Blockchain and Its Application

CS 740 (3-1-0) 4

Dr. Sourav Kanti Addya,
Dept of CSE, NIT Karnataka, Surathkal

Crypto Primitives

- Basic cryptographic primitives behind the blockchain technology
  - Cryptographically secured hash function
  - Digital signature
- Hash Function: Used to connect the “blocks” in blockchain in a tamperproof way.
- Digital Signature: Digitally sign the data so that no one can “deny” about their own activities.

Cryptographic Hash Function

- Can be applied to any sized message *M*
- For blockchian produces **fixed-length** output *h* (256 bit used for Blockchain)
- Easy to compute *h*=H(*M*) for any message *M*

Three security Properties

Collision Free

- If two message are different, then there digest also differs

Hiding

- Hide the original message; remember about the **avalanche effect**

Puzzle Friendly

- Given X and Y, find out k such hat Y = H(X||k) – used to solve the mining puzzle in Bitcoin PoW.

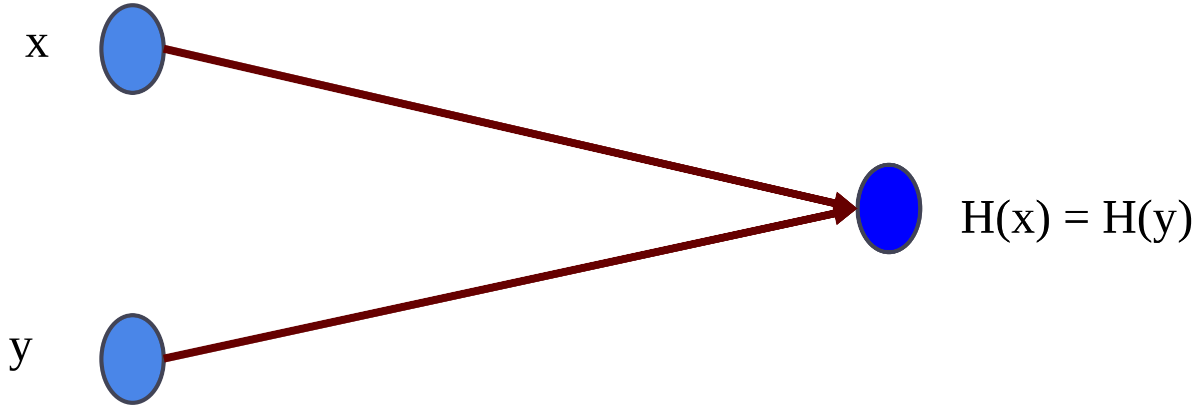
Property 1: Collision-free

It should be difficult to find two different messages m_1 and m_2 such that hash(m_1) = hash(m_2)

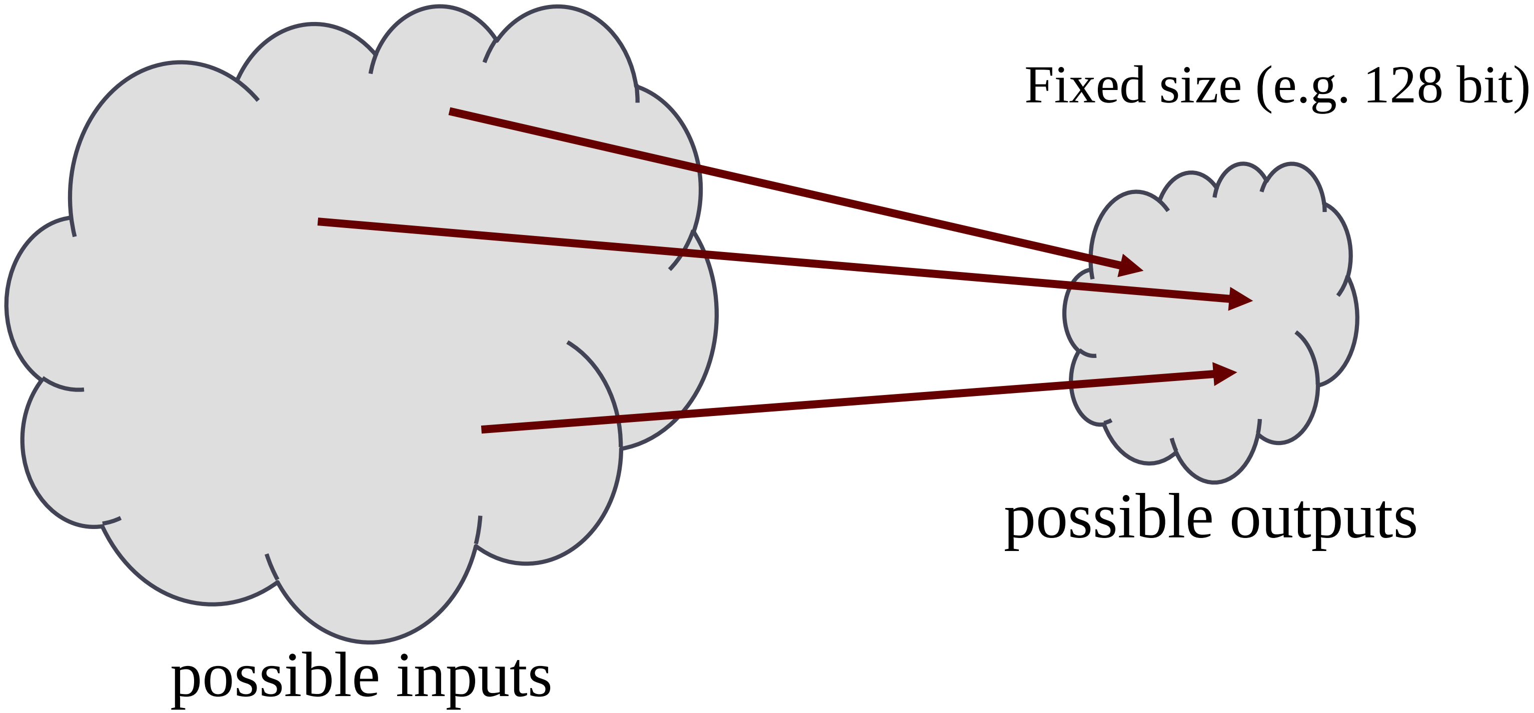
Try with randomly chosen inputs to find out a collision – but it takes too long.

Nobody can find x and y such that

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



Many to One Mapping Exists



Exists Collisions !!

It is unlikely that attacker can find out two such points (inputs) which can produce same output (digest).

How to find Collision?

- Choose inputs randomly
- This works no matter what H is …

- … but it takes too long

Birthday Paradox

- It may be easy to findout collision for some hash function.
- Find the probability that in a set of n randomly chosen people, some of them will have the same birthday.
  - By Pigeonhole Principle, the probability reaches 1 when number of people reaches 366 (not a leap year) or 367 (a leap year).
  - 0.999 probability is reached with just ~70 people, 0.5 probability with only ~23 people.
- Assumptions
  - Nobody was born on February 29
  - People's birthdays are equally distributed over the other 365 days of the year

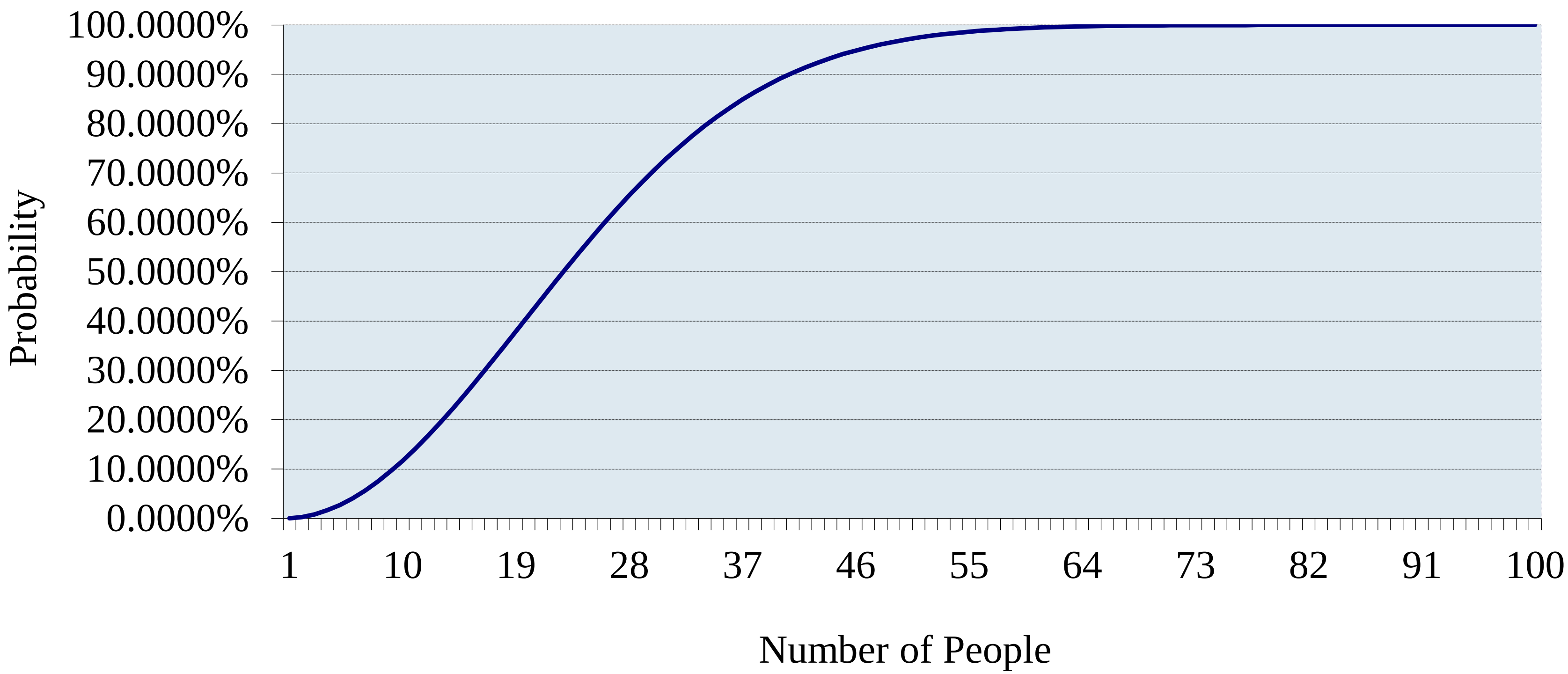
Birthday Attack

- Birthday paradox places an upper bound on collision resistance.
- If a hash function produces N nits of output, an attacker can compute only $2^{N/2}$ hash operations on a random input to find two matching outputs with probability > 0.98.
- For a 256 bit hash function, the attacker needs to compute 2^{128} hash operations – this is significantly time consuming. – if every hash computation takes 1 Millisecond, then it takes ~ 10^{28} years.

The Birthday Paradox

Dr. Sourav Kanti Addya

How large must *k* be so that the probability is greater than 50 percent? The answer is 23



It is a paradox in the sense that a mathematical truth contradicts common intuition

Property 2: Hiding

Given a hash value *h* it should be difficult to find any message *m* such that *h* = hash(*m*)

H("heads")

H("tails")

easy to find x!

Property 2: Hiding (Cont.…)

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.

Property 3: Puzzle Friendly

Puzzle Friendly:

Given X and Y, find out k such that $Y = \mathrm{H}(X \,\|\, k)$

A search Puzzle (used in Bitcoin Mining)

- X and Y given and k is the search solution.

Puzzle friendly property implies that random searching is the best strategy to solve the above puzzle.

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.

The Secure Hash Algorithm (SHA)

- Developed by the National Institute of Standards and Technology (NIST)
- Published as a federal information processing standard (FIPS 180) in 1993
- A revised version was issued as FIPS 180-1 in 1995
  - Generally referred to as SHA-1
- Based on the MD4 algorithm
  - A message digest algorithm that was developed by Ron Rivest at MIT (R of RSA)

SHA

- MD4 was later replaced with the popular MD5 algorithm also by Ron Rivest
  - However advances in cryptanalysis and computing power have led to their decline in popularity
- Both MD4 and MD5 produce a 128 bit message digest
- However, SHA-1 produces a 160 bit
- In 2002, NIST produced a revised version of the standard, FIPS180-2, that defined three new versions of SHA
  - SHA-256, SHA-384, and SHA-512

SHA

- A revised document (FIP PUB 180-3) issued in 2008, which added 224 bit version
- In 2015, NIST issued FIPS 180-4, which added two additional algorithms: SHA-512/224 and SHA512/256

SHA256 Algorithm -

Preprocessing:

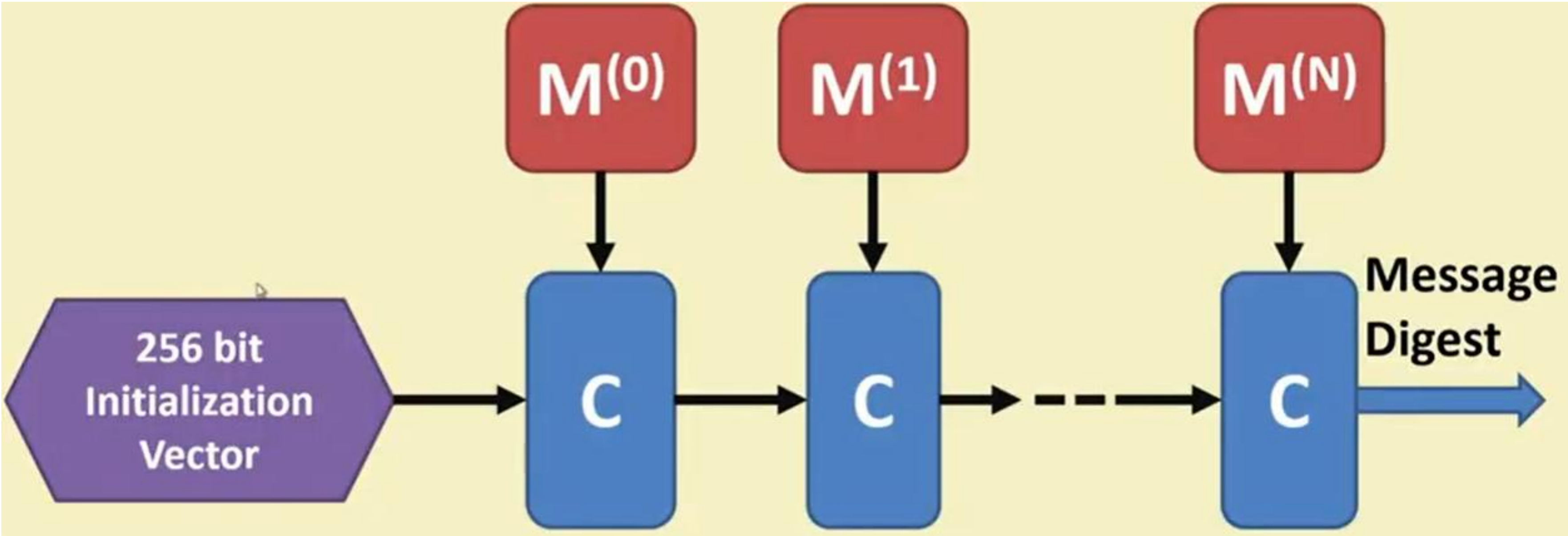
- Pad the message such that the message size is multiple of 512.
  - Suppose that the length of the message M is l; l mod 512 $\neq$ 0
  - Append the bit "1" at the end of the message
  - Append k zero bits, where k is the smallest non-negative solution to the equation l + 1 +k $\equiv$ 448 mod 512
  - Append the 64 bit block wchich is equal to the number l written in binary
  - The total length gets divisible by 512
- Parse the message into N 512 bits blocks $M^{(1)}, M^{(2)}, \ldots, M^{(N)}$
- Every 512 bit block is further divided into 32 bit sub blocks $M_0^{(i)}, M_1^{(i)}, \ldots, M_{15}^{(i)}$

SHA256 Algorithm -

- The message blocks are processed one at a time
- Start with a fix initial hash value $H^{(0)}$
- Sequential compute $H^{(i)} = H^{(i-1)} + C_{M^{(i)}}\left(H^{(i-1)}\right)$; C is the SHA-256 compression function and + means mod 2^{32} addition. $H^{(N)}$ is the hash of M. C is the compression function.

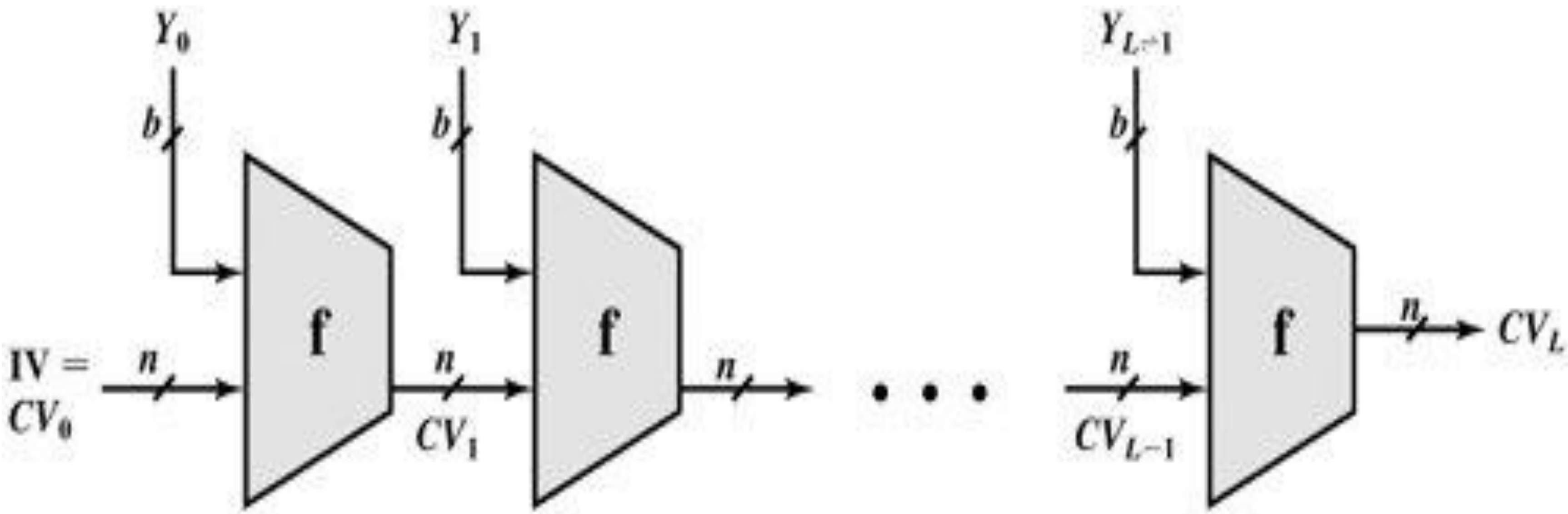
SHA256 Algorithm -

Dr. Sourav Kanti Addya



Dr. Sourav Kanti Addya

General structure of Secure Hash code



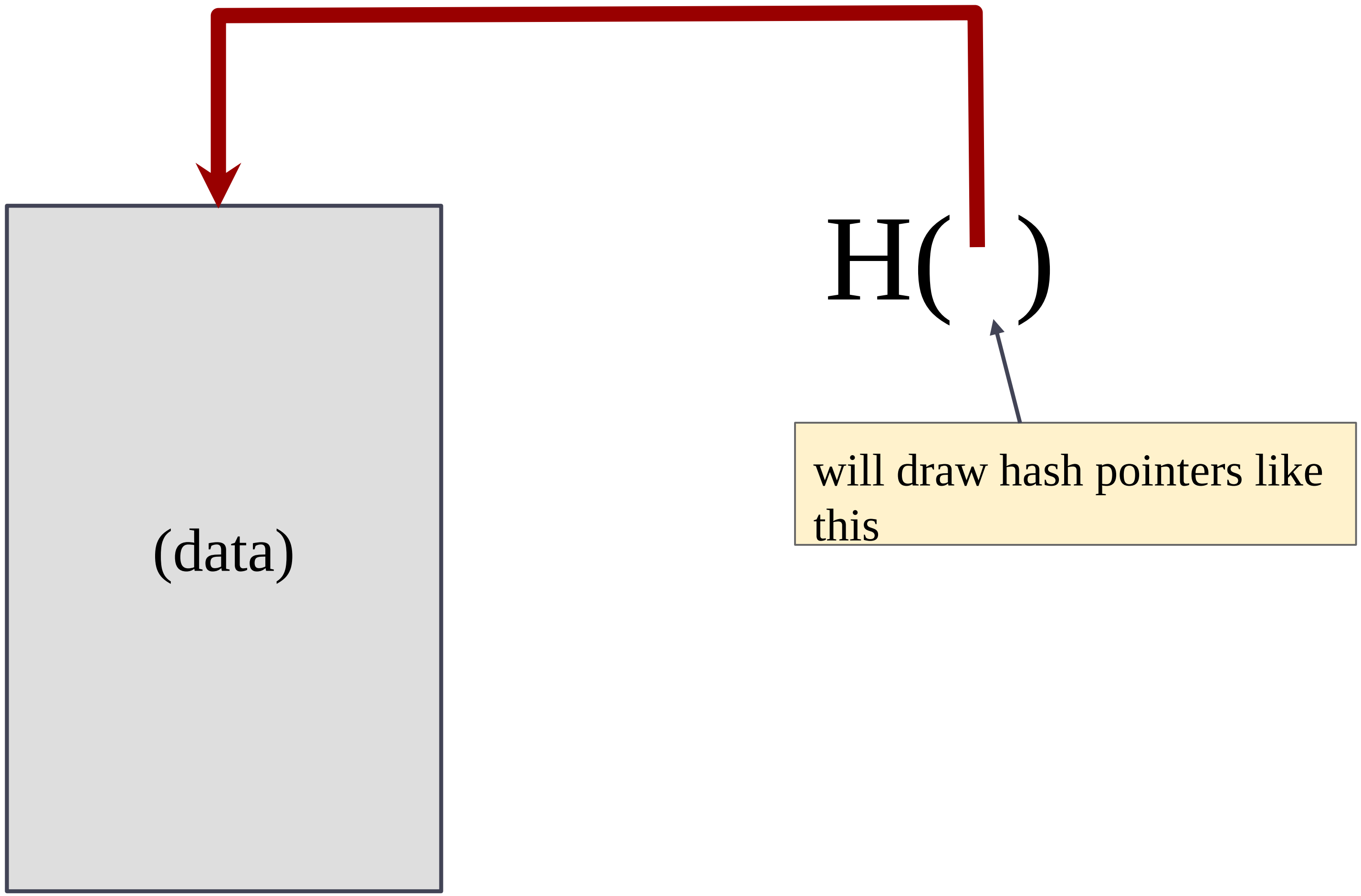
IV = Initial value
CV_i = Chaining variable
Y_i = ith input block
f = Compression algorithm

L = Number of input blocks
n = Length of hash code
b = Length of input block

Hash Pointer

- Point to where some info is stored, and (cryptographic) hash of the info
- Hash of the information is stored

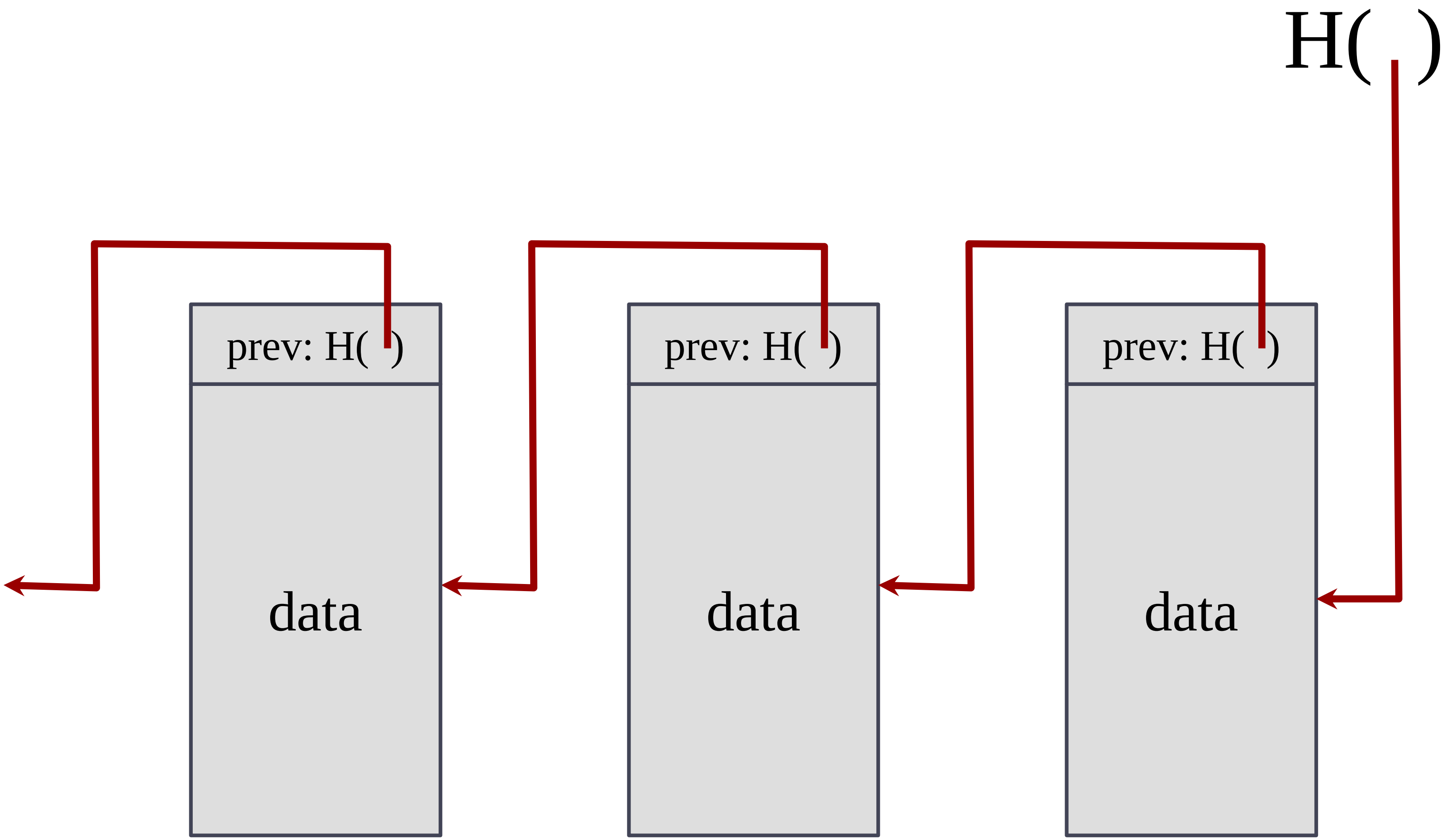
- If we have a hash pointer, we can
  - Ask to get the info back (Retrieve the information), and
  - Verify that it hasn’t changed (by computing the message digest and then matching the digest with the stored hash value).


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

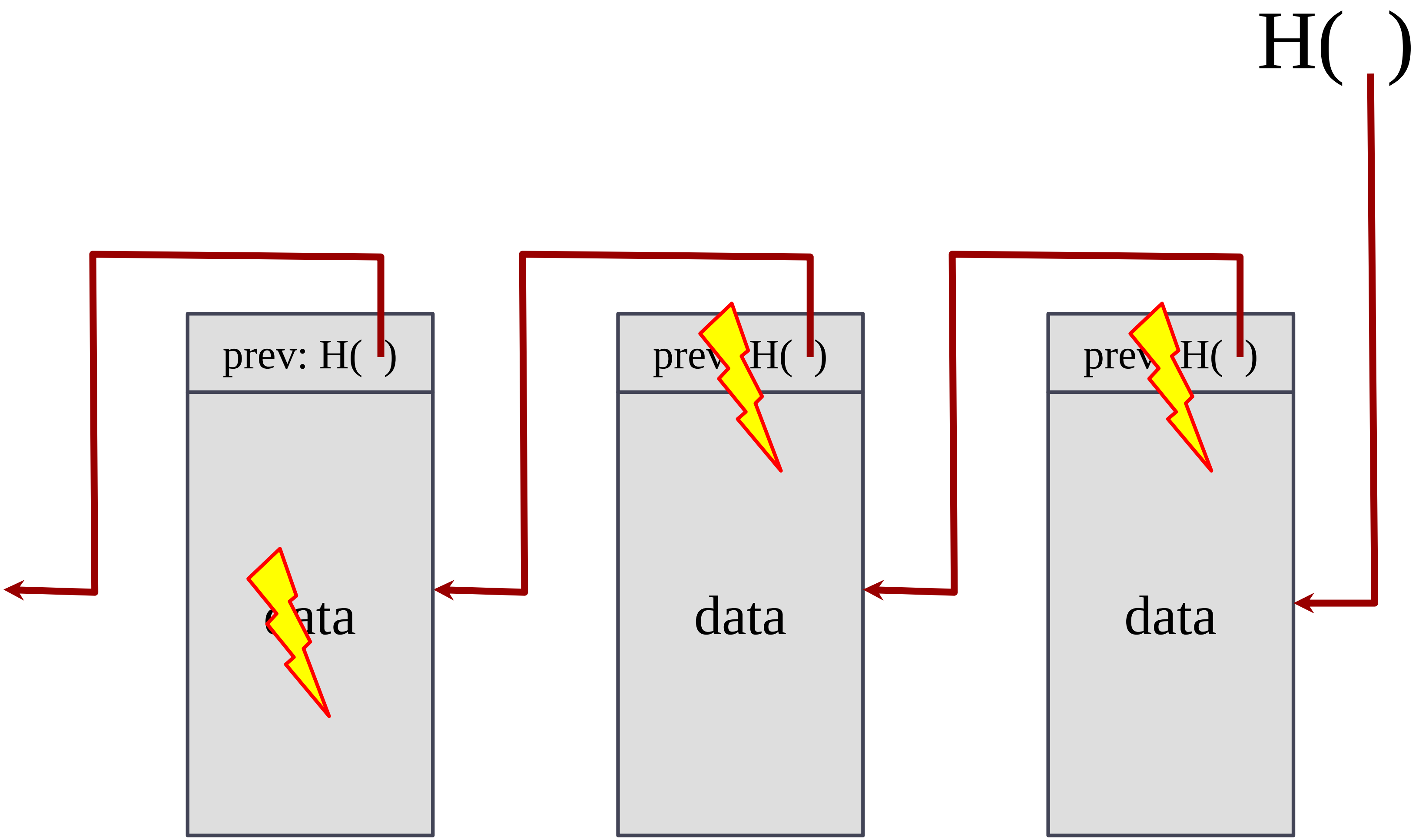
Key Idea:

Build data structures with hash pointers

Linked list with hash pointers

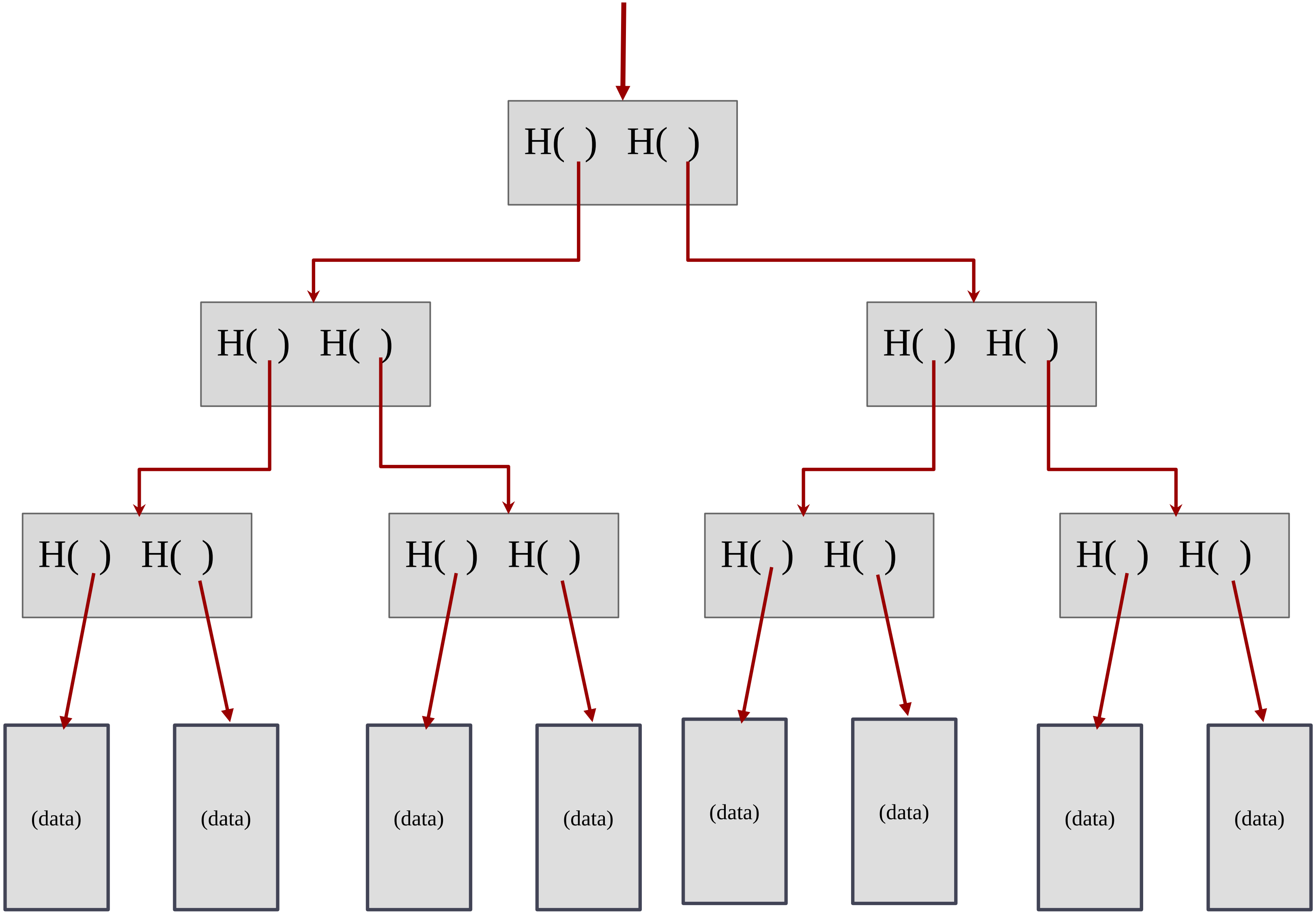


Detecting tampering



use case: tamper-evident log

Binary tree with hash pointers = “Merkle tree” (Ralph Markle, 1979)/ Hash Tree



Digital Signature

- A digital signature is a digital code, which can be include with as electronically transmitted document to verify
  - The content of the document is authenticated
  - The identity of the sender
  - Prevent non-repudiation – sender will not be abel to deny about the origin of the document

Purpose

- Only the signing authority can sign a document, but everyone can verify the signature.
- Signature is associated with the particular document
  - Signature of one document cannot be transferred to another document.

- Reading **Public Key Cryptography**