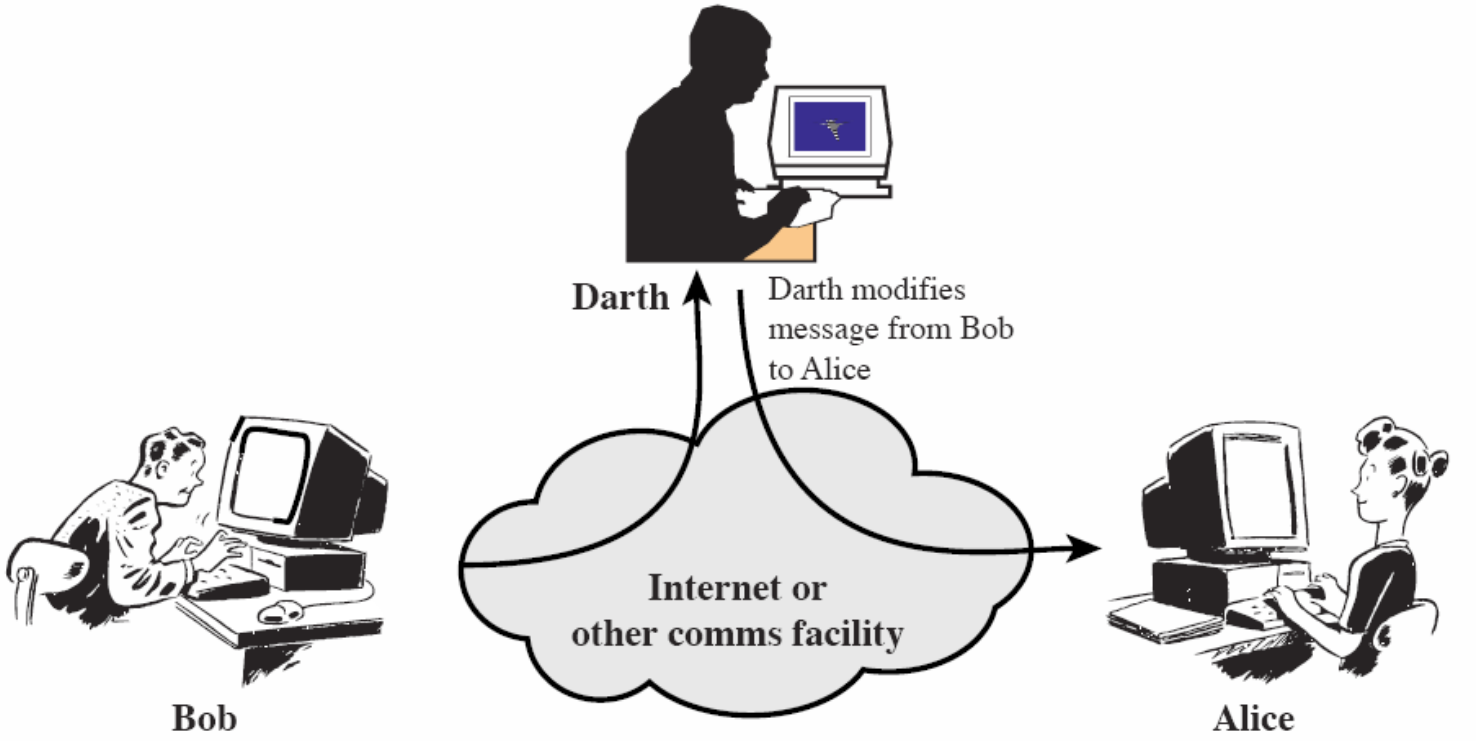
usfCS
UNIVERSITY of SAN FRANCISCO
department of computer science

Hash Functions

EJ Jung

Integrity checks

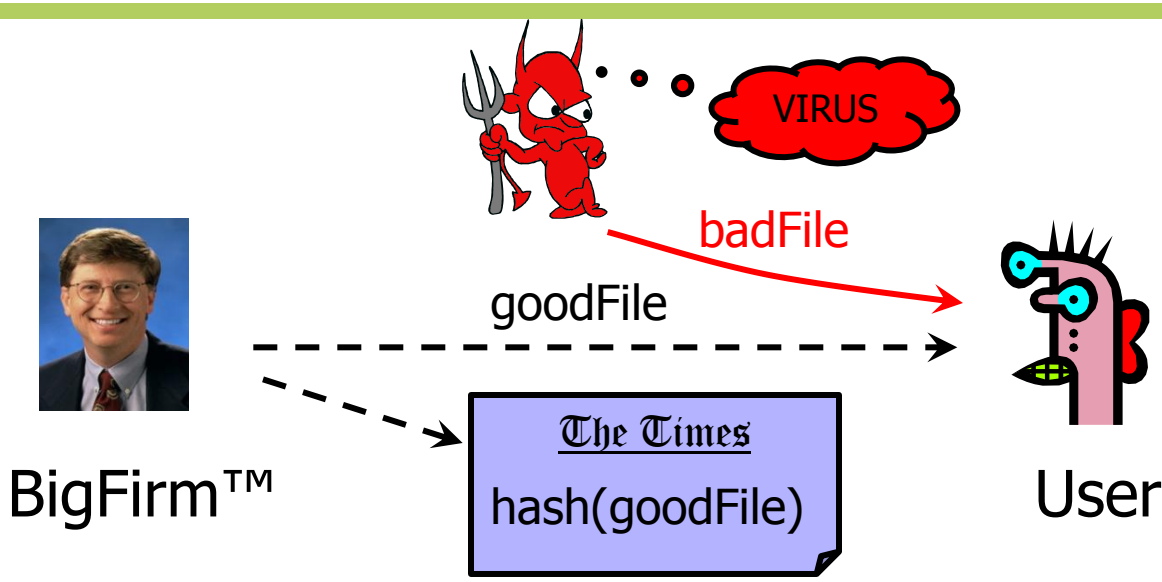


(c) Modification of messages

Integrity vs. Confidentiality

- Integrity: attacker cannot tamper with message
- Encryption may not guarantee integrity!
  - Intuition: attacker may able to modify message under encryption without learning what it is
    - Given one-time key K, encrypt M as M⊕K… Perfect secrecy, but can easily change M under encryption to M⊕M' for any M'
    - Online auction: halve competitor's bid without learning its value
  - This is recognized by industry standards (e.g., PKCS)
    - "RSA encryption is intended primarily to provide confidentiality… It is not intended to provide integrity"
  - Many encryption schemes provide secrecy AND integrity

More on Integrity



Software manufacturer wants to ensure that the executable file is received by users without modification…

Sends out the file to users and publishes its hash in NY Times

The goal is integrity, not confidentiality

Idea: given goodFile and hash(goodFile),
very hard to find badFile such that hash(goodFile)=hash(badFile)

Hash Functions

- Purpose of the HASH function is to produce a ”fingerprint”.
- But, what do you mean by fingerprint??

Secure Hash Functions

- Properties of a HASH function H :
  1. H can be applied to a block of data at any size
  2. H produces a fixed length output
  3. H(x) is easy to compute for any given x.
  4. For any given block x, it is computationally infeasible to find x such that H(x) = h
  5. For any given block x, it is computationally infeasible to find $y \neq x$ with H(y) = H(x).
  6. It is computationally infeasible to find any pair (x, y) such that H(x) = H(y)

Simple hash function

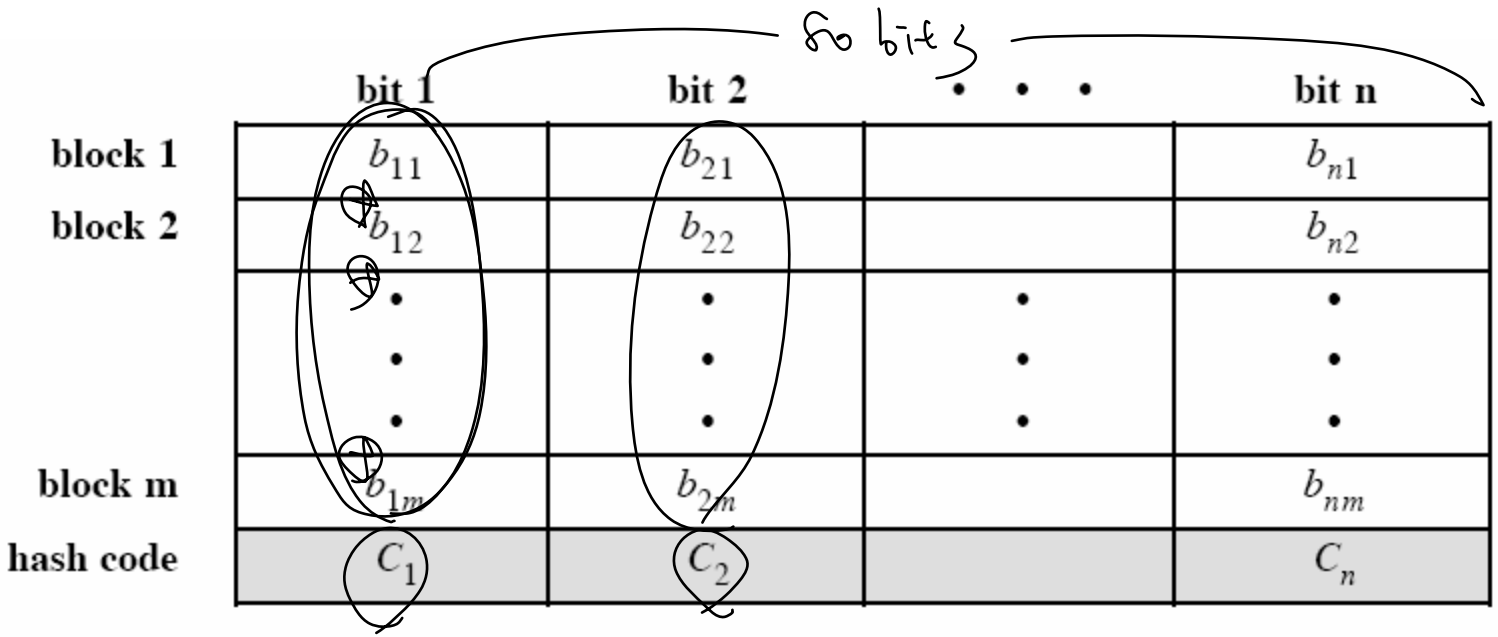
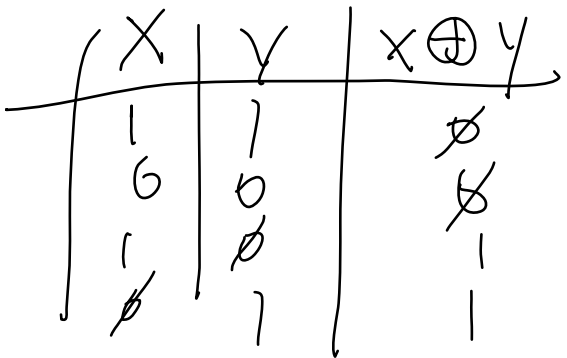
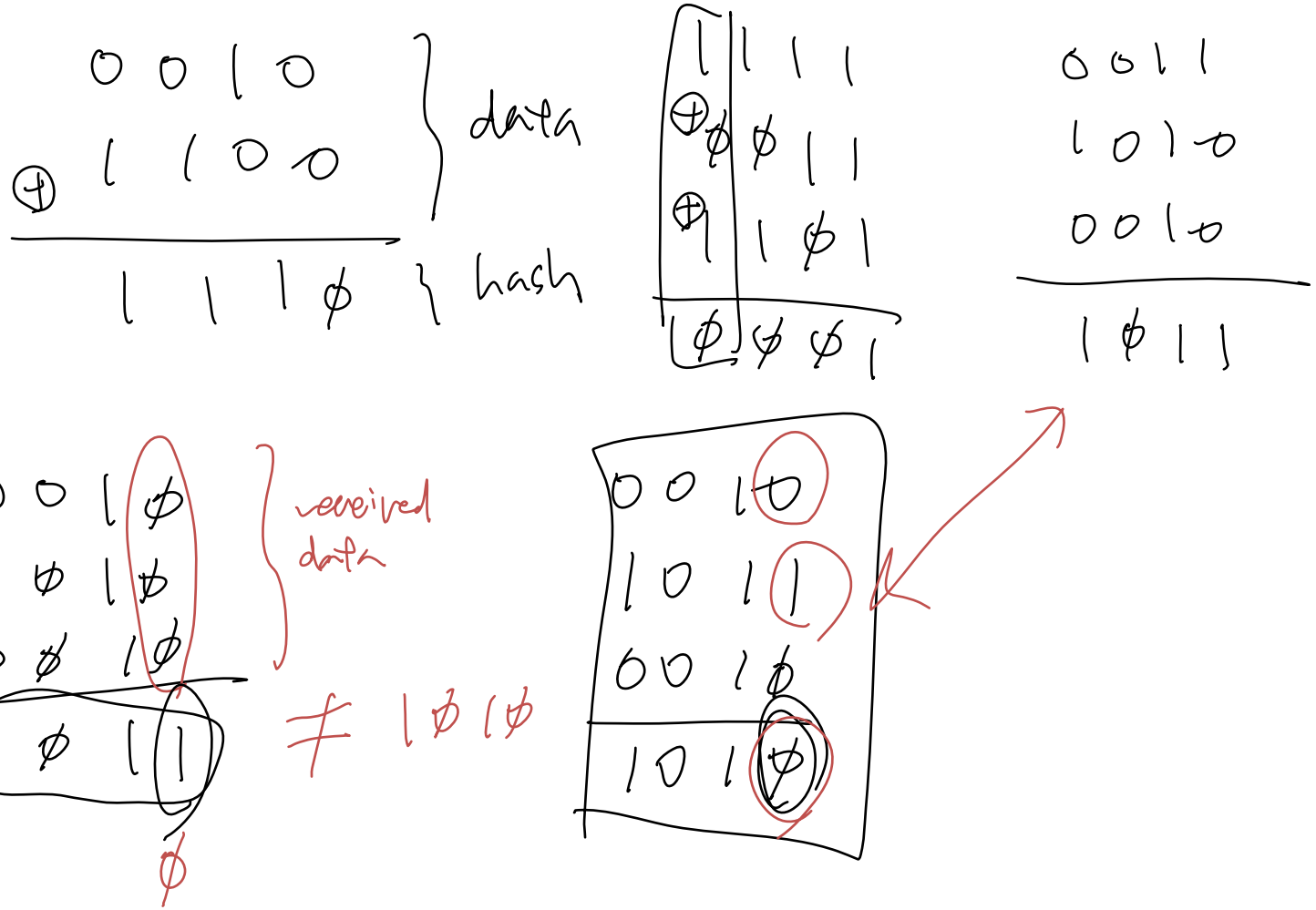


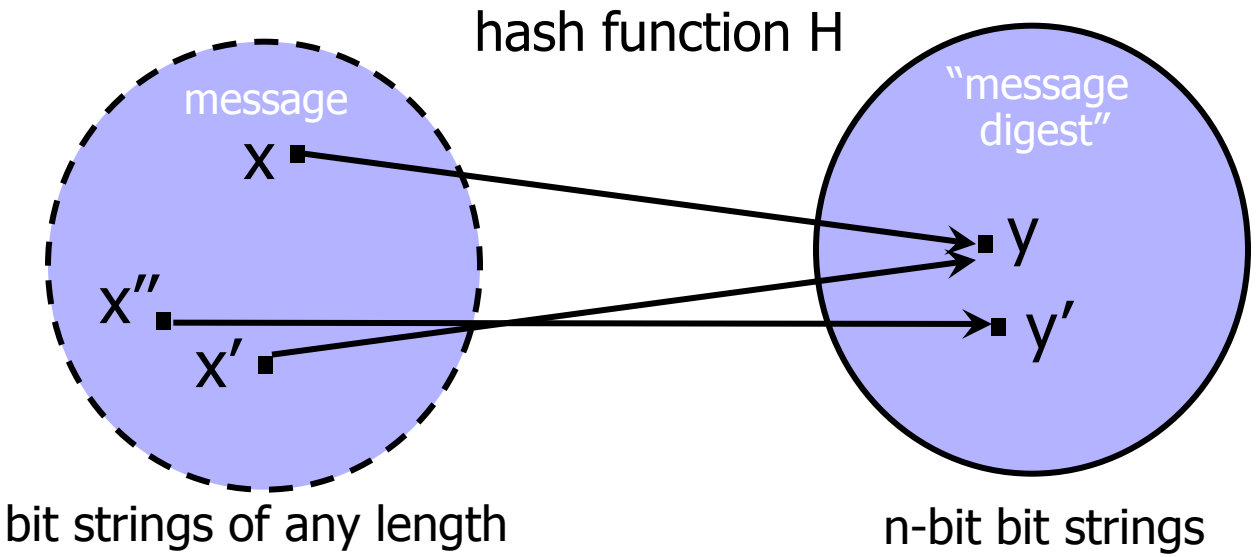
Figure 3.3 Simple Hash Function Using Bitwise XOR



Example



Hash Functions: Main Idea



- H is a lossy compression function
  - Collisions: h(x)=h(x′) for some inputs x, x′
  - Result of hashing should "look random" (make this precise later)
    - Intuition: half of digest bits are "1"; any bit in digest is "1" half the time
- Cryptographic hash function needs a few properties…

One-Way

- Intuition: hash should be hard to invert
  - "Preimage resistance"
  - Let $h(x')=y \in \{0,1\}^n$ for a random x'
  - Given y, it should be hard to find any x such that $h(x)=y$
- How hard?
  - Brute-force: try every possible x, see if $h(x)=y$
  - SHA-1 (common hash function) has 160-bit output
    - Suppose have hardware that'll do 2^{30} trials a pop
    - Assuming 2^{34} trials per second, can do 2^{89} trials per year
    - Will take 2^{71} years to invert SHA-1 on a random image

usfcs
UNIVERSITY of SAN FRANCISCO
department of computer science

"Birthday Paradox"

- T people
- Suppose each birthday is a random number taken from K days (K=365) – how many possibilities?
  - K^T (samples with replacement)
- How many possibilities that are all different?
  - $(K)_T = K(K-1)\ldots(K-T+1)$ samples without replacement
- Probability of no repetition?
  - $(K)_T/K^T \approx 1 - T(T-1)/2K$
- Probability of repetition?
  - $O(T^2)$

usfcs UNIVERSITY of SAN FRANCISCO department of computer science

Collision Resistance

- Should be hard to find x, x′ such that h(x)=h(x′)
- Brute-force collision search is $O(2^{n/2})$, not $O(2^n)$
  - n = number of bits in the output of hash function
  - For SHA-1, this means $O(2^{80})$ vs. $O(2^{160})$
- Reason: birthday paradox
  - Let T be the number of values x,x′,x″… we need to look at before finding the first pair x,x′ s.t. h(x)=h(x′)
  - Assuming h is random, what is the probability that we find a repetition after looking at T values? $O(T^2)$
  - Total number of pairs? $O(2^n)$
  - Conclusion: $T \approx O(2^{n/2})$

One-Way vs. Collision Resistance

- One-wayness does not imply collision resistance
  - Suppose g is one-way
  - Define h(x) as g(x') where x' is x except the last bit
    - h is one-way (to invert h, must invert g)
    - Collisions for h are easy to find: for any x, h(x0)=h(x1)
- Collision resistance does not imply one-wayness
  - Suppose g is collision-resistant
  - Define h(x) to be 0x if x is n-bit long, 1g(x) otherwise
    - Collisions for h are hard to find: if y starts with 0, then there are no collisions, if y starts with 1, then must find collisions in g
    - h is not one way: half of all y's (those whose first bit is 0) are easy to invert (how?); random y is invertible with probab. 1/2