Secure Software Design

Andey Robins

Spring 23 - Supplemental 1

An In-depth Look at RSA

Outline

- Foundations
  - Groups
- RSA Function
- Key Generation
- Takeaways

Serious Cryptography

A Practical Introduction to Modern Encryption

Jean-Philippe Aumasson

Foreword by Matthew D. Green

Groups

What's in a Group

A set with elements obeying these axioms:

- **Closure:** For any two x and y in a group, $x * y$ is also in the group
- **Associativity:** For any x, y, z in a group, $(x * y) * z = x * (y * z)$
- **Identity Existence:** There's some element i such that $i * x = x * i = x$
- **Inverse Existence:** For any x in a group, there is some y such that $x * y = y * x = i$

Other Properties of Groups

Outside of the axioms, these are useful features of many groups.

- **Commutative:** $x * y = y * x$
- **Cyclic:** There is some element g such that *gˆ1, gˆ2, gˆ3, etc.* span all distinct elements.
- **Generator:** If the group is cyclic, the element g is called the generator.

Group Proof Exercise

As an excercise to the student, show this is true. **Z** is the integers and the subscript 4 indicates our group is the integers modulo 4.

$$\mathbb{Z}_4^* = \{1, 3\}$$

Two is not coprime with 4, which is the intuition behind why it is not within the group. Further, what is the generator for this group?

RSA Function

RSA Encoding

RSA encodes a message as a single, positive integer between 1 and n - 1 where n is a large number called the *modulus*. More specifically, it works on all the numbers less than n which are coprime with n (no common prime factors). These numbers form the group:

$$\mathbb{Z}_n^*$$

Encryption with RSA

Let x be the number to be encrypted which belongs to $\mathbb{Z}_n^*$.

Then, RSA encrypts this number as $y = x^e \bmod n$

Or the encrypted message multiplied by itself e times modulus n. e and n thus make up the public key.

Decryption with RSA

Let us denote the "private key" as d.

$$y^d \bmod n = (x^e)^d \bmod n = x^{ed} \bmod n = x$$

Since we select d to be the inverse of e, $ed = 1$.

Euler's Totient Function

This function gives the number of elements coprime with n. If n is the product of prime numbers:

$$\phi(n) = (p_1 - 1) \times (p_2 - 1) \times ... \times (p_m - 1)$$

Since RSA operates with large prime numbers such that $n = pq$,

$$\phi(n) = (p - 1)(q - 1) = |\mathbb{Z}_n^*|$$

This is important, because our choices for *ed* $= 1$ is all mod *phi(n)*.

Therefore, if you can compute *phi(n)*, you can break any RSA encryption since d can be derived from *phi(n)* and *e*.

So where do the *p* and *q* that define *phi(n)* come from?

Key Generation

Generating Keys for RSA

1. Pick two large prime numbers p and q
2. Calculate n as pq
3. Calculate *phi(n)* as *(p-1)(q-1)*
4. Pick a random prime number less than *phi* to be e
5. Calculate the value d from *phi* and e
6. Share n and e as the public key

Assumptions

1. Factoring is a hard problem
2. Calculating e -th roots is a hard problem
   - This is an assertion that appears to derive from the study of rings, something I mention but have no further ability to comment on.

"*These seem closely connected, though we don't know for sure whether they are equivalent.*" - Jean-Phillipe Aumasson

Takeaways

1. If using RSA, ensure your generation of p and q are done in secure ways since the entire system relies on their secrecy.
2. Pick large numbers for p and q (standard is to look for numbers which yield an n with 4096 bits).
3. p and q should be unrelated, random primes of similar size that are not too close in value.