FairCom White Paper

Linux Performance and Safety Advisory

Audience

Developers

Subject

Learn how a known Linux bug affects data intregity and the precautions you should take

FairCom®

Contents

1. Linux Implementation Affects Data Safety & Performance

Tech Bulletin: FairCom DB V10.3 - Performance Change on Linux Systems

The FairCom DB V10.3.1.21834 (Build 140307) release changed behavior from prior Linux releases:

- **Releases prior to this build were vulnerable to a data integrity issue due to a known issue in the Linux kernel.**
- **Releases beginning with this build (and later) address this issue to provide data integrity with some loss in performance.**

FairCom DB releases prior to the build listed above are vulnerable to this issue and should be updated to the latest release.

> **WARNING**: By default, most current Linux systems might not be configured in a way that guarantees recoverability of data in the event of a loss of power to the system. This paper discusses the effects of changes to address this issue and a possible impact on performance.

This document highlights a potential transaction data loss on recovery operations for FairCom DB Servers running on Linux systems due to the implementation of the Linux file system. A workaround allows recovery in case of a power failure. The configuration keywords discussed below can have an impact on performance, but may be necessary to ensure data integrity depending on your environment.

1.1 Workaround for Synchronous Write Mode on Linux

A limitation in all implementations of the Linux file system has caused occurrences of FairCom DB errors **64**, **66**, **96**, and **571**.

In a system power-off test on a Linux system, we noticed cases in which automatic recovery failed with the errors listed above. This was because some data had not been written to disk in the transaction logs, data files, and index files.

These errors were seen when using the following FairCom DB Server configuration options:

`COMPATIBILITY LOG_WRITETHRU`	Affects transaction log files
`COMPATIBILITY TDATA_WRITETHRU`	Affects TRNLOG data files
`COMPATIBILITY TINDEX_WRITETHRU`	Affects TRNLOG index files

`COMPATIBILITY SYNC_LOG`	Affects transaction logs and TRNLOG index files (this is an old option that we don't expect to be used anymore)

On Linux systems, when these configuration options are used, an **fcntl()** call is supposed to enable the O_SYNC file mode. However, Linux ignores this mode, which causes disk writes to be stored in the file system cache instead of being written to disk before the write call returns. **This gives very fast performance, but it does not provide the data integrity that was intended.**

The following Linux kernel bug report confirms this limitation in the file system portion of the Linux kernel: **Linux Kernel Bug Tracker - 5994** (**https://bugzilla.kernel.org/show_bug.cgi?id=5994**). The report includes a proposed patch, but we found no indication that the patch has been applied to the Linux kernel for any Linux distributions as of February 2014.

Beginning with FairCom DB V10.3.1.21834 (Build 140307), a change in the FairCom DB logic resolves the problem by enabling O_SYNC when opening or creating a file instead of calling **fcntl()** after opening the file. This change also disables the use of `COMPATIBILITY SYNC_LOG` on Linux because that keyword would turn off the O_SYNC bit.

Note: Releases prior to FairCom DB V10.3.1.21834 (Build 140307) are vulnerable to this issue and should be updated to the latest FairCom DB release.

The Effects of the Change

When rebooting with a power source (e.g., an uninterruptable power supply, UPS) connected to the system with the disk write cache enabled, it was possible to recover the data without any errors. However, when rebooting by disconnecting the power source with the disk cache enabled, we saw the errors mentioned above during automatic recovery.

Based on these results, it appears safe to keep the disk write cache enabled if there is an uninterruptable power supply (UPS) connected to the system. The UPS must be properly configured to safely shut down the computer and FairCom DB Server before the UPS battery fails.

2. Caching Considerations

The use of write barriers and commit delay has an impact on data safety and performance.

Write barriers enforce the orderly flushing of caches to disk by ensuring that all requests issued at levels above the barrier must be satisfied before continuing. Write barriers affect performance because they require the write to be complete before the write call continues. Asynchronous (cached) vs. synchronous (flushed) writes are examined in the section titled Write Barriers (page 3).

Tests were conducted to determine the performance impact of commit delay and disk write cache. The results of these tests and configuration suggestions are presented in the section titled Commit Delay and Disk Write Cache (page 4).

2.1 Write Barriers

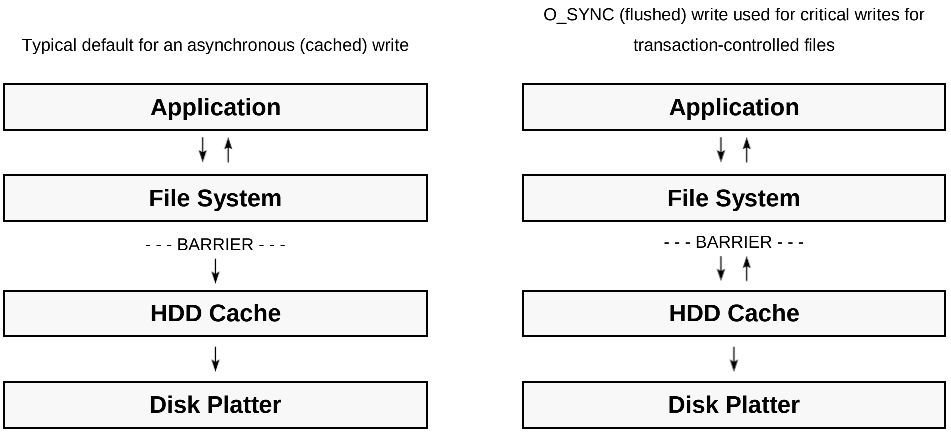
Linux file systems offer several levels of caching, including the user process, file system, transport layer, storage controller, and the hard disk drive (HDD), to name just a few. Think of these layers of caching as a stack with the application at the top and the physical media to which the data is written (the hard disk platter) at the bottom.

When a disk has write cache enabled and the system loses power, the contents of the disk write cache are lost. This means that file data and file system metadata that the application or the file system believed to have been written to disk might not have made it to disk. It is also possible for the data and metadata to have been written to disk in a partial or out-of-order manner, leading to inconsistencies which can cause problems for the file system.

The "write barrier" (or simply "barrier") enforces orderly flushing of these caches to disk. Essentially, the barrier ensures that all requests issued at levels above the barrier must be satisfied before continuing. Write barriers guard against data loss and inconsistency of file system metadata

The barrier affects performance because it requires the write to be complete before the write call continues.

In the default mode, the barrier ensures the data and metadata are written to the File System cache before the write call returns. In O_SYNC mode, the data and metadata must be written to the HDD cache before the write call returns. The diagram below shows this difference:



Typical default for an asynchronous (cached) write

O_SYNC (flushed) write used for critical writes for transaction-controlled files

Asynchronous (cached) vs. synchronous (flushed) writes. Notice that for O_SYNC writes, the call does not return until the data has been written to the HDD cache.

The bug in the Linux kernel mentioned earlier was preventing O_SYNC mode from taking effect, which meant that data was not written to disk before the call returned. This resulted in unexpectedly high performance at the expense of data recoverability in the case of a power loss.

> It should be safe to disable write barriers (using `barrier=0` in */etc/fstab* or using the *-o nobarrier* option for **mount**) in either of the following situations:
> - if disk write cache is disabled; or
> - if disk write cache is enabled and the disk has a *properly configured* battery backup or uninterruptable power supply (UPS).

2.2 Commit Delay and Disk Write Cache

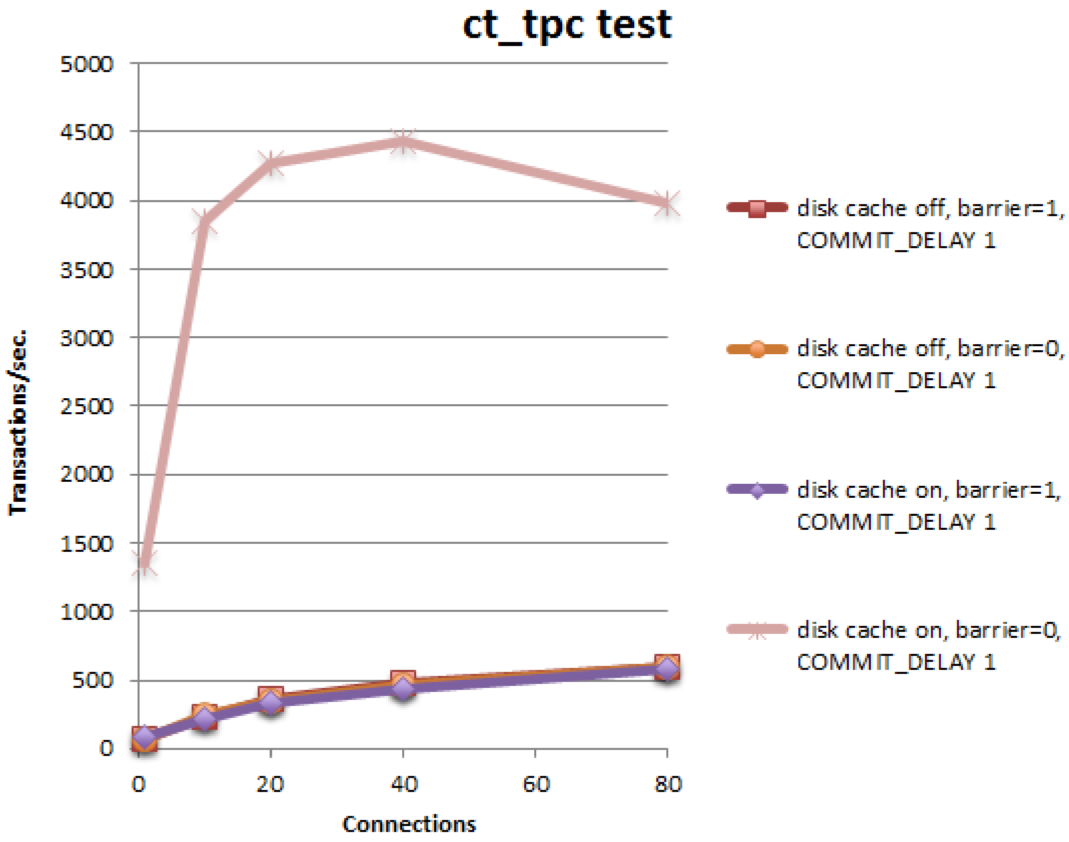
The effects of commit delay and disk write cache on FairCom DB Server performance were tested. All tests used the following options:

```
COMPATIBILITY LOG_WRITETHRU
COMPATIBILITY TINDEX_WRITETHRU
```

The test results are as follows (tps = transactions per second; larger numbers are better):

	ct_tpc test			**With fcntl() bug fix**				
	ext4 file system, data=ordered			**Avg. tps for 20 sec. with # clients:**				
	Disk Write Cache	**Barrier**	**COMMIT _DELAY**	**1**	**10**	**20**	**40**	**80**
	off	1	1	76	236	358	483	600
	off	0	1	76	248	349	463	592
	on	1	1	79	220	332	434	586
	on	0*	1	**1358**	**3848**	**4270**	**4435**	**3980**

* Battery backup or uninterruptable power supply (UPS) required for data integrity.



The results shown above highlight the significant cases from a battery of tests run at different settings.

The following conclusions were drawn based on these tests:

- For all combinations of write cache and barrier settings, using a small commit delay rather than no commit delay improves the performance.
- When disk write cache is enabled (the default for many systems), large commit delays can slow down performance compared to no or small commit delay.

Some options, such as putting an uninterruptable power supply (UPS) on a system, improve recoverability of data without impacting performance.

If you rely on a UPS, be sure to configure it to cleanly bring the system down before the battery is exhausted! Note that the FairCom DB Server will generally come down cleanly if it receives a shutdown signal from the operating system. However, we strongly recommend testing this operation on your system!

The best performance can be achieved by shutting off the barrier, which requires a battery backup or UPS to ensure data integrity.

3. Settings for Best Performance

With the patch installed, the highest transaction rates are seen with the following configuration:

- Disk write cache enabled
- File system mounted with `barrier=0`
- `COMMIT_DELAY 1`

Best Practices

The best configuration for your system depends on many factors. There is no substitute for performing tests to determine the best settings for your environment.

- Use the FairCom DB Load Test Utility to get a general overview of performance on your system. This test program is supplied with the FairCom DB Professional Developer's Kit.
- Use your own application and sample data to experiment with different settings.

Checklist

In evaluating your configuration, consider the questions in this checklist:

1. Which file system are you using and what cashing options does it provide?
2. Which layers of cache do you have on your system (file system, disk controller, etc.)?
3. Do you have an uninterruptable power supply available?
   - Have you properly configured the UPS to bring the system down cleanly before the battery is exhausted?
   - Have you tested to be sure the FairCom DB Server will be brought down in a safe manner by your UPS-invoked shutdown?
4. Should disk write cache be enabled?
5. What is the best setting for the barrier option?
6. What is the best setting for commit delay?

4. Document History

10/01/2015	Corrected the build date of the change from FairCom DB V10.3.1.20772 (Build 140216) to V10.3.1.21834 (Build 140307).
7/1/2014	Updated information on the first and second pages about affected builds and the fact that FairCom DB releases prior to this change are vulnerable to an issue with the Linux kernel.
6/20/2014	Document first published.

Last published Thursday, April 15, 2021.

Copyright Notice

Trademarks

Open Source Components

Like most software development companies, FairCom uses third-party components to provide some functionality within our technology. Often those third-party components are selected because they are a standard in the industry, they offer specific functionality that is easier to license than to develop and maintain in the long run, or they provide a proven and inexpensive solution to a particular business need. Examples of third-party software FairCom uses are the OpenSSL toolkit that provides Transport Layer Security (TLS) for secure communications and the ICU Unicode libraries to provide wide character support (think international characters and emojis).

Some of these third-party components are the subject to commercial licenses and others are subject to open source licenses. For open source solutions that we incorporate into our technology, we include the package name and associated license in a notice.txt file found in the same directory as the server.

The notice.txt file should always stay in the same directory as the server. This is particularly important in instances where your company has redistribution rights, such as an ISV who duplicates server binaries and (re)distributes those to an eventual end-user at a third-party company. Ensuring that the notice.txt file "travels with" the server binary is important to maintain third-party and FairCom license compliance.

4/15/2021

5. Index