The RSA algorithm (or how to send private love letters)

April 10 2013, by Adrian Dudek

Sending secure information? You could do a lot worse than employing the RSA algorithm. Credit: Seq

A couple of days ago on The Conversation, I set myself up with a task: to defend the usefulness of so-called "useless" maths. Today, that defence continues, with a look at the RSA algorithm.

I finished last time by pointing out that three mathematicians – Ron Rivest, Adi Shamir and Leonard Adleman – created the RSA algorithm in 1977, in one fell swoop establishing a practical use for number theory in the modern world.

So, to look at this idea more closely, let's take a detour into the lives of Alice and Bob, two fictional characters who are infatuated with each other, despite never having met.

Suppose Alice wants to send Bob a private note outlining her affections. She could place this in a box, snap a padlock on it and ship it off. The problem is, obviously, that this would be useless to Bob without the key.

Alice could send this key across, but if both were intercepted (possibly by Bob's jealous ex-wife, Eve) there would be trouble – and the fact this could happen means the method isn't really secure.

One solution to this problem runs in parallel with the RSA algorithm.

Bob knows people (Alice, in particular) want to send him secret messages, so he goes out and buys a stack of identical padlocks, all of which open with a single key he keeps hidden in his left shoe. Bob unlocks all of these padlocks and makes them available at, say, Bunnings.

If Alice wants to send Bob a secret message, she simply needs to go to Bunnings and get one of these open padlocks, then use it on the box she wants to send Bob. This will be a safe transmission, seeing as Bob is the only one with the key.

But what a complicated way of securing information: for people to receive secret messages, they need to have public padlocks available to everybody!

These days, of course, more people opt for email rather than snail mail. So how do we know Alice and Bob's online communications aren't being monitored by Eve?

Now we get into the RSA algorithm, which is the strongest possible way to encrypt and decrypt information online.

Feel the rhythm

The algorithm's design and strength are the work of historic results in number theory, and its security is guaranteed by the following fact:

It's easy for a computer to multiply two large prime numbers together (Google will do this without breaking a sweat).

But let's say you multiply two large prime numbers together to get a resulting number: if you gave this new number to a computer and asked it to tell you what prime numbers you multiplied to construct it, the computer would struggle.

This is called a trapdoor, meaning it's easy to go one way, but very hard to go the other.

Which two prime numbers did I multiply together to get 194477? A computer can probably unlock this number easily, but not if the prime numbers I use are much larger.

The mathematical difficulty of the above problem is what ensures the strength of our encryption (or lock).

Crunching the numbers

The RSA algorithm works as follows:

First, I find two huge (at least 100 digits each!) prime numbers p and q, and then I multiply them together to get the even bigger number N. I also combine p and q in a different way to generate another number e (details of this below).

I publish these two numbers (N,e) as my "public key", with the knowledge that there is enormous difficulty with even the world's fastest computers breaking N into its constituent prime atoms p and q.

It turns out the numbers N and e can be used by people to encrypt a message to send to me, which I can use my secret primes p and q to decrypt.

To illustrate this beautiful piece of mathematics, we can return to our fictional lovers.

Bob wants to encrypt and send Alice his age – 42. Of course, the RSA algorithm deals with sending numbers, but seeing as any text can be converted to digits in a variety of ways, we can securely transmit information of any type.

Also, I will use small prime numbers in this example, just to ease the calculations.

1) Alice knows that Bob wants to send her a message, so she selects two prime numbers. Let's say she picks p=17 and q=29 (though in reality they would be much larger so as to ensure better security).

Alice then multiplies p and q together to get the number N:

$p \times q = 17 \times 29 = 493$

So Alice now has that N=493.

2) Alice also needs to generate another number e. She creates this

number first by subtracting 1 off of each of her prime numbers p and q. This gives her:

p – 1 = 16
q – 1 = 28

Alice then multiplies these two together to get:

(p-1) x (q-1) = 448.

This number is not e. The number e is allowed to be any number, which has no factors in common with this new number 448. To see what I mean, we can break 448 into a bunch of prime numbers multiplied together:

448 = 2 x 2 x 2 x 2 x 2 x 2 x 7

Then e is just any number which, when broken down into primes, does not possess a 2 or a 7 as a factor. So there are lots of possibilities. Let's suppose Alice chooses e=5.

3) Alice then gives the numbers N=493 and e=5 to Bob. She could even place them on the local billboard if she wanted to. The important part is that the number N should be hard for anybody to break down into p and q.

We will just assume the number 493 scares people and so nobody tries to decompose it.

4) With these two numbers N and e, Bob can now encrypt his secret message, which, in this case is his age – 42. He starts by putting 42 to the power of e, which he knows is 5. This just means he multiplies his age by itself five times:

42 x 42 x 42 x 42 x 42 = 130,691,232

5) Bob has now half-locked his message. At this point, he just needs to use N to finish the encryption process. He takes the above number 130,691,232 and divides it by the number N=493.

He is interested in the remainder upon performing this division, which is 383. So Bob has encrypted 42 as 383, which is the number that he sends to Alice.

6) Now, the messy bit. Alice has received the number 383 from Bob, and she needs to decrypt it to get his age. Her first step, is to use her secret prime numbers p and q and the public number e to form another number d, which she can use to decrypt Bob's message.

Alice once again considers the number 448, which she obtained in step 3 by multiplying (p – 1) by (q – 1). Alice needs to find a multiple of e=5 which is exactly one more than a multiple of 448.

Number theory grants that this is always possible and we will show a tedious but straightforward way of doing this.

Alice lists the multiples of 5:

5, 10, 15, 20, 25, 30, 35, 40 …

… and the multiples of 448:

448, 896, 1344, 1792, 2240, 2688 …

She needs to find a number in the first sequence which is exactly one more than a number in the second sequence. If Alice goes far enough along in the first sequence – to the 269th term which is 1345 – she can see that this is exactly one more than 1344, the third term in the second sequence. Success!

Note that it was finding the number d=269, the place of the relevant multiple of e, which was the whole point of this step.

7) Finally, Alice can decode the message by doing one big calculation. She has Bob's encrypted number, 383, and her new number, d=269. It turns out Bob's age can be decoded by calculating 383 to the power of 269, and then finding the remainder upon dividing by N=493.

This looks like it will be quite a nasty calculation, but some tools from number theory can be employed here. Once Alice completes this calculation, the remainder will be 42, which is Bob's age.

RSA is the best possible type of public key cryptography, yet due to the high computation involved it is often not used to encrypt/decrypt simple messages.

Instead, it's used for signatures and protocol negotiations to allow two sides to receive private keys to use for their communication.

The RSA algorithm is but one of many systems where a set of mathematical theorems, often from number theory, can be synthesised to construct an encryption scheme.

There is some elegant mathematics going on behind the scenes ensuring the algorithm's success. Of course, in reality, large prime numbers guarantee secure encryption, while the calculations are performed by computers.

Importantly, the algorithm parallels the physical situation where one makes open padlocks available to the public.

Mathematically speaking, our numbers N and e form the open padlock, and our secret primes p and q combine to form a key that we can use to retrieve the locked information.

But if somebody came up with a brilliant formula for breaking numbers into their prime factors (turning padlocks into keys), there would indeed be security issues around the world.

I rest my case

The inherent randomness of the primes is the force that currently prevents such a formula from existing.

Number theorists are at the forefront of this knowledge, but are still a while away from coming up with such a tool.

The world should rest assured that, as problems are solved, many more are created, and this progress not only allows the advancement of technology, but also endows the mathematician with endless motivation.

Just because something doesn't have real-world applications yet doesn't mean it won't have in the future.

And so concludes my defence of the usefulness of useless maths. Did it work? Are you convinced? I'd be interested to hear your views.

This story is published courtesy of The Conversation *(under Creative Commons-Attribution/No derivatives).*

Source: The Conversation

Citation: The RSA algorithm (or how to send private love letters) (2013, April 10) retrieved 23 April 2024 from https://phys.org/news/2013-04-rsa-algorithm-private-loveletters.html