

# SILC Server Configuration

## v0.9.2

Mika Boström <bostik@lut.fi>

2003-12-18

# Contents

<b>1 Prerequisites</b>	<b>3</b>
1.1 Silc-server has been installed . . . . .	3
1.2 Basic knowledge and experience of SILC . . . . .	3
1.3 Root access . . . . .	3
<b>2 Introduction</b>	<b>4</b>
2.1 Why SILC? . . . . .	4
<b>3 Basic configuration</b>	<b>5</b>
3.1 Create an account for silcd . . . . .	5
3.2 Create a SILC keypair for silcd . . . . .	5
3.3 Copy your own SILC public key to silc-server's adminkey . . . . .	5
<b>4 Configure your silcd</b>	<b>6</b>
4.1 Enable Perfect Forward Security . . . . .	6
4.1.1 What is Perfect Forward Security (PFS) anyhow? . . . . .	6
4.2 Change detachment timeout . . . . .	6
4.2.1 Detach? Timeout? . . . . .	6
4.3 Set your server's information in place . . . . .	6
4.3.1 What information is that? . . . . .	7
4.4 Change to server's user account . . . . .	7
4.4.1 Who is this SILC user and what group is he in? . . . . .	7
4.5 Give yourself administrator privileges in silc-server . . . . .	7
4.5.1 Why define administrator only by his public key? . . . . .	7
4.6 Comment out the ServerConnections and RouterConnections . . . . .	8
4.6.1 But I plan to link my SILC servers . . . . .	8
<b>5 Run silcd</b>	<b>9</b>
<b>6 Document history</b>	<b>10</b>

# 1 Prerequisites

This document is written with the following assumptions:

## 1.1 Silc-server has been installed

Installing a silc-server is not difficult and is covered in full detail in silc-server's own documentation.

## 1.2 Basic knowledge and experience of SILC

You need to have some experience of using SILC. Knowledge of the protocol is not required but you must have used a SILC client and have your SILC keypair at hand.

## 1.3 Root access

The SILC server (silcd) binds to TCP port 706. Hence running a silc-server requires either root privileges or explicitly granted capabilities to bind a process to lower ( $\leq 1023$ ) ports.

Also, if the installation process did not add a user and group for silcd, you need to add these. Only root can add or remove users on \*nix systems.

## 2 Introduction

(The following text is copied verbatim, sans typographic corrections, from Silcnet's description.)

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.

### 2.1 Why SILC?

Simply put: conversing information online is easy and straight-forward. What you do or say should be nobody's business but yours.

### 3 Basic configuration

When silc-server is installed, it should create a sample configuration file at `/etc/silc/silcd.conf`. This file is a template and should not be even attempted to use as it is.

Before starting, make sure you have user and group silcd created on your system. If using binary packages, they should add silcd account. If the account and group are missing because of incomplete packaging or you built silc-server yourself, just follow the instructions.

The following examples assume silc-server was installed in its default location, inside `/usr/local/silc` hierarchy.

#### 3.1 Create an account for silcd

If you already have silcd account set up, skip this step.

First, create a group silcd:

```
# groupadd silcd
```

Then create a user account silcd that belongs to group silcd:

```
# useradd -g silcd -s /bin/sh -d /usr/local/silc silcd
```

#### 3.2 Create a SILC keypair for silcd

SILC servers are entities as much as clients. As each SILC entity needs an identity, they must have unique keypairs for each. All SILC keys have an identifier of the following format:

```
UN=<user name>, HN=<hostname>, RN=<real name>, E=<email address> C=<2-letter country code>
```

Make sure you have write privilege to `/usr/local/silc/etc` and issue the following command:

```
% /usr/local/silc/sbin/silcd -C /usr/local/silc/etc --identifier="UN=[your login name], HN=[server FQDN], RN=[your real name], E=[your email address], C=[2-letter country code of your country]"
```

The argument after `-C` tells silcd which directory to output the keys.

#### 3.3 Copy your own SILC public key to silc-server's adminkey

If you are planning to run a silc-server, you have no doubt used SILC before. Thus, you have a keypair of your own already. Server administrators can be identified by several methods, but using a public-key authentication is by far the best solution.

Users' public keys are invariably stored in their silc-client directories. In your home directory, under `~/.silc` there should be a file named `public_key.pub` if your silc-client has generated it for you.

Create a special directory for silc-server's administrators' public keys:

```
# mkdir /etc/silc/adminkeys
```

Then, copy your personal public key to this directory, under a name `admin.pub`:

```
# cp /home/<your login>/.silc/public_key.pub /etc/silc/adminkeys/admin.pub
```

## 4 Configure your silcd

Now it's time to tweak your silc-server's configuration. The very basic setup doesn't require you to change more than a few lines. Each modified line will be explained separately. So, edit `/etc/silc/silc.conf` and change the following lines, in order from top to bottom.

### 4.1 Enable Perfect Forward Security

Find the line that has string `key_exchange_pfs`, uncomment it and change it to read

```
key_exchange_pfs = true;
```

#### 4.1.1 What is Perfect Forward Security (PFS) anyhow?

During cryptographically protected sessions it is both common and advisable to change used sessions keys every once in a while. SILC does this by itself, hidden from users' eyes. In active cryptographic sessions it is sometimes desirable to speed up these rekeying events. In such a case, the new session key is derived from old key and thus is vulnerable to exhaustive searches.

PFS is simply a mode of operation where each rekey happens individually and thus keys do not have any mathematical relation to one another. Downside of this is that rekeyings take longer and are computationally more expensive, as new session keys require parties to exchange new key material.

### 4.2 Change detachment timeout

Next up, we need to enable a timeout for detached SILC sessions. Find the line that has a setting `detach_timeout`, uncomment it and change it:

```
detach_timeout=2160; # 36 hours
```

#### 4.2.1 Detach? Timeout?

SILC as a protocol allows user to maintain his or her presence while not physically connected to SILC network. Normally a user simply issues a QUIT and leaves network. It is however possible to detach a session, disconnect the client (for instance when upgrading to newer version) and then reattach to the same session. For other users and the SILC network in general the user never actually left the network. The client simply was off-line for some time.

Without timeout, a detached client session would stay visible and "present" indefinitely. This setting allows detached but never reattached sessions to die off eventually. This frees some server resources since it can eliminate completely stale client entries.

The setting also naturally limits the time a user can be detached before his or her detached session and presence is terminated.

### 4.3 Set your server's information in place

If you plan on running a SILC server, you certainly want to have the server show the correct information. Find a subsection named `ServerInfo`, and change the following lines:

```
hostname = "[your server's FQDN]";
```

```
ip = "[your server's IP address]";
```

Few lines down there are four lines more that you need to change:

```
ServerType = "[Purpose of your server]";
Location = "[geographic location of server]";
Admin = "[SILC server administrator's full name, ie. your's]";
AdminEmail = "[SILC server administrator's email address]";
```

#### 4.3.1 What information is that?

Those six lines carry all the information your server tells your users about the setup. Hostname and IP are naturally the Internet address.

ServerType is merely an informal string that tells what your server does. It could be an internal SILC server or it could be part of a private SILC network. Likewise, Location only tells where the server is located. It usually contains city and country.

Admin and AdminEmail are informational fields that present the user the information about who exactly is running the server.

## 4.4 Change to server's user account

In step 3.1 you created a dedicated user account for silc-server, named silcd. Find and change the following two configuration lines:

```
User = "silcd";
Group = "silcd";
```

#### 4.4.1 Who is this SILC user and what group is he in?

Silc-server does not run as root, unless explicitly told to run in debug mode. It only requires root privileges when starting up. Whenever silc-server starts, it first binds the port it listens on (TCP 706) and then immediately changes from root to specified user and group. This way the possible damage a malfunction or even a security-related bug gets limited by a wide margin, compared to what would happen if SILC server ran as root.

## 4.5 Give yourself administrator privileges in silc-server

Almost at the end of the configuration file there is a section that starts with Admin {. Jump there and edit the lines inside this section to reflect your own identification data. We are only going to use public key authentication, everything else is redundant.

```
# Host = " ";
# User = " ";
# Nick = " ";
# Passphrase = "[anything at all, a random string is good]";
PublicKey = "/etc/silc/adminkeys/admin.pub";
```

#### 4.5.1 Why define administrator only by his public key?

In general it would be good to limit administrator's identification as much as possible, but in SILC each person can be identified reliably with their public keys by using a simple challenge-response authentication. Hence it is enough to specify only the administrator's public key to use for authentication.

This also allows the administrator to use his privileges from anywhere with a personal laptop and still claim the privileges securely. The defined location of the admin's public key is, incidentally, the same where you copied your own public key in step 3.3.

## **4.6 Comment out the ServerConnections and RouterConnections**

It is very likely that you want to setup a stand-alone SILC server at first. It does not require nor have access to other SILC servers, so any configuration options regarding such connections need to be removed.

### **4.6.1 But I plan to link my SILC servers**

Building a SILC network is not a basic task. How this is done, suits far better to advanced silc-server configuration.

## 5 Run silcd

Now that silc-server has been properly configured, you can start it. As root, execute the silcd binary:

```
# /usr/local/silc/sbin/silcd
```

It should start without any errors. After this, you can connect to defined server address with a silc-client and claim operator privileges with the OPER command.

## **6 Document history**

2003-12-18: Wrote version 0.9.2

2003-10-20: Wrote version 0.9.1

2003-09-21: Wrote version 0.9.