Bern University
of Applied Sciences

DONAU

Tax-deductable Donations for GNU Taler

Course of study	Bachelor of Science in Computer Science
Author	Johannes Casaburi and Lukas Matyja
Advisor	Prof. Dr. Christian Grothoff
Co-advisor	Prof. Dr. Emmanuel Benoist

Version 1.0 of April 24, 2024

- Technik und Informatik
- Mikro- und Medizintechnik

Abstract

This bachelor thesis describes and implements a theoretical contcept of a donation authority system. The donation authority or in short Donau is privacy friendly and free software. It examines the usability by asking a tax authority about their current donation system and obtaining their opinion on the Donau project.

Donau is a GNU Taler project. It depends on the code of the GNU Taler environment, but is completely independent of the Taler payment system.

The Donau environment includes three stakeholder. Donors, charities and of course the tax authority. The centerpiece, the Donau, would be operated by the tax authority itself. The Donau issues donation receipts for the Donor via charity and validates the receipts from the donor. Issuance by the authority prevents possible donation receipt forgeries. The automation of the donation receipt validation process would also allow the donor's various donation receipts to be combined into one. Donation data is often sensitive data. Therefore, in order to protect the donor, the system is designed to collect and store as little data as possible and to anonymize the data where possible.

results...

Contents

1 Introduction

1.1 Motivation

Donations can often be deducted from taxes. To do so, donors must submit donation receipts to the tax authorities. Nowadays, donation receipts can only be checked with disproportionate effort. This is the reason why receipts for small amounts are often not checked at all. Furthermore, donations are sensitive data that should be anonymized as much as possible. Even towards the tax authorities. For example, a donation to an AIDS aid organization can lead to conclusions that could be unpleasant for the donor.

1.2 Goals

The project description is as follows: The goal of this project is to add anonymous donation receipts to Taler. Someone donating to a charitable organization may wish to do so anonymously, but still want to deduce that amount of their tax. This can be done by reusing cryptography present in Taler. A great part of the project will be specifying the details, as well as implementing the Donau (Donation authority). Furthermore, the Taler merchant part and the wallet will have to be developed/adapted. Optionally, a small Android donation verification app will also be provided.

2 Protocol

2.1 Notation & Definitions

2.1.1 Notation

- $\langle a, b, ... \rangle$: Pair/tuple

2.1.2 Definitions

- **Cryptographic Hash Function** $h := H(m)$ where m is a message and h the resulting hash.
- **BlindKeygen** $\langle K_x^{pub}, K_x^{priv} \rangle := Keygen^B(\omega)$ where ω is a source of entropy and x is the associated value (e.g. 2 EUR). The resulting key pair represents a donation unit. The result is a public key K_x^{pub} and private key K_x^{priv}. The equivalent in Taler is a "denomination".
- **DonauKeygen** $\langle D^{pub}, D^{priv} \rangle := Keygen^D(\omega)$
- **CharityKeygen** $\langle C^{pub}, C^{priv} \rangle := Keygen^C(\omega)$
- **Donor Identifier** $i := H(\texttt{taxid}, s)$ where s is a random salt with sufficient entropy to prevent guessing attacks to invert the hash function.
- **Unique Donor Identifier** $u := \langle i, n \rangle$ where n is a high-entropy nonce to make the resulting hash unique per donation.
- **Blinding function** $\overline{u} := blind(u, b, K_x^{pub})$ where u is the value to blind, b the blinding factor to apply and K_x^{pub} the public key of the donation unit that will be used for signing. The blinding can be done with either the RSA blind signature scheme or the Blinded Clause-Schnorr signature scheme. The $\overline{u}$ is a blinded unique donor identifier which is blinded to protect the privacy of the donor.
- **Signing**
  - **Normal signing (e.g. EdDSA):**

$$\boxed{s := sign(m, k^{priv})} \tag{2.1}$$

where m is a message and k^{priv} is the private key used to sign the message, for example $k^{priv} = D^{priv}$ or $k^{priv} = C^{priv}$.
Applications:

* Signatures over **Blinded Unique Donor Identifier-key-pair** or **BUDI-key-pairs**:

$$\mu := \langle \overline{u}, H(K_x^{pub}) \rangle \tag{2.2}$$

$$\boxed{\vec{\mu}_s := sign(\vec{\mu}, C^{priv})} \tag{2.3}$$

where $H(K_x^{pub})$ indicates which donation unit key should be used by the Donau to sign the resulting donation receipt. Thus, this hash carries the information about the exact value the final donation receipt should carry.

A charity signs a collection of *BUDI-key-pair* before transfering them to the Donau to issue *Donation Receipts*

* Signing over **Donation Statement signatures**:

$$\sigma := \langle i, a_\Sigma, \texttt{year} \rangle \tag{2.4}$$

$$\boxed{\sigma_s := sign(\sigma, D^{priv})} \tag{2.5}$$

where D^{priv} is the private key from the Donau. These signatures attest the amount donated in a particular year by a specific donor.

The Donau computes the *donation statement signature* for a donor for a specific year

– **Blind signing(e.g. RSA/CS):**

$$\boxed{\overline{\beta} := blind\_sign(\overline{u}, K_x^{priv})} \tag{2.6}$$

where $\overline{u}$ is a blinded value and K_x^{priv} is the private key used to blind sign the message.
Application:

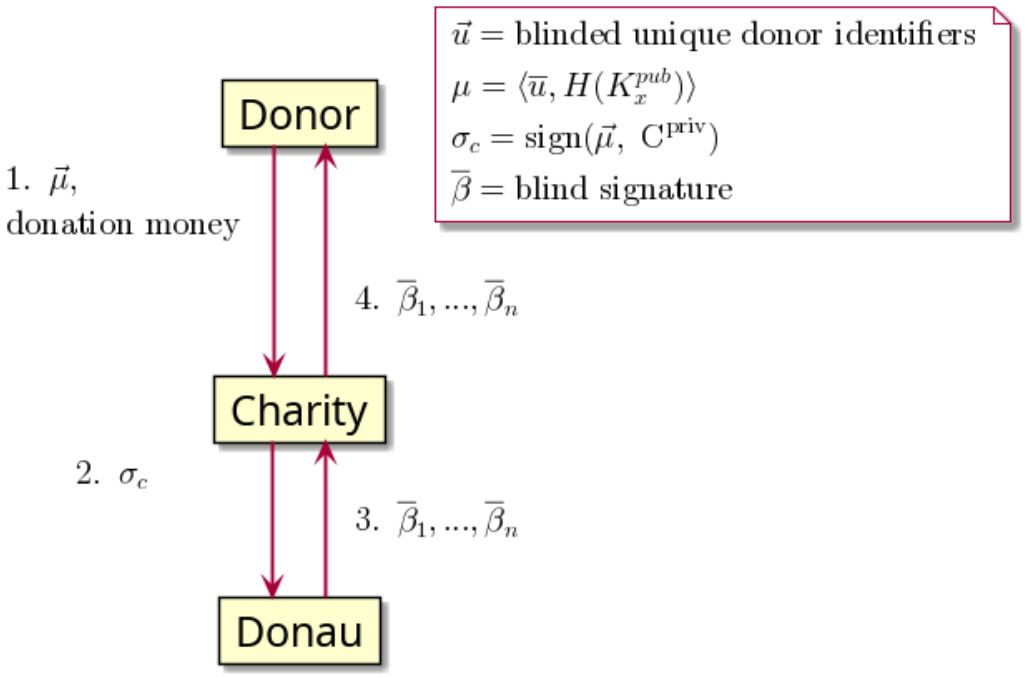
* The Donau blind signs *Blinded Unique Donor Identifier*s received from the charity with the private key matching the public key in the received *BUDI-key-pair*

- **Unblinding function** $\beta := Unblind(\overline{\beta}, b, K_x^{pub})$ where $\overline{\beta}$ is the value to unblind, b the blinding factor to apply and K_x^{pub} the public key of the donation unit that was used for signing. The unblinding must be carried out using the same signature scheme that has already been used for blinding. The unblinded value β is a unique donor identifier.

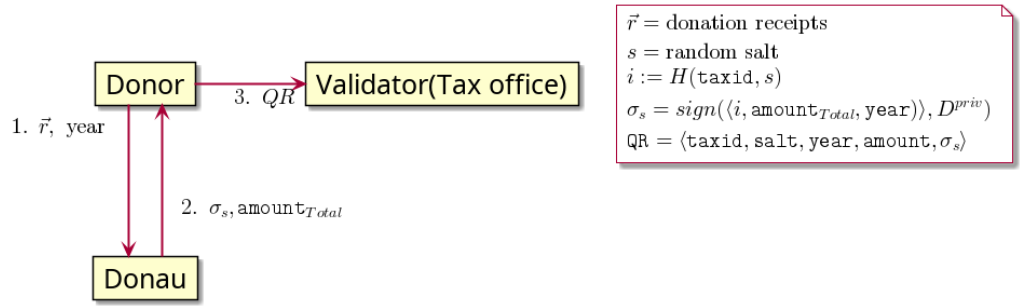
- **Verify functions** - to verify the signatures
  m corresponds to the message and s to the signature:
  - **normal verify**
    $verify(m, s, P^{pub})$ where P^{pub} can be the public key of the Donau D^{pub} or of the charity C^{pub}.
  - **blind verify** - verify a signature that was made blind
    $verify\_blind(m, s, K_x^{pub})$ verifies only signatures made with K_x^{priv}.
- **Donation Receipt** $r := \langle u, \beta, H(K_x^{pub})\rangle$ where β is the unblinded signature: Sent to the Donau to get the donation Statement.

2.2 Overview

2.2.1 Donation: spend and get donation receipt



2.2.2 Get donation statement for taxes after tax period



2.3 Protocol Detail

2.3.1 Key generation and initial setup

Initial Donau setup

1. The Donau generates a public key D^{pub} and private key D^{priv} for EdDSA signing.
2. The Donau generates the *donation unit keys* consisting of K_x^{pub} and K_x^{priv} where x is the associated value.

Charity setup (Charity side and Donau side)

1. The **charity** generates the key pair (C^{pub}, C^{priv}) and downloads the *donation unit public keys* from the donau.
2. The **charity** transmits C^{pub} and the desired yearly donation limit to the party responsible for Donau administration using a **secure channel**.
3. The party in charge of **Donau administration** ensures that the applying party is authentic and if it is publicly recognized as charity organisation. Furthermore, it ensures that all eventual other checks required by law are done. If everything is clear, it registers the public key C^{pub} and sets the requested yearly donation limit for the charitiy.

2.3.2 Continuously during tax period: get donation receipts

Overview

Donor donates to charity and transmits unique donor ids (future donation receipts)

1. The donor downloads the *donation unit public keys* K_x^{pub} for the corresponding year from the Donau. (if not already done)
2. The donor splits the donation amount into a sum of *donation units* offered by the Donau.
   Example: With donation units {1,2,4} available, and a donation with a total value of 7, the donation amount is split into the sum 4+2+1.
3. The donor generates as many *unique donor identifiers* as there are terms in the calculated sum. *Example: In our example, there will be 3 unique donor identifiers: one per donation unit, so one for the value 4, one for the value 2, one*

for the value 1.[1]

$$i := h(\texttt{taxid, salt}) \tag{2.7}$$
$$u_1 := \langle i, \texttt{nonce}_1 \rangle \tag{2.8}$$
$$u_2 := \langle i, \texttt{nonce}_2 \rangle \tag{2.9}$$
$$u_3 := \langle i, \texttt{nonce}_3 \rangle \tag{2.10}$$

4. The donor blinds the *unique donor identifiers* using a **different** blinding factor b for every *unique donor identifier*.
Example:

$$\overline{u}_1 := blind(u_1, b_1, K_1^{pub}) \tag{2.11}$$
$$\overline{u}_2 := blind(u_2, b_2, K_2^{pub}) \tag{2.12}$$
$$\overline{u}_3 := blind(u_3, b_3, K_4^{pub}) \tag{2.13}$$

5. So far, the *unique donor identifiers* do not carry information about their value. The **intended effective value is now indicated** by grouping each *unique donor identifier* with the according (hash of the) *donation unit public key* K_x^{pub}. We call these pairs *blinded unique donor identifier-key-pair*, *budi-key-pair* or even shorter BKP.

   It is only the **intended effective** value because the value will only be attributed later on with the signature of the Donau.

   Example: Note: The public key is not in relation with the sequential index of the budi-key-pair, it only relates to the value of the pair!

$$\overline{\mu}_1 := \langle \overline{u}_1, h(K_1^{pub}) \rangle \tag{2.14}$$
$$\overline{\mu}_2 := \langle \overline{u}_2, h(K_2^{pub}) \rangle \tag{2.15}$$
$$\overline{\mu}_3 := \langle \overline{u}_3, h(K_4^{pub}) \rangle \tag{2.16}$$

$$\vec{\mu} := \langle \overline{\mu}_1, \overline{\mu}_2, \overline{\mu}_3 \rangle \tag{2.17}$$

6. The donor sends all *BKP's* the $\vec{\mu}$ as well as the corresponding payment to the charity.

[1] If one donation unit is present more than once in the sum, then there is more than one unique donor identifier required for said donation unit. This depends upon the offered donation units.

Charity sends signed BKP's to Donau

1. The charity verifies that the amount requested (based on the $h(K_x^{pub})$) for signing is lower or equal to the effective amount of the donation.
2. The charity signs (using EdDSA) a structure containing all unsigned BKP's coming from the donor.

$$\sigma_c = sign(\vec{\mu}, C^{priv}) \tag{2.18}$$

3. The charity sends this structure $\vec{\mu}$ and the signature σ_c to the Donau.

Donau sends back the blind signed UDI's to charity

1. The Donau:
   a) verifies the signature σ_c on the structure.

$$verify(\vec{\mu}, \sigma_c, C^{pub}) \tag{2.19}$$

   b) increments the current amount of donations received per year of the charity. This value is increased by the total amount of the $BUDI$'s, if the increment does not exceed the annual limit.
   c) blind signs all the $blinded\,UDI$'s, the $BUDI$'s, using the *donation unit private keys* K_x^{priv} matching the public keys $h(K^{pub})$ used in the BKP's.

$$\overline{\beta_1} = blind\_sign(\overline{u}_1, K_1^{priv}) \tag{2.20}$$
$$\overline{\beta_2} = blind\_sign(\overline{u}_2, K_2^{priv}) \tag{2.21}$$
$$\overline{\beta_3} = blind\_sign(\overline{u}_3, K_4^{priv}) \tag{2.22}$$

   d) sends back all created blind signatures $\overline{\beta_1}, \overline{\beta_2}, \overline{\beta_3}$ to the charity.
2. The charity forwards the blind signatures to the donor.
3. The donor verifies the signatures.

$$verify\_blind(u_1, \overline{\beta_1}, K_1^{pub}) \tag{2.23}$$
$$verify\_blind(u_2, \overline{\beta_2}, K_2^{pub}) \tag{2.24}$$
$$verify\_blind(u_3, \overline{\beta_3}, K_4^{pub}) \tag{2.25}$$

4. The donor unblinds the signatures of the $BUDI$'s to get the signatures of the UDI's. This results in a collection of **Donation Receipts** DR's each consisting of the UDI, the signature β and the Hash of the *donation unit public key* $h(K_x^{pub})$.

$$\beta_1 = Unblind(\overline{\beta_1}, b_1, K_1^{pub}) \tag{2.26}$$
$$\beta_2 = Unblind(\overline{\beta_2}, b_2, K_2^{pub}) \tag{2.27}$$
$$\beta_3 = Unblind(\overline{\beta_3}, b_3, K_4^{pub}) \tag{2.28}$$

$$r_1 = \langle UDI_1, \beta_1, h(K_1^{pub}) \rangle \tag{2.29}$$
$$r_2 = \langle UDI_2, \beta_2, h(K_2^{pub}) \rangle \tag{2.30}$$
$$r_3 = \langle UDI_3, \beta_3, h(K_4^{pub}) \rangle \tag{2.31}$$

2.3.3 After effective tax period: get tax statement for period from Donau

Donor sends the *Donation receipts* to the Donau to get the *Donation Statement*.

1. The donor sends the collection of all *donation receipts* r_1, r_2, r_3 to the Donau. This happens manually once per period.
   It is not done continuously to obtain *unlinkability* between the **issuance** of the donation receipts (which happens upon donation) and their **submission** for the *donation statement*.
2. For each *donation receipt* the Donau:
   - checks that K_x^{pub} is known.
   - verifies that the signature β is correct using the corresponding public key K_x^{pub}.
   - verifies that the *donor identifier* is the same as in other *donation receipts*.[2]
   - verifies that the `nonce` is unique and was not used before by the donor for the corresponding year.
3. The Donau signs over the total `amount`, `year` and *donor identifier* and sends the signature and the total amount so far back to the donor. This results in a final signature called the ***Donation Statement signature***.

[2] With multiple wallets each wallet must simply obtain a separate *donation statement*!

$$\sigma_s = sign(\langle i, \texttt{amount}_{Total}, \texttt{year}\rangle), D^{priv}) \tag{2.32}$$

Donor sends the QR Code to a validator (tax office)

1. The donor generates a QR code which contains the following:

$$\texttt{QR} = \langle \texttt{taxid}, \texttt{salt}, \texttt{year}, \texttt{amount}, \sigma_s \rangle \tag{2.33}$$

2. The validator scans the QR code and verifies the signature σ_s.

$$verify(\langle i, \texttt{amount}_{Total}, \texttt{year}\rangle), \sigma_s, D^{pub}) \tag{2.34}$$

Declaration of Authorship

I hereby declare that I have written this thesis independently and have not used any sources or aids other than those acknowledged.

All statements taken from other writings, either literally or in essence, have been marked as such.

I hereby agree that the present work may be reviewed in electronic form using appropriate software.

April 24, 2024

L. Matyja

J. Casaburi

List of Figures

List of Tables

Listings

Glossary

This document is incomplete. The external file associated with the glossary 'main' (which should be called `donau-thesis.gls`) hasn't been created.

Check the contents of the file `donau-thesis.glo`. If it's empty, that means you haven't indexed any of your entries in this glossary (using commands like `\gls` or `\glsadd`) so this list can't be generated. If the file isn't empty, the document build process hasn't been completed.

If you don't want this glossary, add `nomain` to your package option list when you load `glossaries-extra.sty`. For example:

```
\usepackage[nomain]{glossaries-extra}
```

Try one of the following:

- Add `automake` to your package option list when you load `glossaries-extra.sty`. For example:

  ```
  \usepackage[automake]{glossaries-extra}
  ```

- Run the external (Lua) application:

  ```
  makeglossaries-lite.lua "donau-thesis"
  ```

- Run the external (Perl) application:

  ```
  makeglossaries "donau-thesis"
  ```

Then rerun LaTeX on this document.

This message will be removed once the problem has been fixed.

A First Appendix Chapter