Understanding Data Survivability in Archival Storage Systems

Yan Li   Ethan L. Miller   Darrell D. E. Long

Storage Systems Research Center & Computer Science Department
University of California, Santa Cruz
{yanli,elm,darrell}@cs.ucsc.edu

Abstract

Preserving data for a long period of time in the face of faults, large and small, is crucial for designing reliable archival storage systems. However, the *survivability* of data is different from the *reliability* of storage because typically, data are stored in more than one storage at a given moment. Previous studies of reliability ignore the former. We present a framework for relating data survivability and storage reliability, and use the framework to gauge the impact of rare but large-scale events on data survivability. We also present a method to track all copies of data and the condition of all the online and offline media, devices and systems on which they are stored uninterruptedly over the whole lifetime of the data. With this method, the survivability of the data can be closely monitored, and potential dangers can be handled in a timely manner. A better understanding of data survivability can be used in reducing unnecessary data replicas, thus reducing the cost.

1. Introduction

As computer systems are taking more and more responsibility in processing critical information and data, the need for reliable data storage is ever-increasing. However, from time to time we hear of high profile data loss accidents, such as NASA's missing Apollo project tapes that contained the original footage of the Apollo 11 moonwalk [16], online backup provider Carbonite's 2004 accident [32] that damaged over 7,500 customers' data, and the recent accident in which cloud computing provider Amazon lost data in one of its Availability Zones within the US East Region in 2011 [31]. These accidents indicate that we need a better understanding of data survivability as well as improved methods for calculating it in large-scale and complex systems.

SYSTOR'12   June 4–6, 2012, Haifa, Israel.
Copyright © 2012 ACM 978-1-4503-1448-0/12/06...$10.00

There has been a great deal of research in improving the understanding of storage reliability by using reliability metrics, measurement of failure trends in the field, and error detection and recovery. However, the storage solutions in the real world are often so complex that calculating data survivability is difficult, to the degree that people believe "our inability to compute how many backup copies we need to achieve a reliability target is something we are just going to have to live with." [24]

Previous research on storage reliability falls short in several aspects. First, it fails to differentiate *survivability of data* from *reliability of storage*, even though in most cases these two measures are not the same. In most archival storage systems, data are replicated across several systems, sites, and backup media, so the survivability of the data is based on the combined reliability of these storage. However, a systematic study on data survivability in complex storage is absent.

Second, there's still no easy way to answer a question like "Is our data safe enough?" People try to increase the safety of their data by following practices like deploying more backup servers and creating offsite replicas. But quite often these practices are carried out without having a quantitative analysis of their contribution to data survivability versus their cost; thus, data loss (safety lower than expectation) and overspending (safety higher than necessary) are still common.

Third, the correlation between failures, especially those caused by rare but large-scale events, are not taken seriously enough. There has been some work on the impact of correlated failures in wide-area systems [19], but little work on quantifying this problem and providing a framework to minimize its impact. Events that impact many replicas at once, such as human errors and earthquakes, can greatly reduce data surviviability. To address this issue, system designers and maintainers have developed conventions, such as acquiring devices from multiple vendor, and keeping replicas geologically distributed. There are few if any quantified analysis of them; thus, people are not sure if these practices are sufficiently good for archival storage, which must preserve data for a very long time.

There are some other situations that are often overlooked in reliability studies, such as data safety during transportation. Suppose a data center is to be relocated, and storage and media must be put on a vehicle and transferred to another location. The chance for a vehicle to crash on the road is much higher than that of storage safely kept in a data center; thus, during transportation, the data's safety is very likely to be lower than the requirement. Shall we create extra replicas before transportation? Or shall we plan the transportation carefully so that no two replicas of one data object are transported at the same time?

This paper proposes a systematic way to solve these problems. Instead of using a microscope and studying the reliability of a device or a system, we step back to have a broader view, in order to quantify the survivability of data objects stored in heterogeneous storage systems. The contribution of this paper includes:

1. A model to calculate the survivability of data objects stored on heterogeneous storage systems. This model is easy to understand and use, targeting real world data center management.
2. The use of a *Reliability Transition Function* to calculate reliability of a storage system from the reliability of underlying devices.
3. Combining the device's reliability model and S.M.A.R.T. events in calculating the reliability of devices.
4. Quantifying the impact of large-scale events, such as earthquakes, on the survivability of data.
5. Using the result from this model to reduce the cost of storage systems and achieve better data survivability by techniques such as improving data layout algorithm and scrubbing planning.

2. Background

2.1 Demand and Regulations

In the storage system research area, there is a rising trend to take reliability of storage more and more seriously. One reason is that much of the culturally and historically significant information is born digital these days and, if not preserved correctly, is easily lost forever. To address this concern, a lot of effort is going into designing archival storage systems for organizations such as libraries and the government [10, 11, 18]. Understanding the survivability of data is paramount for archival storage systems because the most important design goal for such systems is to ensure the survival of the data for a very long period of time—the whole idea of archival storage would be meaningless if the safety of the data can't be guaranteed.

On the industrial and business side, regulations such as the Health Insurance Portability and Accountability Act of 1996 (HIPAA) and Sarbanes-Oxley Act of 2002 demand that important business and medical data must be retained for varying period of time, requiring better data preservation methods.

2.2 Metrics and Models

Since Patterson and Gibson's work on RAID [22], MTTDL (Mean Time To Data Loss) has been widely used in both research and industry as a standard metric for analyzing the reliability of storage systems. However, as Greenan et al. [15] point out, MTTDL is an expectation of time to fail over an infinite interval, which is good for quick, relative comparison, but not meaningful for understanding the real survivability of data. To address this issue, he proposed to use Normalized Magnitude of Data Loss (NOMDL) for measuring the reliability of systems. NOMDL_t is the expected amount of data lost (in bytes) in a system within time t normalized to the system's usable capacity. The importance of this study is that it quantified data loss rate per unit time, broaching the idea that different data objects stored on the same devices can have drastically different survivability.

At the lowest level of a storage solution lies the devices, such as hard drives, tapes, and solid-state drives (SSD). A traditional rotating magnetic platter hard drive is a complex system, and many studies have revealed its reliability characteristics, including the Mean Time Between Failure (MTBF) and Annual Failure Rate (AFR) often given by vendors. However, MTBF and AFR are population statistics and are not relevant to individual units. A vendor-quoted MTBF implies that half the drives in a large population will fail within that time of operation. Several studies [23, 25] provide real world data of hard drive failure patterns. Tape is still widely used in enterprise data center and as archival storage. Gartner and Storage Magazine reported that about 34% of companies never test a restore from tape. Of those that do test, 77% experienced failures in their tape backups [7]. SSD is still a comparatively new kind of device, so research and statistics about their reliability are scarce. However, considering the high complexity in both the manufacturing process and control software algorithms, the current generation of SSD products is not expected to be as reliable as the old rotating platter hard drives in data center usage [27].

At the system level, Markov models are used widely for modeling, since they are suitable for analyzing systems that can be precisely defined as transitions between finite states, where the transitions follow some known distribution that can be expressed in closed-form expressions. Within this category, a lot of study has been done using both analytical and simulation methods [5, 12, 13, 15]. Markov model-based analysis and simulation can give useful information for understanding a specific device given all the detailed internal information of that device, and can also be used to describe simple replica and erasure code-based systems like RAID. However, its usage in analyzing complex devices with unknown properties is limited.

When it comes to storage algorithms and system designs that are of high complexity, people tend to withdraw from these formal mathematical tools and simply assume that creating more replicas means higher safety. There's a disparity between studies that model a specific storage medium or device and studies that introduce new storage algorithms or systems. For the former, the designer normally uses some kind of probability distribution model to describe the failure of components, which is based upon the observation that as devices grow older, they will become more and more unreliable [3, 9, 12, 13, 15, 22]. For the later category of studies, people simply assume that creating more copies means higher safety for the data, largely due to that fact that a precise mathematical model is so hard to build due to the sheer complexity of the new algorithm or system [18, 28, 30, 33].

2.3 Increasingly Heterogeneous Storage

Broadly speaking, most data in mid-scale to large-scale organizations are stored in heterogeneous storage solutions, as illustrated in Figure 1.

Gladney [11] and Giaretta [10] studied the motivation behind using heterogeneous archival storage systems. Generally speaking, factors that contribute to the evolution of a storage solution into a heterogeneous system include:

- Technology obsolescence: old vendors may go away, and spare parts for old devices are no longer produced.
- Leveraging new technology for better performance and total cost of ownership: new storage products are introduced to the market every day, so it's natural for users to pick up the most cost-effective device when replacing old parts. [23].
- Budget and resource changes may introduce new constraints: budget, power, rack space, staff, etc.
- To avoid spike correlated failures among devices, it's a common practice for data centers to procure devices from competing vendors from the market [23].

A given piece of data in such a heterogeneous environment can be stored on one or more of those systems at a given moment. Systems such as Logan [29] can help automate management of heterogeneous archival storage systems, alleviating the management overhead of data migration and device upgrading. However, there's no easy way to understand the survivability of data in such a complex solution.

2.4 Large-scale Disasters

Since archival storage is expected to survive for a very long time, it is likely that during their lifetime some rare but large-scale events may eventually occur. These events include intrusion, malware infection, earthquakes, terrorist attack, theft, fire, war, or even just a power surge. Such disasters can trigger other types of threat, such as media, hardware, and organizational faults [3]. Unfortunately, quantitative analysis of these rare events are scarce.

3. The Model

This model is an analytical model of data survivability. Generally speaking, it examines all devices used in an organization in a bottom-up manner. Each data object is treated individually according to its storage allocation on physical devices. First, for each object, the underlying devices on which it is stored are examined and the reliability of each device is calculated. Second, the reliability information of each device is merged together at the system level, taking into consideration the system's design and the storage algorithm's characteristics. Third, the effect of large-scale events that may affect more than one storage systems are calculated. Fourth, all the information above is combined to give an estimation of the data's current survivability, as well as the projection for survivability into future. The survivability can also be checked against set policies and standards, raising an alert if it is below the desired level.

3.1 Terminology

In the remainder of this paper, we use the following terms:

Data object (DO) The smallest unit of information. In popular file based systems, files can be directly treated as DOs. A DO is itself meaningful and can't be broken down further. For a DO without error-correcting code (ECC) protection, flipping one random bit can corrupt it and render it useless. Objects that contain ECC can tolerate some degree of damage.

Device A physical device that stores bits. Common storage devices include hard drive, SSD, tape, compact disc, etc. It's worth noting that in this paper's context, removable media are also devices. Since it's a common practice to store DOs on both online and offline backup media, unifying both kinds of storage under the name "device" helps us address the calculation of reliability in a coherent way. Because removable media can be stored, read and written separately, one piece of removable media is treated as one device in the following discussion.

Device property A physical property of a device that can be measured and tracked. For a rotating-platter hard drive, the properties may include: power cycle count, head load/unload count, seek error rate, read error rate, power on hours, etc. Device properties vary from device to device; for example, an SSD device can also have the property Power On Time, but Head Seek Error rate is meaningless for it.

Storage system A storage system consists of storage devices. A storage system can be roughly seen as one or more controlling units plus one or more devices. A desktop RAID network-attached storage system (NAS) is a storage system. A large storage cluster that has hundreds of boxes and thousands of disks is also a storage system. Distributed systems that span multiple sites are conceptually divided into one or more system per site for easy cal-

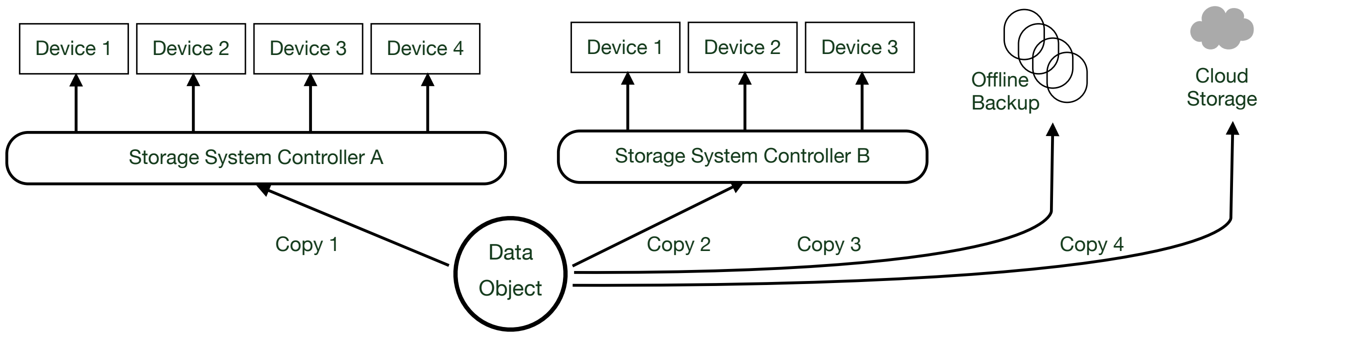

Figure 1. A Heterogeneous Storage Solution

culation when we talk about large-scale events later. An online storage service provider is also seen as a storage system in this paper, though it may not have any physical device that we can track. Storage system is abbreviated as "system" in most cases.

Storage solution A storage solution consists of storage systems. It is the whole solution an organization (or an individual) deploys to store its data. In the context of this paper, a storage solution is distinguished from a storage system in the sense that for one organization, there is only one storage solution, which includes all deployed storage systems. For example, a large organization may have one storage solution that consists of large storage clusters with hundreds of boxes and spans multiple sites, plus nearline storage and offline tape/CD backup, as we have shown in Figure 1. No matter how complex it may be, it is still one storage solution. Storage solution is abbreviated as "solution" in the following text.

Time Time in this paper flows continually from 0. The variable's unit is *hour* in practical calculations and charts.

The relationship between storage devices, systems and solution can be expressed as:

1. DOs are kept in one storage solution.
2. A storage solution consists of one or more storage systems, which can be from one or more vendors.
3. A storage system consists of one or more storage devices.

There's no perfect model that can precisely reflect the complex world of long-term storage, but in order to make our model useful for guiding the management of DOs and systems, we make this first assumption for the calculation of survivability: we always aim at the theoretical lower bound. Instead of trying to get a precise figure for survivability by using complex simulation models, we use an analytical model to get the theoretical lower bound of survivability. Since the goal of calculating survivability in this paper is to guide solution design and deployment, it's not bad if the system performs better than the model predicted. However, it will be a disaster if the model predicts that data is safer than it really is, since this may result in a much higher likelihood of unexpected system failure. Therefore, instead of using simulation methods that require too much simplification and may ignore the complexity of the problem, we choose to use the analytical method and aim at getting the theoretical lower bound of the survivability.

We also note that, even though we are aiming at a "lower bound", there's no way to guarantee that we can achieve it, because the more threats are considered, the lower the calculated survivability is, and it is impossible to cover all threats. This is the reason we we call it "theoretical lower bound". In order to get a better lower bound, we should cover all major threats and carefully choose the calculation method we use.

The second assumption is that a Data Object (DO) is the basic unit of stored digital data. As described above, DO is the smallest unit of data that preserves the meaning and further dividing a DO is meaningless. Although some DOs have built-in ECC and can recover from minor damage, in this analysis it doesn't matter whether ECC is stored at DO or file system level; thus, both approaches can be treated identically mathematically. For the sake of model simplicity, we assume ECC is always done at file system level, and that a DO has no built-in ECC, so changing one bit of a DO corrupts it. This assumption simplifies matters, since ECC implemented as part of the DO format may cover different parts of the DO at different levels. For example, an archive may suffer only minor damage if a sector within an archived object is damaged, but may be totally unrecoverable if the file header or table of contents is corrupted.

3.2 Data Object's Survivability

Most users care more about their data than about the reliability and lifetime of a storage system. Thus, if a storage system goes down, as long as the data is backed up somewhere else, the user just needs to fix or replace the broken system. On the other hand, loss of a data object might be a big threat to the user's business goals. This drives our first major objective: calculating the likelihood that a data object will survive over time.

In existing literature, the survivability (survival rate) of a data object is often treated as equal to the reliability of a storage system, as described in the introduction and background section. For most storage solutions, however, that is not the case. In this section, we derive a basic model for the survivability of a Data Object.

We begin by giving the formal definition of Data Object's Survivability (DOS) used in this paper. The DOS of a data object is defined as:

> the probability that a data object will survive during a specified period of time (t) under stated conditions.

Mathematically, this can be expressed as the cumulative distribution function (CDF) of the failure probability density function of the devices on which it's stored:

$$\begin{aligned}\mathrm{DOS}(t) &= Pr\{T > t\} = \int_t^\infty f(x)\,\mathrm{d}x \\ &= 1 - Pr\{0 < T \le t\} = 1 - \int_0^t f(x)\,\mathrm{d}x \end{aligned} \quad (1)$$

t is the length of the period of time (assumed to start from 0), $f(x)$ is the failure probability density function, and random variable T is the DO's lifetime.

If the user has only one copy of an object stored in one storage system, then the *upper bound* of the survivability of the data object is the reliability of that storage system. It's an upper bound because the object isn't expected to survive if the storage system fails. And in reality, this upper bound can never be achieved because there are many events that can destroy the storage device, such as device loss and an earthquake. Storage device designers won't consider these events when calculating their system's reliability, even though they all contribute to the lowering the DOS to below the storage system's reliability. Section 3.5 discusses the modeling of large-scale events.

When one storage system cannot meet the increasing demand of the user, it will be expanded or new systems will be deployed along with the old system. In that process, data objects will be migrated from the old system to the new or expanded system, potentially meaning that they will reside on more than one storage system. For a solution consisting of n storage systems, its failure rate ($\mathrm{DOS}_n(t)$) can be expressed as a function of the CDF of the underlying systems' failure rates:

$$\mathrm{DOS}_n(t) = g(F_1(t), F_2(t), \cdots, F_n(t)) \quad (2)$$

In the simplest form, using only plain replicas without erasure code, the failure of these systems are uncorrelated—we will discuss large-scale events that can affect more than one system in Section 3.5 later—and the object will only be lost if *ALL* of these n storage systems fail. The probability for this loss event to occur (the CDF of it, $F_{lost}(t)$) can be expressed as:

$$F_{lost}(t) = \prod_{i=1}^{n} F_i(t) \quad (3)$$

The DOS in this case is the probability that F_{lost} doesn't occur, and it can be expressed as

$$\begin{aligned}\mathrm{DOS}_n(t) &= 1 - F_{lost}(t) \\ &= 1 - \prod_{i=1}^{n} F_i(t) \quad (4) \\ &= 1 - \prod_{i=1}^{n} \Big(1 - R_i(t)\Big) \quad (5)\end{aligned}$$

For one device, the CDF of failure rate $F(t)$ and the survivability $R(t)$ have this simple relation: $F(t) = 1 - R(t)$, and both of them will be used in the following discussion.

3.3 Reliability Transition Function

Models that assume only simple replication are not sufficient for long-term storage because, in practice, storage systems employ algorithms such as n/m erasure codes to reduce the space and time overhead of creating simple replicas. Therefore, we introduce the *Reliability Transition Function* (RTF) to cover these designs.

Consider the storage system configuration shown in Figure 1. The DOS depends on the reliability of Storage System Controller A (R_{sysA}), which in turn depends on the reliabilities of underlying devices. We propose to use the RTF to denote this relationship. Suppose the reliability of the underlying devices are $R_{\mathrm{dev1}}(t), R_{\mathrm{dev2}}(t), \cdots, R_{\mathrm{devN}}(t)$. The RTF can be defined as:

$$\begin{aligned}&R_{\mathrm{sysA}}(\mathrm{ObjID}, t) = \\ &\mathrm{RTF}_{\mathrm{sysA}}\big(\mathrm{ObjID}, R_{\mathrm{dev1}}(t), R_{\mathrm{dev2}}(t), \cdots, R_{\mathrm{devN}}(t)\big)\end{aligned} \quad (6)$$

In the above equation, ObjID is the DO's ID, which can be used by the storage system to identify a DO. It is needed because different DOs can be stored on different devices even within the same system. In calculation, only the devices where the DO resides are considered; the other devices will be ignored by the RTF.

RTF describes the reliability of a single storage system and it can be implemented either analytically or by simulation. Greenan et al. [14] show that this task can be complex for even relatively simple systems, and, in order to get the precise reliability of a storage system, some simulation methods must be used. With simulation, however, the precise reliability of a system can't be expressed in a closed-form expression; rather, the simulation must be run for each point in time. In practice, it's not unusual that the DOS of millions objects have to be tracked and calculated, making the simulation method impractical if not impossible for large systems.

For commercial storage systems that use proprietary algorithms, we expect the storage system vendor to provide this RTF to enable end-users to calculate the DOS. We propose that, before a new storage system is purchased and deployed, the user should require the system vendor to provide an RTF for this specific system. Even better, the vendor can provide two or more RTFs: one of them can use simulation methods and be precise and the other can be fast using some form of approximation.

Jiang et al. [17] discovered that "disk failures contribute to 20–55% of storage system failures," and other important factors that shouldn't be overlooked include failures of the system controller, physical interconnects, and protocol stacks. There are two ways to cover these non-device failures in the analytical model. The first is to include them into the RTF, and the second is to handle them at a higher level, when we later aggregate the RTF from many systems to calculate DOS. We expect the device vendors to provide the RTF of commercial storage systems, and they should include in the RTF the analyses results for the non-device related failures. If not, these failures must be handled at a high level.

Here's a sample showing the RTF of a DO that is stored on an erasure code-based system that divides the object into m fragments and recodes them into n copies ($n > m$), as is done for most RAID systems. For the data to survive, at least m devices must survive, which means in order to destroy the data, at least $n - m + 1$ drives must fail. Remember that we are looking at the storage system as a dynamic system, of which one drive might fail and a new drive might be added at any time. Therefore the $R(t)$ of the system's drives are not identical. Our task to get the DOS has not become much harder with this dynamic view because we are only aiming at the lower bound. The RTF for an n/m erasure code based system can be expressed as:

$$\mathrm{RTF}(\mathrm{ObjID}, t) = \Big(1 - \prod_{k=1}^{n-m+1} \Big(1 - R_k(t)\Big)\Big)\Big(1 - F_{sys}(t)\Big) \tag{7}$$

$R_k(t)$ denotes the $n - m + 1$ devices that have the lowest $R(t)$. $F_{sys}(t)$ is the combined failure rate of non-device causes as we have discussed above. In order to get the lower bound of the system's reliability, the key point here is that we pick out $n-m+1$ devices that has the lowest reliability from all the n devices installed. This can be done by calculating $R(t)$ of all the n devices and sort the result, using only the least reliable $n - m + 1$ devices in the above equation. The reason we need to use those least reliable devices is that, even though erasure codes are normally designed for being used with identical devices, in reality devices within one system usually have different reliabilities due to reasons like failed device will be replaced by new one so they end up with difference power-on times, and reading may not be well-balanced so some of the devices receive more wear than the others.

If de-duplication is used in the storage system, the survivability of DO will be affected because the ability to recover a DO now depends on many chunks, each of which might be stored on a different set of devices. Due to the normally proprietary nature and subtleties in de-duplication implementations, the requirement for the storage system vendor to provide the RTF is overwhelming. Again, here our goal is to get the lower bound of these systems, so that we can use a simple form of the equations. If the precise survivability is expected, simulation based methods might have to be used.

With RTF, we can expand our previous DOS equation (5) to:

$$\mathrm{DOS}(\mathrm{ObjID}, t) = 1 - \prod_{i=1}^{n} \Big(1 - \mathrm{RTF}_i\big(\mathrm{ObjID}, R_{\mathrm{dev}i}(t)\big)\Big) \tag{8}$$

Next, we continue to the lower level of the storage solution and derive a good expression for $R_{\mathrm{dev}}(t)$.

3.4 Modeling Devices

Different devices have drastically different reliability models. As shown in Equation (8), each $R_{\mathrm{dev}}(t)$ is an independent function and can be expressed separately by using results from previous research. Therefore, for storage solutions such as the one shown in Figure 1, the reliability models of hard drive, tape and cloud storage will have to be figured out. Here we are using a hard drive as a sample subject for the analysis.

Among the reasons that lead to the failures of devices in a data center, aging is the biggest contributor, as described by the extensive analyses of hard drive reliability [8, 23, 25]. Recent research shows that a Weibull reliability model gives good results [12, 25], and it reflects both failures from infant mortality and aging. With the Weibull model, the reliability function of a hard drive is $\mathrm{R}(t) = e^{-(t/\eta)^{\beta}}$, where β is the *shape parameter* and η is the *scale parameter*. Previous analyses of large deployments of hard drives produce Weibull distributions with slightly different parameters.

For this study, the specific value of these two parameters is not of great importance for three reasons. First, the model we are discussing here is agnostic of the underlying reliability model of devices. Here, we choose the Weibull distribution for demonstrating analysis of hard drive-based systems, but, for other devices, it's possible that other models are more suitable. Second, these are empirical values and are highly brand and model correlated, so they should be picked according to the particular devices deployed. There is no expectation that different hard drive models from a single vendor, let alone those from different vendors, will have the same values for η and β. Third, we will be able to use Equation (15) to adjust these parameters dynamically as shown later. Therefore for simplicity, we use $\beta = 1.12$, $\eta = 100,000$ in the following discussion. These parameters align with values cited in a previous study on failure rates of large numbers of hard drives [25].

In the following sample, we apply Weibull reliability model to the n/m erasure code system and expand Equation (7), without considering the possibility of F_{sys}:

$$\mathrm{DOS}_n(t) = 1 - \prod_{i=1}^{n-m+1} \left(1 - e^{-(\frac{t}{\eta})^\beta}\right) \tag{9}$$

Figure 2 illustrates the effect of device aging. The solid blue curve shows DOS of an object stored in a two-drive mirrored RAID 1 system using two new hard drives, and for comparison, the dashed red curve shows the DOS if one of these two hard drives is four years old at the beginning of the experiment.

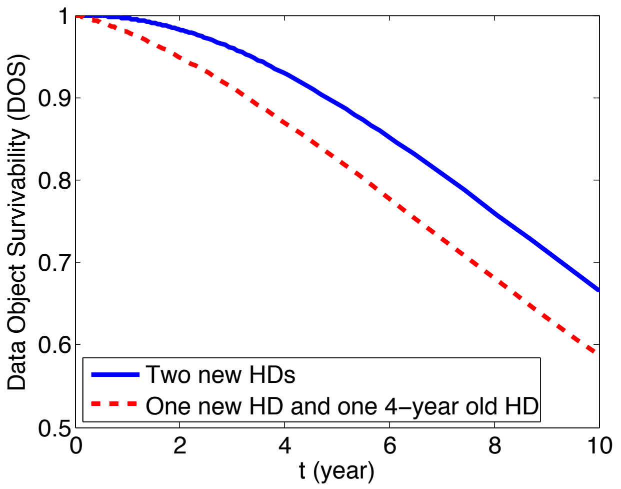

Figure 2. DOS of object stored in mixed old and new hard drives, Weibull model, $\beta = 1.12$, $\eta = 100,000$ hr

The survivability described by Equation (9) is only an upper bound that can never be reached because we have not considered many other failure events.

It is possible to use more complex methods than that of Equation (5) to get a more precise DOS, as proposed by Greenan [13], but they are much more difficult to model than the analytical model used here. In order to use the Markov model, the time to recover from a system failure must be constant or follow a known possibility distribution that can be handled analytically. For some well engineered storage systems, this estimation is possible. However, if a whole storage system in a company's data center goes down, normally the system adminstrator has to rely on the vendor's customer support staff to diagnose and repair the failed system, resulting in difficult-to-predict resolution times. Thus, instead of focusing on the precise modeling of the internal of a complex storage system, we use a simple but effective way to get the lower bound of survivability of DOs stored in more than one systems.

3.5 Modeling Events

An "event" is a thing that occurs and has an impact on system and device reliability. We observe and study events and how they change systems and devices, helping us to understand the storage solution during its whole lifetime. We classify events into two categories: failure events and operational events. Failure events are those that lead to data loss (not necessarily the loss of all replicas), such as device failures, bit rot, natural disasters, critical software and hardware failures, human errors, and even organizational failures. Operational events include normal and abnormal operations of devices and systems that affect their life, such as power cycles, disk head load/unload, surface scan errors and erasing a block of SSD memory.

3.5.1 Failure events

The failures of devices have been studied extensively in literature and are covered by the reliability CDF functions we described above. Thus, we start from quantifying events that fall out of normal device and system vendors' consideration.

There are many kinds of disastrous large-scale events that could destroy all storage systems located in one place, such as earthquakes, fire, flood, military actions, etc. Among them, earthquakes are a relatively well studied and powerful disaster that can easily destroy the entire building where all of the devices are located. Whether the earthquake at a location is a memoryless event or not is still debatable, but for analyzing the risk of future earthquakes, it's sufficient to treat earthquakes as events in a memoryless Poisson process [6]. This assumption leads to the exponential distribution $\mathrm{P}(t) = 1 - e^{t/M}$, where M is the mean time between earthquakes. In order for a DO loss to happen, either all storage systems the object resides on go bad (described by Equation (9)) or one earthquake occurs, destroying all of the storage systems. The possibility for either of them to happen can be calculated by using the inclusion-exclusion principle $\mathbb{P}(A_1 \cup A_2) = \mathbb{P}(A_1) + \mathbb{P}(A_2) - \mathbb{P}(A_1 \cap A_2)$. Applying Equation (3) and the possibility distribution of earthquakes, we can get the DOS when we take the impact of earthquakes into consideration:

$$\mathrm{DOS}_n(t) = 1 - \left(F_n(t) + F_{eq}(t) - F_n(t) \times F_{eq}(t)\right) \tag{10}$$

where $F_n(t)$ is the combined failure rate of all devices and $F_{eq}(t)$ is the happen rate of earthquake.

An earthquake seems to be a very rare event. In order to show its impact, we use the sample configuration we have discussed above in section 3.4, a two-drive mirrored RAID 1 system stored in one building that can be destroyed by a single earthquake, and assume these devices are located somewhere in a coastal city in southern California. According to Akçiz et al. [1], the average time interval between the last six earthquakes that ruptured the San Andreas fault in the Carrizo Plain is 88 ± 41 years. Using Equation (10), the impact of earthquakes on data survivability is shown in Figure 3.

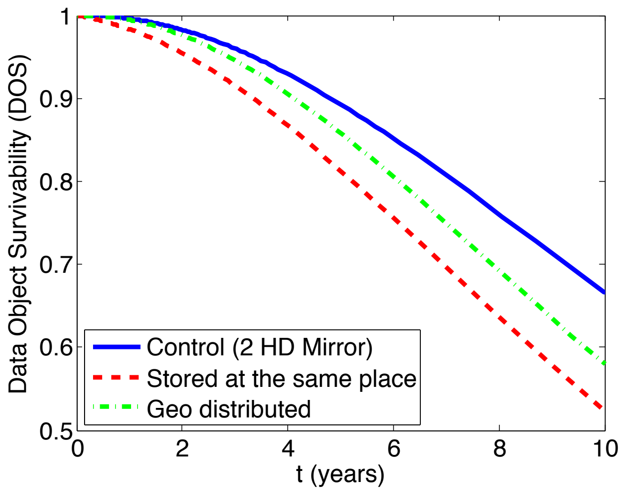

Figure 3. Impact of earthquake on DOS

The solid blue line curve shows the DOS without considering the impact of the earthquake, and the dashed red curve shows the result of Equation (10) after considering the impact of earthquakes. To mitigate the threat to data of a large earthquake, the most straightforward practice is to deploy the storage systems in a geologically distributed way, such as keeping one storage server in San Francisco and the other in New York. Then, if we assume that a single earthquake can only destroy a single storage system, the data loss can only happen if one of the following four events happens:

1. Both hard drives fail.
2. Hard drive A fails and B is destroyed by an earthquake in New York.
3. Hard drive A is destroyed by an earthquake in California and B fails.
4. Both hard drives are destroyed by two (separate) earthquakes.

The overall possibility is calculated by using the inclusion-exclusion principle, with the result shown in Figure 3 by the green dashed dot line curve. As the figure shows, it's slightly safer than the red dashed line curve, but still the impact on the DOS even when the storage systems are geologically distributed can't be ignored.

Besides earthquakes, there are many kinds of failure events that should be modeled: notably hardware and software defects, human errors, malware infections, and security breaches. Hardware and software may cause data corruption or loss if they have some hidden defect in their design, and tend to show correlated failure patterns if they are from the same vendor [21]. Human errors, such as system misconfiguration, accidental data deletion and data mislabeling, may cause unrecoverable data loss no matter how many replicas are created. How these events can be modeled has been studied by prior research to some degree, but is still not enough.

To generalize the equation into calculating the DOS when we consider m kinds of events that can wipe out all storage devices, we can use the general case for the inclusion-exclusion principle and write it in this closed form:

$$\mathrm{DOS}_m(t) = 1 - \sum_{k=1}^{m}(-1)^{k-1}\sum_{\substack{I\subset\{1,\ldots,m\}\\ |I|=k}} F_I(t) \tag{11}$$

3.5.2 Operational events

While events that can destroy entire storage systems are catastrophic and have a major impact on data survivability, smaller events that are not that serious are far more common. For example, most modern hard drives are shipped with the "Self-Monitoring, Analysis and Reporting Technology" (S.M.A.R.T.) monitoring system, which collects internal events and running status that can be queried by the system. S.M.A.R.T. records events that have been found to correlate with the reliability of the storage device in a previous study [23].

S.M.A.R.T. collects a plethora of raw information, so we should first try to identify those that are useful for our analyses and use them to define some polices, hoping this can reduce the cost and/or improve the precision of the calculated system reliability. For example, one of the most conspicuous events is sector error. A previous study showed that even a small number of sector failures presages the overall drive failure [2].

Another interesting S.M.A.R.T. event is the Scan Error. Modern hard drives scan the disk surface during idle time, looking for bad sectors. Getting a Scan Error doesn't mean that the drive is broken or data is lost because most Scan Errors can be automatically recovered (by repeatedly reading the bad sector). But a large number of Scan Errors is a good indicator of surface defects and are believed to lower the predicted device reliability. One study [23] found that a group of drives with Scan Errors are ten times more likely to fail than the group with no such error. With this knowledge, when a Scan Error event is detected on a hard drive, the reliability CDF $F(t)$ of which should be adjusted to $1/10$ of its previous value; thus, its new CDF is:

$$\begin{aligned} F_{\text{new}}(t) &= 1 - \frac{R(t)}{10} \\ &= 1 - \frac{1-F(t)}{10} \end{aligned}$$

It is worth noting that, according to previous studies, S.M.A.R.T. data alone can't be used effectively to predict future failures [23]. Since we are calculating the lower bound of DOS, S.M.A.R.T. events are still a good indicator because there's a sufficiently high correlation between device failure rate and some of the error events listed above.

Generally speaking, this category of events contains many kinds of operations that can be tracked and used in the calculation of DOS. Given that we know event K's effect on the device's reliability can be calculated by $E_K()$, and the CDF of the device is $F(t)$, the new CDF of the device after event K happens is:

$$F_{\text{new}}(t) = 1 - E_k(1 - F(t)) \tag{12}$$

And after a series of event from 1 to K, their whole effect on $F(t)$ can be calculated by using:

$$F_{\text{new}}(t) = 1 - E_k(E_{k-1}(\cdots E_1(1 - F(t))) \tag{13}$$

Equation (13) can be combined with our previous DOS, Equation (8), resulting in:

$$\text{DOS}(\text{ObjID}, t) = 1 - \prod_{i=1}^{n} \Big(1 - \text{RTF}_i\big(\text{ObjID}, E_i\big(R_i(t)\big)\big)\Big) \tag{14}$$

Equation (14) is our final equation for computing DOS. It covers both the aging of devices and the impact of two categories of events to the DOS. Additional events can be included in the model in similar ways.

3.5.3 Events that should be omitted

The nature of analyzing the DOS for objects that are stored on more than one storage system leads to the reconsideration of some common events used in survivability analyses.

Bit rot is an event that falls within this category. In the study of any storage system that runs for more than a few years or employs more than a few dozens of devices, the impact of bit rot must be taken seriously. However, the impact of bit rot event should already be covered by the Reliability Transition Function as described in section 3.3. Obviously, for the rare cases when the DOs are stored directly on bare metal drives (and there's no RTF), the bit rot event must be taken into consideration when calculating the DOS. Normally, however, archival storage systems have higher-level mechanisms for mitigating the impact of bit rot errors.

3.6 Tracking Events

In this model of events, tracking events is an important task. By incorporating the knowledge of every event that happened we can have a better understanding of the DOS.

Suppose we have a DO that is stored in two (or three) systems, and at time t_1 one of the systems failed. If the system's reliability follows the Weibull distribution, the DOS of this object can be calculated by using Equation (9), as shown in Figure 4.

Now, if the failed system is replaced at time t_2 ($t_2 > t_1$), what would the DOS look like? We know that if the newly installed system is a brand new system, its reliability function should take $t - t_2$ as parameter. Therefore the DOS after the new system is installed is:

$$\text{DOS}(t) = 1 - F_1(t) \times F_2(t - t_2)$$

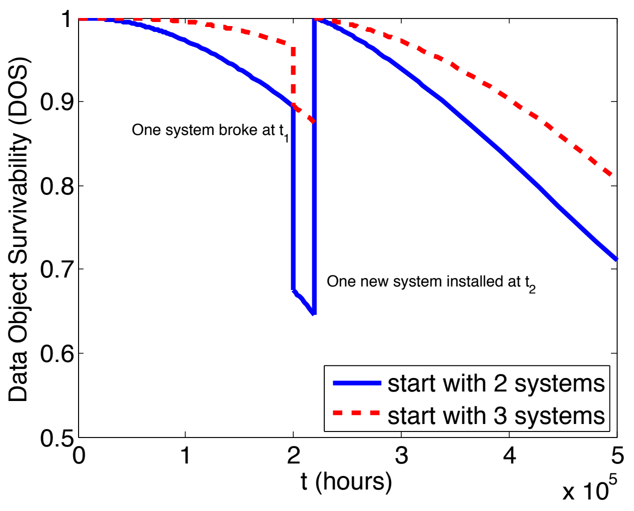

Figure 4. Effect of a system failure that is later repaired.

As can be observed in Figure 4, the DOS drops more quickly after t_2 than at the beginning because after t_2 the system becomes a heterogeneous system that consists of one old system and one new system, and the new combined survivability isn't as high as that of two brand new systems. With a chart like this, system implementers and users will have a better understanding of the survivability of their data after a series of events.

3.7 Refining DOS Models Using Failure Statistics

A key factor for calculating a correct DOS lies in obtaining the correct lower bound of reliability of a device (the $R(t)$ function). In previous examples, we used the empirical value for the Weibull distribution parameters. However, other research on the failure patterns of large numbers of hard drives shows that a hard drive's reliability differs greatly from one brand to another, or even from one shipment to another [17, 23]. In this section we propose that data can be gathered from the field during a storage system's or device's lifetime to fine tune our DOS calculation.

When a storage system is deployed and the initial DOS is to be calculated, we can normally get the "Mean Time to Failure" value straight from a device's specification. However, in most cases vendors fail to specify what kind of failures they considered when calculating MTTF. More troubling is the observation that, in the field, the replacement rate of hard drives is generally much higher than the value calculated from the vendor's MTTF [25].

With these considerations in mind, we propose the following method to tune the reliability function of a device. We divide the devices into groups by their manufacturing batch (or shipment) because previous studies show that hard drives from the same shipment show similar failure patterns [17, 23]. Let N be the count of failures we observed, and T_i ($1 < i \leq N$) be the lives of these N failed devices. Using the Weibull model as described in Equation (9), we adjust the scale parameter η in this way:

$$\eta = \frac{100,000 \times \alpha + \sum_{i=1}^{N} T_i}{\alpha + N} \quad (15)$$

where α ($\alpha \geq 1$) is the *weight parameter* of the empirical value. The larger α is, the more weight the empirical value has over the field gathered data.

In this example, we demonstrated how we calibrate the $R(t)$ for rotating platter hard drives using real-world failure statistics that can be gathered for any moderate to large-scale installation. Similar analysis can also be done for other storage devices such as SSD; we just need the CDF and initial empirical values for MTTF. This technique is thus particularly useful for new technologies for which good estimates of longer-term MTTF may not exist.

4. Applications and Future Work

While we have only provided simple examples, the model described in this paper allows the inclusion of increasingly detailed information about failure characteristics as well as real-world failure information. At the same time, we also perceive that it is impossible to model all large events and there must be some big events that fall through the cracks of the analysis. Another problem is that some storage system vendors might not be willing to provide an RTF for their systems; thus, building a model around their devices can be difficult or impractical.

We are currently studying the possibility of designing a smart data layout algorithm. Greenan [14] proposed that when designing an erasure code-based system to be implemented on a system with a mixture of heterogeneous devices with different reliabilities, the survivability of data varies among different layout algorithms. Similarly, systems for de-duplicated data may require higher reliability for chunks that are components of many objects [4]; our models allow system designers to optimize placement to achieve this goal.

Similar phenomena also exist when not only heterogeneous devices but also heterogeneous systems are deployed. Therefore, one of our future goals is a more general "survivability-aware layout algorithm" which not only considers the survivability of data but also cost constraints. DOs will be grouped according to their importance. For example, metadata are normally more important than normal data objects. In the simplest form, for better survivability, we can store the high priority DOs to mid-age devices, which are supposed to have better reliability. Less important DOs will be kept on brand new or old devices or with fewer replicas. This new layout algorithm can also help to lower cost: unreliable devices can be used safely because we can control the amount of important data stored on them, limiting the risk of loss. By allowing the use of less-reliable devices and systems, implementers can extend device replacement cycle, reducing cost.

Another application is adjusting the scrubbing interval [26] for various devices and systems to optimize DOS. Scrubbing is very important for preventing bit rot in archival storage systems, but too much scrubbing may also be harmful [20]. DOS models can help designers balance the need for shorter scrubbing intervals for aging, less-reliable devices and systems against the damage that increased scrubbing may cause them and fine-tune the scrubbing intervals.

The result of the study can also improve the planning and designing of storage solutions. For example, consider the design of a storage solution for a government archive where the budget permits us to make either three local replicas, or two geologically distributed replicas. Which design is better? Per previous discussion, many factors will have to be considered. Distributed replicas might be safer considering the impact of earthquakes, but at the same time it normally requires longer recovery time in case of media failure. The method discussed in this paper can help quantify these factors and help the decision making process.

We are also planning to build a Survivability-Aware Storage System Manager. The Storage System Manager plays a very important role in today's enterprises for helping more efficient usage of the storage systems and reducing both cost and downtime. However, current designs haven't taken survivability into consideration. To let the user have a better understanding of the DOS, an interface that can display a graph of DOS is needed. When all the methods we proposed above are used in the calculation, even getting the graph of one DOS can be tedious. Therefore a computer system should be designed and built to automate this task. Our initial design is shown in Figure 5.

Simply speaking, the "Survivability Monitor" should implement the algorithms we have discussed in previous section. It monitors and collects field data from each device in use, sends them through Reliability Transition Functions of the component storage systems and applies events' probabilities, and finally generates a continually updating "Survivability View", which can be used to monitor the dynamic changes of the survivability of data objects. When an event occurs, the event data will be automatically retrieved by the Survivability Monitor from the device if they are device-related events, or input by system admin if they are external events, and the survivability view will be updated in real time.

The Survivability Monitor can also handle predctions of data survivability for future events. For example, when one data center is planned to be taken offline and transported to somewhere else, the possibility for device damage during transportation is much higher than when the systems are maintained in-place. Therefore, this planned action and related events should be inputted into the Survivability Monitor as part of the planning process, to ensure the DOS is kept at the expected level during the duration of the transportation.

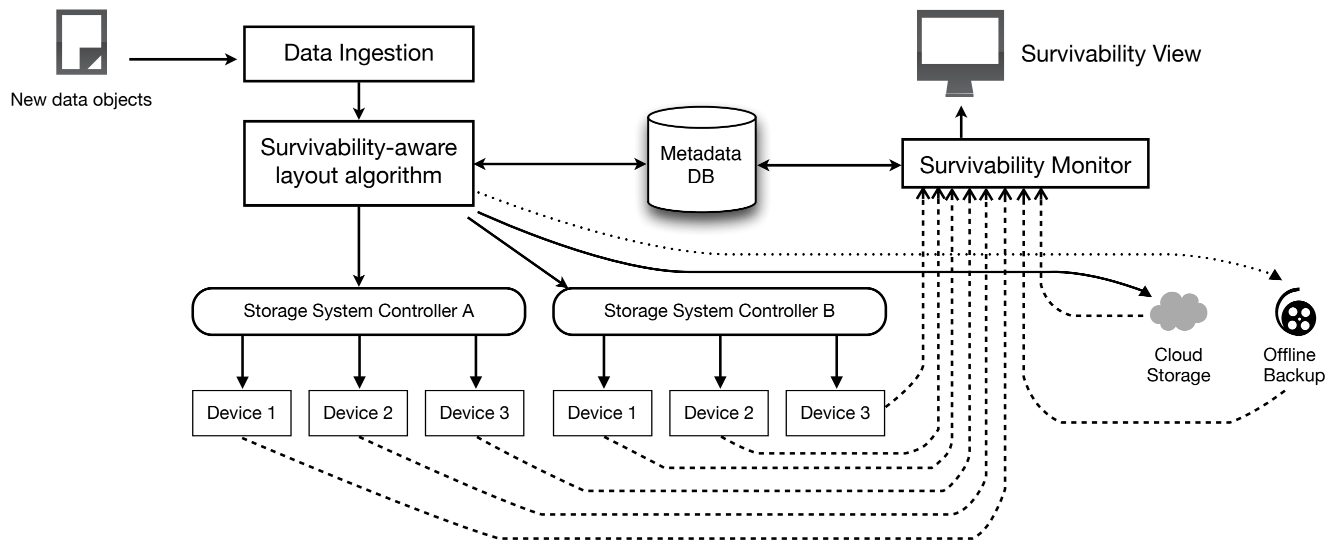

Figure 5. Survivability-aware Storage System Manager

5. Conclusions

Data survivability, not reliability, is the best metric for systems preserving data for the long-term, since it better encapsulates the desire for information to be available in the future. The models we have developed better capture the notion of data survivability for individual data objects than existing models, which focus on system reliability. Our modeling techniques allow for heterogeneous systems with geographic diversity, and can model the impacts of both common and uncommon events on long-term data viability.

We showed that uncommon events that are often omitted in long-term storage modeling, such as earthquakes, can have a real impact on data survivability. Even effects such as the replacement of a middle-aged system with a new one whose reliability is less well-understood can affect data survivability.

Our modeling approach also allows system designers to better plan for handling of objects whose importance may vary. By taking device aging, system configuration, and field-measured reliability metrics into account, we can provide better estimates of long-term data survivability, allowing system designers to provide effective archival storage at lower cost.

Acknowledgments

We would like to thank the faculty and students of the Storage Systems Research Center (SSRC) for their help and guidance, and the reviewers who gave us valuable comments and suggestions. This research was supported in part by the National Science Foundation under awards CNS-0917396 (part of the American Recovery and Reinvestment Act of 2009 [Public Law 111-5]) and IIP-0934401, and by the industrial sponsors of the SSRC, including EMC, Hewlett Packard Laboratories, IBM Research, NetApp, Northrop Grumman, and Samsung.

References

[1] S. O. Akçiz, L. G. Ludwig, J. R. Arrowsmith, and O. Zielke. Century-long average time intervals between earthquake ruptures of the San Andreas fault in the Carrizo Plain, California. *Geology*, 38:787–790, Sept. 2010.

[2] L. N. Bairavasundaram, G. R. Goodson, B. Schroeder, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. An analysis of data corruption in the storage stack. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies (FAST)*, pages 223–238, Feb. 2008.

[3] M. Baker, M. Shah, D. S. H. Rosenthal, M. Roussopoulos, P. Maniatis, T. Giuli, and P. Bungale. A fresh look at the reliability of long-term digital storage. In *Proceedings of EuroSys 2006*, pages 221–234, Apr. 2006.

[4] D. Bhagwat, K. Pollack, D. D. E. Long, E. L. Miller, J.-F. Pâris, and T. Schwarz, S. J. Providing high reliability in a minimum redundancy archival storage system. In *Proceedings of the 14th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '06)*, Monterey, CA, Sept. 2006.

[5] A. M. Blum, A. Goyal, P. Heidelberger, S. S. Lavenberg, M. K. Nakayama, and P. Shahbuddin. Modeling and analysis of system dependability using the system availability estimator. In *Proceedings of the 24th International Symposium on Fault-Tolerant Computing (FTCS '94)*, pages 137–141, 1994.

[6] Y. Bozorgnia and V. V. Bertero. *Earthquake engineering: from engineering seismology to performance-based engineering*. CRC Press LLC, 2006.

[7] R. Chalfant. Tape: A collapsing star. `http://www.mainframezone.com/storage/backup-recovery-business-continuity/tape-a-collapsing-star`, 2010.

[8] J. G. Elerath. Specifying reliability in the disk drive industry: No more MTBF's. In *Proceedings of 2000 Annual Reliability and Maintainability Symposium*, pages 194–199. IEEE, 2000.

[9] J. G. Elerath and M. Pecht. Enhanced reliability modeling of RAID storage systems. In *Proceedings of the 2007 Int'l Conference on Dependable Systems and Networking (DSN 2007)*, pages 175–184. IEEE, June 2007.

[10] D. Giaretta. *Advanced Digital Preservation*. Springer, 2011.

[11] H. M. Gladney. *Preserving digital information*. Springer, 2007.

[12] K. Gopinath, J. Elerath, and D. Long. Reliability modelling of disk subsystems with probabilistic model checking. Technical Report UCSC-SSRC-09-05, University of California, Santa Cruz, May 2009.

[13] K. M. Greenan. Reliability and power-efficiency in erasure-coded storage systems. Technical report, University of California, Santa Cruz, Dec. 2009.

[14] K. M. Greenan, E. L. Miller, and J. J. Wylie. Reliability of flat XOR-based erasure codes on heterogeneous devices. In *Proceedings of the 2008 Int'l Conference on Dependable Systems and Networking (DSN 2008)*, pages 147–156, June 2008.

[15] K. M. Greenan, J. S. Plank, and J. J. Wylie. Mean time to meaningless: MTTDL, Markov models, and storage system reliability. In *Proceedings of the 1st Workshop on Hot Topics in Storage and File Systems (HotStorage '10)*, 2010.

[16] N. Greenfieldboyce. Houston, we erased the Apollo 11 tapes. National Public Radio, `http://www.npr.org/templates/story/story.php?storyId=106637066`, July 2009.

[17] W. Jiang, C. Hu, Y. Zhou, and A. Kanevsky. Are disks the dominant contributor for storage failures? In *Proceedings of the 6th USENIX Conference on File and Storage Technologies (FAST)*, Feb. 2008.

[18] P. Maniatis, M. Roussopoulos, T. J. Giuli, D. S. H. Rosenthal, and M. Baker. The LOCKSS peer-to-peer digital preservation system. *ACM Transactions on Computer Systems*, 23(1):2–50, 2005.

[19] S. Nath, H. Yu, P. B. Gibbons, and S. Seshan. Subtleties in tolerating correlated failures in wide-area storage systems. In *Proceedings of the 3rd Symposium on Networked Systems Design and Implementation (NSDI)*, 2006.

[20] A. Oprea and A. Juels. A clean-slate look at disk scrubbing. In *Proceedings of the 8th USENIX Conference on File and Storage Technologies (FAST)*, Feb. 2010.

[21] B. Panzer-Steindel. Data integrity. CERN/IT, 2007.

[22] D. A. Patterson, G. Gibson, and R. H. Katz. A case for redundant arrays of inexpensive disks (RAID). In *Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data*, pages 109–116. ACM, 1988.

[23] E. Pinheiro, W.-D. Weber, and L. A. Barroso. Failure trends in a large disk drive population. In *Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST)*, Feb. 2007.

[24] D. S. H. Rosenthal. Keeping bits safe: How hard can it be? *Communications of the ACM*, 53, Nov. 2010.

[25] 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 *Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST)*, pages 1–16, Feb. 2007.

[26] T. J. E. Schwarz, Q. Xin, E. L. Miller, D. D. E. Long, A. Hospodor, and S. Ng. Disk scrubbing in large archival storage systems. In *Proceedings of the 12th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '04)*, pages 409–418, Oct. 2004.

[27] A. L. Shimpi. The SandForce roundup: Corsair, Kingston, Patriot, OCZ, OWC & MemoRight SSDs compared. AnandTech, Aug. 2011.

[28] M. Storer, K. Greenan, E. L. Miller, and C. Maltzahn. Potshards: Storing data for the long-term without encryption. In *Proceedings of the 3rd International IEEE Security in Storage Workshop*, Dec. 2005.

[29] M. W. Storer, K. M. Greenan, I. Adams, E. L. Miller, D. D. E. Long, and K. Vorugant. Logan: Automatic management for evolvable, large-scale, archival storage. In *Proceedings of the 3rd Petascale Data Storage Workshop (PDSW '08)*, Nov. 2008.

[30] M. W. Storer, K. M. Greenan, E. L. Miller, and K. Voruganti. Pergamum: Replacing tape with energy efficient, reliable, disk-based archival storage. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies (FAST)*, Feb. 2008.

[31] T. A. W. S. Team. Summary of the Amazon EC2 and Amazon RDS service disruption in the US East Region. Amazon Web Services, `http://aws.amazon.com/message/65648/`, Apr. 2011.

[32] R. Weisman. Data backup firm sues 2 hardware suppliers. The Boston Globe, Mar. 2009.

[33] L. L. You, K. T. Pollack, D. D. E. Long, and K. Gopinath. PRESIDIO: a framework for efficient archival data storage. *ACM Transactions on Storage*, 7(2), July 2011.