Technological Feasibility Documentation

11/09/2023

Team

Kowalski

Sponsor

Western Digital

Rajpal Singh

Team Mentor

Saisri Muttineni

Team Members

Erick Salazar, Bailey McCaulsin, Jake Borneman, Nick Wiltshire

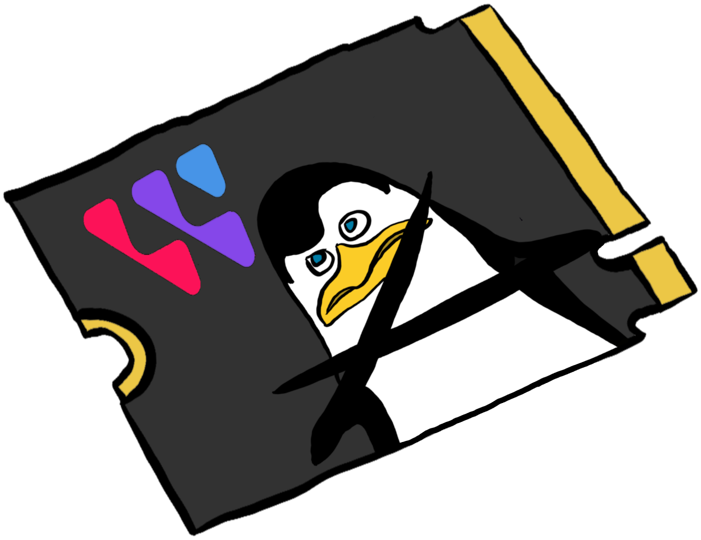


Table of Contents

1. Introduction

In the rapidly evolving digital landscape, data is the lifeblood of any organization. For a company like Western Digital, that produces Solid State Drives (SSDs), the ability to monitor, analyze, and gain insights from the vast amounts of data generated during the research and development process is crucial. Our sponsor at the company, Rajpal Singh, has tasked us to create an insight, analytics, and observability platform to revolutionize this process.

Currently, Western Digital has testing facilities where it runs different types of validations on SSD’s based on the end-user use cases that vary from Client Desktop, OEM Appliances, and cloud use cases. The validation teams who test these SSDs to make sure the product quality meets the customer’s product specifications requirements. Currently there is a lack of an efficient observability framework to capture the insights of system’s kernel behaviors at various stack levels, from the Application to Device Driver level. And then collect, store, analyze, and provide the visibility and monitoring via visual dashboards. This also eliminates the possibility of automation of proactive detection of failures in the drive in the real-time and near real-time scenarios which will allow the validation teams to get notified as soon as issue is detected rather than waiting till end of the validation. This not only costs Western Digital time and money, it also impacts the customer satisfaction indexes.

By implementing our Observability framework, Validation teams will be proactively able to detect issues in the SSD through the eye of the Kernel stack behaviors. On top of that, the data they collect will be presented in an organized, efficient, and effective manner. They will be able to see their data graphically represented in a comprehensive dashboard, equipped to alert when certain performance thresholds are not met. This will have a great impact on the company’s ability to improve their products, and gain a competitive edge over their competitors.

In the following sections, we will be discussing the technological challenges we could possibly encounter during this project, our plan of action for development, and how we will integrate our solution, bringing our project to life. First, we will explore challenges that could present themselves early on in the development process and present possible plans of action for when said challenges arise.

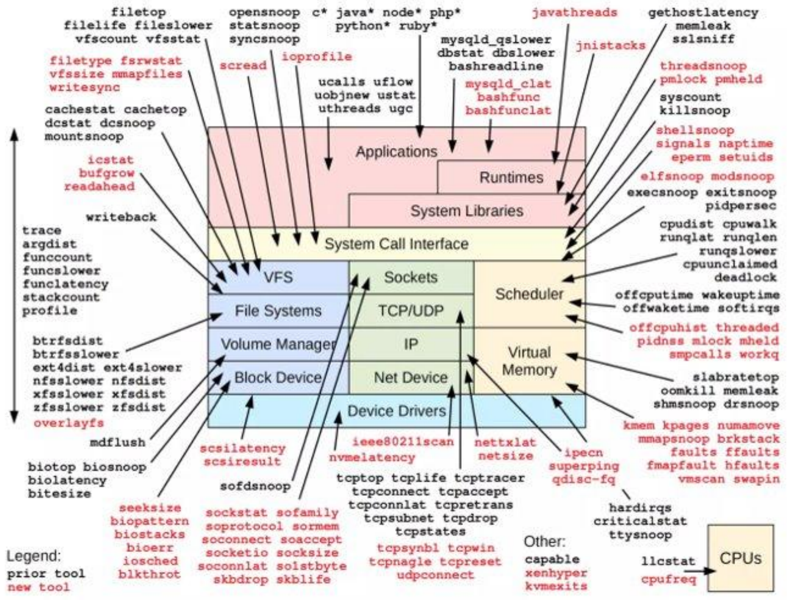
2. Technological Challenges

Every project, regardless of its nature, is bound to encounter challenges. Our solution is divided into five key categories - Collection, Analysis, Delivery, Observability, and Automation. Each category is interdependent, and a failure in one could potentially disrupt the entire system. Therefore, it’s crucial to identify and address potential issues in each category as follows:

2.1 Collection

- **Open “Black Box” Insights:** eBPF tracing technology enables the event-based probing at the kernel stack layers, which will help to capture the stack snapshots and the relevant data with key situational details. Observing a complex level would require high accuracy and understanding of what is occurring for the object being probed.

The “Black Box”



- **Data Relevance**: The data collected must be relevant to the client’s validation use cases. This requires a proper understanding of the client’s product specification requirements and the specific metrics to be met that are important to certain end-customer use cases.

- **Data Transformation**: The raw data collected needs to be transformed into a format that can be easily processed by the other categories. This involves data cleaning, normalization, and possibly feature extraction.

2.2 Analysis

- **Identifying Data of Interes**t: Among the vast amount of data collected, it’s crucial to identify what is valuable and what is redundant. This requires sophisticated data analysis techniques and possibly machine learning algorithms to discern patterns and anomalies.

2.3 Observability

- **Data Presentation**: The analyzed data needs to be presented in a user-friendly manner. This involves designing an intuitive user interface and effective data visualization techniques to communicate the insights derived from the data.

2.4 Automation

- **End-to-End Data Pipeline:** The automation feature must be able to collect kernel insights using eBPF technology, push it to the AWS cloud S3 storage for transformation and analytics, and create the visual dashboards for insights. Enabling real-time and non-real-time logs on critical issues detection on the dashboard.

Having provided an overview of the technological challenges we may encounter in our project, it’s now time to delve deeper into the technological aspects. In the following section, we will conduct a thorough technological analysis. This will involve examining each component of our proposed solution in detail, assessing its feasibility, and determining how it contributes to overcoming the challenges we’ve outlined.

3. Technological Analysis

Western Digital operates numerous data centers, each housing a wealth of information about various data storage devices and their respective models. This information includes details about errors, efficiencies, quantities, and more. However, analyzing this data is a laborious task, as it requires initiating a manual data analysis process for each individual device. This becomes particularly challenging when there's a need to analyze data from thousands of devices simultaneously - a capability that Western Digital currently lacks.

The data analysis is conducted in a cascading manner. Initially, all data stored in the data centers is accessed and probed using eBPF tools at the kernel level. These probes yield data such as latencies, IOPS, firmware behaviors and errors at the system kernel levels. The resulting data is then collected, analyzed, and refactored into a text or CSV file using Python. Once refactored, the data is uploaded to AWS S3 storage, where it undergoes sanitization based on file structure. Ultimately, the data is presented on a virtual dashboard, graphically illustrating the findings for easier comprehension.

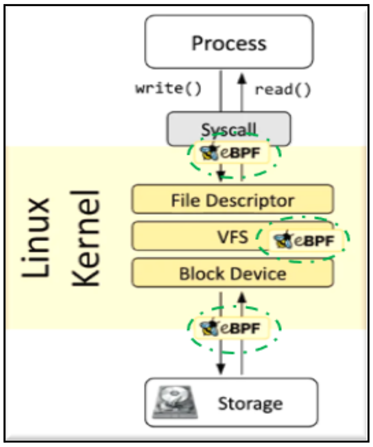
We aim to establish the foundational structure for this system, which involves outlining several expected functionalities as follows:

3.1 Expected Functionalities

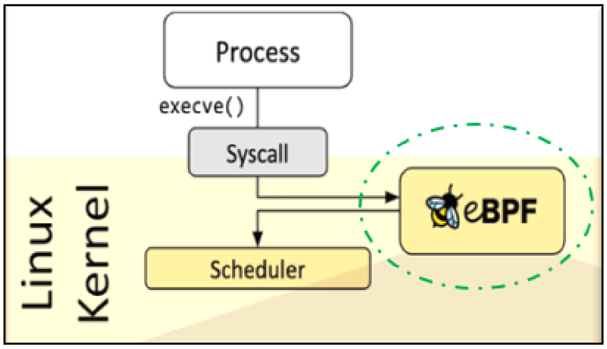
Collection

- Drive Probing
  - Drive data will be collected using eBPF probing programs developed in C and C++.
  - eBPF programs will be attached to Linux and Windows virtual machines at the kernel level using Python scripts.
  - Probes will collect information relevant to storage devices.

- Structuring/Storing Data
  - All output data from probing will be structured and stored into .csv files using Python scripts.



Non-Predefined hooks that can be created to attach to any part of the kernel. In this case and project, onto the Device Drivers, IO and Block devices



Predefined hooks that attach themselves to the kernel. Easily callable as it has been created prior

How and where eBPF event-based hooks can be attached.

Analysis

- File Transfer to Cloud Storage
  - Resulting .csv files will be transferred to AWS S3 Buckets, organized by drive type.

- Crawling and Sanitizing Data
  - Raw data present in .csv files will be converted into tables, ready for parsing and analyzing.
  - Data will be crawled and sanitized using AWS Glue.

- Parsing of Transactional and Master Data
  - Transactional data contains performance statistics and error logs, which will be stored in a noSQL database.

    - Master data contains drive information and specifications, which will be stored in a mySQL database.
    - Data will be Parsed using AWS Glue.

- Specified Data Analysis
    - Data will be organized and thresholds/alerts will be set by desired specifications.
    - Master Data will be analyzed using AWS Athena.
    - Transactional Data will be analyzed using AWS OpenSearch.

Observability

- Merging and Visualizing Data
    - Master and transactional data will be merged and presented in a graphical overview that supports alerts.
    - Data will be merged and visualized in AWS QuickSight.

Automation

- Scripting and API Calls
    - Python scripts executing probing programs will be timed in conjunction with API calls, sending collected data to AWS S3 storage.
    - Automation frequency will be determined by the needs of Western Digital, and will be easily configurable.

Ideally, the solution should be capable of taking the information from the data centers after probing and tracing it, to then collect it and refactor it before having it uploaded to a database on the AWS cloud storage that will later be transformed and display the analyzed data onto a dashboard. It’s in need of mention that this system should be automated to a point where the data for many devices, similar or different, can be analyzed and displayed at once, without requiring individual analyzes for different devices.The main characteristics of this solution are:

3.2 Solution Characteristics

Kernel Level Insight and “Black Box” Understanding

- With not much information about the kernel level’s actions and interactions, being able to be displayed and show the user what is going on at the system stack level can help them understand the corrections and situations that cause issues to certain devices. It would identify the detailed information that leads to object breaking occurrences such as silent failures (failures or occurrences which might not cause a considerable impact now but might become a major issue in the long-term. For eg. Paging error, FileSystem error, etc..)
- Upon creation, the “Black Box” would be more readable at the kernel level that are occuring giving access to the stack layers event-based behaviors much easier with accuracy.

Speed/Cost from AWS Services

- All AWS Services need to be optimized to reduce cost when storing, crawling, analyzing, and querying. Having this characteristic will be essential to keep AWS Services costs to a minimum for Western Digital, as well as, to keep the processing speeds as constant and effective as possible.

Automation

- Automation needs to be easily modified to facilitate company needs. Engineers at Western Digital will need to find the most optimal frequency of automation for our program to be as functional and efficient as possible. This would provide a system that can be used under any schedules the engineers wish to run the program under.

Reliability

- Connection between application level and data pipeline must be reliable in multiple circumstances. This is critical for the program to collect and present necessary data.

Compatibility

- Program must be compatible with drives on windows and linux virtual machines as per the requirements from the client. Meaning the client will test it on different types of virtual machines from different versions of creation. This could range from things as old as Windows 96 to as new as Windows 11 or old linux to the most recent version.

Versatility

- The program must be capable of handling as many devices as needed. This links with the automation process that would be implemented as part of the system.

Maintainability

- The program would be structured in a way that it is easy to maintain, refactored if necessary, and add or subtract any future necessary logic.

3.3 Solution Alternatives

The implementation of these functionalities and characteristics can take various paths, as there are multiple routes to achieve our goal. It is important to discuss possible alternatives for each part of any proposed solution, however, given the structure of our project we are somewhat constrained in the different paths we can take. Our client has already provided us with all the resources and technologies we'll use for the project, but there's some room for interpretation regarding the modules and libraries we'll need to use with specific languages. Our dynamic options are as follows:

Python Requests Library

- Created by Kenneth Reitz in 2011, this library is used for HTTP interactions. It can be used to send collected data to databases for storage.

Python Pandas Library

- Created by Wes McKinney in 2008, this library is designed for data analysis. It will be used to store, analyze, and compile the collected data into a CSV file.

Additional Tools or Libraries

- Any tools or libraries not built in Python must be approved by the client and their company. These could potentially provide alternative methods for data collection, analysis, and storage.

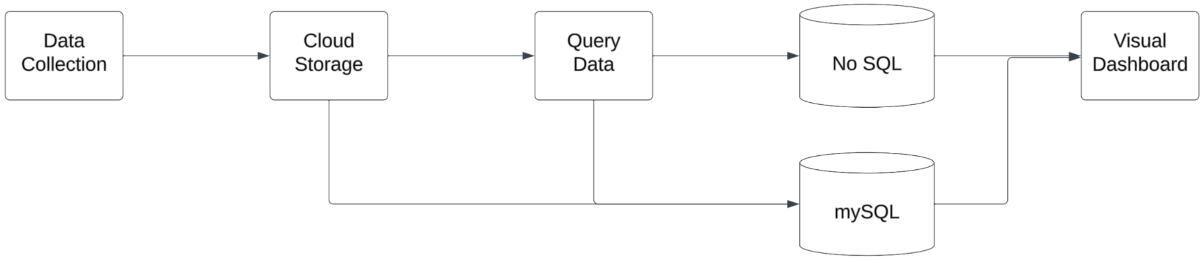
Demo Development

- A live demo will be developed by the end of the semester to test and validate the solution. This demo will probe systems using eBPF, generate SSD performance data, send this data to an S3 bucket, and pipeline it into a simple dashboard.

4. Technology Integration

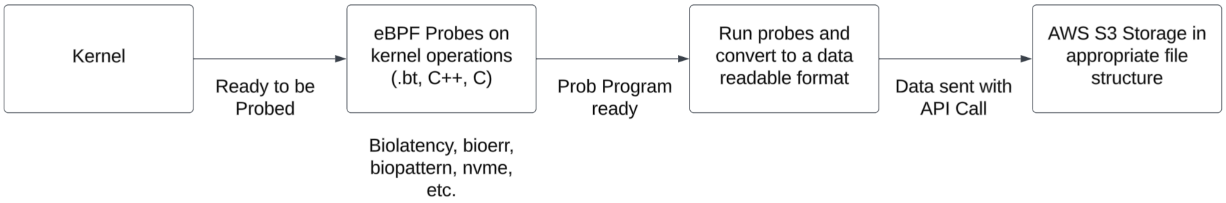
With all the sub-solutions laid out, we need to create a coherent system that consolidates each step into a working product. Each step needs to seamlessly integrate into the next in order for our platform to be a feasible solution for Western Digital.

Product Solution Overview:



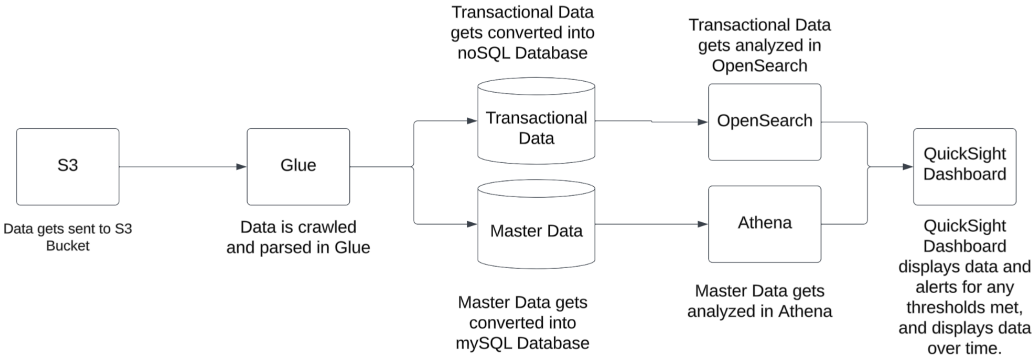
This diagram represents the overall system pipeline, from collecting drive data, sending it through the cloud pipeline, and visualizing it.

Data Collection Overview


Kernel
Ready to be Probed
eBPF Probes on kernel operations (.bt, C++, C)
Biolatency, bioerr, biopattern, nvme, etc.
Prob Program ready
Run probes and convert to a data readable format
Data sent with API Call
AWS S3 Storage in appropriate file structure

This diagram represents the process of probing, formatting, and sending data to the cloud data pipeline.

Cloud Pipeline Overview

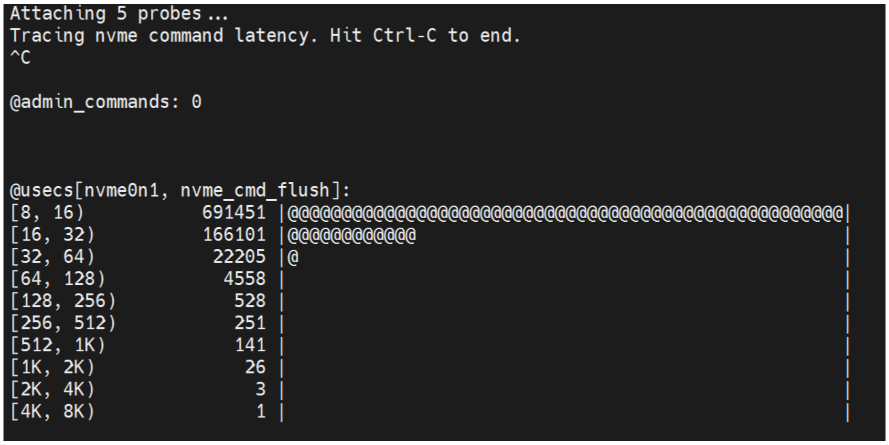

Transactional Data gets converted into noSQL Database
Transactional Data gets analyzed in OpenSearch
S3
Glue
Transactional Data
OpenSearch
QuickSight Dashboard
Master Data
Athena
Data gets sent to S3 Bucket
Data is crawled and parsed in Glue
Master Data gets converted into mySQL Database
Master Data gets analyzed in Athena
QuickSight Dashboard displays data and alerts for any thresholds met, and displays data over time.

This diagram represents the AWS cloud pipeline from stored data in S3, to the observability platform in QuickSight.

5. Use Cases

Automated Kernel-Level Data Collection

- *Description:* The system automatically collects kernel-level data from storage devices using eBPF probes and tracepoints.
- *Actors:* Data Collection System, Kernel-level Probes.
- *Steps:*
    1. Probes and tracepoints are attached to the Linux kernel.
    2. Relevant data, including errors and latencies, is collected from storage devices.
    3. Collected data is structured and stored in a CSV file.
- Sample logs: NVMe Latency



```
Attaching 5 probes...
Tracing nvme command latency. Hit Ctrl-C to end.
^C

@admin_commands: 0



@usecs[nvme0n1, nvme_cmd_flush]:
[8, 16)           691451 |@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@|
[16, 32)          166101 |@@@@@@@@@@@@                                        |
[32, 64)           22205 |@                                                   |
[64, 128)           4558 |                                                    |
[128, 256)           528 |                                                    |
[256, 512)           251 |                                                    |
[512, 1K)            141 |                                                    |
[1K, 2K)              26 |                                                    |
[2K, 4K)               3 |                                                    |
[4K, 8K)               1 |                                                    |
```

Kernel Level Insight

- *Description:* The probing of data would collect and result in accuracy of the data useful for understanding the Kernel Level's actions and understanding
- *Actors:* Kernel-Level Probes, Data Collection System
- *Steps:*
  1. Capturing useful behavior that would describe the Kernel-Level's behavior.
  2. Identifying the data collected validating the SSD and observing "Silent Failures" .

Data Analysis and Abnormality Detection

- *Description:* The collected data is analyzed to identify abnormalities, errors, and performance issues.
- *Actors:* Data Analysis System, Python Wrapper.
- *Steps:*
  1. Data from CSV files is parsed and analyzed to identify patterns and abnormalities.
  2. Specific data points are marked as objects of interest.
  3. Abnormalities are detected and categorized based on predefined thresholds.

Data Transfer and Sanitization

- *Description:* Collected and analyzed data is transferred to AWS S3 storage and sanitized for further processing.
- *Actors:* Data Transfer System, AWS S3, Glue (for sanitization).
- *Steps:*
  1. Analyzed data is transferred to AWS S3 cloud storage.
  2. Data is sanitized and organized into separate tables using Glue.

Data Integration and Storage

- *Description:* Sanitized data is parsed into transactional (NoSQL) and master data (MySQL) for efficient storage and retrieval.
- *Actors:* Database System, NoSQL Database, MySQL Database.
- *Steps:*
  1. Transactional data (performance data and error logs) is stored in the NoSQL database.
  2. Master data (product information and specifications) is stored in the MySQL database.

3. Data is organized and indexed for quick retrieval and analysis.

Data Visualization and Dashboard Creation

- *Description:* The integrated data is visualized and displayed on a user-friendly dashboard for end-user observation.
- *Actors:* Visualization System, QuickSight (or similar tool), End Users.
- *Steps:*
    1. Merged data from NoSQL and MySQL databases is processed.
    2. Data is visualized using QuickSight, applying appropriate filters and thresholds.
    3. Users can access the dashboard to observe kernel-level data, errors, and performance metrics.

Automated Data Collection and Continuous Monitoring

- *Description:* The system is automated to continually run, collect recent data, and update the dashboard without manual intervention.
- *Actors:* Automation System, Scheduler, Data Collection System.
- *Steps:*
    1. Automation system triggers data collection processes at defined intervals.
    2. Recent data is collected and transferred to storage systems.
    3. Dashboard is automatically updated to reflect the most recent data.

6. Conclusion

Western Digital is a lead manufacturer of storage devices. They currently do not have a system for collecting kernel level data and displaying it to an end user. With that being said, they need a way to measure performance, error detection, and data collection of operations occurring at the kernel level in a way that is convenient and effective. So our goal is to create a platform that promotes insight, automation, efficiency, and convenience to our client Western Digital. The platform will do this by establishing a data pipeline that spans 4 levels: Data Center, AWS S3 storage, Database, and Dashboard. This structure allows for the flow of data from the kernel level to a data dashboard without user interference. This way the end user can see kernel level operation details as it pertains to memory devices and be able to identify the kernel level's functionalities. Having understanding of the validation process of the tested hardware and the Silent Failures that might occur upon testing. It completes without having to get their hands dirty and manually dealing with the kernel level.

Though there may be technical challenges along the way, our team is dedicated to producing a product for our client that meets all requirements. It is our mission to overcome technological barriers concerning: collection, insight, analyzing, delivering, observability, and automation. We understand that overcoming these challenges will require us to find alternative solutions, team collaboration, and ask for support from our client.

Through our technological analysis, we understand the technologies we will need to use and why we are using them. Each layer we express in the technological analysis is vital to the project's success. That is, all the while, making sure that we take all seven main characteristics of the solution into consideration. As a road map we plan on using the provided diagrams in the technological integration section.

From here, we will be getting our hands dirty with the technologies we will be using. This involves reading documentation, creating sample programs, and trying different things out. We will also be meeting with our client once or twice a week to discuss progress, questions, and other information to ensure a strong communication between our capstone team and our client. We also have already been assigned our tasks by our client so everyone will be working on learning things in their respective area. With this being said, we are very excited to get this project rolling and work its way to completion.