Final Report

Securing NoSQL databases using blockchains

MongoDB + ARK Blockchain

Gurpreet, Paul and Fernando

April 12, 2018

CS4411

Securing NoSQL databases using blockchains

Introduction

The contents of a modern production database should be secure and interactions should be easily verifiable. We would like to propose a solution that strengthens the security of modern NoSQL databases using bleeding edge blockchain solutions.

In current database management software, the only way to verify the interactions that occur within a database is by checking the log created by the system. The problem is that logging can easily be disabled or reduced to a level where malicious actions could occur without leaving a trace.

The solution we are proposing is to use hashing to capture the active state of a Mongo database and use an Ark blockchain to verify it's integrity. We would like to be able to specify a query resulting in a set of documents. The result will then be hashed and stored on the blockchain to a corresponding address. This functionality could then be automated to check the database's secured queries periodically to effectively secure it.

Motivation

The motivation for combining these two technologies into a security solution comes from pre-existing database logging functionality. The common way to trace back the interactions that occur within a database management system is to look at its logs. Database systems have the ability to disable logs or only record logs up to a specific error level. When relying on these logs to make sure no malicious activity has taken place, the investigator has to assume that the logging is setup correctly and no one disabled or tampered with it during the attack. This is where the problem arises; there is no completely secure way to verify the state of a database and ensure no one has tampered with the documents.

Databases provide fast and reliable data storage systems and blockchains provide secure, distributed and immutable data storage. The problem is interesting because it has to potential to leverage the best qualities of both these technologies and improve data storage all together.

The idea of using a blockchain to secure the data stored in other types of storage systems has been attempted by a web service called Tierion using their token TNT [2]. This system provides end points where you can send a hash and it sends back a timestamp ensuring it has been pushed to

a blockchain. A downside to this system is that it is a Software as a Service (SAAS) and therefore costs money to use on a large scale deployment. Private companies may not wish to have to send hashes of their data to an external service and Tierion does not allow for you to deploy your own instance of their system. The system itself is not very impressive either as it just provides a simple interface to the Bitcoin and Ethereum blockchains, neither of which are set up to handle such a task. Both BTC and ETH have long transaction times and were developed with a different purpose in mind. Tierion can take more than 10 minutes to push a hash to one of the chains which is too slow for an automated system. The idea is much better suited to data oriented blockchains such as the own we will be using called Ark. Ark has 7 second block times which will allow fast automated processing.

Ark is a Delegated Proof of Stake (DPOS) blockchain which allows transactions to carry a small amount of data [1]. Each transaction has more than enough space to carry the hash of a single document. Using this free and open source blockchain also allows us to run a local instance, which means we don't have to pay to secure our data and no information is exposed to the outside world.

Architecture and Environment

Implementation

Developing the ArkMongo project came with many difficulties and challenges because it was quite an innovative idea using new technologies. We didn't have many resources to base the implementation off of and we were all fairly inexperienced with the use of databases combined with a blockchain.

In order to gain enough experience with the technology to develop such a solution we looked at it in major sections and tackled each individually. The first section and challenge was creating a hash of a document set within mongodb. Since we were going to write the command line utility in Ruby, we also did all of our prototyping in Ruby as well. The solution we came to for hashing document sets was to run a query and add each document returned to a sha256 hash. To do this we used the Digest built in library in Ruby. One of the bugs we encountered in this stage was that the hash wasnt being stored to the database correctly. We later discovered that we had to encode the hash in hex form so we could view it correctly.

The next challenge was saving to the Blockchain and database. The database was fairly simple because we learned how to do this in class, but setting up a local blockchain was not an easy task. It took us many iterations of configurations to have deployed a blockchain that worked correctly for our task. The major challenge we faced in this step was discovering a protocol that would allow us to discover data we send to the blockchain. We decided to allocate one address per individual query and then store that address inside our caching database so we can discover it again and add to it later.

Results

Conclusion and Future Work

Currently, database logging isn't a completely reliable way to verify if someone has tampered with the documents in a database. By combining databases fast and reliable data storage and blockchains secure, distributed and immutable data storage, we would be able to secure a locally stored traditional database without compromising much of the speed of the database system. By choosing Ark as our blockchain technology, we will be able to deliver a feasable solution for securing databases.

References

[1] The ARK Crew. *A Platform for Consumer Adoption*. February 25, 2018. URL: https://ark.io/Whitepaper.pdf.

[2] Jason Bukowski Wayne Vaughan and Glenn Rempe. *A global platform for verifiable data*. February 25, 2018. URL: https://tokensale.tierion.com/TierionTokenSaleWhitePaper.pdf.