CS 161 – Signatures & Secret Sharing

18 September 2006

Attacks on cryptography

- Direct attack
  - example: exhaustive search
- Known plaintext
- Chosen plaintext

- Usual assumptions: chosen plaintext attack; attacker knows E, D but not key

Notation

- Ciphertext = Encryption (Plaintext, encryption-Key)
  - sometimes we use "cleartext" instead of "plaintext"
- Key $\in$ Keyspace
- Keysize = $\log_2($ |Keyspace| $)$
- $c=E(m,k)$ (or $c=E_k(m)$ or $c=\{m\}_k$)
- Also Plaintext = Decryption(Ciphertext, decryption-Key)
- encyption-Key = decryption-Key (symmetric)
- encyption-Key ≠ decryption-Key (asymmetric)
- $m=D(c,k)=E^{-1}(c,k)$ (or $c=D_k(m)$)

RSA

- Rivest, Shamir, Adleman (1978 – published 1979)
- Idea:
  - Given e, find d, such that $ed = K(p-1)(q-1)+1$ for some K
  - Encryption: $c = E(m) = m^e \bmod pq$
  - Decryption: $D(c) = c^d \bmod pq$
  - So $D(E(m)) = m^{ed} \bmod pq = m^{K(p-1)(q-1)+1} \bmod pq = m$
- Issues:
  - Given e, how can we find d?
    - Answer: use EGCD (extended greatest common divisor)
      - Euclidean algorithm
    - Given x, y, EGCD finds $Ax + By = GCD(x, y)$
    - Let $x=e$, $y=(p-1)(q-1)$, then $Ae = (-B)(p-1)(q-1) + 1$
  - How can compute exponentiation modulo pq fast?
    - Repeated squaring mod pq – use binary form of number

RSA allows for "public keys"

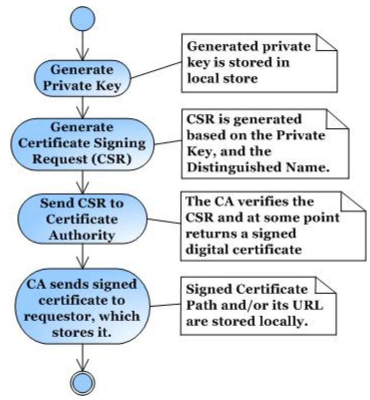
- Encryption key public, decryption key private
  - Easy way to send secret messages
  - If we can guess plaintext, we can break (so we add random bits)
  - Decryption only by intended recipient
  - Perfect for distributing symmetric keys
- Encryption key private, decryption key public
  - Only I can send messages, anyone can verify (and read)
  - A type of "digital signature"
  - We will develop this idea in detail

Asymmetric crypto

- Advantages
  - Doesn't require advance set up
  - Strongest forms are as hard as factoring
  - Perfect for solving key distribution problem
  - Good for building protocols
- Disadvantages
  - Slow, slow, slow (& takes space too)
  - Secrecy & source authentication takes two encryptions
  - Need to find a way to prove "public keys" are honest
    - Future lecture: public key hierarchy

How do we know a public key?

- One approach – the big directory (white pages)
  - Need to make secure big directory
  - Need to keep it updated
- Better approach: allow one party to attest to another
  - Public key infrastructure (PKI)
  - Public key certificate (PKC)
  - Certificate authority (CA)



A hypothetical public-key hierarchy

Doug Tygar's public key is …
Love, Arnold Schwarzenegger

Digitally signed by AS

A hypothetical public-key hierarchy

Arnold Schwartzenegger's public key is …
Love, George Bush Jr.

Digitally signed by W

↓

Doug Tygar's public key is …
Love, Arnold Schwarzenegger

Digitally signed by AS

A hypothetical public-key hierarchy

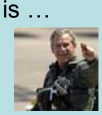
George Bush Jr.'s public key is …
Love, Kofi Annan

Digitally signed by Kofi

↓

Arnold Schwartzenegger's public key is …
Love, George Bush Jr.

Digitally signed by W



↓

Doug Tygar's public key is …
Love, Arnold Schwarzenegger

Digitally signed by AS

Replay attacks

- Cryptosystems are vulnerable to replay attacks.
- Record message; playback later identically
- "Yes"/"No"

- Solution: use nonces (random bits; timestamp) etc.

- Message is <text, timestamp>

Keeping a secret

- Suppose we want to keep a secret among t people
- One way to do this is to set secret = $\sum \text{secret shares} \pmod{n}$
- Another way is exploit linear equations

$$f(x) = x^q + a_{q-1}x^{q-1} + \cdots + a_1 x + a_0 \pmod{p}$$

- Secret = a_0
- Distribute $f(1), f(2), \ldots, f(t)$
- Now a quorum q of those people can recover the secret

Factoring & RSA

- Factoring is easy → RSA is easy
- We have not proved that RSA is as hard as factoring.

- We need better cryptosystems
  - Secret sharing – allows party to store message secretly
  - Rabin signatures – equivalent to factoring

Chinese Remainder Theorem

- We can represent numbers mod pq
- Alternatively as a pair mod p and mod q

- 1 = <1 mod 3, 1 mod 5>
- 7 = <1 mod 3, 2 mod 5>
- 12 = <0 mod 3, 2 mod 5>

Square roots

- This means that a square root mod pq has four roots.
- Suppose that $r^2 = m \bmod pq$
- And r = <s mod p, t mod q>
- Then for square roots are:
- <s mod p, t mod q>
- <-s mod p, t mod q>
- <s mod p, -t mod q>
- <-s mod p, -t mod q>
- If we can find the square roots, then we can factor pq
- <s mod p, t mod q> + <-s mod p, t mod q> =
  <0 mod p, 2t mod q> = multiple of p

Rabin Signature algorithm

- If we can factor pq, it is easy to take square roots
- This means square roots are a great signature
- Easy to verify (just take a square)
- If someone has a square root taking algorithm then he can factor easily.

- Square roots $\leftrightarrow$ factoring