Class 6: Public key cryptography V1

Note Title

In the next two lectures, we learn several different, but related, techniques: (examples in green)

key exchange Diffie-Hellman	used to establish a shared secret, which can then be used as the key for symmetric key crypto
public key encryption RSA	used to encrypt a message with the recipient's public key (or padlock)
private key decryption RSA	used to decrypt a message with the recipient's private key
message authentication code (MAC) (any cryptographic hash function)	used to show the message has not been altered (i.e. guarantees integrity)
digital signature RSA	used to guarantee the identity of the sender of a message

(1) Mathematical preliminaries

(a) We need exponentiation e.g. $3^4 = 3 \times 3 \times 3 \times 3 = 81$

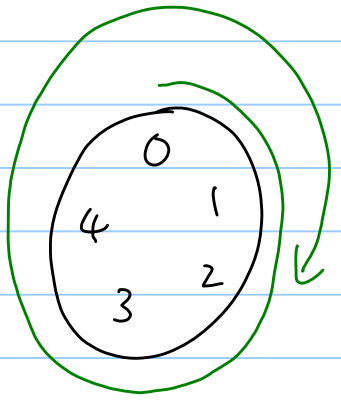
exponent

base

(b) We need clock or modulo arithmetic

e.g. $7 \equiv 2$ with clock size 5:

Also written $7 \equiv 2 \pmod 5$



also called a modulus.

More examples: $3^6 \equiv 9 \pmod{10}$

easy to work out, since we can throw away multiples of the clock size whenever we want:

$$3^6 \equiv 3 \times 3 \times 3 \times 3 \times 3 \times 3$$
$$\equiv 27 \times 27$$
take mod 10 here
$$\equiv 7 \times 7$$
$$\equiv 49$$
$$\equiv 9 \pmod{10}$$

② Diffie-Hellman key exchange

Alice and Bob want to set up a shared secret. Eve can see all their communication, but won't know the secret.

(a) With paints:

step 1 A and B choose private colors

e.g. A private col = red

B private col = green

step 2 Someone announces the public color

e.g. public col = yellow

step 3 A and B send public-private mixture

e.g. A sends red + yellow

B sends green + yellow

step 4 A and B combine received public-private mixture with private color, creating shared secret.

e.g. A receives green+yellow, adds red, gets g+y+r

B receives red+yellow, adds green, get r+y+g

Note: Eve cannot make r+y+g. She only has r+y and g+y, so ends up with too much yellow. We assume she can't unmix the yellow.

(b) With numbers:

Step 1 A and B choose private numbers
e.g. A's private number = 9
B's private number = 8

Step 2 Someone announces clock size and base
e.g. clock size = 11
base = 2

Step 3 A and B each mix their private and public info according to

$$\text{base}^{\text{private num}} \pmod{\text{clock size}}$$

and send the result

e.g. A's mixture is $2^9 \pmod{11} \equiv 6$
B's mixture is $2^8 \pmod{11} \equiv 3$

Step 4 A and B each mix the received info with their private numbers, according to

this is the shared secret! → $$\text{received mixture}^{\text{private num}} \pmod{\text{clock size}}$$

e.g. A calculates: $3^9 \pmod{11} \equiv 4$
B calculates: $6^8 \pmod{11} \equiv 4$

Not required: Why does it work?

Because $(2^9)^8 = 2^{9\times 8} = 2^{8\times 9} = (2^8)^9$

($(2^9)^8$: A's calculation; $(2^8)^9$: B's calculation)

This process is called Diffie-Hellman key exchange.

It is often used in practice e.g. when you visit a web address starting with "https".

Exercises:

a) Alice's private color is mauve. Public color is grey. What should Alice send to Bob?

b) As in (a). Alice receives brown from Bob. What is the shared secret?

c) Clock size is 17, base is 3. Alice's private number is 4. What should Alice send to Bob?

d) As in (c). Alice receives 2. What is the shared secret?

(3) Public key encryption and decryption with RSA

(a) Encryption

Suppose A wants to send to B.

Step 1 B chooses a private key and announces a public key. (They must satisfy certain properties that we don't study.)
The private key is a single number — an exponent.
The public key is 2 numbers: an exponent and a clock size
(also called a modulus)

e.g. B's private key: exponent 13
B's public key: modulus 34, exponent 5

Step 2 A encrypts the message m according to:

$$m^{(\text{B's public exponent})} \pmod{\text{B's modulus}}$$

e.g. $m = 6$, with above keys.
Then

$$\text{ciphertext} = 6^5 \pmod{34}$$
$$= 24$$

compute via Google or the accompanying handout

(b) Decryption: Suppose B wants to decrypt message from A. Then B uses same formula with private key instead of public:

$$\text{plaintext} = \text{ciphertext}^{(\text{B's private key})} \pmod{\text{B's modulus}}$$

e.g. If ciphertext is 24, plaintext $= 24^{13} \pmod{34}$

$= 6$

google doesn't work on this one. use the handout.

Exercises:

1. Keys for A and B as on handout.
   B sends plaintext 19 to A. What is ciphertext?

2. A receives ciphertext 3 from B. What is plaintext?

Optional: Why does it work? Because the private exponent is a specially-selected value that exactly reverses the effect of the public exponent.
You can't calculate the private key from the public key efficiently, unless you know some extra information. The extra info is the prime factorization of the modulus minus one. Crazy!! But, it works.

Thus RSA depends on the hardness of factorization — believed to be difficult, but not proven!!