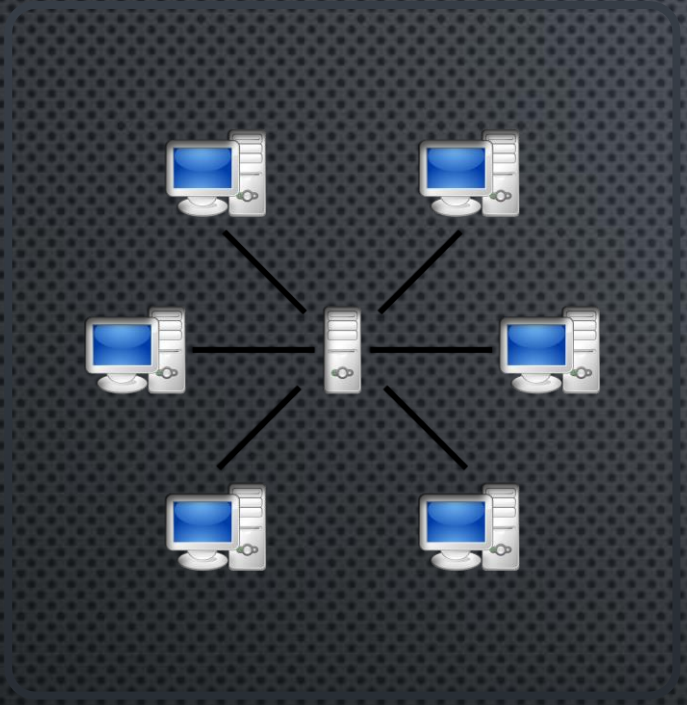
Pirates.io

Lee Elliott

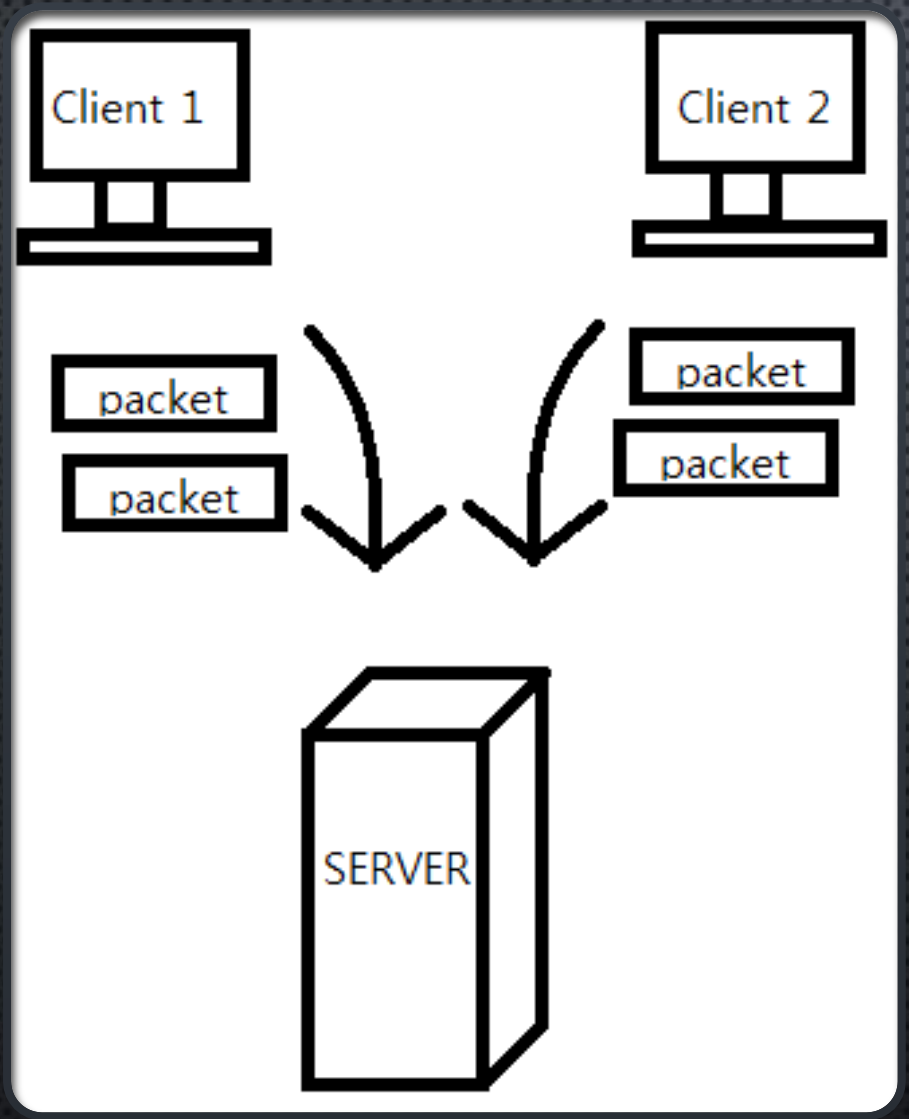
1602346

Network Architecture



- Server/client
  - Multiple clients connected
  - Maximum of 50 clients
  - Clients only have to send one message
- Transport Layer Protocol
  - User Datagram Protocol (UDP)

- Application Layer Protocol
  - SFML packets
  - Only 3 packet types
    - Update of status from client
      - Position, Size, Origin and Rotation
    - Update of “enemies” from client
      - Packet type 1
      - Number of connected clients
      - Loop to add “ID”, Position, Size, Origin and Rotation
    - Server full message
      - Packet type 2
      - String containing message
  - Packets sent every 0.2s



Network API

- Sockets
  - SFML UDP sockets
    - Similar to WinSock
    - WinSock was designed for the client/server model
    - Simple to implement
    - Portable so can be used on most platforms
  - UDP
    - Computationally less expensive than TCP
    - Tethered connection not necessary
    - UDP less reliable than TCP
    - Packet loss can be accounted for using other means

Network Structure and Integration

- Client / Server model
- Client sends single packet containing their status
- Client expects to receive one of two packet types
  - Packet type 1 provides client with status details of all connected clients, which the client uses to update their stored list
  - Packet type 2 provides a polite server full message, which stops client sending messages to the server but allows client to “play offline”
- Server sends one of two packet types
  - Packet type 1 sent to connected clients
  - Packet type 2 sent to prospective clients when server is full

Prediction and Interpolation

- Interpolation
  - Position data received from server is stored in a variable in the enemy.
  - In the update function, this is compared to the current position of the enemy
  - If these do not match, interpolation is used to move the enemy towards the server provided position
  - A similar method is used to control the rotation of the enemy

Evaluation

- Tested using Clumsy
  - Noticeable issues when using several functions at the same time
  - The interpolation does an adequate job of minimizing the effects of lag, dropped packets and throttle
  - Duplicate and out of order had no effect at all
  - Tamper is extremely effective at disrupting this application
- Interpolation of rotation method made the application seem jerky
  - This method was altered and an improvement was observed