CS 4770: Cryptography

CS 6750: Cryptography and Communication Security

Alina Oprea
Associate Professor, CCIS
Northeastern University

March 12 2018

Announcements

- Homework 3 will be out today
  - Due date Fri 03/23
- Distinguished speaker on Thu 03/22
  - Location 97 Cargill, 3-4:30pm
  - Prof Mike Reiter, UNC Chappel Hill
  - Title: “Side channels in multi-tenant environments”
  - Extra credit for next homework: submit a paragraph about his talk
- If anyone is interested in meeting him 4:30-5pm, please email me

Recap

- Collision-resistant hash functions are useful for many tasks
- Constructing hash functions using Merkle-Daamgard paradigm
  - Traditional designs: MD5, SHA-1, SHA-2
- SHA-3 is the new standard
  - Explicit collision found in MD5
  - Structural waeknesses in SHA-1
- Birthday paradox implies n/2 level of security for n-bit hash function in best case

Outline

- Birthday attack
  - Prove lower bound
  - Generic attack on hash functions
- Construction of HMAC
  - More efficient than CBC-MAC
- Applications of hash functions
  - Merkle trees
- Introduction to number theory

Collision-resistant hash functions

short **H(m)**

a hash function

$H : \{0,1\}^* \rightarrow \{0,1\}^n$

long **m**

collision-resistance

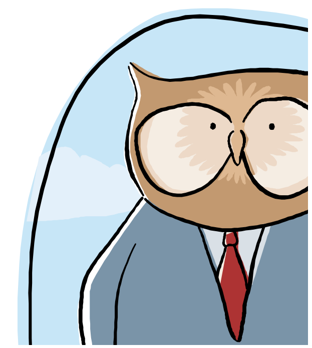
a "collision"

Requirement: it should be hard to find a pair **(m,m')** such that **H(m) =H(m')**

Hash functions – the security definition

security parameter
n

outputs **(m,m’)**



H is a **collision-resistant hash function** if

$\forall$ polynomial-time adversary **A**

Pr[A outputs m, m’ such that H(m)=H(m’)]
is negligible

Birthday paradox

- If we choose q elements $y_1, \ldots y_q$ at random from {1,…,N}, what is the probability that there exists i and j such that $y_i = y_j$?

365 possible days

What is the probability that two people have the same birthday?

Upper bound

- If we choose $y_1, \ldots y_q$ uniformly at random from {1,…,N}, the probability of collision is upper bounded by:

$$\mathrm{Coll}(q, N) \leq \frac{q(q-1)}{2N}$$

- Proof: (Union bound)

$$\Pr[\mathrm{Coll}(q, N)] = \Pr\left[\exists\, i, j\ st\ y_i = y_j\right]$$

$$\leq \sum_{i,j} \Pr\left[y_i = y_j\right] = \binom{q}{2} \frac{1}{N} = \frac{q(q-1)}{2N}$$

Lower bound

- If we choose $y_1, \ldots y_q$ uniformly at random from {1,…,N} and $q \leq \sqrt{2N}$, the probability of collision is lower bounded by:

$$\mathrm{Coll}(q, N) \geq 1 - e^{-\frac{q(q-1)}{2N}} \geq \frac{q(q-1)}{4N}$$

- Proof: NoColl_i = Event no collision in $y_1, \ldots y_i$

$\Pr[\mathrm{NoColl}_q] = \Pr[\mathrm{NoColl}_1] \Pr[\mathrm{NoColl}_2 | \mathrm{NoColl}_1] \ldots \Pr[\mathrm{NoColl}_q | \mathrm{NoColl}_{q-1}]$

$\Pr[\mathrm{NoColl}_1] = 1$

$\Pr[\mathrm{NoColl}_i | \mathrm{NoColli}_{i-1}] = 1 - (i-1)/N$

Lower bound

- If we choose $y_1, \ldots y_q$ uniformly at random from {1,…,N} and $q \leq \sqrt{2N}$, the probability of collision is lower bounded by:

$$\mathrm{Coll}(q, N) \geq 1 - e^{-\frac{q(q-1)}{2N}} \geq \frac{q(q-1)}{4N}$$

- Proof: NoColl_i = Event no collision in $y_1, \ldots y_i$

$$\Pr[\mathrm{NoColl}_q] = \prod (1 - i/N)$$

$$\Pr[\mathrm{NoColl}_q] \leq \prod_i e^{-i/N} \leq e^{-\sum i/N} = e^{-q(q-1)/2N}$$

$$1 - \Pr[\mathrm{NoColl}_q] \geq 1 - e^{-q(q-1)/2N}$$

$$\geq q(q-1)/4N$$

Lower bound

- If we choose $y_1, \dots y_q$ uniformly at random from {1,…,N} and $q \leq \sqrt{2N}$, the probability of collision is lower bounded by:

$$\frac{q(q-1)}{4N} \leq \mathrm{Coll}(q, N) \leq \frac{q(q-1)}{2N}$$

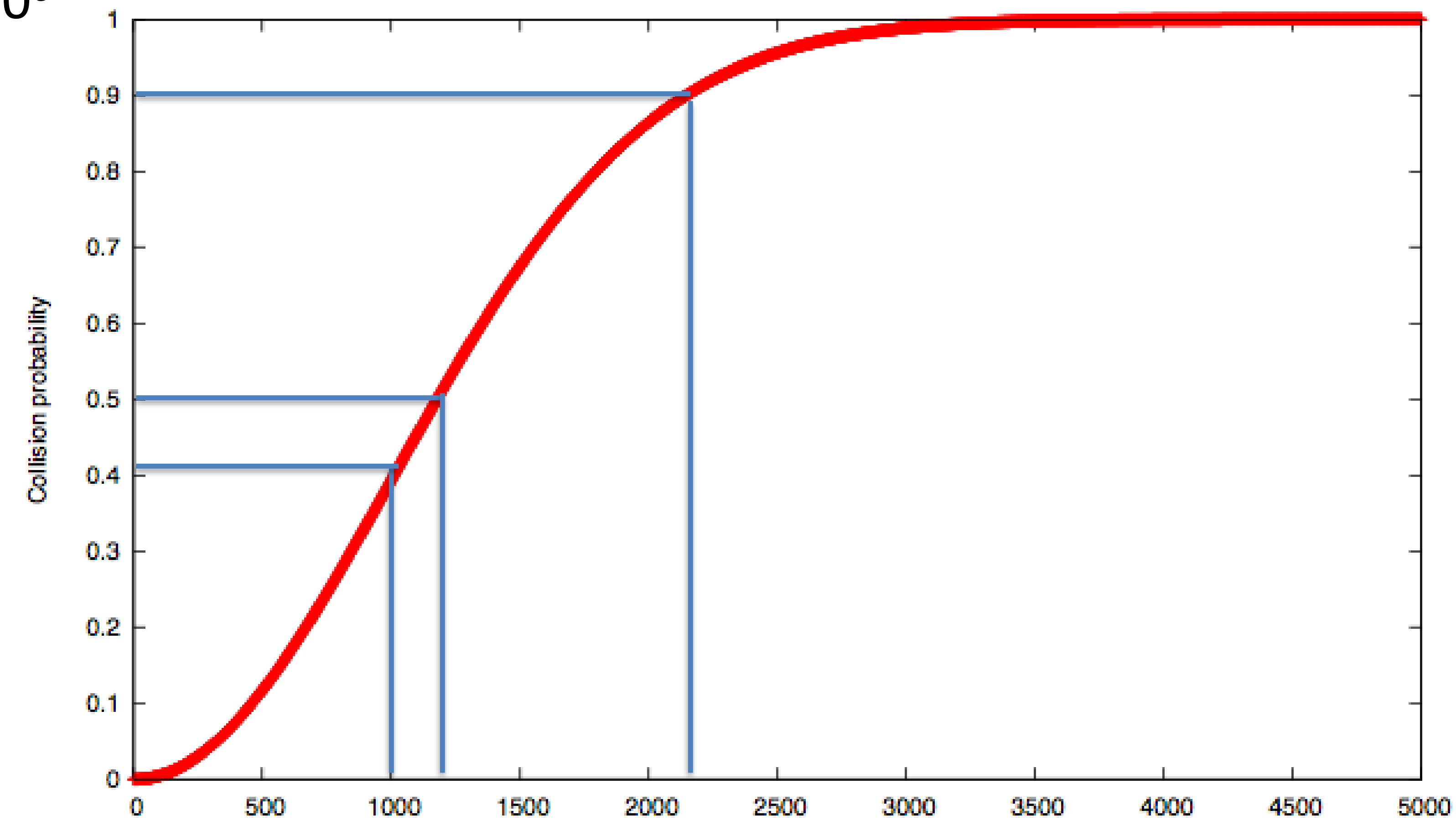
If $q = \Theta\left(\sqrt{N}\right)$, then $\mathrm{Coll}(q, N)$ is approx. ½

Birthday paradox: N = 365, q = 23

Hash functions: $N = 2^n$, $q = 2^{n/2}$

Collision probability

$N=10^6$



samples n

Generic attack on collision resistant hash functions

Let H: M → $\{0,1\}^n$ be a hash function (|M| >> 2^n)

Generic alg. to find a collision **in time** $\mathbf{O(2^{n/2})}$ hashes

Algorithm:

1. Choose $2^{n/2}$ random messages in M: $m_1, \ldots, m_{2^{n/2}}$ (distinct w.h.p)
2. For i = 1, …, $2^{n/2}$ compute $t_i = H(m_i)$
3. Look for a collision ($t_i = t_j$)
4. If not found, got back to step 1

Running time: $\mathbf{O(2^{n/2})}$ (space $O(2^{n/2})$)

Sample C.R. hash functions:

Crypto++ 5.6.0 [Wei Dai]

AMD Opteron, 2.2 GHz (Linux)

	function	digest size (bits)	Speed (MB/sec)	generic attack time
NIST standards	SHA-1	160	153	2^{80}
	SHA-256	256	111	2^{128}
	SHA-512	512	99	2^{256}

Best known collision finder for SHA-1 requires 2^{51} hash evaluations

Security experiment for MAC

- Experiment $\text{Exp}_{\Pi,A}^{\text{MAC}}(n)$:
  1. Choose $k \leftarrow Gen(n)$
  2. *m,t* $\leftarrow A^{Tag()}(n)$
  3. Output 1 if Ver(*m,t*) = 1 and *m* was not queried to the Tag() oracle
  4. Output 0 otherwise

We say that **(Gen, Tag,Ver)** is **a secure** MAC if:

For every **PPT** adversary $A = (A_1, A_2)$:

Pr[$\text{Exp}_{\Pi,A}^{\text{MAC}}(n)$ = 1] is negligible in n

MACs from Collision Resistance

Let (Tag,Ver) be a MAC for short messages over (K,M)

Let H: $M' \to M$ be a collision resistant hash function

Def: (Tag', Ver') over (K, M') as:

$Tag'(k,m) = Tag(k,H(m))$ **$Ver'(k,m,t) = Ver(k,H(m),t)$**

Thm: If (Tag,Ver) is a secure MAC and H is collision resistant then (Tag', Ver') is a secure MAC.

Example: (k,m) = CBC-MAC(k, SHA-256(m)) is a secure MAC.

MACs from Collision Resistance

$Tag'(k, m) = Tag(k, H(m))$; $Ver'(k, m, t) = Ver(k, H(m), t)$

Collision resistance is necessary for security:

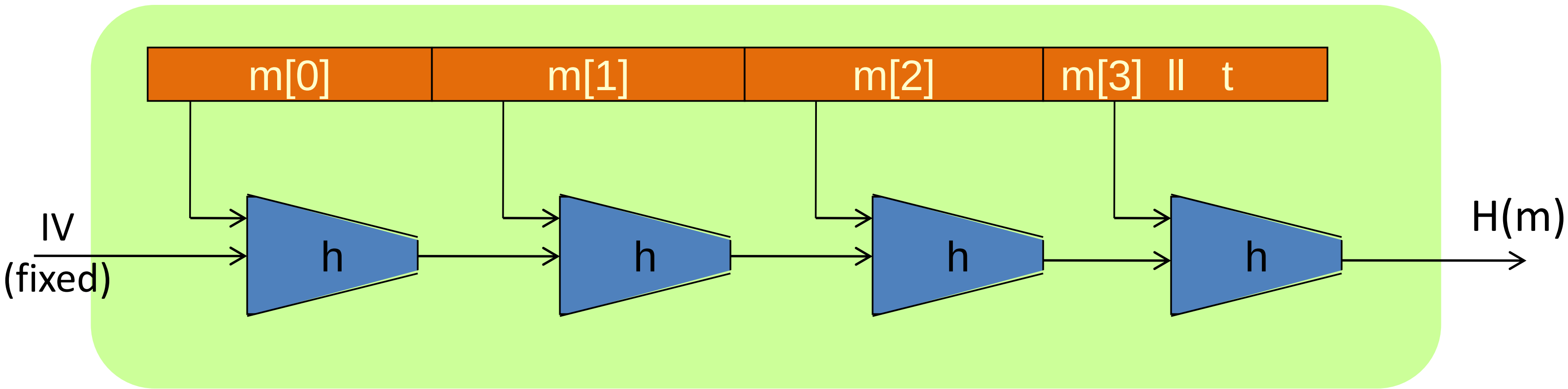
Suppose adversary can find $m_0 \neq m_1$ s.t. $H(m_0) = H(m_1)$

Then: **(Tag',Ver')** is insecure under chosen msg attack

step 1: adversary asks for $t \longleftarrow Tag(k, m_0)$

step 2: output (m_1, t) as forgery

The Merkle-Damgard iterated construction



Thm: h collision resistant ⇒ H collision resistant

Can we use H(.) to directly build a MAC?

MAC from a Merkle-Damgard Hash Function

$H: X^{\leq L} \longrightarrow T$ a C.R. Merkle-Damgard Hash Function

Attempt #1: **Tag(k, m) = H(k ll m)**

This MAC is insecure because:

Given H(k ll m) can compute H(w ll k ll m ll t) for any w.

Given H(k ll m) can compute H(k ll m ll w) for any w.

→ Given H(k ll m) can compute H(k ll m ll t ll w) for any w.

Anyone can compute H(k ll m) for any m.

Standardized method: HMAC (Hash-MAC)

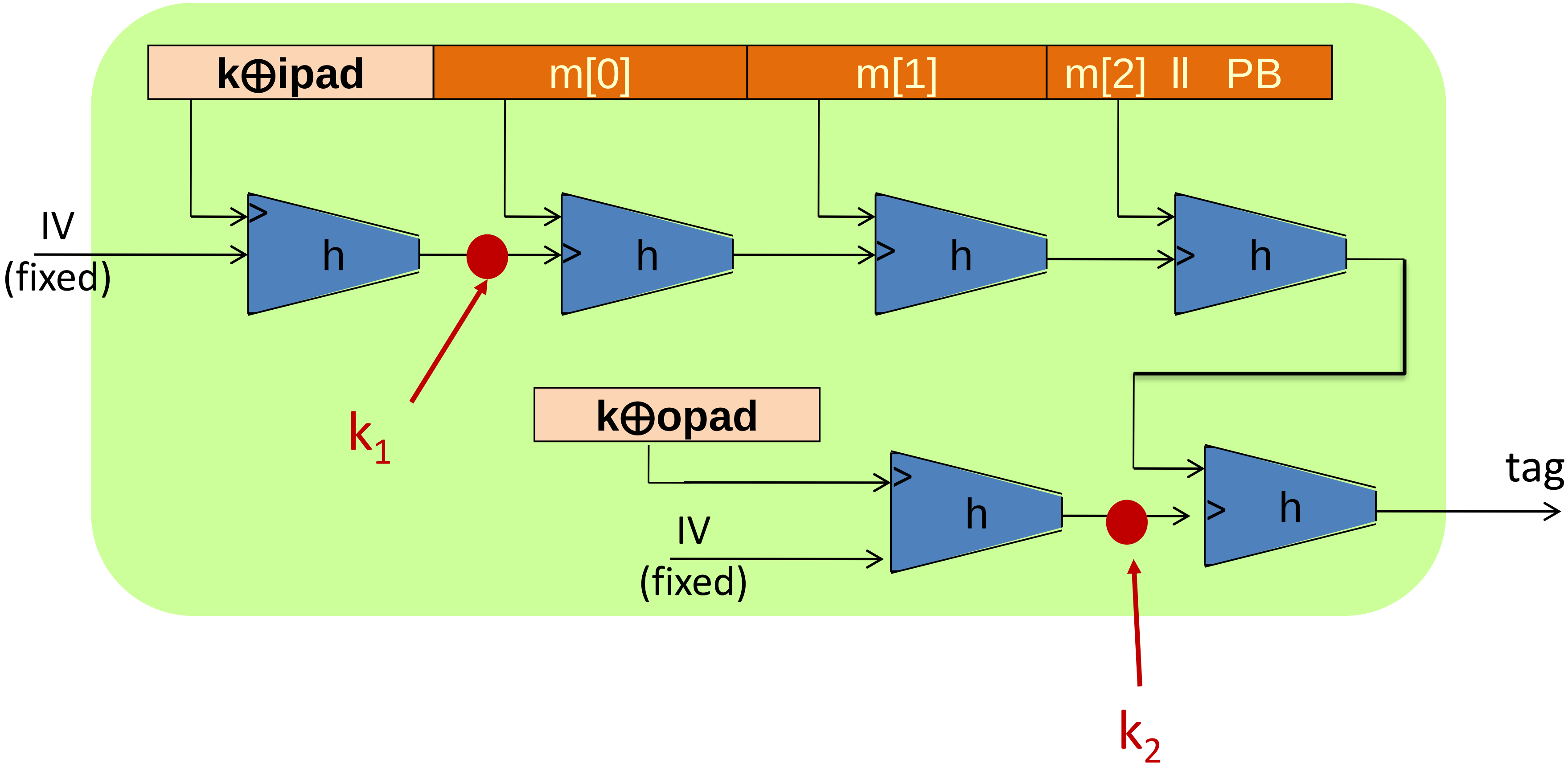
Most widely used MAC on the Internet.

H:	hash function.
	example: SHA-256 ; output is 256 bits

Building a MAC out of a hash function:

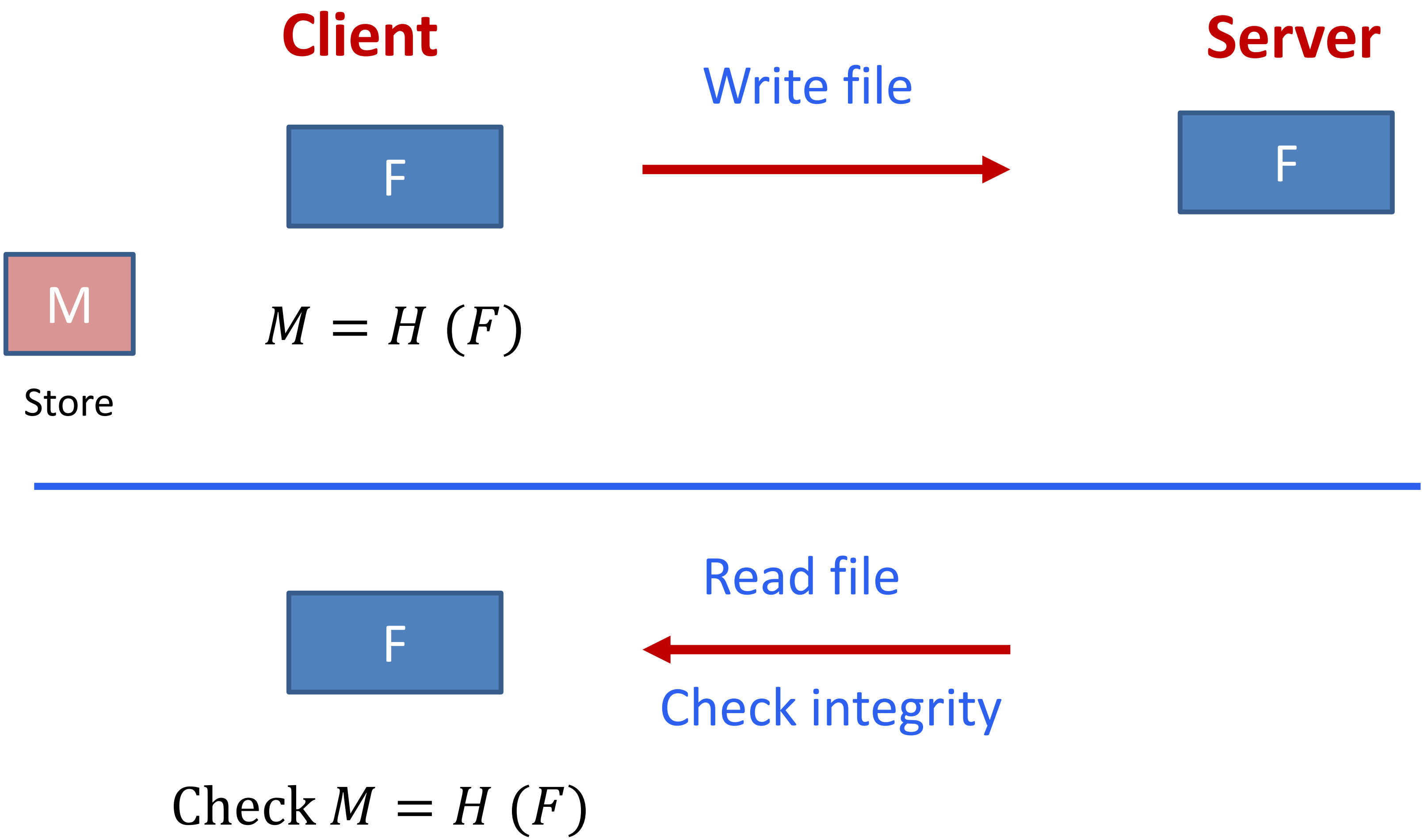
$$\text{HMAC: Tag}(k,M) = H(k \oplus \text{opad}, H(k \oplus \text{ipad} \,||\, m))$$

HMAC in pictures

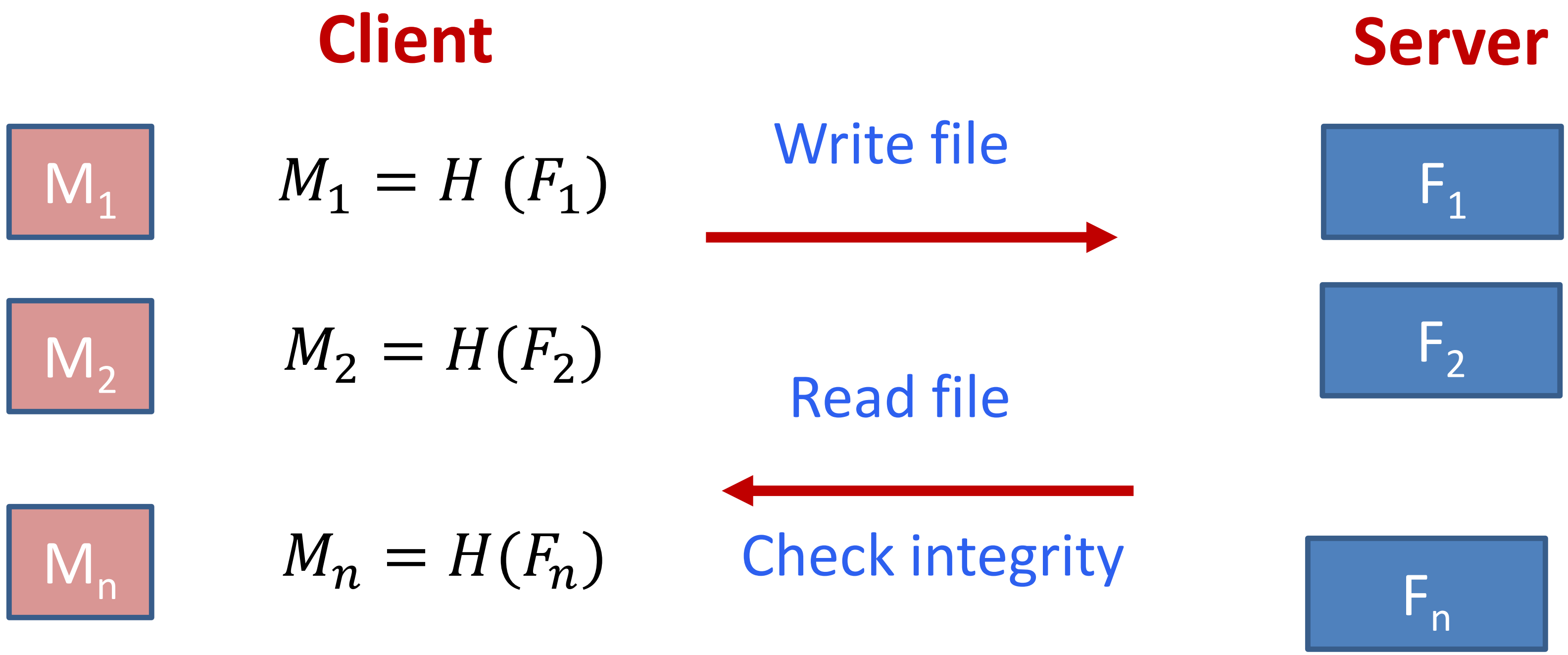


Applications of hash functions: Merkle trees

Authenticate a file using its hash



How to authenticate multiple files?

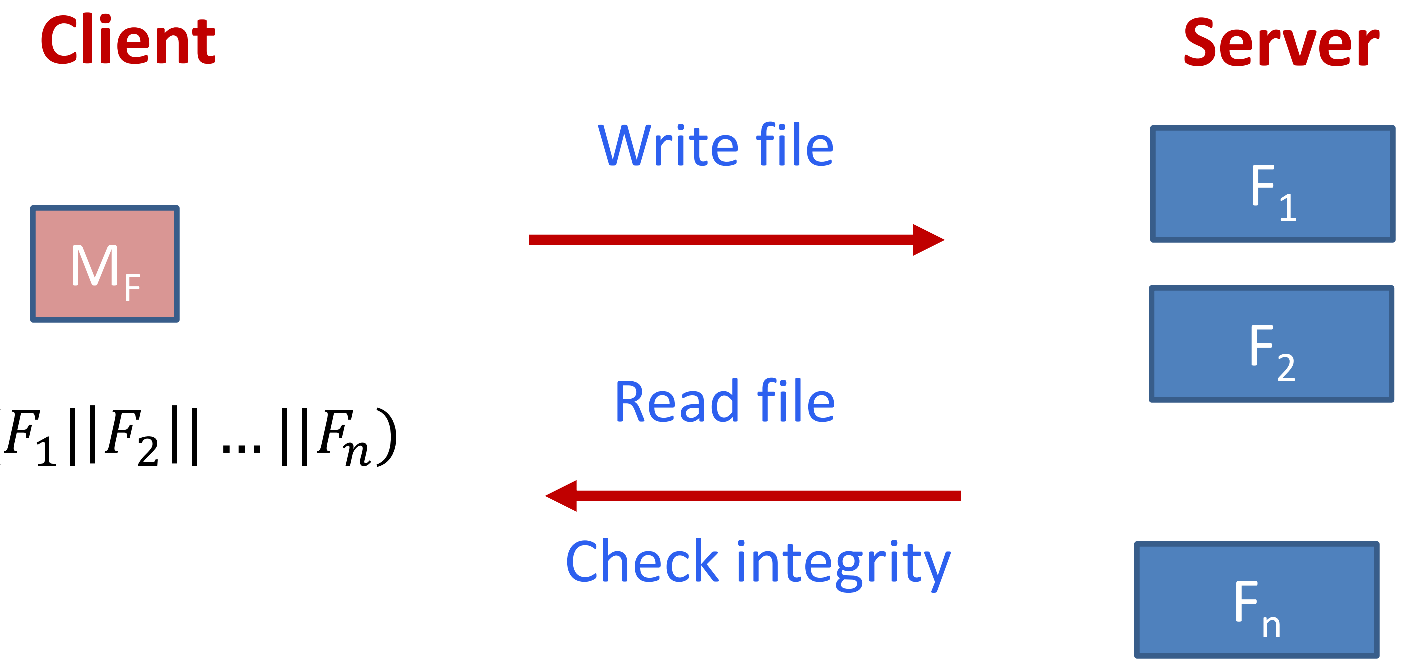


1. Compute and store a hash per file

   + Fast to check integrity and update file

   - Linear storage on client

How to authenticate multiple files?



$$M_F = H(F_1||F_2||\ldots||F_n)$$

2. Compute and store a hash for all files

+ Small storage on client

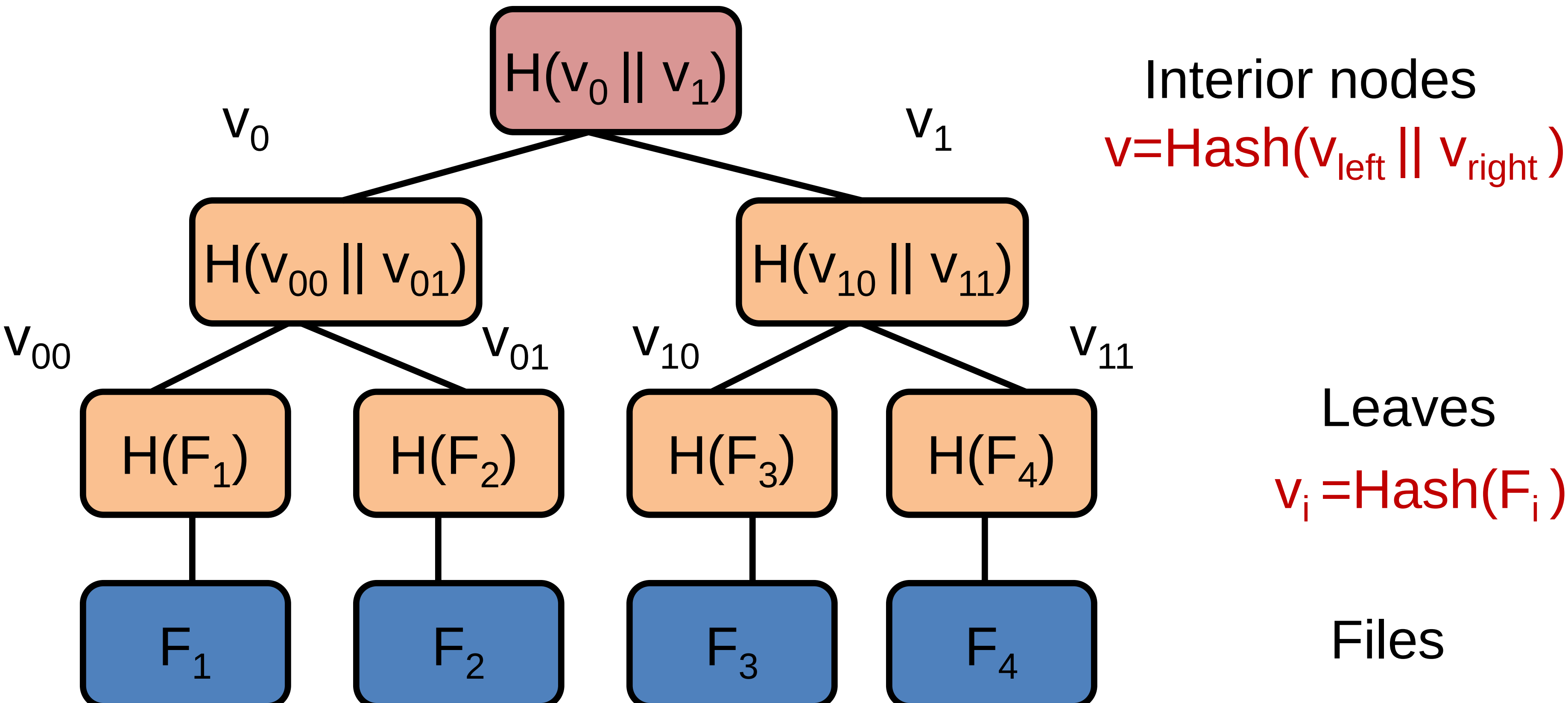
- Linear time to check integrity and update file

Merkle trees

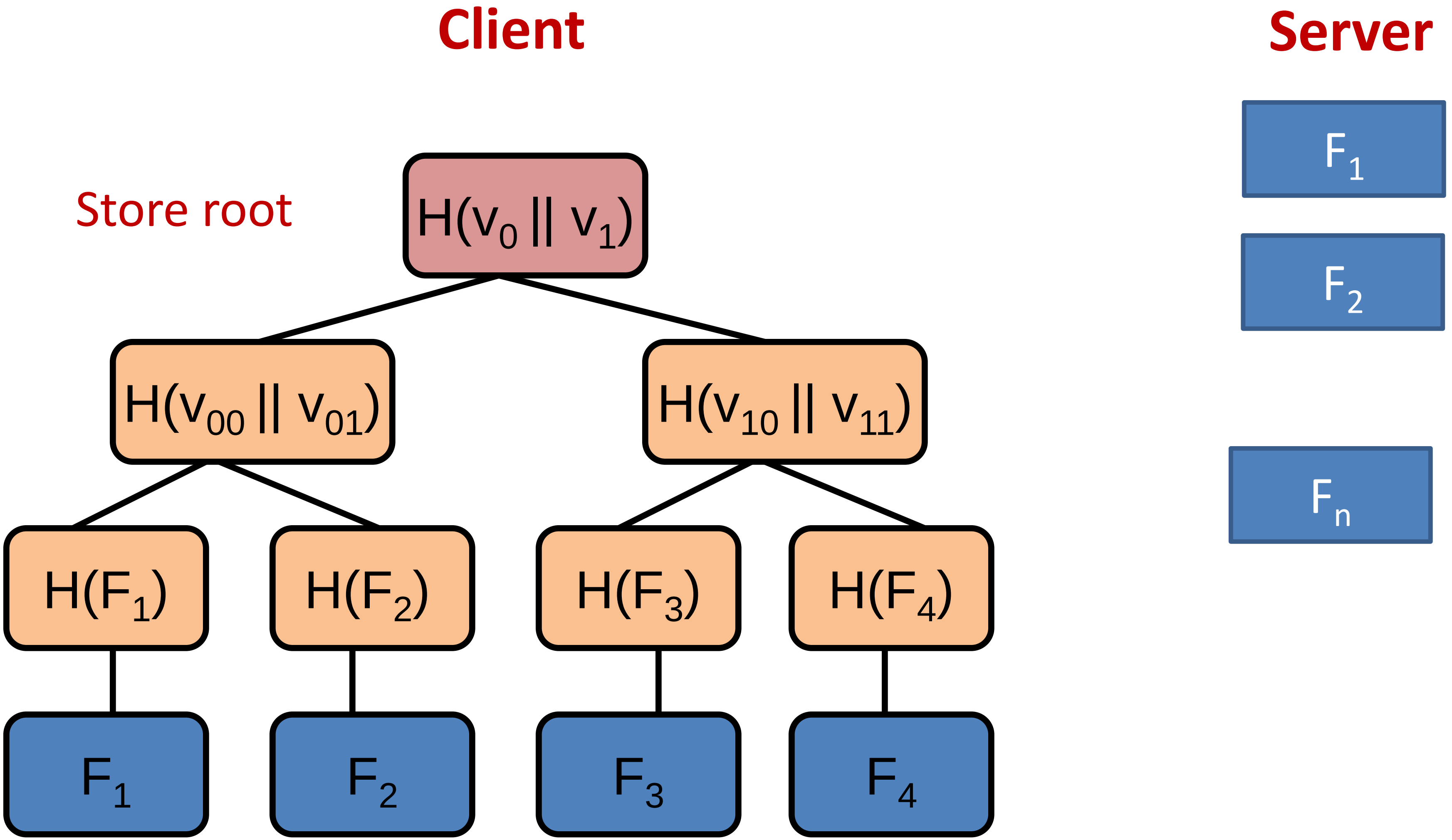
- Introduced by Ralph Merkle, 1979
  - “Classic” cryptographic construction
  - Involves combining hash functions on binary tree structure
- An efficient data structure with many practical applications
- Constant amount of storage on client
- Logarithmic update and verification cost

Merkle tree data structure

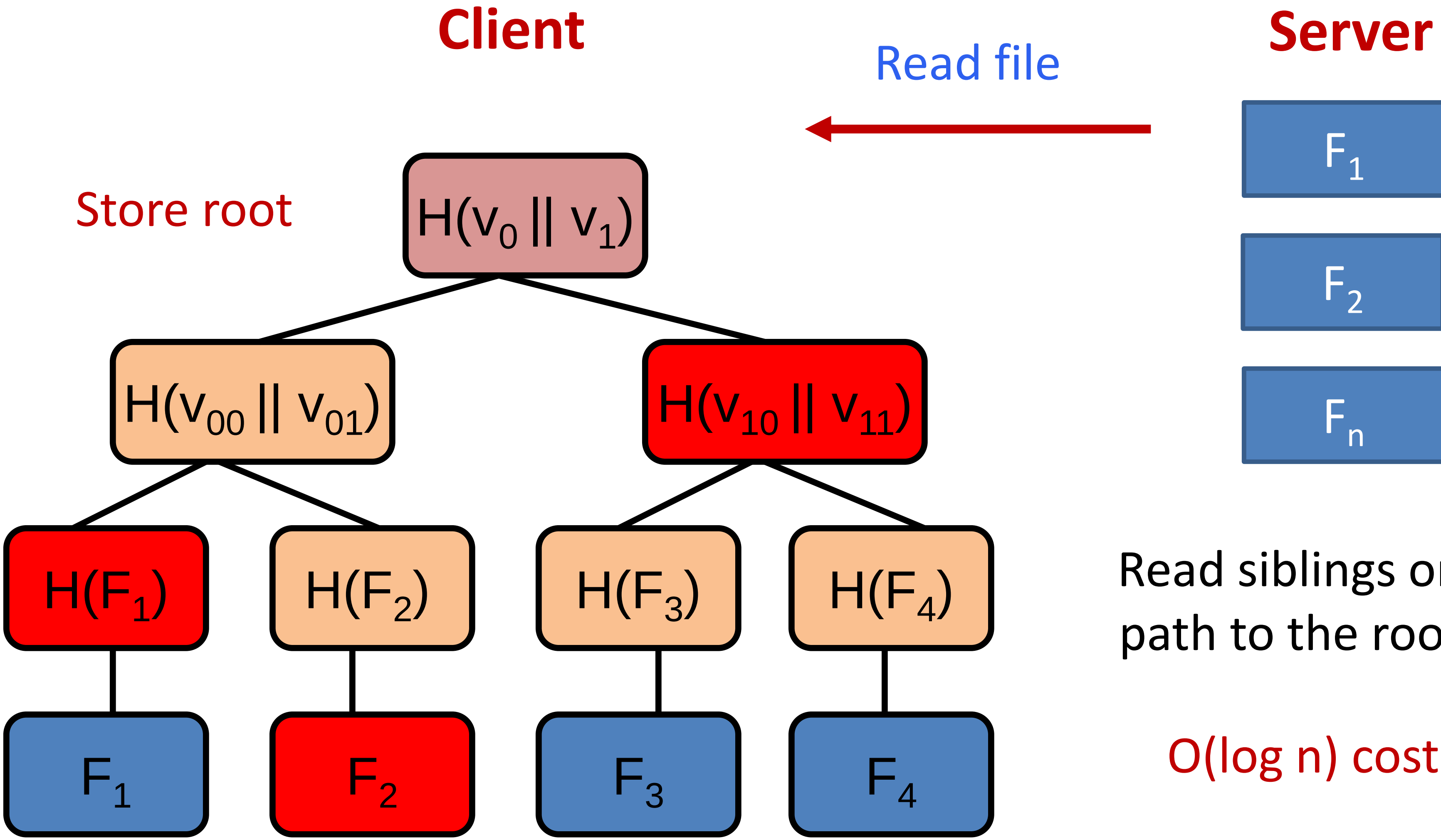
- Binary tree, nodes are assigned fixed-size values
- Files associated to each leaf



How to authenticate multiple files?



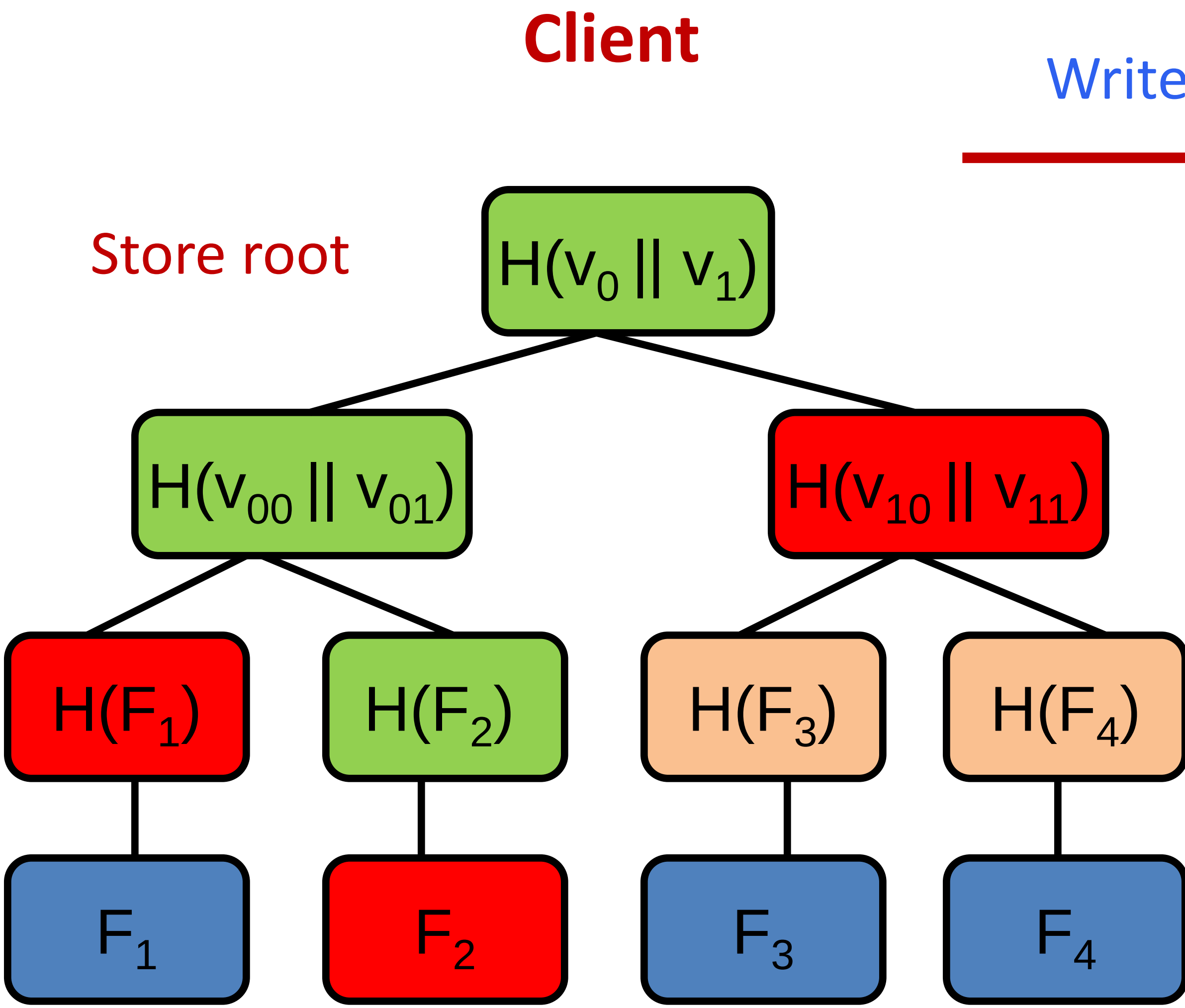
Read/authenticate file

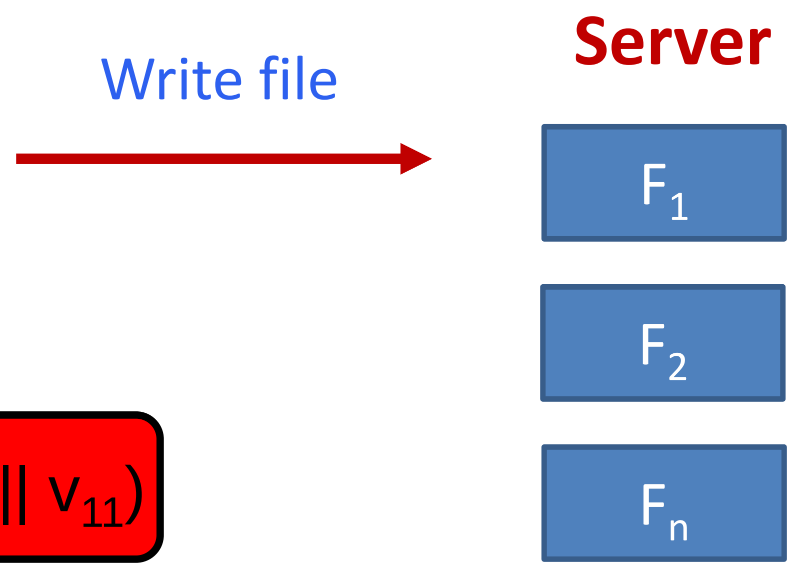


Read siblings on path to the root

O(log n) cost

Write/authenticate file





- Read siblings on path to the root
- Modify nodes on path to root

O(log n) cost

Number theory review

Prime Numbers

- An integer $p > 1$ is a *prime number* iff its only positive divisors are 1 and p
  - E.g., 3,5,7,11,13
- Otherwise, an integer that has other divisors is called *composite*
  - E.g., 4,6,8,10,25,39
- Theorem [Fundamental theorem of arithmetic]

  Any integer $a > 1$ *can be factored* in a unique way as

$$a = p_1^{a_1} p_2^{a_2} \ldots p_t^{a_t}$$

where $p_1 < p_2 < \ldots < p_t$ are primes and a_i are positive integers

- Theorem [Infinite prime numbers]

  The number of prime numbers is infinite

Notation

From here on:

- N denotes a positive integer.
- p denote a prime.

Notation: $Z_N = \{0,1, \dots N-1\}$ group of size N

Can do addition and multiplication modulo N

Modular arithmetic

Examples: let N = 12

9 + 8 = 5 in $\mathbb{Z}_{12}$

5 × 7 = 11 in $\mathbb{Z}_{12}$

5 − 7 = 10 in $\mathbb{Z}_{12}$

Arithmetic in $\mathbb{Z}_N$ works as you expect, e.g $x \cdot (y+z) = x \cdot y + x \cdot z$ in $\mathbb{Z}_N$

Greatest common divisor

Def: For integers x, y: **gcd(x, y)** is the *greatest common divisor* d such that d|x and d|y

Example: gcd(12, 18) = 6

Fact: for all integers x, y there exist a, b such that

$$\mathbf{a \cdot x + b \cdot y = gcd(x,y)}$$

Coefficients a,b can be found efficiently using the *extended Euclidean algorithm*

If gcd(x,y)=1 we say that x and y are **relatively prime**
Example: gcd(14,25) = 1

Facts on gcd

Proposition: If c|ab and gcd(a,c) = 1, then c|b

Proof: If c|ab, there exists a value u such that:

cu = ab

Since gcd(a,c) =1, there exists some constants v and w such that: av + cw = 1

Multiply by b: avb +cwb = b ⇒ cuv + cwb = b

⇒ c(uv+wb) = b ⇒ c|b

Corolary: If p is prime and p|ab, then p|a or p|b

Proof: If p prime, then p|a or gcd(p,a) = 1. Then p|a or p|b

Modular inversion

Over rationals, inverse of 2 is ½ . What about Z_N ?

Definition: The **multiplicative inverse** of x in Z_N is an element y in Z_N such that $\mathrm{x} \cdot y = 1$ in Z_N

y is denoted x^{-1}

Example: Let N be an odd integer. What is the inverse of 2 in Z_N?

$$2 \cdot \frac{N+1}{2} = \mathrm{N} + 1 = 1 \bmod \mathrm{N}$$

Acknowledgement

Some of the slides and slide contents are taken from
http://www.crypto.edu.pl/Dziembowski/teaching
and fall under the following:

We have also used slides from Prof. Dan Boneh online cryptography course at Stanford University:

http://crypto.stanford.edu/~dabo/courses/OnlineCrypto/