Big Data Blockchain

Christopher Nastasi
Department of Computer Science
Texas A&M University-San Antonio
San Antonio, Texas, USA
cnast01@jaguar.tamu.edu

***Abstract*—This research paper aims to answer the question of using blockchain to secure transactions for big data while maintaining the data integrity. In this paper we discuss the background of blockchain and traditional database systems. Create a test environment for each module using a python server. Run the tests and compute the times of the test, compare how each system does, and discuss the scalability and security. The results of the tests are that the blockchain is slower but has better security mechanisms to prevent malicious events. The traditional database system is faster but has less security. The results of the tests show that the blockchain is slower but has better security mechanisms to prevent malicious events. The traditional database system is faster but has less security. The results of the tests show that the blockchain is slower but has better security mechanisms to prevent malicious events. The traditional database system is faster but has less security.**

***Index Terms*—Blockchain, Database, Secure, Data, Integrity, Comparison**

I. Introduction

In this paper we want to explore how blockchain for big data can preserve integrity and stack up against a traditional database system. We also want to analyze and compare how scalable and secure blockchain can be against a traditional database system. But before we explore this topic let's start from the very foundation of what a blockchain is. In short, blockchain is a technology like a linked list that is exactly what it sounds like, a block and a chain. What this technology does is almost makes a self-maintaining connection of nodes that each validate each other's work and before it can be authorized and sent to an immutable chain. A traditional database on the other hand is one that is common in today's environments. It is mostly a SQL relational database that is maintained by the owners of the server.

II. Background

A. Blockchain

To be able to understand how this experiment was conducted we need to explore the fundamentals of blockchain systems and a traditional database system a little more. Blockchain is not a relatively new idea when you compare it to a linked list, but the application of a linked list to maintain itself and be decentralized is. This is blockchain in a nutshell. There are four key components to a blockchain, the transaction, the block, the consensus mechanism, and the blockchain itself. What makes blockchain great is the immutability, decentralization transparency, and cryptography of blockchain. All these concepts are wrapped into what a blockchain is.

1) Key Components of Blockchain: When we dissect the blockchain, we see the four components. Starting with a transaction, this is where the end-user can create a new piece of data or event that is to be sent to the blockchain. Typically, there needs to be some kind of authentication to prove you are the person and you own what you are sending. I will dive into project specifics in the project setup of the blockchain. But this brings up an important and maybe one of the most important concepts of blockchain.

2) Block: Moving to the next component, we have a block. This is a collection of transactions are stored together to await the consensus mechanism validation to be stored on the blockchain. The block is a key part of what cryptographic hashing is for and improves the performance of a blockchain. This is because it would be incredibly resource intensive to have each transaction be verified and stored on the blockchain by itself and the chain would be exponentially longer than storing it in blocks. Cryptographic hashing is a key role in this step. To make a blockchain tamper proof, a cryptographic hash is created that encapsulates the block with unique values like the time it was created and specific hashing algorithms that can include other properties of the block. This is one way that blockchain can be secured when having data transactions on a supply chain. By having the transaction which is authorized by the user having a specific key or account, we can then add further security to the blockchain by using cryptographic hashing for each block.

3) Consensus Mechanism: This brings us to the next component, which is the consensus mechanism. In this step we have a validation technique, and this is probably the most critical part of making it decentralized and secure. This consensus mechanism is what is used for nodes (a separate instance of the chain) and ledgers (copies of transactions on a chain) on the blockchain systems to maintain itself. There are more than a few ways to do this but in terms of this project we simulated a proof-of-work concept which tries to solve the hash created for the block that can then be validated and added to the blockchain. POW is an interesting concept because it uses computational power to solve this very complex hash, and typically miners (the nodes that try to solve the hash to add to the chain) are rewarded for their compute and correctness.

4) Blockchain: Lastly, we have the blockchain itself, which is a collection of the blocks that have been verified by the nodes and are ready added to the overall network of nodes and ledgers. This is considered immutable and highly tamper-proof

due to the hash that creates a link between the previous and next block. The Cryptographic hashing in a block is completed by adding the previous hash of the block into the next block. This makes it nearly impossible to alter previous blocks and get away with it.

B. Traditional Database

A traditional database does not require as much background information as a blockchain because it is widely used and if you are reading this paper, you have some knowledge of a traditional database. But in a traditional database we have the physical database itself, which is a centralized database because it is not distributed amongst nodes, and it is controlled by one or very few entities.

We then have the authorizations and authentications which are performed by the database management system (DBMS) to ensure that only authorized users can access or modify the data. This is typically done through a combination of user accounts, passwords, and permissions that are set at various levels (e.g., table, row, column) within the database.

III. Methodology/Approach

A. Project Overview

The project aims to implement a simple blockchain system using Python. The primary objective is to understand the key components of blockchain technology and how they interact with each other. The project will also explore the differences between blockchain and traditional database systems, particularly in terms of security and data integrity.

B. Project Setup

The project setup is a simple client-server architecture. The server is a python server that is running the blockchain and the client is a simple python script that sends data to the server. The server is running on port 5000 and the client is running on port 5001. The server is a simple REST API that accepts POST requests and returns a JSON response. The client sends a POST request to the server with the data to be stored in the blockchain. The server then creates a new block and adds it to the blockchain. The server also has a GET request that returns the entire blockchain. The client can also send a GET request to the server to get the entire blockchain. The server is running on a local machine and the client is running on a different machine. The server is running on port 5000 and the client is running on port 5001. The server is a simple REST API that accepts POST requests and returns a JSON response. The client sends a POST request to the server with the data to be stored in the blockchain. The server then creates a new block and adds it to the blockchain. The server also has a GET request that returns the entire blockchain. The client can also send a GET request to the server to get the entire blockchain. The server is running on a local machine and the client is running on a different machine. The server is running on port 5000 and the client is running on port 5001

C. Test Environment

The test environment is a simple client-server architecture. The server is a python server that is running the blockchain and the client is a simple python script that sends data to the server. The server is running on port 5000 and the client is running on port 5001. The server is a simple REST API that accepts POST requests and returns a JSON response. The client sends a POST request to the server with the data to be stored in the blockchain. The server then creates a new block and adds it to the blockchain. The server also has a GET request that returns the entire blockchain. The client can also send a GET request to the server to get the entire blockchain. The server is running on a local machine and the client is running on a different machine. The server is running on port 5000 and the client is running on port 5001.

D. Test Cases

The test cases are simple and straightforward. The first test case is to send a POST request to the server with a simple string as the data. The server then creates a new block and adds it to the blockchain. The second test case is to send a POST request to the server with a JSON object as the data. The server then creates a new block and adds it to the blockchain. The third test case is to send a GET request to the server to get the entire blockchain. The server then returns the entire blockchain as a JSON object.

IV. Results/Demonstration

The results of the tests are that the blockchain is slower but has better security mechanisms to prevent malicious events. The traditional database system is faster but has less security. The results of the tests show that the blockchain is slower but has better security mechanisms to prevent malicious events. The traditional database system is faster but has less security. The results of the tests show that the blockchain is slower but has better security mechanisms to prevent malicious events. The traditional database system is faster but has less security.

1) Performance Comparison: When looking at the performance of the two systems, we can see that the blockchain system is slower than the traditional database system. This is due to the fact that the blockchain system has to go through a consensus mechanism to validate each transaction before it can be added to the blockchain. The traditional database system does not have this overhead and can process transactions much faster.

2) Security Comparison: In terms of security, the blockchain system offers several advantages over traditional database systems. The decentralized nature of blockchain makes it more resistant to attacks, as there is no single point of failure. Additionally, the use of cryptographic techniques ensures the integrity and authenticity of the data stored on the blockchain. On the other hand, traditional database systems are more vulnerable to attacks, as they rely on a centralized architecture and are often targeted by malicious actors.

A. Scalability Comparison

When it comes to scalability, traditional database systems have the upper hand. They can easily scale vertically by adding more resources to a single server or horizontally by distributing the load across multiple servers. In contrast, blockchain systems face challenges in scaling due to their consensus mechanisms and the need to maintain a copy of the entire blockchain on each node. This can lead to performance bottlenecks as the number of transactions increases.

V. Conclusion

In conclusion, the blockchain system offers enhanced security features compared to traditional database systems, albeit at the cost of speed. The results indicate that while the blockchain is slower, it provides better mechanisms to prevent malicious activities. On the other hand, traditional database systems, while faster, lack the same level of security. These findings highlight the trade-offs between speed and security in data management systems.

References

[1] G. Eason, B. Noble, and I. N. Sneddon, "On certain integrals of Lipschitz-Hankel type involving products of Bessel functions," Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955.

[2] J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[3] I. S. Jacobs and C. P. Bean, "Fine particles, thin films and exchange anisotropy," in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] R. Nicole, "Title of paper with only first word capitalized," J. Name Stand. Abbrev., in press.

[6] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, "Electron spectroscopy studies on magneto-optical media and plastic substrate interface," IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.

[8] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," 2013, arXiv:1312.6114. [Online]. Available: https://arxiv.org/abs/1312.6114

[9] S. Liu, "Wi-Fi Energy Detection Testbed (12MTC)," 2023, gitHub repository. [Online]. Available: https://github.com/liustone99/Wi-Fi-Energy-Detection-Testbed-12MTC

[10] "Treatment episode data set: discharges (TEDS-D): concatenated, 2006 to 2009." U.S. Department of Health and Human Services, Substance Abuse and Mental Health Services Administration, Office of Applied Studies, August, 2013, DOI:10.3886/ICPSR30122.v2

[11] K. Eves and J. Valasek, "Adaptive control for singularly perturbed systems examples," Code Ocean, Aug. 2023. [Online]. Available: https://codeocean.com/capsule/4989235/tree