Asymmetric Cryptography

Prof. Ravi Sandhu
Executive Director and Endowed Chair

Lecture 3

ravi.utsa@gmail.com
www.profsandhu.com
Asymmetric Encryption
Public-Key Encryption

INSECURE CHANNEL

A

Encryption Algorithm E

Ciphertext

B’s Public Key

B

Decryption Algorithm D

Plain-text

SECURE CHANNEL
Confidentiality
Integrity

B’s Private Key
Symmetric-Key Encryption

INSECURE CHANNEL

Plain-text → Encryption Algorithm E → Ciphertext → Decryption Algorithm D → Plain-text

A
Symmetric Key shared by A and B
K
K

SECURE CHANNEL
Confidentiality
Integrity
Public-Key Encryption

- reduces the key distribution problem to a secure channel for authentic communication of public keys
- requires authentic dissemination of 1 public key/party
- scales well for large-scale systems
  - with N parties we need to generate and distribute N public keys
Known Public-Key Attack

- confidentiality based on infeasibility of computing B's private key from B's public key
- key sizes are large (2048 bits and above) to make this computation infeasible
public key runs 1000 times slower than symmetric key
- think 2g versus 4g on smartphone
This large difference in speed is likely to remain
- Maybe reduce to 100 times

Use public keys to distribute symmetric keys, use symmetric keys to protect data
RSA Cryptosystem

- public key is \((n,e)\)
- private key is \(d\)
- encrypt: \(C = M^e \mod n\)
- decrypt: \(M = C^d \mod n\)
RSA Cryptosystem

- public key is \((n,e)\)
- private key is \(d\)
- encrypt: \(C = M^e \mod n\)
- decrypt: \(M = C^d \mod n\)

This naive use of RSA is not secure but will suffice for our purposes.
RSA Key Generation

- choose 2 large prime numbers p and q
- compute n = p * q
- pick e relatively prime to (p-1)*(q-1)
- compute d, e*d = 1 mod (p-1)*(q-1)
- publish (n,e)
- keep d private (and discard p, q)
- compute $d$, $e\cdot d = 1 \mod (p-1)\cdot(q-1)$
- if factorization of $n$ into $p\cdot q$ is known, this is easy to do
- security of RSA is no better than the difficulty of factoring $n$ into $p$, $q$
Asymmetric Digital Signatures
Public-Key Digital Signature

INSECURE CHANNEL

Plain-text → Signature Algorithm S → Plaintext + Signature → Verification Algorithm V → Yes/No

A's Private Key → A

A's Public Key → B

SECURE CHANNEL
Confidentiality
Integrity
Compare Public-Key Encryption

INSECURE CHANNEL

A

Encryption Algorithm E

B's Public Key

B

Decryption Algorithm D

B's Private Key

Plain-text

Ciphertext

SECURE CHANNEL
Confidentiality
Integrity

World-Leading Research with Real-World Impact!
Compare Symmetric Key MAC

INSECURE CHANNEL

Plain-text → MAC Algorithm M → Plaintext + MAC → Verification Algorithm V → Yes/No

SECURE CHANNEL
Confidentiality
Integrity

A → K → B → K

© Ravi Sandhu

World-Leading Research with Real-World Impact!
RSA has a unique property, not shared by other public key systems
- Encryption and decryption commute
- \((M^\theta \mod n)^d \mod n = M\) encryption
- \((M^d \mod n)^\theta \mod n = M\) signature
- Same public key can be use for encryption and signature
  - But not recommended
Message Digest
- public key runs 1000 times slower than symmetric key
  - think 2g versus 4g on smartphone
- This large difference in speed is likely to remain
  - Maybe reduce to 100 times

- Use public keys to distribute symmetric keys, use symmetric keys to protect data
- public key runs 1000 times slower than symmetric key
  - think 2g versus 4g on smartphone
- This large difference in speed is likely to remain
  - Maybe reduce to 100 times

- Sign the message digest (or hash) not the message
Message Digest (Hash)

original message
no practical limit to size

message digest algorithm

message digest
256 bit

m = H(M)

M = H⁻¹(m)
Desired Characteristics

- weak hash function
  - difficult to find M' such that H(M') = H(M)
- given M, m = H(M) try messages at random to find M' with H(M') = m
  - $2^k$ trials on average, $k = 128$ to be safe
Desired Characteristics

- strong hash function
  - difficult to find any two $M$ and $M'$ such that $H(M')=H(M)$
- try pairs of messages at random to find $M$ and $M'$ such that $H(M')=H(M)$
  - $2^{k/2}$ trials on average, $k=256$ to be safe

Birthday paradox
Message Authentication Code

Symmetric Encryption Based

CBC-MAC

MAC has same size as block size of underlying cryptosystem

CCM mode
Provides confidentiality and integrity

Message-Digest Based

HMAC

Hash the message and a symmetric key

MAC has same size as underlying hash function or can truncate

Revisiting after discussing message digests
Asymmetric Key Exchange
Diffie-Hellman Key Agreement

A

\[ y_A = a^{x_A} \mod p \]

public key

B

\[ y_B = a^{x_B} \mod p \]

public key

private key

\[ x_A \]

private key

\[ x_B \]

k = \( y_B^{x_A} \mod p = y_A^{x_B} \mod p = a^{x_A \cdot x_B} \mod p \)

system constants: \( p: \) prime number, \( a: \) integer
security depends on difficulty of computing $x$ given $y=a^x \mod p$

called the discrete logarithm problem
Diffie-Hellman Man-in-the-Middle Attack

Public keys need to be authenticated