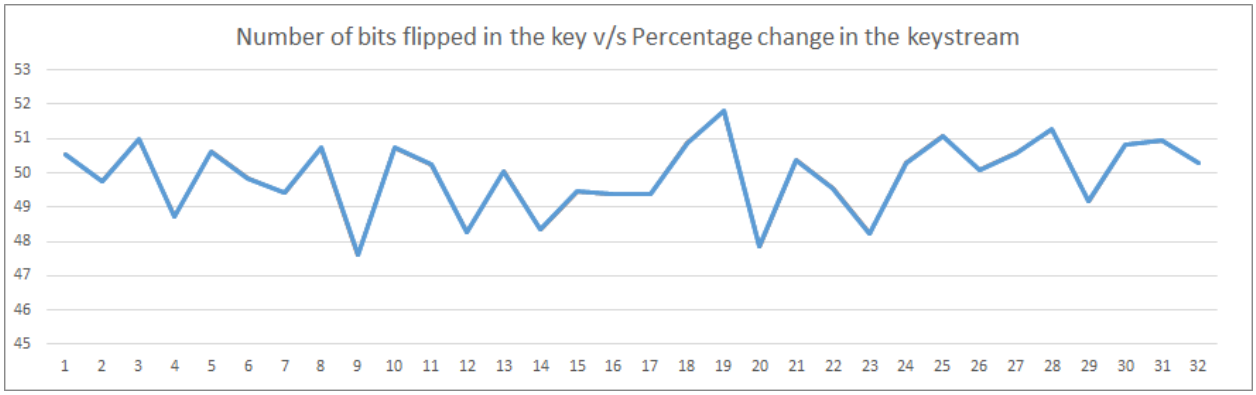
Assignment 3: Analysis of RC4 encryption algorithm

Details :

It is observed that taking a keystream1 from the RC4 by using some key and then taking another keystream2 by flipping some bits from that key and applying RC4 on it, the difference of bits in the two keystreams obtained by XORing output keystream lies between 40 to 60%. On average, it lies nearly at 50% (49.9176 %).



Analysing the differential output bits for randomness:

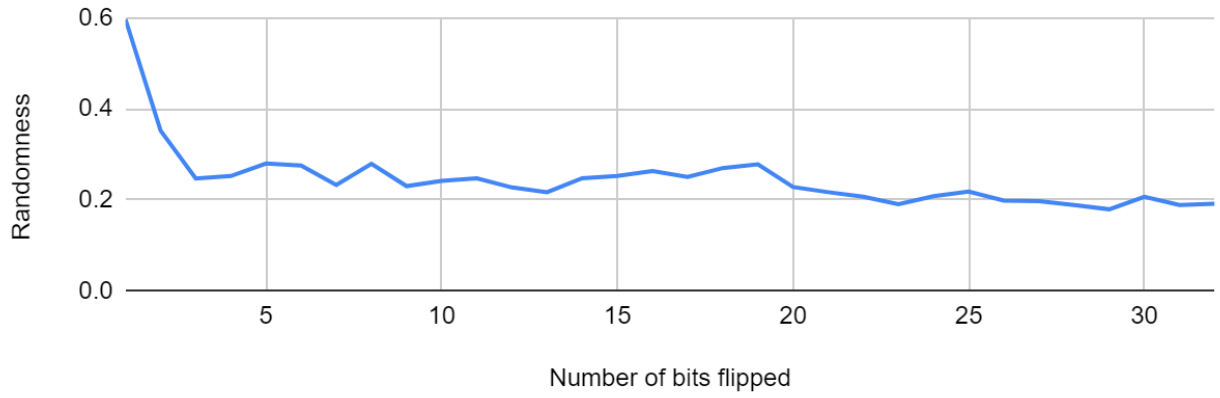
An n-bit counter is taken and then n-bit sequences from the output stream are taken and the corresponding counter is incremented along with that the account of the number of samples is also stored. As mentioned that if the two streams were different then more ones should be expected, the same was observed in the experiments.

1. n = 8(8 - bit counter)

Number of counters(C) = 2^8 (256) counters

key = “1011011010111010111010110110101110101110111110001010111110001010” (64-bits)

The randomness obtained by flipping 1 - 32 bits in the key is plotted as follows :



As we increase the number of bits flipped the randomness decreases and the rate of decreasing randomness is very steep in the beginning while it is gradual at the end.

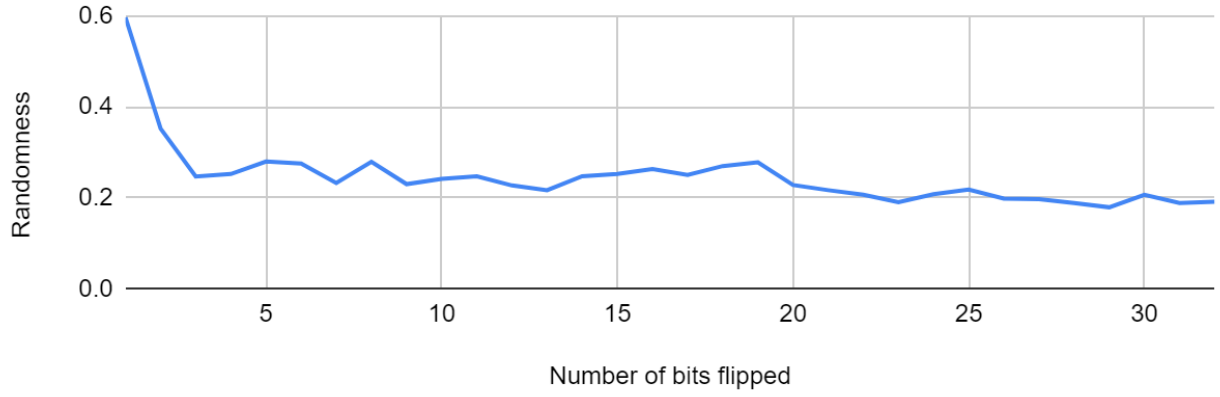
Checking the effect on randomness by changing the number of counters:

By decreasing the number of counters the randomness values will decrease as randomness is directly proportional to the number of counters.

1. n = 8(8 - bit counter)

   Number of counters(C) = 2^8 (256) counters

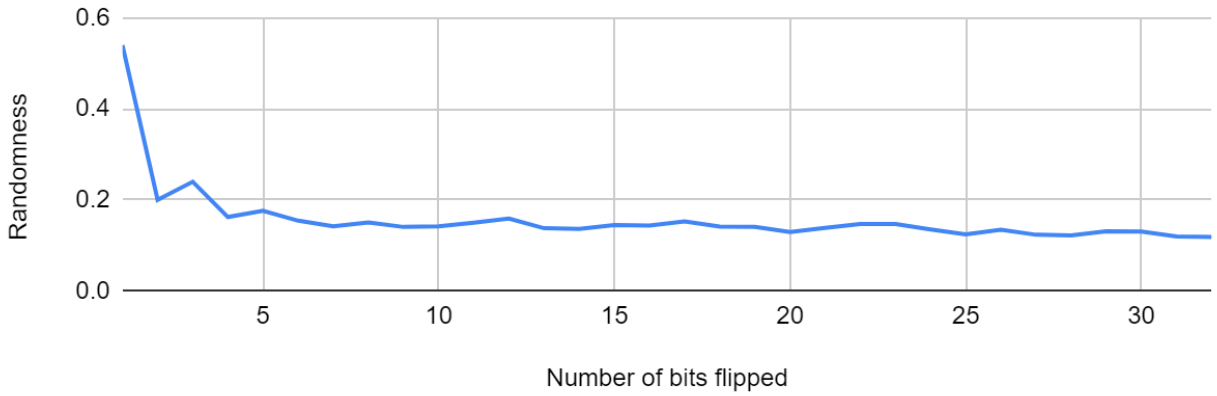
   Range of Randomness = 0.18 - 0.60



2. n = 7(7 - bit counter)

Number of counters(C) = 2^7 (128) counters

Range of Randomness = 0.12 - 0.58



It was observed that if we decrease the number of counters the range of randomness is decreasing as expected.

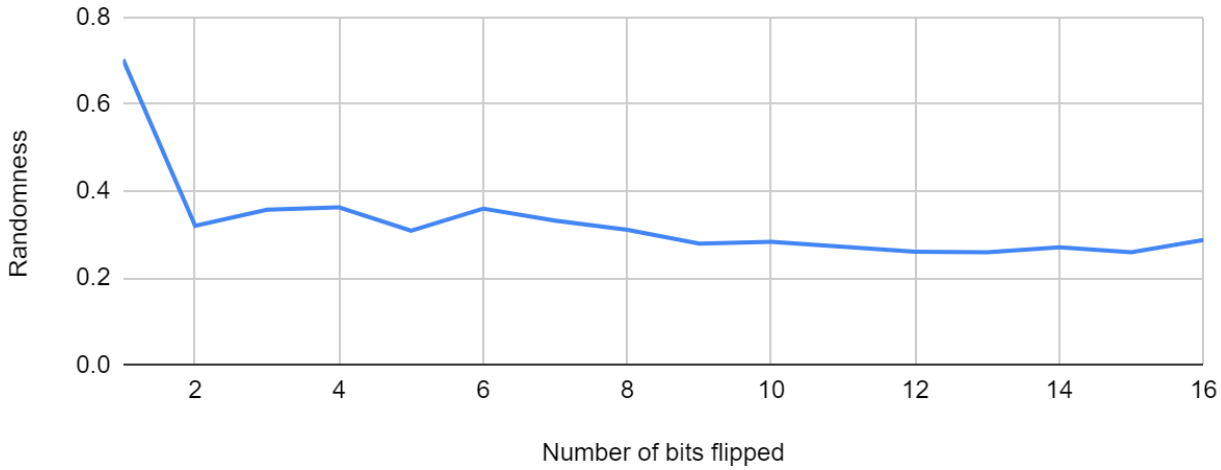
Running the randomness test, collecting the results and plotting the graphs :

The randomness test was performed for the different sizes of keystreams ranging from 4 - 1024 bytes. The key size is 64 bits for all the cases other than 4 and 8 bytes keystream case, for these two cases the key is taken of 16 and 32 bits respectively. The key bits were flipped from 1 - 32 bits for all the cases other than 4 bytes keystream case, for 4 bytes case bits were flipped from 1-16 bits and an average of 100 observations were taken for every case of flipping the bits. A length of 256 counters was taken for all the cases.

The following plots were obtained for these cases :

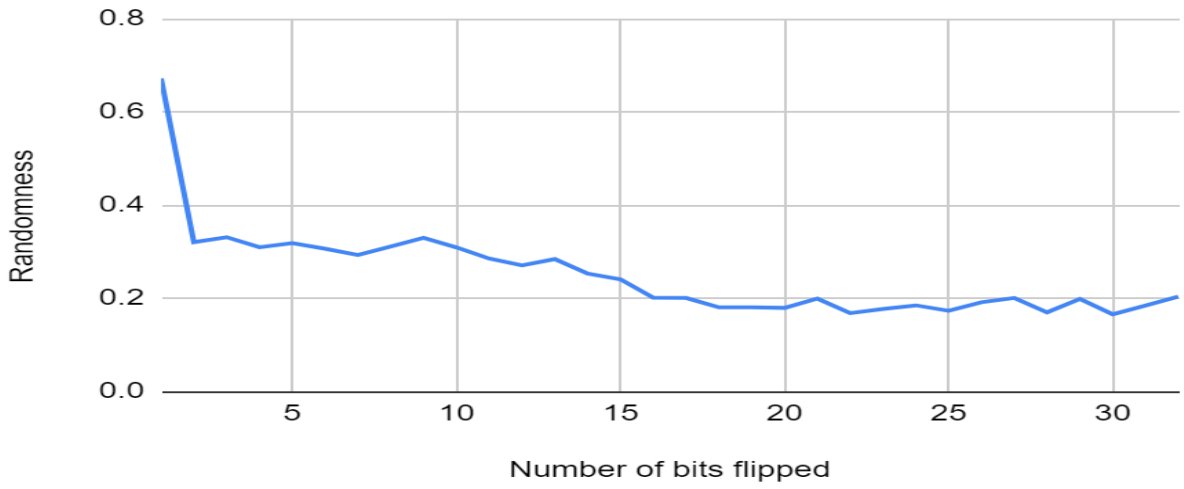
1. 4 bytes :

Range of Randomness: 0.28 - 0.7



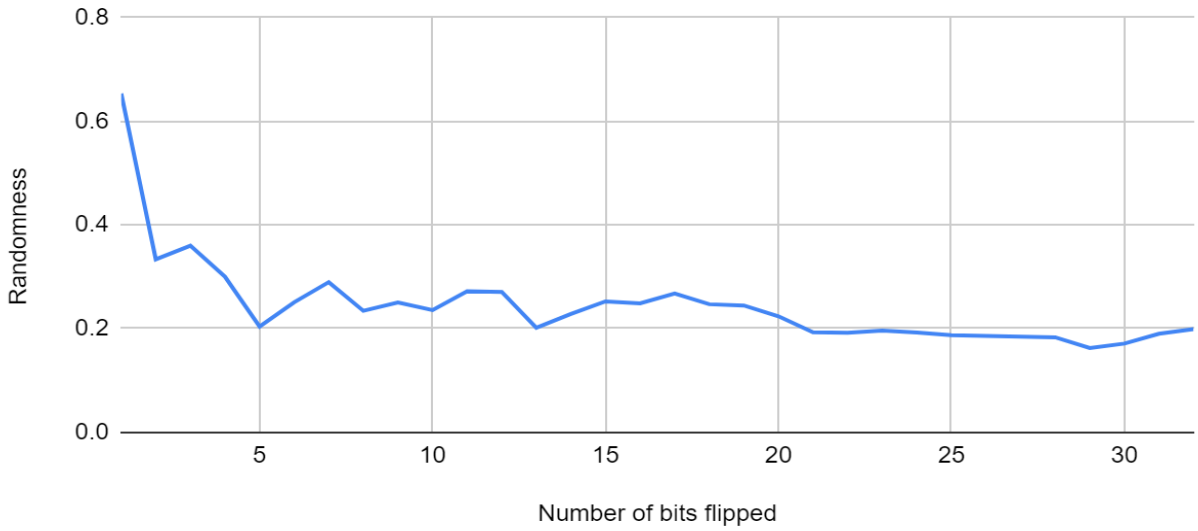
2. 8 bytes :

Range of Randomness: 0.17 - 0.68



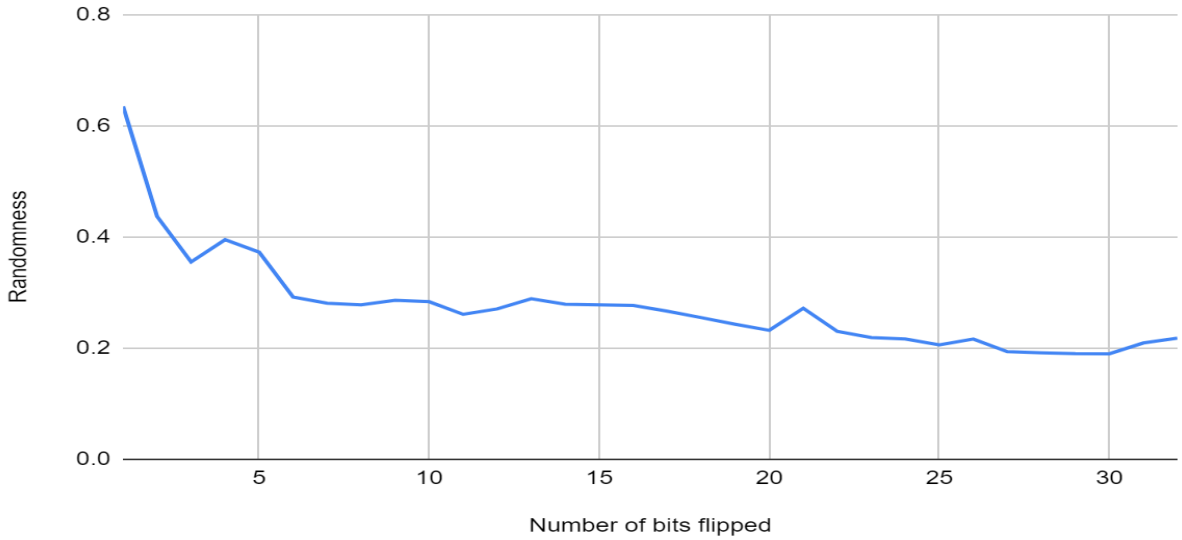
3. 16 bytes :

Range of Randomness: 0.16 - 0.65



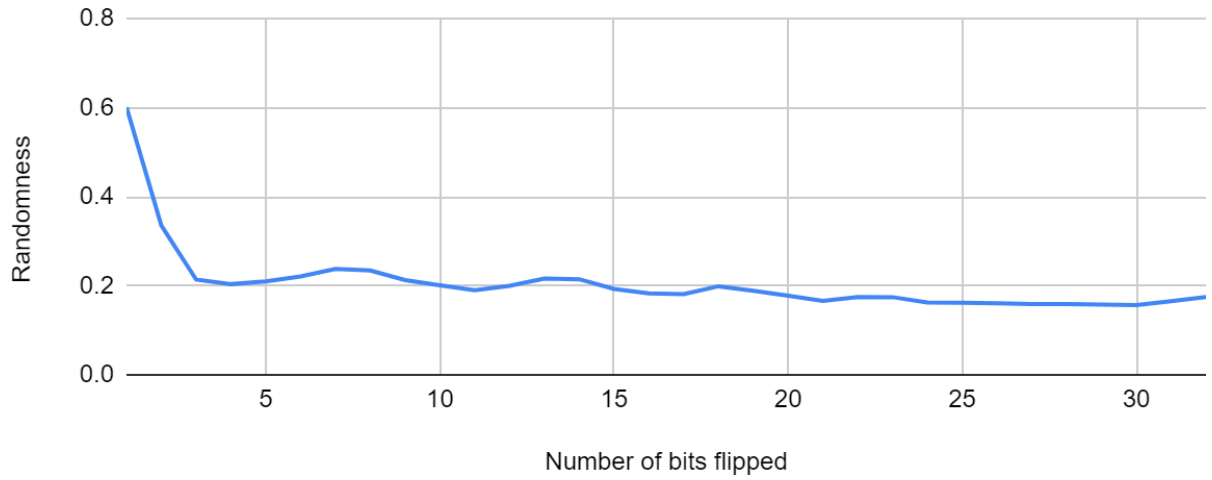
4. 32 bytes :

Range of Randomness: 0.19 - 0.63



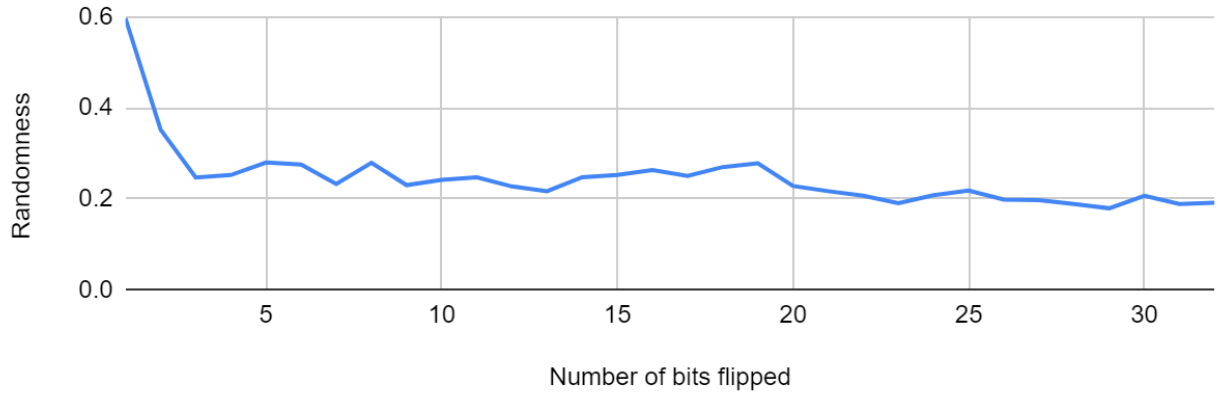
5. 128 bytes:

Range of Randomness: 0.15 - 0.61



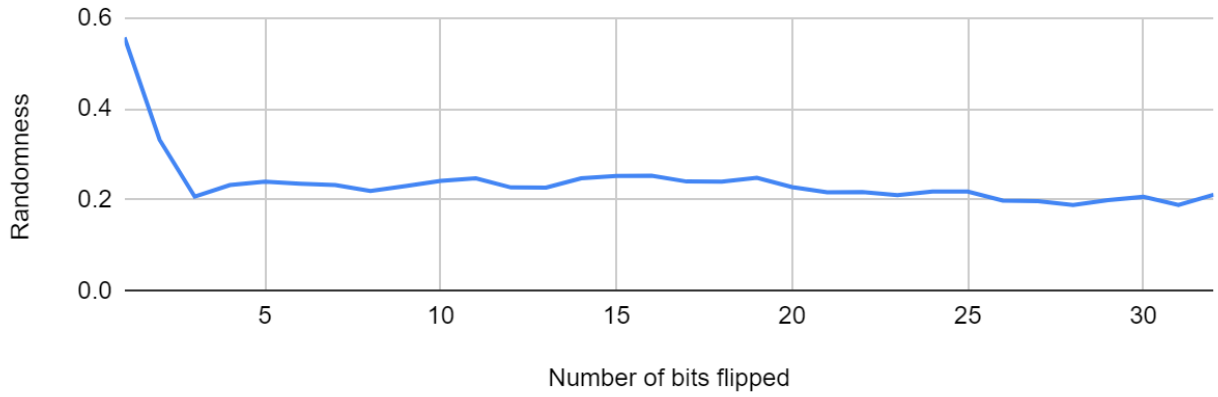
6. 256 Bytes:

Range of Randomness: 0.18 - 0.60



7. **1024 Bytes:**

Range of Randomness- 0.18 - 0.55



Discuss :

How many bits need to be flipped (on average) before the differential bitstream looks random? Does this tell you something about the maximum useful key length for RC4?

In the graphs above, we observe that on average we need to flip between 3-6 bits to get more randomness because of the sudden decrease in the randomness and then afterwards almost steady change in the randomness.

As we have observed that most of the randomness is obtained by toggling some 5-6 bits so the useful size of the key should be more than 7-8 bits.

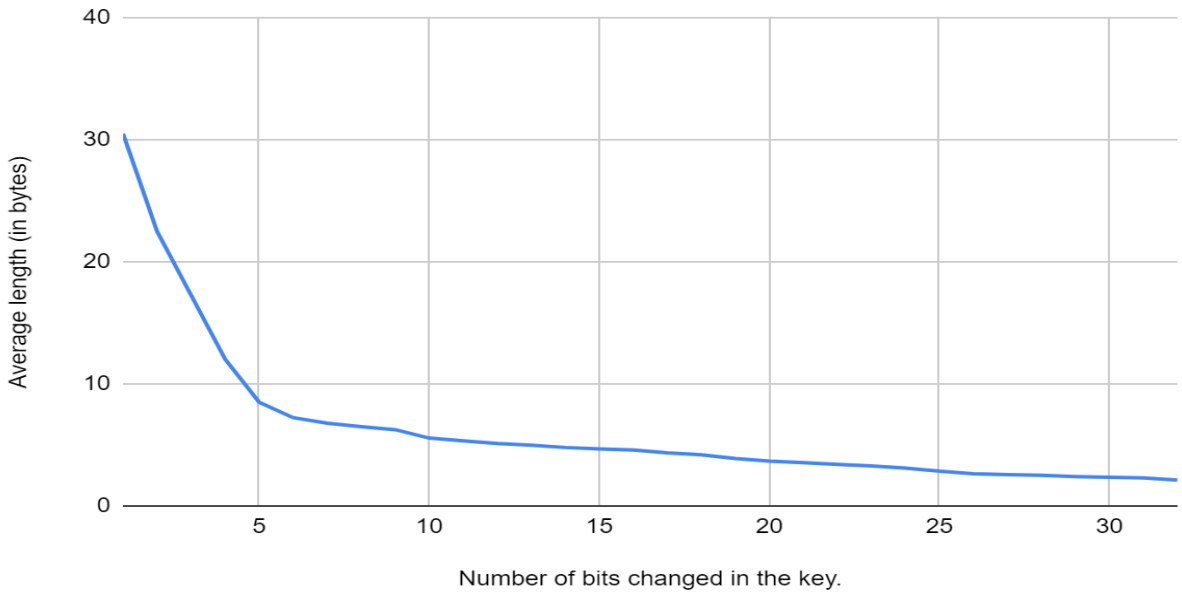
If you use a 128-bit RC4 key, each key bit is replicated 16 times in the internal key. If toggling 16 bits isn't enough to generate a random differential output, then that would imply 128-bit keys are not usefully strong.

We observe that toggling 16 bits in the 128 bits key is not correct because it doesn’t generate the random differential output. So, 128 bit keys are not strong. We observe that the first initial bytes of the output keystream are not perfectly random. It indirectly provides some information about the key.

If a vendor wished to ship a product using RC4 to encrypt short messages (50 bytes max), perhaps they should throw out some of the initial output from RC4 to let the key bits mix properly. How much?

In the previous part, we found that it leaks some information about the key. So, for compensation, we need to remove some parts of the initial output to let the key bits mix properly. Generally, a multiple of 256 bytes is removed from the keystream. So, in this case, 256 bytes should be removed.

Rather than randomness tests, you might observe that two RC4 systems initialized with similar keys may initially generate identical values but will eventually diverge. Try to measure the average length (in bytes) of identical output as a function of the number of bits you change in the key.



We have observed that there is an inverse relationship between the number of bits flipped and the average length of identical output streams in bytes of the two streams. This is evident from the fact that if more the number of bits flipped less will be the correlation between the two output streams. We can also observe that after flipping 5-6 bits the dip in the identical stream length is very less which shows that by flipping 5-6 bits in the key we can get almost maximum randomness.