Intel® Cache Acceleration Software v3.8 for Linux*

Release Notes

December 2018

Document Number: 328497-025US

Contents

Revision History

Document Revision	Software Release Description	Software Release Date
001B	Initial Beta release of document.	January 2013
001	Initial public release of document.	February 2013
002	Documentation Updates for v2.1	May 2013
003	Documentation Updates for v2.5	August 2013
004	Documentation Updates for v2.6	December 2013
005	Documentation Updates for v2.6.1	April 2014
006	Documentation Updates for v2.7	May 2014
007	Documentation Updates for v2.7 GA	July 2014
008	Documentation Updates for v2.7.1 GA	October 2014
009	Documentation Updates for v2.8 GA	December 2014
010	Documentation Updates for v2.9 GA	June 2015
011	Documentation Updates for v3.0 GA	December 2015
012	Documentation Updates for v3.1 GA	May 2016
013	Documentation Updates for v3.5 GA (QEMU)	April 2017
014	Documentation Updates for v3.5 Beta	June 2017
015	Documentation Updates for v3.5 GA	July 2017
016	Documentation Updates for v3.5.1 Beta	December 2017
017	Documentation Updates for v3.5.1 GA	January 2018
018-I	Documentation Updates for v3.6 GA	May 2018
019-I	Documentation Updates for v3.6.1 GA	June 2018
020-I	Documentation Updates for v3.7 Beta	July 2018
021-I	Documentation Updates for v3.7 GA	August 2018
022-I	Documentation Updates for v3.8 Beta	October 2018
023-I	Documentation Updates for v3.8 Beta (Kernel/Ansible updates)	October 2018
024-I	Documentation Updates for v3.8 GA (Kernel/Ansible updates)	November 2018
025	Documentation Updates for v3.8 GA (GM)	December 2018

1 Release Description

This document describes features, extensions, and deviations (if any) from the release functionality presented in the *Intel® Cache Acceleration Software (Intel® CAS) for Linux* Administrator Guide*.

1.1 Features

- **QEMU Hot-Plug Support.** Users are now able to attach or detach a cache drive from a QEMU cache instance without stopping or restarting QEMU.
- **Write-invalidate.** This new QEMU cache mode allows users to invalidate data in the caching drive during a write operation.
- **Fallback Pass-through**. This new features allows QEMU users to fallback to pass-through mode caching after a certain number of errors have been encountered in the caching drive.
- **Ansible* Integration**. This new feature makes use of Ansible® via a plugin module, intelcas-ansible, to automate the configuration and installation of Intel CAS.
- **Maximum I/O Size Enhancement**. The maximum I/O size exposed on by Intel CAS is now inherited from the core device only instead of the smaller value from cache and core devices.

1.2 Supported SSD Drives

Any Linux flash device (NVMe, PCIe*, SATA, SAS, Fibre Channel, RAID) is supported, and can be direct attached, expander attached, or attached via Storage Area Network (with a single worker).

Although Intel CAS supports most SSD brands, some Intel SSD's contain advanced features that can be addressed by Intel CAS to enable enhanced caching performance.

1.3 Limitations

This release has the following known limitations:

- An “ext3” file system is limited to 16 TiB max core drive capacity (this is an intrinsic limitation of the “ext3” file system, not an Intel CAS limit.).
- For VMs, paravirtualized drivers are not supported.
- System sleep (S3) and hibernation (S4) power states and resumption from these states is not supported.
- Hot-plugging of either core or cache drive is not supported.
- For best performance, it is recommended to use the *noop* IO scheduler on the cache device (SSD).
- NUMA node IO transfers can result in performance degradation on multi-CPU platforms with high number of worker threads, if the threads send IO to a CPU that is not directly attached to the PCIe* device that is the target of the IO. This is a known limitation of multi-CPU platform architecture and not of Intel CAS.
- Linux file systems with less than 4096 byte (4KiB) block sizes may encounter less than ideal performance when workload mix has a high percentage of writes in write-back mode.
- Sub-partitioning of an intelcas virtual block device (e.g. /dev/intelcas1-1) that is built on an existing partition of the core device (eg. /dev/sdc1) is not supported.
- Stack overrun can occur in kernels with 8KiB stack size that are caching to an NVMe* SSD, when IO payloads have gaps. This is due to Linux implementation of NVMe* IO using Physical Region Page (PRP) lists, which require there to be no gaps in the IO payload. When a payload is received with IO gaps, the kernel splits it into many sub IOs that meet PRP list requirements. As these IOs complete, it is possible to exceed the 8KiB stack size limit, which would result in a stack overrun. This is a known limitation of Linux NVMe implementation and not of Intel® CAS. To avoid this issue, it is recommended to use a kernel with 16KiB stack size, such as RHEL 6.7 or newer.
- With older versions of LVM, you must first create a partition on the core device (for example, if the core device is /dev/sdc, you must create /dev/sdc1) prior to accelerating that device with Intel CAS device and creating the physical volume or creation of the logical volume will fail. If you see the following warning: *“WARNING: Ignoring duplicate config node: types (seeking types); Found duplicate PV”*, then you must use this workaround.
- When utilizing the Atomic Writes feature, you must reboot the servers after formatting the NVMe devices, before any further changes are made to the system.
- When using large RAID volume starting cache on such device could take longer time than usual. This is not a CAS issue, but is related to how RAID device handles DISCARD/TRIM requests.
- In SysV based Linux distributions (ex. CentOS/RHEL 6.x) if the filesystem created on the CAS device is being used by a user process, the CAS service is not able to umount that filesystem and CAS cannot be stopped. In these distributions there are no methods to provide dependencies between various services for proper ordering of startup/shutdown. To prevent startup issues after this situation we recommended using the load option in intelcas config file. This issue does not affect SystemD based distros.
- It is not recommended to create a file system on an Intel CAS device. The creation process may take an inordinate amount of time. If, however, a file system is created on the device, it may be used with Intel CAS without issues.
- Certain methods for creating partitions on devices (CentoOS/RHEL 6.x) may result in a partition alignment message concerning possible performance degradation. This message may be disregarded.

1.4 Supported Platforms

The following table lists the supported platforms for 64-bit processors.

Table 1-1 Supported Platforms

Operating System	Kernel
Red Hat* Enterprise Linux* (RHEL*) 6.9	x86_64, Kernel 2.6.32-696.20.1
Red Hat* Enterprise Linux* (RHEL*) 6.10	x86_64, Kernel 2.6.32-754
Red Hat* Enterprise Linux* (RHEL*) 7.4	X86_64, Kernel 3.10.0-693.17.1
Red Hat* Enterprise Linux* (RHEL*) 7.5	X86_64, Kernel 3.10.0-862
SUSE* Linux* Enterprise Server (SLES*) Version 12 SP2	x86_64, Kernel 4.4.103-92.56-default
SUSE* Linux* Enterprise Server (SLES*) Version 12 SP3	x86_64, Kernel 4.4.103-6.38-default
Other distros - Intel CAS will install and compile from source on other distros and other kernels, but the user may be required to reproduce any issues on a validated distro & kernel to receive support.	Other kernels

The following table lists the OS combinations that Intel CAS for QEMU supports.

Table 1-2 Supported Intel CAS for QEMU OS Combinations

QEMU Version	Supported Operating System	Kernel
2.1.2	CentOS* 7.2	x86_64, Kernel 3.10.0-327

2 Known Issues

The following are current known issues:

Table 2-1 Known Issues

Reference	Description	Impact/Status/Workaround
	None	

3 Resolved Issues

The following table lists issues Intel has resolved or closed since the previous release.

Table 3-1 Resolved Issues

Reference	Description	Impact/Status/Workaround
	None	