Enhancing the security of gaming transactions using blockchain technology

Ciprian Paduraru
ciprian.paduraru@unibuc.ro
University of Bucharest
Romania

Cristea Rares
rares.cristea@unibuc.ro
University of Bucharest
Romania

Alin Stefanescu
alin.stefanescu@unibuc.ro
University of Bucharest
Romania

ABSTRACT

In this paper, we propose GameBlockchain, an open-source blockchain framework designed to support secure transactions of NFTs in modern computer games. Its purpose is to enable game industry stakeholders such as game developers, content creators, and regular gamers to create and exchange game assets in a more secure and trusted environment. The security of traditional databases and potential data tampering or dangerous user behavior is improved, as outlined in the paper, by blockchain technology, which is used to record critical operations in a ledger, preserving the identity of the user at all times. From a technical perspective, the main goal is to provide an architecture that is easy to use, flexible, understandable, and has an extensible SDK. Using the framework, game developers and regular users should be able to create and trade assets without third-party providers, and use all related services directly in the game interface itself, without having to switch between applications or pay additional transfer fees to providers. We also encourage the development of games with shared marketplaces and wallets on both the developer and user sides, making it easier to monetize assets and services.

KEYWORDS

blockchain, NFT, transaction, games, framework

ACM Reference Format:
Ciprian Paduraru, Cristea Rares, and Alin Stefanescu. 2022. Enhancing the security of gaming transactions using blockchain technology. In *37th IEEE/ACM International Conference on Automated Software Engineering (ASE '22), October 10–14, 2022, Rochester, MI, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3551349.3560504

1 INTRODUCTION

Video games are among the most important revenue generators in the entertainment industry [17]. A prominent feature in many games, and originating in "freemium" video games is the concept of microtransactions. According to [23], "Microtransaction commonly refers to a business model, where users can purchase virtual goods via micropayments [...] The player pays for microtransactions either directly with real world currency or with some form of fantasy virtual currency (e.g. gold). The latter is typically earned during gameplay or can (often alternatively) be purchased with real world money". It has been found that video games that use microtransactions generate a large share of revenue [20], [12]. In the literature there are certain gaps in the current infrastructure of modern games that may limit monetization opportunities. First, we found that each game has its own closed-loop microtransaction system, meaning that users cannot own and spend tokens, even between games from the same developer, let alone between different developers. We believe that these limitations stem from the fact that there is no consistent way to connect different games and developer consortia in the same market in a secure way. Inspired by the development of blockchain technologies, in this work we decided to implement an open source framework to address the observed gaps. The GameBlockchain is an open-source prototype blockchain *HF* in the rest of the paper) and available on GitHub[1]. The proposed architecture enables game industry stakeholders, such as developers, regular users-players, marketers, service providers, etc., to create, manage, and sell game assets in a secure and trusted environment. The goal of the framework is to create a platform that is easy to understand and extend by game developers or content producers, hiding the specifics of the blockchain and enabling easy deployment with minimal knowledge of security.

From a game development perspective, our framework offers the following advantages over other industry solutions:

- It allows users to trade assets directly in-game without using a third-party application that manages transactions, promoting ease of use and trustworthiness. Transaction security is achieved by authenticating users at all times in a permissionless/private blockchain solution, with critical transactions recorded in the ledger.
- It provides an open source code API specifically designed to integrate with popular game engines and AAA games and it is highly customizable. Assets that are created or transferred can be customized depending on the game or user so that they fit different types of products.
- It exhibits an event system to automatically trigger assets at runtime through generic smart contracts.
- It includes a transparent system that enables companies to work together in order to provide users with the benefits of transferring resources for regular users from one game to another. For example, a regular user can sell a player in FIFA [2] to get coins in its account balance and later buy a weapon in a shooter game produced by a different developer, such as FarCry 6 [3] without intermediates, directly from the interface of the game.

From an academic perspective, and to our understanding, this is the first work to propose, implement, and evaluate a prototype framework that can combine game development tools with permissioned blockchain technologies.

The work continues as follows. Section 2 compares our framework to other similar solutions. The architectural choices of the framework were driven by discussions with local game developers

[1] https://github.com/unibuc-cs/GameBlockchain
[2] https://www.ea.com/en-gb/games/fifa/fifa-22
[3] https://www.ubisoft.com/en-gb/game/far-cry/far-cry-6

and regular (game) users to understand their needs and get their perspective. We describe the use cases explored in section 3 in detail. A brief technical architecture of our framework is presented in section 5. Some preliminary evaluation results are shown in section 6. Finally, conclusions and ideas for future work are presented in the last section.

2 RELATED WORK

Currently, the video games that implement blockchain technologies [15] [13], are generally indie games focused on cryptocurrency exchanges and use proprietary blockchain technologies and deployments, typically implemented via web interfaces.[4]

We found *Enjin*[5] to be the most complete framework that could address the connection between blockchain services and game development. It gives developers a software development kit (SDK) to bring common blockchain concepts such as *wallet* management, *transactions*, *NFTs*, and simplified payment gateway interfaces to their games. It does, however have a number of architectural limitations. First, Enjin is a commercial tool and not open-source code, which limits the ability to extend it to the entire video game market, from AAA developers to the indie developers. Secondly, the SDK is written to provide interoperability between many generic or entertainment assets, with applications for sports, music, art, etc., therefore, the overhead per transaction is higher, e.g., only 150 users can be supported per blockchain transaction. Users' wallets are shared by these applications. Finally, the blockchain implementation in the backend is based on a public blockchain, the Ethereum mainnet [6]. While a public blockchain has its own advantages, we discuss in section 4.4 why this type of implementation hinders the need of fast responses and throughput of millions of transactions. The latency and performance limitations of a public blockchain for real-time applications are further explored by [28], [19], [21]. Our base platform proposal, Hyperledger Fabric [7], addresses all these issues by being an open-source blockchain platform, supporting customization of consensus mechanisms for higher throughput, and, via correct architectural choices, minimizing the number of blocks entering consensus by validation and ordering steps previously performed as pruning. These advantages are also mentioned in [11] and [29] and further elaborated in our work in section 4.4.

A higher-level drawback of *Enjin* for our use case is that the solution itself is a Platform-as-a-Service (PaaS). This provides little control through the SDK of the client application that the game developer ends up using. The services are all provided on the *Enjin* platform, while the application developers control a limited set of APIs for managing users' wallets and transactions. While PaaS architectures and exclusive control over APIs may be sufficient for small games, AAA raises questions for game developers about security, trustworthiness, and potential data breaches for their users or their own businesses. We believe that the slow adoption of blockchain technologies in game transactions is related to the aforementioned performance issues and PaaS services. Our framework addresses these gaps by allowing game developers to deploy their own infrastructure, control all services, make the source code visible in all layers involved, and last but not least, follow the principles from the literature that provide the required throughput.

Some other platforms used by cryptocurrency games offer partial, focused services. One of them is *Keepin App* [8], which helps manage identities of users connected through a single unified application. Public Key Infrastructure (PKI) and certificates are used to validate users and their rights, similar to our implementation in the proposed framework. *Dareplay* [9] focuses on offering cryptocurrency services for gamification applications that connect reality with digital environments. *Decentraland* [10] is a fully virtual environment where users can create their own assets and environments using cryptocurrencies. It is not a reusable framework as we are aiming for, but rather an online game built on top of the blockchain infrastructure.

3 REQUIREMENTS

In this section, we discuss the identified requirements from the perspective of both users and game developers. The use cases and the interaction between the different stakeholders are shown in Figure 1 and are discussed in more detail in the following text.

The *Marketplace* component can be viewed as an electronic storage method where sellers (owners) can post items, while buyers can obtain them by paying the amount required by the owners. Two main types of actors with different roles were defined:

- *Developers*: In the literal sense, they are the developers of a video game. Their role in our scenario is to produce content and put it on the *Marketplace*. Examples of content may vary from between video games, but usually include non-fungible items such as: in-game characters, maps, levels, missions, graphical objects, etc. Through a secure environment, the main developer organization could allow other third-party content developers to put content on the marketplace. This is desirable as it can help keep up with the demands of a particular game if the main developer does not have enough resources to keep up with user growth and its demands.
- *Users*: These are the video game users, aka gamers, that play. They interact with the marketplace to query current asset offerings and their expirations (in an auctioning context). However, what is desired and can gain more trustworthiness by using the blockchain infrastructure is the direct trading of assets. This could simplify the process of buying assets (as opposed to using the marketplace as an intermediary) and encourage social communication within the game itself.

Additionally other types of roles might emerge, such as those that perform data analytics and then provide recommendation systems to developers ([27], [24]), but in its current version, the framework focuses mainly on solving the requirements for securely buying and selling assets in a game environment.

Since the amount of data of each asset can be very large, the InterPlanetary FileSystem (IPFS) framework [25] is used to store the content itself, while assets transferred between participants or the marketplace itself use hashes of the data and address. We further detail this mechanism in the 5 section.

[4] https://dappradar.com
[5] https://enjin.io
[6] https://ethereum.org/en/enterprise
[7] https://www.hyperledger.org/use/fabric
[8] https://www.keepin.biz
[9] https://dareplay.io
[10] https://decentraland.org

We have identified five main use cases, shown in Figure 1.

- **U1**. Developers can push content to the marketplace in the form of *assets* Eq. 1.

  $$Asset = (AssetID, Data, AppraisedValue, OwnerID). \quad (1)$$

  The *Data* parameter is a cryptographic hash of the asset's properties and concrete data using a SHA256 function, but other similar functions may be used depending on the needs and trade-offs considered [2]. The *AppreisedValue* is the owner's valued price (in coins), while the *OwnerID* is the owner's unique identifier as specified in their identity certificate (see section 5.1 for more details).
  Developers are assumed to manage their own wallets and account balance infrastructures without relying on third-party providers. This will not only increase the trustworthiness of the platform, but also reduce the cost of transaction fees by eliminating the need for intermediaries. The virtual coins issued by developers are purchased in bulk by regular users as packages paid for with real-world money (e.g., cash, bitcoin, etc.). Also, it is a common trend among developers to give (private) discounts to some members on the products they offer for various reasons. One example could be the automatic detection and prevention of users from leaving the gaming platform [4].
- **U2** Gamers are able to buy assets on the marketplace by transferring the requested coin amount from their personal wallet to the owner of the asset. The marketplace component can be considered as a matchmaking component that brings sellers and buyers together.
- **U3** Gamers are able to make transactions directly between each other, using the market as an intermediary. Such cases occur in real games as players are often involved in chat conversations, can see what other players own during the game, etc.
- **U4** Enabling an auction house system, where the user is able to ask for products or places a bid, and the bids must be matched by the owners of the goods.
- **U5** Statistics gathering and data mining component can be viewed as a service involving both users and developers. Extracting patterns from this data can improve both game quality and monetization over time by monitoring the needs of gamers.

4 BLOCKCHAIN INTRODUCTION

This section briefly describes the blockchain space, for more details we recommend [31], [1] for a further understanding of applications of BC in this area.

4.1 Basics

A *blockchain* is an immutable ledger of transactions maintained on a distributed network of peer nodes. This ledger is replicated and kept in sync among a group of peers using a consensus protocol. Transactions in the ledger are grouped into linked lists of blocks, with each node (block) containing a hash that binds it to the previous block. Probably the best-known blockchain application is the cryptocurrency Bitcoin. Another well-known alternative is Ethereum [7], which also introduces the concept of smart contracts to create a platform for distributed applications. The class of blockchain that Bitcoin and Ethereum belong to is known as *public permissionless blockchain* technology. These public networks allow participants to interact without disclosing their identity.

4.2 Motivation for using blockchain in the gaming industry

Based on our observations and discussions with industry partners, the implementation of trading mechanisms in games currently relies mainly on centralized architectures and gateways. Centralized architectures that use gateways are known to be vulnerable to data forgery, tampering, denial of services, and other common attacks. Cases such as *Nintendo*, *Gigaleak*[11] and Microsoft Xbox games data leak[12] are clear examples of a vulnerability generated by this architecture. To adapt the security level concerns to our use-case of in-game transactions, we identified the following requirements:

- Confidentiality of user data: private data should be accessible only to those who need it and transmitted in encrypted form; the literature describes in detail the benefits of blockchains for identity protection [8], [30], [31], [1].
- Data integrity: tampering with data processed in game transactions could lead to data leaks throughout the distributed system.
- Transaction balance security: ensure that parties correctly agree on a transaction price and the balance account is correctly updated.

Blockchain technologies can help in this regard, as shown in the literature and its use cases in the enterprise space, since operations performed on the network are written to ledgers and agreed upon by multiple peers. This simultaneously makes it harder to falsify data and easier to track and automatically detect data leaks and their source.

4.3 Public vs permissionless blockchain

Recently, blockchain has attracted several enterprise sectors, such as finance [16], healthcare [3], or supply chain [26]. However, one of the requirements in enterprise use cases, from both a performance and security perspective, tends to target the need for a permissioned blockchain rather than a public, anonymous blockchain. In many cases, such as finance, participant identity is a hard requirement. In finance, for example, transactions must comply with certain regulations such as anti-money laundering (AML) and know-your-customer (KYC). This is also very close to our studied use case, gambling trading, as users need to be identified and authenticated for security reasons and their trading behavior needs to be tracked. This is necessary to ensure that no content counterfeiting is possible.

4.4 Motivation for Hyperledger Fabric

Our work is based on the Hyperledger Fabric (HF) solution [6]. HF is an open-source, enterprise-grade distributed ledger technology

[11] https://en.wikipedia.org/wiki/2020\T1\textendash21_Nintendo_data_leak
[12] https://comicbook.com/gaming/news/xbox-microsoft-console-data-leak

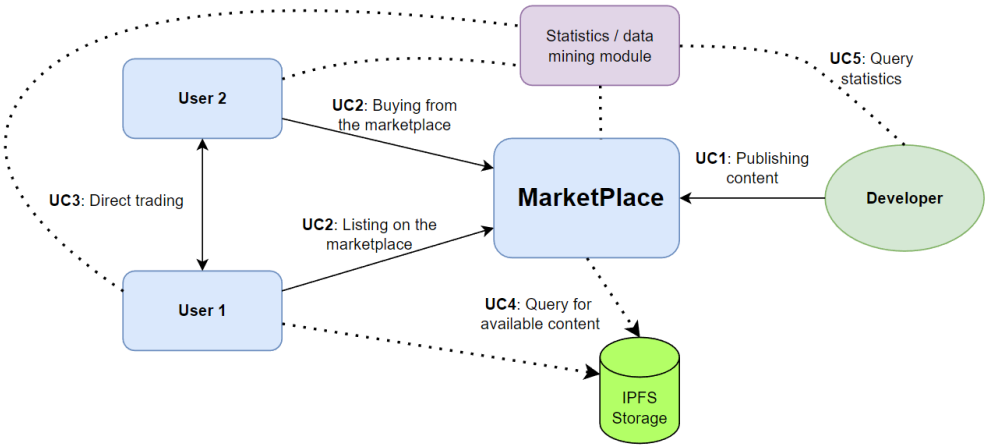

Figure 1: Use cases have been established for managing assets from publication to trading, either directly or through a marketplace interface. The IPFS is used as external storage to shop the storage requirements of the transactions. The data mining module helps developers better understand progress and needs and respond accordingly.

platform with several features such as high modularity and configurability. Application developers can write smart contracts to describe automated business logic.

Other well-known options for implementing a permissioned ledger that we have explored include Ethereum [5] and Corda R3 [19]. For our use cases, we chose HF because we were interested in its features. The most important decision was related to the pluggable consensus protocol feature. This allows the consensus algorithm of HF and the list of participating peers to be customized and executed faster than Ethereum's Proof of Work (PoW) mechanism, which is known to be impractical for applications that require fast real-time response, such as games. [28] [19] [21]

As mentioned in [11], [29], the performance of HF is mainly achieved by using different peers of the network to maintain the state of the ledger, where the process of verification and approval is divided into different steps: (a) endorsing phase - simulates and receives transactions, approves or denies them, (b) ordering phase - orders the agreed transactions into blocks, (c) committing phase - peers agree to the new blocks with fast consensus mechanisms and keep their own synchronized state of the ledger up to date.

Other features such as fine-grained access control, i.e., participants can be restricted via policies and channels for reading, creating or updating data, or the fact that Corda is primarily used in the financial sector contribute to the choice of the BC solution. In terms of the requirements of the General Data Protection Regulation (GDPR), which is a notoriously difficult issue with blockchain solutions in general, HF has an important advantage over the others in that it allows users to pseudo-delete their records from the ledger, i.e., its API allows programmers to mark the ledger's data as deleted so that queries can no longer retrieve it.

There are also drawbacks that are typical of systems that are intended to be as adaptable as possible: It increases implementation costs for developers and operators to connect and define the network.

4.5 Tokens

Currently, there are two categories of tokens in the blockchain space: (a) fungible tokens - they typically represent a currency (such as USD, Euro or ETH), and (b) non-fungible tokens (NFTs) - they are used to represent physical items, digital content, real estate, etc.

For trading items in games, NFTs are the most appropriate token category. The process of *minting*, in short, means creating a unique NFT with a specific owner and content properties. These tokens can then be *transferred* between owners through a transaction agreed upon by both parties. Both processes are recorded as operations in the blockchain ledger. This makes tracking the history of the owners and the transactions performed over each NFT more trustworthy and easier, as one can automate this verification process. This is in contrast to a typical database where only the latest state of the elements would be available.

Throughout the rest of this paper, we replace the term NFT with *Asset*, which is more common in gaming industry terminology, and in line with HF's syntax.

5 FRAMEWORK TECHNICAL ARCHITECTURE

This section presents how the GameBlockchain framework solves the requirements discussed in Section 3 at different levels.

5.1 Network setup and identity management

Our framework uses *Public Key Infrastructure (PKI)* [10], [18] to ensure secure communication on the network. In blockchain terminology, an organization consists of multiple users with different access rights controlled by attributes and/or *Access Control Lists (ACL)*, computing nodes, and a defined *Certificate Authority (CA)* that assigns identities to each participant in the network. In order to participate in transactions, network nodes, administrators, and users, either from the developer side or regular users, must have: (a) a public certificate and (b) a private key to verify their identity.

In our framework and from a game perspective, an organization is defined as the main developer of a game product. Thus, it has full administrative rights over the network and the participants. At the next level, there are other organizations that are considered as third-party content providers for the game product in the deployment phase. Figure 2. In our architecture, the main developer has the rights to the *CA* text type and *ACL*. Any other participant, whether a third- party developer or a user, must have access to these in order to obtain an identity.

To create a marketplace between different game products, developers, and regular users, the framework supports the idea of using a consortium of organizations instead of a single one. For example, users can sell an asset in a game G_1 from developer $D1$ and use the tokens received in another game G_2 from the same developer or in a game G_3 from another developer D_4. The certificates issued by CA are encapsulated in a *X.590* [22] digital certificate, along with the attributes that determine access rights to network resources. Eq. 2 describes the high-level data that each participant of the network has for identification.

$$\begin{aligned} clientIdentity = \{&certificate - C; \\ &unique\ id - clientID \\ &public\ and\ private\ key - PU, PR; \\ &attributesset - ATTR\} \end{aligned} \tag{2}$$

Each participant has a certificate, a private key, a public key, a client ID, issued by the configured *Membership Service Provider (MSP)* of the underlying *HF* framework, an attribute associated with the client identity that defines its roles, permissions, and custom fields. The custom fields can be used, for example, to define what type of subscription to the game a particular user has (e.g., premium, free, etc.). Thus, in practice, those who have a premium subscription could get more benefits and discounts.

In our architecture, a separate instance of *ledger* is created for each individual *communication channel*, as shown in Figure 2. The main reason for the split is performance, as transactions can be verified faster and large queries can be executed in less time. A second reason is the ability to track and collect data from different locations depending on interest, rather than using a completely monolithic system.

There are two categories of channels, and their respective ledger is capable of recording all transactions that take place within that instance of the channel:

- Regular users to deployed game products: They are used to interact from the user organization to the games deployed in a public way using a component called *Marketplace*, but also between regular users directly when they play in the same game. Private transactions use private data in the ledger and a private communication channel, as described in the text below in this section.
- Main and 3rd party developers to game products: this is used to push/retire content to the deployed games.

Assets in real-world games are nowadays often sold via *auction* models, where players bid privately for products. To cover these aspects, our framework provides support for *HF*'s private data API. When a transaction is created at the blockchain level, smart contracts have access to a *transient* map that is added to the channel ledger, but is only visible to a limited group of members. The same mechanism serves the use case of developers offering promotions to some premium members, for example. The private data may be needed in this case to hide the discounts offered to them.

5.2 Wallets and Tokens

Each participant's wallet is a tuple as defined in equation 5.2. The *ClientID* is the one given by his public key, *AssetsOwned* is the list of assets he owns, and finally *AccountBalance* defines the number of virtual coins the participant currently owns. Assets, which are represented by NFTs in blockchain terminology, are implemented in our framework according to the *Erc-721* [9] standard. Tokens are implemented using key-value pairs, where the key is of the form *assetPrefix.ownerID.assetID*. Account balance is implemented in a similar manner, where the key in each case is of the form *balance.ClientID*. These keys are used as indexes for the CouchDB implementation to quickly track the history of transactions for a particular asset, find a user's account balance, or populate the marketplace dashboard (5.5 section).

$$W = \{ClientID, AssetsOwned, AccountBalance\}.$$

The difference between participants behavior is that only regular users are supposed to buy assets and virtual coins from the main developer.

5.3 Storage

The role of the IPFS system (Figure 1) is to allow the ledger to only store cryptographic hashes of data addresses that may physically exist either in local storage for critical operations or in the storage of cloud/self-development servers. This is done in order to speed up the operations of creating, validating, and querying transactions in the blockchain. Due to security concerns, it is usually decided on a case-by-case basis if big data should be stored in the cloud or in the developer's own infrastructure.

5.4 Assets transactions

In our proposed framework, asset transactions can take place between two entities, *Seller* and *Buyer*. The *Seller* can be either a developer or an ordinary user. The *Buyer* is always a regular user. Figure 3 shows a sequence of steps behind the framework implemention to transfer an item between the two entities. It is important to understand that each transaction is recorded in the general ledger. This increases the overall trustworthiness of the system because, compared to a typical database where only the last state of the system is shown, recording operations in a distributed peer ledger can solve problems that are reported in a correct way by tracing the operations performed. The hashing operation over the properties and content of the data, first transferred by the seller in step 1, observed by the buyer in steps 2 and 3, and finally verified by the backend in step 4, ensures that the integrity of the transferred asset is maintained between the point of offer and the point of sale. In the end, both receive a receipt of the transaction. The ledger and backend work as observers, responding to events of agreements sent by customers rather than pooling. This is done for efficiency reasons. The backend computations required to endorse and validate

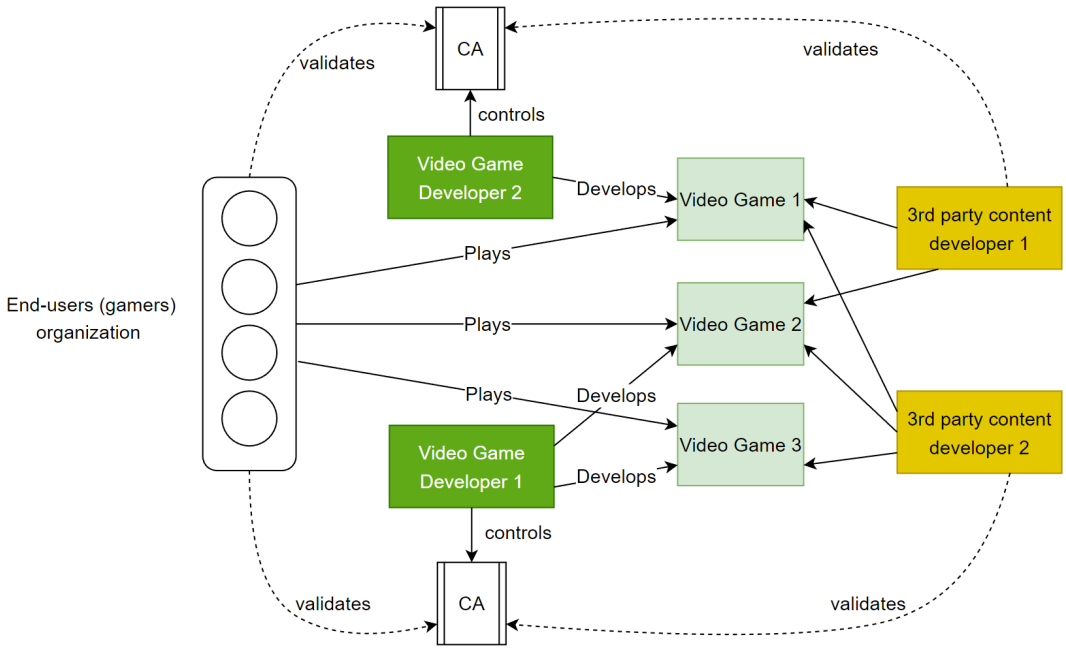

Figure 2: The figure shows an example of network deployment in a consortium of two main developers, Video Game Dev 1 and Video Game Dev 2, and two third-party developers. VG Dev 1 produces two games, VG2 and VG3, while VG Dev 2 produces only VG1. 3rd party Dev 2 produces content for all three game products in the network, while 3rd party Dev 1 produces content for VG1 and VG2. Each main developer has its own certification authority (CA) that registers user identities for both third-party developers and user groups (organized as a single fixed organization) to use the products they provide. To participate in a game and trade items and coins, each user must register with the main developer CA.

blockchain transactions and compile data into blocks are performed by developer peer nodes.

5.5 Marketplace

.

The marketplace component in Figure 1 acts as a dashboard that can be accessed through the framework's user interface and API functions. Each marketplace M is specific to the game product that G_M is deployed on. On the backend, the API queries content from the channels' ledger between the *UserOrg* and G_M and those of developers and G_M and creates a union of saleable assets and their properties. Using the underlying HF API and our implementation over the *CouchDB* infrastructure, our framework is able to leverage rich and performant queries. Behind the scenes, *pagination* and indexing of common properties such as key-value pairs are built by *(assetid, properties)* to support fast queries.

6 EVALUATION

If on the security side we rely on blockchain literature to prove its correctness in synchronizing data and peer agreements, on the performance side, which is a key factor in game development, we conducted synthetic tests in the open-source game engine Unreal Engine 5 [13]. A publicly available demo called *ShooterGame* [14] was used for testing. During the tests, our framework was used as a Flask server[15], with the client game REST API sending requests for various high-level operations, such as: (a) creating an asset at runtime, (b) retrieving the list of saleable items with various properties for the marketplace component, (c) transferring items between two owners. As shown in Figure 4, a request is parsed and sent to the corresponding smart contracts written in *Go* [16] language for further processing. The implementation is responsible for validating the requests, recording the required operations over the ledger, returning the query results, and acting as an observer and event system, as shown in the transaction example in Figure 3.

To synthetically test the above three processes, our test setup ran processes on an Intel Xeon server with 20 cores and 2 threads each. It is assumed that the number of RAM or the hard disk capacity and speed are not important for the test, i.e., do not represent a bottleneck. In the HF deployment, there is one process for ordering peers, one for endorsement, and one for committing. A single process was used to analyze requests, with each request containing less than 1KB of data (note that the IPFS protocol is used for large datasets, so this is not a bottleneck nor does it affect the simulation results if the datasets are ever larger). The benchmark simulated 32 client games sending different requests to the infrastructure simultaneously. For each of the three operations, the results of an average of 100 runs are recorded for the following metrics: (a) maximum latency of an

[13] https://www.unrealengine.com
[14] https://docs.unrealengine.com/4.27/en-US/Resources/SampleGames/ShooterGame
[15] https://flask.palletsprojects.com/en/2.1.x/
[16] https://go.dev

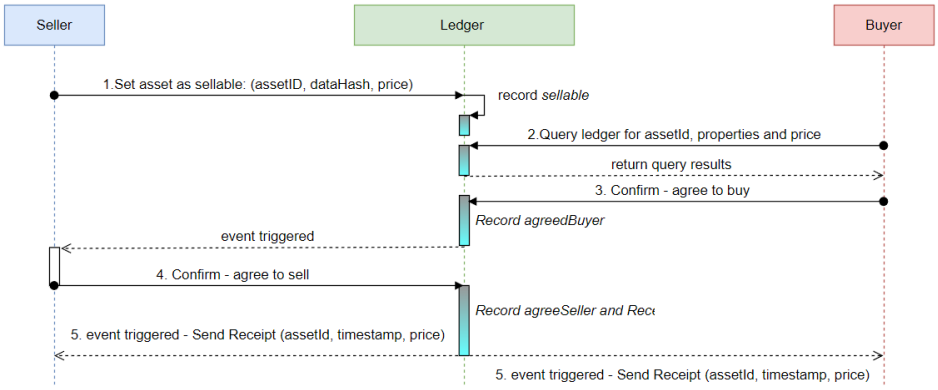

Figure 3: A sequence diagram showing a use case for the successful trading of an asset between a seller and a buyer. The protocol ensures that the agreed price and the characteristics of the data to be transferred are not changed between the offer to sell and the end of the transaction. Note that all transactions are written to the general ledger so that reported issues can be automatically tracked.

.

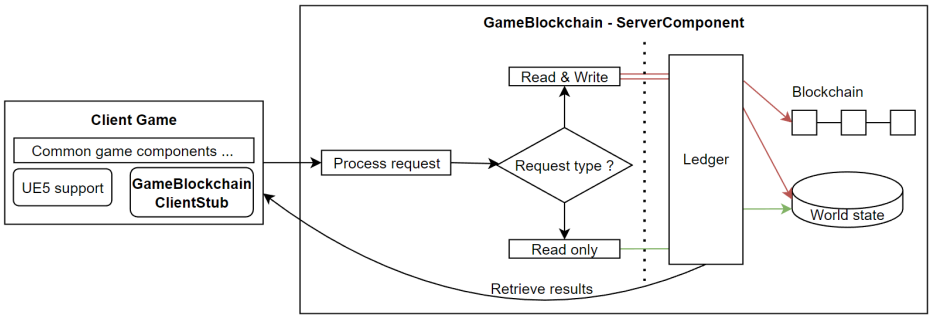

Figure 4: The flow of message requests for assets creating, trading, and queries needed for marketplace component. In Hyperledger Fabric, the ledger component is stored as both blockchain and world state, which is composed of key-value pairs, so read-only operations (flow of the green arrow) that query only the current values do not require blockchain evaluation at all. Note that this is a generic representation of queries for a single ledger, but there is one ledger instance for each communication channel in the network shown in Figure 2.

.

operation, (b) minimum latency, (c) transactions per second (TPS) throughput. The TPS is determined by flooding the server component with fictitious requests from clients at maximum sending capacity. This is to determine how many requests the server can handle in each transaction. Pagination, caching, and indexing are used to set up CouchDB (on each HF network peer), as described in section 5. The results are shown in Table 1. Based on the observed latency metrics, the results can confirm that the method is suitable for real-time deployment by using a separate background thread in the client application that performs the operations and returns the results. It is also important to note that the results are preserved even when the number of clients increases, as demonstrated in the HF [29], [11] studies, if the number of processing nodes and peers used is increased proportionally at the same time.

Test type	Max latency (s)	Avg latency (s)	TPS
Create assets	0.53	0.29	1781
Marketplace query (b=50)	3.4	2.1	90
Marketplace query (b=100)	5.1	3.4	78
Trade	1.03	0.57	1233

Table 1: Table showing benchmark results for the three metrics: average and maximum latencies, throughput transactions per second. The query tests were performed using two batch sizes, to show that, as expected, batching more requests can increase the throughput, but also the latency is increased.

7 ECONOMIC CHALLENGE

The presented framework is not just theoretical, but it is accompanied by a proof of concept, backing the proposed architecture. A

challenge out of the scope of this paper, but nonetheless critical, is the economical opportunity for the developers to implement a shared market system. Currently, microtransactions account for a big part of the revenue stream of the developers and publishers. [17] Adhering to a shared network such as the one this study proposes, will enable the gamer to engage more freely in the economics of video games, and open up possibilities of exploitation. A comparable use case can be observed in the Steam Marketplace, which enables the developers to use the Marketplace to buy and sell assets, but only few developers engage this feature. Enabling a framework such as the one described in this study would limit predatory monetization practices [14].

8 CONCLUSION AND FUTURE WORK

In this paper, we present GameBlockchain, a framework that aims to link game development with blockchain technologies in order to enable a more secure and trustworthy way of transacting assets between content developers and users. Architectural decisions were made by observing the gaps in current tools and after discussions with both stakeholders. We tested usability and performance using a publicly available demo in Unreal Engine. However, from a higher-level perspective, the framework still needs to be adapted and evaluated in real game products. Our plans for the future include developing the framework further as a plugin interface to increase the chance that it will be adopted by games developed on at least publicly available game engines. From a technical point of view, another problem is to remove from the chain some of the data representing the state of the game or users that can only be read.

ACKNOWLEDGMENTS

This research was supported by the European Regional Development Fund, Competitiveness Operational Program 2014-2020 through project IDBC (code SMIS 2014+: 121512). We would also like to thank our game development industry partners from Amber, Ubisoft, and Electronic Arts for their support.

REFERENCES

[1] Abdelzahir Abdelmaboud et al. 2022. Blockchain for IoT Applications: Taxonomy, Platforms, Recent Advances, Challenges and Future Research Directions. *Electronics* 11, 4 (2022). https://www.mdpi.com/2079-9292/11/4/630 Number: 4 Publisher: Multidisciplinary Digital Publishing Institute.

[2] Aishah Alfrhan et al. 2021. Comparative study on hash functions for lightweight blockchain in Internet of Things (IoT). *Blockchain: Research and Applications* 2, 4 (Dec. 2021). https://www.sciencedirect.com/science/article/pii/S2096720921000312

[3] McSeth Antwi et al. 2021. The case of HyperLedger Fabric as a blockchain solution for healthcare applications. *Blockchain: Research and Applications* 2, 1 (March 2021). https://www.sciencedirect.com/science/article/pii/S2096720921000075

[4] Kelly Bergstrom. 2019. Temporary Break or Permanent Departure? Rethinking What It Means to Quit EVE Online. *Games and Culture* 14, 3 (May 2019), 276–296. Publisher: SAGE Publications.

[5] Vitalik Buterin. 2013. Ethereum White Paper: A Next Generation Smart Contract & Decentralized Application Platform. *White-paper* 1, 1 (2013). https://github.com/ethereum/wiki/wiki/White-Paper

[6] Christian Cachin et al. 2016. Architecture of the hyperledger blockchain fabric. In *Workshop on distributed cryptocurrencies and consensus ledgers*, Vol. 310. Chicago, IL, 1–4.

[7] Praveen M Dhulavvagol, Vijayakumar H Bhajantri, and S G Totad. 2020. Blockchain Ethereum Clients Performance Analysis Considering E-Voting Application. *Procedia Computer Science* 167 (Jan. 2020), 2506–2515. https://www.sciencedirect.com/science/article/pii/S1877050920307699

[8] Maya Dotan, Yvonne-Anne Pignolet, Stefan Schmid, Saar Tochner, and Aviv Zohar. 2021. Survey on Blockchain Networking: Context, State-of-the-Art, Challenges. *Comput. Surveys* 54, 5 (May 2021).

[9] William Entriken, Dieter Shirley, Jacob Evans, and Nastassia Sachs. 2018. EIP-721: Non-Fungible Token Standard. https://eips.ethereum.org/EIPS/eip-721

[10] Abba Garba et al. 2020. BB-PKI: Blockchain-Based Public Key Infrastructure Certificate Management. In *2020 IEEE 22nd International Conference on High Performance Computing and Communications; IEEE 18th International Conference on Smart City; IEEE 6th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 824–829.

[11] Christian Gorenflo, Stephen Lee, Lukasz Golab, and Srinivasan Keshav. 2020. FastFabric: Scaling hyperledger fabric to 20 000 transactions per second. *International Journal of Network Management* 30, 5.

[12] Karin A. Haberlin and David J. Atkin. 2022. Mobile gaming and Internet addiction: When is playing no longer just fun and games? *Computers in Human Behavior* 126 (Jan. 2022).

[13] Hyun-joo Jeon, Ho-chang Youn, Sang-mi Ko, and Tae-heon Kim. 2021. *Blockchain and AI Meet in the Metaverse*. Vol. 73. IntechOpen. https://www.intechopen.com/chapters/undefined/state.item.id

[14] Daniel L King and Paul H Delfabbro. 2019. Video game monetization (eg,'loot boxes'): A blueprint for practical social responsibility measures. *International Journal of Mental Health and Addiction* 17, 1 (2019), 166–179.

[15] Cosimo Laneve and Igor Ershov. 2019. Blockchain gaming: An analysis of the use of blockchain technology in the video gaming industry. *Alma Mater Studiorum · Universita di Bologna* (2019).

[16] Chaoqun Ma, Xiaolin Kong, Qiujun Lan, and Zhongding Zhou. 2019. The privacy protection mechanism of Hyperledger Fabric and its application in supply chain finance. *Cybersecurity* 2, 1 (Jan. 2019).

[17] Jesús Manuel Palma-Ruiz et al. 2022. An overview of the gaming industry across nations: using analytics with power BI to forecast and identify key influencers. *Heliyon* 8, 2 (Feb. 2022).

[18] Paul Plessing and Olamide Omolola. 2022. Revisiting Privacy-aware Blockchain Public Key Infrastructure. 415–423.

[19] Julien Polge, Jérémy Robert, and Yves Le Traon. 2021. Permissioned blockchain frameworks in the industry: A comparison. *ICT Express* 7, 2 (June 2021), 229–233. https://www.sciencedirect.com/science/article/pii/S2405959520301909

[20] Alia Reza, Sabrina Chu, Adanna Nedd, and Daniel Gardner. 2022. Having skin in the game: How players purchase representation in games. *Convergence* (May 2022). Publisher: SAGE Publications Ltd.

[21] Krishnapriya S and Greeshma Sarath. 2020. Securing Land Registration using Blockchain. *Procedia Computer Science* 171 (Jan. 2020), 1708–1715. https://www.sciencedirect.com/science/article/pii/S1877050920311649

[22] Stefan Santesson et al. 2013. X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. RFC 6960 (June 2013). https://datatracker.ietf.org/doc/rfc6960

[23] Sebastian Schwiddessen and Philipp Karius. 2018. Watch your loot boxes!–recent developments and legal assessment in selected key jurisdictions from a gambling law perspective. *Interactive Entertainment Law Review* 1, 1 (2018), 17–43.

[24] Matt Stanton, Ben Humberston, Brandon Kase, James F. O'Brien, Kayvon Fatahalian, and Adrien Treuille. 2014. Self-refining games using player analytics. *ACM Trans. Graph.* 33, 4 (2014), 73:1–73:9.

[25] Mathis Steichen, Beltran Fiz, Robert Norvill, Wazen Shbair, and Radu State. 2018. Blockchain-Based, Decentralized Access Control for IPFS. In *2018 IEEE International Conference on Internet of Things*. 1499–1506.

[26] Ciza Thomas, Paula Fraga-Lamas, and Tiago M. Fernández-Caramés. 2020. Computer Security Threats. IntechOpen London. Google-Books-ID: CJYtEAAAQBAJ.

[27] Ahmed Tlili and Maiga Chang (Eds.). 2019. *Data Analytics Approaches in Educational Games and Gamification Systems*. Springer. https://link.springer.com/book/10.1007/978-981-32-9335-9

[28] Xiwei Xu et al. 2017. A Taxonomy of Blockchain-Based Systems for Architecture Design. In *2017 IEEE International Conference on Software Architecture (ICSA)*. 243–252.

[29] Xiaoqiong Xu et al. 2021. Latency performance modeling and analysis for hyperledger fabric blockchain network. *Information Processing & Management* 58, 1 (Jan. 2021). https://www.sciencedirect.com/science/article/pii/S0306457320309298

[30] Rui Zhang, Rui Xue, and Ling Liu. 2019. Security and Privacy on Blockchain. *Comput. Surveys* 52, 3 (July 2019), 51:1–51:34.

[31] Yijun Zou, Ting Meng, Peng Zhang, Wenzhen Zhang, and Huiyang Li. 2020. Focus on Blockchain: A Comprehensive Survey on Academic and Application. *IEEE Access* 8 (2020), 182–201. Conference Name: IEEE Access.

[17] https://www.gamedeveloper.com/business/microtransactions-now-account-for-nearly-half-of-take-two-s-revenue