

## Re: Faster way to use RSA...

**Source:** <http://www.derkeiler.com/Newsgroups/sci.crypt/2003-03/2293.html>

---

**From:** Scott Fluhrer ([sfluhrer@ix.netcom.com](mailto:sfluhrer@ix.netcom.com))

**Date:** 03/31/03

From: "Scott Fluhrer" <[sfluhrer@ix.netcom.com](mailto:sfluhrer@ix.netcom.com)>

Date: Sun, 30 Mar 2003 16:51:03 -0800

"Beta-Ray Bill" <[nSePtAnMeSwUsC@KaSttbi.com](mailto:nSePtAnMeSwUsC@KaSttbi.com)> wrote in message  
news:DeJha.266963\$3D1.144923@scrnsc01...

> *Ease of attack for RSA is based on not being able to factor  $N$  ( $N = P \times Q$ )  
due*

> *to  $N$  being known to all (e.g.  $(e, N)$  is my private key and  $(d, N)$  being my  
> public  
> key).*

Actually, I believe that the more normal nomenclature is for  $e$  to be the  
public exponent and  $d$  to be the private exponent. I'll follow your  
terminology, this comment is for your general enlightenment.

> *As computers get faster RSA users must select  $P$  and  $Q$  values that are  
larger*

> *to remain safe( $r$ ).*

>

> *My questions are as follows:*

>

> *1. If  $N$  could be kept secret (e.g. just between the people who  
communicate)*

> *then would RSA key sizes stop increasing?*

If  $N$  is secret, then what is the advantage of RSA over a symmetric key  
system (such as AES)? If speed is what you're worried about, symmetric key  
systems are \*much\* faster.

>

> *2. Can cryptanalysis determine the value of  $N$  from analysis of the*

> *ciphertext without knowledge of the public key, which of course  
contains*

>  *$N$ ?*

> *From the ciphertext, no. The ciphertext of random plaintexts has a uniform  
distribution between 0 and  $N-1$  — he'd be able to guess that  $N$  is somewhat  
larger than the largest ciphertext he's seen, but he wouldn't be able to  
deduce anything more than that.*

If he had a small set of plaintexts and corresponding ciphertexts, then yes,  
if  $d$  isn't too large. In particular, for any particular plaintext  $P$  and

corresponding ciphertext C, we have:

$$P^{**d} - C = 0 \text{ mod } N$$

That is, if we have two different plaintexts P1, P2 and ciphertexts C1, C2, then:

$$kN = \text{gcd}(P1^{**d} - C1, P2^{**d} - C2)$$

where k would appear likely to be small.

Note that this observation requires the exact value of P1, P2, C1, C2 given to the RSA algorithm -- if some randomized padding is used, then the validity of this attack assumes that the attacker can (somehow) reconstruct that padding. In addition, this assumes that d is small, so that P\*\*d can be reasonably expressed. This attack also assumes that d can also be guessed by the attacker -- however, the requirement that d be small covers that, as the attacker can just try all small d's.

- > *Why doesn't someone select smaller P and Q values, which result in faster encryption, and generate e and d. Encrypt a message using smaller P and Q, e*
- > *and d. Bob*
- > *could use RSA in the standard way (e.g. with large P, Q and large N) to send*
- > *smaller P and Q, e*
- > *and d to Alice. Bob would also send the encrypted message as well. Alice*
- > *would reverse the*
- > *process. Wouldn't the attacker then have to brute force pick smaller P and Q*
- > *pairs*
- > *and for each further test multiple e and d pairs to find solutions?*

What would the advantage of that being over using the large P, Q, N to send over an AES key?

- >
- > *After all, RSA is a very clever way to reconstruct a value from knowing its*
- > *remainder and the decryption key - of course.*
- >
- > *I didn't see this in the FAQ.*

Maybe it isn't all that frequently asked...

--  
poncho