Public Key Cryptography Series - Session 1

Kaushik S Kalmady
IEEE CompSoc Systems and Security Group

What we will cover today:

Introduction to public key cryptography - What is it and why do we need it?

RSA Cryptosystem - Encryption and Signing

Application in Digital Certificates (HTTPS)

Before We Begin

“The quality of any advice anybody has to offer has to be judged against the quality of life they actually lead.”

— Douglas Adams

https://github.com/rsa-ctf/write-ups https://ctftime.org/team/62190

What is cryptography?

Cryptography is the practice and study of techniques for secure communication in the presence of third parties called adversaries.

It is about constructing and analyzing protocols that prevent third parties or the public from reading private messages.

- Wikipedia

https://en.wikipedia.org/wiki/Cryptography

Repeat with me ...

Our main aim is to perform

“SECURE COMMUNICATION IN AN INSECURE CHANNEL”

Encryption

In cryptography, encryption is the process of encoding a message or information in such a way that only authorized parties can access it and those who are not authorized cannot.

1. Symmetric Key
2. Public Key

https://en.wikipedia.org/wiki/Encryption

Symmetric Key Cryptography

Uses same keys for encryption and decryption!

- Encryption of Message M with key K,
  C = E(M , K)
- Decryption of Ciphertext C with same key,
  M = D(C, K)
- Examples: Vigenere, AES, etc.

What’s good about them?

- Extremely fast
- Scales well even for large inputs

https://en.wikipedia.org/wiki/Symmetric-key_algorithm

So what’s the problem?

Both parties need to agree on the same key.

How do they share a common key in an insecure channel?

Repeat with me ...

Our main aim is to perform

“SECURE COMMUNICATION IN AN INSECURE CHANNEL”

Where really is the security?

In almost all of crypto, the security stems from the privacy of the key. The algorithms are all in the public domain.

So the security of your encryption relies completely on the privacy of your key.

Therefore, If you can’t set up a secure channel to exchange keys, Symmetric Key crypto provides no security.

Asymmetric/ Public Key Cryptography

- Here, each party has its own (public key, private key) pair
- Each party's public key is out in the public. Anyone can use it to encrypt their message and send it to the intended party.
- Ideally, only the intended party can decrypt this message using their private key(which is kept secret by each party).

  If Alice wants to talk to Bob, she sends C = E(M, BobPublicKey)

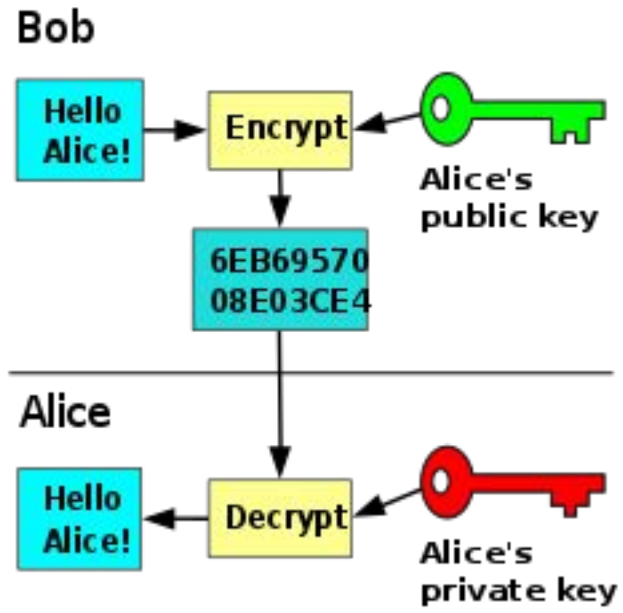
  On receiving the message, Bob does M = D(C, BobPrivateKey)

Disclaimer: This slide is an oversimplification

But what does it all really mean ?

Key Insights (pun intended)

- The public key helps us to ***establish*** the secure channel to communicate with the intended party
- The private key is the backdoor that the intended party possesses to decrypt the communication
- So a person’s public key is like a signpost put out on the road saying “Use this to set up a secure channel to talk to me!”
- The public key and private key cancel out each other’s effect.



But wait ...

- If the public and private keys cancel out each others effects, then there must be some relationship between the two?
- Since the encryption, decryption algorithms are public, what is stopping an attacker from obtaining the private key given the public key ?
  - Note: Since the algorithm is public, so is the relationship between the public key and private key.
  - How then, are these cryptosystems secure?

Things get a little intense from here on

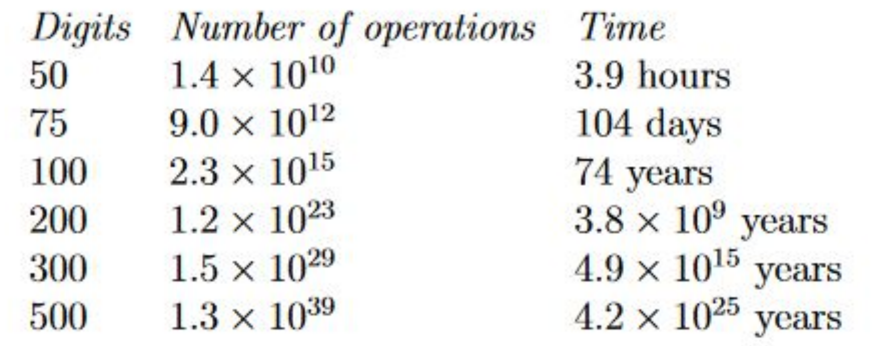
How hard can it be? NP

- The relationship between public and private key is modelled such that obtaining private key knowing the public key is computationally intractable.
- At the same time, the cryptosystem should ensure that encryption, decryption and key generation is trivial for the intended user.

Again, we can observe, security of the encryption depends on the privacy of the key, which in this case depends on the “hardness” of the underlying problem it is modelled on.

RSA and Integer factorisation

- Public Key cryptosystem modelled on the Integer Factorisation problem
- Difficulty of integer factorisation



Digits	*Number of operations*	*Time*
50	1.4×10^{10}	3.9 hours
75	9.0×10^{12}	104 days
100	2.3×10^{15}	74 years
200	1.2×10^{23}	3.8×10^{9} years
300	1.5×10^{29}	4.9×10^{15} years
500	1.3×10^{39}	4.2×10^{25} years

https://people.csail.mit.edu/rivest/Rsapaper.pdf https://en.wikipedia.org/wiki/Integer_factorization

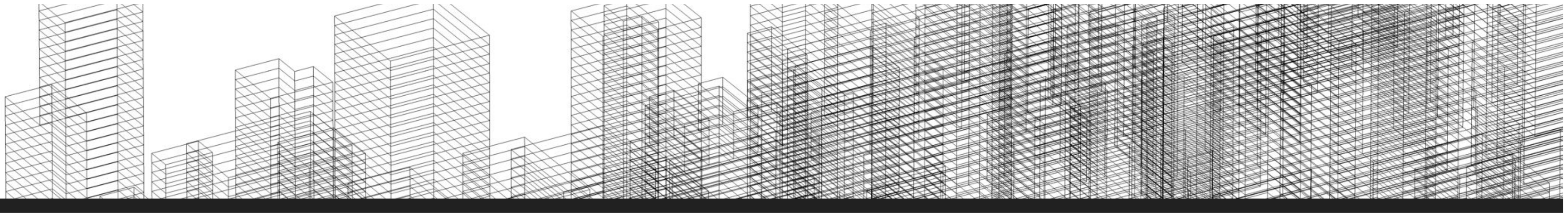
Before diving into the algorithm

- Euler Totient Function $\phi(n)$: counts the positive integers up to a given integer n that are relatively prime to n.
- Euler's Theorem:

  if n and a are coprime positive integers, then

  $a ^ \phi(n) \equiv 1 \pmod{n}$

https://en.wikipedia.org/wiki/Euler%27s_theorem



Disclaimer: You’ll need a bit more math to understand this clearly.

Go back and do some homework, most of it is highschool math

The RSA Algorithm

https://en.wikipedia.org/wiki/RSA_(cryptosystem)
https://people.csail.mit.edu/rivest/Rsapaper.pdf

Key Generation

1. Choose two distinct prime numbers p and q.
2. Compute $n = pq$.
    a. n is used as the modulus for both the public and private keys.
3. Compute $\phi(n) = \phi(p).\phi(q) = (p-1)(q-1)$
4. Choose an integer e such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$; i.e., e and $\phi(n)$ are coprime.
5. Determine d as d ≡ e^−1 (mod $\phi(n)$); i.e., d is the modular multiplicative inverse of e modulo $\phi(n)$
    a. $ed \equiv 1 \pmod{\phi(n)}$
6. e is released as the public key exponent.
7. d is kept as the private key exponent.

https://en.wikipedia.org/wiki/RSA_(cryptosystem)#Key_generation

Encryption and Decryption

Encryption (uses public key):

$$C = M ^ e \bmod (n)$$

Decryption (uses private key):

$$M = C ^ d \bmod (n)$$

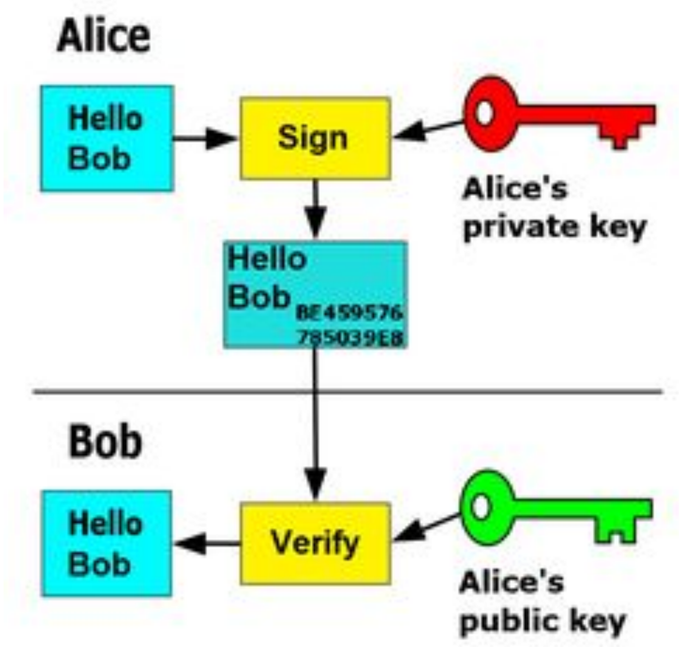
It works, refer the paper, wiki for the math and proof.

Important Questions

- Can any random primes p, q be chosen?
- Why/How do the decryption and encryption operations cancel each other out?
- What ensures the security of the algorithm?
- The encryption looks deterministic!?

What if I encrypt using the private key ?

- We achieve Digital Signatures!
- Alice can sign a message using her private key. Any other entity can verify that the message was indeed signed by Alice by using her Public Key!
- Since no one other than Alice has her private key, it must have been signed by Alice!



Problems

- Not as fast as symmetric key algorithms
- Gets slower as the message size increases

This is mitigated using a simple trick

1. Use Public Key crypto to exchange keys for Symmetric Encryption (Similar to setting up a secure channel to exchange keys)
2. Use Symmetric Encryption with shared key for all future communication!

[redacted]

Digital Certificates

https://robertheaton.com/2014/03/27/how-does-https-actually-work/

Resources for further reading

Twenty Years of Attack on the RSA cryptosystem -
https://crypto.stanford.edu/~dabo/papers/RSA-survey.pdf

Block Cipher mode of encryption (How AES handles variable length plaintext) -
https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

Cryptography Series by Christof Paar -
https://www.youtube.com/channel/UC1usFRN4LCMcfIV7UjHNuQg

Hacking: The Art of Exploitation - Book by Jon Erickson: Please read this. It is not exactly a book on crypto, but a great example of how we can learn things by breaking them. That philosophy applies to all of computer science (Especially in this age of copypasta).

Crypto challenges from CTFs

For crypto(and other) challenges of varying difficulty - https://2018game.picoctf.com/

RSA Power Analysis Side Channel Attack - https://www.youtube.com/watch?v=bFfyROX7V0s

RSA Blinding attack - https://github.com/rsa-ctf/write-ups/tree/master/noxctf18/Decryptor

Ectf crypto challenges - https://github.com/rsa-ctf/ectf2018/tree/master/crypto

(You can mail me for writeups, make sure you explain all the approaches you have tried)