GNU Taler Merchant Manual

Release 0.6.0pre1

GNU Taler team

Sep 19, 2019

CONTENTS

CHAPTER

ONE

INTRODUCTION

1.1 About GNU Taler

GNU Taler is an open protocol for an electronic payment system with a free software reference implementation. GNU Taler offers secure, fast and easy payment processing using well understood cryptographic techniques. GNU Taler allows customers to remain anonymous, while ensuring that merchants can be held accountable by governments. Hence, GNU Taler is compatible with anti-money-laundering (AML) and know-your-customer (KYC) regulation, as well as data protection regulation (such as GDPR).

GNU Taler is not yet production-ready, after following this manual you will have a backend that can process payments in "KUDOS", but not regular currencies. This is not so much because of limitations in the backend, but because we are not aware of a Taler exchange operator offering regular currencies today.

1.2 About this manual

This tutorial targets system administrators who want to install a GNU Taler merchant *backend*.

We expect some moderate familiarity with the compilation and installation of free software packages. An understanding of cryptography is not required.

This first chapter of the tutorial will give a brief overview of the overall Taler architecture, describing the environment in which the Taler backend operates. The second chapter then explains how to install the software, including key dependencies. The third chapter will explain how to configure the backend, including in particular the configuration of the bank account details of the merchant.

The last chapter gives some additional information about advanced topics which will be useful for system administrators but are not necessary for operating a basic backend.

1.3 Architecture overview

crypto-currency KUDOS Taler is a pure payment system, not a new crypto-currency. As such, it operates in a traditional banking context. In particular, this means that in order to receive funds via Taler, the merchant must have a regular bank account, and payments can be executed in ordinary currencies such as USD or EUR. For testing purposes, Taler uses a special currency "KUDOS" and includes its own special bank.

The Taler software stack for a merchant consists of four main components:

- frontend A frontend which interacts with the customer's browser. The frontend enables the customer to build a shopping cart and place an order. Upon payment, it triggers the respective business logic to satisfy the order. This component is not included with Taler, but rather assumed to exist at the merchant. This manual describes how to integrate Taler with Web shop frontends.

- back office A back office application that enables the shop operators to view customer orders, match them to financial transfers, and possibly approve refunds if an order cannot be satisfied. This component is again not included with Taler, but rather assumed to exist at the merchant. This manual will describe how to integrate such a component to handle payments managed by Taler.
- backend A Taler-specific payment backend which makes it easy for the frontend to process financial transactions with Taler. The next two chapters will describe how to install and configure this backend.
- DBMS Postgres A DBMS which stores the transaction history for the Taler backend. For now, the GNU Taler reference implemenation only supports Postgres, but the code could be easily extended to support another DBMS.

The following image illustrates the various interactions of these key components:

```
Missing diagram image
```

RESTful Basically, the backend provides the cryptographic protocol support, stores Taler-specific financial information in a DBMS and communicates with the GNU Taler exchange over the Internet. The frontend accesses the backend via a RESTful API. As a result, the frontend never has to directly communicate with the exchange, and also does not deal with sensitive data. In particular, the merchant's signing keys and bank account information is encapsulated within the Taler backend.

CHAPTER

TWO

INSTALLATION

This chapter describes how to install the GNU Taler merchant backend.

2.1 Installing Taler using Docker

This section provides instructions for the merchant backend installation using 'Docker'.

For security reasons, we run Docker against a VirtualBox instance, so the `docker` command should connect to a `docker-machine` instance that uses the VirtualBox driver.

Therefore, the needed tools are: "docker", "docker-machine", and "docker-compose". Please refer to Docker's official[1] documentation in order to get those components installed, as that is not in this manual's scope.

Before starting to build the merchant's image, make sure a "docker-machine" instance is up and running.

Because all of the Docker source file are kept in our "deployment" repository, we start by checking out the `git://taler.net/deployment` codebase:

```
$ git clone git://taler.net/deployment
```

Now we actually build the merchant's image. From the same directory as above:

```
$ cd deployment/docker/merchant/
$ docker-compose build
```

If everything worked as expected, the merchant is ready to be launched. From the same directory as the previous step:

```
# Recall: the docker-machine should be up and running.
$ docker-compose up
```

You should see some live logging from all the involved containers. At this stage of development, you should also ignore some (harmless) error message from postresql about already existing roles and databases.

To test if everything worked as expected, it suffices to issue a simple request to the merchant, as:

```
$ curl http://$(docker-machine ip)/
# A greeting message should be returned by the merchant.
```

[1] https://docs.docker.com/

2.2 Generic instructions

This section provides generic instructions for the merchant backend installation independent of any particular operating system. Operating system specific instructions are provided in the following sections. You should follow the operating system specific instructions if those are available, and only consult the generic instructions if no system-specific instructions are provided for your specific operating system.

2.2.1 Installation of dependencies

The following packages need to be installed before we can compile the backend:

- autoconf >= 2.69
- automake >= 1.14
- libtool >= 2.4
- autopoint >= 0.19
- libltdl >= 2.4
- libunistring >= 0.9.3
- libcurl >= 7.26 (or libgnurl >= 7.26)
- GNU libmicrohttpd >= 0.9.39
- GNU libgcrypt >= 1.6
- libjansson >= 2.7
- Postgres >= 9.4, including libpq
- libgnunetutil (from Git)
- GNU Taler exchange (from Git)

Except for the last two, these are available in most GNU/Linux distributions and should just be installed using the respective package manager.

The following sections will provide detailed instructions for installing the libgnunetutil and GNU Taler exchange dependencies.

2.2.2 Installing libgnunetutil

GNUnet Before you install libgnunetutil, you must download and install the dependencies mentioned in the previous section, otherwise the build may succeed but fail to export some of the tooling required by Taler.

To download and install libgnunetutil, proceed as follows:

```
$ git clone https://gnunet.org/git/gnunet/
$ cd gnunet/
$ ./bootstrap
$ ./configure [--prefix=GNUNETPFX]
$ # Each dependency can be fetched from non standard locations via
$ # the '--with-<LIBNAME>' option. See './configure --help'.
$ make
# make install
```

If you did not specify a prefix, GNUnet will install to `/usr/local`, which requires you to run the last step as `root`.

2.2.3 Installing the GNU Taler exchange

exchange After installing GNUnet, you can download and install the exchange as follows:

```
$ git clone git://taler.net/exchange
$ cd exchange
$ ./bootstrap
$ ./configure [--prefix=EXCHANGEPFX] \
              [--with-gnunet=GNUNETPFX]
$ # Each dependency can be fetched from non standard locations via
$ # the '--with-<LIBNAME>' option. See './configure --help'.
$ make
# make install
```

If you did not specify a prefix, the exchange will install to `/usr/local`, which requires you to run the last step as `root`. Note that you have to specify `--with-gnunet=/usr/local` if you installed GNUnet to `/usr/local` in the previous step.

2.2.4 Installing the GNU Taler merchant backend

backend The following steps assume all dependencies are installed.

Use the following commands to download and install the merchant backend:

```
$ git clone git://taler.net/merchant
$ cd merchant
$ ./bootstrap
$ ./configure [--prefix=PFX] \
              [--with-gnunet=GNUNETPFX] \
              [--with-exchange=EXCHANGEPFX]
$ # Each dependency can be fetched from non standard locations via
$ # the '--with-<LIBNAME>' option. See './configure --help'.
$ make
$ make install
```

Note that you have to specify `--with-exchange=/usr/local` and/or `--with-exchange=/usr/local` if you installed the exchange and/or GNUnet to `/usr/local` in the previous steps.

2.3 Installing Taler on Debian GNU/Linux

Wheezy Debian Debian wheezy is too old and lacks most of the packages required.

On Debian jessie, only GNU libmicrohttpd needs to be compiled from source. To install dependencies on Debian jesse, run the following commands:

```
# apt-get install \
  autoconf \
  automake \
  autopoint \
  libtool \
  libltdl-dev \
  libunistring-dev \
  libcurl4-gnutls-dev \
  libgcrypt20-dev \
  libjansson-dev \
```

(continues on next page)

(continued from previous page)

```
  libpq-dev \
  postgresql-9.4
# wget https://ftp.gnu.org/gnu/libmicrohttpd/libmicrohttpd-latest.tar.gz
# wget https://ftp.gnu.org/gnu/libmicrohttpd/libmicrohttpd-latest.tar.gz.sig
# gpg -v libmicrohttpd-latest.tar.gz # Should show signed by 939E6BE1E29FC3CC
# tar xf libmicrohttpd-latest.tar.gz
# cd libmicrohttpd-0*
# ./configure
# make install
```

For more recent versions of Debian, you should instead run:

```
# apt-get install \
  autoconf \
  automake \
  autopoint \
  libtool \
  libltdl-dev \
  libunistring-dev \
  libcurl4-gnutls-dev \
  libgcrypt20-dev \
  libjansson-dev \
  libpq-dev \
  postgresql-9.5 \
  libmicrohttpd-dev
```

For the rest of the installation, follow the generic installation instructions starting with the installation of libgnunetutil. Note that if you used the Debian wheezy instructions above, you need to pass `--with-microhttpd=/usr/local/` to all `configure` invocations.

CHAPTER

THREE

HOW TO CONFIGURE THE MERCHANT'S BACKEND

taler-config taler.conf The installation already provides reasonable defaults for most of the configuration options. However, some must be provided, in particular the database account and bank account that the backend should use. By default, the file `$HOME/.config/taler.conf` is where the Web shop administrator specifies configuration values that augment or override the defaults. The format of the configuration file is the well-known INI file format. You can edit the file by hand, or use the `taler-config` commands given as examples. For more information on `taler-config`, see *Using taler-config*.

3.1 Backend options

The following table describes the options that commonly need to be modified. Here, the notation `[$section]/$option` denotes the option `$option` under the section `[$section]` in the configuration file.

Service address The following option sets the transport layer address used by the merchant backend:

UNIX domain socket TCP

```
[MERCHANT]/SERVE = TCP | UNIX
```

If given,

- `TCP`, then we need to set the TCP port in `[MERCHANT]/PORT`
- `UNIX`, then we need to set the unix domain socket path and mode in `[MERCHANT]/UNIXPATH` and `[MERCHANT]/UNIXPATH_MODE`. The latter takes the usual permission mask given as a number, e.g. 660 for user/group read-write access.

The frontend can then connect to the backend over HTTP using the specified address. If frontend and backend run within the same operating system, the use of a UNIX domain socket is recommended to avoid accidentally exposing the backend to the network.

port To run the Taler backend on TCP port 8888, use:

```
$ taler-config -s MERCHANT -o SERVE -V TCP
$ taler-config -s MERCHANT -o PORT -V 8888
```

Currency Which currency the Web shop deals in, i.e. "EUR" or "USD", is specified using the option

currency KUDOS

```
[TALER]/CURRENCY
```

For testing purposes, the currency MUST match "KUDOS" so that tests will work with the Taler demonstration exchange at https://exchange.demo.taler.net/:

```
$ taler-config -s TALER -o CURRENCY -V KUDOS
```

Database DBMS In principle is possible for the backend to support different DBMSs. The option

```
[MERCHANT]/DB
```

specifies which DBMS is to be used. However, currently only the value "postgres" is supported. This is also the default.

In addition to selecting the DBMS software, the backend requires DBMS-specific options to access the database.

For postgres, you need to provide:

```
[merchantdb-postgres]/config
```

Postgres This option specifies a postgres access path using the format `postgres:///$DBNAME`, where `$DBNAME` is the name of the Postgres database you want to use. Suppose `$USER` is the name of the user who will run the backend process. Then, you need to first run

```
$ sudu -u postgres createuser -d $USER
```

as the Postgres database administrator (usually `postgres`) to grant `$USER` the ability to create new databases. Next, you should as `$USER` run:

```
$ createdb $DBNAME
```

to create the backend's database. Here, `$DBNAME` must match the database name given in the configuration file.

To configure the Taler backend to use this database, run:

```
$ taler-config -s MERCHANTDB-postgres -o CONFIG \
  -V postgres:///$DBNAME
```

Exchange exchange To add an exchange to the list of trusted payment service providers, you create a section with a name that starts with "exchange-". In that section, the following options need to be configured:

- The "url" option specifies the exchange's base URL. For example, to use the Taler demonstrator use:

```
$ taler-config -s EXCHANGE-demo -o URL \
  -V https://exchange.demo.taler.net/
```

- master key The "master_key" option specifies the exchange's master public key in base32 encoding. For the Taler demonstrator, use:

```
$ taler-config -s EXCHANGE-demo -o master_key \
  -V CQQZ9DY3MZ1ARMN5K1VKDETS04Y2QCKMMCFHZSWJWWVN82BTTH00
```

Note that multiple exchanges can be added to the system by using different tokens in place of `demo` in the example above. Note that all of the exchanges must use the same currency. If you need to support multiple currencies, you need to configure a backend per currency.

Instances instance The backend allows the user to run multiple instances of shops with distinct business entities against a single backend. Each instance uses its own bank accounts and key for signing contracts. It is mandatory to configure a "default" instance.

- The "KEYFILE" option specifies the file containing the instance's private signing key. For example, use:

```
$ taler-config -s INSTANCE-default -o KEYFILE \
  -V '${TALER_CONFIG_HOME}/merchant/instace/default.key'
```

- The "NAME" option specifies a human-readable name for the instance. For example, use:

```
$ taler-config -s INSTANCE-default -o NAME \
  -V 'Kudos Inc.'
```

- The optional "TIP_EXCHANGE" and "TIP_EXCHANGE_PRIV_FILENAME" options are discussed in Tipping visitors

Accounts wire format In order to receive payments, the merchant backend needs to communicate bank account details to the exchange. For this, the configuration must include one or more sections named "ACCOUNT-name" where `name` can be replaced by some human-readable word identifying the account. For each section, the following options should be provided:

- The "URL" option specifies a `payto://`-URL for the account of the merchant. For example, use:

```
$ taler-config -s ACCOUNT-bank -o NAME \
  -V 'payto://x-taler-bank/bank.demo.taler.net/4'
```

- The "WIRE_RESPONSE" option specifies where Taler should store the (salted) JSON encoding of the wire account. The file given will be created if it does not exist. For example, use:

```
$ taler-config -s ACCOUNT-bank -o WIRE_RESPONSE \
  -V '{$TALER_CONFIG_HOME}/merchant/bank.json'
```

- The "PLUGIN" option specifies which wire plugin should be used for this account. The plugin must support the wire method used by the URL. For example, use:

```
$ taler-config -s ACCOUNT-bank -o PLUGIN \
  -V taler_bank
```

- For each `instance` that should use this account, you should set `HONOR_instance` and `ACTIVE_instance` to YES. The first option will cause the instance to accept payments to the account (for existing contracts), while the second will cause the backend to include the account as a possible option for new contracts.

  For example, use:

```
$ taler-config -s ACCOUNT-bank -o HONOR_default \
  -V YES
$ taler-config -s ACCOUNT-bank -o ACTIVE_default \
  -V YES
```

  to use "account-bank" for the "default" instance.

Depending on which PLUGIN you configured, you may additionally specfiy authentication options to enable the plugin to use the account.

For example, with `taler_bank` plugin, use:

```
$ taler-config -s ACCOUNT-bank -o TALER_BANK_AUTH_METHOD \
  -V basic
$ taler-config -s ACCOUNT-bank -o USERNAME \
  -V user42
$ taler-config -s ACCOUNT-bank -o PASSWORD \
  -V pass42
```

Note that additional instances can be specified using different tokens in the section name instead of `default`.

3.2 Sample backend configuration

configuration The following is an example for a complete backend configuration:

```
[TALER]
CURRENCY = KUDOS

[MERCHANT]
SERVE = TCP
PORT = 8888
DATABASE = postgres

[MERCHANTDB-postgres]
CONFIG = postgres:///donations

[INSTANCE-default]
KEYFILE = $DATADIR/key.priv
NAME = "Kudos Inc."

[ACCOUNT-bank]
URL = payto://x-taler-bank/bank.demo.taler.net/4
WIRE_RESPONSE = $DATADIR/bank.json
PLUGIN = taler_bank
HONOR_default = YES
ACTIVE_default = YES
TALER_BANK_AUTH_METHOD = basic
USERNAME = my_user
PASSWORD = 1234pass

[EXCHANGE-trusted]
URL = https://exchange.demo.taler.net/
MASTER_KEY = CQQZ9DY3MZ1ARMN5K1VKDETS04Y2QCKMMCFHZSWJWWVN82BTTH00
CURRENCY = KUDOS
```

Given the above configuration, the backend will use a database named `donations` within Postgres.

The backend will deposit the coins it receives to the exchange at https://exchange.demo.taler.net/, which has the master key "CQQZ9DY3MZ1ARMN5K1VKDETS04Y2QCKMMCFHZSWJWWVN82BTTH00".

Please note that `doc/config.sh` will walk you through all configuration steps, showing how to invoke `taler-config` for each of them.

3.3 Launching the backend

backend taler-merchant-httpd Assuming you have configured everything correctly, you can launch the merchant backend using:

```
$ taler-merchant-httpd
```

When launched for the first time, this command will print a message about generating your private key. If everything worked as expected, the command

```
$ curl http://localhost:8888/
```

should return the message

```
Hello, I'm a merchant's Taler backend. This HTTP server is not for humans.
```

Please note that your backend is right now likely globally reachable. Production systems should be configured to bind to a UNIX domain socket or properly restrict access to the port.

CHAPTER

FOUR

TESTING

The tool `taler-merchant-generate-payments` can be used to test the merchant backend installation. It implements all the payment's steps in a programmatically way, relying on the backend you give it as input. Note that this tool gets installed along all the merchant backend's binaries.

This tool gets configured by a config file, that must have the following layout:

```
[PAYMENTS-GENERATOR]

# The exchange used during the test: make sure the merchant backend
# being tested accpets this exchange.
# If the sysadmin wants, she can also install a local exchange
# and test against it.
EXCHANGE = https://exchange.demo.taler.net/

# This value must indicate some URL where the backend
# to be tested is listening; it doesn't have to be the
# "official" one, though.
MERCHANT = http://localbackend/

# This value is used when the tool tries to withdraw coins,
# and must match the bank used by the exchange. If the test is
# done against the exchange at https://exchange.demo.taler.net/,
# then this value can be "https://bank.demo.taler.net/".
BANK = https://bank.demo.taler.net/

# The merchant instance in charge of serving the payment.
# Make sure this instance has a bank account at the same bank
# indicated by the 'bank' option above.
INSTANCE = default

# The currency used during the test. Must match the one used
# by merchant backend and exchange.
CURRENCY = KUDOS
```

Run the test in the following way:

```
$ taler-merchant-generate-payments [-c config] [-e EURL] [-m MURL]
```

The argument `config` given to `-c` points to the configuration file and is optional – `~/.config/taler.conf` will be checked by default. By default, the tool forks two processes: one for the merchant backend, and one for the exchange. The option `-e` (`-m`) avoids any exchange (merchant backend) fork, and just runs the generator against the exchange (merchant backend) running at `EURL` (`MURL`).

Please NOTE that the generator contains *hardcoded* values, as for deposit fees of the coins it uses. In order to work against the used exchange, those values MUST match the ones used by the exchange.

The following example shows how the generator "sets" a deposit fee of EUR:0.01 for the 5 EURO coin.

```
// from <merchant_repository>/src/sample/generate_payments.c
{ .oc = OC_PAY,
  .label = "deposit-simple",
  .expected_response_code = MHD_HTTP_OK,
  .details.pay.contract_ref = "create-proposal-1",
  .details.pay.coin_ref = "withdraw-coin-1",
  .details.pay.amount_with_fee = concat_amount (currency, "5"),
  .details.pay.amount_without_fee = concat_amount (currency, "4.99") },
```

The logic calculates the deposit fee according to the subtraction: `amount_with_fee - amount_without_fee`.

The following example shows a 5 EURO coin configuration - needed by the used exchange - which is compatible with the hardcoded example above.

```
[COIN_eur_5]
value = EUR:5
duration_overlap = 5 minutes
duration_withdraw = 7 days
duration_spend = 2 years
duration_legal = 3 years
fee_withdraw = EUR:0.00
fee_deposit = EUR:0.01 # important bit
fee_refresh = EUR:0.00
fee_refund = EUR:0.00
rsa_keysize = 1024
```

If the command terminates with no errors, then the merchant backend is correctly installed.

After this operation is done, the merchant database will have some dummy data in it, so it may be convenient to clean all the tables; to this purpose, issue the following command:

```
$ taler-merchant-dbinit -r
```

CHAPTER

FIVE

ADVANCED TOPICS

5.1 Configuration format

configuration In Taler realm, any component obeys to the same pattern to get configuration values. According to this pattern, once the component has been installed, the installation deploys default values in ${prefix}/share/taler/config.d/, in .conf files. In order to override these defaults, the user can write a custom .conf file and either pass it to the component at execution time, or name it taler.conf and place it under $HOME/.config/.

A config file is a text file containing sections, and each section contains its values. The right format follows:

```
[section1]
value1 = string
value2 = 23

[section2]
value21 = string
value22 = /path22
```

Throughout any configuration file, it is possible to use `$`-prefixed variables, like `$VAR`, especially when they represent filesystem paths. It is also possible to provide defaults values for those variables that are unset, by using the following syntax: `${VAR:-default}`. However, there are two ways a user can set `$`-prefixable variables:

by defining them under a `[paths]` section, see example below,

```
[paths]
TALER_DEPLOYMENT_SHARED = ${HOME}/shared-data
..
[section-x]
path-x = ${TALER_DEPLOYMENT_SHARED}/x
```

or by setting them in the environment:

```
$ export VAR=/x
```

The configuration loader will give precedence to variables set under `[path]`, though.

The utility `taler-config`, which gets installed along with the exchange, serves to get and set configuration values without directly editing the .conf. The option `-f` is particularly useful to resolve pathnames, when they use several levels of `$`-expanded variables. See `taler-config --help`.

Note that, in this stage of development, the file `$HOME/.config/taler.conf` can contain sections for *all* the component. For example, both an exchange and a bank can read values from it.

The repository `git://taler.net/deployment` contains examples of configuration file used in our demos. See under `deployment/config`.

Note

Expectably, some components will not work just by using default values, as their work is often interdependent. For example, a merchant needs to know an exchange URL, or a database name.

5.2 Using taler-config

taler-config The tool `taler-config` can be used to extract or manipulate configuration values; however, the configuration use the well-known INI file format and can also be edited by hand.

Run

```
$ taler-config -s $SECTION
```

to list all of the configuration values in section `$SECTION`.

Run

```
$ taler-config -s $section -o $option
```

to extract the respective configuration value for option `$option` in section `$section`.

Finally, to change a setting, run

```
$ taler-config -s $section -o $option -V $value
```

to set the respective configuration value to `$value`. Note that you have to manually restart the Taler backend after you change the configuration to make the new configuration go into effect.

Some default options will use $-variables, such as `$DATADIR` within their value. To expand the `$DATADIR` or other $-variables in the configuration, pass the `-f` option to `taler-config`. For example, compare:

```
$ taler-config -s ACCOUNT-bank \
               -o WIRE_RESPONSE
$ taler-config -f -s ACCOUNT-bank \
               -o WIRE_RESPONSE
```

While the configuration file is typically located at `$HOME/.config/taler.conf`, an alternative location can be specified to `taler-merchant-httpd` and `taler-config` using the `-c` option.

5.3 Merchant key management

merchant key KEYFILE The option "KEYFILE" in the section "INSTANCE-default" specifies the path to the instance's private key. You do not need to create a key manually, the backend will generate it automatically if it is missing. While generally unnecessary, it is possible to display the corresponding public key using the `gnunet-ecc` command-line tool:

```
$ gnunet-ecc -p                                    \
  $(taler-config -f -s INSTANCE-default \
                 -o KEYFILE)
```

5.4 Using the SEPA wire transfer method

SEPA EBICS The following is a sample configuration for the SEPA wire transfer method:[2].

Then, to configure the EBICS backend for SEPA payments in EUR, the following configuration options need to be set:

```
$ taler-config -s TALER -o CURRENCY -V EUR
$ taler-config -s ACCOUNT-e -o PLUGIN -V ebics
$ taler-config -s ACCOUNT-e -o URL \
 -V payto://sepa/XY00111122223333444455556666
$ taler-config -s ACCOUNT-e -o WIRE_RESPONSE
 -V '${DATADIR}/b.json'
```

Please note that you will also have to configure an exchange and/or auditors that support SEPA. However, we cannot explain how to do this yet as such entities do not yet exist. Once such entities do exist, we expect future versions of the Taler backend to ship with pre-configured exchanges and auditors for common denominations.

5.5 Tipping visitors

tipping Taler can also be used to tip Web site visitors. For example, you may be running an online survey, and you want to reward those people that have dutifully completed the survey. If they have installed a Taler wallet, you can provide them with a tip for their deeds. This section describes how to setup the Taler merchant backend for tipping.

There are four basic steps that must happen to tip a visitor.

5.5.1 Configure a reserve and exchange for tipping

gnunet-ecc reserve key To tip users, you first need to create a reserve. A reserve is a pool of money held in escrow at the Taler exchange. This is the source of the funds for the tips. Tipping will fail (resulting in disappointed visitors) if you do not have enough funds in your reserve!

First, we configure the backend. You need to enable tipping for each instance separately, or you can use an instance only for tipping. To configure the "default" instance for tipping, use the following configuration:

```
[INSTANCE-default]
# this is NOT the tip.priv
KEYFILE = signing_key.priv
# replace the URL with the URL of the exchange you will use
TIP_EXCHANGE = https://exchange:443/
# here put the path to the file created with "gnunet-ecc -g1 tip.priv"
TIP_RESERVE_PRIV_FILENAME = tip.priv
```

Note that the KEYFILE option should have already been present for the instance. It has nothing to do with the "tip.priv" file we created above, and you should probably use a different file here.

Instead of manually editing the configuration, you could also run:

```
$ taler-config -s INSTANCE-default \
    -o TIP_RESERVE_PRIV_FILENAME \
    -V tip.priv
$ taler-config -s INSTANCE-default \
    -o TIP_EXCHANGE \
    -V https://exchange:443/
```

[2] Supporting SEPA is still work in progress; the backend will accept this configuration, but the exchange will not work with SEPA today.

Next, to create the TIP_RESERVE_PRIV_FILENAME file, use:

```
$ gnunet-ecc -g 1   \
  $(taler-config -f -s INSTANCE-default \
      -o TIP-RESERVE_PRIV_FILENAME)
```

This will create a file with the private key that will be used to identify the reserve. You need to do this once for each instance that is configured to tip.

Now you can (re)start the backend with the new configuration.

5.5.2 Fund the reserve

reserve close To fund the reserve, you must first extract the public key from "tip.priv":

```
$ gnunet-ecc --print-public-key \
  $(taler-config -f -s INSTANCE-default \
      -o TIP-RESERVE_PRIV_FILENAME)
```

In our example, the output for the public key is:

```
QPE24X8PBX3BZ6E7GQ5VAVHV32FWTTCADR0TRQ183MSSJD2CHNEG
```

You now need to make a wire transfer to the exchange's bank account using the public key as the wire transfer subject. The exchange's bank account details can be found in JSON format at "https://exchange:443//wire/METHOD" where METHOD is the respective wire method (i.e. "sepa"). Depending on the exchange's operator, you may also be able to find the bank details in a human-readable format on the main page of the exchange.

Make your wire transfer and (optionally) check at "https://exchange:443/reserve/status/reserve_pub=QPE24X..." whether your transfer has arrived at the exchange.

Once the funds have arrived, you can start to use the reserve for tipping.

Note that an exchange will typically close a reserve after four weeks, wiring all remaining funds back to the sender's account. Thus, you should plan to wire funds corresponding to a campaign of about two weeks to the exchange initially. If your campaign runs longer, you should wire further funds to the reserve every other week to prevent it from expiring.

5.5.3 Authorize a tip

When your frontend has reached the point where a client is supposed to receive a tip, it needs to first authorize the tip. For this, the frontend must use the "/tip-authorize" API of the backend. To authorize a tip, the frontend has to provide the following information in the body of the POST request:

- The amount of the tip
- The justification (only used internally for the back-office)
- The URL where the wallet should navigate next after the tip was processed
- The tip-pickup URL (see next section)

In response to this request, the backend will return a tip token, an expiration time and the exchange URL. The expiration time will indicate how long the tip is valid (when the reserve expires). The tip token is an opaque string that contains all the information needed by the wallet to process the tip. The frontend must send this tip token to the browser in a special "402 Payment Required" response inside the X-Taler-Tip header.

The frontend should handle errors returned by the backend, such as missconfigured instances or a lack of remaining funds for tipping.

5.5.4 Picking up of the tip

The wallet will POST a JSON object to the shop's "/tip-pickup" handler. The frontend must then forward this request to the backend. The response generated by the backend can then be forwarded directly to the wallet.

5.6 Generate payments

testing database The merchant codebase offers the `taler-merchant-benchmark` tool to populate the database with fake payments. This tool is in charge of starting a merchant, exchange, and bank processes, and provide them all the input to accomplish payments. Note that each component will use its own configuration (as they would do in production).

The tool takes all of the values it needs from the command line, with some of them being mandatory. Among those, we have:

- `--currency=K` Use currency *K*, for example to craft coins to withdraw.
- `--bank-url=URL` Assume that the bank is serving under the base URL *URL*. This option is only actually used by the tool to check if the bank was well launched.
- `--merchant-url=URL` Reach the merchant through *URL*, for downloading contracts and sending payments.

The tool then comes with two operation modes: *ordinary*, and *corner*. The first just executes normal payments, meaning that it uses the default instance and make sure that all payments get aggregated. The second gives the chance to leave some payments unaggregated, and also to use merchant instances other than the default (which is, actually, the one used by default by the tool).

Note: the abilty of driving the aggregation policy is useful for testing the backoffice facility.

Any subcommand is also equipped with the canonical `--help` option, so feel free to issue the following command in order to explore all the possibilities. For example:

```
$ taler-merchant-benchmark corner --help
```

will show all the options offered by the *corner* mode. Among the most interesting, there are:

- `--two-coins=TC` This option instructs the tool to perform *TC* many payments that use two coins, because normally only one coin is spent per payment.
- `--unaggregated-number=UN` This option instructs the tool to perform *UN* (one coin) payments that will be left unaggregated.
- `--alt-instance=AI` This option instructs the tool to perform payments using the merchant instance *AI* (instead of the *default* instance)

As for the `ordinary` subcommand, it is worth explaining the following options:

- `--payments-number=PN` Instructs the tool to perform *PN* payments.
- `--tracks-number=TN` Instructs the tool to perform *TN* tracking operations. Note that the **total** amount of operations will be two times *TN*, since "one" tracking operation accounts for `/track/transaction` and `/track/transfer`. This command should only be used to see if the operation ends without problems, as no actual measurement of performance is provided (despite of the 'benchmark' work used in the tool's name).

APPENDIX

A

GNU FREE DOCUMENTATION LICENSE

Version 1.3, 3 November 2008

A.1 0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

A.2 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

A.3 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

A.4 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

A.5 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various partiesâ€"for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

A.6 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

A.7 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

A.8 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

A.9 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

A.10 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

A.11 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See https://www.gnu.org/licenses/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

A.12 11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

A.12.1 ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  YEAR  YOUR NAME.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
A copy of the license is included in the section entitled "GNU
Free Documentation License".
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with â€¦ Texts." line with this:

```
with the Invariant Sections being LIST THEIR TITLES, with the
Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.