Interface Graphique pour l'application PEAR :

Après avoir étudié la documentation du protocole Tox, nous nous sommes tournés vers son implémentation pour créer une application fonctionnelle. Cependant, comme nous ne pouvions pas tous travailler en même temps sur cette partie du projet, nous avons jugé utile de faire une interface graphique afin de pouvoir tout d'abord visualiser les différents utilisateurs dans le réseau, puis, par la suite, peut-être développer une véritable application de messagerie, comme Messenger.

Dans cette partie, nous faisons donc une interface graphique pour pouvoir visualiser la manière dont l'application fonctionne. Pear permet de connecter directement deux utilisateurs en parcourant le réseau formé par l'ensemble de tous les utilisateurs. Les utilisateurs possèdent une table de hashage qui permet de référencer certain nœud connu. Un utilisateur peut alors se connecter (envoyer des paquets) à n'importe quel nœud du réseau en passant par un ou plusieurs utilisateurs intermédiaires, afin d'obtenir son adresse IP et son port pour communiquer.

Détail et utilisation de l'application web :

Dans cette représentation graphique, une première page sert à lister tous les nœuds du réseau dans un tableau. On peut aussi connaître leurs nombres totaux de connexions et leurs différents utilisateurs présents dans leur propre table de hashage.

De plus une représentation graphique permet de visualiser ces connexions avec des liens physiques pondérés par leur poids (qui correspond au nombre total de connexions qu'ils possèdent). Cela crée un graphique avec des forces de type « gravité » qui permet d'observer les points centraux (plus riches en informations) du réseau.

Pour naviguer sur la page web et passer d'une visualisation à une autre, on peut utiliser le menu de gauche. Un bouton permet aussi d'alterner entre les visualisations 3D et 2D pour les graphiques. Les graphiques se manipulent intuitivement en déplaçant les nœuds.

Données d'entrée :

L'application graphique prend en entrée un exemple de réseau et d'utilisateurs. Nous avions initialement prévu de faire une API pour que la page web puisse communiquer avec le programme en java et permettre une mise à jour automatique. Malheureusement, par manque de temps, nous n'avons pas pu.

Choix des langages :

Nous avons utilisé les langages web habituel : html, JavaScript et Css (et un peu de scss pour aller plus vite). En effet, le but était d'obtenir une application de type « cross Platform » faites à partir de nos connaissances. Le JavaScript permet de charger les graphiques et de gérer les différents boutons. Pour les graphiques nous avons pris une librairie pour le graphique 2D et 3D.

Améliorations possibles :

Si nous avions eu plus de temps pour développer l'interface graphique, nous aurions pu implémenter l'API qui permettra de visualiser le réseau en temps réel. Nous aurions également aimé faire une véritable application de messagerie.

Bilan :

Dans cette partie, des améliorations étaient possibles en accord avec l'avancement de la partie « technique » du projet. Cela a été contraignant et n'a pas pu permettre la finalisation de cette partie. Le développement de l'API aurait été un plus pour expliquer comment fonctionne le réseau mais il aurait fallu un travail non négligeable pour faire une application de messagerie fonctionnelle, au niveau interface graphique, mais surtout au niveau technique.