TEE-based Scalable and Cheat Resistant Online Video Game Architecture

Terufumi Hata[*]
terufumihata@sslab.ics.keio.ac.jp
Keio University, Japan

Pierre-Louis Aublin
pl@sslab.ics.keio.ac.jp
Keio University, Japan

Kenji Kono
kono@sslab.ics.keio.ac.jp
Keio University, Japan

Introduction. The online video games industry represents, as of 2020, more than 79 billion USD and more than 2.5 billion players. In this context, cheating is an important problem that can negatively impact the outcome of a gaming competition, damage the video game industry economy, or degrade the gaming experience of honest players.

To prevent cheating, many video games rely on a client-server architecture, where a central server validates players input before redistributing them to a subset of the players with the objective of minimizing the game state known by the players. This has the advantage to offer a good cheat-protection, but at the expense of the scalability. As an example, games today allow only a few dozens of players in a single online session.

To improve scalability, some games move functionalities from the server to the player side. Unfortunately it makes cheating easier. An example of such cheat is a wall-hack cheat, where the cheating player can see the position of all the other avatars on the map. This is possible if the server does not restrict access of a given avatar's position only to the players in the field of view.

At the other end of the spectrum is the peer-to-peer (P2P) architecture, where players exchange updates to their game state with every other players without relying on a central server. While it provides a better scalability, this architecture is not resilient to cheating. First, it is easy for a cheating player to tamper with the game state without being detected, for example to gain access to the position of other players on the game map or to modify its avatar characteristics. Second, to keep the game state consistent across all players, they have to exchange all their updates with every other player, thus sending critical information (such as the player avatar's position) to the cheating player.

To address this problem we propose a new online video game architecture that is both scalable and cheat resistant. *Scalability* is achieved by a P2P architecture, where players communicate directly between them without relying on a central server. *Cheat-resistance* is achieved by making use of a Trusted-Execution Environment (TEE), a special execution mode that provides integrity and confidentiality guarantees to code and data even in an untrusted environment (software and hardware) controlled by a malicious entity. Example of TEEs in commodity processors include Intel SGX, ARM TrustZone or AMD SEV-SNP.

Design. The main challenge of our architecture is to minimize the amount of information exposed to a cheating player. This is done in two steps: (i) storing the sensitive code and data that could be leveraged by a cheating player into the TEE; and (ii) securing the network communications.

First, to prevent a cheating player to get access to or tamper with the game state, it must be stored inside the TEE. This includes the game map, the list, position and characteristics of avatars, etc. The TEE needs to encapsulate any information that could be leveraged by a cheating player who wants to gain an advantage in the game as well as the code that manipulates it. The rest of the code, such as handling graphics or user inputs, remains outside of the TEE.

Second, to prevent a cheating player from extracting information from the network messages it receives, as well as to prevent him from modifying messages sent to other players, the network traffic must be encrypted, with the en/decryption done inside the TEE. We use AES-GCM, a cryptographic cipher providing both integrity and confidentiality.

To ensure all players execute the correct code inside the TEE and to provide encryption keys, we need to rely on an authentication server. This server is similar to the authentication server used by players when login to the game session, thus requiring minor changes from the game developper.

Interface. The interface between the TEE and the untrusted code is a good vantage point for the cheating player to launch attacks. It is thus of prime importance to harden it, with extra security checks (checking the origin of pointers, preventing the copy of secure data, adding control-flow integrity, etc.).

The interface is also important when considering the performance of the game, as a complex interface will increase the performance overhead. Video games have tight performance requirements (low latency and high video frame rate) in order to provide a good experience to the players.

Furthermore, different types of video games (real-time strategy, first-person shooter, etc.) have different APIs as well as different performance requirements. Providing a generic interface that can be easily used by various types of games is an interesting challenge.

Evaluation. We have implemented a P2P game prototype and evaluated it using Intel SGX-capable processors. Our preliminary results show that our architecture provides cheat-resistance similar to the client-server architecture while being as scalable as the P2P architecture.

[*] Presenter and student