

# User's Guide

## *Cornfed SIP User Agent*

Version 0.1.3

Cornfed Systems  
<http://www.cornfed.com>



# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Installation</b>	<b>5</b>
<b>3</b>	<b>Product Overview</b>	<b>7</b>
3.1	Features . . . . .	7
3.2	Requirements . . . . .	8
<b>4</b>	<b>Basic Operation</b>	<b>9</b>
4.1	Initiating a Call . . . . .	10
4.2	Answering an Incoming Call . . . . .	12
4.3	Ending a Call . . . . .	13
<b>5</b>	<b>Configuration</b>	<b>15</b>
5.1	Local Endpoint Settings . . . . .	16
5.2	Dialog State . . . . .	17
5.3	RTP . . . . .	18
5.4	Sound Card . . . . .	19
5.5	Configuring a Residential Gateway . . . . .	19
5.6	Turning NAT Translations Off . . . . .	21



# Chapter 1

## Introduction

Welcome and thank you for choosing the *Cornfed SIP User Agent*. The *Cornfed SIP User Agent* is a Session Initiation Protocol (SIP) based softphone for your IBM-compatible Personal Computer running the Red Hat 9 distribution of the Linux operating system. The *Cornfed SIP User Agent* allows you to make Internet-based telephone calls using the Advanced Linux Sound Architecture (ALSA) or Open Sound System (OSS) sound card with speakers and microphone as your telephone handset.

The *Cornfed SIP User Agent* was written by Frank W. Miller, a veteran of two Voice-over-IP equipment vendor startup companies. The goal is to produce a client that is rock-solid in its operations and available on a variety of different platforms. The initial release is focused on the Linux operating system in order to develop and mature the SIP, RTP, and voice codec components of the product. Later releases will port the client to other operating systems and add features and Graphical User Interface elements.

The *Cornfed SIP User Agent* is provided as a binary distribution only. The binary executable program is provided free of charge for personal use for users of the Linux operating system. For other terms of use, please contact Cornfed Systems.



# Chapter 2

## Installation

The *Cornfed SIP User Agent* is distributed in a compressed `tar` file. A distribution file with name similar to `cornfedsipua.tgz` contains the binary executable file and other files including this document. To extract the files, type following command:

```
$ tar xzvf cornfedsipua.tgz
cornfedsipua/
cornfedsipua/sip
cornfedsipua/user.pdf
$
```

The result is a directory called `cornfedsipua` that contains all the files in the distribution. In this example, the file named `sip` is the executable binary and `user.pdf` is an Adobe Portable Document Format (PDF) file containing this User's Guide.

The `sip` binary file needs to be copied to a directory that contains binary executable files, e.g. `/usr/local/bin`. The important point about the directory to which the binary is to be copied is that it needs to be part of the path of any users that want to use the program. To copy the binary file, execute the following commands:

```
$ cd cornfedsipua
$ su
Password: <Enter root password>
#
# cp sip /usr/local/bin
# exit
$
```

This sequence copies the binary executable to the `/usr/local/bin` directory. Note

that the copy requires the user to be logged in as the superuser. The User's Guide PDF file can be placed anywhere in the system as required to make it available for users to reference.

This completes the installation process. To run the client program type `sip` at the Linux shell prompt. There are currently no command line arguments.

# Chapter 3

## Product Overview

This chapter provides a high-level view of the *Cornfed SIP User Agent* product. It specifies the available features and the set of software and hardware requirements that a user must meet in order to run the client.

### 3.1 Features

- Supported protocols: SIP (RFC 3261), SDP (RFC 2327), RTP (RFCs 3550 and 3551)
- Automatic detection of Residential Gateways using Network Address Translation (NAT)
- Interoperable with Grandstream HandyTone-286 and Asterisk PBX devices
- Support for G.711  $\mu$ -Law voice codec
- Supports re-INVITES for changes to media transport
- Flexible command-line interface
- Multi-threaded implementation

## 3.2 Requirements

- An IBM-compatible (Intel *x86*-based) Personal Computer running the Red Hat 9 distribution of the Linux operating system
- An ALSA or OSS compatible sound card (tested with AC97-compatible and SoundBlaster-compatible sound cards)
- A high-speed Internet connection. This will most likely take the form of a Local-Area Network (LAN) or Cable or Digital Subscriber Line (DSL) broadband connection.
- *Pthreads* library installation

# Chapter 4

## Basic Operation

The *Cornfed SIP User Agent* is started by executing the binary file provided in the distribution. Once the program is started you will see something similar to the following output:

```
$ sip

Cornfed SIP User Agent
Copyright (C) 2004 Cornfed Systems
Written by Frank W. Miller
```

```
sip>
```

Any time you see the `sip` prompt, you can enter any of the available commands. If you type the `local` command you will get a list of the local endpoint URI parameters. An example is given below:

```
sip> local

user      : [fwmiller]
host      : [192.168.1.103]
visible   : [128.158.13.92]
port      : 5060
```

This command provides you with the `user` name, `host` and `visible` IP addresses, and SIP UDP `port` associated the *Cornfed SIP User Agent* client on your computer. *The Cornfed SIP User Agent currently handles IP addresses only, domain names*

*not currently supported.* The `host` address is the IP address associated with the `eth0` network interface. The Cornfed SIP User Agent *currently assumes eth0 for its network connection.* The `visible` address is the IP address that other computers on the Internet see when communicating with your computer. This address may be different than the `eth0` address if you are using a Residential Gateway that uses NAT to connect to the Internet.

By default, the executable program is distributed with debugging turned off. When debugging is enabled, a significant amount of output is dumped to the screen when placing phone calls, primarily dumps of incoming and outgoing SIP messages. Debugging can be turned on by executing the following command:

```
sip> debug on
sip>
```

If you want to turn debugging back off, execute this command:

```
sip> debug off
sip>
```

If you want to see whether debugging is turned on or off, execute this command:

```
sip> debug
debug is on
sip>
```

## 4.1 Initiating a Call

To initiate an Internet phone call, you must set the remote endpoint and then dial. The remote endpoint is the user and Internet host to which you make phone calls. A SIP endpoint is generally labeled by a Uniform Resource Indicator (URI) that looks something like this:

```
sip:fwmiller@192.168.1.103:5060
```

The `sip:` element is a prefix indicating that standard SIP signaling is to be used. The other elements comprise an endpoint label. The `fwmiller` element is called the *user*, the `192.168.1.103` element is called the *host*, and the `5060` element is called the *port*.

To see the current remote endpoint settings, type the following command:

```
sip> remote
```

```
user : [fwmiller]
host  : [192.168.1.103]
port  : 5060
```

The `user` field contains the SIP URI `user` element, the `host` field contains the SIP URI `host` element, and the `port` field contains the SIP URI `port` element.

To change the `user` field to `CornfedSIPUA`, type the following commands:

```
sip> remote user CornfedSIPUA
sip> remote
```

```
user : [CornfedSIPUA]
host  : [192.168.1.103]
port  : 5060
```

Observe that the `user` field has been changed to `CornfedSIPUA`.

The `host` and `port` fields can be changed similarly:

```
sip> remote host 65.202.222.192
sip> remote port 5061
sip> remote
```

```
user : [CornfedSIPUA]
host  : [65.202.222.192]
port  : 5061
```

Observe that the `host` and `port` fields have been changed as specified.

Once the remote endpoint is set, you simply issue a `dial` command to initiate a phone call.

```
sip> dial
sip>
```

This command initiates a call to the specified remote endpoint. If the other side answers, the call is connected and you can begin a conversation with the other party.

## 4.2 Answering an Incoming Call

It is also possible for another IP phone user to call your client. If they know the local user, host, and port associated with your *Cornfed SIP User Agent* client, they can make calls to you. You can use the `local` command to see the local endpoint settings. Similar to the `remote` command, the following command shows the local endpoint settings:

```
sip> local

user      : [CornfedSIPUA]
host      : [192.168.1.103]
visible   : [141.157.12.97]
port      : 5062
```

Note that this output includes the visible IP address for the host on which the client is running. If you try to make calls through a Residential Gateway that uses NAT, this address will be used in the appropriate ways to allow calls to be made through the NAT translation.

When another party calls your client, you will see a line similar to the following displayed periodically on your terminal:

```
incoming call from sip:user@192.168.1.101:5061
```

When you see this line appear, you have two choices, you can answer the call or refuse the call. If you answer the call, a voice connection is made and you can begin your conversation. If you refuse the call, the other party is told that you are not currently available to take the call. To answer the call use the following command:

```
sip> answer
call answered
sip>
```

To refuse the call, use the following command:

```
sip> refuse
sip>
```

## 4.3 Ending a Call

To end a call in progress, use the **hangup** command. At the command line, issue the following command:

```
sip> hangup  
sip>
```

The call is terminated and the conversation in progress is ended, i.e. the speakers go silent and the microphone becomes inactive. If the other party hangs up, the only indication is the conversation being discontinued.



# Chapter 5

## Configuration

This chapter provides descriptions of the remaining *Cornfed SIP User Agent* commands. At the command line, you can type a ? (or any other invalid command) and a list of all available commands will be displayed:

```
sip> ?

answer      : Answer incoming call
debug       : Debug settings
dialog      : Dialog state
dial        : Initiate call
hangup      : Terminate call in progress
local       : Local endpoint settings
nat         : Local endpoint NAT settings
soundcard   : Soundcard settings
refuse      : Refuse to accept an incoming call
remote      : Remote endpoint settings
reset       : Clear dialog state
rtp         : RTP settings
```

Several of these commands were introduced in the previous chapter. Descriptions of the remaining commands are given here.

An important command to be aware of is the **reset** command. The **reset** command simply clears the SIP dialog state. This is useful when your client and the other endpoint have gotten out of sync. Any retransmissions or waiting that your client is doing will be ended. In addition, it may be necessary to use the **soundcard**

`flush` command, which causes the sound card input buffer to be cleared. The *Cornfed SIP User Agent* client maintains a high-water mark on the input buffer and will automatically flush it if too much input data has been queued. However, it may be useful to issue this command if you hear strange sounds or not sound at all. These two commands can generally be issued as often as needed in order to return the client to an idle state.

## 5.1 Local Endpoint Settings

The local endpoint settings are used to identify your user name and IP host address and port to parties that you call. To see the current local endpoint settings, type the following command:

```
sip> local

user      : [fwmiller]
host      : [192.168.1.103]
visible   : [128.158.13.92]
port      : 5060
```

The `user` field contains the local SIP URI `user` element and the `port` field contains the local SIP URI `port` element. Note that both the local `host` IP address (associated with the `eth0` network interface) and the `visible` IP address are shown. One or the other of these addresses will be used as the local SIP URI `host` element, depending on whether you are calling across a Residential Gateway that uses NAT.

To change the `user` field to `CornfedSIPUA`, type the following commands:

```
sip> local user CornfedSIPUA
sip> local

user      : [CornfedSIPUA]
host      : [192.168.1.103]
visible   : [128.158.13.92]
port      : 5060
```

Observe that the `user` field has been changed to `CornfedSIPUA`.

To change the port field to 5061, type the following commands:

```
sip> local port 5061
sip> local

user      : [CornfedSIPUA]
host      : [192.168.1.103]
visible   : [128.158.13.92]
port      : 5061
```

Observe that the `port` field has been changed to 5061.

It is not possible to change the `host` or `visible` fields from the command line. These fields are obtained automatically.

## 5.2 Dialog State

The `dialog` command can be used to show the current state of the SIP dialog between your client and another SIP endpoint. The following is an example output just after a call that was initiated by your client has been connected:

```
sip> dialog

state      : SIPS_CONNECTED
Via host    : []
Via port    : -1
Via branch  : [z9hG4bKEqmJrsDu2]
Call-ID     : [nLwDmyz4Evz3]
local tag   : [y2N0nG6p]
remote tag  : [e29450da9e627139]
local seq   : 1
remote seq  : -1
local URI   : [sip:CornfedSIPUA@192.168.1.103:5062]
remote URI  : [sip:fwmiller@192.168.1.101]
remote target : []
last response :
[]
```

In this example, the `state` field indicates the call is connected. The branch tag value and local and remote tag values are stored as well as the Call-ID. The local and

remote CSeq numbers are stored. Finally, the local and remote URIs, used in the From and To headers, respectively, are shown.

The next example is similar, except that the call was initiated by the remote party.

```

sip> dialog

state      : SIPS_CONNECTED
Via host   : [192.168.1.103]
Via port   : 5060
Via branch : [z9hG4bK1c5835f5]
Call-ID    : [3ca55d100fd06ccfaa8c@192.168.1.103]
local tag  : [gv44RLti]
remote tag : [as0126da38]
local seq  : -1
remote seq : 103
local URI  : [sip:cornfedSIPUA@192.168.1.103:5062]
remote URI : [sip:1921681101@192.168.1.103]
remote target : [sip:1921681101@192.168.1.103]
last response :
[]

```

The output shows that the remote CSeq number is filled in rather than the local CSeq number and we also show the remote target (taken from the Contact header of the incoming INVITE).

### 5.3 RTP

The `dialog` command provides information on the SIP dialog signaling state. The `rtp` command shows information regarding the RTP audio transport associated with a call. The following shows an example output of the `rtp` command:

```

sip> rtp

local host  : [192.168.1.103]
local port  : 9096
remote host : [192.168.1.101]
remote port : 11900

```

In this example, a call is in progress between 192.168.1.103:9096 (our local host IP address and port) and 192.168.1.101:11900 (the remote host IP address

and port). The *Cornfed SIP User Agent* uses port 9096 for its RTP port. *This port number cannot be changed in the current release.* The remote party specifies the IP address and port of its side of the voice call in the SDP associated with the SIP signaling for the call.

## 5.4 Sound Card

In addition to flushing the sound card using the `soundcard flush` command, information about the soundcard status can be obtained. Issue a simple `soundcard` command to generate this data:

```
sip> soundcard

soundcard input buffer
avail fragments : 112
total fragments : 256
fragment size   : 256
avail bytes     : 28608

soundcard output buffer
avail fragments : 9
total fragments : 9
fragment size   : 256
avail bytes     : 2304
```

This information describes the parameters associated with the input and output channels of the sound card device. *The Cornfed SIP User Agent currently assumes the use of the /dev/dsp device for all sound card operations.*

## 5.5 Configuring a Residential Gateway

The *Cornfed SIP User Agent* has the ability to operate behind a Residential Gateway that uses Network Address Translation (NAT) and that has firewall functionality. Figure 5.1 illustrates a typical network topology when using a Residential Gateway with NAT.

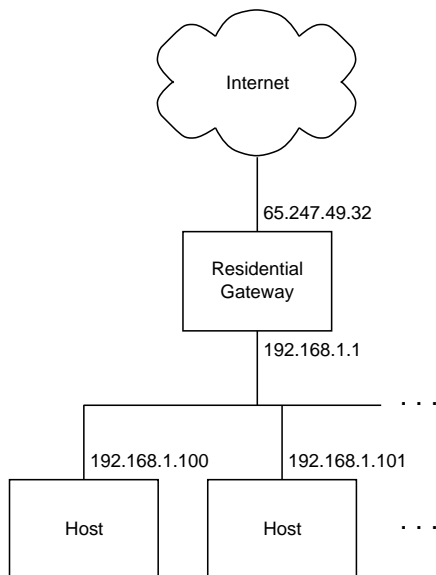


Figure 5.1: A Typical Residential Gateway NAT environment

In this example, any of the computer hosts sitting *behind* the Residential Gateway, i.e. on the private 192.168.1.X subnet, can make use of the *Cornfed SIP User Agent* to talk to parties on the Internet. *In the current release, only one of the hosts can use the client simultaneously.* Each client automatically detects the visible IP address seen by hosts on the Internet, in this case, the 65.247.49.32 IP address.

NAT detection is automatic and makes use of a dedicated Internet web site called <http://www.myipaddress.com>. The client queries this website for the IP address it sees and uses that as the visible IP address on the Internet.

In addition to NAT, Residential Gateways generally provide a *firewall* function. That is, they limit the types of traffic originating from the open Internet that can reach hosts on the private subnet. To get around these limitations, you must make two changes to your Residential Gateway configuration for each client you wish to use on the private subnet. For each client, two *pinholes* must be opened in the Residential Gateway firewall, one for SIP traffic and one for RTP traffic.

By default, the *Cornfed SIP User Agent* uses UDP port 5060 for SIP traffic and UDP port 9096 for RTP traffic. You must configure the Residential Gateway firewall to allow UDP traffic coming from the Internet with these two destination UDP port numbers to be routed to the IP address of the host on which you wish to run the

*Cornfed SIP User Agent client. The limitation on changing the RTP port number in the client is the reason why only a single client can be used on the local subnet in the current release.*

## 5.6 Turning NAT Translations Off

By default, the executable program is distributed with NAT translation capability turned on. It may be necessary to disable this feature for your environment. NAT translations can be turned off by executing the following command:

```
sip> nat off
sip>
```

If you want to turn NAT translations back on, execute this command:

```
sip> nat on
sip>
```

If you want to see whether NAT translations are turned on or off, execute this command:

```
sip> debug
debug is on
sip>
```