SISTEM DE AVERTIZARE ÎN TRAFIC BAZAT PE TEHNOLOGIA VLC

Autori: Ana STOIANOV[1], Gabriel-Ovidiu DAVID[2]
anastoianov96@gmail.com, davidgabriel364@gmail.com

Coordonatori: Conf.univ.dr.ing. **Simona RÎUREAN[3]**, Șef lucr.dr.ing. **Marius- Nicolae RÎSTEIU[3]**

[1] *Universitatea din Petroșani, IME, Calculatoare, anul IV*
[2] *Universitatea, din Petroșani, IME, Automatică și Informatică aplicată, anul IV*
[3] *Universitatea din Petroșani, IME, Departamentul de Automatică, Calculatoare, Inginerie Electrică și Energetică*

Rezumat

Lucrarea se concentrează pe îmbinarea avantajelor comunicatiilor wireless in spatiul de radiofrecventa cu cel din zona luminii vizibile integrate într-un prototip de tip mașina. Prototipul constă într-o mașină echipată astfel incat să primească mesaje atât prin Bluetooth cât și prin lumina provenită de la LED. Mașina, pentru a se misca (“Forward”, “Backward”, “Left”, “Right”), este comandată wireless prin Bluetooth prin intermediul unei aplicații Android realizată în MIT App Inventor, care are înregistrate comenzile vocale ale utilizatorului prototipului. Trecerea mașinii pe sub stalpul de iluminat cu LED-ul aprins permite ascultarea mesajelor vocale (de informare/averizare) primite prin intermediul luminii. Menţinerea mașinii in zona razei luminoase permite ascultarea continuă a mesajului vocal.

Cuvinte cheie

Lumină vizibilă, smart city, mașina mobila.

1. Introducere

Comunicaţia wireless în spectrul de lumină vizibilă (VLC) este o tehnologie nouă, în plină dezvoltare care are o serie de avantaje față de comunicaţia prin unde de frecvenţă radio. Deoarece lumina nu poate trece prin mediul fizic (obstacole mate/dure), VLC este o tehnologie sigură, care păstrează confidențialitatea, de aceea este soluția optimă din punct de vedere al securității securitatea datelor.

Tehnologia VLC este ecologică și rentabilă. Necesită lumină vizibilă pentru a transmite date și are nevoie de mai puține componente în comparație cu tehnologia radio. Astfel, VLC este mai ieftin decât WiFi și cu siguranță are potențialul de a prelua piața comunicațiilor wireless în viitorul apropiat. Ceea ce este mai fascinant este amploarea aplicării sale. Pentru început, VLC poate ajuta la reintroducerea surselor de lumină interioară și exterioară ca bază pentru numeroase aplicații IoT în medii de întreprindere / industriale și orașe inteligente. Tehnologia va crea un mediu avansat tehnologic în, de asemenea, spitale, industrii grele și instituții de învățământ printr-o conectivitate mai sigură și sigură, fără interferențe electromagnetice.

În afară de aceste domenii, VLC poate fi implementat în sistemele de divertisment în timpul zborului prin intermediul comunicației wireless, prin simpla folosire a lămpilor de perete pentru citire.

Cred că este sigur să spunem că domeniile de aplicare ale acestei tehnologii sunt incredibil de diverse iar posibilele idei de implementare sunt interminabile.

2. Descrierea proiectului

Prin acest proiect dezvoltam o simpla implementare a tehnologiilor de comunicaţie wireless care integrate in conceptul de Smart City sa aduca beneficii suplimentare locuitorilor orasului.

Smart City este un oraș viu, asemănător într-o oarecare măsură, unui organism viu. Strategia pentru Smart City, prin complexitatea ariilor tematice și prin abordarea integrată pe care o propune, acoperă o varietate de obiective și direcții de acțiune frecvente incluse pe agenda de priorități globală.

”Smart City” este un oraș care folosește instrumentele tehnologice ale societății informaționale pentru a oferi servicii comunitare la standarde superioare, spre beneficiul locuitorilor săi - un ”sistem al sistemelor” care operează în mod integrat.

Conceptul de Smart City presupune implementarea tehnologiilor informatice integrat care cuprind o multitudine de subsisteme de cloud computing, Internet of Things (IoT), Open Data, Big Data și aplicații mobile, conectate la internet prin intermediul unor rețele sigure. Acestea permit administrației locale să interacționeze direct cu cetățenii și cu infrastructura orașului.

Pentru a deplasa masina folosim comenzi vocale precum: „Înainte”, „Înapoi”, „Stânga”, „Dreapta” . Mașina creata de noi este controlată prin bluetooth hc-05 printr-o aplicație pentru smartphone.

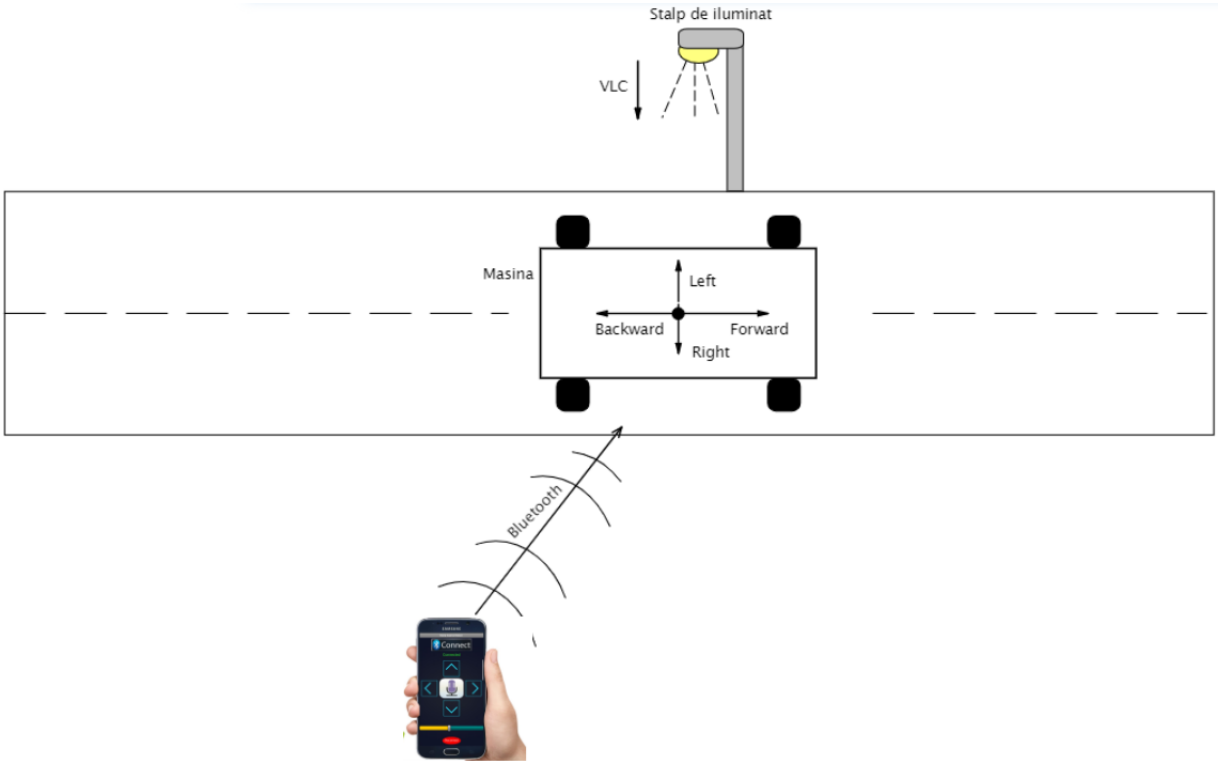
Aplicația este dezvoltată în așa fel încât să convertească comanda vocală în text și să transfere textul pe dispozitivul Bluetooth conectat. Bluetooth-ul conectat pe placa Arduino primește text din aplicația Android sub formă de caractere și le stochează în șirul alocat. Există cuvinte preprogramate (merge înainte, înapoi, întoarce la dreapta, întoarce la stânga, stop și break).

Atunci cand masina se afla sub spotul luminos al stalpului, conductorul auto primeste mesaje de avertizarte precum:

- Sunteti in zona periculoasa

- Nu stationati amenda este,,,

In figura 2. este prezentata schema conceptuala a proiectului nostru.



3. Etapele de proiectare, design, dezvoltare, implementare şi testare a sistemului

Implementarea proiectului necesită mai întâi o scurtă evaluare a componentelor hardware necesare și a software-lui open source disponibil pentru a dezvolta o comunicare simplă și utilă telefon – mașina.

Elementele componente necesare pentru construirea prototipului și utilizate pentru a dezvolta proiectul menționat mai sus sunt prezentate în continuare.

- ***Hardware***

Mașina e creată din umătoarele componente:

Arduino Uno,
Kit cu 4 roți,
modul bluetooth hc-05,
Driver de motor L293D,
Cabluri jumper,
Suport baterie 18650 – 2 celule,
Baterie 18650 3,7 V x 2,
Switch on-off

- ***Software***

Arduino IDE for Windows;

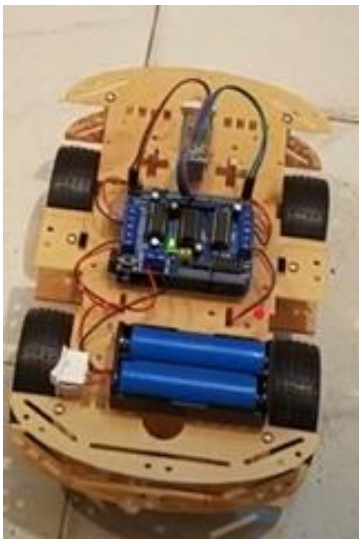
MIT App Inventor.

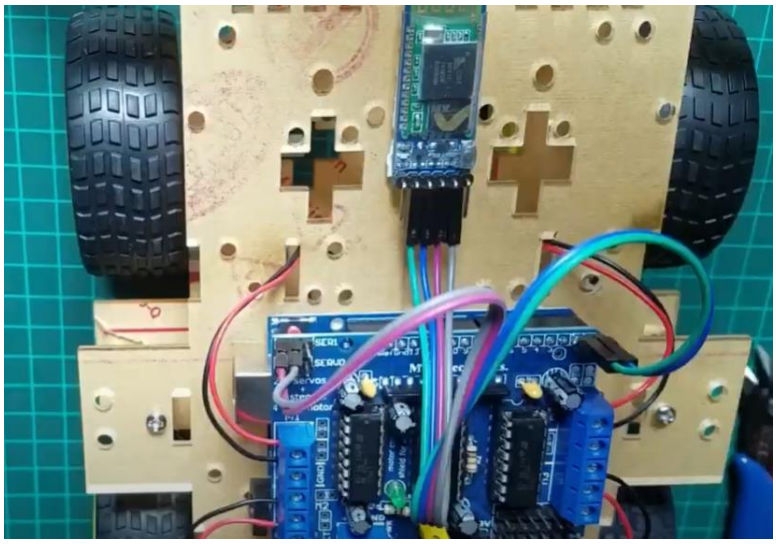
Pentru a realiza acest proiect am ales placa Arduino Uno ținând cont de tipul de microcontroler încorporat deoarece este ușor programabil, versatil, are un cost redus.
Placa este echipată cu seturi de pini de intrare/ieșire (I/O) digitale și analogice care pot fi interfațate cu diferite plăci de expansiune sau Breadboards și alte circuite. Arduino dispune de interfețe de comunicații seriale, inclusiv Universal Serial Bus (USB) disponibil pe modelul nostru, fiind folosit ca port de comunicație pentru a încărca programe de pe computerele personale.

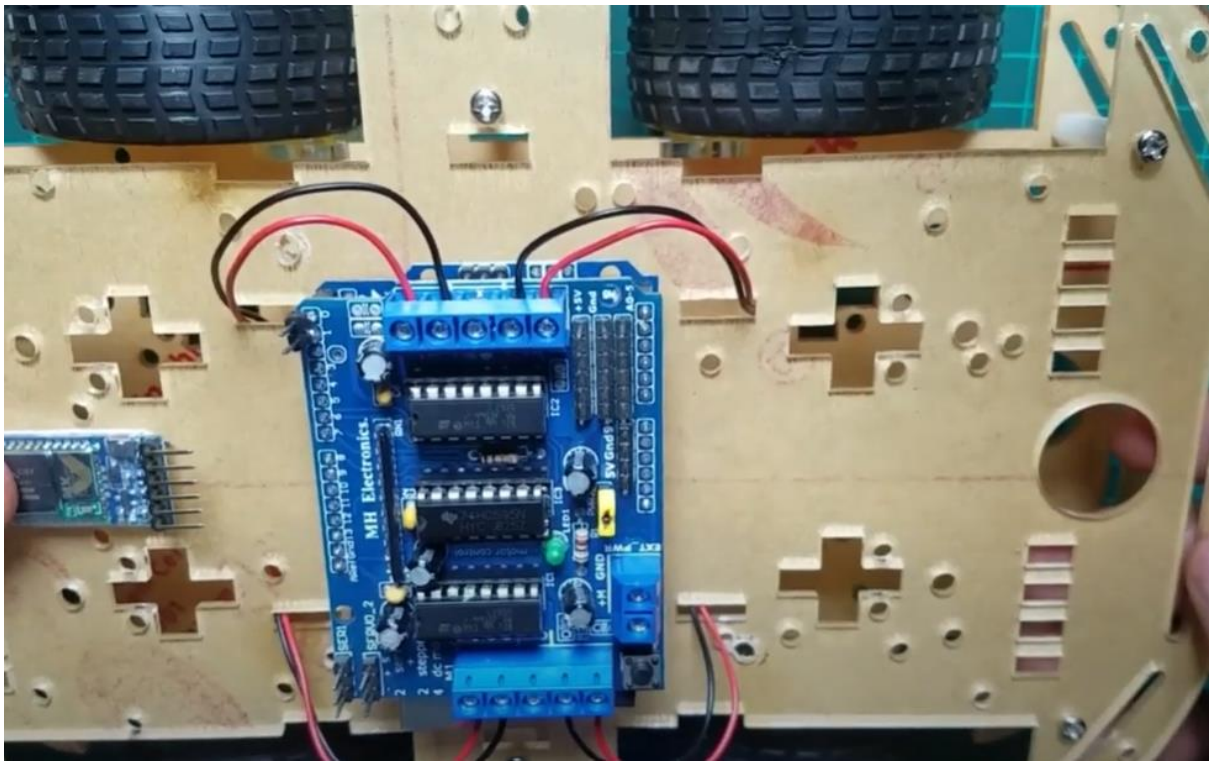
- ***Implimentarea***

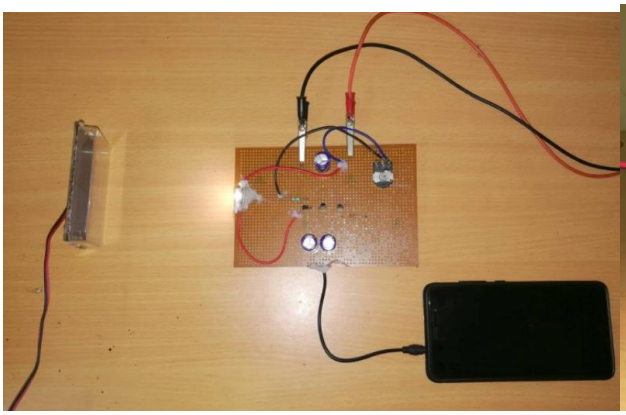
Inițial ne propunem construirea mașinii propriu zise pe care vom monta o celulă solară care va avea rol de senzor de lumină pentru activarea semnalului acustic. În momentul trecerii mașinii pe sub o sursă de lumină, celula solară va reacționa și va trimite comandă către amplificatorul de sunet care va comunica un mesajul de avertizare. La construirea circuitului LI-FI a trebuie sa luam in considerare iluminarea camerei astfel incat circuitul nostru trebuie sa ofere o iluminare constanta. Iluminare trebuie sa fie constanta astel incat sa nu nu existe un fenomen de de pâlpâire

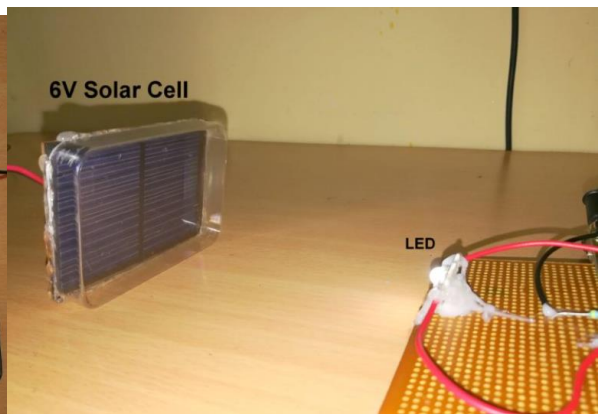
sesizabil de ochiul liber in timpul transferului de date. Schimbare nivelului de lumina al LED-ului, necesara pentru transmiterea datelor nu este perceptibila de ochiul uman ea poate fi detectata cu usurinta doar cu ajutorul celulei solare.

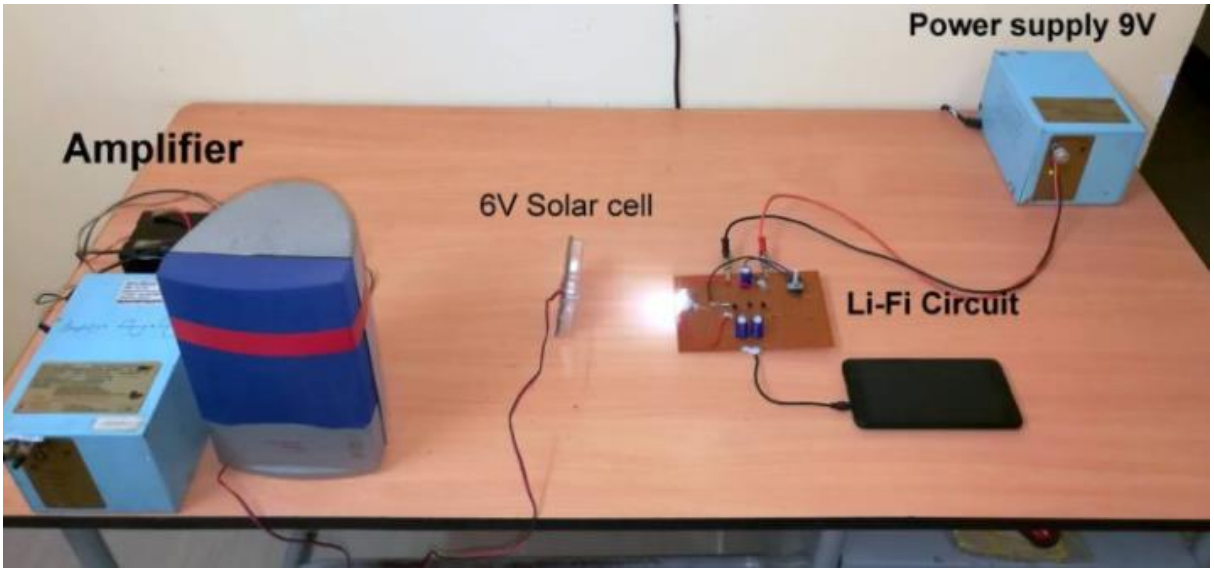












4. Concluzii

În urma realizării acestui proiect, am ajuns la concluzia că prin intermediul luminii se pot realiza o multitudine de proiecte pentru dezvoltarea orașelor inteligente și chiar pentru implimentarea lor în viața de zi cu zi.

Bibliografie:

1. https://www.bsigroup.com/en-GB/smart-cities/
3. https://www.gov.uk/government/collections/future-of-cities
4. https://www.runwithcode.com/what-is-smart-city/
5. https://ro.scribd.com/document/395680409/Smart-City
6. Simona Riurean, Razvan Stoica, Monica Leba. (2017) *Visible Light Communication for Audio Signals*. International Journal of Communications, 2, 24-27

APLICAŢIE ANTIVIRUS DE SCANAREA FIŞIERELOR PENTRU ÎMBUNĂTĂŢIREA SIGURANŢEI CIBERNETICE

Autor: Daniel-Sorin PĂULESCU[1]
paulescudaniel40@yahoo.com

Coordonator: Conf.univ.dr.ing. **Simona RÎUREAN**[2]

[1] *Universitatea din Petroşani, Facultatea IME, Specializarea Calculatoare anul IV*
[2] *Universitatea din Petroşani, Facultatea IME, Departamentul ACIEE*

Rezumat

În lucarea de faţă prezint o serie de elemente referitoare la modalităţile de a creşte securitatea datelor prin prevenirea atacurilor cibernetice asupra dispozitivelor electronice inteligente prin scanarea tuturor fişierelor primite pentru a detecta fişerele cu conţinut maliţios. Am creat o aplicaţie antivirus care scanează fişerele pentru a identifica orice conţinut nedorit care poate afecta siguranţa cibernetică a utilizatorilor, cu scopul de a asigura protecția dispozitivelor.

Cuvinte cheie

Antivirus, siguranță cibernetică, protecție.

1. Introducere

În anul 1978 au fost lansate primele programe anti-virus, acestea fiind: Dr. Solomon's Anti-Virus Toolkit, AIDSTEST și AntiVir. La sfarșitul lui 1990 erau disponibile deja 19 produse antivirus, inclusiv Norton AntiVirus și McAfee VirusScan, acestea fiind foarte folosite și în prezent.

Între anii 2005 și 2012 au început să apară programele pseudo-antivirus-software care nu sunt programe antivirus. Scopul lor era de a înșela utilizatorii și de a realiza un profit cât mai mare, unele fiind chiar programe rău intenționate.

În noiembrie 2014, organizația internațională pentru drepturile omului, Amnesty International, a lansat programul Anti-Virus Detect, un program care era destinat pentru detectarea software-ului rău intenționat, distribuit de agențiile guvernamentale, pentru a spiona activiștii societății civile și oponenții politici. Acesta, în opinia creatorilor, efectuează o scanare mai profundă a hard diskului decât antivirusul obișnuit

Termenul care descrie o gamă largă de programe maliţioase, denumit malware se referă la troieni, viermi, rootkits, ransomware, ameninţări cibernetice şi chiar programe cu potenţial nedorit (PUP). De obicei, este instalat în sistem fără ştirea sau aprobarea utilizatorului, exploatând vulnerabilităţile din securitate.

Software-urile maliţioase sunt utilizate în general pentru a iniţia activităţi neautorizate în calculator. Poate fi proiectat pentru a fura informaţii personale, precum nume de autentificare şi date bancare, sau poate încerca să cripteze fişierele importante din calculator şi să facă pe proprietarul acestora să plătească o recompensă în schimbul cheii de decriptare.

Unele versiuni de malware (adware, browser hijackeri şi altele similare) sunt utilizate doar pentru a afişa conţinut promoţional în calculatoarele oamenilor şi să genereze venituri pay-per-click. Aproape fiecare ameninţare malware are abilitatea de a bloca un software de securitate legitim. În plus, se pot actualiza singure, să descarce malware adiţional sau să cauzeze găuri în securitatea sistemului de PC afectat.

Tipuri de malware:
- Viruși;
- Troieni;
- Viermi;
- Ransomware;
- Spyware;
- Adware;
- Rootkit, time bombs, backdoor, etc

Programele de protecție - programe antivirus, care realizează simultan următoarele activități:
- prevenirea contaminării;
- detecție infectare;
- eliminarea virusului.

Software-urile antivirus sunt programe care încearcă sa identifice, neutralizeze sau sa elimine malware-ul. Deși termenul "antivirus" se referă exclusiv la virușii de calculator totuși marea majoritate a software-ului antivirus modern este făcut pentru a combate o gama larga de amenințări, incluzând viermi,atacuri pishing, rootkits, și troieni, descriși colectiv ca malware.

Software-ul de scanare antivirus, sau un scanner de viruși, este un program care examinează toate fișierele din locații specificate, conținuturile de memorie, sistemul de operare, regiștrii, comportamentul imprevizibil al programelor

și oriunde este relevant cu intenția de a identifica și înlătura orice malware. Sunt folosite doua abordări diferite pentru a identifica malware, deseori în combinație:

- examinarea(scanarea) fișierelor de viruși cunoscuți care se potrivesc cu înregistrările dintr-un dicționar de viruși;

- analiza euristică - identificarea comportamentului suspicios din partea oricărui program care ar putea indica infecție, putând include captură de date, monitorizarea porturilor și alte metode.

Software-ul antivirus bazat pe dicționar - examinează în mod obișnuit fișierele când sistemul de operare al computerului creează, deschide, închide, sau folosește e-mail-ul. În acest fel poate detecta un virus cunoscut imediat ce-l primește. Software-ul antivirus poate fi programat scaneze toate fișierele din hard discul computerului după un anumit principiu.

Deși metoda dicționarului poate fi efectivă în circumstanțele potrivite, autorii de viruși au încercat sa fie cu un pas înainte scriind viruși oligomorfici, polimorfici și mai nou metamorfici, care își encripteaza anumite părți sau pe de alta parte se modifica pentru a nu se potrivi cu semnăturile din dicționarul de viruși.

O tehnica inovatoare de neutralizare a malware-ului este whitelisting. În loc sa caute doar malware cunoscut, acesta tehnica previne executarea oricărui cod cu excepția celor care au fost înainte identificate ca fiind de încredere de către administratorul de sistem. Adițional, aplicațiile de computer care nu sunt dorite de administratorul de sistem sunt împiedicate sa pornească din moment ce ele nu se afla în whitelist.

Comportament suspicios-euristic - nu încearcă sa identifice viruși cunoscuți în schimb monitorizează comportamentul tuturor programelor. Daca un program încearcă sa scrie date către un program executabil, software-ul antivirus poate semnala acest comportament suspicios, alertând utilizatorul. Prin urmare oferă protective împotriva noilor viruși care nu exista încă în dicționarele de viruși.

Un atac cibernetic este un atac lansat de la unul sau mai multe computere împotriva unui alt computer, a mai multor computere sau rețele.

Atacurile cibernetice pot fi împărțite în două categorii: atacuri în care scopul este de a dezactiva computerul țintă sau a-l pune offline sau atacuri în care obiectivul este de a avea acces la datele computerului țintă și de a obține privilegii de administrare pe acesta.

2. Ghid de bune practici pentru îmbunătățirea siguranței cibernetice

Securizarea stațiilor de lucru (PC-uri, laptopuri) și a altor dispozitive conectate la rețele, cu sau fără fir, este o condiție esențială atât pentru asigurarea confidențialității și autenticității datelor sensibile, cât și pentru desfășurarea activităților uzuale la nivelul utilizatorilor

APLICAȚII ȘI SUITE DE SECURITATE

Se recomandă instalarea unor aplicații antimalware sau a unor suite de securitate complexe, performante, care să asigure protecția la cele mai recente tipuri de amenințări cibernetice (ransomware, troiani). Actualizarea permanentă a bazei de date cu semnături malware este o condiție necesară pentru detecția celor mai recente forme de amenințări.

CRIPTAREA DATELOR SENSIBILE

Se recomandă utilizarea unor terțe aplicații sau sisteme de operare ce dețin implementate facilități pentru criptarea datelor sensibile la nivel de fișier individual, folder sau un întreg drive logic.

SECURIZAREA SISTEMULUI DE OPERARE

Se realizează atât prin repararea breșelor de securitate și a erorilor software la nivelul tuturor componentelor sistemului de operare (prin aplicarea periodică, automată sau manuală, a actualizărilor), cât și prin controlul accesului utilizatorilor la resurse (drepturi de acces la fișiere, servicii și aplicații).

ACTUALIZAREA APLICAȚIILOR

Este o acțiune absolut necesară deoarece previne unele atacuri cibernetice și scurgeri costisitoare de date, ajutând la păstrarea în siguranță a datelor sensibile. Utilizatorii trebuie să activeze actualizarea automată a tuturor aplicațiilor esențiale (la nivel de sistem de operare, antivirus, firewall sau IDPS).

COPII DE REZERVĂ A DATELOR

Datele trebuie periodic salvate (backup) și stocate pe suporturi magneto-optice de încredere, depozitate în locuri sigure și eventual criptate pentru a evita accesul neautorizat. Aceste copii trebuie păstrate în mai multe locații fizice (sedii) pentru a evita atât dezastrele naturale, cât și amenințările interne din cadrul companiei.

GESTIONAREA PAROLELOR

În anumite situații se poate recomanda utilizarea unui manager de parole pentru a stoca parole complexe, unice, generate de computer. Parolele folosite trebuie să fie puternice (utilizând caractere alfanumerice și simboluri speciale), să nu fie refolosite la mai multe conturi și trebuie schimbate periodic.

AUTENTIFICAREA CU DOI FACTORI

Este o metodă foarte eficientă și modernă, care folosește un dispozitiv suplimentar (ex. token de securitate sau un smartphone) pentru a confirma într-un pas suplimentar identitatea persoanei care se autentifică. O autentificare suplimentară poate fi realizată folosind datele biometrice.

UTILIZAREA UNOR CONTURI CU DREPTURI LIMITATE

Utilizarea unor conturi cu drepturi limitate în locul unui cont de administrator va bloca accesul la zone sensibile ale sistemului de operare și va bloca implicit atacurile ce vizează serviciile sistemului de operare, fișierele sau bibliotecile sale.

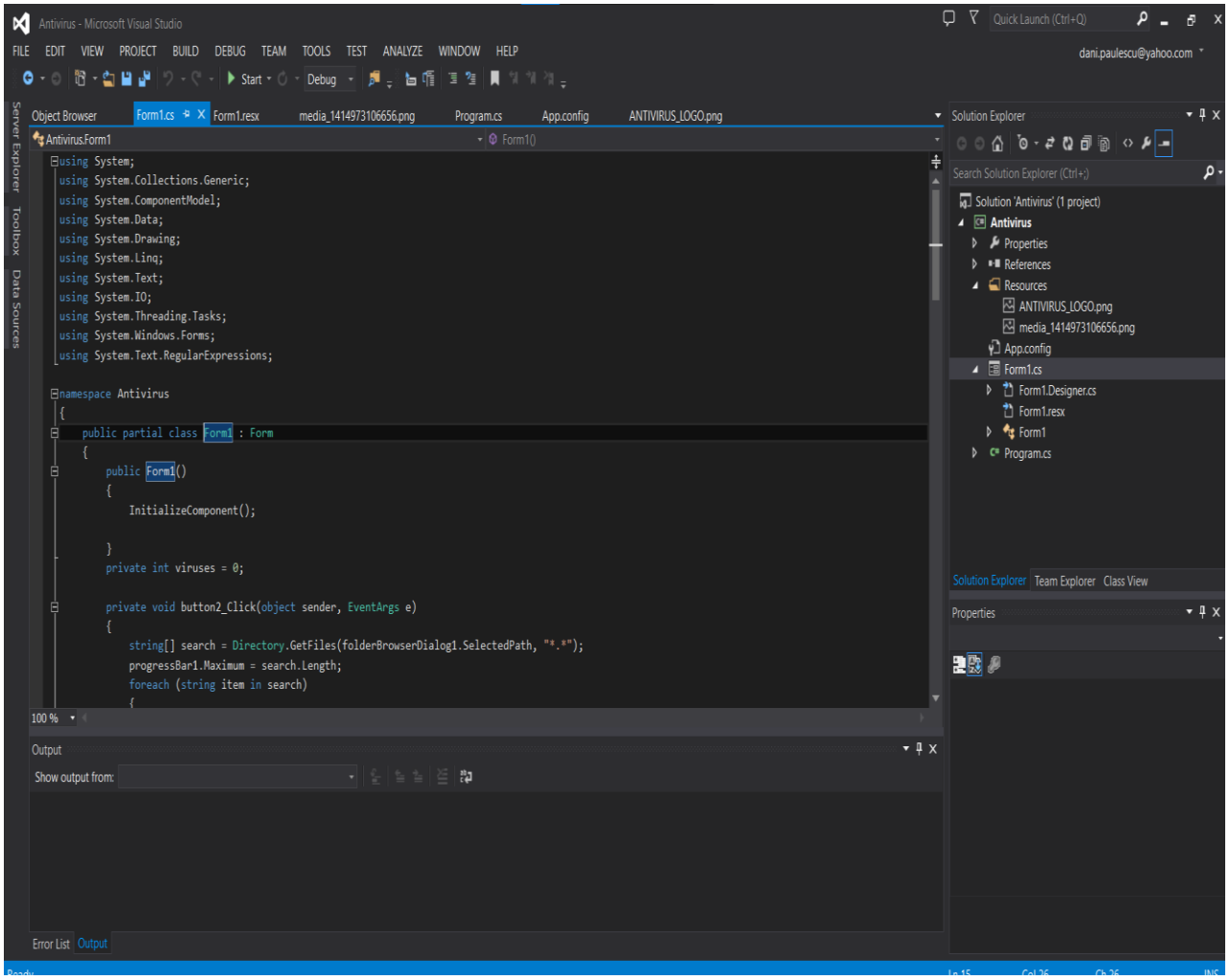
3. Descrierea programului creat

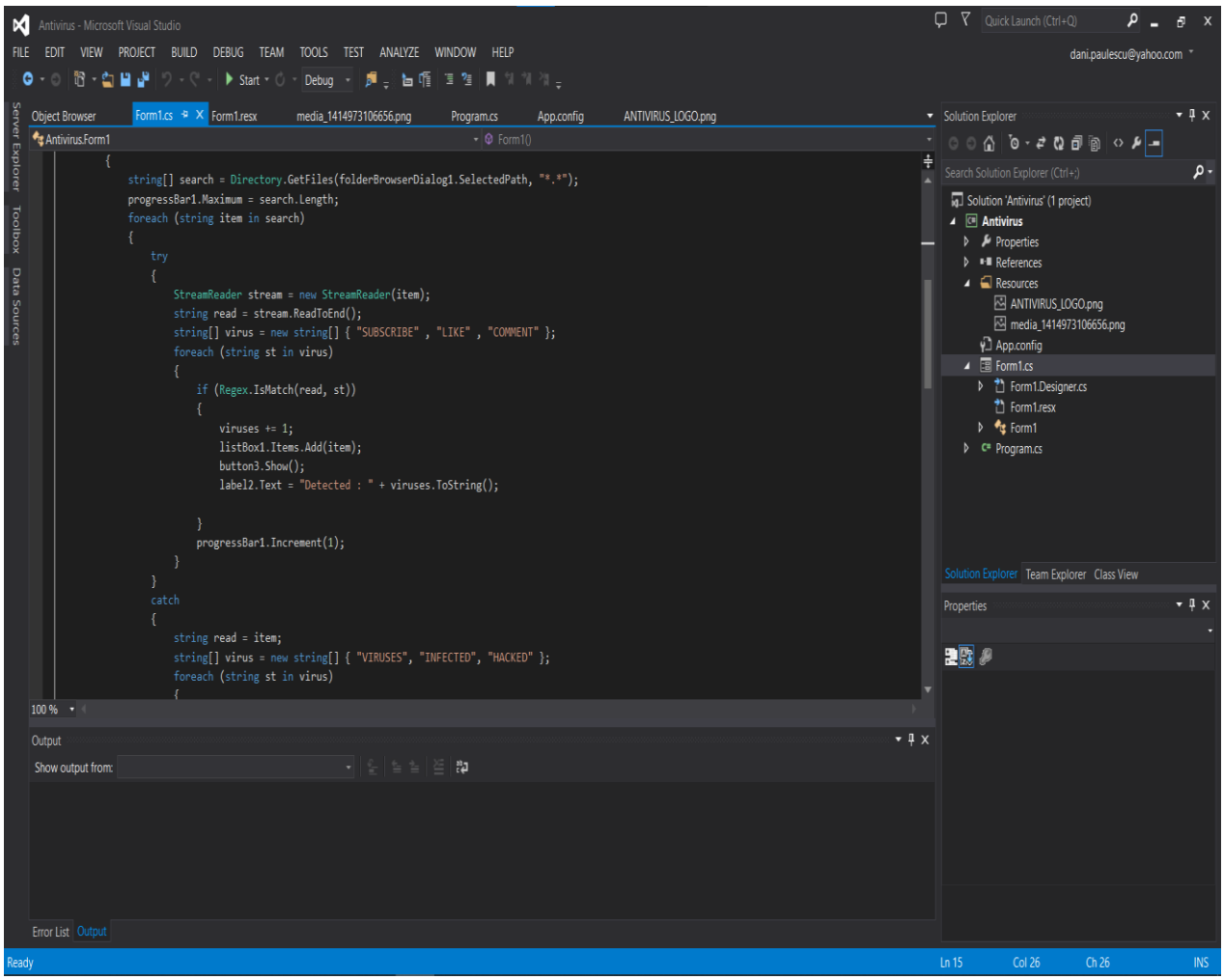
Folosind cunoștintele asimilate pe parcursul celor patru ani de facultate mi-am propus să crez un program antivirus mult mai sigur pentru toți utilizatorii.

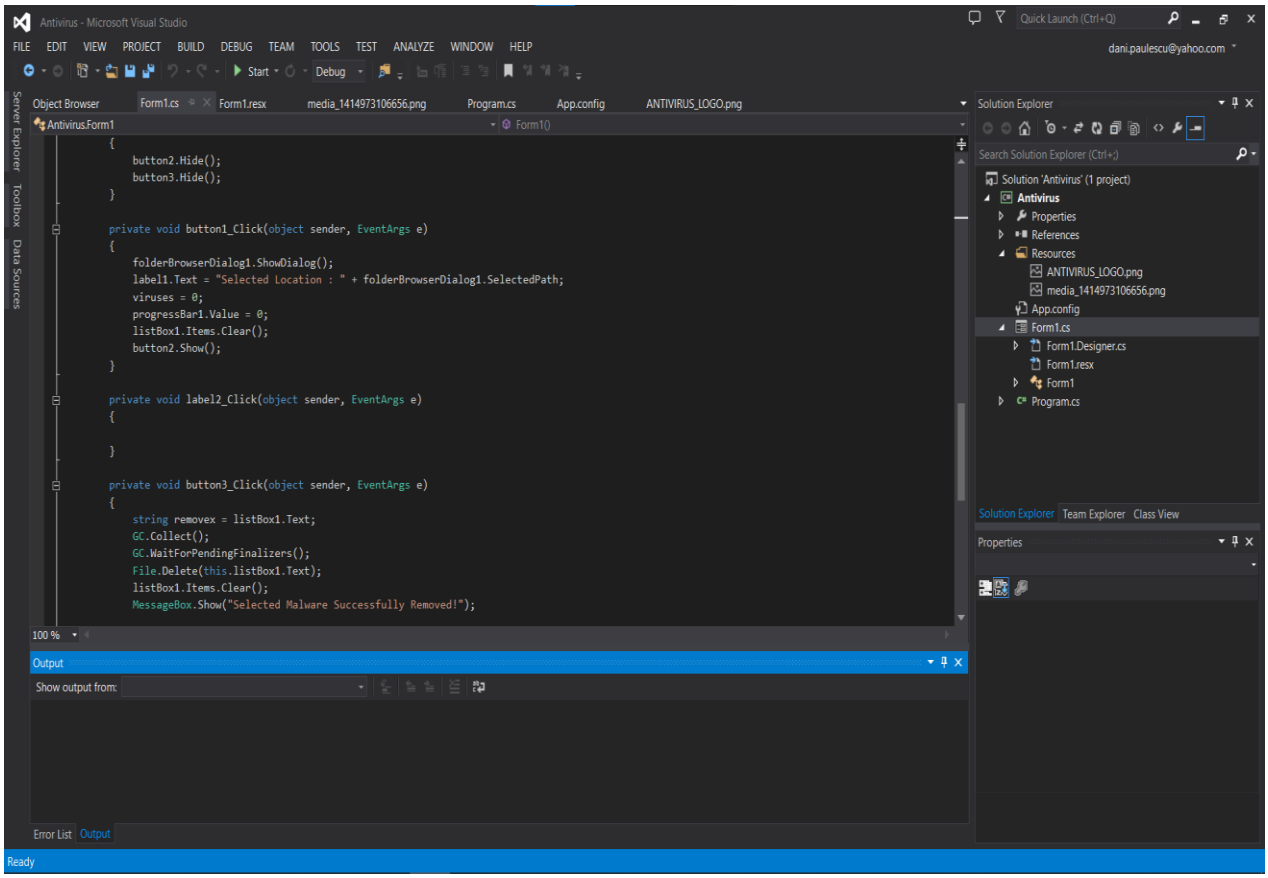
Ca suport software am folosit Microsoft Visual Studio, unde am utilizat limbajul de programare C#, pentru a scrie codul programului antivirus pe care l-am creat, după cum se poate observa mai jos.

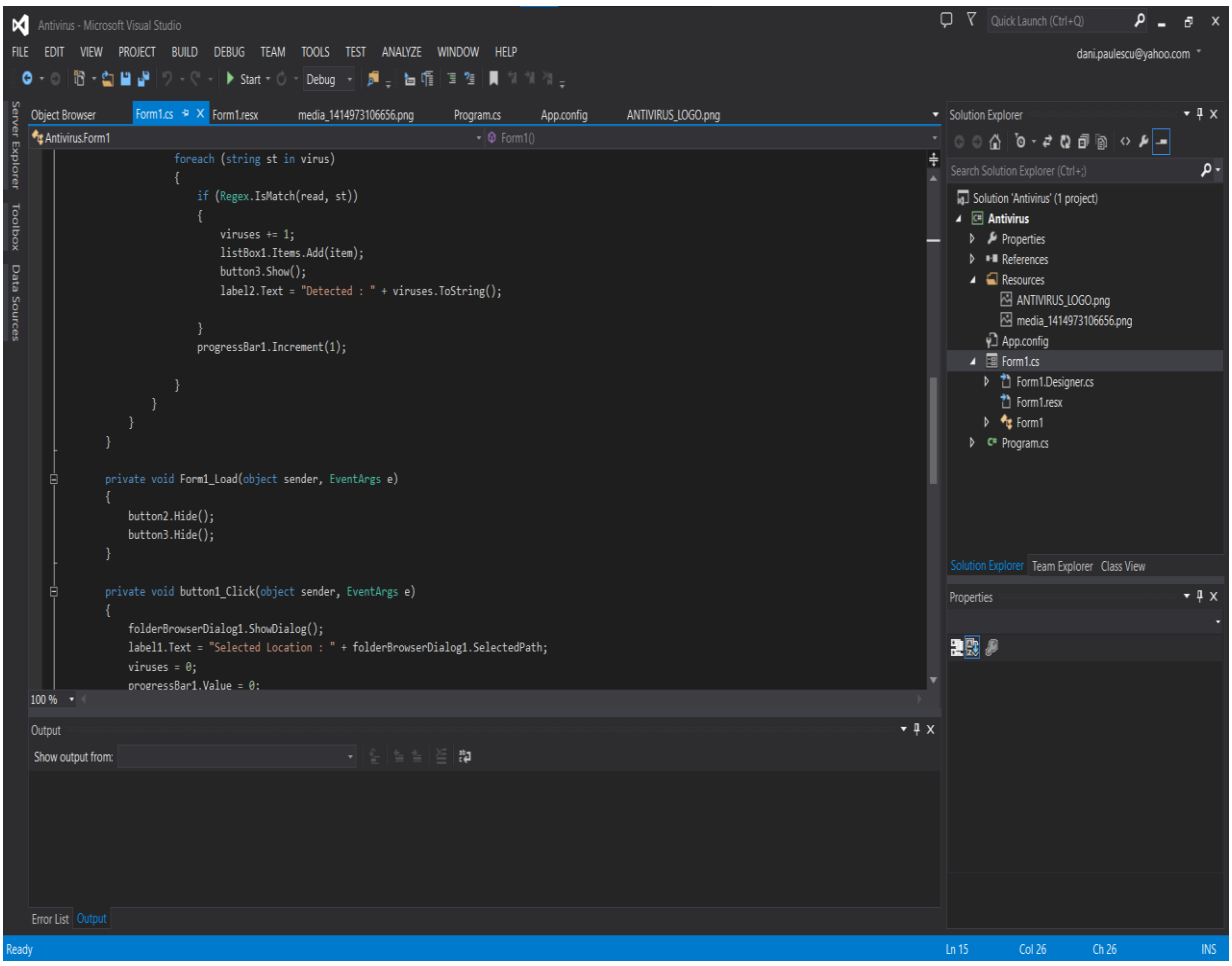
3.1. Scrierea codului

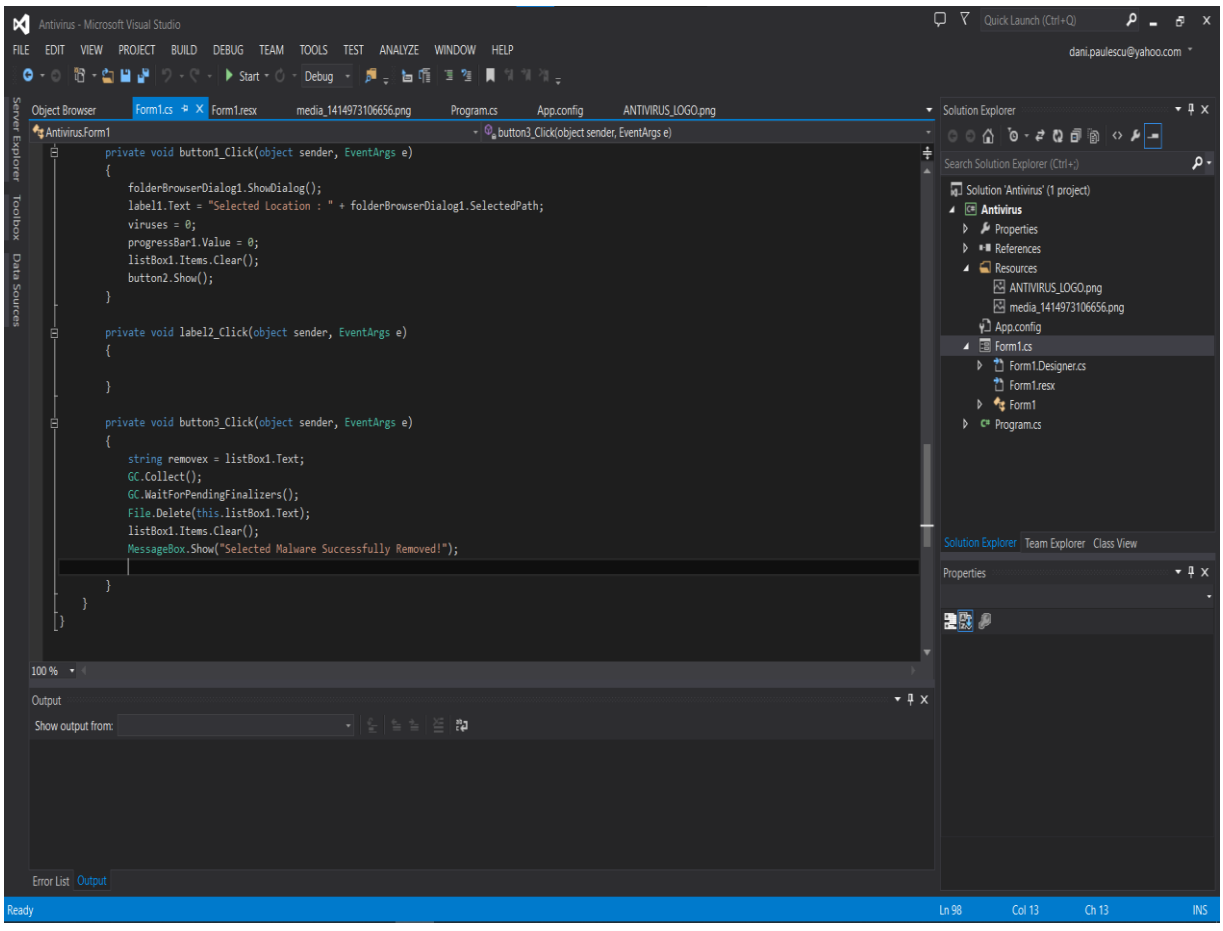
In figurile urmatoare sunt prezentate liniile de cod stau la baza programului de scanare antivirul pe care l-am creat.




Antivirus - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
dani.paulescu@yahoo.com
Start Debug
Object Browser Form1.cs Form1.resx media_1414973106656.png Program.cs App.config ANTIVIRUS_LOGO.png
Antivirus.Form1 Form1()
{
string[] search = Directory.GetFiles(folderBrowserDialog1.SelectedPath, "*.*");
progressBar1.Maximum = search.Length;
foreach (string item in search)
{
try
{
StreamReader stream = new StreamReader(item);
string read = stream.ReadToEnd();
string[] virus = new string[] { "SUBSCRIBE" , "LIKE" , "COMMENT" };
foreach (string st in virus)
{
if (Regex.IsMatch(read, st))
{
viruses += 1;
listBox1.Items.Add(item);
button3.Show();
label2.Text = "Detected : " + viruses.ToString();
}
progressBar1.Increment(1);
}
}
catch
{
string read = item;
string[] virus = new string[] { "VIRUSES", "INFECTED", "HACKED" };
foreach (string st in virus)
{
100 %
Output
Show output from:
Error List Output
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Antivirus' (1 project)
Antivirus
Properties
References
Resources
ANTIVIRUS_LOGO.png
media_1414973106656.png
App.config
Form1.cs
Form1.Designer.cs
Form1.resx
Form1
Program.cs
Solution Explorer Team Explorer Class View
Properties
Ready
Ln 15 Col 26 Ch 26 INS

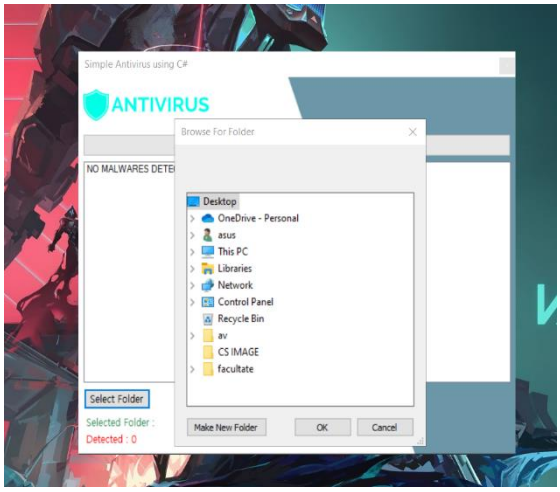

Antivirus - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
dani.paulescu@yahoo.com
Start Debug
Object Browser Form1.cs Form1.resx media_1414973106656.png Program.cs App.config ANTIVIRUS_LOGO.png
Antivirus.Form1 Form1()
{
button2.Hide();
button3.Hide();
}
private void button1_Click(object sender, EventArgs e)
{
folderBrowserDialog1.ShowDialog();
label1.Text = "Selected Location : " + folderBrowserDialog1.SelectedPath;
viruses = 0;
progressBar1.Value = 0;
listBox1.Items.Clear();
button2.Show();
}
private void label2_Click(object sender, EventArgs e)
{
}
private void button3_Click(object sender, EventArgs e)
{
string removex = listBox1.Text;
GC.Collect();
GC.WaitForPendingFinalizers();
File.Delete(this.listBox1.Text);
listBox1.Items.Clear();
MessageBox.Show("Selected Malware Successfully Removed!");
100 %
Output
Show output from:
Error List Output
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Antivirus' (1 project)
Antivirus
Properties
References
Resources
ANTIVIRUS_LOGO.png
media_1414973106656.png
App.config
Form1.cs
Form1.Designer.cs
Form1.resx
Form1
Program.cs
Solution Explorer Team Explorer Class View
Properties
Ready

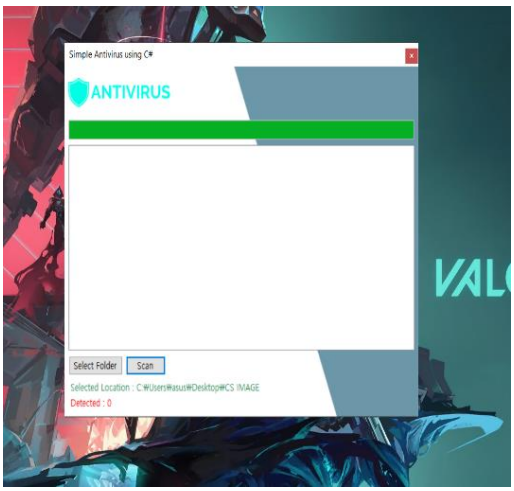

Antivirus - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
dani.paulescu@yahoo.com
Start Debug
Object Browser
Form1.cs
Form1.resx
media_1414973106656.png
Program.cs
App.config
ANTIVIRUS_LOGO.png
Antivirus.Form1
Form1()
foreach (string st in virus)
{
if (Regex.IsMatch(read, st))
{
viruses += 1;
listBox1.Items.Add(item);
button3.Show();
label2.Text = "Detected : " + viruses.ToString();
}
progressBar1.Increment(1);
}
}
}
}
private void Form1_Load(object sender, EventArgs e)
{
button2.Hide();
button3.Hide();
}
private void button1_Click(object sender, EventArgs e)
{
folderBrowserDialog1.ShowDialog();
label1.Text = "Selected Location : " + folderBrowserDialog1.SelectedPath;
viruses = 0;
progressBar1.Value = 0;
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Antivirus' (1 project)
Antivirus
Properties
References
Resources
ANTIVIRUS_LOGO.png
media_1414973106656.png
App.config
Form1.cs
Form1.Designer.cs
Form1.resx
Form1
Program.cs
Solution Explorer Team Explorer Class View
Properties
100 %
Output
Show output from:
Error List Output
Ready
Ln 15 Col 26 Ch 26 INS


Antivirus - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
dani.paulescu@yahoo.com
Start Debug
Object Browser
Form1.cs
Form1.resx
media_1414973106656.png
Program.cs
App.config
ANTIVIRUS_LOGO.png
Antivirus.Form1
button3_Click(object sender, EventArgs e)
private void button1_Click(object sender, EventArgs e)
{
folderBrowserDialog1.ShowDialog();
label1.Text = "Selected Location : " + folderBrowserDialog1.SelectedPath;
viruses = 0;
progressBar1.Value = 0;
listBox1.Items.Clear();
button2.Show();
}
private void label2_Click(object sender, EventArgs e)
{
}
private void button3_Click(object sender, EventArgs e)
{
string removex = listBox1.Text;
GC.Collect();
GC.WaitForPendingFinalizers();
File.Delete(this.listBox1.Text);
listBox1.Items.Clear();
MessageBox.Show("Selected Malware Successfully Removed!");
}
}
}
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'Antivirus' (1 project)
Antivirus
Properties
References
Resources
ANTIVIRUS_LOGO.png
media_1414973106656.png
App.config
Form1.cs
Form1.Designer.cs
Form1.resx
Form1
Program.cs
Solution Explorer Team Explorer Class View
Properties
100 %
Output
Show output from:
Error List Output
Ready
Ln 98 Col 13 Ch 13 INS

3.2. Crearea interfeței utilizator

Pentru interfață am folosit cunoștiințele de Visual Basic și modelare. Programul scanează fișierul selectat pentru a verifica integritatea acestuia așa cum se poate observa și în urmatoarele imagini



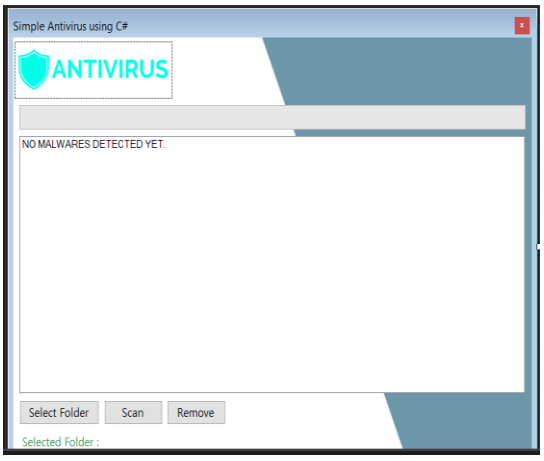


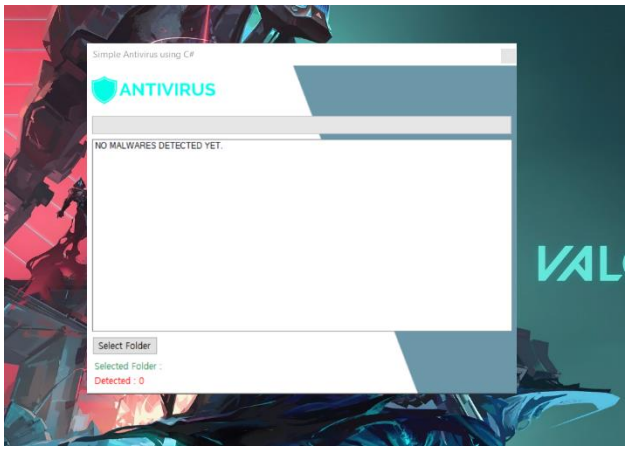
4. Testarea programului

După deschiderea și rularea programului cum se poate observa în imaginile de mai sus, se apasă pe butonul „Select Folder” pentru a alege fișierul dorit pentru scanare.

Ca exemplu am selectat fișierul „CS IMAGE”, unde se afla mai multe poze, apoi, apasând pe butonul „Scan” programul ANTIVIRUS începe să scaneze conținutul fișierului și integritatea acestuia. În momentul în care s-a finalizat operațiunea, bara verde de sus este încarcată la maxim.

Cu verde în partea de jos este afișat ceea ce am scanat și locul unde se află fișierul, pentru a fii mai ușor de găsit pentru ștergere în cazul în care sunt probleme ce nu pot fii rezolvate de către program, iar cu roșu tot în partea de jos, sunt afișate problemele pe care programul ANTIVIUS le-a depistat, în cazul de față, afișează „0”, deoarece nu a găsit nicio problemă.





5. Concluzii

În timp îmi doresc ca acesta să devină un program antivirus eficient și complex, care să poata să combată atât programele maleware de orice tip, așa numiții viruși, dar să poată sa oprească atacurile de orice fel și totodată să depisteze și programele de tip keyloger.

Bibliografie:

1. Szor, Peter (2005), *The Art of Computer Virus Research and Defense*, Addison-Wesley, ISBN 0-321-30454-3
2. https://dnsc.ro/vezi/document/ghid-securitate-cibernetica-2021
3. https://www.arasec.ro/
4. https://www.raisa.org/
5. https://www.cyberlearning.ro/cybersecurity-guide/

INTERFAȚĂ PENTRU VIZUALIZAREA IMAGINILOR DE FUNDAL

Autor: Denis-Cristian RĂDULEA[1]
raduleadeniscristian@yahoo.com

Coordonator: Conf.univ.dr.ing. **Angela EGRI**[2]

[1] *Universitatea din Petroşani, Facultatea IME, Specializarea Automatică și Informatică Aplicată, anul II*
[2] *Universitatea din Petroşani, Facultatea IME, Departamentul ACIEE*

Rezumat

Aplicatia a fost dezvoltată în limbajul de programare Java, pentru sistemul de operare Android. Ea execută sarcini simple precum extragerea unei liste cu imagini predefinite dintr-o bază de date online, care se updatează automat la cererea utilizatorului. Pentru a se realiza cu succes cererea furnizarii listei cu imagini, este necesara doar o conexiune la internet pe dispozitivul respectiv, oferind un mod de vizualizare simplu dar eficient, de tip scroll in care se poate selecta imaginea dorită; și de asemena o previzualizare la scara 1:1 pentru aceasta. În prezent aplicația este disponibilă gratuit pe Google Play store, având peste 1000 de utilizatori internaționali activi. Un ultim punct important de menționat este că a fost testată de catre echipa de testeri Google înainte de a fi aprobată pentru listarea în magazin, si nu s-a gasit nicio eroare.

Cuvinte cheie

Android, Java 8, Google Play Store, Cloud Database

1. Introducere

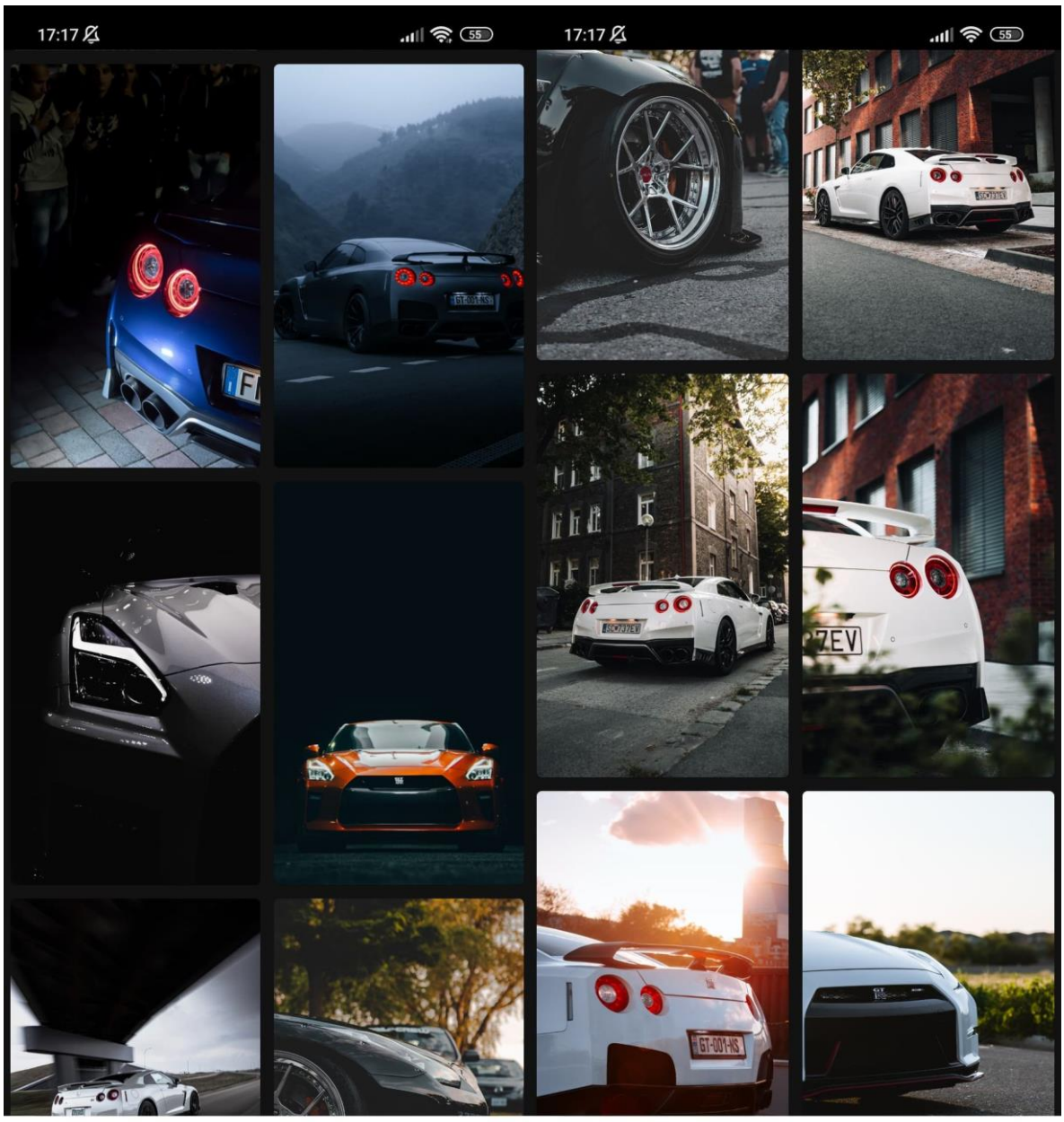
Aplicația este conceputa sub forma unei scurtături / shortcut pentru utilizator și o interfață mai prietenoasa sau mai accesibilă decât o căutare online unde trebuie să folosești filtre sau să fi extrem de specific în legătură cu ceea ce cauți. În cazul utilizării aplicației este deja existentă o varietate de poze de înaltă calitate cu poza dorită, fără a fi nevoie de prea multă filtrare și timp irosit. Limbajul de programare folosit initial a fost Kotlin, dupa care s-a trecut la Java 8, fiind o tehnologie mai noua , mai avansata si mai usor de folosit iar programul in care s-a scris codul in sine se numeste Android Studio.

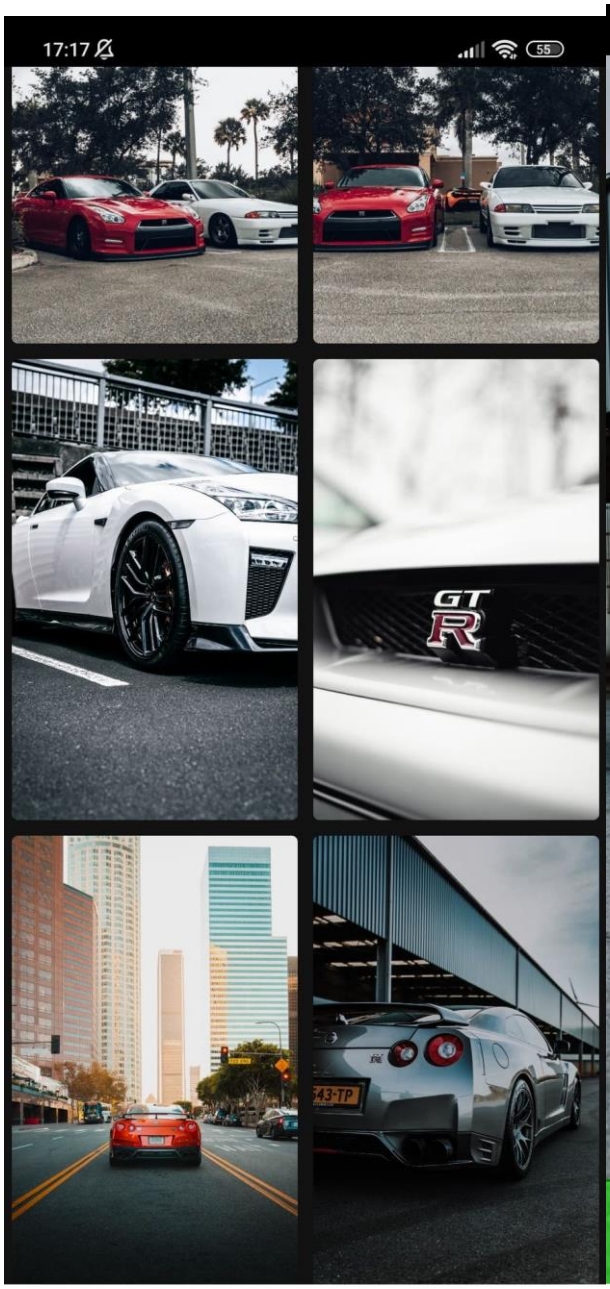
2. Scop

Scopul principal al aplicației este de a oferi accesibilitate și comoditate pentru utilizator la apăsarea unor simple butoane, cu o interfață ‘’curată’’ și intuitivă. Ca parte de scriere a software-ului efectiva a fost folosita documentatie oficiala de pe site-ul Java, care va fi inclus in bibliografie la sfarsitul materialului, si s-au urmat pasii respectivi pentru fiecare feature in parte (Exemplu: Creearea unui buton, schimbarea fontului, implementarea functionalitatii, afisarea textului etc).

3. Descrierea zonei / obiectivului studiat si concluzie

Obiectul final a fost obtinerea unei aplicatii complet functionale cu un design pe placulvutilizatorilor, cerintelor Google Play Store, cu o bună stare de funcționare și fară erori fără a irosi foarte mult timp. Mai jos sunt atasate cateva capturi de ecran in timp real din aplicatie, pentru a putea prezenta un rezultat concret. Aplicatie se deschide ca orice terță parte software descarcata din Google Play store, si anume prin atingerea ecranului dispozitivului, pentru intrarea in aplicatie, apoi folosirea functiei “scroll” inclusa pe dispozitiv, fie ea prin miscarea degetului in sus sau in jos pentru a cauta printre imagini. Dupa ce utilizatorul este satisfacut cu una dintre imagini, o poate atinge o data pentru selectarea acesteia, apoi va fi dus la un meniu cu previzualizarea pozei la o scara 1:1, cum se poate observa in figurile 4 si 5. Apoi, daca utilizatorul doreste sa o seteze ca imagine de fundal, atat ca si ecran de blocare, precum si ca imagine de fundal, se va apasa inca o data pe butonul “Set Wallpaper”, iar acesta isi va schimba culoarea in verde, confirmand schimbarea cu success a imaginii de fundal, cum se poate observa in figura 4.


17:17
55
17:17
55


17:17
GT
R


17:18
J-543-TP
Set wallpaper

Bibliografie:

1. .https://github.com/;https://stackoverflow.com/;https://docs.oracle.com/javase/8/docs/;
2. https://developer.android.com/guide;https://www.pexels.com/;https://unsplash.com/;
3. .https://firebase.google.com/docs;https://play.google.com/console/about/
4. https://kotlinlang.org/docs/home.htm
5. .https://www.google.com/intl/ro_ro/adsense/start/resources/;

TESTAREA STANDARDULUI DE CRIPTARE AES CU ALGORITMUL MODIFICAT PENTRU CREŞTEREA RATEI DE TRANSFER

Autor: Bogdan FUSTEI[1]
mihai_bogdan666@yahoo.com

Coordonator: Conf.univ.dr.ing. **Simona RÎUREAN**[2]

[1] *Universitatea din Petroşani, Facultatea IME, Specializarea Calculatoare anul IV*
[2] *Universitatea din Petroşani, Facultatea IME, Departamentul ACIEE*

Rezumat

În lucrarea de faţă sunt prezentate toate etapele necesare rulării AES conform algoritmului cu modificările propuse de mine. Sunt prezentate etapele necesare criptării şi decriptării informaţiilor ce pot fi transmise wireless. Acest algoritm poate fi implementat în protocoale specifice comunicării cu tehnologii novatoare care au viteze de transfer mari dar care nu necesită resurse hardware complexe. Modificările pe care le-am adus algorimului AES au ca scop îmbunătăţirea ratei de transfer a datelor prin scăderea complexităţii resurselor hardware necesare în sistemul de comunicaţii şi creşterea ratei de transfer a datelor fără să fie afectată securitatea datelor transmise wireless. Tocmai de aceea, un protocol de comunicaţii care are ca bază acest algoritm de criptare se poate implementa în tehnolgii noi, cum ar fi cea de comunicaţii în spaţiul de lumină vizibilă.

Cuvinte cheie

Criptare, transfer, expansiune.

1. Introducere

AES (Advanced Encryption Standard) este un standard folosit in criptarea simetrică și operează cu blocuri de 128 biți și chei de lungimi diferite:128, 192, 256 biți. La baza AES stă algoritmul Rijndael (Daemen and Rijmen-Belgia). Rijndael a fost adoptat ca standard AES în 2 octombrie 2000 fără a suferi modificări.

AES e bazat pe designul SP-network (substitution-network) iar standardul a înlocuit algoritmii DES și 3DES. Este format din trei variante, AES-128, AES-192 și AES-256, care sunt considerate foarte rapide pentru implementarea software sau hardware. Cifrurile AES operează pe blocuri de 128 de biți (16 octeți), folosind chei de 128, 192 și respectiv 256 de biți. AES-128 utilizează 10 runde de criptare, AES-192 utilizează 12 runde, iar AES-256 utilizează 14 runde. Toate aceste lungimi de chei asigură o protecție adecvată a informațiilor, până la nivelul SECRET de 128 de biți, iar 192 de chei de nivel TOP SECRET sunt de 256 de biți. În procesarea textului original, fiecare rundă este împărțită în 4 etape:

a) Substituția de octeți în blocul de intrare (S-box) presupune că fiecare element suferă o transformare neliniară, folosind un tabel de căutare cu proprietăți matematice speciale.

b) ShiftRows (permutarea rândurilor) permite lucrul la nivel de octet.

c) Coloane de înlocuire (MixColumns) este o operație matrice care combină 4 octeți fiecare.

d) Adăugarea cheii (key addition) este operația logicş XOR între blocul curent și cheie.

Ca și în cazul algoritmului DES, fiecare rundă de generare a cheilor AES utilizează o cheie secundară. Operațiile din etapa de procesare AES folosesc calcule pe câmpuri Galois.

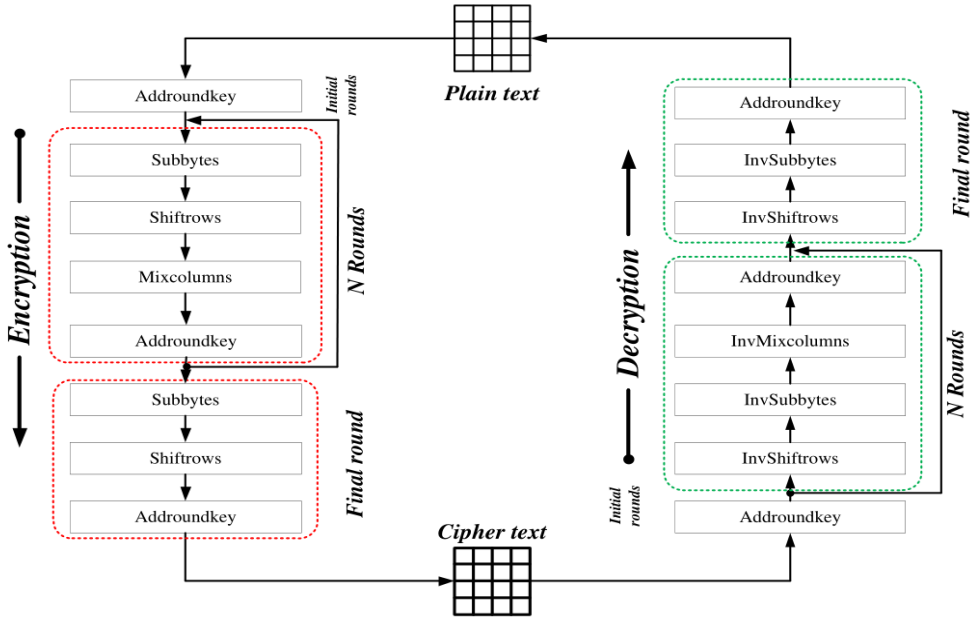


Fig.1. *Schema logică a algorimului AES clasic*

Câmpul finit conține 256 de elemente, folosind notația GF (2^8). Acest câmp a fost ales deoarece fiecare element poate fi reprezentat printr-un octet. Pentru pașii de înlocuire a octeților și coloanelor, AES tratează fiecare octet din blocul de date ca pe un element din câmpul GF (2^8) și realizează operații aritmetice în acest câmp. În runda 10 a AES-128 biți transformarea mix columns nu se realizează. În figura 1 este prezentată schema logică a algoritmului AES clasic.

2. Descriere metodei propuse

Algoritmul AES a fost studiat intens de specialiști de-a lungul timpului, și, deși nu este singurul algoritm existent, este considerat unul dintre cele mai eficiente dacă nu cel mai eficient algoritm pentru criptarea simetrică. Deși excelent, datorită volumului mare de calcule AES afectează capacitatea de stocare și consumă mare parte din resursele hard de procesare. Metoda propusă va aduce modificări procesului de criptare/decriptare.

2.1. Procesul de criptare

Considerăm funcția ca f_{ct} fiind funcția textului codificat, cu următoare formulă:

$$f_{ct} = P \oplus K \quad (1)$$

unde:

P - Textul simplu (Plain text);

K - cheia.

1). Vom alege textul simplu pentru procesul de criptare și o cheie de 128 biți. Vom împărți textul și cheia într-o matrice de 4×4.

2). Asupra textului simplu se va efectua operația XOR cu cheia inițială

$$C_0 = P \oplus K_0 \quad (2)$$

unde:

$P -$ textul simplu;

$C_0 -$ textul codat

$K_0 -$ cheia inițială

3). După operația XOR, toate datele obținute vor fi rearanjate într-o matrice 4×4. Avem 16 biți de date pentru cheie și textul codat.

4). Cheia inițială va fi modificată pentru runda a doua. Realizăm o deplasare la stânga a primului rând din matricea 4×4. În mod analog al doilea rând realizează 2 ori o deplasare la dreapta. Al treilea rând este schimbat după blocul de intrare (S-box), ce este implementat în baza noatră de date. În al patrulea rând folosim o tehnică suplimentară pentru a schimba valoriile, obținem astfel cheia a doua K_1.

5). $C_1 = C_0 \oplus K_1$.

6). Astfel vom obține încă 9 chei cu această ethnică de ajustare și XOR, fiecărei chei îi corespunde un nou text codat. $K = (0, 1, \dots 9)$ și $C = (0, 1, \dots, 8)$ după 10 runde.

7). A zecea cheie va fi obținută printr-un alt tip de modificări: vom urma tot procesul de ajustări cu excepția celui de-al 4 rând care va fi păstrat neschimbat. Astfel a zecea cheie este obținută: K_{10}.

8). Asupra ultimei chei se va realiza operația XOR cu ultimul text codat obținut C_9, și vom obține textul codat principal: $C_{10} = C_9 \oplus K_{10}$.

In tabelul 1. este prezentată sursa (S-box) de sustituire a bitilor pe byte in xy (format hexadecimal) pentru faza de codare.

Tabelul 1. *Procesul de criptare*

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

2.2. Procesul de decriptare

Pentru procesul de decriptare considerăm funcția f_{pt}

$$fpt = C \oplus K \tag{3}$$

unde:

fpt este funcția textului simplu(plain text)

$C -$ textul codat; $K -$ cheia

Pașii pentru procesul de decriptare sunt următori:

1). Avem textul codificat principal C_{10} și cheia primită pe care o vom considera a 11-a cheie K_{10}. Acestea vor fi convertite în forma unor matrici de 4×4, pentru folosirea lor în procesul decriptării.

2). Asupra celor două matrici obținute la punctul de mai sus vom aplica operația XOR. Obținem astfel textul codat C_9 (al zecelea text codat). Deci vom putea scrie că:

$$C_9 = C_{10} \oplus K_{10}.$$

3). Cheia numărul 11 va fi supusă procesului reversibil. Vom deplasa spre dreapta primul rând. Analog, al doilea rând va fi deplasat spre stânga de 2 ori, invers procesului criptării. Al treilea rând va fi substituit de o valoare ce este aleasă din S-box inversat, ce a fost implementat sub forma unei baze de date în program. Vom obține astfel cheia a zecea K_9.

4). Al nouălea text codat C_8 este obținut astfel:

$$C_8 = C_9 \oplus K_9.$$

5). Vom modifica în continuare cheia zece, asupra primelor 3 rânduri vom urma etapele de la punctul 3. Asupra randului al patrulea vom adăuga valoarea decrementiilor. Astfel vom obține cea de-a noua cheie K_8.

6). Aplicăm XOR asupra C_8(textul codat 9) și K_8(cheia a noua) pentru a obține textul codat C_7(textul codat al 8-lea). În mod analog cheile următoare vor parcurge pași prezentați anterior (relație de recurență) până când vom obține textul codat și cheia cu numărul 0. Vom obține astfel următoarele:$K = (10, 9 \dots 0)$ și $C = (10, 9 \dots 0)$.

7). Vom aplica XOR asupra textului codat și a cheii rămase obținând astfel textul dorit. Deci, vom putea scrie următoarele:$P = C_0 \oplus K_0$, unde P este textul simplu (plain text).

În tabelul 2 sunt prezentate valorile de substituție Inverse S-Box pe Byte în coordonate xy.

Tabelul 2. *Procesul de decriptare*

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

2.3. Expansiunea cheii

Expansiunea cheii se referă la procesul în care toate cheile sunt transformate și schimbate în una singură.

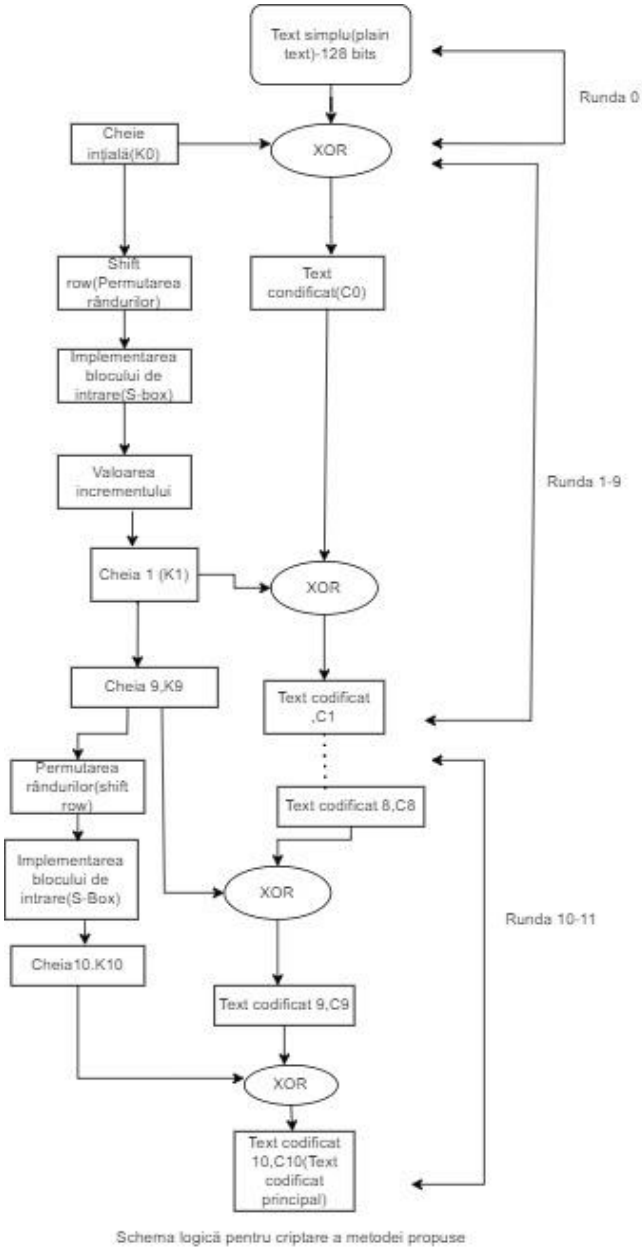
Cheia va trece prin 3 etape diferite de trasformare, ce vor fi implementate pe rând. Acești pași sunt următori:

1). Operația de permutare a rândurilor (shift row) este executată. Rândurile sunt mutate de la un block la altul. Când o cheie este reprezentată printr-o matrice de 4×4 primele 2 rânduri ale acestei matrici trec prin această etapă.

2). Este implementat procesul de substituiere a octeților din blocul de intrare(implementarea S-BOX). Toate calculele se vor realiza în rândul cu nr. 3 din matrice. Ne vom folosi de tabelele S-BOX și Inverse S-BOX. S-BOX va fi folosit în procesul de criptare iar Inverse S-BOX va fi utilizat în realizarea decriptării

3). În ultima etapă are loc procesul valorii Incrementului și a Decrementului, calculele realizându-se în rândul nr. 4 din matricea corespunzătoare cheiilor. Se vor aduna/elimina octeți în fiecare al 4 rând din matrice. Incrementul v-a fi folosit pentru criptare iar decrementul v-a fi folosit pentru decriptare. Ultima cheie din procesul de criptare și prima cheie din procesul de decriptare NU vor fi supuse acestor calcule.

In figurile 2 și 3 sunt prezentate schemele logice pentru criptarea, respectiv decriptarea datelor conform algoritmului cu modificările propuse.



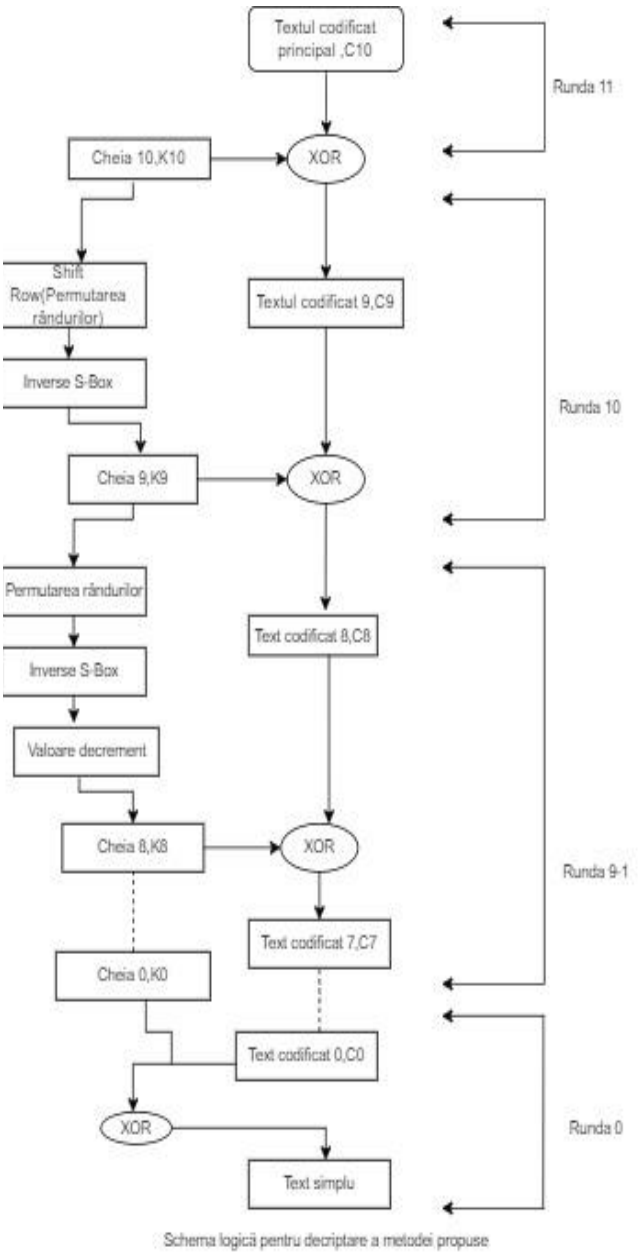


Fig.2. *Schema logică pentru criptarea* **Fig.3.** *Schema logică pentru decriptarea*

3. Testarea algoritmului modificat

In figura 4 este prezentată expansiunea cheii pentru procesul de criptare (caracterizat de valoarea incrementului).

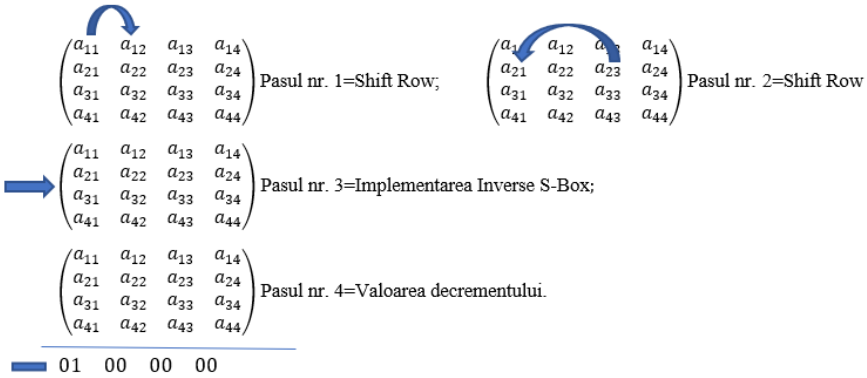


Fig. 4. Expansiunea cheii pentru procesul de criptare

Toți cei 4 pași se vor realiza pentru cheile: K_0 până K_9.

Primi 3 pași se vor realiza pentru cheile: de la K_9 până la K_{10}.

In figura 5 este prezentată expansiunea cheii pentru procesul de decriptare (caracterizat de valoarea decrementului).

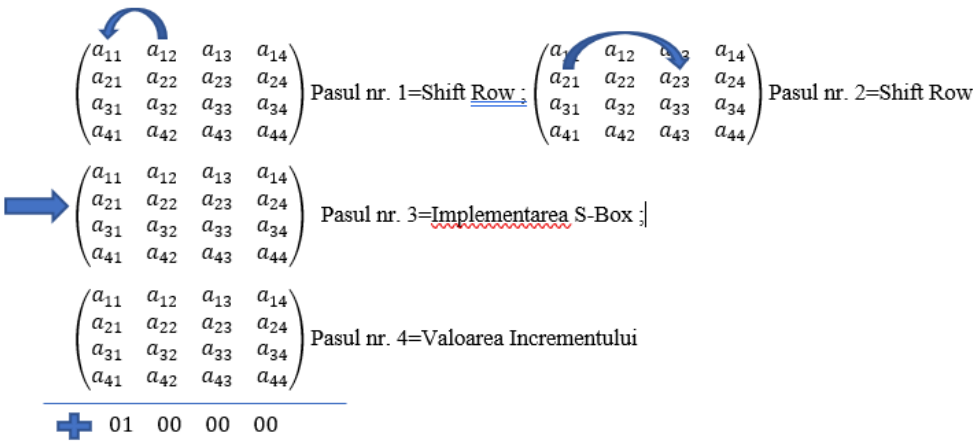


Fig. 5. *Expansiunea cheii pentru procesul de decriptare*

Toți cei 4 pași se vor realiza pentru cheile:K_9 până la K_0.

Primi 3 pași se vor realiza pentru cheile:K_{10} până la K_9.

În figurile 6 și 7 este prezentată rularea în aplicație algoritmul prezentat cu modificările aduse AES.

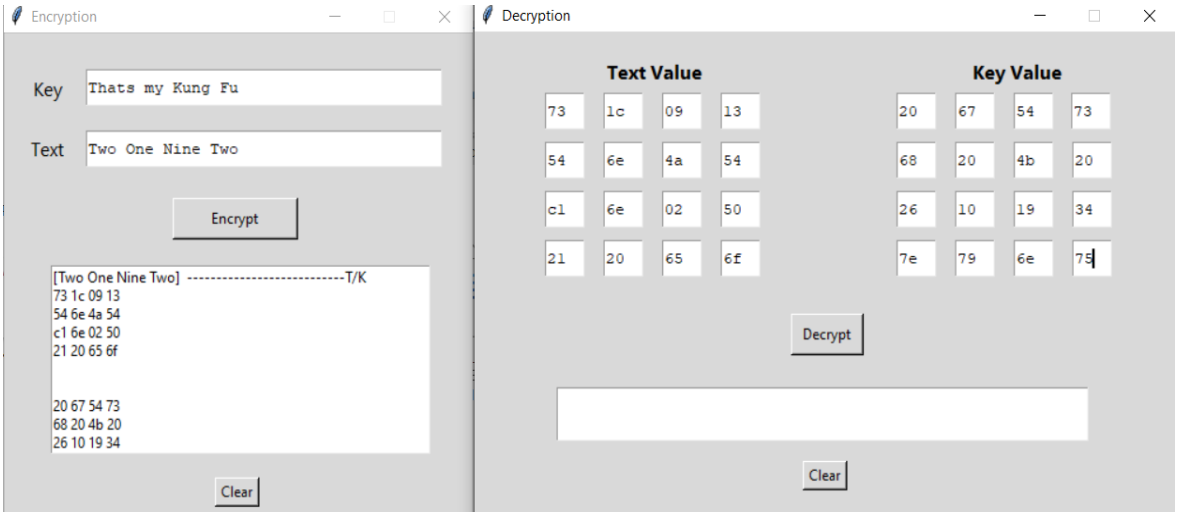


Fig. 6. *Criptarea şi decriptarea textului*

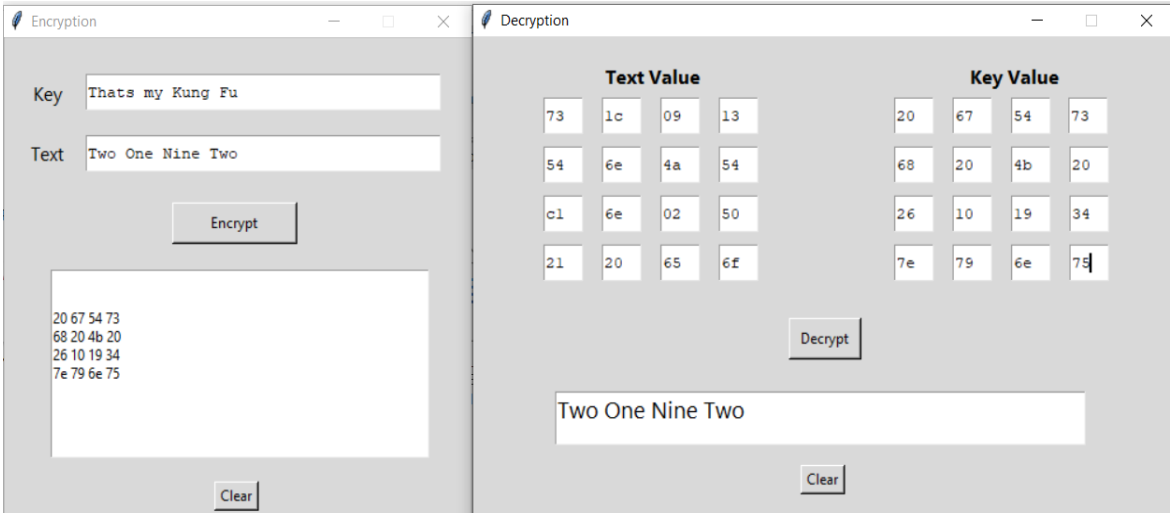


Fig. 7. *Criptarea şi decriptarea textului şi rezuştatul decriptării*

4. Concluzii

Prin prezentarea operațiilor ce au loc în AES-128 modificat și schemelor logice corespunzătoare proceselor de criptare, respectiv decriptare etapa Mix Columns este eliminată, împreună cu toate calculele complexe din cadrul acesteia. Securitatea algoritmului modificat rămâne aceeași ca a algoritmului clasic.

Ambele etape rămân cu un grad de vulnerabilitate minor în fața atacurilor brute. Datorită eliminarii calculelor complexe (realizate în cadrul etapei Mix Columns) algoritmul modificat devine automat mai rapid decât algoritmul clasic.

De asemenea, datorită eliminării etapei Mix Columns presiunea pe resursele hardware va fi una cu mult mai scăzută.

AES-128 modificat va putea fi implementat în toate aplicațiile unde AES-128 clasic este implementat, însă viteza algoritmului modificat fiind una mai bună decât cea a algoritmului clasic.

Bibliografie:

1. Daemen Joan, Rijmen Vincent , *The Design of Rijndael The Advanced Encryption Standard* (AES), Editura Spinger 2020.
2. Forhad M. S. A., Riaz S., Hossain M. S., and Das M, *An improvement of advanced encryption standard*, INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND NETWORK SECURITY, vol. 18, no. 11, pp. 159–166, 2018.
3. Riurean S., Leba M. and L. Crivoi, *Enhanced Security Level for Sensitive Medical Data Transmitted through Visible Light, 2021* International Symposium on Networks, Computers and Communications (ISNCC), 2021, pp. 1-6, doi: 10.1109/ISNCC52172.2021.9615732.
4. Stallings William (2005). *Cryptography and Network Security*, 4th edition. Prentice Hall. ISBN 0-13-187319-3
5. Zobaed S., Salehi M. A., Zomaya A., and S. Sakr, *Big data in the cloud*, Encyclopedia of Big Data Technologies, Editors: Sherif Sakr, Albert Y. Zomaya, 2019 Edition.