Example 118. To (naively) brute-force DES, how much data must we encrypt?

Solution. By brute-forcing, we mean that, given a pair of 64-bit blocks m,c, we go through all 2^{56} possibilities (DES uses 56-bit keys) for k and look for k such that $E_k(m)=c$? We need to encrypt 2^{56} times 64 bits.

This is $2^{56}\cdot 8=2^{59}$ byte, or 512 pebibyte (binary analog of petabyte) or 576 petabyte (since $2^{59}\approx 5.76\cdot 10^{17}$).

How long will this take? Of course, this depends on your machine. Assume we are able to encrypt 1 GB/sec. Then, this will take us about $5.76\cdot 10^8$ sec, or about 18.3 years.

Of course, such a brute-force attack can be fully parallelized to quickly bring this number down to less than an hour for a powerful attacker. Also, the attack can be sped up considerably by careful design (like early aborts).

For comparison. Though mostly of theoretical value, for DES, some possibilities for attacks better than brute-force are known: for instance, as of 2008, linear cryptanalysis can mount an attack with 2^{43} known plaintexts in about 2^{40} (instead of 2^{56}) steps.

Example 119. (bonus challenge) Using DES, are there blocks m,c such that $E_k(m)=c$ for more than one key k?

This would mean that, using just a single plaintext-ciphertext pair, the above brute-forcing might uncover more than one possible key.

[I don't know the answer (but expect that it is "yes") and couldn't find it easily. Maybe you are more skilled?]

Example 120. (3DES) A simple approach to increasing the key size of DES, without the need to design and analyze a new block cipher, is **3DES**. It consists of three applications of DES to each block and is still considered secure.

$$c=E_{k_3}(D_{k_2}(E_{k_1}(m)))$$

The 3DES standard allows three keying options:

- k_1,k_2,k_3 independent keys: $3\times 56=168$ key size, but effective key size is 112
- $k_1=k_3$: $2\times 56=112$ key size, effective key size is stated as 80 by NIST
- $k_1=k_2=k_3$: this is just the usual DES, and provides backwards compatibility (which is a major reason for making the middle step a decryption instead of another encryption).

Comment. The reason for the reduced effective key sizes is the meet-in-the-middle attack. It is also the reason why something like 2DES is not used. See next example!

Comment. NIST approved 3DES until 2030 for sensitive government data.

Example 121. (no 2DES) Explain why "2DES" does not really provide extra security over DES.

Solution. Let's denote DES encryption with E_k and decryption with D_k. The keys k are 56 bits.

Then, 2DES encrypts according to $c=E_{k_2}(E_{k_1}(m))$. The key size of 2DES is $56+56=112$ bits.

- A brute-force attack would go through all possibilities for pairs (k_1,k_2), of which there are $2^{56}\cdot 2^{56}=2^{112}$, to check whether $c=E_{k_2}(E_{k_1}(m))$. That requires 2^{112} 2DES computations.
- On the other hand, note that $c=E_{k_2}(E_{k_1}(m))$ is equivalent to $D_{k_2}(c)=E_{k_1}(m)$.

  Assuming sufficient memory, we first go through all 2^{56} keys k_2 and store the values $D_{k_2}(c)$ in a lookup table.

  We then go through all 2^{56} keys k_1, compute $E_{k_1}(m)$ and see if we have stored that value before. (Even though this is a huge table, the cost for checking whether an element is in the table can be disregarded; thanks to the magic of hash tables!)

  Comment. In this second step, we see that m and c should be more than one block (otherwise we get too many candidate keys $k=(k_1,k_2)$).

  The total number of DES computations to break 2DES therefore is $2^{56}+2^{56}=2^{57}$, which is hardly more than for breaking DES!

This is known as a meet-in-the-middle attack.

https://en.wikipedia.org/wiki/Meet-in-the-middle_attack

Comment. The price to pay is that this attack also requires memory for storing This sort of approach is referred to as a time-memory trade-off. Instead of brute-forcing 2DES in 2^{112} steps, we can attack it in 2^{57} steps while storing 2^{56} values of the size of m.

Comment. This applies to any block cipher, not just DES!

Comment. For some block ciphers it is the case that for all pairs of keys k_1, k_2, there is a third key k_3 such that $E_{k_2}(E_{k_1}(m)) = E_{k_3}(m)$. In that case, we say that the cipher is a group, and double (or triple, or quadruple) encryption does not add any additional security! DES, however, is not a group.

Example 122. Explain why 3DES, used with three different keys, only has effective key size 112.

Solution. (fill in the details!) Instead of going through all k_1, k_2, k_3 to check whether

$$c = E_{k_3}(D_{k_2}(E_{k_1}(m)))$$

(which would take $2^{56} \cdot 2^{56} \cdot 2^{56} = 2^{168}$ DES computations), we can use that the latter is equivalent to

$$D_{k_3}(c) = D_{k_2}(E_{k_1}(m)).$$

Now proceed as in the previous example $\ldots$ to see that we can break 3DES with $\sim 2^{112}$ DES computations. How much memory do we need?

Example 123. (extra; use as PRG) ANSI X9.17 is a U.S. federal standard for a PRG based on 3DES.

Input: random, secret 64 bit seed s, key k for 3DES (keying option 2)

Produce a random number as follows:

- obtain current time D, compute $t = 3\mathrm{DES}_k(D)$
- output $x = 3\mathrm{DES}_k(s \oplus t)$ (that's the pseudo-random output)
- update the seed to $s = 3\mathrm{DES}_k(x \oplus t)$ for future use

Comment. ANSI (American National Standards Institute) X9 are standards for the financial industry.

https://en.wikipedia.org/wiki/Cryptographically_secure_pseudorandom_number_generator

Comment. The same approach can be applied to any block cipher.

Comment. It is common practice to add in time for PRGs that are used to generate enormous amounts of data. If nothing else, it slightly increases the entropy and reduces the likelihood of "short" periods.

Example 124. (extra; DES-X) To increase the key size of DES, the following variation, known as DES-X, was proposed by Ron Rivest in 1984:

$$c = k_3 \oplus \mathrm{DES}_{k_2}(m \oplus k_1)$$

What is the key size of DES-X? What about the effective key size?

Solution. k_1 and k_3 are 64 bit, while k_2 is 56 bits. That's a total key size of 184 bits for DES-X.

However, just like for 3DES, proceeding as in a meet-in-the-middle-attack (without the need of much storage) reduces the effective key size to at most $184 - 64 = 120$ bits.

Comment. This approach of xoring with a subkey before and after everything else is known as **key whitening**. This features in many modern ciphers, including AES.

https://en.wikipedia.org/wiki/DES-X