Public Key Cryptography - RSA

Information Security – Lecture 08

Aadil Zia Khan

Symmetric Key Cryptography

- Symmetric/private/secret/single key cryptography uses only one key
  - Sender and receiver share the key
- If key is disclosed the data and communication would be compromised
- Does not protect sender from the receiver forging a message and claiming it was sent by sender – because both share the same key
- Does not ensure Non repudiation – because both share the same key

The biggest problem here is that the key is shared

Public Key Cryptography

- Uses two keys
  - Public key – anyone can access it
  - Private key – only the owner can access it

- It is asymmetric
  - Owner of the private key is not equal to the people having access to the public key
  - Private and public key are used for separate tasks – this separation of tasks make it so powerful

Why the Need for Public Key Cryptography
Scenario 1: Encryption

- Assume we are using symmetric key encryption
- Suppose Alice encrypts a message and sends it to Bob
- There is only one private key
- How will she share the key with Bob?
  - She can send it over the same channel – but if eve can intercept the message, she can also intercept the key
  - She can send it after a long delay so that the key is sent separately from the message – this would just increase delays and besides, anything sent over an insecure channel is inherently not safe

Ideal Solution - Scenario 1: Encryption

- Suppose Bob has two keys – one public (known to everyone) and one private (known only to Bob)
- Alice uses Bob's public key to encrypt the message and sends it
- This message can only be decrypted using Bob's private key (which no one has access to)
- Even if the message is intercepted by Eve, the private key would be safe because it is never shared

Why the Need for Public Key Cryptography
Scenario 2: Authentication

- If a message is encrypted using a symmetric key – can we say that the owner of the key is the owner of the message as well?
- No
  - Assume we are using symmetric key encryption
  - Suppose Alice and Bob have shared the key
  - A message is created and encrypted using this key
  - Who encrypted it - Alice or Bob?
  - If Bob created something illegal – Alice could get caught

Ideal Solution - Scenario 2: Authentication

- Suppose Alice created a document
- Suppose Alice has two keys – one public (known to everyone) and one private (known only to Alice)
- She will generate a hash value of that document
- She will encrypt the hash value using her private key (only she can do that – no one else can)
- Anyone who wants to check if the document was created by Alice will
    - Calculate the hash value of the document
    - Decrypt the hash value sent by Alice using Alice's public key
    - Compare the two hash values – if both are same it means that the document was not modified and was created by Alice

What If We Want Both Encryption And Authentication

- Suppose Alice wants to send Bob a confidential message and also wants to assure Bob that the message is from her
- Alice first encrypts the message (or its hash) using her private key
  - When any receiver decrypts using Alice's public key – he would know the it is from Alice
- Alice then encrypts the message using Bob's public key
  - Only Bob can decrypt it because no one else has Bob's private key

Public Key Cryptography Uses

- Can be classified into three broad categories:
    - Encryption/Decryption (for confidentiality)
    - Digital signatures (for authentication)
    - Key exchange

Requirements That Any Public Key System Should Meet

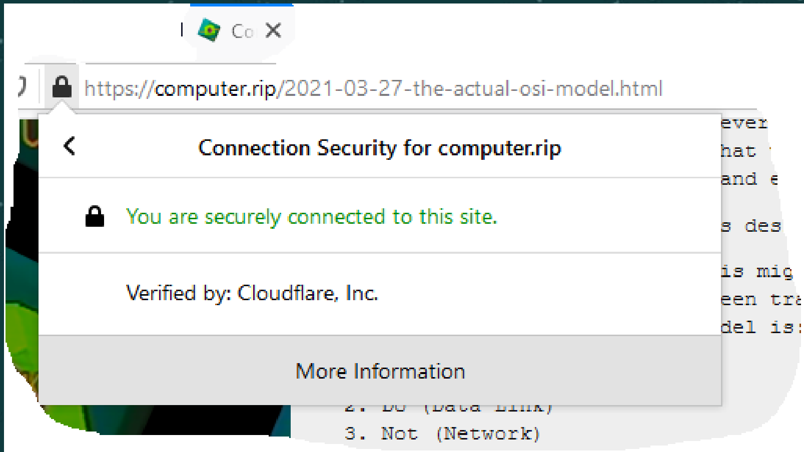
- Three main requirements
  - It must be easy to encrypt or decrypt a message given the appropriate key
  - It must be computationally infeasible to derive the private key from the public key
  - It must be computationally infeasible to determine the private key from available plaintext-ciphertext pairs

RSA

Public Key Info

Algorithm	RSA
Key Size	2048
Exponent	65537
Modulus	A3:04:BB:22:AB:98:3D:57:E8:26:72:9A:B5:79:D4:29:E2:E1:E8:95:80:B1:B0:E3:5B:8E:2B:29:9A:64:DF:A1:5D:ED:B0:09:05:6D:DB:28:2E:CE:62:A2:62:FE:B4:88:DA:12:EB:38:EB:21:9D:C0:41:2B:01:52:7B:88:77:D3:1C:8F:C7:BA:B9:88:B5:6A:09:E7:73:E8:11:40:A7:D1:CC:CA:62:8D:2D:E5:8F:0B:A6:50:D2:A8:50:C3:28:EA:F5:AB:25:87:8A:9A:96:1C:A9:67:B8:3F:0C:D5:F7:F9:52:13:2F:C2:1B:D5:70:70:F0:8F:C0:12:CA:06:CB:9A:E1:D9:CA:33:7A:77:D6:F8:EC:B9:F1:68:44:42:48:13:D2:C0:C2:A4:AE:5E:60:FE:B6:A6:05:FC:B4:DD:07:59:02:D4:59:18:98:63:F5:A5:63:E0:90:0C:7D:5D:B2:06:7A:F3:85:EA:EB:D4:03:AE:5E:84:3E:5F:FF:15:ED:69:BC:F9:39:36:72:75:CF:77:52:4D:F3:C9:90:2C:B9:3D:E5:C9:23:53:3F:1F:24:98:21:5C:07:99:29:BD:C6:3A:EC:E7:6E:86:3A:6B:97:74:63:33:BD:68:18:31:F0:78:8D:76:BF:FC:9E:8E:5D:2A:86:A7:4D:90:DC:27:1A:39

- Rivest-Shamir-Adleman
- Most well-known public key system
- Used in TLS/HTTPS
  - For key exchange and digital signing



RSA: Key Generation

Note: Two numbers are relatively prime if the only positive integer that evenly divides both of them is 1

For example, 12 and 13 are relatively prime, but 12 and 14 are not

- Choose two very large primes, p and q
  - Each should be more than 100 or 200 digits
- Compute the product $N = p \times q$
- Compute the product $(p-1) \times (q-1)$
  - This is called the Totient Function
- Choose a random positive number e which is less than N such that e and $(p-1)(q-1)$ are relatively prime
- Determine the positive integer d such that
  - $e \times d \equiv 1 \bmod (p-1)(q-1)$ which is the same as $e \times d \bmod (p-1)(q-1) = 1$

Public Key: {e,N} Private Key: {d} Discard p and q without revealing them

RSA: Encryption & Decryption

- Divide the message into blocks of size less than N
- Let c be the ciphertext and m be the plaintext message
- To encrypt, use the following formula
  - $c = m^e \bmod N$
- To decrypt, use the following formula
  - $m = c^d \bmod N$

RSA: Example

- Suppose we selected $p = 47$ and $q = 71$
- $N = 47 \times 71 = 3337$
- $(p-1) \times (q-1) = 46 \times 70 = 3220$
- Let us randomly select $e = 79$ because i) it is less than 3337 and ii) 79 and 3220 are relatively prime
- Solving for $e \times d \bmod (p-1)(q-1) = 1$ we see that $d = 1019$
    - $79 \times 1019 / 3220$ gives a remainder of 1
- Now suppose message $m = 6882326879666683$
- Split m into 3 digit blocks: 688 232 687 966 668 003

- First plaintext block is encrypted as $688^{79} \bmod 3337$ which gives the ciphertext 1570
- First ciphertext block is decrypted as $1570^{1019} \bmod 3337$ which gives the plaintext 688
- Do this for all blocks

RSA: Encryption Vs Digital Signature

- Encryption (Alice sends an encrypted message to Bob)
  - Alice uses Bob's public key to encrypt the message and sends it
  - This message can only be decrypted using Bob's private key

- Digital Signature (Alice signs a message)
  - Alice encrypts the message (or its hash) using her private key
  - When any receiver decrypts using Alice's public key – he would know the it is from Alice

RSA For Text Data

- Does RSA only work with numeric data?
  - No – all data (even text) is a sequence of bits that can be treated as a number for RSA

Securing RSA

- RSA's strength lies in the fact that given a very large number, it is currently impossible to break it down into two factors which are very large prime numbers
  - If the numbers are very small, it would be very easy

- Therefore, p and q should be at least 512 bits each and N at least 1024 bits

RSA Running Speed Comparison With DES

- A while back, the comparison results were:
  - In hardware: RSA is 1000 times slower than DES
  - In software: RSA is 100 times slower than DES