U.C. Berkeley
CS276: Cryptography
Professor David Wagner

Lecture 28
April 27, 2006
Scribe: Scott Monasch

Lecture 28: Applications of Crypto Protocols

1 Electronic Payment Protocols

For this section we are considering the scenario that Alice wants to pay Bob money (electronically). Several models were suggested for accomplishing this task:

- **PayPal** Alice establishes a secure channel to the bank, using any of the primitives we've seen, and tells the bank she wants to transfer $100 to Bob's account. Bob can then contact the bank and verify that his account has the money in it.
- **Check** This model is analogous to paper checks: Alice signs a statement requesting a transfer of money to Bob's account, and gives the signed statement to Bob. Bob can then later give the statement to the bank, and the bank will check that the statement is signed by Alice, and give Bob the money. Electronically, this is done with public key signatures. Alice, when she opens her account, tells the bank her public key. To write a check to Bob, she does: $\mathsf{sign}(K_A, \text{“Please transfer \$100.00 to Bob (seq \#101)”})$. The statement is signed with her private key, K_A, and also includes a sequence number to prevent Bob from depositing the same check more than once. This scenario allows transactions to be performed offline, which may be useful if there is no way to connect directly to the bank at the time. The disadvantage is that if Bob doesn't deposit the check immediately, he can't be sure it is valid.
- **Credit Card** To pay Bob, Alice provides her account number to Bob, and the amount of the purchase. Bob goes to the bank with Alice's account number and says “Alice wants me to have $100.” The bank gives him $100. This protocol is completely insecure from a technical viewpoint.

This lecture focuses on protocols where we can achieve security by technical means.

2 Security Goals

These are some of the security goals we might want to have for an electronic payment system:

- **Integrity** Protection against fraud, no one can steal money.
- **Payment verification** Recipient can verify payment validity.
- **Conservation of money**
- **Dispute Resolution** If there is a dispute later over whether the money or goods were received, there is a way to arbitrate it.
- **Privacy/Anonymity** There are several goals we may want to achieve in terms of privacy and anonymity, including:

1. Alice doesn't know Bob's identity.
2. Bob doesn't know Alice's identity.
3. Alice and the bank together can not learn Bob's identity (payee anonymity)
4. Bob and the bank together can not learn Alice's identity (payer anonymity)

In thinking about privacy, checks are not the best metaphor to use, since they must identify whose account should be debited and whose should be credited. The typical metaphor used is instead cash, and in particular, coins. A coin has no history of who withdrew it from the bank, so it inherently has some strong anonymity properties. What we would like is an electronic equivalent of a coin, which has the same privacy properties as a coin, in addition to the integrity properties and other security goals we wish to achieve.

To solve this problem, one might imagine a protocol such as the following. This protocol will not fully work, it won't provide very strong privacy, but it's a starting point.

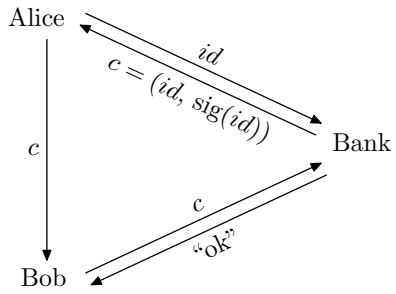


Figure 1: Strawman protocol

In this scheme, assume every coin is worth one cent. First, Alice will withdraw a coin from the bank. Her account is debited one cent, and she ends up with a coin, which is a bearer instrument; that is, it's just bits, and anyone who knows those bits can initiate a protocol to deposit it. She gives the coin to Bob, who later deposits it.

In the strawman scheme, we have to be able to tell one coin from another, so each must have a unique identifier. For Alice to withdraw a coin, she supplies a unique *id*, which may be, say, a random 128-bit number. The bank responds with a signature on the *id*, created using the bank's private key. A valid coin consists of a pair $id, \text{sig}(id))$, and anyone can check that the signature is valid using the bank's public key. Alice can at any time in the future give the coin to Bob. Bob gives it to the bank, which verifies that the coin contains the bank's signature, and credits Bob's account one cent.

From a privacy point of view, there are some limits on what privacy this scheme can provide. The bank learns everything about all transactions that happen in the system, since every coin has a unique identifier. So the bank can determine that Alice paid the money to Bob.

But this scheme has a larger security hole. Alice still has the coin after giving it to Bob, so she can give it to someone else, say, Carol, and Carol can also redeem it. One way around this problem is for the bank to keep a list of the coins it has redeemed, and only accept each *id* once. This protects the bank from double spending of a coin, but it does nothing for Bob. He can not ensure that a coin has any worth, unless he actually deposits it. Otherwise, someone else could deposit the coin

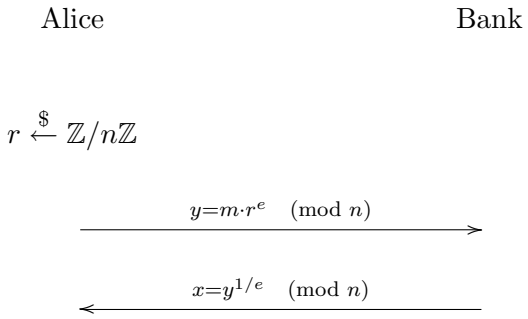
first and make it worthless to him. So in this scheme, Bob must deposit the coin before he gives Alice what she is paying for (e.g. an MP3).

If we assume all communication in this scheme is done over secure channels, we ensure that no eavesdropper can steal a coin. If the bank is trusted, there is privacy, but there are severe restrictions on how money can be spent. Bob can not hold a coin and use it to pay for something else, he must deposit it immediately.

Next we will look at David Chaum's "e-cash" scheme, which builds on this protocol, and provides anonymity for Alice. Even if Bob and the bank are colluding they can not find out where a coin came from. Chaum's protocol changes the withdrawal phase by using blind signatures. A blind signature allows Alice to get the bank's signature on an *id*, without the bank knowing what it's signing. The bank knows it's signing something, but does not learn the value of the *id*. It does know it's signing something for Alice, so it can debit Alice's account.

3 Blind Signatures

Blind signatures are a primitive that are interesting in their own right. In blind signatures we have 2 parties (we'll use the same names as before, Alice and the Bank). For this example, we will see blind signatures using RSA. The bank has a public key (n, e), and a private key d. Alice has a message m she wants signed. She wants a signature of m under the bank's key, but she doesn't want the bank to learn m.



Alice computes $\frac{x}{r} = m^{1/e} \pmod{n}$, which is the signed message. Furthermore, the bank has learned nothing, assuming e is chosen properly. In RSA, e is chosen to be relatively prime to $\varphi(n)$, so the map $r \mapsto r^e$ is a bijection and thus r^e has a uniformly random distribution, independent of m. What the bank sees is random and uniform, so even with unlimited computing power it can gain no advantage.

4 E-cash

In David Chaum's e-cash system, a coin is a pair (u, v) such that v is a full-domain hash RSA signature on u, that is, $H(u) = v^e \pmod{n}$. Assume for simplicity now that $e = 3$. We now have just a slight change from the strawman scheme.

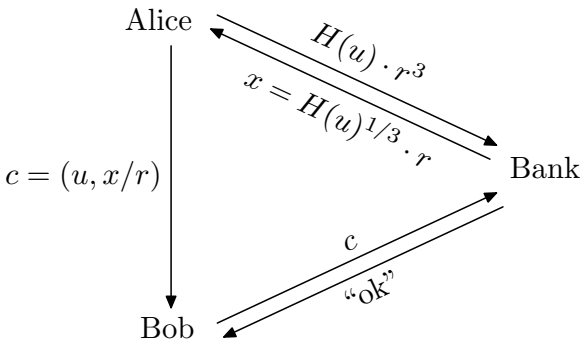


Figure 2: e-cash protocol

This scheme is payer-anonymous. The bank can not tell anything about the identity of Alice, even by colluding with Bob. Of course if Alice is the only one who has withdrawn a coin, she can be traced, but in general others will also withdraw coins before she spends it, and of those people it is impossible for Bob and the Bank to tell which one spent the coin. A similar issue arises if Alice is the only one who has withdrawn enough coins to make a certain payment to Bob (or is one of a small group, in which case Bob and the Bank can narrow down who she might be.)

The scheme is not payee anonymous. If Alice gives a coin to Bob, and Bob deposits the coin, Alice and the Bank can together learn Bob's identity. Alice knows the coin's identifier u, so the Bank can check who deposited that coin. Chaum considered this a feature of the system rather than a flaw. It protects against the selling of illegal items. Authorities can simply make one purchase from Bob, and then find out who he is.

Next we will explore ways to extend this scheme to provide any desired level of anonymity or accountability. First we will make a change that does not alter any of the anonymity properties, but just allows Bob to choose the identifier u rather than Alice.

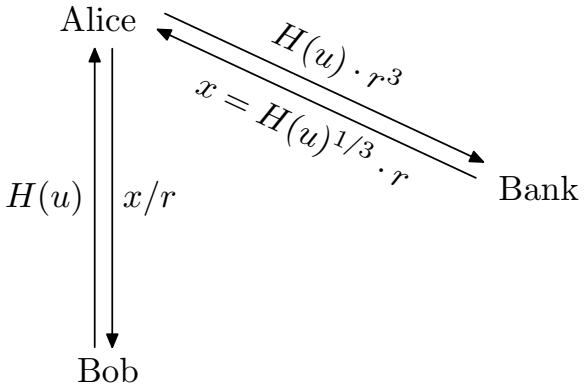


Figure 3:

Bob sends Alice a hash of u, which she blinds and gets signed by the bank, then she computes the signature of $H(u)$ and returns it to Bob. Bob is the only one who knows u, so he can now safely

defer depositing his coin. But Alice and the Bank can still track Bob because they both know the signature of $H(u)$, which is x/r.

The next scheme is a slight tweak which achieves payee anonymity, but not payer anonymity. In this scheme Bob chooses both u and r.

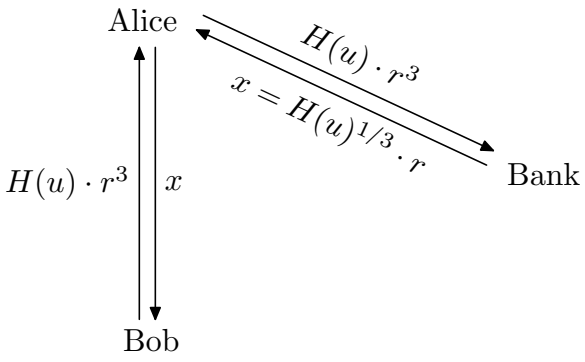


Figure 4:

Bob sends $H(u) \cdot r^3$ to Alice, who simply gets it signed by the Bank and returns it to Bob. Bob's coin is $(u, x/r)$. What Alice is doing now is essentially providing a service by which she will offers to sign any number, in exchange for Bob's goods. Alice's account gets debited, and Bob gets a coin which he can deposit at any time. Bob is completely anonymous now; he is the only one who knows either of the values in the coin. But Bob and the Bank can easily determine Alice's identity, since Alice is passing a value directly from Bob to the Bank.

The next scheme is fully anonymous. It is similar to the previous one, but now Alice picks a value $s \xleftarrow{\$} \mathbb{Z}/n\mathbb{Z}$ which she uses to blind what she sends to the bank.

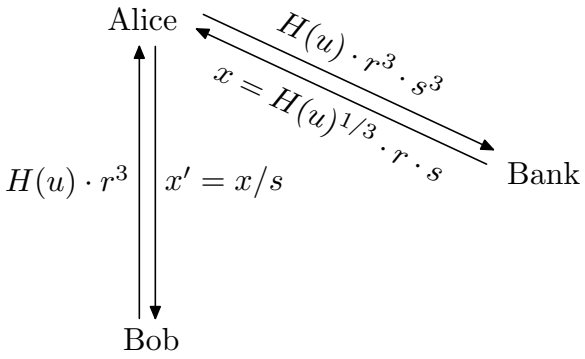


Figure 5: Fully anonymous protocol

Bob's coin is $(u, x'/r)$. Bob is still the only one who knows u and r, so he can not be traced. Alice is the only one who knows s, so everything she gives to the Bank and Bob appears uniformly random to them, and her identity is safe as well. This scheme is both payer- and payee-anonymous.

Another way to achieve anonymity is by the use of a moneychanger. A moneychanger is a con-

struction which allows anyone to exchange one coin for another, where the new coin can not be linked to the person. It follows the same principles as the previous scheme.

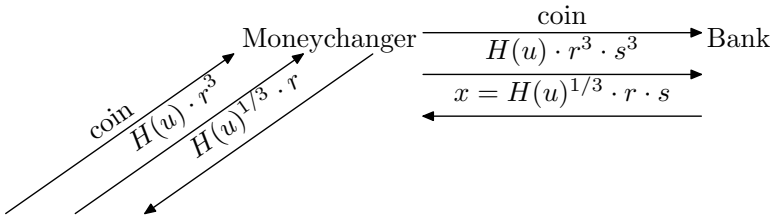


Figure 6: Moneychanger

The moneychanger takes as input a coin and a value, which we will assume is $H(u) \cdot r^3$. The moneychanger reblinds the value with s, and pays the bank one coin to sign the reblinded value. From the signed value, it computes $H(u)^{1/3} \cdot r$ and returns it to the user. The user now has a new coin, which is untraceable for the same reasons as in the fully anonymous scheme.