IT486 v3.0

Bitcoin motivation, recap of crypto hash and signatures

Pre-course questions

1. Have you taken Introduction to Cryptography (IT325) ?
2. Can you write Python code?

What we will cover

- Technological aspects
  - Bitcoin mining, consensus, 51% attack, hard fork, silkroad, altcoin, proof of stake, permissioned Blockchain, etc.
- Not finance aspects of cryptocurrencies

Course Material

- Slides/lectures are the most important!
- Textbook
  - Bitcoin and Cryptocurrency Technologies, Arvind Narayanan, Joseph Bonneau, Edward W. Felten, Andrew Miller, Steven Goldfeder, Jeremy Clark
  - http://bitcoinbook.cs.princeton.edu/
- Selected papers and references

Grading

- Midsem exam: 16%
- Endsem exam: 32% (cumulative)
- Term paper: 20%
- Assignments: 32%

Traditional online payments

- Alice and Bob have accounts at some institution like a credit card company, paypal or a bank
- In order to pay Bob digitally, Alice sends a request to the bank
- Bank checks that Alice has enough in her account to pay Bob
- The bank then debits Alice's account and credits Bob's account (money transfer)

Pros/cons of banks

- Pros
  - Digital payments

Pros/cons of banks

- Pros
  - Digital payments
- Cons
  - Not peer-to-peer (bank must be online during every transaction)

Pros/cons of banks

- Pros
  - Digital payments
- Cons
  - Not peer-to-peer (bank must be online during every transaction)
  - Bank can censor withdrawals and deposits

Digital money (ecash)

- Digital tokens representing money that we can pass back and forth
- With digital tokens, new forms of cheating become possible

Digital money (ecash)

- Digital tokens representing money that we can pass back and forth
- With digital tokens, new forms of cheating become possible
  - I can re-use the same coin token in different transactions $\rightarrow$ spend twice!

Digital money (ecash)

- Digital tokens representing money that we can pass back and forth
- With digital tokens, new forms of cheating become possible
  - I can re-use the same coin token in different transactions $\rightarrow$ spend twice!
- How can we prevent a "double-spend"? (More on this later.)

1990s Ecash

- First serious digital money proposal by David Chaum
- Pros
  - Peer-to-peer
  - Double spending detection
  - Privacy

1990s Ecash

- First serious digital money proposal by David Chaum
- Pros
  - Peer-to-peer
  - Double spending detection
  - Privacy
- Cons
  - Bank needed to avoid double-spending

1990s Ecash

- First serious digital money proposal by David Chaum
- Pros
  - Peer-to-peer
  - Double spending detection
  - Privacy
- Cons
  - Bank needed to avoid double-spending
  - Bank can censor withdrawals and deposits

Centralization leads to control

- Banks can:
  - Freeze assets
  - Deny transactions
  - Enjoy a monopolistic/oligarchic market
  - Know everything we do

Bitcoin

- 2009: Bitcoin announced by Satoshi Nakamoto (pseudonym)
  - Not linked to any fiat currency, i.e. currency issued by a government
- Bitcoin supports user–to–user transactions without needing a trusted arbiter
- The transactions are verified by the participants themselves; everyone shares the same transaction log (ledger)

What is Blockchain?

- A blockchain is an append-only data structure that records information in history
  - Append-only – items can be added, but cannot be changed or removed at any time
- In Bitcoin, the blockchain is a log of all transactions till date

Why is blockchain important for Bitcoin?

- To prevent fraudulent transactions
- Prevent people from creating their own coins
- Keep people from reversing transactions or double-spending their currency

What are hash functions?

- Definition: A function h is called a hash function if:
  - Compression: h maps an input x of arbitrary finite bit length to an output $h(x)$ of fixed bit length n:

$$h : \{0,1\}^\star \to \{0,1\}^n$$

  - Ease of computation: Given x and h it is easy to compute $h(x)$

What are hash functions?

- Definition: A function h is called a hash function if:
  - Compression: h maps an input x of arbitrary finite bit length to an output $h(x)$ of fixed bit length n:

$$h : \{0,1\}^\star \to \{0,1\}^n$$

  - Ease of computation: Given x and h it is easy to compute $h(x)$
- What are further desirable properties of cryptographic hash functions?

Additional properties for cryptographic hash functions

- Definition: Pre-image resistance
  - h is a hash fuction
  - for essentially all pre-specified outputs y, it is computationally infeasible to find an x such that $h(x) = y$.
  - h is also called a one-way function

Additional properties for cryptographic hash functions

- Definition: 2nd Pre-image resistance
  - Given x, it is computationally infeasible to find any second input x' with $x \neq x'$ such that $h(x) = h(x')$.

Additional properties for cryptographic hash functions

- Definition: Collision resistance
  - A function h is said to be collision-resistant if it is infeasible to find two values, x and y, such that $x \neq y$, yet $h(x) = h(y)$.

A bad hash function

Suppose we define a hash function h as follows (n is public):

$$h(m) = m \pmod{n}.$$

Which properties hold for h?

A bad hash function

Suppose we define a hash function h as follows (n is public):

$$h(m) = m \pmod{n}.$$

Which properties hold for h?

- pre-image resistance

A bad hash function

Suppose we define a hash function h as follows (n is public):

$$h(m) = m \pmod{n}.$$

Which properties hold for h?

- pre-image resistance
- second pre-image resistance

A bad hash function

Suppose we define a hash function h as follows (n is public):

$$h(m) = m \pmod{n}.$$

Which properties hold for h?

- pre-image resistance
- second pre-image resistance
- collision resistance

The SHA family

A family of standardized hash functions by NIST

- Message digest 4 / 5 (MD 4/5) Considered broken!
- Secure Hash Algorithm 1 (SHA-1) Considered broken!
- Secure Hash Algorithm 2/3 (SHA-2/SHA-3) At the moment safe to use

Review Q

Which of the following is true of SHA-256?

Review Q

Which of the following is true of SHA-256?

1. It has been proven that there is no fast way to find collisions

Review Q

Which of the following is true of SHA-256?

1. It has been proven that there is no fast way to find collisions
2. No collision has ever been publicly found

Review Q

Which of the following is true of SHA-256?

1. It has been proven that there is no fast way to find collisions
2. No collision has ever been publicly found
3. It has been proved not to have a collision

Review Q

Which of the following is true of SHA-256?

1. It has been proven that there is no fast way to find collisions
2. No collision has ever been publicly found
3. It has been proved not to have a collision
4. We hope that there are no collisions

Digital signatures – Overview

- Based on asymmetric cryptography algorithms like RSA or ECC
- We need two properties of digital signatures to hold in the digital world:
  - Only an entity is able to create a signature of its own, but everyone can verify it.
  - This signature is tied to data that gets signed. A signature cannot be used for different data.

Definition: Digital Signature Scheme

- Three algorithms
  - (sk,pk) := generateKeys(keysize)
    - sk is the secret key and is used to sign messages. pk is the public key and is given to everyone. With the pk, they can verify the signature.
  - sig := sign(sk, message)
    - the sign method takes the message and the secret key, sk, as input and returns a signature for message under sk

Definition: Digital Signature Scheme

- Three algorithms
  - (sk,pk) := generateKeys(keysize)
    - sk is the secret key and is used to sign messages. pk is the public key and is given to everyone. With the pk, they can verify the signature.
  - sig := sign(sk, message)
    - the sign method takes the message and the secret key, sk, as input and returns a signature for message under sk
  - isValid := verify(pk, message, sig)
    - The verify method takes a message, a signature, a public key as input. It will return true if the signature was generated out of the message and the secret key, otherwise false.

Definition: Digital Signature Scheme

- Three algorithms
  - (sk,pk) := generateKeys(keysize)
    - sk is the secret key and is used to sign messages. pk is the public key and is given to everyone. With the pk, they can verify the signature.
  - sig := sign(sk, message)
    - the sign method takes the message and the secret key, sk, as input and returns a signature for message under sk
  - isValid := verify(pk, message, sig)
    - The verify method takes a message, a signature, a public key as input. It will return true if the signature was generated out of the message and the secret key, otherwise false.
  - Such that verify(pk, message, sign(sk, message)) == true and signatures are unforgeable

What does unforgeable mean?

- The attacker knows your public key pk
- The attacker sees your signature sig on an arbitrary amount of messages
- Unforgeable means, that the attacker is not able to create a signature on a message that he has not seen

Digital signatures – Algorithms

Two major digital signature schemes are available:

- RSA-based signature schemes
  - Invented 1997 by Rivest, Shamir and Adleman
  - Based on the assumption that the factorization of large composite integers is very hard, but easy with additional information (so called trapdoor one-way functions)
- ECC-based signature schemes
  - Suggested independently by Neal Koblitz and Victor Miller in 1985
  - Based on discrete logarithms