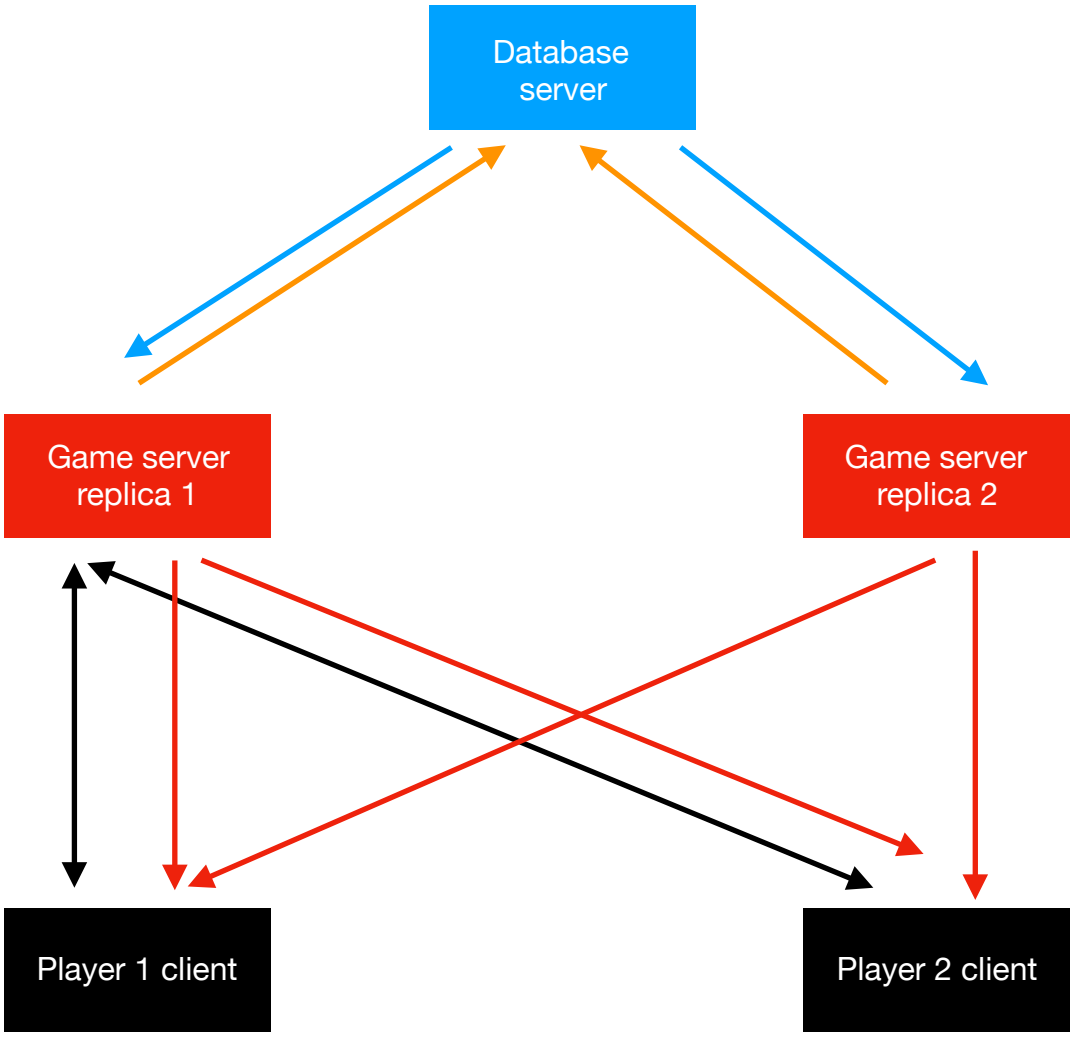

Black - game information (moves, status responses)
Red - heartbeats
Orange - checkpoints and logs
Blue - recovery checkpoints and logs

Fault model: crash fault
We believe that crash faults are the most likely to impact the system and be recovered from. The next most likely fault, in our opinion, would be a communication fault; however this would be hard to recover from in such a system, as losing a player in the game is not something that can be recovered from.

The game server is passively replicated, as recovering from a fault does not necessarily need to be fast in this use case. Players are not constantly sending data that would require a faster recovery. Thus we take advantage of the lesser complexity of a passively replicated system.

A heartbeat is used to determine the status of the game servers. The player clients, with a configurable parameter, send heartbeat requests to the game servers. If heartbeats are missed, the client enters a dormant state and attempts to reconnect. After three missed heartbeat periods, the client then attempts to connect to the other server which, upon receiving its first request from the client, begins a failover procedure. This scheme is used because of the

possibility of a short failure or network outage; in this case, it may be more efficient to simply continue using the original server than to failover to the other replica.

Logs are kept on the database on every move; this is because every move is important, and we cannot lose a single move. A configurable parameter determines how many moves are logged (after which a checkpoint will be generated). If the database goes down, then the server will not accept any moves (will kill itself). This is intentional, as we do not want any moves to not be logged.