CBS COPENHAGEN BUSINESS SCHOOL
HANDELSHØJSKOLEN

Elective Course on Mastering Blockchain: Foundations to Consensus

Recap of Hash Functions and Cryptography

Raghava Mukkamala

Associate Professor & Director, Centre for Business Data Analytics
Copenhagen Business School, Denmark
Email: rrm.digi@cbs.dk, Centre: https://cbsbda.github.io/

Course Coordinator at SRMIST:

Prof. K. Shantha Kumari

Associate Professor

Data Science and Business Systems Department, SRMIST

Shanthak@srmist.edu.in

SRM Institute of Science and Technology, India

EQUIS ACCREDITED AACSB ACCREDITED ASSOCIATION AMBA ACCREDITED CEMS PIM

About Myself

IT Development Experience

- 2000-2009: Software Programmer, software Consultant in Danish IT Industry (C++, C# and Java)
- 2003-2005: M S in Internet Technology from IT University of Copenhagen (ITU), Denmark.

Academic/Research Experience

- 2009-2012: PhD in Theoretical Computer Science & Formal Models from ITU, Denmark.

(PhD Topic: A Formal Model for Declarative Workflows: Dynamic Condition Response Graphs)

- 2012-2015: Postdoc at ITU on Software Verification and Domain specific Modelling languages.
- 2015-2017: Assistant Professor at dept. of IT Management at Copenhagen Business School (CBS), Denmark.
- 2017: Tenured Associate Professor at Department of Digitalization at CBS.
- 2018: Program Director for Masters in Business Administration and Data Science Programme at CBS.
- 2019: Director for Center for Business Data Analytics at Department of Digitalization at CBS.

Research and Teaching

- Applied Data Science, Machine learning, Natural Language Processing
- Blockchain and Smart Contracts, Cybersecurity, Cyber harassment, Hate speech

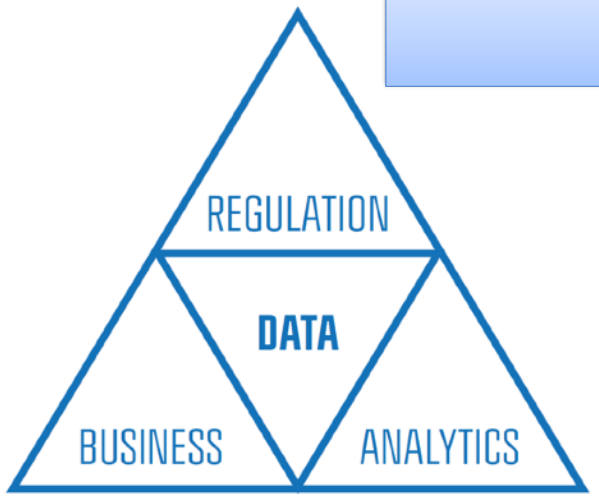
Copenhagen Business School

- Public-funded Social Sciences University in Denmark (~ 100 years)
- "The Triple Crown" Accreditation
- Worldwide rankings vary from 10-50
- 21, 000 students with 4000 international (2-5% of international may be tuition fee paying) and so free education for 99% of students
- 820 faculty + 199 PhD students

https://www.cbs.dk/en/about-cbs/profile/accreditations-and-rankings

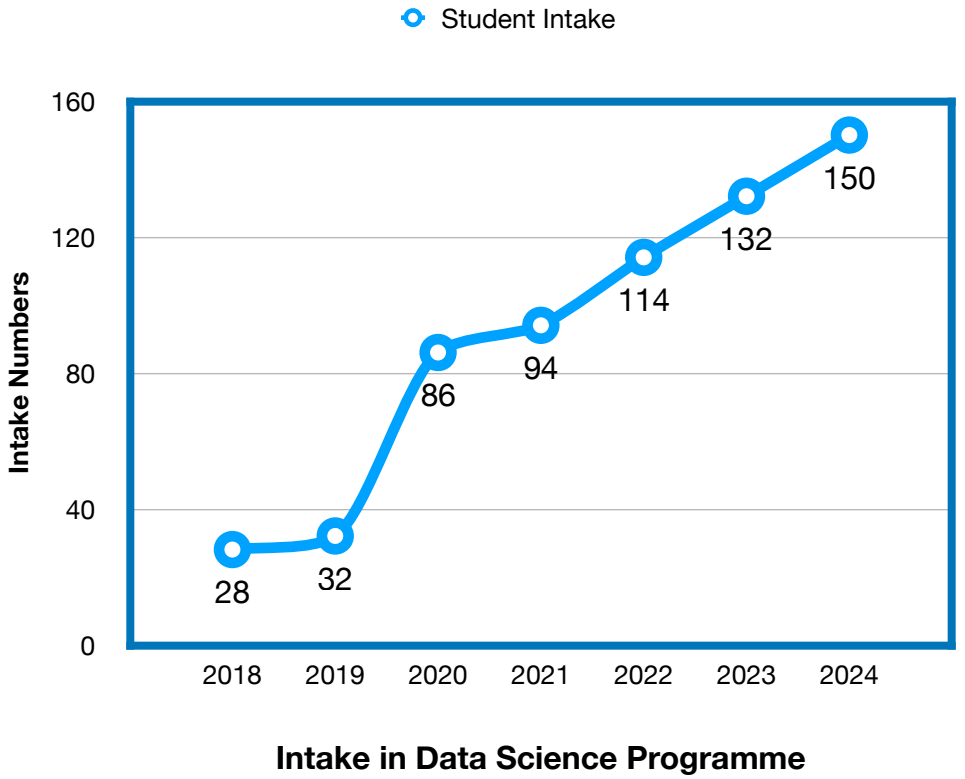
Education: Masters Data Science Programme

Since 2018:
Masters in Data Science



August 2019: Got accreditation from Danish Ministry of Higher Education and Science, as independent Masters programme.

65% students are international students, mostly Scandinavia, Germany, and rest of the European countries



Outline

- Hash Functions
  - collision-free
  - hiding
  - puzzle-friendly
  - Applications of One-Way Hash Functions
- Hash Pointers & Hash-based Data Structures
- Symmetric & Asymmetric Cryptography
- Digital Signatures
- Public Keys in Blockchain

Course Outline

	Date & Time	Topic
1	05-08-2024 Monday 19:15-20:45 IST	A quick recap of Hash Functions, Cryptography, symmetric, asymmetric and elliptic curve cryptography; Digital Signatures, Public/Private keys, Addresses
2	07-08-2024 Wednesday 19:15-20:45 IST	History of Bitcoin: Cypherpunk to 2008 Financial crisis; Chronology of key ideas in Bitcoin and Cryptoeconomics; A Simple Cryptocurrency.
3	12-08-2024 Monday 19:15-20:45 IST	Transactions and transaction-based ledger Immutable coins for Cryptocurrencies Blocks and block structure
4	14-08-2024 Wednesday 19:15-20:45 IST	Storing and spending Bitcoins Online wallets and exchanges Securing Bitcoin by splitting and sharing keys
5	21-08-2024 Wednesday 19:15-20:45 IST	Blockchain Use cases on Identity management and NGOs, NPOs, social businesses and discussion on project reporting template
6	22-08-2024 Thursday 19:15-20:45 IST	Project Consultations and project mentoring session - I

Course Outline

7	29-08-2024 Thursday 19:15-20:45 IST	Bitcoin scripting fundamentals Advanced Bitcoin scripting; Applications of Bitcoin scripts
8	04-09-2024 Wednesday 19:15-20:45 IST	Bitcoin mining and proof-of-work Energy consumption: why is Bitcoin mining unsustainable? Mining pool strategies
9	18-09-2024 Wednesday 19:15-20:45 IST	Trust and consensus in distributed systems Achieving distributed and decentralized consensus; Voting and federated consensus
10	25-09-2024 Wednesday 19:15-20:45 IST	Project Consultations and project mentoring session - II
11	02-10-2024 Wednesday 19:15-20:45 IST	Incentives and mining schemes proof-of-work, proof-of-stake; peer-to-peer networking and Gossip protocol
12	09-10-2024 Wednesday 19:15-20:45 IST	Anonymity basics Deanonymization in cryptocurrencies Anonymity through mixing and altcoins
13		Project Presentations
14		Project Presentations
15		Project Presentations

Course Materials

Dropbox Share Folder link:

- https://www.dropbox.com/scl/fo/i001qu6u8kq8ftzrxm9gd/AB4NbHbDISXVXEw2KKbYTWY?rlkey=le7rvhyokfpipt58kfy7bvkv0&e=1&dl=0

OR

- https://tinyurl.com/MBFTC-2024

HISTORY BEHIND BITCOIN

DOI:10.1145/3132259

Article development led by acmqueue
queue.acm.org

The concept of cryptocurrencies is built from forgotten ideas in research literature.

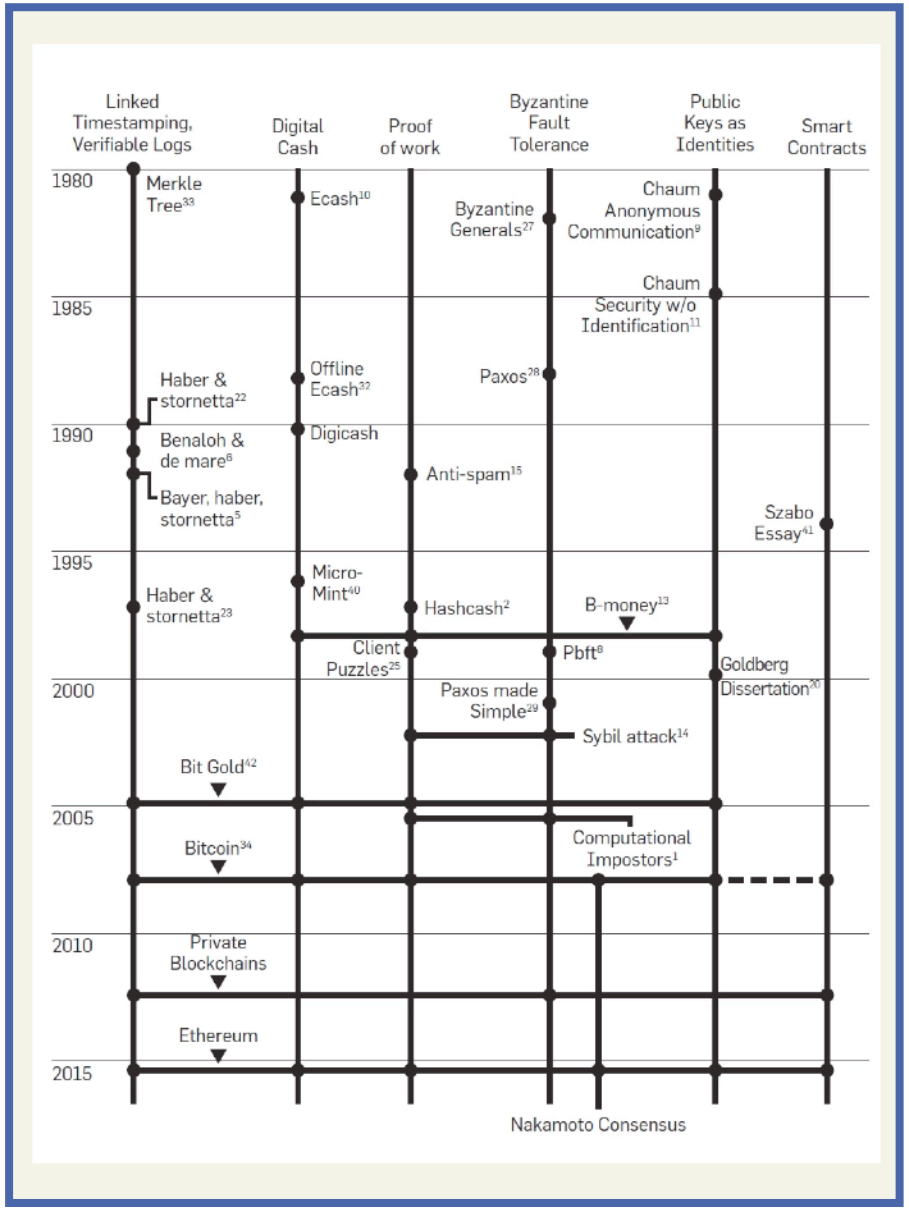
BY ARVIND NARAYANAN AND JEREMY CLARK

Bitcoin's Academic Pedigree

CBS

Chronology of Key Ideas behind Bitcoin

- Cryptocurrencies are built from forgotten ideas from research literature
- Not to diminish Nakamoto's achievement but to prove that he stood on the shoulders of giants
- All of the technical components of bitcoin originated in the academic literature from 1980s and 1990s



1) Source: Narayanan, A., & Clark, J. (2017). Bitcoin's academic pedigree. Communications of the ACM, 60(12), 36-45.

Hash Functions

Many of the slides were taken and adapted from the course on Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/ with due credits to the original author

Image by WorldSpectrum from Pixabay

CBS

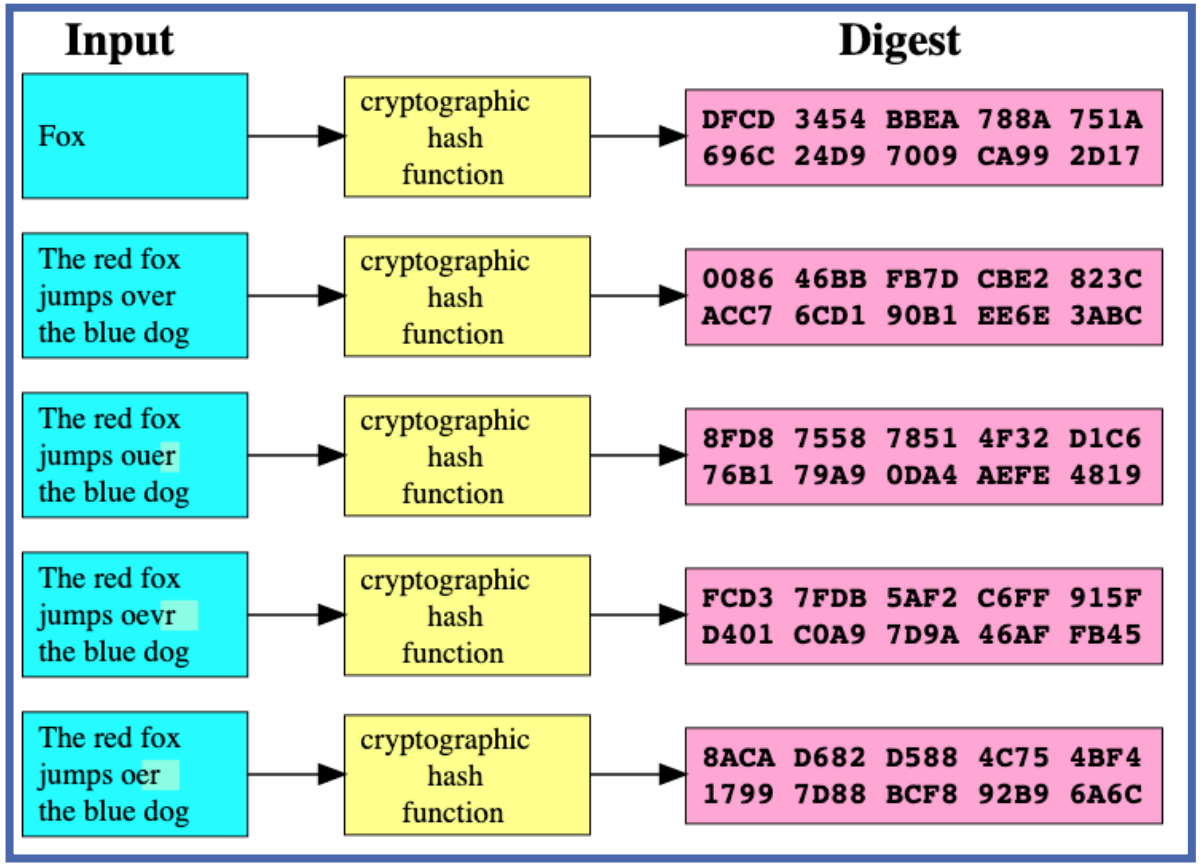
Hash Function

Hash function

- takes any string as input
- fixed-size output (we'll use 256 bits)
- efficiently computable

Security Properties

- collision-free
- hiding
- puzzle-friendly

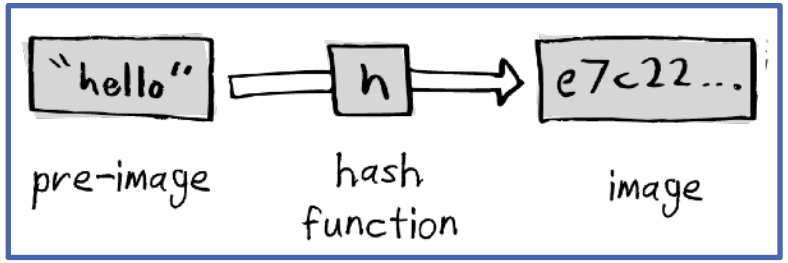


By User:Jorge Stolfi based on Image:Hash_function.svg by Helix84 - Original work for Wikipedia, https://en.wikipedia.org/wiki/Cryptographic_hash_function#/media/File:Cryptographic_Hash_Function.svg

Online Hash Generator: http://onlinemd5.com/

Cryptographic hash functions

- There security of hash functions is defined empirically, if the following problems are found to be computationally infeasible:
  - One way:
    - Given y, find x such that h(x) = y
  - Second pre-image resistant:
    - Given x, find y≠ x such that h(x) = h(y)
  - Collision-resistant:
    - Find y, x, with y≠ x such that h(x) = h(y)



https://mccormick.cx/news/entries/hash-function-attacks-illustrated

One way:Pre-Image Resistance: It should be infeasible (NOT impossible) to find an input 'A' from H(A).

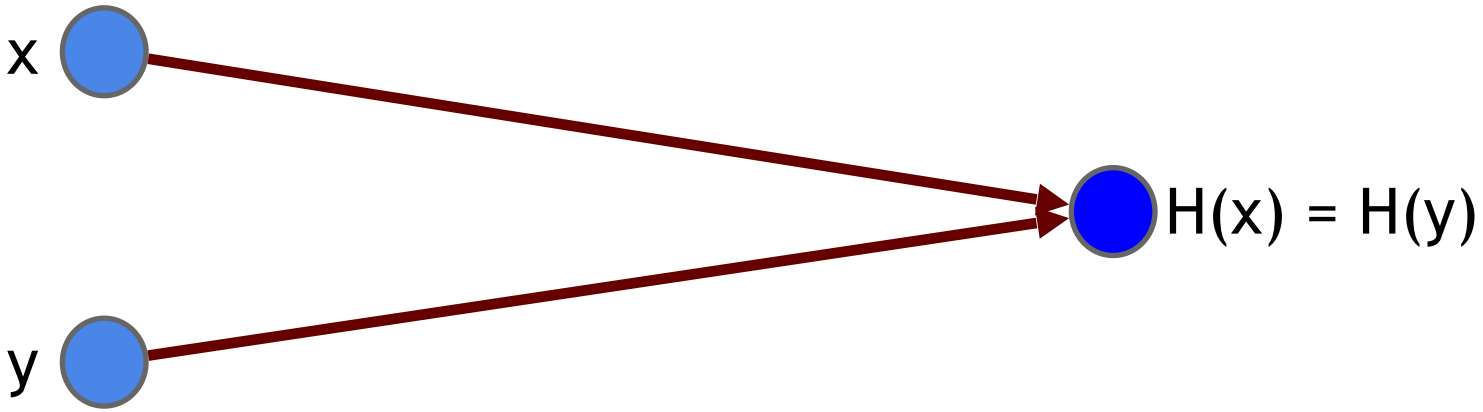
- Deterministic: Hash always be same while executing same input multiple times.
- Unique: Hash values should be unique.

H(A)=A' and H(B)=B', but A' != B'

Hash property 1: Collision-free

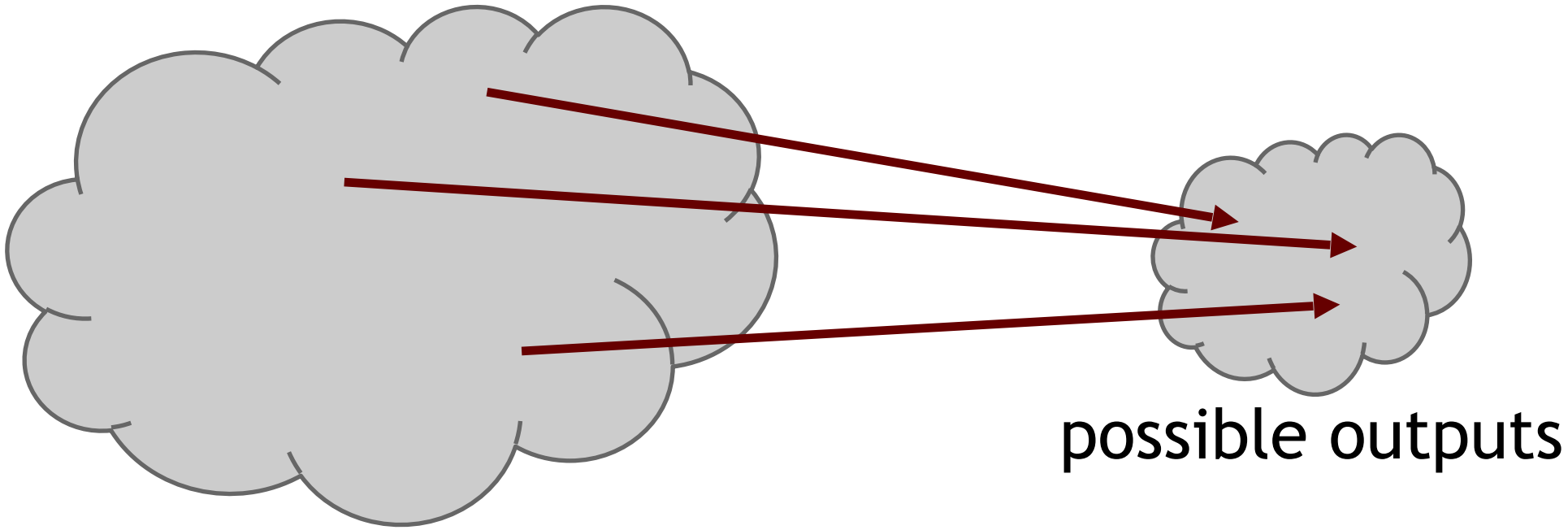
Nobody can find x and y such that

x != y and H(x)=H(y)



Bitcoin and Cryptocurrency Technologies: http://bitcoinbook.cs.princeton.edu/

Collisions do exist …



… but can anyone find them?

How to find a collision

try 2^{130} randomly chosen inputs

99.8% chance that two of them will collide

This works no matter what H is …

… but it takes too long to matter (millions of years …)

Application: Hash as message digest

If we know H(x) = H(y),

it's safe to assume that x = y.

To recognize a file that we saw before,

just remember its hash.

Useful because the hash is small.

Hash property 2: Hiding

We want something like this:

Given H(x), it is infeasible to find x.

H(“heads”)

H(“tails”)

easy to find x!

Hash property 2: Hiding

Hiding property:

If r is chosen from a probability distribution that has *high min-entropy*, then given H(r | x), it is infeasible to find x.

High min-entropy means that the distribution is “very spread out”, so that no particular value is chosen with more than negligible probability.

Application: Commitment

Want to “seal a value in an envelope”, and
“open the envelope” later.

Commit to a value, reveal it later.

Commitment API

com := commit(*msg*, *key*)

match := verify(*com*, *key*, *msg*)

To seal *msg* in envelope:

com := commit(*msg*, *key*) -- then publish *com*

To open envelope:

publish *key*, *msg*

anyone can use verify() to check validity

Commitment API

com := commit(*msg*, *key*)

match := verify(*com*, *key*, *msg*)

Security properties:

Hiding: Given *com*, infeasible to find *msg*.

Binding: Infeasible to find *msg != msg'* such that

verify(commit(*msg*), *msg'*) == true

Hash property 3: Puzzle-friendly

Puzzle-friendly:

For every possible output value y,

if k is chosen from a distribution with high min-entropy,

then it is infeasible to find x such that H(k | x) = y.

> ***Puzzle friendliness.*** A hash function *H* is said to be puzzle-friendly if for every possible n-bit output value *y*, if k is chosen from a distribution with high min-entropy, then it is infeasible to find *x* such that H(k || x) = y in time significantly less than 2^n.

Application: Search puzzle

Given a "puzzle ID" *id* (from high min-entropy distrib.),
and a target set *Y*:
Try to find a "solution" *x* such that

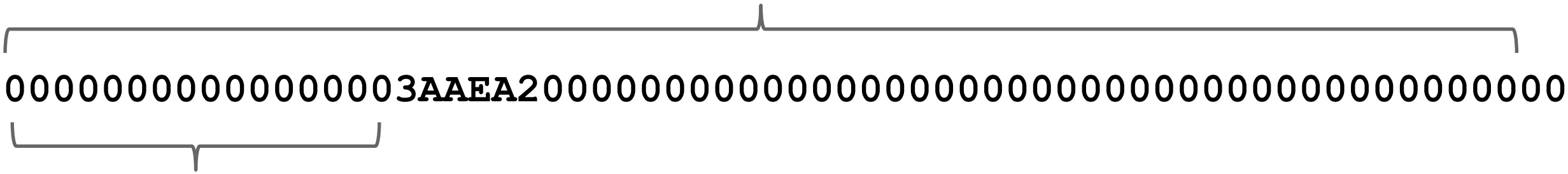
$$H(id \mid x) \in Y.$$

Puzzle-friendly property implies that no solving strategy is much better than trying random values of *x*.

http://onlinemd5.com/

Mining difficulty “target” (2014-08-07)

256 bit hash output

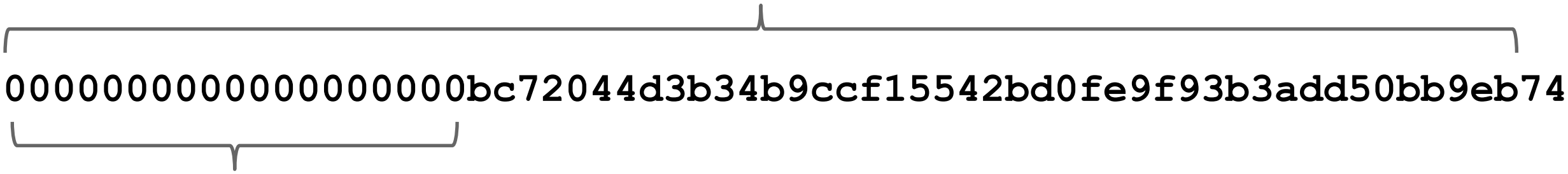


Current difficulty = $2^{66.2}$

=84,758,978,290,086,040,000

Mining difficulty “target” (2019-10-08)

256 bit hash output



Current difficulty = $2^{78.2}$

=12,759,819,404,408

84,758,978,290,086,040,000

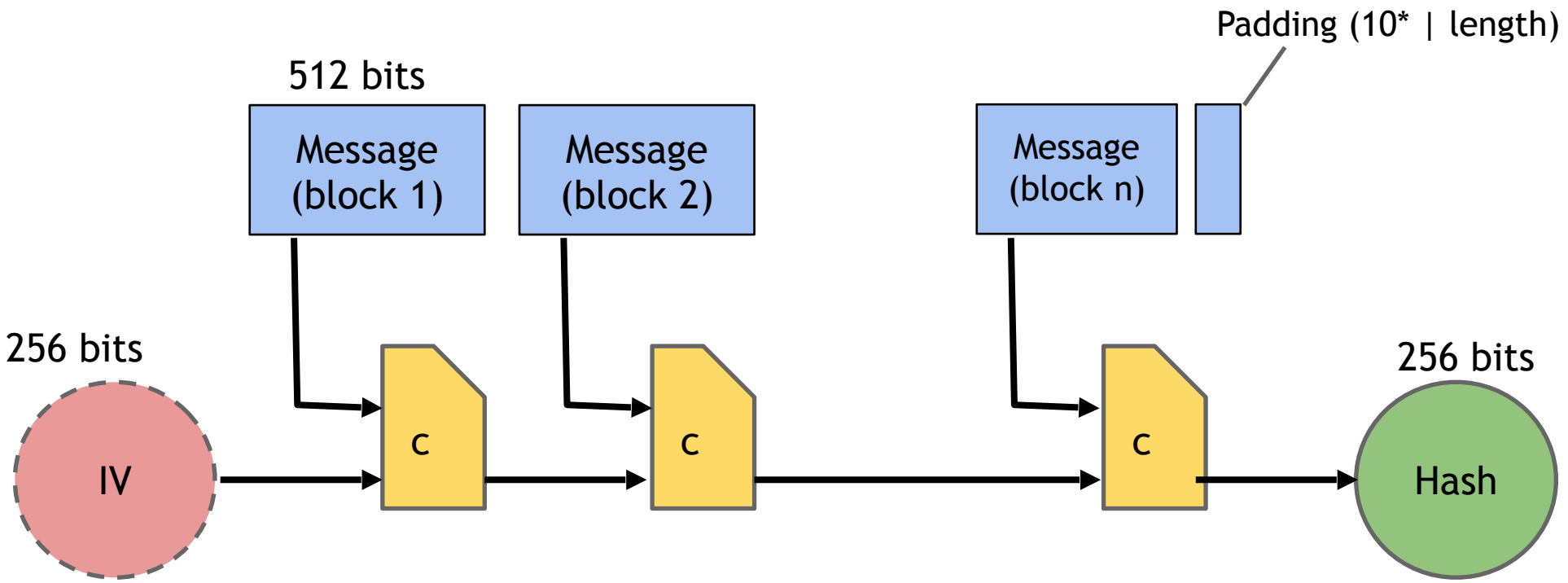
MD One-Way Hash Functions

- MD stands for Message Digest
- Developed by Ron Rivest
- Includes MD2, MD4, MD5,and MD6
- Status of Algorithms:
  - MD2, MD4 - severely broken (obsolete)
  - MD5 - collision resistance property broken, one-way property not broken
  - MD6 - developed in response to proposal by NIST

SHA

- Published by NIST
- Includes SHA-0, SHA-1, SHA-2, and SHA-3
- Status of Algorithms:
  - SHA-0: withdrawn due to flaw
  - SHA-1: Designed by NSA; Collision attack found in 2017
  - SHA-2: Designed by NSA; Includes SHA-256 and SHA-512; Other truncated versions; No significant attack found yet
  - SHA-3: Released in 2015; Has different construction structure (compared to SHA-1 and SHA-2)

SHA-256 hash function



Theorem: If c is collision-free, then SHA-256 is collision-free.

Committing a Secret Without Telling It

- One-way property
  - Disclosing the hash does not disclose the original message
  - Useful to commit secret without disclosing the secret itself
- Usage Example - Stock Market
  - Need to make prediction about the stock market about a certain day
  - Publish the hash of the secret on your website
  - On the particular day, release the secret
  - Your audience can verify it against the hash

Password Verification

- To login into account, user needs to tell a secret (password)
- Cannot store the secrets in their plaintext
- Need for:
  - Password storage where nobody can know what the password is
  - If provided with a password, it verified against the stored password
- Solution: one-way hash function
- Example: Linux stores passwords in the /etc/shadow file

```
seed:$6$wDRrWCQz$IsBXp9.9wz9SG(omitted)sbCT7hkxXY/:17372:0:99999:7:::
test:$6$a6ftg3SI$apRiFL.jDCH7S(omitted)jAPXtcB9oC0:17543:0:99999:7:::
```

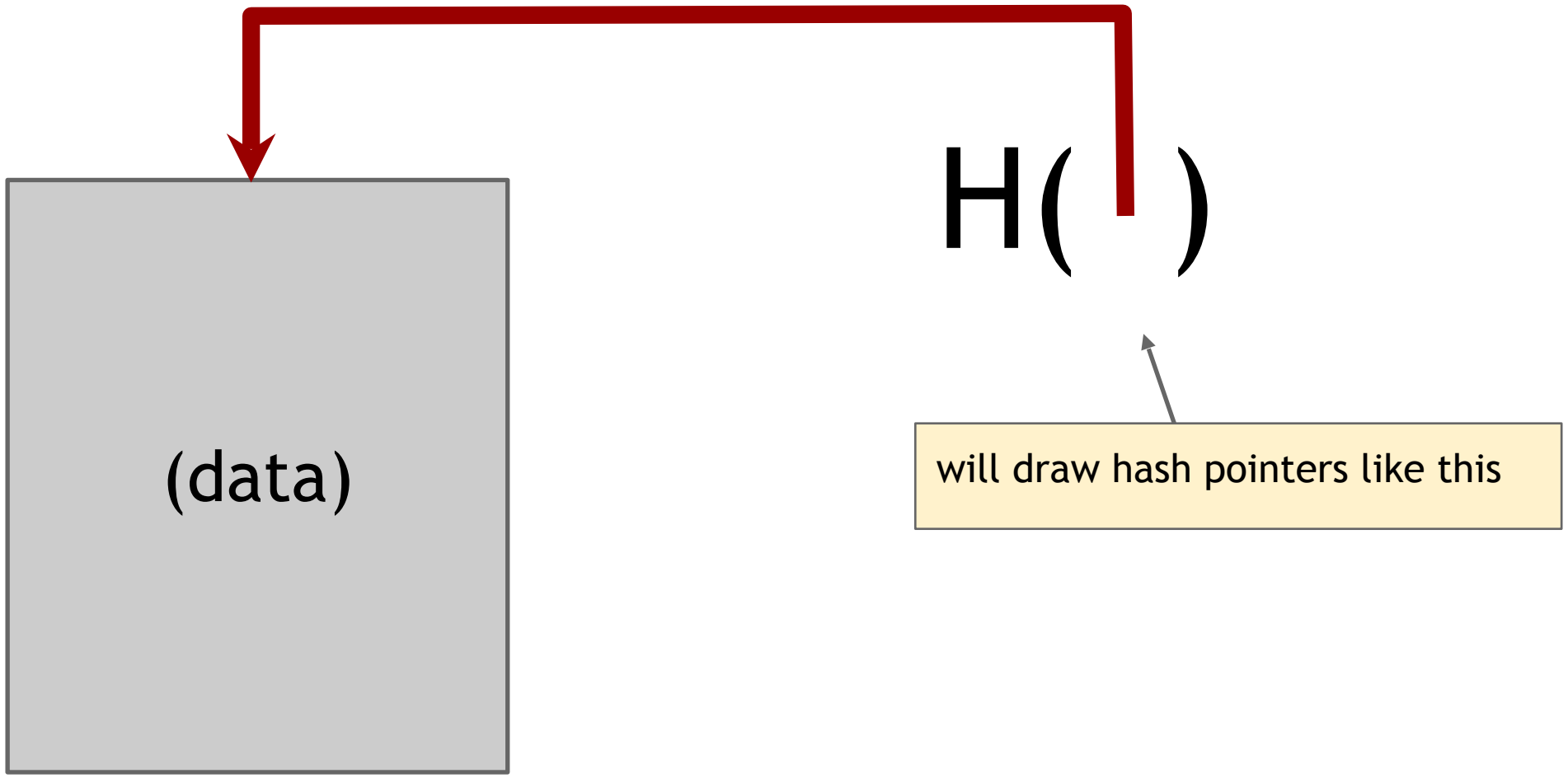
Hash Pointers & Hash-based Data Structures

hash pointer is:

* pointer to where some info is stored, and
* (cryptographic) hash of the info

if we have a hash pointer, we can

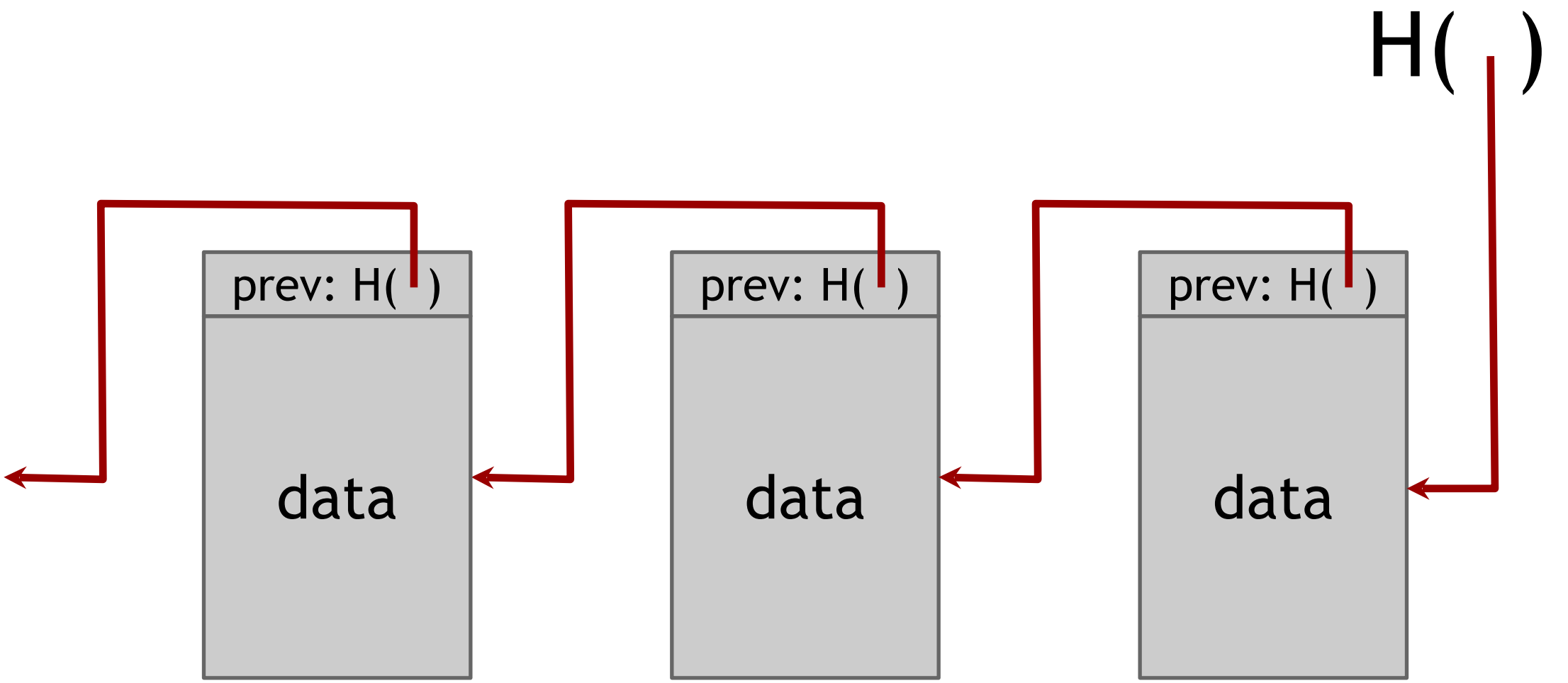
* ask to get the info back, and
* verify that it hasn’t changed


H( )
(data)
will draw hash pointers like this

key idea:

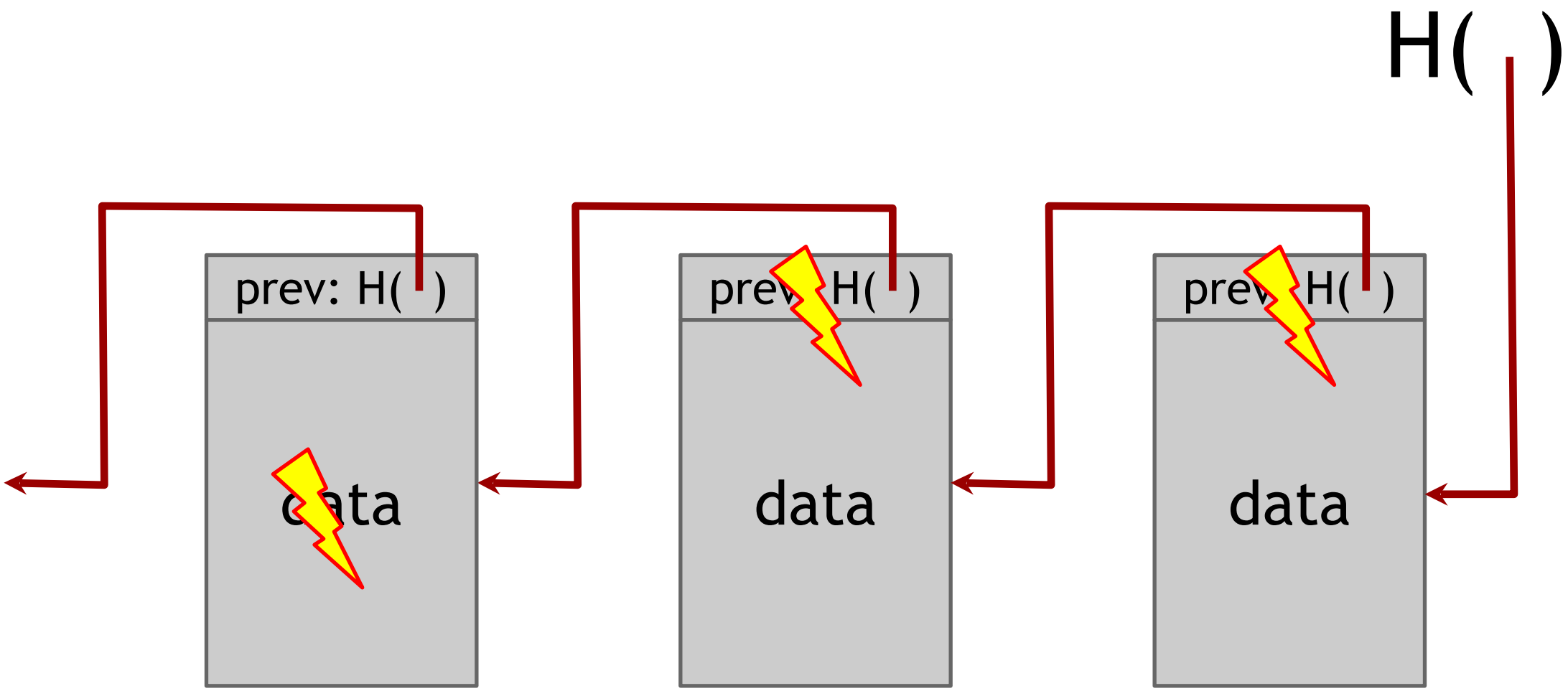
build data structures with hash pointers

linked list with hash pointers = “block chain”



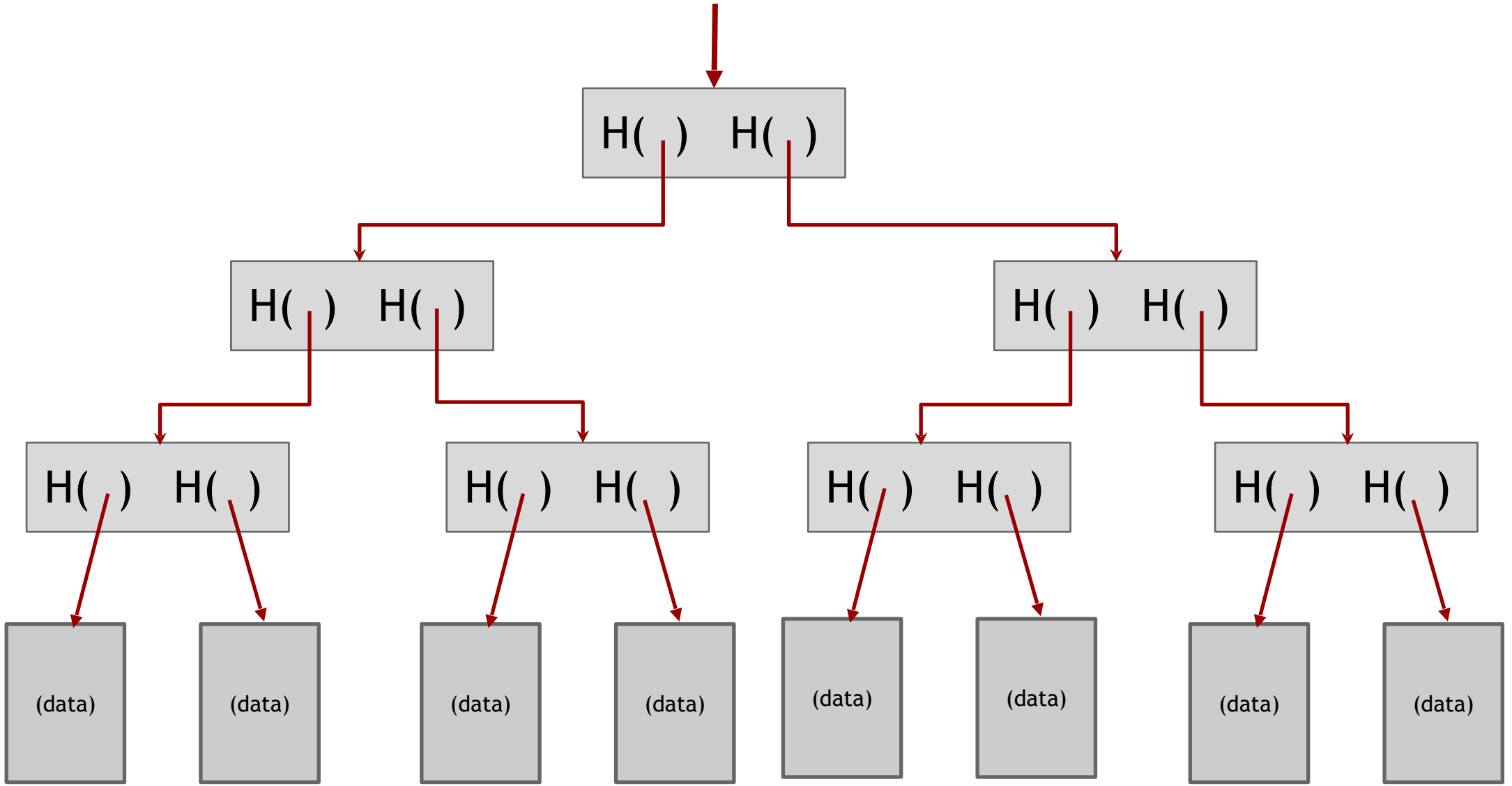
use case: tamper-evident log

detecting tampering

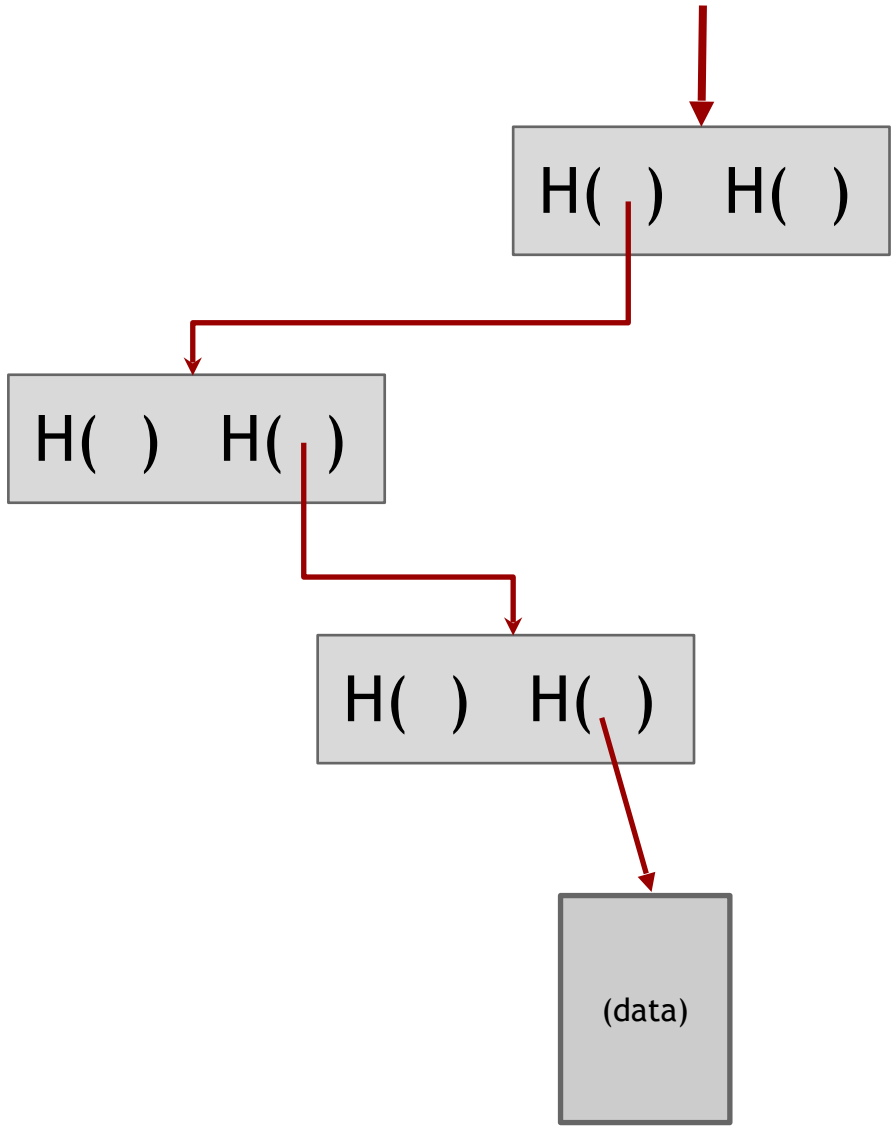


use case: tamper-evident log

binary tree with hash pointers = “Merkle tree”



proving membership in a Merkle tree



show O(log n) items

Advantages of Merkle trees

Tree holds many items

but just need to remember the root hash

Can verify membership in O(log n) time/space

Variant: sorted Merkle tree can verify non-membership in O(log n)

If n = 1000, then log(1000) = 3, so if you have 1000 transactions, then you can verify using just 3 lookups

More generally ...

can use hash pointers in any pointer-based data structure that has no cycles

Cryptography

Cryptography

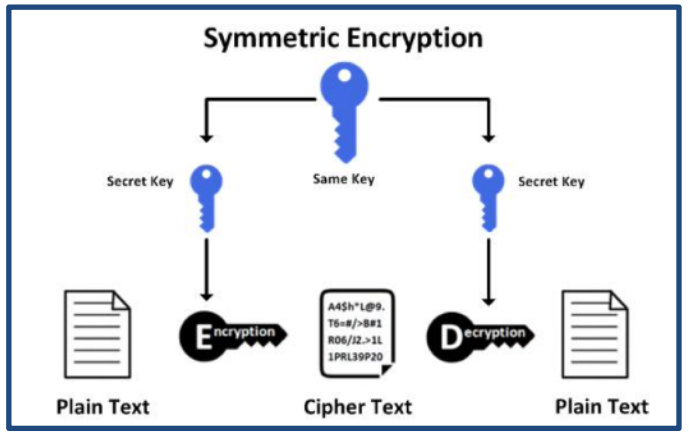
Table 8.1 History of Cryptography

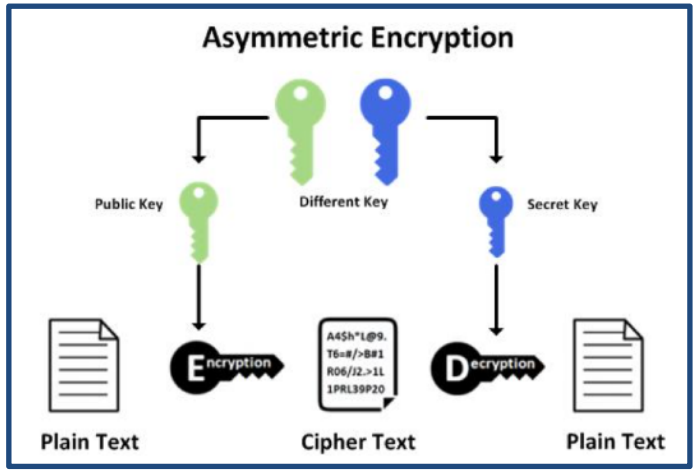
Year	Cryptography
500 to 600 BC	Atbash
< 10 BC	Polybius Square
45 BC to 45 AD	Caesar Cipher
50–120 AD	Scytale
1553 AD	Vigenere′ Cipher
1910 to 1940	Enigma
1976	DES and Diffie-Hellman
1977	RSA
1985	Elliptic Curve Cryptography
1993	Blowfish Cipher
1998	Rijndael Published
2001	Rijndael selected as AES

Easttom, Chuck. Computer security fundamentals. Pearson IT Certification, 2019. P.no. 208

Encryption Methods

- Encryption is a process of encoding information
- It converts plain/original information into ciphertext/encrypted blob
- associated with keys for encrypting/ decrypting
- Two types:
  - Symmetric = use same key
  - Asymmetric = use different keys

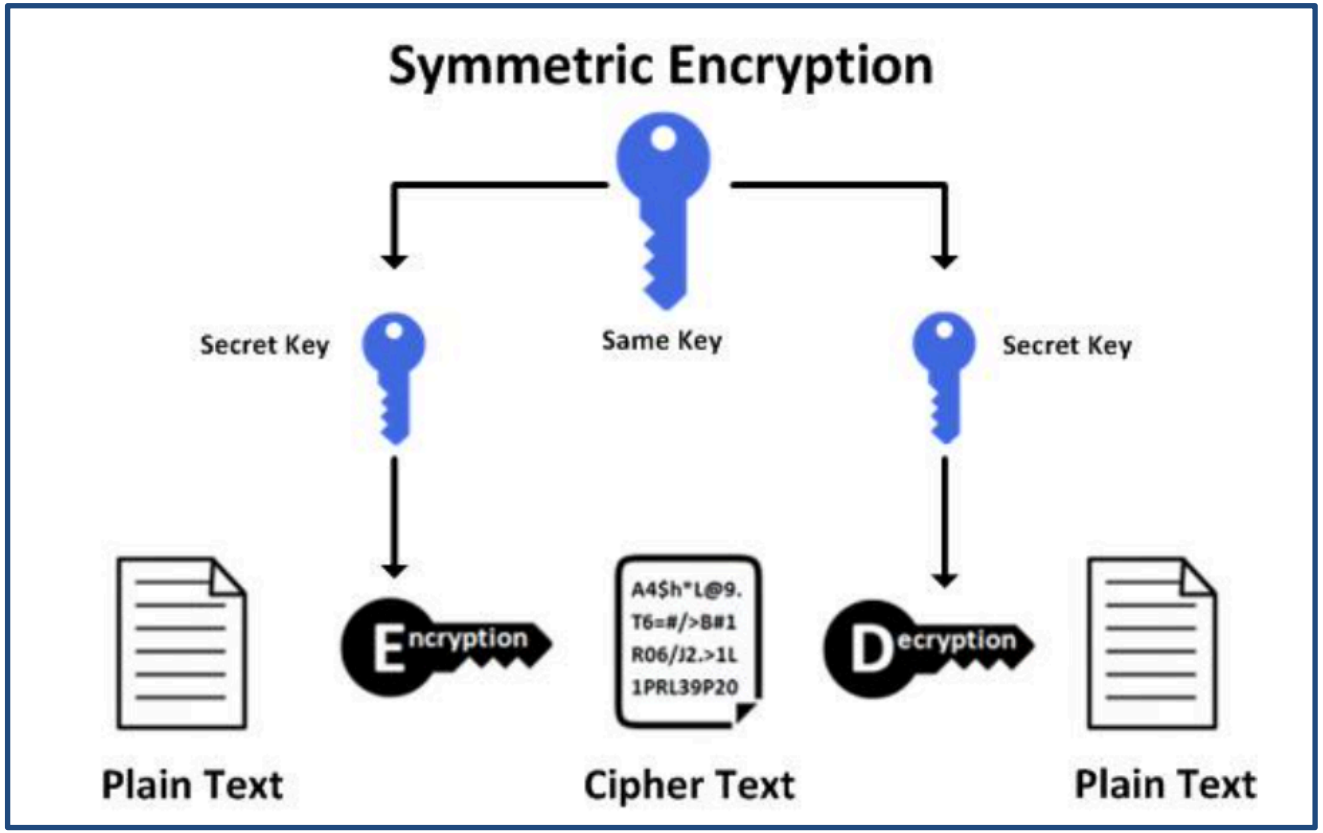




Ons Jallouli. Chaos-based security under real-time and energy constraints for the Internet of Things. PhD thesis. Signal and Image processing. UNIVERSITE DE NANTES, 2017.

Symmetric Encryption

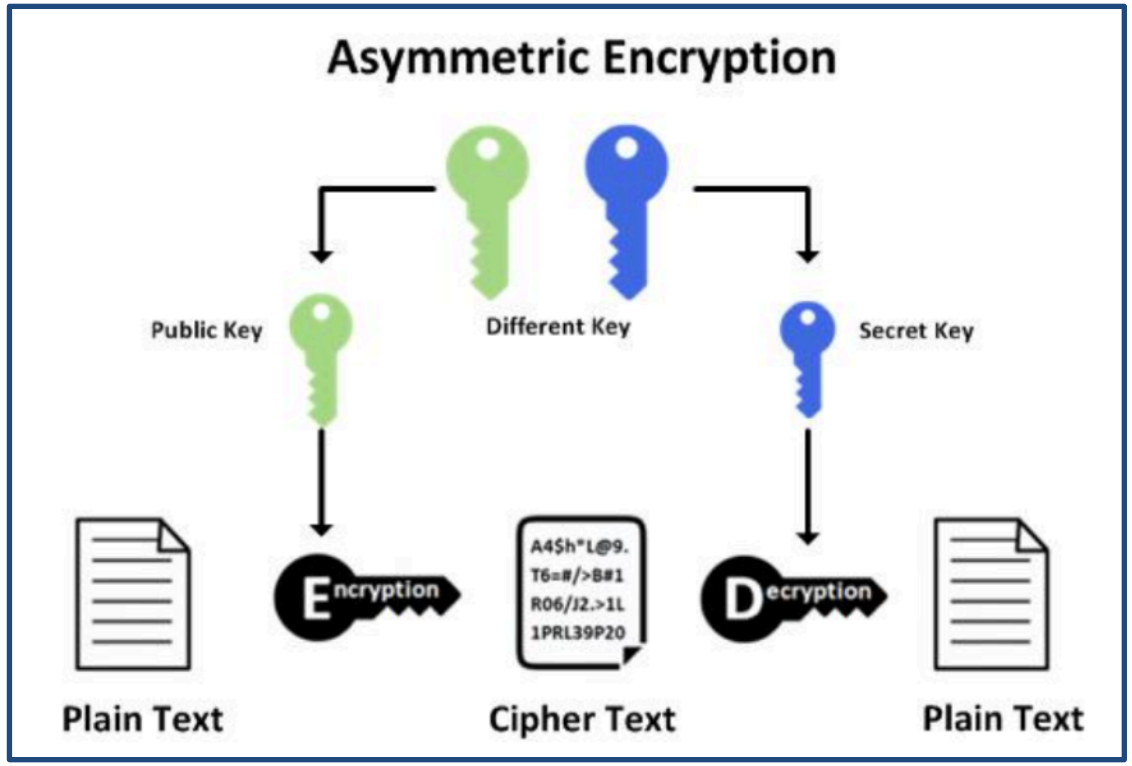
- Symmetric: use same key to encrypt and decrypt
- Both parties need to know the key
- How do we distribute the key securely?
- Algorithms: AES (Advanced Encryption Standard), DES



Ons Jallouli. Chaos-based security under real-time and energy constraints for the Internet of Things. PhD thesis. Signal and Image processing. UNIVERSITE DE NANTES, 2017.

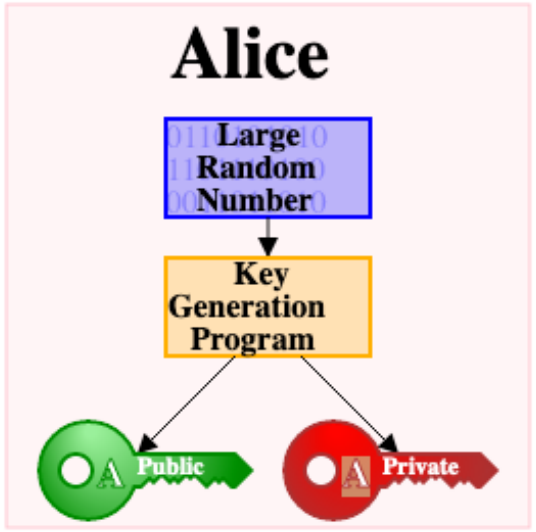
Asymmetric Encryption

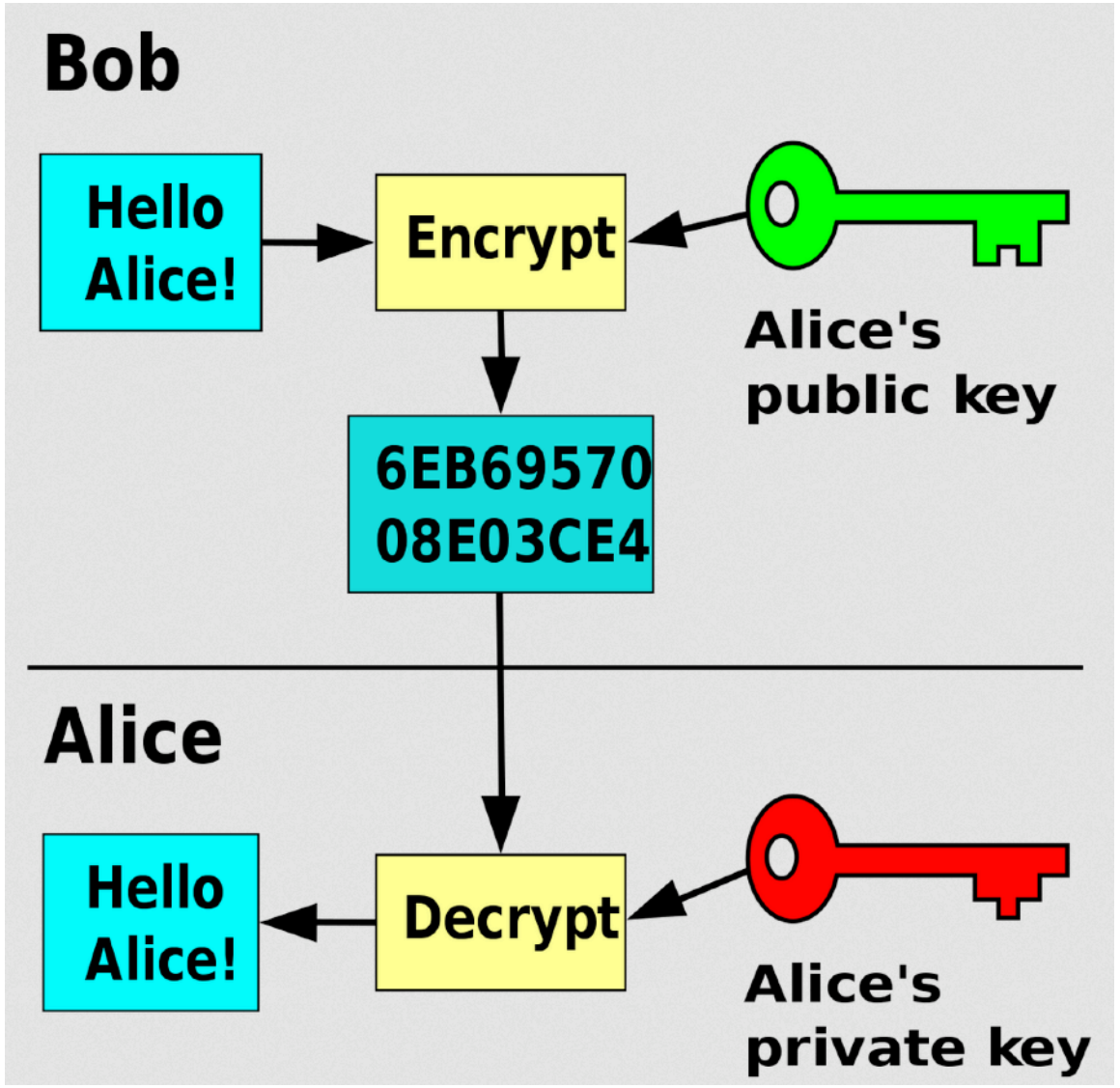
- Use a key pair
  - public key
  - private/secret key
- Corresponding keys: if we use one key to encrypt, then we can only decrypt with other key
- Algorithms: RSA (Rivest–Shamir–Adleman)



Ons Jallouli. Chaos-based security under real-time and energy constraints for the Internet of Things. PhD thesis. Signal and Image processing. UNIVERSITE DE NANTES, 2017.

Asymmetric Cryptography - Encryption

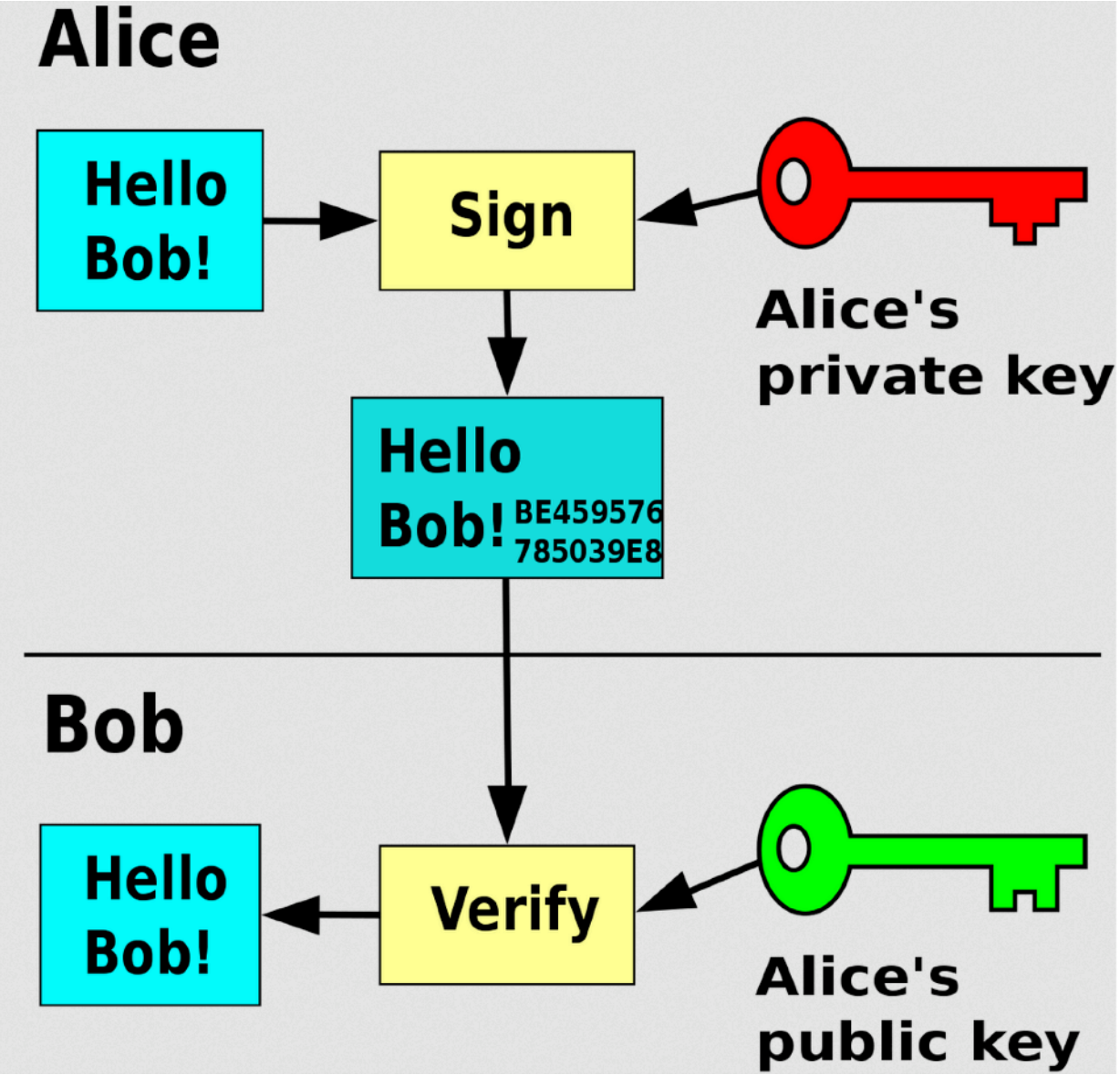




Goal is to send secure message so that only Alice (none other) can read the message.

https://en.wikipedia.org/wiki/Public-key_cryptography

Asymmetric Cryptography - Signing

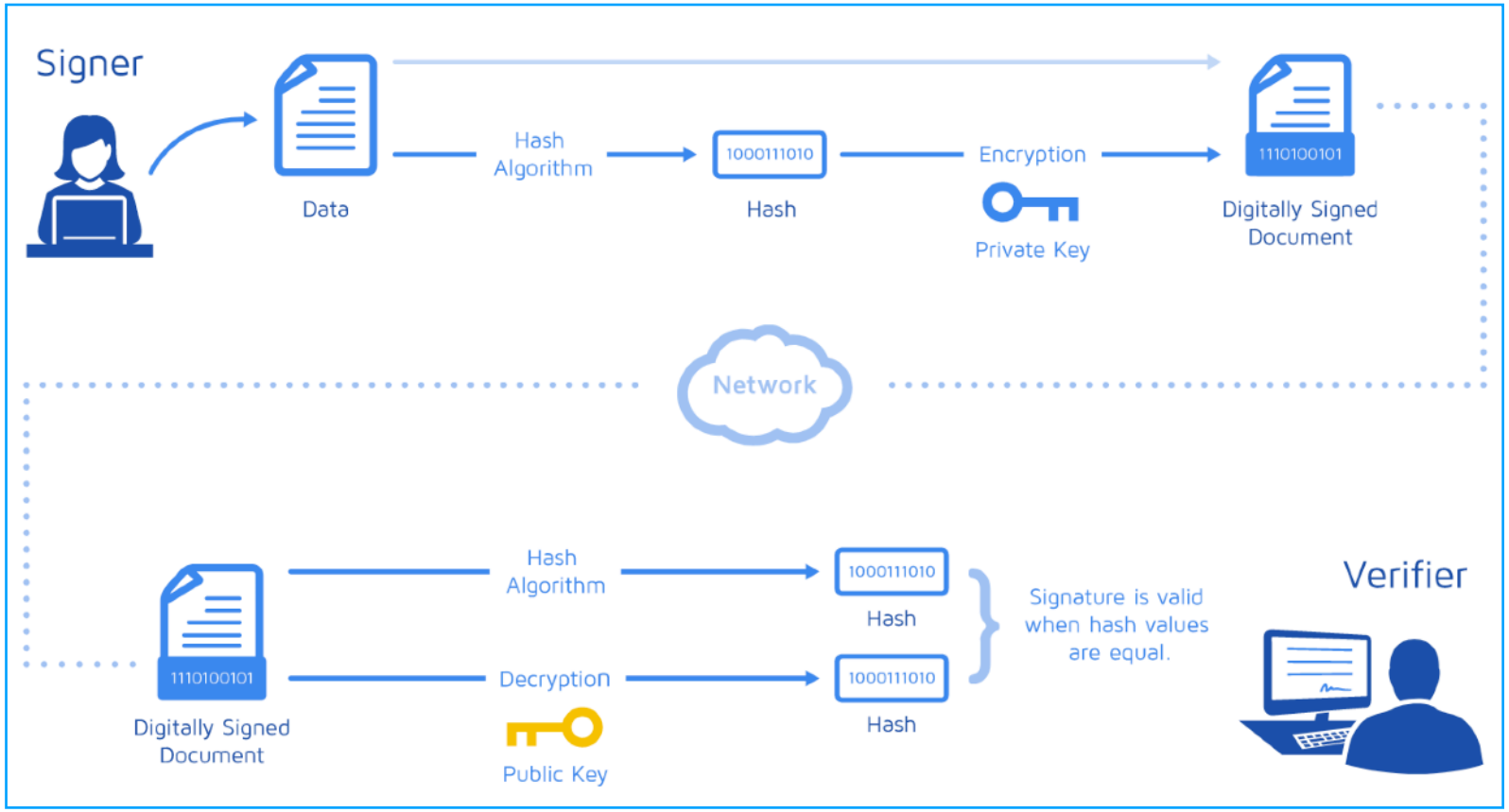


Goal is to prove that it is only Alice (none other) sent this message i.e to prove authenticity of the sender

https://en.wikipedia.org/wiki/Public-key_cryptography

Digital Signatures

- Only you can sign, but anyone can verify
- Signature is tied to a particular document
- Using pair of keys (private and public)



What we want from signatures

Only you can sign, but anyone can verify

Signature is tied to a particular document

can’t be cut-and-pasted to another doc

Public Keys in Blockchain

Useful trick: public key == an identity

if you see *sig* such that *verify(pk, msg, sig)==true*,

think of it as

pk says, "*[msg]*".

to "speak for" *pk*, you must know matching secret key *sk*

https://www.quora.com/What-is-a-simple-example-algorithm-to-explain-to-someone-how-public-key-encryption-works

How to make a new identity

create a new, random key-pair *(sk, pk)*

pk is the public “name” you can use

[usually better to use Hash(pk)]

sk lets you “speak for” the identity

you control the identity, because only you know *sk*

if *pk* “looks random”, nobody needs to know who you are

Thank you!

Raghava Mukkamala

rrm.digi@cbs.dk

https://www.cbs.dk/staff/rrmdigi

https://raghavamukkamala.github.io/

https://cbsbda.github.io/