Example 56. A ciphertext only attack on the one-time pad is entirely hopeless. Explain why!

Solution. The attacker only knows $c = m \oplus k$. The attacker is unable to get any information on m, because every other message m' (of the right length) could have resulted in the same ciphertext c.

Indeed, if the key was $k' = k \oplus m' \oplus m$, then the encryption of m' is $m' \oplus k' = m' \oplus (k \oplus m' \oplus m) = k \oplus m = c$ as well! [Moreover, every plaintext m' is equally likely because it corresponds to a unique key.]

What about the other attacks?

Attacks like known plaintext or chosen plaintext don't apply if the key is only to be used once.

Yet, the one-time pad by itself provides **little protection of integrity**. The next example shows how tampering is possible without knowledge about the key.

Example 57. Alice sends an email to Bob using a one-time pad. Eve knows that and concludes that, per email standard, the plaintext must begin with `To: Bob`. Eve wants to tamper with the message and change it to `To: Boo`, for a light scare.

Alice sends $c = m \oplus k$. If Eve changes the ciphertext c to $c' = c \oplus e$, then Bob receives c' and decrypts it as $m' = c' \oplus k = \overbrace{\underbrace{m \oplus k}_{=c} \oplus e}^{c'} \oplus k = m \oplus e$.

- Eve wants to change the 7th letter from b to o.
- Since b is $0x62$ and o is $0x6F$, we have $b \oplus o = 0x0D$. Hence, $b \oplus 0x0D = o$.
- Eve chooses $e = 0x\underbrace{000000000000}_{6 \text{ characters}}0D00\ldots$ because then $\underbrace{\text{“TO: Bob...”}}_{m} \oplus e = \underbrace{\text{“TO: Boo...”}}_{m'}$.

Example 58. (homework) One thing that makes the one-time pad difficult to use is that the key needs to be the same length as the plaintext. What if we have a shorter key and just repeat it until it has the length we need?

That's essentially the Vigenere cipher (in a different alphabet).

Solution. Assuming the attacker knows the length of our key (if she doesn't she can just try all possibilities), this is equivalent to using the one-time pad several times with the same key. That should never be done! Even using a key twice means that we become susceptible to a ciphertext only attack (see Example 49).

So, repeating the key is a terrible idea. However, the idea to create a longer (random) key out of a shorter (random) key is not (these are pseudorandom generators, to be discussed next).

Let us emphasize that, in order to be perfectly confidential, the key for a one-time pad must be chosen completely at random (otherwise, an attacker can make assumptions on the used keys).

Indeed, the need to generate random numbers shows in every modern cipher.

3.2 Stream ciphers

Once we have a way to generate **pseudorandom numbers**, we can use the idea of the one-time pad to create a **stream cipher**.

Start with key of moderate size (say, 128 bits).

Use the key k and a PRG (**pseudorandom generator**) to generate a much longer **pseudorandom keystream** $\mathrm{PRG}(k)$. Then encrypt $E_k(m) = m \oplus \mathrm{PRG}(k)$.

We lost perfect confidentiality. Security relies on choice of PRG (must be unpredictable).

As with the one-time pad, we must never reuse the same keystream! That does not mean that we cannot reuse the key: for instance, we can prefix the key with some bits (different ones every time), called the **nonce**, then use that enlarged key and pass the nonce along with the message.

3.3 How to generate random numbers?

Natural randomness is surprisingly difficult to harness.

You can for instance play around with a Geiger counter but our department is short on these and getting lots of random numbers is again challenging.

(linear congruential generator) Let a, b, m be chosen parameters.
From the seed x_0, we produce the sequence $x_{n+1} = a x_n + b \pmod{m}$.

The choice of a, b, m is crucial for this to generate acceptable pseudorandom numbers.

For instance, glibc uses $a = 1103515245$, $b = 12345$, $m = 2^{31}$. (This is one of two implementations.) In that case, each x_i is represented by precisely 31 bits. [Note that the choice of m makes this very fast.]

Linear congruential generators are easy to predict and must not be used for cryptographic purposes. More generally, all polynomial generators are cryptographically insecure. They are still used in practice, because they are fast and easy to implement and have decent statistical properties.

Example 59. Generate values using the linear congruential generator $x_{n+1} = 5x_n + 3 \pmod{8}$, starting with the seed $x_0 = 6$.

Solution. $x_1 = 1$, $x_2 = 0$, $x_3 = 3$, $x_4 = 2$, $x_5 = 5$, $x_6 = 4$, $x_7 = 7$, $x_8 = 6$. This is the value x_0 again, so the sequence will now repeat. Note that we went through all 8 residues before repeating. Period 8.

Note. Because $8 = 2^3$ we can represent each x_i using exactly 3 bits. Then $x_1, x_2, x_3, \ldots = 1, 0, 3, \ldots$ corresponds to the bit stream $(001\ 000\ 011\ \ldots)_2$.

Example 60. (homework) Observe that the sequence produced by the linear congruential generator $x_{n+1} = a x_n + b \pmod{m}$ must repeat, at the latest, after m terms. (Why?!)
One can give precise conditions on a, b, m to achieve a full period m. Namely, this happens if and only if $\gcd(b, m) = 1$ and $a - 1$ is divisible by all primes (as well as 4) dividing m.

(a) Generate values using a linear congruential generator $x_{n+1} = 2x_n + 1 \pmod{10}$, starting with the seed $x_0 = 5$. When do they repeat? Is that consistent with the mentioned condition?

(b) What are possible values for a so that the linear congruential generator $x_{n+1} = a x_n + 11 \pmod{100}$ has period 100?

(c) As mentioned above, glibc uses $a = 1103515245$, $b = 12345$, $m = 2^{31}$. After how many terms will the sequence start repeating?

Solution.

(a) $x_1 = 1$, $x_2 = 3$, $x_3 = 7$, $x_4 = 5$. This is the value x_0 again, so the sequence will repeat. Period 4.
[The period is less than 10. This is as predicted by the mentioned condition, because $a - 1$ is not divisible by 2 and 5.]

(b) We need that $a - 1$ is divisible by 4 and 5. Equivalently, $a \equiv 1 \pmod{20}$. Hence, possible values are $a = 1, 21, 41, 61, 81$.

(c) Clearly, $\gcd(b, m) = 1$. Also, $a - 1$ is divisible by 4 (and no primes other than 2 divide m). Hence, for every seed, values repeat only after going through all 2^{31} residues.