

# SILENT DATA CORRUPTION AND PARITY POLLUTION PREVENTION THROUGH ITRENEW STORAGE ARRAYS

## *Abstract*

Recent large academic studies have identified the surprising frequency of silent read failures that are not identified or resolved in enterprise hard disk arrays despite the typical integrity functions. Such errors result in corrupt data being provided by the disk array to the application without any notification or warning, potentially leading to erroneous operations and results. This “silent data corruption” includes misdirected writes, partial writes, and data path corruption with evidence pointing to the errors being uncaught in 1 in 90 drives. By using an extended data integrity feature, silent data corruption can be identified and handled so corrupt data is not sent to the application. Furthermore, data that otherwise would be lost can be recovered.



## Introduction

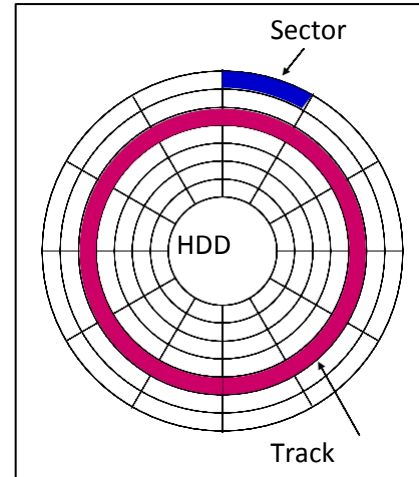
Data integrity is the core of what enterprise systems rely upon. When applications and other systems receive data from their storage system that is malformed or corrupted in some way, it leads to wrong results or downed systems. Even though storage systems contain many layers of data integrity checking and correction, users still encounter corrupt data and its detrimental effects.

## Background: Storage system architecture

### **Sectors and ECC**

The smallest externally addressable unit on a hard disk drive (HDD) is a sector, which typically contains 512 bytes of data. When data is written to a hard drive, it is written in a data “block”, which is just a set of contiguous sectors.

Hard drives contain error detecting and correcting mechanisms to recover from mis-read or mis-written data. For each sector, extra space is set aside to store an error correcting code (ECC) that can be used to detect and correct mis-read or corrupted data. This extra space accounts for much of the difference between an unformatted hard drive’s capacity and the actual usable capacity. But, as we will see later, the hard drive sometimes does not detect that data has been mis-written.



**Figure 1: Hard Disk Drive Structures**

Disk drives commonly experience errors that are recoverable via the ECC. However, various factors can lead to degeneration of data integrity on the hard drive, making it harder to recover from such errors. At some point, data becomes unreadable leading to “read failures,” which can cause systems to crash, hang, or become unstable.

### **Disk Arrays and RAID**

For greater reliability, people use RAID systems in which read failures cause the array controller to mark the effected disk drive as “failed” but maintain data availability using other drives in the RAID group. RAID technology uses multiple disks to achieve greater performance and data availability than afforded by individual disks alone. By spreading data across multiple disks and storing a parity calculation of data, RAID can reconstruct what would, in the event of a single disk failure, have been lost data. It is important to note the reconstruction period of a failed disk can take hours to complete, and performance in the interim is negatively affected.

RAID can also catch and address errors flagged by hard drives. However, there is a set of disk errors not caught by the hard drive and thus not recoverable by RAID technology, and this type of disk error can lead to incorrect data being returned to the server.

## Silent data corruption

### Introduction

There are certain types of storage errors that go completely unreported and undetected in other storage systems which result in corrupt data being provided to applications with no warning, logging, error messages or notification of any kind. Though the problem is frequently identified as a silent read failure, the root cause can be that the write failed, thus we refer to this class of errors as “silent data corruption.” These errors are difficult to detect and diagnose, yet what’s worse is they are actually fairly common in systems without an extended data integrity feature.

### Misdirected write

In some instances when writing to a hard disk, data that is supposed to write to one location actually ends up being written in another location. Through some fault, the disk doesn’t recognize this action is in error and will return a success code. As a result, the “misdirected write” is not detected by the RAID system because it takes action only when the hard disk signals an error.

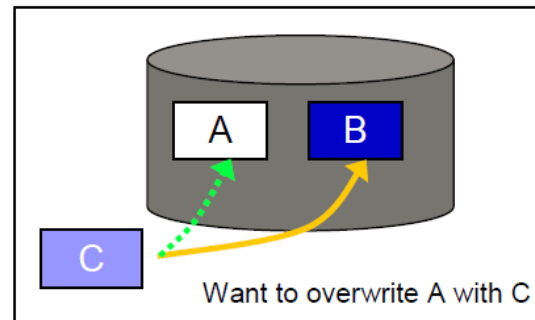


Figure 2: Misdirected Write

Thus, not only has an undetected error occurred, but there has been data loss as well. In Figure 2, data block C was supposed to overwrite data block A but instead overwrote data block B. Thus data block B is lost and data block A still contains the wrong data!

As a result, data has been written to the wrong location; one area has old, wrong data; another area has lost data, and this error will not have been detected by the RAID system or the HDD itself. Accesses to retrieve B or C will result in incorrect data being returned without any warning.

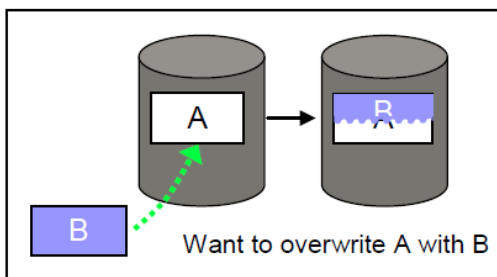


Figure 3: Torn write

### Torn write

In other instances, only some of the sectors that are supposed to be written together end up on the disk. This is called a “torn write” and results in a block of data that contains partially the original data and partially the new data. Some of the new data has been lost, and some reads would return the old data. Again, the hard disk is not aware of this error and returns a success code, so it goes undetected by RAID.

Accesses to retrieve B would return partly incorrect data, which is an untenable situation.

### ***Data path corruption***

The data path from the controller to the hard drive consists of several components: the controller itself, the initiator adapter, the cable, the target adapter, the disk enclosure, and the hard drive. Each of these components is comprised of smaller components and modules. Though data is sometimes protected with error detecting (such as Cyclic Redundancy Check) or correcting (ECC) code while it is being transported from component to component, the data is often exposed to corruption while still within the hardware modules and chips. For example, the RAID module could change data while calculating the parity or some bits may be erroneously flipped while stored in the disk controller cache.

There are thousands (if not hundreds of thousands) of lines of low-level firmware code running on most pieces of hardware – when communicating data, these are collectively called the “protocol stack.” And as much as we want it to be, that code is not bug-free. So a logic error in the firmware can change data in a way that error checking is circumvented when in fact it is not a proper change. Because these can be logical errors rather than a corruption caused by a malfunction, the likelihood that these errors would be detected elsewhere is reduced.

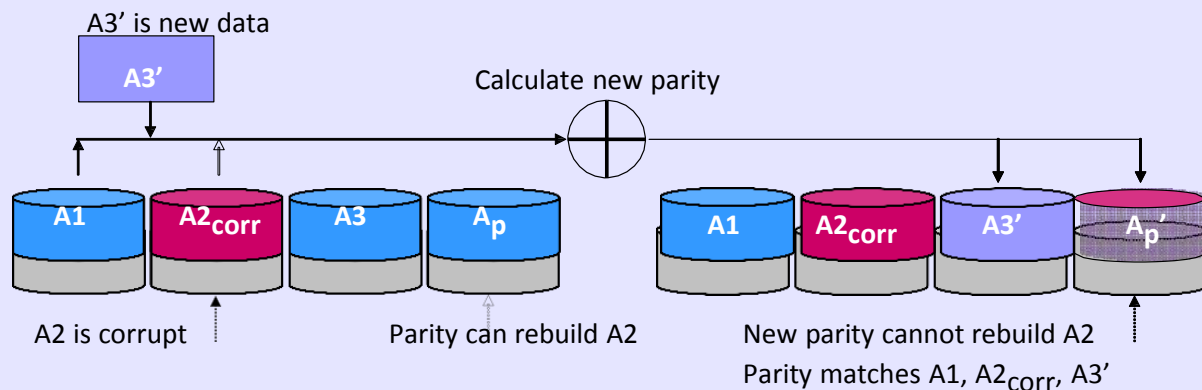
With the combination of firmware code bugs and potential hardware malfunction, there is a possibility during the round trip from controller to disk and back, data can be corrupted without the application’s knowledge.

### ***Parity pollution***

Once a silent corruption has occurred, the error can be compounded to a point where the original data can no longer be retrieved or detected even in a RAID system.

When new data is written in a system that uses a RAID level with parity, a new parity is calculated before the data is written to disk. The RAID system reads the data across the stripe and combines it with the new data to calculate the new parity. In a system experiencing silent data corruption, this situation leads to unrecoverable and undetectable data loss. When corrupt data is used to calculate the new parity, the RAID system can no longer use the parity to restore the non-corrupt data, and this scenario is called “parity pollution.”

### Parity Pollution: An Example



A data stripe on a RAID group (bottom left) originally contained data A1, A2, and A3, and A<sub>p</sub> is the parity calculation for A1, A2, and A3. Subsequently, there was silent data corruption in A2 so that disk now holds A2<sub>corr</sub>. A<sub>p</sub> does not change and could be used to rebuild the original A2 if the corruption were detected.

Some new data comes in to overwrite A3 – it is called A3'. The RAID system reads A1 and A2<sub>corr</sub> in order to calculate the new parity before A3' is written to disk. However, the parity of A1 and A2<sub>corr</sub> is usually not checked at this time, so the corruption goes undetected. The RAID system calculates the new parity, called A<sub>p</sub>' and writes it to disk along with A3'. Because the new parity was calculated with A2<sub>corr</sub>, it is no longer possible to rebuild the original A2. Furthermore, because the new parity matches all of the data on disk, the original corruption is undetectable.

### Silent corruption frequency

A recent academic study [1] of 1.5 million HDDs in the NetApp database over a 32 month period found that 8.5% of SATA disks develop silent corruption. Some disk arrays run a background process to verify that the data and RAID parity match, a process which can catch these kinds of errors. However, the study also found that 13% of the errors are missed by the background verification process.

When you put those statistics together, you find on average that, 1 in 90 drives will experience silent data corruption not caught by the background verification process. So when those data blocks are read, the data returned to the application would be corrupt, but nobody would know. For a RAID-5 (4+P) configuration at 930GB usable per 1TB drive, that calculates to an undetected error for every 67TB of data, or 15 errors for every petabyte of data. If a system were constantly reading all that data at 200MB/sec, it would encounter an error in less than 100 hours.

Another very large academic study [2] looked at failure characteristics for entire storage systems, not just the disks. In the 39,000 storage systems analyzed, the protocol stack (firmware) accounted for between 5% and 10% of storage failures. These are the kinds of failures brought on by faulty code in the firmware.

Thus firmware is not error-free and as previously noted, is capable of introducing silent data corruption. It is clear from these studies that introduction of data corruption into the data path is an all-too real scenario.

## **Solutions to the Silent Data Corruption Problem**

### ***T10-DIF and SCSI-based Protection Information***

Typical Enterprise-class SCSI-based drives (i.e., Fibre Channel and SAS) in disk arrays use a 512 byte sector size, thus for every 32KB block, there are 65 sectors.

The INCITS T10 Technical Committee on SCSI Storage Interfaces is developing updates to the SCSI Primary Commands (SPC-4) and SCSI-3 Block Commands (SBC-3) standards. In these updates, they are introducing 8 additional bytes to each sector for checksum and other uses. This additional data is commonly called the Data Integrity Field (DIF), and the DIF can be used to detect silent data corruption.

The T10-DIF standard applies to SAS hard drives and will provide end-to-end protection against data corruption.

ITRENEW handles silent data corruption by only using storage arrays that support the T10-DIF implementation of 520-byte sectors for enterprise grade applications.

### ***Real World Experience***

Many storage administrators can tell you they have witnessed the presence of undetected data corruption, though there is often no proof of the problem. Proof is difficult to gather and IT prioritizes fixing down systems over root-cause-analysis. These factors prevent many organizations from spending further resources to investigate the problem.

For scientific work, data integrity must be at its highest level so that performing the same analysis on the same data leads to the same results. When the same results are not achieved, however, it makes it easier to identify data corruption as the culprit. For example, CERN, Fermilab, and Desy all experienced data corruption cases, with the investigation at the CERN Computer Center pointing the finger at various causes, including silent data corruption [3].

Based on such investigations, ITRENEW will only supply storage systems using T10-DIF Protection Information for the High Performance Computing (HPC), government, and research markets where data integrity is of demonstrable importance and is readily apparent.

### **Summary and Conclusions**

The evidence presented in academic studies and customer experience validates the existence and unfortunate regularity of errors not caught by typical RAID systems. Silent data corruption is an unrecognized threat that may affect as many as 1 in 90 drives. The ITRENEW Storage Arrays address silent data corruption, while also preventing parity pollution.

## References and Further Reading

[1] Bairavasundaram et al. "An Analysis of Latent Sector Errors in Disk Drives." *Proceedings of the International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS'07)*. June 2007.

Bairavasundaram et al. "An Analysis of Data Corruption in the Storage Stack." *Proceedings of the 6th USENIX conference on File and Storage Technologies (FAST'08)*. February 2008

[2] Jiang et al. "Are Disks the Dominant Contributor for Storage Failures?" *Proceedings of the 6th USENIX conference on File and Storage Technologies (FAST'08)*. February 2008

Krioukov et al. "Parity Lost and Parity Regained." *Proceedings of the 6th USENIX conference on File and Storage Technologies (FAST'08)*. February 2008

[3] Panzer-Steindel. "Data Integrity." *Internal CERN/IT study*. 8 April 2007  
(<http://indico.cern.ch/getFile.py/access?contribId=3&sessionId=0&resId=1&materialId=paper&confId=13797>)

Note: Figure 2 and Figure 3 are derived from concepts in the presentation given on "Parity Lost and Parity Regained."