Project : A study of Public-Key Cryptosystems

Amit Kumar Ghosh
amitgho@microsoft.com

Table of Contents

1. Prologue:

For a long time I had a strong desire to learn the RSA encryption scheme and related other technologies like digital signatures, certificates etc. So I have used this opportunity to study the underlying theory and understand how they work. I will start with couple of number theoretic lemmas which are the base of the RSA Cryptosystem. Then will describe the RSA scheme, its algorithmic complexity and level of security. Followed by traditional private-key (aka symmetric key) encryption and its short-comings that motivates system like RSA. Then the usage of RSA in encryption and authentication (digital-signatures). Finally conclude with the public key distribution problem and certificates as a solution.

2. Number theoretic fundamentals:

□ **Lemma-1:** $\forall p, q \in I,\ \exists \alpha, \beta \in I\ s.t.\ \alpha p + \beta q = \gcd(p, q)$.
□ **Proof:** Let $c = \alpha' p + \beta' q$ be the smallest positive number in S= $\{\alpha p + \beta q : \alpha, \beta \in I\}$.
Let $p = kc + r, 0 \leq r < c$
$\Rightarrow p = k(\alpha' p + \beta' q) + r$
$\Rightarrow r = (1 - k\alpha')p + (-k\beta')q \in S$
$\Rightarrow r = 0, cause\ c\ is\ smallest\ positive\ in\ S$
$\Rightarrow c|p, Similarly\ c|q$

Now $\forall x\big((x|p \wedge x|q) \Longrightarrow x\big|(\alpha' p + \beta' q)\big)\ i.e., \forall x\big((x|p \wedge x|q) \Longrightarrow x|c\big)$
Hence $\gcd(p,q) = c = \alpha' p + \beta' q$. **[Proved]**

□ **Lemma-2:** *If p is a prime and a is any integer then* $a^p \equiv a\ (mod\ p)$
□ **Proof:** $(a+1)^p = a^p + \frac{p}{1}a^{p-1} + \frac{p.(p-1)}{1.2}a^{p-2} + \cdots + 1$, Clearly, each term in the expansion except a^p *and* 1 are divisible by p. Hence, $(a+1)^p - a^p - 1 = kp$

$(a)^p - (a-1)^p - 1 = k_1 p$
$(a-1)^p - (a-2)^p - 1 = k_2 p$
…
$(a - \overline{a-1})^p - (a-a)^p - 1 = k_a p$

Adding, $a^p - a = (k_1 + \cdots + k_a)p \Longrightarrow a^p \equiv a\ (mod\ p)$ **[Proved]**
Corollary: *If a is relative prime to p then* $a^{p-1} \equiv 1\ (mod\ p)$. [**Fermat's little theorem**]
Proof: $a^p \equiv a\ (mod\ p) \Longrightarrow p|(a^p - a)) \Longrightarrow p|a(a^{p-1} - 1)) \Longrightarrow p|(a^{p-1} - 1)$ **[Proved]**

□ **Lemma-3:** $a \equiv b\ (mod\ p) \wedge c \equiv d\ (mod\ p) \Longrightarrow ac \equiv bd\ (mod\ p)$.
□ **Proof:** $a \equiv b\ (mod\ p) \Longrightarrow a = k_1 p + b, and$
$c \equiv d\ (mod\ p) \Longrightarrow c = k_2 p + d$
$ac = (k_1 p + b)(k_2 p + d) = (\ldots)p + bd \Longrightarrow ac \equiv bd\ (mod\ p)$ **[Proved]**

□ **Lemma-4:** *If p, q are relatively prime to each other and $a, b \in I$*
then $a \equiv b\ (mod\ p) \wedge a \equiv b\ (mod\ q) \Longrightarrow a \equiv b\ (mod\ pq)$.
□ **Proof:** $a \equiv b\ (mod\ p) \Longrightarrow p|(a-b) \Longrightarrow (a-b) = k_1 p, and$
$a \equiv b\ (mod\ q) \Longrightarrow q|(a-b) \Longrightarrow q|k_1 p \Longrightarrow q|k_1 (as\ p, q\ are\ relative\ prime) \Longrightarrow k_1 = kq$
$\Longrightarrow (a-b) = kqp \Longrightarrow a \equiv b\ (mod\ pq)$ **[Proved]**

□ **Lemma-5:** $(a\ mod\ n)^x \equiv a^x mod\ n$
□ **Proof:** Let $a = kn + a'$, hence $(a\ mod\ n)^x \equiv a'^x mod\ n \equiv a^x mod\ n$ **[Proved]**

3. RSA Scheme:

- Generate two large prime numbers $p\ and\ q$.
- Set $N = pq$
- Take e s.t. e is relatively prime to $(p-1)(q-1)$.
- Choose d s.t.$ed \equiv 1\ mod(p-1)(q-1)$ [Possible, by **lemma-1**]
- With these choices $M^{ed} mod\ N = M,\ 0 \le M < N$ /* M is the message to be encrypted */

Proof:
$(p-1)(q-1)|(ed-1) \Longrightarrow (p-1)|(ed-1)\ and\ (q-1)|(ed-1)$
$\Longrightarrow (ed-1) = k(p-1)$
$Now\ M^{ed} = M^{k(p-1)+1} = M^{k(p-1)}M$
Case-1: M is co-prime to p,
$M^{(p-1)} \equiv 1(mod\ p)$ [**Lemma-2 corollary**]
$M^{k(p-1)} \equiv 1^k \equiv 1(mod\ p)$ [**Lemma-3**]
$Hence\ M^{ed} \equiv 1.M \equiv M(mod\ p)$
Case-2: p | M,
$M^{ed} \equiv 0 \equiv M(mod\ p)$
Hence in both cases $M^{ed} \equiv M(mod\ p)$,

Similarly, $M^{ed} \equiv M(mod\ q)$
Combining, $M^{ed} \equiv M(mod\ pq)$ [**Lemma-4**]
$i.e., M^{ed}\ mod\ N = M$ **[Proved]**

- Define encryption as $E(M) = M^e mod\ N$, And denote the pair **(e, N)** as **public-key**.
- Define decryption as $D(C) = C^d mod\ N$, And denote the pair **(d, N)** as **private-key**.
- Note that, $D\big(E(M)\big) = (M^e mod\ N)^d mod\ N = M^{ed} mod\ N = M$, used for encryption
- Similarly, $E\big(D(M)\big) = M$, used for digital-signature. [**Lemma-5**]

4. Complexity of RSA Scheme:

Using "exponentiation by repeated squaring and multiplication" encryption and decryption in RSA becomes $O((\log \mathrm{N})^3)$ $i.e., O(n^3)$

5. How secure RSA is:

Only way to break the RSA, i.e., figuring out the private key (d, N) form the public key (e, N)) is by factorizing N (which is normally more than 1024 bits). And as of now factorizing is computationally intractable.

6. Traditional Private key or Symmetric encryption:

- Single key to encrypt and decrypt. So has to kept secret among the parties.
- Example: Let M be one byte. E(M) = rotate_right(M, 3) = C, D(C) = rotate_left(C, 3) = M. Here the private symmetric key is 3.

7. Problem with Private key encryption:

To exchange the key (private) itself, you need a secure channel first!!

8. Solution - Public Key Encryption:

Public key encrypts, private key decrypts

- 'A' wants to send message M to 'B'. $Key\ pair\ of\ A: \{E_A, D_A\}, Key\ pair\ of\ B: \{E_B, D_B\}$
  - $E_A, E_B: public, known\ to\ everybody$ (may reside in some public directory)
  - $D_{A,}\ D_B: private$ (known only to the respective parties)
- 'A' uses B's public key E_B to encrypt M and send the encrypted message $E_B(M)$ to B.
- B uses his private key D_B to decrypt $E_B(M)$ into $D_B(E_B(M)) = M$.

9. Message Digest and Digital Signature:

Private key signs, public key verifies

The properties of RSA public key encryption can be used to provide a way to authenticate the identity of the sender. That means we can have digital signature attached to the message that uniquely identify the sender as the author of that message. Let us say 'A' sends a message M to 'B'. Using MD5 (which calculates 128 bit hash of a message) 'A' can generate a digest of M. And encrypt that digest with his private key. Then sends M along with the encrypted digest to 'B'. Now 'B' can decrypt the encrypted digest using A's public key and match the value with the MD5 digest of M (calculated again by B). If they match then B can be sure that M is not

tempered and indeed M is coming from A. Clearly enough, for the message M, the digital signature of 'A' is $D_A(MD5(M))$. Note that signature is unique to the message and sender pair.

10. Authentication using public key encryption - Challenge-response protocol:

If you know a person's public key you can authenticate the person on the other end of a network connection. One way of doing this would be to generate a random string and pass it to the person on the other end. They encrypt it with their private key and send it back to you. If you decrypt the reply with the public key of the person you believe you are talking to and get back your original string, then the person on the other end of the connection must know the private key. The other end of the connection can authenticate you in a similar way.

11. Distribution of public key:

- We can have centralized the directory to hold public keys. But what if the sever is down? Or overloaded? Even the retrieval of public key over the internet can be forged!!
- The ideal solution is to send the public key as part of the message. But this has a chicken and egg problem. How can you ensure that you are talking with the right guy? No way to ensure. Implies no way to authenticate!!
- Solution: Certificate and Certification Authority.

12. Certificate and CA(certification authority):

A certificate is a message that contains your name and your public key, digitally signed by the private key of a trusted third party - the certification authority. To obtain a certificate you visit a certificate issuer with whatever form of identification they require. They produce a message which contains something like

Name: amitgho@cs.washington.edu

Public key: ABCDEFGHIJKLMNOPQRSTUW

and digitally sign it with their privates key. When someone wants to communicate with you, you send them this certificate and, as long as they know the certification authority's public key, they can check the signature on the certificate and extract your public key. It is not necessary to know the public key of every certifier, because certifiers can create certificates for other certifiers. For example, the CS department could get a certificate for their public key and then issue certificates to students signed with their private key. The user would send a certificate chain to authenticate themselves - i.e. they would send the department's certificate, signed by the certifying authority, to tell the person on the other end what the department's public key was, and their certificate, signed by the department, to tell the other end what their public key was.

13. Practical Implementation:

Although the above sounds complicated, most of it can be handled automatically behind the scenes. Web browsers, for example, come with the public keys of the major certifying authorities compiled into them. If they receive a certificate signed by an authority they don't know about, then, as long as that certificate is accompanied by another certificate signed by an authority they do know about that authenticates it, they can add the new public key to their list and authenticate users whose certificates have been signed by the new authority from then on.

Public key encryption is generally too slow to be used for extended exchanges, so once the ends of a connection have authenticated themselves to each other, a random symmetric key is generated and sent via public key encryption. This key is then used to encrypt the connection with a faster, symmetric cipher.

14. Tools:

Certmgr.msc (WinNT) – Helps to view various certificates installed in your system. Allows you to request new certificate, etc.

15. Reference:

[1] Original RSA paper: "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems" - R.L. Rivest, A. Shamir, and L. Adleman (http://theory.lcs.mit.edu/~rivest/rsapaper.pdf)

[2] RFC 2585 (PKI): "Internet X.509 Public Key Infrastructure Operational Protocols: FTP and HTTP" (http://rfc.net/rfc2585.html)