

GNU

<Taler>

taler.net

IRC#taler

(on freenode)

twitter@taler

mail@taler.net

**Florian Dold &  
Christian Grothoff**

{dold,grothoff}@taler.net



## A Social Problem

This was a question posed to RAND researchers in 1971:

*“Suppose you were an advisor to the head of the KGB, the Soviet Secret Police. Suppose you are given the assignment of designing a system for the surveillance of all citizens and visitors within the boundaries of the USSR. The system is not to be too obtrusive or obvious. What would be your decision?”*

## A Social Problem

This was a question posed to RAND researchers in 1971:

*“Suppose you were an advisor to the head of the KGB, the Soviet Secret Police. Suppose you are given the assignment of designing a system for the surveillance of all citizens and visitors within the boundaries of the USSR. The system is not to be too obtrusive or obvious. What would be your decision?”*

### **Mastercard/Visa are too transparent.**

“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.”

—Edward Snowden, IETF 93 (2015)

# Payment System Surveillance is Real



TOP SECRET//SI//REL TO USA, FVEY

## Private Networks are Important



- Many targets use private networks.

Google infrastructure	SWIFT Network
French MFA	Petrobras

- Evidence in Survey: 30%-40% of traffic in BLACKPEARL has at least one endpoint private.

TOP SECRET//SI//REL TO USA, FVEY



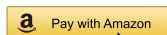
# Credit Card Surveillance

- ▶ When you pay by CC, the information includes your name
- ▶ When you pay in person with CC, your location is also known
- ▶ You often have no alternative payment methods available
- ▶ You hardly ever can use someone else's CC
- ▶ Anonymous prepaid cards are difficult to get and expensive
- ▶ Payment information is typically stored for at least 6 years



# The Bank's Problem

- ▶ Global tech companies push oligopolies
- ▶ Privacy and federated finance are at risk
- ▶ Economic sovereignty is in danger

The PayPal logo, consisting of the word "PayPal" in a blue, italicized sans-serif font with a trademark symbol.The Alipay logo, featuring the Chinese characters "支付宝" in blue and orange, with "Alipay.com" written below in orange.A yellow rectangular button with rounded corners. On the left is the Amazon logo (a lowercase 'a' with a smile). To its right is the text "Pay with Amazon". A mouse cursor is pointing at the bottom right corner of the button.The Apple Pay logo, featuring the Apple logo (a silhouette of an apple with a bite taken out) followed by the word "Pay" in a black sans-serif font.The Samsung Pay logo, which is a blue rounded square containing the word "SAMSUNG" in white uppercase letters at the top and "pay" in white lowercase letters below it.The Android Pay logo, which is a black square containing a white circle. Inside the circle is the green Android robot icon above the word "pay" in white lowercase letters.

# Predicting the Future

- ▶ Google, Apple or Facebook's Libra will be your bank and run your payment system
- ▶ They target advertising based on your purchase history, location and your ability to pay
- ▶ They will provide more usable, faster and broadly available payment solutions; our federated banking system will be history
- ▶ After dominating the payment sector, they will start to charge fees befitting their oligopoly size
- ▶ Competitors and vendors not aligning with their corporate "values" will be excluded by terms of service and go bankrupt
- ▶ The imperium will have another major tool for its financial warfare



# The Distraction: Bitcoin

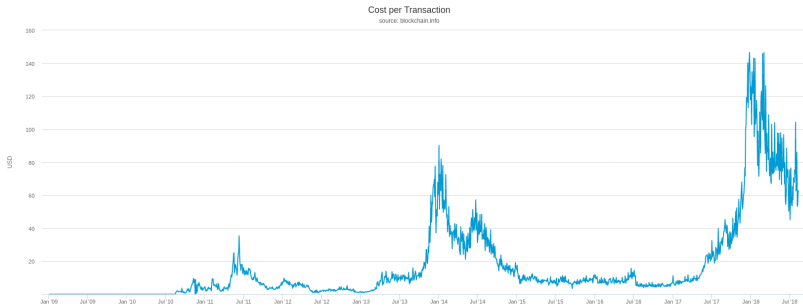
- ▶ Unregulated payment system and currency:  
⇒ lack of regulation is a feature!
- ▶ Implemented in free software
- ▶ Decentralised peer-to-peer system

# The Distraction: Bitcoin

- ▶ Unregulated payment system and currency:  
⇒ lack of regulation is a feature!
- ▶ Implemented in free software
- ▶ Decentralised peer-to-peer system
- ▶ Decentralised banking requires solving Byzantine consensus
- ▶ Creative solution: tie initial accumulation to solving consensus

# The Distraction: Bitcoin

- ▶ Unregulated payment system and currency:
  - ⇒ lack of regulation is a feature!
- ▶ Implemented in free software
- ▶ Decentralised peer-to-peer system
- ▶ Decentralised banking requires solving Byzantine consensus
- ▶ Creative solution: tie initial accumulation to solving consensus
  - ⇒ Proof-of-work advances ledger
  - ⇒ Very expensive banking



Current average transaction value:  $\approx$  1000 USD



Cryptography is rather primitive:

**All Bitcoin transactions are public and linkable!**

⇒ no privacy guarantees

⇒ enhanced with “laundering” services

ZeroCoin, CryptoNote (Monero) and ZeroCash (ZCash) offer anonymity.

**Do you want to have a libertarian economy?**

**Do you want to live under total surveillance?**

**Digital** cash, made **socially**  
**responsible.**

**< T a l e r >**

Privacy-Preserving, Practical, Taxable, Free Software, Efficient

# What is Taler?

Taler is an electronic instant payment system.

- ▶ Uses electronic coins stored in **wallets** on customer's device
- ▶ Like **cash**
- ▶ Pay in **existing currencies** (i.e. EUR, USD, BTC), or use it to create new **regional currencies**

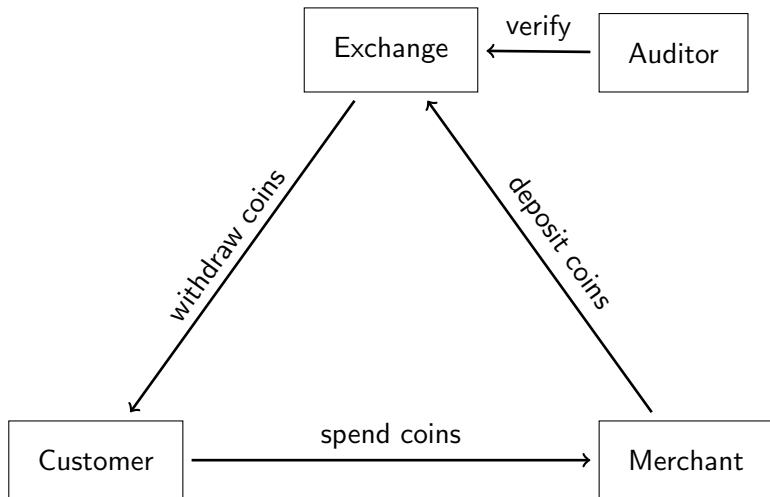


# Design goals for the GNU Taler Payment System

GNU Taler must ...

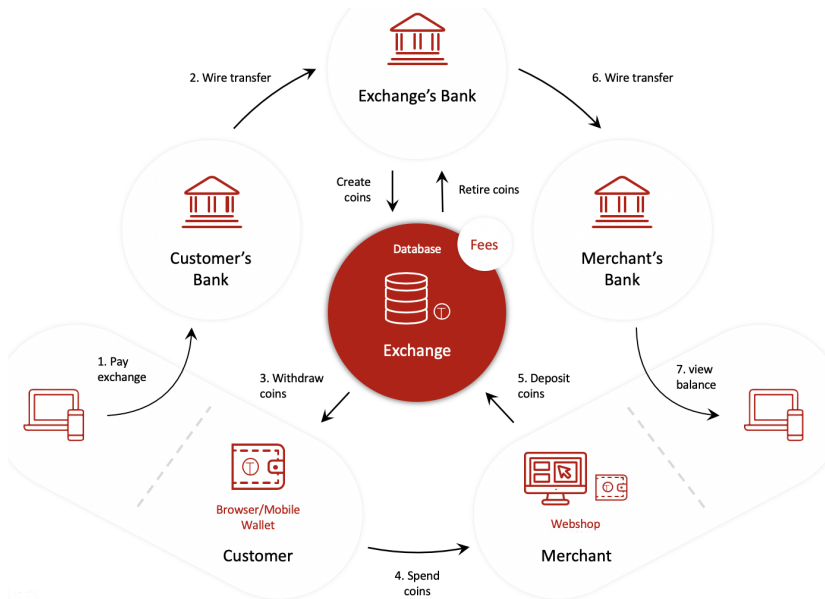
1. ... be implemented as **free software**.
2. ... protect the **privacy of buyers**.
3. ... must enable the state to **tax income** and crack down on illegal business activities.
4. ... prevent payment fraud.
5. ... only **disclose the minimal amount of information necessary**.
6. ... be usable.
7. ... be efficient.
8. ... avoid single points of failure.
9. ... foster **competition**.

# Taler Overview



# Taler in Operation

# Taler in Operation



# Usability of Taler

`https://demo.taler.net/`

1. Install browser extension.
2. Visit the `bank.demo.taler.net` to withdraw coins.
3. Visit the `shop.demo.taler.net` to spend coins.

# Use Case: Journalism

Today:

- ▶ Corporate structure
- ▶ Advertising primary revenue
- ▶ Tracking readers critical for business success
- ▶ Journalism and marketing hard to distinguish

# Use Case: Journalism

Today:

- ▶ Corporate structure
- ▶ Advertising primary revenue
- ▶ Tracking readers critical for business success
- ▶ Journalism and marketing hard to distinguish

With GNU Taler:

- ▶ One-click micropayments per article
- ▶ Hosting requires no expertise
- ▶ Reader-funded reporting separated from marketing
- ▶ Readers can remain anonymous

## Use Case: Anti-Spam

Today, PGP provides authenticated encryption for e-mail:

- ▶ Free software
- ▶ Easy to use opportunistic encryption
- ▶ Available for Outlook, Android, Enigmail
- ▶ Spies & spam filters can no longer inspect content



## Use Case: Anti-Spam

Today, p≡p provides authenticated encryption for e-mail:

- ▶ Free software
- ▶ Easy to use opportunistic encryption
- ▶ Available for Outlook, Android, Enigmail
- ▶ Spies & spam filters can no longer inspect content

With GNU Taler:

- ▶ Peer-to-peer payments via e-mail
- ▶ If unsolicited sender, hide messages from user & automatically request payment from sender
- ▶ Sender can attach payment to be moved to inbox
- ▶ Receiver may grant refund to sender

**Where might this get us exactly?**

# Visions

- ▶ Be paid to read advertising, starting with spam
- ▶ Give welfare without intermediaries taking huge cuts
- ▶ Forster regional trade via regional currencies
- ▶ Eliminate corruption by making all income visible
- ▶ Stop the mining by making crypto-currencies useless for anything but crime

**What is there?**

# Components

- ▶ REST APIs, C APIs
- ▶ Command-line, WebExtension (Firefox, Chrome, Chromium, Brave) and Android wallet
- ▶ GLS bank integration (libeufin, WiP)
- ▶ Escrow/backup solution (Anastasis, WiP)
- ▶ Merchant backend & backoffice (needs love)
- ▶ WooCommerce plugin (needs update)
- ▶ Taler-enabled vending machine (MDB)
- ▶ Sample Web frontends
- ▶ Twister

**How can you help?**

# How to support?

- ▶ Join: `taler@gnu.org`, `#taler`
- ▶ Testing: try it out, report issues (<https://bugs.gnunet.org/>)
- ▶ Translation: translate Web site and software (GNU gettext)
- ▶ Propaganda: spread the word (<https://git.taler.net/marketing.git>)
- ▶ Documentation: explain things better (<https://docs.taler.net/>)
- ▶ Integration: <https://git.taler.net/>
- ▶ Security audits: study our code and design

## Technology deep dive



# 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?

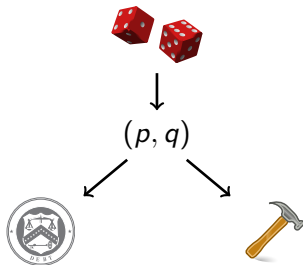
We use a few ancient constructions:

- ▶ 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.

# 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  
 $d := e^{-1} \pmod{\phi(n)}$  exists.
4. Publish public key  $(e, n)$ .



# Merchant: Create a signing key (EdDSA)

- ▶ pick random  $m \pmod{o}$  as private key
- ▶  $M = mG$  public key



↓  
 $m$

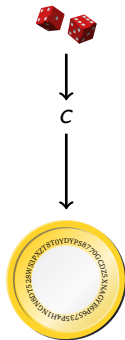
↓  
 $M$

**Capability:**  $m \Rightarrow$



# Customer: Create a planchet (EdDSA)

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

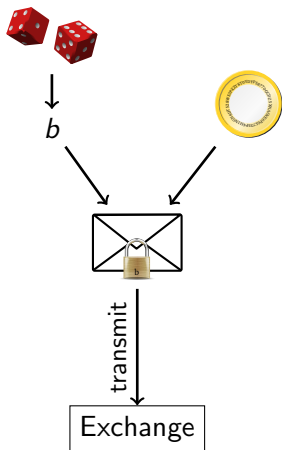


**Capability:**  $c \Rightarrow$



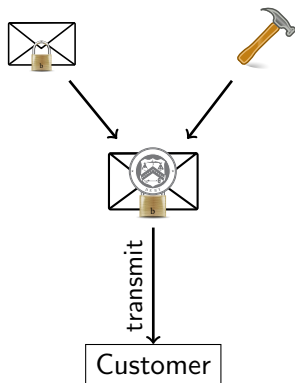
# Customer: Blind planchet (RSA)

1. Obtain public key  $(e, n)$
2. Compute  $f := FDH(C)$ ,  $f < n$ .
3. Pick blinding factor  $b \in \mathbb{Z}_n$
4. Transmit  $f' := fb^e \pmod n$



## Exchange: Blind sign (RSA)

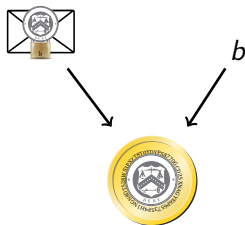
1. Receive  $f'$ .
2. Compute  $s' := f'^d \pmod n$ .
3. Send signature  $s'$ .





# Customer: Unblind coin (RSA)

1. Receive  $s'$ .
2. Compute  $s := s'b^{-1} \pmod n$



## Customer: Build shopping cart



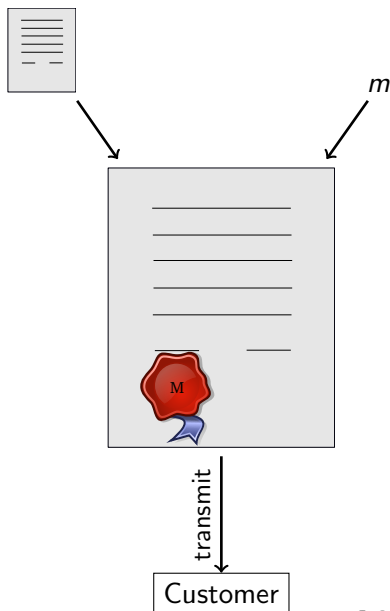
transmit



Merchant

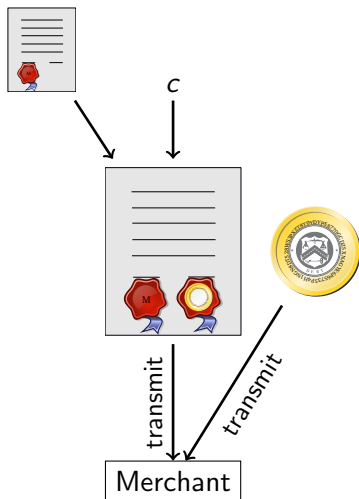
## Merchant: Propose contract (EdDSA)

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)

$$s^e \stackrel{?}{\equiv} FDH(C) \pmod{n}$$



## 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

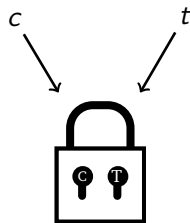
Method:

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



# Diffie-Hellman (ECDH)

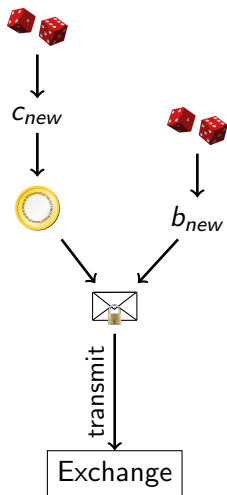
1. Create private keys  $c, t \pmod{o}$
2. Define  $C = cG$
3. Define  $T = tG$
4. Compute DH  
 $cT = c(tG) = t(cG) = tC$



# Strawman solution

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

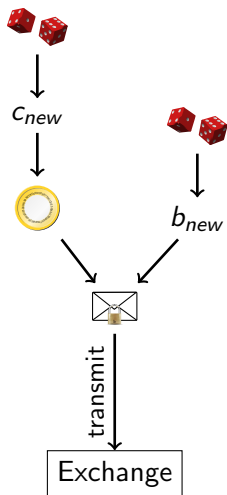
1. Pick random  $c_{new}$  mod  $o$  private key
  2.  $C_{new} = c_{new}G$  public key
  3. Pick random  $b_{new}$
  4. Compute  $f_{new} := FDH(C_{new})$ ,  $m < n$ .
  5. Transmit  $f'_{new} := f_{new}b_{new}^e$  mod  $n$
- ... and sign request for change with  $c_{old}$ .



# 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  $f_{new} := FDH(C_{new})$ ,  $m < n$ .
  5. Transmit  $f'_{new} := f_{new}b_{new}^e$  mod  $n$
- ... and sign request for change with  $c_{old}$ .

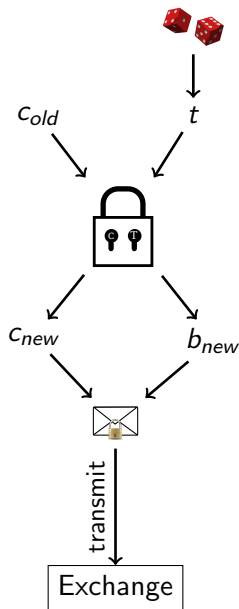


**Problem:** Owner of  $C_{new}$  may differ from owner of  $C_{old}$ !

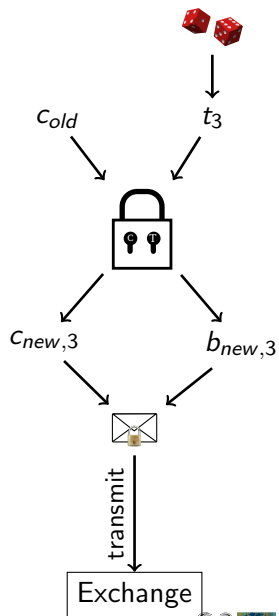
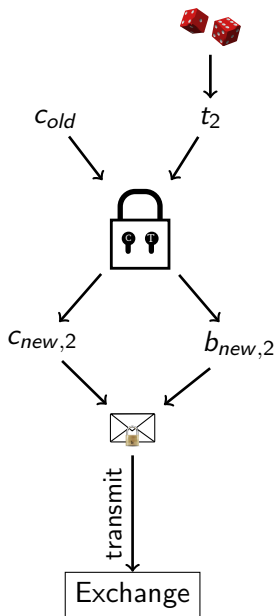
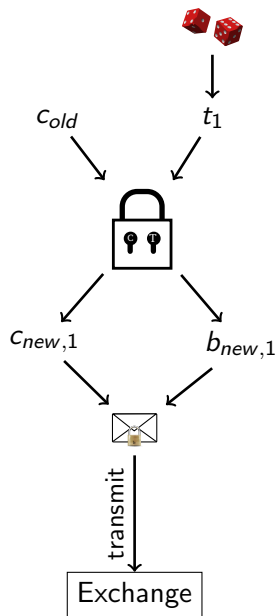
# Customer: Transfer key setup (ECDH)

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

1. Let  $C_{old} := c_{old}G$  (as before)
2. Create random private transfer key  $t \bmod o$
3. Compute  $T := tG$
4. Compute  $X := c_{old}(tG) = t(c_{old}G) = tC_{old}$
5. Derive  $c_{new}$  and  $b_{new}$  from  $X$
6. Compute  $C_{new} := c_{new}G$
7. Compute  $f_{new} := FDH(C_{new})$
8. Transmit  $f'_{new} := f_{new}b_{new}^e$



# Cut-and-Choose



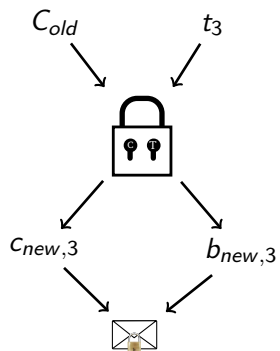
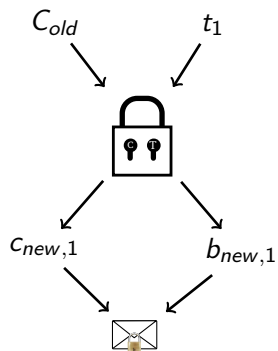
## Exchange: Choose!

Exchange sends back random  $\gamma \in \{1, 2, 3\}$  to the customer.

## Customer: Reveal

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

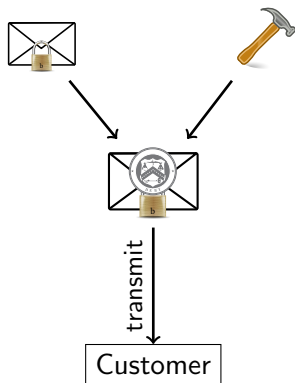
## Exchange: Verify ( $\gamma = 2$ )





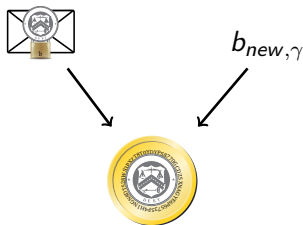
# Exchange: Blind sign change (RSA)

1. Take  $f'_{new,\gamma}$ .
2. Compute  $s' := f'^d_{new,\gamma} \pmod n$ .
3. Send signature  $s'$ .



# Customer: Unblind change (RSA)

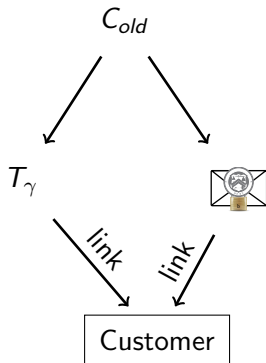
1. Receive  $s'$ .
2. Compute  $s := s' b_{new,\gamma}^{-1} \pmod n$ .



## Exchange: Allow linking change

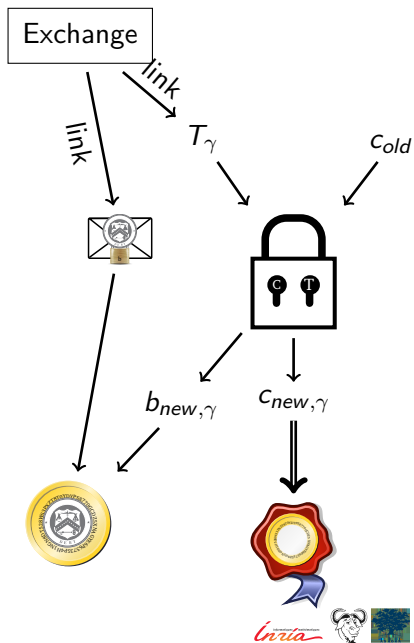
Given  $C_{old}$

return  $T_\gamma, s := s' b_{new,\gamma}^{-1} \pmod n.$



# 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  $s := s' b_{new,\gamma}^{-1} \pmod n$



## Refresh protocol summary

- ▶ Customer asks exchange to convert old coin to new coin
- ▶ Protocol ensures new coins can be recovered from old coin
- ⇒ 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.
- ▶ To handle protocol aborts.

**Transactions via refresh are equivalent to sharing a wallet.**

## Warranting deposit safety

Exchange has *another* online signing key  $W = wG$ :

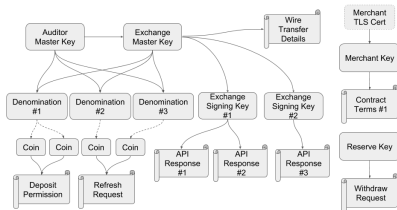
Sends  $E$ ,  $EdDSA_w(M, H(D), FDH(C))$  to the merchant.

This signature means that  $M$  was the *first* to deposit  $C$  and that the exchange thus must pay  $M$ .

Without this, an evil exchange could renege on the deposit confirmation and claim double-spending if a coin were deposited twice, and then not pay either merchant!

# Online keys

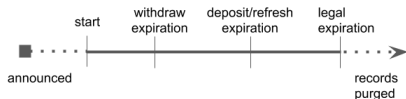
- ▶ The exchange needs  $d$  and  $w$  to be available for online signing.
- ▶ The corresponding public keys  $W$  and  $(e, n)$  are certified using Taler's public key infrastructure (which uses offline-only keys).



**What happens if those private keys are compromised?**

## Denomination key $(e, n)$ compromise

- ▶ An attacker who learns  $d$  can sign an arbitrary number of illicit coins into existence and deposit them.
  - ▶ Auditor and exchange can detect this once the total number of deposits (illicit and legitimate) exceeds the number of legitimate coins the exchange created.
  - ▶ At this point,  $(e, n)$  is *revoked*. Users of *unspent* legitimate coins reveal  $b$  from their withdrawal operation and obtain a *refund*.
  - ▶ The financial loss of the exchange is *bounded* by the number of legitimate coins signed with  $d$ .
- ⇒ Taler frequently rotates denomination signing keys and deletes  $d$  after the signing period of the respective key expires.





## Online signing key $w$ compromise

- ▶ An attacker who learns  $w$  can sign deposit confirmations.
- ▶ Attacker sets up two (or more) merchants and customer(s) which double-spend legitimate coins at both merchants.
- ▶ The merchants only deposit each coin once at the exchange and get paid once.
- ▶ The attacker then uses  $w$  to fake deposit confirmations for the double-spent transactions.
- ▶ The attacker uses the faked deposit confirmations to complain to the auditor that the exchange did not honor the (faked) deposit confirmations.

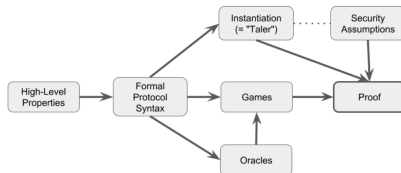
The auditor can then detect the double-spending, but cannot tell who is to blame, and (likely) would presume an evil exchange, forcing it to pay both merchants.

## Detecting online signing key $W$ compromise

- ▶ Merchants are required to *probabilistically* report signed deposit confirmations to the auditor.
- ▶ Auditor can thus detect exchanges not reporting signed deposit confirmations.
- ⇒ Exchange can rekey if illicit key use is detected, then only has to honor deposit confirmations it already provided to the auditor *and* those without proof of double-spending *and* those merchants reported to the auditor.
- ⇒ Merchants that do not participate in reporting to the auditor risk their deposit permissions being voided in cases of an exchange's private key being compromised.

# Technology Summary

- ▶ We can design protocols that fail *soft*.
- ▶ GNU Taler's design limits financial damage even in the case private keys are compromised.
- ▶ GNU Taler does more:
  - ▶ Gives change, can provide refunds
  - ▶ Integrates nicely with HTTP, handles network failures
  - ▶ High performance
  - ▶ Formal security proofs



- ▶ More information at <https://taler.net/>.

# Competitor comparison

	Cash	Bitcoin	Zerocoin	Creditcard	GNU Taler
Online	---	++	++	+	+++
Offline	+++	--	--	+	--
Trans. cost	+	----	----	-	++
Speed	+	----	----	o	++
Taxation	-	--	----	+++	+++
Payer-anon	++	o	++	----	+++
Payee-anon	++	o	++	----	----
Security	-	o	o	--	++
Conversion	+++	----	----	+++	+++
Libre	-	+++	+++	---	+++

# 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, and
- ▶ Enjoy the “benefits” of cash



OR

- ▶ Establish free software alternative balancing social goals!

# Do you have any questions?

## References:

1. Christian Grothoff, Bart Polot and Carlo von Loesch. *The Internet is broken: Idealistic Ideas for Building a GNU Network*. **W3C/IAB Workshop on Strengthening the Internet Against Pervasive Monitoring (STRINT)**, 2014.
2. Jeffrey Burdges, Florian Dold, Christian Grothoff and Marcello Stanisci. *Enabling Secure Web Payments with GNU Taler*. **SPACE 2016**.
3. Florian Dold, Sree Harsha Totakura, Benedikt Müller, Jeffrey Burdges and Christian Grothoff. *Taler: Taxable Anonymous Libre Electronic Reserves*. Available upon request. 2016.
4. Eli Ben-Sasson, Alessandro Chiesa, Christina Garman, Matthew Green, Ian Miers, Eran Tromer and Madars Virza. *Zerocash: Decentralized Anonymous Payments from Bitcoin*. **IEEE Symposium on Security & Privacy, 2016**.
5. David Chaum, Amos Fiat and Moni Naor. *Untraceable electronic cash*. **Proceedings on Advances in Cryptology, 1990**.
6. Phillip Rogaway. *The Moral Character of Cryptographic Work*. **Asiacrypt, 2015**.

**Let money facilitate trade; but ensure capital serves society.**