Breaking RSA

The weak point of the RSA algorithm lies in the key generation: specifically, the difficulty of factoring \(N \) into its prime components. This factorisation is only possible through trial and error, and the algorithm remains secure until someone discovers a heuristic method to do it efficiently.

You wouldn't believe how tricky it is to find \(p \) and \(q \) when \(N = 77 \). Once you've figured it out, think about how you did it. What was the process? How did you come up with the solution?

The RSA Labs used to offer $200,000 for the factorization of 2048-bit numbers. However, they removed the prize because more recent research has shown that the method can be broken if the primes \(p \) and \(q \) have a special relationship.

The RSA Labs previously offered a $100,000 prize to anyone who could submit the prime factors of the following integer \(N \), that is, the two primes whose product is \(N \):

```
N = p x q = 25195908475657893494027183240048398571429282126204
032027777137836043662020707595556264018525880784406
918290641249515082189298559149176184502808489120072
844992687392807287776735971418347270261896375014
9718246911650776133798590957000973304597488084284017
974291006424586918171951187461215151726546322822
168699875491824224336372590851418654620435767984233
871847744479207399342365848238242811981638150106748
104516603773060562016196762561338441436038339044
149526344321901146575444541784240209246165157233507
787077498171257724679629263863563732899121548314381
678998850404453640235273819513786365643912120103
97122822120720357
```

You can sense how difficult this task is: we are dealing with 617 digits, and unfortunately, the last digit is not even.

Example: Breaking RSA if we find \(p \) and \(q \)

Step 1: Given Data

Suppose we know the **public key** \((e, N) \):

- Public exponent \(e = 17 \)
- Modulus \(N = 55 \)

To break the RSA encryption, we need to find the **private key** \(d \). This requires us to factor \(N \) into its prime factors \(p \) and \(q \).

Step 2: Factor \(N \) We need to factor \(N = 55 \):

- \(p = 5 \)
- \(q = 11 \)

These are the two prime factors of \(N \).

Step 3: Calculate \(\phi(N) \) (Euler's Totient Function)

Once we have \(p \) and \(q \), we can compute \(\phi(N) \), which is required to compute the private key \(d \).

\[\phi(N) = (p - 1) \times (q - 1) \] \[\phi(55) = (5 - 1) \times (11 - 1) = 4 \times 10 = 40 \]

Step 4: Find the Private Key \(d \)

The private key \(d \) is the modular inverse of \(e \mod \phi(N) \), which satisfies the equation:

\[d \times e \equiv 1 \mod \phi(N) \]

We need to find \(d \) such that:

\[d \times 17 \equiv 1 \mod 40 \]

Using the extended Euclidean algorithm, we find that:

\[d = 33 \]

So the **private key** is \(d = 33 \).

Step 5: Decrypt the Ciphertext

Now that we have \(d \), we can decrypt the ciphertext. Suppose the ciphertext is \(C = 18 \).

To decrypt, we use the formula:

\[m = C^d \mod N \] \[m = 18^{33} \mod 55 \]

To compute this efficiently, we can use modular exponentiation:

\[18^{33} \mod 55 = 2 \]

So, the original **message** \(m = 2 \).

Summary:

- We started with the public key \((e = 17, N = 55) \) and a ciphertext \(C = 18 \).
- After factoring \(N \) into \(p = 5 \) and \(q = 11 \), we calculated \(\phi(N) = 40 \) and found the private key \(d = 33 \).
- Using \(d \), we decrypted the ciphertext \(C = 18 \) to recover the original message \(m = 2 \).

This example illustrates how RSA encryption can be broken if someone manages to factor \(N \) into its prime factors \(p \) and \(q \).

From:
https://edu.iit.uni-miskolc.hu/ - **Institute of Information Science - University of Miskolc**

Permanent link:
https://edu.iit.uni-miskolc.hu/tanszek:oktatas:techcomm:breaking_rsa?rev=1728313930

Last update: **2024/10/07 15:12**