**IS ZA Public Key Crypto**

Verschlüsseln => public key

Entschlüsseln => private key

RSA

Inverses Element y von x ist => x \* y mod n = 1

Inverses Element finden => Mit Euclids Algorithmus möglich. => wenn x und n gegebene

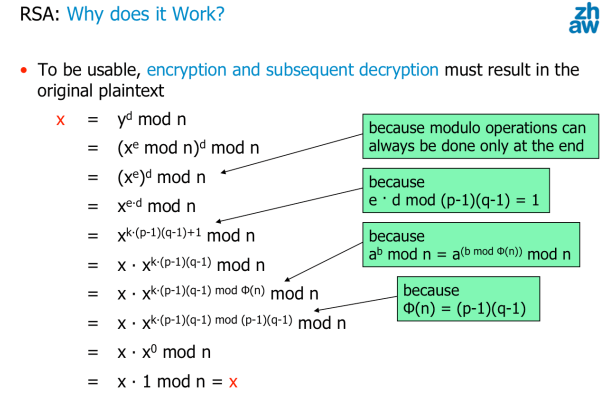
RSA Key Pair generieren:

* Wähle zwei grosse zufällige Primzahlen p und q
* n = p \* q
* Wähle einen zufälligen integer e => e < (p-1)\*(q-1)
  + e und (p-1)(q-1) sollten relativ prim sein
    - d.h sie sollten keine gemeinsamen Primfaktoren haben.
* Eindeutige inverse d von e berechnen
  + e \* d mod (p-1)(q-1) = 1
  + Euklids Algorithmus => effizienter

RSA operates on numbers => binary plaintext must first be mapped to a number before it can be encrypted

Encryption of a plaintext x => y = x^e mod n

Decryption of ciphertext y => x = y^d mod n



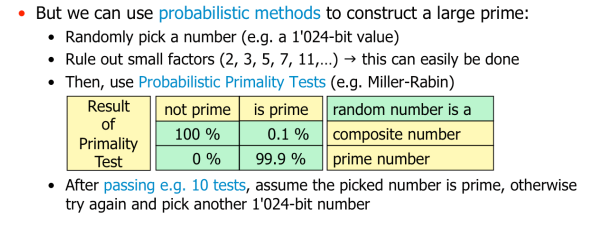
RSA secure:

* To break RSA: an attacker must learn d from the public key n and e
* Attacker doesn’t know p and q
* Factoring large numbers composed of two large primes is a mathematically hard problem, for which no efficient algorithm is known => the security of RSA relies on this assumption

Key size => determined by the modulus size

Complexity to factor large numbers => O(2^n)

Finding large primes :

* Using probabilistic methods
* 

One should never directly encrypt the plaintext with RSA, but use suitable standard encryption formats

Performance of RSA

* RSA 1000 times slower than secret key ciphers
* RSA encryption much faster than RSA decryption
  + e is often chosen to be 65537 (2^16 + 1)

Diffie-Hellman Key Exchange

