1 User Story

Title: UPnP Support to Open Router Ports.

Description: As a user I want my message to be forwarded to or from the gateway router successfully, even if I am in a Local Area Network (LAN). Life is a bitch, fuck it or leave it.

Acceptance criteria: The incorporated UPnP protocol should function successfully for private network users. It should open incoming ports on gateway routers. And one UPnP network address translation is done with the acknowledgment packet from a specific gateway router.

Title: Add to "defaultKnownNodes" Support

Description: As a user I want to add certain IP address to the defaultKnownNode set, then the message would be transmitted within a more robust P2P network.

Acceptance criteria: Read IP information from "appdata" field (defined in defaultKnownNodes.py), and add to the allKnownNodes list. The success of this operation would be reflected in the output (as a display).

Title: Optional Chan Member Authentication Support

Description: As a user I want to have the optional authentication function for my chan members, and this would avoid unnecessary workload since they are trusted.

Acceptance criteria: Implement optional authentication in network level. The success is an obvious workload decrease for chan members, i.e., chan members do not have to provide POW.

2 UPnP Feature How-To

In total the implementation of UPnP feature would take 1 to 2 weeks, testing and documentation included.

- Background Investigation. Refer to UPnP Forum official documents, FAQ and specification of miniupnp project. Since there are many well-established UPnP libraries, there is no need to re-invent the wheel from the very beginning.
- Code Analysis. Refer to miniupnp project's source code (written in C) and other python-based UPnP libraries. Find out their advantages then sketch up the Bitmessage UPnP architecture.
- Code Implementation and Refinement. Implement the UPnP feature then debug with peer review.