Digital Signature

Introduction

Goal

Build a signture-like function for digital world.

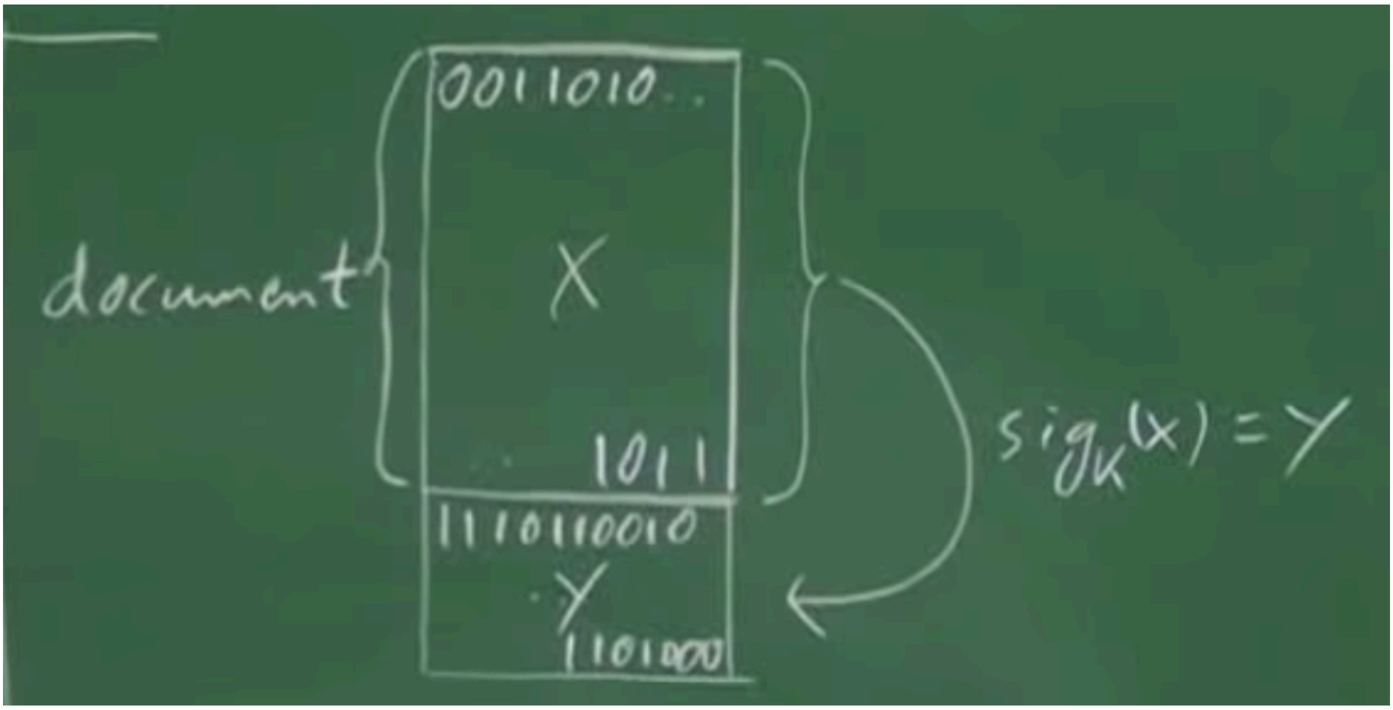
What is a signature?

a signature is a "proof of authentication of the sender".

If we have a signature on something, we can sure that this thing must come from the person who sign it.

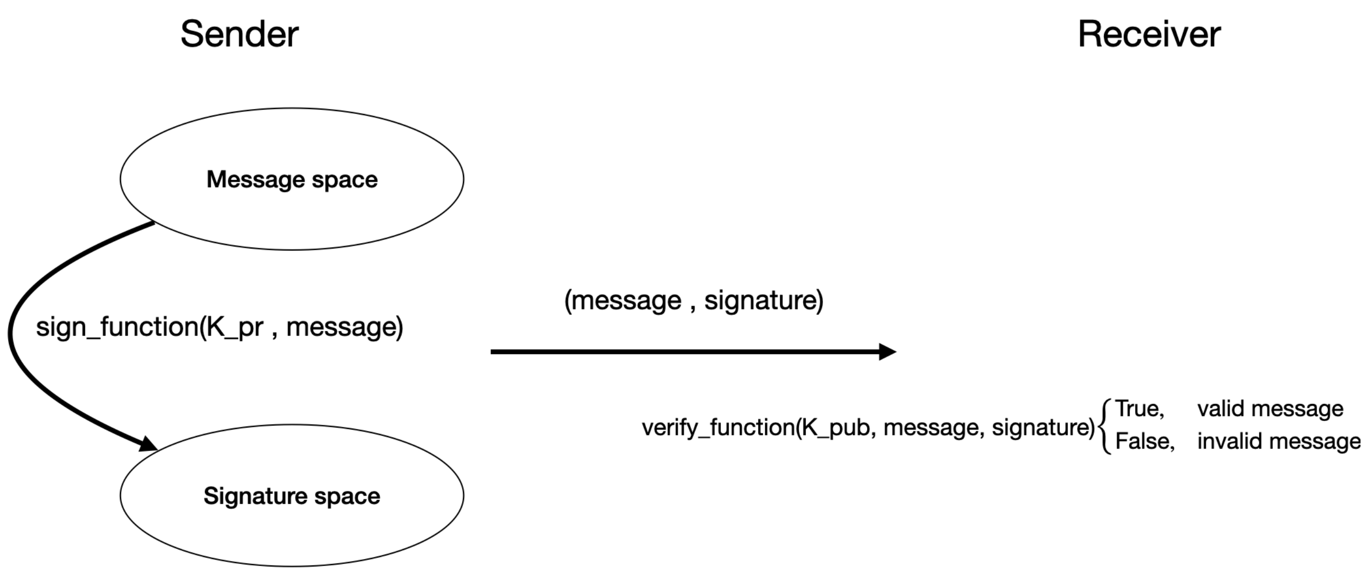
How to do that?

- The method of creating something unique to a person and then append it to the documentation to be signed does not work because once the doc was sent via the Internet, everyone can copy the unique bitstrings.
- we resort to cryptography, building a signing function




```
def sign(message , private_key):
    return signature_for_this_particular_message
# the key should be like a private key, only be known
to the person who sign the message

def verify(message , signature , public_key):
    if valid: return True
    else: return False
```



The receiver can only verify the signature but cannot generate the signature himself due to lack of private key

Security Services

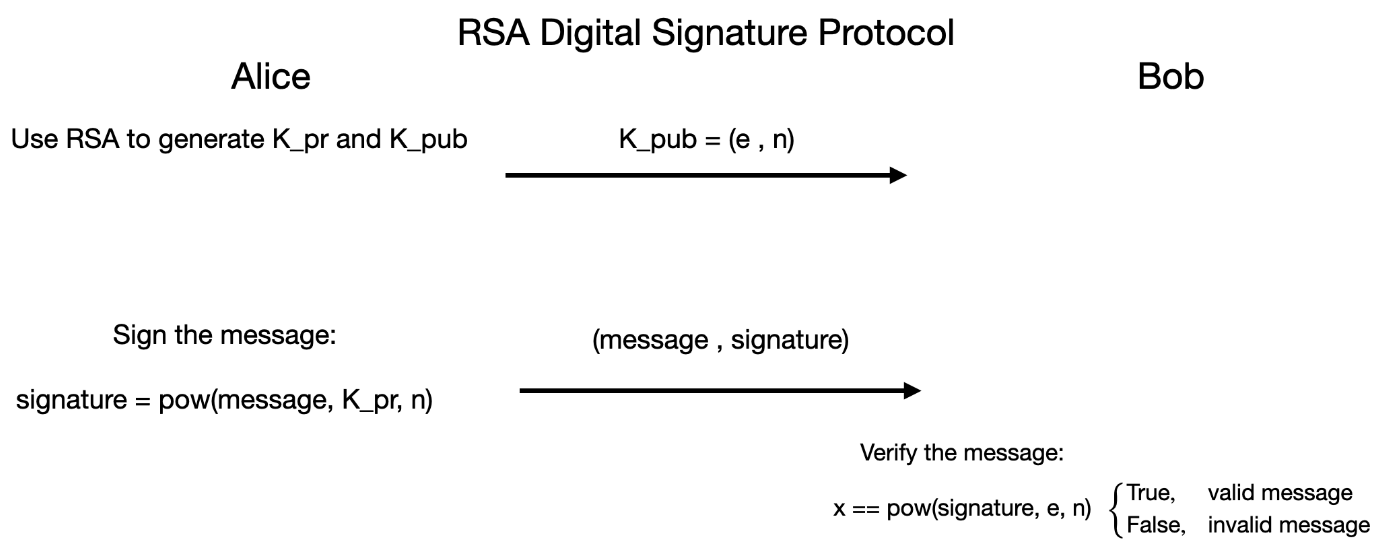
> ## Definition
>
> The objectives of a secure system are called security services

Some important security services

- **Confidentiality** : Information is kept secret from all but the authorized parties.
- **Message authentication** : the sender of the message is authentic

- **Message Integrity** : message has not been modified during transmission.
- **Non-repudiation** : The sender of a message cannot deny the creation of the message.
  - symmetric cryptography will never work here because each side has the same key, we need asymmetric.
- ....

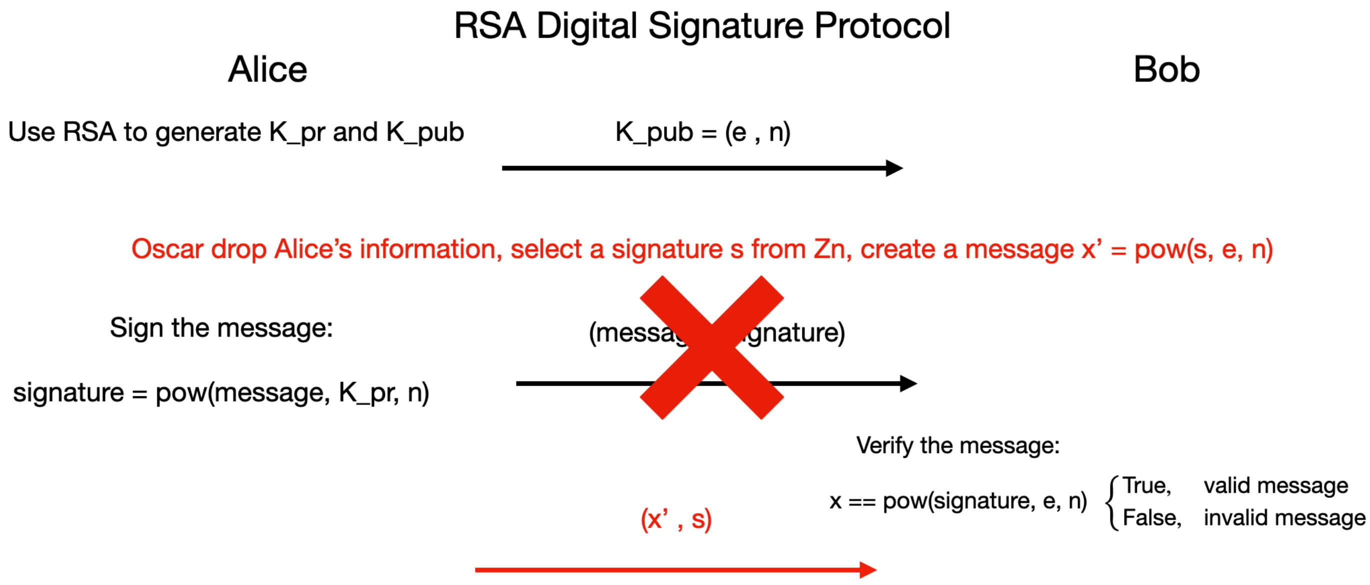
RSA Digital Signature



In order to accelerate the verification, people often use small e *like* $3, 2^{16}-1$ *rather than really really big numbers in practice*

Attack against RSA Digital Signature

Existential Forgery Attack

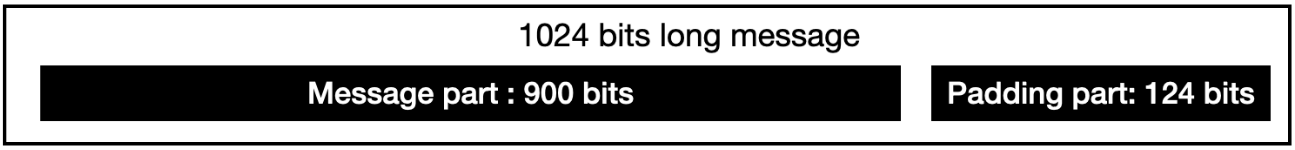


- This attack will work because when verify the message, Bob do the same calculation with Oscar,
  $x'' = s^e \mod n == s^e \mod n = x'$
- The limitation is that Oscar can only control the signature rather than the message

Solution:

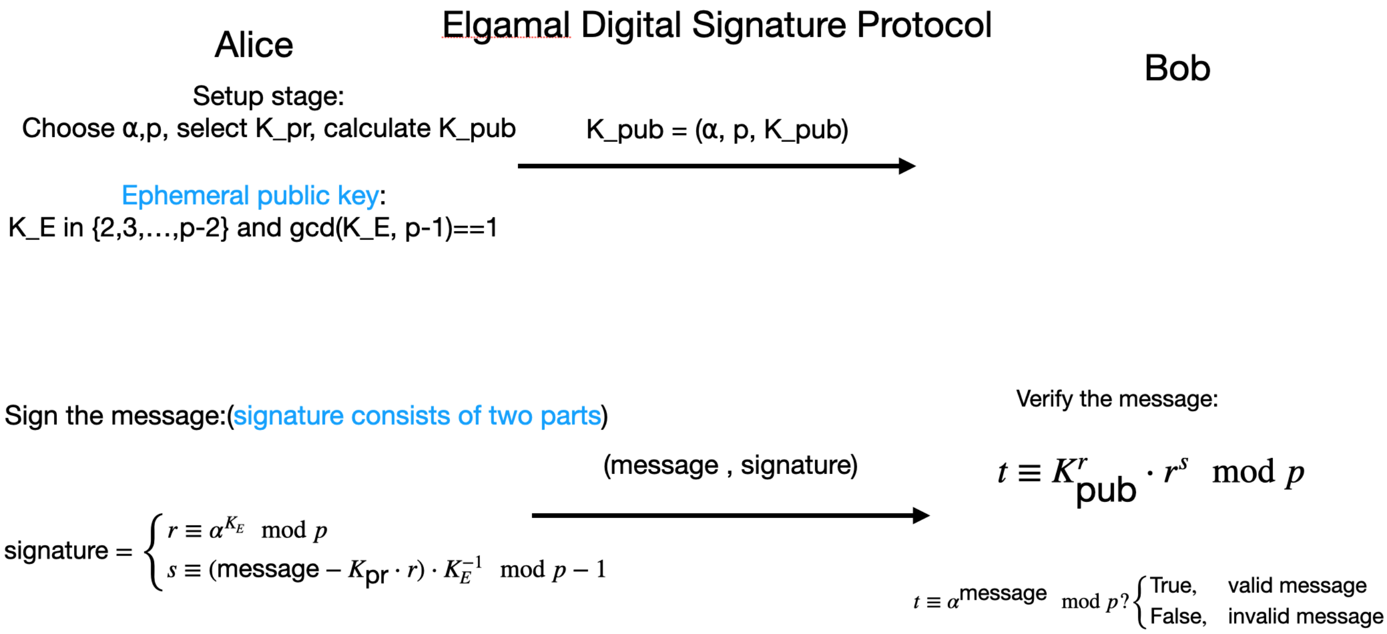
we impose a formatting rule for message which can be checked by the receiver.

For example, we stipulate message must be like the following:



- if the padding part is say to be all 1, then Oscar may need to try 2^{124} different s to generate a message with the correct format

Elgamal Digital Signature



Proof of Correctness

$$\because K_{\text{pub}} \equiv \alpha^{K_{\text{pr}}} \mod p$$

$$\therefore K_{\text{pub}}^r \equiv \alpha^{K_{\text{pr}} \cdot r} \mod p$$

$$\because r \equiv \alpha^{K_E} \mod p$$

$$\therefore r^s \equiv \alpha^{K_E \cdot s} \mod p$$

$$\because s \equiv (\text{message } - K_{\text{pr}} \cdot r) \cdot K_E^{-1} \mod p-1$$

> Fermat's little theorem:
>
> $$\forall m, a \in Z, \text{prime } p$$
>
> $$a^m \mod p \equiv a^{q(p-1)+r} \equiv (a^q)^{p-1} \cdot a^r \mod p$$
>
> $$\because \forall x \in Z, x^{p-1} \equiv 1 \mod p$$
>
> $$\therefore a^m \mod p \equiv a^r \mod p$$
>
> $$\because r \equiv m \mod p-1$$
>
> $$\therefore a^m \mod p \equiv a^{m \mod p-1} \mod p$$

$$\therefore \alpha^{K_E \cdot s} \equiv \alpha^{K_E \cdot s \mod p-1} \equiv \alpha^{\text{message } - K_{\text{pr}} \cdot r} \mod p$$

$$\therefore t \equiv \alpha^{K_{\text{pr}} \cdot r} \cdot \alpha^{\text{message } - K_{\text{pr}} \cdot r} \equiv \alpha^{\text{message}} \mod p$$

Remarks

- The signature is composed with two parts, that is to say, it has twice the bit length of message
- Elgamal DS algorithm is the basis for Digital Signature Algorithm(DSA)

Weaknesses of Elgamal Digital Signature

Reuse of the ephemeral key

compute ephemeral key is arduous, this is why people may reuse this, but it is a really bad idea

Assume that Alice use the same ephemeral key for both message m_1, m_2(ephemeral key should be unique to every message!!!)

Then Oscar can has the following:

$$\alpha, p, K_{\text{pub}}$$

$$m_1, (r_1, s_1)$$

$$m_2, (r_1, s_2)$$

Then he can calculate the K_{pr}

$$s_1 \equiv (m_1 - r_1 \cdot K_{\text{pr}})K_E^{-1} \mod p-1$$

$$s_2 \equiv (m_2 - r_1 \cdot K_{\text{pr}})K_E^{-1} \mod p-1$$

$$K_E \equiv \frac{m_1 - m_2}{s_1 - s_2} \mod p-1$$

go back to one of the two equation

$$K_{\mathrm{pr}} \equiv (m_1 - s_1 \cdot K_E) r_1^{-1} \mod p-1$$

Elgamal existential forgery attack

similar to the attack against RSA, forge the signature and then calculate the message, still can't control the content of message.

details