# Mesh Protocol

## Server

* The server constantly checks for new nodes and that all of the known ones are still in range (using the req broadcast).
* When it receives a reply, it checks if it has already been registered.
  + If a node is unique, it is registered and an IDOK message is sent with the relevant node ID.
  + If a node has already been registered with that ID an IDNOTOK is sent. To differentiate between a new node registering and the same node’s message being bounced around the network, the content field will hold another random number which will be different from both nodes. The IDNOTOK message’s content field will hold the random number received by the rejected node.
* When it needs to send a message to a node it uses flooding to propagate the packet through the network.
* Each packet has:
  + Target address – node that the message should be sent to
  + Time to live – how many hops before the packet expires
  + Packet ID – to avoid the end receiver processing the packet twice
  + Content – whatever you want to send (0’s or 1’s for turning LEDs on and off)
  + Timestamp – Time created to allow packets to expire
* There can only be one server in a network and it’s ID is zero.

## Nodes

* When it receives the general broadcast from the server it replies with an acknowledgement that includes a randomly generated node ID and a randomly generated content field. The server will reply with an IDOK or IDNOTOK. For IDNOTOK with the same random number in the content field the node will resend a new, randomly generated ID and content to be checked again.
  + The randomly generated ID can be any 8-bit????????? number except zero
* Flooding is used to send the packet throughout the network:
  + When it receives a packet, it checks the target ID against its own node ID.
  + If there is a match the packet is processed and an acknowledge reply is sent with a destination of zero.
  + If the target ID does not match it next checks the time-to-live field. If this is not zero it decrements it by one and rebroadcasts the packet. If it is zero it does nothing.
* Server broadcasts and acknowledge replies also use this flooding technique.

## Packet Structure

* Message Types:
  + Req – To discover nodes and ensure current nodes are still available.
  + Ack – Respond to ping message with a node ID. This ID is either a requested ID or an ID already registered with the server.
  + LED – Toggles LED on target device.
  + IDOK – Sent by server to indicate to a node that the requested ID is valid.
  + IDNOTOK – Sent by server to indicate to a node that the requested ID is not valid and a new one must be generated.
* Packet structure:
  + Destination ID – Location to send the message to.
  + Source ID – Location that message originates from.
  + Message Type – type of message
  + Time-to-live – Number of hops before the message expires.
  + Packet ID – unique packet ID
  + Content – for node registration this holds a second random number to differentiate between nodes requesting the same ID.
  + Timestamp – Time when the packet was sent in seconds from the Unix epoch (1 January 1970 00:00:00). Packets are ignored ##### seconds after this.

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Source ID | Destination ID | Content |
| Ping | 0 | - | - |
| Ack | Node ID/ requested node ID | 0 | For request ID:  Random number to differentiate from same ID requests from other nodes |
| LED | 0 | Node ID | - |
| IDOK/IDNOTOK | 0 | Node ID/ requested node ID | Content field of received ack |

* Field values for the different message types (TTL is constant and packet ID is the counter from the node/server):