

Forum:[Hack Forums](#)

Thread: [RSA: Getting private key from public key \(Triviador\)](#)

Xapesy - December 19, 2017, 4:10 am

Hi, I have a public key (so, public exponent and modulus) and I need private key.

I was thinking of bruteforcing but couldn't find any good script (if you know one, I'd be grateful if you posted a link). The private key I am trying to get is used in a game Triviador to encrypt packets that contain questions and answers.

The modulus and public exponent are:

21d_713c41_f170_97aec6_ab97_9d8a28d_4ccfd7_3dc88_5a31199e_89157003_3e0a1651cc_acc
28_3f241c_17ac1f702c_1ed7cfeeca_2e51985_1f0e2c_e97f4c_abdee_3fef67 (Remove the
underscores)
10001

There's also a reward (yes, real money) if you figure out private key. I can provide decompiled AS3 scripts if it helps. If you have any other questions, feel free to ask me.

Thanks!

pussyporn - December 19, 2017, 4:19 am

its not fucking possible to get a fucking private key from a public key are ou on fucking cocaine

CyberStrike - December 19, 2017, 6:04 am

OP it's doable you have the two important pieces of information I am sure that you know that since you posted them.

Xapesy - December 19, 2017, 8:36 am

Hi, thanks for replying. I'm not really good at cryptography, so I'll need some pointers.

Assuming we can't get the private key (I don't know if it's really possible by only using modulus and public exponent, but correct me if I am wrong). I've read on Wikipedia that RSA is a deterministic algorithm, which means given the input, the sequence is determined?

For example (taken from Wikipedia):

Attack at dawn -> Nggnpx ng qnja

We can see here that A -> N is through whole plaintext. Now, if I have different plaintext, will A change to N on that one too?

If that's the case, I won't even need private key at all, because only 32 bytes (so 32 numbers, each is random from 0 to 255) get encrypted.

If anyone has a better idea on how to decrypt the this, please reply.

Thanks.

Yattaze - December 20, 2017, 3:42 pm

It is possible to break RSA encryption, however, it is verrry time-consuming to do so, you could spend the rest of your life running

your cracking program and you still wouldn't get the private key in time.

I suggest you read: <http://www.mathaware.org/mam/06/Kaliski.pdf> if you're really interested in how it works.

But in short, it uses a "Trapdoor" function defined as "A trapdoor function is a function that is easy to compute in one direction, yet

difficult to compute in the opposite direction (finding its inverse) without special information" notice how it says "difficult" and not impossible.

If you were to do about cracking your Public Key you need two things:

The modulus, well give it the letter "n"

and the exponent, "e"

and well give the decryption exponent the letter "d"

You can "break" RSA by knowing how to factor "n" into its "p" and "q" prime factors:

$$n = p * q$$

The easiest way is probably to check all odd numbers starting just below the square root of n:

$\text{Floor}[\text{Sqrt}[n]] = 0$ Or when: " $n \bmod x = 0$ " you know that " $p = x$ " and " $q = n / p$ "

Why?, because d is a special number such that

$$d = e^{-1} \bmod \phi(n)$$

$$= e^{-1} \bmod (p-1)*(q-1)$$

And that's how you would go about breaking it, keep in mind that this takes longer than the heat death of the universe for long rsa keys

[EDIT] 100th post!

Xapesy - December 21, 2017, 1:57 am

Hi, thanks for the reply and congratulations for your 100th post!

I used Python to convert the modulus to actual integer (using `int(key, 16)`), so:

n =

1772352223689553499523382657591107586734440334822954296365208356693849378633858
324876899946660923527875550194354495596668114847388151261943936272297881447

e = 65537 -- Not needed until I find factors, I guess

$n \bmod x = 0$ and $x = p$ then this is:

$n \bmod p = 0$? If I find p then it's easy to find q with $q = n / p$

But why does it seem that this is too easy? If you know any Python script that searches for factors, let me know.

If I find both factors (I already searched factordb to see if I could make this easier, but no such luck), then it's easy to calculate $d = e^{-1} \bmod (p-1)*(q-1)$

I'll try to write a script.

Yattaze - December 21, 2017, 6:17 am

It seems easy until you remember that RSA uses the principle that prime numbers are easy to add together

but its hard to calculate the factor of the product of two primes

Xapesy - December 21, 2017, 7:26 am

Yeah, just found out how hard it is. Took me 10 minutes for checking the p with the Python script I made, which is really inefficient - that's why I would like to find a script that is more efficient with checking.

I found a method called Number Field Sieve algorithm. I cannot post the link since I don't have 50

posts yet, but search for "gilchrist ca nfs guide" and you should find it. It says that it takes months to factor a 155 digit number with GNFS. Is that possible? Or is this the wrong place to search for a script that finds factors?

Yattaze - December 21, 2017, 3:15 pm

Try something like:

```
import math
x = Floor[Sqrt[n]]
c = 0
while (true):
    if n % x == 0:
        print(x)
        break
    c += 1
```