RAID-CUBE: The Modern Datacenter Case for RAID

Jayanta Basak
Randy H. Katz

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2015-4
http://www.eecs.berkeley.edu/Pubs/TechRpts/2015/EECS-2015-4.html

February 5, 2015

Acknowledgement

Research supported in part by NetApp Corporation and the UC Berkeley AmpLab.

RAID-CUBE: The Modern Datacenter Case for RAID

Jayanta Basak
Advanced Technology Group
NetApp India Pvt. Ltd.
Bangalore, India
basak@netapp.com

Randy H. Katz
Department of Computer Science
University of California
Berkeley, California, USA
randykatz@berkeley.edu

***Abstract*—"Big Data" processing in modern datacenters dramatically increases the data volume moving between applications and storage. A major challenge is achieving acceptable levels of availability and reliability in an environment characterized by huge storage capacities, large numbers of disk drives, and very high interconnection bandwidth (e.g., 100 petabytes and 17000 disk drives at CERN[1]). In this paper, we show that existing RAID mechanisms are insufficient, and that the *mean time to data loss* (MTTDL) falls drastically as the number of disks and data volume increase. We introduce a new high availability storage configuration, which we call RAID-CUBE, and show that it is more resilient to data loss as the datacenter scales in capacity than existing RAID dual parity and triple parity schemes. We also identify the limits to capacity of a datacenter (in terms of the number of disks) to maintain an acceptable MTTDL for different data protection mechanisms. Finally, we briefly introduce an effective mechanism for bit error protection for large sequential IOs in this environment.**

I. INTRODUCTION

The emergence of "big data" imposes new challenges for datacenter storage. As storage capacity and the number of disks increase, data loss becomes not just a possibility but inevitable. The sheer size and scale of data movement between applications and storage in a large-scale datacenter renders latent defects more prevalent. When combined with the inevitable operational failures in the disks, this increasingly results in unrecoverable data loss. Data protection in datacenters has emerged as a critical issue. For example, as far back as 2008, Google reported they suffered a disk failure *every* time they executed a 6-hour long petabyte sort across 48,000 disks[2].

In modern datacenters, the storage hierarchy is complex, and assessing it for data loss is complicated. Performance sensitive applications stage their "hot" data into SSDs. The data then spills to local disks, and then to geo-scale storage. Data loss can happen in various system layers, e.g., there can be data corruption due to device wear out in the SSDs, data loss in the local disks due to latent defects and operational failures, data loss in geo-distributed storage, and finally data corruption in the network. In map-reduce workloads, there can be imminent loss during the enormous amount movement.

When the Google file system [6] was originally designed (and later re-implemented in open source as HDFS), RAID techniques, with their parity calculation overheads, were rejected because storage was getting cheaper and redundancy could be achieved simply by keeping multiple copies of files. The data sizes now managed by datacenters, whether data logs or consumer photos, are simply too large and the need to manage multiple copies too difficult. With effective use of memories hierarchies and in-memory processing, and even given that most files die young [1], datacenter storage workloads are becoming increasingly dominated by large block I/Os. This is highly advantageous for avoiding the "small update" Read-Modify-Write overheads of RAID parity processing. The IO block size is also increasing, making bit errors or latent defects even more prominent [18]. The only effective way to protect against these as well as operational failures is to intelligently use more data redundancy.

RAID redundancy mechanisms include simple RAID 5 parity [14], "dual parity" RAID6 [4], erasure coding [2, 9], network coding [5], copy-set replication [3], and RAID triple parity [8]. These techniques certainly protect data. However, as the size and volume of data grows, their effectiveness diminishes. Moreover, techniques like erasure coding and network coding are difficult to scale to larger number of disks because of the computational burden. Also the bandwidth of reconstruction is high for erasure coding.

---

[1] http://home.web.cern.ch/about/updates/2013/02/cern-data-centre-passes-100-petabytes

[2] http://royal.pingdom.com/2008/12/12/wanted-hard-drive-boys-for-our-new-ginormous-data-center/

In this paper, we focus on how best to provide data protection and recovery in a datacenter with a large number of disks, measured in the tens of thousands. Given enough disks, the probability of data loss becomes sufficiently high that data loss becomes operationally inevitable. For example, let the probability of data loss with some protection mechanism for 10 disks be 0.0001. When we scale 1000 times and deploy 10,000 disks, the same probability becomes $1-(1-0.0001)^{1000}$ = 0.095 (i.e., there is a 9.5% chance of data loss) if the disks fail independently. In the case of correlated failure (i.e., due to the effect of physical conditions [16]), the risk of data loss may be further aggravated. In the literature so far, there has not been any formal analysis on the *mean time to data loss* (MTTDL) for very large number of disks in a datacenter (e.g., 20000 disks). We formally analyze the MTTDL for a given mean-time-to-recovery (MTTR) for very large number of disks in a datacenter with different protection mechanisms. We then introduce RAID-CUBE, a RAID configuration that is better equipped to handle data loss than RAID6, with a comparable amount of redundancy or number of parity disks. We also show that for a large number of disks, RAID-CUBE tolerates and protects data loss for large number of disks as compared to RAID6.

RAID-CUBE is not an entirely new idea, with the existing precursors such as 2D RAID [7] and its subsequent enhancements [11, 12, 13, 20]. In 2D RAID [7], n^2 disks are arranged in the form of a $n \times n$ two-dimensional square, and each individual row and column is protected by row parity and column parity disks respectively resulting into a total of 2n parity disks. Since in such an arrangement, each disk is protected by one row parity and one column parity disks, higher resiliency is achieved as compared to single parity RAID 5. A recent paper [13] shows that replicated row or column parities achieves even higher resiliency. In this case, if the row parity disks are replicated then the configuration has 2n row parity disks and a total of 3n parity disks.

RAID-CUBE formally extends the concept of arrangement of disks in a two-dimensional grid into higher dimensions. In this configuration, the disks can be arranged in three, four or in any m-dimensional space. First we show how the parity disks are organized in such higher dimensional configuration. We then generalize the reliability analysis (MTTDL) for any number of disk failures in the presence of a large number of disks. We use the state transition diagram as presented in Paris, et al. [12, 13], and generalize the analysis for any number of disk failures. We then use that generalized analysis to evaluate RAID-CUBE along with RAID6 (Dual parity) [4], RAID-TP (Triple Parity) [8] and the configuration proposed by Paris, et al. [12, 13]. Our analysis results show the superiority of RAID-CUBE compared to these alternatives.

We then extend our RAID-CUBE configurations for the case of large sequential IOs [1] in datacenters and propose a mechanism to protect against bit errors without read-modify-write cycles [15].

Overall, in this report we analyze the data loss probability in terms of MTTDL in the presence of large number of disks for various different data protection schemes. We propose a new data protection mechanism that not only provides better protection against disk failures but also protect the bit errors. The rest of the paper is organized as follows. In Section 2, we describe the general configuration of RAID-CUBE. In Section 3, we present a new and generalized formulation for how to compute the *mean time to data loss* (MTTDL) for a large number of disks in a datacenter. We then use this generalized analysis results for two-dimensional and three-dimensional RAID-CUBEs along with RAID6. In Section 4, we present the MTTDL as provided by the analysis in large datacenter for different protection mechanisms and derive certain insights. We then show how RAID-CUBE can be effectively used for bit error protection and recovery for large sequential IOs in datacenters without read-modify-write cycles. In Section 6, we show the significance of interleaved parity disks that is used in the RAID-CUBE configuration. Finally we summarize and draw conclusions in the Section 7.

II. RAID-CUBE

In two dimensions, RAID-CUBE is similar to that of 2D RAID [7, 11, 12] except that it has two extra "parity of the parity" disks as shown in Figure 1. In Figure 1, we show a 5x5 2D RAID-CUBE which has 25 data disks. For each row, we have a parity disk computing the parity of the data disks in that row. Similarly for each column, we have one parity disk computing the parity of the data disks in that column. We have additional two parity disks, one for row parity disks and the other for column parity disks (interleaved parity disks). The parity of the row parity disks computes the parity of the parity disks over rows, and similarly for the parity of the column parity disks.

Note that, the bits stored by these two parity disks are identical (they are replicants of each other), however, use of two such "parity of parity" disks provide significantly more powerful redundancy as discussed in Section 6.

As we move to three dimensions, the data disks are organized in the form of cube, and for each two dimensional plane there is one parity disk as shown in Figure 2. In 3D RAID-CUBE, as shown in Figure 2, we have 5x5x5 = 125 data disks. Along each dimension we have 5 parity disks computing the parity of the respective data disks in a plane normal to that dimension and positioned according to the parity disk. Therefore, in this case we have 3x5 = 15 parity disks. There are three replicated parity of parity disks additionally. We can view each of them as computing the parity of the parity disks in each dimension. In general, for an m-dimensional RAID-CUBE with n data disks in each dimension, we have n^m data disks, $m*n$ parity disks computing the parity of each (m-1)-dimensional structure, and m replicated parity of parity disks. Therefore, total number of disks becomes

$$N = n^m + m(n+1) \quad (1)$$

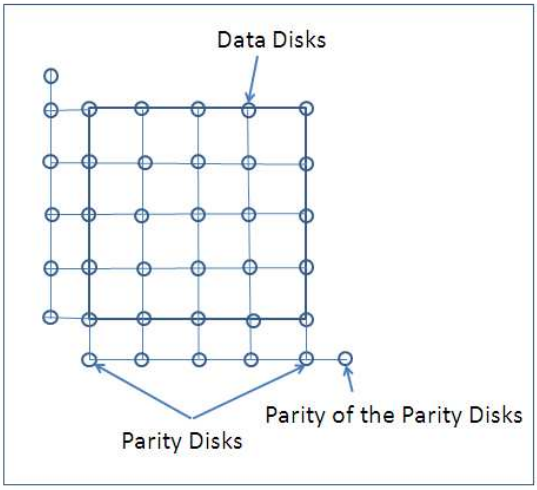

Figure 1: A configurations of 2D RAID-CUBE

For example, if we have 64 data disks, in 8x8 2D RAID-CUBE, we have 2*8+2 = 18 parity disks in total. In three-dimension, for a 4x4x4 3D RAID, we have 3*4+3 = 15 parity disks in total. For the same number of data disks, the resultant number of parity disks decreases as we move to higher dimensional RAID-CUBE.

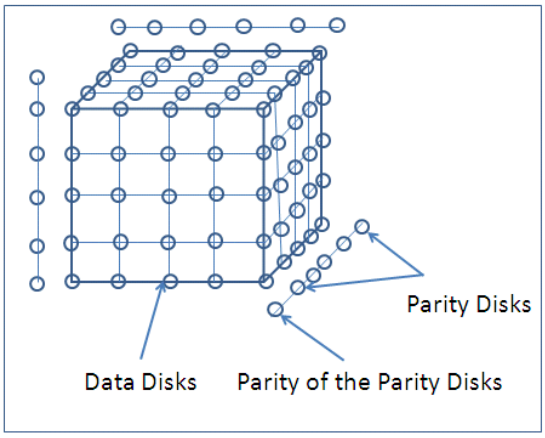

Figure 2: Configuration of 3D RAID-CUBE

III. RELIABILITY ANALYSIS FOR LARGE NUMBER OF DISK FAILURES

We first provide a generalized framework for analyzing the *mean time to data loss* (MTTDL) of a system of disks by extending the model of analysis as provided in Paris, et al. [12, 13]. Next we use the framework to estimate the MTTDL of different protection mechanisms. In general, disk failures may not follow the exponential distribution as studied in [17]. The operational failures can be better modeled by Weibull or Gamma distributions as studied in [17]. The bit errors also do not necessarily follow the exponential distribution. Moreover, the bit errors failure rate may vary substantially depending on temperature, workload pattern, time, and other factors [15, 18]. In this paper, for simplicity of analysis, we assume exponential distribution of failures with a constant rate.

We assume that the disks fail independently or the bit errors happen independently (the assumption may not be true in reality, however, we restrict to this assumption for a computationally realizable analysis of the system). Once a disk fails, the recovery

process immediately starts to recover the data. If multiple disks fail then the recovery of these disks can take place in parallel. We also assume that the recovery rate is exponentially distributed.

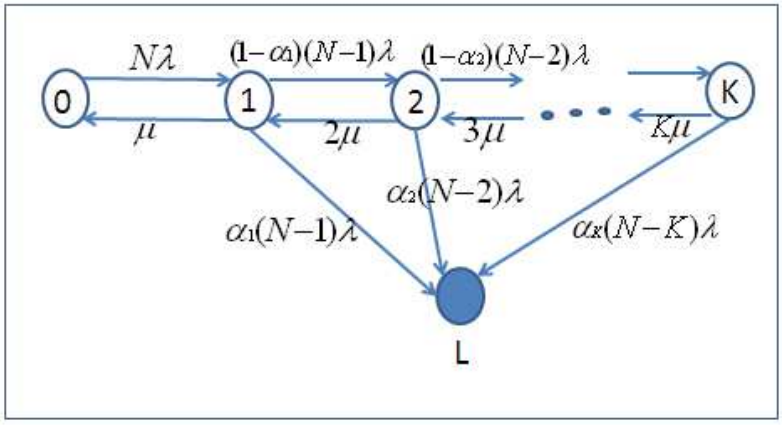

Figure 3: A continuous time Markov chain of the disk failures

Let λ be the failure rate of the exponentially distributed failure process, μ be the recovery rate of the exponentially distributed recovery process. Figure 3 shows a continuous time Markov chain of the entire failure process for different failure rates. The system is initially in state 0 which indicates that there is no failure and all disks are operational. Since there are *N* disks in the system, there can be $N\lambda$ probability of one disk failure. After one disk failure, the system goes to state 1. The failed disk can be replaced and data can be recovered with a recovery probability μ. Once the data is recovered the system returns to state 0. In this analysis we assume that there is no data loss if there is one disk failure. The analysis can be more generalized considering that there is a probability of data loss even if there is one disk failure.

In State 1, if another disk fails then there may be data loss with a probability $\alpha_1\lambda$. Since there are (*N*-1) operational disks in State 1, there is a probability $(N-1)\alpha_1\lambda$ of data loss. If there is data loss, system moves to the data loss state *L*. If there is no data loss then system moves to State 2 with 2 disk failures with a probability $(1-\alpha_1)(N-1)\lambda$. Since both he failed disks can be replaced and recovered simultaneously, the system returns from State 2 to State 1 with a probability 2μ. Extending this logic, we can say that the system can move from a state *i* to state *i*+1 with a probability $(1-\alpha_i)(N-i)\lambda$, and there is a probability $\alpha_i(N-i)\lambda$ of data loss. If the data loss happens then the system moves to data loss state *L*. The system can return from state *i*+1 to state *i* with a probability $(i+1)\mu$. Let there be at most *K* disk failures after which the system cannot be returned to normal state if one more disk fails, and data loss becomes inevitable. In that case, $\alpha_K = 1$ and the system goes to data loss state *L* after one more disk fails. With constant rates of failure and recovery under the assumption of the exponentially distributed failure and recovery processes, we can express the transient changes in the probability of system states $P_0, P_1, P_2, P_3, \ldots, P_K$ as the Kolmogorov system of differential equations [19] given by Equation (2).

$$
\begin{aligned}
&\frac{dP_0(t)}{dt} = \mu P_1(t) - N\lambda P_0(t)\\
&\frac{dP_1(t)}{dt} = N\lambda P_0(t) - ((N-1)\lambda + \mu)P_1(t) + 2\mu P_2(t)\\
&\vdots\\
&\frac{dP_{i+1}(t)}{dt} = (1-\alpha_i)(N-i)\lambda P_i - ((i+1)\mu + (N-(i+1))\lambda)P_{i+1}(t) + (i+2)\mu P_{i+2}(t)\\
&\frac{dP_K(t)}{dt} = (1-\alpha_{K-1})(N-K+1)\lambda P_{K-1} - (K\mu + (N-K)\lambda)P_K(t)\\
&\frac{dP_L(t)}{dt} = \sum_{i=1}^{K} \alpha_i(N-i)\lambda P_i(s)
\end{aligned}
\qquad (2)
$$

In Equation (2), $i+2 \le K$, $P_0(0)=1,$ and $P_i(0)=0$ for all $i>0$.

It is very difficult to obtain closed form solution of the system of differential equations for a large number of system states. First we obtain the Laplace transformation of the system of equations. The mean time to data loss (MTTDL) can be expressed as [19].

$$MTTDL = -\frac{dL}{ds}\bigg|_{s=0} \tag{3}$$

The loss in the system can be expressed as

$$L(s) = sP_L(s) = U(s)/V(s) \tag{4}$$

Where *U(s)* and *V(s)* are given as

$$V(s) = \beta_0(s) \tag{5}$$

Where

$$\beta_i = (s + i.\mu + (N-i)\lambda)\beta_{i+1} - (i+1)(N-i)(1-\alpha_i)\mu\lambda\beta_{i+2} \tag{6}$$

with $\beta_{K+1} = 1$, and $\beta_j = 0$ for all $j > K+1$.

The numerator *U(s)* in Equation (4) is given as

$$U(s) = \sum_{i=1}^{K} \alpha_i \prod_{j=0}^{i} (N-j) \prod_{j=1}^{i-1} (1-\alpha_j)\lambda^{i+1}\beta_{i+1}(s) \tag{7}$$

Therefore the loss function $L(s)$ is determined by $\beta(s)$ which in turn depends on the coefficients α . We derived the expression for *L(s)* by recursively computing the values of $P_1(s), P_2(s), P_3(s)$ and so on. We then find the generalized form of equation that can express the *U(s)* and *V(s)*. Equating the negative of derivative of the loss function and letting *s*=0, we obtain the MTTDL for given values of α .

Therefore,

$$MTTDL = \frac{V(0)\dfrac{dU(s)}{ds}\bigg|_{s=0} - U(0)\dfrac{dV(s)}{ds}\bigg|_{s=0}}{V(0)^2} \tag{8}$$

Since the derivatives are difficult to express in closed form equations, we programmatically compute the MTTDL using recursive formulations instead. Depending on the system configuration, the values of α change and the respective MTTDLs change. Next we analyze the MTTDL for RAID CUBE.

A. Two Dimensional RAID-CUBE

In 2D RAID-CUBE, it is obvious that $\alpha_1 = 0$. Similarly for three disk failures, the data can be recovered unlike the case shown in [12]. If there are four disk failures then there are certain special cases when the data cannot be recovered as shown in Figure 4. In the figure, the transparent circles represent the operational disks whereas the black represent the failed disks.

The four failed disks govern four parity equations; however, only three of these equations are independent, and therefore the data cannot be recovered in this case. Apart from these type special arrangements of the four failed disks, the data can be recovered for other cases. Let there be $n \times n$ data disks in the in 2D CUBE and $2n+2$ parity disks. In order to analyze the probability of

data loss, let us assume that there is only one parity of parity disks. In that case, it is organized as $(n+1)\times(n+1)$ disks. Any two disks can be chosen from a row, and the number of possible combinations is

$$\binom{n+1}{2} \tag{9}$$

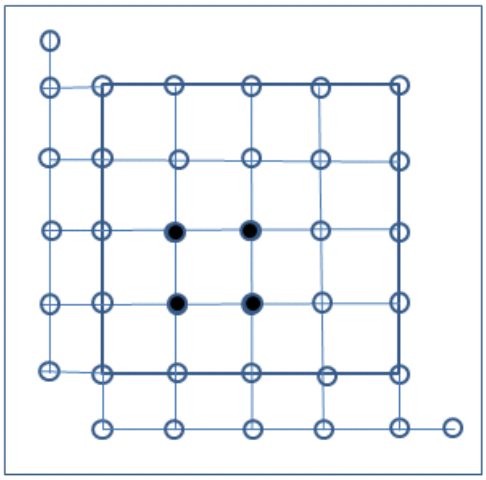

Figure 4: A four disk failure resulting into data loss

Once we choose a row, there are n disks in a column, out of which the third disk can be chosen. So the total number of possibilities is

$$\binom{n+1}{2}n \tag{10}$$

In our $n\times n$ 2D CUBE, the failure of parity of parity disk does not happen because there are 2 such parity disks. If both are failed then we are left with a choice of two disks and therefore, the data loss will not happen. So the number of combinations in which the single parity of parity disk is involved is n^2. We are therefore left with a number of possible configurations in which four disk failure can result in data loss is

$$\binom{n+1}{2}n - n^2 = \binom{n}{2}n \tag{11}$$

If there is a series of l such $n\times n$ 2D CUBEs then we have total number of disks as

$$N = (n^2 + 2n + 2)l \tag{12}$$

We therefore have

$$\alpha_3 = \frac{\binom{n}{2}nl}{\binom{N}{4}} \tag{13}$$

Any combination of five disk failure which has a four disk arrangement as discussed above will lead to data loss. In addition there are certain special cases involving the parity of the parity disks that lead to data loss as shown in Figure 5.

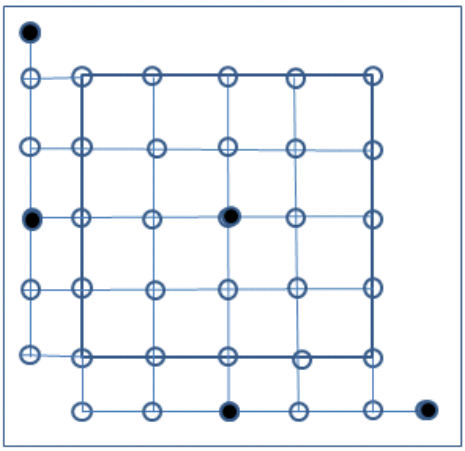

Figure 5: A specific five disk failure resulting into data loss

In Figure 5, both the parity of the parity disks get corrupted and therefore, the row parity and column parity cannot be recovered. As a result the data of the data disk cannot be recovered. This case can happen if any one data disk is selected out of the n^2 data disks and the locations of the parity disks are determined by the location of the data disk. Therefore the total possible combination of data loss configurations due to five disk failures in a $n \times n$ CUBE is given as

$$\binom{n}{2} n\left(n^2 + 2n - 2\right) + n^2 \tag{14}$$

Therefore, we obtain

$$\alpha_4 = \frac{\left(\binom{n}{2} n\left(n^2 + 2n - 2\right) + n^2\right) l}{\binom{N}{5}} \tag{15}$$

In the case of six failures, we can select any one disk in addition to the configurations for five disk failure, and those results into data loss. Therefore,

$$\alpha_5 = \frac{(N-5)\binom{N}{5}\alpha_4}{\binom{N}{6}} = 6\alpha_4 \tag{16}$$

In general, we have

$$\alpha_{i+1} = \min(1,(i+2)\alpha_i) \quad \text{for } i \geq 5. \tag{17}$$

The minimum condition is imposed because as α cannot be greater than unity, and if it goes to unity that conditions an inevitable data loss.

B. Three Dimensional RAID CUBE

Just as in the case of 2D RAID CUBE, in 3D CUBE also, we have $\alpha_1 = \alpha_2 = 0$. However, similar to the 2D CUBE, if four disks fail in the same plane then data cannot be recovered. The situation is illustrated in Figure 6.

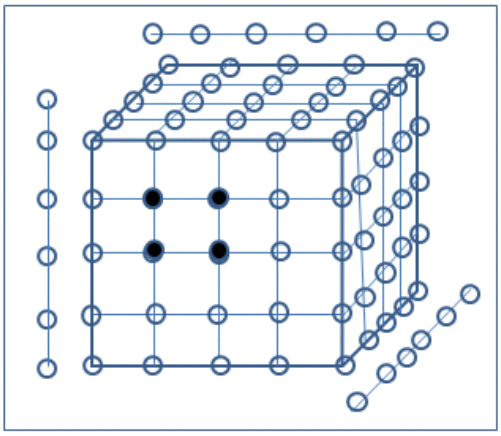

Figure 6: A four disk failure in 3D CUBE resulting into data loss

Since the four nodes encompass a rectangular shape, the parity equations are not independent and the data cannot be recovered. Let us assume that we have $n \times n \times n$ 3D CUBE and the total number of disks in l such 3D CUBEs is given as

$$N = (n^3 + 3n + 3)l \tag{18}$$

Since there are 3 dimensions, the coefficient α_3 is given as

$$\alpha_3 = \frac{3\binom{n}{2}n^2 l}{\binom{N}{4}} \tag{19}$$

Any five disk failure can happen due to a configuration of a four disk failure as above and any one more disk failure. However, in the 3D CUBE, the special case of five disk failure as in 2D CUBE as illustrated in Figure 5 does not happen because of the presence of three replicated parity of the parity disks. Therefore we have

$$\alpha_4 = \min(1,5\alpha_3) \quad \text{and} \quad \alpha_5 = \min(1,6\alpha_4) \tag{20}$$

However, there is a special case of seven disk failures where all three parity of the parity disks fail as illustrated in Figure 7.

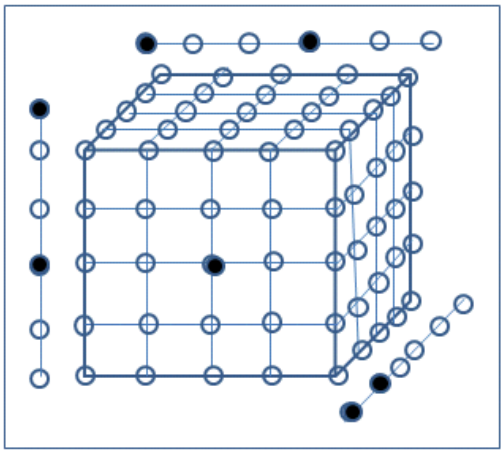

Figure 7: A specific seven disk failure resulting into data loss

In this case, none of the parity disks can be recovered and therefore the data is lost completely. We therefore have

$$\alpha_6 = \min\left(1,7\alpha_5 + \frac{n^3 l}{\binom{N}{7}}\right) \tag{21}$$

For any more disk failures, we have

$$\alpha_{i+1} = \min(1,(i+2)\alpha_i) \text{ for } i \geq 7 . \tag{22}$$

C. RAID6 (Dual Parity)

In RAID6, an array of disks is protected by tow parity disks. For example, in (8+2) RAID-DP, we have 8 data disks and two parity disks. We consider the general case of l arrays of $(n+2)$ RAID-DP. RAID-DP can protect against any two disk failures in an array. Therefore we have $\alpha_1 = 0$. If there are three disks failure in a single array then there is data loss. Therefore we have

$$\alpha_2 = \frac{\binom{n+2}{3} l}{\binom{N}{3}} \tag{23}$$

where $N = (n+2)l$ (24)

For any subsequent failure, we have

$$\alpha_{i+1} = \min(1,(i+2)\alpha_i) \text{ for } i \geq 3. \quad (25)$$

IV. ANALYSIS RESULTS

In our analysis, first we compute the MTTDL for different MTTR (recovery rates) in 2D CUBE and 3D CUBE and compare the performance with RAID6. We consider a standard five-year life span of the disks. Note that, in today's standard the life span of the disks can be more; however, we restrict our analysis to five year life-span. A five-year life-span means $\lambda = 1/(365*5)$. We consider 64 data disks arranged in 8×8 2D CUBE and $4\times 4\times 4$ 3D CUBE. As a comparison, we also study the MTTDL of 8 arrays of 8+2 RAID6.

A. MTTDL for 64 Data Disks

In the case of 2D CUBE, we have 18 parity disks in total; in the case 3D CUBE, we have 15 parity disks, and for RAID6 we have 16 parity disks. The MTTDL is computed in all cases for an MTTR in between 12 to 48 hours (i.e., between half a day and 4 days). Figure 8 illustrates the various MTTDL achieved for different configurations for different MTTR. Evidently, the 2D and 3D RAID CUBEs substantially outperforms the existing RAID6 configuration. 2D RAID CUBE edges over 3D CUBE using additional 3 parity disks for 64 data disks. Interestingly, although RAID6 has one more parity disk as compared to 3D CUBE, its MTTDL is substantially less than 3D CUBE.

We also formally compare the RAID-CUBE performance with replicated row or column parity disks as presented in Paris et al. [13]. In [13], the authors have used a two-dimensional $n\times n$ grid of data disks having n^2 disks, a set of n column parity disks, and a set of replicated 2n row parity disks.

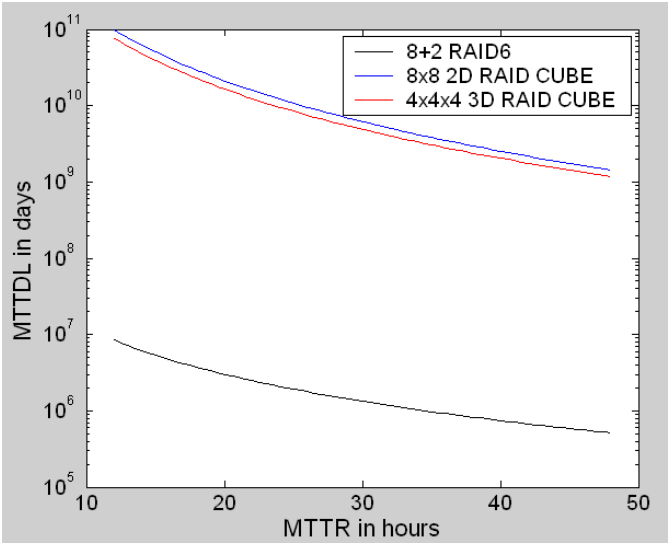

Figure 8: MTTDL vs. MTTR for three different cases in 64 data disks

In such a configuration, we have a total of $n^2 + 3n$ disks for n^2 data disks. Figure 9 shows the MTTDL for 2D and 3D RAID-CUBE and that of Paris et al.[13] for 64 data disks and MTTR ranging from 12 to 48 hours. Here, we consider that the life-time of a disk is 5 years on an average. Note that, the number of parity disks required in Paris, et al.[13] is much more than RAID-CUBE although the performance of RAID-CUBE is better. In 2D RAID CUBE, we have 18 parity disks, 3D CUBE we have 15 parity disks, and in Paris, et al. [13], we have 24 parity disks.

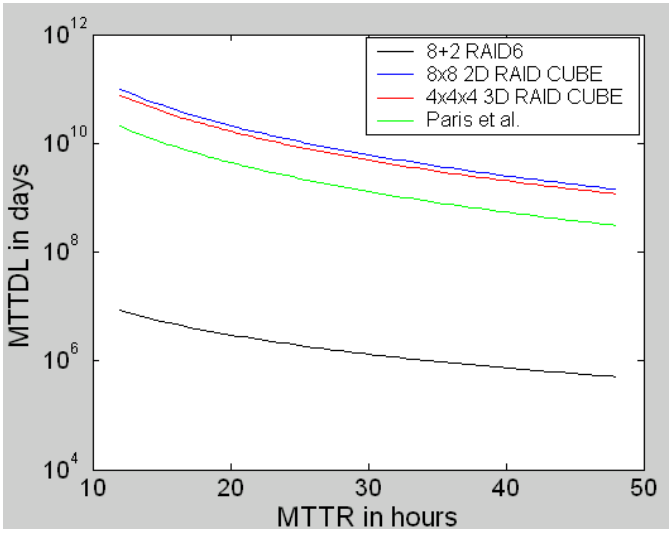

Figure 9: A comparison of the performance of RAID-CUBE with that of Paris et al. [13]

We compare the reliability of the RAID-CUBE with that of RAID Triple parity as presented in [8]. We derive the MTTDL of RAID-TP in the same way as RAID-DP as presented in Equation (23). RAID-TP can always protect against three disks failure. However, in the case of four disks failures, there is a possibility of data loss.

We consider that there are l arrays of $(n+3)$ disks, each array comprising of n data disks and 3 parity disks. If there are three disks failures in any array then also RAID-TP is able to recover the data. Therefore, we have

$$\alpha_1 = \alpha_2 = 0 .$$

If there is a fourth disk failure in one array then there will be data loss. We therefore have,

$$\alpha_3 = \frac{\binom{n+3}{4} l}{\binom{N}{4}} \tag{26}$$

where $N = (n+3)l$. For any subsequent disk failures, we have

$$\alpha_{i+1} = \min(1, (i+2)\alpha_i) \quad \text{for } i \geq 4 . \tag{27}$$

We also model the data loss probability of Reed-Solomon Code (Erasure code). Let us assume that there are l arrays of (n,k) RS codes where k is the number of data disks and (n-k) is the number of parity disks. RS code in that case can tolerate any (n-k) failures. We therefore have $\alpha_1 = \alpha_2 = \alpha_{n-k-1} = 0$. For a subsequent failure, we have

$$\alpha_{n-k} = \frac{\binom{n}{n-k+1}^{l}}{\binom{N}{n-k+1}} \tag{28}$$

where $N = n*l$ is the total number of disks. For any subsequent failure, we have the same equation as in (27).

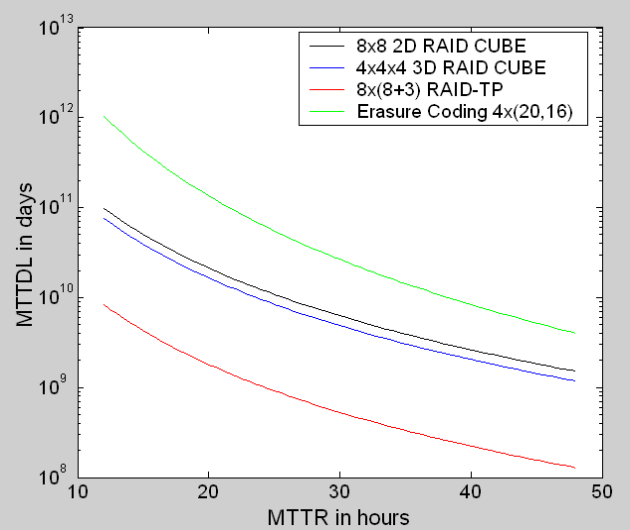

Figure 10: A comparison of the performance of RAID-CUBE with that of RAID-TP and Erasure Coding

We compare the performance of 2D and 3D RAID-CUBE with that of RAID-TP and four arrays of (20,16) erasure code in Figure 10. We consider 64 data disks with 8 arrays of (8+3) RAID-TP configuration. Thus we have 24 parity disks. We maintain the same configurations of 2D and 3D RAID-CUBEs as discussed before having 18 and 15 parity disks respectively for 64 data disks. We illustrate the MTTDL for an MTTR between 12 to 48 hours. From Figure 10, we observe that RAID-CUBE provides superior protection than RAID-TP, even with less number of parity disks. However, erasure code is superior to RAID-CUBE but the former is computationally expensive.

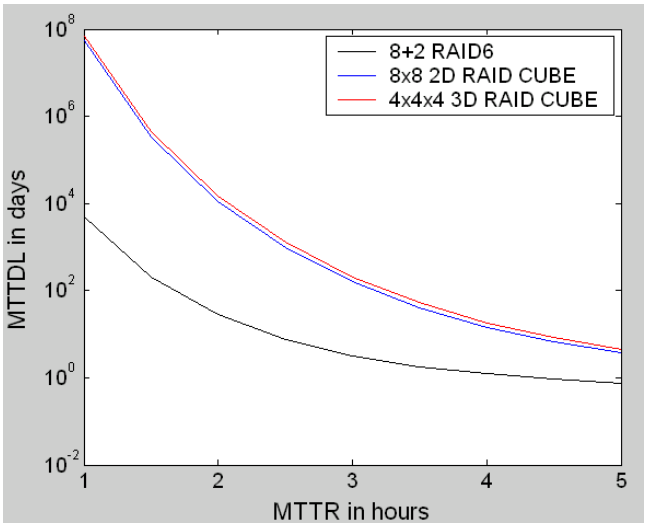

Figure 11: MTTDL vs. MTTR in three configurations for 64000 data disks.

B. MTTDL for Large Number of Disks

Our claim in the introduction was that as the number of disks increases in a datacenter, the data loss becomes inevitable. We substantiate this claim using 64000 data disks. We organize the data disks in three ways: (1) an array of 1000 8×8 2D CUBEs; (2) an array of 1000 $4\times4\times4$ 3D CUBEs and (3) an array of 8000 (8+2) RAID6. We consider an MTTR between 1 to 5 hours. Figure 11 illustrates the expected performance of the three different schemes namely 2D and 3d RAID-CUBEs and RAID-DP.

We observe that for an MTTR = 3 hours, the MTTDL for RAID6 becomes close to one day. That signifies that the data loss in RAID6 becomes inevitable. On the other hand, for an MTTR = 3 hours, both 2D and 3D RAID CUBEs have MTTDL close to 100 days. Therefore, the new scheme is much better in terms of data protection as compared to RAID6.

Next we analyze the chance of data loss as the number of disks increases for a fixed MTTR. We keep the configurations of 2D CUBE, 3D CUBE, and RAID6 same as other two analysis and increase the number of arrays of such configurations. Figures 12, 13, 14, and 15 illustrate the MTTDL of different schemes with an increasing number of disks using a fixed MTTR = 3, 6, 9, 12 hours respectively.

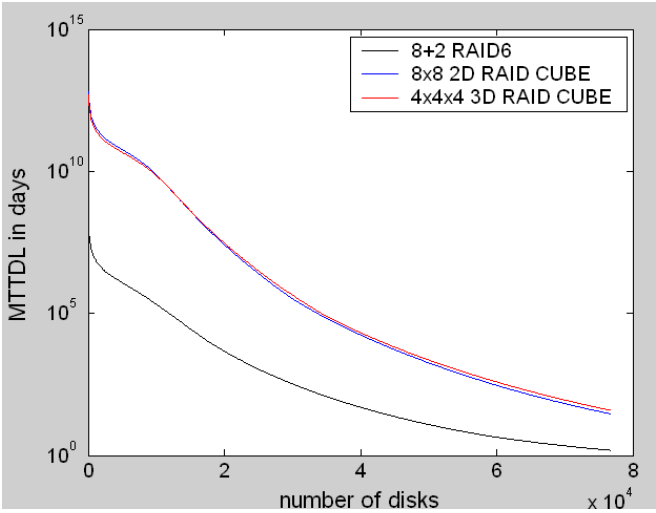

Figure 12: MTTDL vs. number of disks for an MTTR=3 hours.

From Figure 12, we observe that 2D CUBE and 3D CUBE can have approximately 100 days of MTTDL with 80000 data disks. On the other hand, to have 100 days MTTDL, RAID6 can accommodate 50000 data disks in a datacenter. As the MTTR increases, the reliability of all configurations becomes poor. For example, as in Figure 13, with an MTTR = 6 hours, 40000 disks can be accommodated in a datacenter to have 100 days MTTDL for both 2D and 3D RAID CUBEs.

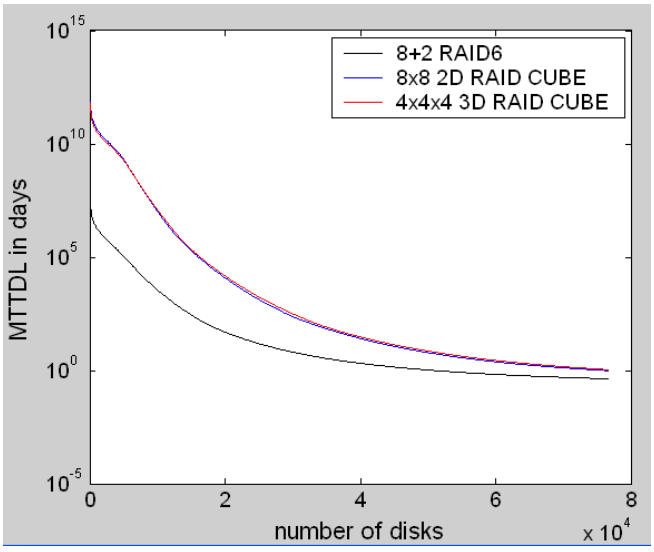

Figure 13: MTTDL vs. number of disks for an MTTR = 6 hours.

On the other hand to have the same MTTDL, RAID6 can only accommodate 20000 disks in a datacenter. The same behavior can be observed from Figures 14 and 15 where 2D and 3D CUBEs are consistently more reliable than RAID6.

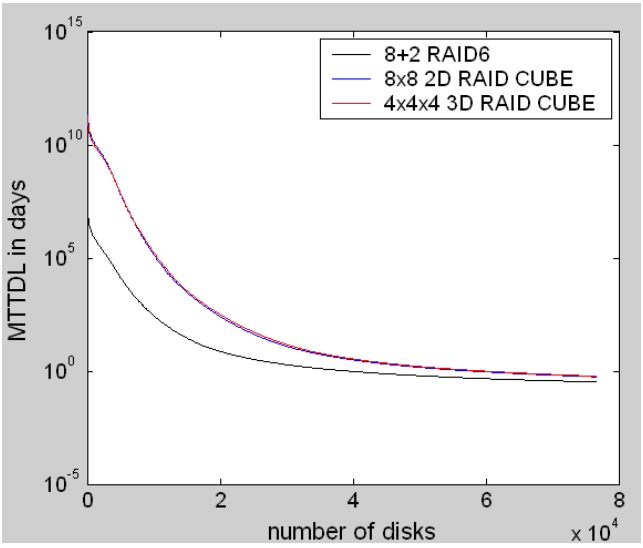

Figure 14: MTTDL vs. number of disks for an MTTR = 9 hours.

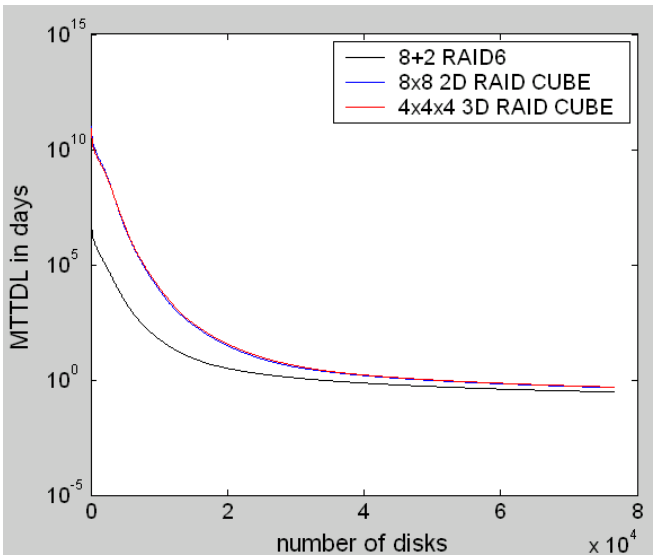

Figure 15: MTTDL vs. number of disks for an MTTR = 12 hours.

For 64000 data disks, we report the MTTDL for the three different configurations for MTTR = 2 – 4 hours in the interval of 5 hours in Table 1. Interestingly, we observe that 3D CUBE has better reliability than 2D CUBE for large number of disks. As the number of disks is relatively smaller, 2D RAID CUBE provides better reliability. However, as the number of disks becomes large, the 3D RAID CUBE performs better.

We can observe that even if we maintain an MTTR = 2 hours, RAID6 is highly unreliable and there is a possibility of data loss in around 28 days. On the other hand, 3D RAID CUBE is able to protect the data for around four years with an MTTR = 2 hours.

Next, in Table 2, we compare the MTTDL for different number of data disks for an MTTR = 3 hours. We observe from Table 2 that if a datacenter is able to maintain MTTR = 3 hours then it can accommodate approximately 32000 disks with an MTTDL =

216 days using RAID6 (8+2) configurations. On the other hand, same MTTDL can be maintained even if double the number of data disks (64000) is accommodated in the datacenter using 3D RAID (4x4x4) configuration.

Table 1: MTTDL vs. MTTR for three configurations with 64000 data disks.

MTTR in hours	MTTDL in days		
	RAID6	2D RAID CUBE	3DRAID CUBE
2.0	28.5	1098.2	1479.8
2.5	7.7	958.9	1288.1
3.0	3.2	157.3	208.4
3.5	1.8	40.4	52.4
4.0	1.2	14.5	18.3

Table 2: MTTDL vs. number of disks for three configurations for an MTTR = 3 hours.

Number of Data Disks in Multiple of 64	MTTDL in number of days		
	8+2 RAID6	8x8 2D RAID CUBE	4x4x4 3D RAID CUBE
300	6133.4	39981000	45085000
400	949.6	1979200	2495300
500	216.8	170800	218210
600	65.1	26690	31858
700	24.2	5587.8	6852.4
800	10.7	1445.6	1829.9
900	5.5	443.2	576.6
1000	3.2	157.3	208.4

We observe that 3D RAID-CUBE provides better MTTDL as the number of disks becomes very large, with fewer parity disks versus 2D RAID-CUBE configurations. Furthermore, the RAID-CUBE achieves higher resiliency as compared to replicated row or column parity disks [13] with much fewer parity disks. We observe that it is very difficult for a datacenter to protect data loss if we have MTTR more than 3 hours. As it increases, data loss becomes inevitable. Erasure coding can provide much better data protection with increased computational complexity. RAID-CUBE provides a higher data protection capability as compared to RAID6 with relatively less complexity as compared to erasure coding.

V. BIT ERROR PROTECTION AND RECOVERY

Our original claim was to protect the data against latent defects and operational failures. In [15], the authors investigated how latent errors followed by operational disk failures result in data loss in the case of RAID. Sometimes it is very hard to detect the latent defect or bit errors even by the parity computation mechanisms. In this section, we provide an outline to protect against the bit errors using the RAID CUBE technology.

Traditionally, both random and sequential IOs are prevalent. As mentioned in Section 1, we believe that a large number of random IOs will be handled or are being handled in the SSD layer, while large sequential IOs will be directed to the local disks. A large sequential IO can be sized so as to be split and distributed among all disks in the CUBE. This has the advantage that Read-Modify-Writes can be avoided, and parity can be computed more efficiently, as we show below. For example, let there be a large sequential IO of size 64MB. Note that with modern applications and systems (e.g., HDFS workload [1,10]), a size of 64MB sequential IO size on a disk subsystem is not atypical. The sequential IO can be divided into 1MB chunks and stored in the 64 disks of an 8×8 2D RAID CUBE or $4 \times 4 \times 4$ 3D RAID CUBE.

To reduce the overhead of parity computation, an I/O can be distributed across the disks with the same PBN (physical block number). Whenever, an IO happens, the chunks are written with the same PBN of different disks. In the same operation, the row parity and the column parity can be computed. In this example, each parity size will be 1MB and stored in the row parity and column parity disks with the same PBN. As an illustration, we show how column parity can be computed for different chunks in Figure 16.

Similarly, the row parity and the parity of the parity can be computed for the overall disk subsystem. One restriction in this mechanism is that I/O is to be symmetrically distributed among all disks. The chunk size can also be less than 1 MB. With this kind of arrangement, if there is a bit error, it can be easily detected during the read operation using the parity check mechanism.

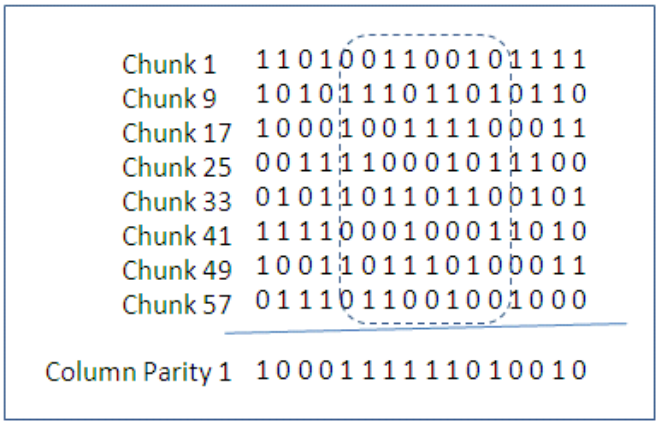

Figure 16: Column parity computation for different chunks.

The write process need not follow a *read-verify-write* mechanism. However, in this scheme if four chunks in the same IO in four different disks (arranged in the form of a rectangle as we discussed before) suffer bit errors then the bit error cannot be recovered. A specific chunk can sustain multiple bit errors since each bit is independently protected by the parity mechanism within the same chunk.

VI. SIGNIFICANCE OF THE INTERLEAVED PARITY

In RAID CUBE, we maintain multiple copies of the interleaved parity. First of all, interleaved parity is important [20]. As shown in [11,12], a similar structure without interleaved parity cannot sustain certain types of three disk failures. Use of interleaved parity enables us to sustain all types of three disk failures.

It is possible to maintain only one "parity of parity" disks (interleaved parity disk) instead of multiple replicated disks. Use of replicated interleaved parity disks has significance both from practical operational point of view and from the reliability angle.

Whenever, there is a write operation in the RAID-CUBE, the interleaved parity is updated which is not true for row and column parity disks. Therefore, the disks storing the interleaved parity can have significantly less life-span than other disks. In our analysis, we assumed that the life-spans of all disks are same which may not be true. Replication of the interleaved parity disks enables the sustenance of such skewed writing operations.

Next we analyze the MTTDL of 2D RAID CUBE and 3D RAID CUBE (having 64 data disks) with that of a similar 2D structure with only one interleaved parity disk. In analyzing the MTTDL, we can see that the chance of data loss for four disk failures in the case of one interleaved parity disk is

$$\alpha_3 = \frac{\binom{n+1}{2} n}{\binom{N}{4}} \tag{29}$$

Any subsequent probability of data loss for larger number of disk failures can be expressed as

$$\alpha_i = \min((i+1)\alpha_{i-1}, 1) \tag{30}$$

Using these values of α s, we can determine the MTTDL for a similar 2D CUBE organization with one interleaved parity disk and 64 data disks. Figure 17 illustrates the MTTDL of a group of 64 disks for different MTTR. Here also we considered the life-span of a disk to be 5 years.

We can observe that 2D CUBE with single interleaved parity disk is comparable with 3D CUBE with 3 parity disks (only for 64 disks); however, it is worse than a similar 2D CUBE with two parity disks. Note that the 3D CUBE shows inferior performance than 2D CUBE only for smaller number of disks. As the number of disks increases, 3D CUBE performs better than 2D CUBE as we discussed before.

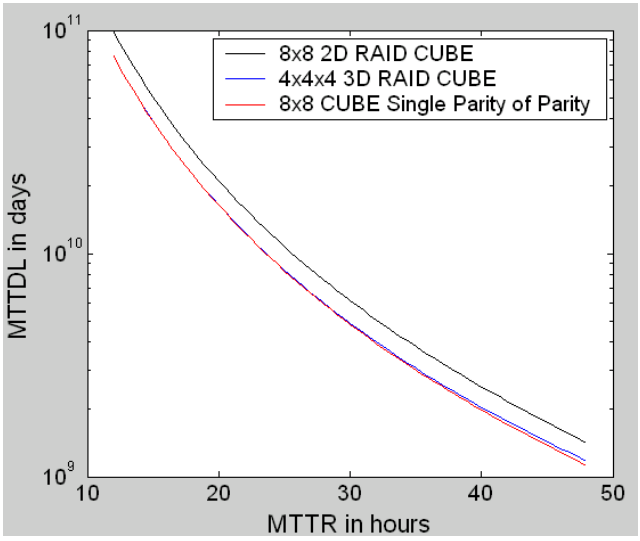

Figure 17: MTTDL vs. MTTR for 2D RAID CUBE with a single interleaved parity disk.

VII. SUMMARY AND CONCLUSIONS

We have shown a new configuration, RAID-CUBE, that is more resilient than existing RAID-based data protection mechanisms, including RAID dual parity [4], RAID triple parity [8], and the two-dimensional RAID layout as proposed by Paris, et al. [13]. RAID-CUBE is essentially a generalization of the configuration proposed by Paris, et al. [11,12], with additional interleaved parity disks. We provided a new and generalized analytical technique to compute the mean time to data loss for a very large number of disks in a datacenter. Using it, we have shown that RAID-CUBE accommodates more capacity (number of disks) in a datacenter as compared to the existing RAID configurations with the same mean time to data loss. We also show analytically that mean time to recovery (MTTR) is a very important parameter to maintain a reasonable limit on the MTTDL. Usually RAID is not well equipped to protect against the bit errors without the read-modify-write cycles. We show a mechanism to protect against the bit errors without using read-verify-write cycles for the kind of large sequential IOs that now dominate modern datacenter workloads [1,10].

In our analysis, we did not differentiate between the disk failure rate and the bit error rate. In practice, bit error rate characteristics are different [18] and much higher than the disk failure rates [15]. Further refinement to our analytical model can be performed to differentiate between these two rates. Our analysis simplistically assumes exponential distribution for bit errors and disk failures. As shown in [17], disk failures do not necessarily follow the exponential distribution, and Gamma or Weibull distribution [17] may be better fit for such failures. However, it is very difficult to analytically express the transient differential equations for data loss using the Weibull distribution. An elaborate simulation technique may possibly reveal the MTTDL more accurately. In modeling the transient state probability equations, we also ignored the other underlying processes such as disk

scrubbing, bursty writes, physical conditions such as temperature [16], etc. We also considered the disks failing independently of each other. In correlated failures (e.g., in [16]), the failure rate can vary substantially. A thorough analysis of the state diagram is required for this purpose.

In our analysis, we did not address the issue of reconstruction bandwidth of the protection mechanism. For example, in 3D RAID-CUBE, all the disks in a single plane will be involved in reconstructing the data of one failed disk. On the other hand, in RAID6, the number of disks involved in the reconstruction process is a single array and less than that of the 3D CUBE. Further analysis is required.

We have provided a generalized analysis to derive MTTDL for very large number of disks in a datacenter. Our analysis and generalization provides insights into how data loss can happen in datacenters with large number of disks, and how a new RAID configuration can provide better protection than the existing variants.

REFERENCES

[1] Abad, C. L., Roberts, N., Lu, Y., and Campbell, R. H.: A Storage-Centric Analysis of Map-Reduce Workloads: File Popularity, Temporal Locality and Arrival Patterns. In Proceedings of the 2012 IEEE International Symposium on Workload Characterization (IISWC '12), pp. 100-109.

[2] Blaum, M., and Roth, R. M. On lowest density MDS codes. IEEE Transactions on Information Theory 45, 1 (January 1999), 46–59.

[3] Cidon, A., Rumble, S. M., Stutsman, R., Katti, S., Ousterhout, J., and Rosenblum, M.: Copysets: reducing the frequency of data loss in cloud storage. Proceedings of the 2013 USENIX conference on Annual Technical Conference, pp. 37-48.

[4] Corbett, P., English, B., Goel, A., Grcanac, T., Kleiman, S., Leong, J. and Sankar, S: Row-diagonal parity for double disk failure correction. In Proc. USENIX Conf. on File andStorage Technologies, pp. 1–14, 2004.

[5] Dimakis, A.G., Godfrey, P.B., Wainwright, M.J., Ramchandran, K.: Network Coding for Distributed Storage Systems. INFOCOM 2007. 26th IEEE International Conference on Computer Communications, pp. 2000-2008.

[6] Ghemawat, S., Gobioff, H., and Leung, S.-T.: The Google File System. In *SOSP'03,* October 19–22, 2003, Bolton Landing, New York, USA.

[7] Gibson, G., Hellerstein, L., Karp, R. M., Katz, R. H., and Patterson, D. A.: Coding Techniques for Handling Failures in Large Disk Arrays. Technical Report, University of California at Berkeley Berkeley, CA, 1988.

[8] Goel, A. and Corbett, P.: RAID triple parity. Operating Systems Review 46(3):41-49 (2012)

[9] Hafner, J. L., Deenadhayalan, V., Rao, K. K., and Tomlin, A. Matrix methods for lost data reconstruction in erasure codes. In FAST-2005: 4th Usenix Conference on File and Storage Technologies (San Francisco, December 2005), pp. 183–196.

[10] Pan, F., Yue, Y., Xiong, J., and Hao, D.: I/O Characterization of Big Data Workloads in Data Centers. In The Fourth workshop on Big Data Benchmarks, Performance Optimization, and Emerging Hardware (Co-located with ASPLOS 2014 – Salt Lake City, Utah, USA).

[11] Paris, J.-F., Schwarz, T.J.E., and Long, D.D.E.: Self-Adaptive Two Dimensional RAID Arrays. In IEEE International Conference on Performance, Computing, and Communications, 2007. IPCCC 2007, pp. 246 – 253.

[12] Pâris,J.-F. Schwarz, T. J. E., Amer, A., and Long, D.: Highly Reliable Two-Dimensional RAID Arrays for Archival Storage. In 31st IEEE International Performance Computing and Communications Conference, Dec 1st - 3d, 2012, Austin, TX, USA.

[13] Paris, J.-F., Schwarz, T. S. J., Amer A., and Long, D.D.E:Protecting RAID Arrays Against Unexpectedly High Disk Failure Rates, In Proceedings of the 20th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC 2014), Singapore, Nov. 2014.

[14] Patterson, D.A., Gibson, G. and Katz, R. H.: A case for redundant arrays of inexpensive disks (RAID). In Proceedings of the 1988 ACM SIGMOD international conference on Management of data, pp 109-116.

[15] Rozier, E. W. D., Belluomini, W., Deenadhayalan, V., Hafner, J., Rao, K. K., and Zhou, P.: Evaluating the Impact of Undetected Disk Errors in RAID Systems. In IEEE/IFIP International Conference on Dependable Systems & Networks, 2009. DSN '09, pp. 83 – 92.

[16] Sankar, S., Shaw, M., Vaid, K., and Gurumurthi, S.: Datacenter Scale Evaluation of the Impact of Temperature on Hard Disk Drive Failures. ACM Transactions on Storage (TOS), Volume 9 Issue 2, July 2013.

[17] Schroeder, B. and Gibson, G. A.: Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you?. In FAST'07: 5th USENIX Conference on File and Storage Technologies, San Jose, CA, Feb. 14-16, 2007.

[18] Schroeder, B., Damouras, S., and Gill, P.: Understanding latent sector errors and how to protect against them. ACM Transactions on Storage (TOS), Volume 6 Issue 3, September 2010.

[19] Trivedi, K. S.: Probability & statistics with reliability, queuing and computer science applications. John Wiley & Sons, 2008

[20] Wildani, A., Schwarz, T. S. J., Miller, E. L., and Long, D. D. E.: Protecting against rare event failures in archival systems. Proc. 17th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS 2009), pp. 246–256, Sep. 2009.