SER HowTo

Dan Austin

Edited by

Nils Ohlmeier

SER HowTo

Edited by and Dan Austinand Nils Ohlmeier

Table of Contents

List of Tables

Chapter 1. Introduction

1.1. Revision

- Version 0.1 12/02/2002 Dan Austin
- Version 0.2 12/03/2002 Dan Austin
- Version 1.0 12/06/2002 Dan Austin
- Version 1.1 12/16/2002 Dan Austin
- Version 2.0 10/04/2003 NSI Ltd.

This document is a result of a merge of two documents http://www.fitawi.com/ser-Howto.html and http://cvs.berlios.de/cgi-bin/viewcvs.cgi/*checkout*/ser/sip_router/INSTALL?rev=ser_0_8_10&content-type=text/plain with some add-ons, made by NSI
team.

1.2. Why SER

SER is an open-source project that aims to make available a fully functional and scalable Session Initiated Protocol server. Call processing is described with a concise scripting language that offers the flexibility of regular expressions and the ability to interface with 3rd party applications for the purposes of call accounting and authorization.

1.3. Where to get SER

SER is available for download from ftp://ftp.berlios.de/pub/ser

The newest release may be found in the folder /latest

Chapter 2. Installation

2.1. Installation Notes

Supported architectures:

- Linux/i386
- Linux/armv4l
- FreeBSD/i386
- OpenBSD/i386
- Solaris/sparc64
- NetBSD/sparc64

(For other architectures the Makefiles might need to be edited) There are various configuration options defined in the Makefile and Makefile.defs.

2.2. Requirements

- gcc or icc : gcc >= 2.9x; >=3.1 recommended (it will work with older version but it might require some options tweaking for best performance)
- bison or yacc (Berkley yacc)
- flex
- GNU make (on Linux this is the standard “make”, on FreeBSD and Solaris is called “gmake”)
- sed and tr (used in the make files)
- GNU tar (“gtar” on Solaris) and gzip if you want “make tar” to work.
- GNU install or BSD install (on Solaris “ginstall”) if you want “make install”, “make bin”, “make sunpkg” to work.
- “mysql” if you need MySQL support.
- “Apache (httpd)” if you want serweb support
- “PHP, MySQL-PHP” for serweb support
- libmysqlclient & libz (zlib) if you want mysql support (the mysql module)
- libexpat if you want the jabber gateway support (the jabber module)

Installing SER on a RedHat Linux distribution for example, is a simple matter of unzipping the downloaded file and using your favorite package manager.

2.3. Install the package

Example:

```
/root>rpm -i ser-08.11-1.i386.rpm
```

Packages for other popular distributions are available, and can be installed using the appropriate package manager for that distribution.

On many platforms you can start the service with:

```
/etc/init.d/ser start
```

RedHat systems will use:

```
/etc/rc.d/init.d/ser start
```

You now have a functioning SIP server, but what can you do with it? At this point not very much. With an SIP client, such as Microsoft MSN Messenger 4.6, you can register with the server, send Instant Messages to other logged on clients of the same server, and even have voice conversations with them.

That sounds pretty good, but maybe you'd like to add a little more security, or make you server accessible to others.

2.4. Serctl Utility

To do so, first set the environment variable SIP_DOMAIN to your domain name, e.g., in Bourne shell (bash), call:

```
export SIP_DOMAIN="foo.bar"
```

If you wont the system to created this variable automatically, you need to add the line

```
export SIP_DOMAIN="foo.bar"
```

in the end of file /etc/profile.

If you are using other than 'localhost' mysql server for maintaining subscriber database, change the variable 'SQL_HOST' to the proper host name in the serctl script.

Run the serctl utility

```
/usr/sbin/serctl monitor
```

If you installed from a tar.gz or Solaris package:

```
/usr/local/sbin/serctl monitor
```

2.5. DNS SVR Resource Records

It is important that your SIP clients can connect to your server for purposes of registration and call control. You might even want to have a redundant server to handle calls if your primary server is unavailable.

These requirements can be meet by using DNS SVR Resource Records, available in BIND 8.X and up releases.

The format for a SVR RR is this:

```
_service._protocol        SVR Priority Weight     Port hostname
```

In this case we want to establish an entry for our primary SIP server, gateway.mydomain.com, that will listen on UDP port 5060. The entry will look like this:

```
_sip._udp         SRV     0  0   5060  gateway.mydomain.com
```

Placement of the new resource record is important. Here is a sample zone file:

```
; zone 'mydomain.com'   last serial 1998071308
$ORIGIN com.
mydomain  86400           IN      SOA     gateway.mydomain.com. postmaster.mydomain.com. (
                                        1998111908 ; Serial
                                        36000 ; Refresh
                                        900 ; Retry
                                        36000 ; Expire
                                        28800 ); Minimum
                IN      NS              gateway.mydomain.com.
                IN      NS              ns3.backupdomain.com.
                IN      MX              1 gateway.mydomain.com.
                IN      A               192.168.0.1

;If we place the SRV record above the next line it fails to load
$ORIGIN fitawi.com.
_sip._udp               SRV  0 0  5060  gateway.mydomain.com.
gateway         IN      A               192.168.0.1
www             IN      CNAME           gateway.mydomain.com.
```

After reloading your zone file you can verify that the entry is working by using dig.

```
dig -t SRV _sip._udp.mydomain.com
```

The results should look something like this:

```
; <<>> DiG 9.1.0 <<>> -t SRV _sip._udp.mydomain.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 32654
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 1

;; QUESTION SECTION:
;_sip._udp.mydomain.com.          IN      SRV

;; ANSWER SECTION:
_sip._udp.mydomain.com.   86400   IN   SRV   0 0 5060 gateway.mydomain.com.

;; AUTHORITY SECTION:
mydomain.com.             86400   IN      NS      ns3.elsewhere.com.
mydomain..com             86400   IN      NS      gateway. mydomain.com.

;; ADDITIONAL SECTION:
gateway. mydomain.com.     86400   IN      A       192.168.0.150

;; Query time: 6 msec
;; SERVER: 192.168.0.150#53(192.168.0.150)
;; WHEN: Tue Dec  3 08:34:17 2002
;; MSG SIZE  rcvd: 132
```

2.6. Adding a database for client information

By leveraging a MySQL database, we can provide support for user credentials, and keeping track of where the clients are logged on during server restarts.

2.7. MySQL setup

To install support for a MySQL database you will need to download the package ser-mysql, which is available from the same download location that you retrieved SER. This package has scripts to create the required database and establish permissions for the accounts needed. A recent release of MySQL is recommended. Earlier versions may have problems with the syntax required to set permissions on the database.

If you do not already have a copy of MySQL installed, download it from your http://www.mysql.com

Once you have MySQL installed and started, execute

```
/usr/sbin/ser_mysql.sh
```

You can verify that the database has been created, and correct permissions assigned by using the mysql management tool and these steps:

```
Mysql> select * from user;
| Host                | User  | Password         | Select_priv | Insert_priv | Update_priv | Delet
| %                   | ser   | 4e633cf914a735a0 | N           | N           | N           | N
| localhost           | ser   | 4e633cf914a735a0 | Y           | Y           | Y           | Y
| %                   | serro | 7cb73a267cb7bd5f | N           | N           | N           | N
| localhost           | serro | 7cb73a267cb7bd5f | Y           | N           | N           | N
```

The above results show that the two user, ser and serro, have been created and granted the permissions needed to access the database. Note that in the above example the permissions have been modified to deny access to these accounts from any system(%) other than local host.

```
mysql> connect ser;
Connection id:    294
Current database: ser

mysql> show tables;
+------------------+
| Tables_in_ser    |
+------------------+
| acc              |
| active_sessions  |
| aliases          |
| config           |
| event            |
| grp              |
| location         |
| missed_calls     |
| pending          |
| phonebook        |
| reserved         |
| silo             |
| subscriber       |
| version          |
+------------------+
14 rows in set (0.00 sec)

mysql> select * from subscriber;
| phplib_id                        | USERNAME | PASSWORD | FIRST_NAME | LAST_NAME | PHONE
| 4cefa7a4d3c8c2dbf6328520bd873a19 | admin    | heslo | first        | admin    | 557-8469     |
```

This last query shows that you have one user account defined and it has administrator privileges.

We'll need to add another account to be the administrator for your realm, which we will do after the next section.

Chapter 3. Configuration

3.1. Modify SER configuration

Now that we have a working MySQL database, we need to modify the configuration file for ser, located on a RedHat, installed in /etc/ser/ser.cfg. The following changes need to be made:

To enable support for the new MySQL database we need to load the appropriate module. That is accomplished by uncomment this line:

```
loadmodule "/usr/lib/ser/modules/mysql.so
```

Next we need to set SER to use the database and write changes instead of just caching them in memory. This is done by means of commenting this line:

```
modparam ("usrloc", "db_mode",  0)
```

And uncomment this line:

```
modparam ("usrloc", "db_mode", 2)
```

Note on db_modes:

- Mode 0

  Disables writes to the database. Contact information will not be preserved if the server is restarted.

- Mode 1

  Writes all changes to the database immediately. Contact information is saved to the database immediately. This can slow the response to clients as they connect.

- Mode 2

  Periodically writes contact information to the database based in the in memory cache.

To enable digest authentication we additionaly need to uncomment the following two lines:

```
loadmodule "/usr/lib/ser/modules/auth.so"
loadmodule "/usr/lib/ser/modules/auth_db.so"
```

We have the option of storing passwords in our database in plain text. This allows for password recovery and makes the initial setup and testing easier. To enable this feature uncomment these lines:

```
modparam (“auth_db”, “calculate_ha1”, yes)
modparam (“auth_db”, “password_column”, “password”)
```

These lines work together. The first tells SER to generate a hash based on username, password and realm. The second tells SER where to look for the plain-text password in the database.

Lastly we need to update the routing section to recognize our realm.

Change (uri==“myself”) to (uri=~“mydomain.com”)

Uncomment these lines and change all instances of iptel.org to mydomain.com

```
if (!www_authorize("mydomain.com", "subscriber")) {
        www_challenge("mydomain.com", "0");
        break;
};
```

We’re now ready to restart ser. On RedHat use

```
/etc/rc.d/init.d/ser restart
```

3.2. Adding an admin for your realm

Now that we have a working database and ser is configured to use it, we need to add some users and at least one of them should have administrator privileges. The administrator role becomes important if you want to use a web management tool such as serweb.

Basic account manipulation can be performed with the serctl script, located in /usr/sbin.

To add a user use these commands

```
serctl add JoeUser qwerty joe@mydomain.com
```

The system notify for “Type MySQL Password”, the default password is “heslo”

To make JoeUser an administrator, we need to logon to MySQL and modify the database.

```
mysql> connect ser;

mysql> update subscriber set perms=?admin? where USER_ID=?JoeUser?;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from subscriber;
| 4cefa7a4d3c8c2dbf6328520bd873a19 | JoeUser     | qwerty |  |   |   | joe@mydomain.com   | 2002-
```

The third from last field shows that Joe has been assigned admin privileges.

At this point Joe can logon to our server, but since he is the only user, there is not much he can do. We can now add additional users using the serctl script, or now is a good time to look at installing serweb, which will allow users to subscribe to our service.

3.3. More on serctl

The script serctl can be used to manage users, access control lists, in memory contacts, and to monitor server health. Executing serctl with no arguments will produce this output:

```
usage:
           * subscribers *
 add <username> <password> <email> .. add a new subscriber (*)
 passwd <username> <passwd> ......... change user's password (*)
 rm <username> ...................... delete a user (*)
 mail <username> .................... send an email to a user
 alias show [<alias>] ............... show aliases
 alias rm <alias> ................... remove an alias
 alias add <alias> <uri> ............ add an aliases

           * access control lists *
 acl show [<username>] .............. show user membership
 acl grant <username> <group> ....... grant user membership (*)
 acl revoke <username> [<group>] .... grant user membership(s) (*)

           * usrloc *
 ul show [<username>]................ show in-RAM online users
 ul rm <username> ................... delete user's UsrLoc entries
 ul add <username> <uri> ............ introduce a permanent UrLoc entry
 showdb [<username>] ................ show online users flushed in DB

   * control and diagnostic *
 moni ... show internal status     start .... start ser
 ps ..... show runnig processes    stop ..... stop ser
 fifo ... send raw FIFO commands   restart .. restart ser
 ping <uri> .. ping a URI (OPTIONS)
 cisco_restart <uri> .. restart a Cisco phone (NOTIFY)

   Commands labeled with (*) will prompt for a MySQL password.
```

```
   If the variable PW is set, the password will not be prompted.

     ACL privileges are: local ld int voicemail free-pstn
```

3.4. Adding and deleting users with serctl

User account management is performed with these commands:

```
serctl add
serctl password
serctl rm
```

The contents of the in memory cache can be managed with the ul argument. Care must be taken to with these commands. For example:

```
serctl ul rm joe
```

Will remove the current contact information about Joe from memory

Whereas “serctl rm joe” will delete joe’s account.

3.5. Examining in memory cache with serctl

The command “serctl ul show” will list any currently registered clients. The output will look like this:

```
===Domain list===
---Domain---
name : 'location'
size : 512
table: 0x402ee6d0
d_ll {
    n    : 2
    first: 0x402f1a74
    last : 0x402f089c
}
lock : 0

...Record(0x402f1a74)...
domain: 'location'
aor   : 'test'
~~~Contact(0x402f708c)~~~
domain : 'location'
aor    : 'test'
Contact: 'sip:test@192.168.0.100:5060'
Expires: 2501
q      :       0.00
```

```
Call-ID: '000a8a93-d4660017-4571a6cd-658ac1bf@192.168.0.100'
CSeq   : 101
State  : CS_SYNC
next   : (nil)
prev   : (nil)
~~~/Contact~~~~
.../Record...
...Record(0x402f089c)...
domain: 'location'
aor   : 'joe'
~~~Contact(0x402f0924)~~~
domain : 'location'
aor    : 'joe'
Contact: 'sip:192.168.0.101:14354'
Expires: 432
q      :       0.00
Call-ID: 'e8d93059-e46e-4fd9-958b-ccb36a1cf245@192.168.0.101'
CSeq   : 11
State  : CS_SYNC
next   : (nil)
prev   : (nil)
~~~/Contact~~~~
.../Record...

---/Domain---
===/Domain list===
```

3.6. Examining server status

Two commands can be used to check the health of the server. The first command serctl ps returns a list of all SER related processes, the ip address and the port they are listening on. For example:

```
[root@gateway /root]# serctl ps
0       31029   attendant
1       31033   receiver child=0 sock=0 @ 127.0.0.1::5060
2       31034   receiver child=1 sock=0 @ 127.0.0.1::5060
3       31035   receiver child=2 sock=0 @ 127.0.0.1::5060
4       31036   receiver child=3 sock=0 @ 127.0.0.1::5060
5       31037   receiver child=0 sock=1 @ 192.168.0.1::5060
6       31038   receiver child=1 sock=1 @ 192.168.0.1::5060
7       31039   receiver child=2 sock=1 @ 192.168.0.1::5060
8       31040   receiver child=3 sock=1 @ 192.168.0.1::5060
9       31049   fifo server
10      31072   timer
```

The second command, serctl monitor, shows the server version, uptime, pending and completed transactions, and the number of major category responses the server has sent. Another example:

```
[cycle #: 1; if constant make sure server lives and fifo is on]
Server: Sip EXpress router (0.8.11 (i386/linux)
Now: Wed Dec  4 10:13:02 2002
Up Since: Mon Dec  2 21:21:11 2002
Up time: 132711 [sec]

Transaction Statistics
Current: 0 (2 waiting) Total: 46 (0 local)
Replied localy: 37
Completion status 6xx: 0, 5xx: 0, 4xx: 23, 3xx: 0,2xx: 22

Stateless Server Statistics
200: 101 202: 0 2xx: 0
300: 0 301: 0 302: 0 3xx: 0
400: 0 401: 0 403: 0 404: 132 407: 0 408: 0 483: 1 4xx: 0
500: 0 5xx: 0
6xx: 0
xxx: 0
failures: 0

UsrLoc Stats
Domain Registered Expired
'location' 2 2
```

Chapter 4. Installing SERweb

The SERweb package can be downloaded from ftp://ftp.berlios.de/ser/latest/serweb

The pages associated with SERweb provide a starting point to customize your SIP user account management tools.

4.1. PHP configuration

Go to PHP configuration file in /etc/php.ini and change "register_globals = Off" to "On"

4.2. Installing SERweb default configuration

If you are installing this package on a server that does not host any other web pages, you can simply extract the files to the document directory of your web server. This presumes that you have a working Web Server.

4.3. Installing SERweb custom file locations

In case your server performs multiple functions, and you want to just add the SERweb tools to an existing web site, we will need to make changes to a number of the php files. The following examples are from an Apache 2.0 on RedHat.

Unzip the files into a temporary directory, such as /root/serweb. The directory will contain these files:

```
-rw-rw-r--    1 827      2020        18561 Sep 25 16:31 COPYING
drwxr-xr-x    2 827      2020         1024 Nov 27 16:43 CVS
-rw-rw-r--    1 827      2020          529 Sep 25 16:29 README
drwxr-xr-x    7 827      2020         1024 Nov 27 22:24 html
drwxr-xr-x    3 827      2020         2048 Sep 26 10:26 phplib
```

On the Linux RedHat Version 8 root directory on web server is /var/www/html/

Move the html directory to the root of your web server:

```
mv html /var/www/html/htdocs/serweb
```

Move the phplib directory to your web server application directory:

```
mv phplib /var/www/html/phplib
```

Following files must be updated with this “new” path to the libraries:

```
./admin/prepend.php
./user_interface/prepend.php
./user_interface/reg/prepend.php
```

For these files the variable: $_PHPLIB[“libdir”] = “../../phplib/”; becomes $_PHPLIB[“libdir”] = “../../../phplib/”;

In the ./admin directory edit the files acl.php, index.php, and users.php will need their path to the forms library updated. For this example, add ../ to the existing line

```
:require "../../../phplib/oohforms.inc";
```

In the ./user_interface directory the following files need the same change:

accounting.php, find_user.php, index.php, missed_calls.php, my_account.php, phonebook.php, send_im.php, notification_subscription.php

Next these files in ./user_interface/reg need the same change, with an additional ../:

Finish.php, get_pass.php, index.php

The last changes occur in the config.php file to provide the location for graphic files, style sheets and time zone information. Update the following variables:

```
$this->root_path="/serweb/";
$this->fifo_server = "/tmp/ser_fifo";
$this->zonetab_file =   "/usr/share/zoneinfo/zone.tab";
//TZ zone descriptions file, usually: /usr/share/zoneinfo/zone.tab
```

Find two variables: “ $this->mail_forgot_pass=”,“ $this->mail_register=” and change line http://oook/~iptel/user_interface/reg/confirmation.php?nr=#confirm#\n\n

with line

http://“.$_SERVER[’HTTP_HOST’].”/htdocs/serweb/user_interface/reg/confirmation.php?nr=#confirm#\n\n

This will insure sending a registration feedback mail to SIP server using its IP address

4.4. Modifying SERweb configuration general

We need to update /usr/local/apache/htdocs/serweb/config.php to represent our realm. The following variables need to be changed to our domain:

```
$this->realm="mydomain.com";
$this->domainname=" mydomain.com";
$this->web_contact="sip:JoeUser@ mydomain.com";
//address of pseudo sender
```

```
$this->default_domain=" mydomain.com";
$this->mail_header_from="Registration@ mydomain.com";
```

Additionally we will want to modify the section for Terms and Conditions, either replacing it with appropriate language for our services, or at least replacing iptel.org with our domain information.

Chapter 5. Issues and Limitation

Since one of the design goals behind SIP is to decentralize the intelligence in communications handling, a basic tenant is that SIP clients need to be able to communicate directly with each other. The problem is that many clients find themselves either behind a firewall or in a NAT fronted address space. When a client registers with the SIP server, it tells the server what it is using for an IP address, and that address may not be accessible to the public.

5.1. More on NAT

There are a couple of ways that we can overcome the problem that NAT introduces. Some SIP client providers are building in options into their products that allow the user to identify the IP address that their phone will appear as to the public. This is a nice simple approach, but presumes that the person installing the client knows what that IP address is, and that it doesn?t change. Cisco has built this feature into their 79XX series SIP phones.

A second solution that is working its way through the standards process is called “Simple Traversal of UDP through NAT”, or STUN. A STUN equipped client is configured to send a who-am-I packet to a known server on the public network. That server will respond with the IP address that the client appears to be communicating from, and the client can then use that address to register with the SIP server. Phones that leverage STUN include: Snom 100, kphone, and sipc .

5.2. Firewalls

SIP clients also present an interesting challenge to configuring a firewall. During registration the SIP client will be assigned a UDP port in the range of 16384 to 32768. Our firewall administrators will not happily open up all of those ports to all of the internal systems, on the chance that a SIP connection may be needed.

This is where the concept of a Firewall Control Protocol, or FCP, comes into play. The design idea is that when a SIP client registers, a FCP agent, or server if you prefer, will dynamically insert a new rule into the firewall policy to permit that client to participate in SIP conversations.

Chapter 6. Diagnostics and tools

Detailed information about the communications between clients and the SIP server is needed to isolate problems. Two tools that can be used to gather such information are sipsak and ngrep.

6.1. ngrep

Ngrep is a capable of listening in on network traffic and filtering it in much the same way as grep can locate patterns in files. To monitor the communications between a client, joe, and the server the following command would be run on the server:

```
ngrep  -n 5060 -d eth0 joe
```

Since SIP communications are ASCII based, all events such as REGISTER, INVITE, SUBSCRIBE, etc. are captured. The output of ngrep can identify problems with SIP addresses, or client identity.

Ngrep should be part of most modern distributions, or can be downloaded from http://sourceforge.net/projects/ngrep/

6.2. Sipsak

Sipsak can be used to determine if your server is responding to requests, and provide information on how your server would route SIP connections. Sipsak and basic documentation on its use can be downloaded from http://sipsak.berlios.de

Chapter 7. Client Configuration

7.1. Microsoft Messenger 4.6

Microsoft Messenger 4.6 can be configured as a SIP client by selecting Tools\Options\Accounts and selecting Communications Service for the account sign in. Clear the check boxes for .NET Passport and Exchange Account. Check the Communications Service account and enter your SIP account name. Click on Advanced and select Configure settings. Enter the IP address of your server, or hostname and choose UDP.

7.2. Cisco 79XX phones

Cisco has complete documentation on how to convert a 79XX series phone to use SIP. The basic steps are:

7.2.1. Configure a DHCP service that provides

- IP address
- Subnet mask
- Default gateway
- DNS server addresses
- TFTP server address

7.2.2. On the TFTP server load these files

- OS79XX - Identifies which firmware the phone should load with no extension. Example: P0S3-04-1-00
- P0S3-04-1-00.bin - The firmware image
- SIPDefault.cnf - Site wide configuration options
- SIPmacaddress.cnf - Phone specific settings, including logon name and password. Example: SIP000A8A93D466.cnf
- RINGLIST.DAT, ringer1.pcm, ringer2.pcm - ring tones

Each time the phone is powered on it will tftp download OS79XX and determine if it needs a firmware update. If no update is needed the next step is to download SIPDefault.cnf, SIPmacaddress.cnf, and optionally a dialplan, ringlist and ring tones.

Calls can be placed to other registered SIP clients, or to a PSTN number provided there is PSTN gateway identified in the SER configuration file.

Chapter 8. PSTN connectivity

Passing calls that originate from a SIP client to the PSTN is a simple matter of permitting SER to relay the session to an established PSTN gateway. Calls that start out on the PSTN and need to be directed to a SIP client requires that the PSTN gateway be aware of where to direct the call.

8.1. Cisco Dial-peer

The Cisco gateway needs to have a PSTN interface, such as FXO ports or a VXB-2TE1+ card, and depending on the model of Cisco device an upgraded IOS revision. The dial-peer itself is simple:

dial-peer voice 999 voip

destination-pattern 555999. ** Associate the number range 555-9990 to 9999 with our SIP server

session protocol sipv2 ** Set this dial-peer to use SIP instead of Cisco protocols

session target sip-server ** Send the call to our SIP server. See SIP-UA below

codec g711ulaw ** Set the default codec to 711-Ulaw (common codec between clients)

!

sip-ua

sip-server ipv4:192.168.0.1 ** IP address of our SIP server

8.2. Relaying PSTN in ser.cfg

The following is an extremely simple sample of how to relay a call from a SIP client to the PSTN

```
# attempt handoff to PSTN
if (uri=~"^sip:9[0-9]*@mydomain.com") {  ##  This assumes that the caller is
    log("Forwarding to PSTN\n");       ##  registered in our realm
    t_relay_to( "192.168.0.2", "5060");  ##  Our Cisco router
    break;
};
```

Chapter 9. SIP Status Codes

The following are the SIP status codes as of RFC3261 (http://www.ietf.org/rfc/rfc3261.txt)

Table 9-1. 1XX-2XX Informational

100	Trying
180	Ringing
181	Call Is Being Forwarded
182	Queued
183	Session Progress
200	OK
202	OK

Table 9-2. 3XX Redirection

300	Multiple Choices
301	Moved Permanently
303	See Other
305	Use Proxy
380	Alternative Service

Table 9-3. 4XX Client-Error

400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
413	Request Entity Too Large
414	Request-URI Too Large

415	Unsupported Media Type
420	Bad Extension
480	Temporarily not available
481	Call Leg/Transaction does not exist
482	Loop Detected
483	Too Many Hops
484	Address Incomplete
485	Ambiguous
486	Busy Here
487	Request Terminated
488	Not Acceptable Here
489	Bad Event
491	Request Pending
493	Undecipherable

Table 9-4. 5XX Server-Error

500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Time-out
505	SIP Version not supported
513	Message Too Large
580	Precondition Failure

Table 9-5. 6XX Global-Failure

600	Busy Everywhere
603	Decline
604	Does Note Exist Anywhere
606	Not Acceptable