Long-term View of the Reliability of Archival Storage Systems

First Step Towards a Reliability-Aware Storage System Manager

Yan Li, Ethan L. Miller
yanli@ucsc.edu, elm@cs.ucsc.edu
Storage Systems Research Center
University of California, Santa Cruz

***Abstract*—In order for data to be stored reliably for a long time, their reliability must be under continual monitoring, because the storage and outer environment are always changing. Current practices for improving data reliability normally boil down to creating more copies, with simple quantitative analyses that either failed to tolerate the complexity of real world storages or ignored the long term impact of large scale events such as the earthquake. This paper proposes a systematic approach to calculate and monitor the reliability of data during their whole lifetime, and discusses how the output of this model can be used to improve data's reliability.**

I. Introduction

As computer systems are taking more and more responsibilities in critical processes, the yearning for a better understanding of the system's reliability is ever increasing. From time to time we hear high profile data loss accidents, from NASA's missing Apollo project tapes that contained the original footage of the Apollo 11 moonwalk [9], online backup provider Carbonite's 2004 accident [15] that damaged over 7,500 customer's data, to the most recent accident of cloud computing provider Amazon's loss of about 0.07% data in one of its Availability Zones within the US East Region in 2011 [14]. These accidents simply mean that our understanding of storage systems' reliability is not good enough.

In the real word, many storage system administrators are still taking the "greedy" approach of "just making more copies," without understanding how reliable the data really are. Because data are normally distributed or backed up to several storage systems of different characters, as shown in Figure 1, and there's no easy way to understand the reliability of data in such a complex solution.

Moreover, storage systems and devices change over time: broken parts will be replaced, new expansion modules will be installed if more spaces are needed. Traditional reliability studies focused on understanding devices and systems, but there's no established way to calculate how safe the data are in a complex real world storage solution. In the rest part of the paper we will use "reliability of data" to mean the safety of data.

Generally speaking, factors that contribute to the evolution of a storage solution into a heterogeneous system include but are not limited to:

- Technology obsolescence: old vendors may go away, spare parts for old devices are no longer produced.
- Leveraging new technology for better performance and cost reduction: new storage products are introduced to the market everyday, therefore it's natural for the end-user to pick up the most cost-effective model when replacing old parts. [11]
- Resource constraints: budget, power, rack space, staff, etc.

Understanding the reliability of data is important because we all know it's a big problem if the actual safety of data is lower than expectation, but too high a safety is also a problem, which means money is wasted on unnecessary devices. The reliability of data might be much lower than what people perceives, as we illustrate in the study: older devices have much lower reliability than new devices, therefore if a very important data object happened to be stored in two old devices, it's reliability maybe very low even though there are replicas. Additionally, rare but large scale events such as earthquake can also have a great impact on the reliability of data.

This problem is especially important for archival storage systems. Although archival storage systems are designed to run for a very long time, many new solutions and algorithms are being proposed every year, as it is still a young and hot research area. Therefore in this background, institutions that deploy these kind of archival storage systems expect to upgrade and expand their installed systems from time to time, or deploy new systems when they become available.

This paper proposed a systematic way to solve this problem, 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 reliability of data objects stored in many storage systems at the same time. The contribution of this paper includes:

1) A model to calculate the reliability of data objects stored on many storage systems, this model is practical enough for being used in real world data center management
2) Using 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 earthquake, on the reliability of data
5) Using the result from this model to reduce the cost

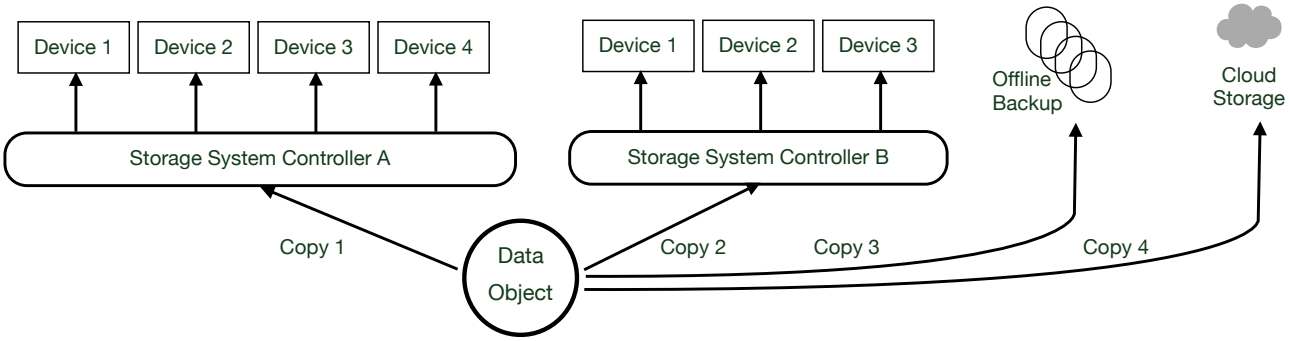

Fig. 1. A Heterogeneous Storage Solution

of storage systems and achieve better reliability by improving data layout and scrubbing algorithm

The rest of the paper is organized as follows: section II introduces the background of reliability analyses and archival storage system; section III introduces the model, from the metric we picked (III-B) to modeling the aging of devices (III-D) as well as large scale events (III-E). For each category of events, we discuss its impact to reliability, how it can be modeled and how to get statistic data to support the model from empirical events data collected from the field. In section IV we discuss the application of the model and the need to build a reliability-aware storage system manager.

II. Background

In the storage system research area, there is a rising trend to take reliability of storage more and more seriously due to regulations like Sarbanes-Oxley Act of 2002, which demands that important business data must be retained for a period of time. Emerging cloud storage vendors promote the shift of the burden of curating data from clients to them as a key competition advantage over old in-door storage solutions. Moreover, understanding the reliability of data is especially important for archival storage systems used in digital preservation, because the most important design goal for an archival storage system is to ensure the survival of the data for a very long period of time, or forever. The whole idea of archival storage would be meaningless if the reliability of the data can't be guaranteed.

Since Patterson and Gibson's work on RAID [10], MTTDL has been widely used in both the research and industry as a standard metric for analyzing the reliability of storage systems. However, as Greenan et al. [8] pointed out that MTTDL is an expectation of time to fail over an infinite interval, which is good for quick, relative comparison, but not very meaningful for understanding the real reliability of data. To address this issue, he proposed to use NOmalized 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 usability capacity. The importance of this study is that it brought the size of data into the study and calculation of reliability.

At the lowest level of a storage solution lies the devices. Traditional rotating magnetic platter hard drive is a complex system and lots of studies revealed many characters of its reliability. The work of Pinheiro et al. [11] and Schroeder et al. [12] provides real world data of the hard drive's failure pattern.

Up to the system level, Markov models are used widely for modeling, and it is suitable for analyzing a system if status of the system can be precisely defined. Within this category, a lot of study has been done using both analytical and simulation methods [8], [6], [2], [5].

III. The Model

A. Terminology and Assertions

The following terms are used in this paper:

Data object (DO) The smallest unit of information. A DO is itself meaningful and can't be broken down further. Therefore changing one bit of a DO can destroy it. In file based systems, files can be directly treated as DOs.

Device A physical device that stores bits. Common storage devices include hard drive, solid state drive (SSD), portable USB thumb drive, 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 to 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.

Device property A physical property of a device that can be measured and tracked. For a rotating-plate 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 hours, 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 unit plus one or more devices. An online storage service provider is also seen as a storage system in this paper, just it doesn't have any physical device that we can track. Storage system is abbreviated as "system" in most cases in this paper.

Storage solution A storage solution (or just solution) consists of storage systems. It is the whole solution a company

(or an individual) deploys to store its data. As we have talked in the introduction section and shown in Figure 1, most solutions consist of several storage systems and are always changing.

Time Time in this paper flows continually from 0. The variable's unit is *hour* in practical calculation 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 impenetrable world, but in order to make our model useful for guiding the management of DOs and systems, we make this assertion for the calculation of reliability: we always aim at the theoretical lower bound. Instead of trying to get a precise reliability by using complex simulation models, we decide to use analytical model to get the theoretical lower bound of reliability. This is because the purpose of calculating the reliability in this paper is to guide the solution design and deployment. It's not bad if the system performs better than the model predicted. But it will be a disaster if the model predicts your data is safe but the system falls apart unexpectedly. 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 reliability.

We also have to point out that even though we are aiming at "lower bound" there's no way to guarantee that we can get it, because the more threats are considered, the lower the calculated reliability is, and it is impossible to cover all threats. That's why 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 we use Data Object (DO) as the basic unit of stored digital data. As described above, DO is the smallest unit of data that preserves the meaning and further diving a DO is meaningless. Therefore even changing one bit of a DO corrupts it.

B. Data Object's Reliability

From end-user's point of view, it's nature that users care more about their data rather than the life of a storage system. If a storage system goes down, as long as the data is backed up somewhere else, the user just need to fix or replace the broken system. On the contrary, loss of a data object might be a big threat to the user's business goals.

However, in literature, the reliability of a data object is often treated as equal to the reliability of a storage system, as we can see in the introduction and background section. In most cases, that's not true.

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

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

Mathematically, it can be expressed as:

$$\mathrm{DOR}(t) = Pr\{T > t\} = \int_t^\infty f(x)\,\mathrm{d}x \tag{1}$$

$f(x)$ is the failure probability density function and t is the length of the period of time (which is assumed to start from time zero).

If the user has only one copy of an object stored in one storage system, then the *upper bound* of the reliability of the data object is the reliability of that storage system. It's an upper bound because you can't expect the object to survive if the storage system fails. However, it's also wrong to assume that this upper bound can be achieved because there are many events that can destroy the storage device such as device losses and earthquake that the storage system's designers won't take into consideration when calculating their system's reliability, even though they all contribute to the lowering the DOR to below the storage system's reliability. Section III-E discusses the modeling of large scale events.

When one storage system cannot meet the always increasing demand of the user, it will be expanded or new systems will be deployed along with the old system. In this process, data objects will be migrated from the old system to the expanded or new system, and they will reside on more than one storage systems. Assume that the DO resides on n storage systems, and the cumulative distribution function (CDF) of these systems' failure rates are expressed by function $F_1(t), F_2(t), \cdots, F_n(t)$. The DOR can be expressed as a function of them:

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

In the simplest form, the failure of these systems are uncorrelated (we will discuss large scale events that can affect more than one systems in section III-E later), and the object will only be lost if *ALL* of these n storage system fail. The CDF for this event can be expressed as:

$$F_n(t) = \prod_{i=1}^{n} F_i(t) \tag{3}$$

And the DOR can be expressed as

$$\mathrm{DOR}_n(t) = 1 - \prod_{i=1}^{n} F_i(t) \tag{4}$$

$$= 1 - \prod_{i=1}^{n} \Big(1 - R_i(t)\Big) \tag{5}$$

Because, for a device, the CDF of failure rate $F(t)$ and the reliability (a.k.a. survival rate) $R(t)$ have this simple relation: $F(t) = 1 - R(t)$, in the following discussion we will use $F(t)$ or $R(t)$ depends on the simplicity of equation.

In next section, we will discuss how to find these functions for a given system.

C. Reliability Transition Function

Consider the storage systems' configuration shown in Figure 1, the DOR 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 a function called Reliability Transition Function (RTF) to denote this relationship. Suppose the reliability of the underlying devices are $R_{\mathrm{dev1}}(t), R_{\mathrm{dev2}}(t), \cdots, R_{\mathrm{devN}}(t)$, RTF can be defined as:

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

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, other devices will be ignored by RTF.

RTF describes the reliability of a single storage system and it can use either analytical or simulation method to implement. Greenan et al. [7] shows 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, the precise reliability of a system can't be expressed in the close form, but instead the simulation must be run for each point in time. In practice, it's not unusual that the DOR of millions objects have to be tracked and calculated, therefore using simulation method may not be possible.

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 DOR. We proposed that before a new storage system is purchased and deployed, the user should require the system vendor to provide RTF for this specific device. Even better, the vendor can provide two or more RTFs, one of them can use simulation method and be precise and the other can be fast using some form of approximation.

Here's a sample showing the RTF of a DO stored on an erasure code based system that divides the object into m fragments and recode them into n copies ($n > m$). This category covers 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 DOR has not becoming much harder with this dynamic view because we are only aiming at the lower bound. We state that the RTF for an n/m erasure code based system can be expressed as:

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

$R_k(t)$ is the $n - m + 1$ devices that have the lowest $R(t)$. In order to calculate the value of DOR, the n devices will be sorted according to their $R(t)$ and only the lowest $n - m + 1$ of them are used in the above equation.

Should de-duplication is used in the storage system, reliability of DO will be affected. Due to the normally proprietary nature and subtleties in implementation, 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 system therefore we can use some simple form of the equations. If the precise reliability is expected, simulation based methods might have to be used.

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

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

Now let's continue to the lower level of the storage solution and take a look at how to get a good expression of $R_{\mathrm{dev}}(t)$.

D. Modeling Devices

Among the reasons that lead to the failures of devices in a data center, aging is the biggest contributor, and there are extensive analyses of hard drive's reliability [12], [11], [4]. A Weibull reliability model gives good results in recent researches [12], [5] and it reflects both failures from infancy 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*.

In a simple example, the devices used for storing objects are hard drives and the DOs are mirrored among them as in a RAID1 system. Without considering the bit rot events, apply equation (5) and we get the DOR for a DO resides on n hard drive:

$$\mathrm{DOR}_n(t) = 1 - \prod_{i=1}^{n} \Big(1 - e^{-(\frac{t}{\eta})^{\beta}}\Big) \quad (9)$$

Figure 2 illustrates the effect of device aging. The solid blue curve shows DOR of an object stored in a two-HD mirrored RAID1 system using two new hard drives, and for comparison, the dashed red line curve shows the DOR if one of these two hard drives is a four year old one at the beginning of the experiment.

The reliability of equation (9) is only an upper bound that can never be reached because we have yet considered many other failure events.

It is possible to use more complex methods than that of equation (5) to get a more precise DOR, as proposed by Greenan in [6], but they are much more harder to model than the analytical model used here, because in order to use the Markov model, the time to recover from a system failure must be known or following a known possibility distribution. For some well engineered storage system, this estimation is possible. However, if a whole storage system in a company's data center goes down, normally the system admin has to rely on the vendor's customer support staff to diagnose and repair the failed system, and the time of which is nigh impossible to predict. The point here is that 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 reliability of DOs stored in more than one system.

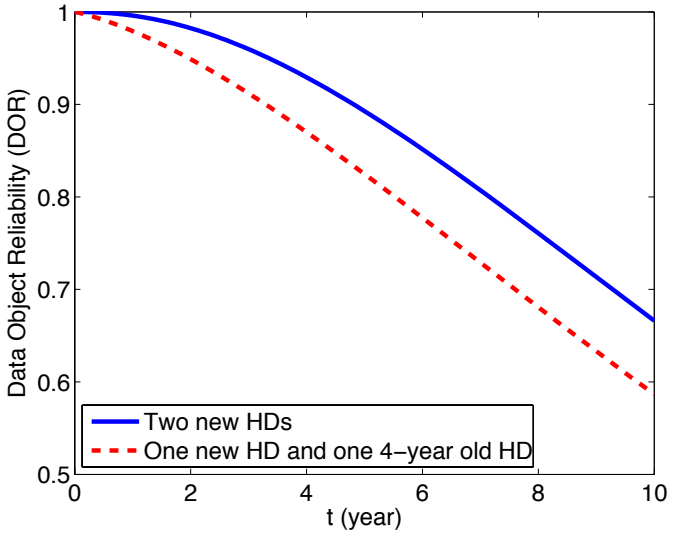

Fig. 2. DOR of object stored in mixed old and new hard drives, Weibull model, $\beta = 1.12$, $\eta = 100,000$ hr

E. Modeling Events

"Event" is a thing that happens and is interesting to the study of systems and devices' reliability. We observe and study events and how they change systems and devices. This helps us to understand the storage solution during it's whole lifetime. We classify events into two categories: failure events and operational events. Failure events include device failures, natural disasters, device losses and bit rot; operational events include normal and abnormal operations of devices and systems that affect their life, such as power cycle, disk head load/unload, surface scan error and erasing a block of SSD.

1) Failure events: The failures of devices have been studied extensively in literature and are covered by the reliability CDF functions we covered above. As such, 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 earthquake, fire, flood, military actions, etc. Among them, earthquakes are a relatively well studied and powerful disaster that can easily destroy the whole building where all the devices are located. Whether the earthquake at a location is a memoryless event or not is still debatable in the academia, but for analyzing the risk of future earthquake, it's enough to treat earthquake as events in a memoryless Poisson process [3]. 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 happens. 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 equations (3) and the possibility distribution of earthquake, we can get the DOR when we take the impact of earthquake into consideration:

$$\mathrm{DOR}_n(t) = 1 - \big(F_n(t) + F_{eq}(t) - F_n(t) \times F_{eq}(t)\big) \quad (10)$$

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 it's impact, let's use the sample configuration we have discussed above in section III-D, a two-HD mirrored RAID1 system stored in one building that can be destroyed by a single earthquake, and assume these devices are located somewhere in California, USA. 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 earthquake is shown in Figure 3.

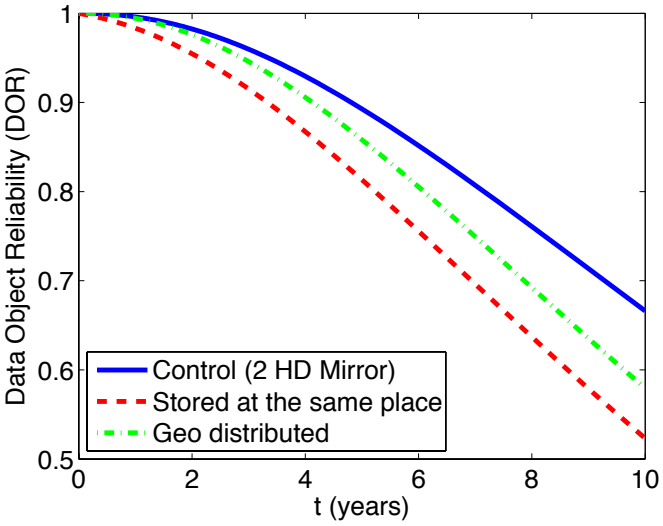

Fig. 3. Impact of earthquake on DOR

The solid blue line curve shows the DOR without considering the impact of the earthquake, and the dashed red curve shows the result of equation (10) after considering the impact of earthquake. To mitigate the threat of earthquake, the most straightforward practice is to deploy the storage systems in a geologically distributed way, such as one storage server in San Francisco and another one in New York. Then we can assume one 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 failed
2) Hard drive A failed and B destroyed by earthquake at New York
3) Hard drive A destroyed by earthquake at California and B failed
4) Both hard drives are destroyed by two earthquakes

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

To generalize the equation into calculating the DOR when we consider m kinds of events that can wipe out all storage devices, we can use the general case for inclusion-exclusion

principle and written it in this closed form:

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

2) Operational events: Besides events that can destroy the whole storage systems, more often we see smaller events that are not that serious. 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 affect the reliability of the storage device in a previous study [11].

S.M.A.R.T. collects a plethora of raw information, so first we should try to identify those that are useful for our analyses, then we should use them to define some polices, hoping this can reduce the cost and/or improve the precision of the calculated system reliability.

One of the events that S.M.A.R.T. records is "Scan Error". Modern hard drive scans the disk surface during idle time. Getting a "Scan Error" doesn't mean the drive is broken or data is lost. In fact, if the S.M.A.R.T. data of a hard drive is not constantly monitored by some process in the operating system, these kind of event may never be discovered. Large number of "Scan Error" events indicates surface defects and are believed to lower the predicted device reliability. Pinheiro et al. [11] found that the group of drives with scan errors are ten times more likely to fail than the group with no errors. Using this data, if we know the CDF of a hard drive is $F(t)$ and we observe a Scan Error event through it's S.M.A.R.T. interface, we know we should adjust it's reliability to $1/10$ of the previous value. Hence it's new CDF is:

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

It is worth noting that according to previous studies by Pinheiro et al., S.M.A.R.T. data alone can't be used effectively to predict future failures [11]. In this paper's context, we are calculating the lower bound of DOR, and in this case, S.M.A.R.T. events can be a good indicator since there's high enough 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 DOR. It's formal mathematical definition can be expressed as: say we know event K's effect on the device's reliability can be calculated by using function $E_k()$, and the CDF of the device is $F(t)$, then the new CDF of the device after event K happens is:

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

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

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

Equation (13) can be combined with our previous DOR equation (8) so we get:

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

Equation (14) is our final equation for computing DOR and it covers both the aging of devices and two categories of events' impact to the DOR.

3) Events that Shouldn't Be Included: The nature of analyzing the DOR, which are stored on more than one storage systems, leads to the reconsideration of some common events used in reliability 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 III-C.

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 DOR.

F. Tracking events

With the model of events, the important task is to track them. By incorporating the knowledge of every event happened we can have a better understanding of the DOR.

Suppose we have a DO that is stored in two (or three) systems, and at time t_1 one of the systems failed. Suppose the system's reliability follows Weibull distribution. The DOR 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 DOR look like? We should aware that if the newly installed system is a brand new system, it's reliability function should take $t - t_2$ as parameter. Therefore the DOR after the new system is installed is:

$$\mathrm{DOR}(t) = 1 - F_1(t) * F_2(t - t_2)$$

And the graph of it is shown in Figure 4 too.

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

G. Learn the failure pattern

A key factor for getting a correct DOR lies in getting the correct lower bound of reliability of a devices (the $R(t)$ function). In previous samples we used the empirical value for the Weibull distribution parameters. However, other research on the failure pattern of large amount of hard drives shows

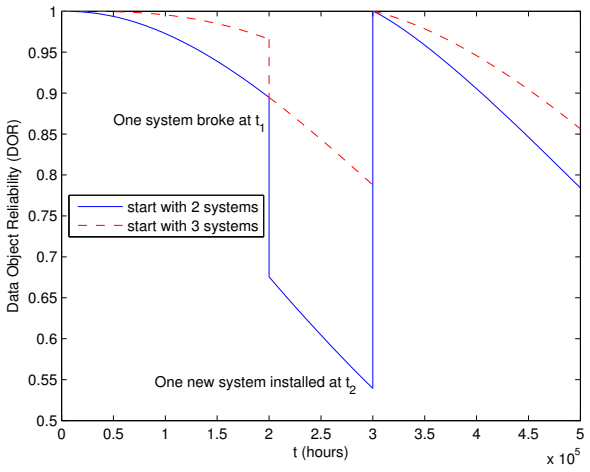

Fig. 4. Effect of one system failure and repaired later

that hard drive's reliability differs greatly from one brand to another, or even from one shipment to another [11]. In this section we propose that data can be gathered from the field during a storage system/device's life time to fine tune our calculation of DOR.

When a storage system is deployed and the initial DOR is to be calculated, we can normally get the MTTF value straight from a device's specification. However, in most cases the vendors failed to specify what kind of failures they considered when calculating "Mean Time to Failure", and what's more irksome is that in the field the replacement rate of hard drives is generally much higher than the value calculated from the vendor's MTTF [12].

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 shipment because previous studies show hard drives from the same shipment show similar failure patterns. Let N be the count of failures we observed, 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:

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

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.

Here we demonstrated how we calibrate the $R(t)$ for rotating platter hard drive. Similar analysis can also be done for other storage devices such as NVM-based devices, and we just need the different CDF and initial empirical value.

IV. Application and Future Work

This model has laid the foundation for future work. We are currently studying the possibility of designing a smart data layout algorithm. Greenan [7] proposed that when designing erasure code-based system and heterogeneous devices with different reliabilities are mingled, the reliability of data varies among different layout algorithms. Similar phenomena also exists when not only heterogeneous devices but also heterogeneous systems are deployed. Therefore one of our future study goals is a more general "reliability-aware layout algorithm" which not only considers the reliability 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 reliability, we can store the high priority DOs to new devices, which are supposed to have better reliability. Less important DOs will be kept on old devices or with fewer replicas. Because old devices can be used safely because we know important data won't be stored on them, this layout algorithm also has the benefit of reducing cost of storage systems.

Another application is adjusting scrubbing (Schwarz et al. [13]) interval according to DOR. Scrubbing is very important for preventing bit rot in archival storage system. But too much scrubbing is also harmful. Therefore intuitively shorter scrubbing interval is needed for aging devices and systems. Understanding the DOR helps us fine tuning the scrubbing interval for each device.

We are also planning to build a Reliability-Aware Storage System Manager. 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, the current designs haven't taken reliability into consideration. To let the user have a better understanding of the DOR, an interface that can display the graph of DOR is needed. When all the methods we proposed above are used in the calculation, even getting the graph of one DOR 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 "Reliability Monitor" should implement the algorithms we have discussed in previous chapter. It monitors and collects field data from each device in use, send them through Reliability Transition Functions of Storage Systems and apply events' probabilities, and finally generates a continual updating "Reliability View", which can be used by the user to monitor the dynamic changes of reliability of data objects. When an event occurs, the event data will be automatically picked by the Reliability Monitor from the device if they are device-related events, or input by system admin if they are external events and the reliability view will be updated in real time.

It is also possible to let the Reliability Monitor handle future events. For example, when one data center is planned to be taken offline and transport to somewhere else, the possibility for device damage during transportation is much higher than when they are kept under a roof. Therefore this planned action and related events should be inputted into the Reliability Monitor as part of the planning process, to ensure the DOR is kept at the expected level during the whole transportation duration.

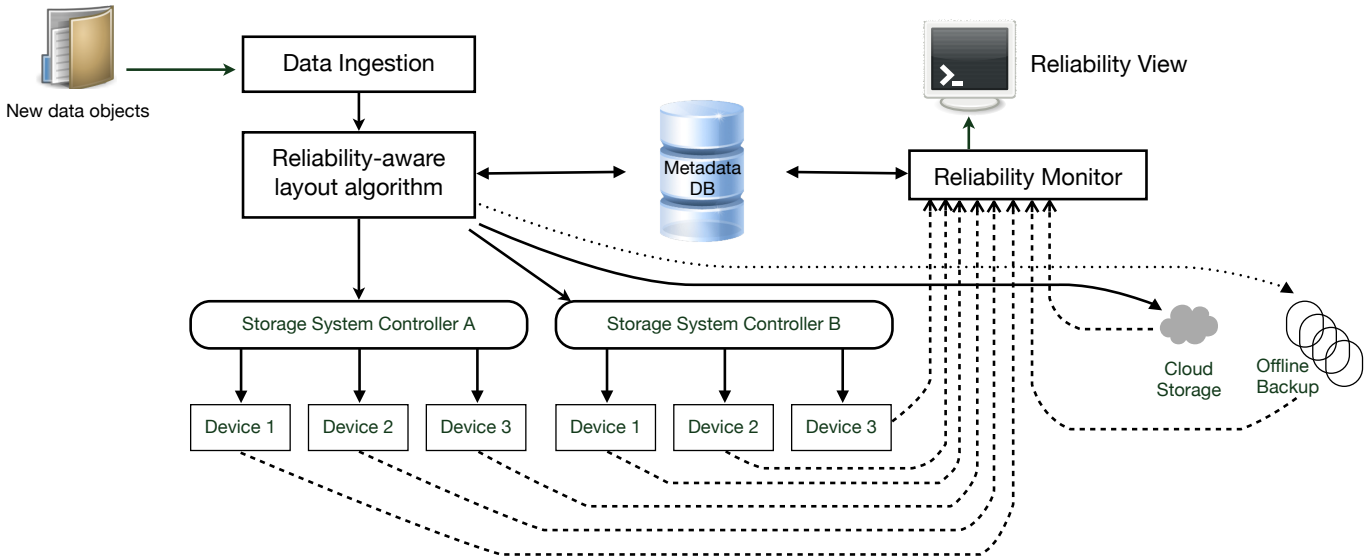

Fig. 5. Reliability-aware Storage System Manager

V. Conclusion

The conclusion of this research is that for end-users, understanding the reliability of data is more important than understanding the reliability of systems or devices. The model proposed in this paper is a systematic approach for calculation the Data Object's Reliability. Also a large amount of device information, which should be useful in understanding the storage systems is lost everyday because there is no established effort on gathering and processing them. We propose to build a reliability-aware storage system manager to gather and use these information for better understanding the reliability of data.

Acknowledgments

We would like to thank the faculty and students of the Storage Systems Research Center for their help and guidance. Support for this research was provided by SSRC industrial partners, including Engenio, Hewlett Packard, IBM, Intel, Microsoft, Network Appliance, Rocksoft, Veritas, and Yahoo!.

References

[1] Sinan O. Akçiz, Lisa Grant Ludwig, J Ramon Arrowsmith, and Olaf Zielke. Century-long average time intervals between earthquake ruptures of the San Andreas fault in the Carrizo Plain, California. 38:787–790, September 2010.

[2] Alvin M. Blum, Ambuj Goyal, Philip Heidelberger, Stephen S. Lavenberg, Marvin K. Nakayama, and Perwez 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.

[3] Yousef Bozorgnia and Vitelmo Victorio Bertero. *Earthquake engineering: from engineering seismology to performance-based engineering*. CRC Press LLC, 2006.

[4] Jon 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.

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

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

[7] Kevin M. Greenan, Ethan L. Miller, and Jay 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.

[8] Kevin M. Greenan, James S. Plank, and Jay 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.

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

[10] David A. Patterson, Garth Gibson, and Randy 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.

[11] Eduardo Pinheiro, Wolf-Dietrich Weber, and Luiz André Barroso. Failure trends in a large disk drive population. In *Proceedings of the 5th USENIX Conference on File and Storage Technologies (FAST)*, February 2007.

[12] Bianca Schroeder and Garth 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, February 2007.

[13] Thomas J. E. Schwarz, Qin Xin, Ethan L. Miller, Darrell D. E. Long, Andy Hospodor, and Spencer 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, October 2004.

[14] The Amazon Web Services 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/, April 2011.

[15] Robert Weisman. Data backup firm sues 2 hardware suppliers. The Boston Globe, March 2009.