Electronics and Computer Science
Faculty of Physical Sciences and Engineering
University of Southampton

Thomas Smith, tcs1g20
November 24, 2022

Using Blockchain for Video Game Distribution

Project Supervisor: Leonardo Aniello
Second Examiner: tbd

A project report submitted for the award of
BSc Computer Science

Abstract

max 200 words

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must <u>change the statements in the boxes</u> if you do not agree with them.

We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

I have not used any resources produced by anyone else.

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself, or with my allocated group, and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs.

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

I have not submitted any part of this work for another assessment.

If your work involved research/studies (including surveys) on human participants, their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

My work did not involve human participants, their cells or data, or animals.

Acknowledgements

I would like to thank my supervisor, Leonardo Aniello, for his support throughout this project.

Contents

Abstract i
Statement of Originality i
Acknowledgements iii

1 Problem Statement **1**
1.1 Goals 1
1.2 Scope 2

2 Background Research **3**
2.1 P2P File Sharing Networks 3
2.1.1 BitTorrent 3
2.1.2 Napster 3
2.2 Ethereum 4
2.2.1 Smart Contracts 4
2.2.2 Proof-of-stake 5

3 Literature Review **6**
3.1 Blockchain-Based Cloud Storage 6

4 Design **8**
4.1 requirements 8
4.2 Justification of Approach 8
4.3 Limitations 8

5 Project Management **10**
5.1 Gantt Chart 10
5.2 Risk Assessment 10

References **12**

Chapter 1

Problem Statement

Video games are large and highly popular pieces of software and this means that individual developers will typically not have the infrastructure to distribute their application at a large enough scale. To solve this, third party platforms, like Steam or Epic Games, are used facilitate the distribution of games and their subsequent updates. Some of the issues with doing this are that:

- they take a cut of all revenue, for example Steam take 30%,
- they are vulnerable to censorship, for example the Chinese version of Steam is heavily censored, and
- if the platform shuts down, the user will likely lose all access to their games.

A distributed solution could solve all of these problems. The developer would not have to pay a cut of revenue but could offer in-game incentives to get users to help seed their software, content cannot be restricted or censored due to the nature of peer-to-peer networks and a user's access to a game is not tied to any one platform.

1.1 Goals

The goal of this application is to create a video game distribution platform, which allows developers to independently release their games whilst being able to offer high availability and be immune to censorship from larger bodies (like governments). Blockchain technology should enable this through its trustless property and means that users can trust the software they receive through the network and be able to use public key infrastructure to verify the uploader of the software.

It is an important consideration of this project that users can be distinguished by whether they have purchased the software or not. This information must be publicly available and verifiable by any node in the network.

Availability is an important factor for games distributed through this network so a user will also need to be able to prove their contribution to the network and have this verifiable by any node in the network. The idea is that the developer can identify users who have helped distribute their game and provide them with rewards to do this. It is expected that developers should provide rewards after certain distribution milestones to encourage long term contribution from their user-base. In-game rewards may be a suitable reward for users and the quality of these will have a direct impact on the contribution by the community.

1.2 Scope

For this project to be successful, it should:

- distribute software between nodes by sharing fixed length shards of data,
- use the blockchain to store software metadata that helps to identify and verify the software,
- allow developers to publish their games to the blockchain,
- allow developers to publish updates to their games already on the blockchain,
- use encryption techniques to provide secure sharding of data between nodes,
- use digital signatures to verify the identity of uploaders,
- suggest a proof of purchase system to be called before sharing data with a node, and
- be deployed to an ethereum testnet.

This project will not:

- offer a graphical user interface for users to locate software, and
- implement a way for users to pay for games through the network.

Chapter 2

Background Research

2.1 P2P File Sharing Networks

2.1.1 BitTorrent

BitTorrent [?, 3, 8] is easily the most popular p2p file-sharing platform, in which users will barter for chunks of files by downloading and uploading them in a tit-for-tat fashion, such that peers with a high upload rate will typically also have a high download rate. For a user to download data from BitTorrent they would:

1. Find the corresponding .torrent file that contains metadata about the torrent such as the location of a tracker, file information such as name, size and path in the directory.
2. The user will find peers also interested in that torrent through a tracker and will establish connections with them.
3. The data is split into constant-sized blocks and are downloaded individually. BitTorrent uses a tit-for-tat mechanism that incentivises users to contribute by providing preferable treatment to nodes who upload data as well.
4. The user will download blocks based upon the following priority:
   (a) **Strict Priority** Data is split into pieces and sub-pieces with the aim that once a given sub-piece is requested then all of the other sub-pieces in the same piece are requested
   (b) **Rarest First** Aims to download the piece that the fewest peers have to increase supply.
   (c) **Random First Piece** When a peer has no pieces, it will try to get one as soon as possible to be able to contribute.
5. The node will continuously upload blocks it has while active.

It is commonly suggested that availability of torrents is the biggest issue surrounding BitTorrent as *'38% of torrents become unavailable in the first month'* [3] and that *'the majority of users disconnect from the network within a few hours after the download has finished'* [8]. This paper [7] looks at how the use of multiple trackers for the same content and DHTs can be used to boost availability.

2.1.2 Napster

Napster uses a large cluster of central servers that maintains an index of every file that are currently being shared by peers in the network. Every peer has to maintain a connection to the central servers and will send queries to it to find files and will be given a list of peers and metadata such as their bandwidth; peers will then form a connection and share files. This means that although peers download files off of each other, they are still reliant on a third party to find each other.

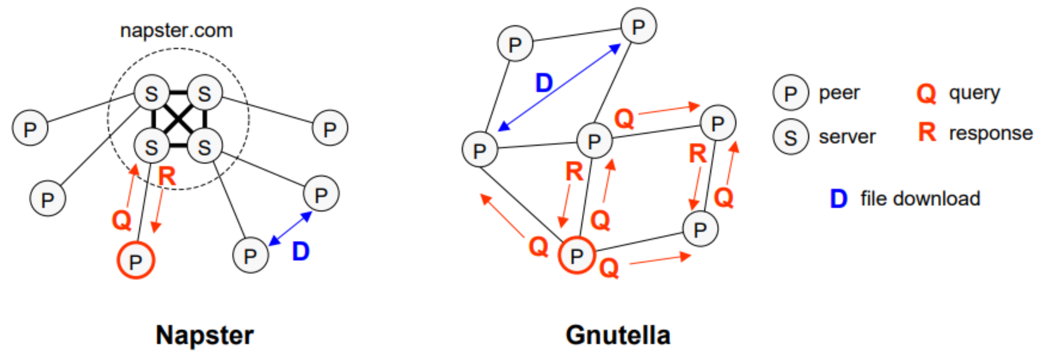

Figure 2.1: Napster vs Gnutella [9, Figure 1]

Gnutella

Peers using Gnutella will form an overlay network by having connections with sets of neighboring nodes, which are found using *ping, pong* messages, where if a node receives a *ping* message they send a *originator* and forward the *ping* to its neighbors. To download a file, a node will flood a message to its neighbors, who will check if they have it and return a message saying so. Regardless, the node will continue to flood through the overlay to find suitable peers.

2.2 Ethereum

Ethereum is a distributed, transaction-based blockchain that allows the deployment of decentralized applications through the use of smart contracts.

2.2.1 Smart Contracts

Each Ethereum blockchain will contain a smart contract that is an executable piece of code, written in Solidity, that is to be ran by every node in the network, using the Ethereum Virtual Machine. Smart contracts provide a concise set of instructions that produces a predictable outcome, such that given the same input will always produce the same result. As smart contracts are public, every node on the network can see the result of each execution and track things like currency and asset transfers across the network.

Gas is a unit of measurement that is used to specify the computational effort required to execute operations on the Ethereum network. Each transaction must set a limit on the amount of gas can be used during code executing, and this is paid using ether. However, this can lead to smart contracts failing to execute due to *running out of gas*. By tying the computational effort of a smart contract to ether, the Ethereum network reduces the risk of DoS attacks as an attacker will likely not have the funds to perform such an attack.

Some example use cases of smart contracts are: creating & distributing digital assets, decentralized gaming, insurance policies, and financial services.

2.2.2 Proof-of-stake

Ethereum was initially a proof-of-work [6] based blockchain but has since moved over to a proof-of-stake system [6], with the idea that it can save energy. The idea behind proof-of-stake is that a user with a larger stake in the network will have a greater chance of mining a block. It uses the idea of coin age, which takes into account the amount of currency a user has and how old it is.

Chapter 3

Literature Review

3.1 Blockchain-Based Cloud Storage

In table 3.1, I detail some examples of how blockchain has been used to solve problems within cloud storage systems. One interesting finding from these papers [11, 2] is that blockchain can been used to provide additional services to cloud storage systems, such as monitoring access control or helping to verify the integrity of the data on these systems.

Paper	**Description of Solution**
Blockchain Based Data Integrity Verification in P2P Cloud Storage [12]	This paper uses Merkle trees to help verify the integrity of data within a P2P blockchain cloud storage network as well as looking at how different structures of Merkle trees effect the performance of the system.
Deduplication with Blockchain for Secure Cloud Storage [5]	This paper describes a deduplication scheme that uses the blockchain to record storage information and distribute files to multiple servers. This is implemented as a set of smart contracts.
Block-secure: Blockchain based scheme for secure P2P cloud storage [4]	A distributed cloud system in which users divide their own data into encrypted chunks and upload those chunks randomly into the blockchain, P2P network.
Blockchain-Based Medical Records Secure Storage and Medical Service Framework [1]	Describes a secure and immutable storage scheme to manage personal medical records as well as a service framework to allow for the sharing of these records.
A Blockchain-Based Access Control System for Cloud Storage [11]	This paper describes a method for using blockchain to facilitate the access control over a cloud storage system. The blockchain stores an immutable record of all '*meaningful security events*', such as key generation, access policy, assignment, etc.

Cloud Data Provenance using IPFS and Blockchain Technology [2]	Uses blockchain technology and IPFS to provide an efficient way to securely store provenance [1] data such that it is out of reach of adversaries, but can be used to verify the integrity of data on a cloud storage system.

Table 3.1: *Examples of blockchain cloud storage systems [10]*

[1]Provenance data are access logs of stored data that can trace the integrity of data and will contain private user information.

Chapter 4

Design

4.1 requirements

ID	Description
	Must
M1	Store software data on a blockchain, including a reference to a previous block where appropriate
M2	Construct a Merkle Tree of any given data object
M3	A node must request individual shards from its peers
M4	A node must be able to discover peers relevant to the software it wants
M5	Software must be updatable through the blockchain
M6	A node must be able to upload software
	Should
S1	Allow users to restrict their software to only a specific set of nodes
S2	Implement a CLI for users to interface with
S3	Allow a node to prove they have helped distribute software
	Could
C1	Allow users to request specific software versions
C2	Allow nodes to join groups for automatic identity verification

4.2 Justification of Approach

4.3 Limitations

One of the major limitations of this project is that it will not support any of the social features, such as achievements, message boards, friends, etc., of platforms like Steam. Whilst, many of these can be supplemented through social platforms, like Discord, it may provide a disconnected social experience for playing games.

On top of this due to the discontinuous nature of P2P file-sharing, the long-term availability of games is dependant on an active community or the developers ability to upload themselves. This means that as games get older or support from developer is discontinued, a game may no longer be available to download at all.

Chapter 5

Project Management

5.1 Gantt Chart

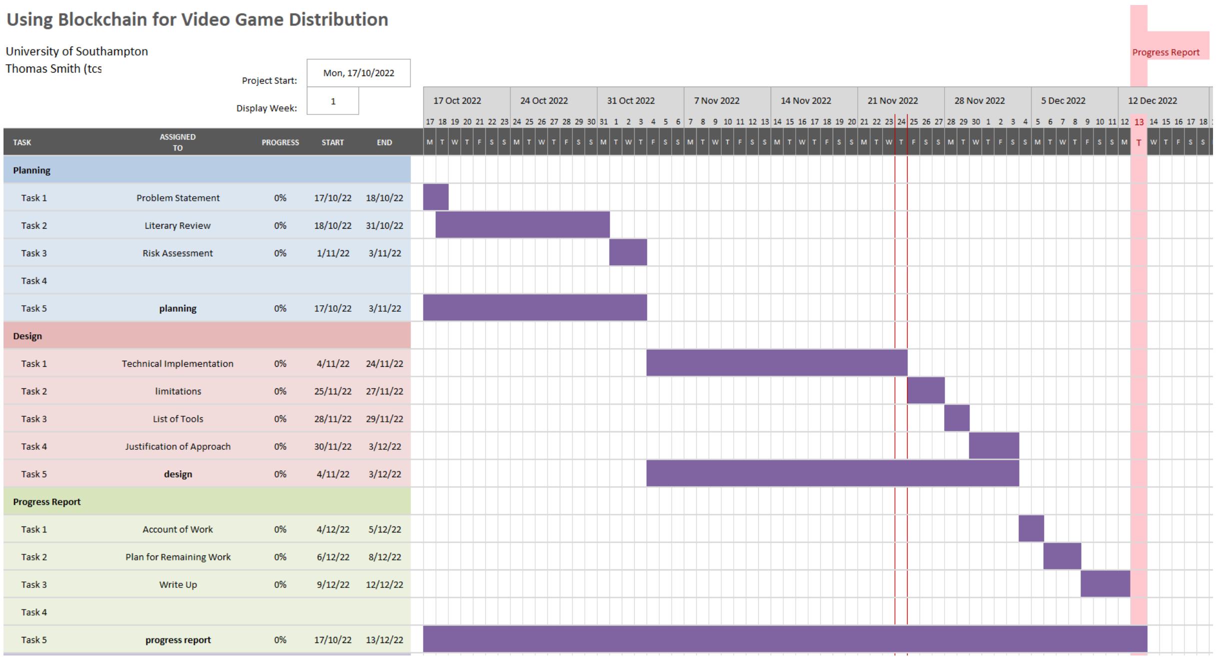

Figure 5.1: *A Gantt chart for my work up until the progress report.*

5.2 Risk Assessment

Risk	Loss	Prob	Risk	Mitigation
Laptop damaged or lost	3	1	5	All work is stored using version control and periodic backups will be made and stored locally and in cloud storage. I have other devices that could be used to continue development.

Difficulty with blockchain development	2	3	6	I will seek advice from my supervisor about how to tackle certain problems and if necessary, what aspects of my project I should change.
The application is not finished	1	3	3	Using agile development will ensure that I will at least have a minimal working application. If I feel that I am running out of time, I will focus on expanding test cases and improving the write-up.
No suitable large scale test environment	2	5	10	I do not have the infrastructure to test this project on a large network, however small scale tests will be possible.
Personal illness	3	2	6	Depending on the amount of lost time, I may have to not complete some of the SHOULD or COULD requirements.

Table 5.1: *The risk assessment of this project.*

Bibliography

[1] Chen, Y., Ding, S., Xu, Z., Zheng, H., and Yang, S. Blockchain-based medical records secure storage and medical service framework. 5.

[2] Hasan, S. S., Sultan, N. H., and Barbhuiya, F. A. Cloud data provenance using IPFS and blockchain technology. In *Proceedings of the Seventh International Workshop on Security in Cloud Computing*, SCC '19, Association for Computing Machinery, pp. 5–12.

[3] Kaune, S., Rumín, R. C., Tyson, G., Mauthe, A., Guerrero, C., and Steinmetz, R. Unraveling BitTorrent's file unavailability: Measurements and analysis. In *2010 IEEE Tenth International Conference on Peer-to-Peer Computing (P2P)*, pp. 1–9. ISSN: 2161-3567.

[4] Li, J., Wu, J., and Chen, L. Block-secure: Blockchain based scheme for secure p2p cloud storage. 219–231.

[5] Li, J., Wu, J., Chen, L., and Li, J. Deduplication with blockchain for secure cloud storage. In *Big Data*, Z. Xu, X. Gao, Q. Miao, Y. Zhang, and J. Bu, Eds., Communications in Computer and Information Science, Springer, pp. 558–570.

[6] Mingxiao, D., Xiaofeng, M., Zhe, Z., Xiangwei, W., and Qijun, C. A review on consensus algorithm of blockchain. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE, pp. 2567–2572.

[7] Neglia, G., Reina, G., Zhang, H., Towsley, D., Venkataramani, A., and Danaher, J. Availability in BitTorrent systems. In *IEEE INFOCOM 2007 - 26th IEEE International Conference on Computer Communications*, pp. 2216–2224. ISSN: 0743-166X.

[8] Pouwelse, J., Garbacki, P., Epema, D., and Sips, H. The bittorrent p2p file-sharing system: Measurements and analysis. In *Peer-to-Peer Systems IV*, M. Castro and R. van Renesse, Eds., Lecture Notes in Computer Science, Springer, pp. 205–216.

[9] Saroiu, S., Gummadi, P. K., and Gribble, S. D. Measurement study of peer-to-peer file sharing systems. In *Multimedia Computing and Networking 2002*, vol. 4673, SPIE, pp. 156–170.

[10] Sharma, P., Jindal, R., and Borah, M. D. Blockchain technology for cloud storage: A systematic literature review. 1–32.

[11] Sukhodolskiy, I., and Zapechnikov, S. A blockchain-based access control system for cloud storage. In *2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, pp. 1575–1578.

[12] Yue, D., Li, R., Zhang, Y., Tian, W., and Peng, C. Blockchain based data integrity verification in p2p cloud storage. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pp. 561–568. ISSN: 1521-9097.