Off-line Payments with Auditable Tracing

Dennis Kügler and Holger Vogt

Department of Computer Science*
Darmstadt University of Technology
D-64283 Darmstadt, Germany
{kuegler|hvogt}@cdc.informatik.tu-darmstadt.de

Abstract Tracing is an important mechanism to prevent crimes in anonymous payment systems. However, it is also a threat to the customer's privacy as long as its application cannot be controlled. Relying solely on trusted third parties for tracing is inadequate, as there are no strong guarantees that deanonymizations are only applied legally.
A recent tracing concept is auditable tracing, where the customer has the power to control the deanonymization. With auditable tracing no trust is required, while it offers comparable tracing mechanisms.
We present the first off-line payment system with auditable tracing. Our payment system supports coin and owner tracing as well as self deanonymization in the case of blackmailing.

1 Introduction

Anonymous electronic payment systems based on blind signatures [Cha83] have been suggested to protect the privacy of customers. However, von Solms and Naccache [vSN92] have shown that anonymity may be misused for untraceable blackmailing of customers, which they called "perfect crime". Furthermore, anonymity may prevent investigations of money laundering and illegal purchases. Due to these anonymity related problems payment systems with revocable anonymity have been requested by governments and banks, and thus tracing methods have been invented, where the withdrawal and the deposit of coins can be linked by two complementary tracing mechanisms [SPC95]:

Coin tracing: The withdrawn coins of a suspicious or blackmailed customer are deanonymized so that the bank will recognize these coins at deposit.

Owner tracing: The coins deposited by a suspicious merchant are deanonymized so that the identity of their withdrawer is revealed.

Payment systems with trusted third parties (e.g. [CMS96,JY96,FTY96,DFTY97]) can provide both tracing mechanisms very effectively, but also have several shortcomings: The introduction of a trusted third party causes additional costs, which

* This work was supported by the Deutsche Forschungsgemeinschaft (DFG) as part of the PhD program (Graduiertenkolleg) "Enabling Technologies for Electronic Commerce" at Darmstadt University of Technology.

neither the bank nor the customer is willing to pay for. But even worse, the achieved level of anonymity is uncertain, as any misuse of tracing by the trusted third party cannot be detected. Demanding a quorum of trusted third parties to cooperate for tracing only guarantees increasing costs, while conspiring trusted third parties may still be able to undetectably trace without permission.

Recent approaches for payment systems abandon the idea of trusted third party tracing [STS99, PS01, KV01a], but they only protect against blackmailing and lack support for coin and owner tracing. It was shown in [KV01b, KV01c] that coin and owner tracing also can be implemented without any trusted third party by introducing an audit concept. However, these payment systems require the bank to be *on-line* at payment.

In this paper we show that coin and owner tracing can also be implemented *off-line* without any trusted third party. Our payment system is based on [CMS96, CMS97], but replaces their trustee based tracing mechanism with the concept of *auditable tracing*: At a certain point of time customers can detect whether their spent coins have been traced or not and whether this tracing was performed with the permission of a judge or the customer himself. As every application of illegal tracing (i.e. tracing without permission) can be prosecuted, only legal tracing will be applied in practice. Thus, auditable tracing provides better protection of the customer's privacy than uncontrolled trusted third party based tracing.

Additionally, we support the self deanonymization mechanism of [PS01], which enables a blackmailed customer to always trace his blackmailed coins.

The remainder is structured as follows: In the next section we explain our idea for off-line auditable tracing. Section 3 presents the building blocks of our payment system, which is implemented in Section 4. The tracing mechanisms and the audit mechanism are described in Section 5. The general trade-off between tracing mechanisms and privacy is discussed in Section 6.

2 Off-line Auditable Tracing

In payment systems with passive trustees (e.g. [CMS96, DFTY97]) a single public tracing key exists, for which only the trustee knows the corresponding private tracing key. At withdrawal the customer is forced to use the public tracing key to encrypt some information identifying this withdrawal. This information can only be decrypted by the trustee, which enables to link the payment and withdrawal view of this coin. Thus, coin and owner tracing can be performed.

Our payment system is not based on trusted third party tracing, but also uses tracing keys for encrypting identifying information. In contrast to payment systems with passive trustees a different public tracing key for each customer and merchant is issued by the bank. All tracing keys have a limited validity period and must be exchanged regularly against new keys. The application of tracing is restricted as follows:

- Coin tracing is possible, if the bank knows the private tracing key for the customer, who withdraws the coin.

– Owner tracing is possible, if the bank knows the private tracing key for the merchant, who deposits the coin.

Whether or not the bank knows a private tracing key depends on the *key generation parameter* for this tracing key. The bank reveals the key generation parameter, after a tracing key has expired. This enables the customer or the merchant to check and even prove whether they have been traced or not.

2.1 Legal Application of Tracing

When the bank creates a tracing key, it issues a *user certificate* containing an identifier for the user, the public tracing key, the activation date, the expiration date, the audit date, a transaction limit, and a symmetric encryption of the key generation parameter. We distinguish two types of user certificates:

– The *customer certificate* can only be used to withdraw coins until it becomes invalid.
– The *merchant certificate* can only be used to deposit coins until it becomes invalid.

Both types of user certificates (and thus the tracing keys) are valid from the activation date until the expiration date or until the transaction limit has been reached. The transaction limit additionally restricts the usage of a certificate e.g. to the maximum amount of money that can be withdrawn/deposited or to the maximum number of withdrawals/deposits.

When the audit date is reached, the customer or merchant can ask the bank for the symmetric key that was used to encrypt the key generation parameters in the certificate. Then the user knows whether the bank was able to trace or not.

To determine whether a detected tracing was legal or illegal, the user can request a *tracing certificate* from the bank. This tracing certificate has to be issued by a judge, who has permitted tracing this user. The tracing certificate is simply the user certificate signed by the judge.

2.2 Audit of Tracing Keys

Due to security and efficiency coin generations are necessary [Sch97] and every coin has only a limited life time. The bank issues coins during a *generate phase* and accepts them for deposit during an *accept phase*. After the bank stops issuing coins of a generation, it immediately starts issuing coins of the next generation.

The user certificates are related to coin generations as follows:

– **Customer certificates:** The activation and expiration dates of a customer certificate must be during the generate phase of the same coin generation. The audit date of the customer certificate is always a certain time period Ω after the end of the accept phase of this coin generation. As the coins are no longer valid for payments, it is possible to reveal at the audit date, whether coin tracing has been applied to these coins or not.

– **Merchant certificates:** The activation and expiration dates of a merchant certificate must be during the accept phase of the same coin generation. The audit date of the merchant certificate is always a certain time period Δ after the expiration date of the merchant's certificate. As this merchant certificate is no longer valid for deposits, it is possible to reveal at the audit date, whether owner tracing has been applied to these coins or not.

The relation of user certificates to coin generations is shown in Figure 1. Coin tracing is guaranteed to be undetectable for at least the time period Ω, and owner tracing is at least undetectable for the time period Δ. The time periods Ω and Δ can be used for investigations of suspected criminal activities. The length of Ω and Δ can even be chosen differently for different users, e.g. a previously convicted person may have later audit dates.

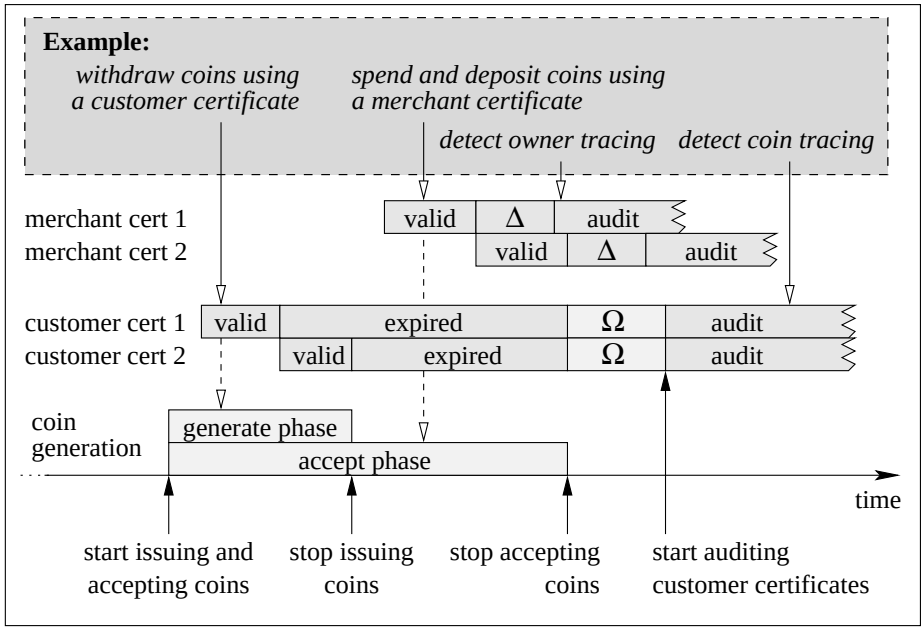

Figure 1. The relation between tracing keys and a coin generation.

2.3 Tracing Blackmailers

The tracing mechanism described so far has the property that if a customer is traced, the bank alone executes the coin tracing at withdrawal. If the customer is not traced, nobody can execute coin tracing later. However, if the customer is blackmailed, coin tracing will be desired by the customer. Thus, he has to instruct the bank to apply coin tracing at withdrawal. The customer may use

a distress mechanism to inform the bank about the blackmailing, as described in [KV01a]. The bank will now create a public tracing key for which it knows the corresponding private key. As a blackmailer will demand a huge amount of money, the amount limit of the current customer certificate will be reached, and the bank can apply the new tracing key for coin tracing. Thus it is guaranteed that the bank can trace the majority of the blackmailed coins with the new tracing key. These coins can be invalidated by blacklisting and even refunded to the customer, if the blackmailer has not spent them yet.

While this is a solution to the blackmailing problem, it depends on the customer to inform the bank about the blackmailing *before* the coins are withdrawn. If the customer did not dare to notify the bank, all the coins withdrawn by the blackmailer will usually be untraceable. A simple protection against this is a policy issued and signed by the customer in advance. This customer policy is used to identify suspicious transactions and instructs the bank to trace those coins. An example for such a customer policy is the withdrawal of more than \$500 per day, which will trigger tracing with the next tracing key.

Another approach is a self deanonymization mechanism proposed by Pfitzmann and Sadeghi [PS01]. There only the customer is able to trace his own coins in the case of blackmailing. We adopt this idea in a way that it does not restrict the tracing capabilities of the bank. Our approach has the following properties: If the bank has performed coin tracing at withdrawal, it can trace alone. Otherwise, tracing is possible only if the customer assists the bank. This kind of tracing is available to the customer at any time so that deanonymization and blacklisting of coins are possible afterwards.

However, if the customer fears that he may be forced to deanonymize himself with this tracing method (e.g. due to laws that demand to reveal decryption keys like the British Regulation of Investigatory Powers Act), he can provably renounce the ability to trace himself.

The three mechanisms described above provide a very good choice for the customer to protect himself against blackmailing. Depending on his preferences he can choose one or more of these mechanisms when he opens his bank account. However, the blackmailer must not be able to alter the chosen mechanisms. To change the customer's preferences for those mechanisms e.g. the customer himself has to appear at the bank or alternatively the change should take effect after a certain time period.

Note that we omit bank robberies [JY96] as generic countermeasures have been proposed against this attack [KV01b], which can also be applied to this payment system.

3 Building Blocks

All computations are done in a cyclic group G of prime order q, where q is chosen so that the discrete logarithm problem is infeasible. For every coin generation the bank computes new parameters for the payment system. The three generators g, g_1, and g_2 are created by a pseudo-random function, which ensures that nobody

knows the discrete logarithm between any of these generators. The bank chooses a private signature key $x \in_R \mathbb{Z}_q$ and computes the public keys $y = g^x$, $y_1 = g_1^x$, and $y_2 = g_2^x$. Then G, g, g_1, g_2, y, y_1, and y_2 are published.

3.1 Merchant Tracing Keys

To implement owner tracing every merchant is assigned a public tracing key y_M. The merchant receives this key from the bank, who also provides the merchant certificate for this key. The corresponding private tracing key is the discrete logarithm of y_M to the base g_1.

The bank chooses a secret value $x_M \in_R \mathbb{Z}_q^*$ to compute the public tracing key y_M. If the merchant is traced, the bank computes $y_M = g_1^{x_M}$ so that x_M is the private tracing key. If no owner tracing is wanted, y_M is computed as g^{x_M}. This guarantees that the bank does not know the discrete logarithm of y_M to the base g_1.

The merchant does not know whether y_M was computed as $g_1^{x_M}$ or g^{x_M}. Thus, he cannot determine whether he is traced or not. The bank encrypts x_M in the merchant certificate and will reveal the symmetric encryption key only after the audit date.

3.2 Customer Tracing Keys

When the customer opens his bank account, he has to register a public key g_C. If he wants to use the self deanonymization mechanism, he must know the discrete logarithm of g_C to the base g_2. Otherwise self deanonymization is impossible.

The customer chooses a secret value $x_s \in_R \mathbb{Z}_q^*$ and either computes $g_C = g_2^{x_s}$ to enable self deanonymization or $g_C = g^{x_s}$, which later enables the proof that he does not know the discrete logarithm of g_C to the base g_2.

To implement coin tracing the bank assigns the customer a public tracing key y_C. The customer receives this key from the bank, who also provides the customer certificate for this key. The corresponding private tracing key is the discrete logarithm of y_C to the base g_2.

The bank chooses a secret value $x_C \in_R \mathbb{Z}_q^*$ to compute the public tracing key y_C. If the customer is traced, the bank computes $y_C = g_2^{x_C}$ so that x_C is the private tracing key. If no coin tracing is wanted, y_C is computed as $g_C^{x_C}$. This guarantees that the bank does not know the discrete logarithm of y_C to the base g_2. However, if the customer has chosen $g_C = g_2^{x_s}$, the bank and the customer together know the discrete logarithm and can apply coin tracing.

The bank encrypts x_C in the customer certificate using symmetric encryption. Only when the symmetric key and thus x_C is revealed after the audit date, the customer can determine how y_C was created and whether his coins are traced or not. If the bank encrypts junk instead of x_C, this will be revealed at the audit date. As the customer certificate contains all necessary information, this fraud can be proven by the customer.

3.3 Proving Correct Usage of the Tracing Keys

The tracing keys are used to encrypt identifying information during withdrawal and payment. To prove the correctness of these encryptions we use the Chaum-Pedersen protocol [CP92], which is an extension of the Schnorr signature scheme [Sch91] and proves the equality of discrete logarithms. As the proof is computed non-interactively, we require a hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$.

The proof of equality of discrete logarithms $log_{a_1} b_1$ and $log_{a_2} b_2$ is called $PEQDL(a_1, b_1, a_2, b_2) = (c, s)$. This proof can only be given, if the prover knows the discrete logarithm $x = log_{a_1} b_1 = log_{a_2} b_2$. The prover chooses a secret $r \in_R \mathbb{Z}_q$ and computes:

$$c = H(a_1, b_1, a_2, b_2, a_1^r, a_2^r)$$
$$s = r - cx \bmod q$$

Then the verifier has to check, whether c is equal $H(a_1, b_1, a_2, b_2, a_1^s b_1^c, a_2^s b_2^c)$. It is also possible to compute a $PEQDL(m, a_1, b_1, a_2, b_2)$ dependant on a message m, if the hash value is computed as $H(m, a_1, b_1, a_2, b_2, a_1^r, a_2^r)$.

4 Implementation of the Payment System

In this section we describe the implementation of the our payment system and present the protocols for withdrawal, payment, and deposit. Our implementation is based on the off-line payment system of [CMS96, CMS97].

4.1 Withdrawal

For every coin the customer chooses a secret value $\alpha \in_R \mathbb{Z}_q^*$ that is used to compute $(h_p, z_p) = (g_1 g_2^\alpha, y_1 y_2^\alpha)$. This value α has to be escrowed at the bank as $d = y_C^\alpha$ using the customer tracing key y_C. Then the customer receives a blindly issued proof $PEQDL(g, y, h_p, z_p)$ from the bank that the pair (h_p, z_p) has the same discrete logarithm as (g, y). The following withdrawal protocol is also shown in Figure 2.

1. The customer transforms the pair (h_p, z_p) from the payment view to the withdrawal view $(h_w, z_w) = (h_p^{\alpha^{-1}}, z_p^{\alpha^{-1}})$.
2. The customer proves that the pair $((h_w/g_2), g_1)$ has the same discrete logarithm as (y_C, d). Therefore he sends the bank the proof $U = PEQDL((h_w/g_2), g_1, y_C, d)$, h_w, and d. The bank verifies the proof U and only proceeds, if this verification succeeds.
3. The bank and the customer interact in a protocol that blindly issues the proof that (h_w, z_w) from the bank's view respectively (h_p, z_p) from the customer's view has the same discrete logarithm as (g, y).
   - The bank randomly selects a secret $r' \in_R \mathbb{Z}_q$ and computes two commitments $t'_g = g^{r'}$ and $t'_h = h_w^{r'}$, which are sent to the customer.

- The customer chooses blinding factors β and γ and blinds the commitments to $t_g = t'_g g^\beta y^\gamma$ and $t_h = {t'_h}^\alpha h_p^\beta z_p^\gamma$. Then the customer computes the challenge $c = H(a, g, y, h_p, z_p, t_g, t_h)$, where a random $r \in_R \mathbb{Z}_q$ is used to create the coin number $a = g_2^r$. The challenge is blinded to $c' = c - \gamma \bmod q$ and sent to the bank.
- The bank signs this challenge by computing $s' = r' - c'x \bmod q$ and returns this value to the customer.
- The customer unblinds the response to $s = s' + \beta \bmod q$ so that (c, s) is a valid $PEQDL(a, g, y, h_p, z_p)$.

At the end of this withdrawal the customer stores r, a, α, h_p, z_p, c, and s, while the bank only needs to memorize h_w and d.

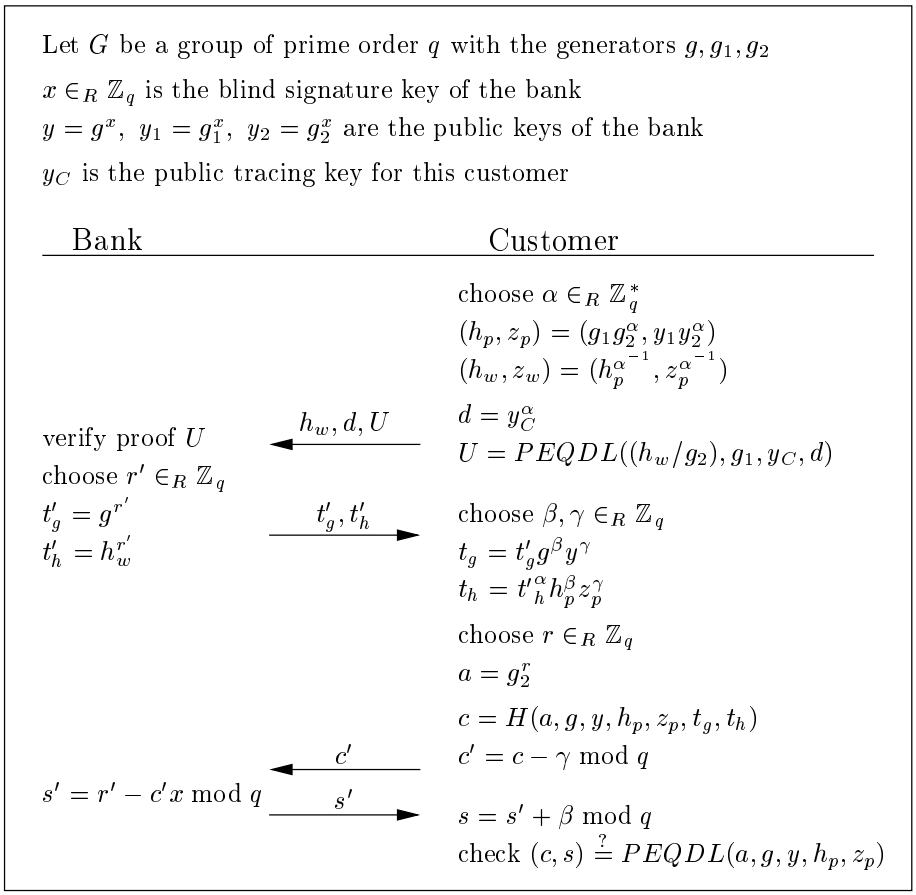

Figure 2. Withdrawal of a coin.

4.2 Payment

At payment the merchant sends the customer a unique value $m = (ID_{merchant}, counter)$ and his public tracing key y_M together with the corresponding merchant certificate of the bank. The customer checks this merchant certificate and proceeds, if it is valid and has not expired.

For every coin the following steps are executed (see also Figure 3):

1. The customer sends a, h_p, z_p, c, and s to the merchant, who verifies that (c, s) is a valid $PEQDL(a, g, y, h_p, z_p)$.
2. To enable owner tracing the customer has to compute the value $d' = y_M^{\alpha^{-1}}$. Then he proves that this value uses the same α as the values h_p from the $PEQDL$ of step 1: He computes $c_p = H(m, d', y_M, g_2, (h_p/g_1), d'^r, a)$ and $s_p = r - c_p\alpha \bmod q$ which are a valid $PEQDL(m, d', y_M, g_2, (h_p/g_1))$. Then d', c_p, and s_p are sent to the merchant.
3. The merchant checks the validity of this $PEQDL(m, d', y_M, g_2, (h_p/g_1))$ and additionally verifies that the same $a = g_2^r$ as committed in the coin's c was used to compute c_p. This forces the customer to use the fixed commitment a so that spending a coin twice will reveal α (see section 5.1).

Finally, the merchant stores a, h_p, z_p, c, s, m, c_p, s_p, and d'.

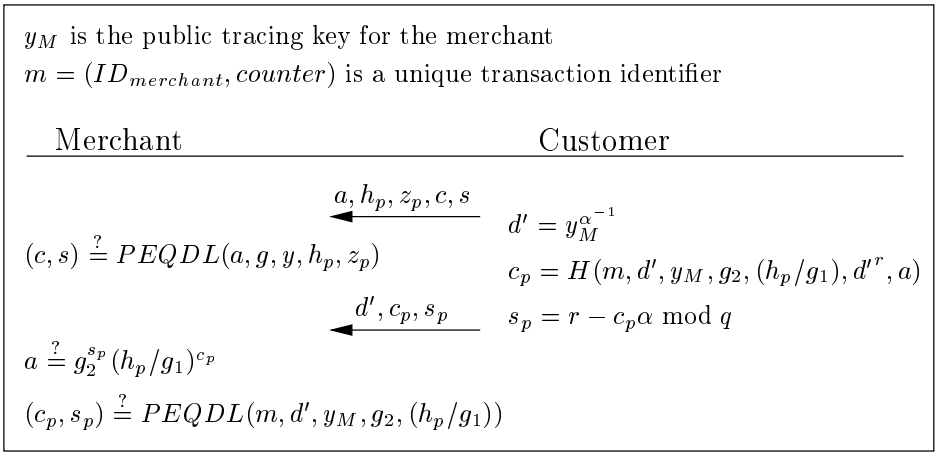

Figure 3. Off-line payment with a coin.

4.3 Deposit

The merchant forwards all stored values to the bank, who does the same verifications as the merchant during the payment. Note, that the bank has to check everything, even if the merchant is not traced and recovering the withdrawal view

for this coin is not possible. Furthermore, the bank has to ensure that the used y_M belongs to the depositing merchant and that the corresponding merchant certificate is still valid.

5 Tracing and Audit

We describe all tracing methods provided by our payment system and show how coin and owner tracing can be audited.

5.1 Tracing Double-spenders

At payment the customer has to compute $(c_p, s_p) = PEQDL(m, d', y_M, g_2, (h_p/g_1))$, where he is forced to always use $a = g_2^r$ as the commitment. If he does this twice, the bank receives

$$s_p = r - c_p\alpha \bmod q$$
$$s'_p = r - c'_p\alpha \bmod q$$

from the double-spender. As s_p, s'_p, c_p, and c'_p are known, the bank simply computes $\alpha = (s_p - s'_p)/(c'_p - c_p) \bmod q$. Now the bank looks up, who has withdrawn the coin with $h_w = g_1^{\alpha^{-1}} g_2$, and accuses this person of double-spending. The bank cannot falsely claim that somebody is a double-spender, as it can only compute α after a double-spending.

5.2 Coin and Owner Tracing

Coin tracing is only possible at withdrawal, while owner tracing can only be performed at payment.

- **Coin Tracing:** In the case of coin tracing the customer is traced. The bank knows the private tracing key x_C, which is the discrete logarithm of $y_C = g_2^{x_C}$ to the base g_2. Thus it can "decrypt" $d = y_C^\alpha$ by computing $g_1 d^{x_C^{-1}} = g_1 g_2^\alpha = h_p$. This value reveals the payment view of the coin so that the coin is recognized at deposit.
- **Owner Tracing:** In the case of owner tracing the merchant is traced. The bank knows the private tracing key x_M, which is the discrete logarithm of $y_M = g_1^{x_M}$ to the base g_1. Thus it can "decrypt" $d' = y_M^{\alpha^{-1}}$ by computing $d'^{x_M^{-1}} g_2 = g_1^{\alpha^{-1}} g_2 = h_w$. This value reveals the withdrawal view of the coin so that the identity of the withdrawer can be looked up.

5.3 Self Deanonymization

Self deanonymization is a special form of coin tracing. It can only be applied, if the customer knows x_s, the discrete logarithm of $g_C = g_2^{x_s}$ to the base g_2.

1. The customer requests the value $d = y_C^{\alpha}$ from the bank, which belongs to one of his withdrawn coins that has to be deanonymized.
2. He removes his secret exponent by computing $d^{x_s^{-1}}$ and returns this value to the bank. To prevent the customer from cheating, he must prove the correctness of his computation by providing the $PEQDL(d^{x_s^{-1}}, d, g_2, g_C)$.
3. If the customer sends the correct value, the bank computes h_p:
   - If the customer has not been traced, then $y_C = g_C^{x_C}$ and the bank computes $g_1(d^{x_s^{-1}})^{x_C^{-1}} = g_1 g_2^{\alpha} = h_p$.
   - If the customer has been traced, then $y_C = g_2^{x_C}$ and the bank computes $g_1 d^{x_C^{-1}} = g_1 g_2^{\alpha} = h_p$.

   If the coin has not been deposited, the bank can blacklist the coin with this value h_p. Otherwise, the bank can look up which merchant deposited the coin.

5.4 Audit

For every audit date the bank publishes a single symmetric key that has been used to encrypt the key generation parameters x_C and x_M in the customer and merchant certificates. Now every customer and merchant with a user certificate of this audit date can detect, whether his certificate was issued for tracing. The published symmetric key enables to decrypt the key generation parameters x_C and x_M so that tracing can be audited as follows:

- Coin tracing is detected by the customer, if y_C is not equal to $g_C^{x_C}$.
- Owner tracing is detected by the customer or merchant, if y_M is not equal to g^{x_M}.

If tracing has been detected, the customer or merchant can request a tracing certificate from the bank. If the bank cannot provide a tracing certificate, the user certificate can be given to a judge. This user certificate has been signed by the bank and contains all necessary information to prove the application of tracing. The judge checks whether tracing indeed has been applied and whether this tracing was legal. The judge denounces any detected illegal tracing.

6 Tracing Capabilities and Privacy

In this section we don't consider payment systems without deanonymization, because they allow misuse of anonymity. For deanonymization two kinds of tracing mechanisms exist:

Self deanonymization: The deanonymization process is under the control of the withdrawer. Thus, tracing can only be used against blackmailers.

Enforced deanonymization: The deanonymization process cannot be influenced by the withdrawer. Thus, tracing can be used for any kind of investigations.

Both mechanisms significantly differ in their anonymity guarantees. Only self deanonymization can provide unconditional anonymity [KV01a] or at least computational anonymity [PS01].

If enforced deanonymization is desired, only weaker anonymity can be guaranteed. Payment systems with a trustee concept (e.g. [CMS96,FTY96,DFTY97, JY96]) have the best tracing capabilities, as tracing is always possible. However, the level of anonymity is uncertain, because tracing is undetectable for the customer and even if tracing has not been applied yet, it still can be applied in the future.

The solution to this problem is the audit concept: Tracing is only possible at either withdrawal or payment and can be detected by the customer after the audit date. Thus, the level of anonymity is only unknown till the audit date. Afterwards an untraced coin remains either unconditional anonymous [KV01b, KV01c] or computational anonymous, which we have presented in this paper. The advantage of this reduction to computational anonymity is the possibility of off-line payments. Furthermore self deanonymization can be applied at any time, as it is not restricted to the withdrawal.

Compared to the directly related off-line payment systems [CMS96,DFTY97] and [PS01] our new solution is clearly superior, as we provide both, a high level of anonymity and at the same time strong deanonymization mechanisms.

7 Conclusion

We have presented the first off-line payment system that provides auditable tracing. In our payment system the bank alone has the power to perform coin and owner tracing. However, the application of tracing is restricted, as the customers and merchants can later audit their payments and detect every deanonymization.

In an alternative implementation of our payment system all the tracing capabilities are transferred to a *tracing authority*, who is responsible for deanonymizations and issues tracing keys and user certificates. The advantage of this implementation is that the tracing capabilities are completely separated from the payment functionality, which allows the bank to focus on the payment system. Anyway, the tracing authority need not be trusted, as the audit concept will reveal any misuse of tracing.

Compared to previous payment systems based on the audit concept, we additionally provide off-line payments and a self deanonymization mechanism that allows the customer to trace his own coins at any time. The self deanonymization is especially useful, if the customer was blackmailed, because the decision about tracing the blackmailed coins is not restricted to the withdrawal.

References

[Cha83] D. Chaum. Blind signatures for untraceable payments. In *Advances in Cryptology – CRYPTO '82*, pages 199–203. Plenum, 1983.

[CMS96] J. Camenisch, U. Maurer, and M. Stadler. Digital payment systems with passive anonymity-revoking trustees. In *Computer Security – ESORICS '96*, volume 1146 of *Lecture Notes in Computer Science*, pages 31–43. Springer-Verlag, 1996.

[CMS97] J. Camenisch, U. Maurer, and M. Stadler. Digital payment systems with passive anonymity-revoking trustees. *Journal of Computer Security*, 5(1), 1997.

[CP92] D. Chaum and T. P. Pedersen. Wallet databases with observers. In *Advances in Cryptology – CRYPTO '92*, volume 740 of *Lecture Notes in Computer Science*, pages 89–105. Springer-Verlag, 1992.

[DFTY97] G. Davida, Y. Frankel, Y. Tsiounis, and M. Yung. Anonymity control in e-cash systems. In *Financial Cryptography – FC '97*, volume 1318 of *Lecture Notes in Computer Science*, pages 1–16. Springer-Verlag, 1997.

[FTY96] Y. Frankel, Y. Tsiounis, and M. Yung. "Indirect discourse proofs": Achieving efficient fair off-line e-cash. In *Advances in Cryptology – ASIACRYPT '96*, volume 1163 of *Lecture Notes in Computer Science*, pages 286–300. Springer-Verlag, 1996.

[JY96] M. Jakobsson and M. Yung. Revokable and versatile electronic money. In *3rd ACM Conference on Computer and Communications Security – CCS '96*, pages 76–87. ACM Press, 1996.

[KV01a] D. Kügler and H. Vogt. Marking: A privacy protecting approach against blackmailing. In *Public Key Cryptography – PKC 2001*, volume 1992 of *Lecture Notes in Computer Science*, pages 137–152. Springer-Verlag, 2001.

[KV01b] D. Kügler and H. Vogt. Fair tracing without trustees. In *Financial Cryptography – FC 2001*. Preproceedings, 2001.

[KV01c] D. Kügler and H. Vogt. Auditable tracing with unconditional anonymity. In *Proceedings of the 2nd International Workshop on Information Security Applications (WISA 2001)*, pages 108–120, Seoul, Korea, 2001.

[PS01] B. Pfitzmann and A.-R. Sadeghi. Self-escrowed cash against user blackmailing. In *Financial Cryptography – FC 2000*, volume 1962 of *Lecture Notes in Computer Science*, pages 42–52. Springer-Verlag, 2001.

[Sch91] C.P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, 1991.

[Sch97] B. Schoenmakers. Security aspects of the ecash payment system. In *COSIC '97 Course*, volume 1528 of *Lecture Notes in Computer Science*, pages 338–352. Springer-Verlag, 1997.

[SPC95] M. Stadler, J.-M. Piveteau, and J. Camenisch. Fair blind signatures. In *Advances in Cryptology – EUROCRYPT '95*, volume 921 of *Lecture Notes in Computer Science*, pages 209–219. Springer-Verlag, 1995.

[STS99] T. Sander and A. Ta-Shma. Auditable, anonymous electronic cash. In *Advances in Cryptology – CRYPTO '99*, volume 1666 of *Lecture Notes in Computer Science*, pages 555–572. Springer-Verlag, 1999.

[vSN92] B. von Solms and D. Naccache. On blind signatures and perfect crimes. *Computers and Security*, 11(6):581–583, 1992.