

Re: The crazy encryption madmans codebook

Source: <http://www.derkeiler.com/Newsgroups/sci.crypt/2007-03/msg00138.html>

- *From:* jt64@xxxxxxxx
 - *Date:* 4 Mar 2007 04:40:45 -0800
-

On 4 Mar, 01:55, "Joseph Ashwood" <ashw...@xxxxxxxx> wrote:

<j...@xxxxxxxx> wrote in message

news:1172925208.343013.187300@xx

I will try to be **a little** more precise if you lay out the text little more.

It certainly would be easier if you told me what k and i, i guess k is keyword and i is integer or index?

Example: Suppose database ranging 0-5 000 000 indexed word and phrases where each entry have an index, realworld word or phrase and a madman word or phrase.

Which gives you 22 bits of possible block, at which point I can easily declare it broken.

However it would not be a good idea to use an algorithm that can be reversed to find the **offset text key**.

Quite the contrary with such small numbers it is critical that it be reversible. Without it being a reversible process you will have collisions and misses, this results in bias, and compromises security even further.

So we would have to find an approach/algorithm that do not compromise the text "characters/ letters" when used off course you could use a hash algorithm that put out values with a range from 0-5 000 000. Because we want collisions in the algorithm. An easy way to get collisions is by using anagrams. By let the characters be multiplied with eachoter, we assure that if the key is 6 letter we have 6! of equivalent values that actually will give same result.

Re: The crazy encryption madmans codebook

This is why you should leave the designs to professionals, you have added a huge amount of complexity specifically to weaken the system.

Suppose the encoding key for the specific phrase "this is big" is letter of course you should not pick characters next to each other but this is an example.

l*e*t*t*e*r=r*e*l*e*t so for any hashed entry we infact have 6! possible keys before applying the hash that put out the hashvalue for the database.

So you've less than 10-bits available, lets see, on a 1GHz machine (we all agree this is out of date) it should take 0.0000007 seconds to break.

You can of course let the hash but out bigger values then 5 000 000 and search the database circular this will make things even harder.

It doesn't, see my earlier proof, in the first or second response I gave you I detailed why this simply isn't the case. I used a design that was a superset of yours, showed it's equivalence to known cryptographic quantities, which in turn lent itself to relying on well established proofs that were very specific that the concept for the system is flawed.

I hope i've been clear so far.

You've been clear enough all along, now if you would understand the argument that has been presented, you'd understand that this design cannot be made secure, it can only be made slower.

I basically say that i can encrypt the word *telephone* to have any *madman phrase* let us say have telephone have the indexvalue 3456789 in the database.

And it has been shown to you multiple times that this is completely false, the entire premise of your security is "Multiplication is secure" it is not, and has not been considered secure since before anyone alive was born.

Since this is a phrase database we say average phrase to encode 16 letters, this will give us 16 bytes of keymaterial for each key (how we pick the keyletters from the decoded text is not that important).

Re: The crazy encryption madmans codebook

So the entire system has an absolute maximum key entropy of 64-bits, and because you are limiting yourself to phrases in a given language you are discarding almost all of those resulting in an approximate entropy of 20-25 bits, we're back to that 0.00000 time frame for a break.

I hope the idea is pretty clear now Joe.

It has been clear all along. It has also been clear all along that you cannot escape the argument I laid out early in this conversation which provides the maximum security for the system, and that security is very small.

Joe

Well Joe I never said that the database only had 5 000 000 entries it was an example, and if the encrypted codebooks used NG on internet really had just 10 bit of entropy why don't you break 'it's a miracles codebook' there are a lot of messages in sci.physics relativity.

"But the truth is you can not, nobody can".

I think you know that you can not because the algorithm used is an expanded key that add keymaterial all time. The actual language used has as much entropy that there is entries in the database.

And that is why you really fear this approach to cryptography, it is your buisness and these type of algorithms rends your knowledge of attacks useless.

And that is why you try to very fast discard the whole approach as useless Joe.

Best regards Jonas Thörnvall

.