An introduction to RSA

Saša Radomirović

February 29, 2000

1 What is *RSA* ?

RSA is a crypto system developed by Rivest, Shamir, and Adleman in 1977. It allows Alice to *publicly* announce or distribute a key which Bob or anybody else can use to send a secret message to Alice. Nobody but Alice will be able to decipher the secret message.

2 What is the difference between *RSA* and Diffie–Hellman (DH)?

DH allowed Alice and Bob, who might have never met before, to establish a secret in order to communicate securely. The whole communication including the generation of the secret takes place while Eve is listening.
In order to establish the secret, both Alice and Bob had to perform some calculations and they had to talk to each other while doing this.
In *RSA* only the receiver of the message, who will be Alice in our case, needs to perform calculations to establish what is called a *secret key* and a *public key*. She doesn't need to know from whom she will receive secret messages. All she has to do is prepare *one* secret key and *one* public key and then distribute the public key. She can even give her public key to Eve! Everybody who has Alice's public key can send secret messages to Alice, but only Alice can decipher the messages and nobody else.

3 How does this work?

3.1 Background

The most important thing to remember is Euler's equation: For distinct primes p and q,

$$m^{(p-1)(q-1)} = 1 \bmod pq \tag{1}$$

Furthermore, for any numbers d and e (and regardless of any "mod"),

$$(m^d)^e = (m^e)^d = m^{d \cdot e}. \tag{2}$$

Please don't confuse this with another very important equality

$$m^d \cdot m^e = m^{d+e}. \tag{3}$$

Remember that equations like $2 \cdot x = 1 \bmod N$, or more generally

$$e \cdot x = 1 \bmod N \tag{4}$$

can be solved easily and rapidly. We will assume this. If you are interested in how such equations can be solved, a description of the process is in your text (pages 108–111).

3.2 Alice's preparation

Alice's first step Alice chooses two large different prime numbers p and q and computes $n = pq$.

Alice's second step She chooses an e and computes a d by solving the equation

$$\boxed{e \cdot x = 1 \bmod (p-1)(q-1)}$$

which is easy to solve according to (4). Her d will be the x which solves this equation.

Question Why does she do that?

Answer As the homework showed, for any number m

$$(m^e)^d = m \bmod n$$

Question Why is that so?

Answer First look at what $1 \bmod (p-1)(q-1)$ means. Since at this point there's nothing special about the $(p-1)(q-1)$ we can write $1 \bmod N$ for more convenience. Notice now, that

$$\begin{array}{rcl} 1 & = & 1 \bmod N \\ N+1 & = & 1 \bmod N \\ N+N+1 & = & 1 \bmod N \\ \vdots & & \vdots \\ N+\ldots+N+1 & = & 1 \bmod N \end{array}$$

If we plug in $(p-1)(q-1)$ for N, we get

$$\underbrace{(p-1)(q-1)+\ldots+(p-1)(q-1)}_{\text{Any \# of times}}+1 = 1 \bmod (p-1)(q-1)$$

On the other hand, Alice has found out in step A2 that

$$e \cdot d = 1 \bmod (p-1)(q-1)$$

so

$$e \cdot d = \underbrace{(p-1)(q-1)+\ldots+(p-1)(q-1)}_{\text{some \# of times}}+1.$$

We don't know how many times $(p-1)(q-1)$ appears, but we shall see that won't matter. Euler's equation tells us that

$$m^{(p-1)(q-1)} = 1 \bmod pq$$

and from the homework we know

$$m^{(p-1)(q-1)+1} = m \bmod pq.$$

We can keep multiplying by "1" any number of times mod pq. That's exactly the same as multiplying by $m^{(p-1)(q-1)}$, so

$$m^{(p-1)(q-1)+\ldots+(p-1)(q-1)+1} = m \bmod pq.$$

But the left hand side is just $m^{e \cdot d}$ and from equation (2) we know that that is equal to $(m^e)^d$. So Alice has found two numbers e and d satisfying the equation which follows.

$$\boxed{(m^e)^d = m \bmod n}$$

Now Alice is prepared.

3.3 Alice's announcement

What Alice has prepared in the last step are a so called *secret key* and a *public key*:

Alice's secret key n and d.

Alice's public key n and e

Alice's third step She gives the numbers n and e to everybody she knows.

3.4 Bob's encryption

So Bob got n and e from Alice. Suppose m is Bob's message.

Bob encrypts Bob computes m^e mod n and calls this result, r.

Bob transmits Bob sends the result r to Alice.

3.5 Alice's decryption

Alice gets m^e mod n from Bob. What Alice wants is m.

Alice decrypts In order to recover the message m Alice computes

$$r^d = (m^e)^d = m \bmod n.$$

r is Bob's encryption of his message. Let's try to be cryptanalysts now, and attack the system.

3.6 What Eve might try to do

Question What does Eve know?
Answer She knows n, e <u>and</u> $r = m^e$. She even knows that r <u>is</u> m^e mod n because we will assume Eve knows not only the specific numbers she can observe but also knows the general idea of the system: she knows that Alice and Bob are using RSA . This is a generous assumption, but it is one which has been shown historically to be useful. Generally it is assumed that an attacker knows the cryptosystem, but does not know the key. This assumption, formulated explicitly about a century ago, is called "Kerckhoff's maxim" and is named after a Dutch cryptographer who spent most of his life in France. (You can see `http://www.cl.cam.ac.uk/~fapp2/kerckhoffs/`.)

Question What does Eve want to find out?
Answer What Eve wants is m.

Question What does Alice know that Eve doesn't know?
Answer Alice knows d.

Question How did Alice compute d?
Answer With the equation

$$e \cdot x = 1 \text{ mod } (p-1)(q-1).$$

So *one* thing Eve could try is to discover p and q from n. If she had p and q, then she could then compute $(p-1)(q-1)$ and get d just as Alice did. If Eve could find the factors of n, breaking the system is quite easy. But

$$\boxed{\textbf{Factoring seems to be very hard}}$$

when numbers are very large, that is if the primes p and q are very large. That's an obstacle Eve faces when she tries to discover the message m.

4 An Example

In "the real world" nobody would ever use numbers as small as those we use here. This is just for instructional purposes.

Alice's first step Alice chooses 11 and 5 as her primes, and computes $n = 11 \cdot 5 = 55$.

Alice's second step She chooses $e = 3$ and finds that $d = 27$.

Question How does she find the d?
Answer She solves the equation $3 \cdot x = 1$ mod $10 \cdot 4$.
In `Maple` she would type: `msolve(3*x=1,10*4);` What she gets is $x = 27$, so she sets $d = 27$.

Alice's third step She gives $n = 55$ and $e = 3$ to Bob.

Bob's got $n = 55$ and $e = 3$ from Alice. Suppose Bob's message is $m = 9$.

Bob computes He computes m^e mod n, which is 9^3 mod $55 = 14$.

Bob transmits He sends 14 to Alice.

Alice gets 14 from Bob.

Alice decrypts She computes $(m^e)^d \bmod n$, which is $14^{27} \bmod 55$. What she gets is $m = 9$, Bob's message!

What about Eve? What has Eve heard? $n = 55$, $e = 3$, and secret message: $r = 14$. She should know d to decrypt this easily. But to find d efficiently, she has to know p and q! In this case, Eve has no problems factoring n. Everybody knows that $5 \cdot 11 = 55$. So Eve sits at her computer, runs `Maple` and types: `msolve(3*x=1,4*10);`. Rapidly `Maple` tells her that $x = 27$. Now Eve knows Alice's secret key and can decrypt every message Alice gets the same way that Alice does.

5 The Real World

In the real world now the primes p and q each have about 170 digits.

Lots of people are trying to factor numbers which are the product of two big primes.

The largest such number factored so far is a product of two 78 digit primes and is known as RSA-155. This result was announced August 22, 1999 by Dr. H. J. J. te Riele at the CWI in Amsterdam (which is the national research institute for mathematics and computer science in the Netherlands) and was a result of joint work involving many organizations and people.

The number

1094173864157052742180970732204035761200373294544920599091384213147634998428893
4784717997257891267332497625752899781833797076537244027146743531593354333897

can be written as the product of the two 78-digit primes:

102639592829741105772054196573991675900716567808038066803341933521790711307779

and

106603488380168454820927220360012878679207958575989291522270608237193062808643

The factoring process for RSA-155 took about 7 months, more than 300 workstations were involved, and in the end even a huge Cray supercomputer was needed.

I've estimated that the amount of computer time spent on this new factoring world record is the equivalent of over 100 years on a standard PC. This is really not a useful comparison, because the amount of memory needed was almost 20 times as much as a standard PC has. Some details: 2048 MB of RAM and up to 3.7 GB of hard disk space were needed.

The factoring process had various stages. At the end a huge system of linear equations had to be solved. This is the part that required the enormous amount of memory, because there were about 6,700,000 equations with 6,700,000 unknowns to be solved.

The original paper on *RSA* was *A method for obtaining digital signatures and public-key cryptosystems* by R. Rivest, A. Shamir, and L. Adleman. It appeared in the February 1978 issue of the Communications of the ACM (this is the Association for Computing Machinery), volume 21, pages 120–126. The original paper on the factorization of RSA-155 can be obtained from `ftp://ftp.cwi.nl/pub/herman/NFSrecords/RSA-155`. You can read about it on the web page `http://www.rsasecurity.com/rsalabs/challenges/factoring/rsa155.html`.

Acknowledgment

Professor Greenfield is kindly acknowledged for discussions, answering questions, proofreading, and editing the manuscript.