# *XMPP clients - usage and mitigation*

# Introduction

UPDATE October 2022: I just thought I might turn this into a general XMPP guide. So let's go.

XMPP is a communication protocol, which means you can use it as a replacement for Discord, Facebook, etc. for talking to people. But why do so? What are the advantages?

- You do not have to give personal data (such as a phone number or real name) to shady tech companies in order to use it.
- You can use it on every device and platform, e.g a smartphone or a computer with Linux, Windows, macOS, Haiku...
- It is just as easy (or even easier) to use as the usual communication "services" (as I'll show you in the next section).
- You can choose from the many clients available, instead of being chained to the

officially supported one.

- You can host your own server, if you know how to - this means you have full control over its functioning, data storage, etc. If you don't want to, pick one out of the hundreds of available servers run by volunteers, instead of - again - being chained to the "official" one.
- The aforementioned decentralization (ability to choose client and server) means you are not dependent on a single entity to decide whether you're allowed to stay in or get kicked out. You can easily use multiple accounts, whereas e.g Facebook would discourage it. **XMPP will never die**, as long as there is at least one person hosting a server and some client still exists on the internet, even old versions (this is pretty much a certainty). FB, Discord, etc. will die the moment the companies running them go down.
- All the features you'd expect are available; adding friends, group chats, file uploading, voice / video calls, etc.
- You can easily encrypt your messages with the OMEMO plugin, which means a malicious middleman cannot spy on them.

Okay, so I hope I have convinced you to switch (or at least use it in addition to the violating services). Let us check out just how easy it is (I will use the Psi+ client for this tutorial; you can install it on any operating system - Windows, Linux, macOS or Haiku). However, the same procedure with other clients (Dino, Gajim, Conversations...) is possible - though I will not cover it.

# Using XMPP - Psi+

# Installing Psi+

Here I will show the method I have used for installing Psi+ on Slackware-based distros. For other ones, you can use the AppImage; and for Windows, the installers. These probably work, but are untested by me (**Edit**: the linked AppImage works, and has all the icons and plugins installed by default, but ignores qt5ct). Also, the AppImage has a **broken version of Client Switcher**; you could install an older AppImage, perhaps. But I will teach you a method in which everything works properly. **You need Slackware 15 for this to work**, as Psi+ versions 1.5+ require openssl 1.1.x - installing which would break older Slackware versions; they might also need newer binutils and / or glibc. **You need to compile this with cmake 3.21.4**; newer versions (eg. 4+) result in a buggy OMEMO which you have to turn off for every interlocutor's message if it doesn't work, since it will automatically turn itself back on. And you need to recompile plugins with the same cmake if they were done earlier with any other! Maybe some other combinations are viable but this works for sure. Anyway let's begin:

First, download the Psi+ source code from here by clicking on the *"tar.gz"* link. 1.5.1642 is the version I've used for the installation, though there are newer ones available now. You can also do it by typing *"wget https://github.com/psi-plus/psi-plus-snapshots/archive/1.5.1642/psi-plus-snapshots-1.5.1642.tar.gz"* into the terminal, where *"1.5.1642"* is the version. If using the wget command, you **have to replace both instances of the version with the one you want to install**, e.g *"https://github.com/psi-plus/psi-plus-snapshots/archive/1.5.9999/psi-plus-snapshots-1.5.9999.tar.gz"* if the version you want to install is 9999. Note the folder that you have downloaded the source to; we will assume it is */home/username/Downloads* for the purposes of this guide.

Now edit (sudo needed) the */etc/slapt-get/slapt-srcrc* file and ensure that the line *"SOURCE=https://www.slackbuilds.org/slackbuilds/15.0/"* is in there; if it's not - add it. Run the terminal command *"sudo slapt-src --update"* to recognize the newly added repository. Now download the Psi+ SlackBuild with *"sudo slapt-src --fetch psi-plus:1.5.1600"*. We are specifying the version because otherwise, an old script from different repos might be pulled, which is incompatible with new versions of Psi+. The SlackBuild will be downloaded to */usr/src/slapt-src/network/psi-plus*. Move your Psi+ source code there with the command *"sudo mv /home/username/Downloads/psi-plus-snapshots-1.5.1642.tar.gz /usr/src/slapt-src/network/psi-plus"* (replace the version if you downloaded a newer one earlier, e.g 1644 instead of 1642).

Now we will move to the slapt-src folder to perform operations there. So type *"cd /usr/src/slapt-src/network/psi-plus"* in the terminal. Edit the SlackBuild script to change the version that will be installed, since originally, it tries to install 1.5.1600 - and we're using a newer one. Type *"sudo leafpad psi-plus.SlackBuild"* (leafpad can be replaced with whatever text editor you are familiar with). Replace the line *"VERSION=${VERSION:-1.5.1600}"* with *"VERSION=${VERSION:-1.5.1642}"* (or 1644 or whichever other version of the source you have downloaded in the previous step). Type *"sudo sh psi-plus.SlackBuild"* to begin the compilation; it might take hours. After it's finished, the terminal will spit out something like *"Slackware package /usr/src/slapt-src/network/psi-plus/psi-plus-1.5.1642-x86_64-1_SBo.tgz created."*. Type *"sudo installpkg /usr/src/slapt-src/network/psi-plus/psi-plus-1.5.1642-x86_64-1_SBo.tgz"* to install it. Oh, and remember to back it up so that you don't have to repeat the process later. E.g *"sudo cp /usr/src/slapt-src/network/psi-plus/psi-plus-1.5.1642-x86_64-1_SBo.tgz /home/username/Packages"* (directory needs to exist).

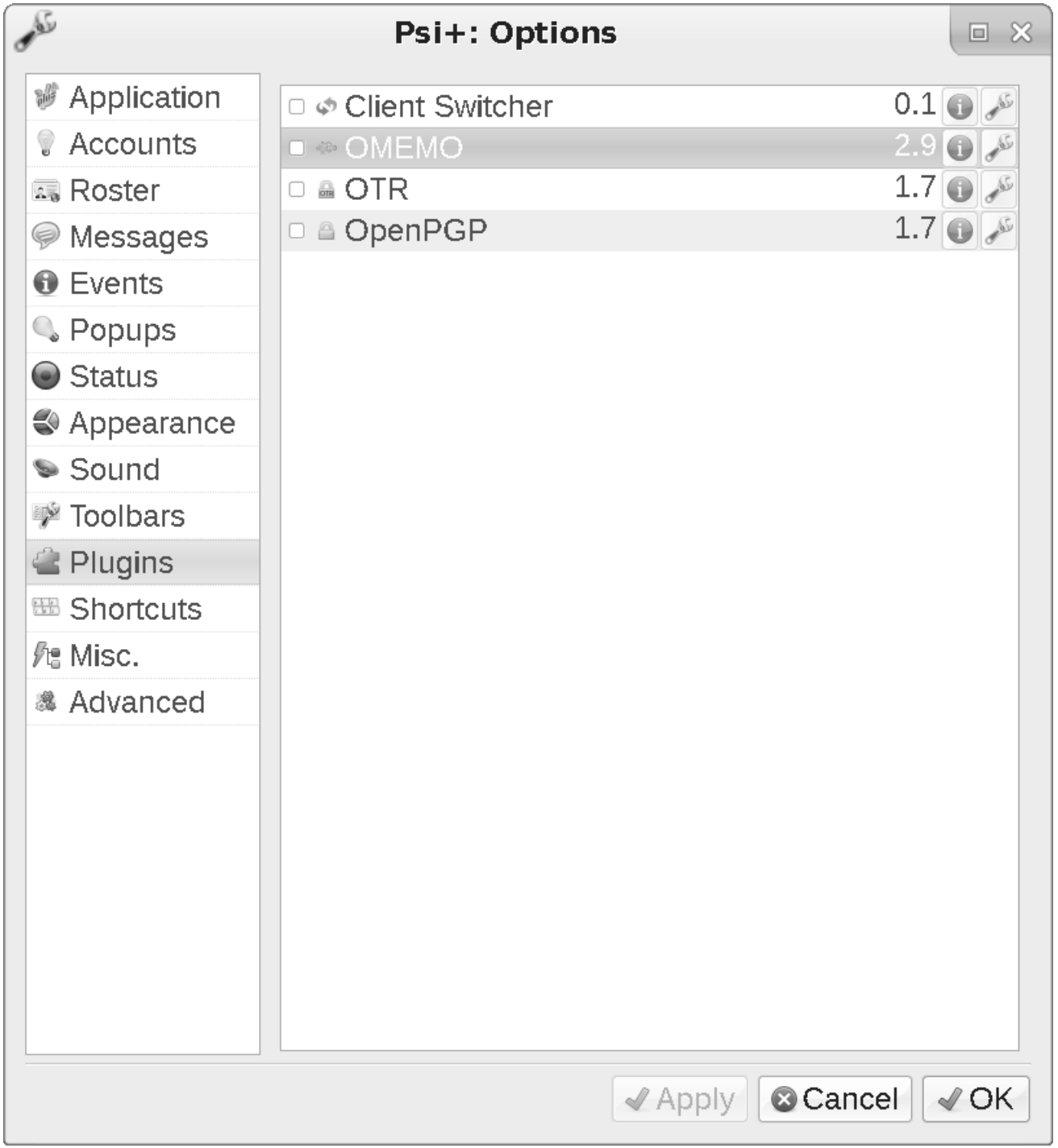
And there we go, Psi+ has been installed. You can move on to the relevant section now to learn

how to use it. You might be wondering, why not just use the command *"sudo slapt-src --install psi-plus"* and skip all the fluff? Because when I tried it, it didn't work at all - the compilation broke in the middle for reasons I couldn't debug. This method works perfectly, and you can install whichever version you want with it. And it's better to know how to do stuff more manually, anyway. **Plugins are not installed this way** - we have to do that separately which I will show how to right now:

## Installing plugins

Because the SlackBuild does not do that by default, we have to do it manually again. Slapt-src extracts all compiled sources to */tmp/_SBo/*, but since Psi+ itself was already compiled, we're only interested in the sources for plugins (which are also contained in the Psi+ source snapshots). Type the command *"cd /tmp/_SBo/psi-plus-snapshots-1.5.1642/plugins/"*, which is where they are stored. Make a new directory where the actual plugins will be created (type *"sudo mkdir build"* for example). Now type *"cd build"* and then *"sudo cmake ../ -DBUILD_PLUGINS= "omemoplugin;openpgpplugin;otrplugin;clientswitcherplugin;conferenceloggerplugin" -DPLUGINS_PATH= "" -DCMAKE_INSTALL_PREFIX= """. "-DBUILD_PLUGINS"* decides which plugins will be installed; we're choosing the three encryption plugins and the one that allows you to disable data leaks, as well as the conference logger that is actually needed to store MUC logs (yes, it's not a default) - but you can add any other that exist in */tmp/_SBo/psi-plus-snapshots-1.5.1642/plugins/generic* (or just delete the whole *"-DBUILD_PLUGINS"* part, which will install them all). You can add any other plugins later though; just repeat the process with other names in the *"-DBUILD_PLUGINS"* section (e.g *skinsplugin* if you want to install skins); I am not familiar with all the plugins so you will have to check what they do for

yourself. The paths are defined because for some reason, cmake randomly adds stuff in there, and you want the plugins installed right into the directory which Psi+ reads from, without useless additions (remove the path definitions and you will see what I mean). Now type *"sudo make"* then when that finishes, *"sudo make install DESTDIR= "/usr/share/psi-plus/plugins""* (if you hadn't defined the paths earlier, the plugins would be stuck in a folder Psi+ doesn't check, and would be unavailable for use). That's it! You should be able to see them in the Psi+ *Options* menu like this:

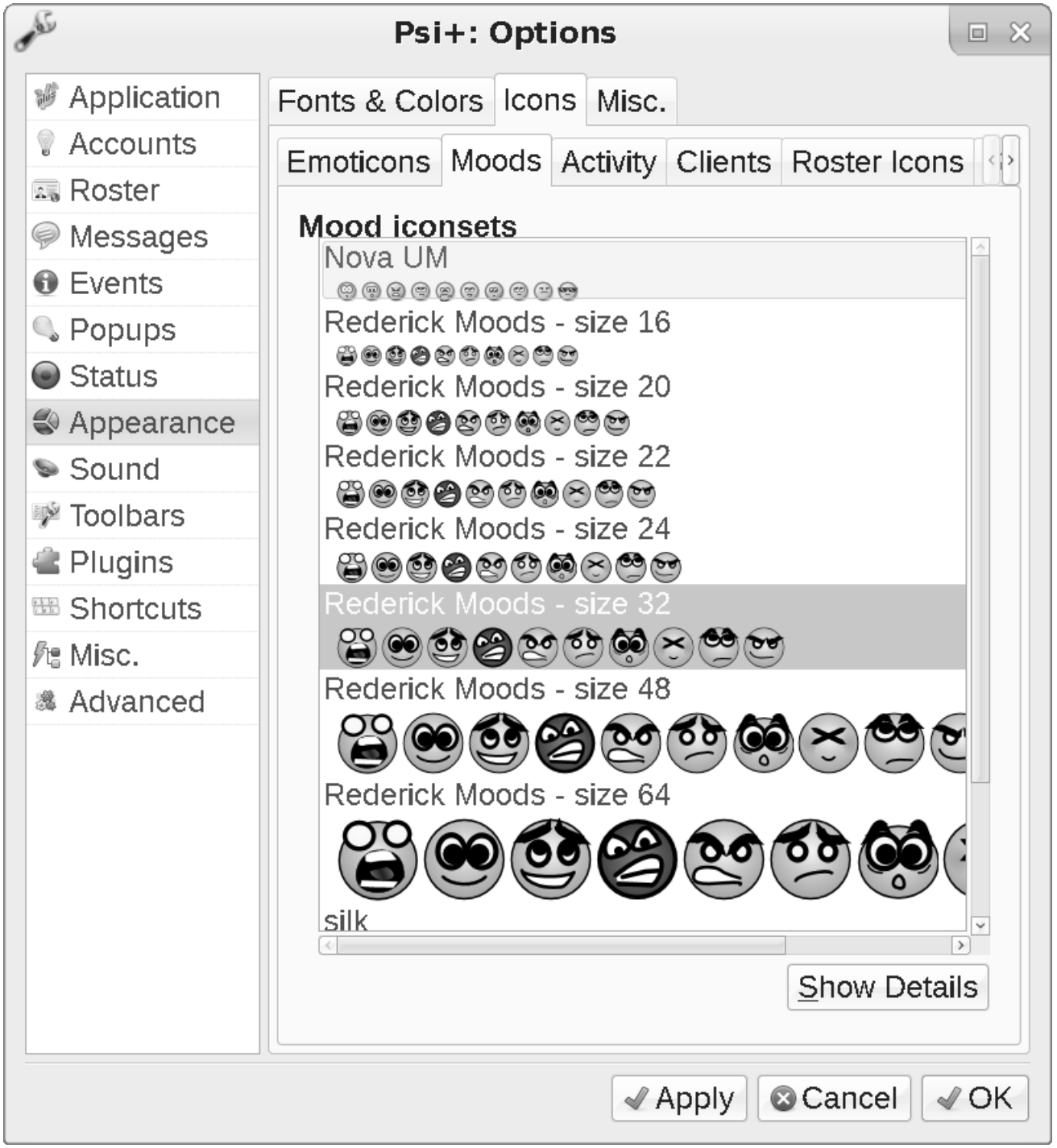



**WARNING**: okay fuck, it seems that the new (0.1) version of *Client Switcher* lacks important

functionality. You will have to compile one from an older Psi+ source code (1.4.1460 works) or download the binary I've compiled (shove that into */usr/share/psi-plus/plugins*). Why do devs have to destroy things that work fine? And after 7 years of absence, as well...No matter, 0.018 works perfectly.

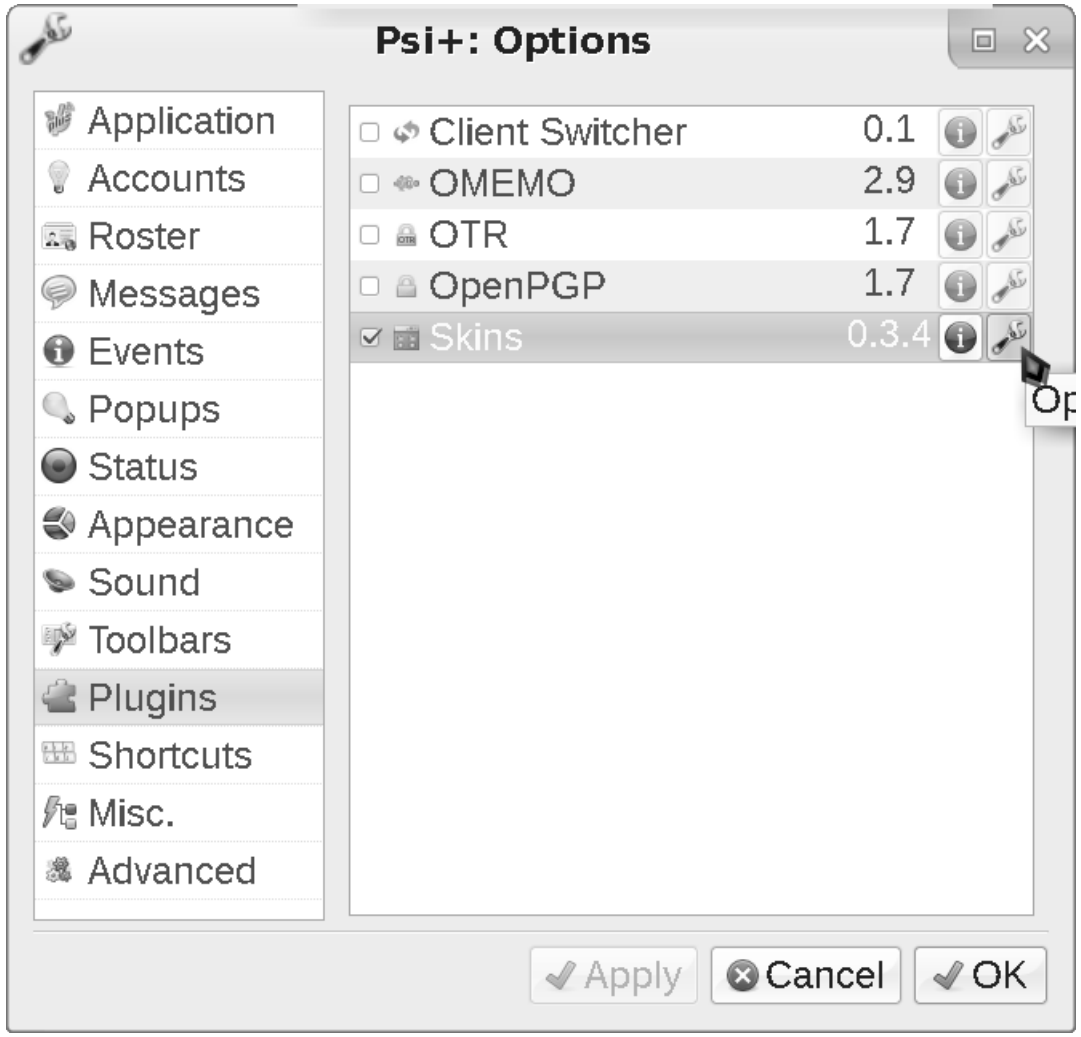
## Installing icons

The SlackBuild doesn't do that by default, either. Go to the build directory (*"cd /tmp/_SBo/psi-plus-snapshots-1.5.1642"*, replacing the version if necessary). Then copy the *iconsets* directory into where Psi reads its icons from (*"sudo cp -R iconsets /usr/share/psi-plus/"*). If you did it right, you will be able to see them in the options screen:

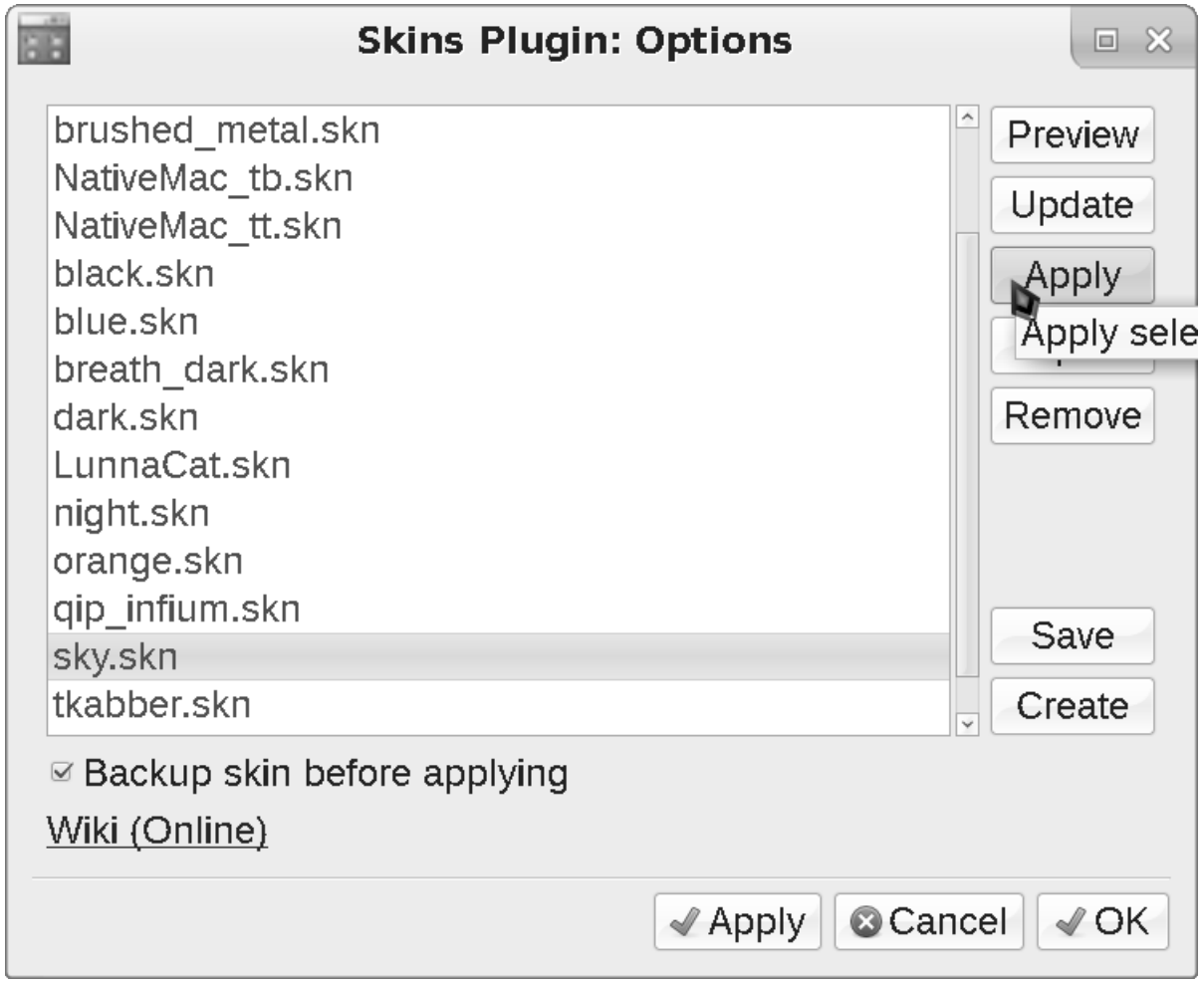

Psi+: Options
Application
Accounts
Roster
Messages
Events
Popups
Status
Appearance
Sound
Toolbars
Plugins
Shortcuts
Misc.
Advanced
Fonts & Colors
Icons
Misc.
Emoticons
Moods
Activity
Clients
Roster Icons
Mood iconsets
Nova UM
Rederick Moods - size 16
Rederick Moods - size 20
Rederick Moods - size 22
Rederick Moods - size 24
Rederick Moods - size 32
Rederick Moods - size 48
Rederick Moods - size 64
silk
Show Details
Apply
Cancel
OK

# Installing skins

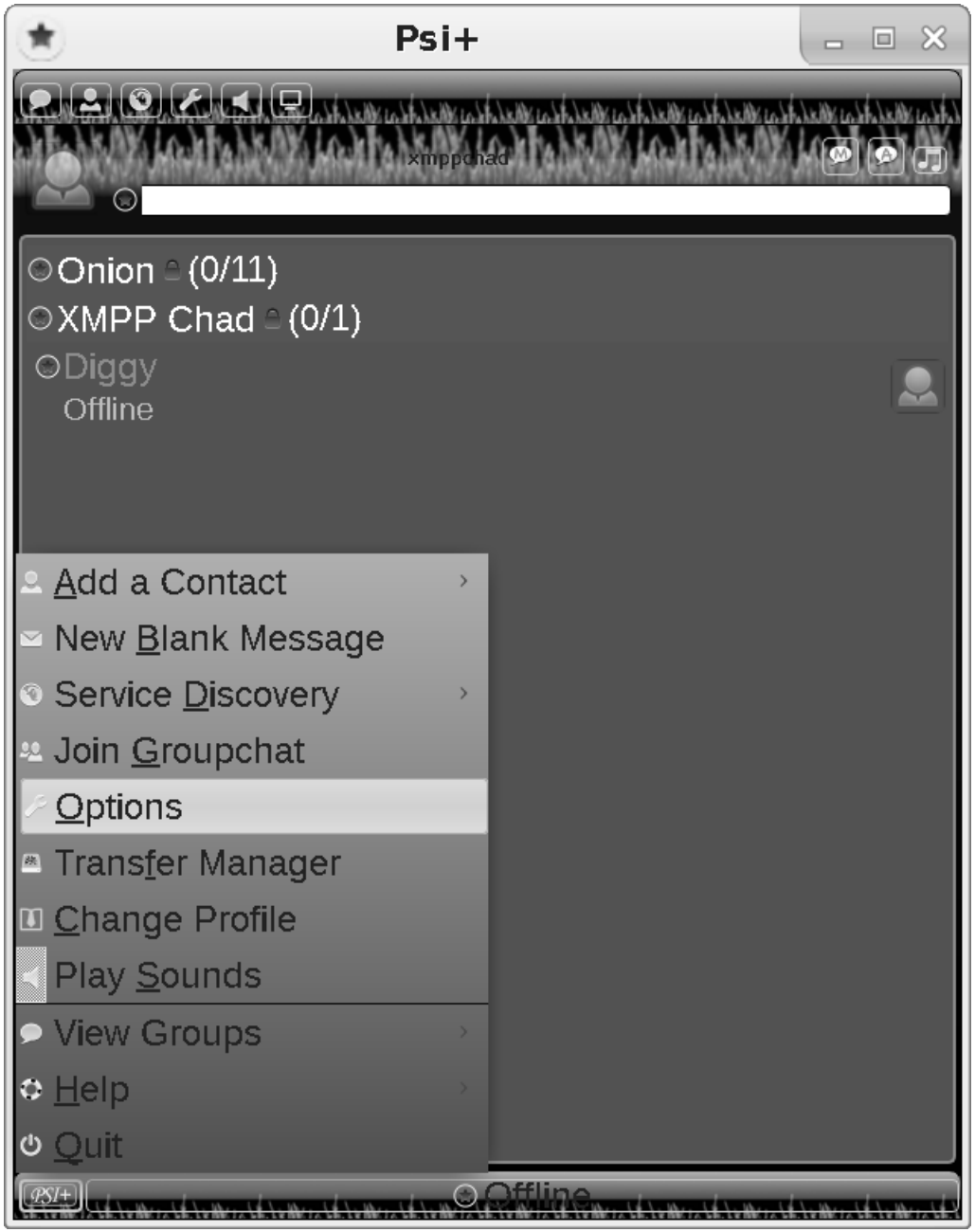
You need to have installed the skins plugin in the Plugins section for this to work. If you have done so, type *"cp -R /tmp/_SBo/psi-plus-snapshots-1.5.1642/skins /usr/share/psi-plus"*. This is the same exact process as with icons - just copying the skins from the Psi+ source folder to the one Psi+ actually reads them from. Enable the *Skins* plugin in the plugin window, and enter its settings like this:



Pick a skin and apply it (ensure the *"Backup skin before applying"* option is enabled, which will allow you to restore your previous configuration later):




This is the Night skin (some can change more, like fonts etc. it seems):



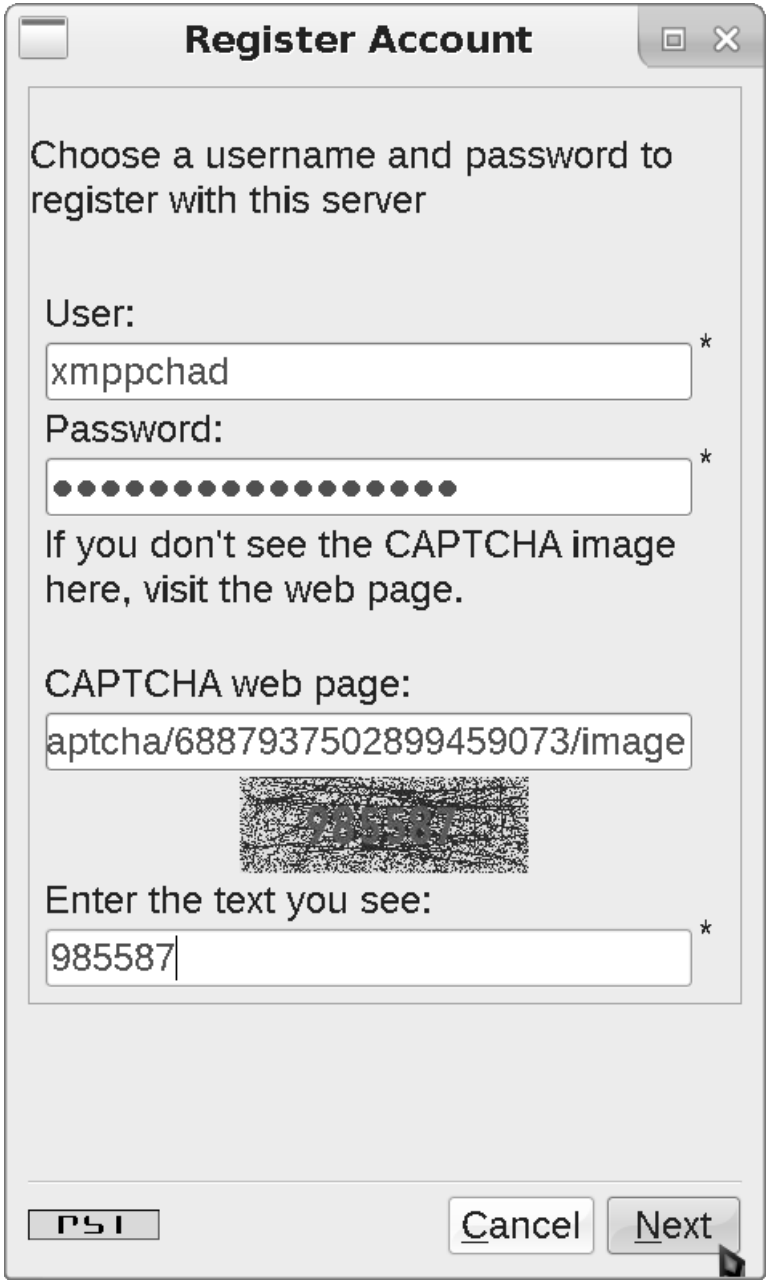

Anyway, if you're done installing plugins, icons, and skins, you can burn the build folder (*"sudo rm -R /tmp/_SBo/psi-plus-snapshots-1.5.1642"*) to not leave litter around.

# Adding an account

When you turn on Psi+, this screen will appear:


To use XMPP, you need to register an account first. Press the *Register new account* button as shown, and the screen will switch to this:

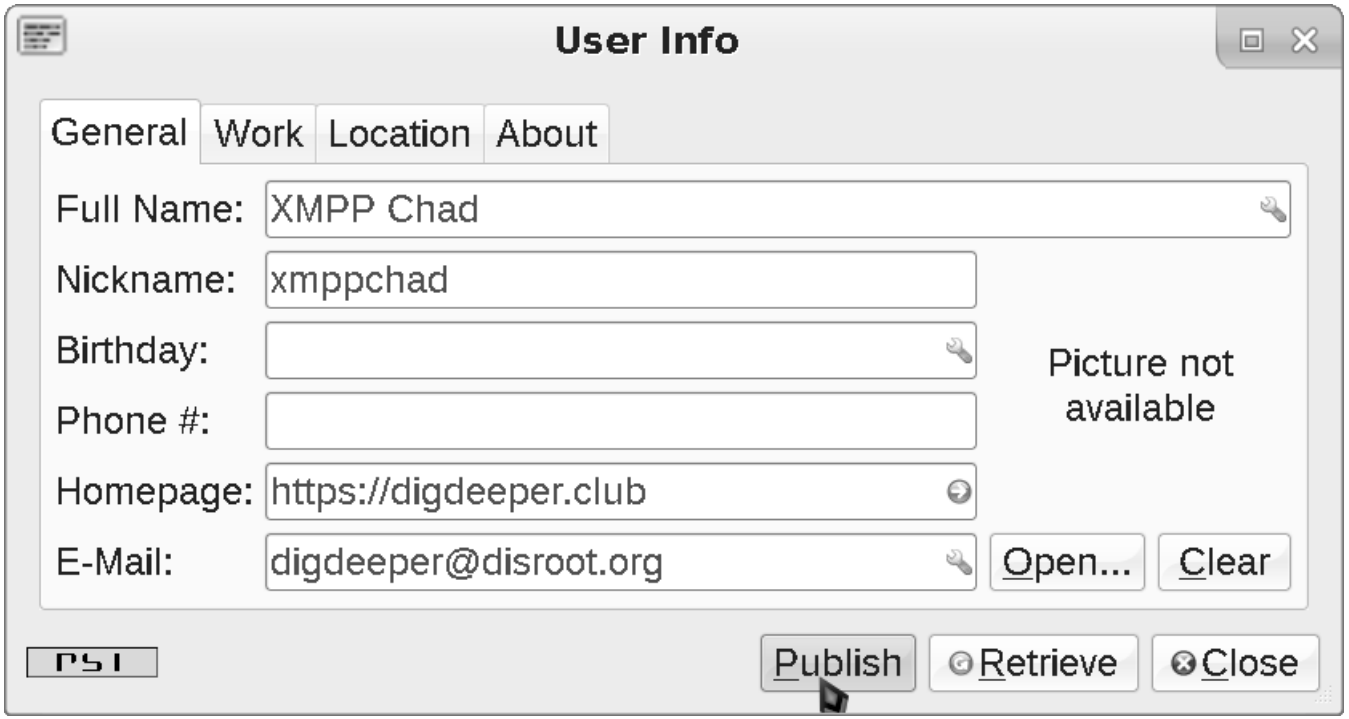
I am using *jabber.cz* for our server, but you can choose any other from the list. However, there is no guarantee all the functionality will be available on every server. Click *Next* and the screen will change again:



Input your desired username (this can be anything) in the relevant field, as well as a strong password (16+ characters that are not easily guessable like birth dates or your kids' names). You might be asked to solve a captcha, so type the number in the picture into the last field. Then click *Next*, and our registration is finalized:


## Filling personal information

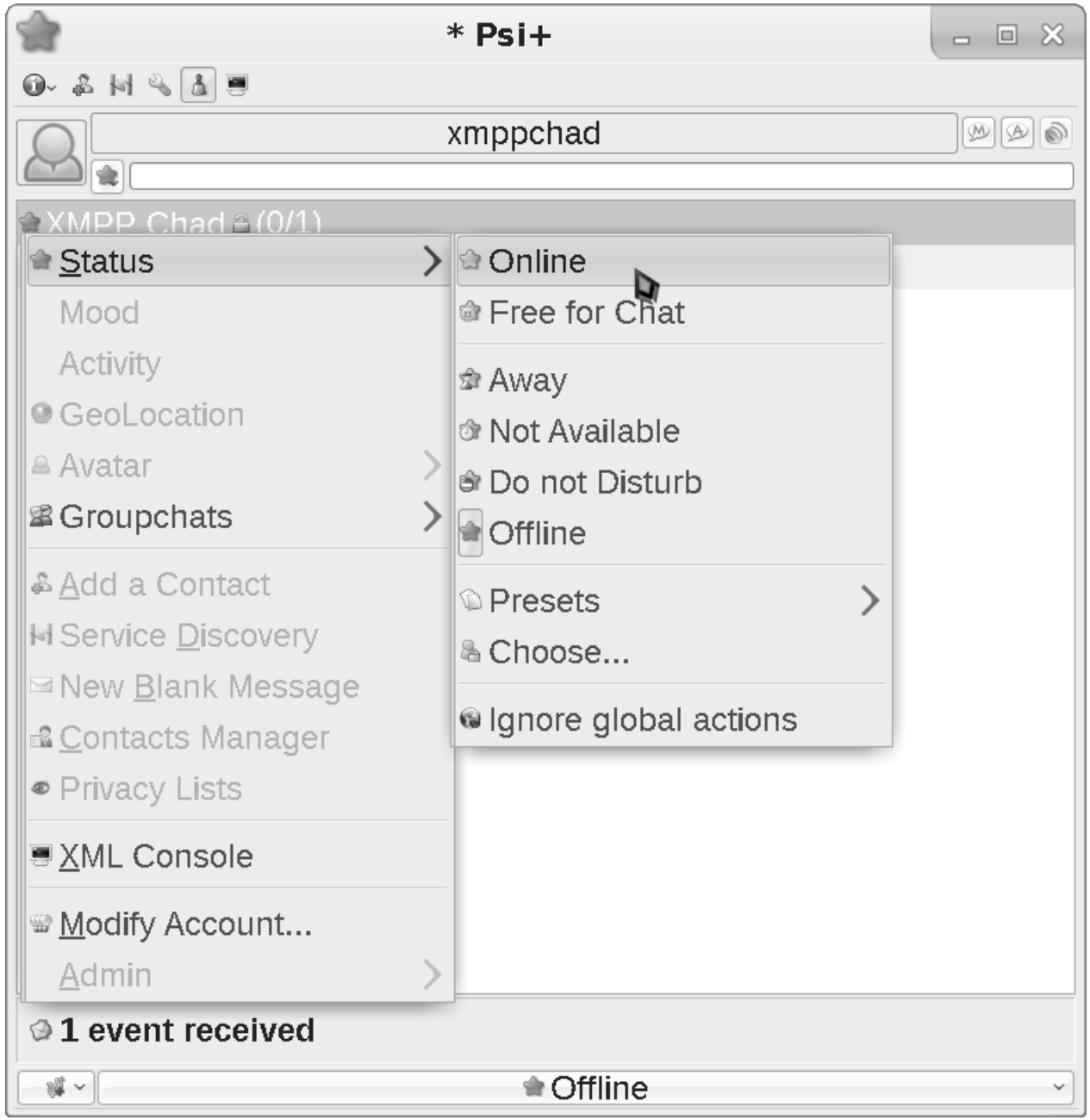
You can fill the *User Info*, but don't have to:




Click *Publish* (if you did fill it), then *Close*. Your server - as well as everyone who you add to your friends list or is in the same group chat - will be able to see this information, but **no real data has to be submitted at any point** (you can leave all the fields empty, if you want to, as well).

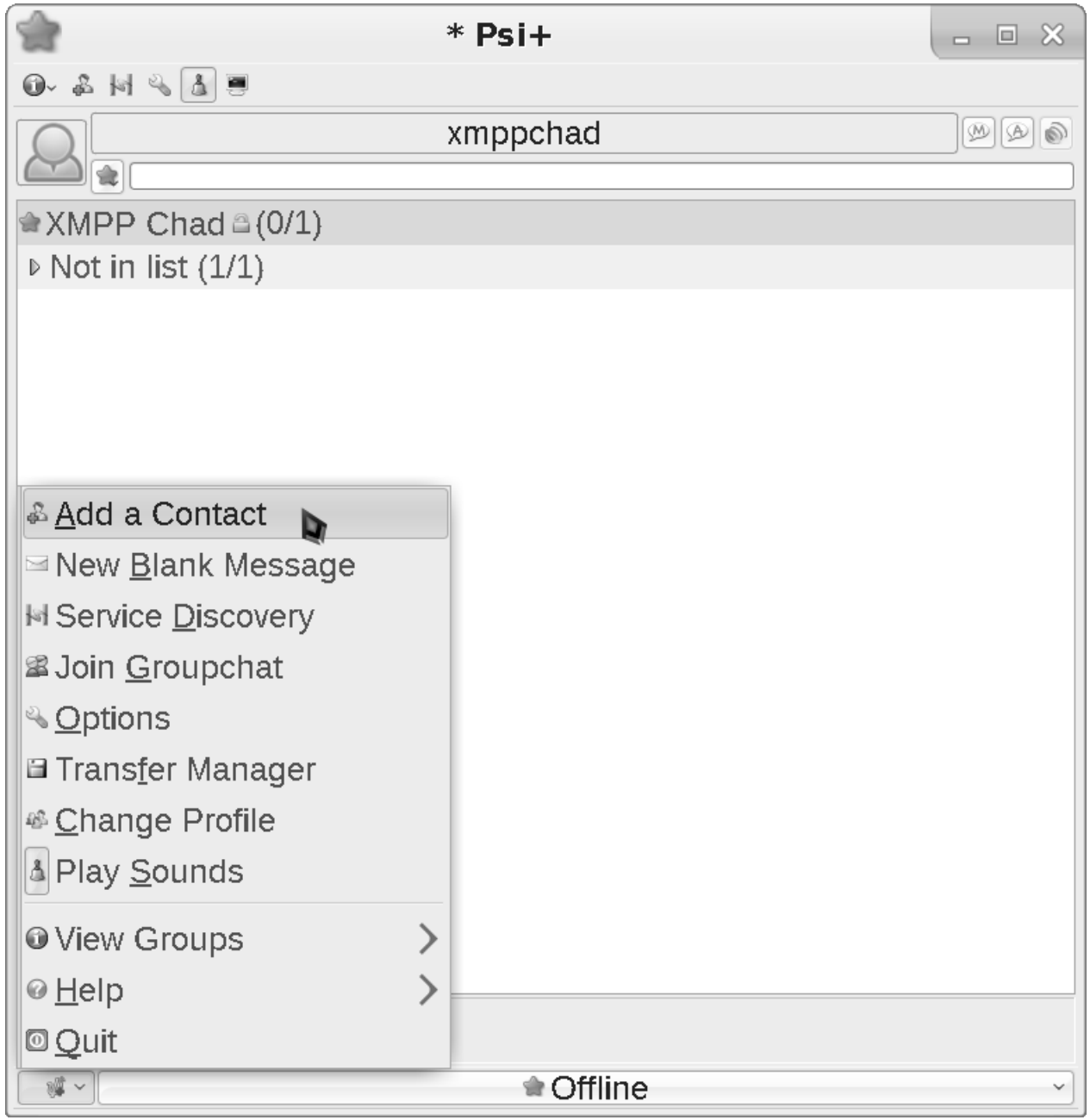
## Going online

Right click your username, and change your status to *Online*:



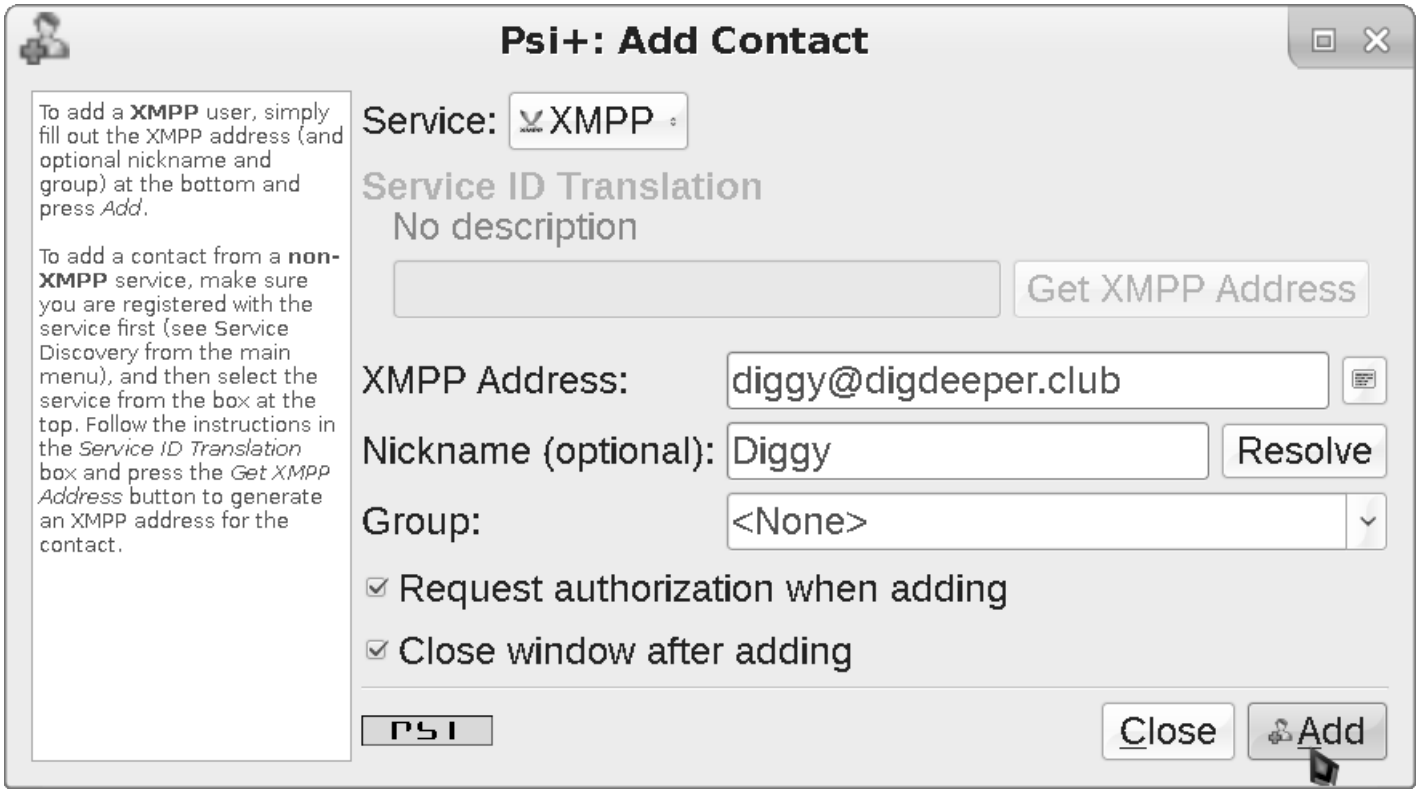

Adding a contact

Click the Psi logo at the bottom left, and add a contact:




You will need the XMPP address of your friend, and whatever name you want to call them

with. Here is an example filled with my details:




When you click *Add*, this will appear:


Your friend (me in this case) will get a friend request, which they will have to approve so that you will get added to their friends list. Though this isn't strictly necessary to message someone

(you can do so while knowing only their address), it will allow you to see each others' status (online, away, etc) as well as set up encryption later.

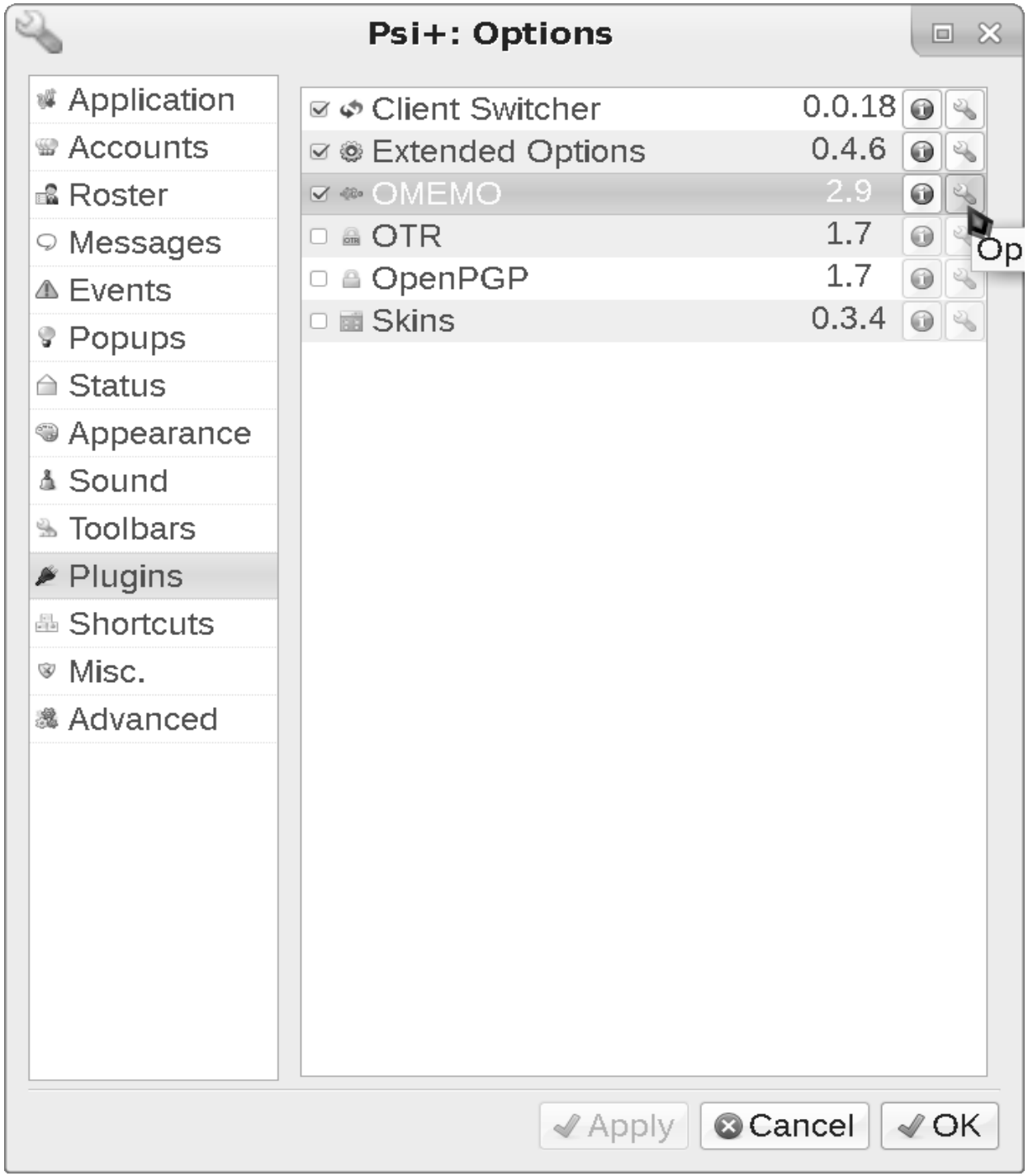
## Starting a conversation

Double-click on your friend's name to bring up the talk window:

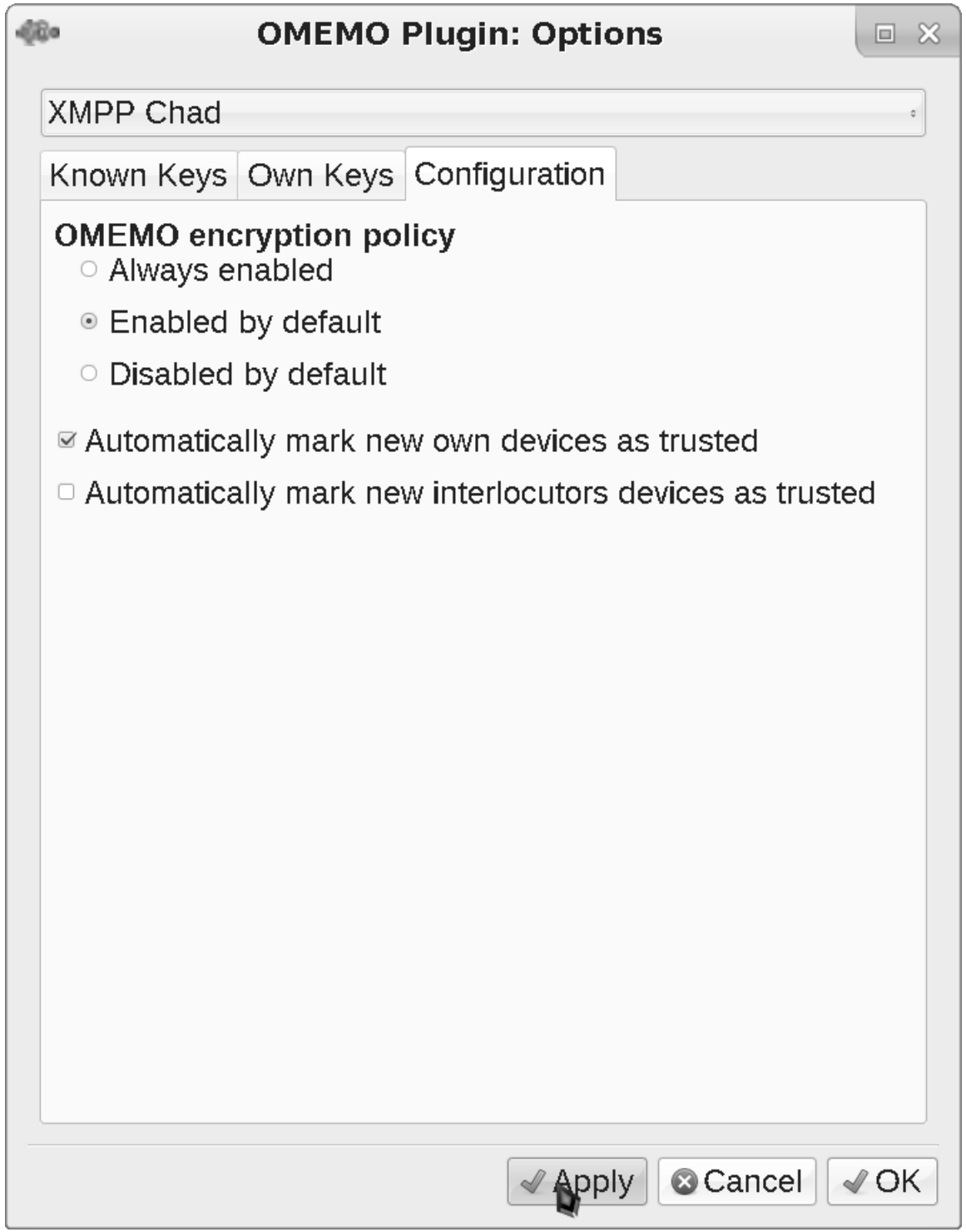
Type whatever you want in the bottom field, and press *Enter* to send. That's it for the basics (I told you it was easy).

## Setting up OMEMO encryption

OMEMO is message-level encryption, which means no one other than the intended recipient can read your message - not even the XMPP servers you or your contact are using. To activate it, go to the *Plugins* menu, enable it and go to its settings like this:




The screen will change. Go to the *Configuration* tab and enable OMEMO by default:


OMEMO Plugin: Options
XMPP Chad
Known Keys
Own Keys
Configuration
OMEMO encryption policy
Always enabled
Enabled by default
Disabled by default
Automatically mark new own devices as trusted
Automatically mark new interlocutors devices as trusted
Apply
Cancel
OK

This will prevent sending unencrypted messages unless you explicitly choose to - but will still allow you to do so when talking with clients that don't support OMEMO. This is the sanest default possible. In some Psi+ versions, it might also be needed to move the OMEMO fish to the message window (*Toolbars*, then click *OMEMO* and press the right arrow) so that you get a notification of whether OMEMO is active, and are able to view and approve / deny your interlocutor's fingerprints. Now *Apply* and leave the menu, and let's finally try to message someone:

We see the OMEMO toggle is lit up, yet the prompt shows us that the *"OMEMO encryption is not available"*. Here is what happens when we send a message in that case:


This message won't be received by your intended recipient. This is because they need to verify your fingerprint before the encrypted conversation can proceed. This is a security feature that's intended to prevent MitM attacks - *"If a client were to opportunistically start using sessions for sending without asking the user whether to trust a device first, an attacker could publish a fake*

*device for this user, which would then receive copies of all messages sent by/to this user."* Ideally, everyone should be publishing their OMEMO fingerprints on a website or elsewhere, so that they could be compared to the one your XMPP client shows (I do that here). The reason it needs to be done out of band (as in, outside of the current communication channel, which in this case is the XMPP conversation) is that presumably the possible attacker does not control the other place, as well. The point is to have something trustworthy to verify the shown fingerprint **against**. Imagine if you tried to compare the fingerprint to a message sent through XMPP itself. If the attacker substituted a fake device, they could just as well insert a fake message (*"The fingerprint XX XX XX that has just appeared to you is totally real, bro"*), and pretend the shown key is real. Of course - again - your recipient needs to have an outside channel for this to matter in the first place. Anyway, once your interlocutor verifies your fingerprint like this:

If you try to type an encrypted message again, this happens (re-logging might be necessary):


Before you trust the fingerprint, verify it's the same as the one shown on my site. Of course, if your recipient does not provide their authentic fingerprint anywhere else, you cannot do that and must accept (which is less secure, but still much more than sending messages without OMEMO; you'd need to be a victim of a targeted attack right at the point of the first key exchange for it to matter) or deny it blindly. Verifying the fingerprints out of band is a good

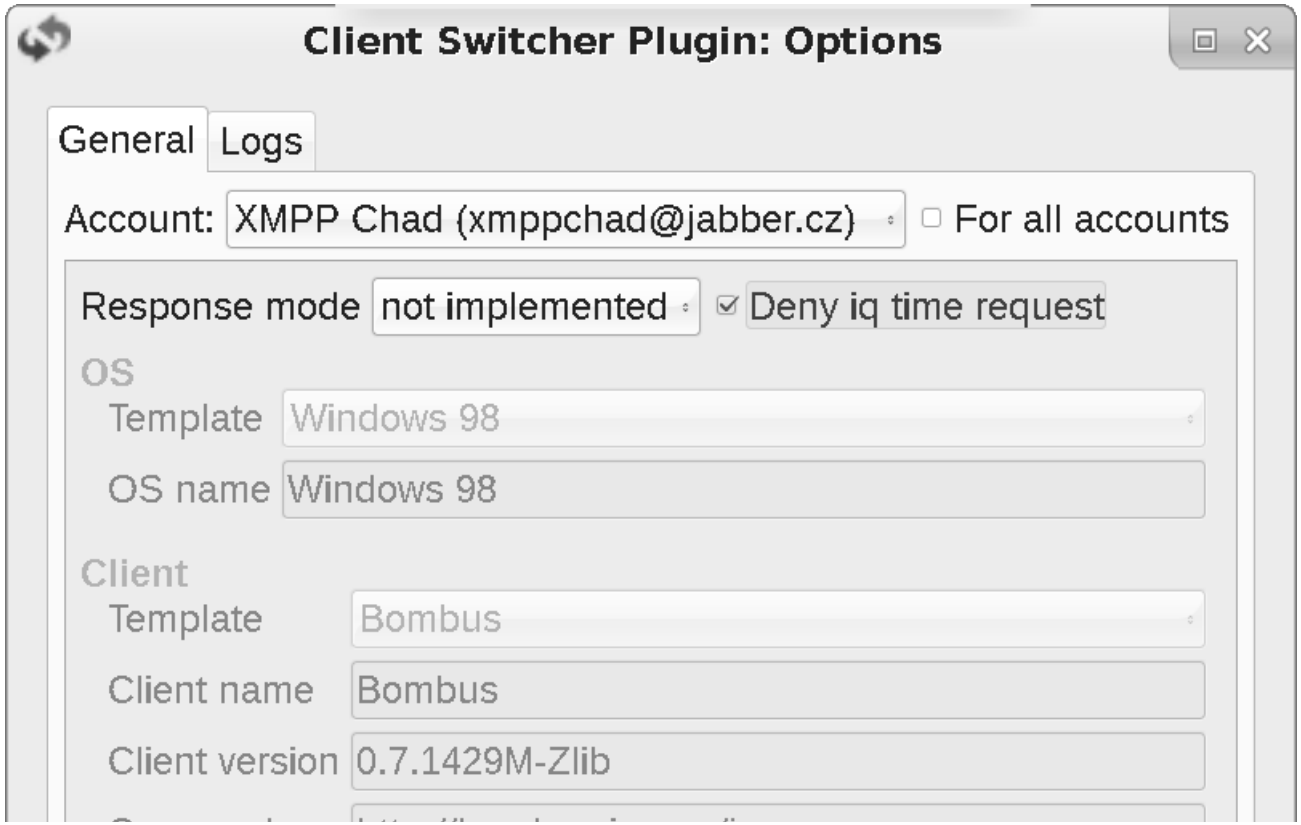
habit to have if your OPSEC requirements are high, though. Okay, now the encrypted conversation can finally proceed:


Now, remember the OMEMO plugin settings menu, and the *"Automatically mark new interlocutors devices as trusted"* option? Well, in light of the above explanations, you should understand this option as simply skipping the fingerprint verification - and realize how insecure it is. That's why we keep it unchecked. Some clients do this by default, by the way - it's called

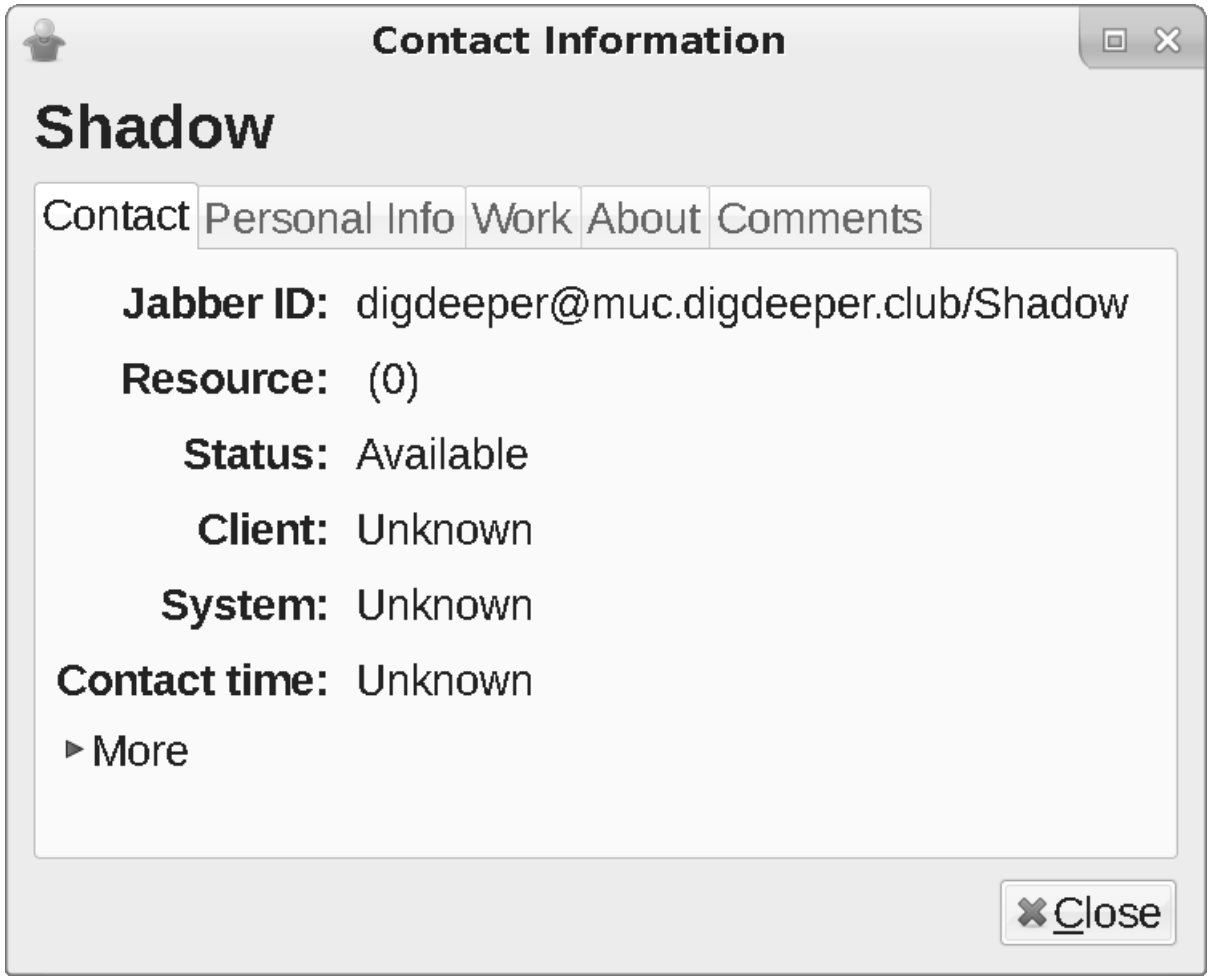
TOFU (*Trust On First Use*), and again - it's not secure, allowing undetectable MitM.

## Privacy mitigations

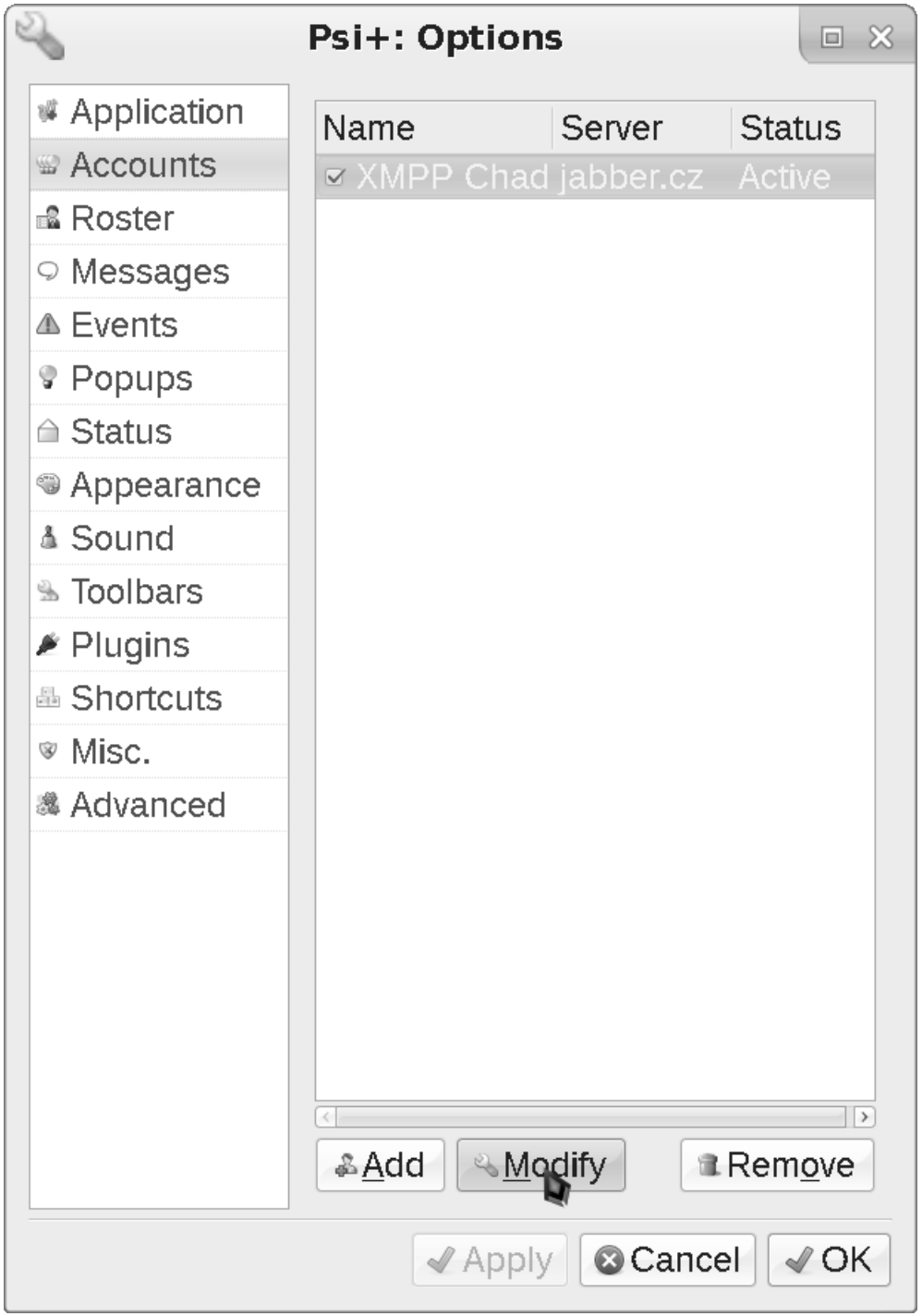
Psi+, by default, reveals your timezone, client (version and creation date), and system information (OS / Linux distro version + kernel version). The way to mitigate it is by using the plugin *Client Switcher*. Choose *"not implemented"* as the *"Response mode"*. Then mark *"Deny iq time request"* (this disables the timezone leak). Finally, you need to *"Enable for"* *"Contacts"* and *"Groupchats"* (self-explanatory). This will show *Unknown* for all categories when inspected. You can also falsify your client and system info - **only Psi+ can do that** (not shown in this example though).




Caps node
http://bombus-im.org/java
Caps version
0.7.1429M-Zlib
Enable for:
Contacts
Groupchats
Show popup at version iq
never
Save queries to log
never
Attention! Thoughtless usage of Client Switcher Plugin may cause to inability of using OMEMO and OpenPGP encryption. Use functions of this plugin very carefully...
Wiki (Online)
Apply
Cancel
OK

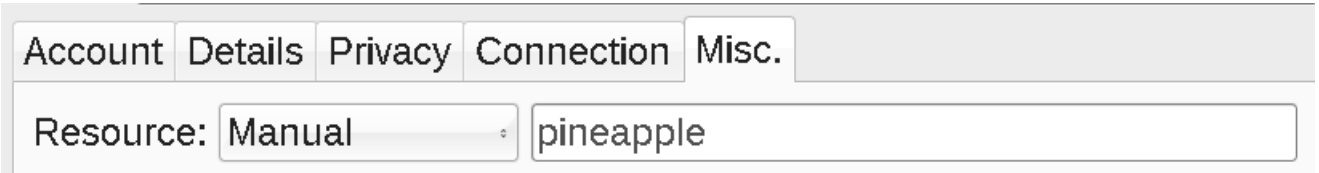



Every client still reveals a *Resource* header to group chat admins and all roster contacts, and in Psi+ that contains your hostname (what you see when you type *"hostname"* in the terminal) by default, which can be sensitive. To disable that, go to the accounts screen, and click *"Modify"*:

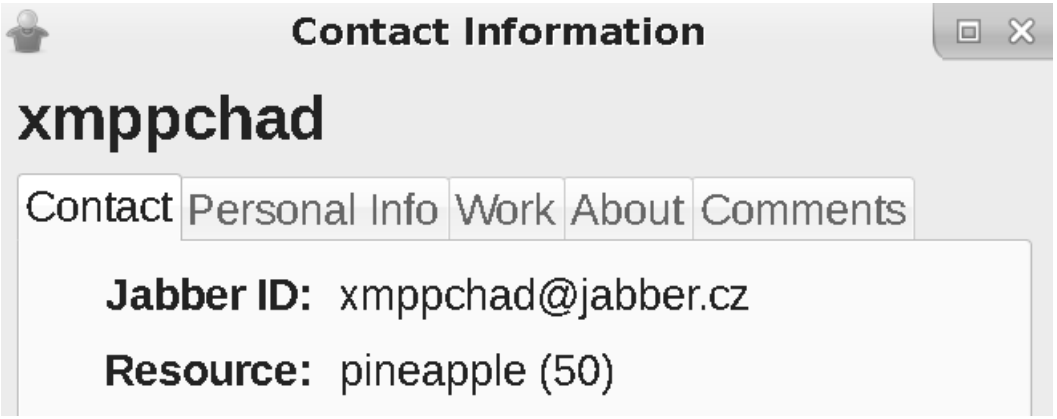



You can change it to whatever, but remember that Gajim and Psi+ are the only clients that are

able to modify the resource header and not leak anything else, so your contacts will know you're using one of those two, either way. You can also pretend to be Dino by setting your Resource header in the format of *"dino."*+8 numbers and letters. E.g *"dino.cfg231e9"*. Save after, and re-login.




Results in this when inspected by a contact or group chat member:




# Using XMPP - Gajim 0.16.9

## Introduction

**UPDATE April 2023**: I guess it is fitting to post a guide for the client I actually daily drive. Version 0.16.9 includes all relevant features (but no bloat unlike Psi+), a good UI (it is the only

usable client still compatible with gtk2) and OTR support (nuJim dropped that and won't bring it back - *"The plugin won't work with Gajim >= 1.0 as it was not ported to python3. There is no port planned."*). The reason OTR support is important is because there are clients out there (e.g mcabber) that do not support OMEMO but do OTR, and the lack of it would prevent encrypted communication with them. There are no known security issues with this version, either. Newer versions have increasingly become more Discord-like with nothing else to justify them; you can read the complaints on the internet - but in this guide, we'd rather be focusing on something good which is Gajim 0.16.9. **UPDATE June 2026**: one "vuln" that I didn't notice up until recently is the fact that SCRAM doesn't appear to be supported, meaning **all passwords are saved in plaintext**. This means anyone or anything with access to your home folder can steal them. Of course, this only matters if you do save passwords in the first place.

## Installation

To install Gajim 0.16.9 in Slackware 14-based distros, you need these packages:

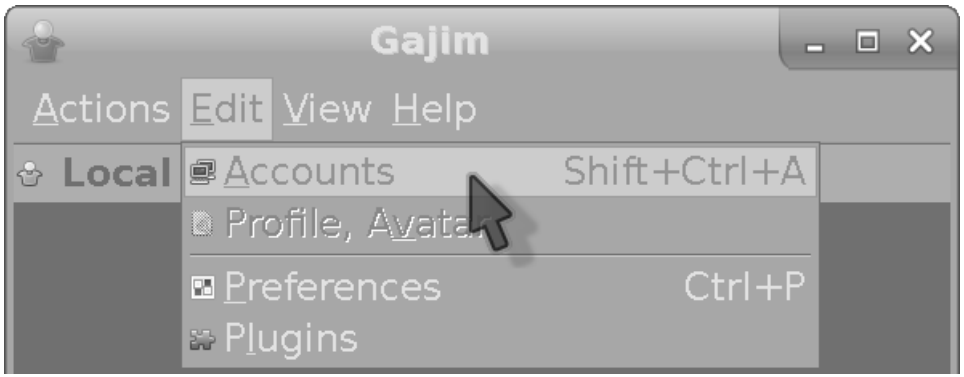
- asn1crypto-0.24.0
- at-spi2-atk-2.18.1
- gajim-0.16.9
- gnome-python-2.28.1
- idna-2.6
- ipaddress-1.0.16
- libunistring-0.9.10
- notify-python-0.1.1
- protobuf-2.6.1

- pyOpenSSL-0.15.1
- pyasn1-0.1.9
- pyasn1-modules-0.0.8
- pycrypto-2.6.1
- pycurl-7.43.0
- pygobject-2.28.6
- pygtk-2.24.0
- python-axolotl-0.1.39
- python-axolotl-curve25519-0.1
- python-cffi-1.5.2
- python-cryptography-1.3.1
- python-enum34-1.1.2
- python-nbxmpp-0.6.2
- python-six-1.11.0
- service_identity-14.0.0

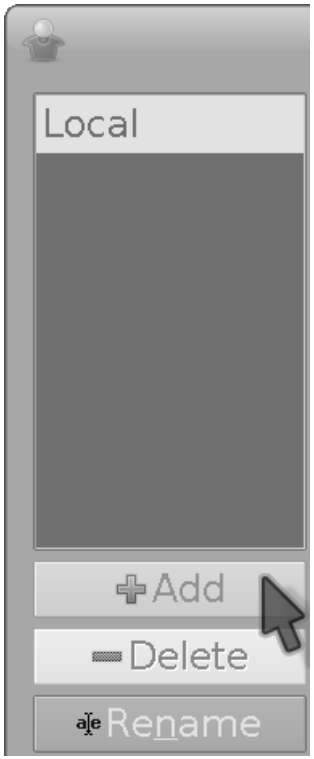
Versions need to match at least for some of them. Newer e.g pyOpenSSL will be undetected, while other mismatches might make Gajim fail with different random errors. I've spent hours (and a handful of pulled hair) hunting down this exact set of dependencies, and I know it works; if you want to try another, you're on your own (have fun digging through the useless or even misleading errors Gajim will spit). **This applies only to Slackware 14.2**; you can still run this program on 15, but the requirements are a little different - *cffi* changes to *1.13.2*; *enum34* to *1.1.10* or higher; *idna* to *2.9* or higher; *pyasn1* to *0.4.4*; *pyasn1-modules* to *0.2.2*; *cryptography* to *3.3.2*; *pygobject* to *2.28.7*; *pyOpenSSL* to *21.0.0* or higher; and *six* to *1.14.0* or higher.

**UPDATE October 2025**: The needed axolotl versions seem to be missing from repos completely now; I think the support for 0.16.9 is dying. All the deps take about 10mb compressed; while for Psi+, qt5 alone (plus the stuff it pulls) is over 100mb compressed and over 600mb uncompressed. Gajim 0.16.9 is contained in e.g the Ponce's repository, or you can use slapt-src (remember the no-dep option or it will pull some BS deps). This client can also be installed on 32-bit versions of Slackware. Anyway - assuming the installation went successfully - let's dive right in and register an account (you can do it inside the client itself):

## Adding an account




Enter the menu shown, and the screen will switch to this:

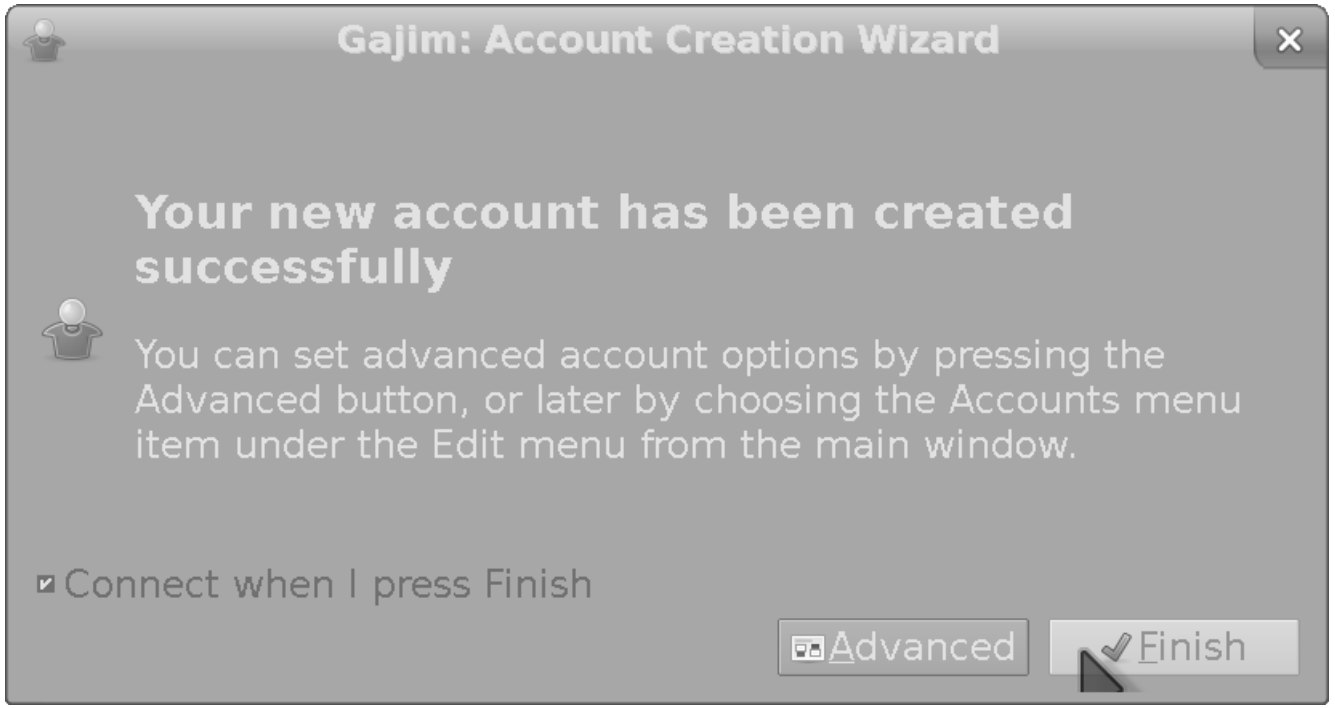



Click *Add* here.


Pick the relevant option and click *Forward*.

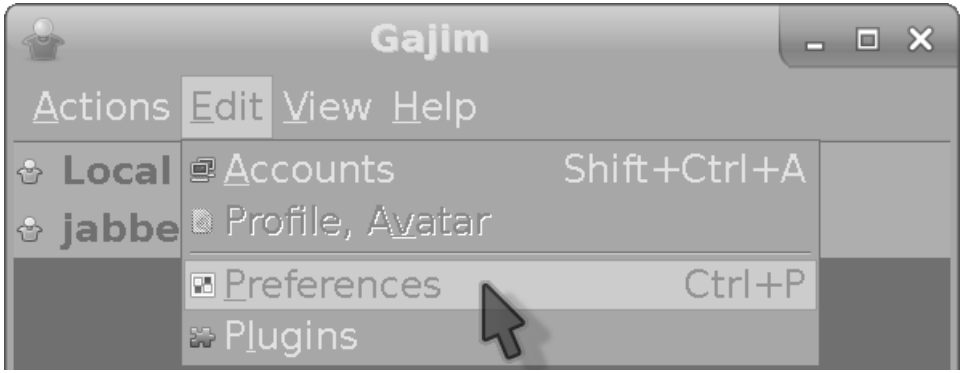

In this screen we choose the server our account will be on. We will use *jabber.sk* for this guide, but you can pick any other. Keep in mind they all have different properties; some might block TOR, in-band (client) registration or lack certain features. Anyway, jabber.sk does allow us to sign-up through the client while using TOR; if you want to do that, just expand *Advanced* and set it as the proxy (this is an in-built option in Gajim). Remember you need to have TOR running on your machine in the first place for it to work. You can leave the proxy empty but will be revealing your IP to the server, then (unless using a system-wide VPN). Click *Forward* as usual.


Insert the username and password that you want to have (the latter you can change later). Then, type in the number you see in the displayed image into the captcha (last) field; not all servers require this. If the registration was successful (e.g you didn't take too long to fill the captcha), the screen will change to this:

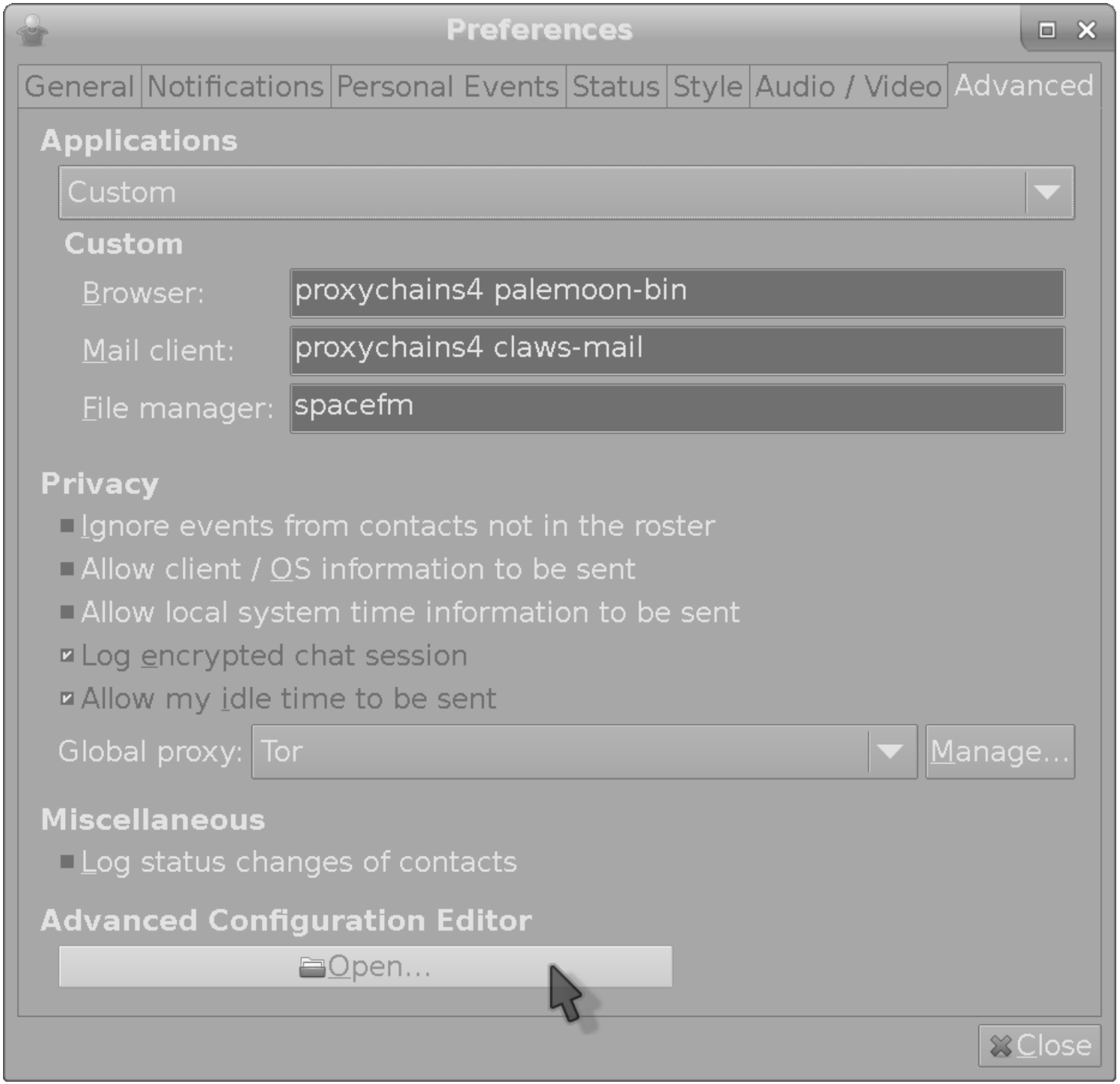



## Privacy mitigations

We're going to do some privacy mitigations for Gajim now. I will assume here you are using a TOR-only setup and want everything to go through it. Enter the *Preferences* menu:

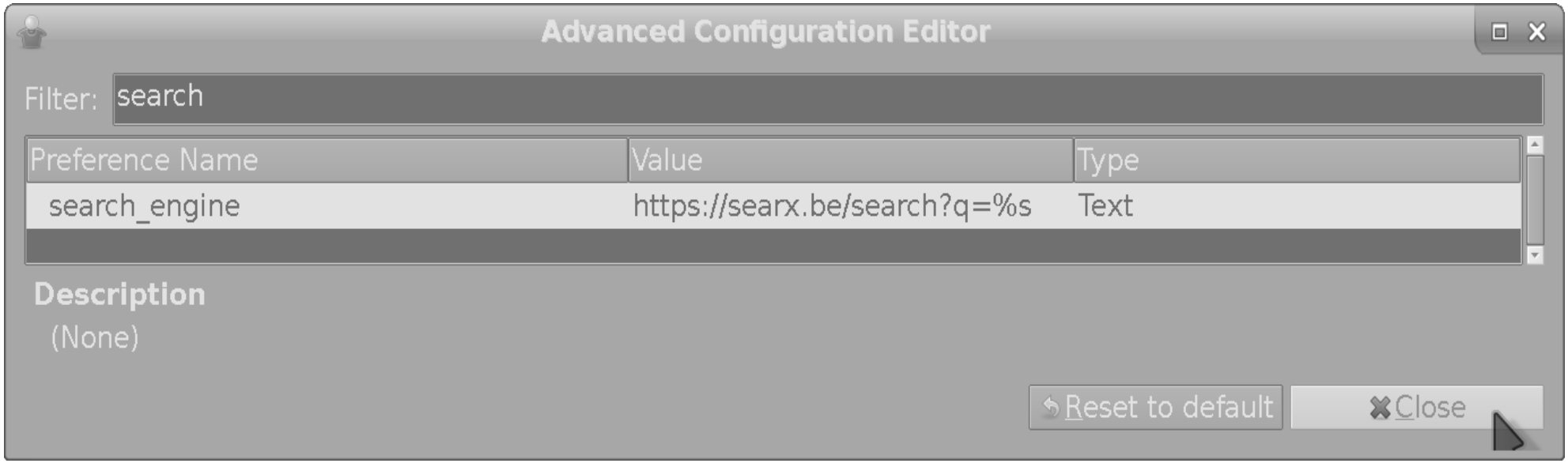


Now switch to the *Advanced* tab and set everything like this:


Preferences
General
Notifications
Personal Events
Status
Style
Audio / Video
Advanced
Applications
Custom
Custom
Browser:
proxychains4 palemoon-bin
Mail client:
proxychains4 claws-mail
File manager:
spacefm
Privacy
Ignore events from contacts not in the roster
Allow client / OS information to be sent
Allow local system time information to be sent
Log encrypted chat session
Allow my idle time to be sent
Global proxy:
Tor
Manage...
Miscellaneous
Log status changes of contacts
Advanced Configuration Editor
Open...
Close

We're setting custom applications so that whenever someone posts a website or E-mail link in a chat - and you click it - Gajim will open it through TOR instead of in the clear which would reveal your IP address. This requires proxychains to be installed and setup in the first place. Of course, if you're using something other than Claws Mail and Pale Moon, input those instead (but keep the *"proxychains4"* part).

We're setting TOR as the global proxy, for obvious reasons. This means every account - even ones made after this - will use it. Disabling *"Allow client / OS information to be sent"* as well as *"Allow local system time information to be sent"* prevents Gajim from leaking data which it does by default. An important to note fact is that **Gajim retardedly regenerates the leaks upon the creation of every new account**, so you will have to disable them again every time you create one. Don't close the *Preferences* menu yet and instead open the *Advanced Configuration Editor*.

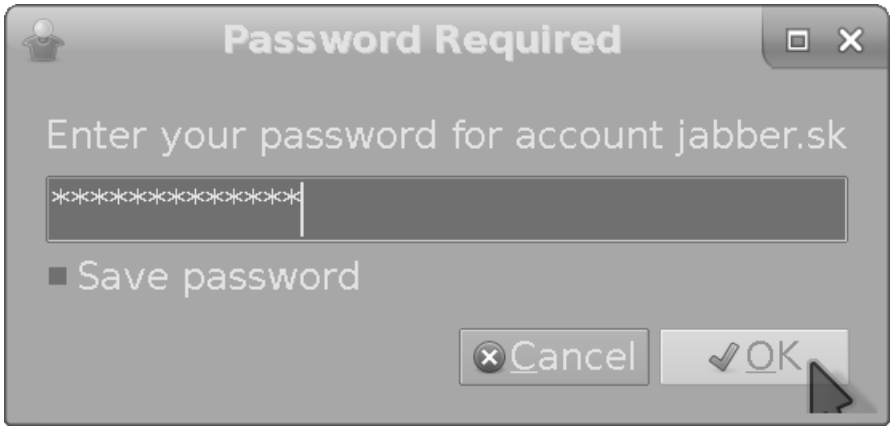



By default - when you right click on a selection and choose the *Web Search for It* option - Gajim will send your query to Google (ugh). Find the *search_engine* preference, click the *Value*

part and change it to your favorite SearX instance, or even something like Qwant Lite with *"https://lite.qwant.com/?q=%s"*. Press Enter and finally *Close*.

## Going online

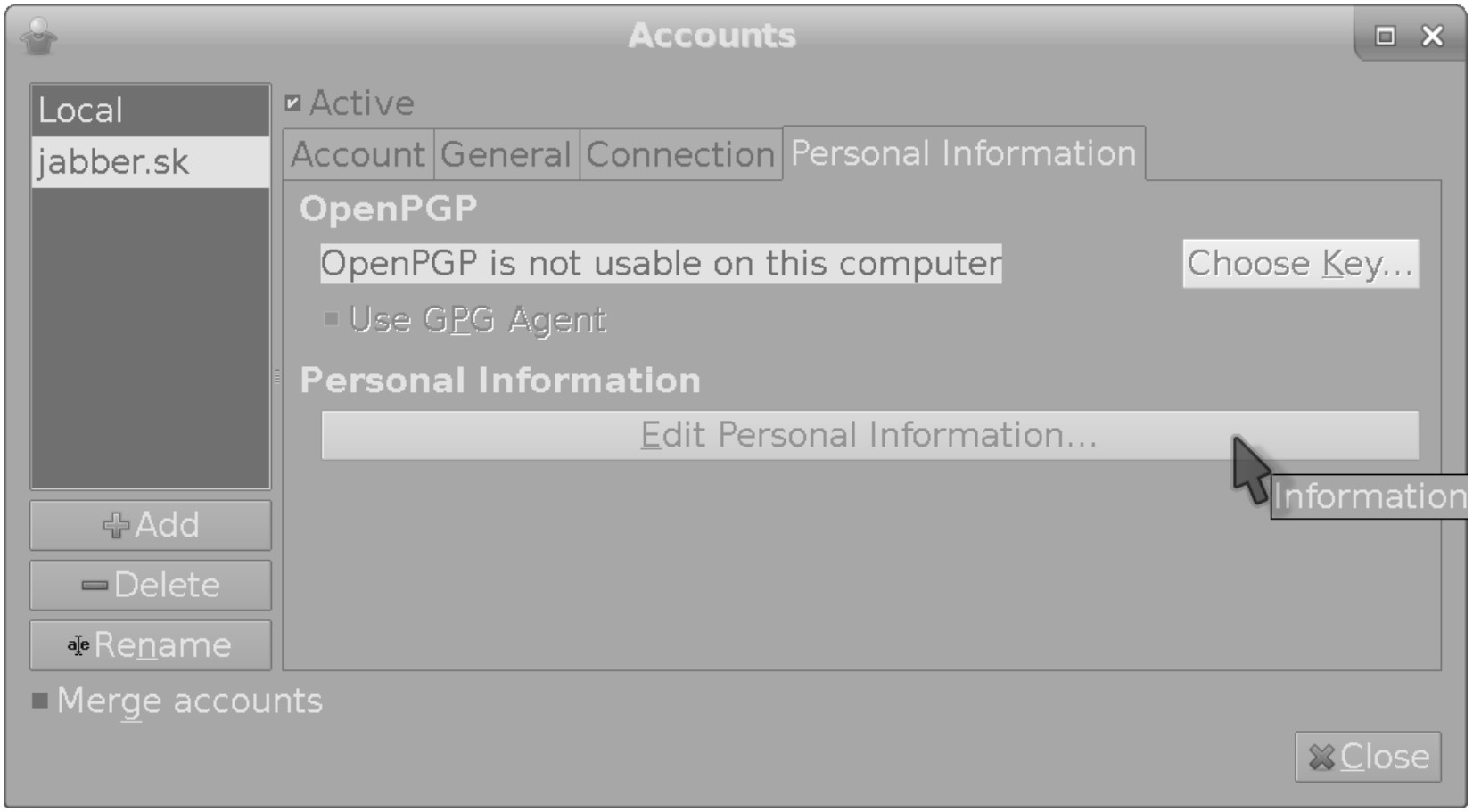

We can go online now. Right click your account name and change your status to *Available*. Then the screen will change to this:




Enter the password you've set before and click *OK*. If you want to login automatically on launch, you can mark the *Save password* option. I don't like it because it exposes you to local attack, e.g by a girlfriend. Anyway, we're going to set up our profile now (this step is **completely optional**):

## Filling personal information

Enter the *Accounts* menu again, but this time switch to the *Personal Information* tab:

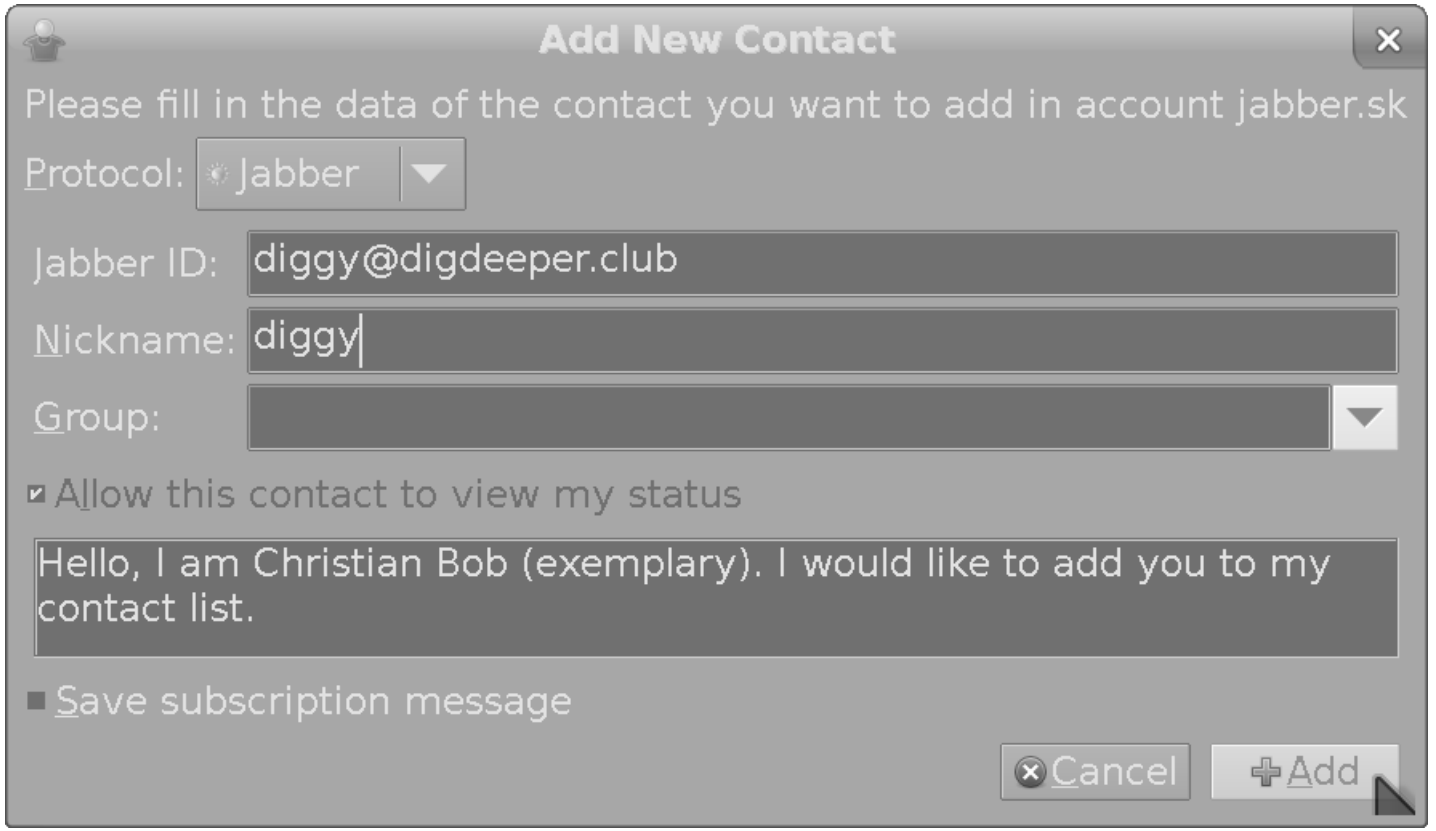



Click *Edit Personal Information* and the screen will change to this:


Everyone you add to your friend list or enter a group chat with will be able to see this information, so be careful of what you put there; you can also leave it empty. Some of this data is also automatically inserted into your subscription message for whenever you add someone to your roster. To be perfectly clear, **you do not have to fill in any of this**. But if you did, click *OK* to publish the information and let's finally add a contact:

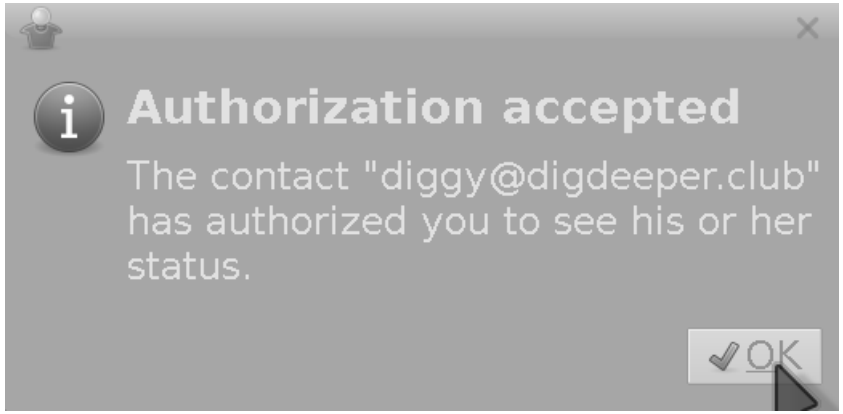
## Adding a contact


Enter this menu and the screen will change to this:




*Jabber ID* is the XMPP address of your friend. Nickname can be anything you want, but better leave it as something that will remind you of the person you're adding. The subscription message at the bottom is what your contact will see when they login and receive the friend request (it's the default, and fills in the info you've put in the *Personal Preferences* menu, as you can see). We will leave the *Group* empty since it's not necessary. When you send the friend request, this is what your recipient will see:


If they accept, you will see this:




## Starting a conversation

Double-click your friend's nickname in the roster window...


and the talk window will appear:


You can type messages in the bottom field, and send them by clicking *Send* or just pressing Enter.


The above is what your recipient will see.

## Setting up OMEMO encryption

Go to *Edit* -> *Plugins* -> *Available* and set the checkmark near *OMEMO*, then click *Install / Upgrade* on the right. You will get a security warning because Gajim won't be able to verify the cert; if you're worried you can download the zip from their site with your browser, which I have also mirrored locally just in case they deprecate the older version's plugins. After that, the procedure is similar to the Psi+ one. Just message someone with OMEMO enabled and a list of fingerprints will appear, then accept (first confirm out of band if possible). By the way, you need to add someone to your roster for Gajim to receive the "device list" and display the

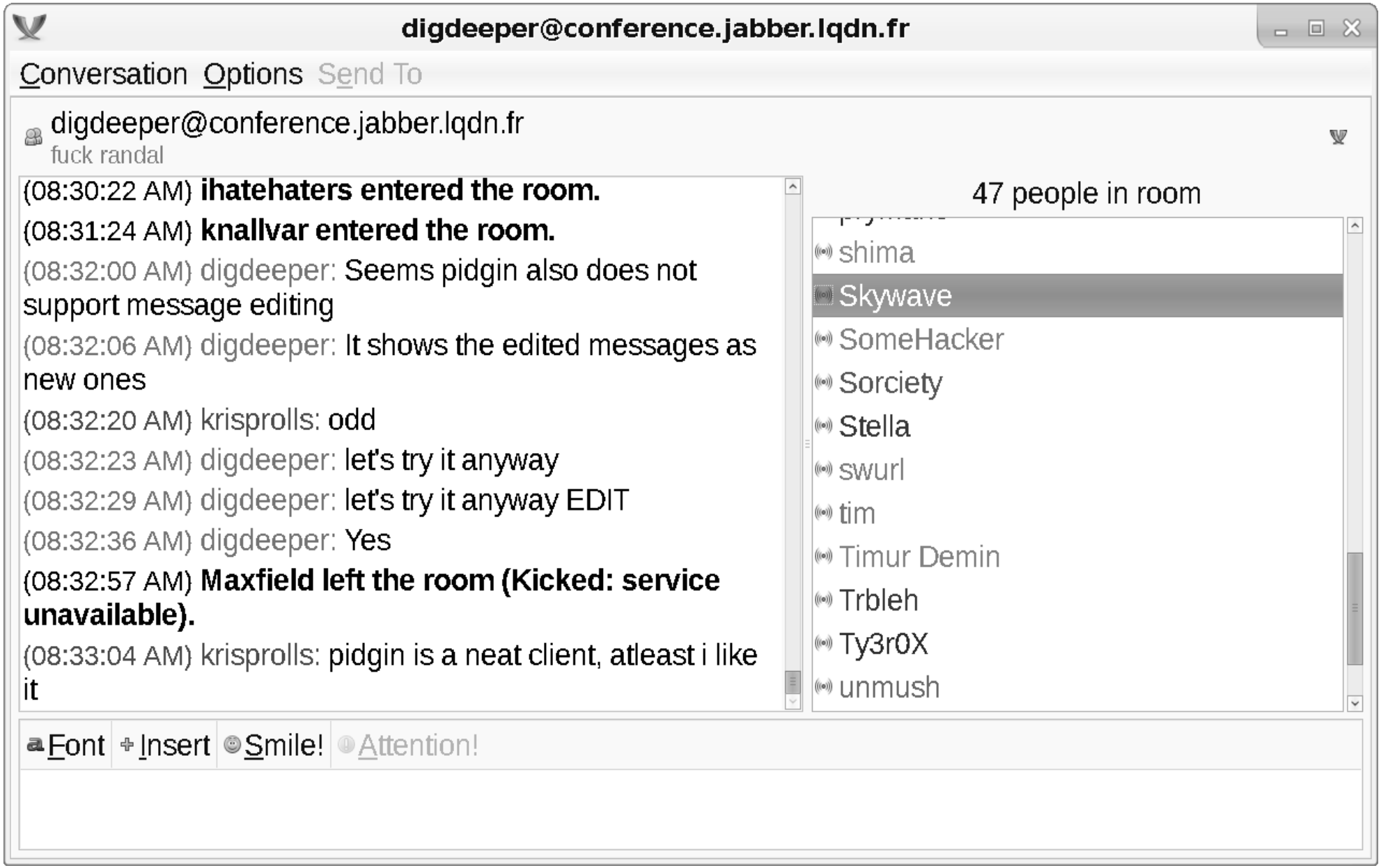
fingerprints. But, you can revoke the subscription later and keep using OMEMO with someone who is no longer in your roster. However, if that person gets a new fingerprint, you won't be able to receive it and will have to re-add them. This is the case in even the newest Gajim versions it seems. **UPDATE February 2026**: and - if someone gets a new fingerprint, and you send an OMEMO-encrypted message before accepting it - it won't reach the target. This probably works different in TOFU clients, but the way Oldjim does it, is more secure. Just telling you this so that you are not surprised if - at some point - you sent a message to someone but they act like they didn't receive it.

# Other clients

## Pidgin

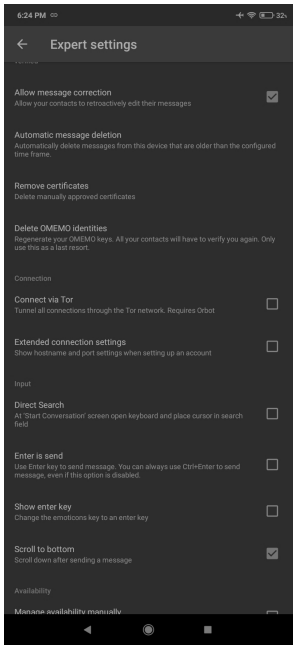
Doubles as a client for IRC and some other protocols. I really like the UI, which relies on GTK2. Leaks your client and version (for example, *Pidgin 2.11.0 (libpurple 2.11.0)*) and timezone, but **not system info** unlike Psi+ (it shows up as *Unknown*); these leaks also cannot be disabled. Supports OTR and OMEMO, but the OMEMO plugin is terminal-based and sucks (cannot accept or remove fingerprints). Decent program in terms of design, but due to the data leaks and bad OMEMO support, cannot be recommended. However, if you really want to use Pidgin, **you can nullify the timezone leak by changing the system's timezone**; this does not fully prevent it, just displays a fake value. This way will also affect other things in your system, like the clock. **Edit**: on Windows, RunAsDate can be used to spoof the timezone. **Pidgin cannot connect to onion domains**, failing with the *"SSL handshake failed"* error - regardless of any settings. Yet another significant flaw, which means Gajim is superior unless you need the other prococols. Pidgin does not support message editing either, showing the modified messages as

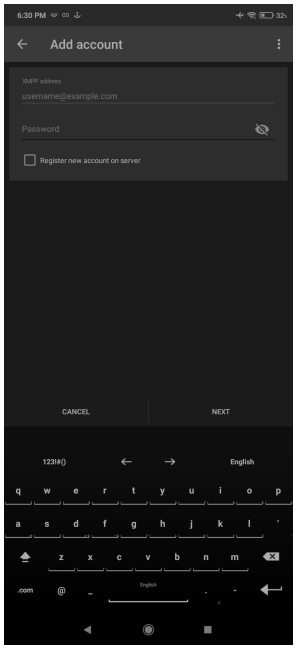
new ones:

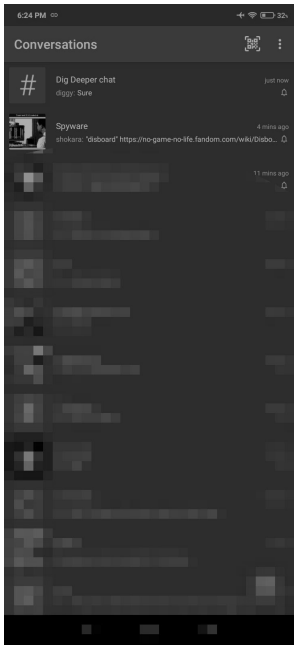


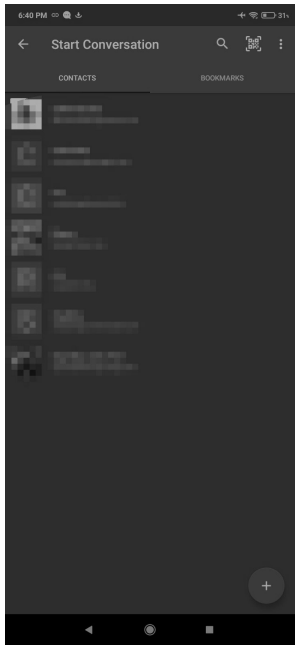

Conversations

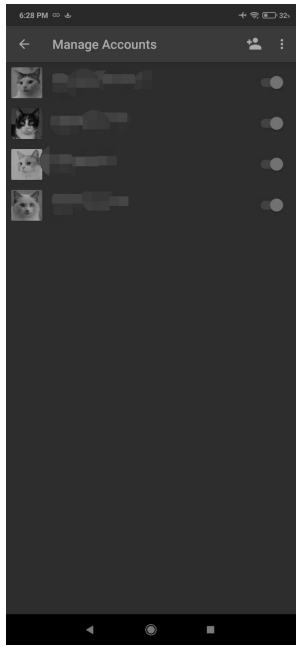
Leaks client and version (for example, *Conversations 2.10.2+fcr*) plus timezone and the fact that it's on Android (but no specific system info). This applies to all the Conversations family (blabber etc). **Using them through TOR prevents the timezone leak**, but still leaks everything else. All Android XMPP clients auto-invite you to group chats, leading to easy trolling. There is no way to mitigate any phone client in full unless you use Diggy's black magic. Usability wise, compared to ChatSecure, *"It's a little clunky... Scrolling and menu changes jump around and switching between group and one on one chats is horrendous. I also have to go through two sub sections to change OMEMO settings to blindly accept or save or reject. With ChatSecure I could look at OMEMO keys in each chat. Conversations requires me to go out of chat to main settings and then a sub configuration menus. Just tedious. I don't get any indication on your status. ChatSecure had a green ring around each avatar when someone was online and how many hours ago."* For something more positive, *"ChatSecure never told me when someone left but Conversations actually did with a visual change too"*.



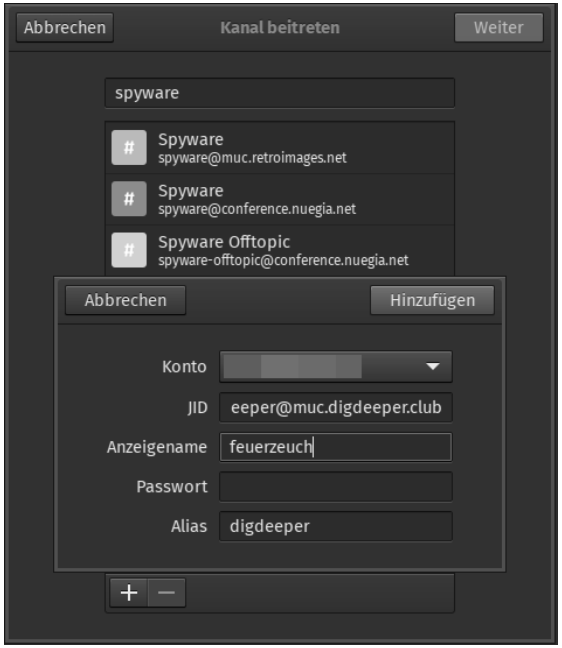



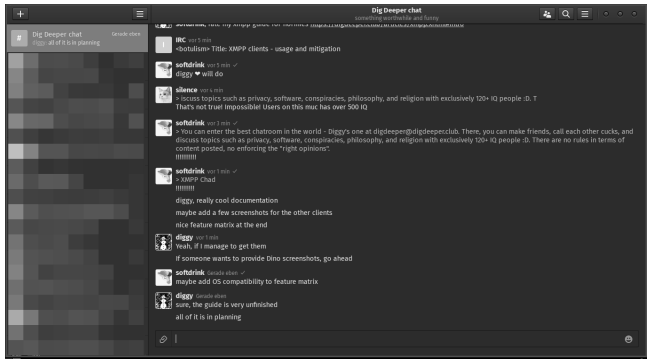



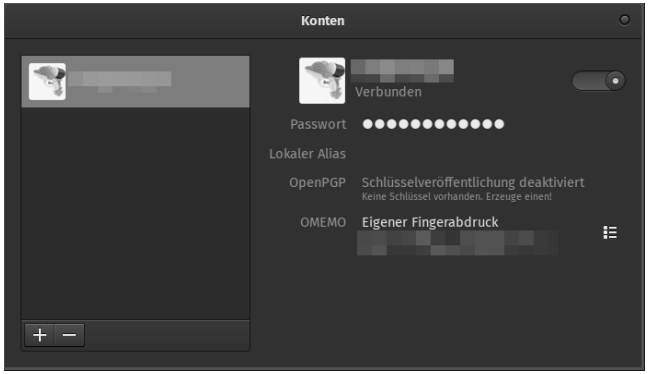



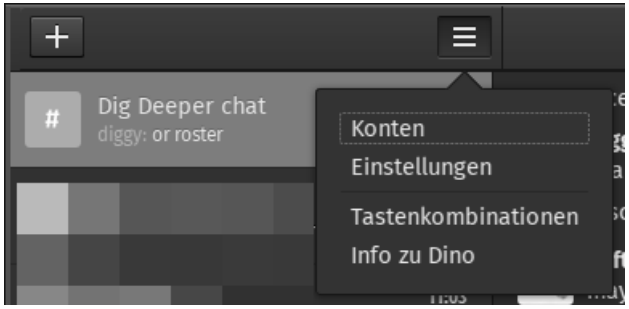
# Dino

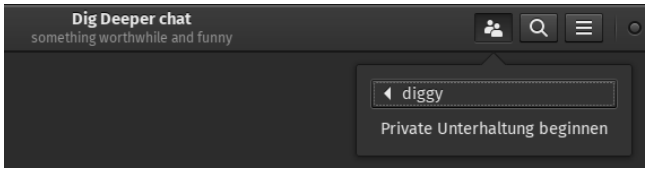
Does not leak timezone, client version, or system information. Does leak its name in the resource header - this means that all clients expose it, instead of just Psi and Gajim. **UPDATE May 2022**: new version has multiple account support and MUC invitations, so it's not as useless now. It is still pretty barren if you look at the screenshots - almost like a phone client brought to desktops. **Cannot connect to onions**. GTK3 dependence. No OTR; does support OMEMO well though.






## Profanity

Terminal-based. **UPDATE July 2026**: I don't know much about it. Supposedly it is really good now, but I haven't had the chance to truly test it yet.

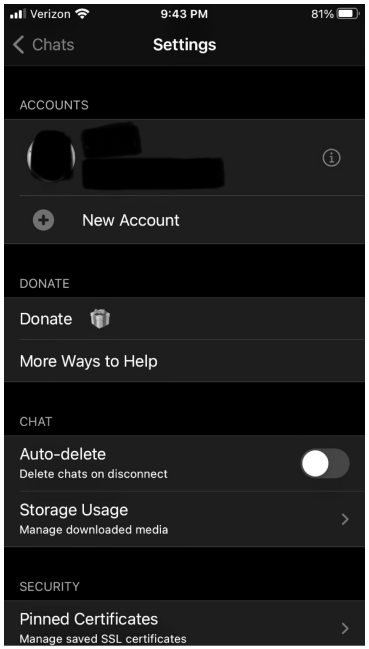
## ChatSecure

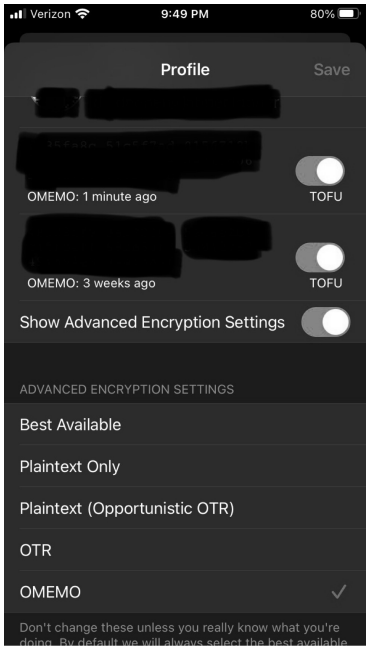
An open source and fully available on GitHub for the IOS App. Unfortunately it's designed specifically and only for IOS mobile devices which are inherently insecure due to the need for Apple's store, and their proprietary data collection methods that cannot be mitigated. In addition to a myriad of privacy risk with using wifi and cellular data plans, Apple's proprietary exclusivity to everything really. Compared to the Android clients, **ChatSecure does not reveal its version, timezone, or system information** - only the client name in the resource header; this makes it one of the best in terms of data leaks. Notable features that give it some grace:

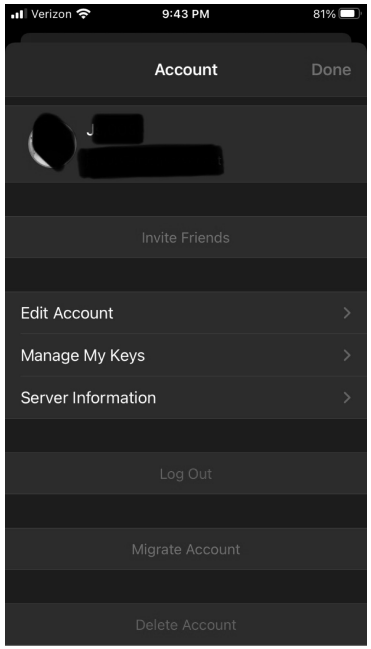
ChatSecure **supports OTR**. Not auto-joining group chats or auto accepting OMEMO keys. Has the ability to OMEMO group chats on a per chat basis. ChatSecure also lets users reject or save new OMEMO keys. If you and a friend have not chatted for a while a default message appears

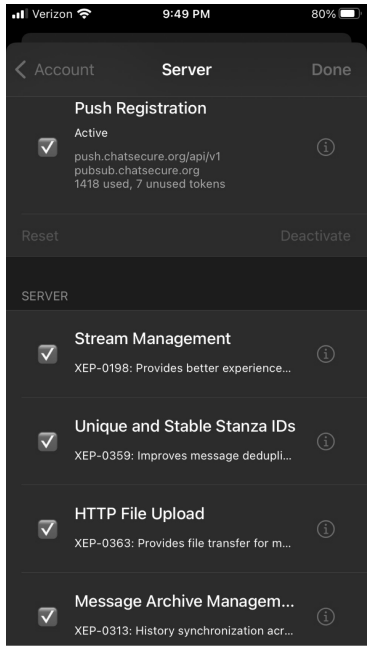
and encourages you to verify who you are messaging. Users can also enable auto chat delete when disconnecting and manage downloaded media. Also configuration options for pinned certificates to manage saved SSL Certs. It grants users to add multiple accounts and edit their server information too. It appears to work cross client well too with users using older versions and current production versions. Ironically it lacks the basic copy and paste ability (big frustration with other languages). It is easy for anyone but lacks many features that are necessary for the goal of XMPP/OMEMO, such as MUC whispers. Oh, and you can't disable typing notifications, so your interlocutor will always know the exact moments you are typing, or stopping, and then resuming...

Written by an anonymous author, who also provided these screenshots:



# Summary

Psi+ is **the only client that is able to hide its identity** - all other clients can be exposed by Psi+ and Gajim even after attempted mitigations. If you care about client concealement, then you have to use Psi+. Also, when I refer to the resource header, that can **only be seen by MUC admins**. The most important leak is the timezone, and it can be mitigated in all clients except Pidgin - so don't use it. **All Android clients leak timezone unless used through TOR**; they also reveal their versions. Profanity can hide everything except its name and version. ChatSecure leaks only the client name, but requires an iPhone to use it. Dino and Gajim (after mitigation) leak only the client name. I did not review any clients that don't support OMEMO, as that is the encryption that is expected today. Even though OTR is a good replacement, and isn't weaker in any relevant way - I can't justify recommending clients that don't support OMEMO, since they won't be able to have encrypted communication with popular clients that have now dropped OTR (e.g nuGajim). **All clients can leak your country through your status messages**, if they are in any other language than English. So, use English as your system language or turn off the messages. XMPP servers will always store your roster - the only way to avoid this is to not add anyone to the roster.

| Client / Feature | Psi+ | Pidgin | Conversations | Dino | Profanity | Gajim | ChatSecure |
|---|---|---|---|---|---|---|---|
| OMEMO | Yes | Partial | Yes | Yes | Partial | Yes | Yes |
| OTR | Yes | Yes | No | No | Yes | Yes (old versions) | Yes |
| | | 3rd party, | | | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| PGP | Yes | didn't test | Yes | Yes | Yes | Yes | No |
| Mitigations for version | Yes | No | No | Yes | Yes | Yes | Yes |
| Mitigations for timezone | Yes | No | External (TOR) | Yes | Yes | Yes | Yes |
| Mitigations for OS | Yes | Yes | No | Yes | Yes | Yes | Yes |
| Mitigations for client name | Yes | No | No | No | Yes | Partial | Partial |
| Onion connectivity | Yes | No | Yes | No | Yes | Yes | No |
| OS support (official) | Linux, Windows, macOS, Haiku, BSD | Linux, Windows, FreeBSD | Android | Linux, BSD | Linux, Windows, BSD, OSX, Android | Linux, Windows, FreeBSD | iOS |

Back to the front page