Game Protocol: A Decentralized Economy for Creating Games

The GamyTech Team

Contents

1 Mission

Real money competitive games are a genre of games in which players compete based on their ability, rather than luck or chance, to win real money. GamyTech is a multiplayer game platform that lets players compete in mobile devices and PCs with others around the world for real cash. After successfully releasing 8 titles played by millions of users, we at GamyTech decided to open up our platform to the public so that everyone can produce and release their own games. This will make GamyTech the first open game platform where 1) players can enjoy a deluge of new titles to play with, and 2) game producers can create new games in their own full vision, without bending to the whims of publishers.

To that end, we plan to develop Game Protocol (GP), a first-of-a-kind blockchain-based gaming economy to enable people with different skills and resources to come together to create and experience new games on our open GamyTech platform. People in the GP community exchange Game Protocol Tokens (GPTs), a native crypto-currency of the economy, for services and resources provided by others in their own mutually agreed terms. This market-based exchange mechanism constitutes the foundation of the GP economy. Specific interaction patterns (protocols) between participants of the GP economy are cemented into smart contracts to streamline and automate the exchange processes.

GPT will be listed on all major crypto-currency exchanges. A healthy economy is one in which everyone collaborates to grow the economy and subsequently benefits from the growth. Exchange tradable GPT enables contributors to the GP economy to be rewarded financially for their efforts.

2 Game Protocol Services

The first obstacle an independent game producer needs to overcome is to raise enough funds for his new project. Fundraising has always been a time-consuming, expensive process; yet it is a prerequisite for any new game development. In the GP economy, we provide GameStarter, a smart contract based, fully automated crowdfunding system to help producers get most from their fundraising efforts.

During the new game development phase, the producer needs to pay developers to work on his new project. One major concern both have is that the cost of a direct on-chain payment transaction is too high. To address this issue, the GP economy provides is a low-cost off-chain payment channel via which anyone can pay anyone else with a fraction of the on-chain cost.

Promotion and marketing is another hurdler the producer needs to overcome after finishing his new game. The GP economy provides a cross marketing service in which a new game can be promoted by established games to reach their existing user bases.

Besides these three services, we also provide a developer's tool that contains specific utilities for real money competitive games. Developers can utilize these tools to quickly code a new game.

GamyTech has extensive experience in producing new games. We experienced and successfully overcome the aforementioned pain points. In the past 3 years, we raised over 3 million dollars and released 8 new games. Our games have been downloaded more than 4 million times. Our existing premium user base (users who actually spent money in the game) has about 700K members and they collectively won about 27 million US dollars.

The GP services and tools will be a great enhancement to the GamyTech platform. By leveraging these services, independent producers can be confident that they can quickly release new games and get direct access to a large user base on the GamyTech platform.

3 GameStarter

GameStarter is a one-stop crowdfunding platform for new game projects. A game producer issues a new Game-Specific Chip (GSC) via GameStarter to fund his project. By crowdfunding games on GameStarter, producers can create games in their own full vision, without bending to the whims of publishers or dealing with the bureaucracy that exists in a fiat based crowdfunding venture.

Investors exchange GPTs, the native currency, for the GSCs issued for a new game project. The utility of a new GSC is completely at the game producer's discretion: they can, for example, exchange for in-game special features, provide full access to the new game, act as a substitute for real cash in the game, etc.

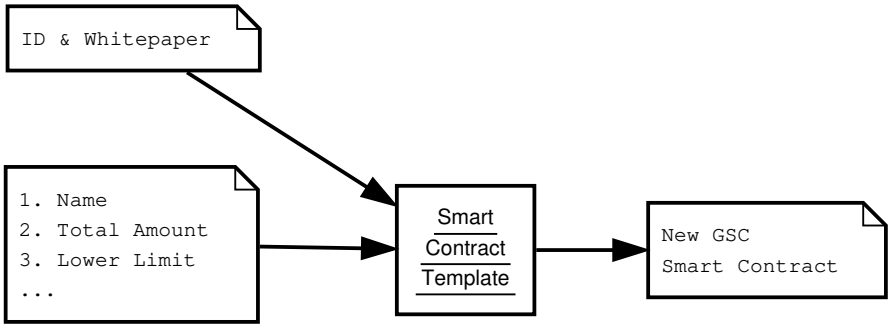

Figure 1: Generate GSC smart contract.

The GameStarter fundraising process consists of 2 phases: In the first phase, the producer submits a proper identification and a whitepaper to GameStarter. GameStarter will publish the id and the whitepaper as a potential new project on the GameStarter bulletin board so that investors can examine and evaluate the proposal. In the second phase, a smart contract will be automatically generated for the producer to manage and maintain the new GSC.

Figure 1 illustrates the process of generating a new GSC smart contract. Note that ID and whitepaper submitted by the producer in phase one will be embedded into the generated smart contract and stored in the blockchain. This ensures that all information relevant to every fundraising campaign in GameStarter is permanently back-trackable. To generate a new smart contract, the producer is required to provide the information shown in Table 1. This information is combined with a smart contract template as illustrated in Figure 1 to generate the final GSC smart contract.

The generated GSC smart contract accomplishes two tasks: 1) manages the funds during the fundraising campaign, i.e., receiving GPTs from investor and allocating GSCs to them, and 2) maintains a liquidity pool from which anyone can buy or sell the GSC at any time after the fundraising campaign finishes.

To better understand how the generated GSC smart contract achieves both functions, we need to first describe how a GSC ledger works.

Index	Field	Explanation
1	Name	Name of the GSC
2	Total amount	Total amount of GSC to issue (e.g., 1M GSCs)
3	Lower limit	The minimum amount of GPTs to be raised for this campaign to be considered a success (e.g., 1000 GPTs)
4	End time	End time of this campaign
5	GSC campaign price	The price investors pay for the GSC using GPT in this campaign (e.g. 0.5 GPT per GSC),
6	Producer's account	The account address that will hold the raised GPTs if the campaign succeeds,
7	Reserve ratio	Used by smart contract to determine the spot price of the GSC after the campaign (e.g. 30%)

Table 1: GSC smart contract inputs.

3.1 Tale of Two Ledgers

Every GSC has its own ledger on the blockchain, just like the main currency GPT, as shown in Table 2. However, unlike GPT, which can be traded at exchanges for other crypto-currencies or fiat currencies, a GSC can only be exchanged for GPT. Furthermore, a GSC is not listed on any exchange and has its own liquidity pool, from/to which anyone can buy/sell the GSC. Therefore, GPT serves as the reserve currency in our economy, while every new game project can issue its own sub-currency (i.e., GSC) that can be exchanged for GPT.

Exchanging GSC for GPT involves two ledgers: the main GPT ledger, and the GSC ledger. To take an example, let's say account 0x0cd2a wants to buy 10 GSCs from account 0x227b at the price 0.5 GPT/GSC. In this case, we need to transfer 5 GPTs from the former account to the latter on the main GPT ledger, and at same time transfer 10 GSCs from the latter account to the former on the GSC ledger.

Since each ledger is controlled by its respective smart contract (the GPT smart contract and the GSC smart contract respectively), these two smart contracts need to coordinate to complete this 2-legged transaction. This coordination is achieved by registering the GSC contract as a "trusted contract" to the main GPT contract. The main GPT contract contains a privileged function that transfers GPTs between accounts on the main GPT ledger. Only trusted contracts can access this function. Clearly, this prevents bad actors from stealing GPTs by calling the privileged function.

GPT Ledger	
...	...
0x0cd2a...	100 GPT
...	...
0x227b...	0 GPT
...	...

GSC Ledger	
...	...
0x0cd2a...	0 GSC
...	...
0x227b...	1M GSC
...	...

Table 2: GPT and GSC ledgers.

3.2 Fundraising Campaign

During fundraising campaign, all GPTs raised will be stored in an auxiliary account address generated by GameStarter, over which the producer has no control. This structure is necessary as not all campaigns will succeed. If they don't succeed, all GPTs raised will be returned to their original owners by GameStarter. If they do succeed, the total balance will be transferred to the producer's account (input 6 in Table 1) that he controls.

To start a fundraising campaign, the GSC smart contract first sets up two separate account entries on the GPT and GSC ledgers respectively, using the generated auxiliary account address. The entry on the GPT ledger will have zero balance as no GPTs has been raised yet; and the entry on the GSC ledger holds the total amount of GSCs to be issued (input 2 in Table 1). Table 2 shows these entries with auxiliary account address 0x227b.

An investor calls a function "*campaign_ buy(address investor_ address, uint GSCs_ to_ buy)*" within the GSC smart contract to buy GSCs. This function will first call the main GPT contract to check if the investor has enough GPT balance. If he does, the contract will transfer the specified amount of GSCs from the auxiliary account to the investor's account on the GSC ledger and then it will call the main GPT smart contract to withdraw the equivalent amount of GPTs from the investor's account to the auxiliary account on the GPT ledger. The exchange price is specified by the producer (input 5 in Table 1).

If the lower limit (input 3 in Table 1) is met at the end time (input 4 in Table 1), then the campaign succeeds. When this happens, function "*campaign_ buy*" will be disabled, i.e., no one can buy the GSC from this function anymore.

3.3 Liquidity Pool

If the campaign succeeds, the auxiliary account will be closed. All the assets, including the raised GPTs and the remaining GSCs, will be transferred to the producer's account (input 6 in Table 1). From that point on, the producer has total control over all the assets he owns. He can, for example, withdraw the GPTs or GSCs to pay a programmer to work on this project. Like the auxiliary account, the producer's account has two sides too: one on the PG ledger holding the GPTs he raised and the other on the GSC ledger holding the remaining GSCs. Both entries have the same account address provided by the producer.

The assets (GPTs and GSCs) on the producer's account constitute a reserved liquidity pool that enables anyone to buy/sell the GSC from/to the pool at any time using GPT. The spot price of the GSC is dynamically calculated on the fly by a Bancor formula:

$$spot\ GSC\ Price = \frac{GPT\ in\ liquidity\ pool}{outstanding\ GSC \times reserve\ ratio},$$

where

1. "outstanding GSC" is the total amount of GSCs in circulation (i.e., the total amount of GSCs issued (input 2 in Table 1) minus the GSC balance on the producer's account);

2. "GPT amount in liquidity pool" is the total amount of GPTs held on the producer's account;

3. "reserve ratio" (input 7 in Table 1) is the fraction of the outstanding GSCs that is backed by the GPTs on the producer's account.

Using this formula to determine the spot price of GSC, the smart contact basically acts as an automatic market-maker for the newly issued GSC on behalf of the producer. The above formula gives the price of trading an infinity small unit of GSC, to determine the amount of GSCs one can buy for a given amount of GPTs for a buy transaction, or the amount of GPTs one receives by selling a given amount of GSCs for a sell transaction, one needs the following two formulas respectively. Refer to Bancor[1]for details.

[1] `https://about.bancor.network/`

$$GSC\ received = outstanding\ GSC \times ((1 + \frac{GPT\ paid}{GPT\ in\ liquidity\ pool})^{reserve\ ratio} - 1),$$

$$GPT\ received = GPT\ in\ liquidity\ pool \times (1 - (1 - \frac{GSC\ sold}{outstanding\ GSC})^{1/reserve\ ratio}).$$

Figure 2 illustrates the connections between GSC, GPT, Bancor, other cryptocurrencies and fiat currencies. One can exchange GPT for any GSC issued on the GP platform. GPT is listed on crypto exchanges, and is also supported by Bancor. Anyone can either go to an exchange or trade directly with Bancor to buy/sell GPT,

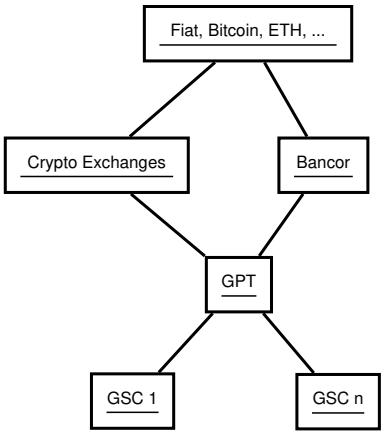

Figure 2: Connections between GSC, GPT, Bancor, and other cryptocurrencies and fiat currency.

4 Off-chain Payment Channel

We anticipate there will be a large volume of direct payments occurring between GP community members. To lower the transaction cost, we plan to implement an off-chain payment channel for the GP community. Our goal is to lower the cost to a small fixed fraction (say, 5%) of the total amount transferred.

To achieve this, we cache transactions into batches off-chain into a database, and then execute them on the blockchain until the batch size reaches a certain threshold. What this means to payment receivers is that they will not be able to cash which they received immediately. As

	micro payment account balance	$user_1$	$user_2$	...	$user_i$	...
$user_1$	27.89	0	5	...	2.1	...
$user_2$	30	10	0	...	0	..
...	...	...	...	...	...	...
$user_i$	98.35	56	2	...	30	..
..	...	...	...	...	...	...

Table 3: Cached payment transactions.

this may not be desirable for some, they always have the option to execute their transactions on-chain directly (and pay the on-chain transaction cost).

A user who wants to utilize our off-chain payment channel first needs to setup an off-chain payment account with GamyTech. Once this is done, he needs to fund that account with a minimum amount of tokens. Only tokens held in the off-chain account can access the low-cost payment channel.

Each payment from every off-chain account is recorded into a database. Conceptually, all these records can be organized as a table, as shown in Table 3. Every time a new payment occurs, we update the table and check if there is an opportunity to execute all the cached transaction in a batch. An outline of the algorithm in shown in Algorithm 1.

In Algorithm 1, S_{ij} is the accumulated payment from $user_i$ to $user_j$. At any point in time either S_{ij} or S_{ji} or both can be zero too, this is because the algorithm always nets out the total payment between any pair of users. This netting-out procedure is implemented from line 7 to line 15. At the end of the algorithm, we estimate the total cost of clearing all cached transaction to the blockchain. If the cost is less than a fraction of the total token value, we clear all the transactions.

Many variations of this algorithm can be considered in practice. For example, we can guarantee the time of payment instead of the cost. To do so, we simply need to clear the cached transactions periodically. We can also mix these two criteria: try to seek for the low-cost clearance opportunities within a given time period and clear the cached transactions if the deadline has passed.

Input : $user_i$ pays n tokens to $user_j$

if *$user_i$'s mircopayment account balance* $< n$ **then**
  trasaction fails;
else
  reduce $user_i$'s mircopayment account balance by n;
  let S_{ij} and S_{ji} be the values at cell ij and ji respectively;
  assert ($S_{ij} == 0$ or $S_{ji} == 0$);
  if $S_{ji} == 0$ **then**
    $S_{ij} = S_{ij} + n$;
  else
    if $S_{ji} \geq n$ **then**
      $S_{ji} = S_{ji} - n$;
    else
      $S_{ij} = n - S_{ji}$;
      $S_{ji} = 0$;
    end
  end
  let $a = \sum_{ij|S_{ij}>0} 1$, $b = \sum_{ij} S_{ij}$, $c =$ current average on-chain transaction cost, $d =$ current token value;
  if $\frac{ac}{bd} < 5\%$ **then**
    clear all cached transactions to the blockchain;
  else
    return;
  end
end

Algorithm 1: Off-chain payment algorithm.

5 Cross-Marketing

The cross-marketing service enables an established game to promote a new game. This type of marketing typically is very effective since the targeting users have already been exposed to the established game.

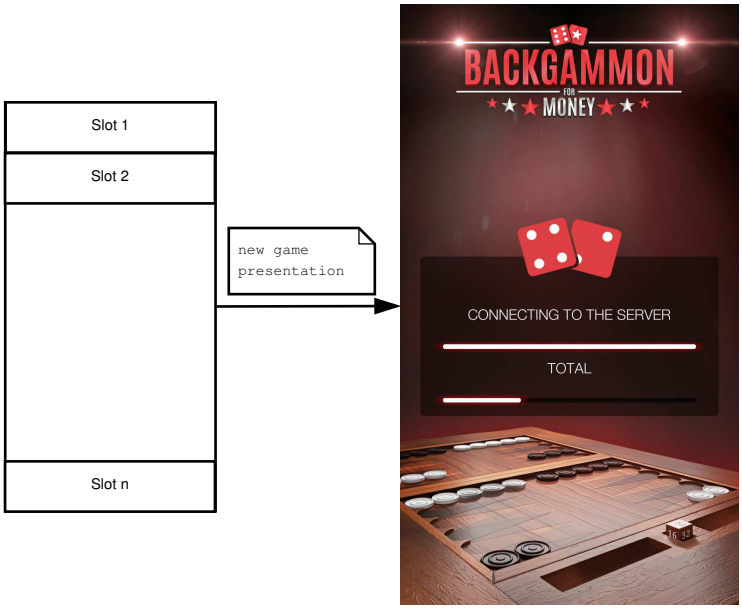

Figure 3: Cross marketing.

The cross-marketing system maintains a list of presentation slots for an established game on a server. A new game can purchase these slots from the established game at a mutually agreed price. Once again, the terms and prices are solely determined by the two parties; and they also have the option to choose the low-cost off-chain payment channel to transact their payments.

Every time a user starts to play the established game, a presentation of a new game in a slot will be selected in a round-robin fashion and shown to the user. Figure 3 illustrates such a process. Backgammon4Money shown in the illustration is one of GamyTech's most profitable games.

Since real money games always need to connect to the game server for each play, it is a perfect time to present a promotion every time when a fresh play starts while the user is

waiting for the connection. Another presentation opportunity is when the game finishes. It is generally undesirable to present any promotions during the game, as it is considered an extremely un-welcomed distraction for players who are competing to earn real money.

Given our large and loyal user base, we expect the cross marketing service to become one of the most attractive features of the GP economy to game producers.

6 Developer's Tool

The developer's tool contains a set of utilities that developers can leverage to quickly build a new real money game. These tools are carefully designed to address common issues of real money games; and they will greatly simplify a developer's work.

Specifically, the developer's tool provides API access to 1) a credit card payment gateway for real money, 2) a crypto-wallet to hold player's crypto-currency, 3) a wagering smart contract, 4) a SDK for easy accessing the central game server, and 5) a provably fair random number generator.

6.1 Real Money Payment Gateway

One sought after feature that our platform provides to game producers is a gateway to receive real money in their games. Maintaining such a gateway is very expensive and thus it is typically beyond an independent producer's reach. By building and releasing new games on the GamyTech platform, a producer can access this gateway through an API we provide.

This service is production-ready and field-tested as we currently use it to process real money in our own games.

6.2 Crypto-wallet

Any crypto-currency supported by the GP economy, including ETH, GPT, and the GSC issued by a game can be used as a utility token within the game. To support such a function, we have developed a multi-currency crypto-wallet plugin for the Unity game engine (Unreal engine will be the next). This wallet securely holds a player's assets, and it also provides a secure, explicit

authorization API to access, the assets. Specifically, the crypto-wallet supports the following functions:

1. create, import and export wallets with encrypted keyfile;
2. mnemonic sentence and private key recover;
3. direct transfer tokens between wallets;
4. add any existing smart contract for use in the wallet;
5. use contract functions and transactions.

Since the wallet is integrated with the game engines, it enables a player to easily spend tokens in the game (with explicit authorization), and to collect rewards from a game as well. The crypto-wallet is open-source, and the code will be posted on GitHub to allow developers to examine it and adapt it to their needs.

6.3 Wagering Smart Contract

The main purpose of the wagering smart contract is to keep the tokens of the players during a game in a trusted third party. Before any game starts the player will have to give permission to the wagering contract to facilitate transactions on his behalf (allowance).

The allowance amount should be at least the amount the player is planning to play on.

The wagering contract will contain 3 functions that will be used by the server.

1. Game Started - Will receive the player's addresses and the game variables like game ID and bet. the function will transfer token from the players using the allowance to keep until the game is finished. if one of the transfers fails the other player will be refunded and the game will not start.
2. Game Finished - Will receive the end game state (winning player). Once the game is finished successfully the contract will transfer the wagering pot to the winning player and the fee to contract owner.
3. Game Cancelled - If the game encountered any issue or error, the server will decide to refund both players. The funds will be returned to both players wallets.

6.4 Server SDK

Game Protocol provides an SDK (software development kit) for game developers to assist them in developing games. This SDK will remove the need to develop a dedicated server or integrate the blockchain themselves. That will allow game developers to fully focus on developing their game.

Initially, the client will use the crypto-wallet to acknowledge their funds to know which matches they can participate in accordance to their balance. After the wallet's funds have been acknowledged, the client will connect to the server via a persistent connection and will pass the wallet information so that the server can verify the user and their currency (verify with the blockchain). After the successful verification, the user will have to state how much currency he would like to wager with and will give permission to the server to reduce that chosen amount on the blockchain when the match begins on the server. After allowance approval, the server will start searching for a matching opponent with the same chosen bet. Once a match is found, the server will turn to the blockchain and will remove the funds from the two opponents and will store them via the wagering smart contract and there the funds will wait until a winner is proclaimed.

Since the transfer can take some time, the server will not wait for the approval of the transferred funds and let the match start while he waits for the approval. In case one of the transfers have failed the match will be canceled and the funds will be returned.

6.5 Random Number Generator

One concern many players have is the fairness of the random elements (e.g. dice) in games. Especially if they lose, they tend to believe that there is dice manipulation which hurts their position in the game. To address this concern, we have constructed a blockchain-based random number generator that is provably fair, i.e., any game participants can check and prove that no one can manipulate the random number generated in the game.

One simple approach is to use the blockchain as the source of randomness. Specifically, here is the algorithm.

A few notes about Algorithm 2:

1 Each player i logins into the central server;
2 After registering all the players, the central server waits for K new blocks to be confirmed on the blockchain;
3 The central server uses the hash of the last confirmed block (i.e., the K-th block) as the seed to generate a random number in the gameplay;
4 Repeat the above steps if another random number is needed in the game;

Algorithm 2: Simple random number generator

1. The random number generator runs on the central game server.
2. Everyone can confirm and verify the random number generated is seeded by the hash of the K-th block.
3. Since no one knows the hash of the K-th block beforehand, this algorithm is provably fair.
4. We have to use a new seed every time a random number is needed to make sure that no player can simulate the game.
5. Players have to wait for a period of time before they start playing the game as the server is waiting for the confirmation of blocks.

One concern over Algorithm 2 is that blockchain miners have control over what blocks to generate in what order. Even though it is highly unlikely any miner will be able to strategically generate blocks in such a way that hurts the winning chance a game player, we still provide another algorithm that eliminates this concern as shown in Algorithm 3.

A few notes about this algorithm:

1. Since every player uploads his private number p_i only after everyone else hash hp_j has been published, he can be sure that nobody can collude with the central server to manipulate the random number generator.
2. The central server cannot manipulate the random number either since the seed hp is determined by all the players collectively.

1 Each player i generates a private number p_i, calculates the hash of p_i as $hp_i = SHA3(p_i)$ and publishes hp_i to the blockchain;
2 Each player polls the blockchain. A player i sends the central server his private number p_i only after he has collected the hashes hp_j of all the other players;
3 After receiving p_i from a player i, the central server checks if $hp_i = SHA3(p_i)$. If not, abort and log player i as a bad actor;
4 After validating all p_i's, the central server calculates $p = \sum_i p_i$, and $hp = SHA3(p)$;
5 The central server uses hp as the seed and generates a random number for the gameplay;
6 Repeat the above steps if another random number is needed in the game;
7 After the game is over, the central server publish all private numbers received from all the players to the blockchain;

Algorithm 3: Random number generator version 2.

3. A fresh seed is used to generate every single random number, a player can be sure that the central server can't collude with his opponent in any way to increase the winning chance of the latter.

This is an expensive algorithm as participants need to access the blockchain multiple times to establish unalterable records to prove fairness. This is because we assume that none of the game participants (including the players and the central server) trusts each other. Since this algorithm uses the blockchain merely as a public whiteboard for information sharing (instead of transaction processing), we can use free and fast blockchains like IOTA or Openchain for that purpose. This will significantly reduce the cost and waiting time.

7 Token Offering Details

Only 150,000,000 GPTs will be created. There will be no dilution and no further token production. 58% of GPTs will be distributed to the public. 10% of the GPTs issued will be retained in a special Game Support Fund. This fund will allow the company to support promising game projects that will be listed on GameStarter. The company will retain the ability to designate which game projects they will choose to support. 20% of the remaining GPT's will be retained by the company (locked 25% per 6 months). Table 4 shows the full distribution details of Game Protocol.

Distribution	Amount	Percentage
Total GPTs	150,000,000	100%
Crowd-sale	87,000,000	58%
Gaming Support Fund	15,000,000	10%
Bounty Program	3,000,000	2%
Advisers and Partnership	15,000,000	10%
Company (Locked 25% per 6 months)	30,0000,000	20%

Table 4: GPT distribution.

8 Team and Advisers

Information about all team members and advisers can be found at:

`http://gameprotocol.io`

9 Market Analysis

The global gaming industry expands across multiple sectors and platforms. Most analysts split it up in three categories: PC, Console, and Mobile. It is estimated that 2.2 billion gamers worldwide participate in these three gaming mediums.

Newzoo, a prominent gaming analyst, estimates the 2017 global games market to be valued at $108.9 billion, representing an increase of 7.8% or $7.8 billion from the year before. Of this, PC games account for more than a fifth of the market, representing a $29.4 billion share. 62% of the global gaming population use a PC. Of the $29.4 billion of global PC gaming sales, the most popular gaming genre, around 37% of the market, is turn based strategy and arcade gaming. With the global gaming market increasing at an average rate of about 5% a year, scam and hack attacks are becoming more and more prevalent. Within the gaming market, PCs are the most frequently hacked device. Despite these attacks, PCs are frequent gamer's most used device at 56% of the population. The ESA defines a frequent gamer as playing at least three hours a week. Evidently, even the gaming industry's most experienced users are not protected from attacks. These gamers are at risk of both external and internal frauds. External fraud happens as the result of outside hackers, while internal fraud comes from the program owners themselves.

Conclusively, gamers are at an increased risk of scam and hackings as the global market expands, therefore a decentralized, transparent, and trusted solution is necessary to protect the growing population of gamers.