Release Summary

Decentralized Internet

The Distributors

6/5/23

<u>Key User Stories and Acceptance Criteria</u>:

User Story 1: As a user, I want to be able to instantiate a network between myself and other users without having to rely on a centralized third party through which all of the network's data would pass.

Acceptance Criteria:

1. User can create a new network from the application interface.
2. User can invite others to join the created network using their IP.
3. Data transmitted within the network is not passing through a central server or third party.

User Story 2: As a user, I want to be able to save data to the network and access that data later so that I can use the network for storage.

Acceptance Criteria:

1. User can upload data files to the network from their local system
2. Uploaded data is successfully saved on the network.
3. User can view a list of all data saved on the network.
4. User can retrieve and access previously saved data from the network.

User Story 3: As a user, I want other users in the network to be able to access data that I have written so that I can use the network for sharing data.

Acceptance Criteria:

1. Users within the network can view a list of all shared data.
2. Users can access and download shared data from the network.

Known Problems:

Major Bugs: the current implementation has no major bugs

Minor Bugs: due to some heavy weight backend computation and database persistence, there occasionally may occur database write issues, specifically when simulating the entire network on a single machine via a Docker container. We expect the program to work as intended for small network sizes.
There are ocaisanal hangups (500 error) when viewing files. This happens when files are larger than a simple webpage (images, etc.); typically this happens because the image still needs time to be loaded, so a simple refresh will typically help with this issue

Missing functionality: File deletion, changing the structure of the network, optimizations for reducing the scope of knowledge each node has about the entire network. Peer 2 peer functionality

Design Shortcuts: frequent conversion between binary data and Python strings has added overhead to the system

Product Backlog:

1. Improved Network snooping functionality:
   We want to limit the amount of information each node has about the entire network. For example for very large networks it would be inefficient to store the entire list of nodes on each node. Instead you can implement a searching protocol to handle node discovery.
2. Dynamically changing networks:
   In P2P systems nodes leave and join at all times, and data distribution needs to account for that to constantly keep all data highly available
3. Leaving and joining active networks:
   As a user we may want to be a part of many different networks and switch between them arbitrarily.