Enabling secure Web payments with Taler

Christian Grothoff

Institut National de Recherche en Informatique et en Automatique (Inria)
The GNU Project
Ashoka Fellow

24.11.2016
Motivation

“I think one of the big things that we need to do, is we need to get a way from true-name payments on the Internet. The credit card payment system is one of the worst things that happened for the user, in terms of being able to divorce their access from their identity.”

GNU Taler

Digital cash, made socially responsible.

Taxable, Anonymous, Libre, Practical, Resource Friendly
Architecture of GNU Taler

- **Exchange**
  - verify
  - deposit coins
  - withdraw coins

- **Auditor**

- **Customer**
  - spend coins

- **Merchant**
Usability of Taler

https://demo.taler.net/

1. Install Chrome extension.
2. Visit the bank.demo.taler.net to withdraw coins.
3. Visit the shop.demo.taler.net to spend coins.
Value proposition: Customer

- Convenient: pay with one click
- Guaranteed: never fear being rejected by false-positives in the fraud detection
- Secure: like cash, except no worries about counterfeit
- Privacy-preserving: payment requires no personal information
- Stable: no currency fluctuations, pay in traditional currencies
- Free software: no hidden “gadgets”, third parties can verify
Value proposition: Merchant

- Fast: transactions at Web-speed
- Secure: signed contracts, no legitimate customer rejected by fraud detection
- Free software: competitive pricing and support
- Low fees: efficient protocol + no fraud = low costs
- Flexible: any currency, any amount
- Ethical: no fluctuation risk, no pyramid scheme, not suitable for illegal business
- Legal: complies with Regulation (EU) 2016/679 (GDPR)\(^1\)

---

\(^1\)Requires privacy by design and data minimization for all data processing in Europe after 25.5.2018.
Value proposition: Government

- Free software = commons: no monopoly, preserve independence
- Taxability: reduces black markets
- Efficiency: high transaction costs hurt the economy
- Security: signed contracts, no counterfeit
- Audited: no bad banks
- Privacy: protection against foreign espionage
Taxability

We say Taler is taxable because:

- Merchant’s income is visible from deposits.
- Hash of contract is part of deposit data.
- State can trace income and enforce taxation.
Taxability

We say Taler is taxable because:

- Merchant’s income is visible from deposits.
- Hash of contract is part of deposit data.
- State can trace income and enforce taxation.

Limitations:

- withdraw loophole
- *sharing* coins among family and friends
How does it work?

We use a few ancient constructions:

▶ Cryptographic hash function (1989)
▶ Blind signature (1983)
▶ Schnorr signature (1989)
▶ Diffie-Hellman key exchange (1976)
▶ Cut-and-choose zero-knowledge proof (1985)

But of course we use modern instantiations.
Global setup: Pick an Elliptic curve

Need:

\[ G \] generator in ECC curve, a point

- size of ECC group, \( o := |G| \), \( o \) prime

Now we can, for example, compute:

\[
A = G + G \\
= 2G \\
B = A + G \\
= 3G \\
C = cG \text{ for } c \in \mathbb{Z}
\]

Note:

\[ G = (o + 1)G \]
Exchange setup: Create a denomination key (RSA)

1. Pick random primes $p, q$.
2. Compute $n := pq$
   $\phi(n) = (p - 1)(q - 1)$
3. Pick small $e < \phi(n)$ such that $d := e^{-1} \mod \phi(n)$ exists.
4. Publish public key $(e, n)$. 
Merchant: Create a signing key (EdDSA)

- pick random $m \mod o$ as private key
- $M = mG$ public key

Capability: $m \Rightarrow M$
Customer: Create a planchet (EdDSA)

- Pick random $c \mod o$ private key
- $C = cG$ public key

Capability: $c \Rightarrow$
Customer: Blind planchet (RSA)

1. Obtain public key \((e, n)\)
2. Compute \(m := FDH(C), \quad m < n\).
3. Pick blinding factor \(b \in \mathbb{Z}_n\)
4. Transmit \(m' := mb^e \mod n\)
Exchange: Blind sign (RSA)

1. Receive $m'$.
2. Compute $s' := m'^d \mod n$.
3. Send signature $s'$.

Customer
Customer: Unblind coin (RSA)

1. Receive $s'$.
2. Compute $s := s'b^{-1} \mod n$. 
Customer: Build shopping cart

www

Merchant

transmit

Merchant
Merchant: Propose contract (EdDSA)

1. Complete proposal $D$.
2. Send $D$, $EdDSA_m(D)$
Customer: Spend coin (EdDSA)

1. Receive proposal $D$, $EdDSA_m(D)$.
2. Send $s$, $C$, $EdDSA_c(D)$
Merchant and Exchange: Verify coin (RSA)

\[ s^e \equiv m \pmod{n} \]
Giving change

It would be inefficient to pay EUR 100 with 1 cent coins!

- Denomination key represents value of a coin.
- Exchange may offer various denominations for coins.
- Wallet may not have exact change!
- Usability requires ability to pay given sufficient total funds.
Giving change

It would be inefficient to pay EUR 100 with 1 cent coins!

▶ Denomination key represents value of a coin.
▶ Exchange may offer various denominations for coins.
▶ Wallet may not have exact change!
▶ Usability requires ability to pay given sufficient total funds.

Key goals:

▶ maintain unlinkability
▶ maintain taxability of transactions
Giving change

It would be inefficient to pay EUR 100 with 1 cent coins!

- Denomination key represents value of a coin.
- Exchange may offer various denominations for coins.
- Wallet may not have exact change!
- Usability requires ability to pay given sufficient total funds.

Key goals:

- maintain unlinkability
- maintain taxability of transactions

Method:

- Contract can specify to only pay *partial value* of a coin.
- Exchange allows wallet to obtain *unlinkable change* for remaining coin value.
Strawman solution

Given partially spent private coin key $c_{old}$:

1. Pick random $c_{new} \mod o$ private key
2. $C_{new} = c_{new} G$ public key
3. Pick random $b_{new}$
4. Compute $m_{new} := FDH(C_{new}), m < n$.
5. Transmit $m_{new}' := m_{new} b_{new}^e \mod n$

... and sign request for change with $c_{old}$.
Strawman solution

Given partially spent private coin key $c_{old}$:

1. Pick random $c_{new}$ mod o private key
2. $C_{new} = c_{new} G$ public key
3. Pick random $b_{new}$
4. Compute $m_{new} := FDH(C_{new}), m < n.$
5. Transmit $m'_{new} := m_{new} b_{new}^e$ mod n

... and sign request for change with $c_{old}$.

Problem: Owner of $c_{new}$ may differ from owner of $c_{old}$!
Diffie-Hellman (ECDH)

1. Create private keys $d, h \mod o$
2. Define $D = dG$
3. Define $H = hG$
4. Compute $DH := d(hG) = h(dG)$
Customer: Transfer key setup (ECDH)

Given partially spent private coin key $c_{old}$:

1. Let $C_{old} := c_{old}G$ (as before)
2. Create random private transfer key $t \mod o$
3. Compute $T := tG$
4. Compute $X := c_{old}(tG) = t(c_{old}G) = tC_{old}$
5. Derive $c_{new}$ and $b_{new}$ from $X$ (KDF)
6. Compute $C_{new} := c_{new}G$
7. Compute $m_{new} := FDH(C_{new})$
8. Transmit $m'_{new} := m_{new}b_{new}$
Cut-and-Choose

\[ c_t \]

\[ c \]
\[ c_{new,1} \]
\[ b_{new,1} \]

transmit

Exchange

\[ c \]
\[ c_{new,2} \]
\[ b_{new,2} \]

transmit

Exchange

\[ c \]
\[ c_{new,3} \]
\[ b_{new,3} \]

transmit

Exchange
Exchange: Choose!

Exchange sends back random $\gamma \in \{1, 2, 3\}$ to the customer.
1. If $\gamma = 1$, send $t_2, t_3$ to exchange
2. If $\gamma = 2$, send $t_1, t_3$ to exchange
3. If $\gamma = 3$, send $t_1, t_2$ to exchange
Exchange: Verify ($\gamma = 2$)
1. Take $m_{new,\gamma}'$.
2. Compute $s' := m_{new,\gamma}'^d \mod n$.
3. Send signature $s'$.
Customer: Unblind change (RSA)

1. Receive $s'$.
2. Compute $s := s' b_{new,\gamma}^{-1} \mod n$. 

\[
\begin{align*}
&b_{new,\gamma} \\
\end{align*}
\]
Exchange: Allow linking change

Given $C_{old}$

return $T_\gamma, s := s' b_{new,\gamma}^{-1} \mod n.$
Customer: Link (threat!)

1. Have $c_{old}$.
2. Obtain $T_{\gamma}$, $s$ from exchange
3. Compute $X_{\gamma} = c_{old} T_{\gamma}$
4. Derive $c_{new,\gamma}$ and $b_{new,\gamma}$ from $X_{\gamma}$
5. Unblind $s := s^{'b_{new,\gamma}^{-1}} \mod n$
Refresh protocol summary

- Customer asks exchange to convert old coin to new coin
- Protocol ensures new coins can be recovered from old coin
  $\Rightarrow$ New coins are owned by the same entity!

Thus, the refresh protocol allows:

- To give unlinkable change.
- To give refunds to an anonymous customer.
- To expire old keys and migrate coins to new ones.

Transactions via refresh are equivalent to sharing a wallet.
Current technical developments

- Tutorial for merchants
- Tutorial for Web shop integration
- Improving wallet (error handling, features, browser support)
- Ongoing work on exchange auditing
Business considerations

- Exchange needs to be a legal (!) business to operate.
- Exchange operator income is from *transaction fees*.
- Created Taler Systems S.A. in Luxemburgh.
- Now trying to find partners and financing for startup.
Conclusion

What can we do?

▶ Suffer mass-surveillance enabled by credit card oligopolies with high fees, and
▶ Engage in arms race with deliberately unregulatable blockchains

OR

▶ Establish free software alternative balancing social goals
Do you have any questions?

References:


