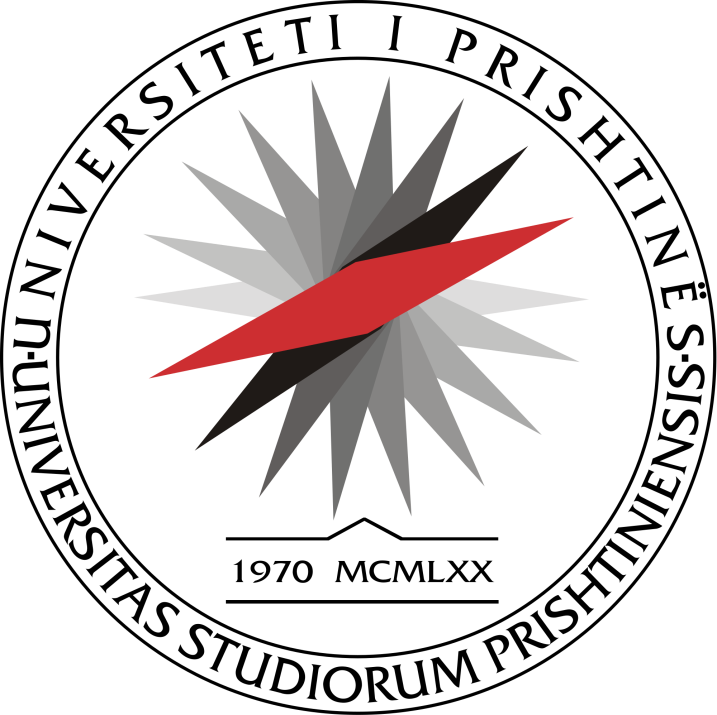
Universiteti i Prishtinës  
“Hasan Prishtina”  
Fakulteti i Shkencave Matematike Natyrore  
Departamenti i Matematikës



Lënda: SIGURIA E TË DHËNAVE   
DETYRA 1

Studentet:   
Suzana Ternava

Arbana Grabanica

Profesor: Artan Berisha

Asistent: Besnik Duriqi

Prishtinë, Mars 2023

Hyrje

Të *sigurosh të dhëna* do të thotë të zbatosh masat e duhura për të siguruar integritetin dhe konfidencialitetin e të dhënave.

Këtë e bejmë me anë të *enkriptimit*(përfshin transformimin e të dhënave në një kod sekret që mund të deshifrohet vetëm me një çelës. Kjo ndihmon për të parandaluar aksesin e paautorizuar në të dhëna), *backup dhe recovery*(përfshin krijimin e kopjeve rezervë të të dhënave në rast të humbjes së tyre apo dëmtimit), *sigurisë fizike*(kamerave etj.) dhe disa mënyrave të tjera.

Algoritmet klasike

Algoritme klasike i’u referohemi algoritmeve që janë zhvilluar para ardhjes së teknologjive moderne kompjuterike. Disa prej më të rëndësishmeve jane:

*Algoritmi i Euklidit*: Ky algoritëm u zhvillua nga matematikani i lashtë grek Euklidi në shekullin III para Krishtit. Përdoret për të gjetur pjesëtuesin më të madh të përbashkët (GCD) të dy numrave. Algoritmi i Euklidit përdoret gjerësisht edhe sot, dhe përbën bazën për shumë algoritme të tjera në teorinë e numrave.

*Binary Search*: Ky algoritëm përdoret për të kërkuar një artikull në një listë të renditur artikujsh duke e ndarë vazhdimisht listën në gjysmë derisa ai të gjendet. Binary Search është një algoritëm themelor i përdorur në shkencën kompjuterike.

*QuickSort*: QuickSort është një algoritëm i renditjes së shpejtë që u zhvillua nga shkencëtari kompjuterik Tony Hoare në vitin 1960. Ai funksionon duke e ndarë një listë në dy nënlista, njëra me elemente më të vegjël se elementi kryesor i zgjedhur dhe një me elemente më të mëdha se elementi kryesor. Më pas, algoritmi i rendit në mënyrë rekursive nënlistat. QuickSort përdoret gjerësisht edhe sot dhe është një nga algoritmet më të njohura të renditjes.

Detyra e parë

Dizajnoni skemën dhe pastaj e implementoni në Java një Nonlinear Feedback Shift Register me këto elemente:

• Vektor inicializues IV (10 bit)

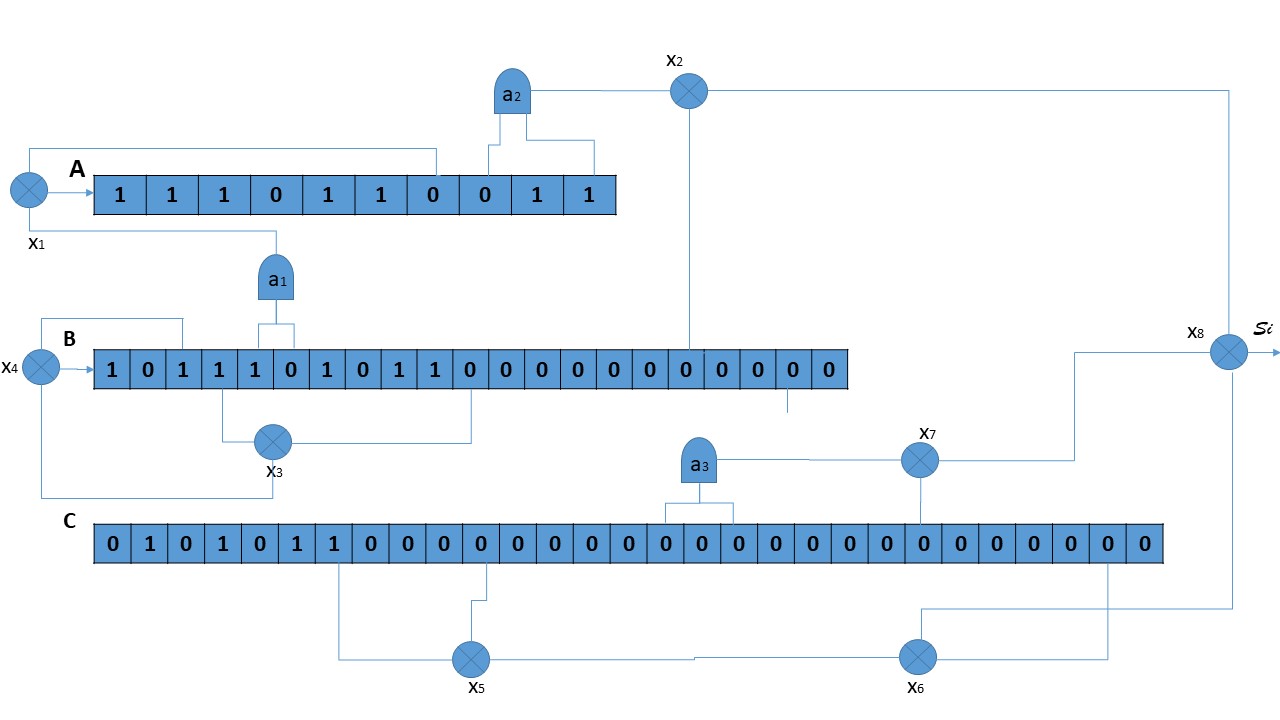
• Çelës K (10 bit)

• Tre regjistra A (11 bit), B (21 bit) dhe C (29 bit)

• Tre operatorë AND

• Tetë operatorë XOR

Dhjetë bitët e parë të regjistirit A mbushen me vektorin inicializues IV, dhjetë bitët e parë të regjistrit B mbushen me çelësin K ndërsa dhjetë bitët e parë të regjistirit C mbushen me rezultatin e fituar nga IV XOR K. Të gjithë bitët tjerë të të tre regjistrave kanë vlerën 0. Së pari llogaritet biti i parë s1 pastaj bëhet zhvendosja e bitëve të regjistrave për një pozitë djathtas dhe pastaj biti i parë në të dy regjistrat zëvendësohet me vlerën e gjetur (sipas dizajnit që keni bërë).



ALGORITMI:

Input IV dhe K

For i = 1 to n

Hapi 1: a1 = B4 dhe B5

Hapi 2: a2 = A7 dhe A9

Hapi 3: a3 = C15 dhe C17

Hapi 4: x1 = a1 xor A6

Hapi 5: x2 = a2 xor B16

Hapi 6: x3 = B10 xor B3

Hapi 7: x4 = B2 xor x3

Hapi 8: x5 = C6 xor C10

Hapi 9: x6 = x5 xor C27

Hapi 10: x7 = C22 xor a3

Hapi 11: x8 = (x2 xor x6) xor x7

Hapi 12: Bëhet zhvendosja te të dy regjsitrat për një bit djathtas

endFor

Kodi në java:

**public** **class** Detyra1 {

//Metoda zhvendos bën zhvendosjen për një bit djathtas

**public** **static** **boolean**[] zhvendos(**boolean**[] a, **boolean** b ) {

**boolean**[] c = **new** **boolean**[ a.length ];

**for** ( **int** i = 1; i < a.length; i++ ){

c[i] = a[i-1];

c[0] = b;

}

**return** c;

}

**public** **static** **void** main(String[] args){

**int** n = 150;

**boolean** a1, a2, a3, x1, x2, x3, x4, x5, x6, x7, x8;

//Vektori inicializues 10 bitesh me vlera 1110110011

**boolean**[] IV = **new** **boolean**[10];

IV[0] = IV[1] = IV[2] = IV[4] = IV[5] = IV[8] = IV[9] = **true**;

// Celesi 10 bitesh me vlera 1011101011

**boolean**[] K = **new** **boolean**[10];

K[0] = K[2] = K[3] = K[4] = K[6] = K[8] = K[9] = **true**;

**boolean**[] A = **new** **boolean**[10];

**boolean**[] B = **new** **boolean**[20];

**boolean**[] C = **new** **boolean**[28];

**boolean**[] s = **new** **boolean**[n];

/\*Dhjetë bitët e parë të regjistirit A mbushen me vektorin inicializues IV dhe

\*dhjetë bitët e parë të regjistrit B mbushen me çelësin K dhe të gjithë bitët tjerë të regjistrave A dhe B e pambushur kanë vlerën binare 0.

\*ndërsa dhjetë bitët e parë të regjistirit C mbushen me rezultatin e fituar nga IV XOR K dhe bitët tjerë me vlerën binare 0\*/

**for** ( **int** i = 0; i < 9; i++ ){

A[i] = IV[i];

B[i] = K[i];

C[i] = IV[i] ^ K[i];

}

//Algoritmi

**for** ( **int** i = 0; i < n; i++ ){

a1 = B[4] && B[5];

a2 = A[7] && A[9];

a3 = C[15] && C[17];

x1 = a1 ^ A[6];

x2 = a2 ^ B[16];

x3 = B[10] ^ B[3];

x4 = B[2] ^ x3;

x5 = C[6] ^ C[10];

x6 = x5 ^ C[27];

x7 = C[22] ^ a3;

x8 = (x2 ^ x6) ^ x7;

s[i] = x8;

A = *zhvendos*(A, x1);

B = *zhvendos*(B, x4);}

//Dalja

**for** ( **int** i = 0; i < n; i++ ){

**if** ( s[i] ) { System.***out***.print(1); }

**else** { System.***out***.print(0); }

}

}

}