Payment systems

Tuomas Aura
CSE-C3400 Information security

Aalto University, autumn 2014

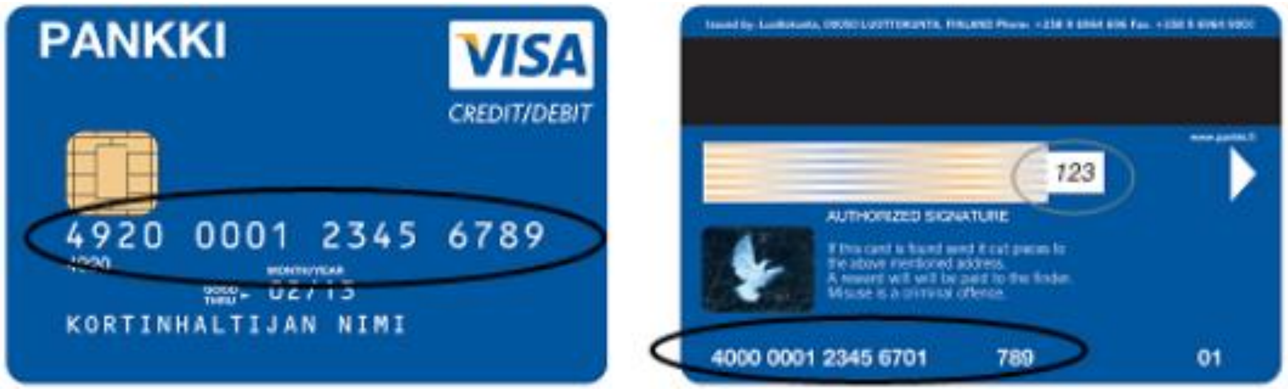
Outline

1. Card payments
2. Anonymous payments and BitCoin

CARD PAYMENT

Bank cards

- Credit or debit card
  - Card number, card holder, expiration date, CVV2
  - Magnetic stripe
  - Chip in integrated circuit card (ICC)
  - Contactless (NFC) interface
  - Card holder signature
  - Hologram
- PIN
- Terminals
  - Point of sale (POS)
  - Automated teller machine (ATM) = cash machine



[Picture: www.korttiturvallisuus.fi, Nets Oy]

Historical mag-stripe bank cards

- Magnetic stripe contains card number, holder name, expiration date, service code, PVKI, PVV, CVV1
- CVV1 is a cryptographic MAC of the PAN, name, expiration and service code (based on 3DES)
  → It is possible to copy but not change the mag stripe data
- PIN is a function of data on mag stripe and a secret key
  → offline PIN verification at disconnected POS or ATM
- Offline terminals have a security module to store the card and PIN verification keys
- Service code: e.g. Visa Electron 121 where 2=online only
- CVV2 to make online fraud harder
  – 3-4 digits printed on card but not on mag stripe
  – Required for card-not-present transactions (web and phone)
  – Verified but not stored by merchant → safe from server hacking
  – Vulnerable to phishing, though

Mag-stripe Visa PIN verification

- Input from magnetic stripe:
  - Primary account number (PAN) i.e. 15-digit card number
  - PIN verification key indicator (PVKI, one digit 1..6)
  - PIN verification value (PVV, 4 decimal characters)
- Verifier must have
  - PIN verification key (PVK, 128-bit 3DES key)
  - PVKI is an index of PVK to enable key updates for PVK
- Create security parameter (TSP):
  1. Concatenate 11 rightmost digits of PAN, PVKI and PIN
  2. The 16-digit concatenation is one hexadecimal DES block
- PVV generation:
  1. 3DES encryption of TSP with the key PVK
  2. Decimalization of the encryption result to 4-digit PVV
- Decimalization happens by taking the 4 leftmost digits 0..9 from the hexadecimal encrypted block
  - If less than 4 such digits, take 4 first digits A..F and map A=0,B=1,C=3...

[For details see IBM]

Chip-and-PIN bank cards

- EMV standard (Europay, Mastercard, Visa)
- Smart card chip (ICC) on the bank card
  - Tamperproof ICC stores a cryptographic (RSA) signature key
  - Card also contains a certificate
- Online vs. offline PIN verfication
  - Online: PIN sent to card issue for verification
  - Offline used mainly for credit, online for debit
- Online vs. offline authorization
  - Cash withdrawal from ATM always online
  - Some cards, e.g. Visa Electron always online

Offline transactions

- Three levels of secure offline transactions:

1. Static data authentication (SDA):
   - Certificate verification only; no longer used in Finland because certificate can be copied
2. Dynamic data authentication (DDA):
   - Card signs a random challenge sent by terminal with RSA
   - Currently main offline payment method
3. Combined DDA and application cryptogram (CDA):
   - Card signs transaction details incl. random challenge

- Card holder authenticated with PIN or signature
  - PIN usually sent to the card, which answers yes/no
- Offline risk parameters on the card limit offline transactions

Contactless (NFC) payment

Picture: visa.ca

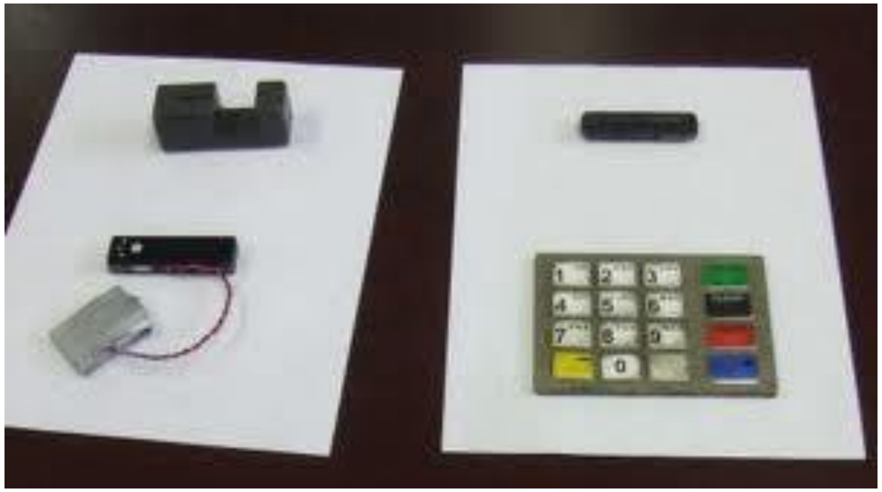
- Fast DDA (fDDA)
  - optimized signed message for contactless transactions
- No PIN verification
- Risk parameters for maximum offiline use
  - After a certain number of transactions and total amount of money spent, an online contact transaction with PIN is required
  - Soft and hard limits: after soft limit, online transaction is preferred but not required

EMV security issues

- Not possible to copy the chip
- Mag stripe can still be copied

  → Possible to create a copy of the mag stripe: use in the USA or as the fallback method after chip failure

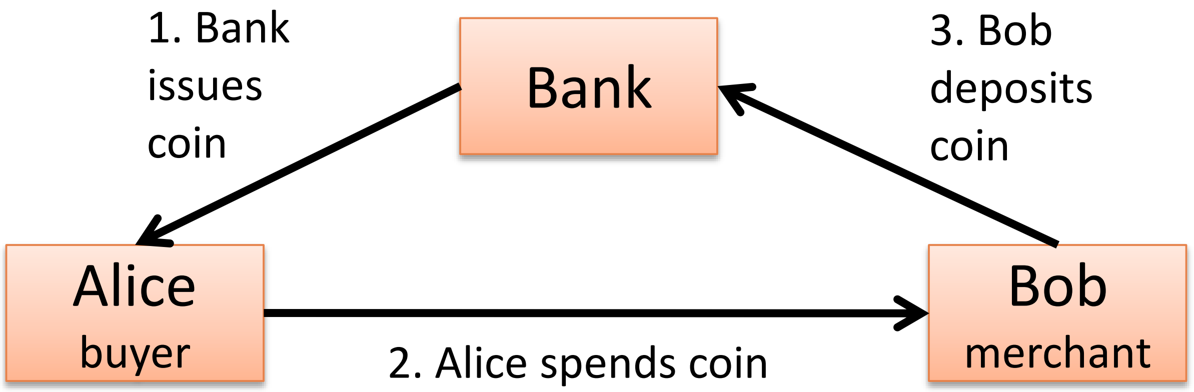
  – Mag stripe data can also be read from the chip
- PIN used frequently → easier to capture



ANONYMOUS PAYMENTS

Anonymous digital cash

- David Chaum 1982, later DigiCash product — never really used but an influential idea
- Participants: bank, buyer Alice, merchant Bob



- Anonymous:
  - Bank cannot link issued and deposited coins, not even with Bob's help
- Not transferable: must be deposited to bank after one use
- Uses blind signatures: bank signs coins without seeing their contents → cannot link events of coin issuing and use

Blind signature

- Idea 1: blind signature:

  Bank has an RSA signature key pair key (e,d,n) for signing 1€ coins (and different keys for 10€, 100€,...)

  1. Alice creates a coin from random "serial number" SN and redundant padding required for the RSA signature;

     Alice generates a random number R, computes $coin \cdot R^e \bmod n$, and sends this to the bank
  2. Bank computes $(coin \cdot R^e)^d \bmod n = coin^d \cdot R \bmod n$ and sends this to Alice
  3. Alice divides with R to get the signed coin $coin^d \bmod n$

  → Bank has signed the coin without seeing it and cannot link the coin to Alice
- Alice can pay 1€ to Bob by giving him the coin
  - Bob deposits coin to bank; bank checks signature and only accepts the same coin once
- Double-spending: Customers are anonymous; if someone pays the same coin to two merchants, who was it?

Double-spending detection

- Idea 2: double-spending detection with secret splitting
  - Alice computes SN = h(h(A,C) | h(A xor "Alice",D)) where A,C,D is a random number
  - After Alice has given the coin to Bob for bind signing, Bob decides which it wants to see: either h(A,C),A xor "Alice",D or A, C, h(A xor "Alice",D)

  → Bob can check that the values are correct by recomputing SN

  → Neither choice reveals the name "Alice", but together they do

  → In double spending, Alice reveals her name with 50% probability
- Make each 1€ coin of k separately signed sub-coins
  → name recovery probability $p = 1-2^{-k}$
  - Coins will be quite large: k=128 with 2048-bit RSA signatures makes 32kB/coin
- Remaining problem: What forces Alice to compute SN this? How can the bank check the contents of the message that it signs blindly?

Cut and choose

- Idea 3: cut and choose
  - Alice creates k pairs of sub-coins for signing
  - Bank asks Alice to reveal N for one sub-coin in each pair and signs the other one
    → probability of detecting malformed coins is $p = 1-2^{-k}$

→Alice can make anonymous payments but will be caught with probability $p = 1-2^{-k}$ if she tries to create an invalid coin or spend the same coin twice

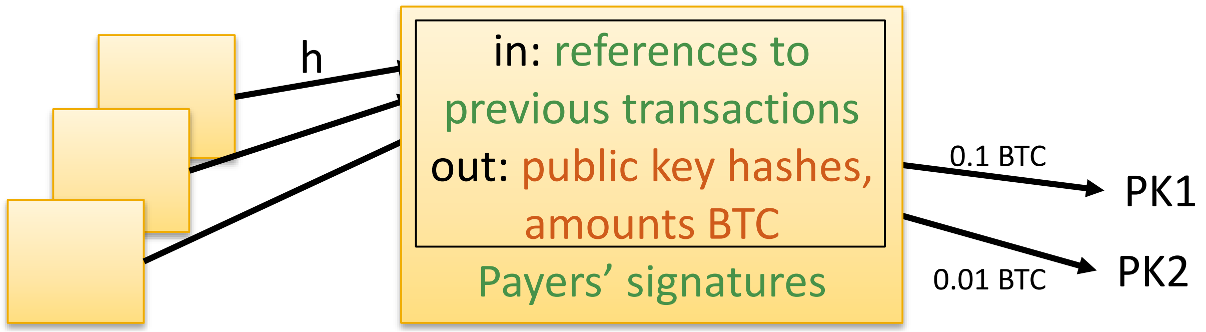
BITCOIN

Bitcoin

- Transferable digital money
  - Based on cryptographic signatures and hash functions
- P2P system, no central bank or trusted issuer
- "Fair", competitive mechanism for the initial issue
- Amount of money in circulation capped
  - Max 21 million BTC
  - Coins can be subdivided to 0.00000001 BTC
- Created in 2008 by pseudonym Satoshi Nakamoto

Bitcoin transaction

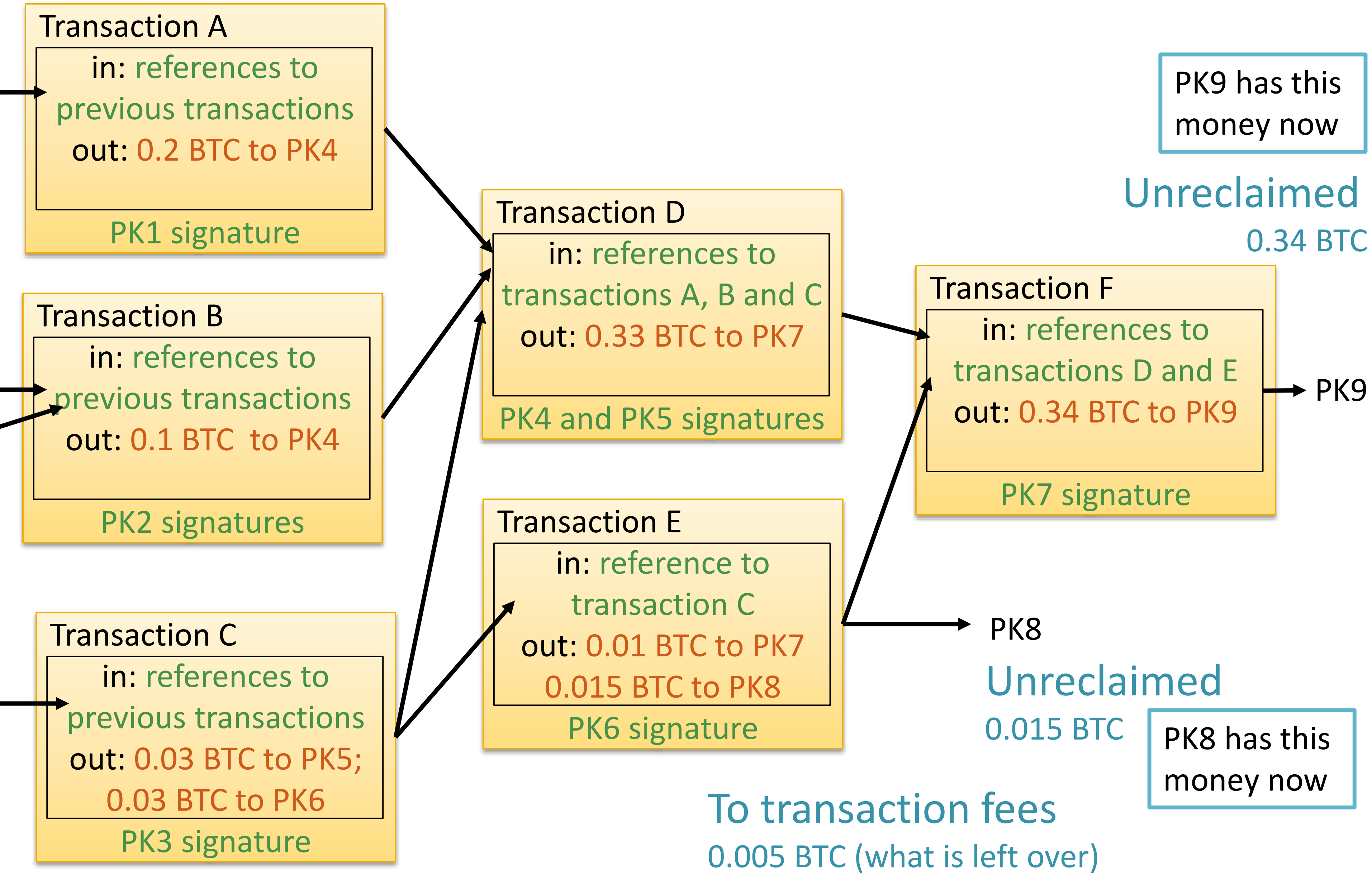
- Direct transactions between public key pairs:
  - Transaction record contains (1) inputs, (2) outputs
  - Input info: (a) pointer to the previous transactions i.e. when did the payers receive this money, (b) payer signature(s)
  - Output info: (a) payee public key hashes, (b) transfer amounts
  - Total inputs from previous transactions must be ≥ total outputs


h
in: references to previous transactions
out: public key hashes, amounts BTC
Payers' signatures
0.1 BTC
PK1
0.01 BTC
PK2

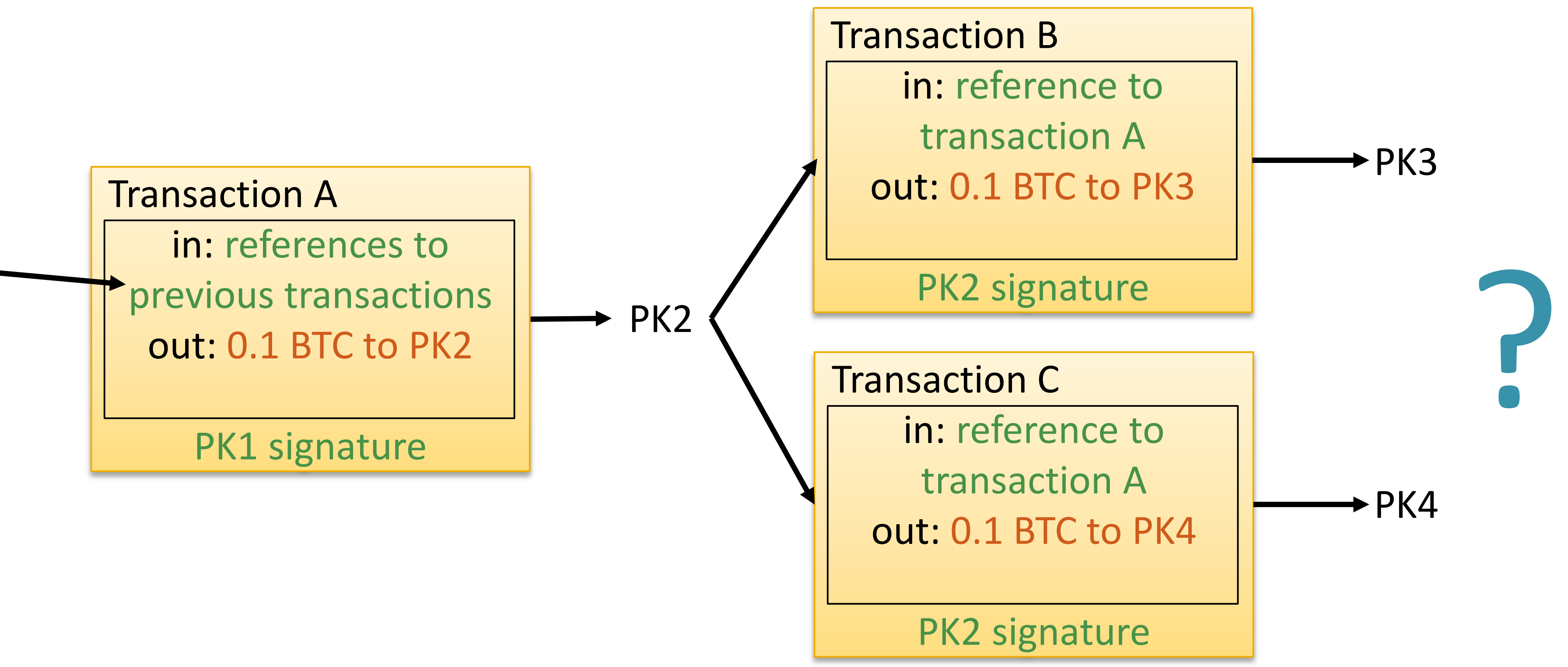
- History of signed transactions proves who has the money now
  - Need to know the complete history of all transactions ever!

- Questions:
  - How to bundle received small outputs, or get change for a large input?
  - What if the outputs ≤ inputs?
  - Who stores the history and checks the signatures?

Transaction history



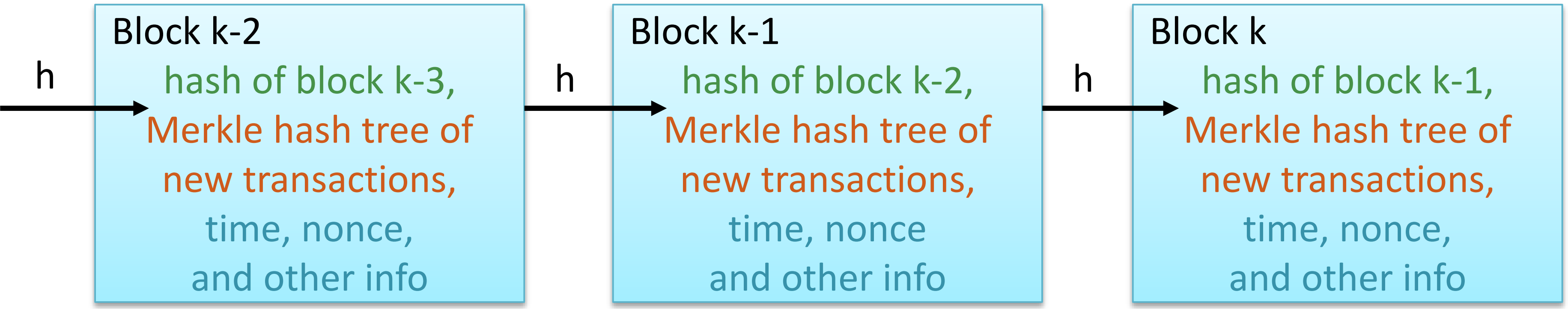
Double spending



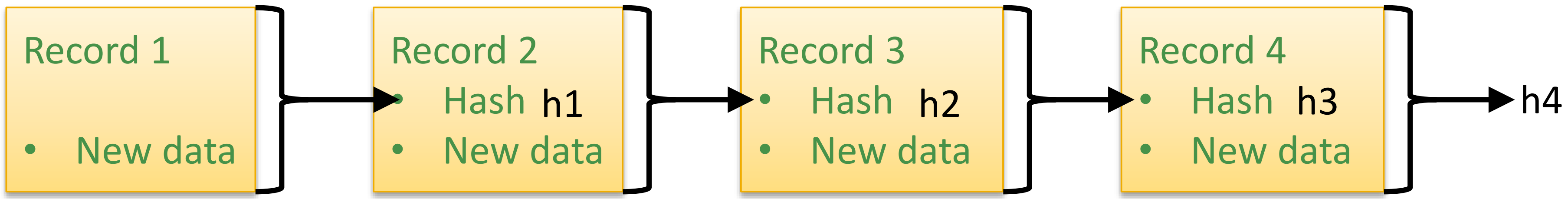
How to prevent double (or over) spending?

Public transaction log

- Public transaction log of all past transactions:
  - All transactions ever made incl. signatures
  - Updated every 10 minutes, on the average
  - Used to check for double spending
- Block chain: public chain of log entries, updated every 10 minutes
  - Block contains hash of the previous block and Merkle hash of new transactions
- The latest block is, in effect, a hash of all transactions ever
- Log size grows over time
- Q: Who can be trusted to maintain the log? A: global P2P network



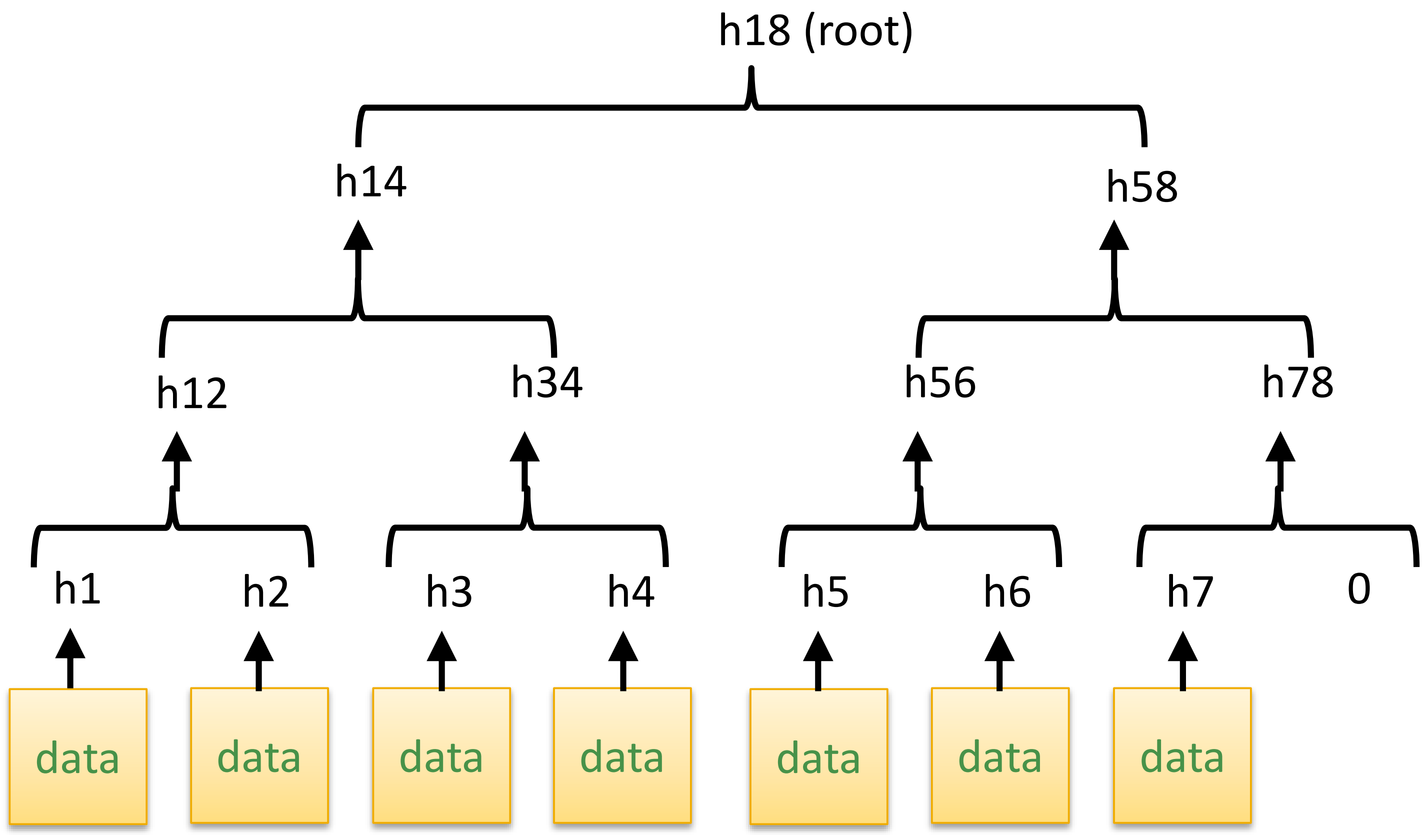
Background info: hash chain



- h1=h(data1,0)
  h2=h(data2,h1)
  h3=h(data3,h2)
  …
- Cumulative hash of data
  - Backward-linked list, with a hash value as the unambiguous reference to the previous record
- Verifying that some data is in the chain costs O(N)
- Appending a data item costs O(1), but updating the hash costs O(N) if earlier data changes

Background info: Merkle tree

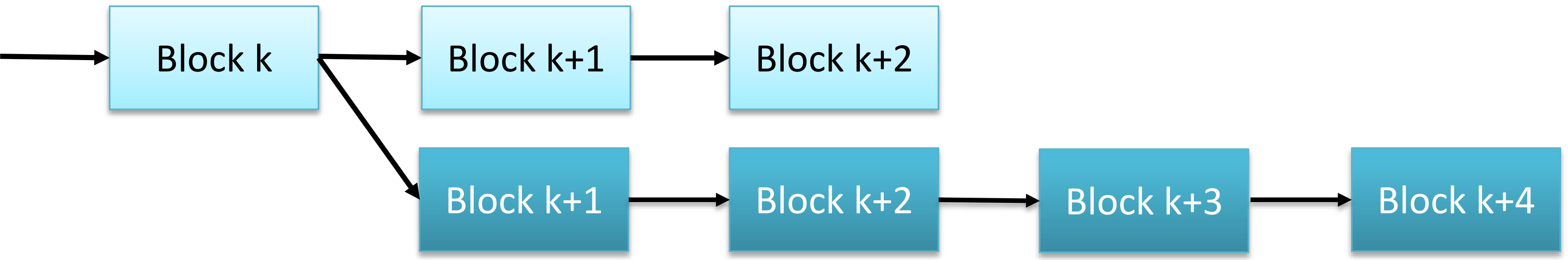
- Binary or n-ary tree of hash values
- Verifying the presence of data costs O(log N) both for computation and communication
- Adding new data or updating old data costs O(log N)



Mining

- Anyone can add blocks to the block chain
  - To do so, you must perform proof of work i.e. solve a cryptographic puzzle with adjustable difficulty, which requires you to do a brute-force search
  - Find a nonce (any number) such that the SHA-256 hash of the block is smaller than a target value
    h(block header(nonce)) ≤ target value
  - The first to find a solution gets a reward and everyone moves to search for the next block
  - The difficulty of the next block is adjusted to keep the predicted block generation time at 10 minutes
- Issuing coins
  - The reward for generating a new block is new BTC (currently 25 BTC per bock) – this is how the coins are initially issued!
  - Transactions may also include a small transaction fee to encourage mining

Security of the block chain



- Double spending detection depends on the block chain not branching
  - Client software always chooses the longest branch if many are available
  - After receiving the payment, sellers publish the transaction to the P2P network and wait until 6 new blocks include it – then the transaction is complete
- If someone controls more than 50% of the global hash rate, they can double spend

Bitcoin philosophy

- Anonymous transactions like cash money?
  - Not exactly: input and output linkable, unlike in DigiCash
  - Signature keys own money, not users
- Digital equivalent of gold:
  - Limited supply on earth
  - Cannot be controlled or inflated by a government
  - Opposition to current monetary policy (quantitative easing)
- Potential problems:
  - Exchange rate volatility
  - Unlike gold; competing electronic currencies easy to create
  - Max transaction rate ~21M BTC / hour, 60-minute latency
  - Security based on wasting energy in CPUs
  - Security reduced if there are many such currencies (mining capacity can move around)
  - Favorite currency for drug trade and other crime – but maybe this is why it will succeed?
  - Tax authorities will be interested
  - Market bubble?

Possible security issues

- No way to reverse transaction without the payee's cooperation
- Block chain branching, double spending
  - Should to wait 60 minutes or more to confirm a transaction
- Software bugs
- Bank robbery by hackers
- Malware attacks against wallets
- Police and government attempts to control
  - Silk Road raided by FBI in Oct 2013
- Competing digital currencies easy to create

Why would anyone use Bitcoin?

Even the most dysfunctional money is better than not having a means for economic exchange

100 000 000 000 000 RESERVE BANK OF ZI
I promise to pay the bearer on demand
ONE HUNDRED TRILLION DOLLARS
for the Reserve Bank of Zimbabwe
Dr. G. Gono Governor
100 000 000 000 000 HARARE 2008
AA3905431

Reading material

- Ross Anderson: Security Engineering, 2nd ed., chapter 10
- Interesting reading online:
  - University of Cambridge Security Group: http://www.cl.cam.ac.uk/research/security/banking/
  - BitCoin wiki: https://en.bitcoin.it/wiki/Main_Page
  - Scam baiting sites have stories about advance-fee fraud etc. (e.g. http://www.419eater.com) — but these site can be unpleasant to read

Exercises

- What are the main threats in
  a) online card transactions?
  b) POS transactions?
  c) ATM cash withdrawals?

  What differences are there in the way credit cards and bank debit cards address these threats?
- Could you (technically) use bank cards or credit cards
  a) as door keys?
  b) as bus tickets?
  c) for strong identification of persons on the Internet?

  (This question may require quite a bit of research.)
- How could a malicious merchant perform a man-in-the-middle attack against chip-and-PIN transactions?
- When a fraudulent bank transaction occurs, who will suffer the losses? Find out about the regulation and contractual rules on such liability.
- Bank security is largely based on anomaly detection and risk mitigation. In what ways could a bank reduce the risk of fraud in mag-stipe or chip-and-PIN payments?
- Even though DigiCash coins are unlinkable, what ways are there for the merchant or bank (or them together) to find out what Alice buys?
- Find a Bitcoin block explorer web site with the full transaction record and browse around. Find the latest blocks and transactions, and the first block ever. See how the mining difficulty has changed over time.

The course ends here, what next?

CSE-C3400	Information Security --- DONE!
CSE-E5480	Mobile Systems Security (Asokan, spring)
T-110.5241	Network Security (Tuomas Aura, autumn, period II)
T-110.5102	Laboratory Works in Networking and Security (spring)
T-110.6220	Special Course in Information Security P (usually spring)
T-110.5291	Seminar on Network Security P (autumn + spring)
T-110.6101	Special Assignment in Networking and Security P
T-79.4502	Cryptography and Data Security (Kaisa Nyberg, autumn)
T-79.5501	Cryptology P (Kaisa Nyberg, spring)