the default/optimal block size for the core device?

4KB

Appendix A Glossary

Term	Definition
Cache	The transparent storage of data so that future requests for that data can be served faster.
Cache Hit	When requested data is contained in (and returned from) the cache.
Cache Miss	When requested data is not in the cache, and therefore must be retrieved from its primary storage location.
Core Device	The device to be cached.
Dirty data	When data is modified within cache but not modified in main memory, the data in the cache is called dirty data.
Guest	An operating system running on a Virtual Machine (VM) environment.
Host	The operating system running (hosting) the Virtual Machine (VM) environment, on which guest Operating Systems can be installed.
I/O	Abbreviation for Input/Output as it relates to the flow of data.
NAS (Network-Attached Storage)	File-level data storage (such as fixed disk and magnetic tape drives) that are directly linked to a storage area network or other network.
Primary Storage	As it relates to caching, the storage system or location (DAS, SAN, NAS, etc.) where the data is stored.
SAN (Storage Area Network)	Framework used to attach remote computer storage devices to servers. Storage devices appear as if they were attached locally to the operating system.
SSD (Solid-State Disk)	A device used for data storage that utilizes memory chips instead of a revolving disk.
Write-Through	A write caching policy where every write to the cache causes a synchronous write to primary storage.

Appendix B Setting Up the Intel® SSD 910 Series Device

When using an Intel® SSD 910 Series device as a cache drive, it is highly recommended that the entire SSD cache drive is utilized as a whole either "RAW" (using RAID-0) or, if partitioning is required, use only one single partition (using the `parted` command). For optimal caching performance, the Intel® SSD 910 Series device should not be broken up into multiple partitions.

"RAW" partition using RAID-0:

If the entire SSD cache drive is to be RAIDed together, the following command may be used to do so optimally.

Note: The `mdadm` command does not save the configuration across system reboots or power downs. You will need to add the following command in the system files such as: `/etc/madadm.conf` and `/etc/fstab` if your filesystem is mounted on RAID.

```
mdadm --create --verbose /dev/md0 --level=0 --chunk=64 --raid-
devices=2 /dev/sdx /dev/sdy

# where md0 is the resultant output RAID-0 logical drive.
# sdx and sdy are the corresponding two RAID input drives
(applicable to the 400 GB Intel® SSD 910 Series device)
# RAID level is set to 0.
# Chunk size is set to 64K
```

Single partition using the `parted` command:

The following commands can be used to create a single 200 GB partition cache drive using the 400GB Intel® SSD 910 Series device:

```
parted -s -- /dev/sda mkpart primary 1 191488

# where sda is the single 200 GB partition created
```

Note: Intel recommends using the `parted` utility instead of `fdisk`, because older revisions of `fdisk` may cause misaligned partitions (`util-linux-ng` package 2.17.2 and lower revisions). The `parted` tool will align partitions to 1MB boundaries by default.