Computer and Network Security: Integrity and Authentication

Kameswari Chebrolu

Outline

- **Modern Cryptography**
  - Overview
  - Confidentiality
    - Background: Definition, Crypto-analysis, One Time Pads
    - Symmetric key encryption, Block modes
    - Asymmetric key encryption
  - **Integrity (includes Authentication)**
    - Hashes, MAC, **Digital signature**

Outline

- Hashes/Message Digests: Data Integrity
- Message Authentication Codes (MACs): Integrity and Authentication
  - Based on Symmetric key model
- **Digital Signatures: Integrity and Authentication**
  - Based on Asymmetric key model

MAC vs Digital Signatures

- Example: Software company releasing periodic patches; integrity of patches important
  - How many keys?
  - How to ensure trust? Bind document to author
- Digital Signatures (based on public key systems)
  - Scalable
  - Easy to verify identity
  - Disputes can be resolved by third parties

Digital Signature: Properties

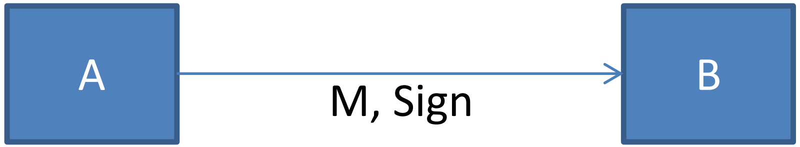
Signing a document: Desirables?

- Non forgeable: No one else other than signer signed it
- Authentic: Signer deliberately signed the document
- Non repudiation: Signer cannot claim she didn’t sign it
- Tamerproof: Document cannot be altered after signature
- Non malleable: signature cannot be cut /paste to another document

Unlike manual signature, digital signature which is a function of document changes from doc to doc

Details

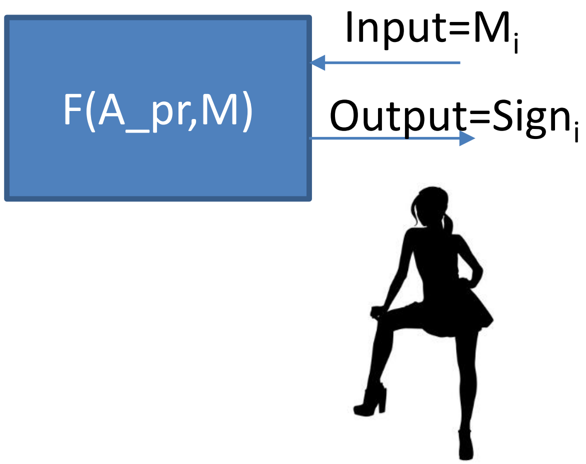
- M: message; A_pu: Public key; A_pr: private key
- A sends message and sign
- B verifies received message with sign
  - Matches, accept (authentic + untampered)
  - No match, reject (tampered/unauthentic or corrupted)



sign = F(A_pr,M)

V(A_pu,M,Sign)
= 1 (accept)
= 0 (reject)

Security Model



Attacker does not know A_pr but knows A_pu

Attacker can input *any* messages M_1, …, M_n of its choice and get corresponding sign

Attacker succeeds if it outputs a <u>forgery</u>; i.e., (M, sign)

$M \neq M_i$ for all i

V(A_pu,M,sign) = 1

Want Pr[winning] ~ 0 (time bound)

Construction: Focus on RSA

Recap:

- p,q: two large prime numbers
- n = p*q; $\phi(n)$= (p-1)(q-1)
- Pick e relative prime to $\phi(n)$
- Set $d = e^{-1} \, mod \, \phi(n)$ (p,q and $\phi(n)$ immaterial)
- Public key (e,n)
- Private key (d,n)

- Encrypt m (<n) to generate ciphertext c

$$c = m^e \, mod \, n$$

- Decrypt c to recover m

$$m = c^d \, mod \, n$$

Signature Construction Details

- Public key (e, n); private key (d,n)
- Signature (S) on Message M: S = M^d mod n
- Verification of (M,S): check whether S^e == M nod n (accordingly V(A_pu,M,sign) is 1 or 0)

$$S^e mod n = M^{de} \, mod\, n = M^{de \, mod \, \phi(n)} \, mod \, n = M \, mod \, n$$

Properties

- Non forgeable: Attacker has to produce M^dmod n but without knowing d (crux of RSA)
- Authentic: Only signer has access to d; any third party can easily verify it
- Tamerproof: Signature is tightly bound to M
- Non malleable: ?
- Non repudiation: ?

Attack-1 (Malleable)

- Attacker has two valid signatures on messages M_1 and M_2

$$S1 = M_1^d \, mod \, n \, ; \, S2 = M_2^d \, mod \, n$$

- Attacker can produce (on his own, without knowing d) new signature S on message $M = M_1 . M_2$

$$S = S_1.S_2 \, mod \, n = (M_1.M_2)^d \, mod \, n$$

Attack-2 (Repudiation)

- Attacker wants A's (say notary public) signature S on message M
- Pick arbitrary x and get A to sign $M' = Mx^e \bmod n$
- A returns $S' = (Mx^e)^d \bmod n$
- $S = S'/x = M^d \bmod n$
- S is a valid signature of M

Attack-3 (Encryption)

- Attacker has ciphertext c (of A), wants to read original message m; m = c^d mod n
- Attacker chooses random r < n and calculates

$$x = r^e \, mod \, n; y = xc \, mod \, n$$

- Attacker gets A to sign y (i.e. gets y^d mod n)
- Then computes

$$r^{-1} y^d \, mod \, n = r^{-1} x^d c^d \, mod \, n = c^d \, mod \, n = m$$

Digital Signatures with Hash

- Digital Signature on message
  - Expensive operation if M is long
  - Can be insecure (Attack-1 and 2)
- In practice, digital signatures are applied to hash of messages
  - Send M, S where S = F(A_pr, hash(M))
  - Receive M,S; verify whether V(A_pu,hash(M),S) == 1
- Insecure if hash is not collision resistant

Lessons Learnt

- Use different keys for encryption and digital signature
  - Makes it more secure (prevents attack-3)
  - Can surrender one key (to authorities) while retaining the other
  - Both can have different lifetimes
- Always sign hash of messages

DSS Standard

- Proposed by NIST for digital signatures in 1991
- Algorithm is DSA, a variant of ElGamal signature scheme (royalty free)
  - Difficult and non-intuitive math
  - Why not RSA? Likely due to patenting issues
- https://en.wikipedia.org/wiki/Digital_Signature_Algorithm (steps are straightforward)

Digital Signatures and Encryption

- Want confidentiality, integrity and authentication
- A signs message with private key (S= F(A_pr,M)) or (S= F(A_pr,hash(M)))
- A encrypts signed messaged with B's public key (C=F(B_pu,S)) or (C=F(B_pu,M|S))

Digital Signatures and Encryption

- B decrypts message with private key (F(B_pr,C)= S)
- B verifies with A's public key and recovers the message (F(A_pu,S) = M)
- Signing before encryption important (natural, secure and legal)

Summary

- Integrity + Authentication crucial in many scenarios
- Achieved by
  - MACs based on symmetric key crypto
  - Digital signatures based on asymmetric key crypto
- MACs: Construction both with symmetric key algo and hashes; vulnerabilities and solutions
- Digital Signatures: Properties; RSA based signatures; Attacks and Fixes; DSS standard