A Fresh Look at the Reliability of Long-term Digital Storage

Mary Baker
HP Labs

Mehul Shah
HP Labs

David S. H. Rosenthal
Stanford University Libraries

Mema Roussopoulos
Harvard University

Petros Maniatis
Intel Research Berkeley

TJ Giuli[*]
Ford Research and Advanced
Engineering

Prashanth Bungale
Harvard University

ABSTRACT

Emerging Web services, such as email, photo sharing, and web site archives, must preserve large volumes of quickly accessible data indefinitely into the future. The costs of doing so often determine whether the service is economically viable. We make the case that these applications' demands on large scale storage systems over long time horizons require us to reevaluate traditional system designs. We examine threats to long-lived data from an end-to-end perspective, taking into account not just hardware and software faults but also faults due to humans and organizations. We present a simple model of long-term storage failures that helps us reason about various strategies for addressing some of these threats. Using this model we show that the most important strategies for increasing the reliability of long-term storage are detecting latent faults quickly, automating fault repair to make it cheaper and faster, and increasing the independence of data replicas.

Categories and Subject Descriptors

E.5 [**Data**]: Files—*backup/recovery*; D.4.5 [**Software**]: Operating Systems—*reliability*; H.3.7 [**Information Systems**]: Digital Libraries—*systems issues*

General Terms

Design, Management, Reliability

Keywords

Digital Preservation, Storage Systems

---

[*]Work performed while at Stanford University

1. INTRODUCTION

In this paper we make the case that long-term reliable storage presents challenges that differ from traditional problems in the storage literature, warranting increased interest from systems and storage researchers.

Frequent headlines remind us that bits, even bits stored in expensive, professionally administered data centers, are vulnerable to loss and damage [20,22] [1]. The vulnerabilities grow when large volumes of data must be stored indefinitely into the future, as required by emerging web services such as e-mail (e.g., Gmail)[2], photo sharing (e.g., Ofoto), and archives (e.g., The Internet Archive). The economic viability of these services depends on storing data at low cost, while acceptance by their customers depends on their keeping data unaltered and accessible with low latency.

Doing so over long periods of time would be easy if fast, cheap, reliable disks were available, and if threats to the data were confined to the storage subsystem. Unfortunately, neither is true. The economics of high-volume manufacturing provide a choice between consumer-grade drives, which are cheap, fairly fast, and fairly reliable, and enterprise-grade drives, which are vastly more expensive, much faster but only a little more reliable (§6.1). For short-lived data, current levels of drive reliability might not pose a problem, but for long-lived data, faults are inescapable. Further, long-term storage faces many threats beyond the storage system: obsolescence of data formats, malicious attacks, and economic and structural volatility of the host organizations.

To make our case, we start by motivating the need for *digital preservation* – storing immutable data over long periods (§2). We then list the threats to data survival using examples from real systems (§3), and examine mismatches between the design philosophy of many current storage systems and these threats (§4).

To explore the implications of this problem, we introduce a simple reliability model (§5) of long-term replicated storage systems. Though inspired by the reliability model for RAID [38], we take a more end-to-end – rather than device-

---

[1]All URLs in the references were verified 13 March 2006. Except for [25, 30, 33, 41, 48, 51, 52], their pages were also archived at the Internet Archive Wayback Machine (`http://www.archive.org/`).

[2]All trademarks mentioned in this paper belong to their respective owners.

oriented – approach and address a wider range of faults. Our model explicitly incorporates both *latent faults*, which occur long before they are detected, and *correlated faults*, when one fault causes others or when one error causes multiple faults (§4).

Our model is simplistic but useful; it highlights needed areas for gathering reliability data, and helps to evaluate strategies for improving long-term storage reliability (§6). For example, we would like to be able to answer questions such as: Do latent faults occur frequently enough that we need to worry about them? (Yes, §5.4.) How often should we search for latent faults if doing so can itself cause damage? (The increase in fault rate must be balanced by a quadratic reduction in the time to detect and repair the faults, §5.4.2.) Is it better to increase the mean time between visible faults or between latent faults? (Perhaps neither if it significantly decreases the other, §5.4.) Is it better to increase replication in the system or increase the independence of existing replicas? (Both, but replication without increasing independence does not help much, §5.5.) Some of these strategies have been proposed before (§8), but we believe they are worth revisiting in the context of long-term storage.

We conclude (§9) that the most important strategies for increasing the reliability of long-term storage are detecting latent faults quickly, automating repair to be faster and cheaper, and increasing the independence of replicas at all levels. Our analysis and conclusions should motivate a renewed look at the design of storage systems that must preserve data for decades in volatile and even hostile environments.

2. THE NEED FOR PRESERVATION

Preserving information for decades or even centuries has proved important. Shang dynasty (12th century BC) Chinese astronomers inscribed eclipse observations on "oracle bones" (animal bones and tortoise shells). About 3200 years later researchers used these records, together with one from 1302BC, to estimate that the accumulated clock error was just over 7 hours, and from this derived a value for the viscosity of the Earth's mantle as it rebounds from the weight of the glaciers [37].

Longitudinal medical studies depend upon accurate preservation of detailed patient records for decades. In 1948 scientists began to study the residents of Framingham, Massachusetts [14] to understand the large increase in heart disease victims throughout the 1930s and 40s. Using data collected over decades of research, scientists discovered the major risk factors that modern medicine now knows contribute to heart disease.

In 1975, the former USSR sent probes Venera 9 and 10 to the surface of Venus to collect data and imagery. The low quality images attracted little interest. About 28 years later, an American scientist used modern image processing algorithms on the diligently preserved data to reveal much more detail [57].

These timescales of many decades, even centuries, contrast with the typical 5-year lifetime for computing hardware and digital media. Beyond just scientific data, legislation such as Sarbanes-Oxley [2] and HIPPA [1] require many organizations to keep electronic records over decades. Consumers used to analog assets such as mail and photographs, which persist over many decades, are now happily entrusting their digital versions to online services. The associated marketing literature [17] encourages them to expect similar longevity.

3. THREATS TO PRESERVATION

In this section we list threats to long-term storage and provide real examples to motivate them. While some apply also to short-term storage, others are unique to long-term preservation (e.g., media obsolescence).

Large-scale disaster. During the life of archival data we must expect large-scale disasters (floods, fires, earthquakes, acts of war). Such disasters typically trigger other types of threat, such as media, hardware, and organizational faults, as was the case with many data centers affected by the 9/11 attack.

Human error. Users or operators often accidentally delete content they still need, or purposefully delete data for which they later discover a need. Sometimes the errors affect preservation hardware (e.g., losing tapes in transit [41]), software (e.g., uninstalling a required driver), or infrastructure (e.g., turning off the air-conditioning system in the server room or swapping out the wrong disk in an array that has suffered a disk failure). Human error is increasingly the cause of system failures [40].

Component faults. Taking an end-to-end view of a system, any component may fail. Hardware components suffer transient recoverable faults (e.g., temporary power loss) and catastrophic irrecoverable faults (e.g., a power surge fries a controller card). Software components, including firmware in disks, suffer from bugs affecting stored data. Ingestion of data into a preservation system over the network may itself fail. External license servers or the companies that run them might no longer exist decades after an application and its data are archived. Domain names will vanish or be reassigned if the registrant fails to pay the registrar, and a persistent URL [36] will not resolve if the resolver service fails to preserve its data with as much care as the storage system client.

Media faults. The storage medium is a vital component. No affordable digital storage media are completely reliable over long periods of time. They are subject to gradual accumulation of irrecoverable bit errors – often called *bit rot* – and to sudden irrecoverable loss of bulk data such as disk crashes [49].

Bit rot is particularly troublesome, because it occurs without warning and might not be detected until it is too late to make repairs. A familiar example might be CD-Rs. Manufacturers claim lifetimes up to 100 years, but even when properly stored actual lifetimes may be only 2 to 5 years [21, 31]. Similarly, a previously readable disk sector can become unreadable or may be readable but contain the wrong information due to firmware bugs or misplaced sector writes [5].

Media/hardware obsolescence. Over time, media and hardware components can become obsolete – no longer able to communicate with other system components – or irreplaceable. This problem is particularly acute for removable media, which though readable may have outlived any suitable reader device [26]. 9-track tape and 12-inch video laser discs are typical examples. The recently ubiquitous PC floppy is no longer part of industry specifications and will shortly share their fate.

Software/format obsolescence. Software obsolescence is similar, often manifested as *format obsolescence*: the bits in which the data were encoded remain accessible, but the

information can no longer be correctly interpreted. Proprietary formats, even popular ones, are equally vulnerable. For instance, digital camera companies have proprietary, often undocumented "RAW" formats for recording camera data. When a company ceases to exist or to support its format, photographers can lose valuable data [51].

Loss of context. Metadata, or more generally "context," includes information about the subject and provenance of content, the layout, location, and inter-relationships among stored objects, and the processes, algorithms and software needed to manipulate them. Preserving context is as important as preserving the actual data, and it can even be hard to recognize all required context in time to collect it. Encrypted information is a particularly challenging example, since the decryption keys must be preserved as well as the encrypted data. Unfortunately, over long periods of time, secrets (such as keys) can get lost, leak, or get broken [15]. Short-term storage applications are less vulnerable; their assets rarely live long enough for the context to be lost or the information to become uninterpretable.

Attack. Traditional repositories are subject to long-term malicious attack, and there is no reason to expect their digital equivalents to be exempt. Attacks include destruction, censorship, modification and theft of repositories' contents, and disruption of their services [52]. The attacks can be short-term or long-term, legal or illegal, internal or external. They can be motivated by ideological, political, financial or legal factors, by bragging rights or by employee dissatisfaction. Short-term storage is also vulnerable, but typically to the better studied abrupt, intense attacks rather than slowly subversive attacks. Because much abuse of computer systems involves insiders [25], a digital preservation system must anticipate attack even if it is completely isolated from external networks.

Examples of attacks include US government websites being "sanitized" to match the administration's world view [33, 50]. "Operation Cancelbunny" is an example of an attack motivated by religion. Followers of the Church of Scientology launched attacks against a Usenet newsgroup by canceling articles that were critical of the church [3]. Yet another example was George Goble's immensely popular web site about lighting barbecues with liquid oxygen, an activity for which he won the 1996 Ig Noble prize for chemistry. In early 2003, Goble was asked by the authorities to remove this web site [16]. Incomplete versions of the site can still be found at the Internet Archive.

Organizational faults. A system view of long-term storage must include not merely the technology but also the organization in which it is embedded. These organizations can die out, perhaps through bankruptcy, or change missions. This can deprive the storage system of the support it needs to survive. Planning must include data "exit strategies" that envisage the possibility of the asset represented by the preserved content being transferred to a successor organization.

During organizational changes, a large IT company closed a research lab and requested the lab's research projects be copied to tape and sent to another of its labs. Unfortunately, the tapes languished without documentation of their contents, because few knew about them. When it became clear that some of the project data would be useful to current researchers, enough time had passed that nobody could identify what would be on which tape, and the volume of data was too huge to reconstruct an index [28].

Storage services can also make mistakes, and assets dependent on a single service can be lost. An Ofoto user's digital photographs of her wedding were deleted when she did not make a purchase in the required interval. She had not updated her e-mail address so did not receive the warning [29]. Even if she had, at that time Ofoto provided no "data exit strategy" by which she could have retrieved her high-resolution originals.

Economic faults. Organizations often stretch their limited budgets simply to get their collections online, leaving little or nothing to ensure continued accessibility. There are ongoing costs for power, cooling, bandwidth, system administration, equipment space, domain registration, renewal of equipment, and so on. Information in digital form is much more vulnerable to interruptions in the money supply than information on paper, and budgets for digital preservation must be expected to vary up and down, possibly even to zero, over time. These budget issues affect our ability to preserve as many collections as desired: many libraries now subscribe to fewer serials and monographs [7].

The lack of tools to predict these on-going costs makes it difficult to motivate an investment in preservation [18], especially if the future target audience does not exist at the time decisions are made. While budget is an issue in the purchase of any storage system, a shorter lifespan makes planning easier.

4. DANGEROUS ASSUMPTIONS

These threats are not new, so why are archives still losing data? A recent study for the National Archives [48] noted the lack of explicit threat models for archival storage systems, implying that systematic design and an end-to-end perspective are lacking. To that issue we add common, but potentially dangerous assumptions. These include visibility of faults (§4.1), independence of faults (§4.2), and unlimited budgets (§4.3), as described below.

4.1 The fault visibility assumption

While many faults are detected at the time an error causes them, some occur silently. Media errors are the best known of many sources of these *latent faults*. A disk sector might become unreadable or bits might rot, but this will not be detected until an attempt is made to read them.

Silent media errors and faults occur more frequently than commonly assumed. Schwarz et al. suggest that silent block faults occur five times as often as whole disk faults [45]. Our study of data from the Internet Archive (IA) (§5.4) shows a smaller but still significant rate.

In aggregate, systems like the IA might supply users with data items at a high rate, but the average data item is accessed infrequently. Detecting loss or corruption only at user accesses renders the average data item vulnerable to an accumulation of latent faults.

Beyond media faults, the threats of §3 can lead to many types of latent faults:

Human error: Accidental deletion or overwrite might not be discovered until the affected material is needed.

Component failure: The reliance on a failed system component or a third-party component that is no longer available might not be discovered until data depending on that component are accessed.

Media/hardware obsolescence: Failure of an obsolete,

seldom-used reader might not be discovered until information on its medium needs to be read. It might then be impossible or too costly to repair or replace the reader.

Software/format obsolescence: Upon accessing old information we might discover it is in a format belonging to an application we can no longer run.

Loss of context: We might not discover we are missing crucial metadata about saved data until we try to make sense of them. For instance, we might not have preserved an encryption key.

Attack: Results of a successful censorship or corruption attack on a data repository might never be discovered, or might only become apparent upon accessing the data long after the attack.

4.2 The independence assumption

Data replication is necessary for preventing data loss, but it is not sufficient. Analysis of replication schemes is often based on the assumption that replicas fail independently. Alas, in practice, faults are not as independent as we might hope. Taking an end-to-end perspective, the threats of §3 provide many sources of fault correlation:

Large-scale disaster: A single large disaster might destroy all replicas of the data. Geographic replication clearly helps, but care must be taken to ensure it provides sufficient independence. For example, the 9/11 disaster in New York City destroyed a data center. The system correctly failed over to a replica data center on the other side of a river. Unfortunately, the replicas were not independent enough; the chaos in the streets prevented staff from getting to the backup. Eventually it was unable to continue unattended [53].

Human error: System administrators are human and fallible. Unfortunately, in most systems they are also powerful, able to destroy or modify data without restriction. If all replicas are under unified administrative control, a single human error can cause faults at all of them.

Component faults: If all replicas of the information are dependent on the same external component, the loss of that component causes correlated faults at every replica. Talagala [49] logged every fault in a 368-disk system at UC Berkeley over six months and showed significant correlations between them due to shared failed components such as power connections, cooling and SCSI controllers.

Media faults: Temperature and vibrations of tightly packed devices in machine room racks [5] are sources of correlated media faults.

Loss of context: Losing metadata for archival content might cause correlated faults across replicas.

Attack: Attacks can cause correlated faults, e.g., a flash worm affecting many replicas at once, or a censorship attack affecting the same materials in many places.

Organizational faults: Correlated organizational faults – the simultaneous demise of all subsidiaries of the same defunct parent company – can cause all copies of an archive to fail at the same time.

4.3 The unlimited budget assumption

The biggest threats to digital preservation are economic. Most of the information people would like to see live forever is not in the hands of organizations with unlimited budgets; it cannot be preserved by optimal but expensive techniques such as synchronous mirroring of RAIDs across widely dispersed geographic replicas. Although we mention cost aspects of some of the strategies we explore in this paper, detailed cost modeling of long-term storage will require much more real-world data to be meaningful.

5. MODELING DATA LOSS

Abstract models such as those for RAID [10, 38] are useful for reasoning about the reliability of different replicated storage system designs. In this section, we build upon these models, increasing the focus on issues important to long-term storage by incorporating the effect of latent and correlated faults on the overall reliability of an arbitrary unit of replicated data. Our model is agnostic to the unit of replication; it can be a bit, a sector, a file, a disk, a collection of objects, or an entire storage site. We therefore attempt to develop a more abstract model that can be interpreted in a more general, holistic fashion. *The model is not intended to provide exact answers*, and its mathematical representation is only a coarse approximation. Instead, the model provides a conceptual methodology for reasoning about the relative impact of a broad range of faults, their detection times, their repair times, and their correlation. The model helps point out what strategies are most likely to increase reliability, and what data we need to measure to resolve trade-offs between these strategies.

Our model also distinguishes between the size of the unit of replication and the size of the fault. The damaged data may be bigger or smaller than the replication unit. For instance, while we might replicate data at the file level, a fault might only affect a few bytes of the file. Traditionally, in looking at block-level or disk-level replication strategies, faults have sometimes been assumed to affect a whole disk, even if some of the information is salvageable from the disk. We separate replication size and fault size in our model to make it possible to identify more generally what data are actually damaged.

The disk has traditionally been the convenient unit to work with, because manufacturers report MTTFs for disks. In contrast, our model allows us to work at any granularity, incorporating more specific information about the likelihood of failures for different units of replication as the information becomes available. Determining the most effective, lowest-cost method of replication for a system might require parallel analysis at several different choices of replication unit.

We start with a simple, abstract definition of latent faults. We then derive the mean-time-to-failure of mirrored data in the face of both immediately visible and latent faults. This derivation includes both temporal and spatial coincidence of faults. We extend this equation to include the effect of faults correlated either temporally or spatially. Finally, we discuss the implications of this equation on the reliability of long-term storage under various scenarios.

5.1 Visible versus latent faults

We distinguish between immediately visible and latent faults, as shown in Figure 1. Visible faults are those for which the time between their occurrence and detection is negligible. Examples of such faults include entire-disk or controller failures. We denote the mean time to a visible fault by MV and the associated mean time to repair by MRV (see symbol key in Table 1). Latent faults are those for which the time between occurrence and detection is significant. Examples include misdirected writes, bit rot, un-
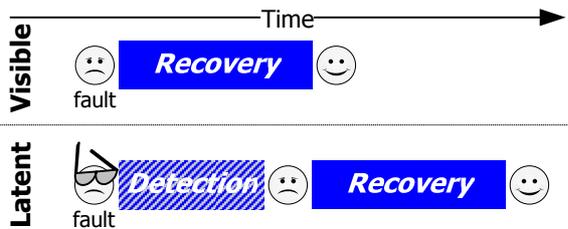
Figure 1: *Types of replica faults. Time flows from left to right. Above, when a visible fault (sad face) is detected, recovery begins immediately. At the end of successful recovery, the fault has been corrected (happy face). Below, when a latent fault occurs (face behind sunglasses), nothing happens until the fault is detected. Then, as with visible faults, recovery takes place.*

α	Temporal correlation factor
β_{ab}	Prob. a and b overlap spatially
MDL	Mean time to detect latent fault
ML	Mean time to latent fault
MRL	Mean time to repair latent fault
MRV	Mean time to repair visible fault
MTTDL	Mean time to data loss
MTTF	Mean time to fault
MV	Mean time to visible fault
WOV	Window of vulnerability
$a \cap_T b$	a coincides in time with b

Table 1: *Key to symbols and acronyms in the model.*

readable sectors, and obsolete data formats. We denote the mean time to a latent fault by ML and mean time to repair by MRL. We only consider latent faults that are detectable; hence, they have a finite mean time between occurrence and detection, denoted by MDL.

5.2 Assumptions in the model

The mathematical representation of our model is developed by starting with simple assumptions and adding complexity in stages. Following RAID [38], we start by assuming that the processes generating faults are memoryless. That is, the probability, $P(t)$, of a fault occurring within time t, is independent of age. This assumption leads to the exponential distribution $P(t) = 1 - e^{-t/\text{MTTF}}$, where MTTF is the mean time to the fault. For many parts of our derivation, we consider the case where $t \ll \text{MTTF}$, so the following approximation holds:

$$1 - e^{-t/\text{MTTF}} \approx 1 - (1 - \frac{t}{\text{MTTF}}) = \frac{t}{\text{MTTF}} \qquad (1)$$

This approximation and similar ones below are used only to simplify the expression for the exponential in the probability and are not fundamental to the model.

Initially, we assume that all faults occur independently of one another, both temporally and spatially. Subsequently, we introduce correlated faults that are also exponentially distributed but with an increased rate of occurrence. For simplicity, we model this increase by a multiplicative correlation factor, the same for both latent and visible faults. We also accommodate faults with an increased likelihood that resulting damage affects the same data in both replicas. This approach accounts for faults correlated either temporally or spatially, but does not account for cross correlations across

space and time.

Our equations provide a mean-value analysis of reliability. If the distributions of faults, recovery times, or detection times are not exponential (e.g., bimodal) our use of means is inexact. A distribution analysis would provide better understanding of the distribution of data loss probability, but we have not yet attempted it.

Finally, we also make assumptions about detection and repair of latent faults. Latent errors manifest themselves in two basic ways: as inaccessible data or as corrupted data. If data are inaccessible, then noticing the fault upon access is straightforward. We can repair this fault by creating a new copy from the remaining redundant copies. If the data are corrupted, then noticing the fault requires further work such as comparing the data to a copy. To decide which copy is corrupted, though, we need additional information. For instance, we might use the information that one copy decrypts or decompresses to something sensible, while the other produces gibberish. We might replicate content more aggressively and use majority consensus among the replicas. Or we might use collision-resistant checksums. Although checksums cannot be used to repair a damaged copy, they can provide votes for the consensus. For the model described in the next section, we assume that we can detect damage by comparing two copies of the content, and that we have sufficient information to determine which copy to use for repair.

5.3 The model

Our model only considers redundancy schemes that replicate the entire body of data. For most of this section we concentrate on mirroring, the simplest form of replication.

Mirrored data become irrecoverable when there are two successive faults, one in each copy, such that a) the second fault occurs before the initial fault can be repaired (temporal overlap) and b) the damaged portions of the copies intersect (spatial overlap).

MTTDL is the mean time to data loss, and $\frac{1}{\text{MTTDL}}$ is equal to the rate of double faults that lead to data loss. In this section, we derive an expression for this quantity, which represents the reliability of mirrored data, to understand how it is affected by visible, latent, and correlated faults. We first estimate the probability of temporal overlap and subsequently account for the probability of spatial overlap. Note that a double fault may occur without a user being aware of the data loss. Nevertheless, it still counts toward the double-fault rate.

In our model, we experience **temporal overlap** when a fault occurs at a second replica during the *window of vulnerability (*WOV*)* after a fault at the first replica. The WOV is the time during which the first fault remains unrepaired. We denote this intersection in time by $\cap_T$. Since faults may be visible or latent, we need to consider the WOV after each type, as illustrated in Figure 2.
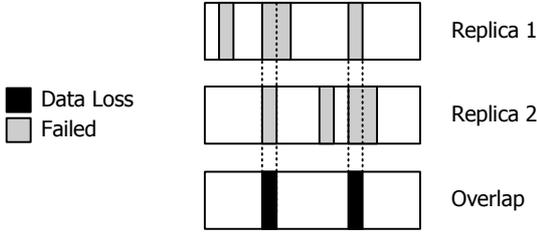
First, consider the WOV **after a visible fault**, V_1, which on average is MRV. During this WOV, both latent and visible faults can occur. The probability that another visible fault, V_2, overlaps in time with the WOV of the first fault, given that the first fault has occurred, is

$$P(V_2 \cap_T V_1 | V_1) \approx \frac{\text{MRV}}{\text{MV}} \qquad (2)$$

where MRV $\ll$ MV. We obtain this result by using the
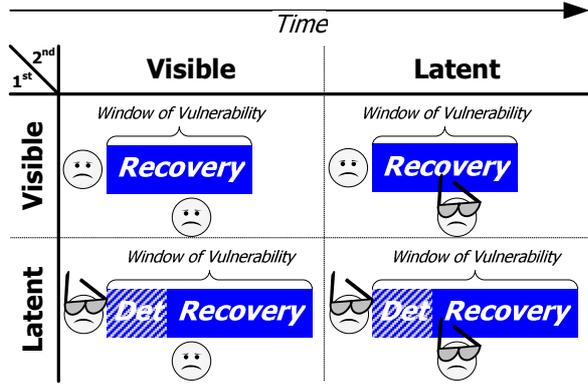
Figure 2: *Temporal overlap of faults causing data loss. The y axis indicates the type of the first fault (first sad face) and the x axis indicates the type of the second fault (second sad face). After the first fault occurs, there is a **window of vulnerability** during which the occurrence of a second fault can lead to data loss. After visible faults, this window only consists of the recovery period. After latent faults (faces behind sunglasses), this window also includes the time to detect the fault.*

Figure 3: *Spatial overlap of faults. Both replicas have suffered areas of damage. Intersecting damage causes data loss.*

approximation in Equation 1.

The probability that another latent fault, L_2, occurs is

$$P(L_2 \cap_T V_1 | V_1) \approx \frac{\text{MRV}}{\text{ML}} \qquad (3)$$

where $\text{MRV} \ll \text{ML}$. The difference between $P(V_2 \cap_T V_1 | V_1)$ and $P(L_2 \cap_T V_1 | V_1)$ arises only from the different rates of visible and latent fault occurrence.

Next, consider the WOV **after a latent fault**, L_1, which on average is $\text{MDL} + \text{MRL}$. Again both latent and visible faults can occur in this WOV.

The probability that another visible fault, V_2, overlaps with the WOV of the first fault, given that the first fault has occurred, is

$$P(V_2 \cap_T L_1 | L_1) \approx \frac{\text{MDL} + \text{MRL}}{\text{MV}} \qquad (4)$$

and the probability that another latent fault, L_2, occurs is

$$P(L_2 \cap_T L_1 | L_1) \approx \frac{\text{MDL} + \text{MRL}}{\text{ML}} \qquad (5)$$

As before, $\text{MDL} + \text{MRL} \ll \{\text{MV}, \text{ML}\}$.

To obtain MTTDL, we need the overall probability that a second fault overlaps with a first fault; hence, we next

account for **spatial overlap** of the damaged areas, as illustrated in Figure 3. For instance, the likelihood that two faults overlapping in time would also damage the same materials in two replicas depends on their size and the areas they affect. Assuming that failures across replicas are not cross-correlated with space and time, we obtain the overall probability by multiplying each term for temporal overlap with a term, β, representing the probability that the faults overlap spatially. For example, for two successive visible faults,

$$P(V_2 \cap V_1 | V_1) \approx \beta_{VV} \cdot P(V_2 \cap_T V_1 | V_1) \qquad (6)$$

where $\cap$ denotes the overall overlap.

Spatial overlap may be either physical or logical. An example of physical overlap is where the same sectors on mirrored disks suffer faults. An example of logical overlap is where the same files hosted at two sites suffer faults; the same materials, regardless of their physical layout, are damaged at both replicas. Moreover, β can be a function of time. The amount of damage in a replica will accrue over time, and the longer the detection and recovery times, the more likely it becomes that damage in the second replica overlaps with the damage in the first replica.

To estimate the **total double-fault failure rate** we multiply the rate of the first fault by the probability that a second fault overlaps with the first, and then sum the products for each fault-type. Thus, $\frac{1}{\text{MTTDL}}$ is approximately

$$\frac{P((V_2 \vee L_2) \cap V_1 | V_1)}{\text{MV}} + \frac{P((V_2 \vee L_2) \cap L_1 | L_1)}{\text{ML}} \qquad (7)$$

where the first term counts the fraction of the visible faults that result in double-failures, and the second term counts the fraction of latent faults that result in double-failures. This equation actually reports the mean time to the first fault resulting in data loss, which approximates the mean time to the second fault when detection and recovery times are short. Substituting using Equation 6 and its counterparts and leaving out second-order terms, $\frac{1}{\text{MTTDL}}$ is approximately

$$\frac{\beta_{VV} \cdot P(V_2 \cap_T V_1 | V_1) + \beta_{LV} \cdot P(L_2 \cap_T V_1 | V_1)}{\text{MV}} +$$
$$\frac{\beta_{VL} \cdot P(V_2 \cap_T L_1 | L_1) + \beta_{LL} \cdot P(L_2 \cap_T L_1 | L_1)}{\text{ML}} \qquad (8)$$

Note that if MDL becomes large – latent faults are not detected – the approximations in Equations 4-5 in the second term do not hold. Instead, the combined $P((V_2 \vee L_2) \cap L_1 | L_1)$ approaches 1.

To account for **temporally correlated faults**, we assume that the probability of the second fault (conditioned on the occurrence of the first) is also exponentially distributed, but with a faster rate parameter, as illustrated in Figure 4. We introduce a multiplicative correlation factor $0 < \alpha \leq 1$ that reduces the mean time to the subsequent fault once an initial fault occurs. In this case, Equations 2–5 are multiplied by $1/\alpha$. Note that Equation 8 folds the α's into the temporal overlap probabilities.

To account for **spatially correlated faults**, such as might result from a successful censorship attack, we increase the probability that spatial overlap occurs by adjusting β closer to 1.

This is undoubtedly a vast simplification of how faults correlate in practice. We are not alone in using this simplification [12], because more accurate modeling of correlations
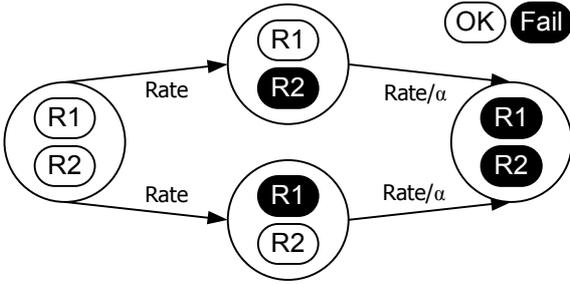
Figure 4: *Temporal correlation of faults. This state machine models temporal correlation of faults. In the first state, both replica R1 and R2 are good. Then one of the replicas suffers a fault at a rate based on the mean time to a visible or a latent fault. We represent correlation of faults through the increased rate (multiplied by $\frac{1}{\alpha}$ where $0 < \alpha \leq 1$) at which the second replica also experiences a fault.*

	Mirrored disk	Mirrored archive
MV	120,000 hrs	20 hrs
ML	9.7 yrs	1531 hrs
MRV	1.4 hrs	4.4 hrs
α	1	1
$\beta_{VV}, \beta_{LV}, \beta_{VL}$	1	1/1795
β_{LL}	0	0

Table 2: *Parameters used for reliability estimation.*

is still very difficult; it relies on modeling a particular system instantiation and its component interactions, as well as external influences such as attacks. We are instead seeking a more general way to reason about replicated materials regardless of the level of replication. An alternative would be to introduce a distinct MTTF for correlated faults that is less than the independent MTTF, as done for RAID [10].

5.4 Implications

To understand the implications of Equation 8, we investigate the behavior of two systems – a single pair of mirrored disks and a large geographically replicated archive. To do so, we parameterize the model with real data to explore the differences in reliability under three scenarios: 1) the effect of latent errors is ignored, 2) latent errors exist but the system does not try to detect or repair them, and 3) latent errors exist and the system detects and repairs them. Proactively searching for latent disk errors is often called "scrubbing." We prefer to use "auditing" to apply to all system levels and avoid confusion with deleting data. Our analysis confirms that the assumptions behind these scenarios lead to orders of magnitude differences in reliability. In this Section, we first explain how we obtain the parameters to calculate reliability, as shown in Table 2, for each of the two systems. Then, we proceed with the reliability estimation.

In the first example system, we consider replicated consumer ATA drives and take into account only faults from media errors. We base the visible fault rate on operations at a large disk farm as reported by Gray and van Ingen [19]. Assuming an exponential failure distribution and a 7% per year disk failure rate, we calculate an MV of 120K. We derive ML from one of several quoted worst-case unrecoverable error rates of 1 bit in 10^{15}. This is a read error rate, but for lack of additional information we assume this rate is

the rate of latent errors. Using an analysis similar to Chen et al. [10], this bit error rate results in a .164% probability of an unrecoverable read of a 200 GB disk. Assuming the disk is 99% idle and supports a 40 MB/s [19] transfer rate, we would read a disk capacity worth of data approximately 63 times a year. This results in an unrecoverable read once every 9.7 years, which we use for ML. Moreover, the mean time to recovery, MRV, for mirrored disks (reading the whole of the surviving replica and in parallel writing a hot standby) takes about 1.4 hours. To calculate the spatial overlap parameters for this system, we assume that all visible faults affect a whole disk, which is an entire replica. Thus, $\beta_{VV} = \beta_{VL} = \beta_{LV} = 1$. Since latent faults affect at a minimum a single sector and there are 4×10^8 sectors, we assume $\beta_{LL} \approx 0$ because it is small and its effect in the upcoming reliability equations is insignificant compared to the effect of the other β parameters.

Our second example system consists of two geographically separate just-a-bunch-of-disks (JBOD) replicas networked together, each with 1795 200GB drives. We base this example on a preliminary analysis of failure data collected from the IA. We derive our parameters from aggregate statistics computed over a subset of this failure data. A more detailed analysis that accounts for numerous other factors, such as growth of the archive during the data collection period or correlations among failures, is still pending.

Although the IA does not yet routinely audit replicas, IA staff collected 27.7K hours of monthly checksums and other data on about 1.5M immutable files, each on average about 43MB in size with about 836 files per disk. We observed 1366 unexplained checksum changes. Of these changed files, 1218 had been moved from one drive to another by the normal archive maintenance mechanisms without an error being detected during migration. Assuming these could be fixed by simple end-to-end checks after transfers, the remaining 148 are latent faults that were distributed across 18 disks. Further assume these disks experience latent faults that obey an exponential probability of failure. Using Equation 1, we obtain an ML of 2.7×10^6 hours for each disk or an ML of 1531 hours for a replica. From the IA data, we observe a mean time to disk replacement of 36×10^3 hours, resulting in a replica MV of 20 hours. We assume visible faults affect an entire disk, but latent faults affect a file. In this case if either of two temporally overlapping faults is visible, then their spatial overlap probability is $\beta_{VV} = \beta_{LV} = \beta_{VL} = 1/1795$, because there are 1795 disks in a replica. If the two temporally overlapping faults are both latent, then their spatial overlap probability is much smaller, $\beta_{LL} = 1/(1.5M)$, since there are 1.5M files. At this magnitude we assume $\beta_{LL} \approx 0$, because its effect in the reliability equations is insignificant. In this scenario, MRV is longer because recovery involves copying data over a wide-area network. Assuming a transfer rate of 100 Mbps, MRV for a 200GB disk is 4.4 hours.

These parameters incorporate faults arising from the day-to-day operation of a large archive. Such faults may result from errors at all levels of the storage stack, from low-level media up to the human operator. Thus, although not comparable to the ones in the first system, these parameters may be more realistic. In what follows we assume that all faults are independent ($\alpha = 1$).

5.4.1 No latent errors

First, consider the scenario most often explored in the lit-

erature, in which latent faults are completely ignored. Because we are ignoring the effects of latent faults, we assume $MV \ll ML$ and $MDL + MRL \ll MV$. As a result, the second term in Equation 8 becomes negligible compared to the first, which simplifies to

$$\text{MTTDL} \approx \frac{\alpha \cdot MV^2}{\beta_{VV} \cdot MRV}. \qquad (9)$$

Plugging in the parameters for our mirrored disks, MTTDL is 1.2×10^6 years. This seems optimistic. Using the parameters for the replicated archive, we calculate MTTDL to be 18.6 years. Note, even if we were to remove all possible latent errors, this is the best reliability we could achieve for a replicated archive of this scale and type. This points to the need for additional redundancy, perhaps at the level of the entire archive or perhaps at lower levels, e.g., using RAID within an archive replica to boost MV.

5.4.2   Latent errors without auditing

Next, given the evidence that latent errors occur in practice, we show their impact on reliability by removing some assumptions. First, we remove our neglect of latent faults. Without auditing, latent errors in archival systems will typically be discovered only by infrequent user accesses (§4.1). In this case, $MDL + MRL \gg ML$, which means the numerator in the second term of Equation 7 approaches 1. This limit results because of an increasing chance of both spatial and temporal overlap when detection times for latent faults are large. Second, we remove the constraint $MV \ll ML$ from above. Then, MTTDL simplifies to:

$$\text{MTTDL} \approx \frac{\alpha \cdot MV^2}{MRV \cdot \frac{\beta_{VV}ML + \beta_{LV}MV}{ML} + \frac{\alpha MV^2}{ML}}. \qquad (10)$$

This equation provides a quantitative metric for determining when latent errors are negligible. To achieve the same reliability as before, ML must be large enough so that the right half of the denominator is negligible compared to the left half, i.e., $ML \gg MV^2$. Moreover, when recovery and detection times are long, this equation underestimates MTTDL because it actually reports the mean time to first fault. To get a better estimate on the mean time to second fault, i.e., data loss, we add in a correction factor of at most $min(MV, ML)$ to the MTTDL obtained from Equation 7. We omit the result due to space.

Substituting our parameters for the mirrored disks, without auditing, MTTDL is 8.5×10^4 hours or 9.7 years. For the replicated archive example, MTTDL is 1517 hours without the correction factor, and 1537 hours with the correction factor (add 20 hours), or a little more than 64 days. Compared to the previous reliability numbers, the benefit of redundancy in the presence of latent errors is undermined.

5.4.3   Latent errors with auditing

Given that latent errors are not negligible, we examine the effect of auditing on reliability. Following Schwarz et al. [45], we assume a periodic process that checks and fixes all data every 4 months, which leads to an MDL of 1460 hours (half the auditing period). Since recovery of latent faults involves a sector copy or file copy of at most 100MB, we assume $MRL \approx 0$ in both cases. Moreover, we substitute $MV = kML$ into Equation 8 and simplify. In the mirrored disk case, $k = 1.41$ and in the replicated archive case, $k =$

.013. This substitution simplifies MTTDL to

$$\frac{\alpha \cdot k^2 \cdot ML^2}{MRV(\beta_{VV} + k\beta_{LV}) + k(MDL + MRL)(\beta_{VL} + k\beta_{LL})}. \qquad (11)$$

For the mirrored disk case with auditing, MTTDL is 7.0×10^6 hours or 795 years. For the IA example, if we audit every 4 months, MTTDL is 3.0×10^4 hours or 3.4 years. Auditing every two weeks ($MDL = 168$ hours) improves the overall reliability to 12.3 years. We see that a sufficiently proactive detection process may increase reliability by orders of magnitude.

In this equation, MDL+MRL plays a role similar to MRV. With all other variables fixed, this result implies that reducing detection time can be as beneficial as reducing recovery times.

Since detecting latent errors is a proactive process that can tax and wear out compute and storage resources, auditing can potentially induce an increase in the rate of visible or latent errors. The above equation implies that any multiplicative increase in the error rate due to auditing must be accompanied by a quadratic reduction in detection and repair time of latent errors. That is, assuming $MDL + MRL$ dominates the denominator, if $ML_{new} = ML_{old}/x$, then $(MDL + MRL)_{new} < (MDL + MRL)/x^2$ for auditing to be effective.

Finally, these three specializations of the model point out an additional trade-off between the rates of occurrence of latent and visible faults. In Equations 9-11 MTTDL varies strongly with both MV and ML and in particular is limited by the smaller of MV and ML. Thus, we must consider occurrence rates for both fault types to improve overall system reliability. We must be careful not to sacrifice one for the other, which might happen in practice, since MV and ML can often be anti-correlated depending upon hardware choice and detection strategy.

5.5   Replication and correlated faults

In this section, we show that additional replication does not offer much additional reliability without independence. To simplify our reliability analysis for higher degrees of replication, we assume that we have instrumented the system so that MDL is short, and we assume that latent and visible faults have similar rates and repair times, MV. We roughly estimate the MTTDL of a system with a degree of replication, r, by extending the analysis along the same lines as for mirrored data. We calculate the probability that $r - 1$ successive, compounding faults after an initial fault will leave the system with no integral copy from which to recover.

To do so, we calculate the probability of $r - 1$ successive faults occurring within the vulnerability windows of each previous fault. For simplicity, this analysis assumes that the vulnerability windows, each of length MRV, overlap exactly. In that case, the probability that the k-th copy fails within the WOV of the previous $(k - 1)$ failed copies is roughly $\frac{MRV}{\alpha \cdot MV}$. Thus, the probability that successive $r - 1$ copies all fail within the WOV of previously failed copies is the product of those $r - 1$ probabilities, $\left(\frac{MRV}{\alpha \cdot MV}\right)^{r-1}$. Since the first fault occurs with rate $\frac{1}{MV}$, the overall mean-time-to-data-loss is

$$\text{MTTDL} \approx MV \cdot \left(\frac{\alpha \cdot MV}{MRV}\right)^{r-1} \approx \frac{\alpha^{r-1} \cdot MV^r}{MRV^{r-1}} \qquad (12)$$

This equation shows that although increasing the level of

replication, r, geometrically increases MTTDL, a high degree of correlated errors ($\alpha \ll 1$) would also geometrically decrease MTTDL, thereby offsetting much or all of the gains from additional replicas.

6. STRATEGIES

This simple model reveals a number of strategies for reducing the probability of irrecoverable data loss:

- Increase MV by, for example, using storage media less subject to catastrophic data loss.

- Increase ML by, for example, using storage media less subject to data corruption.

- Reduce MDL by, for example, auditing the data more frequently to detect latent data faults.

- Reduce MRL by, for example, automatically repairing latent data faults rather than alerting a human operator to do so.

- Reduce MRV by, for example, providing hot spare drives so that recovery can start immediately rather than once a human operator has replaced a drive.

- Increase the number of replicas enough to survive more simultaneous faults.

- Increase α and decrease β by increasing the independence of the replicas.

While we generally describe these strategies in terms of the more familiar hardware and media faults, they are also applicable to other kinds of faults. For instance, in addition to detecting faults due to media errors, auditing can detect corruption and data loss due to attack. As another example, we can use a similar process of cycling through the data, albeit at a reduced frequency, to detect data in endangered formats and convert to new formats before we can no longer interpret the old formats. In the rest of this section we examine the practicality and costs of some techniques for implementing these strategies.

6.1 Increase MV or ML

Increasing MV and ML will provide higher reliability, but the cost trade-offs must be considered in each case. For example, should we build an archive using more reliable, but more expensive, enterprise drives, or use lower-cost consumer drives and more replication? Based on Seagate's specifications, a 200GB consumer Barracuda drive has a 7% visible fault probability in a 5-year service life [46], whereas a 146GB enterprise Cheetah has a 3% fault probability [47]. But the Cheetah costs about 14 times as much per byte ($8.20/GB versus $0.57/GB – prices from TigerDirect.com 6/13/05). The Barracuda has a quoted irrecoverable bit error rate of $< 10^{-14}$ and the Cheetah of $< 10^{-15}$. Instead of using Cheetahs we could implement the archive using RAID5 arrays of Barracudas, increasing the archive's MV, *and also* replicate the entire archive, increasing MTTDL given the MV.

6.2 Reduce MDL

A potentially less expensive approach to addressing latent faults is to detect the faults as soon as possible and repair them. The only way to detect these faults is to *audit* the replicas by reading the data and either computing checksums or comparing against other replicas.

Assuming (unrealistically) that the detection process is perfect and the latent faults occur randomly, MDL will be half the interval between audits, so the way to reduce it is to audit more frequently. In other words, one can reduce MDL by devoting more disk read bandwidth to auditing and less to reading the data. Recent work suggests that in many systems a reasonable balance of auditing versus normal system usage [39, 45] can be achieved.

On-line replicas such as disk copies have two significant advantages over off-line copies such as tape backups. First, the cost of auditing an off-line copy includes the cost of retrieving it from storage, mounting it in a reader, dismounting it and returning it to storage. This can be considerable, especially if the off-line copy is in secure off-site storage. Second, on-line media are designed to be accessed frequently and automatically. Auditing off-line copies, on the other hand, is a significant cause of highly correlated faults, from the error-prone human handling of media [41] to the media degradation caused by the reading process [4].

The audit strategy is particularly important in the case of digital preservation systems, where the probability that an individual data item will ever be accessed by a user during a disk lifetime is vanishingly small. The system cannot depend on user access to trigger fault detection and recovery, because during the long time between accesses latent faults will build up enough to swamp recovery mechanisms. A system must therefore proactively audit its replicas to minimize MDL. The LOCKSS system [32] is an example.

Note that relying on off-line replicas for security is not foolproof. Off-line storage may reduce the chances of some attacks, but it may still be vulnerable to insider attacks. Because it is harder to audit, the damage due to such attacks may persist for longer.

6.3 Reduce MRL or MRV

Reducing the mean time to repair a visible fault is important in reducing the window of vulnerability, and therefore in improving reliability. Although the mean time to repair a latent media fault will normally be far less than the mean time to detect it, it is important that we do not inadvertently increase MRL to be as big as MDL.

Again, on-line replicas have the major advantage that repair times for media faults can be short – a few media access times. No human intervention is needed, and the process of repair is in itself less likely to cause additional correlated faults. Repairing from off-line media incurs the same high costs, long delays and potential correlated faults as auditing off-line media.

6.4 Increase replication

Off-line media are the most common approach to increasing replication. But the processes of auditing and recovering from faults using off-line backup copies can be slow, expensive, and error-prone.

Some options for disk-based replication strategies include replication within RAID systems, across RAID systems, and across simple mirrored replicas. Replication within RAID

systems does not provide geographical or administrative independence of the replicas. If we require these kinds of independence, a global view of system reliability and costs must weigh the costs of increasing the reliability of individual sites versus increasing the level of geographic replication using cheaper components. OceanStore [27] is an example of using a large number of replicas on cheap disks.

6.5 Increase independence

In just the six months of the Talagala study [49], many correlated faults were observed, apparently caused by disks sharing power, cooling, and SCSI controllers, and systems sharing network resources. Our model suggests that in most cases, even with far lower rates of correlated faults, increasing the independence of replicas is critical to increasing the reliability of long-term storage.

Long-term storage systems can reduce the probability of correlated faults by striving for diversity in hardware, software, geographic location, and administration, and by avoiding dependence on third-party components and single organizations. Examples include:

Hardware: Disks in an array often come from a single manufacturing batch, exhibiting similar fault characteristics. However, the increased cost that would be incurred by giving up supply chain efficiencies of bulk purchase might make hardware diversity difficult. Note, though, that replacing all components of a large archival system at once is likely to be impossible. If new storage is added in "rolling procurements" over time [8], then differences in storage technologies and vendors over time naturally provide hardware heterogeneity.

Software: Systems with the same software are vulnerable to epidemic failure; Junqiera et al. have studied how the natural diversity of systems on a campus can be used to reduce this vulnerability [23]. However, the increased costs caused by encouraging such diversity, in terms not merely of purchasing, but also of training and administration, might again make this a difficult option for some organizations. The system at the British Library (BL) [8] is unusual in explicitly planning to develop diversity of both hardware and software over time. Nevertheless, given the speed with which malware can find all networked systems sharing a vulnerability, increasing the diversity of both platform and application software is an effective strategy for increasing α.

Geographic location: Many systems that use off-line backup store replicas off-site, despite the additional storage and handling charges that implies. Digital preservation systems, such as the BL's, establish each of their on-line replicas in a different location, again despite the possible increased operational costs of doing so.

Administration: Human error is a common cause of correlated faults among replicas. Again, the BL's system is unusual in ensuring that no single administrator will be able to affect more than one replica. This is probably more effective and more cost-effective than attempts to implement "dual-key" administration, in which more than one administrator has to approve each potentially dangerous action. In a crisis, shared pre-conceptions are likely to cause both operators to make the same mistake [40].

Components: System designs should avoid dependence on third-party components that might not themselves be preserved over time. Determining all sources of such dependence can be tricky, but some sources can be detected by running systems in isolation to see what breaks. For example, running a system in a network without a domain name service or certificate authority can determine whether the system is dependent on those services. Such an example is the LOCKSS system, which is built to be independent of the survival of these services.

Organization: Taking an end-to-end view of preservation systems, it is also important to support their organizational independence. For instance, if the importance of a collection extends beyond its current organization, then there must be an easy and cost-effective "exit strategy" for the collection if the organization ceases to exist. For example, a bickering couple might not want to store their babies' photos on a home server that requires their continued cohabitation for access.

Unfortunately, these strategies are not necessarily orthogonal, and some can have adverse effects on reliability. Auditing media to detect latent faults increases reliability but can introduce other channels for data corruption. An example would be attacking a distributed system through the audit protocol itself [32], which therefore must be designed as carefully as any other distributed protocol. Automated recovery can reduce costs and speed up recovery times, but if buggy or compromised by an attacker, it can also introduce latent faults. This can be dangerous because even visible faults can now (though seemingly having been recovered) turn into latent ones. Strategies such as increasing the independence of administrative domains and diversity of hardware and software also come with increased costs and organizational overhead.

In summary, the main techniques for increasing the reliability of long-term storage are replication, independence of replicas, auditing replicas to detect latent faults, and automation to reduce repair times and costs.

7. FUTURE WORK

There is much more to be done. Our simple reliability model points out areas where we are in great need of further data to validate the model and to evaluate the potential utility of the reliability strategies described in this paper. In particular, there is little published about the types and distribution of latent faults, both due to media errors and also due to the other threats we describe. Moreover, the correlations between faults are poorly understood.

We have started to analyze logs from the IA that contain periodic monitoring data about the file system, checksums of individual files, SMART records, and loads on individual hosts. While we use initial results in Section 5.4, we hope to extract the distribution of faults and their correlations with loads on the hosts and drives. However, this data set lacks information about workload, system administration activities, root causes of failure, and costs in terms of machine and human effort. The data set only relates to this unique system. Logging these details for a range of systems is essential to developing better long-term storage systems.

We should thus instrument a range of existing systems. We should log visible faults and detected latent faults, and occurrences of data loss, for example by following the IA in regularly logging a checksum for each immutable object. Ideally, the log entries would include timestamps and information about the location of the fault: block, sector, disk, file, application, etc. We should also log information about

recovery procedures performed (e.g., replacement of disks or recovery from tape), their duration, and outcomes. Together these logs would allow us to validate our model. If we were also able to collect cost data we could start to compare the cost-effectiveness of the strategies of §6. Finally, logging SMART data from disks, workload data, and administrative actions together would allow us to identify correlated faults and perform root cause analysis.

8. RELATED WORK

In this section we review related work, showing how others address problems, such as correlated and latent faults, that arise when large volumes of data must remain unaltered yet accessible with low latency at low cost. We start with low-level approaches that focus on single devices and RAID arrays and then move up the stack.

Evidence of correlated faults comes from studies of disk farms. Talagala [49] logs every fault in a large disk farm (of 368 disk drives) at UC Berkeley over six months and shows significant correlations between them. The study focuses primarily on media (drive failures), power failures, and system software/dependency failures.

Gray and van Ingen share our interest in looking at failure rates in storage systems and have been monitoring failures at many levels in disks arrays [19].

Chen et al. [10] explore the tradeoffs between performance and reliability in RAID systems, noting that system crashes (i.e., correlated faults) and uncorrectable bit errors (i.e., latent faults) can greatly reduce the reliability predicted in the original RAID paper [38]. Kari's dissertation [24] appears to be the first comprehensive analysis of latent disk failures. His model also shows that they can greatly reduce the reliability of RAID systems, and presents four auditing algorithms that adapt to disk activity, using idle time to flush out latent errors. In contrast with these works, we explore a broader space that includes application faults and distributed replication.

Enterprise storage systems have recognized the need to address latent and correlated faults. Many high-end storage arrays perform block-level data audits [30]. Network Appliance's storage threat model includes two whole-disk failures, and a whole-disk failure with a latent fault discovered during recovery (our $P(L_2 \cap_T V_1)$), but seems to ignore the cases in which latent faults occur first. They employ row-diagonal parity and suggest periodic auditing of disks to improve reliability [12]. Schwarz et al. [45] show that *opportunistic* auditing, piggy-backed on other disk activity, performs well. Like us, they do not depend on the disk to detect a latent fault but actually check the data. Our exploration also includes higher-layer failures.

Database vendors have implemented some application-level techniques to detect corruption. DB2's threat model includes a failure to write a database page spanning multiple sectors atomically. It sprinkles page consistency bits through each page, modifying and checking them on each read and write. This scheme only detects forms of corruption that affect the consistency bits [34]. Tandem NonStop systems write checksums to disk with the data, and compare them when data are read back [9].

The IRON File System's [39] threat model includes latent faults and silent corruption of the disk. It protects the file system's metadata using checksums and in-disk replication. Moreover, it employs checksums for data blocks and

a single-block parity for a user file which enables recovery from the failure of at most one block. Although much better than traditional file systems, this design is still vulnerable to correlated corruption of multiple blocks on a storage device.

An alternative to tightly-coupled replication such as RAID is more loosely-coupled distributed replication. Saltzer suggested in 1990 that digital archives need geographically distributed replicas to survive natural disasters, that they must proactively migrate to new media to survive media obsolescence, and that heavy-duty forward error correction could correct corruption that can accumulate when data are rarely accessed [44]. More recently, these ideas inform the design of the BL's digital archive [8].

Many distributed peer-to-peer storage architectures have been proposed to provide highly-available and persistent storage services, including the Eternity Service [6], Intermemory [11], CFS [13], OceanStore [27], PAST [43], and Tangler [54]. Their threat models vary, but include powerful adversaries (Eternity Service) and multiple failures. Some (OceanStore) use cryptographic sharing to proliferate n partial replicas from any $m < n$ of which they can recover the data. Others (PAST) replicate whole files. Weatherspoon's [55] model compares the reliability of these approaches. The model does not include latent errors or correlated errors, but instead takes into account the storage and bandwidth requirements of each approach. Later work [56] identifies correlations among replicas (e.g., geographic or administrative) and informs the replication policy to reduce correlation.

Deep Store [58] and the LOCKSS system [32] share the belief that preserving large volumes of data for long periods affordably is a major design challenge. Deep Store addresses the cost issues by eliminating redundancy. The LOCKSS system addresses costs by using a "network appliance" approach to reducing system administration [42] and large numbers of loosely coupled replicas on low-cost hardware. Both recognize the threats of "bit rot" and format obsolescence to long-term preservation.

9. CONCLUSIONS

We have examined the need for, and threats to, long-term storage of digital information. Using an extended reliability model that incorporates latent faults, correlated faults, and the detection time of latent faults, we reason about possible strategies for improving reliability. The cost of these strategies is important, since limited budget is one of the key threats. We find that the most important strategies are auditing to detect latent faults as soon as possible, automating repair so that it is as fast, cheap, and as reliable as possible, and increasing the independence of data replicas.

Many of the systems we describe in §8 incorporate a subset of the strategies we outline in §6 to counter a subset of the threats we identify in §3. Thus far, none has identified the tradeoffs of incorporating all of the strategies nor demonstrated over a long period of time the ability to balance these tradeoffs in defending against all the threats. We conclude with a possible system architecture that would do so.

A good architecture will maintain replicas that are geographically independent, administratively independent, and heterogeneous in platform. Geographic replication increases the chance content will survive large-scale disasters. Administrative independence ensures that no single administrator

can damage more than one replica. Platform independence provides resistance to many attacks and some component and obsolescence faults. Such independence of replicas can be a hard position to take for large repositories while the IT industry is promoting increased consolidation and homogeneity of systems, but it is important for digital preservation.

A good architecture will permit inexpensive audit of its content to root out and fix latent faults before they accrue to the extent that repair is impossible. This means that we cannot rely on backup to high-latency off-line media such as tape for detecting faults in the system; we must have some replicas accessible with relatively low latency for audit purposes. Currently, this suggests that disk-based replicas are a good choice, even if the disks are not kept spinning continuously. Furthermore, it means we cannot rely solely on local auditing of replicas within a single site; crossing platform, format, or organizational boundaries is essential to identify latent errors that monocultures mask away.

A good architecture will also allow for on-going introduction of new components, avoiding lock-in to any exotic technology or particular vendor's system. As the repository grows in size, or as it becomes appropriate to decommission old components, it must be possible to add new storage in whatever is a widely-used technology at the time. These "rolling procurements" imply that the architecture will always consist of heterogeneous components, as it is infeasible to convert large repositories all at once to entirely new systems.

Finally, a good architecture will include background processes at many levels of the storage system to detect endangered content, including formats that are becoming obsolete, access controls that no longer have meaning and metadata that must be updated. Digital archives are not static systems. To survive, they require continual vigilance, auditing and refreshing.

Data loss in systems designed on this basis should be extremely infrequent, but it will surely happen. The host institution will typically cover up such incidents. The experience base on which to improve the architecture will thus grow slowly and with difficulty. A system similar to NASA's Aviation Safety Reporting System [35] should be established, through which operators of storage systems could submit reports of incidents, even those not resulting in data loss, for others to read in anonymized form and from which they can learn how to improve the reliability of their own systems.

10. ACKNOWLEDGMENTS

11. REFERENCES

[1] 104th Congress, United States of America. Public Law 104-191: Health Insurance Portability and Accountability Act (HIPAA), Aug. 1996.

[2] 107th Congress, United States of America. Public Law 107-204: Sarbanes-Oxley Act of 2002, July 2002.

[3] D. Akst. Postcard from Cyberspace. *Los Angeles Times*, Jan. 1995.

[4] AMIA2003. Fact Sheet 5 - Estimating Tape Life. `http://www.amianet.org/publication/resources/guidelines/videofacts/tapelife.html`, 2003.

[5] D. Anderson, J. Dykes, and E. Riedel. More Than an Interface – SCSI vs. ATA. In *FAST*, 2003.

[6] R. J. Anderson. The Eternity Service. In *1st Intl. Conf. on the Theory and Applications of Cryptology*, 1996.

[7] ARL – Association of Research Libraries. ARL Statistics 2000-01. `http://www.arl.org/stats/arlstat/01pub/intro.html`, 2001.

[8] M. Baker, K. Keeton, and S. Martin. Why Traditional Storage Systems Don't Help Us Save Stuff Forever. In *Proc. 1st IEEE Workshop on Hot Topics in System Dependability*, 2005.

[9] W. Bartlett and L. Spanhower. Commercial Fault Tolerance: A Tale of Two Systems. *IEEE Transactions on Dependable and Secure Computing*, 1(1):87–96, 2004.

[10] P. M. Chen, E. L. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High Performance, Reliable Secondary Storage. *ACM Computing Surveys*, 26(2):145–185, June 1994.

[11] Y. Chen, J. Edler, A. Goldberg, A. Gottlieb, S. Sobti, and P. Yianilos. A Prototype Implementation of Archival Intermemory. In *Intl. Conf. on Digital Libraries*, 1999.

[12] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar. Row-Diagonal Parity for Double Disk Failure Correction. In *FAST*, 2004.

[13] F. Dabek, M. F. Kaashoek, D. Karger, R. Morris, and I. Stoica. Wide-area Cooperative Storage with CFS. In *SOSP*, 2001.

[14] T. Dawber, G. Meadors, and F. Moore. Epidemiological Approaches to Heart Disease: the Framingham Study. *American Journal of Public Health*, 41(3):279–81, Mar. 1951.

[15] W. Diffie. Perspective: Decrypting The Secret to Strong Security. `http://news.com.com/2010-1071-980462.html`, Jan. 2003.

[16] G. Goble. `http://ghg.ecn.purdue.edu/~ghg/`.

[17] Google, Inc. About Gmail. `http://gmail.google.com/gmail/help/about.html`, June 2005.

[18] J. Gray, A. Szalay, A. Thakar, C. Stoughton, and J. vandenBerg. Online Scientific Data Curation, Publication, and Archiving. Technical Report MSR-TR-2002-74, Microsoft Research, July 2002.

[19] J. Gray and C. van Ingen. Empirical Measurements of Disk Failure Rates and Error Rates. Technical Report MSR-TR-2005-166, Microsoft Research, Dec. 2005.

[20] E. Hansen. Hotmail Incinerates Customer Files. *News.com*,

http://news.com.com/Hotmail+incinerates+customer+files/2100-1038_3-5226090.html, June 2004.

[21] J. Horlings. CD-R's Binnen Twee Jaar Onleesbaar, 2003. *PC Active*, See http://www.cdfreaks.com/news/7751.

[22] IT Committee Inst. of Chartered Accountants of India. Tape backup vis-à-vis online Backup. *Harmony IT*, http://isaicai.org/Harmony/2004-07/index_plain.htm, July 2004.

[23] F. Junqueira, R. Bhagwan, A. Hevia, K. Marzullo, and G. M. Voelker. Surviving Internet Catastrophes. In *Usenix Annual Technical Conference*, 2005.

[24] H. Kari. *Latent Sector Faults and Reliability of Disk Arrays*. PhD thesis, Computer Science Department, Helsinki University of Technology, Finaland, Espoo, Finland, 1997.

[25] M. Keeney, E. Kowalski, D. Cappelli, A. Moore, T. Shimeall, and S. Rogers. Insider Threat Study: Computer System Sabotage in Critical Infrastructure Sectors. http://www.secretservice.gov/ntac/its_report_050516.pdf, May 2005.

[26] K. Keeton and E. Anderson. A Backup Appliance Composed of High-capacity Disk Drives. In *HotOS*, May 2001.

[27] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. OceanStore: An Architecture for Global-Scale Persistent Storage. In *ASPLOS*, 2000.

[28] R. Lau. Personal Communication, Sept. 2004.

[29] D. Lazarus. Precious Photos Disappear. *San Francisco Chronicle*, http://www.sfgate.com/cgi-bin/article.cgi?file=/chronicle/archive/2005/02/02/BUG7QB3U0S1.DTL, Feb. 2005.

[30] P. Luse and M. Schmisseur. Understanding Intelligent RAID 6. *Technology@Intel Magazine*, http://www.intel.com/technology/magazine/computing/RAID-6-0505.htm, 2006.

[31] R. Malda. The Myth of the 100 Year CD-Rom. *Slashdot*, http://slashdot.org/article.pl?sid=04/04/22/1658251\&mode=flat\&tid=137\&ti, Apr. 2004.

[32] P. Maniatis, M. Roussopoulos, T. Giuli, D. S. H. Rosenthal, and M. Baker. The LOCKSS Peer-to-Peer Digital Preservation System. *ACM TOCS*, 23(1), Feb. 2005.

[33] D. Milbank. White House Web Scrubbing, Dec. 2003. *The Washington Post*, http://www.washingtonpost.com/ac2/wp-dyn?pagename=article&node=&contentId=A9821-2003Dec17&notFound=true.

[34] C. Mohan. Disk Read-Write Optimizations and Data Integrity in Transaction Systems Using Write-Ahead Logging. In *ICDE*, 1995.

[35] NASA. Aviation Safety Reporting System. http://asrs.arc.nasa.gov/.

[36] OCLC. Persistent Uniform Resource Locator. http://purl.oclc.org/.

[37] K. Pang, K. Yau, and Hung-Hsiang Chou. The Earth's Palaeorotation, Postglacial Rebound and Lower Mantle Viscosity from Analysis of Ancient Chinese Eclipse Records. *Pure and Applied Geophysics*, 145(3-4):459–485, Sept. 1995.

[38] D. Patterson, G. Gibson, and R. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). In *SIGMOD*, 1988.

[39] V. Prabhakaran, N. Agrawal, L. Bairavasundaram, H. Gunawi, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. IRON File Systems. In *SOSP*, 2005.

[40] J. Reason. *Human Error*. Cambridge University Press, 1990.

[41] Reuters. Time Warner Says Employee Data Lost by Outside Storage Company. *The New York Times*, http://www.nytimes.com/2005/05/02/business/business-tech-timewarner.html?ex=1272686400&en=39cc177d5da055d2&ei=5090&partner=rssuserland&emc=rss, May 2005.

[42] D. S. H. Rosenthal. A Digital Preservation Network Appliance Based on OpenBSD. In *Proceedings of BSDcon 2003*, San Mateo, CA, USA, Sept. 2003.

[43] A. Rowstron and P. Druschel. Storage Management and Caching in PAST, A Large-scale, Persistent Peer-to-peer Storage Utility. In *SOSP*, 2001.

[44] J. H. Saltzer. Fault-tolerance in Very Large Archival Systems. In *SIGOPS European Workshop*, 1990.

[45] T. Schwarz, Q. Xin, E. Miller, D. Long, A. Hospodor, and S. Ng. Disk Scrubbing in Large Archival Storage Systems. In *MASCOTS*, 2004.

[46] Seagate. ST3200822A Configuration and Specifications. http://www.seagate.com/support/disc/specs/ata/st3200822a.html, Sept. 2003.

[47] Seagate. Cheetah 15K.4. http://www.seagate.com/cda/products/discsales/enterprise/tech/0,1084,656,00.html, 2005.

[48] R. F. Sproull and J. Eisenberg. Building an Electronic Records Archive at the National Archives and Records Administration: Recommendations for a Long-Term Strategy. http://www.nap.edu/catalog/11332.html, June 2005.

[49] N. Talagala. *Characterizing Large Storage Systems: Error Behavior and Performance Benchmarks*. PhD thesis, CS Div., Univ. of California at Berkeley, Berkeley, CA, USA, Oct. 1999.

[50] The Memory Hole. Department of Education to Delete Years of Research From Its Website. http://www.thememoryhole.org/edu/ed-info.htm, 2002.

[51] The OpenRAW Working Group. The RAW Problem. http://openraw.org, 2005.

[52] J. Tom. When Mutilators Stalk the Stacks. http://gort.ucsd.edu/preseduc/bmlmutil.htm, 2000.

[53] S. Towers. Personal Communication, July 2004.

[54] M. Waldman and D. Mazières. Tangler: A Censorship-Resistant Publishing System Based On Document Entanglements. In *ACM Conf. on Computer and Communications Security*, 2001.

[55] H. Weatherspoon and J. Kubiatowicz. Erasure Coding vs. Replication: A Quantitative Comparison. In *IPTPS*, Mar. 2002.

[56] H. Weatherspoon, T. Moscovitz, and J. Kubiatowicz. Introspective Failure Analysis: Avoiding Correlated Failures in Peer-to-peer Systems. In *Intl. Workshop on Reliable Peer-to-Peer Distributed Systems*, 2002.

[57] D. Whitehouse. Reworked Images Reveal Hot Venus. *BBC News*, Jan. 2004.

[58] L. L. You, K. T. Pollack, and D. D. Long. Deep Store: An Archival Storage System Architecture. In *ICDE*, 2005.