CAS CS 357

InClass Note 14

1. RSA
    a. Algorithm that one of the things we can do is digital encryption
    b. We can also create keys and signatures with this
    c. RSA can be applied to more difficult encryption
2. RSA algorithm
    a. Choose two large prime numbers p and q at random and multiply them together
    b. Important that p does not equal q and the numbers p and q are not secret
    c. N = p * q (N is the RSA modulus) and N is public
    d. We need to realize that "factoring is hard" for N: given N it is hard to find the prime factors p and q
    e. If p and q are larger enough, the computation of N will equal exponential time
    f. There is an algorithm to factor N but requires the usage of quantum computer (but the number of bits have to be small → choose large p and q so that N is large enough)
3. RSA algorithm
    a. Choose an number e such that 1 < e < lowest common multiple (p-1, q-1) that we are going to include defining public key and gcd(e, (p-1)*(q-1)) = 1
    b. Therefore, e and (p-1)*(q-1) should not share a factor in common (prime relative to each other)

    c. e, in most cases, is defined as $2^{16} + 1$ (this number generally works and matches the conditions explained above as long as p and q does not equal e → however, $2^{16} +1$ is too small for p and q so the two variables will not equal e in most cases)
    d. e is known to the public
    e. Public key = (e, N)
4. RSA algorithm
    a. Given that the message is m, there is a value d such that $(m^e)^d = m \bmod N$
    b. There is an algorithm that if you know the p and q, we can compute d
    c. $d = e^{-1} \bmod (p-1)*(q-1)$
    d. The notion is that p and q are kept secret and therefore, calculating d will be difficult
    e. In other words, $ed = 1 \bmod (p-1)*(q-1)$
    f. ex) $(3+4) \bmod 3 = 7 \bmod 3$

       $= 1 \bmod 3$
    g. Private key → (d,N)
    h. If we know d, we can find (p,q)
    i. If we know (p,q), we can find d
5. RSA assumption
    a. For all integers m, and given (e,N), it is hard to find d such that $(m^e)^d = m \bmod N$
    b. If you know the RSA public key, you can compute RSA secret key
6. How to sign with DSA

a. $h = Hash(m) \rightarrow$ turns the message into a large integer (in case the message is too short such as the value 1) ex) SHA256

b. $Sign_{SK}(m)$:

    h = hash (m)

    o = h^d mod N

    output o

c. $Ver_{PK}(m,o)$:

    h = hash(m)

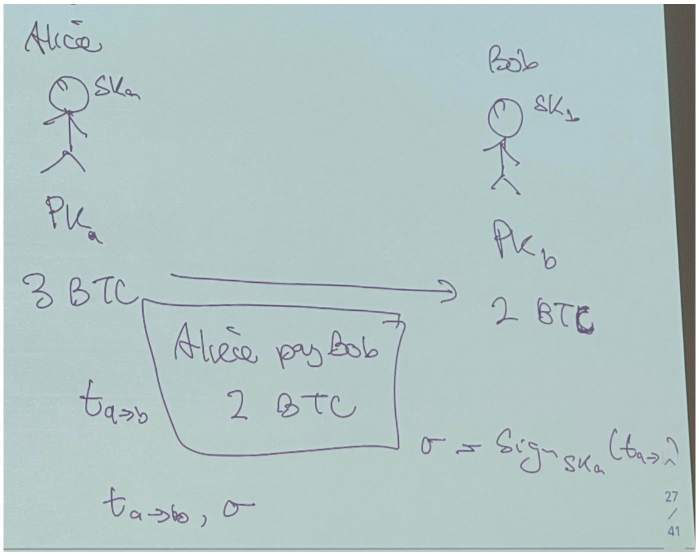
    if o^e = h mod N:

        output 1

    else:

        output 0

7. Why it works?
    a. o^e = (h^d)^e = h mod N
8. Concept
    a. One direction is difficult, one direction is easy
    b. Difficult: o = [H(m)]^d mod N given m and (N,e)
    c. Easy: o = [H(m)]^d mod N given m and (N,d)
    d. Difficult to find p and q from N
    e. Easy to find N from p and q
9. Application of digital signatures: Digital Ledger → Bitcoin
    a. Transaction 1: alice pays Bob (signed by secret key of alice)
    b. Transaction 2: alice pays Charlie (signed by secret key of alice)

c. 

d. The notion is that if Alice keeps her key secret and Alice's secret key is not stolen, the transaction will work

e. Ledger → multiple people can have different versions of ledger (one ledger might say A pays B 2 BTC, A pays C 100 BTC) → bitcoin has found a method to find a ledger that people agree upon (blockchain algorithms)

10. What if the secret key is stolen?

    a. If secret key is stolen, they can take all the BTC (money) inside and send it to adversaries

    b. Secret key should be preserved as money