

*The only thing necessary for the triumph of evil is for good men to do nothing.*

## **RasLenny Project by StrangePolack**

### **Credits**

Many thanks to:

- *RealLenny @ Reddit*  
An anonymous Australian IT Guy, who used his incredible voice skills to create and share Lenny's monologs.
- *Chriss from Crosstalk aka Toao.net @ YT*  
An American IT Guy, who contributed with his willingness of knowledge sharing. Thanks to his lucid tutorial, I was able to start slowly with IPT and carry on.
- *Joly Roger @ YT*  
Another Australian IT Guy, whose contribution in combating phone spam is significant and who also (although indirectly) gave me inspiration to move on.

### **Disclaimer**

The project is in a very early, immature, pre-alpha stage and may contain bugs.

There are huge fields for improvement in security, functionality and ergonomics of this project.

As well as in quality of this tutorial itself.

If you do not appreciate my amateur efforts and expect a professional, fully reliable solution, this is not the project, you're looking for!

So, if you still decide to use it in other than a home/hobbyist way, I did warn you!

Also, I am not advocating or supporting any of the products, programs, technologies or systems that I will be mentioning in this tutorial.

I just need to cite their proprietary names to make some examples.

## **Purpose**

When you are bothered with unwanted calls on your mobile, all you can do is to hang up and add the number to the black list.

While a universal solution for mobile phones will not be available for a long time, still it is possible to use a mobile's functionality to divert the unwanted call, instead of blocking or rejecting it.

And then connect the call to a human imitating machine, so you can later have fun listening spammer's conversation with a robot and publishing the result dialogs.

You might have heard of a phone chat bot starting conversation with "Hello, this is Lenny!".

You may find many Lenny's conversations available in the internet, just by typing the above search phrase.

That is a good way to waste the time of scammers, abusive telemarketers, debt collectors, etc.

This tutorial is to help you out to build a cheap home phone system for such purposes.

You may easily record a fully customizable set of monologs in your own language and with your own voice.

I require you to be familiar with using a PC.

But no advanced IT skills are needed. Just read this tutorial and you will be able to build your own answering machine within one extended weekend.

(RPi is an abbreviation for the Raspberry Pi microcomputer, that will be used from now on, throughout this tutorial, together with RasPbx which means a telephone system built on RPi.).

## **Functionality**

First, let me disappoint you a bit.

There is no ready solution for mobile phones.

There is no way, that you can simply install an app on your favourite mobile and enjoy the show.

So, if this is what you expect you may skip reading.

Still, if you want to learn a bit about the idea of the IP Telephony, I encourage you to read the article.

What you actually can achieve, if you decide to implement this idea, is the ability to forward an answered mobile call to your own telephone robot, based on a cheap RPi and send them to Lenny.

Furthermore, you may even utilize your own set of recordings in your own language and with your own voice.

So, this idea works brilliant for "landline" phone numbers (registered with a SIP provider).

*If you really want to have a solution, that does not require an interaction, a GSM to SIP gateway is the only option.*

*That device is a bit costly and can be used inhouse only - not a portable solution, sorry.*

*If you still think of buying a random/cheap one online - the chance, that you will buy a model, that will not work is very big. So why waste money?*

*Except Yeastar, I am not aware of any other reliable brand producing these unpopular devices, which are very rarely used outside very specific call centres.*

*Plus operating with SMSes is much more challenging.*

*Therefore, this solution is outside the scope of this tutorial, at least with this version...*

Also if you still have an old landline, there are converters that you can use, to change the output to SIP (also outside scope of this tutorial as barely used).

### **What is required**

For a basic functionality you will need to purchase:

- SIP account with an active phone number - the cheapest point (next chapters explain).
- Raspberry Pi 3B (or 3B+)  
with additional equipment (like USB charger, cables and casing)
- Micro SD card with at least 32GB capacity

Having the above items (configured), you will be able to forward unwanted calls to your own chat bot.

Remember: there is no way to turn your mobile into a working prank box!

Also, from that moment on, to shorten writing, I will be using in this tutorial these terms interchangeably:

- RPi as an abbreviation for a Raspberry Pi box
- RasPbx - as an abbreviation for the same Raspberry Pi box  
but with an installed and running FreePbx OS

### **Registering your mobile app with a SIP provider**

If you understand the idea of free internet-calling mobile apps, this topic should be very easy to comprehend. You certainly know apps like: Skype, Viber, WhatsApp, etc.

Some of such apps may be even using SIP protocol, while others use a proprietary one.

So, the possibility of making voice calls, without mobile network or a landline phone should not seem surprising to you, at all.

The purpose of this chapter is to explain, how you can get a phone number with a chosen provider.

If you already use this technology or/and are familiar with this topic, you may skip this chapter.

This is the only chapter you may freely skip if you find it too obvious.

The rest of you, please read and follow the instructions below.

Being familiar with the above topic, you will go just one little step further.

A company offering public telephone numbers with the SIP technology, is called a **SIP provider**.

From now on, in this tutorial, for simplicity it will be simply called "**provider**".

Please do not confuse it with an internet provider.

Let's get started...

In addition to normal mobile calling capability (GSM etc), mobile phones offer apps that allow to use a public "landline" phone number without using a mobile network whatsoever, as long as your phone is connected to the internet.

You just need to register with the provider, make up a username and a password and you can start calling!

When SIP comes into play, you need to install an app on your mobile (or PC).

That will let you use a public, landline phone number from your country and make call (as well as pick up) calls anywhere in the world you are.

Plus, in the very unlikely case, that you are still using a traditional, old phone line at home, you can (and should!) migrate it to a SIP provider. And that will be way cheaper!

(You can even purchase and configure a desktop SIP phone so the person using it so far will not see much visual difference.)

It is just a matter of selecting a proper SIP company, that offers you, what you need.

You are about to activate your new SIP based phone number!

### ***Finding a SIP provider***

Open a search page, type a phrase like "sip number yourcountry" and start searching for your local provider.

I am unable to help you out with your searching, I can only advise on things that you should consider, when comparing different offers.

- SIP accounts with fully functional phone number also for incoming calls.
  - This is the key requirement!
  - You may find SIP offers that allow you only register your existing mobile number within an app, so you can make outbound calls cheaper. This is not the solution you're looking for. So, read the offers carefully.
- Reasonable (maybe equal zero?) monthly fees.
  - E.g. in Poland one can find an offer for 1PLN, in the UK for 3GPB (hm...), in Switzerland for free.
  - Prices for making calls are the other factor - they should be comparable with prices for mobile calls.
- Check out for payment and cancelation options.
  - As payment options, they should be accepting bank transfers or known, secure online paying methods. If they require your credit card number - skip that offer!
  - Same is for cancelation - that should be easily available through their webpage.
- Skip offers that disallow you using the number outside your country.
  - The idea of IP Telephony is a worldwide calling from any locations for a local price.
- Prices for calls within the same network.
  - Prices between customers registered within the same provider should be free or very cheap. If they are not, keep looking for another offer.

Once you find an offer that you like, you must register.

(Some providers may require, that you do not register to their portal from abroad.

In such a case you need to ask someone in your country or use a proxy server for your browser.

It is still fine, as long as they later let you use their service from abroad.

To make explanations easier, I would like you to create a fictional example below. Then you will follow the analogy.

Suppose, we found a webpage of our soon will be provider in Poland. Their address is:

"http://goodsipcompany.local"

After answering a bunch of questions, providing the email, other required data, passing captcha tests, etc finally we make it till the end and have an account on their web portal.

*We created a username: "booboo"*

...and secured the account

*...with a password: "portalpass"*

This not your SIP account, yet!!!

These are just for login to their webpage (portal).

Just like you login to an internet forum, auction service, social account, etc.

This is your panel, where you administer your account and settings.

When logged in, you can e.g.: check calling history, top up your balance, activate/deactivate your phone numbers, etc.

Before you actually activate a new phone number, you may need to be prepared for some more formalities possibly required by the law in your country.

E.g. in Poland it is necessary to provide a proof of identity.

In other countries, they may even ask you for a proof of residence.

While maybe in some other countries, you may activate a phone number totally anonymously.

And of course, they will want you to pay them before creating a phone number.

Even if they let you activate a new phone number for free, it is very unlikely, that it will be fully functional without some balance.

So probably this is the moment, when you will have to top up your account.

Suppose you pass all formalities, top up your personal account and it becomes fully active.

So now you are finally ready to open a SIP account and select a phone number.

### ***Activating a SIP phone number***

This step has similarities with opening an email account.

There is one very important info, that you need to find on your provider's webpage.

And that is the address of their SIP server.

For our fictional provider, it goes like:

*"serversip.goodsipcompany.local"*

Like mentioned before, a SIP company will use one distinct address for a webpage, where a customer can login to their panel and administer the account.

And They will have another address for their SIP server, that will provide the actual telephony services.

Apart from the server info, you may find some other details and settings to follow.

You may even find a detailed instructions step by step, how to configure your favourite mobile app for your new number - please follow them!

Let us say, that we were lucky and from our fictional pool of numbers at our fictional provider, we found available this number:

*"11111111"*

Upon booking that we were obliged to provide a password to secure it, so we took:

*"mynumpass"*

After the successful registration we will be an owner of a SIP account like below:

*username: "11111111"*

*password: "mynumpass"*

*registration domain: "serversip.goodsipcompany.local"*

What is this for? What is this all about? We wanted a number not a username.

Why do we need that email-like looking text strings?

When your mobile app registers to the SIP server, it uses the above form:

*11111111@ serversip.goodsipcompany.local*      (*secured with password : "mynumpass"*)

That is your SIP username. It may be (and usually is but not always) same as your phone number.

(Imagine: all the other SIP users around the world may actually contact the above username.)

This is what SIP understands, phone number is just an additional option, only for us people.

The SIP protocol does not really need a phone number to work but because we wanted,

we were also given a public number:

*"0048 11111111"* (0048 = Prefix for Poland)

You now probably see a huge resemblance to an email address. You are right!

The SIP protocol was designed to mimic as much as possible the well know HTTP.

Including response codes.

(E.g. it also has the 404 error, that you get when you try to browse a non-existent page.)

So, to sum up.

Registering our SIP account was like registering an email.

Please digest that info above as I am about to confuse you a bit more with one more possibility.

It may happen that upon the registration of our new number, the provider will assign (to your chosen phone number) a random username like that:

*user2020coronav@serversip.goodsipcompany.local*



This username does not indicate a phone number. Not a good practice. But it does not matter. After you register, your selected phone number will be also active.

***More on activating a phone number***

So finally, you must end up having these SIP details:

- username
- password
- SIP server address
- phone number (not important for registration process - only for you)

As long as you know the 3 top details (2 top are for your eyes only), you will always register.  
(Sooner or later, even if some settings are non-standard.)

If any non-standard settings are used by the provider of your choice, they will be reflected on their webpage.

### **Choosing and configuring the SIP app on your mobile**

At this moment you should have an idea how you can order a public telephone number at a provider of your choice.

Perhaps you already have your own SIP account number.

So you should be equipped with registration details from the provider.

Or if you do not want to buy any and you are reading this tutorial just for fun, you should more less know how you deal with the above.

Time to finally install a SIP app on your mobile phone to make and pickup phone calls.

For simplicity I assume you will install Zoiper but there are more to choose from:

Bria, Gandstream Wave, Linphone, Sipdroid, MizuDroid and more...

For more options, all you need to do is to search, typing e.g "sip client for mobiles".

Once you install the app, time to register your SIP account.

*You remember what 3 details of your SIP account you need to remember for registration?*

The registration process for all SIP clients follows the same logic but as the apps have different GUIs, I will not go into details. Every app will have it explained on their webpage.

I need to admit, that registering your SIP account on a mobile is the step, where things may not be working "out of the box" at the first time.

Also I am said to say but your initially stable connection or registration may break up "from time to time".

(Using a desktop SIP phone is more stable but it costs money.)

The most common problems are breaking wifi and mistyped registration details. Keep trying.

Consider yourself successful with registration when your SIP app says "registered".

What you will have to fill in case of Zoiper in the "SIP Account Section":

- Account Name (just for your convenience, meaningless for registration)
- Host (that is the SIP server's url)
- Username
- Password

You can also confirm your phone's registration on the provider's portal, usually in the section with your accounts/numbers.

It should be showing details of the registered endpoint like:

IP address and endpoint type. eg "Zoiper"

After a successful registration, test your new phone number, making both outgoing and incoming calls. Pick them up as you need to check the audio as well.

And that finally would be the end of this chapter.

If you find these things simple, that is cool, we will extend the knowledge in the next chapter.

If you got confused with this basic stuff, I am very sorry but there is not much I can do to help you.

And sorry to mention, but if you read it and still cannot comprehend this chapter afterwards, chances are very little, that you will succeed with the rest.

In the next chapter we will build up a cheap home telephone system.

### **Initial setup of your PC and first RPi booting**

I assume you are an MS Windows user.

This is what you need to do.

First, download and install Python 3 on you PC.

Then open the command prompt of your MS Windows (type "cmd").

*From now on I will refer to this activity, simply as: "Open cmd".*

Execute one by one, the following commands:

```
pip install rich
pip install fabric
pip install pyfiglet
pip install pathlib
```

If no error is thrown, you are lucky.

Now, grab and format your micro SD card (eg. using "SD Card Formatter").

Download: <http://download.raspberry-asterisk.org/raspbx-10-10-2020.zip>

Then unpack the zip and install the img file on the micro sd card  
(eg. using "Win32DiskImager").

These 2 steps are widely described online, I will not be explaining them.

Do not try to upgrade your freshly installed RaspPbx like they say you should, right after installing the image. It will fail.

Now, insert the card to your RPi box, plug it to ethernet and power and start it.

## Connecting to the customized CLI of your RasPbx

*As the last sort option, you can always attach HDMI, mouse and keyboard if things go wrong. But that is the least convenient option.*

I have created for you a customized menu (in IT language is it called "CLI"), that you will use from now on to connect to your RasPbx.

Do as follows...

Copy the files of this project to the folder of your choice.

*From now on, I assume it is: "D:\raslenny\".*

And I will be referring to this path all the time.

So the folder should now contain:

- many ".py" files
- text files named: "init1.txt" and "init2.txt"
- a folder named: "audios", "recordings" and "tshoot"

Open cmd.

Navigate to the "Lenny" folder by executing:

*"cd D:\raslenny" and then "D:"*

Issue the following commands:

*"mkdir D:\raslenny\recordings" and then "mkdir D:\raslenny\tshoot"*

First way is to connect manually. (Do not try this one yet, please just keep reding.)

If you want to login manually, execute:

*python conask.py*

Then you will have to provide:

- IP or hostname of your RPi:  
(most likely the initial "raspbx.local" or the new one: "raslenny.local")
- username (the default is "root")
- password (the default is: "raspberry")

It may take ca 15s to get you logged in.

If the logon is successful, you will be greeted with a prompt that looks similar to cmd.

this is how the login process should look like.

[illegible]

The other way to connect (good for lazy people like me) is to use the script for an automatic connection.

First make sure you have only this one RPi connected to your home network.

Let us also assume you have not changed your logon credentials, yet.

In such a case you can always login by typing the command:

*python condisco.py*

If it does not work (eg, you changed credentials or you are not using MS Win), try the manual connection, as described several lines above.

In either case you should be greeted with a prompt:

```
type "?" for help, "exit" to disconnect, "shutdown" to shutdown or enter a regular Linux command.  
command:>
```

You may type "?" to get help. The options are to configure your FreePbx.

If you want to exit, type:

*exit*

You may also (to a limited extend) use genuine Linux commands.

If you want to shutdown, type:

*shutdown*

From now on, whenever I want you to open the customized CLI - I will just shortly write:

*"Open cCLI"*

If in any of these steps you run into problems, you may use a program like "MobaxTerm" to connect manually, transfer the files into the RPi and locally run the initial scripts described below.

But these require some knowledge. If you do not know what to do - sorry, you are on your own.

### **Initial setup of your RaspPbx**

Connect in the way described previously.

(For help, press "?".)

Now you will initialize the system.

Type "*init1*" to initialize the config. Wait for a long time (>20 mins).

After the system is back, type "*init2*" to apply further config. Wait (>5 mins).

Type "*init3*" to apply the last portion of the config. Wait (not too long).

When all the above inits are done without errors, you need to assign static IP settings for your RPi.

Type "*ipstatic*" to apply static IP setting. This step is very important!

Just follow the prompts and keep providing the info.

After you successfully provide all needed pieces, the system will go to reboot and start with the new static IP settings.

Please write down the statically configured IP settings of your RaspPbx, as you will be using it!

To check it when you are logged in, use the genuine Linux command "ip address"

Note:

You can always revert to automatic IP settings, by typing "ipauto".

You may like to do it, if your plan to move your RPi to a different location.

### **Using the GUI of your RasPbx device**

Open your favourite browser and navigate to:

http://(staticly configured IP of your RPi)

(Be careful! The browser may know better, what you want and try to redirect you to "https:")

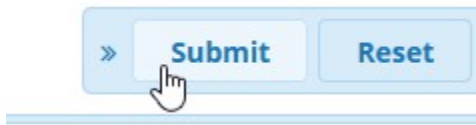
Username: " webadmin"

Password: " webpass"

### ***Applying changes***

Whenever you see the phrase "**APPLY!**" in this tutorial, do as follows.

Navigate to the bottom right corner, find a menu element with buttons "Submit" and "Reset" and click:



Then navigate to the top right corner and click:



Give it a moment to complete... Done!

### ***Hiding and unhiding the "Submit" button***

The mentioned element with buttons "Submit" and "Reset" may sometimes obfuscate other important GUI parts. In such a case just click:



And the other way around - in case the buttons seems to be missing/hidden, click:



I assume now you will always know how to hide/unhide this element, so even if this step is needed in the process, it will not be described again.

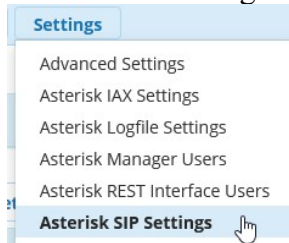
Also from now on all the steps described in this chapter to open the GUI will be referred as:

*"Open gui"*

## **Setting up codecs**

Open gui.

In the main menu go to:

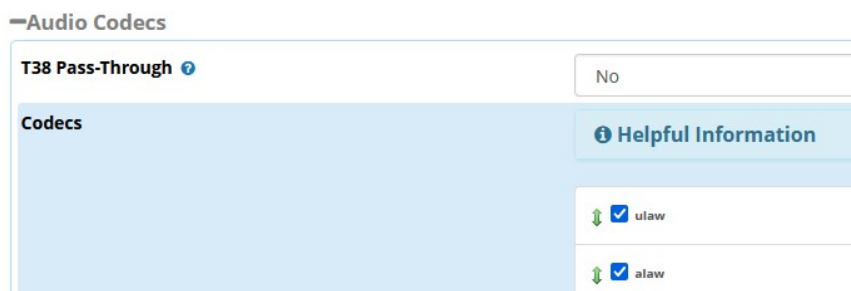


Stay in the section "General SIP settings:

### **SIP Settings**



Scroll down to "Audio Codecs"



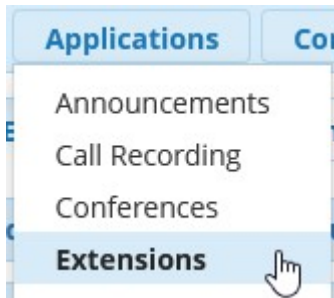
Now select all the codecs from the top to the bottom.

***APPLY!***



## **Registering a local extension**

Go to:



and select:

**+ Add New SIP [chan\_pjsip] Extension**

Fill this section like below:

Add PJSIP Extension **33**

General Voicemail Advanced Pin Sets

— Add Extension

This device uses PJSIP technology listening on Port 5060 (UDP)

User Extension ?	33
Display Name ?	33
Outbound CID ?	
Emergency CID ?	
Secret ?	pass33

Really Weak

Yeah, I know... This is just an example, right?

**APPLY!**

Congratulations!

You have created your first local extension!

Let's try it.

### **Registering your mobile app with the RasPbx**

If now you have any other accounts on your mobile SIP app (eg. the ones of your newly created remote SIP server), please delete everything.

Again, we assume Zoiper as your SIP app.

In the "SIP Account Section" fill in:

- Account Name (eg: "33")
- Host (the static IP of your RPi, that you have previously configured)
- Username: "33"
- Password: "pass33"

You should be now registered with your mobile app into your home RasPbx.

Now, dial:

\*60

If you can hear the speaking clock, well done!

### **Registering your RasPbx device with the SIP provider**

Make sure your mobile app is not configured with the provider anymore.

You may also like to re-read the chapter:

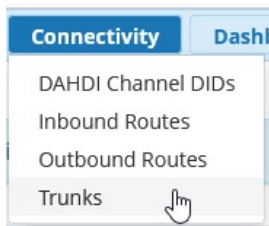
"Registering your mobile app with a provider"

Similar terminology will be used.

To recapitulate the meaning of terms from the previous examples:

11111111	fictional username of your SIP number account (do not confuse it with the login to their webpage!!)
mynumpass	fictional password protecting the above account
serversip.goodsipcompany.local	fictional url of a fictional SIP provider

So let's get to work. Open the gui and go to:



Next click:



Then fill in the General section.

As the "Trunk Name" we assume to choose "my\_chansip\_trunk".

The "Outbound CallerID" will be here the fictional account name: "111111111".

A screenshot of the 'General' tab in a configuration interface. It shows three fields: 'Trunk Name' with the value 'my\_chansip\_trunk', 'Hide CallerID' with a 'Yes' button selected and a 'No' button, and 'Outbound CallerID' with the value '111111111'.

It has absolutely no significance, it is just so the system does not nag you with a popup like below:

It is highly recommended that you define an Outbound CallerID on all trunks, undefined behavior can result when nothing is specified. The CID Options can control when this CID is used. Do you still want to continue?



But even if it happens, that you see such a popup, just click "OK".

After filling the "General" tab, click on "sip Settings" and stay with the "Outgoing" section.

The initial setting should look like below:

A screenshot of the 'sip Settings' tab, 'Outgoing' section. It shows a 'Trunk Name' field with the value 'my\_chansip\_trunk' and a 'PEER Details' section with the following text: 'host=\*\*\*provider ip address\*\*\*', 'username=\*\*\*userid\*\*\*', 'secret=\*\*\*password\*\*\*', and 'type=peer'.

You will need to fill in this section with the values, given by your SIP provider.

Look at the fictional settings below:

A screenshot of the 'sip Settings' tab, 'Outgoing' section. It shows a 'Trunk Name' field with the value 'my\_chansip\_trunk' and a 'PEER Details' section with the following text: 'host=serversip.goodsipcompany.local', 'username=11111111111', 'secret=mynumpass', and 'type=peer'.

The picture above shows fictional values!

Then navigate to "Incoming":

#### Add Trunk

General	Dialed Number Manipulation Rules	sip Settings
Outgoing	Incoming	
USER Context ?		
USER Details ?		secret=***password*** type=user context=from-trunk

Again, use the settings given by your SIP provider.  
Look at the fictional settings below:

General	Dialed Number Manipulation Rules	sip Settings
Outgoing	Incoming	
USER Context ?		111111111111
USER Details ?		host=serversip.goodsipcompany.local type=peer
Register String ?		111111111111:mynumpass@serversip.goodsipcompany.local

#### **APPLY!**

Just as you did when registering your mobile app, check the account on your provider's webpage. In a matter of a minute, you should see an "FPBX" device registered. Of course just the successful registration confirmed is not in any use yet. Nobody can call you and you cannot call the outside world yet, neither. Let's change it.

## Setting Inbound Routes for testing

Go to:



Click:



Configure it in the following way:

### Inbound Routes

Route: all\_in

General	Advanced	Privacy	Other
Description ?	all_in		
DID Number ?	ANY		
CallerID Number ?	-		
CID Priority Route ?	<input checked="" type="radio"/> Yes <input type="radio"/> No		
Alert Info ?	None		
Ringer Volume Override ?	None		
CID name prefix ?			
Music On Hold ?	Default		
Set Destination ?	Feature Code Admin Speaking Clock <*60>		

**APPLY!**

You have now set up a test inbound route for none-restricted numbers.

(If you want to accept also restricted calls, just leave the "CallerID Number" field empty)

Now, call your new SIP number from an external phone number.

If you are getting connected to a speaking clock, that is great.

One of the most challenging steps is behind us.

In case something goes wrong here.

Sorry, not much I can advise except restarting the RPi.

You may have a SIP vs NAT problem with a "smart" router.

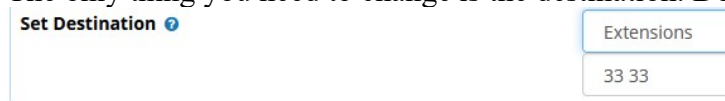
### **Setting real Inbound Routes**

I assume this moment you are able to call the speaking clock from your extension "33" and from an external number. Now it is time to connect both ends.

Go again to the "Inbound Routes" and edit the "all\_in" that you have created for tests by clicking this button:



The only thing you need to change is the destination. Do it as follows:

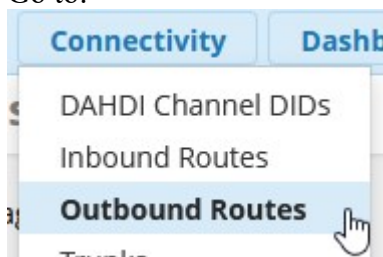
A screenshot of a web form titled 'Set Destination'. It has a label 'Set Destination' with a help icon. To the right, there is a dropdown menu labeled 'Extensions' with the value '33 33' selected.

**APPLY!**

Now if call your SIP number from the outside word, you should be receiving a call on your mobile sip app. Try it.

### **Setting Outbound Routes**

Go to:



Click:



Fill in the following settings:

Route Settings | Dial Patterns | Import/Export Patterns | Notifications | Additional Settings

Route Name [?](#) out\_route

Route CID [?](#)

Override Extension [?](#) Yes No

Route Password [?](#)

Route Type [?](#) Emergency Intra-Co

Music On Hold? [?](#) default

Trunk Sequence for Matched Routes [?](#) + my\_chansip\_trunk

Then navigate to "Dial Patterns".

Route Settings | Dial Patterns | Import/Export Patterns | Notifications | Additional Settings

Dial Patterns that will use this Route

Pattern Help

prepend prefix D

All you need to do is put a dot (".") in the 3rd column (right lower corner of this picture).

***APPLY!***

Now you should be able to make and answer phone calls on you SIP app, whenever your mobile is connected to the same local network as the RasPbx.

## **Testing the original "This is Lenny!" bot**

Navigate to the folder:

D:\raslenny\audios\

*You can see there is already present a sub-folder named "clap".*

*Do not worry about it now, we will discuss it in the next chapter.*

Create a sub-folder and name it "Lenny".

Download the audio files for the Lenny bot.

The following files should be copied to the newly created "Lenny" folder:

- files from "Lenny1.ulaw" to "Lenny16.ulaw"
- file named "backgroundnoise.ulaw"

This is all you need! Do not copy any other files.

Eg. in the package from some sources, you may find a file named "Lenny16-rickroll.ulaw".

No! Do not copy it to your new "Lenny" folder!

Open the cCLI.

Now, to transfer your local dialog files to the RasPbx, you must issue:

**sendaudios**

Remember! If you already have any other custom monologs previously put on your RasPbx, they will be lost!

*Also, note that the files from all other subfolders (except Lenny) will be transferred to the RPi.*

*This will of course result in creating more custom extensions. We will discuss it later.*

So now, issue:

**makeulaws**

Then, issue:

**addcustoms**

Your RasPbx will reboot afterwards.

when it comes up, re-open its gui and navigate to "Inbound Routes".

This time in the "Set Destination" fields, put this:



Set Destination ?
Custom Destinations
Lenny

**APPLY!**

From that moment on, all incoming calls from non-restricted numbers will go to Lenny.

Later, I will show you how to forward calls selectively.

Now, call your SIP landline number from an external phone and talk to Lenny.



The bot works in a way, that when you speak, it listens.

But after 1.5s of silence, it plays its next monolog.

After playing the 16th dialog it should return to the 3rd one.

Sadly, I was unable to replicate the functionality, that runs another monolog when the silence lasts for too long.

In multiple Lenny's recordings, you may find a special monolog being played when the silence is hanging for too long. Like "Hello, are you there?" etc.

So, what is the function of the "background" file?

At this point it is just a dummy, useless audio file containing silence.

For now, it just plays silence in both cases: when the caller speaks and also when they listen.

But it still must be there to maintain compatibility.

It should not be like that, I agree. Let us see what the future will bring.

### **Testing a sample custom extension**

Open you favourite file explorer and navigate to the folder:

`D:\raslenny\audios\clap\`

This folder contains some simple, example audio files to let you test another custom extension. By learning how this one works, you will understand better how to create your own. Let us discuss all the present files in there:

#### **- Several mp3 audio files**

These are the main monologs. Obviously, you will need them for your robot to talk to spammers. They will be played in order.

#### **- File named "background.mp3"**

The same useless, silence audio file as explained in the section dedicated to "Lenny".

#### **- File named "setings.txt"**

Open it and you will see just one line:

`jump=4`

That means that after playing all monologs, the 4th one will be re-played again.

If you forget to have the "settings.txt" file after you create your own extension sub-folder, the system will assume the default value of jump=3.

That is why, this file is not needed in case of Lenny.

Again, connect to the cCLI, issue one by one, the same commands as you did with Lenny:

***sendaudios, makeulaws, addcustoms***

If you are confused, go back one section above.

After the system reboots, open the gui and go to "Inbound Routes".

This time select the below:



The screenshot shows a 'Set Destination' dialog box. On the left, there is a label 'Set Destination' with a blue circular icon. On the right, there is a list box titled 'Custom Destinations' containing the text 'clap'.

### ***APPLY!***

And you are ready to play. The bot should resume from the 4th dialog after finishing the last one. After testing this combination, navigate to the "clap" folder on your PC and do some alterations. Eg. you can open the settings.txt file set jump value=2. Plus you can delete the last dialog. Then again, issue the 3 command sentence to send the files and reconfigure the RasPbx.

### **Creating your own custom extension (bot)**

Suppose you want to create a new custom extension, name it "Lenka", give it 5 monologs and you want the 2nd monolog to be re-played after the last one.

Create a new folder, where you will put your custom monolog audio files:

[D:\raslenny\audios\Lenka\](#)

Obviously, you need to record your own audio sentences.

The common accepted audio formats are: "m4a", "mp3" and "wav".

So, the built in Windows sound recorder will make it.

After creation of your own monologs, you do not need to rename them one by, like you can see they are in the "clap" folder. The scripts will convert the filenames for you.

Look at the table below.

Recording.m4a	Recording (1).m4a	batman.wav
Recording (2).m4a	Recording (2).m4a	Lenka01.mp3
Recording (3).m4a	Recording (4).m4a	Lenka4.mp3
Recording (4).m4a	Recording (6).m4a	superman.m4a
Recording (5).m4a	Recording (9).m4a	zorro.wav

Any of these will work just fine.

Afterwards, they will be all converted to files named from: "Lenka1.ulaw" to "Lenka5.ulaw"

So, your input audio files may have wrong filenames as long as they all have proper content.

It may happen if in the middle of your work you must delete or replace an audio file.

But do not overcomplicate the names! And remember: the limit is: 99.

Now add the "settings.txt" file with a line: "jump=2".

If you don't, jump=3 will be set up.

When all needed files (that means main audios, a background file and the settings.txt file) are in the "Lenka" folder, open cCLI, run the commands to send them, convert them and make your extension active (as described in the "Lenny" section).

Then open the GUI and setup the Inbound Routes pointing to "Lenka".

No pictures needed as you know how to do it, right?

## Selective re-routing of incoming calls to a bot

As of now you know how to divert all calls to your favourite robot.

That is not particularly useful. You still would like to pickup most of the calls on your mobile SIP app and only some of them routed to robot.

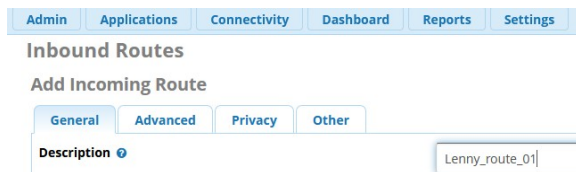
Let us say, you want for some reason every call coming from the capital of Poland, to go to Lenny. In that case the number will start from:

004822

Again, (just like you did in the section "Setting Inbound Routes for Testing") go to:

Connectivity>Inbound Routes>Add Inbound Route

In the "Description" field put eg.:



The screenshot shows the 'Add Incoming Route' form. At the top, there are tabs: Admin, Applications, Connectivity, Dashboard, Reports, and Settings. Below these is the 'Inbound Routes' section. Under 'Add Incoming Route', there are sub-tabs: General, Advanced, Privacy, and Other. The 'General' tab is selected. In the 'Description' field, the text 'Lenny\_route\_01' is entered.

You also need to fill in is this field:



The screenshot shows the 'CallerID Number' field. The field is empty, and there is a small question mark icon to its right. Below the field is a button labeled 'ANY'.

So what should be there? It depends...

- It may be: "\_004822." which would be the ideal option.  
That means your SIP provider sends the full caller's number to you, including the international prefix and does not strip leading zeros.
- It may be "\_4822." which means your SIP provider also sends the full caller's number to you, including the international prefix but strips leading zeros.
- It may be "\_0022." which means most likely you have a Polish SIP provider, you are receiving a domestic call and your SIP provider also sends the full caller's number to you.
- It may be "\_22." which means most likely you have a Polish SIP provider, you are receiving a domestic call and your SIP strips leading zeros.

Are you confused?

When you pickup up calls on your mobiles how can you easily recognize domestic callers from the abroad ones?

Calls from abroad will always include the prefix of "+xx" or "00xx"

Similar is with SIP.

So what should you put in: "CallerID Number" afterwards?

The functionality of your SIP account should include the call registry.

This is how you figure it out.

Also remember: the more specific (or longer you can say) the info you providing is, the more priority it has.

Therefore the below will work as you expect:

DID	CID	Description	Destination
Any	-	all_in	Extensions: 33 33
Any	_4822.	Lenny_route_01	Custom Destinations: Lenny

Only the very specific calls will go to Lenny.  
The rest of your callers will be transferred to your mobile SIP app.

### **Download the recordings**

After your bot receives some calls, it is time to download the result recordings.  
Login to the customized CLI and issue:

*download*

The result recordings will be downloaded to the folder: "D:\raslenny\recordings"  
The name of each file consists of: timestamp, custom extension name and calling number.

### **Maintenance and clean-up**

Once you are sure, you have downloaded all result recordings, issue:

*delrecordings*

This will release your RPi's memory card from storing too many audio files.

To delete all custom extensions, including audio files, you must issue:

*delcustoms*

Once per month execute:

*init1*

Whenever you change anything in the scripts, execute:

*init3*