Palestine Polytechnic University

College of Information Technology and Computer Engineering

Cryptography

NIST Tests Project

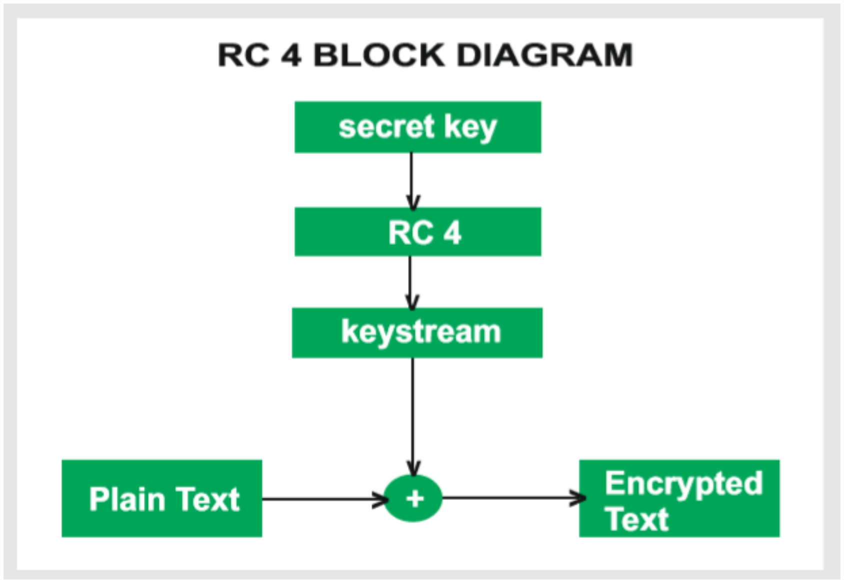
Dr. Mousa Farajallah

Student name

Yaseen Joba

Number :181033

RC4 (Rivest Cipher 4) : it's an encryption algorithm (is a stream cipher) and It is a variable key-size stream cipher with byte-oriented operations.



NIST tests (Randomness tests): are used to analyze the distribution of a set of data to see if it can be described as random or not.

And we will see some of these tests in this report and the result of each of them:

1. **The Frequency (Monobit) Test.**
2. **Frequency Test within a Block.**
3. **The Runs Test.**
4. **The Cumulative Sums (Cusums) Test.**
5. **The Random Excursions Variant Test.**

<u>Test number 1: Frequency (Monobit) Test.</u>

The purpose of this test is to determine whether the number of ones and zeros in a sequence are approximately the same as would be expected for a truly random sequence.

And this can happen only when the number of ones and number of zeros should be about the same. And the probability of zeros equals the probability of ones and equal to ½.

And to do this test we just convert the numbers from zeros and ones to 1 and -1 so if we make a summation between these numbers, we will get value **Sn.**

This value can be ignored or not, so to see that, we need to find S_{obs}. Which given by:

S_{obs} = Sn/sqrt(Key size).

And then we can find the specify P-value = erfc(S_{obs} / sqrt(2)).

<u>In our case:</u>

Key (input to RC4) (Bytes): 65,66,67,68,69,90,150,200,14,23,78,233,44.

Generated key : 1000000 bits .

Sn = -942.

S_{obs} = 0.942.

P-value = 0.346193.

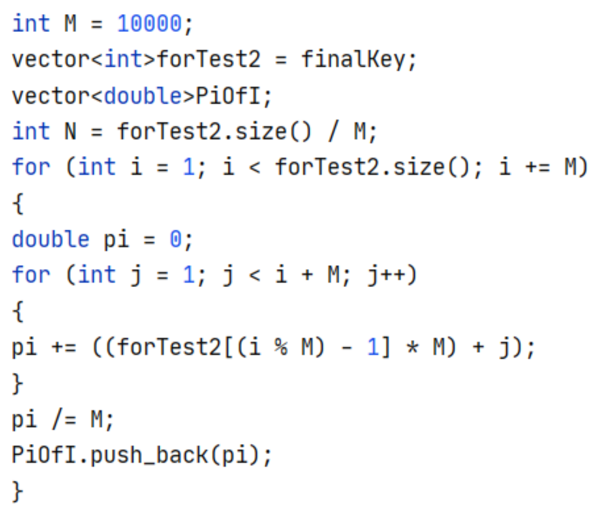
➔ P-value > 0.01 ➔ Test 1 Passed (random key).

<u>Test number 2: Frequency Test within a Block.</u>

In this test we divide the key into N blocks, and then we count the number of ones in each block and see if it approximately equal M/2.and the different between this test and previous test is the size of M, so in the previous test M was equal 1 and we use the same logic.

We just divide the generated key into N blocks each of them has size M = 10000.so in our case N = 100 Blocks.

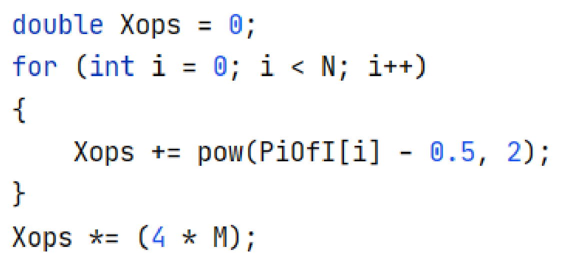
For each blook we need to find **πi** which can be calculated by this loop:



```
int M = 10000;
vector<int>forTest2 = finalKey;
vector<double>PiOfI;
int N = forTest2.size() / M;
for (int i = 1; i < forTest2.size(); i += M)
{
double pi = 0;
for (int j = 1; j < i + M; j++)
{
pi += ((forTest2[(i % M) - 1] * M) + j);
}
pi /= M;
PiOfI.push_back(pi);
}
```

Note: the size of the vector (PiOfI) equal to 100 (number of blooks).

And then we need to calculate X^2(obs) which given by equation expressed using this loop:



```
double Xops = 0;
for (int i = 0; i < N; i++)
{
    Xops += pow(PiOfI[i] - 0.5, 2);
}
Xops *= (4 * M);
```

Finally, we will find P-value using **igamc** function (Incomplete Gamma Function).and we find it online.

In our case:

Key (input to RC4) (Bytes): 65,66,67,68,69,90,150,200,14,23,78,233,44.

Generated key : 1000000 bits .

X^2(obs) = 2.05034e+21.

P-value = igamc(N/2 , Xops / 2) = igamc(50 , 1,025,170E+15) = **UNDERFLOW.**

➔ P-value < 0.01 ➔ Test 2 Failed.

Test number 3: Runs Test.

Run: is an uninterrupted sequence of identical bits bounded before and after with a bit of the opposite value.

The purpose of the runs test is to determine whether the number of runs of ones and zeros of various lengths is as expected for a random sequence.

To do this test we need to calculate **π** for the key by make a summation for all number and divide the result by n (key length).

And then we need to define function r(k) by this rules :

If ($x_i == x_{i+1}$) r(x) = true (1).

Otherwise, r(x) = false (0).

And the summation of all r(k) plus one equal to V_n(obs).

Finally we can find P-value using this law :

$$P\text{-}value = \boldsymbol{erfc}\left(\frac{|V_n(obs) - 2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)}\right).$$

In our case:

Key (input to RC4) (Bytes): 65,66,67,68,69,90,150,200,14,23,78,233,44.

Generated key: 1000000 bits.

Π = 0.499529.

V_n(obs) = 500306.

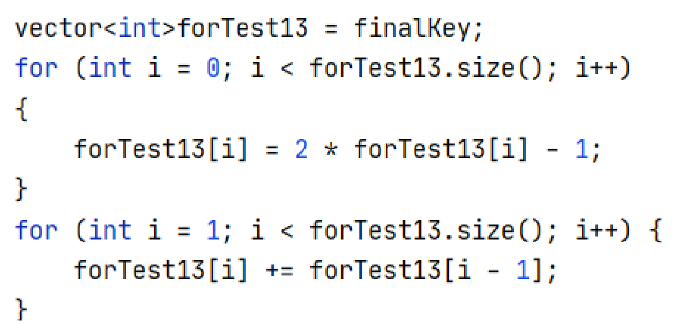
P-value = 0.538039. **P-value > 0.01 ➔ Test 3 Passed (accept the sequence as random).**

Test number 13: The Cumulative Sums (Cusums) Test.

The purpose of the test is to determine whether the cumulative sum of the partial sequences occurring in the tested sequence is too large or too small relative to the expected behavior of that cumulative sum for random sequences.

And to see how accumulative sum is walk we need to convert the sequence from ones and zeros to -1 and +1.

And then we need just to calculate the accumulative sum across the sequence using this loop from my code:



```
vector<int>forTest13 = finalKey;
for (int i = 0; i < forTest13.size(); i++)
{
    forTest13[i] = 2 * forTest13[i] - 1;
}
for (int i = 1; i < forTest13.size(); i++) {
    forTest13[i] += forTest13[i - 1];
}
```

After that we need to find $S_{max}(Z)$ and implement this equation :

$$\text{Compute } P\text{-}value = 1 - \sum_{k=\left(\frac{-n}{z}+1\right)/4}^{\left(\frac{n}{z}-1\right)/4}\left[\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right)-\Phi\left(\frac{(4k-1)z}{\sqrt{n}}\right)\right] +$$

$$\sum_{k=\left(\frac{-n}{z}-3\right)/4}^{\left(\frac{n}{z}-1\right)/4}\left[\Phi\left(\frac{(4k+3)z}{\sqrt{n}}\right)-\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right)\right]$$

In our case:

Key (input to RC4) (Bytes): 65,66,67,68,69,90,150,200,14,23,78,233,44.

Generated key: 1000000 bits.

Z (Smax) = 2059.

P-value = 0.0789885.

➔ P-value >0.01 ➔ **Test 3 Passed (accept the sequence as random).**

Test number 15: The Random Excursions Variant Test.

The focus of this test is the total number of times that a particular state is visited, and we test this in specific states from [-9,9] (18 state).

After we find accumulative sum for modified sequence -1 and 1 , we will check the total number of times that state x appear in my sequence.

And for each state we need to find P-value by this equation:

$$P\text{-}value = \mathbf{erfc}\left(\frac{|\xi(x) - J|}{\sqrt{2J(4|x| - 2)}}\right)$$

And we will construct this table:

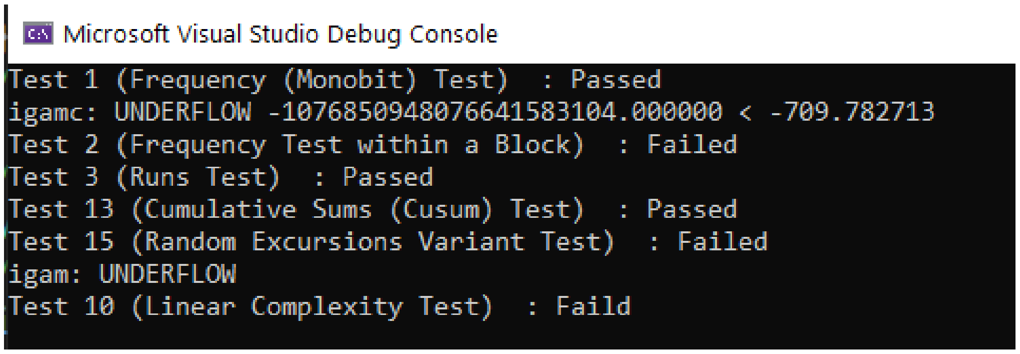
State	count	P-value
0	436	-nan(ind)(small value)
1	456	5.19817e-80
2	516	6.95469e-25
3	610	5.62646e-13
4	654	7.11131e-09
5	662	4.28301e-07
6	681	7.88371e-06
7	746	0.000156821
8	780	0.000784562
9	739	0.000848024
-1	400	1.06128e-88
-2	400	9.52219e-31
-3	428	3.31174e-18
-4	453	6.97354e-13
-5	463	3.58595e-10
-6	430	4.77998e-09
-7	432	7.64387e-08
-8	452	9.12756e-07
-9	451	3.90864e-06

Note: all P-value are less than 0.01 so, **Test 15 failed**

Overview :

Test	State
1.The Frequency (Monobit) Test.	Passed
2.Frequency Test within a Block.	Failed
3.The Runs Test.	Passed
13.The Cumulative Sums (Cusums) Test.	Passed
15.The Random Excursions Variant Test.	Failed
10.Linear Complexity Test	Failed

Program output when key = 65,66,67,68,69,90,150,200,14,23,78,233,44.



```
Test 1 (Frequency (Monobit) Test)  : Passed
igamc: UNDERFLOW -1076850948076641583104.000000 < -709.782713
Test 2 (Frequency Test within a Block)  : Failed
Test 3 (Runs Test)  : Passed
Test 13 (Cumulative Sums (Cusum) Test)  : Passed
Test 15 (Random Excursions Variant Test)  : Failed
igam: UNDERFLOW
Test 10 (Linear Complexity Test)  : Faild
```

Thank you