Naive RSA Encryption System

with proof of correctness

Sanjay Bhattacherjee

Setup. The setup algorithm takes as input the security parameter ν and generates two $\nu/2$-bit prime numbers p and q. Let $N = p \cdot q$. Since N is composite, $\mathbb{Z}/N\mathbb{Z} = \{0, 1, 2, \ldots, N-1\}$ is a ring with respect to addition and multiplication modulo N. We also have $(\mathbb{Z}/N\mathbb{Z})^*$ as the set of all non-zero elements of $\mathbb{Z}/N\mathbb{Z}$ that are mutually prime to N. It forms a group with respect to multiplication modulo N. The number of elements in this group is given by

$$|(\mathbb{Z}/N\mathbb{Z})^*| = \phi(N)$$

where $\phi(N)$ is Euler's totient function. In particular, we have $\phi(N) = \phi(p)\phi(q) = (p-1)(q-1) = N-(p+q)+1$. Note that there are p multiples of q between 0 and $N-1$ namely those in the set

$$\{0 \cdot q, 1 \cdot q, \ldots, (p-1) \cdot q\}.$$

Similarly, there are q multiples of p between 0 and $N-1$ namely those in the set

$$\{0 \cdot p, 1 \cdot p, \ldots, (q-1) \cdot p\}.$$

Since 0 is a multiple of both p and q, hence the total number of integers outside of $(\mathbb{Z}/N\mathbb{Z})^*$ is $(p+q-1)$.

Let $M = \phi(N)$. We will invoke the notation $\phi(N)$ when we need to keep in mind that $\phi(N) = (p-1)(q-1)$. Otherwise, we will treat this number only as a composite integer M.

For large values of p and q, the fraction $\frac{\phi(N)}{N}$ is close to 1. In other words, the set $(\mathbb{Z}/N\mathbb{Z})^*$ is almost as large a set as $\mathbb{Z}/N\mathbb{Z}$.

We recollect that the extended Euclidean algorithm takes as input two integers a and b and finds their *greatest common divisor* d in a form

$$d = x \cdot a + y \cdot b.$$

The coefficients x and y are also returned by the algorithm. We note here that the extended Euclidean algorithm is a very efficient algorithm.

The next step is to find an integer e such that $\gcd(e, M) = 1$. In other words, we will have to find an element $e \in (\mathbb{Z}/M\mathbb{Z})^*$. We may generate a random integer $x \leq M$ and check if $\gcd(x, M) = 1$. [1] We note here that $|(\mathbb{Z}/M\mathbb{Z})^*| = \phi(M)$ has to be large to easily find such an e. Once we have found such an e, the output of the extended Euclidean algorithm will be in the form

$$1 = x \cdot e + y \cdot M.$$

Applying the modular operation $\pmod{M}$ on both sides of this equation, we get

$$x \cdot e = 1 \pmod{M}.$$

This element x output by the algorithm is the inverse of e in the multiplicative group $(\mathbb{Z}/M\mathbb{Z})^*$. We denote this inverse element by d. So,

$$d \cdot e = 1 \pmod{M}. \tag{1}$$

In summary, the output of the setup phase is:

public key $\mathfrak{pk} \leftarrow (N, e)$

secret key $\mathfrak{sk} \leftarrow (N, d)$

Note that the primes p and q may only be kept hereafter for improving the efficiency of exponentiation as has been explained in Section 6.3.2 in the book [1]. The value of $\phi(N)$ may not be stored at all.

Encryption. The user's message is mapped to an element $m \in \mathbb{Z}/N\mathbb{Z}$. The encryption algorithm takes as input the message m and the public key $\mathfrak{pk} = (N, e)$, and finds the ciphertext c as follows.

$$c \leftarrow m^e \pmod{N}.$$

[1] There may be more efficient ways to find such an integer x depending upon the choices of the primes p and q.

Decryption. The decryption algorithm takes as input the ciphertext c and the secret key $\mathfrak{sk} = (N, d)$, and finds the message m' as follows.

$$m' \leftarrow c^d \pmod N.$$

We say that the system works correctly if the message m that was encrypted is indeed the one that has been found as the output of decryption. In other words, the system works correctly if $m = m'$.

Correctness. To show that the scheme is correct, we have to show that

$$\begin{aligned} m = m' = c^d &\pmod N \\ = (m^e)^d &\pmod N \\ = (m)^{e\cdot d} &\pmod N \end{aligned} \tag{2}$$

At this point, we recollect from Equation 1 that $e \cdot d = 1 \pmod M$. So, we may write $e \cdot d = 1 + s \cdot M$ for some integer s. Substituting this value of $e \cdot d$ in Equation 2, the new statement to be proved is

$$\begin{aligned} m = m^{1+s\cdot M} &\pmod N \\ \implies m = m \cdot m^{s\cdot\phi(N)} &\pmod N \\ \implies 0 = m \cdot \left(m^{s\cdot\phi(N)} - 1\right) &\pmod N. \end{aligned} \tag{3}$$

To prove Equation 3, we will show that

$m \cdot \left(m^{s\cdot\phi(N)} - 1\right)$ *is a multiple of* N.

We recollect at this point that $m \in \mathbb{Z}/N\mathbb{Z}$. So,

- either $m \in (\mathbb{Z}/N\mathbb{Z})^*$,
- or $m \notin (\mathbb{Z}/N\mathbb{Z})^*$.

In case $m \notin (\mathbb{Z}/N\mathbb{Z})^*$, then either m is one of the q multiples of p, or m is one of the p multiples of q. When m is a multiple of p, it is mutually prime with q and vice versa. Only when $m = 0$, both p and q divide m. In summary, there are four possible scenarios as listed below.

	divisibility by p		divisibility by q
Case 1:	$\gcd(m, p) = 1$	and	$\gcd(m, q) = 1$
Case 2:	$m = 0 \pmod p$	and	$\gcd(m, q) = 1$
Case 3:	$\gcd(m, p) = 1,$	and	$m = 0 \pmod q$
Case 4:	$m = 0 \pmod p),$	and	$m = 0 \pmod q$

We look at each of these cases in the following.

1. This is the most common case as mentioned earlier. In this case, since $\gcd(m, p) = 1$ and $\gcd(m, q) = 1$, hence $m \in (\mathbb{Z}/N\mathbb{Z})^*$. By Euler's theorem, we have

$$\begin{aligned} m^{\phi(N)} = 1 &\pmod N \\ \implies \left(m^{\phi(N)}\right)^s = 1 &\pmod N \\ \implies m^{s\cdot\phi(N)} = 1 &\pmod N). \end{aligned} \tag{4}$$

   Hence, N divides $m \cdot \left(m^{s\cdot\phi(N)} - 1\right)$ and hence Equation 3 is proved to be correct.

2. Since $\gcd(m, q) = 1$, we use Fermat's little theorem to note that

$$\begin{aligned} m^{(q-1)} = 1 &\pmod q \\ \implies \left(m^{(q-1)}\right)^{(p-1)} = 1 &\pmod q \\ \implies m^{\phi(N)} = 1 &\pmod q \\ \implies \left(m^{\phi(N)}\right)^s = 1 &\pmod q \\ \implies m^{s\cdot\phi(N)} = 1 &\pmod q). \end{aligned} \tag{5}$$

   From Equation 5 we get that q divides $\left(m^{s\cdot\phi(N)} - 1\right)$. We already know that in this case, p divides m. Combining these two facts, we get that N divides $m \cdot \left(m^{s\cdot\phi(N)} - 1\right)$ and hence Equation 3 is proved to be correct.

3. The arguments in this case are the same as Case 1, by interchanging the prime q with p.

4. For $m = 0$, Equation 3 is trivially correct. However, note that $c = m^e \pmod{N} = 0$ which is unchanged from m.

Note on the choice of m. In practice, for any choice of $N = p \cdot q$, there will be $(p+q-1)$ elements $m \in \mathbb{Z}/N\mathbb{Z}$ for which $\gcd(m, N) > 1$. These are precisely the elements $m \notin (\mathbb{Z}/N\mathbb{Z})^*$ and hence $\gcd(m, N) \neq 1$. If an attacker of the system (with only the knowledge of the public key $\mathfrak{pk} = (N, e)$) is able to find such a message m, then they can compute $\gcd(m, N)$ to get one of the two primes p or q. This will provide the factorisation for N and as a result, the cryptosystem will be broken.

Now, if all messages $m \in \mathbb{Z}/N\mathbb{Z}$ are equally likely to occur, then the probability that an attacker gets hold of such a message m for which $\gcd(m, N) > 1$ is negligibly small. The precise probability is given by

$$\frac{p+q-1}{N}.$$

We note that p and q are $\nu/2$-bit primes. So, $p + q - 1$ would be of around that size as well while N will be approximately of size ν bits. So, the probability of finding such an element m is around $\frac{1}{2^{\nu/2}}$. Hence, the cases 2, 3 and 4 of the correctness proof are extremely unlikely to occur.

However unlikely, if an innocent sender of messages actually takes up one such $m \notin (\mathbb{Z}/N\mathbb{Z})^*$ and tries to encrypt it, the encryption and decryption mechanisms as designed, will work correctly. That is because of cases 2, 3 and 4 above where we have shown that the correctness property still holds in these two cases.

Note on padding and the use of $m = 0$. The reader may have noted that when $m = 0$, the ciphertext $c = 0$. For the naive RSA scheme, a passive observer will know that the message corresponding to $c = 0$ is nothing but $m = 0$. The purpose of encryption will thus be defeated. This may lead us to think that the element $m = 0$ should be left out from representing any message that is to be encrypted using naive RSA.

We first note that for uniformly distributed messages, the probability $1/N$ of $m = 0$ occurring is extremely low. This is even lower than the probability that an adversary accidentally picks up an m outside of $(\mathbb{Z}/N\mathbb{Z})^*$ and finds the factorisation of N. So, we may assume that to be extremely unlikely.

Nevertheless, if naive RSA is indeed used on its own for encrypting messages $m \in \mathbb{Z}/N\mathbb{Z}$, then this will remain an issue to be careful about. However, in practice naive RSA is never used directly. As an example, let us note that the elements of $\mathbb{Z}/N\mathbb{Z}$ are ν bits long. One would usually be encrypting messages that are much larger than ν bits in size. Such a message will have to subdivided into smaller chunks of bits, each of which are smaller than ν bits. These smaller chunks are appended with a few additional bits and then mapped to elements in $\mathbb{Z}/N\mathbb{Z}$ in a pseudo-random manner. This mechanism is called *padding*. One of the most popular padding schemes in use today was invented by Bellare and Rogaway and is called Optimized Asymmetric Encryption Padding (OAEP). Please refer to section 16.2.1 of [1] for some more details on OAEP.

When using a padding scheme, its output is an element $m \in \mathbb{Z}/N\mathbb{Z}$ that is then encrypted by the RSA encryption algorithm to create the ciphertext. The pseudo-random nature of the padding mechanism ensures that a message does not get mapped to the same m or the corresponding ciphertext c every time. More specifically, even if $m = 0$ occurs as the output of the padding scheme (and consequently results in $c = 0$ to be observed by the adversary), it does not give out the original message that has been mapped in a pseudo-random fashion with $m = 0$ by the padding scheme.

The main purpose of using a padding scheme however, is to prevent the *dictionary attack*. Note that in naive RSA, once a key pair $(\mathfrak{pk}, \mathfrak{sk})$ has been fixed, a message will always get encrypted to the same ciphertext. A passive adversary may create a dictionary of some messages of their choice and the corresponding ciphertexts that they themselves can create using the $\mathfrak{pk}$. Whenever a ciphertext occurs that is in the adversary's dictionary, they will know which message is being sent. Using RSA with a padding scheme maps a message with elements of $m \in \mathbb{Z}/N\mathbb{Z}$ on the fly in a pseudo-random manner. It removes the deterministic mapping between the message and the ciphertext that a passive adversary may have exploited. Hence, such an adversary will no longer be successful (with any non-negligible probability) in identifying the messages just by observing the ciphertexts.

Acknowledgement. The detailed proof of correctness is not available in any resource that the author is aware of. The overall idea and the structure of the proof has been arrived at in discussion with Prof. Palash Sarkar. He has also kindly reviewed an earlier version and helped add several details.

References

[1] Nigel P. Smart. *Cryptography Made Simple*. Information Security and Cryptography. Springer, 2016. Kent LibrarySearch Link: https://link-springer-com.chain.kent.ac.uk/book/10.1007%2F978-3-319-21936-3.