# RSA

- By Rivest, Shamir & Adleman of MIT in 1977
- Best known & widely used public-key scheme
- Uses large integers (e.g., 1024 bits)
- Security due to cost of factoring large numbers, and difficulty of computing discrete logarithm.

# Background

- Totient function $\phi(n)$
  - Number of positive integers less than *n and relatively prime to n*
  - *Relatively prime means with no factors in common with n*
- Example: $\phi(10) = 4$
  - 1, 3, 7, 9 are relatively prime to 10
- If $n=pq$, $\phi(n) = (p-1)(q-1)$

- Euler's Theorem: $a^{\phi(n)} \bmod n = 1$ where $\gcd(a,n)=1$
  - Example: $n=10$, $a=3$.

# RSA Key Setup

- Each user generates a public/private key pair by:
- Selecting two large primes at random $p, q$
- Computing their system modulus $n=p*q$

  $\phi(n)=(p-1)(q-1)$
- Selecting at random the encryption key $e$

  where $1<e<\phi(n)$, $\gcd(e,\phi(n))=1$
- Solve following equation to find decryption key $d$

  $e*d=1 \bmod \phi(n)$ and $0\le d\le n$
- Publish their public encryption key: PU={e,n}
- Keep secret private decryption key: PR={d,n}

p and q are kept secrete.

# RSA Use

- To encrypt a message M, the sender:
  - obtains **public key** of recipient $PU=\{e,n\}$
  - computes: $C = M^e \bmod n$, where $0 \leq M < n$
- To decrypt the ciphertext C the owner:
  - uses their private key $PR=\{d,n\}$
  - computes: $M = C^d \bmod n$
- Note that the message M must be smaller than the modulus n (block if needed)

# Why RSA Works

- RSA we have:
  - $n=p.q$
  - $\varnothing(n)=(p-1)(q-1)$
  - carefully chose $e$ & $d$ to be inverses mod $\varnothing(n)$

    $e*d=1 \bmod \phi(n)$ and $0 \le d \le n$
  - hence $e*d=1+k*\varnothing(n)$ for some $k$
- Hence :

$$
\begin{aligned}
C^d \bmod n &= (M^e)^d \bmod n \\
&= M^{1+k.\varnothing(n)} \bmod n \\
&= M*(M^{\varnothing(n)})^k \bmod n \\
&= M*(1)^k \bmod n \quad \text{(Euler's Theorem, Fermat's little Theorem)} \\
&= M \bmod n
\end{aligned}
$$

# RSA Example - Key Setup

1. Select primes: $p$=17 & $q$=11, and keep secret.
2. Compute $n = pq$ =17 x 11=187
3. Compute $ø(n)=(p-1)(q-1)$=16 x 10=160
4. Select e: gcd(e,160)=1; choose $e$=7
5. Determine d: $d*e$=1 mod 160 and $d$ < 160
   Value is d=23 since 23x7=161= 1x160+1
6. Publish public key PU={7,187}
7. Keep secret private key PR={23,187}

# RSA Example - En/Decryption

- sample RSA encryption/decryption is:
- given message `M = 88`
- encryption:

  $C = 88^7 \bmod 187 = 11$

- decryption:

  $M = 11^{23} \bmod 187 = 88$

# Basic Algorithms in RSA

1. Encryption and Decryption
   Exponentiation

2. Key Setup
   How to find d given e?

# “Efficient” powering to compute a^n

power($a, n$)
set $r \leftarrow 1$
if $n = 0$, return $r$
while $n > 1$ do
    if $n$ is odd, set $r \leftarrow r * a, n \leftarrow n - 1$
    set $n \leftarrow n/2, a \leftarrow a * a$
return $a * r$

Loop invariant: power($a, n$)$r$ is constant.

# Euclidean Algorithm

- An efficient way to find the GCD(A,B)
- Uses theorem that:(A>B)
  - GCD(A,B) = GCD(B, A mod B)
- Euclidean Algorithm to compute GCD(A,B) is:

```
Euclid(A,B)
        if (R=0) then return B;
        else return Euclid(B, A mod B);
```

GCD(16, 12)=GCD(12, 4)=4

# Euclidean Algorithm

Find the GCD(12, 33).

| Q | A | B | R |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |

# Euclidean Algorithm

Find the GCD(12, 33).

| Q | A | B | R |
|---|---|---|---|
| 2 | 33 | 12 | 9 |
| | | | |
| | | | |
| | | | |

$$\begin{array}{r|l} & 2 \\ \hline 12 & 33 \\ & 24 \\ \hline & 9 \end{array}$$

# Euclidean Algorithm

Find the GCD(12, 33).

| Q | A | B | R |
|---|---|---|---|
| 2 | 33 | 12 | 9 |
| | | | |
| | | | |
| | | | |

# Euclidean Algorithm

Find the GCD(12, 33).

| Q | A | B | R |
|---|---|---|---|
| 2 | 33 | 12 | 9 |
| 1 | 12 | 9 | 3 |
| | | | |
| | | | |

1
9 | 12
9
3

# Euclidean Algorithm

Find the GCD(12, 33).

| Q | A | B | R |
|---|---|---|---|
| 2 | 33 | 12 | 9 |
| 1 | 12 | 9 | 3 |
| 3 | 9 | 3 | $\boxed{0}$ |
| | | | |

$$\begin{array}{r|l} & 3 \\ \hline 3 & 9 \\ & 9 \\ \hline & 0 \end{array}$$

# Euclidean Algorithm Exercise

- gcd(309,171)

# Extended Euclidean Algorithm

- Extended Euclidean (a,b)

- This function calculates not only GCD but also x & y:
- ax + by = gcd(a, b), where x and y will have opposite signs.
- Follow sequence of divisions for GCD but assume at each step i, can find x &y:
- $$r = ax + by$$
- In the end find GCD value and also x & y
- If GCD(a,b)=1, y and b are inverses mod a

# Find Private Key in RSA

How to calculate d s.t. `e*d=1 mod` $\phi$ `(n)` using Extended Euclidean algorithm ?

Given $\phi(n)$ and e (public key) in the RSA, we run

Extended Euclidean($\phi(n)$, e)

Since $\phi(n)$ and e are co-prime, this function will return x and y s.t., $\phi(n)\, x + e\, y = \gcd(\phi(n), e) = 1$

On both sides, mod $\phi(n)$, we get $0 + e\, y \bmod \phi(n) = 1$

Then y is the inverse of e mod $\phi(n)$.

We set d = y as the private key.

# Extended Euclidean Algorithm Example (get multiplicative inverse modulo n)

## Multiplicative Inverse using EEA

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Extended Euclidean Algorithm Example (get multiplicative inverse modulo n)

## Multiplicative Inverse using EEA

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

**Points to Ponder**

A > B


$T_1 = 0$ and $T_2 = 1$

$T = T_1 - T_2 \times Q$

$T_2$ is the M.I.

# Multiplicative Inverse using EEA

Example 1: What is the multiplicative inverse of 3 mod 5.

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 2 | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

1

3 | 5

3

2

# Multiplicative Inverse using EEA

Example 1: What is the multiplicative inverse of 3 mod 5.

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 2 | 0 | 1 | -1 |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

$T_1 = 0$ and $T_2 = 1$

$T = T_1 - T_2 \times Q$

$T = 0 - 1 \times 1$

$T = 0 - 1$

$T = -1$

# Multiplicative Inverse using EEA

## Example 1: What is the multiplicative inverse of 3 mod 5.

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 2 | 0 | 1 | -1 |
| | 3 | 2 | | 1 | -1 | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

1
2 | 3
2
1

# Multiplicative Inverse using EEA

Example 1: What is the multiplicative inverse of 3 mod 5.

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 2 | 0 | 1 | -1 |
| 1 | 3 | 2 | 1 | 1 | -1 | 2 |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

$T = T_1 - T_2 \times Q$

$T = 1 - (-1) \times 1$

$T = 1 - (-1)$

$T = 2$

# Multiplicative Inverse using EEA

Example 1: What is the multiplicative inverse of 3 mod 5.

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 2 | 0 | 1 | -1 |
| 1 | 3 | 2 | 1 | 1 | -1 | 2 |
| 2 | 2 | 1 | 0 | -1 | 2 | -5 |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Multiplicative Inverse using EEA

Example 1: What is the multiplicative inverse of 3 mod 5.

| Q | A | B | R | $T_1$ | $T_2$ | T |
|---|---|---|---|---|---|---|
| 1 | 5 | 3 | 2 | 0 | 1 | -1 |
| 1 | 3 | 2 | 1 | 1 | -1 | 2 |
| 2 | 2 | 1 | $\boxed{0}$ | -1 | $\boxed{2}$ | -5 |
| | | | | | | |
| | | | | | | |
| | | | | | | |

# Extended Euclidean Algorithm Implementation

```c
// C function for extended Euclidean Algorithm
int gcdExtended(int a, int b, int *x, int *y)
{
    // Base Case
    if (a == 0)
    {
        *x = 0;
        *y = 1;
        return b;
    }

    int x1, y1; // To store results of recursive call
    int gcd = gcdExtended(b%a, a, &x1, &y1);

    // Update x and y using results of recursive
    // call
    *x = y1 - (b/a) * x1;
    *y = x1;

    return gcd;
}

// Driver Program
int main()
{
    int x, y;
    int a = 35, b = 15;
    int g = gcdExtended(a, b, &x, &y);
    printf("gcd(%d, %d) = %d", a, b, g);
    return 0;
}
```

# RSA Example – Find d

1. Select primes: $p$=17 & $q$=11, and keep secret.
2. Compute $n = pq$ =17 x 11=187
3. Compute $\varnothing(n)=(p-1)(q-1)$=16 x 10=160
4. Select e: gcd(e,160)=1; **choose** $e$=7
5. Determine d: $d*e$=1 mod 160 **and** $d$ < 160