

INTRODUCTION

Chat protocols are very popular on the Internet. They have actually been very popular since the very first chat protocols appeared on the net. The Internet Relay Chat (IRC) was one of the first chat protocols, and quickly gained the status of being the most popular chat on the net. Today, IRC has several competitors from various other so called Instant Messaging (IM) protocols, such as ICQ. However, all of these different chat protocols have something in common; they are all insecure. The security is important feature in applications and protocols in contemporary network environment. The older chat protocols, however have failed to meet the growing security requirements on the Internet. It is not anymore enough to just provide services, like for example chat services. Now, they need to be secure services.

The Secure Internet Live Conferencing (SILC) protocol is a new generation chat protocol which provides full featured conferencing services, just like any other contemporary chat protocol provides. In addition, it provides security by encrypting and authenticating the messages in the network. The security has been the primary goal of the SILC protocol and the protocol has been designed from the day one security in mind. All packets and messages travelling in the SILC Network are always encrypted and authenticated. The network topology is also different from for example IRC network. The SILC network topology attempts to be more powerful and scalable than the IRC network. The basic purpose of the SILC protocol is to provide secure conferencing services. The SILC Protocol have been developed as Open Source project. The protocol specifications are freely available and they have been submitted to the IETF. The very first implementations of the protocol are also already available.

ABOUT SILC

SILC (Secure Internet Live Conferencing) is a protocol which provides secure conferencing services on the Internet over insecure channel. SILC superficially resembles IRC, although they are very different internally. They both provide conferencing services and have almost the same set of commands. Other than that, they are nothing alike. The SILC is secure and the network model is entirely different compared to IRC.

SILC provides security services that any other conferencing protocol does not offer today. The most popular conferencing service, IRC, is entirely insecure. If you need secure place to talk to some person or to group of people over the Internet, IRC or any other conferencing service, for that matter, cannot be used. Anyone can see the messages and their contents in the IRC network. And the most worse case, some is able to change the contents of the messages. Also, all the authentication data, such as, passwords are sent plaintext in IRC.

SILC is much more than just about 'encrypting the traffic'. That is easy enough to do with IRC and SSL hybrids, but even then the entire network cannot be secured, only part of it. SILC provides security services, such as sending private messages entirely secure; no one can see the message except you and the real receiver of the message. SILC also provides same functionality for channels; no one except those clients joined to the channel may see the messages destined to the channel. Communication between client and server is also secured with session keys and all commands, authentication data (such as passwords etc.) and other traffic is entirely secured. The entire network, and all parts of it, is secured. We are not aware of any other conferencing protocol providing same features at the present time.

SILC has secure key exchange protocol that is used to create the session keys for each connection. SILC also provides strong authentication based on either passwords or public key authentication. All authentication data is always encrypted in the SILC network. Each connection has their own session keys, all channels have channel specific keys, and all private messages can be secured with private message specific keys.

Distribution

The SILC is distributed currently in three different packages. The SILC Client package, the SILC Server package and the SILC Toolkit package. Each package has its intended audience.

- SILC Client package is intended for end users who are looking for a good and full featured SILC client. The SILC Client package currently includes Irssi-SILC client that supports all SILC features, themes and much more. It is curses based but has a possibility of adding various other frontends to it. The Irssi-SILC client's user interface is based on the Irssi client (see Irssi project).
- SILC Server package is intended for system administrators who would like to run their own SILC server or SILC router. The package includes the actual server but not the client. If you are running a server and would like to connect it to the silc.silcnet.org router you can contact us.
- SILC Toolkit package is intended for developers and programmers who would like to create their own SILC based applications or help in the development of the SILC protocol. The actual development of the SILC is done in the Toolkit and all the other packages are based on the

Toolkit releases. The Toolkit includes SILC Protocol Core library, SILC Crypto library, SILC Key Exchange (SKE) library, SILC Math library, SILC Modules (SIM) library, SILC Utility library, SILC Client library and few other libraries. It also includes the Irssi-SILC Client, another client as an example how to program with the Toolkit and the SILC Server.

HISTORY

SILC, designed by Pekka Riikonen, was released to the public in the summer of 2000, but both the idea and the original protocol date from 1996. The first lines of code were written in early 1997, and SILC has been rewritten three times since that very first version. The original implementation of SILC included a client, a very preliminary server, and implementations of both the RSA and 3DES encryption algorithms. The server was unusable, but the client looked every similar to the one found in the first public release. That release's random number generator was inspired by the RNG that SSH used; the current one is based on that original implementation but has been rewritten twice since then.

SILC's development was temporarily suspended for a few months in 1997, when Pekka's time was consumed with work and school. It resumed in 1998 when Juha Räsänen and Pekka added an implementation of the ElGamal encryption algorithm. Development stopped again because of time constraints, but in 1998 SILC was rewritten in C++, which seemed like a good idea. Pekka had to stop development yet again in the winter of 1999 as work on his thesis took up his available time.

Later in 1999 it was decided that SILC would be rewritten from scratch, using C instead of C++. Core parts of the protocol were reworked, the

protocol was fully documented, and the specifications were submitted to the IETF. The result of this effort was the original public release in the summer of 2000. Since then, several other people have contributed to the continued development of the project, which continues to this day.

FEATURES OF SILC

- Normal conferencing services such as private messages, channels, channel messages, etc. All traffic is secured and authenticated.
- No unique nicknames. There can be same nicknames in SILC without collisions. SILC has unique Client ID's, Server ID's and Channel ID's to assure that there are no collisions. The maximum length of the nickname is 128 characters. The maximum length of the channel name is 256 characters.
- Channels can have channel operators and a channel founder which is the client who created the channel. Channel founder privileges supersede the channel operator privileges. Also, channel founder privileges may be regained even if the founder leaves the channel. The requirement for this is that the client is connected to the same server it was originally connected. The channel founder cannot be removed (kicked) from the channel using force.
- Channel messages are protected by channel key, generated by the server. The key is re-generated once in an hour. It is possible to set a private key for the channel so that even the server does not know the key. Actually, it is possible to set several private keys so that only specific users on the channel may decrypt some specific messages. Adding the private key significantly increases the security as nobody else but the users on the channel know the key.
- Private messages are protected using session keys, generated when connecting to the server. This means that the private messages are

decrypted and re-encrypted enroute to the true receiver of the message. However, it is possible to set a private key between two clients and protect the private messages with that key. In this case no server enroute can decrypt the message since they don't have the key. The SILC protocol provides an automatic key negotiation between two clients using the SKE protocol. This makes it very easy to negotiate a shared secret key with another client in the network.

- All the other traffic, like commands between client and the server are protected using the session keys. Session keys are re-generated once in an hour. The re-key may be done with or without the PFS (Perfect Forward Secrecy).
- Secure key exchange and authentication protocol. SILC Key Exchange (SKE) protocol provides key material used in the SILC sessions in secure manner. The protocol is immune for example to man-in-the-middle attacks and is based on the Diffie-Hellman key exchange algorithm. The SILC Authentication protocol provides strong authentication. Authentication may be based on pass phrase or public key (RSA) authentication. For clients there is an option not to use authentication when connecting to servers.
- Supports secure file transferring between clients in the network. SILC use the SFTP as the main file transfer protocol.
- All traffic is encrypted and authenticated using the best cryptographic algorithms out there. Cipher keys are, by default, 256 bits in length and public keys, by default, 1024 bits in length.
- Supports the following ciphers: AES (Rijndael), Twofish, Blowfish, Mars, Cast-256, RC5 and RC6. Supports the following hash functions: MD5 and SHA1. Supports the following HMACs: hmac-

sha1-96, hmac-md5-96, hmac-sha1 and hmac-md5. Supports the PKCS #1 (RSA) for public key cryptography.

- Supports data compression with GZIP to improve performance.
- Supports SOCKS4 and SOCKS5 firewall traversal protocols.
- SIM (SILC Module) support. Support for loading of shared objects at run-time that provides new and extended features to both SILC client and server. These can provide extra ciphers and extra features to the software.
- SILC client can be installed and used without root privileges.
- SILC client can be configured by system wide configuration files but with user specific configuration files as well.

SILC PROTOCOL

The Secure Internet Live Conferencing (SILC) protocol provides secure conferencing services over insecure network channel. The SILC is IRC like protocol, however it does not support IRC. Strong cryptographic methods are used to protect SILC packets inside the SILC network. SILC provides all the common conferencing services like channels, channel messages, private messages, nicknames, various commands, and secure file transfer. Difference to other chat protocol is in the design of the protocol. The SILC protocol has been designed from the day one security in mind and it shows in the protocol design.

The packets and messages in the SILC network are always encrypted and authenticated. It is not possible to send unencrypted messages in SILC at all. This assures that end user cannot even accidentally send unencrypted messages while thinking that it is encrypted. This is one of the problems of most of the other chat protocols that provide so called plugin encryption. They are not secure by default but try to provide security by applying external

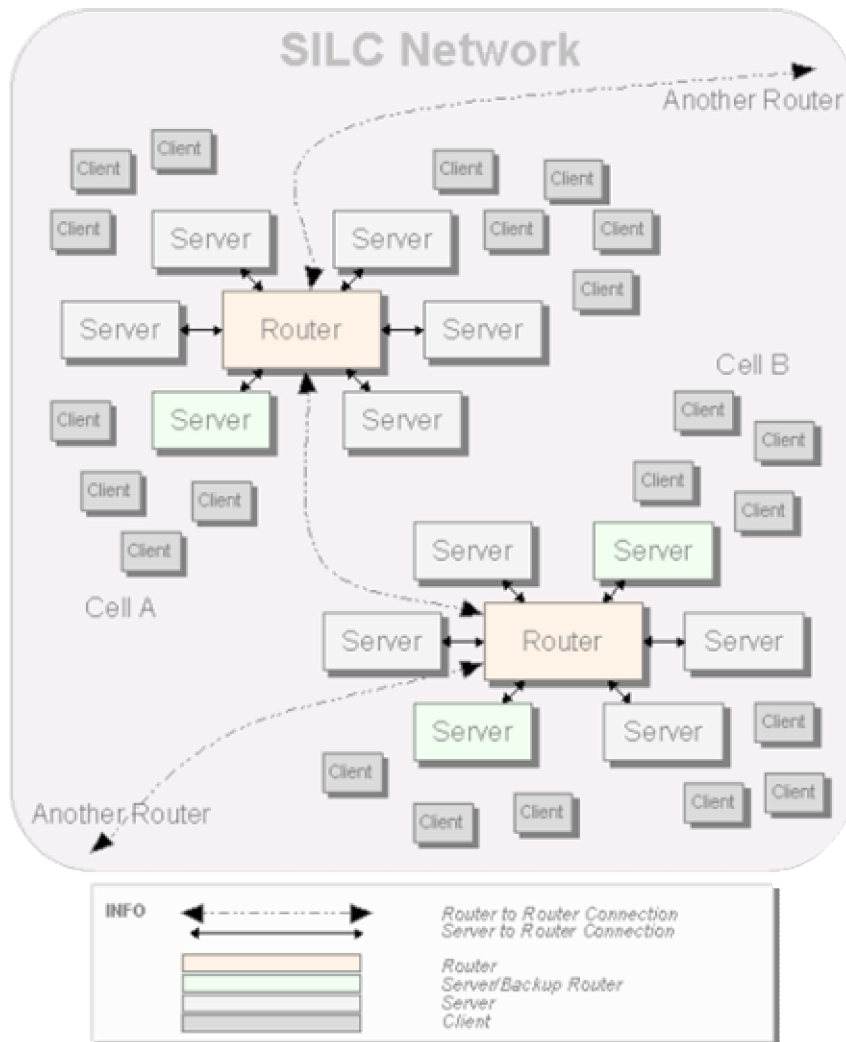
security protocol such as PGP or SSL over the insecure chat protocol. In these cases the security is achieved usually by encrypting the data while key management, message authentication and other security issues may be left out, leaving the implementation vulnerable to various security problems. The other problem is also that the external protocols tend to leave the network only partly secured; usually only two points in the network are secured with for example SSL. While SSL does provide provable security it is not enough to provide security for a chat network as a whole.

SILC is secure in environment of mutual distrust between entities in the network. It is possible to encrypt messages end to end, so that only the sender and the receiver is able to encrypt and decrypt messages. It is also possible to send messages to group of users, so that only the specified group of users is able to encrypt and decrypt messages. Many times the protocol use keys that are generated by the servers, so that if other external key exchange methods fail the network still remains encrypted. However, it is always possible to negotiate and use locally generated keys to secure messages, so that the servers do not know the key.

Like so many other contemporary chat protocols, SILC too provides file transfer. It is possible to transfer files securely between users in the SILC Network. The actual file transfer stream is always sent outside the network peer to peer. Before the file transfer is started a key exchange protocol is executed to negotiate file transfer session key.

The network topology is also different to various other chat protocols, like for example IRC. IRC has tree style network, but SILC network can be described more as an hybrid ring-mesh network. The routers in the network form a ring, but they can also have other direct routers (secondary routes) to

other routers. A router in the network is also called a cell, when it has multiple servers and clients connected to it. The cell can also have backup routers in case the primary router becomes unresponsive.



The diagram above illustrates a portion of the SILC network. It shows two cells that both have several servers, and backup routers and several clients. Clients can connect to server and routers if they want to. The following sections will describe the entities of the SILC Network in greater detail.

Clients

A client is a piece of software connecting to SILC server. The software is usually run by the end user, a real person that is. The purpose of the clients is to provide the end user an interface to the SILC services. They are used to actually engage the conversations on the SILC Network, and they can be used to execute various SILC commands.

The clients are distinguished from other clients by unique Client ID. There cannot be multiple same Client IDs in the SILC Network at the same time. The end user, however does not use Client IDs. The end users usually selects a preferred nickname they want to use, and identifies themselves with that nickname to other users on the network. The nicknames are not unique in the SILC Network. There can be multiple same nicknames at the same time on the network. The maximum length for the nickname is 128 characters.

Most of the other chat protocols have unique nicknames. This is where SILC differs from most of the other chat protocols. The purpose of this feature is to make IRC style nickname wars obsolete, as no one owns their nickname; there can always be someone else with the same nickname.

When client connects to the server the SILC Key Exchange (SKE) protocol and SILC Connection Authentication protocol are executed. The result of the SKE protocol is the session key that the client and server use to secure their communication. All commands, for example, that the client sends to the server are secured with the session key. The session key expires

periodically and the rekey process can be executed with or without the Perfect Forward Secrecy (PFS). The connection authentication protocol is used to authenticate the client to the server. The server may allow the client to connect without authentication, or it may require a passphrase or public key based (or certificates) authentication.

Servers

Servers forms the basis for the SILC Network, by providing a point to which clients may connect. There are two kinds of servers in SILC; normal servers and router servers. The next section describes the function of router server.

Normal servers connect to router server. Normal servers cannot directly connect to other normal servers. Messages that are destined outside the local server are always sent to the router for further routing. The clients usually connect to the normal server, however, clients may connect to router servers as well. The SILC Network diagram above illustrates how normal servers connects to the router server.

The servers are distinguished by other servers in the network by unique Server ID. There cannot be multiple same Server IDs in the SILC Network at the same time. The servers keep track of local information. It knows all locally connected clients and it knows all channels that its clients have joined. However, it does not know any global information. It usually does not keep track of global clients, however, it may cache that information if it was queried. The reason for this is that the server does not need to keep global information up to date and thus makes the server faster (and in the end the entire network faster). They can always query the information from the router.

When server connects to its router the SILC Key Exchange (SKE) protocol and the SILC Connection Authentication protocol are executed, just like when client connects to server. The SKE results in to the session key that is used to secure the communication between the server and the router. The connection authentication protocol is used to authenticate the server to the router. The authentication is always based in either passphrase or public key (or certificates).

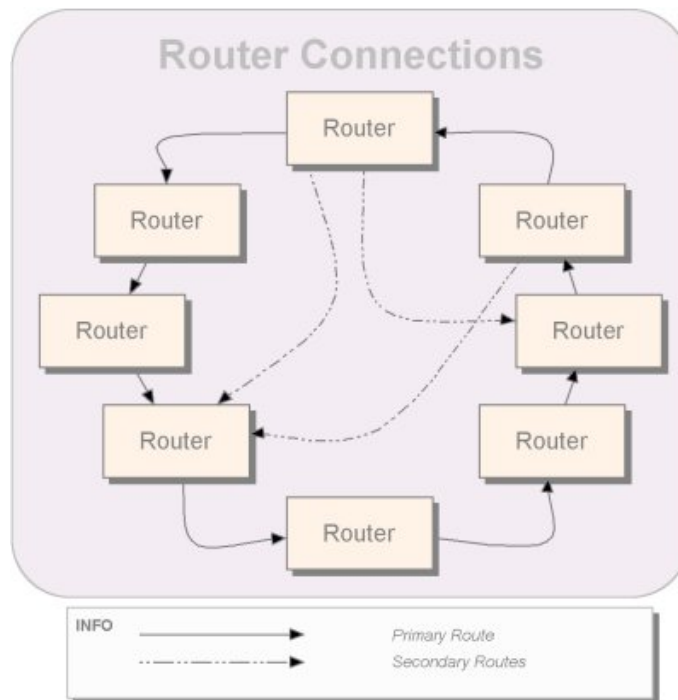
Routers

The router servers are servers that actually handles the message routing in the network. They are, however also normal servers and they do accept client connections. Each of the router in the network is called a cell. A cell can have only one active router and it may have several servers and several clients. The cell, however may have backup routers that can take over the tasks of the primary router if it becomes unresponsive. The switch to the backup router should be transparent and only local connections to the primary router are lost. Other connections in the cell are intact, and clients and servers merely experience some lag in the network connection during the switch to the backup router.

The normal server knows only local information. Router server on the other hand knows local information and global information. It considers the cell as local and outside cells as global. It knows all the clients connected to the network, all created channels, and all routers and servers in the network. The server may query the global information if it is needed. For example, when client sends WHOIS command, the server may query the information from the router. If the router does not know all the details that the WHOIS

command requires it can query the information from a router or a server which knows all the details. It may then cache that information.

The primary purpose of the router server is to route the messages to local servers and local clients, and messages that are destined to outside the cell are routed to the primary route or some other secondary route if it is a faster route. The routers in the network forms a ring. Each router has a primary route to other router in the network. Finally the ring is closed by the last router using the first router in the network as its primary route.



The diagram above illustrates how the routers forms a ring in the network. A router may have several secondary routes which it may use when it routes the packets.

When routers connect to its primary router the SKE and the SILC Connection Authentication protocols are executed just like when normal

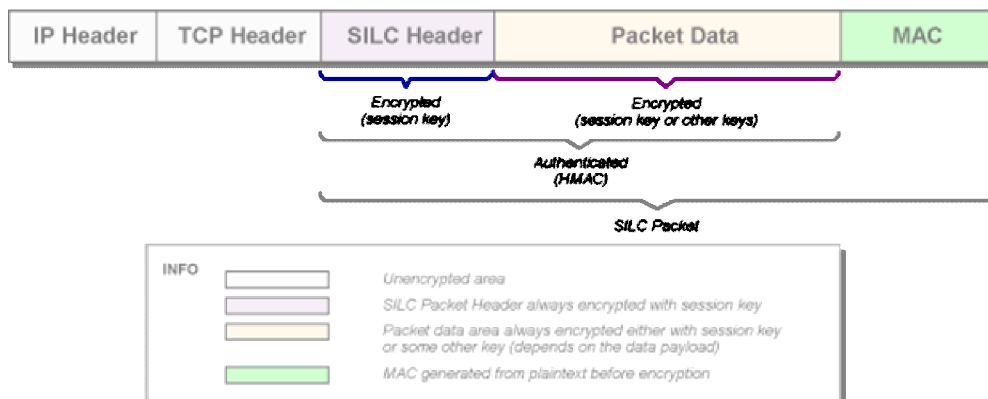
server connects to its router. The session key is used to secure the communication between the routers. All the secondary routes also have their own session keys.

SILC Packet Protocol

The basis of SILC protocol relies in the SILC packets and they are without a doubt the most important part of the protocol. The SILC Packet protocol is a binary packet protocol. The protocol provides secure binary packets and assures that the contents of the packets are secured and authenticated.

Packets are used in the SILC protocol all the time to send for example channel messages, private messages, commands and other information. All packets in SILC network are always encrypted and their integrity is assured by computed Message Authentication Codes (MAC). The protocol defines several packet types and packet payloads. Each packet type usually has a specific packet payload that actually defines the contents of the packet. Hence, the actual data in the packet is the packet payload defined in the protocol.

SILC Packet and Packet Encryption



As the diagram above illustrates the SILC packet is constructed from the SILC Packet Header that is included in all SILC packets, data area that includes the packet payloads, and MAC area which assures the integrity of the packet. Entire SILC packet is always encrypted, except for the MAC area which is never encrypted. The encryption process and the key used, however depends on the packet payload. Some of the payloads are encrypted with the session key and some are encrypted with other keys, for example with channel message keys. The SILC Packet Header is always encrypted with the session key. The MAC is computed from the SILC Packet Header and the data area before encrypting the packet.

SILC Key Exchange Protocol

SILC Key Exchange Protocol (SKE) is used to exchange shared secret between connecting entities. The result of this protocol is a key material used to secure the communication channel. This protocol is executed when, for example client connects to server. It is also executed when server connects to router. And, there is no reason why it could not be executed between two clients too, if two clients would need to create secret key. The purpose of the SKE protocol is to create session keys to be used in current SILC session. The SKE is based on the Diffie-Hellman key exchange algorithm, and is immune to for example man-in-the-middle attacks by using digital signatures.

This is the first protocol that is executed when creating connection to, for example SILC server. All the other protocols are always executed after this protocol. This way all the other protocols are secured since the SKE creates the session key that is used to secure all subsequent packets. The session keys created in the SKE are valid only for some period of time (usually an hour) or at most until the session ends. The rekey process can be executed with or without the Perfect Forward Secrecy (PFS).

The security properties that are used in the SILC session are also negotiated during the SKE. The protocol has initiator and responder. The initiator is the one who starts the SKE negotiation and responder is the one who receives the SKE negotiation. When the protocol is started initiator sends a list of security properties that it supports. The responder then selects the security properties it supports and sends its reply to the initiator. The security properties includes ciphers, hash functions, public key algorithms, HMAC functions and other security properties. The responder can always choose the properties it supports.

After the security properties are selected the protocol continues by performing the Diffie-Hellman key exchange algorithm. At the same time the initiator and responder also sends their public keys or certificates to each other. The initiator and responder also computes a signature that the other party will verify. By default the protocol is executed in so called mutual authentication mode, where both of the parties computes a signature which are verified by each other independently. This way both of the parties will have prove the possession of the private key to the public key they are providing in the protocol. If any of the phases of the protocol are to fail the connection is closed immediately.

The public key or certificate that is received during the SKE protocol must be verified. If it is not verified it would be possible to execute a man-in-the-middle attack against the SKE protocol. If certificates are used they can be verified by a third party Certification Authority (CA). Verifying a public key requires either confirming a fingerprint of the public key over phone or email, or the server can for example publish the fingerprint (public key) on some website. In real life systems accepting the public key without verification,

however is often desired. In many security protocols, such as in SSH2, the public key is accepted without verification in the first time when the connection is created. The public key is then cached on local hard disk. When connecting next time to the server the public key on local cache is verified against the public key server sent. In real life this works most of the time. However, if client (or server) cannot trust this, it must find some other way to verify the received public key or certificate.

SILC Connection Authentication Protocol

Purpose of SILC Connection Authentication protocol is to authenticate the connecting party with server or router. This protocol is executed when for example client connects to server. It is also executed when server connects to router. Its other purpose is to provide information for the server about which type of connection it is. The type of the connection defines whether it is client, server or router. If it is client then the server will create a new Client ID for the client. If it is server then it will expect the server to send its Server ID. Server IDs are created by the servers and routers itself.

Since the SILC Connection Authentication protocol is always executed after the SKE protocol, session keys has been established already. This means that all packets sent in the connection authentication protocol are encrypted and authenticated. The authentication may be based either in passphrase or public key encryption. It is also possible to not require authentication at all. If the authentication is based to passphrase the passphrase is sent to the server. As the packet sent by, for example client, is entirely encrypted it is safe to send the passphrase inside the packet.

If the authentication is based to public key then, for example the client, signs data with its private key and sends it to the server. The server then

verifies this signature by using the client's public key. The packet is also encrypted in the case of public key authentication.

If the authentication is to fail the connection to the server or router will be refused. If it is successful the connection is granted. After this the client is ready to communicate in the SILC Network.

Channels

A channel is a named group of one or more clients which will all receive messages addressed to that channel. The channel is created when first client joins to it, and the channel ceases to exist when the last client leaves it. When channel exists, any client can reference it using the name of the channel. Channel is a place where group of people can engage conversation. Channel names are unique in the SILC Network. There cannot be multiple same channels in the network at the same time. However, channel has also a Channel ID which is actually used to reference the channel in the SILC Network. The maximum length for the channel name is 256 characters. Channels can have operators that can administrate the channel and operate all of its modes. There are two types of operators on the channel: channel founder and channel operator.

The channel founder is the client which created the channel. Channel founder is channel operator with some more privileges. Channel founder can operate all of the channel's modes. Furthermore, channel founder privileges cannot be removed by any other operator on channel and channel founder cannot be removed from the channel by force. It is also possible for the channel founder to regain its privileges at later time, even if they have left the channel. Channel operator is operator that can operate most of the channel's

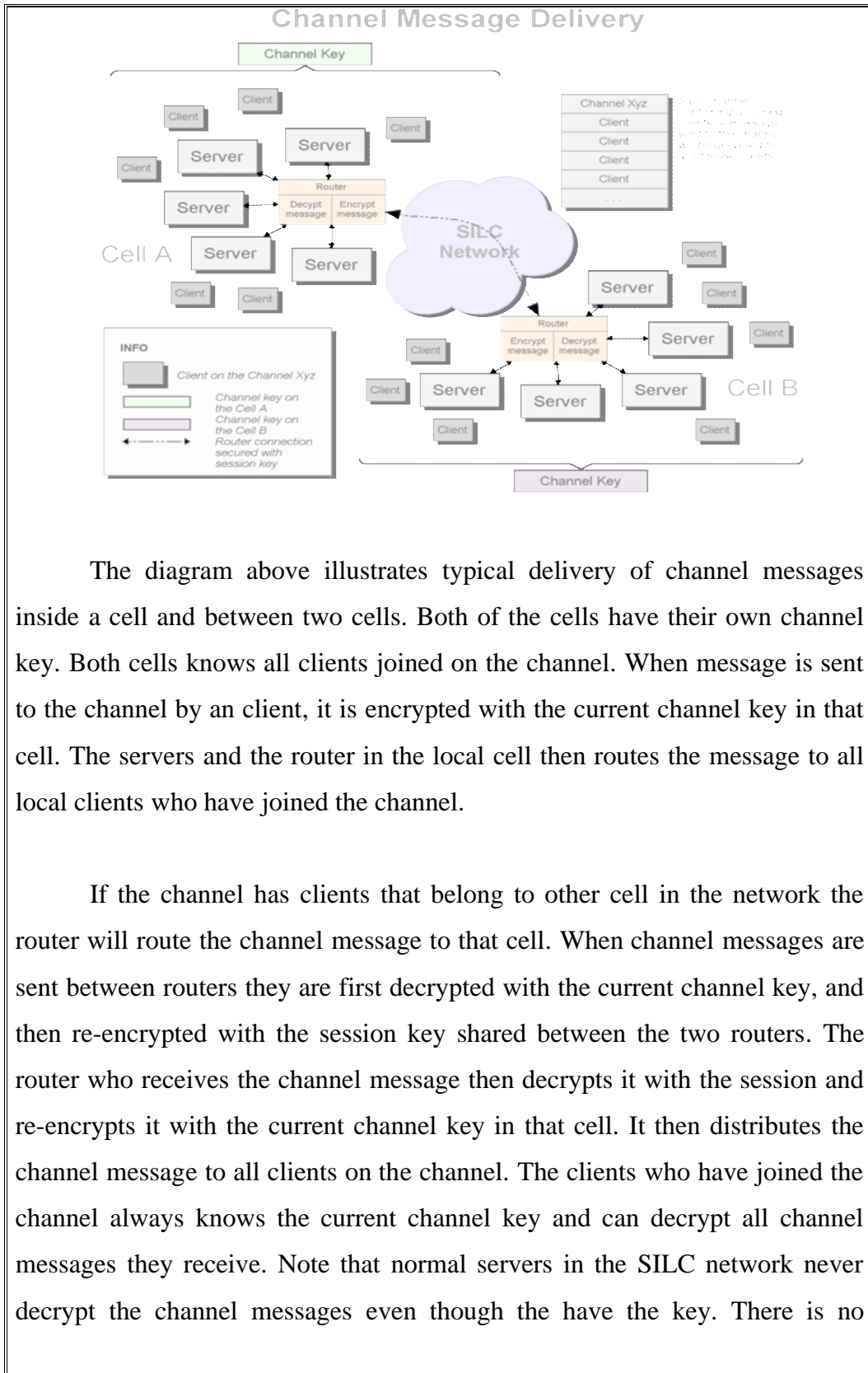
modes and administrate the channel. However, it cannot operate all modes which are strictly reserved for channel founder. Channel operator is, however able to administrate the channel, set some modes on the channel, remove a badly behaving client from the channel, and promote other clients to become channel operator.

Channel Message Delivery

All clients that have joined the channel can send messages to the channel. All channel messages are secured and authenticated by channel key. The channel key is generated by the server when the channel is created, a client joins the channel, or a client leaves the channel. The channel key is also regenerated periodically.

The reason for the regeneration of channel key everytime someone joins or leaves the channel is that it prevents new clients joining the channel, and old clients leaving the channel, to encrypt or decrypt old or new messages. They can encrypt and decrypt channel messages only when they have joined on the channel.

Channel keys are cell specific in the SILC Network. Each cell that have clients joined on a particular channel have also own key for the channel. That key is not shared by other cells in the network. Inside the cell the channel key is known by the router and all servers that have clients on the channel and all clients that have joined the channel.



reason for servers to decrypt the message. The router decrypts the message only when sending it between two routers.

This method of channel message delivery is the default way to send channel messages in the SILC Network. However, this is not perfect solution on all circumstances. If the clients joined on a particular channel cannot trust, or do not want to trust the servers and routers in the SILC Network they can consider the fact, that servers and routers knows the channel key is actually a breach of security.

If the clients on the other hand can trust their servers and routers in the SILC Network this is the recommended way of sending channel messages. This method is the simplest method for end user since it does not require any special settings before engaging the conversation on the channel. The client merely joins the channel, receives the channel key from the server and can start the conversation on the channel.

In addition of encrypting channel messages it also possible to digitally sign all sent channel messages. The receiver could then verify the signature of each of the message using the sender's public key.

Channel Message Delivery With Channel Private Key

If the clients cannot trust the servers and routers in the SILC Network they should not use the default way of sending the channel messages. Instead, they should use channel private keys to encrypt and decrypt the channel messages. Channel private keys are keys that are known only by the clients who have joined the channel. Servers and routers do not know the key and cannot decrypt the messages. When message is sent between two routers they

are merely re-encrypted with the session key but not decrypted since the router do not have the key to do that.

The clients who have joined the channel must first agree on the channel private key they are going to use. The key may generally be anything. It may be a passphrase or a random string, or the key may negotiated using some key exchange protocol which provides negotiating the key for multiple clients at the same time.

As the channel private key is actually entirely local setting in the client, it is possible to set several channel private keys for one channel. It is possible to have multiple channel private keys that are not known by all channel members.

When encrypting messages with one channel private key only the clients who have that key can decrypt the message. The other key could be shared for example by all clients on the channel and thus all clients can decrypt messages encrypted with that key. In this way it is actually possible to have a private group conversation inside the channel while having global conversation at the same time.

Private Messages

Private messages are messages that are sent from one client to another through the SILC Network. They are private because they are not sent to anyone else except to the true receiver of the message. Private messages can be used to engage private conversation with another client if channels are not desired.

As all messages in SILC the private message are also encrypted and

authenticated. There are several ways to secure private messages. By default private messages are encrypted using the session keys established in the SKE protocol.

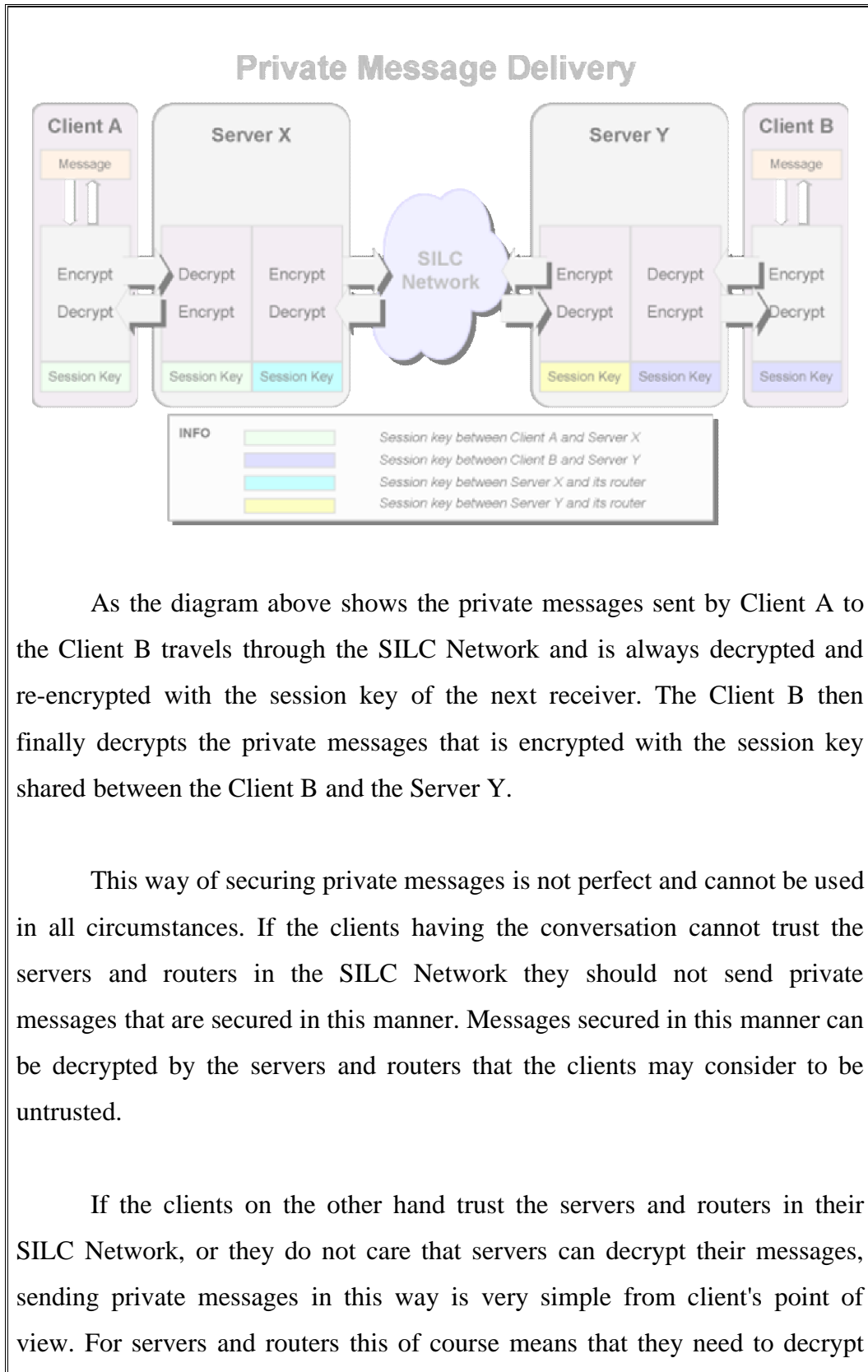
It is also possible to negotiate a private message key between the two clients and encrypt the messages with that key. It is even possible to encrypt the messages with public key cryptosystem, if desired. The next sections will describe all these private message delivery methods.

The SILC protocol provides these three methods of delivering private messages because none of the methods alone can satisfy the security requirements of all people. The end user should decide the acceptable level of risk, the required level of security and other security and usability aspects when deciding what way of sending private message suites for them.

In addition of encrypting private messages it also possible to digitally sign all sent private messages. The receiver could then verify the signature of each of the message using the sender's public key.

Private Message Delivery With Session Keys

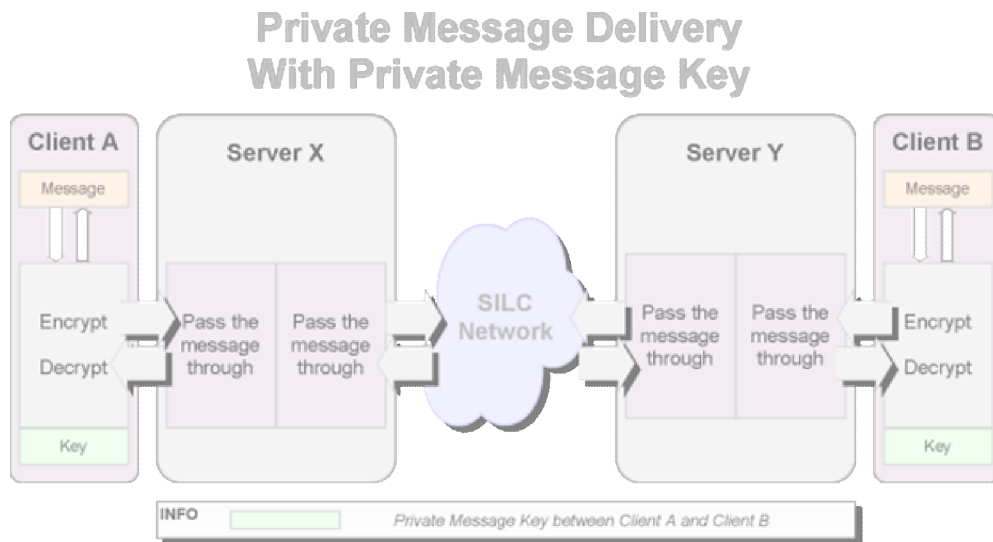
Sending private messages are by default secured with session keys established in the SKE protocol. This means that the private message is always encrypted with the session key of the next receiver of the message enroute to the receiving client. This also means that the message is decrypted and re-encrypted everytime it is sent further to the receiving client.



and re-encrypt each private message. Since this way of securing private message cannot be used at all times the SILC protocol provides other ways of securing private messages.

Private Message Delivery With Private Message Key

Private messages can be secured with private message key as well. This key is known only by the sender of the message and the receiver of the message. This way no one else except the sender and the receiver can encrypt and decrypt the private messages. The message is encrypted by the sender with the private message key and all the servers and routers pass the message through enroute to the receiver. They cannot decrypt the message since they do not have the key. When sending private messages in this way it does not matter whether the clients trust or do not trust the servers and routers in the SILC network.



As the diagram above shows the Client A encrypts the message with private message key and sends the message to the SILC Network. All servers and routers pass the message through since they cannot decrypt it. The Client B then receives the message and decrypts it with the private message key.

Sending private messages in this manner is always secure since the key is shared only by the sender and the receiver. The problem of this method is that the sender and the receiver must somehow agree about the key they are going to use. The private message key can generally be anything. It can be a passphrase that only the sender and the receiver knows. They could have been agreed to use some word or phrase as the key sometime earlier before they started the conversation. Or the key maybe from some random string from a code book that only the sender and the receiver poses. Or it can be a key that is negotiated using some key exchange protocol.

The problem however is fundamental. How to agree to use some key when you cannot reach the other person over secure channel? The SILC protocol solves this problem by providing a possibility to negotiate the key between two clients using the SKE protocol. One or both of the clients can set up the SKE server running in their host and ask the other client to connect to it. In this case the SKE is executed outside the SILC Network. As a result of the SKE protocol the clients have now shared secret that they can use as private message key. The key is known only by the two clients that executed the SKE protocol. They can then use that key to secure all subsequent private messages.

Using this method of private messages delivery is recommended if the clients cannot trust the servers and routers in the SILC Network. The drawback is the extra phase of setting the private message key before starting the conversation. However, using the SKE protocol is the recommended way to negotiate the private message key since it can be automatized and does not cause any extra tasks for end user.

Private Message Delivery With Public Key Encryption

If the clients cannot trust the servers and routers in the SILC Network they can use the private message key. As described in the previous section it is easy to set up with the SKE protocol. However, sometimes the two clients do not want to use any passphrases as private message key or negotiate the key with SKE, or perhaps they are unable to negotiate the key because of some other external problem. The SILC protocol provides yet another way of securing the private messages. This way does not require setting or negotiating private message key. And, in this method also it does not matter whether the clients trust or do not trust the servers and routers in the SILC Network. The method is public key encryption.

The clients can encrypt the private messages with the receiver's public key and send the message to the network. The servers and routers cannot decrypt the messages since they do not have the receiver's private key. The receiver on the other hand has the private key which it uses to decrypt the message.

Even this method of private message delivery is not perfect. The drawbacks of this method is that the sender must first assure that the public key it is using in the encryption is actually the receiver's public key. This is a absolute requirement in this method.

If the sender cannot authenticate the receiver's public key this method of private message delivery should not be used. In SILC protocol clients can fetch other clients public keys from servers, but the client still must verify the key. Use of certificates can solve the problem. The receiver's certificate could be authenticated by a third party Certification Authority (CA).

Usually verifying the public key is not a problem since the receiver can provide the public key on some website, or verify the fingerprint of the key over email, or phone call. The clients can also fetch the public keys from SILC servers. If both of the clients trust that the public keys are authentic using this method of private message delivery is very simple and recommended.

Secure File Transfers

The file transfer support in chat protocols are a absolute requirement nowadays, and chat protocol without one is no chat protocol at all. SILC also supports file transfer with the addition that the file transfer stream is secured. When a user wants to transfer a file to another user, the SILC Key Exchange (SKE) protocol is first executed to negotiate a session key for the file transfer stream. This key is then used to protect the peer to peer stream between users.

The file transfer protocol used in SILC protocol is the SSH File Transfer protocol (SFTP). Even though the name of the protocol relates to SSH, the actual file transfer protocol has nothing to do with Secure Shell. The SFTP is totally independent file transfer protocol and its stream is secured using SILC. The SFTP is very good protocol because in addition of providing simple file transfer support, it can also support complex file and directory manipulation.

The support for file transfer in SILC has been designed so that using practically any file transfer protocol is possible. The mandatory protocol is SFTP but in the future adding support for other protocols is also possible.

FUTURE OF THE PROTOCOL

The current protocol version is 1.0. This does not mean that the protocol is perfect and does not need further development. There is still features that are missing from the protocol, and it is clear that the protocol needs to mature a bit more. There has been a talk about adding features like permanent channels, more wide channel founder privileges, and other similar features.

The network model of the protocol allows powerful routing capabilities, however the routing is not fully defined yet in the protocol and requires more in depth work. The protocol is still in draft phase and is open for new features. However, it is our intention that the protocol will be standardized in the future.

LICENSING OF THE SILC SOFTWARE

The SILC is distributed currently as three different distributions; SILC Client, SILC Server and SILC Toolkit. Each of the distributions may have different licenses and different licensing policy. Currently the licenses of these different distributions are as follows:

- SILC Client: GNU General Public License
- SILC Server: GNU General Public License
- SILC Toolkit: GNU General Public License

If you have special licensing requirements, such as you are going to distribute one of the SILC distributions as commercial product, or you need to

include one of the SILC distributions (such as SILC Toolkit) into your commercial product, or you need to use some parts of the SILC distributions (such as SILC Toolkit) within your commercial product, it is suggested that you would contact us to discuss the licensing policy and other possible licensing methods for the SILC distributions. Note that these licenses apply only to the SILC software. The actual SILC protocol specification is open specification and is freely available from this site and from the IETF.

CONCLUSION

The Secure Internet Live Conferencing (SILC) protocol is a new generation chat protocol that provides all the common conferencing services with strong support for security. It has wide range of security properties that should meet the highest levels of security requirements, while not forgetting easy of use. The network topology offers new architectural solution with better scalability over traditional chat protocols.

REFERENCES

- www.itpapers.com

More detailed information about the SILC protocol is available in the SILC protocol specification documents. There exists currently four Internet Drafts that defines the protocol in great detail. The Internet Drafts are available from the following sources but also from the IETF website.

- Secure Internet Live Conferencing (SILC), Protocol Specification
- SILC Packet Protocol
- SILC Key Exchange and Authentication Protocols
- SILC Commands

CONTENTS

1. Introduction
2. About SILC
Distribution
3. History
4. Features of SILC
5. SILC Protocol
Clients
Servers
Routers
SILC Packet Protocol
SILC Key Exchange Protocol
SILC Connection Authentication Protocol
Channels
Channel Message Delivery
Channel Message Delivery with Channel Private Key
Private Messages
Private Message Delivery with Session Keys
Private Message Delivery with Private Message Key
Private Message Delivery with Public Key Encryption
Secure File Transfer
6. Future of Protocol
7. Licensing of SILC Software
8. Conclusion
9. References

ABSTRACT

The purpose of this paper is to give short but deep enough introduction to the SILC Protocol. The document describes the purpose of the protocol and how the protocol works in practice.

Chat protocols are very popular on the Internet. They have actually been very popular since the very first chat protocols appeared on the net. The Internet Relay Chat (IRC) was one of the first chat protocols, and quickly gained the status of being the most popular chat on the net. Today, IRC has several competitors from various other so called Instant Messaging (IM) protocols, such as ICQ. However, all of these different chat protocols have something in common; they are all insecure.

The Secure Internet Live Conferencing (SILC) protocol is a new generation chat protocol which provides full featured conferencing services, just like any other contemporary chat protocol provides. In addition, it provides security by encrypting and authenticating the messages in the network. The security has been the primary goal of the SILC protocol and the protocol has been designed from the day one security in mind. All packets and messages travelling in the SILC Network are always encrypted and authenticated. The network topology is also different from for example IRC network. The SILC network topology attempts to be more powerful and scalable than the IRC network. The basic purpose of the SILC protocol is to provide secure conferencing services. The SILC Protocol have been developed as Open Source project. The protocol specifications are freely available and they have been submitted to the IETF. The very first implementations of the protocol are also already available.