Software requirements specification for POC

Authors

Malov Alexei

Dubov Alexei

Sartakov Alexander

Introduction

Nowadays the game industry is highly regarded. Many people play computer games, especially online computer games. But as in every sphere in games not all of the people are honest. Some of them use "cheats". Almost all of the online games uses client-server method of data processing. In many cases the data of game is stored in the memory on the user's PC. If the data of the game is reached somehow by the user, it can be easily changed. Another way of dishonest game is to change the API request user sends to the server, so the data of the game are changing in a wrong scenario. And one more, some advanced users can break into the server and change the data on it. In these case we say the user cheats. It almost always affect the course of the game, one person or a group now get an advantage over others. It can ruin the users experience of the game and they might even quit the project, what none of the companies want. Now we can easily determine cheats as wrong way of getting an advantage over other players and our goal is to prevent this.

There are a lot of various anti-cheats solutions in order to do the product unbreakable for cheaters, but all of them are not perfect. Need to admit, that cheating in games are not unique, people trying to do this in every sphere, where some computers runing. One of the most powerful security software system is blockchain. Now it uses in cryptocurrency, when people make operations (buying, selling ot mining cryptocurrency) the data is send as JSON-file with some hash-key, (these file have a special name "smart-contract"), then all of the computers (miners) trying to guess this key, and the first gets the reward, then the information about this writes into blockchain. Unlike centralized banks system, cryptocurrency is decentralized system, i.e. there's no "building" with the servers, that you can hack. All of the informaion about the blockchain is stored on all of the computers (miners), so you can't hack it, because none of the people have such computing powers, to calculate all of the hash-key since some period till the end of the time. This concept we can implement into some game, to prevent our system from being hacked. Right now, this idea can be applied to games only, where the amout of operation per second (commands sent to server) is quite low, but in the future the idea will be developed.

For our project work, we decided to make a clon of "Hearstone" and implement a smart-contracts concept directly in it.

Glossary

Game engine - is a software framework, primarily designed for the development of video games, and generally includes relevant libraries and support program.

Contract - legally binding agreement that defines and governs the rights and duties between or among its parties.

Smart-contracts - is a computer program or a transaction which is intended to automaticaly execute, control or document legally relevant events and actions according to the terms of a contract or an agreement.

Blockchain - is a growing list of records, called blocks, that are linked together using cryptography.

Cryptography - is the practice and study of techniques for secure communication in the presence of adversarial behavior.

Cheat - script which is used by one of the player to have an advantage over another player.

Cheater - player, who uses cheats to get an advantage over other players

Actors

Server: There we store packages with game's data.

Players: They send packages with game's data to the server. One of them is trying to rewrite them, in order to have an advantage over another player.

Functional requirements

Use cases:

1. Player#1 is trying to use cheat in the game without smart-contracts.

   Actors:

   - Players
   - Server

   Goals:

   - To have an advantage over another player.

   Precondition:

   - Players are playing computer card game.

   Trigger condition:

   - One player wants to win with dishonest way.

   Main success scenario:

   1. Player#1 executes program, which rewrites game packages.
   2. Game packages are rewritten successfully. Player#1 gets an advantage.

2. Player#1 is trying to use cheat in the game with smart-contracts. Cheat doesn't work.

   Actors:

   - Players
   - Server

   Goals:

   - To have an advantage over another player

   Precondition:

   - Players are playing computer game.

   Trigger condition:

   - Player wants to win with dishonest way.

Actions:

   1. Player#1 executes program, which rewrite game packages.
   2. Smart-contract does not let rewrite packages .
   3. Player#1 does not get advantage.

3. No one uses cheats, everyone is honest c:

   Actors:

   - Players
   - Server

   Goals:

   - Having fun while spending time

   Trigger condition:

   - Players are playing the game

   Actions:

   - Players are playing the game

System-wide functional requirements

- The system [game] is protected against hacking [attacks] made by players.

Non-functional requirements

Environment

- The system shound run smoothly on machines with 4 GB RAM and a quad-core x86 CPUs
- The application can be used on any windows machine with internet connection and and characteristics not lower than those presented in the paragraph above

Performance

- One server should work smothly with al least 20 students
- Calculations time should be under 3 seconds.

Reliability

- The server should work for a week without restart

Extensibility

- The amount of servers can vary