RSA: Public Key Crypto

- Ex: Alice creates a public RSA key
  - She picks 2 primes: p,q. $p = 7, q = 7.$
    $n = pq = 7 * 11 = 77$
    $e,$
    Need $gcd(e, (p - 1)(q - 1)) = 1$
    $e = 17$
  - Alice's Public Key in $(n, e) = (77, 17)$. Everyone knows these numbers.
  - Bob wants to send the message $m = 9$ to Alice.
  - Bob computes $E(9) \equiv 9^{17}(mod\ 77)$
    $17 = 16 + 1$
    $9^1 \equiv 9(mod\ 77)$
    $9^2 \equiv 81 \equiv 4(mod\ 77)$
    $9^4 \equiv 4^2 \equiv 16(mod\ 77)$
    $9^8 \equiv 16^2 \equiv 256 \equiv 25(mod\ 77)$
    $9^{16} \equiv 25^2 \equiv 625 \equiv 9(mod\ 77)$
    $9^{17} \equiv (9^{16})(9^1) \equiv 9 \times 9 \equiv 81$
  - To decrypt, Alice has to compute $d \equiv e^{-1}(mod\ \phi(n))$ Note:
    $\phi(n) = (p - 1)(q -)$
    $d \equiv 17^{-1}(mod\ (7 - 1)(11 - 1))$ Note: $(7 - 1)(11 - 1) = 60$
    Euclids' Algorithm
    $gcd(17, 60)$
    $60 = 3(17) + 9$
    $17 = 1(9) + 8$
    $9 = 1(8) + 1$
    $1 = 9 - 1(8)$
    $= 9 - 1(17 - 1(9)) = 2(9) - 1(17)$
    $= 2(60 - 3(17)) - 1(17)$
    $1 = 2(60) - 7(17)$
- RSA is secure as long as n is too big to factor.
- What if there was an easier way to compute $\phi(n)$? (A way that didn't require factoring?)
- Computing $\phi(n)$ and Factoring n are equally difficult.
- Suppose you have a fast way to compute $\phi(n)$:
  $\phi(n) = \phi(pq) = (p - 1)(q - 1)$
- Compute
  $V = n - \phi(n) + 1$
  $= pq - (p - 1)(q - 1) + 1$

$= pq - (pq - p - q + 1) + 1$
$= p + q$
p and q are the roots of $x^2 - vx + n = (x - q)(x - p)$
$p, q = \frac{v \pm \sqrt{v^2 = 4n}}{2}$

- Suppose $n = 27,906,817$
  $\phi(n) = 27,894,996$
  Use this info to find p, q.
  - $p = 8563$
  - $q = 3259$
- Alice needs two really large primes p and q, essentially p and q need to be "brand new", prime numbers never used before.
- There are much faster ways to check if a number is prime than to factor it.
- Fermat Primality test "compositeness".
- Fermat's Little theorem: If p is prime and $a \not\equiv 0(mod\ )$ then $a^{p-1} \equiv 1(mod\ p)$.
  - Contrapositive: if $a^{p-1} \not\equiv 1(mod\ p)$ then p is not prime.
- Steps to Fermat's Primality test:
  - We want to test if n is prime.
    1) Pick a randomly $1 < a < n - 1$
    2) Compute $a^{n-1}(mod\ n)$
  - If we don't get 1, n is composite.
  - If we do get 1, n is "probably prime".
- Ex: $n = 5$, $a = 2$. Compute $2^{n-1} \equiv 2^4 \equiv 16 \equiv 1(mod\ 5)$
  - Fermat says 5 is "probably prime".
- Test if $n = 33$ is prime.
  - Pick an a
    - $a = 5$
      $5^{33-1} \equiv 5^{32}$
    - Repeated squaring:
      $5^2 \equiv 25(mod\ 33)$
      $5^4 \equiv 25^2 \equiv (-8)^2 \equiv 64 \equiv 31(mod\ 33)$
      $5^8 \equiv 31^2 \equiv (-2)^2 \equiv 4(mod\ 33)$
      $5^{16} \equiv 4^2 \equiv 16(mod\ 33)$
      $5^{32} \equiv 16^2 \equiv 256 \equiv 25\ mod\ 33$. Note: Not 1 $(mod\ 33)$
  - 33 is **not prime**.

- Test $n = 21$ using Fermat's test and $a = 13$. Compute $13^{20} (mod\, 21)$

  $13^2 \equiv (-\ 8)^2 \equiv 64\ (mod\, 21) \equiv 1\ (mod\, 21)$

  $13^4 \equiv 1^2 \equiv 1\ (mod\, 21)$

  $13^8 \equiv 1$

  $13^{16} \equiv 1\ (mod\, 21)$

  $13^{20} \equiv 13^{16} \times 13^4 \equiv (1)(1) \equiv 1\ (mod\, 21)$
  - Fermat says $n = 21$ is "probably prime".
  - Keep trying more values of a to see if they also give probably prime. Try $a = 2$,

    $2^{20} \equiv (2^{16})(2^4)(mod\, 21)$

    $2^2 \equiv 4\ (mod\, 21)$

    $2^4 \equiv 4^2 \equiv 16\ (mod\, 21)$

    $2^8 \equiv 16^2 \equiv (-\ 5)^2 \equiv 25 \equiv 4\ (mod\, 21)$

    $2^{16} \equiv 4^2 \equiv 16\ (mod\, 21)$

    $2^{20} \equiv 2^{16} \times 2^4 \equiv 16 \times 16 \equiv 4\ (mod\, 21)$. Note: Not 1. 21 is composite.
- If n is composite but $a^{n-1} \equiv 1\ (mod\, n)$, we call n a base-a **pseudoprime**.
  - 21 is a base-13 pseudoprime.
- There exists composite numbers v^n which are pseudoprimes to ever base coprime to n. These are called carmichael numbers. The smallest example is

  $n = 561 = 3 \times 11 \times 17$.