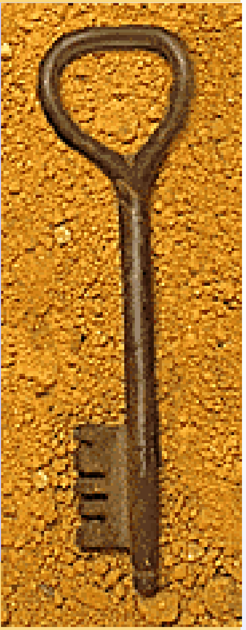
Graduate Course on **Computer Security**

Lecture 3: Public-Key Cryptography

Iliano Cervesato iliano@itd.nrl.navy.mil

ITT Industries, Inc @ NRL - Washington DC

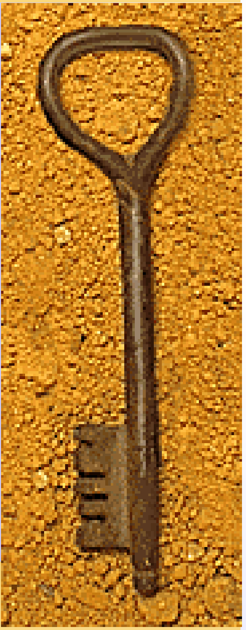
http://www.cs.stanford.edu/~iliano/



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Outline

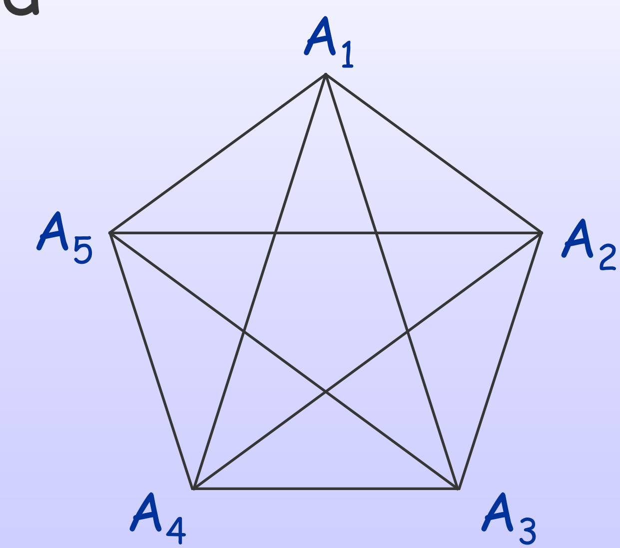
- Motivations
- Elements of number theory
- Public-key encryption
  - Diffie-Hellman key exchange
  - El Gamal encryption
  - RSA encryption
- Hash functions
  - Unkeyed
  - Keyed – MACs
- Digital signatures
- Public-key infrastructures

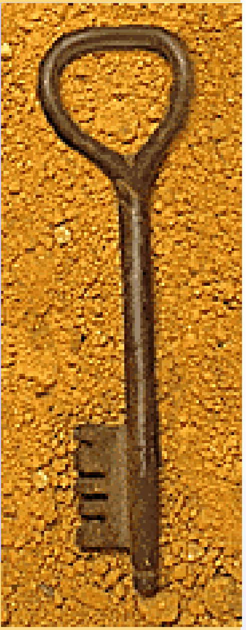


Naïve Key Management

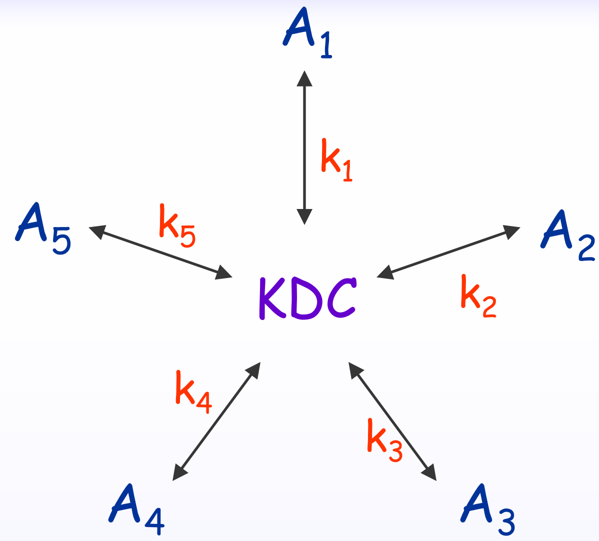
Principals $A_1, \ldots, A_n$ want to talk

- Each pair needs a key
  - n(n-1)/2 keys
- Keys must be established
  - Physical exchange
  - Secure channel
  - ...





Improved Solution



Centralized key-distribution center

- n key pairs needed
- However
  - KDC must be trusted
  - KDC is single point of failure
  - Still n direct exchanges

… if A_i wants to talk to A_j …

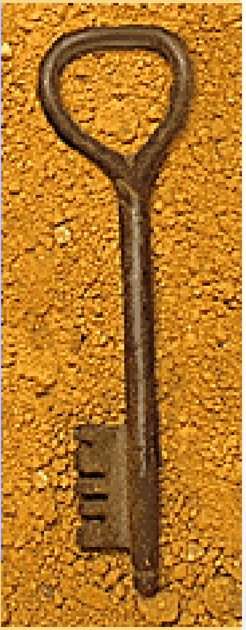
- $A_i \rightarrow$ KDC: "connect me to A_j"
- KDC generates new key k_{ij}
- KDC $\rightarrow A_i$: $E_{ki}(k_{ij})$
- KDC $\rightarrow A_j$: $E_{kj}(k_{ij}$, "A_i wants to talk")

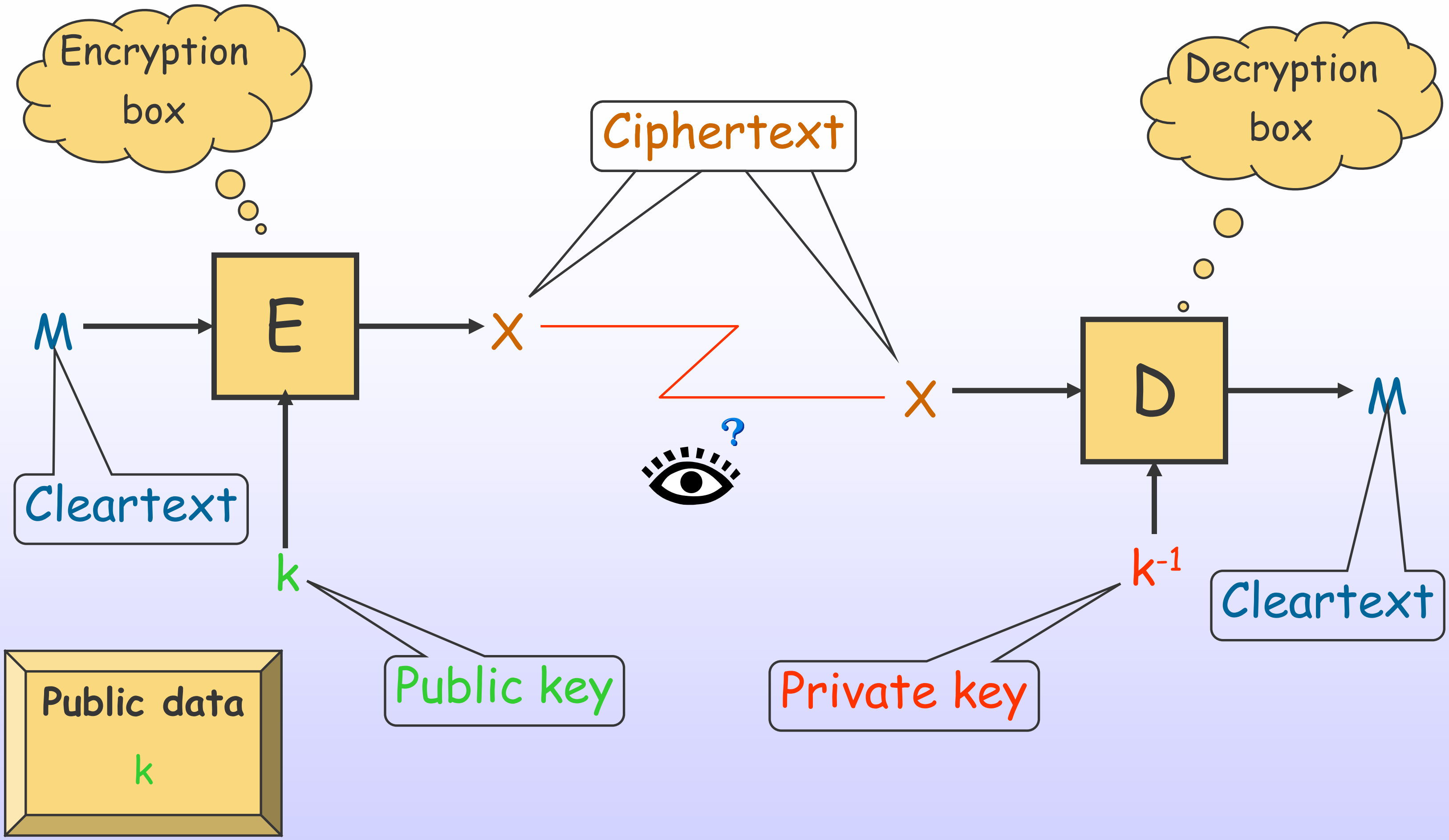
Still naïve

- No authentication

Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

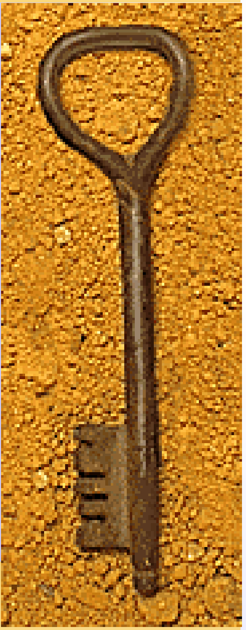
Asymmetric Ciphers

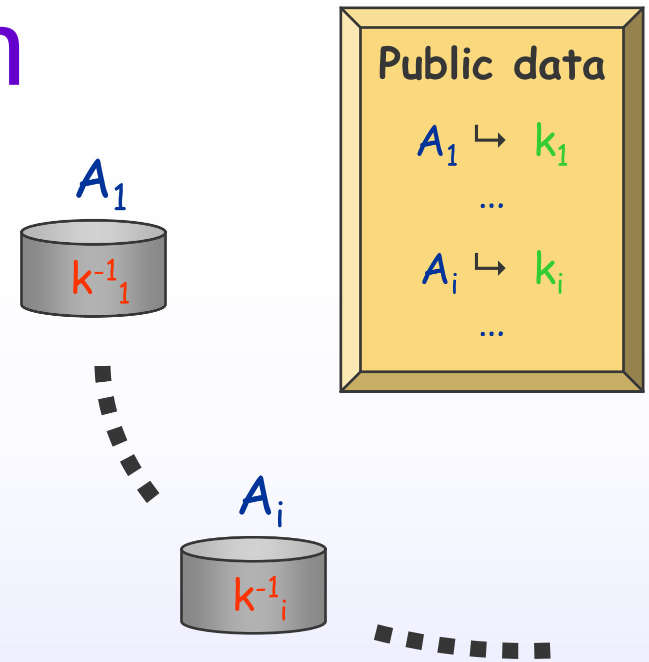




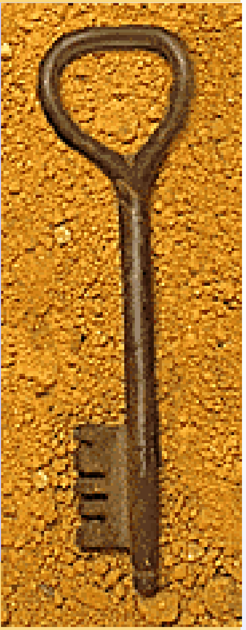
$$D_{k^{-1}}(E_k(m)) = m$$

Public-Key Solution





- Pair (k_i, k_i^{-1}) for each A_i
- k_i's are published
  - Phonebook
- Simple setup
  - A_i generates (k_i, k_i^{-1})
  - A_i publishes k_i
  - … details later



Number Theory – Basics

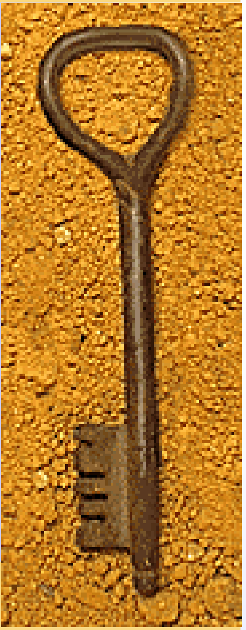
- $Z = \{\ldots, -1, 0, 1, \ldots\}$ is a ring
- $a|b$ if $\exists c.\ ac = b$
  - E.g. $3|6$
- $\gcd(a, b)$ = largest $d \in Z$ s.t. $d|a$ and $d|b$
  - E.g. $\gcd(18,15) = 3$
- $p>1$ *prime* if 1 and p are its only divisors
  - E.g. 3, 5, 7, …
- p and q are *relatively prime* if $\gcd(p,q) = 1$
  - E.g. 4 and 5 are relative primes

Euclid's algorithm

Given $a > b$

- $r_0 = b$, $r_1 = a$
- $r_{i-2} = q_i r_{i-1} + r_i$
- When $r_{n+1} = 0$, set $\gcd(a,b) = r_n$
  - $\exists u,v.\ \gcd(a,b) = ua + vb$

Arithmetic Modulo a Prime

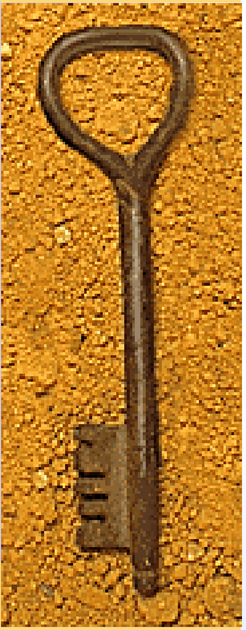


Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

- p prime number
  - For us, typically 1024 bits (~ 300 digits)
- $Z_p = \{0, 1, \ldots, p-1\}$
  - Addition and multiplication are modulo p
  - Exponentiation is iterated multiplication
  - x is the inverse of $y \neq 0$ if $xy = 1 \bmod p$
- All non-null elements of Z_p are invertible
  - $x^{-1} = x^{p-2} \bmod p$
  - We can solve linear equations in Z^*_p
    - If $ax = b \bmod p$, then $x = ba^{p-2} \bmod p$
- $Z^*_p = \{1, \ldots, p-1\}$
  - Contains all invertible elements of Z_p

Fermat's little theorem

If $a \neq 0$, then $a^{p-1} = 1 \bmod p$



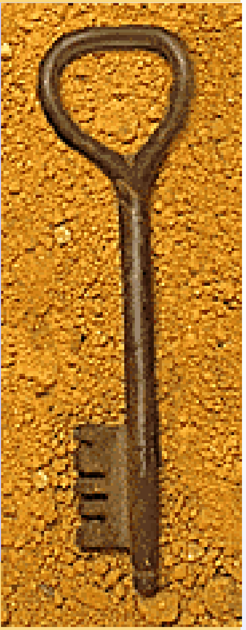
Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Computing in Z_p

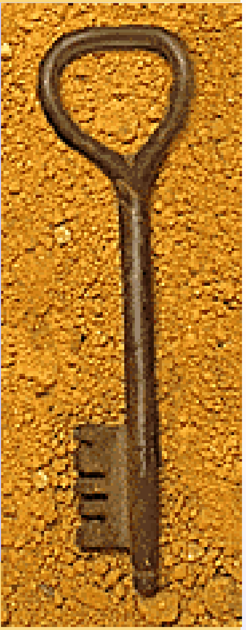
- Let n be the length of p
  - Usually around 1024 bits
- Addition in Z_p done in $O(n)$
- Multiplication is $O(n^2)$
  - Clever (and practical) algorithms achieve $O(n^{1.7})$
  - Same for inverse
- $x^r \bmod p$ computed in $O((\log r)\, n^2)$
  - Repeated squares
    - E.g.: $g^{23} = g^{10111} = g \,.\, g^2 \,.\, g^4 \,.\, g^{16}$ (7 multiplications)
      ↑ ↑ ↑ ↑ ↑ ↑↑
  - Addition chains
    - Saves 20% in average (but shortest chain is NP-complete)
    - $g, g^2, g^3, g^5, g^{10}, g^{20}, g^{23}$ (6 multiplications)
      ↑ ↑ ↑ ↑ ↑ ↑

Complexity in Z_p

- Easy problems
  - Generating p
  - Addition, multiplication, exponentiation
  - Inversion, solving linear equations
- Problems believed to be hard
  - DL: Discrete logarithm
    - Given g and $x \in Z_p$, find r s.t. $x = g^r \bmod p$
  - DH: Diffie-Hellman
    - Given $g, g^r, g^s \in Z_p$, find $g^{rs} \bmod p$
  - Note
    - DL implies DH
    - Unknown if DH implies DL
    - Best known attack on DL requires space and $O(2^{\sqrt{n}})$ time



Diffie-Hellman Key Exchange



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Public data

p, g

A

- Choose random a, $1 \leq a \leq p-1$
- send g^a
- Receive g^b
- $(g^b)^a = g^{ab}$
- $k = f(g^{ab})$

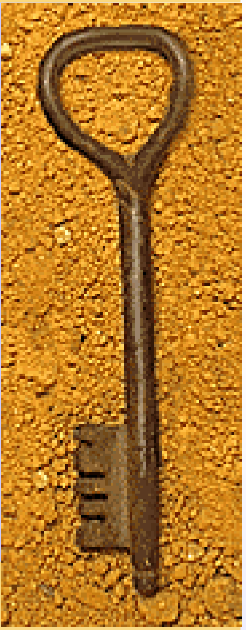
A → B: g^a

B → A: g^b

B

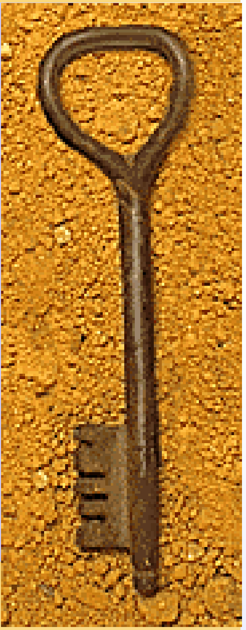
- Receive g^a
- Choose random b, $1 \leq b \leq p-1$
- Send g^b
- $(g^a)^b = g^{ab}$
- $k = f(g^{ab})$

Diffie-Hellman Key Exchange [2]



- Allows 2 principals to produce a shared secret
  - Without secure channel or physical exchange
  - Without a key distribution center
  - f is typically a hash function
    - Agreed upon in advance
- However, no authentication
  - Can be fixed with some infrastructure
- Security relies on hardness of DH

El Gamal Encryption Scheme



A wants to send secret $m \in Z_{p_B}$ to B

Public data

$A_1 \hookrightarrow p_1, g_1, g_1^{\alpha_1}$

…

$A_i \hookrightarrow p_i, g_i, g_i^{\alpha_i}$

…

A (α_A)

- Choose random a
- Send g_B^a, $g_B^{\alpha_B a}$ m mod p_B

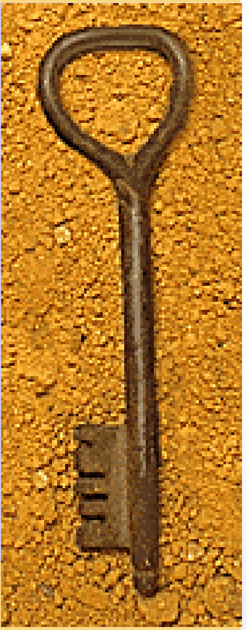
g_B^a, $g_B^{\alpha_B a}$ m mod p_B →

B (α_B)

- Receive g_B^a, $g_B^{\alpha_B a}$ m mod p_B
- $(g_B^a)^{\alpha_B} = g_B^{\alpha_B a}$
- Compute $g_B^{-\alpha_B a}$
- $g_B^{-\alpha_B a} g_B^{\alpha_B a}$ m mod p_B = m

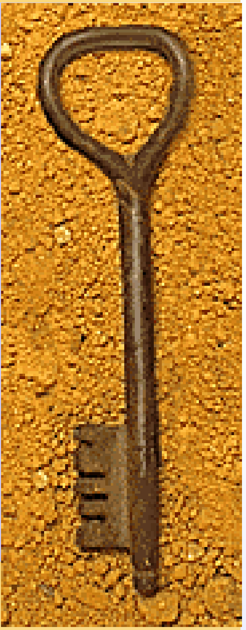
- Security rests on hardness of DL
- Criticisms
  - Transmitted message double of m
  - Public data has to be managed
  - Very slow (~10Kb/sec vs. 250Kb/s of DES)

Arithmetic Modulo a Composite



- n natural number
  - For us, typically 1024 bits or ~ 300 digits
  - Typically n = pq, with p and q primes
- $Z_n = \{0, 1, \ldots, n-1\}$
  - x is inverse of $y \neq 0$ if $xy = 1 \bmod n$
  - x has inverse iff gcd(x,n) = 1
    - $ux + vn = 1$ by Euclid's algorithm so $x^{-1} = u$
    - Works also in Z_p where more efficient than $x^{-1} = x^{p-2}$
  - We can solve linear equations in Z_n
- $Z^*_n = \{x : gcd(x,n) = 1\}$
  - Contains all invertible elements of Z_n

Euler's Totient Function

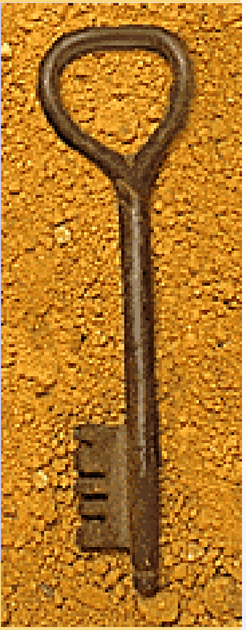


- $\phi(n)$ is the size of Z^*_n
  - If $n = \Pi_i p_i^{e_i}$,
    then $\phi(n) = \Pi_i p_i^{e_i - 1}(p_i - 1)$
  - If $n=pq$,
    then $\phi(n) = (p-1)(q-1) = n - p - q - 1$

Euler's theorem

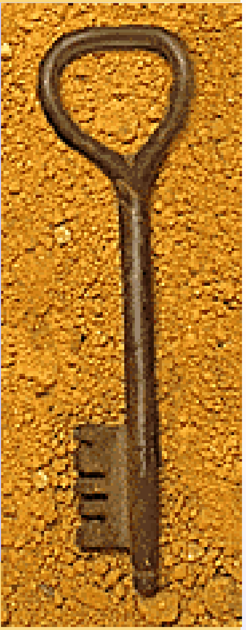
If $a \in Z^*_n$, then $a^{\phi(n)} = 1 \bmod n$

Computing in Z_n



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

- Easy problems
  - Generating p
  - Addition, multiplication, exponentiation
  - Inversion, solving linear equations
- Hard problems
  - Factoring
    - Given n, find p,q s.t. n = pq



RSA [Rivest,Shamir,Adelman '76]

$n_i = p_i q_i$

$e_i d_i = 1 \bmod \phi(n_i)$

A wants to send secret $m \in Z_{nB}$ to B

Public data

$A_1 \hookrightarrow n_1, e_1$

…

$A_i \hookrightarrow n_i, e_i$

…

A — p_A, q_A, d_A

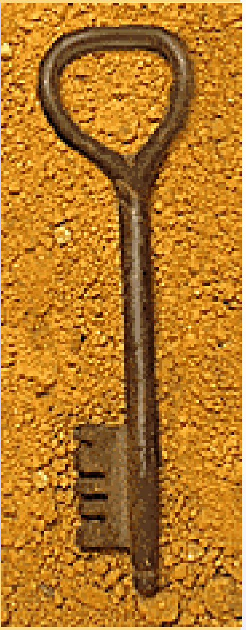
- Send $m^{e_B} \bmod n_B$

$m^{e_B} \bmod n_B$ →

B — p_B, q_B, d_B

- Receive $m^{e_B} \bmod n_B$
- $(m^{e_B})^{d_B} \bmod n_B$
  $= m^{e_B d_B} \bmod n_B$
  $= m^{k\phi(n_B)+1} \bmod n_B$
  $= (m^{\phi(n_B)})^k\, m \bmod n_B$
  $= (1)^k\, m \bmod n_B$
  $= m \bmod n_B$

- Security of RSA rests on
  - Hard to factorize n = pq
    - Hard to compute $\phi(n)$ from n
- Factoring implies RSA
- Unknown if RSA implies factoring



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Attacks on RSA

- Small d for fast decryption
  - But easy to crack if $d < (n^{1/4})/3$ [Wiener]
    - d should be at least 10^{80}
- Small e for fast encryption
  - If m sent to more than e recipients, then m easily extracted
  - Popular $e = 2^{16} + 1$
    - Same message should not be sent more than $2^{16} + 1$ times
    - Modify message (still dangerous)
- Timing attacks
  - Time to compute $m^d \bmod n$ for many m can reveal d
- Homomorphic properties of RSA
  - If $c_i = m_i^e \bmod n$ (i=1,2), then $c_1 c_2 = (m_1 m_2)^e \bmod n$
    - Easy chosen plaintext attack
  - Eliminated in standards based on RSA

One-Way Functions

$f : \{0,1\}^{n'} \rightarrow \{0,1\}^{n}$ is a *1-way function* if

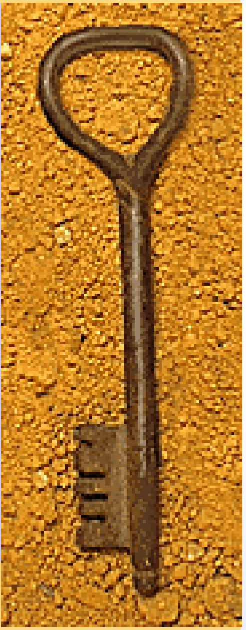
- There is an efficient algorithm that given x outputs f(x)
  - polynomial
- Given y, there is no known efficient algorithm to find x s.t. y = f(x) for non-negligible fraction of y's

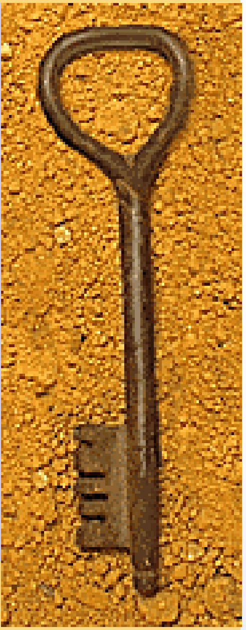
- Examples
  - $f(x) = DES_x(m)$ for a given m
  - $f(x) = g^x \bmod p$ for given g and p as in DH

$f_p : \{0,1\}^{n'} \rightarrow \{0,1\}^{n}$ is a *1-way function with trapdoor*

- $f_p(x)$ is 1-way if p is unknown
- Given p, $f_p(x)$ has efficient algorithm

- Examples
  - $f_d(x) = x^e \bmod n$ for given e and n as in RSA
  - $f_k(x) = DES_k(x)$





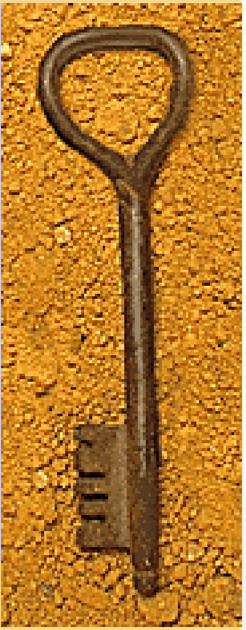
Cryptographic Hashing

$f : \{0,1\}^{n'} \rightarrow \{0,1\}^n$ is a *1-way hash function* if

- n is short
- n′ may be unbounded

Two families

- Non-keyed
  - $h : \{0,1\}^* \rightarrow \{0,1\}^n$ (e.g. $n = 160$)
  - h(m) is the *message digest* of m
  - Used for password protection, digital signatures, …
- Keyed
  - $h_k : \{0,1\}^* \rightarrow \{0,1\}^n$ (e.g. $n = 96$)
  - Used for message integrity



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Preimage Resistance

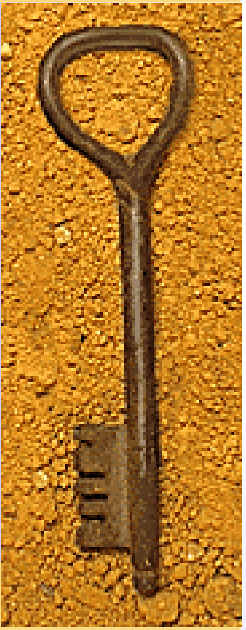
$h : \{0,1\}^* \rightarrow \{0,1\}^n$ is *PR* if

- Given random y
  - It is hard to find m s.t. h(m) = y

Applications:

- Protect password files
  - `/etc/passwd` in Unix

$username_1$	$h(pwd_1)$
$username_2$	$h(pwd_2)$
...	...



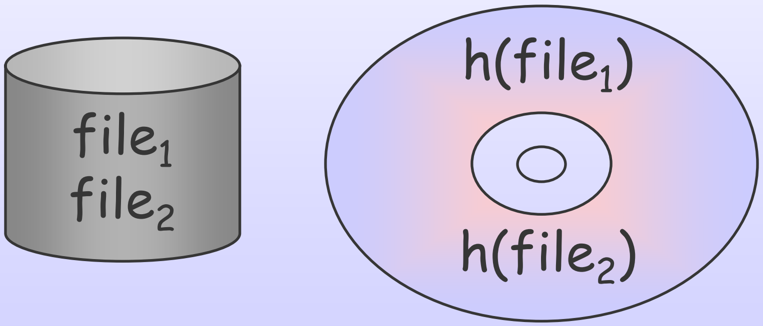
Second Preimage Resistance

$h : \{0,1\}^* \rightarrow \{0,1\}^n$ is *2PR* if

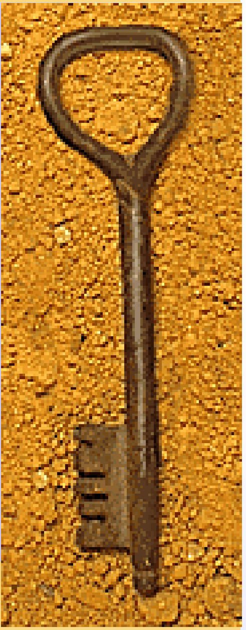
- Given random m
  - It is hard to find m′ s.t. h(m) = h(m′)

Applications:

- Virus protection
  - E.g. Tripwire
  - file and h(files) must be kept separate



- 2PR implies PR



Collision Resistance

$h : \{0,1\}^* \rightarrow \{0,1\}^n$ is *CR* if

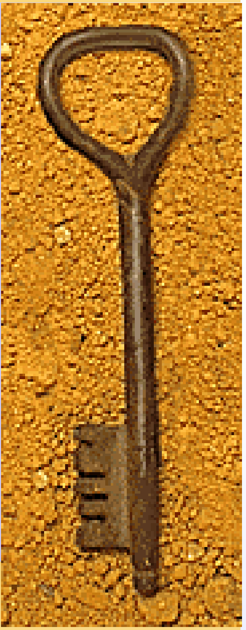
- It is hard to find m and m′ s.t. h(m) = h(m′)

Applications:

- Digital signatures
  - $Sig_k(h(m))$
  - Assume attacker knows m and m′ s.t. h(m) = h(m′)
    - Ask principal to sign m
    - Has automatically signature on h(m′)

- CR implies 2PR (implies PR)
  - Easier to construct CR than 2PR
  - From now on, we focus on CR

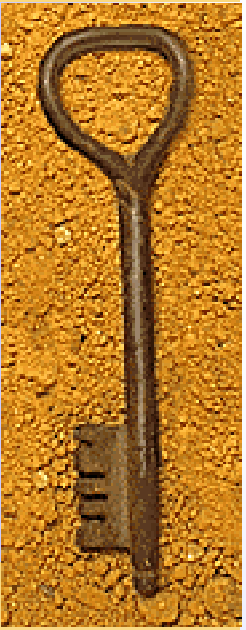
Birthday Paradox



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

There is a 0.5 probability that 2 people have the same birthday in a room of 25

- Given $r_1, \ldots r_n \in [0, 1, \ldots, B]$ independent integers
  - If $n \geq 1.2\sqrt{B}$, then $\text{Prob}[\exists\ i \neq j : r_i = r_j] > \frac{1}{2}$
- For message digest 64 bits long
  - Collision can be found with around 2^{32} tries
  - Typical digest size is 160 bits (SHA-1)
    - Collision time is 2^{80} tries

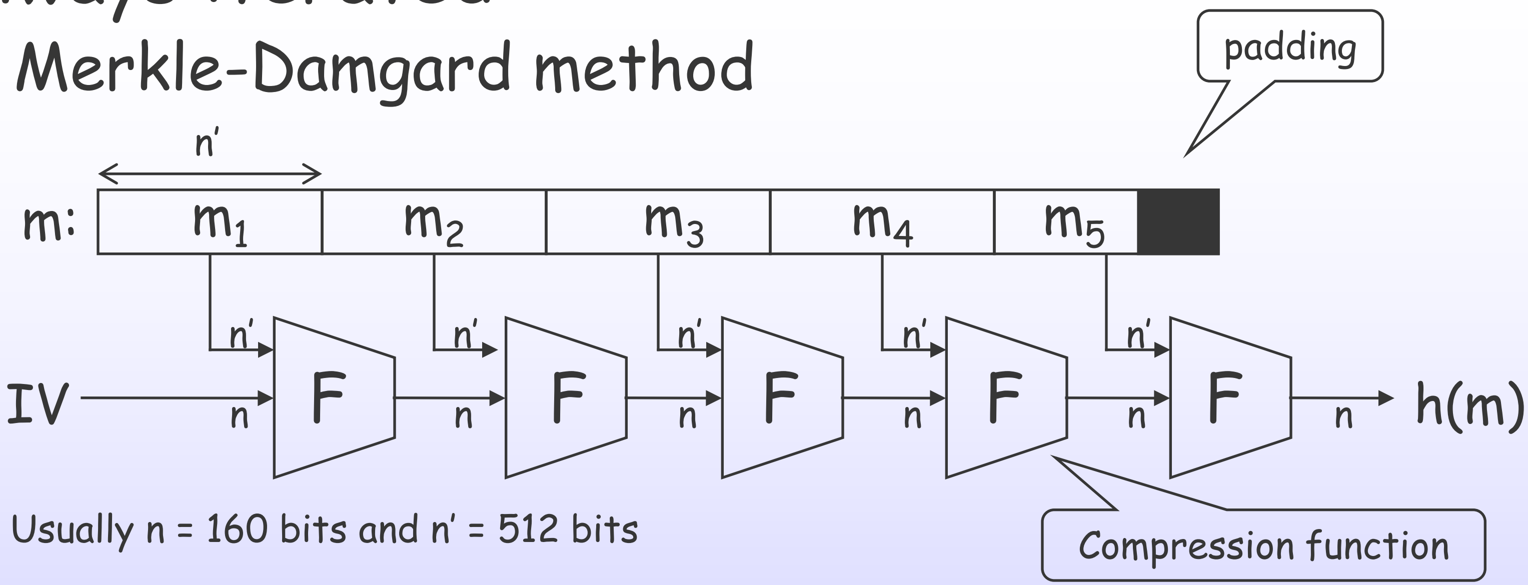


Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Constructions

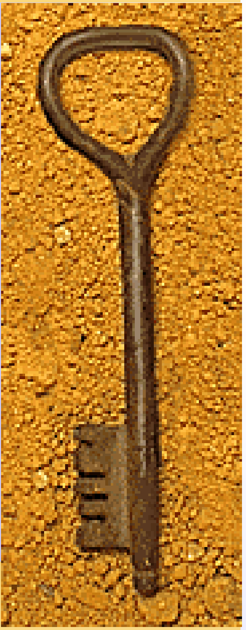
Always iterated

- Merkle-Damgard method



- If F (compression function) is CR, then Merkle-Damgard hash is CR
  - Enough to construct a CR compression function
    - Based on block ciphers (typically slow)
    - Customized design (faster)

Actual Compression Functions



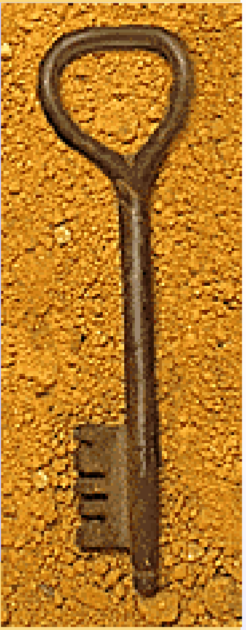
Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

- Based on block ciphers (e.g. DES)
  - Given block cipher $E_k(m)$
  - $F(m,h_i) = E_{m \oplus k_{i-1}}(m)$
  - If $E^k(m)$ is ideal cipher, finding collisions takes $2^{n/2}$ tries
    - Best possible, but black-box security
- Customized compression functions

Name	n	Speed	Comment
MD4	128	?	Proprietary (RSA labs); broken in time 2^{26}
MD5	128	28.5 Mb/s	Collision for compression function
SHA-1	160	15.25 Mb/sec	NIST
RIPE-MD	160	12.6 Mb/s	RIPE

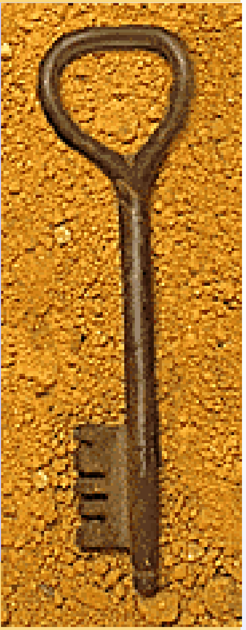
On 200MHz Pentium

Keyed Hash Functions



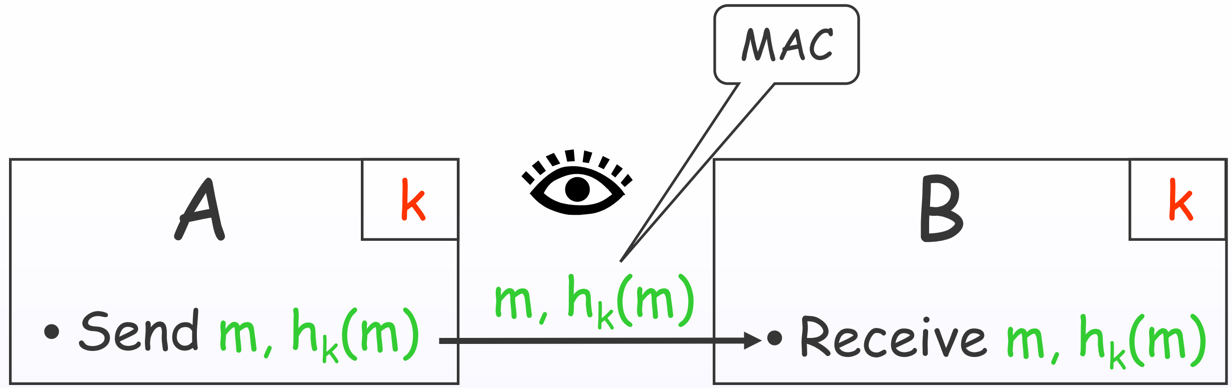
$$h_k : \{0,1\}^* \rightarrow \{0,1\}^n$$

- k needed to evaluate function
- Main application:
  - Message authentication codes (MAC)
    - Guarantees message integrity
- $H_k(m)$ is a cryptographic checksum
  - Ensures that m has not been tampered



Example

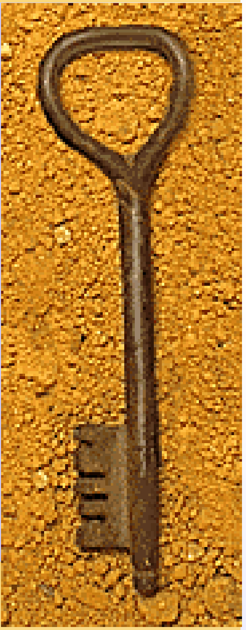
- Network



  - Adversary can't build MAC for $m' \neq m$
  - Note: MAC used for integrity, not secrecy
  - Digital signature work, but are too slow

- File system

  - MAC verified when file is accessed
  - pwd needed to modify file



Constructing MACs

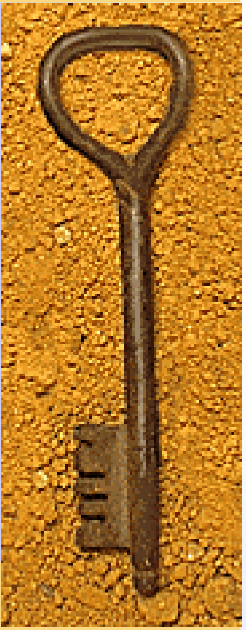
2 methods

- Cryptographic MACs
  - CBC-MAC
    - Based on block ciphers
  - HMAC
    - Based on non-keyed hash functions
- Information-theoretic MACs
  - Based on universal hashing

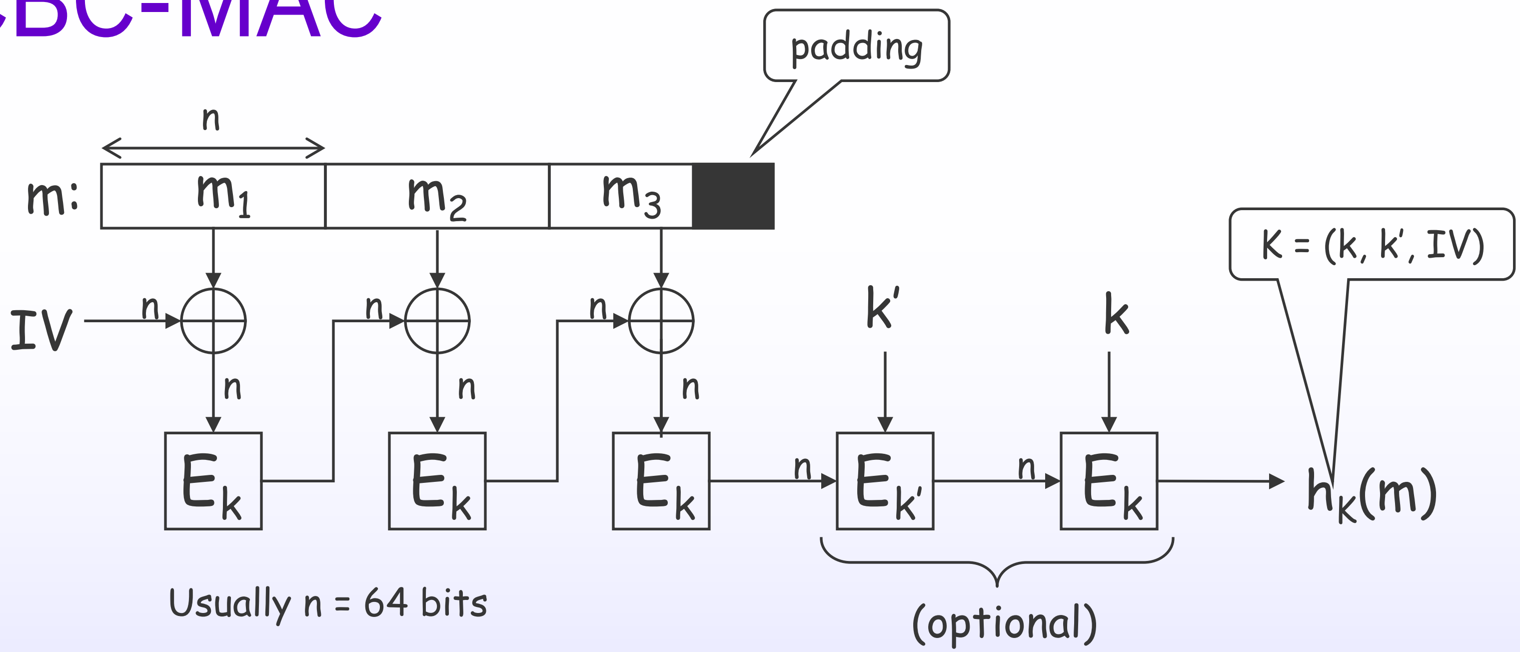
Performance

Name	n	Speed
3DES	64	1.6Mb/sec
IDEA	64	3Mb/sec
MD5	128	28.5 Mb/s
SHA-1	160	15.25 Mb/sec

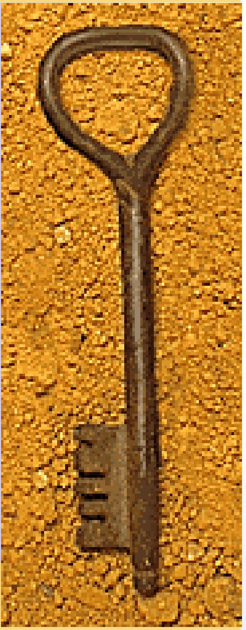
On 200MHz Pentium



CBC-MAC



- Most commonly used in banking industry
- If E is a MAC, then CBC-E is also a MAC
- Note: no birthday attack
  - MACS can be shorter then message digests



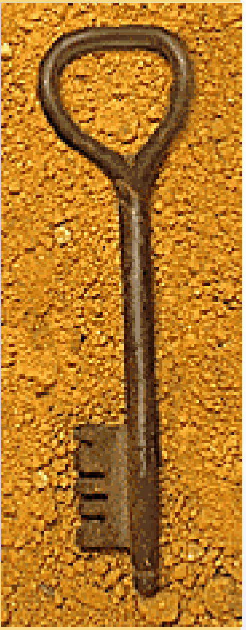
Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Hash-Based MACs

h non-keyed hash function

- Attempt: $MAC_k(m) = h(k\ m)$
  - Extension attack with Merkle-Damgard method:
    - $MAC_k(m\ m') = h(MAC_k(m)\ m')$
- Attempt: $MAC_k(m) = h(m\ k)$
  - Birthday paradox attack
- Envelope method
  - $MAC_{k,k'}(m) = h(k\ m\ k')$
- Prefered method: HMAC
  - $HMAC_k(m) = h(k\ pad_1\ h(k\ pad_2\ m))$
  - If compression function in h is a MAC and h is CR, then HMAC is a MAC
  - IPSec and SSL use 96 bit HMAC

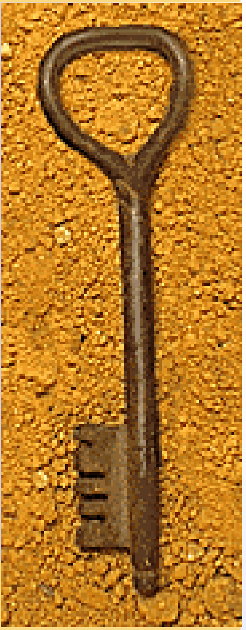
Hash-based MAC



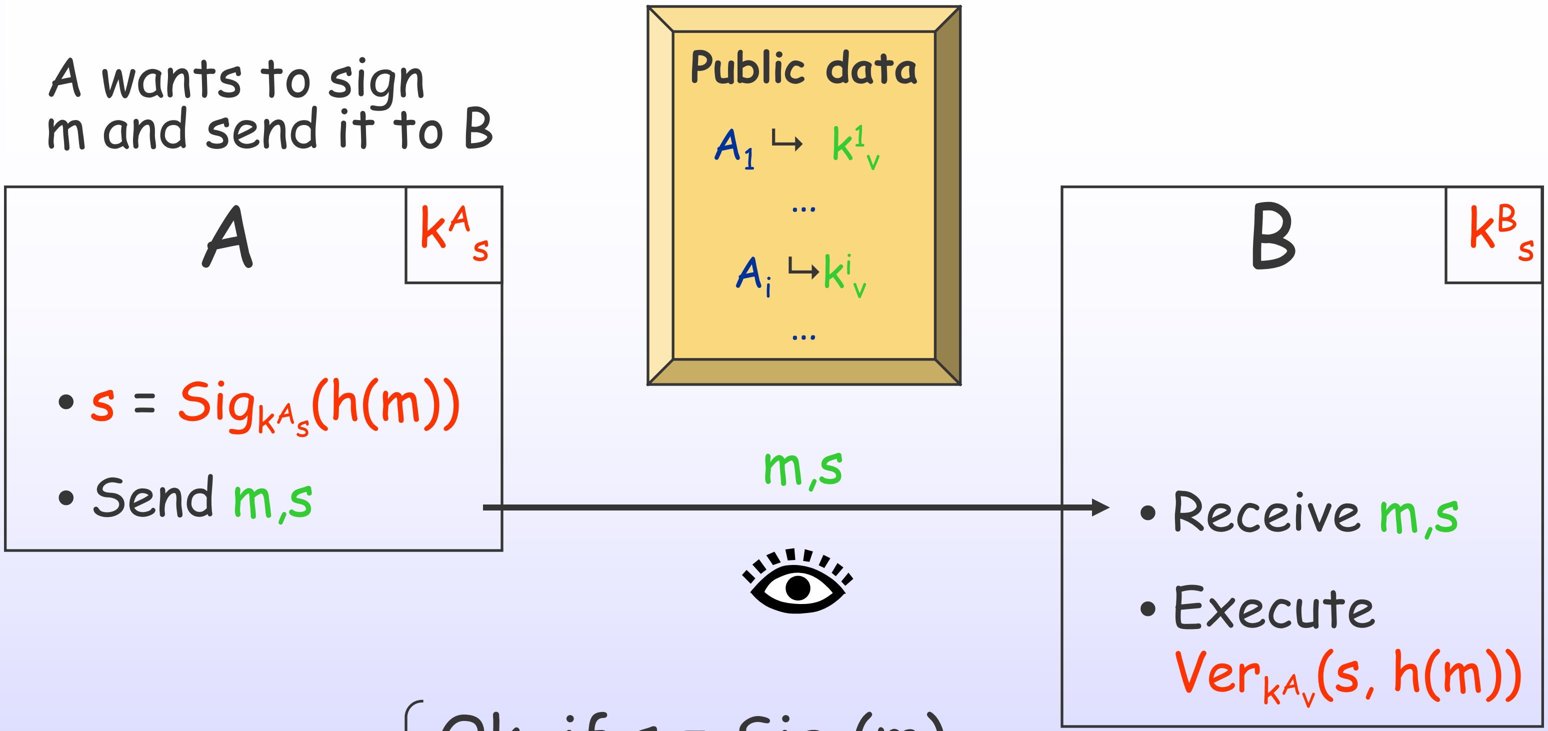
Digital Signatures

- Paper signature guarantees non-repudiation for
  - Identity
  - Contract signing
- Digital signature
  - binds a secret k to a document m
    - s = f(m,k)
  - s can be generated only knowing k
  - s can be verified by anyone knowing m
- Should guaranty
  - Non-repudiation
  - Non-malleability
    - Signature cannot be cut and pasted to other documents
  - Non-forgeability

Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI



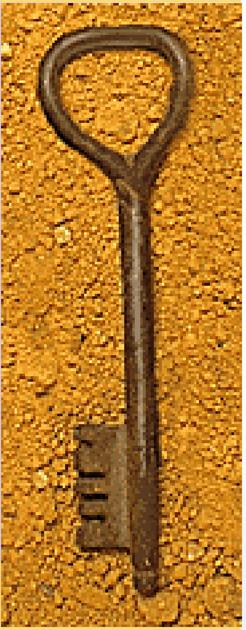
Signature Process



$$Ver_{k^{-1}}(s,m) = \begin{cases} \text{Ok} & \text{if } s = Sig_k(m) \\ \text{No} & \text{otherwise} \end{cases}$$

- h makes signature short

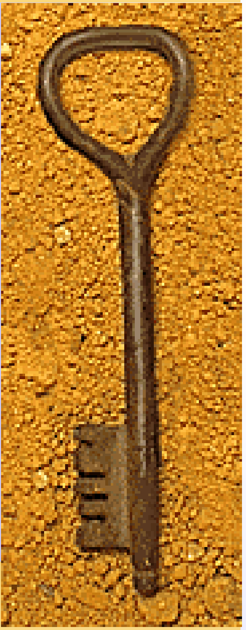
Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Attacks on Digital Signatures

- Signature break
  - Adversary can recover k_s from k_v and intercepted messages
- Selective forgery
  - Adversary can forge signature s for message m of his choice
- Existential forgery
  - Adversary can forge signature s for arbitrary message m

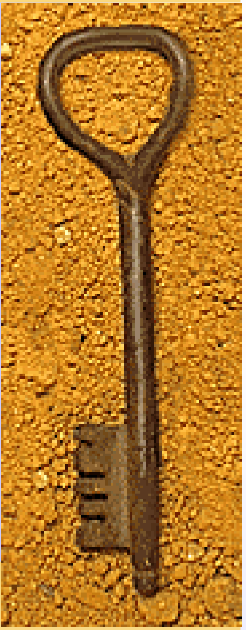


Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Constructions

Signature schemes based on

- RSA
  - E.g.: PKCS#1, Fiat-Shamir, …
  - Easy to verify but hard to generate
    - Ok for certificates
  - Relatively long (1024 bit)
- DL
  - El Gamal , DSS, …
  - Hard to verify, but easy to generate
    - Ok for smart cards
  - Short (320 bit)
- General 1-way functions
  - Lamport, Merkle, …
  - Impractical



Naïve RSA Signature

$n_i = p_i q_i$
$e_i d_i = 1 \bmod \phi(n_i)$

A wants to send signed $m \in Z_{n_A}$ to B

Public data

$A_1 \hookrightarrow n_1, e_1$
…
$A_i \hookrightarrow n_i, e_i$
…

A — p_A, q_A, d_A

- Send $m^{d_A} \bmod n_A$

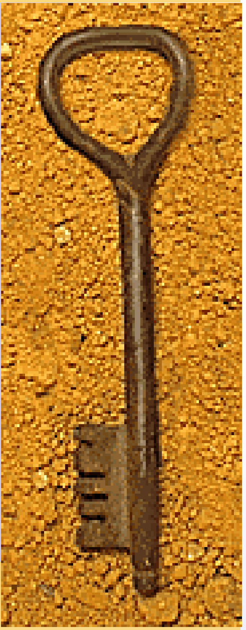
$m^{d_A} \bmod n_A$ →

B — p_B, q_B, d_B

- Receive $m^{d_A} \bmod n_A$
- $(m^{d_A})^{e_A} \bmod n_A$
  $= m^{k\phi(n_A)+1} \bmod n_A$
  $= m \bmod n_A$

- Signature = RSA decryption
  - Achieves confidentiality as well
- Verification = RSA encryption

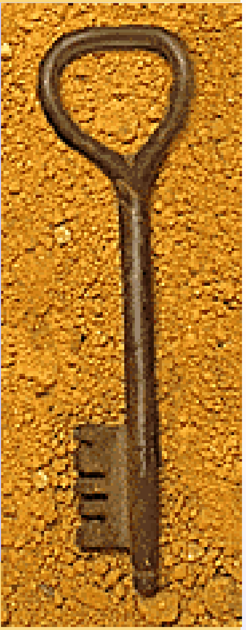
Attacks on Naïve RSA Signature



- Existential forgery
  - $Ver_d(s^e, s)$ = Ok for any s
- Blinding attack

  Adversary wants signature of A on m
  - Pick $r \in Z_{nA}$
  - Get A to sign $m' = mr^e \bmod n_A$
  - A returns $s' = (mr^e)^d \bmod n_A$
  - Deduce then $s = s'/r = m^d \bmod n_A$
  - Then (m, s) is a valid signature pair

RSA Signatures – PKCS#1



$n_i = p_i q_i$
$e_i d_i = 1 \bmod \phi(n_i)$

A wants to send signed $m \in Z_{n_A}$ to B

Public data

$A_1 \hookrightarrow n_1, e_1$
…
$A_i \hookrightarrow n_i, e_i$
…

A — p_A, q_A, d_A

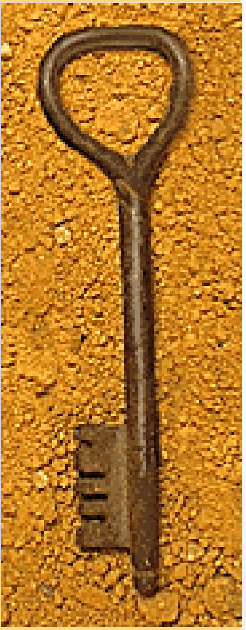
- Compute $s = (PD\ h(m))^{d_A} \bmod n_A$
- Send m,s

m,s →

B — p_B, q_B, d_B

- Receive m,s
- Check if $(m^{d_A})^{e_A} \bmod n_A = (PD\ h(m))^{d_A} \bmod n_A$

- PD = 00 01 11 11 … 11 00 (864 bit)
- h(m) is 160 bit
- Security is unproved
  - ISO standards use other PD's



El Gamal Signature

A wants to send secret $m \in Z_{p_B}$ to B

Public data

$A_1 \hookrightarrow p_1, g_1, g_1^{\alpha_1}$

...

$A_i \hookrightarrow p_i, g_i, g_i^{\alpha_i}$

...

A (α_A)

- Choose random r
- Compute
  - $k = g^r \bmod p_A$
  - $r^{-1} \bmod (p_A - 1)$
  - $s = r^{-1}(h(m) - k\alpha) \bmod (p_A - 1)$
- Send m, k, s

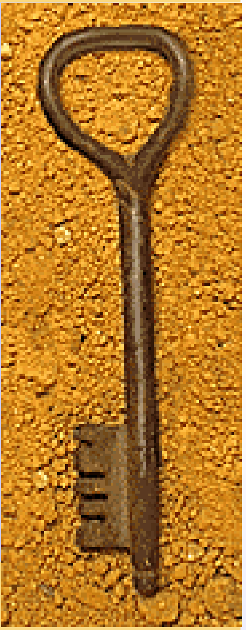
$\longrightarrow$ m, k, s $\longrightarrow$

B (α_B)

- Receive m, k, s
- Check
  $1 \le k \le p_A - 1$
  $g^k k^s = g^{h(m)} \bmod p_A$

- Why does it work?
  - Exercise

Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI



DSS – Digital Signature Standard

$q_i \mid p_i - 1$
$g_i^{q_i} = 1 \bmod p_i$
$y_i = g_i^{\alpha_i} \bmod p_i$

Public data

$A_1 \hookrightarrow p_1, p_1, g_1, y_1$
…
$A_i \hookrightarrow p_i, p_i, g_i, y_i$
…

A wants to send signed $m \in Z_{n_A}$ to B

A (α_A)

- Pick random $r \in Z^*_{q_A}$
- Compute
  - $k = (g_A^r \bmod p_A) \bmod q_A$
  - $s = r^{-1}(h(m) + k\alpha_A) \bmod q_A$
- Send m,k,s

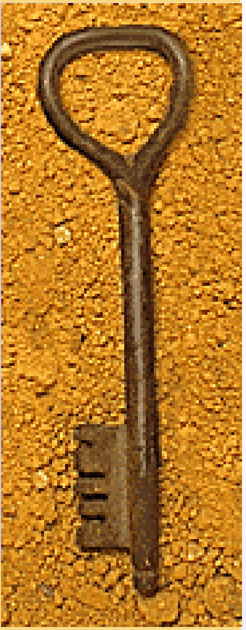
m,k,s →

B (α_B)

- Receive m,k,s
- Check
  $1 \le k,s < p_A$
  $k = g_A^{s^{-1}h(m)} (y_A^{s^{-1}w} \bmod p_A) \bmod q_A$

- p is 1042 bits
- q is 160 bits
- Signature k,s is only 360 bits
- Fast verification methods exist

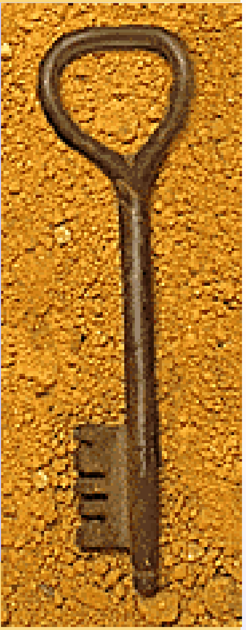
Lamport Signatures



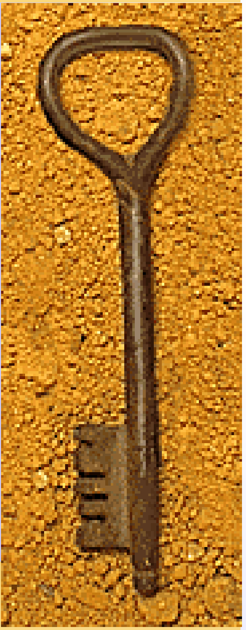
Given CR hash function h

- Key generation
  - Pick random $x_i^{(j)} \in \{0,1\}^n$, for i=1..n, j=0,1
  - Public key: $v_i^{(j)} = h(x_i^{(j)})$, for i=1..n, j=0,1
  - Private key: $x_i^{(j)}$, for i=1..n, j=0,1
- Signature of $m = m_1, \ldots, m_n \in \{0,1\}^n$
  - $s = (x_1^{(m_1)}, \ldots, x_n^{(m_n)})$
- Verification
  - $h(s_i) = x_i^{(m_i)}$, for i=1..n
- Comments
  - Can be used only once
  - Very fast
  - Lots of public data

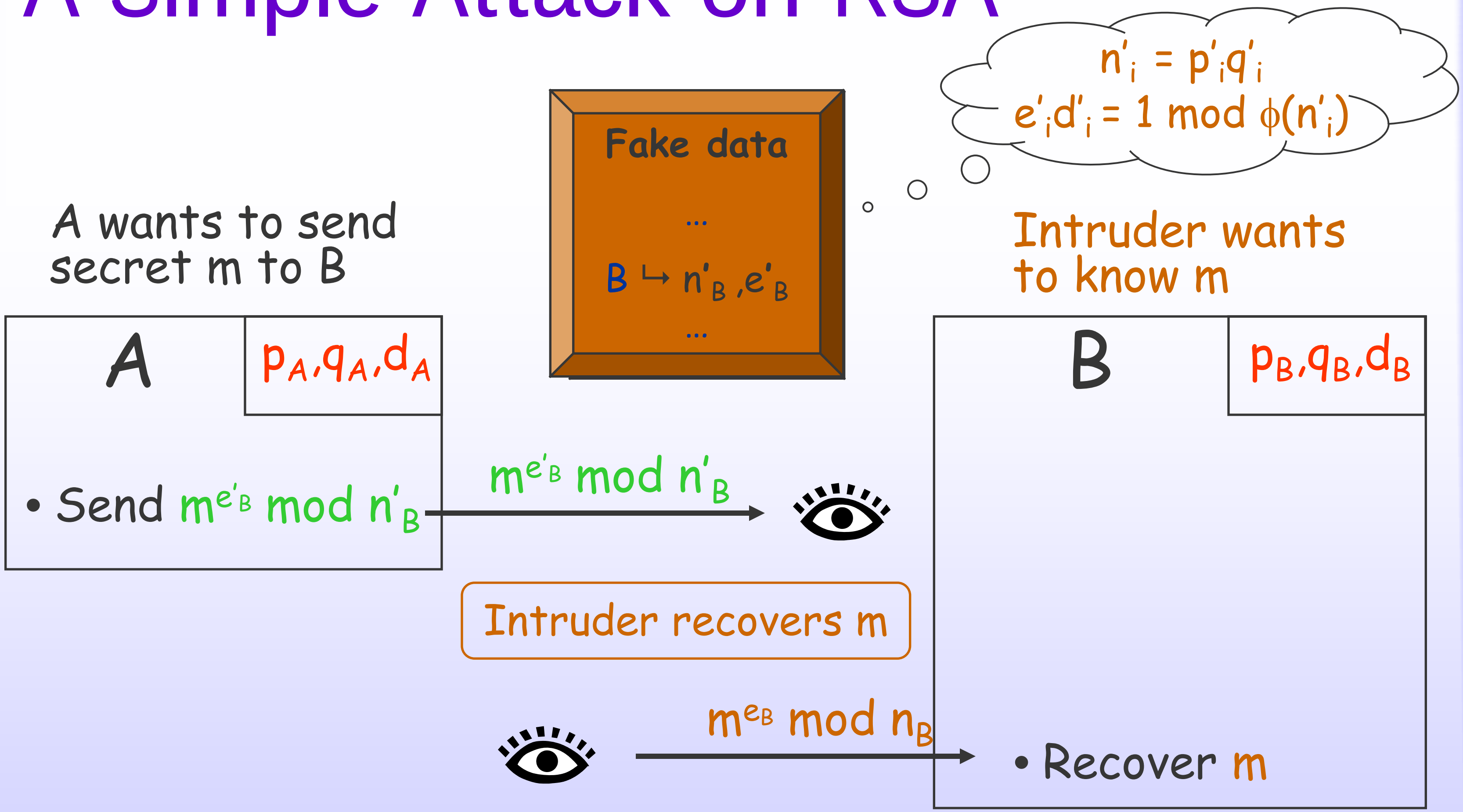
Hashing vs. MAC vs. Signatures



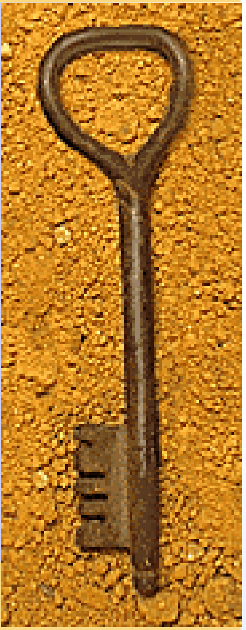
- Hashing: private checksum
  - Produce footprint of a message
  - Must be stored separated from message
- MAC: cryptographic checksum
  - Footprint protected with shared key
  - Can be transmitted over public channel
- Digital signature: taking responsibility
  - Footprint protected with private key
  - No shared secrets with verifier



A Simple Attack on RSA



- How is the public table implemented?



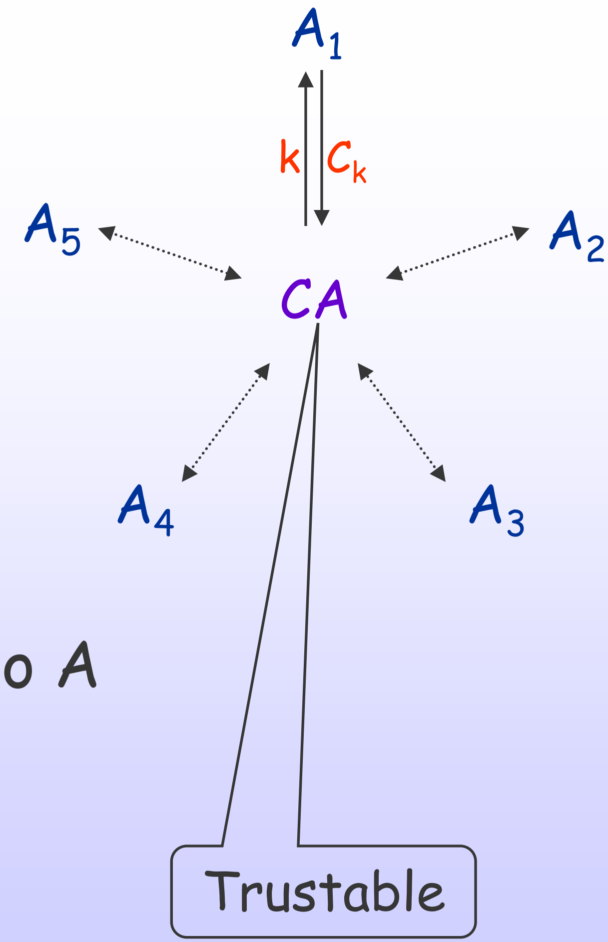
Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

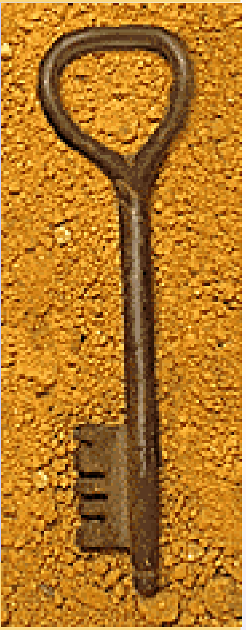
Certification of Published Data

A generates public/private key pair (k,k^{-1}) and wants to publish k on public table

1. A sends k to CA
   - ➢ Certification Authority
2. CA verifies that A knows k^{-1}
   - ➢ Challenge-Response exchange
3. CA generates C_k and sends it to A

- A forwards C_k when using k
  - ➢ Either A volunteers C_k (push)
  - ➢ or sends it on demand (pull)
  - ➢ CA not needed on-line





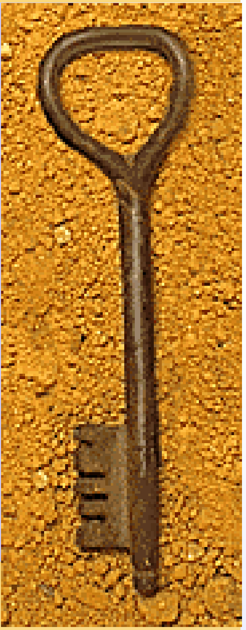
Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

Certificates

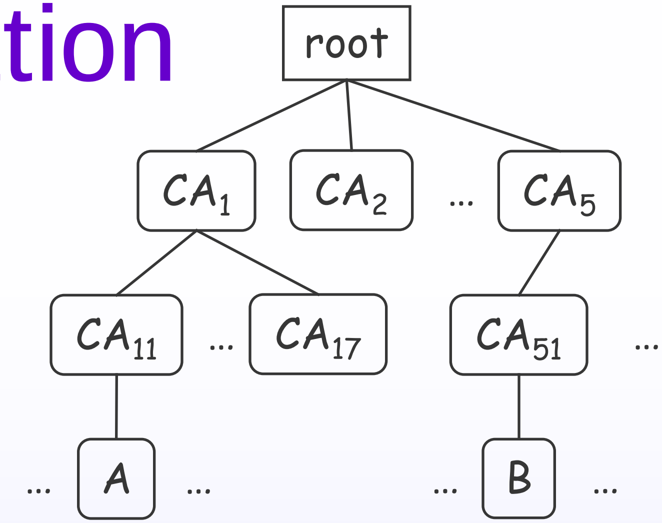
$$C_k = (A, k, t_{exp}, priv, \ldots, sig_{CA})$$

- t_{exp} = expiration date
- priv = privileges
- … = possibly more information

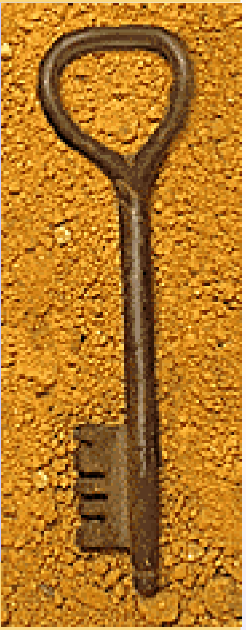
- Everyone knows the verification key of CA
  - Single point of failure
  - Vulnerability as number of principals grows



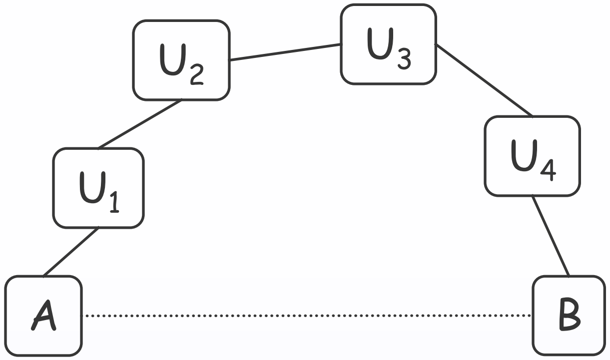
Hierarchical Certification



- Certificate chains
  - Contain certificates of all the nodes to the root
  - Exchanged certificates limited to first common ancestor
- Root signature is trusted and recognizable
  - Redundancy can reduce vulnerability
- Used in SET
  - Developed by Visa/Mastercard
  - Root key distributed among 4 sites

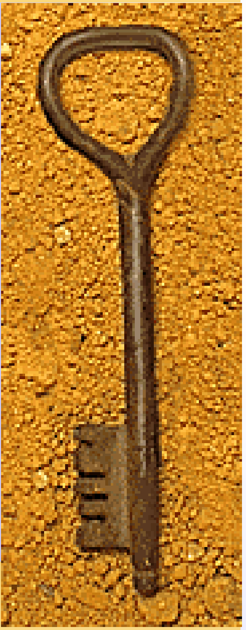


"Web of Trust"



- No central authority
- Users give ratings of keys they used
  - Validity (binding to other user)
  - Trust (none, partial, complete)
- Used in PGP

Certificate Revocation

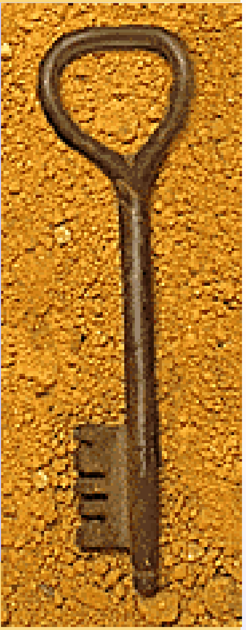


Certificates may be revoked

- A's key is stolen
- Employee leaves the company

- Wait till t_{exp}
  - May be too late
- Certification Revocation List
  - May get blocked
- Validate certificates at fixed intervals

Comparison with KDC



Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI

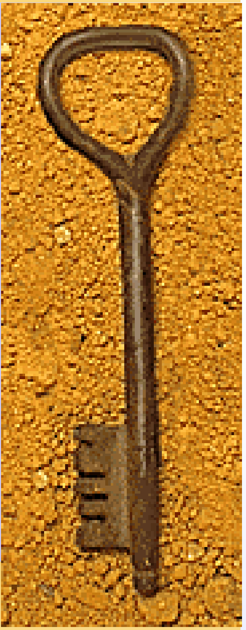
Symmetric keys

- KDC on-line, used at every session
- KDC knows secret key
- If KDC compromised, past and future messages exposed
- Fast

Public key

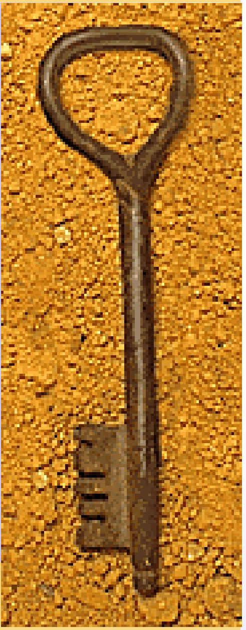
- CA off-line except for key generation
- CA knows only public key
- If CA compromised, only future messages exposed
- Slow

New Trends in Cryptography



- Elliptic-curve cryptography
  - Groups (like Z^*_n) with very hard crypto-analysis
  - Fast and small keys (190 bit ~ 1024 bit of RSA)
  - Complex underlying mathematics
- Quantum cryptography
  - Measuring particle properties destroys them
    - E.g. polarization
  - No eavesdropping without perturbing transmission

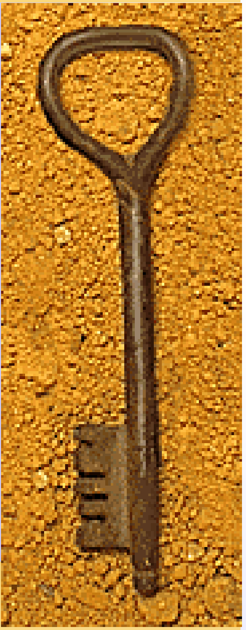
Readings



... references from lecture 2, and also

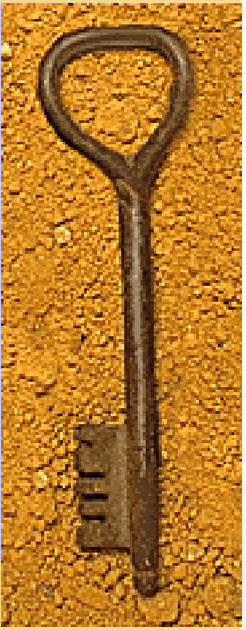
- Douglas Stinson, *Cryptography: theory and practice*, 1995
- Michael Luby, *Pseudorandomness and Cryptographic Applications*, 1996

Exercises for Lecture 3



- Show that Euler's theorem is a generalization of Fermat's little theorem
- Show that El Gamal and DSS signature verifications are correct

Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI



Next ...

- Authentication Protocols

Motivation
Numbers
DH
El Gamal
RSA
Hashing
Signature
PKI