LOGI: An Empirical Model of Heat-Induced Disk Drive Data Loss and Its Implications for Data Recovery

Hammad Ahmad
hammada@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Colton Holoday
cjholoday@gmail.com
MathWorks
Natick, Massachusetts, USA

Ian Bertram
ianbtr@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

Kevin Angstadt
kangstadt@stlawu.edu
St. Lawrence University
Canton, New York, USA

Zohreh Sharafi
zohreh.sharafi@ieee.org
Polytechnique Montréal
Montreal, Quebec, Canada

Westley Weimer
weimerw@umich.edu
University of Michigan
Ann Arbor, Michigan, USA

ABSTRACT

Disk storage continues to be an important medium for data recording in software engineering, and recovering data from a failed storage disk can be expensive and time-consuming. Unfortunately, while physical damage instances are well documented, existing studies of data loss are limited, often only predicting times between failures. We present an empirical measurement of patterns of heat damage on indicative, low-cost commodity hard drives. Because damaged hard drives require many hours to read, we propose an efficient, accurate sampling algorithm. Using our empirical measurements, we develop LOGI, a formal mathematical model that, on average, predicts sector damage with precision, recall, F-measure, and accuracy values of over 0.95. We also present a case study on the usage of LOGI and discuss its implications for file carver software. We hope that this model is used by other researchers to simulate damage and bootstrap further study of disk failures, helping engineers make informed decisions about data storage for software systems.

CCS CONCEPTS

• **Computing methodologies** → **Model development and analysis**; • **Hardware** → *Fault models and test metrics.*

KEYWORDS

n-gram model, disk drive damage, data recovery software

ACM Reference Format:
Hammad Ahmad, Colton Holoday, Ian Bertram, Kevin Angstadt, Zohreh Sharafi, and Westley Weimer. 2022. LOGI: An Empirical Model of Heat-Induced Disk Drive Data Loss and Its Implications for Data Recovery. In *Proceedings of the 18th International Conference on Predictive Models and Data Analytics in Software Engineering (PROMISE '22), November 17, 2022, Singapore, Singapore.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3558489.3559071

PROMISE '22, November 17, 2022, Singapore, Singapore

ACM ISBN 978-1-4503-9860-2/22/11…$15.00
https://doi.org/10.1145/3558489.3559071

1 INTRODUCTION

With the rise of data centers and big data in software engineering, there is increasing demand from industries to store and manage information. It has been predicted that the amount of data will grow over five times between 2018 and 2025 and that over 80% of enterprise bytes will continue to be stored on hard drives [36].

However, as the bit-density of the hard drives increases, hard drives become more vulnerable [3, 33] as even a simple failure may lead to severe data loss which may have negative impacts on company revenue, productivity, or reputation. Additionally, other sectors of the economy, such as aviation and autonomous transportation, store telemetry that is used in the event of accidents [2, 27]. These data recorders can be exposed to extreme physical forces (*e.g.,* physical shock and heat), making recovery of critical data challenging [10, 29]. For example, recovery of such data is considered so critical and challenging that a working group was convened by the French civil aviation accident investigation agency to study alternate methods for collecting and recovering flight data [6]. Ultimately, understanding hard drive failures and the factors that influence recovery is essential for providing reliable data storage in software engineering. Such knowledge would lay the groundwork for developing data loss prevention- and mitigation-strategies.

Previous work studied the failure trends of hard drives either by monitoring their field behavior or by analyzing self-monitoring, analysis, and reporting technology (SMART) data post factum. SMART is an industry-standard technology embedded in most modern drives that records internal parameters such as read/write error rate and temperature periodically during the operation. These studies reported that heat is one of the leading environmental factors that impact hard drive failure. Overheating is among key failure sources at data centers [34], and has caused outages and brought down services of big tech companies, including Microsoft's Cloud service [24], Amazon AWS [26], and Wikipedia [25]. A 2016 survey of 63 data centers reported that the cost of a service downtime would be more than $540,000 per hour, a significant 38% increase over 2010 [16]. While previous studies model mean time to failure (MTTF), there is a current lack in direct studies—and modeling—of the particular pattern of data loss resulting from a failure. This is relevant because where to store data on a disk, and thus how to mitigate loss from failures, can be a software decision.

There have also been research and engineering efforts to develop effective data recovery tools [1, 12, 32], often referred to as *file carvers*, which often reconstruct stored files by examining software or file system data structures [7] or searching the drive for *magic numbers* [37]. Existing analyses and comparisons of file carvers often assume undamaged drives; most benchmarks involve no damage, and an ideal carver could (theoretically) recover all files. Unfortunately, these benchmarks are not indicative of real-world use cases in which drives have been damaged.

In this work, we address limitations in previous studies of both drive damage as well as file carver performance. First, we design a protocol for—and conduct a series of—controlled experiments to systematically damage hard drives by deliberately overheating them. We apply this protocol to 40 OEM disk drives to collect real-world damage patterns. Second, to measure the extent of the damage, we develop a hierarchical sampling algorithm to read the damaged drives and record the damage status at the granularity of the sector level. Modern drives take significantly longer to respond to read requests of damaged areas, even when low-level access methods are used. Where conventional recording methodology would take over a year for an exhaustive assessment, our approach is tunable, trading off fidelity with recovery time to produce accurate approximations of drive damage in 48 hours or less.

Third, using our empirical measurements, we develop LOGI,[1] a formal model that captures the context and extent of heat damage. We use an n-gram model to leverage the spatial locality of damaged sectors. LOGI allows researchers and practitioners to apply indicative damage to data, simulating varying levels of heat-related damage. Our model provides a framework for bootstrapping further studies without the need to induce damage directly on disks under test. Further, our methodologies may be extended to capture other failure modes, such as electronic or physical shock.

We demonstrate one example use case for LOGI by conducting an evaluation of open-source file carvers. We present a partial replication of a previous study while also injecting realistic faults at the sector level. Also, we evaluate the performance of these file carvers across several commodity file systems. In total, we test 9 file system configurations with 6 data sets and 75 generated damage patterns, clarifying the nuanced relationship between data recovery throughput and accuracy as well as software file system type.

For example, we find higher file carver throughput for the simpler `FAT32` compared to other file systems ($p < 0.05$). Our results also show lower performance for file carvers that are highly configurable, emphasizing that current-generation tools are highly context-dependent. Furthermore, our results show the negative impact of heat-induced damage on file carver performance as the damage intensity increases.

The contributions of this paper are as follows:

- We develop a formal protocol for controlled experiments to systematically damage drives with heat. We apply this protocol to 40 drives to form the basis for our modeling and analyses. We make the raw measured disk image data from controlled heating experiments available for further analysis and replication.
- We propose a sampling-based algorithm to record sector-level damage. Our approach trades off fidelity with runtime to produce close approximations of damaged sectors and is two orders of magnitude faster than direct-read approaches.
- We introduce LOGI, an empirically backed formal model of heat damage for hard drives, achieving precision, recall, F-measure, and accuracy values over 0.95. LOGI can be used by researchers to simulate drive damage to support additional studies of data loss and recovery. We include a study of off the shelf file carvers, including software that uses higher-level semantic information (*e.g.,* for image storage).
- As one application of our model, we present a case study of popular commodity file systems to compare file recoverability using open source file carvers. We find statistically significant results relating certain aspects of file system complexity to recovery performance.

[1] In Norse mythology, Logi is a fire giant.

2 BACKGROUND AND RELATED WORK

In this section, we present background on disk failures, our modeling approach, file systems, and file recovery techniques. We also position our work in the context of related efforts from each of these domains.

2.1 Hard Drive Failure Analysis

Previous studies, mainly performed by drive manufacturers, investigate the electromechanical characteristics of hard drives and discuss how several environmental factors and usage activities can impact the device [8, 30, 47]. Temperature, power-on-hours, and duty cycle were found to be the main parameters that affect the failure rate of the drives [8, 47].

A set of user studies provided a detailed error analysis of failure rates by monitoring a broad set of hard drives for a period of time. Elerath *et al.* analyzed a large pool of enterprise hard drives [11, 39, 40]. They reported that a variety of factors, including the drive model and environmental elements, have a significant impact on drive reliability.

Further, previous studies have shed some light on the failure behavior of hard drives by mining and analyzing SMART signals. Pitakrat *et al.* compared the performance of 21 machine learning algorithms on hard drive failure detection and found that some SMART signals are strongly correlated with higher failure probabilities [31]. Conversely, studying a large deployment of hard drives, Pinheiro *et al.* found that that, despite having some recorded strong correlations, a large fraction of failed drives showed no SMART error failure [30].

Our work complements these previous efforts: we develop a sector-level model of failures to predict data loss due to environmental factors (overheating in particular). Our model does not rely on SMART signals and does not predict times between failures, but instead is derived from controlled experimentation on a population of drives. Our work focuses on results that are actionable at the software level (e.g., informing how to lay out or recover data) rather than at the hardware level.

2.2 N-Gram Models

In machine learning, an *n-gram model* is a probabilistic model that predicts the most likely item that might follow in a sequence. Initially proposed for natural language processing [5], n-grams are now widely used for many applications [9, 19, 46]. As a type of probabilistic language model, an n-gram model predicts the next item the sequence of tokens (*e.g.,* letters or bits) based on the pattern of the previous tokens. Further, n-gram models are *generative.* While many models produce *classifiers*, which are able to label previously-unseen data with a category or value consistent with the labels of a training set, an n-gram model can *produce new, artificial data.* This generated data mimics training input, creating a sequence based on the observed patterns of the extracted $n - 1$ sequences.

N-gram models are particularly well-suited for problems where the possible outputs are not extremely diverse. Due to its extreme sparsity, an n-gram model can only generate the exact seen instances or interpret unseen instances with respect to learned training data [15]. Our experiments in Section 3 demonstrate that heat-induced failure patterns in disks have high spatial locality. Also, while the failure patterns are quite long, they consist of a limited, not-diverse set of macro-patterns that inspire us to develop LOGI, a generative n-gram model to produce synthetic damage patterns. With LOGI, researchers can study the impact of sector-level damage due to overheating without the need for inducing or collecting physical damage, reducing the overall burden and cost for many larger-scale experiments.

2.3 File Systems Error Injection and Robustness Analysis

There is a body of research using fault injection to study both robustness to failure and error-handling mechanisms of commodity file systems [17, 33, 43]. Prabhakaran *et al.* analyzed how file systems handle various drive failure modes and proposed the IRON file system, which implements new policies to handle hardware failures [33]. Shehbaz *et al.* performed a similar analysis on Solid-state drives [17]. Moreover, a handful of studies focus on errors in the read and write operation and analyze the crash resiliency and performance of commodity file systems with respect to these types of errors [14, 28, 44].

Our work extends these finding orthogonally by specifically studying how file system recovery handles errors due to overheating. Since such errors are neither transient nor fully random, our model allows us to study the resilience of common commodity file systems and compare their recoverability across low, medium, and high damage patterns.

2.4 File Carvers

In software engineering and data forensics, *file carvers* are programs used to recover data from drives [45]. These software tools can be used by forensic experts to retrieve information for government inquiries or by engineers or consumers to recover deleted files. Carvers reconstruct data from drives using a variety of heuristics, which ultimately reduces to scanning a drive for bit sequences marking the start and end of relevant data and then "carving" the data between these points.

File carvers generally fall into one of two categories: those that recover entire files [37, 41] and those that attempt to discover snippets of data (*e.g.,* credit card numbers) [13]. In this paper, we focus on the former class of carvers to compare with—and build upon—prior work [21, 22]; we leave the latter category for future work. This prior work investigated the quality and gaps of file carvers, identifying precision, recall, F-measure, and throughput as key metrics. To admit direct comparisons, we also evaluate on the the available dfft and dfrws benchmarks from these experiments (cf. [22]).

This work extends previous findings by replicating experiments on more file system formats and a larger corpus of benchmarks. Our proposed model allows us to generate synthetic heat damage, supporting the generation large populations of damaged drive data. Further, we reveal challenges with recovering data from damage drives and expose new opportunities for carver development and innovation.

3 EMPIRICAL MEASUREMENT OF HEAT DAMAGE

In this section, we present the design, execution, and data collection for our controlled heat damage experiments. We heat each drive using a precision-controlled oven. Then, we design a sampling-based algorithm to record the sectors damaged by the heat. We use the results of these experiments to develop LOGI, our generative model of heat damage, in Section 4.2.

3.1 Heating Procedure

We used 40 commodity hard drives from a popular manufacturer, 18 of which had 160 GB capacity and 22 with 320 GB capacity. Prior to heating each drive, we collected a baseline recording for each drive by writing an alternating bit pattern to the disk and re-reading data to verify that the returned pattern matched our written value. We noted any sectors with extant failures (*i.e.,* possibly from age and/or manufacturing defects, but unrelated to our controlled experiments) and exclude these locations from our subsequent analysis. Overall only 10 of the 40 drives had pre-existing damage, and each such case was limited to 0.46% of sectors.

Disk drives are sophisticated electromechanical devices that can exhibit many different failure modes. We are particularly interested in failures that manifest at the block or sector level (*i.e.,* failures in which the drive is still accessible but some data is unreadable). To reduce the likelihood of other failure modes, we remove the disk controller circuitry prior to heating. Thus, we explicitly study heat damage to the mechanical and magnetic components of a hard drive. We leave study of heat damage to the control circuitry for future work.

We conducted all experiments using a Lindberg & Blue M. Gravity Oven, with a temperature range of 40–260°C. This oven has a uniformity of $\leqslant \pm 3\%$ and a temperature fluctuation of $\leqslant \pm 1$°C, thus allowing for very precise control over the ambient temperature. We waited for the oven to preheat to the set temperature, then shut the drives inside the oven for exactly seven hours to ensure that the entire hard drive reached a uniform temperature. We allowed the hard drives to cool overnight once removed from the oven, so drive internals were not still hot while testing for—and recording—damaged sectors. Prior to recording, we replaced the controllers, taking care that each drive received its original controller.[2]

During preliminary testing of our experimental protocol, we discovered high variance in the effect of heating on drives. That is, for

[2]Many manufactures include device-specific ROM on the controller, and thus there is a one-to-one mapping between controller and disk.

Algorithm 1 **Hierarchical Sampling.** This divide and conquer algorithm distributes data probes across a drive to avoid collocated bad sectors and enable early termination with indicative results. The procedure is breadth-first: each subsequent pass over the drive collects finer-grained data about the location of bad sectors. The key configuration parameter is the partitioning factor in **partition_chunk**().

```
procedure HIERARCHICALLY_SAMPLE(start, end)
    queue chunks ← create_chunks(start, end)
    while chunks is not empty do
        chunk ← dequeue chunks
        for sector ∈ chunk do
            if syscall_read(sector) is good then
                report sector is good
            else
                report sector is bad
                for subchunk ∈ partition_chunk(chunk) do
                    chunks ← enqueue subchunk
                end for
            end if
        end for
    end while
end procedure
```

the same duration exposure to the same temperature, some drives would experience total failure while others received only minor damage. Since we are interested in modeling software-actionable damage (*e.g.,* drives with no damage do not need special recovery or file carvers, and drives that are entirely inoperable do not admit commodity tools), we developed a procedure with repeated exposure to maximize the number of non-trivial data points produced by our experiment. In the context of this experiment, we consider a result to be *trivial* if (1) there is less than 10% new damage on the drive, or (2) the drive no longer responds to the system. We initially exposed each drive to 168°C, following the described procedure. Then, we repeated the procedure while increasing the temperature in 3°C increments. From our population of 40 hard drives, we were able to successfully capture 15 different, non-trivial damage patterns, each corresponding to a single iteration of our experimental procedure. Temperature exposure for our data points ranged from 165°C to 186°C.

3.2 Recording Algorithm

Attempting to read a failed sector on a drive takes orders of magnitude longer than a functioning sector (*e.g.,* [35]). Our testing indicates that this time is dictated by both the drive firmware and OS configuration. In our experiments, lower-level utilities (*e.g.,* `dd`, `ddrestore`, etc.) and approaches (*e.g.,* `ioctl`, low-level interfaces, special timeouts) still suffered from long read times on bad sectors or failed to read recovered data. While there is variance in bad sector read times, for an indicative drive in our experiments, reading a bad sector induced eight seconds of latency before a subsequent probe could be made. For a 160 GB drive, which has approximately 312,581,808 sectors, exhaustively assessing non-trivial damage could require months. Note that while special hardware could potentially mitigate such latencies, we focus on a commodity use case applicable to software engineering scenarios.

To gather data in a reasonable time frame, we devise a tunable divide and conquer algorithm to sample a disk drive and approximate heat damage within a user-provided time budget. We refer to our approach as *hierarchical sampling* and outline it in Algorithm 1. At a high level, Algorithm 1 breaks the logical block address space of a disk into chunks. The algorithm scans each chunk sequentially until a bad sector is found. Upon finding a bad sector, the algorithm recursively breaks up the address space into smaller chunks. Processing of these chunks is performed in a breadth-first manner: coarser-grained chunks are processed before subchunks.

Inherent in the formulation of our hierarchical sampling is the assumption that bad sectors exhibit spatial locality. We base this assumption on evidence collected during initial testing, during which we frequently found collocated bad sectors from heat damage. Our chunk-based sampling approach allows the algorithm to jump over bad contiguous segments of the disk. As the algorithm proceeds, it iteratively refines the locations of bad sectors, increasing the fidelity of the recorded damage pattern with each pass.

Given infinite time, our proposed algorithm is isomorphic to a naive reading of all sectors. However, our sector probing strategy enables a tradeoff between time and fidelity. By setting a shorter timeout, the algorithm terminates before probing all sectors, but still produces a partial mapping of bad sectors from which a full approximate mapping can be interpolated later. In our experiment, we used a 48 hour budget, considering the use case of a company conducting a software-based drive recovery over a weekend.

The quality of the damage mask produced by our hierarchical sampling algorithm is affected both by the chunk partitioning scheme and also by how unscanned portions of the disk are interpolated. To distribute probes across a drive, our hierarchical sampling algorithm subdivides a disk into contiguous chunks. When a bad sector is discovered, the algorithm further subdivides the remainder of the chunk. We evaluated multiple partitioning configurations using synthetic damage patterns and compared them against a baseline of random probing. Our highest-performing scheme first divided the logical block address (LBA) space into 1024 chunks at the top-level, then broke each into 512 subdivisions at the second level, and so on. Since the recording algorithm samples sectors within a time budget, some sectors may not be probed. Based on our observed spatial locality of damaged sectors and to simplify model training and file system investigation, we use a nearest-neighbor interpolation scheme to assign sectors a damage value. We note that our initial hierarchical sampling is sound but incomplete (*i.e.,* while correctly reporting those that it probes, it does not probe all sectors); any approximation in our overall recording approach is introduced by the interpolation step.

4 LOGI: A GENERATIVE MODEL OF HEAT DAMAGE

In this section, we first provide the quantitative and qualitative description of the recorded heat damage collected using the procedure described in Section 3.1. Using this data, we develop LOGI, a generative model of heat-based disk damage. Because of the spatial locality of the heat damage, we argue that an n-gram model is appropriate to capture the context (Section 4.1). Next, we present our n-gram model, its characteristics, and evaluation (Section 4.2). Finally, we assess the threat of over-fitting via 10-fold cross-validation and assess the completeness and suitability of our training data set using a perturbation analysis (Section 4.3).

4.1 Damage Characteristics

In this subsection, we analyze the distribution of damaged sectors and investigate the hypothesis that the distribution is positionally symmetric across the drive.

To better understand the distribution of the damaged sectors and find a suitable model, we consider summary statistics. For each recorded drive, we split the data in half and compare the percentage of the damaged sectors in the first and second halves of the drive. The difference between the number of damaged sectors in the first half compared to the second half of the drives is not significant (Wilcox test: $W = 40, p = .25$). On average, 47% of damaged sectors are located in the first half, while the rest are located in the second. We also categorized recorded samples into four quarterlies based on the intensity of the damage, and found that the damage distribution is fairly symmetrical across the drive.

In addition to a quantitative, statistical test of the distribution of damaged sectors across the drive, we also present a qualitative explanation to show that the distribution is independent of sector positions. Figure 1 displays concrete damage distributions of an example drive from each quartile. Each image shows adjacent sectors from bottom to top (then left to right) in increasing memory addresses. That is, the y-axis encodes the minor, low-order bits of the address, while the x-axis encodes the major, high-order bits. Low memory addresses are on the left, and the addresses span 16k per column as we move to the right. Each pixel in the image represents a sector in the drive. Damaged sectors are shown in purple while the healthy ones are yellow. Every full vertical line in the image represents adjacent memory addresses starting from n to $n + 16,000$. The myriad set of rectangular purple segments demonstrates the notable amount of spatial locality. This finding is also aligned with the work of Bairavasundaram *et al.* on silent data corruption in disk drives [4]: based on 400,000 instances of checksum mismatches over a 41-month period, they reported high spatial locality due to consecutive disk sectors developing corruption.

This observation of spatial locality encourages us to look for a model that leverages the history (the context) of the damaged sectors by looking at the status of their neighbors. In the following subsection, we propose such a model, constructed from n-grams to capture the context between adjacent sectors.

4.2 N-gram Model of Heat Damage

In this subsection, we propose LOGI, a generative, n-gram model for producing synthetic damage patterns with spatial locality. LOGI embeds the recorded raw data into linear, numeric vectors to capture the locality of the heat damage. Further, LOGI is computationally inexpensive and predicts the state of a sector (*i.e.,* damaged or healthy) based on its adjacent sectors. Our n-gram model provides the probability of a sector state based on the state of $n - 1$ previous sectors. We formally present our n-gram model, detail parameter tuning, and explain its characteristics.

4.2.1 Formal Representation. To present data in terms of n-grams, each recorded data item (*i.e.,* each recording of a damaged drive), x, is represented as a binary string of damaged (0) and healthy (1) sectors. By considering all windows of n sectors over each data item x, we extract all substrings of length n. These substrings (*n-grams*) are used to build a map to a high-dimensional vector space, where each dimension is associated with the occurrences of one n-gram. Formally, this map Ø is constructed by the set S of all possible n-grams as:

$$Ø : x \longrightarrow (Ø_s\,(x))_{s \in S} \text{ with } Ø_s\,(x) = fr\,(s, x)$$

where the frequency function, $fr\,(s, x)$, returns the probability for the occurrences of the n-gram s in data x.

Table 1: Pilot experiment to choose model order (n) based on accuracy and F-measure. Random guessing is regarded as the baseline. The value selected for our experiments is bolded.

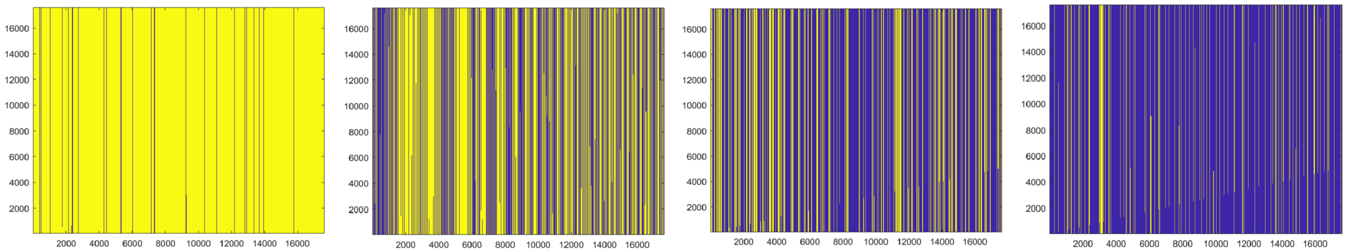
	Accuracy	Precision	Recall	F-measure
Random	0.498	0.517	0.496	0.445
n = 2	0.895	0.895	0.895	0.895
n = 3	0.998	0.990	0.992	0.991
n = 4	0.988	0.989	0.989	0.997
n = 5	**0.999**	**0.999**	**0.999**	**0.999**
n = 7	0.997	0.988	0.989	0.989
n = 10	0.997	0.987	0.989	0.988

Table 2: The number of times that a given 5-gram appear in the training set. [1,1,1,1,1] and [0,0,0,0,0] are the most frequent 5-gram patterns.

Bit pattern	Number of occurrences	
	Zero (0)	One (1)
[1,1,1,1]	53,278	3,546,993,106
[1,1,1,0]	53,279	0
[1,1,0,0]	53,279	0
[1,0,1,1]	0	1
[1,0,0,0]	53,279	2
[0,1,1,1]	1	53,271
[0,1,0,1]	0	1
[0,1,0,0]	1	0
[0,0,1,1]	0	53,271
[0,0,1,0]	1	1
[0,0,0,1]	3	53,271
[0,0,0,0]	3,639,695,709	53,272

4.2.2 Parameter Tuning. We tune performance by adjusting the *order* of the model: the value of "n". Because higher orders offer diminishing accuracy benefits and also increase computation and concerns of overfitting, we consider values 2, 3, 4, 5, 7, and 10. To choose the order for our modeling, we follow the 90/10 rule and divide the recorded data into a training set and a testing set, obtaining the results shown in Table 1. There are small performance gains when moving from order 2 to 5, but growth reverses from order 5 to order 10. We select the most performant configuration, n= 5. However, our results show that our model is not particularly sensitive to order and that any value greater that 2 could be used. Overall, we achieve a very high accuracy and F-measure.

4.2.3 Characteristics. We generate a 5-gram model to characterize heat-induced damage patterns, which is presented in Table 2. Out of 16 possible binary patterns, our empirical training set contains 12. The most dominant patterns are "[1, 1, 1, 1] followed by 1" and



(a) Quartile 1 (1–25%) (b) Quartile 2 (25–50%) (c) Quartile 3 (50–75%) (d) Quartile 4 (75–100%)

Figure 1: Examples of damage patterns, one from each damage quartile. The x- and y- axes encode a sector's major and minor address, respectively. Yellow pixels are functioning sectors; purple pixels are unreadable. Large bands of purple indicate significant spatial locality in our recorded drive samples.

"[0, 0, 0, 0] followed by 0" which demonstrate the sequential locality of the damaged sectors. We also observe that the recorded patterns are not diverse. For example, patterns [1, 0, 1, 1], [0, 1, 0, 1], or [0, 1, 0, 0] occurred only once.

We use this information to generate synthetic patterns of damage that closely conform to our empirical observations. We embed this model in LOGI to generate a sequence of values (*i.e.,* a synthetic pattern of disk damage) where each value corresponds to a single sector (*i.e.,* encoding whether it is damaged or not). Starting with an initial sequence of 5, LOGI generates the next sector's status, using the model to estimate the likelihood of the next sector being damaged. More precisely, given a history of 4 previous sectors, LOGI approximates the probability of the next one being damaged by choosing a random number weighted by the modeled proportions. Looking at the four previous sectors in the example, we observe that the probability of a damaged sector occurring after the pattern [1, 1, 0, 0] is 100%. We select 1 (damaged) and continue with the next sequence of size four and moving forward. For round three (m = 3), the probability of a health sector occurring after the pattern [0, 0, 0, 0] is significantly low, but not zero. In such cases, the generative model flips a biased coin. We continue generating sectors in this fashion until we create a list of the size of the target device.

While our model generally predicts damaged sectors from a context of previous damaged sectors (and undamaged sectors from undamaged contexts), unlikely predicted deviations are possible. Since even our smaller 160 GB drives involve over 300 million sectors (Section 3.2), the interplay between long arrays of predictions and high probabilities of local similarity generates synthetic data closely conforming to the patterns seen in the measured drives.

4.3 Suitability Criteria

To check for overfitting, we perform a 10-fold cross-validation analysis. Our model achieves an average performance of 0.957, 0.957, 0.958, and 0.974 in terms of precision, recall, F-measure, and accuracy, respectively. Our 10-fold cross-validation for all ten groups in terms of accuracy reveals high agreement between folds (groups), giving confidence that the performance of the model is not due to overfitting.

We also look at *perturbation* to characterize the expected completeness of the training data. Per Wressnegger *et al.*, perturbation is "the expected ratio of n-grams in a benign data set that are not part of the training data" [46]. A value of 0 means that we have observed all possible n-grams of the data set in the training phase. A high value indicates that despite training, a large number of unseen n-grams occur during testing. That is, a high perturbation value means the model must often generate the next value from a previously unseen sequence. Low perturbation limits the number of false positives produced by unknown n-grams in testing data [46]. Our data set quickly approaches zero perturbation indicating that our training set is sufficiently large, and we do not require more data or training.

4.4 LOGI Usage

LOGI allows for the generation of synthetic damage patterns indicative of heat damage to a disk. The damage map produced by LOGI can be composed with actual sector data to produce data that has been artificially damaged by heat. Because our model is generative, we support variable drive sizes; test images need not be the same size as our sample drives. Thus, our model enables a wide variety of studies without the need for physically damaging a drive.

In addition to a model trained on all of our recorded drive samples, we also produced configurations of LOGI for low (less than 25% damaged sectors), mid-level (25%–75% damaged sectors), and high (more than 75% damaged sectors) damage. Each of these models can be used to artificially increase damage the same data (*e.g.,* to study file recovery performance as a function of the degree of damage).

5 FILE SYSTEM AND RECOVERY EVALUATION

In this section, we demonstrate a case study on the usage of LOGI, our formal model, to generate damage patterns. First, we present a partial replication of previous studies on the performance of file carvers. Second, we evaluate the impact of heat damage on file recovery for various commodity file systems.

5.1 Experimental Setup

File Carver Selection. We choose three file carvers for this study: Scalpel [37], Foremost [41], and Photorec [7]. All have been studied in prior work (*e.g.,* [22]) and have wide file type support. For availability and replication, we restrict our attention to open-source file carver software.

Data Set Benchmark Selection. We use a total of six data sets to evaluate the performance of file-systems and file carvers: cfd, dfft, dfrws, drupal, open_images, and wiki. We include the Chicago Face Database (cfd) [23] to provide a larger data set with real-world applications. Two data sets, dftt and dfrws, are taken from prior work and are indicative of "challenge" benchmarks in the data forensics community [22]. Our drupal benchmark is a snapshot of a professional organization's webserver, which uses the popular content management system[3] to serve files. Further, we include the open_images benchmark, which is a random 5GB subset of a large, indicative machine learning training data set. Finally, we constructed a benchmark from Wikimedia using a snapshot of uploaded files from March 2013.

These data sets contain a variety of file types commonly stored to disk and are indicative of the types of files that would need to be recovered in the event of disk failure. All files are either documents, images, multimedia, or compressed archives. Our data sets do contain some file types that are unsupported (*e.g.,* PPT, WMV, and TXT files) by our file carvers. In addition, the drupal and wiki data sets include a set of web resource and asset files (*e.g.,* CSS, TTF, and ASP) that none of our file carvers directly support. As a result, we believe these benchmarks are more indicative of general software engineering use cases than are previous forensics-focused benchmarks.

File System Selection. We consider four file systems: `BTRFS`, `Ext4`, `FAT32`, and `NTFS`. `BTRFS` is a modern, copy-on-write file system initially designed by Oracle for use in Linux [38]. The `Ext4` file system is particularly common for Linux systems, and we consider multiple configurations to understand the impact of journaling and metadata overhead on file recovery. While `FAT32` is quite old, FAT-based systems remain relevant due to their use in UEFI-based booting [42, Section 13.3] and embedded systems. Finally, `NTFS` is widely used in Microsoft ecosystems.

Damage Pattern Generation. We use LOGI to generate damage patterns for our data sets. For each data set benchmark and file system format, we generate 250 new disk images that are artificially damaged according to a pattern produced by our formal mathematical model. To generate these images, we seed LOGI with a random damage pattern of 10 sectors.

To study the effects of file carver selection as a function of the amount of damage present, we also use three additional configurations of LOGI as described in Section 4.4 to generate three specific damage patterns: low damage, mid-level damage, and high damage.

Performance Measurement. We use known metrics from previous work [21, 22] to measure file carver performance. *Recall*, *Precision*, and *F-measure* evaluate correct recovery of files. Prior work distinguishes between carving and supported recall; we measure carving recall, which is the fraction of files recovered in a benchmark. Finally, we use *processing speed* (or *throughput*) to evaluate the speed of data recovery. Throughput, measuring total MB transferred per second, is an indicator of how quickly a drive is processed.

[3]https://drupal.org

Table 3: File carving performance scores for all data sets. *Photorec* has the best overall performance.

	Data Set	Recall	Precision	F-measure	Throughput (MB/s)
Foremost	All	0.31	0.79	0.39	23.7
	dfft	0.29	0.72	0.32	2.9
	dfrws	0.21	0.54	0.25	10.5
	cfd	0.51	0.94	0.61	26.1
	drupal	0.09	0.85	0.16	29.9
	wiki	0.24	0.70	0.33	37.2
	open_img	0.52	0.99	0.63	35.3
Photorec	All	0.42	0.73	0.47	31.5
	dfft	0.41	0.77	0.43	33.5
	dfrws	0.40	0.79	0.73	50.4
	cfd	0.50	0.51	0.46	23.7
	drupal	0.26	0.63	0.34	42.9
	wiki	0.41	0.65	0.46	20.3
	open_img	0.52	1.00	0.63	18.8
Scalpel	All	0.18	0.34	0.19	24.1
	dfft	0.13	0.51	0.11	2.9
	dfrws	0.20	0.20	0.15	12.0
	cfd	0.04	0.03	0.03	46.2
	drupal	0.03	0.14	0.05	8.9
	wiki	0.15	0.16	0.14	5.3
	open_img	0.52	0.98	0.63	69.4

5.2 File Carver Impact on File Recovery

First, we partially replicate the results of a previous study of file carver performance [22] using our larger suite of data sets. Then, we study the performance impact of introducing sector failures produced by LOGI.

Operators typically desire file carvers that (a) recover the most possible files, (b) do so quickly, and (c) include as few false positives as possible. We evaluate file carvers along these dimensions and present the results in Table 3.

File Carver File Recovery. *Photorec* achieves the highest overall file recovery performance compared to *Foremost* and *Scalpel*. *Photorec* makes use of heuristics to automatically detect relevant data files while *Foremost* and *Scalpel* use pre-configured sets of supported file types. Two of our data sets (wiki and drupal) include types of files that are not supported by *Foremost* or *Scalpel*. In agreement with the previous work [22], the lowest performance is observed from tools that require expert configuration (*Foremost* and *Scalpel*).

File Carver Speed. *Photorec* yields the overall highest throughput on our benchmarks. Further, our data set allows us to study the performance impact of increasing the number of file types to be recovered. For signature-based carvers (see Section 2.4), the number of file types can have a large impact. Conceptually, the carver must consider all possible signatures for each file type *per chunk of data*. Increasing the number of file types increases number of comparisons per chunk of data, resulting in lower throughput. For

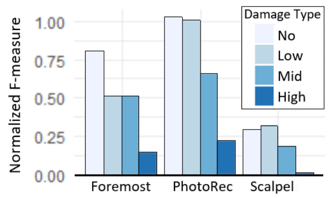


Figure 2: Comparison of file carver F-measures for various damage models.

example, *Scalpel* shows the highest throughput of all carvers (70 MB/s) when only one file signature is present, but shows the lowest throughput (3 MB/s) when more than five signatures are in play. These results are in agreement with previous findings that file carver heuristics can have significant performance implications.

File Carvers and Damage. Finally, we consider the impact of damage rates on file carver performance. To the best of our knowledge, this has not been studied previously. Figure 2 plots the F-measure of each file carver as the amount of damage varies from none to high. While *Photorec* and *Scalpel* are resistent to small amounts of damage, their performance plummets at moderate damage levels. By contrast, *Foremost* shows a sharp degradation for even small amounts of damage, but has similar performance for low and mid damage models.

All three carvers showed low absolute performance for high damage models. Recall that our high damage model was trained from real-world drives with sector damage rates higher than 75%. Informally, if only 25% of the information is available, we expect tool performance to drop by at least 75%. While this simplification abstracts a number of important details (*e.g.,* filesystem metadata), *Photorec* and *Foremost* generally follow this pattern. For example, *Photorec*'s normalized F-measure drops by 80% from no to high damage while *Foremost*'s drops by 85%. By contrast, *Scalpel*'s performance degrades more dramatically.

While we are wary of generalizing from a handful of tools, these results do suggest that there are relevant tradeoffs in file carver heuristics. In our experiments, *Photorec* was particularly resilient to low levels of damage while the *Foremost* tool treated low- to mid-levels of damage almost identically.

5.3 File System Impact on File Recovery

We investigate the effect of file system choice on file recovery.

File Systems and Recovery Performance. When all of our file benchmarks (*e.g.,* images, websites, etc.) are considered and averaged, our results show no significant differences between file systems with respect to file carver performance. Table 4 shows the details.

However, `FAT32` outperforms other file systems with regard to throughput. For two (dfrws and open_images) out of six data sets, we find that `FAT32` admits faster recovery in a statistically significant manner, as shown in Table 5.

Damage Severity and Recovery Performance. We also investigate how the severity of the heat damage impacts the file recovery performance and rates. We considered the high and low damage models (*i.e.,* trained from drives with greater than 75% damage and

Table 4: Summary of the file carvers performance score, averaged for all data sets to compare various file systems.

	BTRFS	Ext4	FAT32	NTFS
Recall	0.3	0.3	0.1	0.3
Precision	0.6	0.6	0.6	0.6
F-measure	0.3	0.4	0.4	0.4
Throughput (MB/s)	26.6	25.4	30.0	26.5

Table 5: Throughput (MB/s) of file systems per data sets. The Non-parametric Wilcox test (α = .05) was used to compare `FAT32` with other file system pair-wise. Significant results (< 0.05 for `FAT32` vs. `BTRFS`) are bolded.

	BTRFS	Ext4	FAT32	NTFS	p-value
dfft	17.58	8.29	17.42	14.74	> 0.05
dfrws	23.31	17.72	25.05	26.67	**< 0.05**
cfd	29.90	33.68	32.43	32.80	> 0.05
drupal	29.06	24.97	32.86	20.49	> 0.05
wiki	20.61	22.09	25.08	19.17	> 0.05
open_img	35.82	42.17	46.81	40.58	**< 0.05**

less than 25% damage, respectively). First, we found no significant differences between file carver throughput as a function of file system with respect to damage amount. That is, no file system supported particularly faster or slower recovery for higher levels of damage.

In addition, we found no significant differences between file carver recovery (*i.e.,* precision, recall or F-measure) as a function of file system with respect to damage amount. That is, for example, no file system emerged as a clear champion for recovering data particularly from high levels of damage.

File System Design Decisions. We also consider the hypothesis that modern file system software techniques, such as journaling and metadata frequency, might interact with file carvers (*e.g.,* because the carvers may look for signatures that may not stored contiguously given such file system design decisions).

We investigate the impact of journaling (which can be present or not) and amount of file system metadata overhead (either high metadata, 2048 bytes per 512 byte inode, or low metadata, 8192 bytes per 128 byte inode) on recovery for `Ext4`. The results are shown in Table 6. While neither journaling nor metadata alone has a statistically significant effect, a generalized linear model reveals an interaction between journaling and metadata. This significant interaction is present for both the F-measure of files recovered ($p < 0.001$, t-value 10.989) and also for throughput rate ($p < 0.001$, t-value 13.273). While the magnitude of the effect is small, the interaction is strongly significant.

This preliminary results calls for further experiments and evaluation on the impacts of such factors. For example, if the particular combination of high metadata overhead and journaling means that 1–2% fewer files can be recovered, on average, but that disadvantage can be mitigated by either removing journaling or reducing metadata rates, administrators may use such information to guide deployment decisions or storage software design decisions.

Table 6: Comparison of the performance scores for `Ext4` Journaling and Metadata, averaged over all data sets.

Journaling	Metadata	F-measure	Throughput (MB/s)
Off	High	0.35	24.58
Off	Low	0.36	25.51
On	High	0.34	24.95
On	Low	0.36	25.35

6 DISCUSSION

In this work, we developed LOGI, a generative model of sector damage due to overheating disk drives. Using this model, we tested the performance of file carvers and file systems in the face of failing sectors. In particular, we found that the performance of file carvers rapidly degrades as damage is introduced (Figure 2). This is relevant because almost all previous work evaluates file carvers in a zero-damage setting [21, 22]. Our findings motivate further development of robust file carvers that are capable of partial recovery of files. Such techniques might include "fuzzy matching" (*e.g.,* within some Hamming or Levenshtein edit distance) rather than the exact matches currently employed. LOGI allows for direct evaluation of prototype carving algorithms by allowing for the generation of synthetic benchmarks.

We see this as an opportunity for research on software that controls data layout and recovery. Our study demonstrates that sector failures exhibit high spatial locality. Thus, a file system that is robust to heat-induced failures should spread data throughout a disk. Unfortunately, this is in direct contradiction to performance studies, which recommend reducing fragmentation to improve performance (*e.g.,* [18, 20]). More relevant to recovery, however, this suggests that more partial files may actually be recoverable, in theory, from high-metadata journaling file systems, even if current-generation file carvers do not employ expensive, system- and journal-aware heuristics to find them. On the one hand, this suggests that partial file recovery benchmarks and metrics should play a larger role going forward; many investigations favor simpler metrics that count only fully-recovered files (*e.g.,* [21, 22, 37]). However, repeated studies of indicative high-damage patterns have not been feasible in the past, to the best of our knowledge. LOGI supports such studies by allowing arbitrary manipulation of a file system under test prior to applying the damage model. RAID has also been used to improve the performance and robustness of stored data at the cost of additional storage space. Future studies using LOGI could also investigate the ability to recover data from multi-disk arrays damaged by heat.

In summary, we present initial findings about the relationship between sector-level data loss due to heat damage and the performance of file systems. We find that a robust file system should balance performance with risk of data loss due to spatial locality. Our model, LOGI, enables subsequent studies to test candidate file systems and recovery algorithms. In particular, we believe that a number of our design decisions, such as including more software-relevant file types and content types in our benchmarks and focusing on software-only actions (*e.g.,* using low-level reads, rather than physically taking apart the disks) and their associated consequences (*i.e.,* orders-of-magnitude time differences between successful and failed reads) mean that our results are more directly applicable to software-level decisions (*e.g.,* how to lay out or recover data given the spatial locality of heat damage).

7 LIMITATIONS

The study carried out in this paper is based on a limited number of hard drives of a single brand. Variation in manufacturing techniques may limit the generality of our results. For this initial study of sector-level heat damage, we chose to use a single manufacturer to increase the likelihood of collecting sufficient data points to produce an accurate model. As noted in Section 4.3, LOGI performs quite well in our testing scenarios. In particular, our model has low perturbation, suggesting that we collected sufficient training data.

Additionally, our data sets are larger than prior work, but remain orders of magnitude smaller than the capacity of modern disk drives. Thus, there is a risk that synthetic damage produced by LOGI is not indicative of real-world damage on larger drives. We tested the performance of *Scalpel* on a 160 GB disk image using real-world recorded damage (~81% of sectors unreadable). We could find no statistically significant difference in precision, recall, and F-measure between these large-scale tests and our experimental data sets. This adds confidence that our results generalize to larger data sets and are indicative of performance on real-world damage.

Finally, we record sector-level damage at a high level of abstraction. Disk controllers map virtual sectors to physical locations and include functionality to remap sectors as they begin to fail. Our recording technique is unable to capture these low-level details. While disassembly and reading of data directly from the platters in a clean room would avoid this problem, such a protocol is unnecessarily prohibitive. File systems are implemented at the OS level and therefore interface with the disk using the same abstracted interface at which we conducted our study. As such, our performance and recovery findings are indicative of real-world deployments.

8 CONCLUSIONS

There is increasing demand to store information in software engineering, even in the face of failures. This paper investigates heat-based disk drive damage patterns. We conduct controlled experiments and propose a novel, sampling-based algorithm to collect damage patterns. We develop LOGI, a model for studying sector-level heat damage. Our model generates synthetic heat damage, which can be leveraged for large-scale studies.

We use LOGI to investigate three file carvers and four file systems, including a partial replication of previous file carver studies on our indicative damage models. These carvers exhibit differing behavior at increasing levels of damage (*e.g.,* some carvers treated no- and low-damage models similarly, while others treated low- and mid-damage models similarly), confirming that carver heuristics can be tailored to specific damage use cases. Our study reveals two trends relating file system complexity and recovery. First, for some data sets, the simple `FAT32` file system admits the highest throughput. Second, we found a significant interaction between journaling and metadata rates for `Ext4`. We hypothesize that both observations relate to file fragmentation and fragment ordering and present an opportunity for next-generation data recovery heuristics.

Although our work investigates the impacts of heat-induced damage on stored data with regard to files, our proposed n-gram model and experimental design may support future investigations of damage (*e.g.,* electronic shock), file carving heuristics, or file system data structures and configurations. To the best of our knowledge, this work presents the first realistic, generative formal model of sector-level, heat-induced damage on commodity drives and uses it to assess file recovery as a function of the file systems and file carvers used. Our data sets, recorded data, and algorithm are publicly available for analysis and replication as a Zenodo artifact.[4]

ACKNOWLEDGMENTS

We gratefully acknowledge the partial support of the NSF (2211749, 1763674) and the AFRL (FA8750-19-1-0501).

REFERENCES

[1] Nadeem Alherbawi, Zarina Shukur, and Rossilawati Sulaiman. 2016. A survey on data carving in digital forensic. *Asian Journal of Information Technology* 15, 24 (2016), 5137–5144.

[2] Michael P Andersen and David E Culler. 2016. Btrdb: Optimizing storage system design for timeseries processing. In *Conference on File and Storage Technologies*. 39–52.

[3] Dave Anderson, Jim Dykes, and Erik Riedel. 2003. More Than an Interface-SCSI vs. ATA.. In *Conference on File and Storage Technologies*, Vol. 2. 3.

[4] Lakshmi N. Bairavasundaram, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dusseau, Garth R. Goodson, and Bianca Schroeder. 2008. An Analysis of Data Corruption in the Storage Stack. *ACM Trans. Storage* 4, 3, Article 8 (Nov. 2008), 28 pages. https://doi.org/10.1145/1416944.1416947

[5] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* 3, null (March 2003), 1137–1155.

[6] Bureau d'Enquêtes et d'Analyses pour la sécurité de l'aviation civile (BEA). 2009. *Flight Data Recovery Working Group Report*. Technical Report.

[7] CGSecurity. 2019. Photorec. https://www.cgsecurity.org/wiki/PhotoRec

[8] Gerry Cole. 2000. Estimating drive reliability in desktop computers and consumer electronics systems. *Seagate Technology Paper TP* 338 (2000).

[9] Marc Damashek. 1995. Gauging similarity with n-grams: Language-independent categorization of text. *Science* 267, 5199 (1995), 843–848.

[10] Alex Davies. 2019. How Investigators Pull Data off a Boeing 737's Black Boxes. https://www.wired.com/story/boeing-737-max-8-ethiopia-crash-black-box-data-lion-air/. [Online; accessed 18-05-2020].

[11] J. G. Elerath and S. Shah. 2004. Server class disk drives: how reliable are they?. In *Annual Symposium Reliability and Maintainability, 2004 - RAMS*. 151–156.

[12] Simson L Garfinkel. 2007. Carving contiguous and fragmented files with fast object validation. *digital investigation* 4 (2007), 2–12.

[13] Simsong L. Garfinkel. 2014. Bulk Extractor 1.5.6. https://github.com/simsong/bulk_extractor

[14] Haryadi S. Gunawi, Cindy Rubio-González, Andrea C. Arpaci-Dusseau, Remzi H. Arpaci-Dussea, and Ben Liblit. 2008. EIO: Error Handling is Occasionally Correct. In *Conference on File and Storage Technologies* (San Jose, California) *(FAST'08)*. USENIX Association, USA, Article 14, 16 pages.

[15] Alon Halevy, Peter Norvig, and Fernando Pereira. 2009. The unreasonable effectiveness of data. *IEEE Intelligent Systems* 24, 2 (2009), 8–12.

[16] Ponemon Institute. 2016. *Cost of Data Center Outages*. https://www.vertiv.com/globalassets/documents/reports/2016-cost-of-data-center-outages-11-11_51190_1.pdf

[17] Shehbaz Jaffer, Stathis Maneas, Andy Hwang, and Bianca Schroeder. 2020. The Reliability of Modern File Systems in the Face of SSD Errors. *ACM Trans. Storage* 16, 1, Article 2 (March 2020), 28 pages. https://doi.org/10.1145/3375553

[18] Cheng Ji, Li-Pin Chang, Sangwook Shane Hahn, Sungjin Lee, Riwei Pan, Liang Shi, Jihong Kim, and Chun Jason Xue. 2019. File Fragmentation in Mobile Devices: Measurement, Evaluation, and Treatment. *IEEE Trans. Mob. Comput.* 18, 9 (2019), 2062–2076. https://doi.org/10.1109/TMC.2018.2869737

[19] Vlado Kešelj, Fuchun Peng, Nick Cercone, and Calvin Thomas. 2003. N-gram-based author profiles for authorship attribution. In *Proceedings of the conference pacific association for computational linguistics, PACLING*, Vol. 3. sn, 255–264.

[20] Haecheon Kim, Dongwoo Kang, Kwanghee Lee, and Jongmoo Choi. 2018. Fragmentation-Less FAT-compatible File System for Internet of Things. In *Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication, IMCOM 2018, Langkawi, Malaysia, January 05-07, 2018*. 61:1–61:5. https://doi.org/10.1145/3164541.3164554

[21] SJJ Kloet. 2007. Measuring and improving the quality of file carving methods. *Almere, Niederlande: Eindhoven University of Technology (Thesis)* (2007), 4–79.

[22] Thomas Laurenson. 2013. Performance analysis of file carving tools. In *IFIP International Information Security Conference*. Springer, 419–433.

[23] Debbie S Ma, Joshua Correll, and Bernd Wittenbrink. 2015. The Chicago face database: A free stimulus set of faces and norming data. *Behavior research methods* 47, 4 (2015), 1122–1135.

[24] Robbert McMillan. 2013. Did Microsoft Fry Its Cloud With an Air-Conditioner Update? https://www.wired.com/2013/03/outlook-outage/. [Online; accessed 23-08-2019].

[25] Rich Miller. 2010. Wikipedia's Data Center Overheats. https://www.datacenterknowledge.com/archives/2010/03/25/downtime-for-wikipedia-as-data-center-overheats/. [Online; accessed 23-08-2019].

[26] Rich Miller. 2012. Generator Fan Failure Triggered AWS Outage? https://www.datacenterknowledge.com/archives/2012/06/21/aws-outage/. [Online; accessed 23-08-2019].

[27] Gianfranco E Modoni, Marco Sacco, and Walter Terkaj. 2016. A telemetry-driven approach to simulate data-intensive manufacturing processes. *Procedia CIRP* 57 (2016), 281–285.

[28] Keshava Munegowda, GT Raju, and Veera Manikandan Raju. 2014. Evaluation of file systems for solid state drives. In *Proceedings of the Second International Conference on Emerging Research in Computing, Information, Communication and Applications*. 342–348.

[29] Ontrack. 2015. Recovering Data From Black Boxes. https://www.ontrack.com/uk/blog/concepts-explained/recovering-data-black-boxes/. [Online; accessed 18-05-2020].

[30] Eduardo Pinheiro, Wolf-Dietrich Weber, and Luiz André Barroso. 2007. Failure trends in a large disk drive population. In *Conference on File and Storage Technologies*. Google.

[31] Teerat Pitakrat, André van Hoorn, and Lars Grunske. 2013. A comparison of machine learning algorithms for proactive hard disk drive failure detection. In *Proceedings of the 4th international ACM Sigsoft symposium on Architecting critical systems*. ACM, 1–10.

[32] Rainer Poisel and Simon Tjoa. 2013. A comprehensive literature review of file carving. In *2013 International conference on availability, reliability and security*. IEEE, 475–484.

[33] Vijayan Prabhakaran, Lakshmi N Bairavasundaram, Nitin Agrawal, Haryadi S Gunawi, Andrea C Arpaci-Dusseau, and Remzi H Arpaci-Dusseau. 2005. IRON file systems. *ACM SIGOPS Operating System Review* 39, 5 (2005).

[34] Raycap. 2015. Implementing Reliable Surge Protection in Data Centers (Product Application Note). https://www.raycap.com/wp-content/uploads/2018/02/Protecting-Data-Centers-G09-00-011.pdf.

[35] RedHat. 2018. Controlling the SCSI Command Timer and Device Status Red Hat Enterprise Linux 7. https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/storage_administration_guide/scsi-command-timer-device-status

[36] David Reinsel, John Gantz, and John Rydning. 2018. The digitization of the world from edge to core. *IDC White Paper* (2018).

[37] Golden G Richard III and Vassil Roussev. 2005. Scalpel: A Frugal, High Performance File Carver.. In *Digital Forensic Research Workshop (DFRWS)*.

[38] Ohad Rodeh, Josef Bacik, and Chris Mason. 2013. BTRFS: The Linux B-Tree Filesystem. *ACM Trans. Storage* 9, 3, Article 9 (Aug. 2013), 32 pages. https://doi.org/10.1145/2501620.2501623

[39] S. Shah and J. G. Elerath. 2004. Disk drive vintage and its effect on reliability. In *Annual Symposium Reliability and Maintainability, 2004 - RAMS*. 163–167.

[40] S. Shah and J. G. Elerath. 2005. Reliability analysis of disk drive failure mechanisms. In *Annual Reliability and Maintainability Symposium, 2005. Proceedings*. 226–231.

[41] SourceForge. 2001. Foremost. http://foremost.sourceforge.net/

[42] UEFI Forum, Inc. 2020. *Unified Extensible Firmware Interface (UEFI) Specification*. Specification Version 2.8 Errata A.

[43] Vijayan Prabhakaran, A. C. Arpaci-Dusseau, and R. H. Arpaci-Dusseau. 2005. Model-based failure analysis of journaling file systems. In *2005 International Conference on Dependable Systems and Networks (DSN'05)*. 802–811.

[44] Yongkun Wang, Kazuo Goda, Miyuki Nakano, and Masaru Kitsuregawa. 2010. Early Experience and Evaluation of File Systems on SSD with Database Applications. In *Proceedings of the 2010 IEEE Fifth International Conference on Networking, Architecture, and Storage (NAS '10)*. IEEE Computer Society, USA, 467–476. https://doi.org/10.1109/NAS.2010.12

[45] Warlock. 2021. File carving. https://resources.infosecinstitute.com/file-carving/

[46] Christian Wressnegger, Guido Schwenk, Daniel Arp, and Konrad Rieck. 2013. A Close Look on N-Grams in Intrusion Detection: Anomaly Detection vs. Classification. In *Proceedings of the 2013 ACM Workshop on Artificial Intelligence and Security* (Berlin, Germany) *(AISec '13)*. Association for Computing Machinery, New York, NY, USA, 67–76. https://doi.org/10.1145/2517312.2517316

[47] Jimmy Yang and Feng-Bin Sun. 1999. A comprehensive review of hard-disk drive reliability. In *Annual Reliability and Maintainability. Symposium. 1999 Proceedings (Cat. No. 99CH36283)*. IEEE, 403–409.

[4] https://zenodo.org/record/7051564