PyTox With Custom Notification System

It's common for someone to wish to receive messages for certain things but not for others. It's okay if your wife is pregnant, and you don't mind if the message that she's going into labor interrupts your meeting. There may be times when you're on call for work, but don't want anything to interrupt your free time. In some cases, a binary on/off notification system is insufficient to meet the needs of a user.

In light of this, a messaging system with customizable notifications might prove to be beneficial. Basically, this would be a form of do not disturb mode, where the user could choose what notifications they are interested in receiving and what notifications they do not.

In order to do this, they will be able to create a series of whitelists and blacklists for the sources of notifications from which they will receive notifications. At any given time, they can choose which will be active. Additionally, the system can automatically reply to blocked messages with a piece of text the user adds or the system will generate a message based on the client configuration. As a result, the user can let others know that they are busy and when they are likely to reply to the message.

However, person-to-person notification settings are not always granular enough. There are times when you want to be interrupted by certain things and not by others. For this reason, the notification settings should allow individual users to override the general silence.

The most reliable solution is to use peer-to-peer systems. Due to the expense of having dedicated servers and the desire to avoid single points of failure. It is true that peer-to-peer systems are very complex, and they do require a certain number of hosts for discovery. Fortunately there is a solution that handles both algorithms and initial hosts, as well as a testable user base: Project Tox. Tox offers an end-to-end encrypted peer-to-peer messaging system under the GNU General Public License v3.0. A library for interfacing with this network written in C, called Tox-core is freely available. In addition there is a python library that brings Tox-core into python, called PyTox.

Tox is very client-centric and so creating the client will not be a trivial task, but it will not have the difficulty of writing algorithms and creating a distributed hash table from scratch either.

In order to use these tools, we simply need to download the libraries on our computers. The development process does not require any additional resources, but we may want to set up some type of version control system for ease of development.

Tentative Schedule

Thursday, October 6th: UI mockup and data structures/classes being used planned out.

Monday, October 31st: Can interface with Tox network, send/receive messages, WIP UI for testing purposes

Monday, November 14th: UI Elements in place, core Tox functionality in UI working

Monday, November 28th: All elements functional

Remaining time dedicated to bug fixing and delays

Group Members

Ethan Moore: eemvzm@umsystem.edu

Pavan Raj Dasari: mdm77@umsystem.edu

Sai Ganesh Pamarti: spbd3@umsystem.edu