Packet System:

I use a byte serialisation system to transmit data between clients and the server. This system uses enums to define the type of information that any one packet contains, then though the use of a Packet class inheritance structure, provides processing for that type of packet, and in the case of the server, redistribution of that packet to the other clients on the network.

Packets start on the client as data attached to a game object. This data is translated into a packet that can contain that datatype (ie, TransformPacket for an object transform), then it is serialised using C# MemoryStream(1) and BinaryWriter(2) if it is generated in the server or on a unity client, and if the packet is generated on an unreal client, it uses FMemoyrWriter(5). This produces either a C# array, or an Unreal TArray (6) (depending on where the packet is generated), which is then sent to the server. The server then checks the packets origin (which is stored at position 0 in the packet), before sending the packet to all other clients currently connected. This is done to minimise the number of excess packets sent to the client.

[Diagram of Packet structure, possibly related to a binary dump of a packet]

//What would I do if I continued extending the engine and server

**Cheating and Exploitation**

Cheating is an ongoing problem in online multiplayer games. Though the use of either software or hardware, unscrupulous actors will use cheats to gain an advantage over the other players in the game.

**Types of Cheating:**

**Aimbotting:**

Aimbotting is the process of automating the aiming process, allowing for a cheater to target enemy players with unnatural speed and accuracy (8). Aimbots can be implemented at either the software or hardware level, with software level aimbots using the same device as the client to manipulate the players aiming ability, whereas hardware aimbots use external computing using either other computers to intercept and manipulate mouse input, or moving the mouse using computer controlled robotics.

**Extra Sensory Perception (ESP)**

Extra Sensory Perception is a subclass of game hacks that reveal information to the player that they otherwise wouldn’t be able to see. This includes wallhacks, which shows enemies though walls though the use of an overlay and radar hacks, which shows enemy locations on a minimap component, allowing players to gain an advantage in having greater knowledge of their enemies positions.

ESP hacks can also provide cheaters with information specific to each other player, such as health total, weapons and consumable counts. In game, this allows cheaters to target players with particularly dangerous weapons or items first, or hunt down players with move valuable gear and equipment.

1. <https://learn.microsoft.com/en-us/dotnet/api/system.io.memorystream?view=net-8.0>
2. <https://learn.microsoft.com/en-us/dotnet/api/system.io.binarywriter?view=net-8.0>
3. <https://learn.microsoft.com/en-us/dotnet/api/system.io.binaryreader?view=net-8.0>
4. <https://dev.epicgames.com/documentation/en-us/unreal-engine/API/Runtime/Core/Serialization/FMemoryReader/__ctor>
5. <https://dev.epicgames.com/documentation/en-us/unreal-engine/API/Runtime/Core/Serialization/FMemoryWriter>
6. <https://dev.epicgames.com/documentation/en-us/unreal-engine/array-containers-in-unreal-engine>
7. Cheating: Gaining Advantage in Videogames, Mia Consalvo