22BCE3799

Apurba Koirala

Cryptography and Network Security Lab Assessment 5

RC4 operations: The same example as class is taken and 8 is used instead of 256

a. Permutation of the S array

Code:

```
#include <iostream>
#include <vector>

using namespace std;

void printS(const vector<int>& S) {
    for (int val : S) {
        cout << val << " ";
    }
    cout << endl;
}

void rc4_permutation(vector<int>& S, const vector<int>& T) {
    int j = 0;
    for (int i = 0; i < 8; i++) {
        j = (j + S[i] + T[i]) % 8;
        swap(S[i], S[j]);
        cout << "Iteration " << i << ": ";
        printS(S);
    }
}

int main() {
    //in this program, we are running the RC4 algorithm to 8 bits, as taking 256 bits would be very difficult to assess, same as class
    int keysize;
    cout<<"Enter Key size: \n";
    cin>>keysize;
```

```
    if (keysize < 1 || keysize > 8) {
        cout << "Invalid key size. Must be between 1 and 8.\n";
        return 1;
    }
    vector<int> S(8), T(8), key(keysize);
    cout<<"Enter Key:\n";
    for (int i = 0; i < keysize; i++){
        cin>>key[i];
    }



    for (int i = 0; i < 8; i++) {
        S[i] = i;
        T[i] = key[i % keysize];
    }

    rc4_permutation(S, T);

    cout << "S after permutation:\n";
    for (int i = 0; i < 8; i++) {
        cout << S[i] << " ";
    }
    cout << endl;

    return 0;
}
```

Output:

```
Enter Key size:
4
Enter Key:
1
2
3
6
Iteration 0: 1 0 2 3 4 5 6 7
Iteration 1: 1 3 2 0 4 5 6 7
Iteration 2: 2 3 1 0 4 5 6 7
Iteration 3: 2 3 1 6 4 5 0 7
Iteration 4: 2 3 1 4 6 5 0 7
Iteration 5: 2 3 5 4 6 1 0 7
Iteration 6: 2 3 5 4 6 0 1 7
Iteration 7: 2 3 7 4 6 0 1 5
S after permutation:
2 3 7 4 6 0 1 5
```

b. Stream Generation

```cpp
#include <iostream>
#include <vector>

using namespace std;

void printS(const vector<int>& S) {
    for (int val : S) {
        cout << val << " ";
    }
    cout << endl;
}

void rc4_keystream(vector<int>& S, int numBytes) {
    int i = 0, j = 0;

    cout << "Generated Keystream:\n";
```

```
    for (int count = 0; count < numBytes; count++) {
        i = (i + 1) % 8;
        j = (j + S[i]) % 8;

        swap(S[i], S[j]);

        int t = (S[i] + S[j]) % 8;
        int k = S[t];

        cout << k << " ";
    }
    cout << endl;
}

int main() {
    vector<int> S(8);
    cout<<"Enter S vector: \n";
    for (int i = 0; i<8; i++){
        cin>>S[i];
    }

    cout << "Initial S: ";
    printS(S);

    int numBytes;
    cout << "Enter number of keys to generate: ";
    cin >> numBytes;

    rc4_keystream(S, numBytes);

    return 0;
}
```

```
Enter S vector:
2
3
7
4
6
0
1
5
Initial S: 2 3 7 4 6 0 1 5
Enter number of keys to generate: 4
Generated Keystream:
5 1 0 1
```

c. Encryption

Code:

```cpp
#include <iostream>
#include <vector>

using namespace std;

void printVector(const vector<int>& vec) {
    for (int val : vec) {
        cout << val << " ";
    }
    cout << endl;
}

vector<int> rc4_encrypt(vector<int>& plaintext, vector<int>& keystream) {
    vector<int> ciphertext;
    for (size_t i = 0; i < plaintext.size(); i++) {
        ciphertext.push_back(plaintext[i] ^ keystream[i]);
    }
    return ciphertext;
}
```

```
int main() {

    int plaintextSize;
    cout << "Enter plaintext size: \n";
    cin >> plaintextSize;
    vector<int> PT(plaintextSize), K(plaintextSize);
    cout<< "Enter plaintext: \n";
    for (int i = 0; i< plaintextSize; i++){
        cin>> PT[i];
    }

    cout<< "Input Generated Keystream: \n";
    for (int i = 0; i< plaintextSize; i++){
        cin>> K[i];
    }

    vector<int> CT =  rc4_encrypt(PT, K);
    cout << "Ciphertext: ";
    printVector(CT);
    return 0;
}
```

Output:

```
Enter plaintext size:
4
Enter plaintext:
1
2
2
2
Input Generated Keystream:
5
1
0
1
Ciphertext: 4 3 2 3
```