UNIVERSITY OF MORATUWA

Department of Computer Science and Engineering

CS4262 - Distributed Systems

Project Design Document

Distributed File Sharing System

S. Arunan 140045E

T. Inthirakumaaran - 140235L

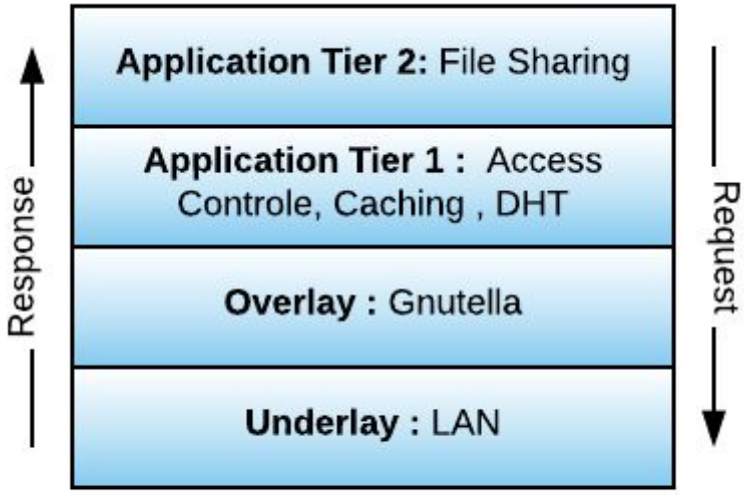
J.M. Isham - 140236P

T. Thuvarakan - 140631X

Proposed Topology

The higher level architecture will be a multi tier architecture with four tier, where layers are defined in cohesive manner and provide well defined interfaces to communicate with other layers.

1. **Application Tier 2 :** This layer provides the interface for the user to interact and provides file sharing services among client
2. **Application TIer 1:** This layer provides the access control to overlay network and handles the file downloading procedure.
3. **Overlay :** This layer defines an unstructured non deterministic protocol to communicate with other peers in the network.
4. **Underlay :** This layer interacts with network hardware and provide an abstract interface for the upper layers to use these hardwares.



Higher Level Architecture - Multi Tier

System Level Architecture - Gnutella

Gnutella is a very basic unstructured nondeterministic peer-peer model well suited for distributed file sharing system. Where each peer is a gnode. In our case, the number of peers in the network is going to be fairly small, hence flooding the network with the query request is not expected to fully utilize the bandwidth.

Apart from the protocols given in the problem description to communicate with the bootstrap server, we are using the following message structures to communicate with the peers.

Ping - Used to actively check if the neighbors on the network. A peer receiving a Ping descriptor is expected to respond with one or more Pong descriptors. It does not use message payload.

Pong - A Gnode sends a Pong message like an answer to a Ping message. The pong message Includes the address of a connected Gnutella peer and also the information regarding the amount of data it is making available to the network.

Query - The primary mechanism for searching the Gnutella network. It is used by a gnode to query other gnodes by a certain resource A server receiving a Query descriptor will respond with a QueryHit if a match is found against its local data set.

QueryHit : This is the answer returned by the gnode for a query. This message consists of the information about the information to share the file which matches the query such as IP address, port number, and network bandwidth of responding peer, number of results and result set.

Joining the network

The following pseudocode explain the process of a gnode joining the network

1. Once a gnode is created, the node sends a register request to the Bootstrap server.
2. The bootstrap server responses with available neighbor gnodes to connect.
3. The gnode sends ping to the neighbors.
4. Neighbors responses with pong. (if the # neighbours <4)
5. Gnodes updates the routing table
6. After joining the network the gnodes periodically sends pings and pong to let the neighbors know the about existence of themselves.

Search for a file

The following pseudocode explains the process of querying within the gnutella network.

1. Create search tokens by splitting the search term by space .
2. Search in the node itself for matching
3. Send each token as a query to randomly selected neighbors with hop limit.
4. If there is a matching in a node send back a QueryHit.
5. Repete 3,4 until the hop limit reached

Downloading a file

After the QueryHit response, the query gnode connects directly to the response gnode and uses the simplified version of hypertext transfer protocol (HTTP) to retrieve the file using the returned uniform resource locator. File download always takes place out of the network.

Leaving the network

The following pseudocode explains the process of a node doing **a graceful departure** from the network.

1. The node will unregister itself from the bootstrap server.
2. It will send a message to its neighbours that it is leaving.
3. The neighbours (based on their number of neighbours) will decide whether to request from the bootstrap for new neighbours. (ie (# neighbours >= 2) ? continue() : request())

The following pseudocode explains the process of a node doing **a sudden departure** from the network.

1. The nodes will be sending periodical pings to its neighbours to check whether they are active.
2. If they don't receive pong messages they will register with new neighbours.

Performance Analysis

Performance analysis is a main phase of this project, since these results justifies the design decisions and applicability of gnutella for distributed file sharing system. The main concern of the performance is latency. Even though this factor highly depends on the network hardware, we still may use this to compare the system with other systems such as chord and kazaa. We only have to consider the query latency in this scenario since the file downloading part does not come under gnutella protocol.

The latency will be measured with respective to the client node, when sending the query, the current time will be added to the payload. When a node reply with a queryHit, it will add the same time that came with the query and return it. Then once a queryHit arrives in the initial client, the client can measure the overall latency of the search lifecycle.

Hop counts are another possible performance measure. The number of hops can be determined by subtracting the allowed hops and remaining hops.