R?ddle: A Fully Decentralized Mobile Game for Fun and Profit

Athanasia Maria Papathanasiou(✉), Chalima Dimitra Nassar Kyriakidou, Iakovos Pittaras, and George C. Polyzos

Mobile Multimedia Laboratory, Department of Informatics, School of Information Sciences and Technology, Athens University of Economics and Business, Athens, Greece

papathanasiou@aueb.gr, nassarkyriakidou@aueb.gr, pittaras@aueb.gr, polyzos@aueb.gr

Abstract. We explore the adoption of Web 3.0 technologies in gaming, focusing on creating new business models for novel decentralized games that attract the attention of all actors involved: players, designers, et al. In particular, we leverage advances in Distributed Ledger Technologies (DLTs) and the InterPlanetary File System (IPFS) to build a flexible, transparent, and fully decentralized mobile game, called R?ddle, in which players, potentially around the world, compete in posing and solving mathematical riddles, designed by a gaming company or anyone, in order to get rewarded accordingly. Furthermore, our presented game supports a fully decentralized version of the Play-to-Earn (P2E) gaming model that allows players to earn money, while they are playing and enjoying the game. Finally, with our proposed architecture, we manage to achieve gas and cost minimization on Ethereum, by using the IPFS effectively.

Keywords: InterPlanetary File System (IPFS) · Distributed Ledger Technologies (DLTs) · Blockchains · Smart contracts · Ethereum · Mobile gaming · Crypto-games · Play-to-Earn (P2E) · Web 3.0

1 Introduction

The growing popularity of Distributed Ledger Technologies (DLTs) has made them one of the most used technologies for several applications in any domain, such as in energy sector, Internet of Things (IoT), supply chains, etc., as they come with many benefits and intriguing properties. One of these domains that can be significantly benefited is the mobile gaming industry. Indeed, many gaming companies have already utilized the blockchain technology within gaming and they have developed and published a variety of games [6]. In addition to these games, the use of blockchains within gaming has brought new business opportunities in the gaming industry. In particular, a new gaming model has been enabled, called Play-to-Earn (P2E), which allows players to generate revenues by just playing games. With this new model, blockchain-based games have attracted a lot of attention (more than $800,000$ unique users) [2].

In a previous work by some of the authors [7], it was studied how blockchains can be utilized in mobile gaming ecosystems, and the benefits they provide were demonstrated, i.e., *decentralization, immutability, availability, transparency, and auditability*, among others. In [3], a fully decentralized system was envisioned that uses the InterPlanetary File System (IPFS) [1]. The presented system enables games having evolvable trading assets, represented as Non-Fungible Tokens (NFTs), completely owned by the users.

In this paper, we use the IPFS and the Ethereum blockchain [10] in a different way. In particular, we present *R?ddle*, a novel, decentralized mobile game. In this game, players solve mathematical riddles at three levels of difficulty; easy, intermediate, and advanced. The riddles are set by the gaming company. However, players can add their own riddles too. Furthermore, players can use hints to solve the riddles, by paying the defined amount of money, in cryptocurrency. A ranking system is used to keep track of the top players, based on their in-game level, in order to get rewarded. Players level up by solving riddles. Finally, players can also get rewards by adding their own riddles in the game.

To develop a proof of concept implementation of the aforementioned game, we leverage the Ethereum blockchain and its support for smart contracts to build a fair, trusted baseline system for the game. Furthermore, we use the IPFS to reliably host, in a decentralized manner, the riddles and the corresponding hints of the game. In this paper, we make the following contributions:

- We design and implement a fully decentralized and transparent baseline system for mobile games. We store the riddles and the hints of the game on IPFS, which provides tamper-proof, decentralized storage, to minimize cost, respecting decentralization.
- We guarantee that the rules of the game (e.g., the ranking system, etc.) will be respected, by using the Ethereum blockchain and smart contracts.
- We enable the P2E gaming model, by allowing users to get rewarded either by playing the game, or by adding their own riddles, and all this in a decentralized manner.
- We achieve gas minimization by using effectively the IPFS.

The remainder of this paper is organized as follows: In Sect. 2, we present the related work in the area. In Sect. 3, we present our proposed system design, as well as, its implementation. In Sect. 4, we quantitatively evaluate our system, while in Sect. 5, we present and discuss its advantages. Finally, in Sect. 6, we conclude the paper.

2 Related Work

We first introduce some typical examples of popular blockchain-based games. A simple one is BetDice, which runs on the EOS platform. In this game, each player selects a number between 1 and 100 and then rolls a dice. In case the target number is less than the number of the dice, the player wins. Due to the fact that it runs on the EOS chain, it does not introduce any gas fee or transaction delay and is therefore preferred from casinos on the Ethereum platform. Another game, which introduced the topic of asset ownership is CryptoKitties [8]. This game uses ERC-721 tokens [9] (kitties), which can be collected and traded among players. It uses Ethereum smart contracts and has a web application interface, so that players can interact easily with Ethereum. Another notable reference is Gods Unchained, a card game, where players purchase cards provided by the game operators, with the use of smart contracts, in limited quantity.

In addition, it is worth mentioning a few of the most important blockchain-based games that encorporate the P2E gaming model, as introduced in [4]. These games take place in Metaverses, which are virtual environments, where users create avatars to duplicate their physical-world experiences on virtual platforms. They also use NFTs, which represent in-game property. One of these games is Decentraland (MANA), which is a virtual game platform that runs on the Ethereum blockchain. Through its features, it allows users to create and monetize their content and property in the game. Another interesting reference is Axie Infinity (AXS) [5]. This game has gained general recognition in the industry. This is due to its seamless and dynamic P2E features. The goal of many players is to earn money, while they are having an entirely new level of enjoyment. Last but not least, one of the most played decentralized P2E games is the SandBox (SAND). This game allows users to create their own in-game avatars, as well as trade and sell content and property on NFT LAND. Like the aforementioned games, SandBox allows anybody to create, share, and monetize their in-game assets.

Thus, we observe from the numerous use cases and applications that blockchain technology can strengthen the gaming industry offering a variety of new interesting properties and features, such as transparency, immutability, availability, etc. Furthermore, using IPFS, we can create fully decentralized systems and games that do not have any trust relationships with centralized third-party servers.

3 System Overview and Implementation

In this section, we present an overview of our solution and its architecture, which is illustrated in Fig. 1. In our system, there is a *player* owning a mobile device, which contains our (Android) application and an Ethereum wallet (a public/private key pair) used for signing transactions sent to the blockchain, a *game administrator* having an Ethereum wallet, who creates the smart contracts, deploys them on the blockchain, and uploads the riddles on IPFS. Finally, there is the Ethereum infrastructure and the corresponding smart contracts, and the IPFS, where the riddles and the corresponding hints are stored.

In our system, there are two smart contracts, the `PlayerContract` and the `RiddleContract`. The PlayerContract implements all the actions for creating

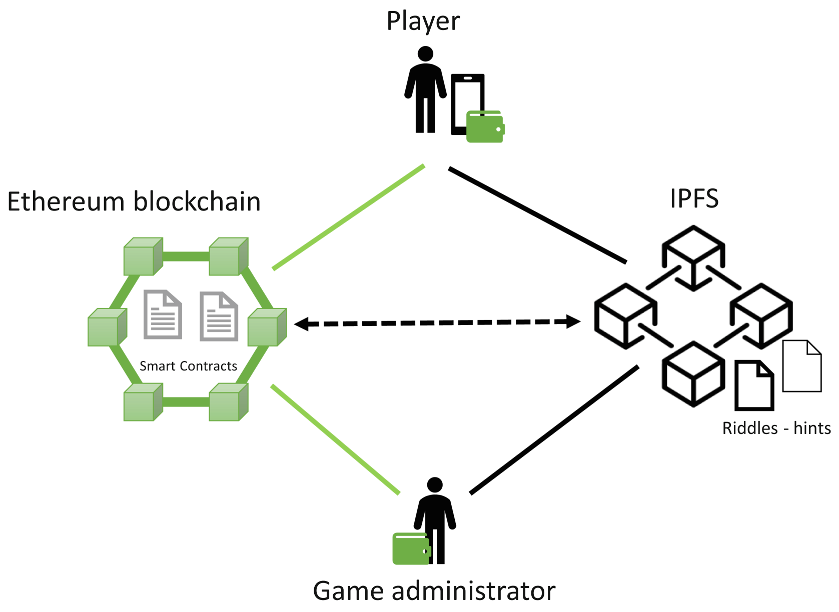

Fig. 1. An overview of the system's architecture.

and managing the players. Namely, the creation of an avatar, the retrieval of its attributes, and the modifier that correlates the avatar's ID with the user's account. Furthermore, it contains functions that check whether a user is eligible to level up, as well as, if she can use a hint. We should note here that it is essential to assure that some actions regarding a player, are only performed by the owner of the avatar. Due to blockchain's transparency every player in the game can view the functions inside the contract. Therefore, we do not want another player, except the owner of the avatar, to send transactions in order to level up, solve a riddle, or get a hint. Due to that fact, a modifier is used to ensure that only the owner of an avatar can call functions regarding their avatar. PlayerContract also includes functions such as play, solve, and reward, which are invoked, when a player tries to solve a riddle. Play function is called when a user chooses a riddle to solve. This function asks the player to pay a defined amount of money, which is transferred on the smart contract's address. Solve function checks whether the player should level up, while reward function shares to players the amount of ethers stored in the smart contract's address.

The latter contract is responsible for managing the riddles. It has one function, which is used for the creation of a riddle and is responsible for assigning each riddle to a specific level of difficulty; easy, medium, and hard. Both of the smart contracts implement appropriate checks to allow only the game administrator to perform administrative tasks.

The game administrator is responsible for adding and "pinning" the riddles of the game and the corresponding hints on IPFS. It is crucial to pin the content after adding it on IPFS, in order to guarantee that it will be live for long term and that it will not be garbage collected. When she uploads a riddle on the

IPFS, a CID is produced (in the form of the hash SHA-256), which is then used for fetching the uploaded riddle. These CIDs are stored on the blockchain, and in particular on the RiddleContract. The CIDs are used in the application, in order to extract the riddles from IPFS and have them appear on the user's mobile devices. The CIDs of the hints are not published anywhere, so the users cannot find them easily on IPFS. However, a malicious user can find them by observing what the node of the gaming company on IPFS uploads. This can be addressed by making the node of the gaming company changing its peer ID regularly. On the other hand, the answers of the riddles are stored within the application, so as to ensure that nobody will find them on IPFS. But, if the mobile phone is rooted, then a malicious user will be able to find and read the answers from the mobile phone. We can address this, by encrypting the answers. To store the keys, we can use the Android Keystore system. Nevertheless, the solution of this issue is out of the scope of our paper, and we have not implemented it.

From a high-level perspective, the system works as follows. Initially, the player launches the application. Then, she can register to create a new account or log in to an existing one. During this phase, the player is required to provide her Secret Recovery Phrase (or seed phrase) of her blockchain wallet, in order for her keys to be extracted from it (the seed phrase is secured by the gaming company and it is not stored on IPFS nor the blockchain). This is to find the user's account credentials. If she does not own an Ethereum wallet, then she has to create one, in order to be able to perform transactions within the application. After the registration phase, the player's avatar is created and stored in the PlayerContract. If the player has already created an account, she logs in using her user name and her seed phrase. Then, her avatar is retrieved from the blockchain. After that phase, the player can browse the application and choose one of the following options; play, settings, or ranking. By selecting "play", the difficulty level menu appears. *R?ddle* is inclusive of players of any mathematical background and provides three levels of increasing difficulty; easy, intermediate, and advanced.

To solve a riddle or request a hint, the player has to pay a defined amount of money in ethers. When the player chooses to solve a riddle, the unique URL that correlates to it, it is extracted from IPFS and the riddle is fetched on the application. Similarly, the same process applies for the hints. In addition, players can track their own, as well as, all the other players' progress in the game by selecting the ranking option from the menu. The ranking system keeps track of the three top players in the game and it is based on the players level.

As previously mentioned, we enable the P2E gaming model. Thus, the amount of money collected from the players, who try to solve a riddle or request a hint for that riddle, is stored on the contract's balance. Then, it is transferred to the first player's wallet in the ranking system. This is done in the case that the riddle is placed by the game administrator. Finally, players can contribute to the game by creating their own riddles, uploading them on IPFS, and setting the corresponding fees. By doing so, they can get rewarded with half of the amount of money that are stored in the contract's balance for that specific riddle, while the rest of it is transferred to the first player of the ranking.

The smart contracts of the system are implemented using Solidity programming language, and they are deployed on the Ropsten Ethereum test network.[1] The mobile application is developed using the Android studio and the Java programming language. To interact with the Ethereum blockchain, we used the Web3j java and android library,[2] while for the interaction with the IPFS, we have used the io.ipfs HTTP API.

4 Evaluation

To evaluate our proof of concept implementation of the *R?ddle* game, we measured the gas used for the key functions, as well as, we calculated the time needed to perform the most important game actions, i.e., when a user solves a riddle. The transaction cost (computational and storage overhead), which in Ethereum is expressed as the cost of gas for executing transactions is calculated every time a function is called. Gas is denoted to gwei.[3] Regarding the execution time, we are considering the time needed to send the transactions to the blockchain by calculating the execution time, which is required for all the functions that take part in the scenario of solving a riddle.

In Table 1, measurements of the basic functions are shown. The UploadIPFS row refers to the execution time needed for a riddle to be added and pinned on IPFS, while the ExtractIPFS row refers to the time needed for a riddle to be extracted from IPFS and appear on the application. LoadWallet row refers to the execution time, which is required to load the Ethereum wallet credentials for a player. Moreover, execution time is calculated for the basic activities; register, ranking, login, and solving a riddle, which are all shown by the corresponding rows of Table 1. The average execution time of 20 executions, for each action, is shown in the second column of Table 1. The deviation in the measurements can be attributed to the load and the traffic of the network and is not affected by the system's implementation.

Table 1. Average execution time of key functions measured in seconds.

Functions	Execution time
UploadIPFS	2.05
LoadWallet	1.25
Register	90.27
Ranking	5.04
Login	1.75
Solve	1.71
ExtractIPFS	1.22

[1] https://ropsten.etherscan.io/.
[2] https://docs.web3j.io/4.8.7/.
[3] A gwei is equal to 0.000000001 ETH = 10^{-9} ETH

As Table 1 indicates, most actions require just a few seconds, except from the register action. Considering Table 2, which measures gas price, the function createNewAvatar, which is called in register activity, has one of the most higher prices, and the highest among non-payable methods. Note that gas is calculated based on the complexity of the function in the smart contract, and createNewAvatar is one of the most complex functions. Due to that fact, it is reasonable to say that the execution time, which is required for register activity will be higher than the rest of the functions, which are less complex. However, the aforementioned issue does not affect the player's gaming experience, as createNewAvatar is only called once, when a player registers for the first time in the application.

Regarding the cost of the game, the gas needed for the key functions in the smart contracts is calculated. The contract functions giveHint, play, and reward are payable and therefore the corresponding gas price on Table 2 takes into account both the gas needed to execute the functions and the amount of ether required to be sent to the contract, as these functions are payable methods. Function createNewAvatar is only called in register activity, while the rest of the functions are called in solve activity, meaning that they are executed every time a player tries to solve a riddle. The rest of the activities from Table 1, which does not appear on Table 2, use only view functions, which are read-only and thus, they do not introduce any cost.

Therefore, as calculated from Table 2, in order for a user to solve a riddle, which is the main aim of the game, an amount of gas equal to 0.00207247 ether is required in case the player does not request a hint, whereas in the other case, an amount of gas equal to 0.00299355 is needed.

Table 2. Transaction gas fee of key actions, measured in gwei.

Functions	Cost (gwei)
CreateNewAvatar	854,930
GiveHint	921,080
Play	720,570
Reward	681,160
Solve	670,740

However, in our P2E gaming model, players can create riddles on their own or try to get a higher place in the ranking and hence receive rewards. In the first case, they receive half of the money for each riddle a player solves, while in the second one, they may even receive the whole amount of money for a riddle. In either cases, players can receive by far more money than the amount of ether they used to play the game. As illustrated in Fig. 2, if six players try to solve a riddle placed by a specific user, the user receives 0.00216171 ether, which is more money than the amount of ether he used to play the game. Consequently, for a number of players greater than six, the user, who created the riddle can make a profit.

In addition, a player can earn more money by getting the first place in the ranking system. In this case, the user will receive half of the amount of money

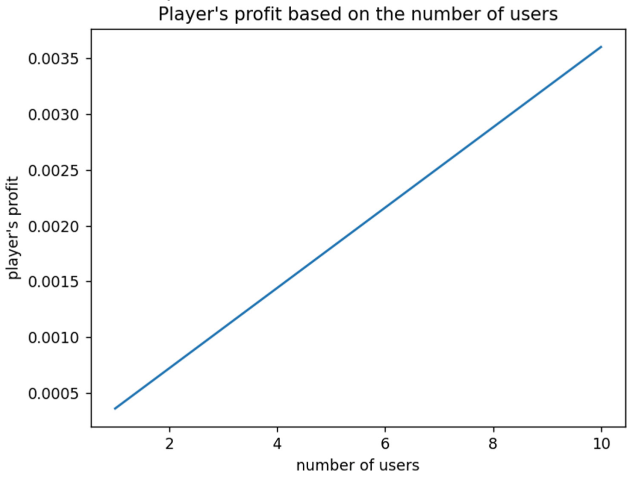

Fig. 2. Player's profit based on the number of users.

for riddles created by other players, in which scenario, as we discussed earlier, for a number of players greater or equal to six, he receives more money than the amount he used to play the game. On the other hand, if the riddle is placed by the game administrator, the player who ranked first receives the whole amount of money that each user paid to solve the riddle. Again, the profit increases linearly with the number of players. In this case, the top player can make a profit even if only three players try to solve a riddle created by the game administrator. If the aforementioned cases are combined, the player, who ranked first can earn a significant amount of money, which gives a motivation to players to try solving riddles and continue playing the game.

5 Discussion

In this paper, we presented a system that benefits from the blockchain technology, as well as enables the P2E gaming model, in which players can make a profit by creating their own riddles or solving riddles and ranking first in the game. Moreover, we used the Ethereum's capability for smart contracts in order to build a totally decentralized mobile game, open to all players, who just own an Ethereum wallet. We also took advantage of IPFS, as a decentralized storage unit, in which riddles and their corresponding hints are uploaded.

Our game has several advantages compared to legacy (blockchain-based) games. First of all, smart contract execution is deterministic and a smart contract once deployed, it cannot be modified. In this way, it is guaranteed that the rules of the game will not change with the time, since transactions do not require a central authority and their records are immutable and distributed across the network, meaning that they are visible by all participants at all times. Thus, our system is transparent, in the sense that players can be sure that the rules of the game will always be respected and no one, not even the gaming company, can modify them. Moreover, the use of smart contracts allows us to easily implement automated decentralized payments without the need of trusted intermediaries, enabling the P2E gaming model.

Furthermore, our system is fully decentralized as it is based on the Ethereum blockchain and IPFS. The riddles and their corresponding hints are stored on IPFS and not on centralized trusted servers owned by the gaming company, as it happens in traditional games that use relational databases owned and managed by the gaming companies. Players can also upload their own riddles on IPFS, without the interception of any centralized party. In this way, trust is promoted as players do not rely on any centralized authority. They can try to solve any riddle they want or create their own as the rules in the smart contracts indicate. Players also have the chance to receive rewards from the game, by contributing to it. Contributions entail uploading their own riddles. However, players can also earn an amount of money in ether by ranking first in the game. In this way, they have the opportunity to solve riddles based on their preferences on riddles, which can increase their probability to receive rewards.

Finally, by using IPFS, we reduce the chances of hardware overload for the gaming company by decreasing the storage requirements of our game. Moreover, with the use of IPFS as a storage unit (and pinning services), we ensure that the riddles and the corresponding hints will be available for long term. Even if the gaming company stops supporting the game, or if the company suffers from network outages, the game will continue to be available and playable.

Despite the numerous (security) advantages in the (mobile) gaming industry that blockchain technology can offer, as we have already mentioned above, it introduces a cost, which is based on the transactions that occur inside the blockchain. Due to the fact that gas is calculated based on the computational cost of a function, complex functions consume more gas. In our system, the function that is responsible for the creation of a player's avatar requires a significant amount of gas. However, it does not affect the player's gaming experience, as the method is only called once, when the player registers in the application.

In order to reduce the amount of gas that is consumed when a function is called, functions should be less complex, which means that smart contracts should consist of only the attributes that are important for the players or the game administrator in order to interact with the blockchain. More specifically, in our design, smart contracts contain only necessary information (e.g., player's account, riddle's ID, etc.) resulting in the minimization of the gas used for all actions. IPFS also plays a significant role in gas minimization, as it stores both riddles and their hints, while only the hashes of the riddles are stored on the blockchain.

6 Conclusions

In this paper, we presented *R?ddle*, a game relying on the Ethereum blockchain and the IPFS for efficiency, enabling a fully decentralized ecosystem for posing and solving (typically mathematical) puzzles. Anyone with access to the blockchain can pose riddles (and set economic parameters) and anyone can choose the riddles to play and solve, being rewarded accordingly. Introducing puzzles, one can be rewarded but also, more importantly, she can decide on their own the game parameters, such as cost to play. On the other hand, players can decide to play games they just like, or games based on their expectation to benefit from. Our design offers transparency, high availability, full decentralization, and enables the P2E gaming model.

Acknowledgments. The work reported in this paper has been funded in part by the Research Center of the Athens University of Economics and Business.

References

1. Benet, J.: Ipfs—content addressed, versioned, p2p file system (2014)
2. Herrera, P.: State of the blockchain game sector (2021). https://dappradar.com/blog/bga-blockchain-game-report-july-2021
3. Karapapas, C., Pittaras, I., Polyzos, G.C.: Fully decentralized trading games with evolvable characters using NFTs and IPFS. In: Workshop on Decentralizing the Internet with IPFS and Filecoin (DI2F), in Conjuction with IFIP Networking Conference 2021, pp. 1–2 (2021)
4. Kaur, M., Gupta, B.: Metaverse technology and the current market (2021)
5. Mavis, S.: Official axie infinity whitepaper. In: Axie Infinity (2021). https://whitepaper.axieinfinity.com/
6. Min, T., Wang, H., Guo, Y., Cai, W.: Blockchain games: a survey. In: 2019 IEEE Conference on Games (CoG), pp. 1–8. IEEE (2019)
7. Pittaras, I., Fotiou, N., Siris, V.A., Polyzos, G.C.: Beacons and blockchains in the mobile gaming ecosystem: a feasibility analysis. Sensors **21**(3) (2021)
8. Sako, K., Matsuo, S., Meier, S.: Fairness in ERC token markets: a case study of cryptokitties. In: International Conference on Financial Cryptography and Data Security, pp. 595–610. Springer, Berlin (2021)
9. Entriken, W., Shirley, D., Evans, J., Sachs, N.: EIP-721: ERC-721 Non-fungible token standard (2018). https://eips.ethereum.org/EIPS/eip-721
10. Wood, G.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Project Yellow Paper 151 (2014)