CISC 2100/2110 — Discrete Structure II
Fall, 2016

Why RSA Cipher works

Let M be the message, p, q be two large prime numbers, d be a number that is relatively prime with $(p-1)(q-1)$, and e be a number that is multiplicative inverse modulo $(p-1)(q-1)$ of d. The (d, pq) is the public key, and (e, pq) is the private key that is kept secret.

The encoding function works as follows to map the message M (that is smaller than pq) to cipher code C:

$$C = M^d \bmod pq \tag{1}$$

The decoding function works as follows to decode the cipher code:

$$M' = C^e \bmod pq \tag{2}$$

We prove that $M' \equiv M \pmod{pq}$, and as $M < pq$, this means $M' = M$, so the decoding of the cipher code C yields the original message M.

Proof: Substituting the Eq(1) to Eq(2), we get

$$M' = (M^d \bmod pq)^e \bmod pq = M^{de} \bmod pq$$

In the second step above, we use Theorem 8.4.3 (4).

From e is multiplicative inverse modulo $(p-1)(q-1)$ of d, we have (by definition),

$$de = 1 + k(p-1)(q-1), \text{ for some integer } k$$

So we have

$$M^{de} \bmod pq = M^{1+k(p-1)(q-1)} = M(M^{p-1})^{k(q-1)} = M(M^{q-1})^{k(p-1)}$$

We consider the following two cases: M is relative prime to pq, or M is not relatively prime to pq.

1. If M is relatively prime to pq, i.e., $gcd(M, pq) = 1$, then we have $p \nmid M$ and $q \nmid M$. From Fermat's little theorem, we have

   $$M^{p-1} \equiv 1 \pmod{p}$$

   and therefore

   $$M^{de} = M(M^{p-1})^{k(q-1)} \equiv M(1)^{k(q-1)} = M \pmod{p}$$

   Similarly, since $q \nmid M$, we have from Fermat's little theorem,

   $$M^{q-1} \equiv 1 \pmod{q}$$

   and therefore

   $$M^{de} = M(M^{q-1})^{k(p-1)} \equiv M(1)^{k(p-1)} = M \pmod{q}$$

2. If M is not relatively prime to pq, then either $p \mid M$ or $q \mid M$.

   (a) If $p \mid M$, then $M \bmod p = 0$, and $M^{de} \bmod p = 0$, so $M^{de} \equiv M \pmod{p}$.
   Since $M < pq$, we have $q \nmid M$ (otherwise, if $p \mid M$ and $p \mid M$, then $M \geq pq$, but $M < pq$). Following Fermat's little theorem, we have

   $$M^{q-1} \equiv 1 \pmod{q}$$

   and therefore

   $$M^{de} = M(M^{q-1})^{k(p-1)} \equiv M(1)^{k(p-1)} = M \pmod{q}$$

   So in this case, we also have $M^{de} \equiv M \pmod{p}$ and $M^{de} \equiv M \pmod{q}$

(b) If $q \mid M$, with similar argument as the case above, we can show that $M^{de} \equiv M (\text{mod } p)$ and $M^{de} \equiv M (\text{mod } q)$

Combining the above two cases, we have $M^{de} \equiv M (\text{mod } p)$ and $M^{de} \equiv M (\text{mod } q)$
This means, $p \mid (M^{de} - M)$ and $q \mid (M^{de} - M)$.
By the definition of divisibility, we have

$$M^{de} - M = pt \text{ for some integer } t$$

So we have $q \mid pt$, and since q and p are distinct prime numbers, $gcd(p, q) = 1$, so by Euclid's lemma, we get

$$q \mid t$$

Therefore, $t = qu$ for some integer u, by definition of divisibility.
So we have

$$M^{de} - M = pt = p(qu) = (pq)u, \text{ where } u \text{ is an integer,}$$

So, we have $pq \mid (M^{de} - M)$, and therefore

$$M^{de} \equiv M (\text{mod } pq)$$

As $M < pq$, the above implies that

$$M' = M^{de} \text{mod } pq = M$$

End of Proof.

Additional Notes:

- The security of RSA algorithm comes from the fact that factoring large number is a hard problem. So it's comptuatioanlly expensive to find p, q from pq, which is needed in order to figure out the e in the private key used for decoding the cipher code.
- For sample application of RSA algorithm, you can read further and the following is a good article describing how RSA algorithm can be used to authenticate user during ssh login (as alternative to enter password everytime):

  `https://www.digitalocean.com/community/tutorials/`
  `ssh-essentials-working-with-ssh-servers-clients-and-keys`