Back-office Web service manual.

Version 0.0.0
16 Jan 2018

Marcello Stanisci (marcello@taler.net)
Christian Grothoff (grothoff@taler.net)

Table of Contents

1 Introduction

The back-office Web service allows a merchant to check on the status of their Taler transactions. Merchants can check if and when they were paid by a wire transfer for coins deposited at the exchange. Also, given a wire transfer they have received, they can ask which Taler transactions correspond to the wire transfer.

This manual guides merchant system administrators through the installation and configuration of this Web service.

2 Installation

The back-office Web service code is available at `git://taler.net/backoffice`. The application can be installed in a GNU-ish fashion.

```
# Get the code:
$ git clone git://taler.net/backoffice

# Bootstrap and configure
$ cd backoffice
$ ./bootstrap
# This step will check if the system is ready to
# allow the installation.
$ ./configure --prefix=<PREFIX>
$ make install
```

Note that to make the application work `<PREFIX>/bin` must be included in the `$PATH`, and `<PREFIX>/lib/python3.5/site-packages/` in `$PYTHONPATH`.

3 Configuration

The following information must be provided in the configuration: on which address the backend should serve the Web site, which currency is used, and which merchant backend this Web service will work with.

The merchant backend is an important agent, as it is responsible for returning information about transactions and wire transfers. The backoffice Web service is merely a frontend for it. A separate manual (available at `https://docs.taler.net/merchant/backend/html/manual.html`) describes the installation and configuration of the merchant backend.

Assuming the reader is familiar with configuration in Taler (if not, read: `https://docs.taler.net/exchange/html/taler-exchange.html#Configuration-format`), a working configuration example is the following one:

```
[taler]
# will be EUR, USD, or whatever currency the merchant
# works with.
currency = KUDOS

# each back-office Web service is associated with a "frontend
# name": this name instructs the application which configuration
# section is going to be read.  Thus <name> is the "frontend name"
# and must be specified on the command line via the "--frontend"
# option.  See the 'Run' chapter for more details on launching the
# application.
[backoffice-<name>]

# This option sets the way the backoffice communicates
# when it is instructed to operate via UWSGI.  Therefore,
# <how> can be: TCP or UNIX.  If TCP is given, then the
# additional UWSGI_PORT option becomes necessary.
uwsgi_serve = <how>

# those options will be read if the Web site is served via
# WSGI.
uwsgi_unixpath_mode = 660
uwsgi_unixpath = /path/to/backoffice-<name>.uwsgi
uwsgi_unixmode = 666

# If UWSGI_SERVE were set as 'TCP', then the following option
# would have been necessary.  It instructs the backoffice service
# about which TCP port should be listened on, to communicate over
# UWSGI.
# uwsgi_port = 8080

# this option will be read if the Web site is served via
# HTTP.
```

```
http_port = 5959

# specifies which merchant backend is going to be used.
backend = http://backend.test.taler.net/

# Informally speaking, each instance points to both a private
# key that can sign proposals and a bank account that can receive
# wire transfers by some exchange.

# Here, <instance_i> is just a string (with no spaces) that will
# make the referenced instance be traceable by the back-office Web
# application.

instances = <instance_1> <instance_2> ..
```

4 Launching the backoffice

The following example shows how to run the Web service.

```
# such invocation will work only if the configuration contains
# a section called "[backoffice-myshop]" which looks like the
# example above.

# As of serving, the Web site will be available via HTTP, at the
# port specified in the configuration option "http_port", at localhost.

$ taler-merchant-backoffice --frontend myshop serve-http
```

Other options, such as those to serve the site via WSGI, are explained in the man page and can be listed using the `--help` argument.