RSA[1] is one of the most widely used encryption schemes in the world. It is a *public-key cipher*: anyone can encrypt messages using the public key, but knowledge of the private key is required for decryption. Knowing the public key does not help crack the private key.

Public-key cryptography makes the secure Internet possible. Before public-key cryptography, keys had to be carefully exchanged between people who wanted to communicate, often by non-electronic means. Now RSA is routinely used to exchange keys without allowing anyone snooping on the channel to understand what has been communicated.

RSA security is based on the fact that there is no known efficient algorithm for factoring large numbers. Deriving a private key from the corresponding public key appears to be as hard as factoring.

Public key cryptography

In public key cryptography, messages are encoded using a *public key* E and decoded using a *private key* D. Any agent A who wants to receive secret messages obtains a key pair (E_A, D_A). The agent A keeps the private key D_A secret and publishes the public key E_A, say on a webpage. Now if agent B wants to send a secret message m to A, B encodes the plaintext message m using A's public key to get an encoded ciphertext $E_A(m)$. The ciphertext is sent to the recipient, who can decode it using the private key to get $D_A(E_A(m)) = m$. For this to work, it must be the case that D_A is a left inverse of E_A, that is, $D_A \circ E_A$ is the identity function. For it to be secure, it must be the case that neither the plaintext message nor the private key can be derived from knowledge of the ciphertext or the public key.

Public key cryptography can also be used to digitally sign a message so that the recipient can be sure that it came from the agent who purported to send it. If A wants to send a signed secret message m to B, then A can send the pair $(E_B(m), D_A(E_B(m)))$ to B. When B receives the message, they can decode it by applying their private key D_B to the first component to get $D_B(E_B(m)) = m$. Then they can authenticate by applying A's public key E_A to the second component to get $E_A(D_A(E_B(m)))) = E_B(m)$ and comparing it with the first component. If they match, then B is assured the message came from A, because A is the only agent who knows D_A. For this to work, it must be the case that D_A is also a right inverse of E_A, that is, $E_A \circ D_A$ is the identity function.

[1]Named for its inventors, Ronald Rivest, Adi Shamir, and Leonard Adelman.

The RSA algorithm

In RSA, the conversion between plaintext and ciphertext is done numerically on the binary representation of the text and involves modular arithmetic on very large numbers, typically much larger than would fit in a 64-bit integer. The Chinese remainder theorem can be used for this purpose.

Key generation

1. Choose two large distinct random prime numbers p and q (we will discuss in another lecture later on how to do this). These must be kept secret. The larger p and q are, the stronger the encryption will be. With current technology, 1024 bits gives a decent level of security.

2. Compute their product $n = pq$. This will be the modulus used for encryption. All arithmetic will be in the ring $\mathbb{Z}_n$ of integers modulo n (often called $\mathbb{Z}/n\mathbb{Z}$).

3. Compute $\varphi(n)$, the *totient* of n. This is the number of positive integers less than n that are *relatively prime* to n, that is, have no prime factor in common with n. If the prime factorization of n is $p_1^{m_1} \cdots p_k^{m_k}$, then $\varphi(n) = p_1^{m_1-1}(p_1 - 1) \cdots p_k^{m_k-1}(p_k - 1)$, so for a product of primes $n = pq$, it is just $(p - 1)(q - 1)$. This is easy to compute from knowledge of p and q, but it is not known how to compute it efficiently from just knowledge of n, and it is conjectured to be intractible.

4. Choose an integer e such that $1 < e < \varphi(n)$ and e is relatively prime to $\varphi(n)$. One way to do this is to take e to be a prime greater than $\varphi(n)/4$. Then e cannot divide $\varphi(n)$, because 4 divides $\varphi(n)$ so e would have to divide $\varphi(n)/4$. Since e is a prime not dividing $\varphi(n)$, it is relatively prime to $\varphi(n)$.

5. Let d be the multiplicative inverse of e modulo $\varphi(n)$. The value of d can be computed from e and $\varphi(n)$ using the extended Euclidean algorithm for integers, which gives integers d and t such that $de + t\varphi(n) = 1$. Then d and e are multiplicative inverses mod $\varphi(n)$, since $de = 1 - t\varphi(n) \equiv 1 \bmod \varphi(n)$.

The public key is the pair (e, n) and the private key is the pair (d, n). To allow people to encode messages to you, you can advertise your public key, say on your webpage, keeping your private key secret.

Encryption

A plaintext message $m \in \mathbb{Z}_n$ is encrypted as ciphertext via the formula: $m \mapsto m^e \bmod n$. Note that encryption can be done using only the publicly known n and e. Exponentiation modulo n can be done recursively:

$$m^{2k} = (m^k)^2 \qquad\qquad m^{2k+1} = m(m^k)^2,$$

reducing modulo n as necessary. The number of arithmetic operations needed to compute $m^e \bmod n$ is $O(\log e)$.

Decryption

A ciphertext c is decrypted to plaintext via the formula: $c \mapsto c^d \bmod n$. Note that this requires knowledge of the private key.

If we encrypt and then decrypt a plaintext message m, we obtain a new plaintext message $(m^e \bmod n)^d \bmod n = m^{ed} \bmod n$. For this to give the original message back, we must have $m^{ed} \equiv m \bmod n$. We will show below that this is true, provided m is relatively prime to n.

Example

Let's take $p = 5$, $q = 11$, $e = 3$, and $m = 8$. Then $n = pq = 55$, $\varphi(n) = 4 \cdot 10 = 40$, and $e = 3$ which is relatively prime to 40. Its inverse mod 40 is 27, as $3 \cdot 27 = 81 \equiv 1 \bmod 40$. Then $(e, n) = (3, 55)$ and $(d, n) = (27, 55)$ are our public and private keys.

Now to encode the plaintext 8, which is relatively prime to $n = 55$, we compute 8^3 mod 55, which gives the ciphertext 17. To decrypt, we compute 17^{27} mod 55, which gives—well, I guess you'll just have to take my word for it—8. It works!

Correctness of RSA

A number $k \in \mathbb{Z}_n$ is relatively prime to n iff it has a multiplicative inverse mod n. This is because of the extended Euclidean algorithm, which gives integers s and t such that

$$sk + tn = \gcd(k, n).$$

If k and n are relatively prime, then $\gcd(k, n) = 1$, so $sk + tn = 1$, so $sk = 1 - tn \equiv 1 \bmod n$, which says that s and k are inverses mod n. Conversely, if k has a multiplicative inverse s mod n, then $sk = 1 + tn$ for some t, so $sk - tn = 1$, and $\gcd(k, n)$ divides k and n, so it must also divide $sk - tn = 1$, so it must be 1.

The set of invertible elements of $\mathbb{Z}_n$ is

$$\mathbb{Z}_n^* = \{x \in \mathbb{Z}_n \mid \gcd(x, n) = 1\} = \{x \in \mathbb{Z}_n \mid x \text{ has a multiplicative inverse in } \mathbb{Z}_n\}.$$

This is a multiplicative subgroup of $\mathbb{Z}_n$ of size $|\mathbb{Z}_n^*| = \varphi(n)$.

The correctness of RSA depends on *Euler's theorem*:

Theorem (Euler's theorem)**.** *If m and n are relatively prime, then $m^{\varphi(n)} \equiv 1 \bmod n$.*

Proof. In other words, if $m \in \mathbb{Z}_n^*$, then $m^{\varphi(n)} = 1$, where arithmetic is the ring $\mathbb{Z}_n$. This is true because m generates a cyclic subgroup $\{1, m, m^2, \ldots, m^{k-1}\}$ of $\mathbb{Z}_n^*$ consisting of powers of m, where k is the *order* of m, the smallest number such that $m^k = 1$. Then k must divide $\varphi(n)$; indeed, for any subgroup H of any finite group G, $|H|$ divides $|G|$. But then $m^{\varphi(n)} = (m^k)^{\varphi(n)/k} = 1^{\varphi(n)/k} = 1$. $\square$

Using Euler's theorem, we can now prove the correctness of RSA. We wish to show that if m is relatively prime to n, then $m^{ed} \equiv 1 \bmod n$. But we know that $ed = 1 \bmod \varphi(n)$, so $ed = 1 + k\varphi(n)$ for some k. Then modulo n,

$$m^{ed} = m^{1+k\varphi(n)} = m(m^{\varphi(n)})^k = m \cdot 1^k = m.$$

If the message m is not relatively prime to n, then this will not work, but in that case we can pad the message with randomly chosen garbage. The chance that the resulting padded message is not relatively prime to n is negligibly small, $\frac{1}{p} + \frac{1}{q} - \frac{1}{pq}$ to be precise.