2015 IEEE 21st Pacific Rim International Symposium on Dependable Computing

Triple failure tolerant storage systems using only exclusive-or parity calculations

Thomas Schwarz, S.J.
Universidad CentroAmericana
La Libertad, El Salvador
tschwarz@calprov.org

Darrell D.E. Long
University of California
Santa Cruz, CA
darrell@cs.ucsc.edu

Jehan-François Pâris
University of Houston
Houston, TX
jfparis@uh.edu

***Abstract*—We present a disk array organization that can survive three simultaneous disk failures while only using exclusive-or operations to calculate the parities that generate this failure tolerance.**

The reliability of storage systems using magnetic disks depends on how prone individual disks are to failure. Unfortunately, disk failure rates are impossible to predict and it is well known that individual batches might be subject to much higher failure rates at some point during their lifetime. It is also known that many disk drive families, but not all, suffer a substantially higher failure rate at the beginning and some at the end of their economic lifespan.

Our proposed organization can be built on top of a dense two-failure tolerant layout using only exclusive-or operations and with a ratio of parity to data disks of $2/k$. If the disk failure rates are higher than expected, the new organization can be super-imposed on the existing two-failure tolerant organization by introducing $(k+1)/2$ new parity disks and $(k+1)/2$ new reliability stripes to yield a three-failure tolerant layout without moving any data or calculating any other parity but the new one. We derive the organization using a graph visualization and a construction by Lawless of factoring a complete graph into paths.

I. Introduction

While flash technology has conquered an important segment of the storage market and while new storage technologies such as phase-change memories hold promises for the future, magnetic disk drives will archive the majority of the hundreds of exabytes of data stored annually. This is because of the attractive cost over capacity ratio of disks, which now (2015) stands at less than three cents per GB of storage.

Recent studies on large-scale storage installations have begun to shed light on the failure behavior of disks. In summary, disks in large installation fail at higher rates than the data sheet specifications. Historically, many researchers assumed that failure rates of disks follow a bathtub curve with high failure rates at the beginning (burn-in) and at the end (wear-out) of the economic lifespan, but this behavior is often not observed. Dependence on utilization and ambient temperature also appear to vary from installation to installation. Latent sector errors are an important source of data loss [1], [20], [21], [19]. Because latent sector errors cannot be ignored and because the observed failure rates are in general less than 10% per year, two-failure tolerance is sufficient for most disks arrays. A recent study by Beach showed however that these rates are average and that individual disk batches can evidence much higher failure rates. Beach reports a batch of 539 disks from a reputable manufacturer that failed at a rate of 25% per year [2].

Under these circumstances, higher levels of failure tolerance for disk arrays become important. In addition, "hardening" a disk array also makes sense [14]. Hardening is the process of adding additional parity information to an array in order to increase the level of failure tolerance. It constitutes a reaction to an increase in the observed disk failure rate beyond the worst failure rates used in the design of the array. Designing storage arrays for failure rates close to the worst case is rarely a good idea as it would result in needless complex and costly systems. Unfortunately, sometimes reality is worse than planned. If a batch of disks turns out to not be quite so trustworthy, hardening allows an administrator to use new parity data stored elsewhere in the data center to have the existing group of disks meet their reliability targets without having to move the data themselves.

We present a three failure-tolerant organization of a disk array using a single exclusive-or operation to calculate the parity in an individual reliability stripe. Our organization is based on the construction presented in previous work, [22], which uses the least number of parity disks and of data disks for a given ratio 2 to $(n-1)$ of parity to data disks. In this organization, each data disk is located in two different reliability stripes with $n-1$ data disks and one parity disk. The organization has $n(n-1)/2$ data disks and n parity disks for a total of $n(n+1)/2$ disks. We define and investigate an organization built on top of this organization. This new organization introduces additional reliability stripes of the same size as the already existing ones such that every data disk now belongs to three reliability stripes. This implies the organization of additional $n/2$ reliability stripes and the introduction of that many new parity drives. The construction is feasible whenever n is even and the resulting layout is three-failure tolerant. The additional protection can be constructed on need (hardening). The construction is based on a theorem in combinatorics by Lawless [12]. As the organization uses fixed parity disks, the organization is suited only for archival storage systems.

We present a visualization of disk array layouts where all data disks belong to two different reliability stripe (each with one parity drive containing the exclusive-or parity of the data

978-1-4673-9376-8/15 $31.00 © 2015 IEEE
DOI 10.1109/PRDC.2015.17

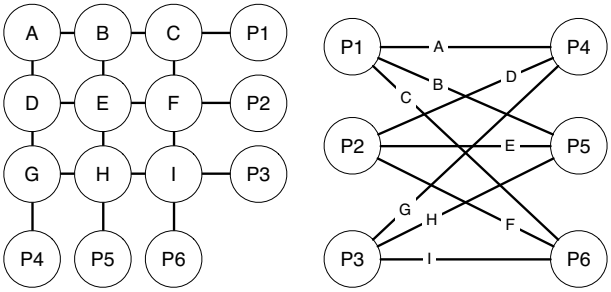

Fig. 1: Left: The two-dimensional layout for a disk array. A, B, ... I are data disks and $P1$, ..., $P6$ parity disks. Right: The corresponding graph visualization.

disks in the stripe), present Lawless construction of factoring a complete graph into paths, and define our three-failure tolerant layout. We then derive the resilience of the design against four and five disk failures. We then present a reliability analysis of the resulting design and show that the design is less likely to loose data than am equally dimensioned RAID Level 6 with three parity disks per stripe given the same number of failed disks. We also show that hardening the two-failure tolerant design results in equal or better reliability at three times a reasonable failure rate.

II. Construction

We consider disk arrays laid out by *flat XOR-codes* [6]. This means that all data disks are organized into reliability stripes with k data and one parity disk. The parity disk contains the exclusive-or of the data disks. Because we separate client data and parity data on different devices, the layout is only appropriate for archival workloads. While not necessary, we also assume that the size k of the reliability stripe is constant.

A. Two-failure tolerant disk array layouts and their relationship to mathematical graphs

In order to be two-failure tolerant, every disk needs to be in two different reliability stripes. Placing a data disk in three (or more) reliability stripes implies updating parity three (or more) times with each change to the disk's content. To limit this operational overhead, we therefore want to place each disk in exactly two reliability stripes. Reversely, two different reliability stripes can only intersect in at most one disk. (If two reliability stripes were to intersect in two disks A and B, then we can pick arbitrary disk contents X, change the contents of A to $A \oplus X$, the exclusive-or of A and X, change the contents of B to $B \oplus X$ without changing the contents of the parity drives of these two stripes because the changes cancel each other out. Therefore, these two stripes cannot possibly protect the contents of these two disks against failure.)

A consequence of these simple, often made observations is that each data disk is determined by the two reliability stripes to which it belongs. We can now represent the layout of such a two-failure tolerant disk array by labeling all reliability stripes with consecutive numbers 0, 1, ..., $n-1$, label the parity disks with the stripe to which they belong, and the data disks with the number of the two stripes to which the data disk belongs. Thus, the parity disks have labels in

$$V = \{0, 1, \ldots, n-1\} = \mathbb{Z}_n$$

(where $\mathbb{Z}_n$ as usual denotes the residue classes modulo n) and the data disks have labels in

$$E = \{(i, j) | i \neq j, i, j \in \mathbb{Z}_n\}.$$

Since an (undirected) graph is given exactly by two sets (V, E) of this form, it turns out that the layout of all disk arrays with two-failure tolerance defines and is defined by an undirected graph. To our knowledge, this was first exploited by Xu and colleagues in the definition of B-codes [26]. There, they used a coloring of the graph encompassing both vertices and edges and derived from a (graph-theoretical) factorization of an enlarged graph to define an array code by collocating data and parity information on a minimum number of disks. We use the same tool here for a different means.

As an example, we use the two-dimensional layout in Figure 1. Parity stripes are defined by the horizontal and vertical lines and each contain three data disks and one parity disk. A, B, ... I are data disks and P_1, ..., P_6 parity disks. Disk B is located in the two reliability stripes with parity drives P_1 and P_5 respectively. It is the only disk that can belong to these two stripes, and we give it the label (P_1, P_5) that characterizes this disk. If we proceed, we get A corresponds to (P_1, P_4), B to (P_1, P_5), C to (P_1, P_6), D to (P_2, P_4), E to (P_2, P_5), F to (P_2, P_6), G to (P_3, P_4), H to (P_3, P_5), and I to (P_3, P_6). The order within a label does not matter. The design of the disk array is given by defining the reliability stripes. We can encode the stripes now in the form of the graph on the left side of Figure 1. In this graph, the vertices correspond to the parity disks and the data disks to the edges. For instance, B has label (P_1, P_5) and corresponds to the edge between vertices P_1 and P_5. The reliability stripe with parity disk P_i is made up of the data disks with a label that contains P_i. For example, the reliability stripe with parity P_5 contains B corresponding to (P_1, P_5), E to (P_2, P_5), and H to (P_3, P_5). The move from disk array layout to a graph representation is in fact a frequent tool in combinatorical mathematics for block designs.

We observed in previous work that the two-failure tolerant disk array with the least number of data disks given a certain ratio of the number of parity disks over the number of data disks is obtained by letting this graph be the complete graph K_n with n disks. The n vertices of this graph are numbered from 0 to $n-1$ and represent the parity disks and the $n(n-1)/2$ edges are given by $(0,1)$, $(0,2)$, ..., $(0,n-1)$, $(1,2)$, $(1,3)$, ... $(n-2,n-1)$ and represent the data disks. Each stripe consists of $n-1$ data disks. We then calculated the resilience (mean time to data loss) of such an ensemble, which we call the *compact layout* and found that it compared favorably with that of an RAID Level 6 organization with an equal amount of data and parity disks [22]. The primary motivation to prefer the compact layout to the RAID Level 6 organization is the simpler calculation of parity and during reconstruction of failed disks. If there is a single failed disk, its contents are

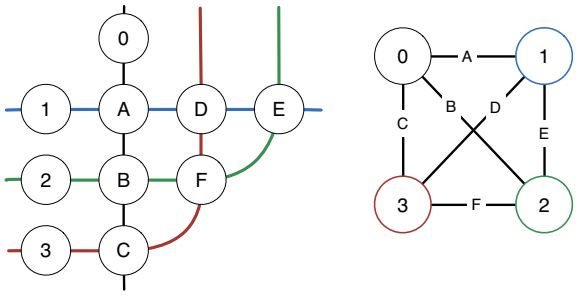

Fig. 2: Two representations of the same small, compact disk array layout with six data and four parity disks. On the left, the edges denote reliability stripes and the vertices all disks. On the right, the vertices denote parity disks and the edges data disks.

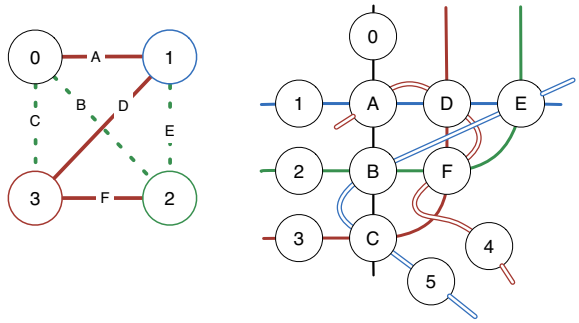

Fig. 3: Adding two reliability stripe to the disk array in Fig. 2. The left side shows the factoring of K_4 into two paths. The result on the right shows the disk array with six data disks and six parity disks. The "new" parity disks 4 and 5 are only implicitly represented in the graph through the factoring.

recovered by a single exclusive-or operation from the contents of the other disk in one of the two reliability stripes containing the failed disk and if there are several failed disks, each one will be recovered by such an operation though in an order restricted by the placement of the failed disks in the graph. Often, we can recover completely in parallel.

Figure 2 gives an example of the compact layout. On the left, the vertices denote all disks. Vertices A, B, C, D, E, and F represent data disks and vertices 0, 1, 2, and 3 represent parity disks. The edges represent membership in a reliability stripe. The right of Figure 2 gives the corresponding graph representation K_4. Data disk E lies in the reliability stripes 1 and 2 and corresponds to the edge $(1,2)$ between the vertices 1 and 2.

An observation by Zhou and colleagues characterizes minimal failure sets of disks as those containing either a cycle of edges or a path where the end vertices have also failed [27].

B. Three-failure tolerant disk array layouts and graphs

We want to arrange data disks in three reliability stripes providing triple-failure tolerance, but starting from a double-failure resilient layout (hardening). We start with the complete graph K_n and then organize all edges in groups of $n-1$ that define additional reliability stripes that consists of $n-1$ data disks and one additional parity disk. While there are many ways of forming the $n(n-1)/2$ edges in these groups, the resulting layout in general will not provide three tolerance. We provide here one layout where the groups are paths of $n-1$. We invoke a result of graph theory that is a variant of Kirkman's schoolgirl problem that stands at the beginning of the theory of combinatorial designs in Mathematics [11]. Whereas Kirkman's problem is posed as

> *Fifteen young ladies in a school walk out three abreast for seven days in succession: it is required to arrange them daily so that no two shall walk twice abreast.*

the variant has dangerous prisoners walking in groups to the prison yard several abreast, and each handcuffed to his neighbor. In less gruesome terms, and after translating from block designs to graphs, we divide the edges of the complete graph K_n into $n/2$ paths of $n-1$ edges each. As Hung and Mendelsohn already showed in 1974, this is always possible if n is even [9]. J. F. Lawless gave a general construction of handcuffed designs using difference sets [12].

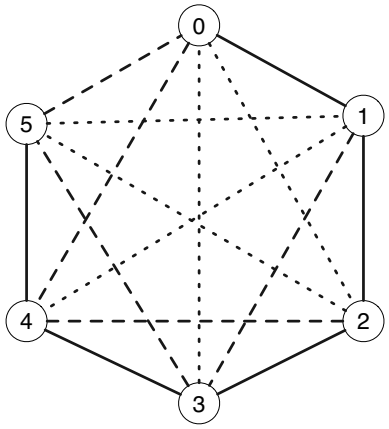

Fig. 4: Factoring K_6 into three paths of five edges without the Lawless construction. The graph corresponds to a disk array with 15 data and 6 parity disks, to which the factoring adds three new parity disks that are not visible in the graph layout.

Figure 3 shows on the left the factoring of K_4 into two paths of length three using Lawless' construction. We marked the edges in the first factor in red and the ones in the second factor in dashed green. The data disks represented by the edges in each path are organized into two new reliability stripes, namely A, D, F and E, B, C. Two new parity disks (4 and 5) are added to the ensemble. These are not depicted in the graph layout on the right of Figure 3, but their existence can be deduced from the factoring.

We give another example of the resulting layout in Figure 4 where we used different strokes to assign all edges of the complete graph K_6 with 6 vertices to three different paths. These paths represents three new reliability stripes. We continue to label data disks with the reliability stripes to which they belong, so that disk (i,j) is the data disk that belongs to the two reliability stripes with parity disks i and j. We recall that in the compact layout, there is always such a data disk.

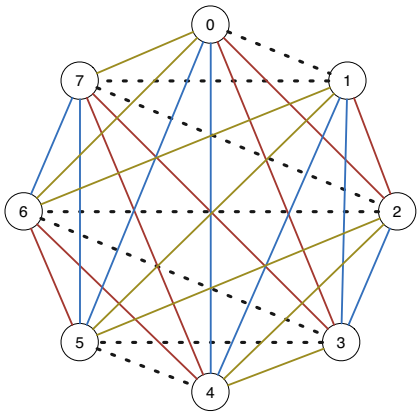

Fig. 5: Factoring K_8 into four paths of six edges with the Lawless construction.

We then have the six reliability stripes defined by the graph. They are

$$0,(0,1),(0,2),(0,3),(0,4),(0,5)$$
$$1,(0,1),(1,2),(1,3),(1,4),(1,5)$$
$$2,(0,2),(1,2),(2,3),(2,4),(2,5)$$
$$3,(0,3),(1,3),(2,3),(3,4),(3,5)$$
$$4,(0,4),(1,4),(2,4),(3,4),(4,5)$$
$$5,(0,5),(1,5),(2,5),(3,5),(4,5)$$

Obviously, each stripe corresponds to a vertex and the adjacent edges to the vertex. In addition to these "old" reliability stripes, there are three new ones defined by the three paths depicted in Figure 4. The path given as a contiguous stroke starts at vertex 0 and passes then through 1, 2, 3, 4 to terminate at vertex 5. The first new reliability stripe therefore consists of the data disks that lay in reliability stripes with parity disks i and $i+1$, $0 \le i \le 5$. The other two graph factors determine the remaining two "new" reliability stripes. The result is

$$(0,1),(1,2),(2,3),(3,4),(4,5)$$
$$(1,3),(3,5),(0,5),(0,4),(2,4)$$
$$(1,4),(1,5),(2,5),(0,2),(0,3)$$

All 15 data disks (i,j) $(0 \le i < j \le 5$ have now been placed into three additional reliability stripes. Vertex 0 corresponds to the parity disk of the first "old" reliability stripe, Vertex 1 to the second "old" reliability stripe, etc. as becomes clear by observing that this is the common label of all the data disks in the stripe. The parity disks for the new stripes corresponding to the path are not depicted in the graph.

For a slightly more complicated example, we turn to the Lawless' factorization of K_8 in Figure 5. The "old" reliability stripes consist of the data disks with a common component in the label, thus we have the old reliability stripes

$$0:(0,1),(0,2),(0,3),(0,4),(0,5),(0,6),(0,7)$$
$$1:(0,1),(1,2),(1,3),(1,4),(1,5),(1,6),(1,7)$$
$$2:(0,2),(1,2),(2,3),(2,4),(2,5),(2,6),(2,7)$$
$$3:(0,3),(1,3),(2,3),(3,4),(3,5),(3,6),(3,7)$$
$$4:(0,4),(1,4),(2,4),(3,4),(4,5),(4,6),(4,7)$$
$$5:(0,5),(1,5),(2,5),(3,5),(4,5),(5,6),(5,7)$$
$$6:(0,6),(1,6),(2,6),(3,6),(4,6),(5,6),(6,7)$$
$$7:(0,7),(1,7),(2,7),(3,7),(4,7),(5,7),(6,7)$$

The new reliability stripe depicted as a dotted black line is

$$(0,1),(1,7),(2,7),(2,6),(3,6),(3,5),(4,5)$$

the red, blue, and yellow ones are

$$(1,2),(0,2),(0,3),(3,7),(4,7),(4,6),(5,6)$$
$$(2,3),(1,3),(1,4),(0,4),(0,5),(5,7),(6,7)$$
$$(3,4),(2,4),(2,5),(1,5),(1,6),(0,6),(0,7)$$

As before, the parity disks for the new stripes have no counterpart in the graph visualization. The data disks in the red stripe are obtained by adding one modulo 8 to the labels in the black one and the data disks in the blue and yellow stripe by adding two and three, respectively.

C. A general construction for K_{2m}

In the special case of interest to us, the construction by Lawless is extremely efficient [12]. It embeds the vertex numbers into $\mathbb{Z}_n$, the integers modulo $n = 2m$, and uses a simple difference set to define the paths. The first path is given by the vertices

$$\{0, 0+1, 0+1-2, 0+1-2+3, \ldots, 0+1-2+3\ldots+(n-1)\},$$

i.e. has *forward differences* 1, -2, 3, $\ldots$ $(n-1)$. The data disks therefore have labels $(0,1)$, $(1,-1 \pmod n)$, etc. The remaining paths are given by adding an element of $\mathbb{Z}_n$ to all vertex numbers in the path. In the graph, the paths follow a zig-zag pattern, that is then rotated $m-1$ times to yield m paths. We apply this construction in Figure 5, where we use dotted lines to give the first path. The other paths are obtained by adding 1, 2, and 3 to the vertex numbers in the first path. We refer to these paths as the *Lawless' paths*. Lawless' construction has the advantage of being intrinsically symmetric which simplifies arguments over failure tolerance.

For the determination of minimal failure patterns, it is useful to notice the relationship between vertices on a Lawless' path that are at distance two of each other. Since the forward difference between these vertex pairs are either $-k+(k+1)$ or $k-(k+1)$, the difference in the vertex numbers is 1. Thus, the vertex pairs lie on the "periphery" of the complete graph, formed by the edges $(i,i+1)$, $i \in \mathbb{Z}_n$. Since there are $n-2$ vertex pairs of distance two in a Lawless path and since there are n edges in the periphery, these vertex pairs make up all the edges in the periphery with the exception of two, namely

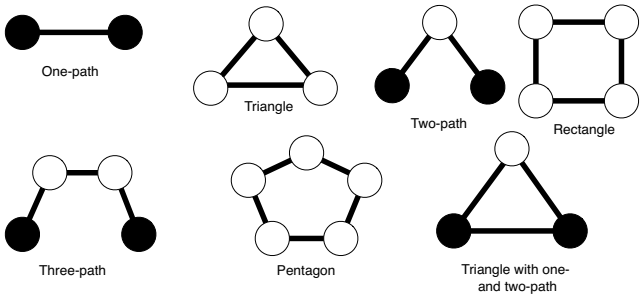

Fig. 6: Failure pattern definitions. Black nodes and edges represent failed elements [22].

$(a, a+1)$ and $(a+\frac{n}{2}, a+\frac{n}{2}+1)$.

III. Minimal Failure Patterns

In the following, we make free use of previous results [22] in order to calculate the data loss probabilities exactly for up to five failures. Every set of failed parity or data disks represents a *failure pattern* in the graph. We call the reliability stripes captured in the graph visualization *old* stripes and the ones corresponding to the paths in the graph the *new* stripes and correspondingly of *old* and *new* parity drives. Thus, only the old parity drives are represented as vertices. If we assume that all new parity drives have failed, then we can show that a failure pattern has to contain a path (two failed vertices and a path between them) or a cycle.

A. Minimal failure patterns of size 3 do not exist

We have two types of three-failure patterns in the compact layout, namely the one-path and the triangle, see Figure 6. The one-path corresponds to a failed data disks together with the failed old parity disks in the two reliability stripes to which the data disk belongs. We can recover the contents of the failed data disks from the new stripe to which the data disk belongs unless the corresponding new parity drive has also failed, but then we have four failures. A three-cycle represents a set of three failed data disks such that each two of the three lay in an old reliability stripe. As the new stripes consists of paths, at most two of the three edges (data disks) of the cycle can be in a single new reliability stripe. Therefore, we can recover first the contents of at least one of the three failed data disks and then the contents of the other two failed data drive. The new (hardened) data layout is therefore three-failure tolerant.

B. Minimal failure patterns of size 4

The design without new reliability stripes has two types of minimal four-failure patterns, namely the two-path consisting of a failed (old) parity disk, an adjoining data disk, a data disk adjoining to this data disk and the adjoined (old) parity disk, and a quadrangle, consisting of four failed data disks (Figure 6). We can recover the data in a failed two-path if the two edges (data disks) are in two different new stripes, but not, if they are in the same new stripe. To count this pattern, we need to count the number of ways in which we can select two adjacent edges in a new stripe, since every such patterns forms a failure pattern. Since the new stripes are formed by a path with $n-1$ edges, there are $n-3$ subpaths of length 2. As there are $n/2$ new stripes, we obtain $(n-2)n/2$ failure patterns of this type.

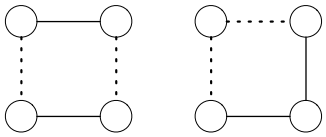

Fig. 7: Two types of minimal four-failure patterns for quadrangles.

TABLE I: Number of quadrangles of type 2.

n	Fl.	n	Fl.	n	Fl.	n	Fl.	n	Fl.
6	3	26	78	46	253	66	528	86	903
8	14	28	259	48	804	68	1649	88	2794
10	10	30	105	50	300	70	595	90	990
12	39	32	344	52	949	72	1854	92	3059
14	21	34	136	54	351	74	666	94	1081
16	76	36	441	56	1106	76	2071	96	3336
18	36	38	171	58	406	78	741	98	1176
20	125	40	550	60	1275	80	2300	100	3625
22	55	42	210	62	465	82	820		
24	186	44	671	64	1456	84	2541		

A quadrangle cannot be reconstructed using the new reliability stripes if and only if every edge in the quadrangle lies in the same new stripe as another edge. This gives us two types of four-failure patterns based on quadrangles, depending on whether opposing or adjacent edges belong to the same Lawless' paths (Figure 7).

We can count the quadrangles of the latter type by using the subpaths of the Lawless' paths of length 2, which corresponds to pairs of vertices at distance two in the path. We select one such subpath from one Lawless' path and then another one from another one. As we have seen, these subpaths form two thirds of a triangle with the other edge being an edge on the periphery. For example, the Lawless' path in K_8 starting at 3 has a subpath of length 2 of $(4,2),(2,5)$ forming a triangle $2,4,5$ and the other edge is edge $(4,5)$ on the periphery. If we select two Lawless' paths, then in order to form a quadrangle of subpaths of length two of these, we just select one such peripheral edge. For example, if in addition to the Lawless' path from 3 to 7 we pick the Lawless path from 1 to 5 in K_8, then 4 and 5 are also the endpoints of a subpath $(4,6),(6,5)$ on that path. This gives us a quadrangle with vertices $[4,6,5,2]$ (in that order). Given two Lawless' paths, they each have $n-2$ of the n peripheral edges and they have $n-4$ in common. Therefore, we count by selection two of the $n/2$ Lawless paths, and then a common peripheral edge to obtain a total of these type of quadrangles of

$$(n-4)\binom{\frac{n}{2}}{2}$$

We were not able to derive a closed-form formula for the number of quadrangles of the second type, where the two edges belonging to the same Lawless' path are in parallel and not adjoint. However, we could easily write a Python program to evaluate all graphs for n even between 6 and 100. The result

TABLE II: Number of pentagons with edges in two Lawless' path.

n	Fl.	n	Fl.	n	Fl.	n	Fl.	n	Fl.
6	24	22	792	38	2584	54	5400	70	9240
8	56	24	936	40	2840	56	5768	72	9720
10	120	26	1144	42	3192	58	6264	74	10360
12	180	28	1316	44	3476	60	6660	76	10868
14	280	30	1560	46	3864	62	7192	78	11544
16	368	32	1760	48	4176	64	7616		
18	504	34	2040	50	4600	66	8184		
20	620	36	2268	52	4940	68	8636		

(Table I shows a strong dependence on the remainder class of n modulo 4.

Finally, we can have a three minimal failure pattern in the old design with the additional failure of one new parity disk. The one-path has only one data disk and if that data disk belongs to a new stripe where the parity disk has failed, then data reconstruction is impossible using the old or the new parity information. There are $\binom{n}{2}$ of these one-paths, each constituting a different minimal four-failure pattern.

The other possibility is the triangle. In order for the data on the three mutually interconnected data disks to be not reconstructible, we need two of them to be in a new reliability stripe (which then has lost two of its members) and one in a new reliability stripe where the new parity disk also has failed. We recall that a Lawless' path cannot contain a cycle, so that this is the only possibility. Now, this pattern is uniquely determined by the two data disks in the same new reliability stripe, which have to be connected. They represent therefore a subpath of length 2 in a Lawless' path of length $n-1$, of which there are $n-2$ per Lawless path for a total of $\frac{n}{2}(n-2)$.

C. Minimal failure patterns of size 5

There are two minimal failure patterns of size 5 in the compact layout without the new reliability stripes. These are the pentagons and the two path. The other failure patterns of size 5 are constituted by a failure of a new parity disk and a minimal four-failure pattern in the old array, or a failure of two new parity disks and a minimal three failure pattern.

We first consider the pentagon. It consists of five data disks such that any two intersect in an old reliability stripe. With the new stripes, reconstruction is impossible if neither one of the failed data disks lie in a Lawless' path. Since a Lawless' path does not include cycles, this gives us a situation where two of the failed data disks lie in one and three of the failed lie in another Lawless' path. It was beyond our Mathematical capabilities to derive a closed form formula and so we used a Python program to count them (Table II).

We now consider the three-path. It consists of three data disks D_1, D_2, and D_3 such that D_1 and D_2 as well as D_2 and D_3 intersect each in a different, old reliability stripe and that the other old parity stripe of D_1 and of D_3 has suffered failure. In order to not be able to reconstruct the data on the data disks, each needs to be collocated with at least one more of the failed data disks in a new reliability stripe. Since there are only three of them, they need to be in the same Lawless' path. Since they also form a path of length three and a Lawless' path has length $n-1$, there are $n-3$ of them per Lawless' path for a total of $\frac{n}{2}(n-3)$.

We now consider the minimal four failures with the old reliability stripes in conjunction with the failure of the parity disk in a new stripe. The two-path contains two failed data disks. As we have seen, they cannot be recovered if both of them are in the same new reliability stripe. If this is not the case, then they belong to two different new stripes. Since only one of these new stripes can have lost the parity disk, we can use the other to reconstruct the data and then we no longer have a failure pattern. Thus, this pattern does not give rise to a new minimal failure pattern.

Next, we have the quadrangle, consisting of four data disks D_1, D_2, D_3, D_4 such that D_1 and D_2, D_2 and D_3, D_3 and D_4, and D_4 and D_1 each intersect in a separate old reliability stripe. As we have seen, if two of the data disks belong to the same new reliability stripe and the two other ones to another new reliability stripe, then they constitute a minimal four-failure pattern. Since with the additional failure of the parity disk of a new reliability stripe, still the contents of the lost four data disks have to remain not reconstructable, we only gain one more pattern, where we have three data disks in one new reliability stripe and the other data disk in another one, and where the parity drive of the latter has also failed. In the graph visualization, the quadrangle then has three sides in the same Lawless' path and the other one in a different path. Since any subpath of length three in a Lawless path has the end vertices connected by necessity with an edge in a different Lawless path, the number of these patterns is equal to set of three paths in a Lawless path, of which there are $\frac{n}{2}(n-3)$.

Finally, we consider minimal three failure patterns in the old design with two new parity disks also having failed. For the triangle, we can have all three data disks belong to different new reliability stripes, but in this case, at least one of the new reliability stripes has the parity intact and the data on the lost disk in it can be recovered. Therefore, at least two data disks need to belong to the same new parity stripe, and the other one needs to lie in a new reliability with failed parity disk, which is a pattern already counted as a minimal four-failure pattern.

If we have the one-path, then there is only one data disk, so that the failure of the parity of a new stripe in addition to that of the stripe, to which the data disk belongs, does not make a difference. Hence, these neither constitute minimal five-failure patterns.

D. Five failure patterns

Whereas all four failure patterns are minimal, some five failure patterns consist of a (by necessity) minimal four-failure pattern and an arbitrary failure. When accounting for these patterns, one also has to take double and triple counts into account as happens when two minimal four-failure patterns with a different additional, failed disk makes up the same five failure pattern. The closest we come is the triangle with one and two path (Figure 6), which in the layout without the new reliability stripes would be counted as a triangle, a

one-path, and a two-path. However, the triangle pattern is a failure pattern only if two of the data disks (edges) belong to a new stripe and the other edge to a new stripe with failed parity drive. This means six and not five failed disks. We can conclude after inspecting all other combinations of minimal four failure patterns that there are no patterns that are over-counted.

Therefore, a five failure pattern consists either of a minimal five failure pattern or of a minimal four failure pattern with one additional, but arbitrary disk failure. Since there are $n(n+2)/2$ disks in total, there are $n(n+2)/2$ possibilities to choose the additional disk to make up the pattern.

IV. Comparisons

A. RAID Level 6 organization with three parity disks per stripe

We compare against a RAID Level 6 organization that is three-failure tolerant. This organization consists of n reliability stripes with k data and three parity disks. The total number of disks is $n(k+3)$ and of data disks is nk. A failure pattern for this organization has lead to dataloss if there are four or more disks in a stripe that have failed. To obtain a formula for the number of patterns with f failed disks where there is dataloss, we determine the number of patterns without dataloss. If there are then a_1 stripes with one failed disk, a_2 stripes with two failed disks, a_3 with three failed disks and $n-a_1-a_2-a_3$ stripes without failed disk, then we count

$$\binom{n(k+3)}{f-a_1-a_2-a_3,a_1,a_2,a_3}\binom{k+3}{1}^{a_1}\binom{k+3}{2}^{a_2}\binom{k+3}{3}^{a_3}$$

failure patterns. The number of failed disks in this pattern is $a_1+2a_2+3a_3=f$. By summing up over all possibilities, this gives us an effective way of calculating the number of failure pattern that do not lead to dataloss:

$$\sum_{a_1+2a_2+3a_3=f}\binom{n(k+3)}{f-a_1-a_2-a_3,a_1,a_2,a_3}\binom{k+3}{1}^{a_1}\binom{k+3}{2}^{a_2}\binom{k+3}{3}^{a_3}$$

B. Robustness evaluation

For the reliability evaluation, we used simulation to obtain the survival probabilities for the data in a disk array organizations in the presence of f failures. We present the results in Figures 8 and 9. We divided the simulations into 20 batches of 200000 simulations each in order to obtain 99% confidence intervals. The resulting intervals are at worst at 0.2 per mil of the obtained probabilities, too small for visible error bars in Figures 8 and 9.

As can be seen, data loss is guaranteed with the same number of failures for both types of organizations. Since both RAID Level 6 and the (hardened or not hardened) compact layout guarantee no data loss with three or two failures, the important results are for numbers f of failures between these two values. We can observe that the hardened or not hardened compact layout always has a lesser probability of data loss than the RAID Level 6 organization. We also observe that the hardened versions show very good data survival probabilities for small numbers of failures.

C. Evaluation with Markov models

We give the standard Markov model for a disk array layout in Figure 10. The state of the system is given by the number of failed disks in the system. The model is parametrized by the failure rate λ (assumed to be constant) and the repair rate ρ also assumed to be constant. The transitions represents disk failure and disk repair. A failure transition goes from State k (with k failed disks) to State $k+1$ with transition rate $p[k](N-k)\lambda$ where $p[k]$ is the probability that the failure of the $k+1^{\text{st}}$ disk does not lead to dataloss, or goes from State k to State F (the absorbing state representing failure) with rate $(1-p[k])(N-k)\lambda$. The repair transitions are from State $k+1$ to State k with rate $(k+1)\rho$.

If an organization suffers dataloss with probability a after failure of k disks and with probability $b>a$ after failure of $k+1$ disks, then the probability that the system is alive after $k+1$ failures is equal to $1-b$, but also to the probability that no dataloss is caused by the $(k+1)^{\text{st}}$ failure if none was caused by the k^{th} one, which is $p[k](1-a)$. Therefore, $p[k]=(1-b)/(1-a)$.

For the RAID Level 6 organization, Section IV.A gives the exact values of $p[i]$. For the graph layout, we obtained $p[3]$, $p[4]$, and $p[5]$ exactly in Section IV. The exactness of the remaining values for $p[i]$ are less important, and we determined them by simulation as reported in the previous sections. We then used this Markov model to calculate the one-year survival probability of the data in a disk array using an average repair time of 36 hours and varied the mean time to failure of disks. While the model is not completely accurate, for example, repair times are definitely not exponentially distributed, and while we neglect the effect of latent sector errors, whose consideration would make the article too long, we consider this modeling accurate enough for comparisons between the various designs.

We give the results in Figure 11, where we measure the one-year reliability of the array in terms of nines. A survival rate of 99.9% corresponds to three nines and one of 99.999% to five nines. Of course, these numbers do not include other causes of failures such as operator error or fire. The numbers shows clearly that the addition of a third reliability stripe in which a data disk is placed, more than compensates the effects of doubling the disk failure rate. For examples, we provide Tables III and IV. An annual failure rate of 5% corresponds to a mean time to failure of 166,688 hours. At this rate, the complete layout with 10 parity disks and 45 data disks has an annual data loss probability (with mean repair times of 36 hours) of 6.29×10^{-7}. If however the disk mean time to failure turns out to be only 50,000 hours, than the annual data loss probability increases to 2.34×10^{-5} (Table III). The layout with 5 additional parity disks however has an annual loss rate of 4.89×10^{-8}, which is actually better than that for the double-failure tolerant array. If we move to a larger layout with 120 data disks and 16 parity disks, with disks at a mean time to failure of 166,688 hours, the one year data loss rate is 2.06×10^{-5}. If we have disks with one tenth of this failure

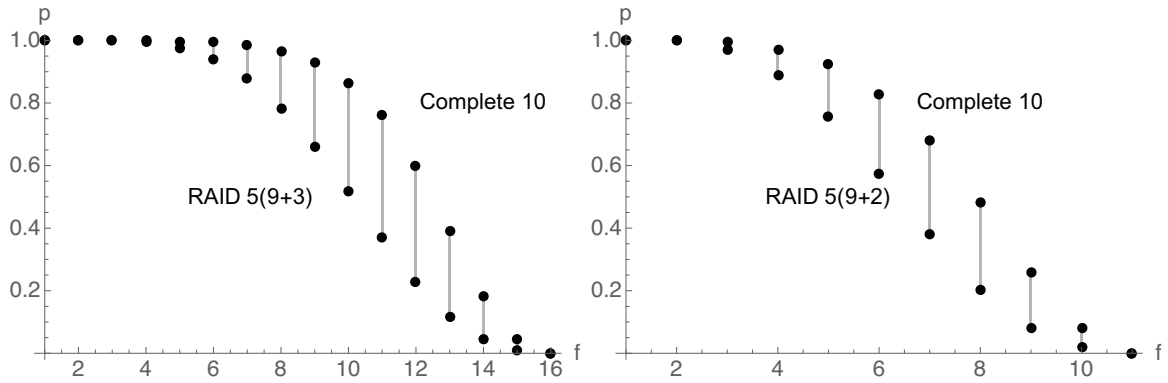

Fig. 8: Probability p of data survival (y-axis) in the presence of f failed disks (x-axis). The left graph gives the numbers for the triple failure resilient graph layout based on K_{10} (top) and for a RAID Level 6 layout with 5 stripes of 9 data disks and 3 parity disks. The right graph depicts the results for the two failure-tolerant base cases.

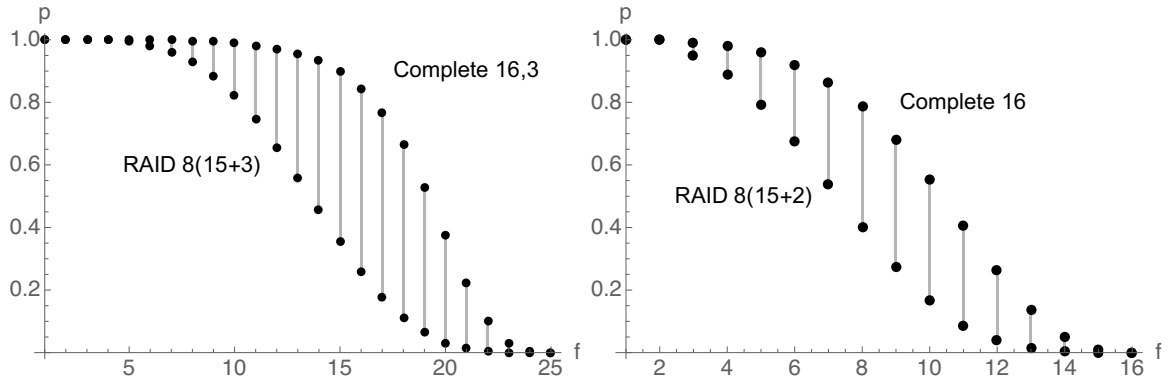

Fig. 9: Probability p of data survival (y-axis) in the presence of f failed disks (x-axis). The left graph gives the numbers for the triple failure resilient graph layout based on K_{16} (top) and for a RAID Level 6 layout with 8 stripes of 15 data disks and 3 parity disks. The right graph depicts the results for the two failure-tolerant base cases.

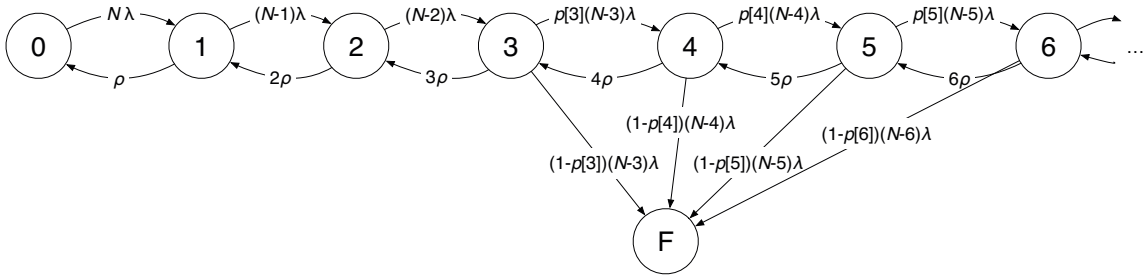

Fig. 10: Markov model for a three-failure tolerant disk array layout.

rate, the data loss rate increases to 1.83×10^{-2}. However, the hardened array with the same mean time to failure of 16,669 hours has a one year data loss rate of 1.52×10^{-5}, which is even a bit less than the one for the original rate with the original storage array. Perusing Tables III and IV shows that this observation holds for the range of values.

V. Related Work

The default storage policy for cloud file systems is triplication, as for example in the Google File System or Hadoop File System [4], [24]. For Windows Azure Storage, erasure coding is now considered as an replacement for triplication; the proposed linear reconstruction code provides triple failure tolerance, but sacrifices being MDS in order to read from less drives for most recovery operations [7].

Block codes offering three failure tolerance are not new. Generalized Reed-Solomon codes are triple-failure tolerant MDS codes, but involve Galois field calculations in their construction. Feng and colleagues present a family of more efficient triple-failure tolerant MDS codes as does Huang and Xu [3], [8]. Current research is now interested in improving the efficiency of such codes during recovery [10]. Wang

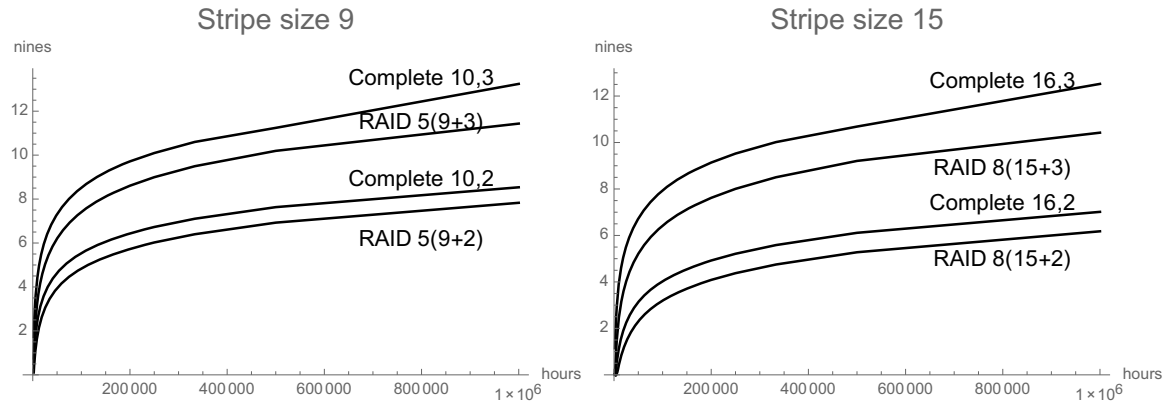

Fig. 11: Comparison of the probabilities for one year survival of all data. The resulting probability is given in number of nines. The x-axis gives the average mean time to failure of a disk.

TABLE III: One year failure probability for the double- and tripe-failure tolerant layouts based on the complete graph K_{10} and the RAID VI with 5 stripes with 9 data disks and two or three parity disks.

MTtF	2-failure tlr.	3-failure tlr.	2-failure tlr.	3-failure tlr.
	Graph		RAID VI	
10000	2.95×10^{-3}	3.04×10^{-5}	1.41×10^{-2}	3.77×10^{-4}
20000	3.66×10^{-4}	1.91×10^{-6}	1.81×10^{-3}	2.41×10^{-5}
30000	1.08×10^{-4}	3.77×10^{-7}	5.41×10^{-4}	4.80×10^{-6}
40000	4.57×10^{-5}	1.19×10^{-7}	2.29×10^{-4}	1.52×10^{-6}
50000	2.34×10^{-5}	4.89×10^{-8}	1.17×10^{-4}	6.26×10^{-7}
60000	1.35×10^{-5}	2.36×10^{-8}	6.81×10^{-5}	3.02×10^{-7}
70000	8.50×10^{-6}	1.27×10^{-8}	4.29×10^{-5}	1.63×10^{-7}
80000	5.70×10^{-6}	7.46×10^{-9}	2.88×10^{-5}	9.58×10^{-8}
90000	4.00×10^{-6}	4.66×10^{-9}	2.02×10^{-5}	5.99×10^{-8}
100000	2.92×10^{-6}	3.06×10^{-9}	1.47×10^{-5}	3.93×10^{-8}
200000	3.64×10^{-7}	1.92×10^{-10}	1.85×10^{-6}	2.46×10^{-9}
300000	1.08×10^{-7}	3.89×10^{-11}	5.47×10^{-7}	4.87×10^{-10}
400000	4.55×10^{-8}	1.19×10^{-11}	2.31×10^{-7}	1.54×10^{-10}
500000	2.33×10^{-8}	5.75×10^{-12}	1.18×10^{-7}	6.40×10^{-11}
600000	1.35×10^{-8}	3.50×10^{-12}	6.84×10^{-8}	3.16×10^{-11}
700000	8.48×10^{-9}	3.17×10^{-12}	4.31×10^{-8}	1.84×10^{-11}
800000	5.68×10^{-9}	1.06×10^{-12}	2.89×10^{-8}	9.89×10^{-12}
900000	3.99×10^{-9}	1.22×10^{-12}	2.03×10^{-8}	6.75×10^{-12}
1000000	2.91×10^{-9}	5.73×10^{-14}	1.48×10^{-8}	3.67×10^{-12}

TABLE IV: One year failure probability for the double- and tripe-failure tolerant layouts based on the complete graph K_{16} and the RAID VI with 8 stripes with 15 data disks and two or three parity disks.

MTtF	2-failure tlr.	3-failure tlr.	2-failure tlr.	3-failure tlr.
	Graph		RAID VI	
10000	7.63×10^{-2}	1.17×10^{-4}	3.94×10^{-1}	3.62×10^{-3}
20000	1.08×10^{-2}	7.32×10^{-6}	6.94×10^{-2}	2.34×10^{-4}
30000	3.33×10^{-3}	1.44×10^{-6}	2.21×10^{-2}	4.68×10^{-5}
40000	1.43×10^{-3}	4.57×10^{-7}	9.62×10^{-3}	1.49×10^{-5}
50000	7.41×10^{-4}	1.87×10^{-7}	5.01×10^{-3}	6.11×10^{-6}
60000	4.32×10^{-4}	9.03×10^{-8}	2.93×10^{-3}	2.95×10^{-6}
70000	2.74×10^{-4}	4.87×10^{-8}	1.86×10^{-3}	1.6×10^{-6}
80000	1.84×10^{-4}	2.86×10^{-8}	1.25×10^{-3}	9.37×10^{-7}
90000	1.3×10^{-4}	1.78×10^{-8}	8.83×10^{-4}	5.86×10^{-7}
100000	9.47×10^{-5}	1.17×10^{-8}	6.46×10^{-4}	3.85×10^{-7}
200000	1.2×10^{-5}	7.3×10^{-10}	8.2×10^{-5}	2.41×10^{-8}
300000	3.56×10^{-6}	1.44×10^{-10}	2.44×10^{-5}	4.77×10^{-9}
400000	1.5×10^{-6}	4.56×10^{-11}	1.03×10^{-5}	1.51×10^{-9}
500000	7.71×10^{-7}	2.06×10^{-11}	5.29×10^{-6}	6.2×10^{-10}
600000	4.47×10^{-7}	8.76×10^{-12}	3.07×10^{-6}	2.98×10^{-10}
700000	2.81×10^{-7}	5.04×10^{-12}	1.93×10^{-6}	1.61×10^{-10}
800000	1.89×10^{-7}	3.31×10^{-12}	1.3×10^{-6}	9.48×10^{-11}
900000	1.32×10^{-7}	1.83×10^{-12}	9.1×10^{-7}	5.9×10^{-11}
1000000	9.66×10^{-8}	$3.\times10^{-13}$	6.64×10^{-7}	3.78×10^{-11}

and colleagues use combinatorial techniques (including graph factorizations, but of a different kind than we use) to define t-erasure correcting MDS codes with a good choice of code-parameters [25].

Greenan *et al.* introduced the notion of *flat XOR codes* to describe the type of code that we presented here, where data disks are stored in various reliability stripes with a single parity disk that contains the exclusive-or of the data disks in the stripe [5], [6]. They describe various constructions for these codes with the help of Tanner codes [23]. They also calculate minimal erasure lists, an enumeration of what we call minimal failure patterns, and the minimal erasure vector, which encodes the dataloss probability given a certain number of failures. The focus of this work is finding good codes for small arrays. For very large sizes of the code work, Low Density Parity Check (LDPC) or Gallager codes [13] have been shown to possess excellent encode and decode costs for data over lossy channels at the cost of sacrificing a small amount of space-efficiency, which is the reason that they have been adapted in several standards such as for the WiFi 802.11n and 802.11ac standard and 10GBase-T ethernet. They are defined by a sparse parity check matrix, as is the case for the types of codes considered in this paper. Plank's work bridges the gap between the large sizes needed for network encoding and the smaller codes appropriate for disk array design [18]. Pâris *et al.* investigate the reliability of three-dimensional RAID arrays that provide three-failure tolerance [15].

In previous work, we proposed to equip a fairly conventional two-dimensional RAID architecture, where each disk belongs to exactly one horizontal and one vertical RAID Level 4 stripe, with a super-parity containing the parity of all parity disks to obtain a three-failure tolerant array with a reliability that compares favorably to an equivalent set of RAID Level 6 stripes [16]. We also proposed to protect against early disk failure by

using free space in a disk array for parity information until the disks are burnt in [17]. We first investigated hardening a disk array in the case of the two-dimensional disk array, where we mirror the parity in one dimension on need [14].

VI. Conclusion

We presented here a layout for a tripe-failure tolerant disk array design that uses a single exclusive-or operation to calculate a parity and equally a single exclusive-or operation to reconstruct the data on a failed drive. The layout can be obtained by adding reliability stripes to a disk array design with the same operational properties, but with only tolerance for double failures. One mode of use for the new design is to harden the compact layout with two-failure tolerance to obtain one with three-failure tolerance. We have shown that hardening tolerates an unexpected decrease in the mean time to failure of a disk array by about a factor of 10. We also showed that the design is substantially more resilient to data loss from disk failures than a RAID Level 6 design with the same number of parity disks and the same size of reliability stripes. In our eyes, this makes this design attractive as the base of organizing disk arrays in a large, archival storage system with thousands of disks, in spite of lacking the flexibility that a RAID Level 6 offers for changing the number of disks by small amounts.

As future work, we want to consider two extensions. First, the compact layout is not the only two failure-tolerant layout based on a flat XOR code, even though it is attractive because of its density. Second, current work in disk array layout is interested in lowering the I/O costs of typical reconstructions. In fact, our triple-failure tolerant layout can fall in line by halving the size of the new reliability stripes. Another promising line of constructions starts with (r,s)-regular graphs where one subset of vertices has degree r and its complement degree s. The techniques used in this paper, extensive simulation to determine failure tolerance given f failures together with Markov modeling for reliability determination allow the study of these layouts.

VII. Acknowledgments

D. D. E. L. was supported in part by the National Science Foundation under awards CCF-1219163 and CCF-1217648, by the Department of Energy under award DE-FC02-10ER26017/DE-SC0005417 and by the industrial members of the Storage Systems Research Center.

References

[1] L. N. Bairavasundaram, G. R. Goodson, S. Pasupathy, and J. Schindler, "An analysis of latent sector errors in disk drives," in *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, 2007, pp. 289–300.

[2] B. Beach. (2014, January) What harddrive should I buy? [Online]. Available: http://blog.backblaze.com/2014/01/21/

[3] G.-L. Feng, R. H. Deng, F. Bao, and J.-C. Shen, "New efficient MDS array codes for RAID. Part I. Reed-Solomon-like codes for tolerating three disk failures," *Computers, IEEE Transactions on*, vol. 54, no. 9, pp. 1071–1080, 2005.

[4] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *ACM SIGOPS Operating Systems Review*, vol. 37(5), 2003, pp. 29–43.

[5] K. M. Greenan, X. Li, and J. J. Wylie, "Flat XOR-based erasure codes in storage systems: Constructions, efficient recovery, and tradeoffs," in *IEEE 26th Symp. on Mass Storage Systems and Technologies*. IEEE, 2010.

[6] K. M. Greenan, E. L. Miller, and J. J. Wylie, "Reliability of flat XOR-based erasure codes on heterogeneous devices," in *IEEE Int. Conf. Dependable Systems and Networks*. IEEE, 2008, pp. 147–156.

[7] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, S. Yekhanin *et al.*, "Erasure coding in windows azure storage." in *USENIX Annual Technical Conference*, 2012, pp. 15–26.

[8] C. Huang and L. Xu, "Star: An efficient coding scheme for correcting triple storage node failures," *IEEE Transactions on Computers*, vol. 57(7), pp. 889–901, 2008.

[9] S. H. Hung and N. Mendelsohn, "Handcuffed designs," *Aequationes Mathematicae*, vol. 11, no. 2-3, pp. 256–266, 1974.

[10] O. Khan, R. Burns, J. Plank, and C. Huang, "In search of i/o-optimal recovery from disk failures," in *3rd USENIX conference on Hot topics in storage and file systems*, 2011.

[11] T. P. Kirkman, "On a problem in combinations," *Cambridge and Dublin Math. J*, vol. 2, pp. 191–204, 1847.

[12] J. Lawless, "On the construction of handcuffed designs," *Journal of Combinatorial Theory, Series A*, vol. 16(2), pp. 76–86, 1974.

[13] M. G. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, "Practical loss-resilient codes," in *Proceedings of the 29th annual ACM symposium on Theory of computing*. ACM, 1997, pp. 150–159.

[14] J.-F. Pâris, T. J. Schwarz, A. Amer, and D. Long, "Protecting RAID arrays against unexpectedly high disk failure rates," in *Proceedings, 20th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2014.

[15] J.-F. Pâris, D. D. Long, and W. Litwin, "Three-dimensional redundancy codes for archival storage," in *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2013 IEEE 21st International Symposium on*. IEEE, 2013, pp. 328–332.

[16] J.-F. Pâris, S. Schwarz, A. Amer, and D. D. Long, "Highly reliable two-dimensional raid arrays for archival storage," in *IEEE 31st International Performance Computing and Communications Conference (IPCCC)*, 2012, pp. 324–331.

[17] J.-F. Pâris, T. J. Schwarz, and D. D. Long, "Protecting data against early disk failures," in *Fifth Intl. Information and Telecommunication Technologies Symposium (I2TS) Cuiabá, MT, Brazil*, 2006.

[18] J. S. Plank, M. Thomason *et al.*, "A practical analysis of low-density parity-check erasure codes for wide-area storage applications," in *IEEE Int. Conf. Dependable Systems and Networks*, 2004, pp. 115–124.

[19] B. Schroeder, S. Damouras, and P. Gill, "Understanding latent sector errors and how to protect against them," *ACM Transactions on storage (TOS)*, vol. 6, no. 3, p. 9, 2010.

[20] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you?" in *FAST*, vol. 7, 2007, pp. 1–16.

[21] ——, "Understanding failures in petascale computers," in *Journal of Physics: Conference Series*, vol. 78, no. 1. IOP Publishing, 2007, p. 012022.

[22] T. Schwarz, D. D. Long, and J. F. Pâris, "Reliability of disk arrays with double parity," in *Dependable Computing (PRDC), 2013 IEEE 19th Pacific Rim International Symposium on*. IEEE, 2013, pp. 108–117.

[23] R. M. Tanner, "A recursive approach to low complexity codes," *Information Theory, IEEE Transactions on*, vol. 27, no. 5, pp. 533–547, 1981.

[24] A. Thusoo, Z. Shao, S. Anthony, D. Borthakur, N. Jain, J. Sen Sarma, R. Murthy, and H. Liu, "Data warehousing and analytics infrastructure at facebook," in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. ACM, 2010, pp. 1013–1020.

[25] G. Wang, S. Lin, J. Liu, G. Xie, and X. Liu, "Combinatorial constructions of multi-erasure-correcting codes with independent parity symbols for storage systems," in *Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on*. IEEE, 2007, pp. 61–68.

[26] L. Xu, V. Bohossian, J. Bruck, and D. G. Wagner, "Low-density MDS codes and factors of complete graphs," *Information Theory, IEEE Transactions on*, vol. 45, no. 6, pp. 1817–1826, 1999.

[27] J. Zhou, G. Wang, X. Liu, and J. Liu, "The study of graph decompositions and placement of parity and data to tolerate two failures in disk arrays: Conditions and existance," *Chinese Journal of Computers (chinese edition)*, vol. 26, no. 10, pp. 1379–1386, 2003.