[CS304] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy
Scribed by: Archit Agrawal (202051213)

Winter 2022-2023
Lecture 19(Week 10)

1 RSA (Rivest Shamir Adleman) Encryption

It is the first public key encryption algorithm. Before beginning any discussion on RSA encryption, let's first recall a few concepts.

- The Euler's Totient Function $\phi(n)$ denotes the number of integers less than n that are co-prime to n, i.e number of x such that gcd(x, n) = 1 where $1 \leq x \leq n-1$. For example, $\phi(8) = 4, \{1, 3, 5, 7\}]$ are co-prime to 8. Also, we know,

$$\phi(p) = p - 1, \text{ if p is prime}$$
$$\phi(p^k) = p^k(1 - \tfrac{1}{p})$$

- Let there be a set $S = \{x \text{ mod m}\}$ such that $|S| = m$.

$$S = \{r_1, r_2, \ldots, r_m\}$$

All the elements in the set S are unique (usually they are from 0 to m-1). Assume an integer a such that $gcd(a, m) = 1$. Let's say that there is another set S_1 such that,

$$S_1 = \{ar_1 \text{ mod } m, ar_2 \text{ mod m}, \ldots, ar_m \text{ mod } m\}$$

Since, $\{r_1, r_2, \ldots, r_m\}$ are different elements and gcd(a, m) = 1, it can be concluded that $\{ar_1$ mod m, ar_2 mod m, $\ldots, ar_m$ mod $m\}$ will also be m unique elements. This can be proved using contradiction. Suppose, if $ar_i = ar_j$ for $r_i \neq r_j$. Therefore,

$$ar_i \equiv ar_j \text{ mod } m$$

Since, gcd(a, m) = 1, therfore, $1 = ab + ms($ from Bezout's Identity). Therefore, there exists an integer b such that $ab \equiv 1$ mod m. The value of b is known as multiplicative inverse of a and it can be found using Extended Euclidean Algorithm. Therefore, on multiplying the above equation by b on both sides gives us,

$$b \cdot a \cdot r_i \equiv b \cdot a \cdot r_j \text{ mod } m$$

$$r_i \equiv r_j \text{ mod m } (\because ab \equiv 1 \text{ mod m})$$

Hence, it is a contradiction to our initial assumption that $r_i \neq r_j$. Hence, elements in the set S_1 will be unique iff gcd(a, m) = 1.

1.1 Euler's Theorem

If gcd(a, m) = 1, then $a^{\phi(m)} \equiv 1$ (mod m).
Let us assume, that we have a set S, such that

$$\text{S} = \{ \text{ x} \mid \text{gcd(x, m)} = 1\}$$
$$\text{S} = \{s_1, s_2, s_3, s_4, ..., s_{\phi(m)}\}$$

Let us consider gcd(a, m) = 1 and create another set S_1 such that

$$S_1 = \{ \ as_1, as_2, as_3..., as_{\phi(m)} \ \}$$

We know from the discussion above that, if $as_i \equiv as_j$ (mod m) $\Rightarrow s_i \equiv s_j$(mod m)
Given that gcd(a, m) = 1 and $b.a \equiv 1$ (mod m)

$$|S| = \phi(m)$$
$$|S_1| = \phi(m)$$

Since a is co-prime with m and s_i is also co-prime with m, then there must be some correspondence between elements of S and S_1.

$$s_i \equiv as_j \text{ (mod m)}$$

Let us now take product on both sides,

$$\prod_{i=1}^{\phi(m)} s_i \equiv \prod_{j=1}^{\phi(m)} as_j (mod\ m)$$
$$\Rightarrow \prod_{i=1}^{\phi(m)} s_i \equiv a^{\phi(m)} \prod_{j=1}^{\phi(m)} s_j (mod\ m)$$

Since gcd(s_i, m) = 1, each s_i will have multiplicative inverse under mod m. So, after simplifying we get,

$$a^{\phi(m)} \equiv 1 (mod\ m)$$

1.2 Fermat's Theorem

If p is a prime number and p does not divide a(means that p is co-prime to a), then

$$a^{p-1} \equiv 1 (mod\ p)$$

Using Fermat's theorem,

$$\Rightarrow a^p \equiv a (mod\ p)$$

Note: If $p|a$(P divides a), then,

$$\text{a} \equiv 0 \text{ mod p}$$
$$\Rightarrow a^p \equiv 0\ mod\ p$$
$$\Rightarrow a^p \equiv a\ mod\ p$$

But the Fermat's theorem will not hold when p does not divide a.

1.3 RSA Cryptosystem

Few facts:

- g(a, m) = 1, then $a^{\phi(m)} \equiv 1(mod\ m)$
- $a^{p-1} \equiv 1(mod\ p)$

Now, let us understand the components of RSA

1. n = pq, where p, q are primes
2. Plaintext space = Z_n
   Ciphertext space = Z_n
3. Key space = {K = (n,p,q,e,d) | $ed \equiv\ (mod\ \phi(n))$}
4. Encryption:

$$E(x, K) = c$$
$$c = E(x, K) = x^e (\text{mod } n)$$

5. Decryption:

$$Dec(c, K) = x$$
$$c = Dec(c, K) = c^d (\text{mod } n)$$

We know that e and d are related as:

$$ed \equiv 1 \text{ mod } \phi(n)$$
$$\Rightarrow ed - 1 = t.\phi(n)$$
$$\Rightarrow 1 = ed + t_1.\phi(n)$$
$$1 = gcd(e, \phi(n)) = ed + t_1.\phi(n)$$

Encryption:

$$c = x^e (\text{mod } n)$$

Decryption:

$$x = c^d (\text{mod } n)$$
$$c^d = (x^e)^d \text{ mod } n$$
$$c^d = x^{ed} \text{ mod } n$$
$$\text{Now using } ed = 1 + t.\phi(n) \text{ from above}$$
$$c^d = x^{1+t.\phi(n)} \text{ mod } n$$
$$c^d = x.x^{t.\phi(n)} \text{ mod } n$$
$$\text{Since p, q are primes and n = pq, then } \phi(n) = (p-1)(q-1)$$
$$c^d = x.x^{t[(p-1)(q-1)]} \text{ mod } n$$
$$\text{Finally, } c^d = x.x^{t[(p-1)(q-1)]} \text{ mod } (pq)$$

Now, let us simplify the part $x^{t[(p-1)(q-1)]}$ mod (pq), where $x \in Z$
We check $x^{t[(p-1)(q-1)]}$ mod p,

$$\equiv x^{p-1^{t(q-1)}} \text{ mod p}$$
$$\equiv 1(\text{mod p}) \text{ [As } x^{p-1} \equiv 1 mod\ p]$$

Now we check $x^{t[(p-1)(q-1)]}$ mod q

$$\equiv x^{q-1^{t(p-1)}} \text{ mod q}$$
$$\equiv 1(\text{mod p}) \text{ [As } x^{q-1} \equiv 1 mod\ q]$$

We finally have,

$$x^{t[(p-1)(q-1)]} \equiv 1(\text{mod p})$$
$$x^{t[(p-1)(q-1)]} \equiv 1(\text{mod q})$$
$$\Rightarrow x^{t[(p-1)(q-1)]} \equiv 1(\text{mod pq})$$

Substituting the above result,

$$c^d = \text{x}.x^{t[(p-1)(q-1)]} \text{ mod (pq)}$$
$$c^d = \text{x}.1 \text{ mod (pq)}$$
$$c^d = \text{x mod (pq)}$$

Hence, our decryption is successful.

Now let us consider a scenario where Alice is trying to communicate with Bob. Here, two keys play the main role-one is public key and the other is secret key. Here, Bob is encrypting the message and sending it to Alice and Alice has both the keys. Public key is known to Bob but the secret key is not known to Bob.

Alice

n = pq and p, q are large prime numbers

ed $\equiv$ 1 (mod ϕ(n))

She chooses e

Public key of Alice = (n, e)

She can generate d using extended euclidean algorithm

Secret key of Alice = (p, q, d)

Bob

Now Bob selects a message x from Z_n

x

He knows n, e for Alice, so he can encrypt

y = $x^e(modn)$

Now the message y is sent to Alice, she can decrypt it with her secret key as :

$$x = y^d \text{ (mod n)}$$

Now, if we were given n, how can we find p and q in polynomial time?
We can run a loop from 2 to $\sqrt{n}$, and everytime we calculate n % i, if it is zero, it means we have found a factor, then we check if the factor is prime. If yes, then we divide n by p, to get q. Getting the prime factors of a number n is computationally hard problem when n is large.
Note: If we are able to compute p,q from n, then we will be able to compute ϕ(n). And then we already have e, so we will be able to find d by extended euclidean algorithm and security of RSA will be broken. So, RSA is based on the fact that factorization problem is hard.

1.3.1 RSA Problem

We have public key (n, e) and c. If from this we can find x ($c = x^e$), we will be able to break security of RSA.
We have an algorithm to solve the RSA problem, i.e., it can find the decryption without the factorization. Is this true? **Note:** If we can break the RSA, there is no guarantee that we can find the factors. But if we have the factors, we can always Break the RSA. But the reverse is not always true. So RSA is secure under two assumptions:

- factorization is hard
- decryption is hard

Note : Public key encryption is generally heavy because of x^e and c^d operation. The exponential operations are heavy. So they are usually avoided.