A Review of Disc Scrubbing and Intra Disc Redundancy Techniques for Reducing Data Loss in Disc File Systems

Genti Daci
Polytechnic University of Tirana
Faculty of Information Technology
Sheshi Nene Tereza 4, Tirana, Albania
gdaci@icc-al.org

Aisa Bezhani
Polytechnic University of Tirana
Faculty of Information Technology
Sheshi Nene Tereza 4, Tirana, Albania
aisa_bezhani@live.com

ABSTRACT
Because of high demand that applications and new technologies have today for data storage capacity, more disk drives are needed, resulting in increased probability to inaccessible sectors, referred as Latent Sector Errors (LSE). Aiming to reduce data loss by LSE, two main techniques are extensively studied lately: Disk Scrubbing, which performs reading operations during idle periods on systems to search for errors and Intra Disk Redundancy which is based on redundancy codes. This paper reviews and discusses the problems of LSE and the main causes that lead to LSE, its properties and their correlation on nearline and enterprise disks. Focusing on reducing LSE with regards to security, processing overhead and disk space, we analyze and compare the latest techniques: Disc Scrubbing and Intra Disk Redundancy aiming to highlight the issues and challenges according to different statistical approaches. Furthermore, based on previous evaluation results, we discuss and introduce the benefits on using both schemes simultaneously: combining different IDR coding schemes with Accelerated Scrubbing and Staggered Scrubbing in particular regions of disc drives that store crucial data during idle periods. Finally, we discuss and evaluate from an extended statistical analysis the best ways on how reduce data loss with a minimum impact on system performance.

Categories and Subject Descriptors
C.4 [**Performance of Systems**]: *Reliability, availability and serviceability*; B.3.2 [**Memory Structures**]: Design Styles—*Mass storage*

General Terms
Performance, Reliability

Keywords
Latent sector errors, disk scrubbing, intra disk redundancy, staggered scrubbing, accelerated scrubbing, data storage capacity, disk drives

1. INTRODUCTION
Except complete disc failures[1,18] there are other factors that leave the HDD functioning but corrupt the data [7,24,26]. The one that we will focus on are latent sector errors. Although, many solutions are proposed for total failure cases, the latter has higher chances of occurrence [15,20]. Both affect enterprise discs which are mostly used by business –critical applications, and also nearline discs which are used for archival purpose. There are several reasons on why we focus on this subject:

The usage of cheaper disk drives, which at the same time are less reliable (like choosing SATA disk drive components over SCSI or FC) and at the same time, the need for more storage and faster performance (using smaller disks) .

Also, when a disk fails in a system using RAID5, during the reconstruction process, good sectors are read, so that all the undamaged data could be written in a free disc. During this process if the reading operation finds a damaged sector, the data will be lost [22]. The same thing happens when using RAID6, with the difference that it is used for every two disk fails.

The main problem with LSE is the fact that until a particular sector is used, we can never know that it is corrupted or damaged, nor we know the exact time the problem occurred. Along with that, understanding error distribution, helps creating new techniques in preventing data loss. Intra Disk Redundancy helps in reducing unrecoverable errors. It aims to add an extra redundancy level to disk drives, along with the existing Inter Disk Redundancy, or RAID. Disc scrubbing [10,16] scans periodically to find media errors and so to prevent data loss, having a positive impact on MTTDL.

Disc scrubbing [2] and Intra Disk Redundancy [4], were first evaluated based on the assumption that LSE is similar to the Poisson distribution. The results of this assumption demonstrated that Disc scrubbing depends on workload and it can never offer the reliability that IPC offers. IPC is a technique of Intra disc redundancy which does not affect noticeably workload. This scheme was thought to give the same reliability on data as an operating system without unrecoverable sectors would give. So, IPC would have been a way better security scheme than Disk scrubbing for a system with high workload. After an expanded analyze of data in [5], a better statistical approach was adopted, the Pareto distribution. As a result, a reconsideration of the older Intra-Disk Redundancy techniques is done on [27] like: simple parity check, simple parity check, interleaved parity, maximum distance separable codes and new techniques were created: hybrid SPC and MDC, and Column Diagonal Parity. The same thing applies to Disk scrubbing: Localized, Accelerated, Staggered, and Accelerated Staggered Scrubbing. As a result of new strategies, we reconsider the simultaneous usage of methods, to detect errors and then to correct data, preventing it from loss.

BCI'12, September 16–20, 2012, Novi Sad, Serbia.

Local Proceedings also appeared in ISBN 978-86-7031-200-5, Faculty of Sciences, University of Novi Sad.

At last, understanding the distribution of the LSE helps in deciding where to put the important information of a file system, like metadata [21], which as known is essential in data loss when the sector in which they are written is damaged. These specific parts of disk drives, require methods like Accelerated Scrubbing and Staggered Scrubbing to detect as soon as possible, errors that can result fatal. Even so, we have to find a proper scrubbing technique and frequency that does not affect our system. We apply different coding schemes on Intra Disk Redundancy arguing on security level, over disk space that the parity sectors occupy. We manage the utilization of idle times with the Busy Bee dynamic algorithm, in order to balance the need for security and performance at the same time. The paper shall include an introduction on the current research in the papers field, original solutions, experimental results analysis, conclusions and references.

2. BACKGROUND

Our review is based on collected data of [5] which analyses causes that lead to LSE and its characteristics. This data was collected from 1.53 million disks during a period of 32 months. The software stack consisted of three layers: The Wafl file system, Raid and storage layer. Logs of important events were registered from a mechanism located in storage data. Latent sector errors are known to be there, only when we try to access a certain sector. Based on the Net Apps's system which scrubbing frequency is 2 weeks, we can approximate the occurrence of the error around the same amount of time. More than 30 models were studied, each of them is found to have its own characteristics but we focus on the general distinction between enterprise and nearline disks. The results of the logs in the systems were latter analyzed according to two main different approaches, and it is based on those two that we argue our decisions on a better system.

3. LATENT SECTOR ERRORS

Impact on Nearline and enterprise disks: Expectedly, experiments [6,27] showed that, nearline discs are more likely to develop latent sector errors, compared to enterprise disks [5]. However, both are more likely to develop other errors, after the first one has occurred. Of all the cases in which this problem was encountered, 60% of them was identified by disk scrubbing. The information retrieved by the logs showed the moment when the problem wass evidenced. It is not known the exact moment when this error was created.

Error bursts: By error burst we mean a series of contiguous errors located in a logical block space. We should be able to answer questions like: If two error burst occur, will they be far enough from each other so that they happen to be in different parity sectors (statistically speaking) ? If an error occurs, will the system have enough time to recover itself before the next error happens? An important factor that helped in answering these questions is error burst distribution. 90%- 98% of error bursts was created by a single error. In most models, more than 2.5% caused 2 continuous errors, the percentage dropped even more for cases with more than two errors. Having the experimental results, five different approaches were studied, and the one who fitted most the results, was the Pareto distribution. Another factor considered was the distance of error bursts. According to the results, 20 – 60 % of errors, were prone to have another error in a distance of less than 10 sectors. Specific regions of the discs are more prone to errors. Depending on the disc type, 20-50 % of all errors happened on the first 10% of the disk, on others, this percentage was focused at the last sectors. In the same way, experiments proved that after the first error was detected, chances for the second error to happen dropped exponentially (1% after 10 weeks, 0.1% after 30 weeks). Another interesting numerical factor was the fact that 58-85 % of errors in disc happened during a two week period.

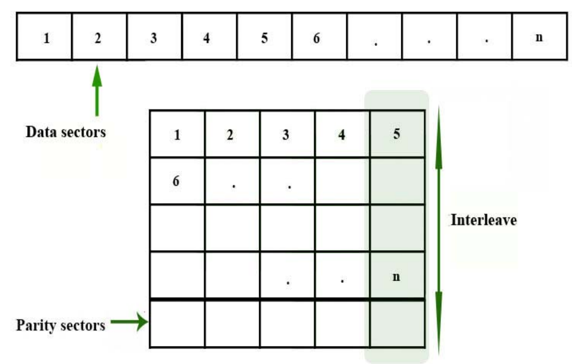

Figure 1: IPC scheme.

Spatial and local correlation: Until late those two factors were studied apart. For example, If we wanted to determine that an inter disk redundancy scheme is appropriate for a certain system, despite of the distance that two errors occur, will the system have the necessary time to recover from the first error before the second one occurs? Over 90% of burst errors are discovered in two weeks, and over 95% are detected in a one month distance. This means that errors are caused by a common factor, that's why they happen at the same time.

4. INTRA DISK REDUNDANCY

Logic structure: Intra disk redundancy has a simple logic: every stripe is divided into segments. There are data bits and parity bits inside each segment. Parity bits are created using different parity codes. A segment is compound of ℓ sectors, of which n sectors are used for data bits, and the remaining (ℓ-n) sectors for parity bits. The schemes proposed change the security level according to disc space and overhead penalties.

Single Parity Check (SPC) uses k consecutive sectors to store data and one sector for parity. SPC is a simple scheme, therefore it implies low overhead .This scheme is able to correct only one error per parity group. To be able to correct more than one error, inter disk redundancy is needed, which means Raid needs to be used. IPC uses an internal scheme which is based on simple XOR[22] operations. This scheme guarantees security in Raid systems, without a noticeable growth in overhead. The way IPC works is: N sequential data sectors are put in a certain way to create a matrix. Parity sectors in columns are created using XOR. When data is updated, the parity sectors are updated too. A column with data along with its parity sector is called interleave. An IPC scheme which has a number of m interleaves, is able to correct one mistake per interleave. According to Poisson distribution results, IPC is an optimal choice for typical frequency and workload. It is stated that it reaches the same reliability state comparable to a system without inaccessible sectors. There are also down sides to this, such as: to reach this state of security, we should increase the dick size by 6% in order to store the same amount of data [11,12,17].

Maximum distance separable codes (MDS) is compound of k data sectors followed by m parity sectors. This scheme can tolerate the loss of m sectors per segment. It can recover the data for m errors, where m is the number of sector that the parity bits occupy. Again this is an adaptation of codes used in raid systems [19]. Obviously there is higher reliability, because parity sectors are calculated by RS codes, but simultaneously it gives higher overhead and parity bits take a considerable part of the hard disk space not usable for actual data. Lately, the overhead that it implies comparing to IPC is discussable [29].

Most disc errors happen at 5 – 15% of the disk space, that is why hybrid methods were created. Hybrid SPC and MDS uses MDS [27] code for parts that are more prone to errors, and a simple SPC 8+1 code for the rest. This helps in finding the balance between security and overhead.

CDP can detect more errors if compared to SPC and IPC, and at the same time it implies less overhead if compared to MDS. This code is based on RDP [25] which is used to recover from double failures on systems that use RAID. Originally, RDP uses p+1 disks, from which p-1 store data and the other 2 left : one is used for diagonal parity and the other for row parity. This code, besides its application to RAID, works with inter disk redundancy too, where the number of data sectors is $k = (p -1)^2$, and the number of parity sectors is $m = 2(p - 1)$. CDP can tolerate 2 error bursts with length p-1, and reconstruct data of a sector /diagonal if the sector /diagonal is found in a column with less than 2 errors. Even in this case we have an inverse correlation of efficient space in disk (k / (k + m)) against I/O overhead and the security that it offers. Expectedly, the most efficient codes have lower reliability. Several experiments, changed the sector size to find the maximum efficiency. If the sector sizes were enlarged, higher overhead was encountered, because of the distance between data and parity sectors. As a result, if two data sectors are far from each other, so will the parity sectors be. On the other hand, accessing the desired sectors will take more time, leading to higher overhead. Experiments showed that, if a change in the MDS scheme was applied from 8+1 to 16 +2, the security level raised by 50%. To achieve this level of security, IPC used schemes of 56+7 and 64+8. When bigger segments were used MDS and CDP would outreach the performance of IPC with a few orders.

5. DISK SCRUBBING

Disk Scrubbing is applied in order to detect silent errors before the damage is done [20, 23]. *Schemes:* Sequential scrubbing is a process which scans and reads disc sectors one after another. The moment it scans the last sectors, the process repeats. Scrubbing period is determined by the capacity and speed by which the disc is checked and the time required to scan the sectors form the first to the last.

Random scrubbing is similar to sequential scrubbing, with the only difference that accessing is done in a random order. The problem in this case is that several sectors may be scanned many times, while others may not be checked at all. Disc scrubbing is affected by workload and the frequency of the reading on disc to identify the inaccessible sectors. Besides its impact on performance, it does not need free disk space. Intelligent staggered strategies were later developed.

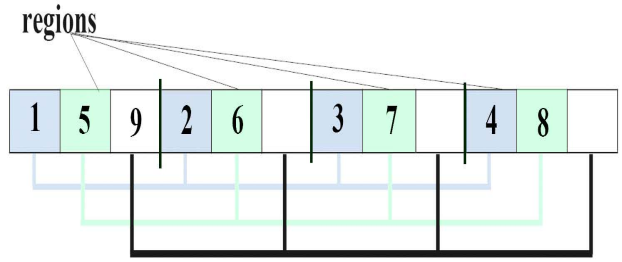

Figure 2: Staggered scrubbing.

Staggered scrubbing, divides the disc in m regions. Each division is compound of r segments. It reads the first segment of every single region according to Logical Block Address (LBA) as shown in Figure 1. After that it starts reading the second segments of the regions, and so on. This technique is effective because if a region has LSE, there is a big chance that some consecutive sectors are damaged. For example, an extreme case would be a region whose sectors are all corrupted by LSE. Staggered scrubbing would be able to detect the error faster than sequential scrubbing. Staggered scrubbing has its own downsides, like: more head movement, but it can be solved when several parameters are chosen. One of those, is request size. The bigger the request size, the better the reading performance. After a few tests done on variable request sizes, on a 16 GB, 7200RPM Hitachi disc drive, the optimal size is 16KB. For larger sizes, the optimization is not noticeable.

Localized scrubbing scans disks for LSE and when the first error is found, being aware of spatial and temporal locality, this scrubbing scheme implies that other errors will likely be near the one it just detected. So, it scans r sectors continuously with an accelerated rate.

Accelerated scrubbing: When an error is detected, the rest of the disk sectors are scanned with an accelerated rate, while Accelerated staggered scrubbing combines both techniques, as the name implies. It detects the erroneous sectors, and then scans the entire disk with accelerated rate.

Comparison: Local scrubbing has a comparative performance to that of standard disc scrubbing. Traditional scrubbing does not accelerate, but since we are talking about contiguous sectors, it will soon detect them. Accelerated scrubbing also doesn't perform well for the same obvious reasons. Both schemes are slow on detecting the first error, and then accelerate. Thus, a different approach is taken in the Staggered scrubbing. It localizes which parts are affected by LSE. It outperforms standard scrubbing in both forms, local and accelerated. The proper segment size is found: big enough to minimize overhead, and small enough, so it doesn't resemble to a standard scrubbing. For a region size of 128 MB the scrubbing effectiveness was similar, for segment sizes from 1 KB to 32MB. If the region size is shrunk to 64MB, accelerated scrubbing effectiveness drops by 50% if compared to that of the standard scrubbing.

6. OUR APPROACH

The idea here, is to use simultaneously Disc Scrubbing to detect media errors on disc and Intra Disk Redundancy to check if the data is corrupted. The risk of data loss can be solved by using mirroring, but this method is time consuming, and applies too much overhead. Having studied the characteristics and the causes that lead to data loss, we can try to predict which of the intradisk redundancy codes along with which disc scrubbing scheme can be a better match. As we know, the data in hard disk drives is separated in to regular data (which is data that we store) and metadata (which is data about the data and it is essential in locating other files that are stored in the disk). Higher reliability is crucial in disks that store critical data. To do so, we need to decrease the MTTDL (in systems using RAID) or MTTF[9,3] (systems that don't use RAID). In order to keep crucial data safe we propose the application of intelligent disc scrubbing strategies such as: staggered scrubbing or accelerated scrubbing to prevent from losing data. Check-summing is also required for checking and reconstructing. We need to find the effective parity coding that gives reliability and does not occupy much disk space.

Knowing also that disc drives have certain parts that are more prone to burst errors, for example 20-50 % of all errors happen on the first 10% of the disc, or at the last 10% of the disk depending on the type. Obviously we avoid storing metadata in these critical areas.

Our aim is to maximize MTTDL time. Being aware of the fact that chances of a LSE occurrence are focused on the first 10% , we scan in a sequential way. The remaining sectors will be scanned with staggered scrubbing so we can be able to detect the error faster. As soon as the first error is detected, we trigger the accelerated staggered scrubbing. By doing this, we can evidence in the fastest way possible all the sectors affected by LSE. We chose accelerated staggered scrubbing based on the fact that there is a high possibility of another error which is more likely to happen after the first error has occurred. Applying accelerated scrubbing will be not efficient because the sectors we are considering are close in space, so accelerated scrubbing would result in a slow performance, just like the one of sequential scrubbing. For the other parts of the disc drive we consider using staggered scrubbing, because we don't want the background activities interfere and create considerable delays in the system performance. Once the error is found, we will need to correct it.

Inter Disc Redundancy parity sectors are obtained using simple operations on data bits. It is depending on these operations that we obtain the security and the overhead. Single Parity Check based on XOR operation is simple, but it is not appropriate since bust errors have spatial locality. This means that if two errors occur in the same logical space, our data can not be reconstructed, and the scrubbing would be pointless. Next approach would be MDS, but using Reed Solomon codes while we have also scrubbing would imply high overhead, predictably not appropriate to use, although it gives the best combination according to reliability. In this case we use Column Diagonal Parity (CDP), which is the adapted code of Raid for double failures. It tolerates a considerable number of failure patterns, while not implying a high overhead. Another reason that leads us to use a simple parity code, is the fact that errors will be detected soon, so there is a lower chance for them to spread and affect multiple bits in a single data-parity sectors combination.

Disk scrubbing in terms of latency can take the same time as a read operation. It is infinite because, once the disc is scrubbed totally, the process restarts. On the other hand, intra disk redundancy needs to update the parity sector(s) every time a change is made. This is reduced to a read process, a calculation and a write of the parity group. Seemingly, we expect a high overhead, if we use both techniques. This would not be appropriate if applied in real systems.

To avoid the overhead we use both, disc scrubbing and intra disk redundancy as background activities, so it can impact as less as possible the normal operation of the system [13, 14]. It is proven that the cooperation of the two gives a high reliability, more than the linear combination of disc scrubbing or intradisk redundancy when they operate alone in a system. Out of all the ways to manage idle times we consider Busy Bee [8]. It is a scheduling algorithm that adapts dynamically, according on the current workload. It has a crucial feature: even if we are operating on a high workload system, this policy doesn't leave the background operation starve. For our system, this means, that if a foreground short job is scheduled, and we evidence a LSE, we can leave the foreground job waiting, and schedule the accelerated staggered scrubbing for a limited time, until we are sure that no other LSE has affected a nearby sector. If the foreground job takes many clock cycles to complete, we can suspend the scrubbing because we do not want to add further latency on the system. As soon as the long job finishes, we restart the scrubbing. However, we must consider, factors such as the disk model or the file system that we are working on, because the approach would change according to the specific case.

```
do {
If (high probability of short_foreground_job_coming)
 {
    if (scanner_head < 0.1*n)  // n- total number of sectors.
            Apply_sequential_scanning //standard frequency

 else if (scanner_head >=  0.1*n)  //if out of the first 10%
            Apply_staggered_scrubbing // standard frequency

else if ( LSE detected)
{
        Apply_accelerated_staggered_scrubbing
     Call intradisk_redundacy_operation_on_affected_sector
}
else wait for foreground job  }}
While (idle_time)
```

The above pseudo-code reveals the concept on which this theory is based. The scrubbing operation happens during idle time, and because it is infinite, we continue until a foreground job is present. If the foreground job is statistically thought to be short, the system continues to scrub. In the case it detects an error burst, weather the foreground job is short or long, the system doesn't stop scanning. In this case we prefer to secure our data, not minding the overhead in this extreme case of risking data loss. Otherwise, the scrubbing stops. There are three scanning schemes that we apply, in order to be sure that if a LSE exists, we would be able to catch it as soon as possible. On the other hand, we can not use accelerated scrubbing strategies during the whole time because it would impact on system performance. According to the mentioned reasons, we scan sequentially the first 10 % of the total number of sectors, knowing that it has the biggest probability of developing latent sector errors. After that we chose sequential scrubbing, knowing that in comparison to the other scrubbing strategies (except staggered ones), it is the one to detect the errors first. Last, if we encounter an error, we want to prevent losing data in any cost. This is why the intradisk redundancy is called to perform a reading of the damaged sector, and also a recalculation, so it can locate the error, and fix it. While doing so, we scan at an accelerated rate, to detect any other possible erroneous areas. We need to state that in real life, there is no way for us to predict if the coming job is going to be long or short. In this case, if we are expecting a short job the scrubbing process will start. But what if a long job comes? The scrubbing will stop, giving precedence to the foreground job, if and only if an error is not detected at that moment.

7. CONCLUSIONS

Based on previous analyses our aim was to find the best solution while using disc scrubbing and intradisk redundancy simultaneously. We proposed a combination of disk scrubbing techniques by which we can detect LSE in the fastest way possible. By doing this we can allow the usage of an Intra Disk Redundancy parity code that will not imply much overhead. Opting for a minimal impact on performance we use both strategies as background operations, applying a dynamic algorithm.

Our idea was based on related studies of temporal and spatial distributions of error bursts. By understanding them, we predicted the best approaches among many disc scrubbing schemes and intra disk redundancy codes possible combinations. We found that an aggressive policy would be appropriate for those parts of disks where critical data is stored, or that are more prone to disk errors. Although, it assures reliability, our theory is yet to be proven in practical terms of overhead.

8. REFERENCES

[1] D. Patterson, G. Gibson, and R. Katz. A case for redundant arrays of inexpensive disks (RAID). In Proc. of SIGMOD, 1988

[2] Alina Oprea , Ari Juels, A clean-slate look at disk scrubbing, Proceedings of the 8th USENIX conference on File and storage technologies, p.5-5, February 23-26, 2010, San Jose, California

[3] Mary Baker , Mehul Shah , David S. H. Rosenthal , Mema Roussopoulos , Petros Maniatis , TJ Giuli , Prashanth Bungale, A fresh look at the reliability of long-term digital storage, Proceedings of the ACM SIGOPS/EuroSys European Conference on Computer Systems 2006, April 18-21, 2006, Leuven, Belgium [doi>10.1145/1217935.1217957]

[4] Ajay Dholakia , Evangelos Eleftheriou , Xiao-Yu Hu , Ilias Iliadis , Jai Menon , K.K. Rao, A new intra-disk redundancy scheme for high-reliability RAID storage systems in the presence of unrecoverable errors, ACM Transactions on Storage (TOS), v.4 n.1, p.1-42, May 2008 [doi>10.1145/1353452.1353453]

[5] Lakshmi N. Bairavasundaram , Andrea C. Arpaci-Dusseau , Remzi H. Arpaci-Dusseau , Garth R. Goodson , Bianca Schroeder, An analysis of data corruption in the storage stack, ACM Transactions on Storage (TOS), v.4 n.3, p.1-28, November 2008

[6] Lakshmi N. Bairavasundaram , Garth R. Goodson , Shankar Pasupathy , Jiri Schindler, An analysis of latent sector errors in disk drives, Proceedings of the 2007 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, June 12-16, 2007, San Diego, California, USA [doi>10.1145/1254882.1254917]

[7] Weihang Jiang , Chongfeng Hu , Yuanyuan Zhou , Arkady Kanevsky, Are disks the dominant contributor for storage failures?: a comprehensive study of storage subsystem failure characteristics, Proceedings of the 6th USENIX Conference on File and Storage Technologies, p.1-15, February 26-29, 2008, San Jose, California

[8] Feng Yan, Alma Riska, Evgenia Smirni, Busy bee: how to use traffic information for better scheduling of background tasks. ICPE '12 Proceedings of the third joint WOSP/SIPEW international conference on Performance Engineering, p.145-156, 2012, doi>10.1145/2188286.2188308

[9] Bianca Schroeder , Garth A. Gibson, Disk failures in the real world: what does an MTTF of 1,000,000 hours mean to you?, Proceedings of the 5th conference on USENIX Conference on File and Storage Technologies, p.1-1, February 13-16, 2007, San Jose, CA

[10] Thomas J. E. Schwarz , Qin Xin , Ethan L. Miller , Darrell D. E. Long , Andy Hospodor , Spencer Ng, Disk Scrubbing in Large Archival Storage Systems, Proceedings of the The IEEE Computer Society's 12th Annual International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'04), p.409-418, October 04-08, 2004

[11] Ilias Iliadis , Robert Haas , Xiao-Yu Hu , Evangelos Eleftheriou, Disk scrubbing versus intra-disk redundancy for high-reliability raid storage systems, ACM SIGMETRICS Performance Evaluation Review, v.36 n.1, June 2008

[12] Ilias Iliadis , Robert Haas , Xiao-Yu Hu , Evangelos Eleftheriou, Disk Scrubbing Versus Intradisk Redundancy for RAID Storage Systems, ACM Transactions on Storage (TOS), v.7 n.2, p.1-42, July 2011

[13] N. Mi, A. Riska, E. Smirni, and E. Riedel. Enhancing data availability in disk drives through background activities. In Proc. of DSN, 2008.

[14] N. Mi, A. Riska, Q. Zhang, E. Smirni, and E. Riedel. “Efficient utilization of idle times” Proceedings of the ACM SIGMETRICS, pp. 371–372, 2007

[15] Rozier, E., Belluomini, W., Deenadhayalan, V., Hafner, J., Rao, K., and Zhou, P., Evaluating the impact of undetected disk errors in raid systems. In Dependable Systems Networks, 2009. DSN '09. IEEE/IFIP International Conference on (29 2009-july 2 2009), pp. 83-92.

[16] Jon G. Elerath , Michael Pecht, Enhanced Reliability Modeling of RAID Storage Systems, Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, p.175-184, June 25-28, 2007 [doi>10.1109/DSN.2007.41]

[17] Evaluation of Applied Intra-disk Redundancy Schemes to Improve Single Disk Reliability Matthias Grawinkel,

Thorsten Schafer, Andre Brinkmann, Jens Hagemeyer, Mario Porrmann (2011) 2011 IEEE 19th Annual International Symposium on Modelling Analysis and Simulation of Computer and Telecommunication Systems p. 297-306

[18] Eduardo Pinheiro , Wolf-Dietrich Weber , Luiz André Barroso, Failure trends in a large disk drive population, Proceedings of the 5th conference on USENIX Conference on File and Storage Technologies, p.2-2, February 13-16, 2007, San Jose, CA.

[19] Alexander Thomasian , Mario Blaum, Higher reliability redundant disk arrays: Organization, operation, and coding, ACM Transactions on Storage (TOS), v.5 n.3, p.1-59, November 2009 [doi>10.1145/1629075.1629076]

[20] Pâris, J.-F. , Improving Disk Array Reliability Through Expedited Scrubbing, 2010 IEEE Fifth International Conference on Networking, Architecture and Storage (NAS), 15-17 July 2010 pp 119 - 125

[21] Vijayan Prabhakaran , Lakshmi N. Bairavasundaram , Nitin Agrawal , Haryadi S. Gunawi , Andrea C. Arpaci-Dusseau , Remzi H. Arpaci-Dusseau, IRON file systems, Proceedings of the twentieth ACM symposium on Operating systems principles, October 23-26, 2005, Brighton, United Kingdom [doi>10.1145/1095810.1095830]

[22] James Lee Hafner , Veera Deenadhayalan , K. K. Rao , John A. Tomlin, Matrix methods for lost data reconstruction in erasure codes, Proceedings of the 4th conference on USENIX Conference on File and Storage Technologies, p.14-14, December 13-16, 2005, San Francisco, CA

[23] Junping Liu, Ke Zhou, Zhikun Wang, Liping Pang, Dan Feng, Modeling the Impact of Disk Scrubbing on Storage System. Journal of Computers, Vol 5, No 11 (2010), 1629-1637, Nov 2010 doi:10.4304/jcp.5.11.1629-1637

[24] Andrew Krioukov , Lakshmi N. Bairavasundaram , Garth R. Goodson , Kiran Srinivasan , Randy Thelen , Andrea C. Arpaci-Dusseau , Remzi H. Arpaci-Dussea, Parity lost and parity regained, Proceedings of the 6th USENIX Conference on File and Storage Technologies, p.1-15, February 26-29, 2008, San Jose, California

[25] Peter Corbett , Bob English , Atul Goel , Tomislav Grcanac , Steven Kleiman , James Leong , Sunitha Sankar, Awarded Best Paper! -- Row-Diagonal Parity for Double Disk Failure Correction, Proceedings of the 3rd USENIX Conference on File and Storage Technologies, March 31-31, 2004, San Francisco, CA

[26] Jon Elerath, Hard Disk Drives: The Good, the Bad and the Ugly!, Queue, v.5 n.6, September/October 2007 [doi>10.1145/1317394.1317403]

[27] Bianca Schroeder , Sotirios Damouras , Phillipa Gill, Understanding latent sector errors and how to protect against them, ACM Transactions on Storage (TOS), v.6 n.3, p.1-23, September 2010

[28] J. L. Hafner , V. Deenadhayalan , W. Belluomini , K. Rao, Undetected disk errors in RAID arrays, IBM Journal of Research and Development, v.52 n.4, p.413-425, July 2008 [doi>10.1147/rd.524.0413]

[29] Plank, J.S., XOR's, lower bounds and MDS codes for storage Information Theory Workshop (ITW), 2011 IEEE, 16-20 Oct. 2011, pp. 503- 507