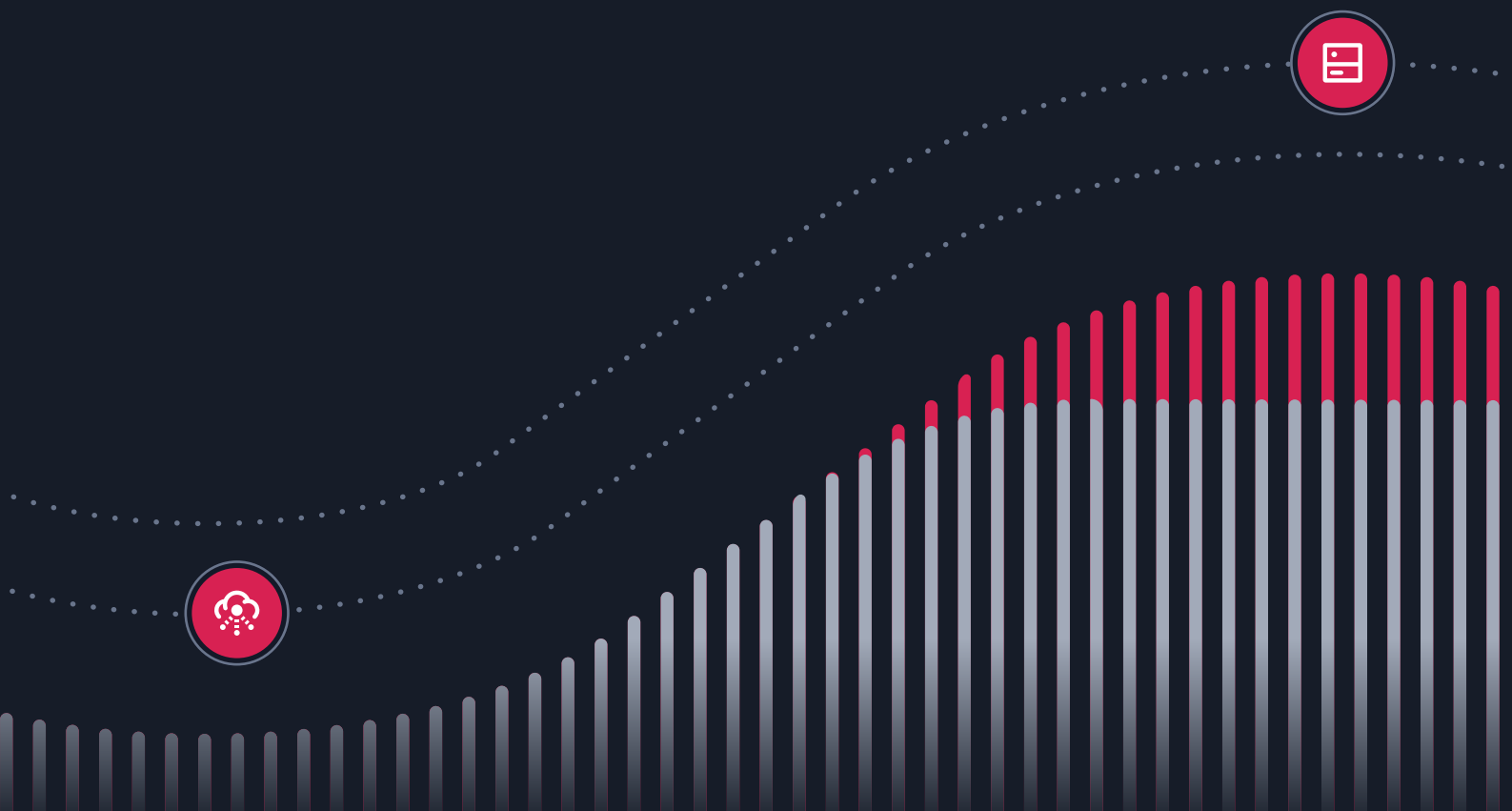
HYDRA

SABER INT. BACKEND TECHNOLOGY WHITE PAPER

Hydra Game Services

A single backend solution for all online aspects of the game



About Hydra

Developers face many technical hurdles related to a game's online experience. They commonly use different services and tools to overcome these challenges. Matchmaking, Dedicated Server Hosting, Game Analytics, Cross-Progression, Voice Chat, Game Mods… the list is seemingly endless.

Skilled developers know that introducing these things into their game can be complex and time-consuming. They must integrate each SDK, resolve potential conflicts & performance issues, write logic coordinating between services, and manage / configure / operate all services in different admin panels.

To solve this problem, Saber Interactive created a single united backend solution covering most of the online needs of the game: **Hydra Game Services**.

Hydra Game Services is a successor of Saber Backend – the corporate backend solution of Saber Interactive – which has proven its high reliability and efficiency being the backend of such AAA titles as Space Marine 2, WWZ, Quake Champions, SnowRunner, etc.

As time passed, we accumulated knowledge and expertise in building our own "ideal" backend, and decided to make it available for others, while reorganizing it from a project-oriented solution to a game services platform.

Features and Functionality

The main features inherited by **Hydra Game Services** from its predecessor are the following:

- Flexible, reliable, scalable backend infrastructure, based on a micro-service architecture
- Wide range of game and LiveOps services including:
  - Skill-based multi-factor Matchmaking
  - Customizable in-game Economy
  - In-game Friends
  - In-game Banner System for displaying messages to players
  - Leaderboards - seasons and tournaments
  - Challenges and Lime-Limited events - personal and community
  - Rewards and Entitlements - Twitch and Redeem campaigns
  - Player Statistics
  - Beta and Playtests
  - Modding System
  - Per-users Cloud Saves
- Powerful analytics with a rich set of predefined metrics and dashboards, extendable with custom events
- Advanced Diagnostics and health control tools, crashes, errors, client logs and KPI reports
- Handy online tool for customer support and game operations
- Smart and efficient system for title configuration on the Developers Portal
- 24/7 LiveOps team as well as second layer support team

This document describes the **Matchmaking** feature in detail.
For information on other features, please submit a request, and we will provide more details.

Services for Multiplayer

Everything you need to support cooperative and player-vs-player modes:

- Matchmaking and lobby for session-based games
- Server browsing
- Dedicated servers and networking library for peer-to-peer multiplayer
- Sessions, party, invites
- In-game friends, recent players, allowlists
- Voice chat, text chats

Matchmaking

1. Overview

Matchmaking (MM) is the process of combining players for a match.

This article provides an overview of the general flow of the MM process, its terms, entities, and features without technical details of their implementation.

2. Main Stages of Matchmaking Process

2. 1. Creation of Matchmaking Session

During the MM process, players can search for an appropriate online match among all other public matches (e.g. the “Find a Game” scenario). Or, they can connect each other directly (e.g. the “Create/Join a Game” scenario).

During this stage, your game will need to use the Presence and Push services.
The result of this stage is always the same — the creation of the group of players that will play together somewhere, the so-called MM Session.

2. 2. Creation of Game Session (Match)

After a group of players is formed, they can start a match. Depending on the game settings, there are three major options for this:

- **DS scheme via Hydra Game Services** — in this case, Hydra launches a separate Dedicated Server (DS) for this match. If you choose to use a DS scheme for your game, you will need your game client to work with the Session Control service, which will be responsible for launching and managing a DS instance. Uploading a build of this DS to the Admin Portal and the configuration of this DS there is also necessary.

- **DS scheme via third-party solution** — in this case, you use the third-party solution to start a DS for the selected players (using data on members of MM session received from the Hydra 5 backend). And, then, manage this DS. This process is out of the scope of Hydra documentation.
- **P2P scheme** — in this case, there is no selected dedicated server of the game. Every member of the session may act as a “server”, i.e. may control some entities or all of the entities of the multiplayer game. If you choose to use this scheme, the game session will not be present on the backend, it would be the responsibility of the game clients to coordinate on the multiplayer game.
  Also, see the Session Browsing Mode section.

However, the result of this stage is always the same — the creation of the group of players that will play together on the particular server, the so-called Game Session.

3. Server and Session Browsing

Server Browsing and **Session Browsing** are two optional modes that can be implemented as a part of the matchmaking process, as described in the subsections below.

Note that those modes integrate with different backend services and do not work together in most cases. The only case where session and server browsing can work at the same time is the managed flow for the DS sessions. In this flow, the creation of a session in the **Presence** component leads to the creation of a session in the **Session Control** component.

3. 1. Server Browsing Mode

The **Server Browsing** mode revitalizes an old-school online multiplayer concept, which becomes popular again. It is based on the client-server model, but servers are hosted by players themselves.

Particularly, in this mode:

- Players can host a standalone server, which registers on the backend via the **Session Control** service.
- Non-hosting players can view the list of such standalone servers (provided by the backend) and can connect to them to play.

This mode works only with the **Session Control** service, so you don't need to work with the **Presence** service when integrating it.

3. 2. Session Browsing Mode

In case a title does not work with dedicated servers in the matchmaking process, it might use the **Session Browsing** functionality of the **Presence** service. The purpose of session browsing is to provide players with data on all matchmaking sessions that they can join.

Having a list of matchmaking sessions with their details, players can browse them and decide which one to join.

This mode works only with the **Presence** service, so you don't need to work with the **Session Control** service when integrating it.

4. Hydra Services involved in Matchmaking Process

During the implementation of MM, you will need to work with the following Hydra services:

- **Presence** — the service is used for initiation of the MM process on the backend, working with Parties of players (invites, joins, etc.), and, in general, keeping track of the operational states of the player, Party, and MM session.
- **Push** — the service allows you to receive updates on the operational states of the player, Party, and MM session from the backend. Roughly speaking, you will initiate some MM operations using Presence and receive their results via Push Updates. Push Updates use Web Sockets.
- **Session Control** — this service is responsible for the management of the Session Control session (group of players within a match as a result of MM), starting and managing dedicated servers for these sessions.

You will need to use the Session Control service only if you choose to use a DS-based scheme for your multiplayer and choose to implement this scheme using Hydra. If you choose to use a P2P scheme, a DS is not needed. If you choose to use a third-party solution for the DS management, you will not use the Session Control service as well.

When you will be implementing the MM process for your game using one of the SDKs, you will use SDK components corresponding to the services listed above.

5. Typical Use Cases of Matchmaking

Your possible scenarios for utilization of Hydra for MM of your game are the following:

1. **You can use Hydra on all stages of MM**: for both the creation of MM Session and launching the DS for the selected players.
2. **You can use Hydra only for the creation of MM Session**. For example, this can be useful, if you want to use a P2P scheme or if you want to use a third-party solution for DS launch and management.
3. You can use a third-party solution for the creation of MM Session and **use Hydra only for launching a DS for the selected players**.

6. Push Updates: Tracking States of Matchmaking Entities

Working with MM in Hydra heavily relies on Push Updates. Their processing is an integral part of any implementation of the Hydra MM.

A Push Update is the update of the information that is sent by the backend (by the Push service) and received by the game client listening to Push Updates.

Using Push Updates, the game client will be able to keep track of its current states in the MM process and related events, including the results of the MM operations initiated on the client side.

There are 3 types of Push Updates:

- **User Push Update** — these push updates provide info about the states of the particular player in the MM process.
- **Session Push Update** — these push updates provide info on the MM session of the player and its properties and events (i.e. a session member joined, a session member left, etc.).

- **Party Push Update** — these push updates provide info about the Party of the player and its properties and events (i.e. a party member joined/left, the Party Owner (host) has changed, etc.).

The game client can subscribe to all types of these push updates. And, after receiving these push updates, can process their information to determine the necessary actions of the game client in MM flow.

For more details on data and properties within the push updates of particular types, see documentation of the particular SDK and gRPC API reference.

Usage of Push Updates allows Hydra to be more stable in the high load/CCU scenarios.

8. Session Owner

A MM session owner is the game client that initiated the MM session creation, or it can be one of the game clients selected by the backend for this role. From the point of view of general game client implementation, this session member is responsible for important parts of the game logic. For example, in the P2P scheme, a session owner is typically responsible for starting a server within its game client. Moreover, a session owner has additional rights and permissions on the content and properties of the MM session, i.e. only the session owner can change the settings of the MM session after its creation, set Search Variants for it, kick other members of the MM session, and so on.

Selection of the session owner based on the connection quality.
Below you can see a few examples of the functionality available to the session owners.
Note that this functionality is enabled by the **Presence** service.

- **Session Restart** — after a game session on a DS is finished, and the backend session is still active, there is an option to start a new game session with the same users. T
- **Change Team** — for the cases of team imbalance in game sessions, there is the MatchmakeSessionChangeTeam method. It provides the session owner with the option of manually allocating players to teams.

9. Party. Party Owner

A **Party** is a group of players that want to play together. In most games with MM, the players can form Parties from the UI of their game clients or the UI of their platform (e.g. by Steam Invites), invite each other to their Party, proceed with this Party to the matchmaking, leave their Party, etc.

The MM process guarantees that all members of the Party, which is formed by the players, will be matchmaked together, put into one MM session, and will be playing the same match. However, if a party member chooses to leave the Party, this player will be matchmaked separately (if not joining another party).

Each Party has a **Party Owner** who controls the Party. By analogy with a Session Owner, a Party Owner is a selected person that has more rights and permissions than any other member of the Party. Particularly, only the Party Owner can start the MM process for the Party, all other party members will just wait for the Party Owner to do so. Only the Party Owner can kick any other party member from the Party. And so on.

10. Search Variants

Search variants are criteria that a player specifies when looking for a match. MM uses these variants to find the matching opponents for the player.

For example, the desired map, difficulty, or other parameters of the game can be the search variants.

MM in **Hydra Game Services** is optimized against searching by these criteria and tries to find the intersection of them when grouping players into MM sessions. For example:

If:

- player 1 wants to play on **any map** and with **Medium difficulty**,
- player 2 wants to play on **map A or B** on **any difficulty**,
- player 3 wants to play on the **B map** with **any difficulty**,

then:
the system will try to join them into one MM session with parameters that match the search criteria of all players. I.e.:

- *If other parameters of these players used in MM (e.g. ratings) are the same*, all of them may be added to one MM session, where the **map is B** and the difficulty is **Medium**.

The amount of search variants used within MM is unlimited.

11. Playlists and Game Modes

Game Mode is a game type selected by the player (for example, deathmatch, CTF, team deathmatch, and so on).

However, from the configuration perspective, all parameters for the game mode are specified in a Playlist. A Playlist can correspond to single or multiple game modes. Matchmaking also works not with game modes, but with playlists.

For each game mode, its playlist has a lot of configuration parameters, necessary for correct matchmaking within this game mode. These configuration parameters can be different for different Playlists.

You can create and manage playlists for your game using the **Developers Web Portal**.

12. Distributions. Matchmaking Duration

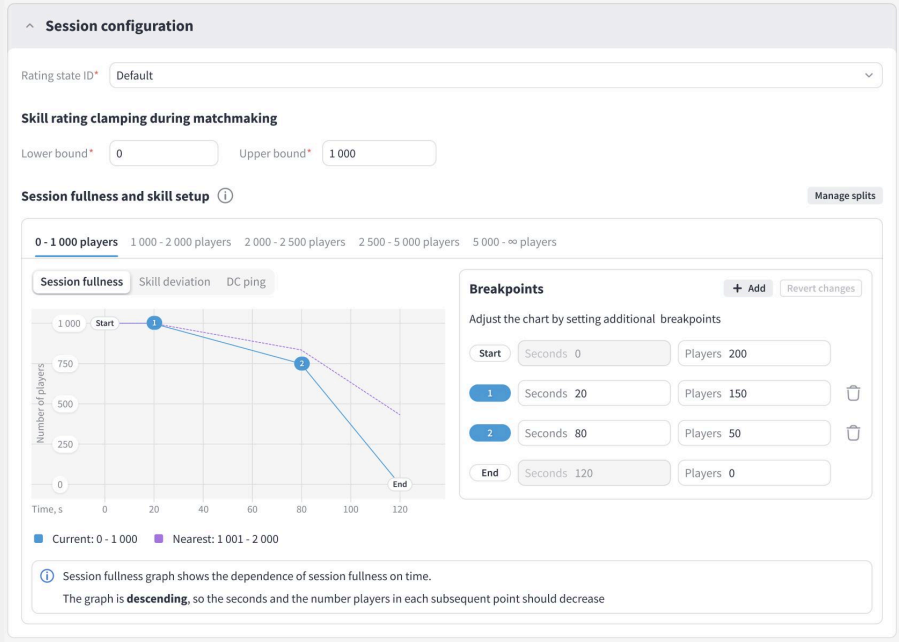
12. 1. MM Duration Setting

A **MM duration** is the time spent by the system to form a match. The value defines how much time it takes for the participating players to be matchmaked into a MM session. It is a very important metric of the MM process since it affects the user experience.

Using the **Developers Web Portal**, you can configure the dependence of **MM duration** from 3 parameters:

- **Session Fullness** — the fullness of the MM session (i.e. how many players are in the MM session, compared to the MM session with the maximum amount of players).
- **Skill Deviation** – the difference in skills of players (i.e. how different are ratings of players).
- **DC Ping** – an average ping deviation of the players (i.e. how different are ping values of players).

On the **Developers Web Portal**, in the **Matchmaking > Session Configuration** section of the Features tab, you can define these dependencies in the form of graphical splines, so-called **Distributions**.



12. 2. The Main Principle of the Distribution Dependency

The graphic splines for each parameter illustrate the dependency between the parameter's value and the time players spent in the matchmaking queue. As more time passes, the value relevant to start a session becomes lower.

In this section we consider an example of the **Session Fullness** parameter, but note that the principle explained here applies to all **MM duration** parameters, including the **Skill Deviation** and **DC Ping**.

On the screenshot below you can see the **Session fullness** spline. The session fullness in percent becomes lower as time passes. It means that the longer players wait in the matchmaking queue, the higher the chances that they will be joined to a session with a lower number of players. I.e., at the moment of joining the matchmaking process, a player can only be directed to a session that then becomes full (100% on the spline), as 60 seconds pass, the player can be directed to a session that is 60% full, and after 150 seconds, the player can be directed to any session with the minimum number of players allowed to start a session (0% on the spline).

Roughly speaking, the more complete the session, the faster it should proceed to a match. I.e., the closer the number of the players in MM session to the maximum amount of players in the session, the faster the MM process should end working with this MM session, stop waiting for other players to join, and treat this session as a formed MM session (for which a DS can be launched if this is configured).

By editing the spline points or adding the new ones, you can edit the relation between the parameter value and time.

Considering the example of the **Session fullness** again, you can move the **END** point of the spline to a greater value, thus setting the MM process to form a MM session for a longer duration when it has the minimum allowed amount of players. For example, it can be decided that players will be ok to wait for 200 seconds to join a more complete session and thus to have a better match quality.

This is just an illustrative example, and you can use splines to configure these dependencies more precisely, finding the balance between the time the players wait for the start of the match and the number of players in this match.

12. 3. Session Fullness value

Session fullness on the **Session Fullness** spline is calculated in percent, based on the comparison of the current number of players in the MM session with the MIN_PLAYERS (**Min Players**) and MAX_PLAYERS (**Max Players**) values set for the Playlist.

I.e., if the current number of players equals MIN_PLAYERS specified for the Playlist, this corresponds to 0% of Session fullness. If the current number of players equals MAX_PLAYERS — this corresponds to 100% of Session fullness.

For example, if the Playlist has MIN_PLAYERS=2 and MAX_PLAYERS=4, and the current amount of players in the MM session is 3 – the Session Fullness of this MM session will correspond to 50%.

12. 4. Relation to Playlist

In **Hydra Game Services**, when you configure a Playlist (see the Playlists and Game Modes section above), in one of its fields you select the **Session Settings** that will be used for it. And, these **Session Settings** contain the Distributions that will be used to form MM sessions within this Playlist. Thus, every Playlist is linked to the particular Distributions that will be used in the MM process for this Playlist.

This allows you to specify different **Distributions** for different **Playlists**. For example, you can specify more narrow criteria on skill difference for game modes targeted at professional players of your game (so, they will wait a little longer, but will have a greater chance to play with players with almost the same skill). And, make these criteria less strict for some game modes (to create game sessions faster). Or, if for some mode of the game (i.e. Playlist) the best user experience is achieved when the match contains, say 4 people (but the minimum is 2 players), you can create the Session Fullness distribution spline that will tell the MM process to wait longer, up to 3 minutes, until the MM session contains at least 4 players, and create minimally filled matches only if players are already waiting for these 3 minutes.

12. 5. Relation to CCU intervals

Moreover, on the **Developers Web Portal**, within Session Settings, all Distributions are assigned to particular intervals of CCU. By default, there is only one interval: from 0 to infinity (∞), but, by adding necessary breakpoints to this interval, you can split it into as many smaller intervals, as you need. For every such interval, a separate Distribution can be configured (and you will see the Distributions from adjacent intervals as dashed splines of a different colour near your spline).

This allows you to specify different Distributions for different sizes of a MM queue. For example, if the number of players in the matchmaking is low (and, correspondingly, it is hard to find matching players for the searching player), you can form a session faster even if the session is not full, or the players' skill or ping differs. And, when there is a high amount of matchmaking users, wait for full sessions and matching skills/ping values (since it will not be a long wait).

13. Join-In-Progress

Hydra Game Services supports Join-In-Progress (JIP) mechanism. I.e., if a MM session or a match has less than their maximum number of players, the system can add to those MM sessions or match some players who are currently searching for a game.

JIP can be enabled or disabled on a per-playlist basis (see the Playlists and Game Modes section above). For some game modes, where playing with particular opponents is very important, it is a good practice to disable JIP. For example, if somebody is playing an e-sportive “Duel” with a particular opponent, this person won’t be happy if, after the opponent leaves, the system joins another player to the match instead of ending the match and awarding victory to the remaining player. On the contrary, for arcade game modes, where the particular opponents are not so important, JIP is useful, since it allows to reduce the MM time and increase the session fullness.

14. Rules of Matchmaking Process

- Only players that have selected the same Playlist (Game Mode, e.g. “duel”, “deathmatch”, etc.) can be matchmaked with each other.
- Only players with the same version of the Game Client can be matchmaked with each other.
- The system does not start the match if the number of players in a MM session is less than the minimum number of players (MIN_PLAYERS amount) specified for this playlist. The maximum number of players in the MM session is also limited by the particular value (MAX_PLAYERS amount) specified for this playlist.
- The system maximizes the number of players in session (i.e. add players until the number of players in a MM session is close to MAX_PLAYERS amount).
  The dependence of the time of formation of the game session from the fullness of the session can be set up on the **Developers Web Portal**. If an existing game session is not full (e.g. someone has left), the system will try to add new matching players from the MM queue to it (Join-In-Progress feature).
- The system performs matchmaking for the player only within particular Data Centers selected by the player in the game client and tries to optimize this process. Particularly, the system starts searching for matching players/MM sessions within DCs with the lowest ping first, and then, if there are no matches there, proceeds to DCs with higher ping.
- Matchmaking is organized using multiple matchmaking queues of users. Users in one queue are matchmaked if lists of DCs selected by them intersect (i.e. they have at least one DC in common).

15. Dedicated Server for a Game Session

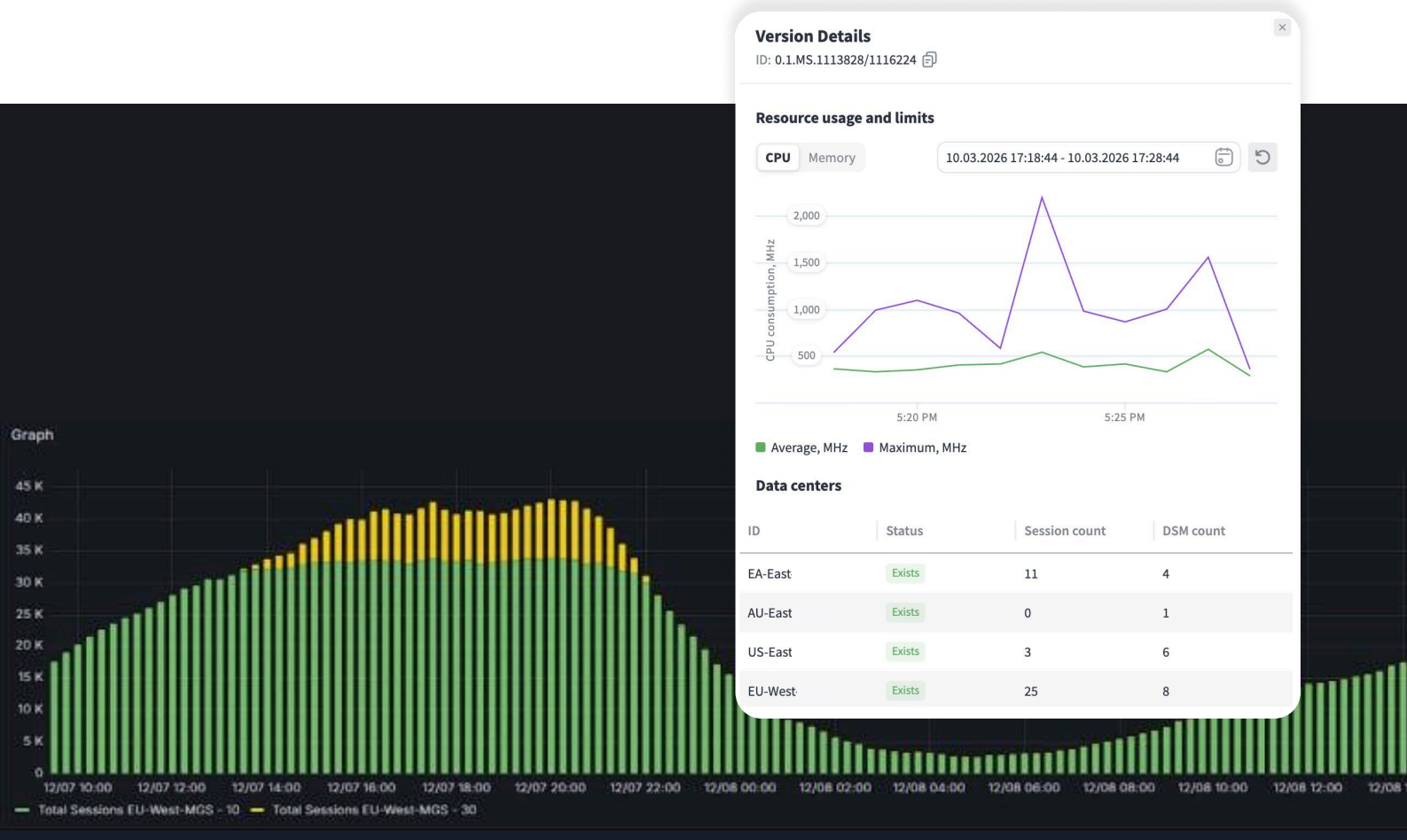
Dedicated Server (DS) – the specific server component that launches within a particular Data Center and hosts the multiplayer game.

More precisely, it is the version of your game that is specifically prepared for this purpose. It shall not contain any graphics and shall use only the minimum amount of resources necessary to host a game session, etc. It can be implemented on either Windows or Linux platform.

Typically, you will implement a Dedicated Server for your game along with the Game Client, and this implemented DS will also use the Hydra SDK (it's server-side methods).

Once a DS is created, a particular build of it needs to be uploaded to the Developers Web Portal and then configured there. Some configuration parameters are passed to the DS at the moment of its launch. These parameters can be presented as command-line arguments for the executable file of this DS. When a DS is configured, Hydra will be able to launch it (either automatically or upon request from the Game Client).

The game client and the DS are rather independent and can use Hydra SDKs for different languages. For example, the game client can use Hydra SDK for C# and the DS can use Hydra SDK for C++.



Data Center (DC) – hardware and infrastructure located in a particular region and used by Hydra to start DS instances. When mentioned here, it corresponds to a particular DC with certain connection endpoints (IP address + port, mainly).

There can be two types of providers within a DC:

- **Bare-metal provider** – provider with fixed capacity, where only a certain number of DSs can be hosted.
- **Auto-scalable provider** – provider with infrastructure that supports scaling virtual machines for the DS hosting based on the load, typically more expensive than bare-metal providers.

Depending on game logic, a DC for a player can be selected automatically, based on the player's ping, or players can select DCs that they want to use in the Game Client.

15. 1. Creation of Game Session

After a group of players is formed, they can start a match. Depending on the game implementation, there are three major options for this:

- **DS scheme via Hydra** – in this case, Hydra launches a separate **Dedicated Server (DS)** for this match. If you choose to use the DS scheme for your game, you will need your game client to work with the **Session Control** service, which will be responsible for launching and managing a DS instance. Uploading a build of this DS to the Admin Portal and its configuration there is also necessary. This case can be additionally split into two sub-cases:
  - **Managed flow sub-case** – when a DS is started automatically by Hydra.
  - **Unmanaged flow sub-case** (still with a Hydra-hosted DS) – when a DS is started manually (by request from one of the game clients).
- **DS scheme via third-party solution** – in this case, you use a third-party solution to start a DS for the selected players (using data on members of the MM session received from the Hydra backend). Then, manage this DS. This process is out of the scope of the Hydra documentation.

- **P2P scheme** – in this case, there is no selected dedicated server of the game. Every member of the session may act as a “server”, i.e., may control some entities or all of the entities of the multiplayer game. If you choose to use this scheme, the game session will not be present on the backend, it will be the responsibility of the game clients to coordinate on the multiplayer game.

This stage results in the creation of a group of players that will play together on a particular server, the so-called **Game Session**.

15. 2. Session Control: General Idea and Intended Use

The Session Control component is responsible for:

- Managing a Game Session, i.e., a group of players within a match created during a MM session.
- Receiving a game client request to start a Dedicated Server launched for a Game Session.
- Returning info on the launched Dedicated Server to the game client.
- The Session Control component also uses the Game Server Hosting component. It is responsible for the management of DS instances, their hosting, actual launch of a DS, as well as its closure and some other operations. However, this component is strictly internal and cannot be connected via gRPC API or Hydra SDKs. When the DS management is necessary, the Session Control component acts as the medium between the Game Server Hosting component and the game client (i.e., it passes all necessary info between them).

After the creation of the MM session, the core MM process in Hydra is finished. I.e., at this stage, you have a set of players that are united in the MM session and ready to start a game session (match). However, you still need to create a match for them.

As described in the Creation of Game Session (Match) section above, there are 3 variants of the further MM flow, starting with this point:

- **Managed flow**: Hydra automatically starts a DS for members of the MM session.

- **Unmanaged flow with DS via Hydra**: at the end of matchmaking, Hydra starts a DS after a request of the MM session owner.
- **Unmanaged P2P flow or Unmanaged flow with DS via a third-party solution**: at the end of matchmaking, a DS is not started by Hydra, neither automatically nor on request. If you choose to use the P2P scheme, a DS is not needed. If you choose to use a third-party solution for the DS management, this process will be out of the scope of Hydra.

To start a DS in the Managed and Unmanaged DS-based via Hydra flow, you will need your game client to work with the Session Control component.

If you want to implement the Unmanaged P2P flow, you won't need Session Control.

If you choose to use a third-party solution for the DS management, you won't need Session Control too (and the whole process will be performed without Hydra).

Thus:

- If you do not need a DS hosted by **Hydra Game Services** for your game, you will not need to work with Session Control.
- If you need a DS hosted by **Hydra Game Services** for your game, you will need to work with Session Control.
  Note that in this case, you will need Session Control regardless of the scenario you select to create a MM session. I.e., even if you decide to use a third-party solution to create a MM session (e.g., the Xbox Live platform for matchmaking) while using **Hydra Game Services** only for launching a DS, you will still need to work with Session Control for that.

This document describes the **Matchmaking** feature in detail.
For information on other features, please submit a request, and we will provide more details.

HYDRA

SABER

A single backend solution
for all online aspects of the game