

# DarkChat: a light weight, [almost] server-free, push-based robust p2p network with a chat payload

Nathan Griffith and Ahmet Aktay

May 12, 2010

## 1 Abstract

The purpose of this project is to create a protocol for clients on a network to function with very little dependence on a server. In the case of *DarkChat*, the clients function as 'chat' clients. The clients are able to communicate with each other information such as 'online'/'offline' state and lists of contacts, with little-to-no interaction with the server.

While *DarkChat* relies on a multi-threaded approach, and also makes no use of authentication, implementation decisions such as this have been made independently of the underlying protocol. As such, the protocol could be implemented in any number of ways, and this is meant as merely one example.

## 2 Clients and Servers

The 'Client' class has been built as a wrapper for the messaging and interface classes, putting them all together into one usable chat client. Note that the term 'client' is slightly misleading, as with the inclusion of a simple flag, a client can function as a server with very few implementation changes. It is possible that this could be reduced even further, such that servers could simply act as 'super-clients' which store information about the entire network, instead of about only their nearby edges and nodes.

In either case, the only purpose of the sever is to act as a secondary or backup repository for storing information about the network. Once the network is fully operational, the server is not needed, as long as enough clients remain online that one cluster of edges does not become separated from any other.

## 2.1 Usage: Parameters

When executing the client, a number of parameters can be passed. With the exception of ‘username’ (‘-u’), all parameters will revert to default values if none is specified:

```
-p [port number, for accepting incoming communication]
-t [the default number of listening threads]
-u [the username of the client]
-sip [ip address of known server]
-sp [port number of known server]
-s (server flag, specifying that this instance of DarkChat is a server)
```

Note that there is currently no reliable centralized server, so in the actual case of starting a network running from scratch, it would be best to ensure that a server is running, and specify the server ip and port for each client. Once the clients are online and have contacted each other, one need only use a macro to contact a currently online client.

## 2.2 Usage: Macros

Once the client has been started, a number of macros have been made available to give the user control:

```
\help
-Display this dialog
\chat <username>
-Switch to conversation with <username>
\users
-See a list of KNOWN users (online or off)
\online
-See a list of known, ONLINE users
\offline
-See a list of known, OFFLINE users
\add <username>
-Attempt to add username to list of known users
\ping <ip> <port>
-Attempt to ping the ip at the given port
-If no port is specified, the current instance’s incoming will be used
\quit or \exit
-Leave the program gracefully
```

Note that these are made available before the client has necessarily made successful contact with a server, allowing one to use the ‘\ping’ macro to manually connect to another client or server.

Once a connection has been made, information about offline and online users should propagate across the network and to the client, at which point the macros ‘\online’, ‘\offline’, and ‘\users’ will provide useful information.

To communicate with an online user, one need only use the ‘\chat’ macro followed by the user name to start a new chat session with that user. From that point on, any line of text that is entered and doesn’t start with a macro ‘\’ will be sent to all online sessions associated with the desired username.

## References

- [1] Atul Adya, John Dunagany, Alec Wolman, *PRS: A Reusable Abstraction for Scaling Out Middle Tiers in the Datacenter*. Microsoft Corporation, Microsoft Research, 2008.