Bit Chat: A Peer-to-Peer Instant Messenger

Shreyas Zare
shreyas@technitium.com
https://technitium.com

December 20, 2015

Abstract. Bit Chat is a peer-to-peer instant messaging concept, allowing one-to-one or group instant messaging & file sharing, using a decentralized peer-to-peer protocol, end-to-end encryption and a trust based peer identification system. Users communicate by forming a full mesh network topology after discovering peer IP addresses to connect using Bit Torrent trackers and Distributed Hash Table (DHT). The system's purpose is to have a secure instant messaging platform for privacy and security.

1. Introduction

Bit Chat is a secure, peer-to-peer, open source instant messenger designed to provide end-to-end encryption that can be used over Internet and private LAN networks for instant messaging and file sharing. The implementation allows ubiquitous and automatic encryption available to all users without them needing to understand the complexities involved.

Most instant messaging platforms use a centralized architecture allowing users to connect to the network and exchange messages while the service provider of such messaging platforms can collect metadata and even retain copy of messages (when end-to-end encryption is not available). Even while having an end-to-end encryption protocol support, the user has to trust and depend upon the messaging service provider for initial contact and key exchange, and still give away metadata such as when and with whom the user chats with.

However, by being a peer-to-peer messaging platform, Bit Chat users connect to each other directly to exchange messages over an end-to-end encrypted channel. Each Bit Chat user needs to do a onetime registration for an email address validated digital profile certificate which is used by the peer-to-peer protocol to authenticate peers on both sides of the channel.

Bit Chat does not have any type of contact management system to invite a user to chat. A user will have to make initial contact to the peer via an email or any other communication channel available, and provide the chat group name or email address to be able to chat. Bit Chat uses an algorithm to generate a network ID based on the chat group name or peer email address, and an optional shared secret. This

network ID is used as an identifier for finding peer IP addresses using Bit Torrent trackers and Distributed Hash Table (DHT). Not having to manage contacts of each peer helps in reducing the metadata footprint at the messaging service provider end.

Since there is no centralized mechanism for message routing, a user can exchange messages with a peer only when that peer is online, that is, there is no offline messaging facility available. Similarly, there is no method to find out if a user left the chat forever or went offline.

Bit Chat is open source and source code is available on GitHub [1] under GNU GPLv3 License [2].

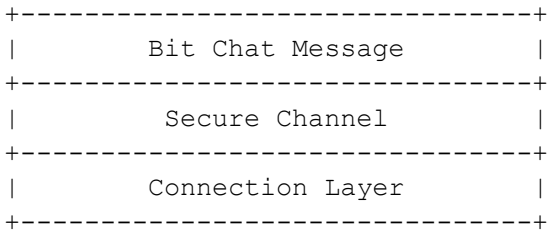
2. Peer-to-Peer Protocol

The Bit Chat peer-to-peer protocol works over TCP protocol for making direct connections between peers. Each peer acts both as a client and server, accepting incoming connections and making outbound connections. A Peer listens on any available random TCP port for accepting incoming connections and advertises both IP address and port number to be discovered by other peers.

A peer can also act as a TCP relay and allow peers, who are behind a Network Address Translation (NAT) or Firewall device, to accept incoming virtual TCP connection.

The peer-to-peer protocol in itself is a stack of three different protocol layers. The Connection Layer forms the base of the protocol responsible to make and accept TCP connections from peers, allows creating virtual channel streams, and provides TCP relay functionality. Virtual channel streams feature allows a single TCP connection to be split into multiple virtual connections which are identified by a channel name. These virtual channel streams are further secured using the secure channel protocol. These secured virtual channel streams provide end-to-end encrypted tunnel from one peer to another for transporting Bit Chat messages. Each Bit Chat network needs a separate secure channel stream connection for each peer in the chat group forming a full mesh network topology.

The connection initiating peer requires opening a virtual data channel to a Bit Chat network identified by network ID. This network ID is generated by each peer using the chat group name or peer email address, and an optional shared secret. A channel name is further generated using peer ID parameters from the connection handshake protocol and the network ID. This channel name is then used to open the virtual data channel, secured by the Secure Channel protocol, required to exchange Bit Chat messages.



Protocol Stack

2.1. Connection Layer

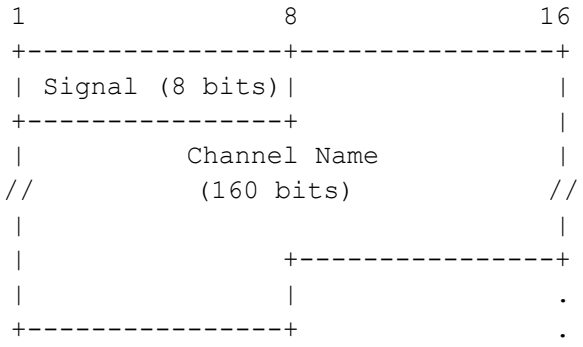
Connection layer uses signal frame for control and to initiate virtual channel streams. Data frame are used for transferring data in virtual channel stream.

When a peer connects to another peer over TCP connection, a connection handshake protocol is initiated. The handshake protocol is designed to detect duplicate TCP connections using peer ID. A peer ID (20 bytes) is a randomly generated identifier by each peer during application startup. The service port number is the TCP port number that the client is listening on for accepting incoming connections.

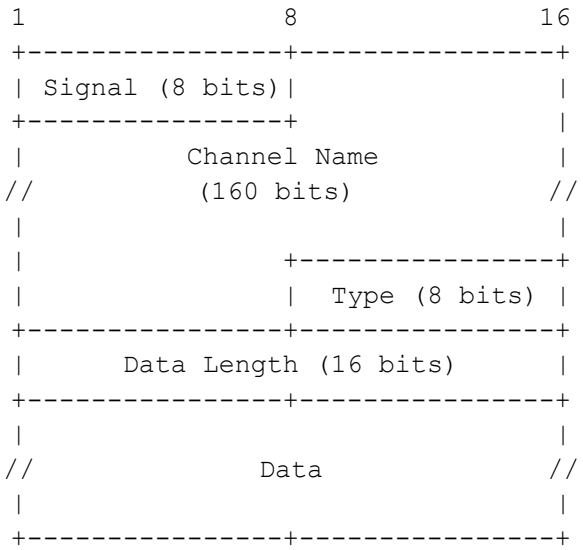
```
                              SERVER       CLIENT
------------------------------------------------------------------------------
                                    <---- version, service port,
                                          client peer id

status (ok/cancel), server peer id  ---->
```

Connection Handshake Protocol



Connection Signal Frame



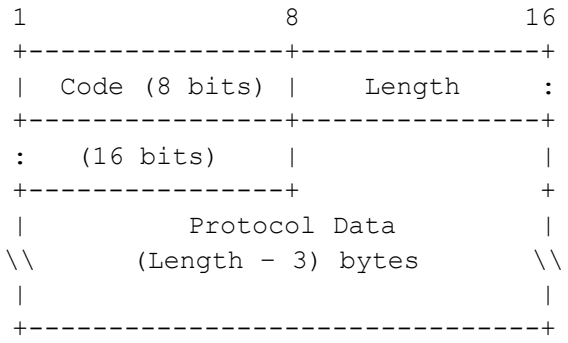
Connection Data Frame

2.2. Secure Channel

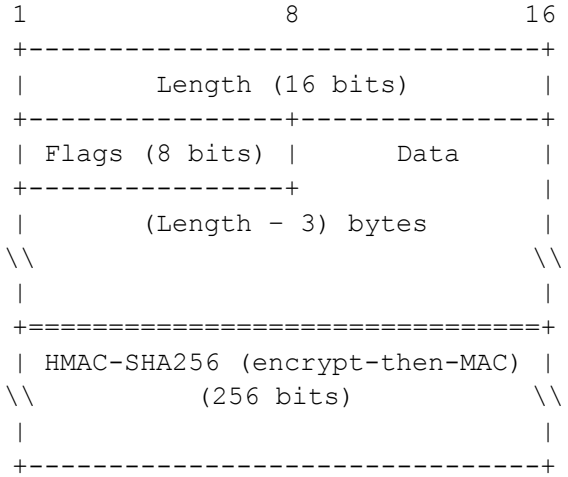
A secure channel provides end-to-end encryption and authentication layer to the virtual channel stream underneath it. Both peers are required to exchange digital profile certificate to authenticate each other before the secure channel is ready for data exchange. The secure channel protocol encrypts the digital profile certificate in transit, preventing identity disclosure to passive sniffing attacks at network level. An optional pre-shared key (PSK) can be used to strengthen the protocol and avoid certificate disclosure to an active attacker during certificate exchange.

The protocol also provides master key re-negotiation feature which when triggered by any peer will start secure channel key exchange and a new master key will be negotiated. Re-negotiation can be triggered automatically by any of the peers on the basis of time the channel is open or the amount of data exchanged.

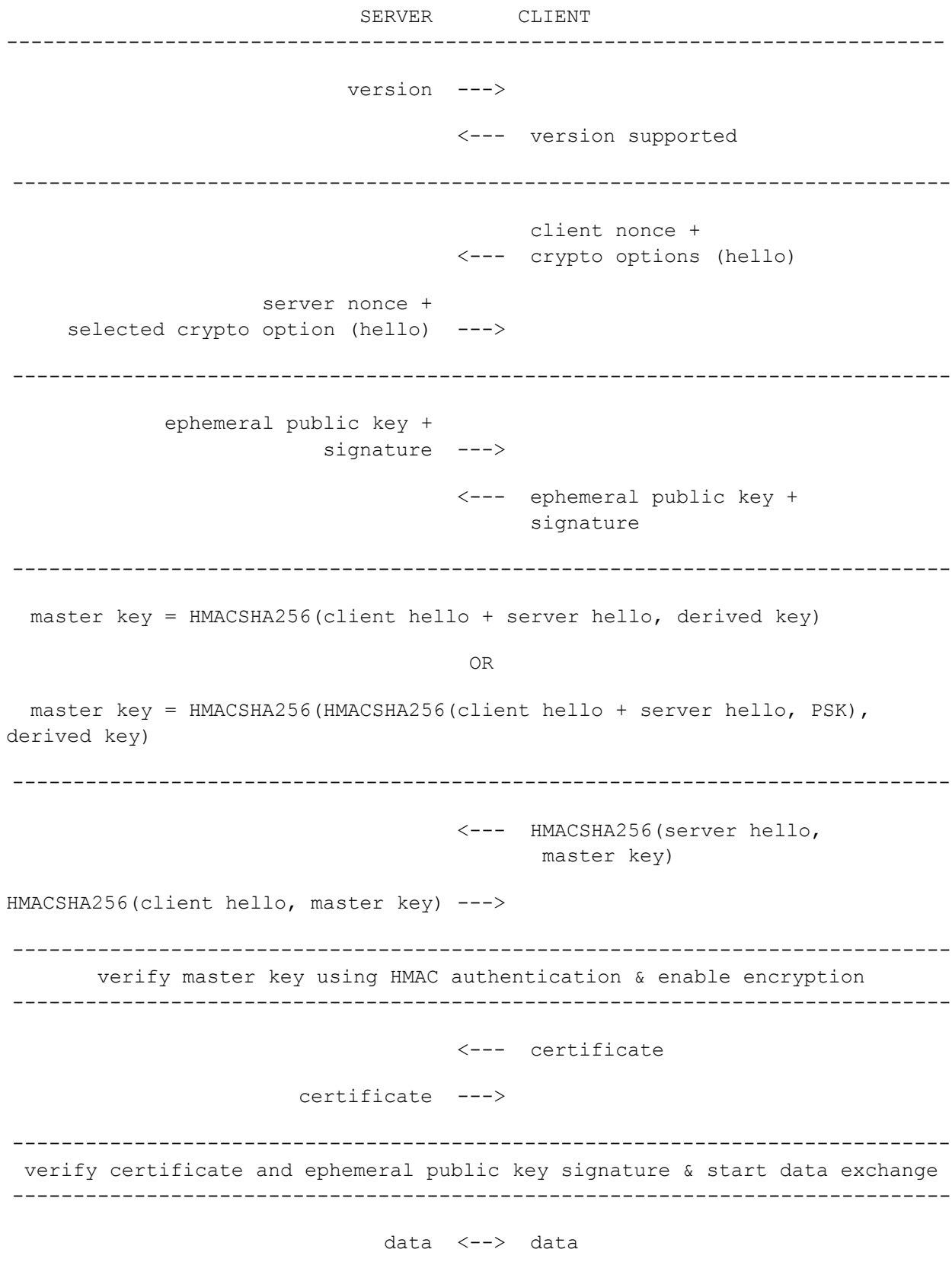
The protocol implements Perfect Forward Secrecy (PFS) using Diffie Hellman (DHE 2048 bits) or Elliptic Curve Diffie Hellman (ECDHE 256 bits) depending on the handshake selection process. For authentication, RSA (4096 bits) signed digital profile certificate is used. The data transmitted is encrypted by AES (256 bits) in CBC mode.



Secure Channel Control Packet



Secure Channel Data Packet



Secure Channel Protocol

Data exchanged after the secure channel establishment is sent as a stream of encrypted packets. Each data packet implements authenticated encryption (encrypt-then-MAC) using HMAC-SHA256. The complete data packet, including the 3 byte header fields, is encrypted and then *HMAC (encrypted packet, master key)* is appended to the packet.

2.3. Bit Chat Message

The Bit Chat Message protocol is used to send text messages, exchange peer information, send keep-alive (NOOP) messages, share files and send notifications. These messages are sent to all the peers connected to the Bit Chat network via a secure channel. The message MUST begin with the message type (8 bit) field. Each type of message has its own message format.

The file sharing feature provided using Bit Chat messages works similar to Bit Torrent file sharing but only for the close group of people connected to the chat network. The file being shared is split into blocks and a file advertisement containing file name, size, hash, and a table of blocks with their hash is sent to each peer connected to the chat network. Peers participating in the file transfer process exchange file blocks with each other such that the peer having the original file does not have to transfer the complete file to each peer individually. Each file block received is verified by hashing received data and comparing it to the hash listed in block table in the file advertisement. Once a peer has all the file blocks downloaded, it keeps sharing the blocks with other peers in need. This allows the initial file sharing peer to go offline once the file is available with another peer in the chat network.

3. Peer Discovery

Bit Chat does not depend upon any centralized mechanism to find peer information like IP address and TCP port number. In this regards, it works similar to Bit Torrent client and even uses torrent trackers to find peer information. Both HTTP [9] and UDP [10] versions of the torrent tracker protocol are supported. Bit Chat also implements a Kademlia [8] based Distributed Hash Table (DHT) for finding peer information. Apart from torrent trackers and DHT, Bit Chat uses IPv4 broadcast and IPv6 multicast options to find peers on the same Local Area Network (LAN).

A Bit Torrent client uses ‘infohash’ to track/find peers to participate in file transfer. Similarly, Bit Chat client uses network ID, corresponding to a unique Bit Chat network, using which peers can find each other. When peer contact information is discovered, the peer-to-peer protocol begins to work.

Once Bit Chat peers are connected to each other, they exchange list of connected peers, allowing the opposite peer to know which other peers need to be connected in order to complete the full mesh network topology. It also has a trigger update mechanism which notifies other peers when a new peer is connected or disconnected allowing quick formation of a full mesh.

The discovered DHT nodes are used by Bit Chat as TCP relays due to the fact that an active DHT node can accept incoming TCP connections. Three nodes are chosen from the list of available DHT nodes to be connected and used as a relay for accepting incoming virtual connections.

Bit Chat may require around a minute's time to discover and connect to all peers to a chat network unlike in a centralized messaging system where a user becomes 'online' almost instantaneously to other peers.

4. Profile & Profile Certificate

Profile certificate is a digital certificate issued to each Bit Chat user upon registration by a certification authority run by Technitium. Bit Chat clients have a hard coded root certificate which is used to verify the chain of certificates. Certificates are issued only after an email address verification process and are essentially email address verified digital certificates.

The profile certificate is exchanged with each peer in the Bit Chat network during the secure channel handshake and it contains all the details that the user provided during the registration process.

Profile certificate use RSA (4096 bit) key pair which the Bit Chat client can automatically generate or the user can import externally generated RSA key pair in PEM format during registration. The RSA private key parameters and Bit Chat client settings are stored in an encrypted local file called as the Profile file. This user profile file is encrypted by AES (256 bits) using a profile password that user is required to enter during registration. Key derivation algorithm PBKDF2 [12] with HMAC-SHA256 and 200,000 iterations is used to generate the AES encryption key from the user provided profile password.

User needs to enter the profile password each time to start Bit Chat with the selected profile file. Since there is no alternate way to access the encrypted profile data without the profile password, in case the user forgets the password, a new profile has to be registered by the user to continue using Bit Chat with the same email address.

The profile file can be moved or copied to another computer to be used with Bit Chat. Bit Chat also supports using multiple computers running Bit Chat client with the same profile file and allows chatting using any of those available computers.

An email address can be used to issue only one profile certificate at a time and the certificate issuing system has a revocation mechanism to allow revoking previously issued certificate which gets automatically triggered when another successful registration for the same email address is done.

This trust based system was chosen to allow people to use something they already have (an email address) to be used as an identifier in the peer-to-peer network. Any other peer-to-peer system that manages peer contacts requires the user to trust the system for initial contact in order to get a peer's public key or an identifier. Any peer-to-peer system that does not have a trust based system to authenticate a user is inherently vulnerable to social engineering attacks since the user needs to trust the peer on the other end with insufficient information. Meeting in person or over voice call to exchange contact info or verify identifiers may not be feasible or may be error prone [7].

5. Privacy

Bit Chat profile certificate registration is the only service which Technitium provides and hence knows the information provided during registration. The same registration information is stored inside the profile certificate which can be viewed by any peer the user chats with. Essentially, a user is sharing the same information with the registration authority and the other peers. It is recommended to the user to provide information brief enough to allow other peers to identify him/her. The RSA private key parameters and the profile encryption password are known only by the user. A detailed privacy policy document is available on the Bit Chat website [3].

Bit Chat supports using HTTP proxy and SOCKS v5 proxy protocols which can be used to hide IP address during registration and chatting. Similarly, user can use any available VPN service to hide IP address. Bit Chat can also be configured to use Tor network by using SOCKS v5 support [4]. User can only make outbound connection via proxy to another peer who can accept incoming connections. If both users configure proxy then they will have to rely on the availability of TCP relay nodes for accepting incoming connection.

Bit Chat network ID is used to discover peers using Bit Torrent trackers and DHT. Any adversary who can figure out the network ID can find a list of peer end points (IP address & port number) and use that info. The network ID is generated using the chat group name or peer email address, and an optional shared secret. When no shared secret is used, network ID can be generated by guessing the group name or peer email address. Thus it is useful to set a shared secret; even a simple one should do a good job.

While using Bit Chat, the message routing is done peer-to-peer and hence there is no metadata collection is possible by Technitium. The peer-to-peer connections shall take the shortest path available such that users who are using same Internet Service Provider (ISP) will have their data being routed within the same ISP network. Messages of users on a private LAN network will never leave the local network. However, it is possible for ISPs to log metadata of TCP connections (like source & destination IP addresses) that are being routed via the networks they control. Any attacker capable of doing passive network sniffing of the network being used by the peer-to-peer connection can log the TCP source & destination IP addresses. The data transferred using Bit Chat over any network shall be end-to-end encrypted with Perfect Forward Secrecy (PFS) in any case.

It should be noted that any peer the user chats with, can view the user's IP address and similarly the user too can view each peer's IP address. This is due to the fact that all peers are connected to each other directly by a TCP connection.

6. Conclusion

Bit Chat provides a simple to use, secure, peer-to-peer, alternative instant messaging platform with end-to-end encryption for people and organizations who are concerned about their privacy and security. Using techniques similar to a Bit Torrent client, a fully peer-to-peer instant messaging network is possible and scalable without requiring much investment to maintain the service availability.

References

[1] Bit Chat Source Code on GitHub, https://github.com/TechnitiumSoftware/BitChatClient

[2] Technitium Bit Chat License Agreement, https://bitchat.im/license.html

[3] Technitium Bit Chat Privacy Policy, https://bitchat.im/privacypolicy.html

[4] "How to Configure Bit Chat to Use Tor Network", http://blog.technitium.com/2015/11/how-to-configure-tor-with-bit-chat-v4.html

[5] Bruce Schneier, "Why We Encrypt", https://www.schneier.com/essays/archives/2015/06/why_we_encrypt.html

[6] Glenn Greenwald, "Why privacy matters", http://www.ted.com/talks/glenn_greenwald_why_privacy_matters

[7] "User Error Compromises Many Encrypted Communication Apps", http://www.technologyreview.com/news/544516/user-error-compromises-many-encrypted-communication-apps/

[8] Petar Maymounkov and David Mazieres, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric", https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf

[9] Bit Torrent Tracker Protocol, https://wiki.theory.org/BitTorrent_Tracker_Protocol

[10] UDP Tracker Protocol for Bit Torrent, http://www.bittorrent.org/beps/bep_0015.html

[11] Bit Torrent Protocol Specification, https://wiki.theory.org/BitTorrentSpecification

[12] PKCS #5: Password-Based Cryptography Specification Version 2.0, https://www.ietf.org/rfc/rfc2898.txt