% How Many Copies Is Enough?
% Micah Altman; Richard Landau
% 2016-08-15
% Revised 2018-04-29 RBL

Information Integrity Over the Long Term -- How Many Copies Is Enough?

Assessing Long-term Durability of Collections Through A Flexible, Replicable Simulation

OUTLINE

- abstract
- summary of specific recommendations
- motivation
- problem definition

core risks

- types and severity of threats
- range of error rates, wide, what's plausible

structure of the model

- basic testing process
- data representation, units and scales, half-lives, logs

Simple and Complex Failures - simplest case: sector errors - similar case: glitches - complex case: shocks

simple cost model

auditing the document collection

- why audit
- simplest case: no auditing
- less simple case: total auditing, random, by popularity

- how many copies do we need with what auditing

observations

- large vs small collections (words)
- large vs small documents (picture)
- large vs small storage structures (words)
- total auditing is essential (picture)
- other types of auditing (words)
- segmented total auditing slightly better (words)
- auditing robust across a wide range of quality (picture)
- finding dead servers (words)
- sampling without and (random) with replacement

recommendations

for the collection owner

- five copies with annual auditing
- beware lack of independence
- have a plan for associated server failures
- be prepared with list of alternate storage vendors

for the digital preservation commons

- develop standards
- share experience of reliability of cloud vendors
- share experience on correlated failures

supplemental material

- details of model
- simplifying assumptions and scaling, Poisson IID, metric years, partitioning of simulations, non-repairable documents
- Poisson assumption, calibration tests
- words about error rates
- why did we choose this range of error rates
- representation and scale of error rates, why half-life
- relationship of MTBF to error rates
- what is MTBF

- the appearances of the graphs, x, y, logs, 1% line, where is zero
- overview of software architecture
- installation on AWS
- how to model various scenarios
- pointers to software how-to docs

where to these fit?

- limitations
- long term bit rot
- medium term if error rates are uncertain
- specific types of threats: finance, billing, adversaries internal and external, HVAC, environment, software failures, admin errors, hardware batches, government censorship, economics
- modeling associated failures, types and ranges
- things not modeled
- threat matrix
- extensions
- parameter value ranges
- large vs small collections
- large vs small documents
- compression
- encryption
- format obsolescence
- adversaries
- parallel between strategy of less reliability + more auditing, RAID, ECC RAM, FAST, Hadoop
- recommendations for more research, cost models, strong adversaries, erasure codes, lower bandwidth for auditing

END OF OUTLINE

OUTLINE WITH TEXT IN IT

Abstract

How to protect a large, valuable, digital document collection?

This article addresses the problem of ensuring information integrity over long periods, and against a diverse range of real-world legal, organizational, technical, and economic threats. Rapid advances information technology have shifted the economics of information production, transmission, and storage -- resulting in a vast amount of information that is increasing stored online or near-line. This shift in information storage changes both the profile of threat to integrity, and the set of feasible methods for mitigating these threats. We develop a general event-based simulation framework that can be used to flexibly and reproducibly simulate the effectiveness of various methods of storage, replication, auditing, and transformation against a diverse portfolio of threats to data loss. We then apply this framework to a range of scenarios that are representative of common baseline threats and current storage technology. From this we derive general guidance for the use of replication, platform diversification, fixity metadata, integrity auditing, compression, and encryption where long-term integrity is desired.

- summary of specific recommendations

TBS

Thou shalt keep multiple copies of thy documents.

Thou shalt audit thy documents fully and regularly, and keep them healthy.

Thou shalt lovingly squeeze and compress thy documents, that they may be better protected from the elements.

Thou shalt cloak thy documents, if they be shy, in secret robes to keep them from prying eyes.

Thou shalt not attend disk dealers who bear false witness of their reliability.

Thou shalt respect and monitor the independence of thy vendors.

Thou shalt befriend more vendors than thou currently doth engage, for they may be friends in lean years of woe and hardship.

Thou shalt engage with thy community to develop standards for the benefit of all.

Thou shalt foresee unforeseen circumstances.

Motivation

- Information production is rapidly increasing.
- The vast majority of the world's data is stored on rotating disks, 90% in 2007 and not changing very rapidly.
- Digital store changes the goal of preservation, from maintaining a constant optimal physical environment for documents to continual curation in order to maintain document integrity and understanding.
- Information systems are generally designed for immediate use, not for long term access and understanding. Much of the information generated is at risk.
- The changing economics of digital production make it difficult for any one institution to safeguard everything it uses.
- The changing economics also provide opportunities for preservation: replication, cloud services, auditing, and a variety of storage options.
- Curators are faced with a set of choices: storage media, replication, cloud vendors, auditing strategy, encryption, compression. There is currently no systematic guidance based on quantitative models.

Problem Definition

- Maintain understanding of large digital collections over time.
- Choose strategy for collection storage quality, replication, auditing, repair, formatting.
- Risks to collections come from a variety of threat types.
- Problem: Keeping risk of object loss fixed: what choices minimize $ in storage, network, etc.
- "Dual" problem: Keeping $ fixed, what choices minimize risk?

Core Risks

TBS

- types and severity of threats
   Various Threats to Library Collections (in the square brackets)

1. Document rot (bit rot) on disk.
2. Environmental failures that accelerate document rot.

3. Server failures that destroy sets of documents.

4. Lack of independence of servers.

5. Business failures: a single business failure may affect more than one server, due, e.g., to consolidation or cross-financing. That is, servers that appear to be independent may not be financially independent in practice.
6. Economic failures.

7. Attack on collection, institution, subject matter.

- range of error rates, wide, what's plausible

Structure of the Model

Our preservation model includes a few very simple objects and operations.

- A coherent aggregation of information is represented as a *document*. A document may be of any size.
- A set of documents forms a *collection*. A collection may include any number of documents. All documents in a single collection are treated equally by the owner of the collection and the service storing a copy of the collection.
- A collection belongs to a *client*, e.g., a library, that wishes to preserve it for the long term.
- The client has two strategies available to preserve the collection: to keep a small number of *copies* of the documents on very high quality storage, or to replicate many copies of the documents using available commercial data storage facilities. Very high quality storage may be very expensive or actually unobtainable; multiple copies on commercial storage may also be expensive, but such storage services are available commodities.
- Documents are stored on *storage servers*, and currently reside largely on rotating disk memories. Storage servers attempt to make their storage more reliable by themselves using a variety of techniques, including duplication, mirroring, RAID, error correction coding, erasure coding, and other technologies that may become available.
- A client may *audit* a collection to determine if all the copies of documents stored on the server(s) are still readable. Such auditing may be accomplished in a variety of ways; see the section on Auditing Strategies.

- An error in storage may cause a document or some portion of it to become unreadable. (For simplicity, we model that a single error in a document makes the document unreadable.) If all copies of a document are unreadable, then the document is permanently lost.
- Errors causing individual document failures are *independent* of one another.
- Errors causing individual document failures are *silent*, that is, they are *latent errors*. They can be discovered by the client only when the document is accessed for reading or for auditing.
- *Small data errors* that damage a single document are distinct from *larger storage failures* that damage an entire disk, RAID set, or storage server.
- Errors of disk data occur in small regions, e.g., blocks of data or small groups of blocks of data. These data errors within a storage service occur randomly and independently among the disk resources of the storage server.
- Documents may occupy more or less storage depending on their *size*. Since failures of blocks of storage are random and independent, a failure is more likely to be located within a large document than within a small one.
- A storage server may fail and cause all the data stored on that server to be lost. Such failures are random and occur at some rate. The rate may vary due to exogenous circumstances. Major storage failures are very rare events compared with individual document failures.
- Storage servers are independent of each other. Each server has a characteristic rate of failures of blocks of stored data. Different storage servers may have different failure rates.
- It is possible that the failure rate within a server is not constant over time. However, over suitably short intervals, a changing rate can be approximated by some mean value in the interval.

(Format obsolescence is not an inherent part of this model, but forms of it may be modeled through extensions involving associated failure. See below.)

Simple and Complex Failures

Simplest Case: Sector Errors

- An error in the storage corrupts a document sector. Errors arrive randomly in a Poisson process. A cosmic ray striking a disk or memory cell is a good model for this type of error.
- If the error occurs in a sector occupied by a copy of a document, that copy is corrupted. For the purposes of this study, we consider the copy to be lost. Manual repair by human inspection is not considered here.

- Errors are silent, that is, no one notices an error until someone tries to read the document and discovers that it is lost.

Similar Case: Glitches

- A *glitch* is a temporary, short-lived, condition that impacts a single server and increases the sector error rate on that server for some short interval.
- What types of glitches might occur in server farms? HVAC weakness or failure; environmental contamination by chemicals or particulates; radiation; electrical noise; and similar. In general, these conditions are not fatal to the server overall, but degrade the integrity of data storage.
- Glitches arrive at random intervals in a Poisson process, and have limited duration.
- Glitches are local phenomena, limited to a single server.
- The effect of a glitch is simply to increase the sector error rate for a short period. Glitches, like the errors they induce, are silent. In this study, we find that it is hard to distinguish glitch-induced increases in error rates from random variations in performance.

Complex Case: Shocks

- A *shock* is a more serious condition affecting storage servers. This is a temporary, short-lived, or possibly permanent, increase in the likelihood of death of a server. All servers are considered to have finite lives (of random length), though the lives may be very long. A shock reduces that lifetime.
- A shock may kill a server immediately, or it may simply increase the likelihood of failure of that server.
- What types of shocks might occur? Natural disasters, such as fire, flood, earthquake, volcano, meteor, etc.; economic downturnsin consumer and financial markets; regional wars; government interference; administrative errors such as billing and credit arrangements; and so forth.
- Shocks arrive at random intervals in a Poisson process, and may be immediately fatal to a set of servers or may simply reduce their life expectancies for a period.
- The impact of a shock is modeled as silent, that is, no one notices the death of a server until someone tries to retrieve a document from a dead server. Note that a server failure results in the loss of all document copies stored on that server.
- Shocks may be regional or administrative phenomena that affect more than one server at a time. One particularly subtle cause of a shock affecting multiple servers is lack of independence of the servers, due to corporate mergers, collocation of server farms, dependence on large power grids, etc.

- When a server is lost, the client is required to find a new server and populate it with the whole collection -- or at least the parts of the collection that can still be found on the remaining servers.

A Very Simple Cost Model

Many storage vendors may be available to a client, each with charge schedules. For the most part, vendors will charge for storage and bandwidth.

- A charge per month per byte stored (usually gigabyte or petabyte).
  - The cost of storage may vary by "quality" of storage, based on its typical error rate or perhaps on speed of retrieval access.
- A charge per month per byte sent in or out ("ingress" and "egress" charges).
  - Bytes sent do not distinguish between user access for normal retrieval and administrative access for auditing.
  - The cost may vary by speed or reserved bandwidth (Mbps).
- Charge schedules may include quantity discounts for storage and transfer

For the purposes of this study, a client will store a collection on a set of servers of the same "quality" level. Documents with differing quality requirements are considered separate collections and are stored and managed separately.

Basic Testing Process

Simulations of the aging of document collections are done in several *runs* through the simulation program. The process is approximately as follows.

- The model employs one client library with one collection that contains a large number of documents. Results for multiple clients, for clients with multiple collections, or for varying number of documents can be extrapolated from this simple case.
- The client assigns some number of external servers to store a copy of the collection of documents. Each server is supposed to maintain, and be able to retrieve on demand, an authentic copy of any document in the collection.
- The model permits both a range of replicated copies of documents, and a wide range of reliability properties of storage servers on which the copies are stored.
- Small errors occur randomly in the storage of the servers. When an error occurs where a document is stored, that document is considered lost.

- When a document fails on a server, the failure is silent. The client is not immediately informed of the failure. Indeed, the server might not be able to sense that the latent failure has occurred until it tries to retrieve the document on a request from the client.
- From time to time, the client may audit the collection by testing the copies of documents on servers to ensure that they can still be read and are still valid copies. The model supports this process of auditing the collection. Audits can be scheduled and performed using a variety of schedules and strategies.
- If in the process of auditing the collection, a document is found to have failed on a server, the client will refresh the failed copy if an intact copy remains on any other server. If no other intact copy remains, the document is considered to be permanently lost.
- At the end of the simulation time period, the model assesses all copies of documents on all servers to determine how many documents have been permanently lost.

This entire simulation cycle is repeated a number of times using different values to seed the (pseudo-)random number generator that drives the simulation. The numbers from all runs are collected and presented in tabular form and in graphical summaries in the supplemental material.

The model does not consider directly the costs of storage or bandwidth. These factors vary widely and change rapidly. Any conclusions based on specific numbers would become obsolete very quickly. However, some possibilities for minimizing or smoothing bandwidth consumption are considered.

The model permits great variety in the structuring of the simulations. Not all possibilities have been used in the simulations reported here. A number of simplifications were used to reduce substantially the size of the problem space to be searched. For example, we report results for a single client library with a single collection of documents. Given the assumptions that clients, collections, and documents all are independent of each other, results for multiple clients, multiple collections, and varying sizes of collections can be extrapolated from these results. Similarly, early testing revealed that document size did not interact with other simulation parameters in any unforeseen ways; so we have reported results for a limited number of document sizes, along with guidelines for scaling document size relative to other adjustable parameters, such as sector error rate. See the section on "Simplifying Assumptions."

The model does not attempt to characterize individual disk drives, RAID sets, erasure code sets, or any other physical storage entities. Our investigations have dealt only with small errors that corrupt stored documents and large failures that destroy entire storage servers.

Data Representation

The likelihood of an error in a disk bit or sector, or even the failure of an entire disk, is a very small number with many zeroes before the first significant digit. We choose to invert the error rate into a function of lifetime of that bit, or of a sector containing many bits. We feel that lifetime is more accessible to the reader and less error-prone than error rate. For example, a probability of a bit failing in a year of 10E-15 becomes a mean lifetime of 100E12 years.

We note that, expressed that way, the example figure does seem to be excessively optimistic; the age of the universe is currently estimated to be only 14E9 years. Data on such a disk would be effectively immortal; that does not correlate with experience and would not require a protective strategy.

Also, we do not consider bit error rates, but choose to nominal sector size of 1MB. All lifetimes are expressed relative to such sectors.

Because of the extremely wide range of lifetimes being considered here, we also choose to plot lifetimes on logarithmic scales.

Lifetime Expressed as Half-Life

Also, we have chosen to use the half-life of objects (sectors, servers) rather than the more common "mean exponential lifetime" used in most statistical formulas. Mean lifetime is a standard statistical measure, but is not intuitive to the non-expert. "By the end of an MTTF period, approximately 63% of the units will have failed" is not easily understood by most non-statisticians. (If we assume Poisson arrivals, the probability of failure in one average lifetime is (1-1/e), about 63%.) We have chosen for all simulations and tables of results to express lifetime instead as half-life. "By the end of a half-life period, approximately half of the units will have failed" is easier to understand, and should be familiar to most people from examples of radioactive decay, chemical reactions, drug absorption, etc.

For Poisson processes (with exponential arrivals), the relationship of half-life to exponential lifetime is simply

```
(half-life) = (lifetime) * ln(2)
```

Auditing the Document Collection

Why Audit?

Auditing the collection to test the validity of remote copies of documents can greatly reduce permanent document losses over time. The auditing process actively patrols for errors before they cause permanent document losses, and corrects them whenever possible. A number of strategies for auditing are possible, and some are measurably better than others.

In all cases, when a document copy is found to be absent (or corrupted), the auditing process attempts to replace the missing copy with a fresh copy obtained from another server. If there is an intact copy on another server, then the missing document is repaired and the process continues. If there is no other intact copy, then the document is considered permanently lost.

Auditing is essential to maintaining the health of a collection. This is the method by which errors are detected and corrected. Without auditing, errors tend to build up in a collection and eventually cause some permanent document losses, regardless of how many copies of the documents we keep. We can think of the auditing process as health care for electronic documents: minor problems will be found and fixed before they cause permanent damage. Of course, it will always be possible for unlikely juxtapositions of errors to cause a document to be lost, but regular auditing of a modest number of copies can minimize permanent losses.

Simplest Case: No Auditing

Without auditing and repair, errors will always cause copies to be lost. Multiple copies of a documents will reduce the likelihood of permanent loss, but the number of copies needed to forestall loss depends directly on the lifetime of storage sectors. With high storage error rates, for instance, sector half-lives less than 10E6 hours, no reasonable number of copies can prevent large numbers of permanent losses. Even with sector half-lives in the range of 10E6 hours, ten or more copies are needed to keep likely permanent losses near zero. If the storage is much more reliable, with sector half-life in the range 100E6 or 1000E6 hours, still at least four or five copies are needed to minimize losses. Given the uncertainty of storage error rates, particularly across servers and over time, the larger numbers of copies are probably required to protect collection documents.

Note that all the figures stated here for times and lifetimes are based on arbitrary scales chosen for the simulations. The numbers should not be applied literally to your situations or experiences. Please see the section on "Simplifying Assumptions" for details on time scales, error rates, storage and document sizes, etc.

Simple Case: Independent Failures & Just Plain Copies

Just make copies -- no auditing? TOO MANY COPIES REQUIRED

TBS

TODO: Why did we choose this spectrum, which goes from rusty garbage-can lids to immortal disks.

It must be stressed that a client should choose a strategy that works for *somewhere* on the broad spectrum of reliability, because one never knows where servers lie on that spectrum. The service level agreements of cloud storage server vendors are not likely to specify precise bit error rates (nor liability for lost documents). And error rates can change in the short term due to environmental glitches, bad disks, and such.

How many copies do you need if ...

Auditing, even at relatively low rates, changes the picture entirely. With a reasonable amount of auditing, five copies of a collection suffice to protect the collection from loss across a very wide range of error rates, starting with sector half-lives greater than 3E6 hours.

[FIGURE: FIVE COPIES ARE ENOUGH]

[FIGURE: ANNUAL AUDITING IS ENOUGH]

Common Auditing Strategies

- **Total auditing**: test all copies of all documents in the collection. This *auditing cycle* is usually repeated at regular intervals, such as annually, quarterly, monthly, etc.

- **(Systematic) segmented auditing**: divide the collection into several segments, and test one segment at each interval. The whole collection is visited during the audit cycle, but only one segment at a time. For example, the collection may be divided into four segments; if each segment in turn is audited at quarterly intervals, then the entire collection will have been audited at the end of a yearly auditing cycle. Of course, more or fewer segments can be used: with an annual frequency, one segment is the same as annual auditing; twelve for monthly segmented auditing, and so forth.

Auditing cycle rates may vary from quarterly (four times per year) to biennial (once every two years). Auditing more frequently than annually does not seem to confer much additional protective benefit. This depends, of course, on the number of copies of the collection that are placed on independent servers. Please consult the tables in the supplementary material for details.

Note that segments need not be fixed portions of the collection. Each segment of the collection might be selected at random when its turn comes, so long as the random selection is made *without* replacement over the audit cycle. This ensures that every document in the collection will be audited exactly once during the complete cycle.

- **Random auditing**: at some interval, audit a random subset of documents chosen from the collection. This often is expressed as, for instance, "audit ten percent of the documents every month." The difference between this random strategy and segmented auditing is that the random selection is chosen *with* replacement. Thus it is likely that some documents will escape auditing entirely for long periods.

  It is important that, during each audit cycle, every document copy be examined. Auditing strategies that examine random subsets of the collection sampled with replacement, are not so effective at protecting the collection. For example, a strategy sometimes suggested of auditing a random ten percent of the collection every month, where the random selection is made with replacement, is slightly less effective than a single, total audit once a year.

- **Auditing by popularity**: divide the collection into sub-collections that represent varying levels of document usage, e.g., small sub-collections for the documents most frequently accessed, medium size sub-collections for documents of intermediate popularity, and large sub-collections for documents rarely accessed. Permanent losses in the popular sub-collections would have much greater negative impact on the customer base. To reduce the likelihood of such expensive losses, the more popular (small) sub-collections of the collection can be audited more frequently than the others with little increased cost in bandwidth and time.

Our simulations tested many auditing strategies, including total, segmented, and random. Tests differed in cycle frequency and in the parts of the collection audited during each segment or cycle.

- All tests occurred on regular schedules.
- Auditing cycles varied from monthly to biennially.
- Segment counts were either one, two, four, ten, or fifty (corresponding to annual, semi-annual, quarterly, monthly, or weekly audits).

- Segments were chosen either systematically (the first quarter of the collection, the second quarter of the collection, etc.) or by uniform random selection, with replacement, of part of the collection.

Observations

Large vs Small Collections

The simulations were done with a fixed collection size of 10,000 documents in a collection. All the statistics we report are stated as percentage of the documents lost. These proportions can be scaled to different collection sizes by linear extrapolation.

Large vs Small Documents

Most of the simulations were done with documents of a fixed size. Because document sizes may vary over a wide range depending on content, format, compression, etc., we ran additional tests over a wide range of document sizes. Results matched expectations based on the Poisson distribution, that document size and error rate (i.e., sector lifetime) vary inversely. Increasing document size by a factor of N and decreasing the error rate by the same factor of N (or, equivalently, increasing the sector lifetime by the same factor of N) result in the same distribution of errors and therefore the same document losses.

The table in Exhibit XX shows the results of tests over a wide range of document sizes, from 5MB to 5,000MB, and a comparable range of sector lifetimes, from 2E6 to 10E9 hours.

Large vs Small Storage Structures

Storage servers will store documents on storage arrays of varying size, depending on disk size, RAID or erasure coding level, and other administrative factors. We ran tests on storage structures varying in size by a factor of 10 and found no differences in document losses.

One minor note: using very large storage structures with small documents result in storage extents that are sparsely populated. This reduces the efficiency of the simulation programs, because many of the simulated "cosmic ray" errors striking the disks land in unoccupied areas that do not affect any documents.

Total Auditing is Essential

- *Observation*: Total auditing of the collection is highly effective at reducing document losses. Without auditing and its attendant repair of damaged documents, minor errors will cause a stored collection to continue to decline over time with no barrier to extinction. No number of redundant copies without auditing, certainly no *reasonable* number, will prevent significant document losses over a long period. In addition, shocks to the system may cause servers to fail, thus reducing the actual number of active copies of a collection and accelerating further deterioration.

Exhibit XX shows unaudited document losses over long periods with large numbers of redundant copies.

However, even a modest regimen of auditing and repair can minimize damage to a collection, across a huge range of server quality, long periods of time, and unpredictable adverse shock conditions.

- *Observation*: The effectiveness of auditing is robust across a wide spectrum of storage quality (i.e., document error rates) and short term variations in storage quality.

Exhibit XX shows how annual total auditing of a small number of copies can keep a collection healthy across a very wide range of server quality.

- However, auditing strategies may not be robust to severe associated failures that compromise multiple servers over short periods. Associated server failures -- whether due to disasters, economic downturns, clerical errors, or lack of independence of servers -- can remove more than one server from service between audit cycles. This reduces the number of active replications of the collection, leaving the collection more vulnerable to minor errors until it is repaired by auditing. If severe shock conditions are anticipated, it may be necessary to increase slightly the number of redundant copies or the frequency of auditing of the collection.

Other Types Of Auditing

- *Observation*: Random auditing, where segment contents are selected with replacement, is less effective than total auditing or, equivalently, segmented auditing *without* replacement. Selection of documents randomly *with replacement* will inevitably miss some documents entirely while sampling others more often than needed.

Exhibit XX shows the higher loss rates for collections audited randomly with replacement as opposed to total auditing.

Segmented Total Auditing Slightly Better

- *Observation*: Total auditing in multiple segments is slightly more effective than auditing the entire collection as one segment with the same cyclic frequency; e.g., auditing a quarter of the collection each quarter is slightly more effective than a single annual audit of the whole collection.

  Note that auditing in a number of segments has two additional advantages: 1. It spreads the bandwidth requirements for auditing throughout the audit cycle. This can reduce recurring (monthly, quarterly) egress charges for large audits. 1. It can find a dead server more quickly. A dead server can be detected only during auditing when a document repair fails. Since all servers are examined quarterly, for instance, rather than annually, documents are exposed less to permanent loss.

- *Observation*: Across a wide range of document error rates, increasing auditing frequency beyond a certain point shows little improvement. Segmented auditing with multiple segments per audit cycle can be advantageous, but returns may diminish with excessive frequency. For example, in most environments, annual total auditing in four segments (quarterly) improves loss rates over one segment (annually). However, the further improvement from monthly or weekly audit segments is minimal.

Recommendations

TBS

For the Collection Owner

- Maintain at least 5 copies of the document collection.
- Use systematic quarterly auditing, that is, audit the entire collection annually, but perform the audit in four segments per year.
- Use compression, with known algorithms, wherever possible on all documents.
- Do not trust MTBF and other similar measures stated by anyone. Choose a strategy to protect your collection over a very wide range of storage quality and adverse

future conditions.

[Remaining questions] - What can we say about document size? - What can we say about collection size? (E.g. twitter corpus) Error rates matter either if collection is big or long-term? - What can we say about encryption? - What can we say about increasing replication in the face of particular correlated threats?

TBS

For digital preservation commons

- Share experience on error rates discovered in auditing, across vendors, numbers of copies, auditing frequency, etc.
- Share experience information on the reliability of cloud storage vendors.
- Share information on experience of correlated failures, exogenous and endogenous.
- Develop standards to improve efficiency and reduce costs.
    1. Develop agreements with cloud vendors for cryptographic auditing that reduces data egress.
    2. Develop standards for reporting failure rates.

TBS

- Parallel between strategy of less-reliability + more auditing with original RAID (inexpensive disks); RAM error correction; FAST array of Wimpy Nodes; Google hardware-failure tolerant hadoop architecture

---

Supplemental Material

ALL TBS

- details of model
- simplifying assumptions and scaling, Poisson IID, metric years, partitioning of simulations, non-repairable documents
- Poisson assumption, calibration tests
- words about error rates

A Few Words About Error Rates

One basic question should be answered before embarking on such simulations: what is the failure rate of stored documents? This is a difficult question due to a lack of real data.

There is data on the failure rate of individual disk drives over time. Thanks to Backblaze, Google, and others, there is some published empirical data on failure rates of disk drives of recent technology vintages. [citations] These figures refer to replacements of entire disk drives during the useful life and wear-out periods of device use. That is, they exclude infant failures but include mid-life and senescence. Unfortunately, we do not get much information on the rates of sector failures, bad block replacements, and so forth.

There is an estimate of unrecoverable failures on consumer-grade SATA drives that is often mentioned in the industry: Pr{a bit fails during a year} = 10E-15. This looks like a small number until one calculates that a single 4TB drive, very commonly deployed today, contains about 40E12 bits of data, plus essential metadata. [I'm pretty sure that this figure is for SCSI and FC disks; the number quoted for consumer grade SATA disks is 1E-14, which is very different. And these numbers probably are not per year, but just per bit.] [Have to find the reference for this number, and have to check the accuracy, too.] Even ignoring the magnitude of the storage involved, it is hard to grasp that the average lifetime of a bit is a thousand times longer than the age of the universe.

We have not encountered data on the performance of disk drives or blocks in RAID and erasure coding configurations, the effect of pre-emptive data scrubbing, etc.

Data errors are always assumed to be silent to the client that owns the document. Active searching for and correction of errors is necessary to ensure continuing data integrity. Note that the multiple-copy storage and auditing procedure explored in this paper is analogous to RAID storage with data scrubbing, but done at a document level rather than a block level.

The Representation and Scale of Error Rates

The likelihood of an error in a disk bit or sector, or even the failure of an entire disk, is a very small number with many zeroes before the first significant digit. We choose to invert the error rate into a function of lifetime of that bit (or sector containing many bits). Thus a probability of a bit failing in a year of 10E-15 becomes a mean lifetime of 100E12 years. Expressed that way, the figure seems excessively optimistic. (The age of the universe is currently estimated to be 14E9 years.) Data on such a disk would be effectively immortal, and that does not correlate with experience.

We agree with Rosenthal (2010) and others that such estimates are merely marketing projections that are not based on empirical data. Using simulations to investigate such nearly immortal disks would be expensive and fruitless. If there are no errors, then no

protective strategy is needed. However, the statement "no errors" does not correlate well with practical experience.

Where, then, to search for information about the effectiveness of replication and auditing of large collections of data? We choose to investigate a region that more nearly matches the experience of computer users, where disks and disk files have lifetimes somewhere between the fruit fly and the universe.

- Error rates are non-zero: there are *some* errors rather than none.
- Larger data is likely to encounter more errors.
- Error rates are not so high as to be crippling to normal usage.

The supplementary material includes comparisons of theoretical and empirically observed loss rates for a wide range of error rates. The theoretical figures are based on simple independent Poisson arrivals of document failures, based on sector lifetime, sector size, document size, and simulation length. Empirical numbers are derived from repeated runs of simulations with the stated parameters, with simple document aging and no auditing. Even with very small samples (twenty runs) the empirical numbers agree very well with the theoretical predictions.

Why Did We Choose This Range of Error Rates

The region of error rates that we investigate generates enough errors to evaluate the impact of storing multiple copies and the impact of various auditing strategies. Our conclusions describe storage and auditing strategies that are robust over very wide ranges of error rates (and the corresponding ranges of bit/block/disk lifetimes), spanning approximately four orders of magnitude.

A back-of-the-envelope calculation is called for here. The scale (of disk quality) that we have used for most of the simulations ranges from a sector half-life of two or three megahours (2E6 or 3E6 hours) up to one thousand or ten thousand megahours (1E9 or 10E9 hours).

- A modest size disk today might contain one terabyte of data. The "sector size" we have used is one megabyte (1MB). Thus our hypothetical modest disk would contain about a million such "sectors".

TODO:RBL: correct the numbers in these several bullets.

- At the extreme low end of the disk quality scale, the sector half-life is, say, two megahours. 1E6 sectors, each with a half-life of 2E6 hours, scaling down for legibility, is comparable to 1E3 sectors with a half-life of 2E3 hours. Every 2,000

hours, about one half of the 1,000 sectors would incur an error. That 2,000 hours is less than a quarter of a year. A disk with a lifetime this low -- or an error rate this high -- would not be usable.

- At the high end of the disk quality scale, the sector half-life is 1000E6 hours. Performing the same scaling, the 1E6 disk sectors with half-lives of 1000E6 hours are comparable to 1E3 disk sectors with half-lives of 1E6 hours. Every million hours -- about 110 years -- half of the sectors will incur an error. In the first year, only xxx % of the sectors will have an error; after ten years, xxx %.
- In the middle of the scale, the sector half-life is 100E6 hours. Again, performing the same scaling, the 1E6 disk sectors with half-lives of 100E6 hours are comparable to 1E3 sectors with half-lives of 100E3 hours. Every 100,000 hours -- about 11 years -- half of the sectors will incur an error. After the first year, about xxx % of the sectors would be expected to have an error.
- In a more realistic area, consider half-lives of 30E6 to 50E6 hours. See table below.

TODO:RBL: table goes here. strictly poisson calculations. - horiz: sector half-life, 3, 10, 30, 50, 100, 1000 mh. - vert: pct sectors lost after 1 yr, 2 yrs, 5, 10; asterisks for not usable.

Relationship of MTBF/MTTF to Block Error Rates

The inverse of error rate is usually expressed in terms of MTBF or MTTF, and, initially, we expressed all parameters as mean exponential lifetime. But MTBF and MTTF are hard even for most experts to grasp, and uninformative or misleading for non-experts.

The disk manufacturing industry tends to express the device lifetime as MTBF or MTTF. This is an expected (mean) exponential lifetime for the *device*, but that does not give much information about the lifetime of data in individual files, blocks, or bits on the disk. There are several layers of error detection and correction in storage systems that tend to mask small errors in disk data and obscure the relationship between small data errors and drive failures.

1. Block error correction on the disk. Disk data is written with error correcting coding that can repair small errors when reading data sectors.
2. Bad block remapping in disk controllers. Smart disk controllers can take unreliable sectors out of service, replacing them with more reliable sectors from a pool of spares. Such remapping is usually transparent to most software, but it may or may not be able to rescue the data residing on the bad blocks.
3. RAID and similar redundant recording of sector data. Sector data may be recorded redundantly in mirror sets, or recorded partially in multiple versions using parity techniques, and so forth.

The use of any or all of these techniques makes it very difficult to assess the relationship between drive failure statistics and block level errors. Small correlations have been found between S.M.A.R.T. error reporting statistics and subsequent drive failures, but there is no clear causal connection. [cite Wikipedia article on S.M.A.R.T.]

It must be stressed at this point that RAID and similar techniques protect only against *drive* failure; they do not protect against bad bits, blocks, or tracks when reading from a drive. If a disk drive fails completely, RAID and erasure code techniques can rebuild the data on that drive from redundant data stored on other drives. However, while a drive is still in operation, individual bad blocks on a drive will still read as bad blocks until the drive is removed from service and its data recovered, *if possible*, from the remaining drives in the redundancy set. The data on a disk drive, even in a redundancy set, can deteriorate incrementally over time and cause documents (in our case documents, but files in general) to be altered badly. If deteriorated data cannot be repaired by block error correction techniques in the disk controller, then the data of the file or document may be permanently lost or corrupted.

The rate of deterioration of data is not directly related to the lifetime of the disk drive itself. Drive failure may result from failures in the mechanisms of moving parts, the controller electronics, circuit boards, connectors, etc. Data errors may result from surface wear, chemistry, radiation, lubrication, magnetic interference, and so forth, none of which seems very clearly related to the major causes of drive failure. (Physical contamination from airborne dust, chemicals, or humidity may be an exception.)

What is MTBF, Really?

MTBF, Mean Time Between Failures, is a slippery notion, much touted by marketing departments and viewed warily by users. If the object in question is removed from service after only one failure, as is the case here with documents, it is perhaps more appropriate to speak of MTTF, Mean Time To Failure. MTTF is intended to be equivalent to the mean lifetime (before failure) of the object, and, if one assumes that failures are a Poisson process, then MTTF is the mean *exponential* lifetime of the object.

How is MTTF calculated before it is published? Several methods might be used, including at least the following.

1. A predicted value based on engineering characteristics of the mechanism, component parts, expected wear patterns, and so forth. Depending on the complexity of the device and the manufacturer's understanding of its components and usage patterns, this can be a very complex and, frankly, questionable estimation.
2. Failure data from life testing, often of large numbers of devices over long periods. Such testing may be done by the manufacturer in-house, or in field testing of early

deployments, or by consumers who use large numbers of devices and track failures carefully.
3. Failure data from accelerated life testing. It is often assumed that operation under high temperature or thermal cycling or high speeds or other stress conditions will cause devices to fail predictably prematurely. For some classes of devices, accelerated life testing has proved to be useful and accurate.
4. Failure data from warranty failures returned during a service period. This may be assessed by the manufacturer or by users of large numbers of devices.

Most non-marketing literature considers MTBF estimates from manufacturers to be exaggerated considerably, by factors of three or four at best. The annual failure rates of disk drives in large collections of drives are much higher than would be expected based on published MTBF estimates of 1E6 hours or more. [citations]

Even if we understood the source and accuracy of stated MTTF estimates for disk drives, we would still not have information about individual sector failures within a drive that cause document failures, nor the relative frequencies of sector failures versus drive failures.

A Few Words About the Graphs

The graphs are generally structured as follows:

- The X axis is the sector lifetime expressed as half-life. As half-life increases from left to right, the sector error rate, which is the inverse of lifetime, decreases. Trying to express the sector life as an error rate was found to be confusing because they are generally small fractions.
- Sector half-life is stated in megahours. The range is typically from 2 or 3 megahours to 1,000 or 10,000 megahours.
- The Y axis is the document failure rate, stated in percent of the collection size. The collection size is 10,000 documents.
- The graphs include a horizontal line at the 1% level, that is, one per cent of the documents permanently lost in the simulation.
- Except as noted, the duration of the simulation is ten years, which is nominally 100,000 hours.

- Both axes on the graphs are drawn with logarithmic scales with base ten. Because some of the sampled numbers are actually zero, we have biased the loss rate numbers away from zero by a small amount. Typically, zero is rendered as 10 parts per million (ppm), which shows up as 0.001% on the graphs. This can be modified with a simple

change in the code that draws the graphs; we have tried 1 ppm, also, and the visual results are very similar.

- overview of software architecture
- installation on AWS

- how to model various scenarios

A wide range of real-world threats may be modeled through varying the parameterization of the model

- pointers to software how-to docs

---

where to these fit?

- limitations

A Few Words About the Graphs

The graphs are generally structured as follows:

- The X axis is the sector lifetime expressed as half-life. As half-life increases from left to right, the sector error rate, which is the inverse of lifetime, decreases. Trying to express the sector life as an error rate was found to be confusing because they are generally small fractions.
- Sector half-life is stated in megahours. The range is typically from 2 or 3 megahours to 1,000 or 10,000 megahours.
- The Y axis is the document failure rate, stated in percent of the collection size. The collection size is 10,000 documents.
- The graphs include a horizontal line at the 1% level, that is, one per cent of the documents permanently lost in the simulation.
- Except as noted, the duration of the simulation is ten years, which is nominally 100,000 hours.

- Both axes on the graphs are drawn with logarithmic scales with base ten. Because some of the sampled numbers are actually zero, we have biased the loss rate numbers away from zero by a small amount. Typically, zero is rendered as 10 parts per million (ppm), which shows up as 0.001% on the graphs. This can be modified with a simple

change in the code that draws the graphs; we have tried 1 ppm, also, and the visual results are very similar.

- long term bit rot

- Long term -- bit rot

  o Using Poisson closed form -- if expectation loss is just less than .5 doc in 10 years . Expect not to see any doc loss in 10 years ... how long to see a doc lost... how long to see 1% failure.

  o Interaction -- fragility of big documents

  o [FIGURE] (above -- suppose we increase doc size by 10, by 100, by 1000 -- how many expected doc lossed in the same period)

  o Cite to Rosenthal previous results on this

- medium term if error rates are uncertain

- Medium term -- if storage error rates are uncertain

  o Storage error rates are difficult to verify

  o [FIGURE] How long for failure of 1% as error rate increases? (TODO:RBL: Q: Sounds like a one-way table, for fixed document size, IV = error rate, DV = hours before loss exceeds 1%.)

  o How to interpret claimed storage error rates

```
- What are the limitations of how MTBF is measured?
- Given an MTBF, what is the possible bounded range of half-lives?
```

- specific types of threats: finance, billing, adversaries internal and external, HVAC, environment, software failures, admin errors, hardware batches, government censorship, economics

Type of Threats

- Server-side Billing Failure
- Server-side Financial Failure
- Unsophisticated adversary outsider attacker
- HVAC Failure/anticipated environmental problem
- Unanticipated Environmental Catastrophe (including local war)
- Local software failure
- Admin failure
- Hardware batch quality
- Formal Government Action
- Powerful External Adversary
- Economic Recession
- Limited Internal Adversary
- Curatorial Failure/Client error

- Common software failure

- modeling associated failures, types and ranges

Modeling Associated Failures

Sources of failures are modeled as a stochastic processes, in a hierarchical model

	Logical Block	Server (Provider) Glitch	Global Shock
Represents	failure of logical block within physically raided storage	event affecting reliability of single provider -- increase logical block error rate	event affecting reliability of multiple providers -- increase server failure rate
Distribution	Poisson IID	Poisson IID	Poisson IID

	Logical Block	Server (Provider) Glitch	Global Shock
Duration	Instantaneous and permanent	Possibly bounded Duration %GLITCH_MAX_LIFE%, Exponential Decay	bounded duration
Effect	loss of single block of single copy of document	Increases logical block failure rate -- parameterized level of impact	increase likelihood of death of k servers
Detection	Loss is detected on audit	Server error itself detected on audit iff. block error rate > %CLIENT_SENSITIVITY (not yet implemented)	Invisible (detected only through effects)
Notes	Failure rate is not known precisely to client	Induces additional block failures, induces non-stationary errors (temporal clusters).	Induces server deaths, and temporal cluster of server failures among

- things not modeled

A number of sources are not modeled, but are assumed to be addressed through storage practices:

- Bathtub curve, accelerate start and end-of-life failures. The model is conditioned on good systems administration practice is in place, including equipment burn-in and scheduled replacement within the expected service life. Thus the error rate observed during the service life of the equipment should not be subject to these failure.
- Raid configuration characteristics, internal raid errors. Standard storage practice may include low-level physical redundancy. Thus the reliability of a logical block will be better than that of an underlying physical storage. When using the model, one should calibrate error rates based on the implied or observed failure rates of logical blocks -- each of which may be represented by redundant physical storage.
- Media format obsolence -- we assume good practice -- migration to new media before end of life.

- Unmanaged File Format obsolescence.
    - Management can mitigate file format obsolescence where documents are stored in multiple formats (or multiple indepent readers) and tested for format characteristics at audit. Failures occur at the level of an entire document, but the threat of loss could be modeled as with a block error rate implying a certain document failure rate for fixed size documents, where the number of servers represents the number of independent formats stored.
    - Unmanaged format obsolescence cannot be addressed through redundancy, etc., and will not show on audits.

What we are not modeling - Sophisticated adversaries that attack the auditing mechanism - Cascading failure/contagion - Environmental glitch that directly affect background rate of server glitch - Server characteristics are in a steady -state equlibrium -- characteristic of servers remain the same over time.

- threat matrix

Threat Matrix

A wide range of real-world threats may be modeled through varying the parameterization of the model

[Well, I'm fairly convinced that pandoc markdown cannot do complex lists inside tables, so we will have to render this sort of table in raw HTML.]

Model Level	Real World Threat Source	Used to predict ...	Use to derive ...
Logical Block

- loss due to media failure
- loss due to raid/internal replication characteristics and failure

| Document loss rate as a function of {number of replicas, auditing strategy, auditing frequency, block error rate} | Document loss as a function of * document size * format fragility - file compression - managed format obsolescence | | Server | - Server-side Billing Failure - Server-side Financial Failure - Unsophisticated adversary outsider attacker - HVAC Failure/anticipated environmental problem - Unanticipated Environmental Catastrophe (including local war) - Local software failure - Admin failure - Hardware batch quality | Document loss rate rate as a function of server error characteristics, given a fixed choice of {replicas, auditing, block error} Increased redundancy needed to maintain

fixed loss rate in presence of server errors, given recommended auditing and repair strategy |

| Global |- Formal Government Action - Legal action - Economic Recession - Limited Internal Adversary - Curatorial Failure/Client error - Common software failure | Document loss rate rate as a function of global error characteristics, given a fixed choice of {replicas, auditing, block error} | Increased redundancy needed to maintain fixed loss rate in presence of global errors |

Effects of encryption key escrow policies

Tradeoff between regional diversification and adding servers.

Server Error Parameterizations

Type	frequency	Impact	lifetime	Notes
Server Billing/Government Takedown	Medium	High (> sensitivity rate)	Instantaneous	Shock. Loss of entire collection on server
Financial (recession)	Low	High	Permanent -> simulation period	Shock. Loss of entire collection on server
Low Resource External Adversary	Low	Medium	Medium	Assume that adversary does not subvert audit
HVAC/power glitch	high	low	low	
Unanticipated Environmental Catastrophe	low	high	instantaneous	shock
Local Software	medium	medium	long	
Administrator Error	medium	medium	medium	
Hardware batch quality	Medium	Medium	Long	

Specific Parameter Value Ranges

Parameter	Range	Cause(s)	Low End	High End

Parameter	Range	Cause(s)	Low End	High End
Frequency	Low	Financial, Low Resource External Adversary, Unanticipated Environment Catastrophe;	5 years	20 years
Frequency	Medium	Local Software; Administrator Error; Hardware Batch Quality Defect, Server Billing/Government	1 year	5 year
Frequency	High	HVAC	0.3 years	1 year
Impact	Low	HVAC	10% reduction in half-life	33% reduction
Impact	Medium	Low Resource External Adversary; Local Software, Hardware Batch, Administrator error	33%	90%
Impact	High	Server Billing/Government takedown, Financial, Unanticipated Server Catastrophe,	1 server	50% of servers
Lifetime	Instantaneous	Unanticipated Environmental Catastrophe, Server Billing	Instantaneous	instantaneous
Lifetime	Low	HVAC/Power Glitch	1 Week	1 Month
Lifetime	Medium	Low Resource External Adversary	1 Month	1 Year
Lifetime	High	Hardware Batch	1 Year	3 year

- extensions

How Many More for

- Environmental

- Institutional Final Failure

- Recession

Extensions

Overview

- In this section we discuss extensions of the simulation results to document size, collection size, use of compression, managing multiple encryption keys, diversification against format failure risk, and protection against clandestine adversaries
- In each of these areas, the current simulation framework provides leverage under well-specified constraints

- Where these constraints may be violated in realistic scenarios, we discuss potential area research

- parameter value ranges

- large vs small collections

Large vs Small collections

- Assumptions: No additional assumptions needed. Percentage of failure is independent of number of documents in collection. However, given a low percentage expected document loss, the probability that at least one document will be lost depends on the number of douments in the collection: [formula]
- Additional Simulation: Formulation may be checked using simulations.
- Implications: For collections with huge numbers of documents, and where the absolute integrity of the collection neeeds to be maintained (for example, legal evidence) very low loss rates are desired

- Generalizations: document size

- large vs small documents

Large vs Small documents

- Assumptions: Without any additional assumptions needed, the relationship between document size and failure rate can derived, and follows the formula : [equation] (TODO:RBL: Q: Simple Poisson with clear list of parameters?)
- Additional simulation: No additional simulation is needed, however this formulation may be checked using simulations: [figure] (TODO:RBL: Q: Or a table, given that the errors are so small?)
- Implications: You can mitgate the risk of increasing size by a factor of [X] by adding [Y] additional copies. (TODO:RBL: Not a closed form calculation, requires actual tests.)

- Generalizations: The model assumes a single failure within a document destroys it. Documents may be redundant -- such that multiple hits are needed to completely destroy them. (TODO:RBL: Q: If we want to present real numbers for this case, we will need some new code.)

- compression

Compression

Documents stored on digital media are fragile; storage errors corrupt the content of a document. How much of a document is corrupted depends largely on the data format of the document. Even small errors in highly compressed or encrypted documents may render part or all of the document unusable.

For documents that might not be fatally corrupted by a single sector error, lossless compression of the document involves a clear trade-off. A smaller document is a smaller target for a randomly occurring error, but a highly compressed document is more fragile. A small error in an audio or video file, or an uncompressed text file, might not be fatal to the document, but a highly compressed text document (or an encrypted document) might be lost.

In these simulations, we have modeled documents as very fragile: one sector error causes the document to be judged as lost. In this model, at least these two considerations should be included in the decision to compress documents.

- Smaller is safer. A smaller document presents a smaller target for random errors. If a document is compressed, say, by 90%, that is, to 10% of its original size, then a

random error is only one-tenth as likely to strike that document. When placed on a storage medium of any given quality level, that smaller, compressed document is likely to persist without error ten times longer than the uncompressed version.

- Smaller is less expensive. A stored collection incurs costs for both storage of the document images and the bandwidth used in auditing and repair. Smaller documents consume less space and less bandwidth and therefore cost less to maintain. On a given budget, a compressed collection can be replicated into more copies and audited more frequently. Both the increased copy count and more frequent auditing contribute directly to reducing or eliminating permanent losses in the collection.

Other discussion points? - Variation: Repairable documents - Small failure damages only one segment

(Model this as a collection of smaller docs?)

- encryption

Encryption

Deriving encryption key loss:

- Assume that your expected collection loss with N_S servers is L% , without encryption
- Assume N_K copies of keys with
- Assume loss of all keys yields 100% loss of collection
- Assume availability and integrity of key is verified on audit

  For example:

Suppose there are only 2 copies of keys. Is the expected document rate due to encryption key loss is equivalent to the expected document loss in the following scenario? - there are two servers - logical block failure is 0 - audits are annual - servers are subject to glitches of the *financial failure* form -- permanent, total loss - What is L*= E(L|KL,N_K) - Can this be derived from failure rate of N_S servers with a rate of KL financial failures - Also affects repairability

- format obsolescence

Format Obsolescence

- can be simulated using reparameterized correlated surver failure model
  - represent
    - each format as server
    - associated failures represent major technology shifts that affect groups of formats (web 4.0)
    - audit period represents time between format reviews -- review of a format for obsolence
  - collections are lost to format failure if all servers fail between audits
- assumptions
  - We have an estimate of failure rate.
  - The audit period is fixed.
  - DO NOT assume that all documents in a format are actually stored on one server.
- excludes interaction between format and document failure rate
  - e.g., if a particular format is extremely fragile, or extremely large
- adversaries

Adversaries

- Assumptions:
  - Weak adversary
- Simulations:
- Implications:
  - Weak adversary -> add [x] copies
  - Open government adversary using legal takedowns -> add [y] copies
  - Clandestine strong adversary, subverting auditing system. Must use distributed auditing. [z] c
- Generalizations: formal crypto models -- see [cite]
- parallel between strategy of less reliability + more auditing, RAID, ECC RAM, FAST, Hadoop
- recommendations for more research, cost models, strong adversaries, erasure codes, lower bandwidth for auditing

Recommendations for research

- Detailed cost models
- Strong adversaries
- Erasure codes...
- Auditing with lower bandwidth -- with cryptographic primitives

References

END OF OUTLINE WITH TEXT IN IT