3D3 – Computer Networks: Project 1

Student Name: Jamie Columb **Student ID:** 19335427

Preferred Focus and Use Case:

For this project I have decided to focus on applying a peer-to-peer architecture to a network of autonomous underwater vehicles (AUV's). The purpose of an AUV is to gather data from the ocean using on-board sensory equipment without the need for a human operator. The data gathered by the AUV is then sent to the surface using a client-server architecture. For more complex operations, it would be beneficial to have a group of AUV's communicating and sharing collected data with each other to improve coordination and efficiency. I propose that these AUV's should use a peer-to-peer network to intermittently communicate where each AUV is a peer capable of sharing and distributing files to the other AUV's in the network.

There are a number of constraints when dealing with underwater communications. The primary constraint is the limited range of communication. AUV's use acoustic modems to send and receive signals and due to the limited battery life, they have to limit the range in which they cand send these signals. To mitigate this, each AUV could use data from received signal from other AUV's to adjust their course to ensure they are always within range of the groups signals.

Protocol Overview:

This design will use a peer-to-peer architecture similar to BitTorrent with each AUV acting as a peer to share/distribute data with each other. Once an AUV collects data with its sensors, it will distribute the data file in chunks across all other peers in the network using TCP connections. In return, other peers in the network will share chunks of data files that they gathered. Once an AUV has the entire file from received chunks, it has no option but to stay in the network and continue sharing chunks with other AUV's. Eventually each AUV will have full data files gathered from neighbouring peers so that if one AUV happened to get lost or deactivate, the data it gathered could still be obtained from a neighbouring AUV.

One AUV will also act as a tracker as well as a peer. Initially each peer will register itself with the tracker and in return, the tracker will give the peer the IP addresses of all other peers in the network. Once all peers in the network registered, they will periodically send information to the tracker, informing the tracker if the peer is connected and what chunks they have stored.

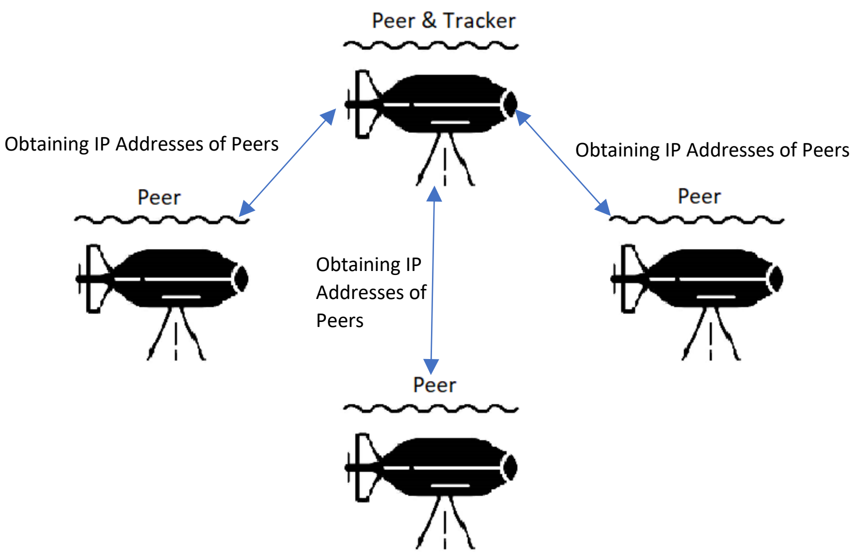

Fig.1 AUV's Registering to the Tracker

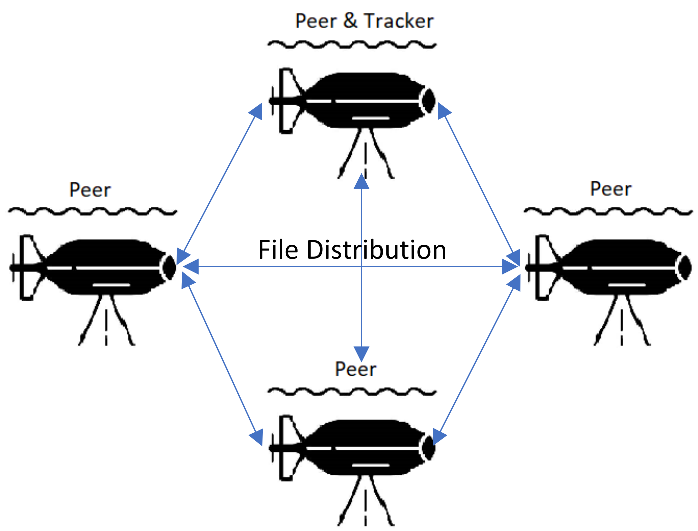

Fig.2 File Distribution Between AUV's Using P2P Architecture

Communication Model:

Using the peer-to-peer communication model, any device could potentially register with the tracker and obtain file chunks from the AUV's already participating in the network. For this particular use case, if a group of AUV's were to be used for military applications, it would be a security risk if an unauthorised device were to connect and obtain confidential data.

To add a layer of security to this network, each AUV will have a recognisable key. When an AUV tries to register with the tracker and gain access to the network it will be asked to provide this key. Any unauthorised device that cannot provide a key or provides the wrong key will not be given the IP addresses on each peer in the network and will therefore not be able to participate in the file distribution.

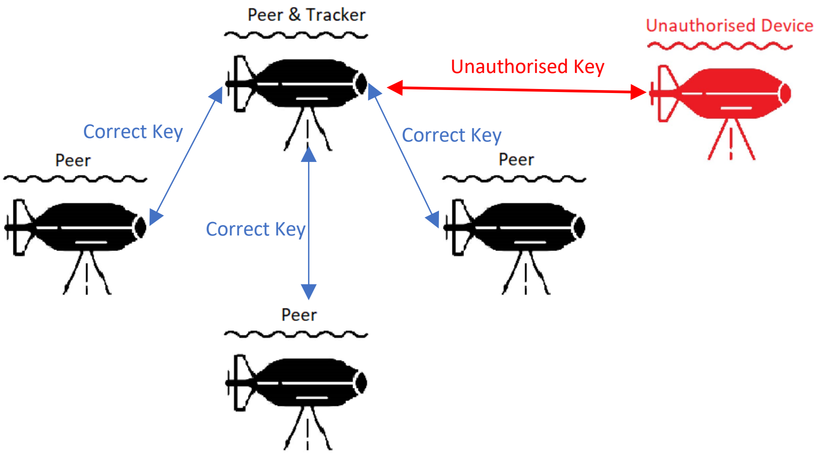

Fig.3 Unauthorised Device Attempting to Connect to Network

Module Descriptions:

Initially, the only interaction is between the device wishing to connect to the peer-to-peer network and the tracker. The device registers itself with the tracker and provides the correct key. In return, the tracker gives the device a list of all the peers in the network. Once this device is connected to the network it becomes a peer, and since any device that connects is previously authorised, they all have equal trust boundaries.

Once the AUV begins to gather data, it can start to share this data in chunks with its neighbouring peers using TCP connections. The AUV will also share location data so that if one AUV begins to veer off course, it knows the location of the rest of the group and can adjust its movements. This is why its beneficial to have equal trust boundaries so that important data can be shared as fast as possible.

Summary of Algorithms:

The first algorithm enables the peer to join the network. It will do this by assigning each peer a global unique identifier (GUID). This is a number that is unique to each peer and will be used to identify where the new peer is sending data.

A socket algorithm will be necessary for a peer to send and receive data from other peers in the network. Since it will be used in a peer-to-peer architecture, each socket will have to both reach out to another peer's socket while also listening at a port for incoming data.

A useful library for implementing peer-to-peer protocols is the Boost.Asio library for c++. This provides support for TCP connections and socket I/O streams which I have mentioned will be used in this network application.

Software Development Practices:

Test driven development will be used when programming this network. This is a comprehensive way of testing your code by first writing code that will satisfy all known test cases in your system before fully developing the code. Defensive programming will also be employed to ensure application will still operate while experiencing unforeseen issues, preventing the network to completely fail.