әл-Фараби атындағы Қазақ ұлттық университеті
ҚР БҒМ ҒК Ақпараттық және есептеуіш технологиялар институты

ӘОЖ 004.421

Қолжазба құқығында

ХОМПЫШ АРДАБЕК

Позициялық емес санау жүйесін қолдану арқылы ақпаратты қорғау алгоритмін құру және зерттеу

6D100200 – Ақпараттық қауіпсіздік жүйелері

Философия докторы (PhD)
дәрежесін алу үшін дайындалған диссертация

Отандық ғылыми кеңесші:
т.ғ.к., Капалова Нурсулу
Алдажаровна
(Ақпараттық және есептеуіш
технологиялар институты,
Алматы, Қазақстан)

Шетелдік ғылыми кеңесші:
Doctor of Science, Pofessor,
Muslum Arıcı
(Kosaeli University, Izmit, Turkey)

Қазақстан Республикасы
Алматы, 2021

МАЗМҰНЫ

НОРМАТИВТІ СІЛТЕМЕЛЕР

Бұл диссертацияда келесі стандарттарға сілтемелер қолданылды:

1. ҚР Үкіметінің 2017 жылғы 30 маусымдағы № 407 қаулысымен бекітілген Киберқауіпсіздік («Қазақстанның киберқалқаны») ТҰЖЫРЫМДАМАСЫ.
2. Киберстратегия және ақпараттық қауіпсіздікті басқару ISO/IEC 27001:2013 сәйкес және сыртқы және ішкі шабуылдардан ақпаратты алдыңғы қатарлы қорғау тәжірибелерімен ақпараттық қауіпсіздікті басқаруды бағалау, енгізу және дамыту.
3. Қазақстан Республикасы Үкіметінің 2017 жылғы 12 желтоқсандағы № 827 қаулысымен бекітілген "Цифрлық Қазақстан" мемлекеттік бағдарламасы.
4. «Диссертацияларды және авторефераттарды рәсімдеу бойынша нұсқаулық», ҚР БҒМ, Жоғары аттестаттау комитеті, Алматы, 2004. МЕСТ 7.1-2003. Библиографиялық жазба.
5. ГОСТ 7.32-2001- Ғылыми-зерттеу жұмысының есебі.

БЕЛГІЛЕУЛЕР МЕН ҚЫСҚАРТУЛАР

ПЕСЖ - позициялық емес санау жүйесі
ПЕПСЖ - позициялық емес полиномды санау жүйесі
ҚКСЖ - қалдықтар класындағы санау жүйесі
ЕМ - exponentiation modul
SP - substitution-permutation (алмастыру-орын ауыстыру)
GF - Galois field (Галуа - өрісі)

КІРІСПЕ

Зерттеу тақырыбының өзектілігі. Бүгінгі таңда Қазақстан индустриялық қоғамнан қазіргі ғылыми-технологиялық революцияның қатаң талаптарымен айқындалатын қоғамдық және экономикалық дамудың түбегейлі жаңа деңгейіне көшудің тарихи қажеттілігіне тап болып отыр. Себебі көптеген дамыған елдерде ақпараттық қоғам мен ақпараттық экономиканың жоғары деңгейде дамығанын ескерсек, онда Қазақстанда бұл мәсселелерді қалыптастыру өзекті мәселелердің бірі. Ал ақпараттық қоғамда оның материалдық қоры ақпараттық экономика екендігі анықталса, қажеттілігі ақпараттық ресурсқа ауысады. Бұл кезде ақпараттық ресурстар басқа қолданушылардың рұқсат етілмеген қол жетімділігінен тұрақты қорғауды қажет ететін елдің стратегиялық ресурстары ретінде орындалады.

Автоматтандырудың жоғары дәрежесі, компьютерлік жүйелердің адам қызметінің әртүрлі салаларына деструктивті кеңінен енгізілуі деректерді өңдеудің автоматтандырылған жүйелерін әр түрлі іс-әсерлерге қатысты өте осал етеді және қоғамды пайдаланылатын ақпараттық технологиялардың қауіпсіздік деңгейіне тәуелді етеді. Сондықтан ол арқылы таралатын ақпараттың қауіпсіздігі кез-келген компьютерлік жүйенің күрделілігі мен мақсатына қарамастан маңызды сипаттамасына айналады.

Диссертациялық жұмыстың **өзектілігі** - рұқсатсыз қол жетімділіктен құпия ақпаратты қорғау міндеті қазіргі кездегі ең көне және толығымен шешілмеген мәселелердің бірі болып табылады.

Сонымен қатар кез-келген мемлекеттің даму стратегиясында басым бағыттарының бірі ұлттық қауіпсіздік екендігін ескерсек, онда оның ең маңызды элементтерінің бірі ақпараттық қауіпсіздік болып табылады. Сондықтан ақпараттық қауіпсіздіктің жаңа технологияларын құру мәселесін шешу, оған қолжетімділікті шектеп, ақпаратты қорғаудың қажетті деңгейін қамтамасыз ету заманауи талаптарға сай келетін ақпараттық қауіпсіздік құралдарын құру өзекті мәселелердің бірі.

«Қазақстанның киберқалқаны» (2017 жылғы 30 маусымдағы) киберқауіпсіздік тұжырымдамасында «Зерттеулерге және қолданбалы математика, ақпаратты криптографиялық қорғау құралдарын әзірлеу, криптология, бағдарламаланатын логикалық интегралдық схемалар бойынша әзірлемелер, кванттық криптография мен ақпарат берудің, өңдеудің және сақтаудың қорғалған жүйелерін, сондай-ақ ақпараттық қауіпсіздік жүйелерін құру бойынша өз мектептерімізге басымдық беру керек» және «Отандық әзірлемелердің сұранысқа аса ие болмау мәселесін жою, өйткені киберқауіпсіздік түптен келгенде отандық IT-саласының және электронды өнеркәсібінің даму деңгейіне байланысты» қажеттілігі көрсетілген [1].

Осыған байланысты отандық криптографиялық құралдар құру бойынша ғылыми зерттеу жұмыстарын жүргізу еліміз үшін **өзекті** болып табылады.

Бұл, ең алдымен, ғылыми-техникалық прогресстің үнемі өсіп келе жатқан қарқынымен байланысты, бұл компьютерлік технологиялардың жетілдірілуіне алып келеді. Олардың пайда болуы қауіпсіздіктің жаңа мәселелерін көтеріп

қана қоймай, сонымен қатар шешілген мәселелерді жаңа тұрғыда ұсынады, ал ақпаратты қорғау проблемасын шешудің күрделілігіне төмендегілер ықпал етеді [2,3]:

- компьютерлік технологияны қолдану арқылы жинақталған, сақталатын және берілетін ақпарат көлемінің ұлғаюы;
- компьютерлік жүйелердің ресурстарына қол жетімділігі бар пайдаланушылар шеңберін кеңейту;
- компьютерлік жүйенің техникалық құралдарының жұмыс режимдерінің күрделенуі;
- деректерді өңдеудің автоматтандырылған жүйелеріндегі техникалық құралдар мен байланыстар санының көбеюі;
- жаңа инфокоммуникациялық технологиялардың кеңінен таратылуы.

Рұқсат етілмеген қол жетімділіктің салдарын азайту үшін қауіпсіздік жүйесін құру қажет. Мұндай жүйені құрудың мақсаты қасақана немесе кездейсоқ деструктивті қатерлерден алдын алу, оның нәтижесінде ақпарат жоғалу, модификациясы немесе өзгертілуі мүмкін. Бұл жағдайда тиімді қауіпсіздік жүйесін құру үшін төмендегілерді қамтамасыз етуі керек [4]:

- барлық ақпараттың немесе оның маңызды бөлігінің құпиялылығы;
- ақпараттың сенімділігі (толықтығы, нақтылығы, дұрыстығы, тұтастығы, түпнұсқалығы), кез-келген уақытта жүйе компоненттерінің жұмыс қабілеттілігі;
- пайдаланушылардың өздеріне қажет ақпараттық және жүйелік ресурстарға уақытылы қол жетімділігі;
- ақпараттық байланыстың белгіленген ережелерін бұзғаны үшін жауапкершілікті күшейту;
- ақпаратты басқару, түзету және алмасу кезеңдерін жедел қадағалау.

Қорғау объектісіне байланысты қауіпсіздік жүйесінің анықталған қасиеттерінің арасында басымдықтардың әр түрлі орналасуы мүмкін екенін атап өткен дұрыс. Осылайша, мемлекеттік құпияларды қорғау мәселесіне ақпараттың құпиялылығына баса назар аударылады. Бұл мақсаты қауіптің ықтималдығын азайту арқылы немесе қауіпті іске асырудың салдарын азайту болып табылатын қарсы қауіп-қатерлерді анықтайды. Бұл шаралар бірлесіп қауіпсіздік саясатын қалыптастырады. Көптеген шетелдік және отандық ғалымдардың жүргізген зертеулері көрсеткендей, көптеген ұйымдастырушылық, әдістемелік және техникалық шаралар арасында ақпаратты криптографиялық қорғау әдістері ерекше орын алады.

Заманауи криптографиялық әдістер, оның ішінде итеративті блоктық шифрлар жылдамдығы жоғары ақпарат тарату желілерінде қауіпсіз ақпарат алмасуды қамтамасыз ететін, сұранысқа ие құралдардың бірі болып табылады. Ақпараттық технологияларды кеңінен қолдану және есептеу қуатының қарқынды дамуы белгілі шифрлардың криптоталдауына қауіп тудырады.

Мәліметтерді криптографиялық қорғау құралдарын құруға бағытталған зерттеулер көбінесе мемлекеттік құпияларға байланысты, сондықтан шетелдік дайын шешімдерді қолдану қауіпсіз емес. Отандық ақпаратты криптографиялық қорғау құралдарын құру, оның ішінде шифрлау

алгоритмдерін құру бойынша зерттеулер жүргізу өзекті және қажетті болып табылады.

Диссертациялық жұмыстың мақсаты - позициялық емес санау жүйесінің (ПЕСЖ) мүмкіндіктерін қолдану арқылы ақпаратты шифрлау алгоритмін құру, құрылған алгоритмнің криптоберіктілігін талдау, алгоритмді бағдарламалық жүзеге асыру болып табылады.

Зерттеу мақсатына жету үшін келесі міндеттер қойылды:

- ақпаратты криптографиялық қорғау әдістеріне шолу және талдау;
- ақпаратты криптографиялық қорғау жүйелеріне қойылатын талаптар мен өнімділік критерийлерін талдау;
- позициялық емес полиномды санау жүйесі негізінде симметриялық блокты шифрлау алгоритмін құру;
- құрылған симметриялық блоктық шифрлау алгоритмін бағдарламалық жүзеге асыру;
- құрылған симметриялық блоктық шифрлау алгоритмінің криптоберіктілігін зерттеу.

Зерттеудің нысаны - криптографиялық қорғау алгоримдері және оларға талдау жүргізу әдістері.

Зерттеудің пәні - позициялық емес полиномды санау жүйесі негізінде құрылған шифрлау және раундық кілттерді түзу алгоритмдері.

Зерттеу әдістері - модулді арифметика, позициялық емес полиномды санау жүйесі, статистикалық тестер, биттік шашырау критерийлері, криптоталдау әдістері.

Зерттеудің ғылыми жаңалығы. Ақпараттарды қорғау мәселесі әліде шешімін таппаған күрделі мәселелердің бірі. Осы мәселелерді назарға алып, ақпаратты қорғау мәселесін шешу алгоритмі ұсынылып, біршама жаңа жетістіктерге қол жеткізілді, бұл ғылыми нәтижелер жоғары рейтингті журналдарда жарияланды. Сол ғылыми нәтижелер жаңалығы диссертациялық жұмыста негізге алынды. Атап айтар болсақ:

- ЕМ түрлендіру әдісін қолдану арқылы жаңа симметриялы блоктық шифрлау алгоритмі құрылды;
- криптоталдау талаптарын қанағаттандыратын S-блок алмастыру кестесі құрылды;
- раундттық кілттерді жасау алгоритмі құрылды;
- шифрлау жылдамдығын арттыру мақсатында таңдап алған жұмыс негіздерінің индекс кестесі құрылды.

Жұмыстың теориялық және практикалық маңызы. Диссертациялық зерттеуде алынған нәтижелер телекоммуникациялық және ақпараттық жүйелер мен желілердегі, электрондық құжат айналым жүйелеріндегі ақпараттарды сонымен қатар отандық ақпараттық-коммуникациялық технологиялардың бағдарламалық өнімдерін, мемлекеттік және жеке тұлғаның рұқсат етілмеген құпия мәліметтерін бөгде адамдардың ұрлауы, өзгертуінен қорғау үшін пайдалануға қолдануға болады. Сонымен қатар жоғарға оқу орындарында оқу үрдістерінде пайдалануға, сондай-ақ жаңа ақпаратты криптографиялық қорғау жүйелері әзірленуде қолдануға болады.

Қорғауға шығарылған негізгі тұжырым. Қазіргі заманауи симметриялық блокты шифрлеу алгоритмдерінің негізгі талаптарына сай келетін ақпараттарды қорғау ЕМ түрлендіру әдісін қолдану арқылы жаңа симметриялы блоктық шифрлау алгоритмі құрылды.

Сонымен қатар алгоритмге қолданылған жаңа S-блок алмастыру кестесін алу әдісі ұсынылып, S-блокқа сызықты және дифференциалды криптоталдау жүргізілді және алынған нәтижелер белгілі алгоритмдермен салыстырылды. Алгоритмнің шифрлау жылдамдығын арттыру мақсатында позициялық емес полиномды санау жүйесі және таңдап алған жұмыс негіздерінің индекс кестесі қолданылды.

Қорғауға ұсынылатын нәтижелер. Жаңа блокты шифрлау алгоритмі құрылды. Алгоритмнің криптоберіктілін анықтау мақсатында бірнеше криптоталдау әдістері арқылы талдаулар жүргізіліп нәтижелері ұсынылды.

Зерттеу нәтижелерін жүзеге асыру. Диссертатциялық жұмысты зерттеу кезеңінде алынған нәтижелер Ақпараттық және есептеуіш технологиялар институтының Ақпараттық қәуіпсіздік зертханасында тексеріліп, BR05236757 - «Жалпы мақсаттағы желілер мен инфокоммуникациялық жүйелерде ақпаратты жіберу және сақтау кезінде оны криптографиялық қорғау үшін бағдарламалық және бағдарламалық-аппараттық кешендерді құрастыру» атты жобада жүзеге асырылды.

Диссертация нәтижелерінің апробациясы. Зерттеу жұмысының басты басты нәтижелері төмендегі конференциялар мен семинарларда баяндалып талқыланды:

– «Көліктегі инновациялық технологиялар: білім, ғылым, тәжірибе" атты XLI Халықаралық ғылыми-практикалық конференциясы (3-4 сәуір, 2017, Алматы, Қазақстан);

– Ақпараттық және есептеуіш технологиялар институтының «Информатика және есептеу технологияларының қазіргі заманғы мәселелері» ғылыми конференциясы (29-30 маусым, 2017, Алматы, Қазақстан);

– II Халықаралық «Информатика және қолданбалы математика» ғылыми-практикалық конференциясы (27-30 қыркүйек, 2017 ж., Алматы, Қазақстан);

– III Халықаралық «Информатика және қолданбалы математика» ғылыми-практикалық конференциясы (26-29 қыркүйек, 2018 ж., Алматы, Қазақстан);

– «XXI ғасыр ғылымы: жаңа көзқарас»: студенттердің, аспиранттардың және жас ғалымдардың XXIII халықаралық ғылыми-практикалық конференциясы (22-23 мамыр, 2019 ж., г. Санкт-Петербург, Россия);

– IV Халықаралық «Информатика және қолданбалы математика» ғылыми-практикалық конференциясы (25-29 қыркүйек, 2019 ж., Алматы, Қазақстан);

– «Қазақстандағы ақпараттық қауіпсіздігінің өзекті мәселелері (АҚӨМ-2020)» халықаралық ғылыми-практикалық конференциясы (15 қаңтар, 2020 ж., Алматы, Қазақстан);

– «Ақпараттық және есептеуіш технологиялар» институты «Информатика, математика және басқарудың өзекті мәселелері» атты ғылыми-практикалық семинарлары (2017-2020, Алматы, Қазақстан);

– Әл-Фараби атындағы Қазақ ұлттық университеті «Ақпараттық технологиялар» факультеті ғылыми семинарлары (2017 - 2020 жж., Алматы, Қазақстан).

Жарияланымдар. Диссертацияның негізгі нәтижелері бойынша 14 мақала жарияланды және 1 авторлық куәлік алынды. Оның ішінде 1 мақала халықаралық рецензияланатын мерзімді басылымдарда, 6 мақала ҚР БҒМ-нің Білім және ғылым саласы бойынша бақылау комитеті ұсынған ғылыми баспаларда, 7 мақала Қазақстан мен шетелдердегі халықаралық ғылыми конференциялар жинақтарында жарияланды. Диссертациялық жұмыс бойынша шыққан мақалалар пайдаланылған әдебиеттер тізімінде келтірілді. Бағдарламалық кешенге алынған авторлық құқық куәлігі қосымшада берілді.

Диссертация құрылымы және көлемі. Диссертациялық жұмыс құрылымы кіріспеден, 3 бөлімнен, қорытындыдан, пайдаланылған әдебиеттер тізімінен және 4 қосымшадан тұрады. Зерттеу жұмысының жалпы көлемі 88 бет, оның ішінде 26 сурет, 28 кесте қамтылған.

Кіріспеде қазіргі кездегі пайдаланушылардан рұқсат етілмеген ақпараттарды криптографиялық қорғаудың заманауи блоктық алгоритмдерге қойылатын талаптарға сай келетін алгоритмдерін құру мәселесінің өзектілігін анықтайды, жұмыстың мақсатын сипайтайды, ғылыми мақсатына жету үшін қойылатын міндеттерді белгілейді, зерттеу нысаны мен пәнін, негізгі қорғауға шығарылатын тұжырым, қорғауға ұсынылған диссертациялық жұмыстың жаңалығы, сынақтан өткізу және оның нәтижелері көрсетілген.

Бірінші бөлімде Диссертациялық жұмыстың өзектілігінде көрсетілген мәселелерді шешудің ең тиімді әдістерінің бірі криптографиялық әдістер екендігін ескерсек, онда бұл бөлімде криптографиялық әдістері туралы негізгі түсініктер, терминдер және криптографиялық түрлендірудің негізгі класстары, симметриялық блокты шифрлау алгоритмдерін құру әдістерінің негізгі кезеңдері, блокты шифрлау алгоритмне қойылатын талаптар қарастырылды.

Екінші бөлімде Симметриялық блокты шифрлау алгоритмдерінің негізгі талаптарын ескере отырып жаңа блокты шифрлау алгоритмі құрылып, ұсынылып отырған алгоритмде қолданылған позициялық емес полиномды санау жүйесінің құру жолдары, алгоритмнің негізгі параметрлері, алгоритмнің шифрлау және шифрды ашу сұлбасы, сонымен қатар алгоритмге қолданылған түрлендіру әдістері ЕМ түрлендіру әдісі, S-блок алмастыру кестесін алу әдісі, таңдап алған жұмыс негіздерінің индекс кестесін алу және дәрежені жылдам есептеу процессі, раундттық кілттерді жасау алгоритмі сипатталды.

Үшінші бөлімде диссертациялық жұмыста ұсынылған жаңа блокты шифрлау алгоритмінің бағдарламалық кешен құрылып сипатталды. Алгоритмнің криптобериктілін тексеру үшін шифрмәтіннің статистикалық қәуіпсіздігі бағалау және графикалық тесттер арқылы тексеріліп нәтижелері ұсынылды. Сонымен қатар алгоритмнің биттік шашырау критериилері арқылы зерттеу, S-блоққа сызықты және дифференциалды криптоталдау, алгоритмнің

барлық түрлендіру процесіндегі дифференциалды талдау нәтижелері сипатталған.

Қорытындыда жұмыстың негізгі қорытындылары мен нәтижелері тұжырымдалды.

Қосымша А: EMCipher шифрлау алгоритмінің бағдарламасы C++ тілінде Microsoft Visual Studio 2013 бағдарламалау ортасында құрылған бағдарлама кодының үзіндісі келтірілді.

Қосымша Ә: «CryptoEM v1.0.1», ЭЕМ-ге арналған бағдарламаға алынған №5450, 24 қыркүйек 2019 ж. авторлық құқық куәлігі тіркелді.

Қосымша Б: Алгоритмнің диссертацияның негізгі бөлімінде көрсетілмеген раундтардағы биттік шашырау нәтижелері, ЕМ түрлендірудің керісі және таңдалған жұмыс негіздерінің индекс кестесін қолдану арқылы алгоритмнің шифрлау жылдамдығы артатыны анықталған салыстыру нәтижелері, раундттық кілттерді жасау алгоритміне қолданылған S-блок алмастыру кесетсі сипатталды.

Қосымша В: Диссертациялық жұмыстың нәтижелерін іске асыру туралы АКТ ұсынылған.

1 АҚПАРАТТЫ КРИПТОГРАФИЯЛЫҚ ҚОРҒАУ ӘДІСТЕРІНЕ ШОЛУ ЖӘНЕ ТАЛДАУ

1.1 Ақпаратты қорғаудың криптографиялық әдістерін талдау

Ақпаратты өңдеу мен басқарудың автоматтандырылған жүйелерінде компьютерлік технологияларды кеңінен қолдану компьютерлік жүйелердегі ақпаратты рұқсатсыз пайдаланудан қорғау қиындықтарының шиеленісуіне әкеліп соғады. Сонымен қатар компьютерлік жүйелердегі ақпараттарға шабул жасайтын көптеген қауіптер бар, оларды сыртқы бұзушылар да, ішкі бұзушылар да жүзеге асыра алады. Сондықтан ақпараттарға жасалатын шабулдардың көбі электронды ақпараттарға жасалатыны белгілі.

Электрондық ақпаратты қорғау қиындықтарын түбегейлі шешу қорғалған автоматтандырылған өңдеу мен деректерді берудің маңызды мәселелерін шешуге мүмкіндік беретін әдістер ретінде криптографиялық әдістерді пайдалану арқылы оң шешім алынады.

Ақпаратты криптографиялық қорғау әдістері - бұл ақпаратарды шифрлау, кодтау және өзге де түрлендірудің арнайы әдістерін пайдалана отырып оның мазмұны шифрмәтінге айналдырады, ал шифрмәтін кілтті білмей кері түрлендіруге қолжетімсіз болады [5,6]. Ақпаратты криптографиялық қорғау ақпараттарды рұқсатсыз пайдаланудан қорғаудың негізгі құралы болып табылады. Бүгінгі таңда дүниежүзінде цифрлық технологиялардың белсенді дамуына байланысты ақпаратты қорғаудың криптографиялық әдістерімен байланысты ғылым салалары да қарқынды дамуда.

Ақпаратты қорғау мәселелерімен криптология (κρυπτος-құпия, λογος-ғылым (сөз), грек тілінен аударғанда) айналысады.

Криптология криптография және криптоталдау болып екі бағытқа бөлінеді. Бұл бағыттардың мақсаты бір бірімен тікелей қарама қарсы.

Криптография ақпараттың мазмұнын жасыру мақсатында ақпаратты түрлендіру әдістерін іздестірумен және зерттеумен айналысатын болса, ал криптоталдау шифрмәтінді кілтсіз ашумен айналысатын бағыт.

Криптографиялық әдістерді пайдаланудың негізгі бағыттары - байланыс арналары арқылы құпия ақпаратты жіберу (мысалы, электрондық жәшік), жөнелтілген хабарламалардың түпнұсқалығын анықтау, ақпаратты (құжаттар, мәліметтер қоры) тасымалдағыштарда шифрланған түрде сақтау [6,7].

Сонымен, криптография ақпаратты оқу (қалпына келтіру) тек кілтті білумен ғана мүмкін болатындай етіп ақпаратты түрлендіруге мүмкіндік береді.

Криптографиялық жүйелере ашық мәтіннің E түрлендірулер тобы болып сипатталады. Бұл топтың қатысушы элементі k символымен индекстеледі немесе белгіленеді; k параметрі әдетте кілт деп аталады. E_k түрленімі тиісті алгоритммен және k кілт мәнімен анықталады.

Кілт - бұл мәтіндерді кедергісіз шифрлау және шифрын ашу үшін қажетті ақпарат. K кілттер кеңістігі - бұл мүмкін кілт мәндерінің жиыны. Әдетте кілт - бұл алфавит әріптерінің тізбектелген қатары [8,9,10,11].

Криптографиялық беріктілік - бұл кілті білместен, оның шифрын ашуға төзімділігін анықтайтын сипаты (яғни, криптоталдау).

Шифрланатын және шифрлануы ашылатын ақпарат ретінде біз белгілі бір алфавит бойынша жасалған мәтіндерді қарастырамыз. Бұл терминдер төменде көрсетілген.

Алфавит - ақпаратты кодтау үшін пайдаланатын көптеген таңбалар жиынтығы.

Мәтін - алфавит элементтерінің жинақталған жиынтығы.

Заманауи ақпараттық жүйелер (АЖ) қолданылатын алфавит мысалы ретінде төмендегілерді келтіруге болады:

- алфавит Z_{33} –орыс алфавитінің 32 әріпі ("ё" қоспағанда) және бос орын;
- алфавит Z_{256}- ASCII және КОИ-8 стандартты кодтарына кіретін таңбалар;
- екілік алфавит - $Z_2 = \{0,1\}$;
- сегіздік немесе он алтылық алфавит.

Шифрлау - ашық мәтін деп аталатын бастапқы мәтінді шифрленген мәтінге айналдыру процесі.

Шифрды ашу – шифрлауге кері процесс. Кілттің негізінде шифрленген мәтін бастапқыға ауыстырылады.

Қазіргі уақытта ақпараттық технологиялар жүзеге асыру құралдарының қарқынды дамуына байланысты, ақпараттық қауіпсіздік мәселесін қамтамасыз ету құралдары үнемі жаңарту мен дамытуды қажет етеді.

Сонымен қатар қауіпсіздікті қамтамасыз ету мәселелерін ақпараттың үлкен көлемін түзету және берудің жоғары жылдамдығын ұйымдастыру мәселелері сияқты ақпаратты түзету кешендерінің ажырамас бөлігі болып табылады.

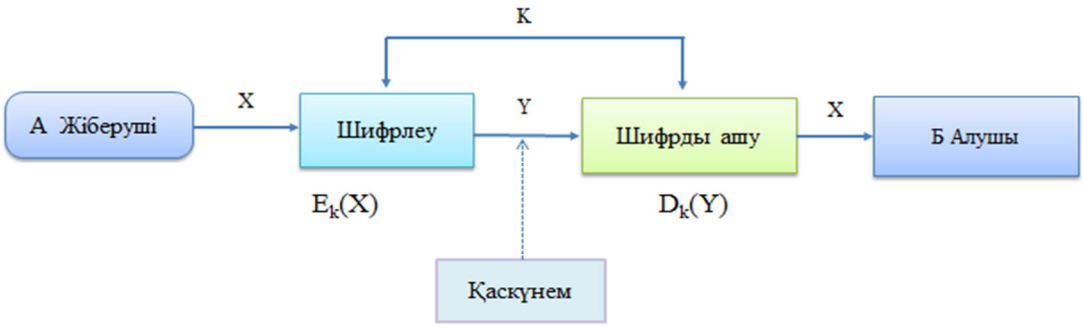
Сондықтан заманауи талаптарға сай келетін ақпараттық қауіпсіздік құралдарын қалыптастыру өзекті мәселелердің бірі [6, 9, 12, 13].

Бұл мәселелерді шешудің ең тиімді әдістерінің бірі криптографиялық әдістері, яғни криптографиялық қорғау жүйелерін қолдану. Ақпаратты криптографиялық қорғау әдістері функционалды міндеттеріне байланысты сан алуан. Заманауи криптография төрт бөлімнен тұрады:

- симметриялы криптожүйелер;
- ашық кілтті криптожүйелер (асимметриялық криптожүйелер);
- электронды сандық қолтаңба;
- кілттерді басқару.

Симметриялық криптожүйелерде шифрлау және шифрды ашу үшін бірдей кілт қолданылады. 1.1-суретте симметриялық криптожүйенің жеңілдетілген құрылымдық сызбасы берілген.

Симметриялы криптожүйені қолданудың алдында пайдаланушылар жалпы k құпия кілтін алып, оған қаскүнемнің қол жеткізуін болдырмауы тиіс. X ашық хабарламасы $E_k(X)$ криптографиялық түрленімге ұшырайды және алынған Y криптограммасы байланыс каналы бойынша алушыға беріледі, онда бастапқы X ашық хабарламасын бөлу мақсатында $D_k(Y)$ кері түрлендіру жүзеге асырылады.



Сурет 1.1 - Симметриялық криптожүйенің жалпы сұлбасы

Ашық кілтті криптожүйелерде бір-бірімен математикалық байланысқан екі кілт қолданылады - ашық және жабық (құпия). Ақпарат барлық пайдаланушыларға қол жетімді ашық кілтпен шифрланады және хабарлама алушыға ғана белгілі жабық кілт арқылы шифр ашылады.

Кілттерді тарату және кілттерді басқару терминдері ақпаратты өңдеу жүйесінің процестеріне жатады, олардың мазмұны пайдаланушылар арасында кілттерді құру және тарату болып табылады [4,6,9,12,14]

Электронды сандық қолтаңба деп мәтінге қосылатын криптографиялық түрлендіруді атайды, бұл мәтінді басқа қолданушы алған кезде хабарламаның авторлығы мен түпнұсқалылығын тексеруге мүмкіндік береді.

Ақпаратты қорғауға арналған шифрлаудің беріктілігін кілттің құпиясын сақтауға және шифрдың криптоберіктілігіне байланысты.

Симметриялық криптожүйелердегі криптографиялық әдістердің барлық түрлерін келесі түрлендірудің төрт класына дейін қысқартуға болады:

- ауыстыру - шифрланған мәтіннің таңбалары алдын ала тағайындалған ережеге сәйкес сол немесе басқа алфавиттің таңбаларына ауысады;
- алмастыру - шифрланғанмәтіннің таңбалары берілетін мәтіннің белгілі блогының бірқатар ережеге сәйкес алмасады;
- аналитикалық қайта құру - шифрланға мәтін кейбір аналитикалық ережеге сәйкес қайта жасалынады, мысалы, гаммалау бастапқы мәтінге кілт қосу арқылы орындалады;
- аралас түрлендіру - бұл шифрланған ақпараттың бөлігіне пайдаланатын негізгі түрлендіру әдістерінің қасиетіне негізделеді.

Симметриялық криптожүйені екі негізгі топқа бөлуге болады: блоктық және ағындық шифрлар. Бірінші топтың криптоалгоритмдері ашық ақпарат блоктарымен криптографиялық түрленімдерді жүзеге асырады. Әдетте блоктың ұзындығы 64 (128) битті құрайды. Деректер ағынын блок бойынша шифрлаудың заманауи жүйелерінде «ауыстырулар» және «алмастырулар» қарапайым операцияларының 16-дан 32-ге дейінгі кезеңдерін қолданады [13,15,16].

Таңбаларды араластыру және шашырату процедураларын қолдану шифрланған мәліметтер блогында шифрдың жоғары төзімділігін және таңбалардың (биттердің) жүйелілігін біркелкі етуге мүмкіндік береді.

Ауыстыру, алмастыру және циклдік жылжытудың қарапайым операцияларын қолдану микропроцессорлық есептеу құрылғыларын қолдана отырып, ақпаратты криптографиялық қорғаудың осы процедураларын іске асыруға мүмкіндік береді. Алайда, бұл шифрлау алгоритмдерінде қазіргі заманғы жоғары жылдамдықты деректерді беру жүйелерінде ақпараттың құпиялығын қамтамасыз етуге мүмкіндік бермейтін бірқатар кемшіліктер бар [9,17,18].

Біріншіден, бұл шифрланған мәліметтердің аз көлемі - 64 (128) бит және итерациялардың көп саны (32 раунд). Екіншіден, блоктық шифрлау жүйелері қатенің блоктың бүкіл ұзындығына таралуы, шифрдың қажетті жылдамдығын қамтамасыз етумен сипатталады. Аталған кемшіліктер қазіргі ақпараттық және телекоммуникациялық жүйелерде блоктық шифрларды кеңінен қолданудың негізгі факторы болып табылады.

Екінші топ бит бойынша шифрлау алгоритмдеріне негізделген. Бит бойынша шифрлау жүйелері шифрлау және шифрды ашу процесінің жоғары тезәрекеттігін қамтамасыз ететіндігі белгілі. Бұл жағдайда деректерді криптографиялық қорғау процедурасы аппараттық және бағдарламалық жасақтамада да ашық ақпараттың түсу жылдамдығына сәйкес жылдамдықпен жүзеге асырылуы мүмкін. Алайда, бұл жүйенің жеткілікті төзімділік деңгейі жоқ. Шифр жалған кездейсоқ биттердің ағынының модуль бойынша түпнұсқалық мәтінмен қосылуына негізделген, жалпы теориялық тұрғыдан танылмайтындығына қарамастан, шифрлау жүйесінің өзі төзімділікпен ерекшеленбейді және түпнұсқа мен шифр мәтінінің кейбір белгілері болғанда бірден ашылуы мүмкін. Бастапқы және таңдалған мәтіндер негізінде жүйенің шабуылдарға осалдығы, мәліметтер ағынын биттік шифрлау кезінде модуль екі таңбасын қосу қайтымды шифрлау функциясын құрудың жалғыз әдісі болып табылады [11,13,17].

Іс жүзінде блоктық шифрлары белгілі бір класстың «таза» түрлендірулеріне қарағанда, олардың криптографиялық беріктілігінің жоғарылығына байланысты жиі кездеседі. Шифрлаудың американдық және ресейлік стандарттары осындай блоктық шифрлар класына негізделіп жасалынған [11,13,19].

1.2 Симметриялы шифрлау алгоритмдерін құрудың жалпы принциптері

Симметриялы блоктық шифрлау алгоритмдерін құру әдістері бірнеше категорияға бөлінеді. Олар:

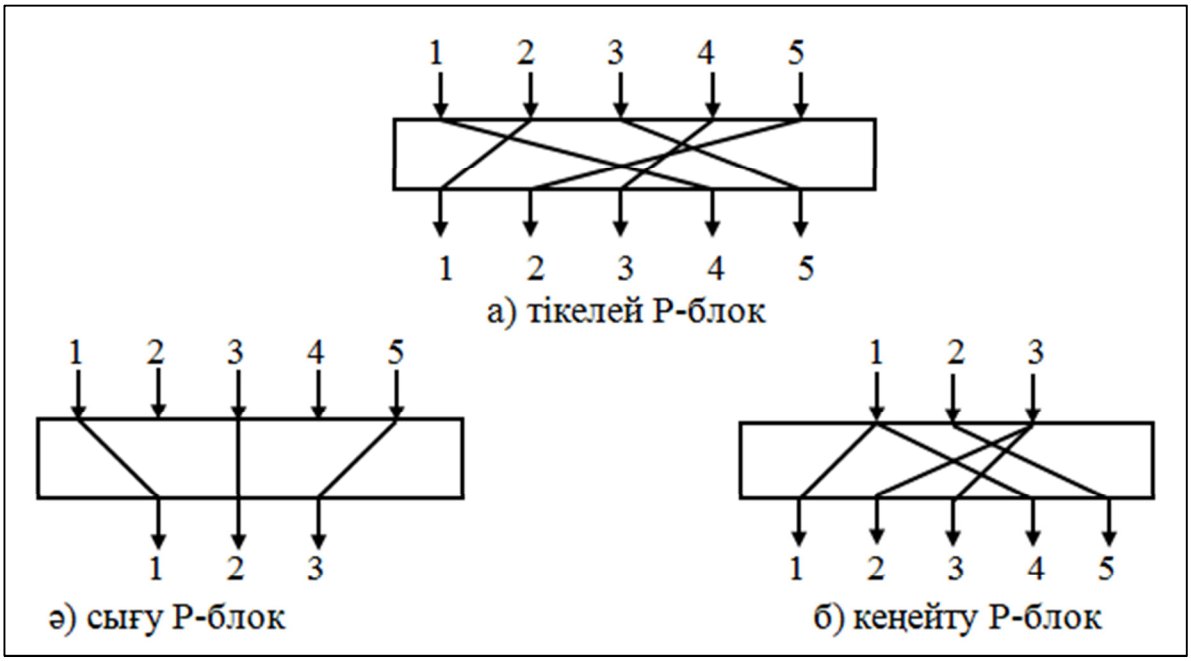
1. Алмастыру-орын ауыстыру желісі (SP-Substitution-permutation network).
2. Фейстель желісі.
3. Тек қана қайтарымды операторларға негізделген квадрат құрылымды шифрлар.

Енді осы көрсетілген компоненттердің жұмыс істеу қызметін жеке-жеке қарастырайық.

1. Алмастыру-орын ауыстыру желісі (SP-Substitution-permutation network).

Қазіргі заманғы блоктық шифрлар әдетте кілттерді алмастыратын шифрлар болып табылады, онда кілт мүмкін болатын кірістерге мүмкін болатын кірістердің ішінара бейнелерін ғана береді. Алайда, бұл шифрлар әдетте бір модуль ретінде жасалынбайды. Ақпаратты шашырату және араластыру сияқты заманауи блоктық шифрдың қажетті қасиеттерін қамтамасыз ету үшін бұл шифр транспозиция модульдері (P-блоктары деп аталады) және ауыстыру модулдері (S-блоктары деп аталады) ретінде қалыптасады [20,21].

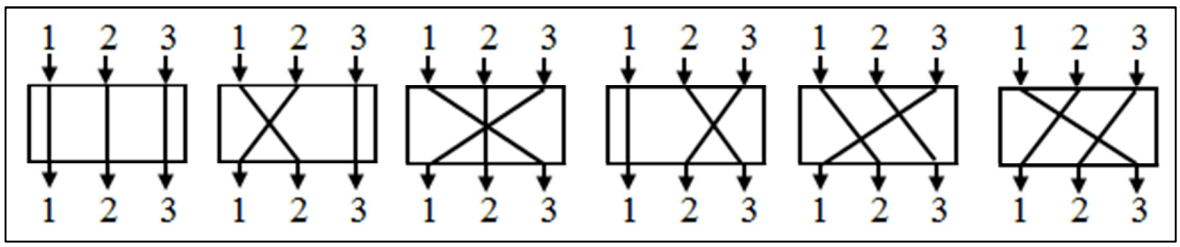
P-блок (алмастыру блогы). 1.2-суретте 5×5 тікелей P-блок, 5×3 сығу P-блок, 3×5 кеңейу P-блок көрсетілген. Әр бір блокты жеке жеке қарастырсақ.



Сурет 1.2 – P-блоктың үш типі

Тікелей P-блок. Тікелей P-блокта n-кірісте мүмкін болатын n! алмастыру бейнесі орындалады да, n-шығыс алынады.

Мысалы: 1.3-суретте 3×3 P-блоктың мүмкін болатын барлық 6 бейнесі көрсетілген.



Сурет 1.3 – 3×3 P-блоктың мүмкін болатын бейнесі

Р-блок n! бейненің бірін анықтау үшін кілт қолдануы мүмкін, алайда Р-блок кілт қолданбайды, яғни бейнелеу алдын ала тағайындалған. Егер Р-блок алдын ала тағайындалса және аппараттық құралға енгізілсе немесе бағдарламалық жасақтама жасалса, ауыстыру кестелері бейнелеу ережесін береді. Екінші жағдайда, кестедегі кірістер шығыс позициялары көрсетілген орындарда көрсетіледі. 1.1-кестеде n=64 болған кезде ауыстыру кестесінің мысалы берілген [13,21,22,23].

Кесте 1.1- Тікелей Р-блок ауыстыру кестесінің мысалы

58	50	42	34	26	18	10	02	60	52	44	36	28	20	12	04
62	54	46	38	30	22	14	06	64	56	48	40	32	24	16	08
57	49	41	33	25	17	09	01	59	51	43	35	27	19	11	03
61	53	45	37	29	21	13	05	63	55	47	39	31	23	15	07

Тікелей Р-блок қайтымды. Бұл дегеніміз тікелей Р-блокты шифрлауда да және шифрды ашудада қолдана аламыз дегенді білдіреді.

Сығу Р-блок. Сығу Р-блокта n кіріс және m шығысы орындалады, мұндағы n>m.Қазіргі заманғы блоктық шифрларда қолданылатын сығу Р-блогы, әдетте биттерді ауыстыру ережелерін көрсететін ауыстыру кестесімен кілтсіз болады. Біз сығу Р-блогының ауыстыру кестесінде m кестелік кіріс бар екенін ескеруіміз керек, бірақ әрбір кесте кірісінің мазмұны 1-ден n-ге дейін болады, ал олардың кейбір кірістері бұғатталған болуы мүмкін. Мысалы 1.2-кестеде көрсетілгендей 32×24 сығу Р-блокта кейбір 7,8,9,16,23,24 және 25 кірістері бұғатталған.

Кесте 1.2 - 32×24 ауыстыру кестесінің мысалы

01	02	03	21	22	26	27	28	29	13	14	17
18	19	20	04	05	85	10	11	12	30	31	32

Сығу Р-блоктары биттерді ауыстыру қажет болған кезде және келесі кезеңдер үшін биттер санын азайту кезінде қолданылады.

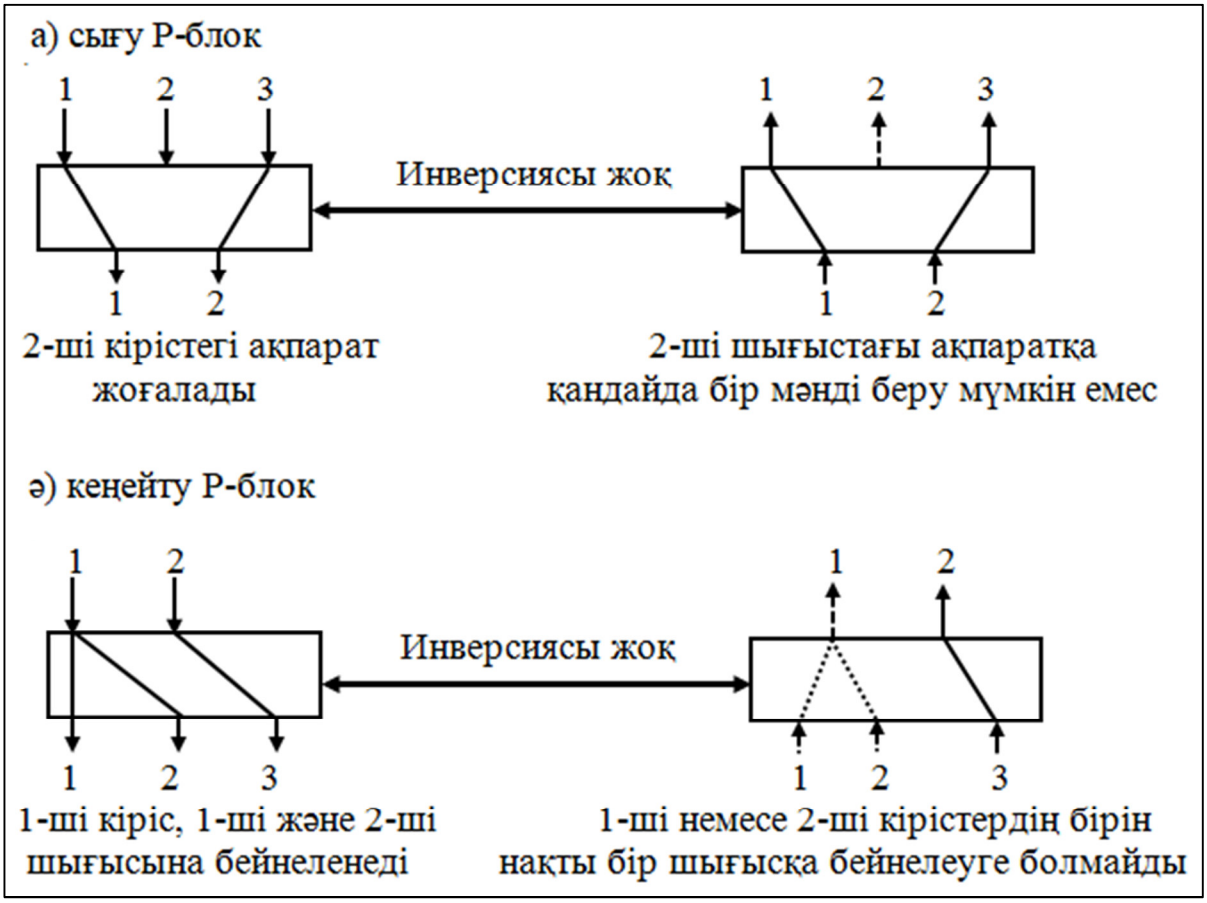
Кеңейту Р-блок - n кірісі және m шығысы бар Р-блогы, мұндағы m> n. Кейбір кірістер бірнеше шығысымен байланысты (1.2- сурет). Қазіргі блоктық шифрларда қолданылатын Кеңейту Р-блокты шифрлар, әдетте кілтсіз орындалады. Битті ауыстыру ережелері кестеде көрсетіледі. Кеңейту Р-блогының орын ауыстыру кестесінде m кестелік кіріс бар, бірақ m - n кіріс (бірнеше ақпараттық шығыстарына байланысты кірістер). 1.3-кестеде 12×16 Кеңейту Р-блоктың ауыстыру кестесі көрсетілген. Онда 1, 3, 9 және 12-нің әрқайсысы екі шығысқа қосылатынын байқаймыз [21,24,25].

Кесте 1.3 - 12×16 ауыстыру кестесінің мысалы

01	09	10	11	12	01	02	03	04	05	06	07	08	09	12

Кеңейту Р-блогы биттерді ауыстыру керек болғанда және келесі шифрлау сатысында биттер санын көбейту кезінде қолданылады.

Сығу және Кеңейту Р-блоктар қайтымсыз. Себебі сығу Р-блокты шифрлау процесіне қолданған кезде кейбір кіріс биттері жоғалады, ал шифрды ашу кезінде жоғалған кіріс биттерін қалпына келтіретін кілт жоқ. Ал Кеңейту Р-блогты шифрлау процесіне қолданғанда кіріс биті бірнеше шығыс битіне бейнелеуге болады, ал шифрды ашу кілті жоқ, осылайша берілген кірісті бірнеше кірістің қайсысының бейнеленуі көрсететіні анықталмайды. 1.4-суретте екі жағдай көрсетілген.



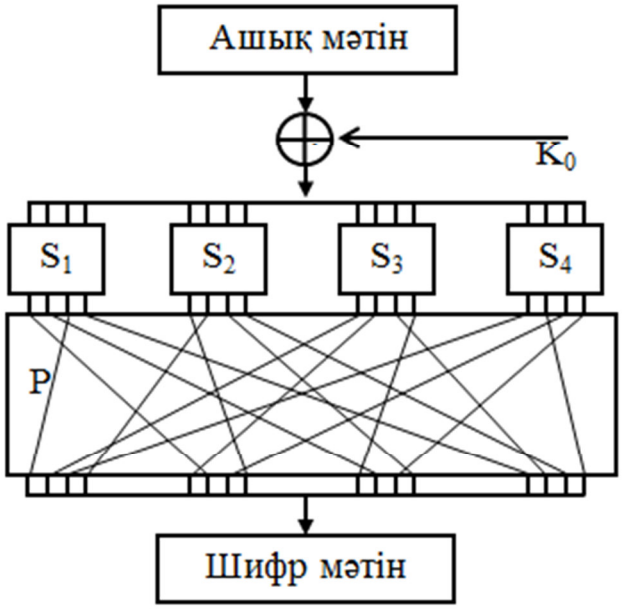
Сурет 1.4 –Сығу және Кеңейту Р-блоктар қайтымсыз компоненттер

1.4-суретте сығу Р-блок Кеңейту Р-блоктың кері шифры және керісінше емес екендігі көрсетілген. Бұл деп отырғанымыз егер біз сығу Р-блокты шифрлауда қолданатын болсақ, онда біз Кеңейту Р-блокты шифрды ашуға немесе керісінше қолдана алмаймыз. Сондықтан сығу немесе Кеңейту Р-блокты пайдаланған шифрлардың нәтижелері басқа түрлендірулерге қарағанда нашар екендігі көрсетілген [21,26,27].

S-блок (ауыстыру блогы). Миниатюралық ауыстыру шифры ретінде қарастыруға болады. Бұл блоктың кіріс және шығыс саны әр түрлі болуы мүмкін. Басқа сөзбен айтқанда, S-блокта n-биттік кіріс және m биттік шығыс болуы мүмкін, мұндағы m және n сандары бірдей болуы міндетті. S-блок кілтсіз

немесе кілтпен бола алатынына қарамастан, қазіргі заманғы блоктық шифрлар көбінесе кілтсіз S-блоктарды қолданады, мұнда ақпараттық кірістерден ақпараттың шығуына дейін алдын ала анықталған.

S (substitution) - ауыстыру блогы және P (permutation) алмастыру блок түрлендірулері SP-желілері атына ие болды, яғни алмастыру және ауыстыру желілері. SP-желі - бұл 1971 жылы Хорст Фейстель ұсынған блоктық шифрдың бір түрі. SP-желілерінің криптографиялық идеясы оңай жүзеге асырылатын, салыстырмалы қарапайым бірнеше түрлендіру үйлесімімен күрделі криптотүрлендіру құруға негізделген. Бірінші криптографиялық алгоритм SP-желісі негізінде «LUCIFER» болды (1971ж). SP-желілері бір раундта толығымен шифрланатын блокты өңдейді. SP-желілерінің 1-раундтағы сұлбасы 1.5-суретте көрсетілген. SP-желілерінің мысалдары ретінде SERPENT немесе SAFER+ алгоритмдерін келтіруге болады [13,15,21,28].



Сурет 1.5 - SP-желілерінің 1-раундтағы сұлбасы

2. Фейстель желісі. Фейстель желісі деп аталатын, көп рет қайталанатын құрылымды қолданатын Фейстель желісі. Қайтымсыз операторларды пайдалануға рұқсат етіледі. Бір ұяшықтан екінші ұяшыққа ауысқан кезде раундтық кілт өзгереді. Шифрлау/дешифрлауды ашу операциялары кейбір жөнге келтірумен сәйкес келіп, кілттерді кері тәртіпте қолдануды ғана талап етеді. Осы құрылымның көмегімен шифрлау бағдарламалық деңгейде, сондай-ақ аппаратты деңгейде оңай іске асырылады, бұл қолданудың кең мүмкіндіктерін ұсынады. (мысалы DES, МЕМСТ 28147-89, RC5, BLOWFISH, TEA, CAST-128 және с.с.). SP желісімен салыстырғанда, Фейстель -желілері бір раундта жартылай шифрланатын блокты өңдейді [13,17,21,28,29].

Фейстель желісі ерікті функцияны (әдетте F-функциясы деп аталатын) көптеген блоктарға ауыстырудың жалпы түрлендіру әдісі болып табылады. Бұл құрастырылымды Хорст Фейстель ойлап тапты және бұл құрастырылымдар

DES және МЕМСТ 28147–89 алгоритмдерінде қолданылды. Фейстель желісінің негізгі құрылыс блогы болып табылатын F-функциясы, әрдайым сызықты емес және барлық дерлік жағдайда қайтарымсыз болып таңдалады.

Формалды түрде F-функцияны келесідей көрсетуге болады:

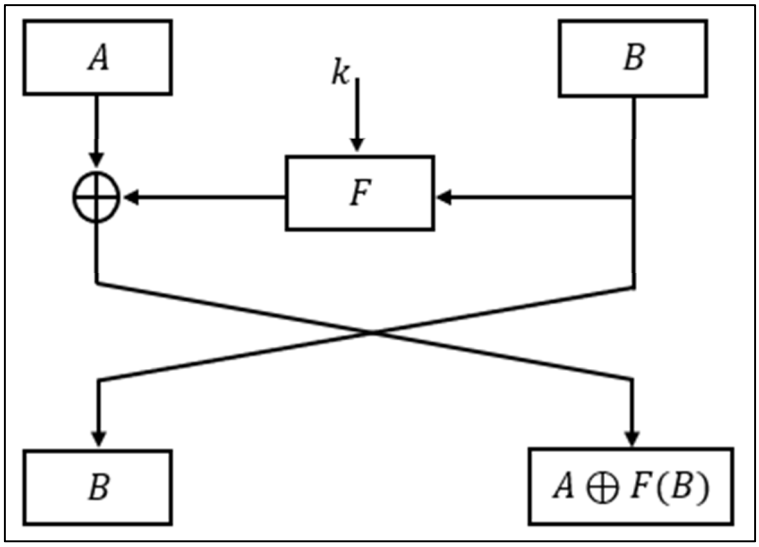
$$F: Z_{2,N/2} Z_{2,k} \rightarrow Z_{2,N/2},$$

мұнда N – түрленетін мәтін блогының ұзындығы (жұп болуы тиіс); k – негізгі ақпаратта қолданылатын кілттің ұзындығы.

X – мәтін блогы болса, оны бірдей ұзындықтағы екі қосалқы блок ретінде көрсетеміз $X = \{A, B\}$. Онда Фейстель желісінің итерациясы (немесе раунд) төмендегідей анықталады

$$X_{i+1} = B_1 || (F(B_1, k_i) \oplus A_i),$$

мұндағы $X_i = \{A_i, B_i\}$, $||$ тіркесім операциясы, ал $\oplus$–биттік қосу НЕМЕСЕ.



Сурет 1.6 - Фейстель желісі итерациясының құрылымы

Фейстель желісі итерациясының құрылымы 1.6-суретте берілген. Фейстель желісі жасалатын шифрдың төзімділігі ескерілуімен анықталатын белгілі бір итерация санынан тұрады, ал соңғы итерация кезінде мәтіндік блоктың жартысын ауыстыру орындалмайды, өйткені бұл шифрдың беріктігіне әсер етпейді.

Бұл шифрлар құрылымының бірқатар артықшылықтары бар, атап айтқанда:

- шифрлау және шифрлауды ашу процедуралары сәйкес келеді, шифрлауды ашқан кезінде негізгі ақпарат кері тәртіпте пайдаланылатын жағдайларды қоспағанда;

- шифрлау құрылғыларын құру үшін шифрлау және шифрлау ашу тізбектерінде бір блоктарды пайдалануға болады.

Кемшілігі, әр итерация кезінде өңделген мәтін блогының жартысы ғана өзгереді, бұл қажетті жылдамдыққа жету үшін итерация санын көбейту қажеттілігіне әкеледі.

F-функциясын таңдауға қатысты нақты стандарттар жоқ, бірақ бұл функция сызықтық емес алмастырулар, ауыструлар мен жылжытулардың кілттеріне тәуелді реттілік болып табылады.

Блоктық шифрларды құрудың тағы бір тәсілі - кілттерге тәуелді қайтарымды түрленулерді пайдалану болып табылады. Бұл жағдайда барлық блок әрбір итерация жағдайында өзгереді, және сәйкесінше, итерациялардың жалпы санын азайтуға болады. Әр итерация - бұл түрлендірулер тізбегі болып табылады («қабаттар» деп аталатын), олардың әрқайсысы өз функцияларын орындайды. Әдетте, сызықты емес қайтымды ауыстыру қабаты, сызықтық араластыру қабаты және *K* кілтін араластырудың бір немесе екі қабаты қолданылады, бұл тәсілдің кемшіліктеріне мыналарды жатқызуға болады, жалпы жағдайда бірдей блоктарды шифрлау және шифрлауды ашу процедураларында қолдануға болмайды, бұл іске асырудың аппараттық немесе бағдарламалық шығындарын арттырады [13,16,21,27].

3. Тек қана қайтарымды операторларға негізделген квадрат құрылымды шифрлар. Мысалы, AES шифры. «Квадрат» құрылымы екі өлшемді байт массиві түрінде шифрленген мәліметтер блогын ұсынумен сипатталады. Криптографиялық түрлендірулер массивтің жеке байттарында, сонымен қатар оның жолдарында немесе бағандарында орындалуы мүмкін. Алгоритм құрылымы өз атауын Square алгоритмінен алды, ол 1996 жылы Винсент Риджменмен (Vincent Rijmen) және Джоан Дейменмен (Joan Daemen) –Rijndael алгоритмінің болашақ авторларымен жасалды, сонымен қатар ол Square-ұқсас құрылымға ие және ол ашық конкурста AES кейін АҚШ-тың жаңа шифрлау стандарты болды [13,29].

Шифрлау алгоритмдерін құру процессінде алгоритмнің криптоберіктікті қамтамасыз ету негізгі мәселе болып табылады. Бұл мәселе сонымен қатар ең күрделісі болып саналады. Криптоберіктікті есептеу ең ұзақ, көп еңбекті қажет ететін және аралас сатылардың бірі болып табылады. Жоғары беріктікке жетудің өзі мақсат емес екенін есте ұстаған жөн. Белгілі бір тәжірибесі бар криптограф үшін аппараттық және бағдарламалық іске асыру үшін анықталмайтын шифрді тез арада жасау қиынға соқпайды. Нақты қолдану үшін қажетті шифр параметрлерін (өнімділік, іске асырудың күрделілігі және т.б.) қамтамасыз ету маңызды. Пайдалану талаптары қолайлы шифрді жасау үлкен талғам мен еңбекті қажет ететіндігін білдіреді. Пайдаланудың нақты шарттары шифрдың құрылысының жалпы сызбасын және бағалау кезеңін таңдауға айтарлықтай әсер етуі мүмкін. Блок шифрларын синтездеу процесі келесі кезеңдермен және оларға сәйкес мәселелермен байланысты:

1. Қолдану аясын зерттеу. Бұл кезеңде криптожүйенің түрін таңдау және оның негізгі параметрлеріне қойылатын талаптарды тұжырымдау жүзеге асырылады. Блоктық шифрлар қазіргі кезеңде кеңінен қолданылуға ие болды.

Олар жеке блоктарды тәуелсіз шифрлау режимінде жоғары беріктікті қамтамасыз етеді, бұл шифрленген мәліметтер массивіне кездейсоқ қол жеткізуге мүмкіндік береді. Ақпараттарды рұқсатсыз кіруден қорғаудың бірқатар құралдары блоктық және ағындық шифрлау белгілері бар біріктірілген типті шифрды қолдана отырып, файлдарды желілік режимде жоғары жылдамдықта шифрлауды қолданады. Шифрлаудың блокты нұсқасын таңдағанда, пайдаланылған блоктың ұзындығын негіздеу қажет. Қолдану салаларын ескере отырып, іске асыру нұсқасының мәні (бағдарламалық, аппараттық, әмбебап), аппараттық қамтамасыз етудің күрделілігі және шифрлау жылдамдығы анықталады.

2. Құпия кілттің ұзындығын таңдау. Ақырғы кілті бар кез-келген криптожүйе үшін әрдайым мүмкін кілттерді мүмкін болатын нұсқаларының бәрін теру әдісі арқылы кілтті табуға болатындығын есте ұстаған жөн. Қазіргі уақытта әмбебап жағдайда кем дегенде 128 биттен кем емес кілтінің ұзындығы ұсынылады, бірақ кейбір жағдайларда ақпарат маңызды емес болса, 64 немесе тіпті 56 битті кілт ұзындығын пайдалануға болады.

3. Кілтті пайдалану кестесін таңдау.

4. Негізгі криптографиялық примитивтерді таңдау және сызбаны жасау. Осы тармақты орындау шифрларды құрудың негізгі тәсілдері, блоктық криптожүйелердің түрлері туралы, сонымен қатар нақты шифрлар мен қолданылатын операциялардың қасиеттері, оларды аппараттық іске асырылуының құны мен енгізілген кідіріс уақыты туралы білімді қажет етеді.

5. Шифрлау алгоритмін іске асыру үшін тұтынылған ресурстарды бағалау. Бағдарламалық және аппараттық іске асыру нұсқалары қарастырылған тәжірибелік шифрлегіштерді және олардың бағдарламалық үлгілерін іске асыру жүзеге асырылуда.

6. Шифрдың өнімділігін бағалау. Бағдарламалық және аппараттық құралдарды іске асырудың әртүрлі нұсқалары үшін шифрлау жылдамдығының мәні анықталады. Егер 5 және 6 қадамдарда алынған нәтижелер 1 қадамда анықталған нәтижелерді қанағаттандырмаса, онда 4 қадам ағымдағы қадамда алынған нәтижелерді ескере отырып қайталанады.

7. Крипто-аналитикалық шабуылдардың мүмкін түрлерін криптоберіктіктін қарастыру. Криптоталдаудың негізгі варианттары, сонымен қатар қолдану мүмкіндіктерін (инженерлік талдау) пайдалана отырып, криптошабуылдың басқа мүмкін нұсқалары қарастырылады. Күрделілігі шифрдың криптоберіктігінің нәтижесін анықтайтын ең тиімді шабуылды анықтау.

8. Алгоритмді алдын-ала талдауды ескере отырып түрлендіру. 7-ші қадамда алынған нәтижелерді ескере отырып, ең тиімді шабуылдың күрделілігін арттыру үшін крипто-сызбаның негізгі түйіндерін оңтайландыру жүзеге асырылады. Содан кейін модификацияланған алгоритмнің беріктігін алдын-ала бағалау жүргізіледі. Қажет болса, алдын-ала беріктік мәні алынғанша осы қадамды бірнеше рет қайталанады.

9. Түрлендірілген шифрге толық талдау жасау. Егер егжей-тегжейлі талдау өзгертілген сызбаның маңызды кемшіліктерін анықтаса, 8-кезеңге немесе тіпті 4-ші қадамға қайта оралу қажет.

10. Статистикалық тесттер мен арнайы эксперименттерді орындау. Бұл қадамда шифрлау алгоритмі бағдарламаланады және теориялық талдаудың толықтығы мен дұрыстығын тексеру үшін талдау нәтижелерін ескере отырып арнайы эксперименттер жүзеге асырылады [13,19,28,30].

Сонымен қатар, деректерді криптографиялық түрлендірудің заманауи жоғары жылдамдықты әдістері деректердің құпиялылығын, тұтастығы мен шынайылығын қамтамасыз етудің ең тиімді құралы болып табылады.

1.3 Заманауи шифрлау алгоритміне қойылатын талаптар

Ақпараттарды криптографиялық қорғау процесі бағдарламалық, сондай-ақ аппараттық түрде жүзеге асырылуы мүмкін. Аппараттық түрде іске асыру едәуір қымбаттығымен айқындалса, алайда оның артықшылықтары да бар: жоғары өнімділік, қарапайымдылық, қауіпсіздік және т.б.

Бағдарламалық іске асыруды пайдалану қолайлы болып табылады, себебі қолдануда белгілі бір икемділікке мүмкіндік береді.

Ақпаратты криптографиялық қорғаудың заманауи криптографиялық жүйелері үшін жалпы келесі қабылданған талаптар тұжырымдалады [5,8,11,13,23]:

- шифрланған мәтін тек кілт болса ғана оқылуы тиіс;
- шифрланған мәтіннің фрагменті және оған сәйкес ашық мәтін бойынша пайдаланылған шифрлау кілтін білу үшін қажетті әрекеттер саны кем дегенде мүмкін болатын кілттердің жалпы санынан тұруы керек;
- барлық мүмкін кілттерді іріктеп алу арқылы ақпараттың шифрын ашу үшін қажет болатын операциялардың саны қатаң ең төменгі шектеулі бағаға ие болуы тиіс және қазіргі компьютерлердің мүмкіндіктерінен асып кетуі керек (желілік есептеулерді пайдалану мүмкіндігін ескере отырып) немесе осы есептеулерге жол берілмейтін үлкен шығындарды талап етеді;
- шифрлау алгоритмін білу қорғаныс сенімділігіне әсер етпеуі керек;
- кілттің шамалы өзгеруі сол бастапқы мәтінді шифрлау кезінде де шифрленген хабарлама түрінің айтарлықтай өзгеруіне әкелуі керек;
- бастапқы мәтіннің болмашы өзгеруі сол кілтті қолданған кезде де шифрленген хабарлама түрінің айтарлықтай өзгеруіне әкелуі керек;
- шифрлау алгоритмінің құрылымдық элементтері өзгеріссіз болуы керек;
- шифрлау процесінде хабарламаға енгізілген қосымша биттер толық және сенімді түрде шифр мәтінінде жасырынуы керек;
- шифр мәтінінің ұзындығы бастапқы мәтінінің ұзындығынан аспауы керек;
- шифрлау процесінде бірізді түрде қолданылатын кілттер арасында қарапайым және оңай орнатылатын тәуелділік болмауы керек;
- көптеген мүмкін кілттердің кез келген кілті ақпараттың сенімді қорғалуын қамтамасыз етуі керек;

- алгоритм бағдарламалық, сондай-ақ аппараттық іске асыруға мүмкіндік беруі тиіс, бұл ретте кілт ұзындығын өзгерту шифрлау алгоритмінің сапалылығына әсер етпеуі керек.

1.4 Бөлім бойынша қорытынды

Диссертация тақырыптың өзектілігіне сәйкес рұқсатсыз қол жетімділіктен құпия ақпаратты қорғау мәселесін шешудің ең тиімді әдістерінің бір криптографиялық әдістер екендігін ескерсек. Онда бірінші бөлімде қарастырылған мәліметтер заманауи ақпаратты қорғауға арналған симметриялық блокты шифрлау алгоритмін құруға негізделген ақпараттар болып табылады. Себебі бұл бөлімде ақпараттарды криптографиялық қорғаудың негізгі бөлімдері және қолдану әдістері келтірілген.

Криптографиялық қорғау әдістерінің бірі симметриялық блокты шифрлау алгоритмдерінде қолданылатын негізгі түрлендіру әдістері және алгоритмге қойылатын негізгі талаптар туралы нақты мәліметтер берілген.

2 ПОЗИЦИЯЛЫҚ ЕМЕС ПОЛИНОМДЫ САНАУ ЖҮЙЕСІ НЕГІЗІНДЕ ЖАСАЛЫНҒАН СИММЕТРИЯЛЫҚ БЛОКТЫ ШИФРЛАУ АЛГОРИТМІ

2.1 Криптографиялық жүйелерде позициялық емес полиномды санау жүйесін қолдану

Зерттеу барысында санау жүйесі, оның ішінде қалдықтар класынындағы санау жүйесі (ҚКСЖ) бойынша көптеген ғалымдардың еңбектеріне шолу жасалды. 20-шы ғасырдың 60-жылдары қалдықтар класының санау жүйесі деп аталатын жүйе кең танымал бола бастады. Чех ғалымдары М. Валаха және А. Свободаның 1955-1957 жылдары жарық көрген мақалалары осы жүйені зерттеуге және одан әрі дамытуға негіз болды. Олардың жұмыстарында қалдықтар жүйелерін компьютерлік технологияда қолдану алғаш қарастырылған. Бұл идеяны кеңес заманының ғалымдары И.Я. Акушский және Д.И. Юдицкий қолдап, қалдықтар жүйесі негізінде есептеу құрылымдарының математикалық теориясын дамытып қана қоймай, осы жүйені қолдана отырып отандық алғашқы ЭЕМ құру бойынша ғылыми жұмыстарды басқарды [31,32].

Сонымен қатар қалдықтар класының санау жүйесі туралы А.И.Галушкиннің «Нейрокомпьютеры в остаточных классах» кітабында қарастырған. Бұл жұмыста қалдықтар класының санау жүйесінде жұмыс істейтін нейрокомпьютерлер құрылымдарының жаңа даму бағыты сипатталған. Қазіргі таңда қалдықтар класының санау жүйесі басқа да көптеген салаларда қолданылуда [32,33].

Модулді арифметиканың даму бағыттарының бірі болып, позициялы емес полиномды санау жүйесін құру, талдау және қолдану бойынша жүргізілген Р.Г.Бияшевтың жұмыстары табылады. Ол полиномдар алгебрасы қандай да бір келтірілмейтін көпмүшелік модулі бойынша өріс болатындығын көрсеткен, қалдықтар жайлы қытай теоремасын көпмүшеліктер үшін дәлелдеген, және де полиномды жүйеде арифметикалық амалдарды жүргізу мен көпмүшелілердің қалдықтары арқылы қалпына келтіру ережелерін анықтаған [34,35,36].

Классикалық ҚКСЖ жүйе негізі ретінде өзара жай сандар алынады, онда кез-келген сан жүйенің негіздеріне бөлгеннен қалған қалдықтары арқылы көрсетіледі. Ал позициялық емес полиномьды санау жүйесі (ПЕПСЖ) ретінде келтірілмейтін көпмүшелік алынады. Позициялық емес полиномьды санау жүйесін (ПЕПСЖ) құруды сипаттайық [36,37,38,39].

Бізге $p_1(x), p_2(x), \dots, p_n(x)$ жұмыс негіздері деп аталатын келтірілмейтін көпмүшелік берілсе, онда қандайда бір $F(x)$ көпмүшелікті таңдалған жұмыс негіздер бойынша қалдықтар тізбегі түрінде өрнектеуге болады (2.1):

$$A(x) = (a_1(x), a_2(x), \dots, a_n(x)) \tag{2.1}$$

бұл жерде $a_i(x)$, $i = \overline{1,n}$ келесі формула бойынша анықталады

$$a_i(x) = A(x) - \left[\frac{A(x)}{p_i(x)}\right] p_i(x), i = \overline{1,n} \tag{2.2}$$

Мұнда $A(x)$ көпмүшелігінің i-ші разрядты $a_i(x)$, $A(x)$ көпмүшелігін $p_i(x)$-ға бөлгенде қалдығы және $a_i(x) < p_i(x)$. Егер $p_i(x)$ жұмыс негіздерінің дәрежесі m-ға тең болса, онда оған сәйкес қалдықтың ең жоғарғы дәрежесі $m_i - 1$-ге тең. Демек, қалдықты бейнелеу үшін қажет болатын екілік разрядттың саны $a_i(x)$, $i = \overline{1,n}$ (2.2), оның жұмыс негіздерінің дәрежесіне тең, ондай болса қалдықтардың саны жоғарыда айтылған екілік разраядтарды қолданып бейнеленетін жұмыс негіздеріндегі көпмүшеліктердің санына 2^{m_i} – ға тең болады. Осылайша, полиномдық санау жүйесінде «ақпараттық» артықтық жоқ, және бұл әр базада барлық мүмкін комбинацияларды қолдануға мүмкіндік береді [36,37,40].

ПЕПСЖ-нің диапазон көлемінің мәні мынаған тең:

$$P(x) = \sum_{i=1}^{n} p_i(x),\ i = \overline{1,n} \qquad (2.3)$$

Қалдықтар арқылы берілген (2.1) формуладағы позициялық емес санау жүйесін $A(x)$ позициялық санау жүйесіне ауыстыру. Ақпаратты сақтау, жіберу және өңдеу жағыдайында ол келесі формула бойынша жүзеге асырылады:

$$A(x) = \sum_{i=1}^{n} a_i(x)B_i(x),\ B_i(x) = \frac{\prod_i^n p_i(x)}{p_i(x)} M_i(x) \equiv 1\big(mod p_i(x)\big); \qquad (2.4)$$

Мұндағы $i = \overline{1,n}$ және $M_i(x)$ көпмүшелігінің мәні (2.3) формулада көрсетілген салыстыруды орындау арқылы есептелінеді. Сақталған және берілетін ақпарат үшін қалпына келтіру келесі формула бойынша орындалады:

$$A(x) = \sum_{i=1}^{n} a_i(x)P_i(x),\ P_i(x) = \frac{P(x)}{p_i(x)},\ i = \overline{1,n} \qquad (2.5)$$

2.2 EMCipher шифрлау алгоритмін құру

Кез-келген мемлекеттің даму стратегиясында басым бағытарының бірі ұлттық қауіпсіздік екендігін ескерсек онда оның ең маңызды элементтерінің бірі ақпараттық қауіпсіздік болып табылады. Сондықтан ақпараттық жүйелерді қауіпсіз және тиімді өңдеуді дамыту әр бір мемлекеттің басым бағыттарының бірі. Ақпараттық қауіпсіздіктің жаңа технологияларын құру мәселесін шешу, бір жағынан өңдеудің жоғары жылдамдығын және ақпараттың үлкен көлемін беруді, ал екінші жағынан оған қолжетімділікті шектеп, ақпаратты қорғаудың қажетті деңгейін қамтамасыз ету қажет.

Сондықтан қазіргі заман талабына сай ақпараттық қауіпсіздік құралдарын құру өзекті мәселелердің бірі болып табылады.

Осы мәселелерді шешудің тиімді әдістерінің бірі - криптографиялық әдістерді қолдану.

Ақпаратты криптографиялық қорғау әдістері олардың функционалдық міндеттеріне байланысты өзгеруі мүмкін. Сонымен қатар ақпаратты

криптографиялық қорғау ақпаратқа рұқсатсыз кіруден қорғаудың негізгі құралы болып табылады [8,36,37,40].

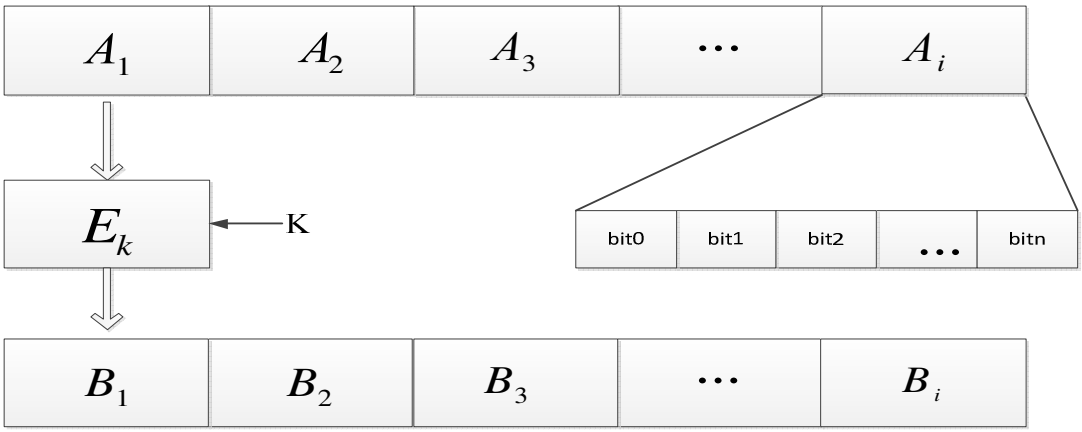
Бүгінгі таңда әлемде цифрлық технологиялардың белсенді өсуіне байланысты ақпаратты қорғаудың криптографиялық әдістерімен байланысты ғылым салалары қарқынды дамуда. Осыған орай шетелдік және отандық ғалымдарда өздерінің еңбектерін жыл сайын жақсартып тиімді шешімдер қабылдап нәтижелерімен бөлісуде. Мысалы [41,42,43] жұмыстарда осы сұрақтар егжей-тегжейлі сипатталған, авторлар ақпаратты криптографиялық қорғау бойынша ғылыми мақалаларды, оның ішінде бейнелер түріндегі ақпаратты қорғауға арналған шифрлау алгоритмдерін шолу бойынша орасан зор жұмыс жүргізді. Сонымен қатар авторлар шифрлаудің кейбір алгоритмдеріне және оларға жүргізілген криптографиялық шабуылдардың нәтижелерін сипатталған [13,25,44,45,46,47].

Криптографиялық қорғау әдістері ақпаратты қорғауда ақпаратқа оған қол жеткізу емес, ақпараттың өзі тікелей қорғағандықтан ең сенімді әдістері болып табылады. Заманауи криптография төрт бөлімнен тұратынын ескерсек. Онда осы диссертациялық жұмысты зерттеу барысында қарастырылған симметриялы криптожүйеге кеңінен тоқталып өтсек. Симметриялық криптожүйелердегі шифрлау тәсілдерінің классификациясы 2.1-суретте көрсетілген. Симметриялы криптожүйеде ақпаратты шифрлау және дешифрлау үшін де бір ғана кілт қолданады

Сурет 2.1 - Симметриялық криптожүйелердегі шифрлау тәсілдерінің классификациясы

Егер блокты шифрлау алгоритмдері ашықмәтінді блоктарға бөліп, әр блокпен жеке шифрлау жұмысын жүргізсе, ал ағынды шифрлау алгоритмі ашықмәтінді бөліктемей әр элементін шифрлеп ағынды күйде жіберіледі. Ал аралас шифрлау алгоритмдерінде блокты және ағынды шифрлау тәсілдерін

бірге қолданады. Блокты шифрлау алгоритмдерінің жалпы сұлбасы 2.2-суретте көрсетілген.



Сурет 2.2 - Блокты шифрлау алгоритмдерінің жалпы сұлбасы

Блокты шифрлар тәжірибеде олардың неғұрлым жоғары криптоберіктілігіне байланысты "таза" түрлендірулерге немесе олардың қандай да бір басқа түрлеріне қарағанда жиі кездеседі. Американдық және Ресей шифрлау стандарттары дәл осы блоктық шифрлардың класына негізделген [13,25,48]. Симметриялық блокты шифрлау алгоритмдері қазіргі уақытта заманауи ақпараттық және телекоммуникациялық жүйелерде ақпаратты өңдеу барысында құпиялылықты қамтамасыз етудің негізгі құралы болып табылады.

Қазіргі шифрлау алгоритмдері Киркхоффс принципіне негізделеген, оған сәйкес шифрдың құпиялылығы шифрлау алгоритмінің құпиялығымен емес, кілттің құпиялығымен қамтамасыз етілуі керек [13,25,49]. Симметриялық блокты шифрлау алгоритмдерінің көпшілігінде келесі операция түрлері қолданылады: алмастыру кестесі, биттердің тобы қандайда бір басқа биттер тобымен алмастырылады (S-box); хабарламаның биттері орынауыстыру арқылы қайта реттеледі; модулі 2 бойынша қосу операциясы (XOR немесе $\oplus$) белгілерімен өрнектеледі; қандайда бір биттердің санына циклдік ығысу [22,23,25,50,51].

Сонымен қатар, заманауи криптографиялық жылдам түрлендіру әдістері жоғары жылдамдықты автоматтандырылған жүйелердің бастапқы өнімділігін сақтауға мүмкіндік береді. Ақпараттарды криптографиялық түрлендіру әдістері құпиялылықты, тұтастықты және тұпнұсқалықты қамтамасыз етудің ең тиімді құралы болып табылады.

Диссертациялық жұмысты зерттеу барысында EM (Exponentiation Module) түрлендіру әдісі және S-блок алмастыру кестесі негізіндегі жаңа блокты шифрлау алгоритмі құрылды. Бұл жаңа позициялық емес полиномьды шифрлау алгоритмі «EMCipher» деп аталды.

Жасалынып отырған шифрлау алгоритмін құру кезінде Галуа $GF(2^v)$ өрісінде дәрежеге шығару операциясына негізделген позициялық емес полиномьды санау жүйесінде жұмыс істейтін EM (exponentiation modul)

түрлендіру әдісі, S-box ауыстыру кестесі қолданылды. Барлық қолданылған әдістер төменде сипатталған. Ұсынылып отырған блокты шифрлау/дешифрлау алгоритмі 2.3 және 2.4-суреттерде көрсетілген.

Алгоритмнің негізгі параметрі:

- блок ұзындығы 128 бит;
- кілт ұзындығы 320 бит;
- раунд саны –16;

Шифрлау процесін 4 кезеңге бөлуге болады:

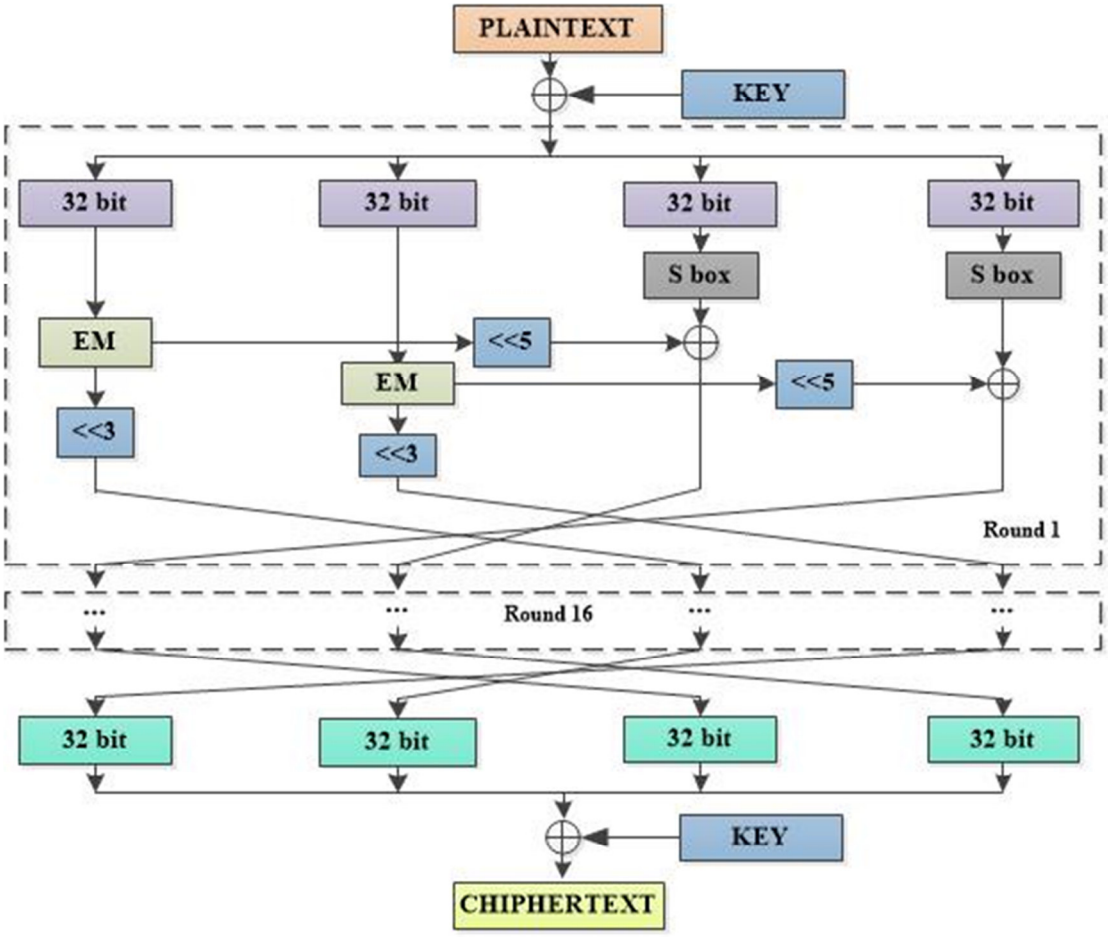
1-кезең. Ашық мәтіннің блогына модуль екі бойынша кілт қосыладыда, шыққан блок 4 ішкі блоктарға бөлінеді және әрбір блок келесі кезеңнен өтеді.

2-кезең. 1-ші және 2-ші ішкі блоктар сұлба бойынша ЕМ түрлендіру әдісінен өтеді және биттік жылжыту операциясы орындалады;

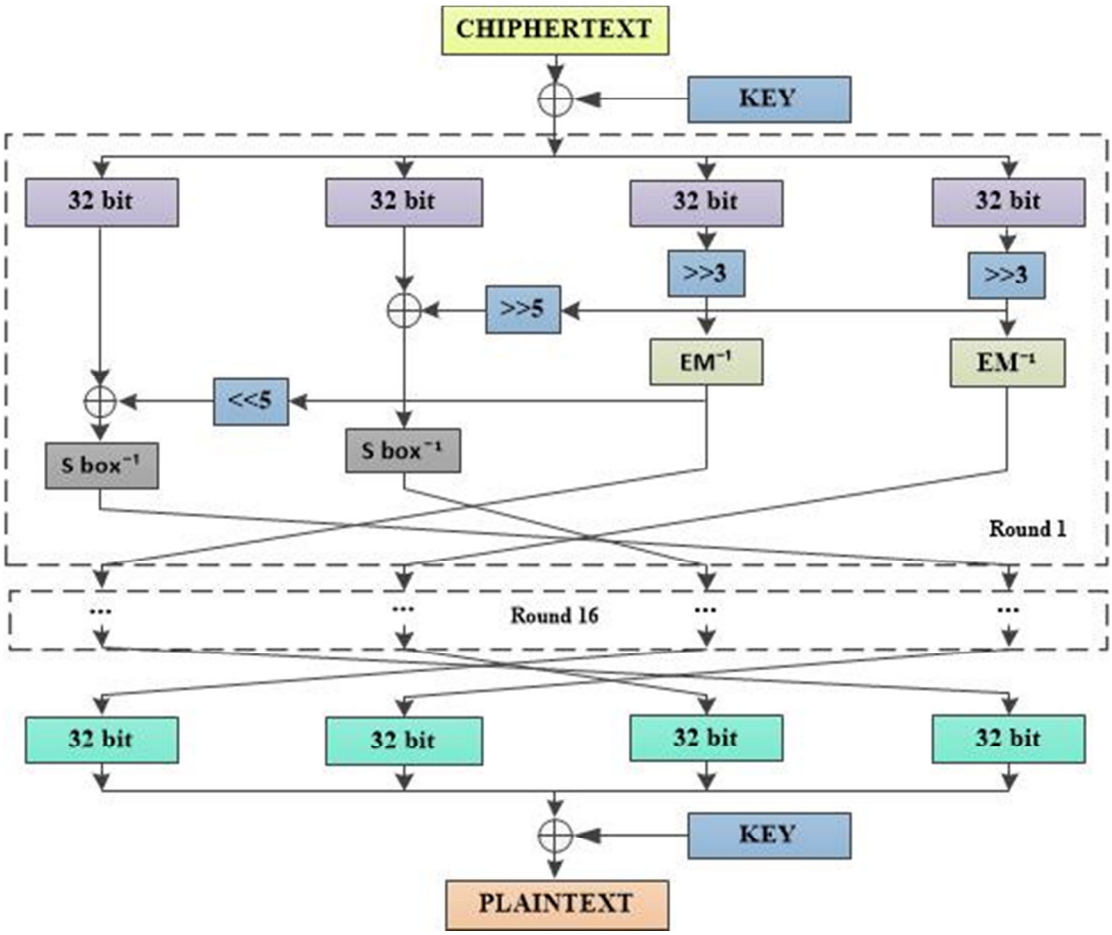
3-кезең. 3-ші және 4-ші ішкі блоктар сұлба бойынша S-box алмастыру кестесінен өтеді, нәтижесі 1-ші және 2-ші ішкі блоктардың 2-кезеңдегі нәтижісімен модуль екі бойынша қосу операциясы орындалады;

4-кезең. Әрбір ішкі блоктар раунд сайын сұлба бойынша орын ауыстырады [52];

Соңғы раундтан кейін шыққан блокқа модуль екі бойынша кілт қосылып шифрмәтін алынады.



Сурет 2.3- «EMCipher» алгоритмінің сұлбасы



Сурет 2.4- «EMCipher» алгоритмінің дешифрлау сұлбасы

2.2.1 EM түрлендіру әдісі

EM (exponentiation modul) түрлендіру әдісі. Кеңейту Галуа $GF(2^v)$ өрісінде дәрежеге шығару операциясына негізделген позициялық емес полиномды санау жүйесінде (ПЕПСЖ) жұмыс істейтін EM (exponentiation modul) түрлендіру әдісі үш кезеңнен тұрады:

- жұмыс негіздер жүйесін құру және олардың орналасу ретін таңдау;
- раундық кілттерді жасау;
- кіріс деректерін түрлендіру және кері түрлендіру.

Бірінші кезең. Жұмыс негіздерін таңдап алу кезеңін қарастырайық. Ол үшін екілік тізбектегі келтірілмейтін көпмүшеліктердің сәйкесінше n_1 дәрежесінің саны m_1-ге, n_2 дәрежесінің саны m_2 -ге, n_S дәрежесінің саны m_S -ге тең болсын [53,54].

Онда жұмыс негіздерінің ең үлкен дәрежесі H-қа тең және m_i, $i = \overline{1,S}$-ке дейінгі дәрежелі жұмыс негіздерін таңдау кезеңдерінде (2.6) теңдікті қанағаттандыратындай алгебралық теңдеудің барлық ықтималды шешімдерін табамыз

$$l_1 m_1 + l_2 m_2 + \ldots + l_S m_S = H. \qquad (2.6)$$

мұндағы $0 \le l_i \le n_i,\ i = \overline{1,S}$-белгісіз коэффициент, l_i- таңдап алынған m_i дәрежелі келтірілмейтін көпмүшеліктердің саны, n_i −барлық m_i дәрежелі келтірілмейтін көпмүшеліктердің саны, мұндағы $1 \le m_i \le H$, онда барлық жұмыс негіздерінің саны мынаған тең:

$$S = l_1 + l_2 + \ldots + l_S. \tag{2.7}$$

Екінші кезең. Кеңейтілген Галуа өрісінде дәрежеге шығару операциясына негізделген түрлендіру әдісін жүзеге асыру үшін раундттық кілттер k_i мәнін раундттық кілттер жасау алгоритімі көмегімен аламыз:

- алынған екілік тізбекті таңдалған жұмыс негіздерінің дәрежелеріне сәйкес бөлшектеу;
- екілік жүйедегі тізбектерді ондық жүйеге ауыстыру;
- алынған k_i мәнді ЕҮОБ(k_i,$p^{deg(p_i(x))} - 1$)=1 болатындай таңдаймыз.

Үшінші кезең. Модуль бойынша дәрежеге шығару операциясын пайдалану негізінде деректерді шифрлау жылдамдығы көп уақытты талап ететіндігі белгілі. Алайда, бұл процедураның есептеу уақытын жылдамдату үшін ПЕПСЖ-ні қолдану орынды [37,52,54].

Ұсынылып отырған түрлендіру әдісінде кіріс деректері 128 ұзындықтағы биттер түрінде берілгендіктен. Оны 32 ұзындықтағы блоктарға бөліп, әр блокпен жұмыс жасаймыз.Түрлендіру әдісінің блок схемасы 2.5-суретте ал керісі (қосымша Б, сурет Қ.1) көрсетілген. Әрбір 32 ұзындықтағы блок жұмыс негіздерінің дәрежесі бойынша бөліктерге бөлінеді. Алынған бөлікті (2.8) формуладағы ПЕПСЖ-дегі қалдықтардың тізбегі ретінде өрнектейміз.

$$A(x) = a_1(x), a_2(x), \ldots, a_n(x), \tag{2.8}$$

мұндағы $a_i(x)$ -алынған бөліктер, $i = \overline{1,n}$.

(2.8) формула бойынша бөлініп алынған блоктарды түрлендіру формуласын төмендегідей өрнектеп аламыз:

$$b_i(x) = a_i^{k_i}(x)\ mod\ p_i\ (x), \quad i = \overline{1,n}. \tag{2.9}$$

(2.9) формуласы бойынша алынған шифрмәтіндер жүйесін былай өрнектейміз:

$$B(x) = (b_1(x), b_2(x), \ldots, b_n(x)). \tag{2.10}$$

Онда керітүрлендіру мынаған тең:

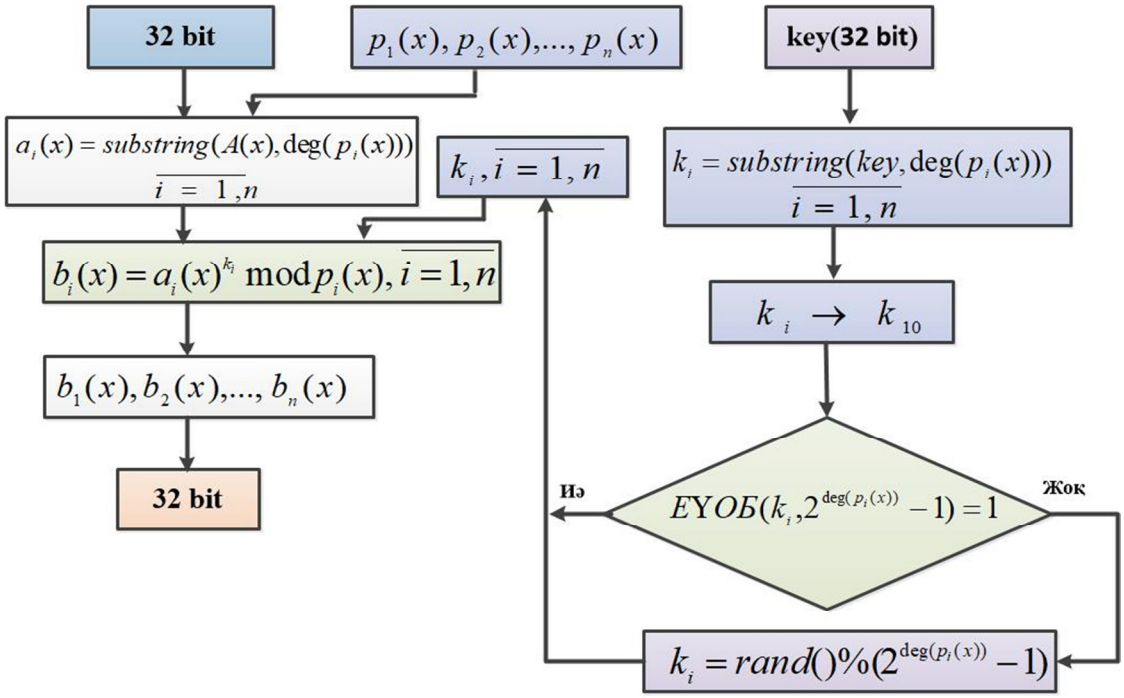
$$a_i(x) = b_i^{k_i^{-1}}(x)\ mod\ p_i\ (x). \tag{2.11}$$

Алынған ашық мәтіндердер жиыны тең (2.12) формулаға тең:

$$A(x) = a_1(x), a_2(x), \ldots, a_n(x). \quad (2.12)$$

Ұсынылған алгоритмде әрбір блокқа қолданылатын кілттің керісін есептейміз:

$$k_i \cdot (k_i)^{-1} \equiv 1 \, mod(\, p^{deg(p_i(x))} - 1), i = \overline{1,n}. \quad (2.13)$$



Сурет 2.5- ЕМ түрлендіру әдісінің блок схемасы

Сипатталған алгоритм бойынша төмендегі мысалды қарастырайық.

Мысалы: $GF(2^3)$ өрісін алып, осы өрісте жататын келесі полиномьдарды алайық (жұмыс негіздерін таңдаймыз).

$$p_1(x) = x^2 + x + 1, p_2(x) = x^3 + x^2 + 1, \; p_3(x) = x^3 + x + 1.$$

Онда жұмыс негіздерінің диапазон көлемі мынаған тең:

$$P(x) = \prod_{i=1}^{3} p_i(x) = x^8 + x^6 + x^5 + x^4 + x^3 + x^2 + 1;$$

Шифрлау қажет биттер ағыны (ашық мәтін) төмендегі екілік тізбектен берілсін:

$$A = \{110111000111101100011011\}.$$

Екілік жүйедегі ашық мәтінді ПЕПСЖ-нің жұмыс диапазонының дәрежесі бойынша блоктарға бөліп және алынған блоктарды көпмүшелік түрінде өрнектейміз:

Біздің жағыдайда:

$A_1 = 11011100;\ A_1(x) = x^7 + x^6 + x^4 + x^3 + x^2;$

$A_2 = 01111011;\ A_2(x) = x^6 + x^5 + x^4 + x^3 + x + 1;$

$A_3 = 00011011;\ A_3(x) = x^4 + x^3 + x + 1;$

Шифрлау процесін диапазон дәрежесі бойынша бөлініп алынған бір блокта сипаттайық. Онда алынған блокты жұмыс негіздерінің дәрежесі бойынша бөліп, оны көпмүшелік түрінде өрнектейік:

$a_1 = 11, a_1(x) = x + 1;$

$a_2 = 011, a_2(x) = x + 1;$

$a_3 = 100, a_3(x) = x^2;$

$$A_1(x) = (x + 1, x + 1, x^2).$$

Кездейсоқ тізбек генераторынан алынған кілттік тізбек мынаған тең болсын:

$$K = (10101110).$$

Алынған тізбектен кілтерді жұмыс негіздерінің дәрежесіне сәйкес кесіп аламыз:

$$k_1 = (10),\ k_2 = (101), k_3 = (110).$$

Блокқа бөлінген екілік жүйедегі кілттерді ондық жүйеге ауыстырамыз.

$$k_i = (2,5,6).$$

Ашық мәтінді (2.9) формула бойынша шифрлаймыз:

$$b_i(x) = (a_i(x))^{k_i}\, mod\, p_i\,(x) = ((x + 1)^2\, mod(\, x^2 + x + 1),\ (x + 1)^5\, mod(\, x^3 + x^2 + 1), (x^2)^6\, mod(\, x^3 + x + 1)) = (x, x^2 + x + 1, x^2 + x + 1);$$

Шифрмәтіндер төмендегідей болады:

$$b_1(x) = x;$$
$$b_2(x) = x^2 + x + 1;$$
$$b_3(x) = x^2 + x + 1;$$

Өз кезегінде бұны (2.13) формулаға сәйкес жазамыз:

$$B_1(x) = (x, x^2 + x + 1, x^2 + x + 1);$$

Енді дешифрлау үшін (2.13) формула бойынша k_i-дің кері элементтерін табамыз, бұл жағыдайда әр алынған жұмыс негіздеріне сәйкес:

- 2-нің кері элементі модуль 3 бойынша 2-ке тең;
- 5 кері элементі модуль 7 бойынша 3-ке тең;
- 6 кері элементі модуль 7 бойынша 6-ке тең;

(2.11) формулаға сәйкес дешифрлаймыз:

$$a_i(x) = b_i(x)^{k_i^{-1}} \, mod \, p_i\,(x)((x^2 \, mod(\, x^2 + x + 1), (x^2 + x + 1)^3 \, mod(\, x^3 + x^2 + 1), (x^2 + x + 1)^6 \, mod(\, x^3 + x + 1)) = (x + 1, x + 1, x^2);$$

Сонымен алынған ашықмәтіндерді келесідей жазып алуға болады:

$$a_1(x) = x + 1;$$
$$a_2(x) = x + 1;$$
$$a_3(x) = x^2; \text{ немесе}$$
$$A_1(x) = (x + 1, x + 1, x^2);$$

Берілген көпмүшеліктерді екілік жүйеде таңдалған жұмыс негіздерінің дәрежесі бойынша сипаттасақ:

$$A_1 = (11011100);$$

Қолданылып отырған ЕМ түрлендіру әдісінде дәрежеге шығару деген ұғымда оны есептеу процессі көп уақытты алатыны белгілі. Бірақ біз ұсынылып отырған алгоритмде дәрежені есептеу таңдалған жұмыс негіздерінің индекс кестесін құру арқылы орындап отырғандықтан есептеу жылдамдығы артады [55,56]. Алгоритмнің жылдамдығын тексеру үшін қарапайым дәрежені және индекс кестесі негізінде есептеу бағдарламасы құрылып нәтижелер сипаттамалары әртүрлі компютерлермен тексеріліп нәтижелер алынды (қосымша Б, кесте Қ.14). Ұсынылған алгоритмнің индекс кестесін пайдалана отырып ЕМ түрлендіруінде таңдап алған жұмыс негіздері $p_i(x)$ бойынша индекс кестесін келесі заңдылық бойынша толтырамыз [48,52]

$$a(x) = \alpha^j \, mod \, p_i\,(x), j = \overline{0, 2^{deg(p_i(x))} - 2}\,, \qquad (2.14)$$

α - $GF(2^v)$ өрісіндегі мультипликативті топты тудырушы элемент.

Мысалы: $GF(2^3)$ *өрісіндегі* жұмыс негіздері $p(x) = x^3 + x + 1$ тең келтірілмейтін көпмүшеліктің индекс кестесін қарастырайық (2.1-кесте).

Кесте 2.1- $p(x) = x^3 + x + 1$ жұмыс негіздерінің индекс кестесі

Индексті берілуі ind	Көпмүшелік түрде берілуі $a(x)$
α^{∞}	0
α^0	1
α^1	x
α^2	x^2
α^3	$x+1$
α^4	x^2+x
α^5	x^2+x+1
α^6	x^2+1

Таңдалған жұмыс негіздерінің индекс кестесіне сәйкес келесі математикалық теңдеуді енгізейік

$$l = \underset{\alpha}{ind} a_i(x) \, mod \, p_i\,(x), \tag{2.15}$$

мұндағы l - $a(x)$ -тің α-бойынша алынған дәрежесі немесе индексі (ind). Ондай болатын болса онда (2.9) формулаға келісідей өзгерту енгіземіз:

$$b_i(x) = (\alpha^l)^{k_i} \, mod \, p_i\,(x) = (\alpha^{(lk_i) \, mod (2^{deg(p_i(x))}-1)}) \, mod \, p_i\,(x). \tag{2.16}$$

Кері түрлендіру процессіндеде k_i-дің орынына кері элементі $(k_i)^{-1}$-ді қолданамыз:

$$\underset{\alpha}{ind} b_i(x) \, mod \, p_i\,(x) \tag{2.17}$$

$$a_i(x) = (\alpha^l)^{k_i^{-1}} \, mod \, p_i\,(x) = (\alpha^{(lk_i^{-1}) \, mod (2^{deg(p_i(x))}-1)}) \, mod \, p_i\,(x) \tag{2.18}$$

EM (Exponentiation Module) түрлендіру әдісі Галуа $GF(2^v)$ өрісінде дәрежеге шығару операциясына негізделген позициялық емес полиномьды санау жүйесінде немесе қалдықтар классының санау жүйесінде жұмыс істейтіндіктен келесі мүмкіндіктерге ие:

- разрядаралық тасымалдың болмауы;
- үлкен көлемдегі позициялық санау жүйесіндегі разрядттарды таңдап алған жұмыс негіздерінің қалдығы ретінде өрнектеу нәтижесінде шифрлау жылдамдығы артады;
- қателерді табу және түзету мүмкіндігі;
- әр түрлі электрондық жүйелер мен желілердегі ақпаратты қорғаудың жоғары тиімді құралдарын жасау мүмкіндігі.

2.2.1 Шифрлау алгоритміне арналған S-блок құру

S-блок - бұл алдын ала тағайындалған ауыстыру кестесі онда, *m*-кіріс мәндері кіредіде, n-шығыс мәндері алынады. S-блоктар әдетте түрлендіру әдісінің бір бөлігі болып табылады және шифрлау алгоритмінің криптоберіктілігі үшін үлкен мәнге ие. S-блокқа кіретін кіріс мәндерін өзгерткен кезде, шығыс мәндеріндегі биттер кез-келген болып таңдалатындай болу керек. Сонымен қатар шығу мәндерінің кіріс мәндеріне тәуелділігі сызықтық болмауы немесе сызықтық функциялармен оңай жақындатылмауы керек (бұл қасиет сызықтық криптоталдауды қолдану кезінде қолданылады). Қазіргі уақытта AES, GOST R 34.12-2015, DES, Twofish және т.б. көптеген симметриялық шифрлау алгоритмдері S-блокты қолданылатыны белгілі [13,23,25,29].

Сондықтан жақсы S блоктарды таңдап алу оңай емес. Қазіргі кезде S блок алу әдістерінің көптеген бәсекелестік тәсілдері бар, олардың ішіндегі негізгі 4 әдісті қарастырсақ. Олар:

- Кездейсоқ таңдау. S-блоктардың беріктілігі неқұрлым кездейсоқ және кілтке тәуелді болса сенімді болады. Кездейсоқ таңдап алу барысында шағын S-блоктар әлсіз, ал үлкен S-блоктар жеткілікті сенімді болуы мүмкін. Қазіргі уақытта сегіз және одан да көп кірісі бар кездейсоқ S-блоктар қолданылатыны белгілі осыған орай кездейсоқ таңдап алынған S-блоктар жеткілікті сенімді болуы мүмкін.
- Кезекті тестілеу арқылы таңдау. Кейбір шифрлерде алдымен кездейсоқ S-блоктары жасалынады, кейін олардың қасиеттері талаптарға сай тексеріледі.
- Қолмен әзірлеу. Бұл жағдайда математикалық аппарат өте аз қолданылады: S-блоктар интуитивтік ойлау әдістері арқылы жасалады. Барт Пренель өзінің мақаласында «... теориялық тұрғыдан қызықты критерийлер жеткіліксіз (S-блоктардың логикалық функцияларын таңдау үшін) ...» және «... арнайы дизайн критерийлері қажет» деп мәлімдеді [21,50,52,57].
- Математикалық әзірлеу. S-блоктар математика заңдарына сәйкес жасалынады, сондықтан олар Дифференциалды және сызықты криптоталдау мен жақсы шашырау қасиеттеріне төзімді. «Математикалық» және «қолмен» әзірлеу тәсілдерін біріктіру туралы ұсыныстар болды, бірақ іс жүзінде кездейсоқ таңдалған S-блоктар мен белгілі бір қасиеттері бар S-блоктары бәсекелесе алады. Соңғы тәсілдің артықшылығы ашудың белгілі әдістері дифференциалды және сызықтық криптоталдауға қарсы оңтайландыруды қамтиды.

ЕМ түрлендіру әдісі негізінде ұсынылған блокты шифрлау алгоритміне қолданылатын S-блокты алу әдісін қарастырсақ.

Ол үшін біз Галуа $GF(2^v)$ өрісіндегі мультипликативті топ құратын келтірілмейтін көпмүшелік және кез-келген негіз деп аталатын көпмүшелік таңдаймыз. Содан кейін таңдалған көпмүшелік арқылы модуль бойынша дәрежеге шығару операциясын орындаймыз:

$$S_i = A(x)^{p_i(x)}(x)\ mod\ P\ (x),\ i = \overline{0, 2^{deg(P(x))} - 1}, \qquad (2.19)$$

мұндағы S_i- S-блоктың элементтері, $A(x)$ - негіз деп аталатын көпмүшелік, $p_i(x)$ - мультипликативті топтың элементтері, $P(x)$ -модуль (келтірілмейтін көпмүшелік). Ұсынылған әдіс бойынша алынған S-блок ауыстыру кестесі және кері ауыстыру кестелері 2.2, 2.3-кестелерде көрсетілген.

Кесте 2.2- S-блок ауыстыру кестесі

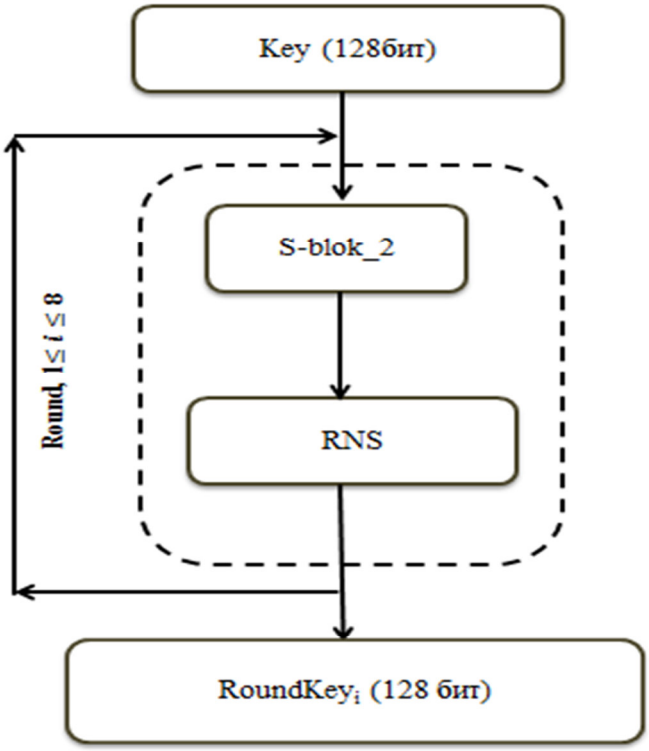
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	1B	34	5E	D4	65	13	EC	8F	C6	92	A8	74	C9	B	F5
1	8D	F0	FA	14	AD	03	2D	5C	E2	D	AF	35	45	E0	3B	C7
2	89	9C	2A	1D	6E	E6	61	7F	3C	86	05	77	E4	57	17	80
3	5F	CF	51	4D	38	EA	D5	7E	27	B2	5B	A3	81	44	FB	F
4	99	5D	F9	39	F1	E1	20	F3	D7	48	4F	E	82	69	A7	ED
5	94	F2	CC	7C	11	DA	E7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

Кесте 2.3- Кері S-блок ауыстыру кестесі

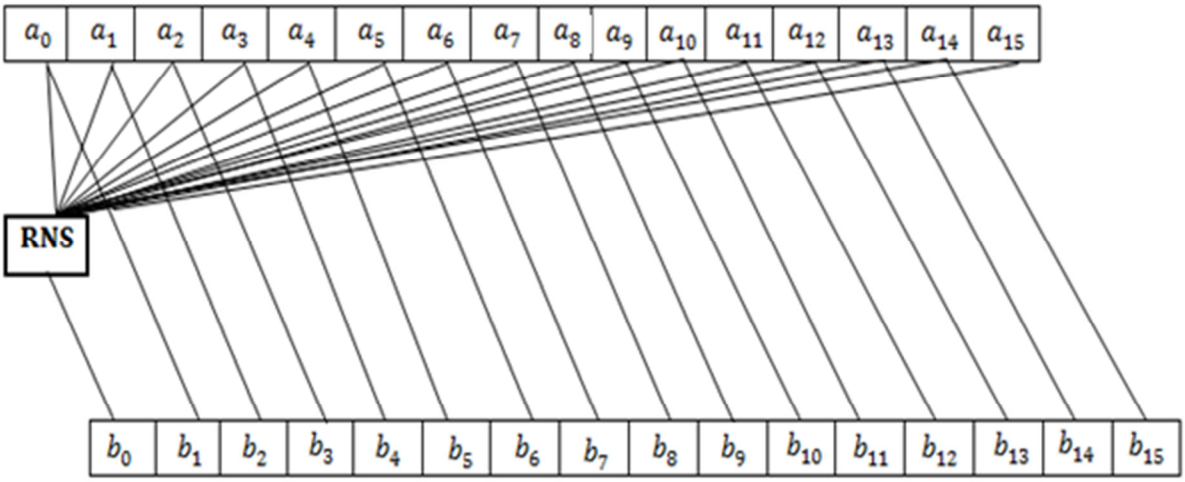
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	FF	00	74	15	E8	2A	89	D6	5D	EB	9E	E	FD	19	4B	3F
1	D1	54	60	06	13	AD	82	2E	72	73	8D	01	BF	23	B3	6A
2	46	7F	C8	EF	D4	67	7A	38	87	E5	22	88	F6	16	A2	8E
3	E6	E4	E7	69	02	1B	75	AF	34	43	97	1E	28	E2	DE	C2
4	BA	D7	F3	B4	3D	1C	64	F7	49	95	DB	58	EE	33	AC	4A
5	EB	32	5A	F9	96	7E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

2.3 Раундттық кілттерді жасау алгоритмі

Ұсынылып отырған раундттық кілттерді жасау алгоритмінің сұлбасы 2.6-суретте көрсетілген. Онда 128 бит ұзындықтағы негізгі кілт S-блок ауыстыру кестесі (Қосымша Б, Кесте Қ.15) өткенен кейінгі 128 бит байттарға ауысып, әр байт 2.7-суретте көрсетілгендей RNS түрлендірунен өтеді.



Сурет 2.6- Раундттық кілттерді жасау алгоритмінің сұлбасы



Сурет 2.7 - RNS түрлендіру әдісі

2.3.1 RNS түрлендіру әдісі

RNS түрлендіру әдісі келесідей орындалады:

Әр бір блоктың байттары таңдап алынған $p_i(x), i = \overline{1,2}$ жұмыс негіздерінің қалдығы ретінде өрнектейміз.

$$a_j(x) = \left(\alpha_{1j}(x), \alpha_{2j}(x)\right), \ \ j = \overline{0,15} \tag{2.20}$$

мұндағы $a_j(x) \equiv \alpha_{ij}(x) \bmod p_i(x) \ \ i = \overline{1,2}$.

Таңдап алынған $\alpha_{ij}(x) -$ ге, i $= \overline{1,2}, \ j = \overline{0,15}$ кез-келген $r(x)$ көпмүшелігі модуль екі бойынша қосылады (2.21-формула) және алынған $\beta_{ij}(x), \ p_i(x)$ жұмыс негіздерінің дәрежесі бойынша бөліктерге бөлінеді. Алынған бөлікті (2.22) формуладағы ПЕПСЖ-дегі қалдықтардың тізбегі ретінде өрнектейміз.

$$\beta_{ij}(x) = \alpha_{ij}(x) \oplus r(x) \tag{2.21}$$

$$\beta_{ij}(x) = (\beta_{1j}(x), \beta_{2j}(x)) \tag{2.22}$$

Алынған ПЕПСЖ-дегі қалдықтардың тізбегін Қалдықтар туралы қытай теоремасын қолдана отырып жалғыз шешімін аламыз.

$$F_n(x) = \sum_{i=1}^{2} \beta_{ij}(x) B(x) mod P(x), \quad j = \overline{0,15}, \ n = \overline{0,15} \tag{2.23}$$

мұндағы $B(x)$-базис (2.4) формула бойынша, ал $P(x)$-жұмыс негіздерінің диапазоны көлемі (2.3) формуламен анықталады.

Енді алынған әр бір $F_n(x)$ үшін келесі формула орындалады.

$$b_1(x) = \oplus \sum_{n=0}^{15} F_n(x) mod P(x) \tag{2.24}$$

Ал қалған байттар 2.7-суретте көрсетілгендей орындалады бұл процесс 16 рет қайталанып раундттық кілт алынады.

2.4 Бөлім бойынша қорытынды

Заманауи симметриялық блокты шифрлау алгоритмдеріне талдау жүргізе отырып екінші бөлімде жаңа EMCipher шифрлау алгоритмі құрылды. Сонымен қатар алгоритмде қолданылатын EM түрлендіру әдісі, S-блок ауыстыру кестесін алу әдісі, раундттық кілттерді түзеу алгоритмінің құрылымы сипатталды. Алгоритмге EMCipher деп ат берілді. Құрылған алгоритм симметриялық шифрлау алгоритміне қойлатын талаптарды қанағаттандырады.

3 ҚҰРЫЛҒАН СИММЕТРИЯЛЫ БЛОКТЫ ШИФРЛАУ АЛГОРИТМІНІҢ СЕНІМДІЛІГІН ЗЕРТТЕУ

3.1 EMCipher алгоритмін бағдарламалық жүзеге асыру

EMCipher алгоритмінің сенімділігін тексеру үшін алгоритм бағдарламалық жүзеге асырылды. Бағдарлама Microsoft Visual Studio 2013 бағдарламалау ортасында C++ тілінде жазылды. Бағдарламаға **«CryptoEM v1.0.1»** файлдарды шифрлау бағдарламасы деген атау беріліп, авторлық куәлік алынды (қосымша Ә). Жасалынған бағдарлама келесі мүмкіндіктерге ие.

Қолдану ортасы:

- телекоммуникациялық және ақпараттық жүйелер мен желілердегі, электрондық құжат айналымы жүйелеріндегі ақпаратты криптографиялық қорғау жүйелерінде;
- отандық ақпараттық-коммуникациялық технологиялардың бағдарламалық өнімдерінде;
- позициялық емес полиномды санау жүйесі негізінде құрылған және Галуа $GF(2^v)$ өрісіндегі модуль бойынша дәрежеге шығару негізінде түрлендіру операция негіндегі криптографиялық ақпаратты қорғау жүйелерінде [58].

Функциональдық мүмкіндігі:

Бағдарлама EMCipher алгоритмін қолдана отырып, әр түрлі форматтағы электрондық файлдарды шифрлауға және дешифрлауға арналған.

Бағдарлама үшін кіріс және шығыс келесі мәлімет болып табылады:

1) кіріс мәліметтері:

- шифрлауға/дешифрлауға арналған файл;
- негізгі шифрлау кілті (ұзындығы 256 бит), оны қолмен және файл түрінде енгізу мүмкіндігі;
- жұмыс негіздерін оларды қолмен (бит түрінде) және файл түрінде енгізу мүмкіндігі бар;

2) шығыс мәліметтері:

- шифрланған/дешифрланған файлдың қайда сақталады және атауы кеңейтуін көрсету.

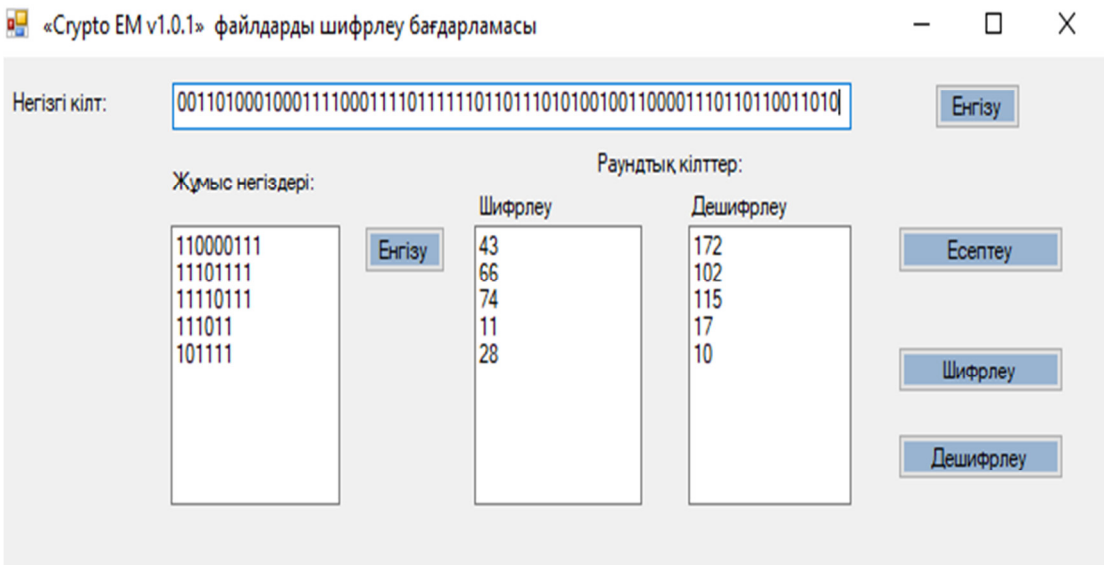
Негізгі техникалық сипаттамалары:

«CryptoEM v1.0.1» компьютерлік бағдарламасының дискідегі бос орын көлемі - 1,13 МБ. Бағдарламалық орнату қажет емес. Бағдарламаны жергілікті немесе тасымалдау дискісіне көшіруге болады. Ол «CryptoEM v1.0.1.exe» негізгі файлынан - Windows ортасындағы қосымшадан тұрады. Осы компьютерлік бағдарламаны файл түрінде пайдалану кезінде алынған мәліметтер пайдаланушы көрсеткен каталогтарда жазылады.

Іске асыратын ЭЕМ-ге қойылатын ең төменгі талаптар:

Pentium/AMD, тактілік жиілік - кемінде 600МГц, оперативті жады – 256М-ден кем емес, қатты дискідегі бос орын – 500 МБ-ден кем емес, операциялық жүйелер - Windows XP / 7.

Бағдарлама «CryptoEM v1.0.1.exe» файлы арқылы іске қосыладыда нәтижесінде негізгі диалогты терезе пайда болады (3.1-сурет).



Сурет 3.1- «CryptoEM v1.0.1» файлдарды шифрлау бағдарламасы

Бағдарламаның негізгі терезесі келесі элементтерден тұрады:

- 4 диалогты терезе тұрады, оның ішінде 2 - мәліметтерді енгізу, 2 – нәтижелерді шығару терезесі.
- 5 командалық батырмалар.

Әрбір диалогты терезелер мен батырмалардың қызметтері:

1-терезе: Негізгі кілтті енгізу терезесі, кілттерді екілік код түрде енгізу керек қолмен немесе файл түрінде енгізуге мүмкіндік бар;

2-терезе: Ұзындығы 32 битке дейінгі жолдармен бөлінген әр түрлі жұмыс негіздерін енгізу терезесі, қолмен немесе файл түрінде енгізуге мүмкіндік бар;

3,4 -терезе: Есептеуден кейін алынған шифрлау немесе дешифрлауге қолданылатын раундттық кілтерді көрсету терезесі;

1-батырма: Негізгі кілт бар файлды енгізуге арналған диалогтық терезені ашуға арналған батырма;

2-батырма: Жұмыс негіздерін орналасқан файлды енгізуге арналған батырма;

3-батырма: Шифрлау және дешифрлауға арналған раундттық кілттерді есептеуге арналған батырма;

4,5-батырма: Шифрлау/Дешифрлау батырмасын басқанда шифрленетін немесе дешифрленетін файлды таңдайтын терезе ашылады, кейін қайда сақтау керектігін көрсететін терезе ашылады.

3.2 EMCipher алгоритмінің статистикалық қауіпсіздігін зерттеу

Криптографиялық алгоритмдердің беріктігін бағалаудың негізгі компоненттерінің бірі оның статистикалық қауіпсіздігін бағалау болып табылады. Шифрлау алгоритмдерден алынған шифрмәтіндердің тізбегі кездейсоқтық қасиеттерін қамтамасыз етсе, онда алгоритм статистикалық тұрғыдан қауіпсіз болып саналады. Сондықтан ұсынылған шифрлау алгоритмінен алынған шифрмәтіндердің статистикалық қасиеттерін зерттеу үшін графикалық және бағалау тестілері қолданылады[59,60].

Графикалық тестілерде құрылған тізбектердің статистикалық қасиеттері графикалық көрініс түрінде бейнеленеді, оған сәйкес тест нәтижелеріне сапалы бағалау жасалады.

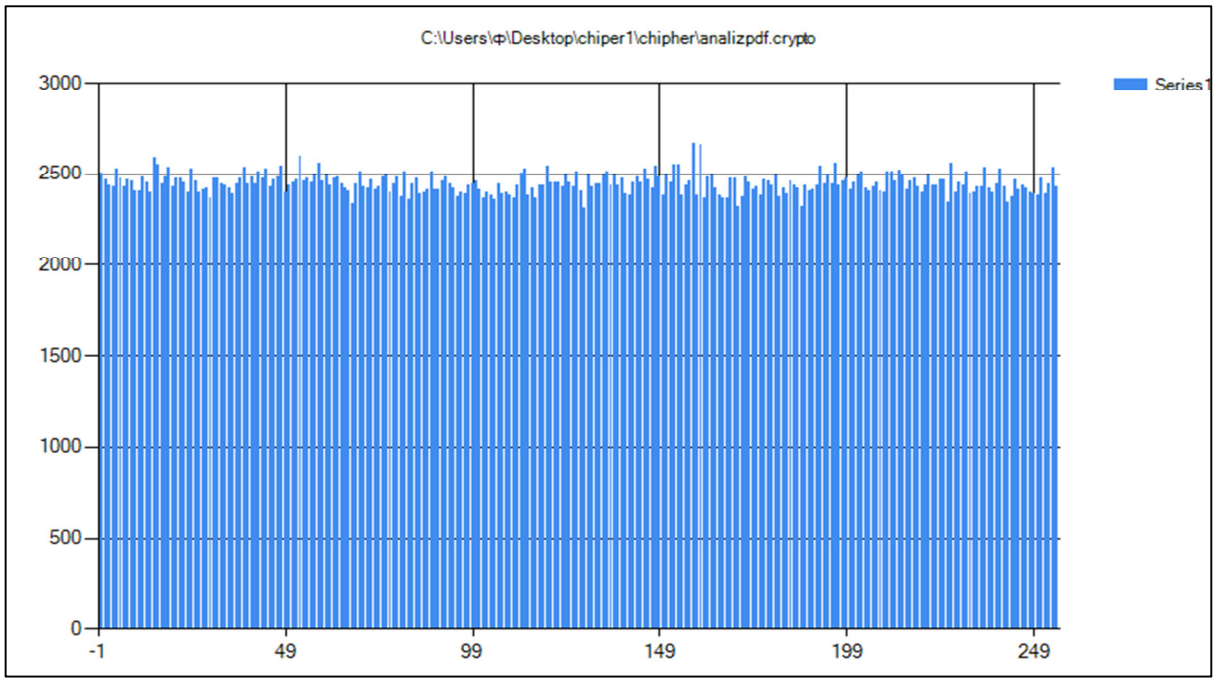
Статистикалық қасиеттерді анықтау үшін келесі графикалық тесттер қолданылды:

- тізбектің элементтерінің таралу гистограммасы;
- жазықтықтағы үлестірім;
- серияларды тексеру;
- монотондылықты тексеру;
- автокорреляция функциясы;
- сызықтық күрделілік профилі;
- графикалық спектрлік тест.

Шифрмәтіннің статистикалық қасиеттерін графикалық тесттер арқылы анықтау үшін өлшемі 646кб pdf файлын шифрлап алынған шифрмәтінге зерттеу жүргізіліп нәтижелері әр-бір тесттерді сипаттау аймағында көрсетілген

«Тізбектің элементтерінің таралу гистограммасы» тестісі. Зерттелетін тізбектегі таңбалардың біркелкі таралуын бағалауға, сонымен қатар белгілі бір таңбаның пайда болу жиілігін анықтауға мүмкіндік береді. Зерттелетін элементтер тізбегінде әр элементтің қанша рет кездесетіні есептеледі, содан кейін элементтердің пайда болу санымен олардың сандық көрінісі арасындағы тәуелділігі графигі жасалады. Тізбек кездейсоқтықтың қасиеттерін қанағаттандыру үшін тізбекте қарастырылатын разрядтағы элементтердің барлық мүмкін жағдайлары болуы керек, сонымен қатар таңбалардың пайда болу жиіліктерінің таралуы нөлге жетуі керек. 3.2-суретте жоғарыда келтірілген шифрмәтіннің тізбектің элементтерінің таралу гистограммасы көрсетілген.

«Жазықтықтағы үлестірім» тестісі. Зерттелетін тізбектің элементтері арасындағы өзара тәуелділікті анықтауға арналған. Жазықтықта үлестірімді құру келесідей жүзеге асырылады. $m \times m$ өлшемдегі өрісте координаталары $(e_i; e_{i+1})$ болатын нүктелер түсіріледі, мұндағы m – алфавиттің қуаттылығы, $0 \leq e_i \leq m$, e_i – зерттелетін тізбектің элементтері, $i = \overline{1, (n-1)}$, n – тізбектің ұзындығы.

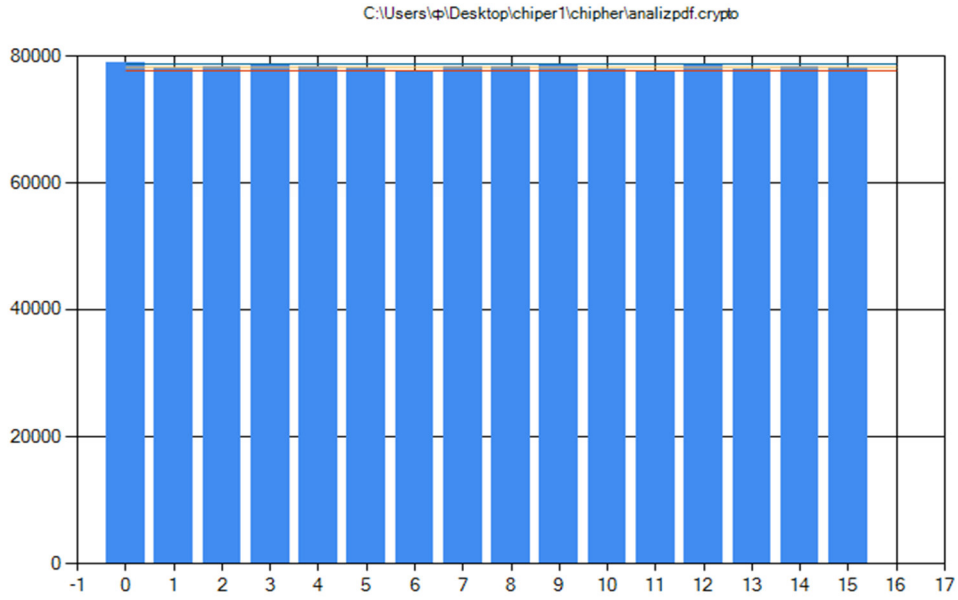


Сурет 3.2 - Тізбектің элементтерінің таралу гистограммасы

Осыдан кейін алынған бейне талданатын болады. Егер тізбектің элементтерінің арасында тәуелділік болмаса, онда өрістегі нүктелер кездейсоқ орналасқан болады. Егер өрісте тәуелділіктер байқалатын болса, яғни қандай да бір "өрнектер" байқалса - тізбек кездейсоқ емес. Алынған бейне біріңғай бір түсті болуы үлкен ұзындықтағы тізбектер үшін жақсы нәтиже. Көріп отырғанымыдай 3.3-суретте кеңейтуі pdf файлынан алынған шифрмәтін жазықтықтағы үлестірім тестісінен өтетіндігін көруге болады.

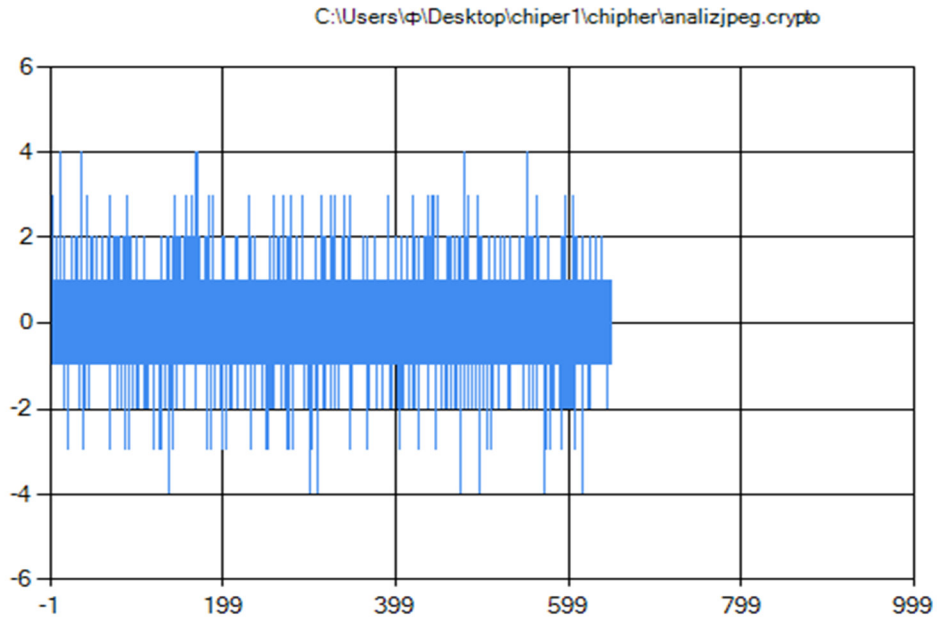
Сурет 3.3 - Жазықтықтағы үлестірім тестісі.

«Сериялардың тексеру» тестісі. k биттен тұратын сериядағы нөлдер мен бірліктердің пайда болу жиілігін талдау негізінде зерттелетін тізбектегі таңбалардың біркелкі таралуын бағалауға мүмкіндік береді. Зерттеу келесідей жүзеге асырылады. Зерттелетін тізбектің биттік көрінісіндегі нөлдер, бірліктер, екілік биттегі сериялар, үштік биттегі сериялар және т.б. неше рет рет кездесетіні есептелінеді. Алынған нәтижелер графикалық түрде қарастырылады. Статистикалық қасиеттері шынайы кездейсоқ тізбектің қасиеттеріне жақын тізбекте нөлдер мен бірліктердің пайда болу сандары арасындағы алшақтық, жұптарының әрбір түрінің арасындағы сериялардың пайда болу сандары арасындағы алшақтық және үштік биттердің әрбір түрінің серияларының пайда болу сандары арасындағы алшақтық нөлге ұмтылуы. 3.4-суреттегі диаграммалардың бір келкі орналасуын ескерсек, оның сериялары тексеру тестісінен өтетіндігін көрсетеді [52,59,60].



Сурет 3.4 - Сериялары тексеру тестісі

«Монотондылықты тексеру» тестісі. Тізбек элементтерінің өсіп кетпеу және кемімеу учаскелерінің ұзындықтарын талдау негізінде зерттелетін тізбектегі таңбалардың біркелкі таралуын бағалауға мүмкіндік береді. Талдау келесідей жасалады. Зерттелетін тізбек дәйектілік бірізділік элементтерінің бір-бірінен кейінгі өспейтін және кемімейтін қиылыспайтын учаскелері негізінде графикалық түрде ұсынылады. Статистикалық қасиеттері шынайы кездейсоқ тізбектің қасиеттеріне жақын тізбекте белгілі бір мөлшерде өсіп кетпейтін (кеміп кетпейтін) аумақтың пайда болу ықтималдығы оның ұзындығына байланысты: ұзындығы неғұрлым үлкен болса, соғұрлым ықтималдығы аз болады. Кері жағдайда тізбек кездейсоқ болып есептелінбейді. Монотондылықты тексеру тестісі арқылы қарастырып отырған .pdf файлынан алынған шифрмәтіннің нәтижесі 3.5-суретте көрсетілген.



Сурет 3.5 - Монотондылықты тексеру тестісі

«Автокорреляциялық функция (АКФ)» тестісі. Зерттелетін тізбектің жылжытылған көшірмелері арасындағы корреляцияны бағалауға арналған. Талданатын ішкі тізбекшелердің арасындағы тәуелділік табылуы мүмкін.

Биттік АКФ. Биттік АКФ талдануы мынадай түрде жүргізіледі. Алдымен зерттелетін тізбек бит түрінде қарастырылады, содан кейін алынған бит тізбегі (1→1,0→-1) түрде қалыпқа келтіріледі. Осыдан кейін корреляция толқындары есептеледі:

$$c_j = \frac{\sum_{i=0}^{n-1} b_i \cdot b_{(i+j) mod\, n}}{\sum_{i=0}^{n-1} b_i^2},$$

мұндағы b_i – нормаланған тізбектің элементтері, n - нормаланған бит тізбегінің ұзындығы, $i = \overline{0,(n-1)}, j = \overline{0,n}$.

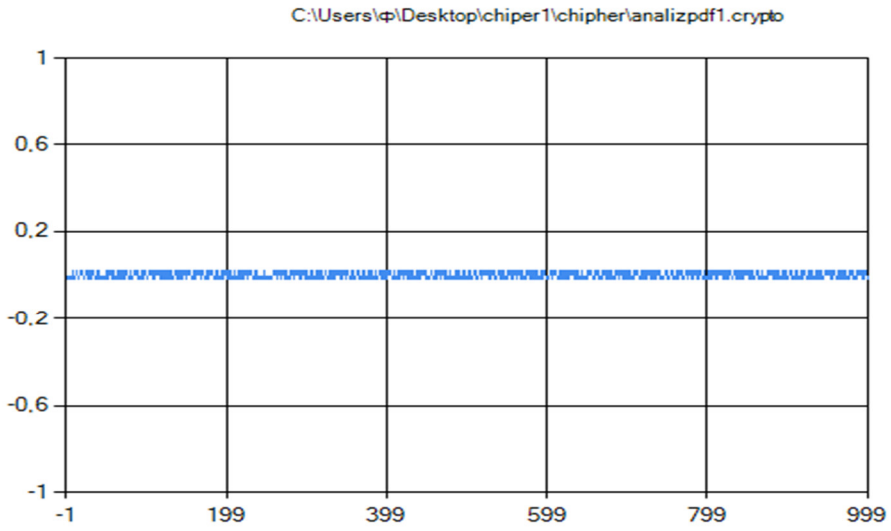
Байттық АКФ. Символдық АКФ талдауы келесідей. Алдымен зерттелетін тізбек қалыпқа келтіріледі. $a_{r-1}a_{r-2}\dots a_0$ (r -санның разрядтылығы) – зерттелетін тізбектің екілік жазбасы болсын. Содан кейін бұл элементтің қалыпты мәні мынандай болады:

$$b_i = \sum_{j=0}^{r-1} \left((-1)^{a_i} \cdot 2^j\right)$$

Осыдан кейін корреляция толқындары есептеледі:

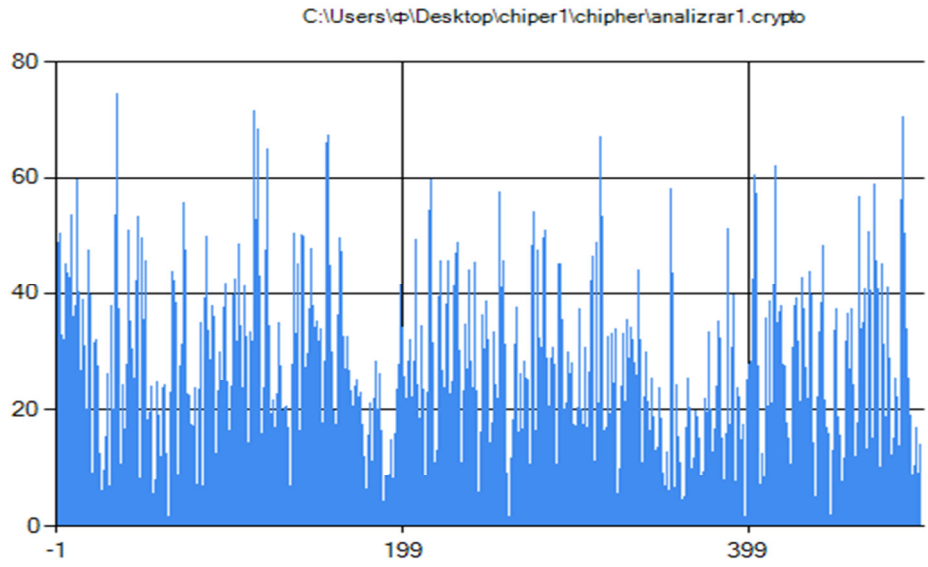
$$c_j = \frac{\sum_{i=0}^{n-1} b_i \cdot b_{(i+j) mod\, n}}{\sum_{i=0}^{n-1} b_i^2},$$

мұндағы b_i – нормаланған тізбектің элементтері, n - нормаланған бит тізбегінің ұзындығы, $j = \overline{0,n}$. Автокорреляциялық функция (АКФ) тестісі арқылы тексерілген шифрмәтіннің нәтижесі 3.6-суретте ұсынылған.



Сурет 3.6 - Автокорреляциялық функция (АКФ) тестісі

«Сызықтық күрделілік профилі» тестісі. Тізбектің сызықтық күрделілігінің оның ұзындығына тәуелділігін талдай отырып, кездейсоқтыққа зерттеуге мүмкіндік береді. Сызықтық күрделілік профилін құру келесідей. $t_1 t_2 t_3 \ldots t_n$ – n ұзындықтағы екілік тізбек болсын. Алғашқы k элементтері бар $t_1 t_2 t_3 \ldots t_k$ ішкі тізбекшелері бірітіндеп рет-ретімен қарастырылады және ішкі тізбекшенің ұзындығына сай L сызықтық күрделіліктің графигі жасалады. Сызықтық күрделілік профилі тестісі арқылы тексерілген шифрмәтінің нәтижесі 3.7-суретте көрсетілген.

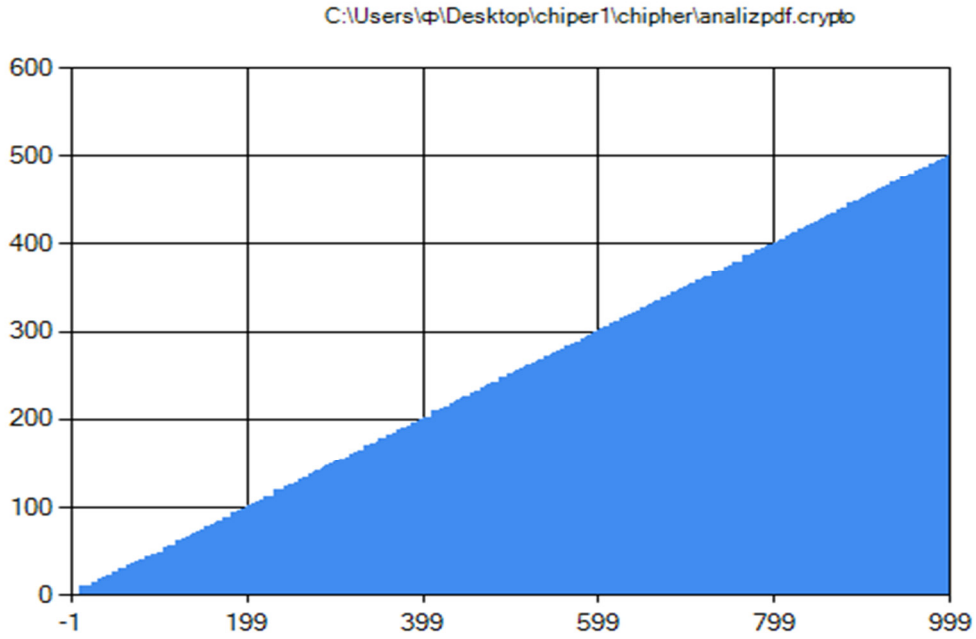


Сурет 3.7 - Сызықтық күрделілік профилі тестісі.

«Графикалық спектрлік тест» тестісі. Фурье түрлендіруінің шығындарының биіктігін талдау негізінде зерттелетін тізбектегі 0 және 1 үлестірудің біркелкілігін тексеруге мүмкіндік береді. оны $x_1x_2x_3 \dots x_n$ тізбегіне түрлендіреміз, мұндағы $x_i = 2t_i - 1$ (т. е. 1→1, 0→-1). Енді қолданылады $x_1x_2 \dots x_n$ тізбегін дискретті Фурье түрлендіруне қолданамыз және гармоника тізбегін алымыз:

$$S_j = \sum_{k=1}^{n} x_k \left[cos\left(\frac{2\pi j}{n}(k-1)\right) + i \cdot sin\left(\frac{2\pi j}{n}(k-1)\right)\right]$$

Табылған S_j күрделі сандардан олардың модулін есептеп, графигін саламыз. Графикалық спектрлік тестісі арқылы қарастырып отырған .pdf файлынан алынған шифрмәтіннің нәтижесі 3.8-суретте көрсетілген.



Сурет 3.8 - Графикалық спектрлік тестісі

Статистикалық қасиеттерді анықтау үшін келесі бағалау тесттер қолданылды:

- 0 мен 1-ге тексеру;
- Тіркеспеген серияларды тексеру;
- Символдық тексеру;
- Аралықтарды тексеру;
- Комбинацияларды тексеру;
- Купон жинаушының тесті;
- Орын ауыстыруды тексеру;
- Корреляцияны тексеру.

Хи-квадрат критерийін (χ^2 – квадрат) k санатына бөлуге болады. n тәуелсіз сынақтар өткізіледі. Сынақ нәтижелері олардың саны көп болсын. p_s -

сынақ нәтижесінің s санатына ену ықтималдығы, ал Y_s -s санатына нақты кіретін сынақтардың саны.

Статистиканы қалыптастырамыз:

$$\chi^2(obs) = \sum_{s=1}^{k} \frac{(Y_s - np_s)^2}{np_s}.$$

Алынған нәтижені бағалау үшін χ^2 үлестірілу кестелері қолданылады. Бұл кестелердің жолдарында *v* еркіндік дәрежелері, ал бағандарда *p* ықтималдығы орналасқан. Егер кестеде *v* жолында және *p* бағанында *x* саны бар болса, онда бұл χ^2 (obs) мәні *p* ықтималдылығымен *x*-тен үлкен болады дегенді білдіреді.

«0 ме 1-ге тексеру» тестісі. 0 және 1-дің үлестірімнің біркелкілігін бағалауға мүмкіндік береді. пайда болу саны v_i есептеледі және статистика шығарылады:

$$\chi^2(obs) = \sum_{i=0}^{1} \frac{\left(v_i - \frac{n}{2}\right)^2}{\frac{n}{2}}.$$

Алынған нәтиже 1-ге тең еркіндік дәрежелері бар χ^2 критерийінің көмегімен талданады.

«Тіркеспеген сериялардың тексеру» тестісі. $t_1 t_2 t_3 \dots t_n$ – тіркеспеген сериялардың екілік тізбегі болсын, *n* ұзындықтағы тіркеспеген сериялар ұзындығын талдай отырып, тізбектің кездейсоқтығын зерттейміз. Әрбір *k* санның ұзындығымен әр түрлі тіркеспейтін $v_{i_1 \dots i_k}$ сериялардың пайда болу санын есептейміз (қосымша биттер алынып тасталады) және статистика есептеледі:

$$\chi^2(obs) = \sum_{v_{i_1 \dots i_k}} \frac{\left(v_{i_1 \dots i_k} - \left[\frac{n}{k}\right] \cdot \frac{1}{2^k}\right)^2}{\left[\frac{n}{k}\right] \cdot \frac{1}{2^k}}.$$

Алынған нәтиже 2^k-*1* еркіндік дәрежесінің санын есепке ала отырып, χ^2 критерийінің көмегімен талданады.

«Символдық тексеру» тестісі. Әр таңбаның пайда болу жиілігін талдау негізінде зерттелетін тізбектегі таңбалардың біркелкі таралуы тексеріледі. $\underline{t_1 t_2 t_3 \dots t_n}$ – *n* – ұзындықтағы k-разрядтық сандарының тізбегі. $v_i, i = \overline{0, (2^k - 1)}$ неше рет кездесетінін есептейміз.

v_i мәні мына аралықта жатуы тиіс

$$\left[\frac{n - 2{,}58 \cdot \sqrt{n(2^k - 1)}}{2^k};\ \frac{n + 2{,}58 \cdot \sqrt{n(2^k - 1)}}{2^k}\right].$$

немесе

$$\chi^2(obs) = \sum_{i=0}^{2^k-1} \frac{\left(v_i - \frac{n}{2^k}\right)^2}{\frac{n}{2^k}}.$$

Алынған нәтиже *2^k-1* еркіндік дәрежесінің санын есепке ала отырып, χ^2 критерийінің көмегімен талданады.

«Аралықты тексеру» тестісі. Барлық элементтері белгілі бір сандық аралыққа жататын ішкі тізбектің ұзындығын талдай отырып, зерттелетін тізбектегі таңбалардың біркелкі таралуын тексереді. $t_1 t_2 t_3 \dots t_n$ тізбегі *n* ұзындықтағы k-разрядтық сандарының тізбегі болсын. α және β – $0 \le \alpha < \beta \le 2^k - 1$ теңсіздігін орындайтын екі бүтін сан болсын. [α; β] аралығындағы сандар арасындағы интервалдардың ұзындығы есептеледі. Осыдан кейін $v_i, i = \overline{0,m}$ интервалдар саны анықталады, мұндағы *0, 1, 2, …, m* үшін статистика есептеледі:

$$\chi^2(obs) = \sum_{i=0}^{m} \frac{\left(v_i - \eta \frac{\beta-\alpha}{2^k}\left(1 - \frac{\beta-\alpha}{2^k}\right)^i\right)^2}{\eta \frac{\beta-\alpha}{2^k}\left(1 - \frac{\beta-\alpha}{2^k}\right)^i},$$

мұндағы: $\eta = \sum_{i=0}^{m} v_i$ – интервалдардың жалпы саны.

Алынған нәтиже еркіндік дәрежелерінің саны *m*-ге тең χ^2 критерийінің көмегімен талданады.

«Комбинацияларды тексеру» тестісі. Зерттелетін ішкі тізбекшелердегі әртүрлі таңбалардың таралуын талдай отырып, қарастырылатын тізбектегі таңбалардың біркелкілігін тексереді. $t_1 t_2 t_3 \dots t_n$ тізбегі *n* ұзындықтағы k-разрядтық сандарының тізбегі болсын. Біз осы тізбекті әрқайсысының ұзындығы *m* болатын тізбекшелерге бөлеміз (қосымша биттер алынып тасталады). i санын қамтитын $v_i, i = \overline{1,r}$ ішкі тізбекшелерінің санын санаймыз, осыдан соң статистика есептелінеді:

$$\chi^2(obs) = \sum_{i=1}^{r} \frac{\left[v_i - \left[\frac{n}{k}\right] \cdot p_i\right]^2}{\left[\frac{n}{k}\right] \cdot p_i},$$

мұндағы $p_i = \frac{d(d-1)\dots(d-i+1)}{d^m} \left\{ {m \atop i} \right\}$, $d = 2^k - 1$, $\left\{ {m \atop i} \right\}$- Стирлинг саны.

Стирлинг саны мына түрде есептелінеді:

$$\left\{ {k \atop n} \right\} = \begin{cases} 1, & \text{егер } k = n \text{ немесе } n = 1 \text{ және } k > 0; \\ 0, & \text{егер } n = 0 \text{ немесе } k < n; \\ n \cdot \left\{ {k-1 \atop n} \right\} + \left\{ {k-1 \atop n-1} \right\}, & \text{қалған жағдайларда.} \end{cases}$$

Алынған нәтиже *r-1* еркіндік дәрежесінің санын есепке ала отырып, χ^2 критерийінің көмегімен талданады.

«Купон жинаушының тесті» тесті. Әр түрлі сандар комбинацияларын талдай отырып, зерттелетін тізбектегі таңбалардың біркелкі таралуын тексереді. $t_1t_2t_3 \dots t_n$ тізбегі n ұзындықтағы k-разрядтық сандарының тізбегі болсын. 0 ден $2^k - 1$-ге дейігі сандар жиынтығын қамтитын $i, i = \overline{2^k, r}$ ұзындықтағы v_iішкі тізбекшелерінің саны есептеліп шығады. Содан кейін статистика саналады:

$$\chi^2(obs) = \sum_{i=d}^{r} \frac{[v_i - \eta \cdot p_i]^2}{\eta \cdot p_i},$$

мұндағы $p_i = \frac{d!}{d^i}\begin{Bmatrix} i-1 \\ d-1 \end{Bmatrix}$, $i = \overline{2^k, r}$, $p_r = \frac{d!}{d^{r-1}}\begin{Bmatrix} r-1 \\ d \end{Bmatrix}$, $\eta = \sum_{i=d}^{r} v_i$, $d = 2^k - 1$.

Алынған нәтиже $r - 2^k + 1$ еркіндік дәрежесінің санын есепке ала отырып, χ^2 критерийінің көмегімен талданады.

«Орын ауыстыруды тексеру» тестісі. Ішкі тізбекшедегі сандардың өзара орналасуын талдай отырып, зерттелетін тізбектегі таңбалардың біркелкі таралуын тексереді.

$t_1t_2t_3 \dots t_n$ тізбегі n ұзындықтағы m-разрядтық сандарының тізбегі болсын. Біз осы тізбекті әрқайсысының ұзындығы k болатын тізбекшелерге бөлеміз (қосымша биттер алынып тасталады). В каждой подпоследовательности возможно k! вариантов относительного расположения чисел. Әрбір ішкі тізбекшеде сандардың орналасуына қатысты *k!* санындағыдай мүмкін болатын жағдай бар. v_i, $i = \overline{1, k}!$ неше рет кездескені саналады және оның статистикасы шығарылады:

$$\chi^2(obs) = \sum_{i=1}^{r!} \frac{\left[v_i - \left[\frac{n}{k}\right] \cdot \frac{1}{k!}\right]^2}{\left[\frac{n}{k}\right] \cdot \frac{1}{k!}}.$$

Алынған нәтиже *k!-1* еркіндік дәрежесінің санын есепке ала отырып, χ^2 критерийінің көмегімен талданады.

«Корреляцияны тексеру» тестісі. Тізбек элементтерінің өзара тәуелділігін тексереді. $t_1t_2t_3 \dots t_n$ тізбегі n ұзындықтағы k-разрядтық сандарының тізбегі болсын. Статистика төмендегідей есептелінеді:

$$\mathrm{C}_j = \frac{n\left(t_0t_j + t_1t_{(1+j)modn} + \cdots + t_{n-1}t_{(n-1+j)modn}\right) - (t_0 + t_1 + \cdots + t_{n-1})^2}{n(t_0^2 + t_1^2 + \cdots + t_{n-1}^2)(t_0 + t_1 + \cdots + t_{n-1})^2}$$

Кез-келген j үшін C_j мәні төмендегі интервалда жатуы тиіс:

$$[\mu_n - 2{,}43\sigma_n;\ \mu_n + 2{,}43\sigma_n],$$

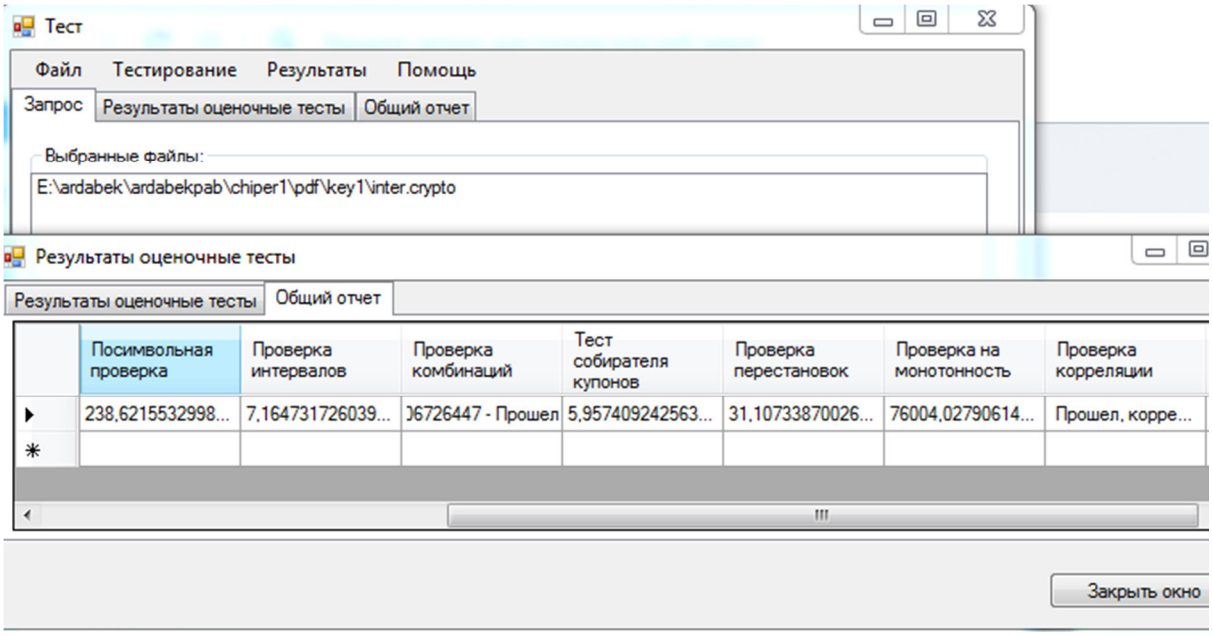
мұндағы:

$$\mu_n = -\frac{1}{n-1}, \sigma_n^2 = \frac{n^2}{(n-1)^2(n-2)}.$$

Статистикалық қасиеттерді анықтау үшін бағалау тесттер арқылы қарастырып отырған .pdf файлынан алынған шифрмәтіннің нәтижесі 3.1-кестеде көрсетілген. Бағалау тесттерінің нәтижесі бізге нақты өтетінін немесе өтпейтіндігін көрсетеді. Ұсынылған нәтиже статистикалық қасиеттерді анықтау үшін, Ақпараттық және есептеуіш технологиялар институтының, Ақпараттық қауіпсіздік зертханасында жасалынған графикалық және бағалау тесттеріне құрылған бағдарламаның (3.9-сурет) көмегімен жүзеге асырылды.

Кесте 3.1 - Бағалау тесттерінің нәтижесі

0 мен 1-ге тексеру	**Тіркеспеген сериялады тексеру**	**Символдық тексеру**	**Аралықтарды тексеру**
1,79116108583207 – Өтті	13,4034923968421 - Өтті	238,621553299887 - Өтті	7,16473172603963 - Өтті
Комбинацияларды тексеру	**Купон жинаушының тесті**	**Орын ауыстыруды тексеру**	**Корреляцияны тексеру**
0,823912306726447 - Өтті	5,95740924256364 - Өтті	31,1073387002683 - Өтті	Өтті, корреляция жоқ



Сурет 3.9 - Графикалық және бағалау тесттерін зерттеу бағдарламасы

Қарастырылған шифрлау алгоритмі үшін алынған шифрланған мәтіндерінің статистикалық қасиеттерін зерттеу процесі келесі дәйекті орындалатын процедураларды қамтиды:

- ұсынылған модельдер бойынша ашық мәтіндерді шифрлау;
- алынған шифр мәтіндер үшін статистикалық тесттер жиынтығын жүргізу;
- шифрмәтіндерін статистикалық тексеру нәтижелерін зерттеу;
- ұсынылған модельдер үшін алынған шифрмәтіндерінің қасиеттері туралы шешім қабылдау.

Ұсынылған шифрлау алгоритмін қолдана отырып алынған шифрмәтіндердің статистикалық қасиеттерін толық тексеру үшін:

- көлемдері және кеңеюі әр түрлі 10 файл (3.2-кесте);
- жұмыс негіздерінің саны әр түрлі 12 кілт (K1, K2, ..., K12) арқылы компьютерлік эксперименттер жүргізіліп нәтижесінде 120 шифрмәтін алынды.

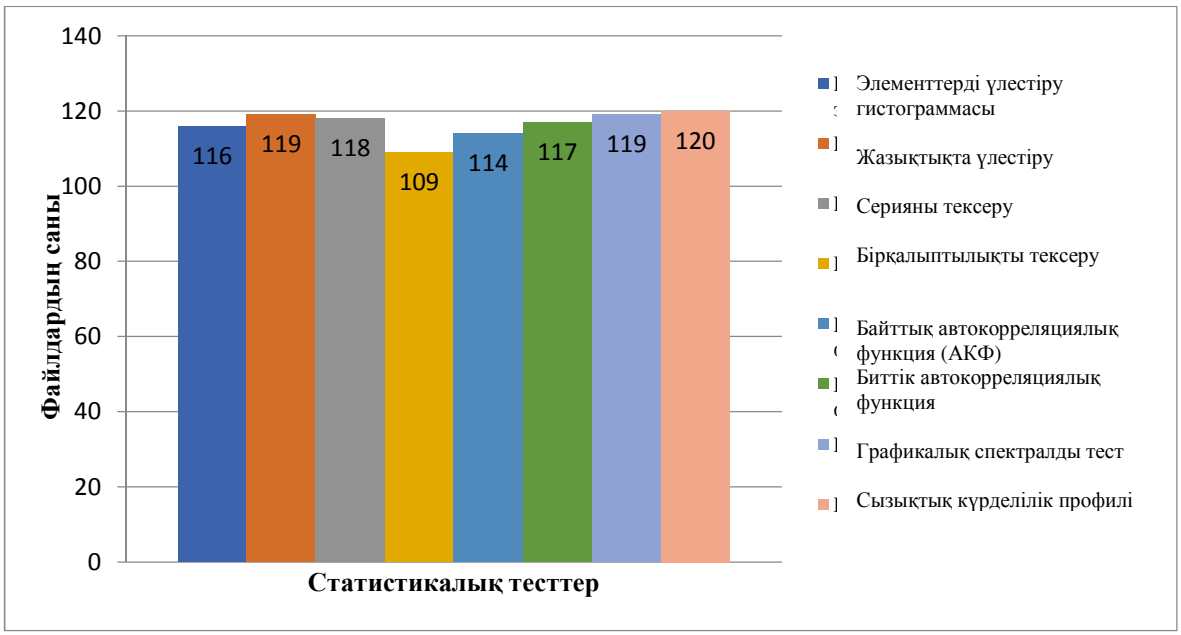
Алынған 120 шифр мәтіннің графикалық тесттер (3.10-сурет) және бағалау тесттері (3.11-сурет) бойынша талдау нәтижелері өткен файлдардың саны көрсетілген.

Кесте 3.2 - Шифрлау алгоритмін тестілеу кезінде қолданылған файлдар

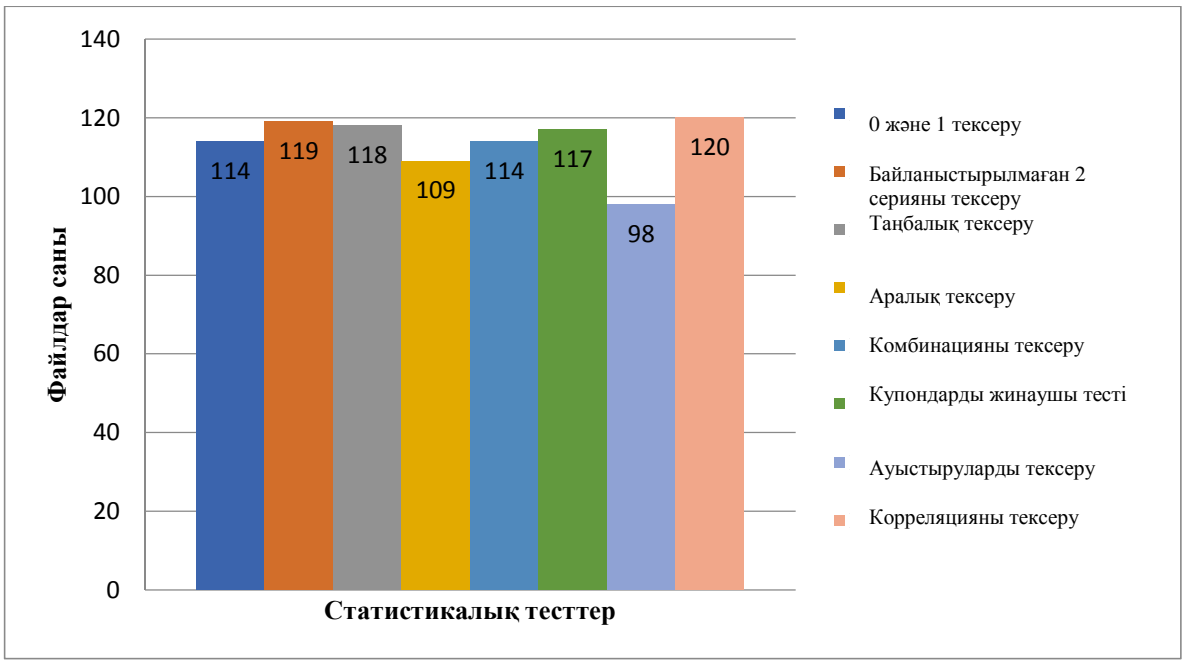
Файл нөмірі	**Файл түрі**	**Сипаттамасы**	**Өлшемі**
1 файл	docx	Microsoft Word құжат	231 КБ
2 файл	xls	Электрондық кесте Microsoft Excel 97-2003	20,0 КБ
3 файл	pptx	Таныстыру Microsoft PowerPoint	2,95 МБ
4 файл	pdf	PDF құжат	7,3 МБ
5 файл	rar	Мұрағат RAR	2,23 МБ
6 файл	zip	Мұрағат файлы ZIP	339 КБ
7 файл	jpg	Joint Photographic Experts Group	856 КБ
8 файл	png	Браузер каталогы Corel PaintShop Photo Pro	1,1МБ
9 файл	txt	Мәтіндік файл файл	78,1 КБ
10 файл	gif	Graphics Interchange Format	548 КБ

Графикалық тестілеудің ұсынылған алгоритмі бойынша жүргізілген зерттеулер шифрленген мәтіндерді тестілеу жақсы нәтиже беретіндігін көрсетті.

Графикалық және бағалау тестілеріне сәйкес зерттелген шифр мәтіндерінің сапасын бағалаудан шифрланған файлдардың статистикалық қауіпсіздігі бар екендігі анықталды. Қарастырылған шифрлау алгоритмі файл түріне байланысты әр түрлі тесттер бойынша әр түрлі нәтижелерді көрсететіні анықталды [52,59,60].



Сурет 3.10 - Графикалық тесттердің нәтижелері



Сурет 3.11 - Бағалау тесттері бойынша нәтижелер

3.3 EMCipher алгоритмін «биттік шашырау» критериі бойынша тексеру нәтижелері

Шифрлау алгоритмдерін әзірлеу барысында оларды әртүрлі криптошабуылдардың түрлеріне төтеп бере алатындай сенімділігіне талдау жүргізу міндетті болып табылады. Қазіргі уақытта ең көп таралған стандартты әдістердің бірі сызықтық және дифференциалды криптоталдау негізіндегі шабуылдар болып табылады.

Дифференциалды талдаудың мағынасы – шығыс биттердің айырымдарының өзгерістерінің кіріс биттердің өзгерістеріне тәуелділігін әр раундтарда қадағалап отыру.

Шифрлау алгоритмінің беріктігін дифференциалды криптоталдаудан қамтамасыз етудің негізгі шарттарның бірі негізгі түрлендіруде «биттік шашырау» әсерінің болуы.

Биттік шашыраудың әсері бастапқы мәтінге (немесе кілтке) кішкентай өзгерістер енгізілсе онда шифрмәтінге айтарлықтай өзгерістер әкелуі мүмкін дегенді білдіреді. Шифрлау алгоритмінің жақсы диффузиялық қасиеттерінің болуы биттік шашырау әсеріне тікелей қатысты болады.

Биттік шашырау әсерінен келесі критериілерді қарастыруға болады:

- биттік шашырау - ашық мәтіннің кезкелген биттін өзгерткен кезде шифрмәтін биттерінің жартысына жуығы өзгеріске ұшырауы қажет;

- қатаң биттік шашырау - ашық мәтіннің кезкелген биттін өзгерткен кезде шифрмәтіннің әрбір биттерінің өзгеру 1/2 ықтималдықта өзгерісті талап етеді.

Егер алгоритмде қажетті деңгейде биттік шашырау әсері болмаса, онда криптоталдаушы нәтижені негізге ала отырып ашықмәтінге шабул жасай алады. Түрлендіру кезінде биттік шашыраудың әсер ету дәрежесін сипаттау үшін биттік шашырау параметрі анықталады. Биттік шашырау критериі үшін биттік шашырау параметрінің мәні келесі формула бойынша анықталады:

$$p = |2k_i - 1| \tag{3.1}$$

мұндағы і – ашықмәтіндегі өзгертілген биттің нөмері, k_i - ашықмәтіннің і-ші биттін өзгерткен кезде өзгертілген ашықмәтінмен шифрмәтінді салыстырғанда өзгерген шифрмәтін биттерінің жартысына жуығы өзгеру ықтималдығы. [61,62].

Жоғарыда көрсетілген критерилерге сайкес жасалынған алгоритмнің биттік шашырауы эксперименталды түрде тексерілді. Тексеру үшін 128 бит ұзындықтағы кездейсоқ алынған ашықмәтін таңдап алынды. Алгоритмнің бесінші, оныншы және оналтыншы раундтағы биттік шашырау нәтижелері 3.3, 3.4 және 3.5 кестелерде ал қалған раундтағы нәтижелер Б қосымшада (кесте Қ.1- Қ.13) көрсетілген.

Кесте 3.3 - EMCipher алгоритмнің 5 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,42	17	0,44	33	0,52	49	0,47	65	0,38	81	0,46	97	0,49	113	0,23
2	0,49	18	0,56	34	0,49	50	0,50	66	0,38	82	0,47	98	0,45	114	0,52
3	0,41	19	0,43	35	0,46	51	0,54	67	0,46	83	0,40	99	0,32	115	0,42
4	0,53	20	0,58	36	0,45	52	0,52	68	0,42	84	0,45	100	0,46	116	0,51
5	0,52	21	0,45	37	0,45	53	0,48	69	0,41	85	0,47	101	0,40	117	0,44
6	0,45	22	0,52	38	0,60	54	0,43	70	0,27	86	0,30	102	0,50	118	0,30
7	0,49	23	0,44	39	0,50	55	0,56	71	0,34	87	0,38	103	0,53	119	0,47
8	0,58	24	0,41	40	0,45	56	0,34	72	0,37	88	0,37	104	0,34	120	0,40
9	0,51	25	0,55	41	0,41	57	0,38	73	0,52	89	0,38	105	0,43	121	0,34
10	0,50	26	0,40	42	0,57	58	0,50	74	0,45	90	0,38	106	0,34	122	0,35
11	0,38	27	0,40	43	0,53	59	0,52	75	0,55	91	0,40	107	0,52	123	0,38
12	0,52	28	0,43	44	0,60	60	0,41	76	0,52	92	0,34	108	0,46	124	0,36
13	0,48	29	0,39	45	0,46	61	0,45	77	0,39	93	0,37	109	0,48	125	0,30
14	0,52	30	0,37	46	0,48	62	0,34	78	0,51	94	0,37	110	0,45	126	0,38
15	0,46	31	0,34	47	0,53	63	0,28	79	0,40	95	0,32	111	0,48	127	0,30
16	0,59	32	0,45	48	0,49	64	0,41	80	0,54	96	0,34	112	0,41	128	0,41

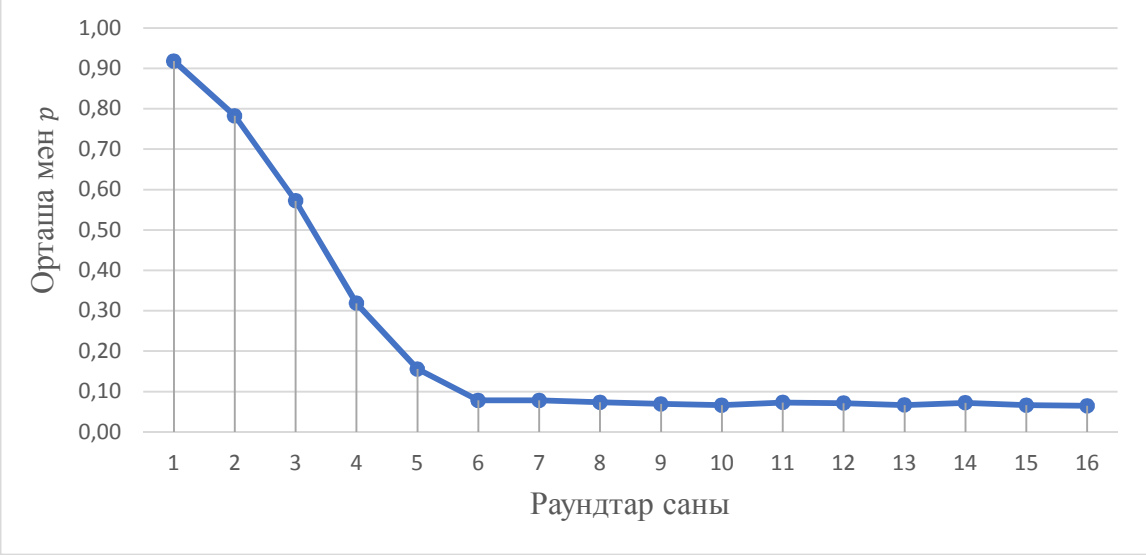
Кесте 3.4 - EMCipher алгоритмнің 10 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,42	33	0,56	49	0,55	65	0,47	81	0,54	97	0,52	113	0,59
2	0,53	18	0,48	34	0,41	50	0,52	66	0,55	82	0,50	98	0,43	114	0,45
3	0,48	19	0,51	35	0,50	51	0,55	67	0,54	83	0,50	99	0,49	115	0,57
4	0,45	20	0,49	36	0,45	52	0,56	68	0,43	84	0,59	100	0,54	116	0,50
5	0,48	21	0,50	37	0,53	53	0,50	69	0,49	85	0,48	101	0,55	117	0,42
6	0,45	22	0,49	38	0,52	54	0,51	70	0,50	86	0,50	102	0,48	118	0,48
7	0,52	23	0,46	39	0,48	55	0,50	71	0,51	87	0,48	103	0,45	119	0,47
8	0,48	24	0,43	40	0,48	56	0,55	72	0,43	88	0,56	104	0,42	120	0,48
9	0,52	25	0,51	41	0,53	57	0,54	73	0,49	89	0,45	105	0,52	121	0,52
10	0,45	26	0,52	42	0,45	58	0,53	74	0,45	90	0,55	106	0,45	122	0,45
11	0,54	27	0,47	43	0,54	59	0,48	75	0,48	91	0,52	107	0,46	123	0,52
12	0,48	28	0,46	44	0,48	60	0,52	76	0,51	92	0,53	108	0,46	124	0,48
13	0,43	29	0,48	45	0,50	61	0,52	77	0,51	93	0,49	109	0,48	125	0,54
14	0,58	30	0,52	46	0,52	62	0,48	78	0,43	94	0,51	110	0,48	126	0,45
15	0,48	31	0,53	47	0,52	63	0,42	79	0,57	95	0,60	111	0,55	127	0,44
16	0,51	32	0,47	48	0,52	64	0,48	80	0,52	96	0,49	112	0,47	128	0,52

Кесте 3.5 - EMCipher алгоритмнің оналтыншы раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,48	33	0,49	49	0,52	65	0,56	81	0,56	97	0,51	113	0,52
2	0,47	18	0,48	34	0,48	50	0,54	66	0,56	82	0,50	98	0,49	114	0,51
3	0,56	19	0,45	35	0,50	51	0,47	67	0,48	83	0,54	99	0,47	115	0,54
4	0,57	20	0,47	36	0,51	52	0,49	68	0,52	84	0,48	100	0,52	116	0,51
5	0,40	21	0,47	37	0,51	53	0,47	69	0,50	85	0,55	101	0,58	117	0,44
6	0,49	22	0,58	38	0,50	54	0,45	70	0,52	86	0,50	102	0,51	118	0,47
7	0,50	23	0,53	39	0,44	55	0,53	71	0,48	87	0,49	103	0,48	119	0,56
8	0,47	24	0,50	40	0,51	56	0,53	72	0,54	88	0,47	104	0,51	120	0,52
9	0,50	25	0,38	41	0,49	57	0,52	73	0,50	89	0,41	105	0,47	121	0,46
10	0,52	26	0,52	42	0,52	58	0,51	74	0,55	90	0,45	106	0,51	122	0,50
11	0,47	27	0,48	43	0,55	59	0,41	75	0,52	91	0,48	107	0,49	123	0,59
12	0,52	28	0,64	44	0,54	60	0,55	76	0,50	92	0,53	108	0,48	124	0,54
13	0,46	29	0,54	45	0,55	61	0,54	77	0,51	93	0,53	109	0,57	125	0,52
14	0,51	30	0,52	46	0,51	62	0,49	78	0,46	94	0,48	110	0,45	126	0,45
15	0,48	31	0,52	47	0,54	63	0,55	79	0,56	95	0,60	111	0,52	127	0,49
16	0,48	32	0,43	48	0,47	64	0,47	80	0,43	96	0,50	112	0,44	128	0,55

Алгоритмнің биттік шашырау p орташа мәні 3.12–суретте көрсетілген. Көріп отырғанымыздай жасалынған алгоритмнің биттік шашырау параметрінің орташа мәні 4 раундттан бастап жақсарғаны байқауға болады. Биттік шашырау параметрлерінің өзгеру диапазоны 0-ден 1-ге дейін. Биттік шашырау параметрінің мәні неғұрлым төмен болса, түрлендіру кезінде биттік шашыраудың әсері соғұрлым күшті болады.



Сурет 3.12 - Биттік шашырау критерийлері параметрінің орташа мәні

3.4 Құрылған S-блокқа жасалған криптоталдау нәтижелері

Блоктық шифрлау алгоритмдерінің криптографиялық беріктілігі S-блоктардың беріктігіне тікелей байланысты екені белгілі. Сондықтан ұсынылған алгоритмде қолданылатын S-блоктар үшін сызықтық және дифференциалды талдау жүргізілді және нәтижелер басқа алгоритмдердің S-блоктарымен салыстырылды (3.6-кесте) [63,64].

Кесте 3.6 –Сызықты және Дифференциалды криптоталдау нәтижелері

Алгоритмдер		**Төменгі мән**	**Жоғарғы мән**
DES	Сызықты	12	48
	Дифференциалды	0	16
Magma (GOST R34.13-2015)	Сызықты	4	12
	Дифференциалды	0	4
Kuznyechik (GOST R34.13-2015)	Сызықты	100	156
	Дифференциалды	0	8
AES-128	Сызықты	111	145
	Дифференциалды	0	5
EMCipher	Сызықты	100	156
	Дифференциалды	0	8

3.5 EMCipher алгоритміне дифференциалдық криптоталдау

Дифференциалдық криптоталдауды EMCipher алгоритміне жүргізуден бұрын, алдымен алгоритмнің 2.3-суреттегі толық схемасын еске түсіре отырып, оның құрамындағы математикалық амалдарға (функциялар немесе операциялар) дифференциалдық талдауды жеке қарастырайық және оның нәтижелерін ескере отырып әрбір раунд үшін жоғарыдан бағалау арқылы арнайы таңдалған ашық мәтіндер мен шифрмәтіндер жұптарына қатысты кілттерді анықтаудағы дифференциалдық талдау қарастырамыз.

Алгоритмнің құрамындағы амалдарды сызықтық және сызықтық емес ретінде талдайық және олардың әрқайсына жеке талдаулар қарастырылады.

Төмендегі сызықтық функциялар үшін дифференциалдық талдаудың ықтималдылығы 1-ге тең, яғни:

- «XOR» биттік қосу амалы үшін $\Delta y = y_1 \oplus y_2 = x_1 \oplus k \oplus x_2 \oplus k = x_1 \oplus x_2 = \Delta x$, яғни XOR функциясының кірісіне бірдей мәндерді қосқаннан кірісінің айырымдары мен шығысының айырымдары өзгермейді;

- «XOR» биттік қосу амалы үшін $\Delta y = y_1 \oplus y_2 = x_1 \cdot k \oplus x_2 \cdot k = (x_1 \oplus x_2) \cdot k = \Delta x \cdot k$, яғни XOR функциясының кірісіне бірдей мәндерге еселеп артырғаннан кірісінің айырымдары мен шығысының айырымдары сонша мәнге еселеп артады [65,66];

- қадам 3 бойынша солға циклді жылжыту үшін: $\Delta y = y_1 \oplus y_2 = \left((x_{1,1} \oplus k_{1,1}) \cdot 2^3 \oplus (x_{1,2} \oplus k_{1,2}) \cdot 2^{29}\right) mod 2^{32} \oplus \left(\left(x_{2,1} \oplus k_{2,1}\right) \cdot 2^3 \oplus \left(x_{2,2} \oplus k_{2,2}\right) \cdot 2^{29}\right) mod 2^{32} = \left(\left(x_{1,1} \oplus x_{2,1}\right) \cdot 2^3 \oplus \left(x_{1,2} \oplus x_{2,2}\right) \cdot 2^{29}\right) mod 2^{32} = (\Delta x_1 \cdot 2^3 \oplus \Delta x_2 \cdot 2^{29}) mod 2^{32}$ болды;

- қадам 5 бойынша солға циклді жылжыту үшін: $\Delta y = y_1 \oplus y_2 = \left((x_{1,1} \oplus k_{1,1}) \cdot 2^5 \oplus (x_{1,2} \oplus k_{1,2}) \cdot 2^{27}\right) mod 2^{32} \oplus \left(\left(x_{2,1} \oplus k_{2,1}\right) \cdot 2^5 \oplus \left(x_{2,2} \oplus k_{2,2}\right) \cdot 2^{27}\right) mod 2^{32} = \left(\left(x_{1,1} \oplus x_{2,1}\right) \cdot 2^5 \oplus \left(x_{1,2} \oplus x_{2,2}\right) \cdot 2^{27}\right) mod 2^{32} = (\Delta x_1 \cdot 2^5 \oplus \Delta x_2 \cdot 2^{27}) mod 2^{32}$ болды;

Сызықтық емес функциялар үшін дифференциалдық талдаудың ықтималдылығын анықтаймыз.

- ЕМ фукнциясы үшін:

$\Delta\, \beta_i(x) = \beta_{i,1}(x) \oplus \beta_{i,2}(x) = \left(\left(\alpha_{i,1}(x)^{k_i(x)}\right) \oplus \left(\alpha_{i,2}(x)^{k_i(x)}\right)\right) mod P_i(x)$, және айырымдардың кірісі мен шығысының кездесуінің үлкен ауытқулары сәйкесінше келтірілмейтін көпмүшеліктерге қарай сегізінші дәреже үшін 2^{-5}, жетінші дәреже үшін 2^{-4} және бесінші дәреже үшін 2^{-2} болады, мұндағы, $\alpha_{i,j}(x), \beta_{i,j}(x) \in \frac{GF\left(2^{\deg(k_i(x))}\right)}{P_i(x)}, i = \overline{1,4}, j = 1,2$;

- енді «S-box» функциясына талдау қарастырылады. Таңдалған «S-box» алмастыруының талдау нәтижесіндегі айырымдардың кездесуі 0-ден 8-ге дейінгі аралықта болады. Әрбір кіріс айырымдарына сәйкес шығыс айырымдары 8 рет кездесуі ең үлкен ауытқуы болғандықтан, осы $\frac{8}{256} = 0{,}03125$ ықтималдылығын аламыз.

Дифференциалдық талдауды бір раунд үшін қарастырайық. Мысалға жоғарыдан бағалауға қолайырақ болу үшін таңдалатын ашық мәтіндердің жұбының айырымдарын 0x0000000000000000000000000000001B-ге тең болатындай алайық. Сонда 2.3-суреттегі сұлба бойынша бір раундтан кейінгі нәтижелердің айырымына 0x0000003C00000000000000000000 сәйкес кілттің анықталатын бөлігінің жоғары ықтималдылығы «S-box» алмастыруына қатысты болғандақтан $0{,}03125 = 2^{-5}$ тең болады.

Екінші раундта айырымның нөлден өзгеше мәні төртінші байт болғандықтан, ол ЕМ фукнциясына қатысты және кірісі 0x0000003C үшін $0{,}01887 \approx 2^{-4}$ болса, онда екінші раундтан кейін белгілі аумақта кілті анықтау

ықтималдылығы 2^{-9} және айырымның мәні: 00000000 $0000x_{13}x_{14}00$ $0000x_{21}x_{22}x_{23}0$ 00000000 болады.

Үшінші раундта айырымның нөлден өзгеше мәні жетінші, он бірінші және он екінші байт болғандықтан, оған ЕМ фукнциясыда және «S-box» алмастыруыда қатысты, бірақ екеуінде алу міндетті емес, тек қайсысының ауытқуы жоғары, соңы таңдаймыз. Біздің жағдайда ЕМ фукнциясының ауытқуы жоғары болғандықтан осы байттың соңынан жүреміз. Сондықтан үшінші раундтан кейін анықтау ықтималдылығы 2^{-9} болады.

Осы айтылғандарды ескере отырып, әрбір раундтағы шығысының ықтималдылығын ғана санайық, өйткені бізге бағалау үшін оның әрбір раундтағы статистикасын 2.3-суреттегі сұлбада көргендей, статистикалық өзгерістерін әрбір функциясының ықтималдылығының үлесіне сәйкес есептейміз. Сондықтан, төртінші раундтың кіріс айырымы 2^{-9} тең, ал шығыс айырымы үшін «S-box»-тан және XOR-дан өткеннен кейін 2^{-13} болады. Дәл осылай тәртіппен төмендегі 3.7-кестедегі мәндерді ала аламыз. Көріп отырғандай ұсынылған алгоритм дифференциальды талдауға беріктігі анықталды.

Кесте 3.7 – Әрбір раундтағы кіріс және шығыс айырымдарының кілтті анықтаудағы бағалануы

Раундтар №		**Айырым ықтималдығы**	**Раундтар №**		**Айырым ықтималдығы**
1	кіріс	1	6	Кіріс	2^{-21}
	шығыс	2^{-5}		Шығыс	2^{-38}
2	Кіріс	2^{-5}	7	Кіріс	2^{-38}
	Шығыс	2^{-9}		Шығыс	2^{-78}
3	Кіріс	2^{-9}	8	Кіріс	2^{-78}
	Шығыс	2^{-9}		шығыс	2^{-128}
4	Кіріс	2^{-9}	9	Кіріс	2^{-128}
	Шығыс	2^{-13}		Шығыс	2^{-215}
5	Кіріс	2^{-13}	10	Кіріс	2^{-215}
	Шығыс	2^{-21}		Шығыс	2^{-361}

Берілген кілт пен ашық мәтін жұптарының айырымдарына қатысты сәйкесінше шифрмәтін жұптарының айырымдарының биттік шашырауының қатынасын мысалдар арқылы бақылайық.

1-мысал. Кілт 288 бит немесе 36 байт (он алтылық санау жүйесінде берілген):
047707D9EA56EDBD9123F72ABF3C98A1404AB70154F9DD33D700F3F32946C9869185297C

Кілт үш бөлікке бөлінеді:

- Алғашқы 128 бит ашық мәтінге «XOR» операциясы бойынша (разрядтық түрде) қосылады;

- Соңғы 32 бит ЕМ операциясына қатысты ПЕПСЖ (позициялы емес полиномды санау жүйесі) раундтық кілттердің қызметтің атқарады;

- 129 биттен 256 битке дейінгі аралықтағы биттер 16 раундтан кейінгі шыққан блоктің нәтижесіне «XOR» операциясы бойынша қосылып шифрмәтінді береді.

Төмендегі мысалдарда әртүрлі ашық мәтіннің жұптары және оған сәйкес шифрмәтін жұптарының айырымы көрсетілген:

1): 1-ші ашық мәтін: 81754b8c671be306adee86fc52174dcd

2-ші ашық мәтін: 81754b8c671be306adee86fc52174dcc

Бірінші мен екінші ашық мәтіндердің айырымы төмендегідей болады:

00000000000000000000000000000001

Осыған сәйкес әрбір раундтан кейінгі нәтижелердің және шифрмәтін жұптарының айырымын қарастырайық:

1-раунд:

00000053000000000000000000000000

2-раунд:

00000000000770000001dc000000000

3-раунд:

00054000000048ce0000000000015000

4-раунд:

002dee4009de00000277800000085dd0

5-раунд:

282f747769e9b600012627008a100005

6-раунд:

83c028753e752333de62444c948dca07

7-раунд:

0525ea6757bb07bc84e527e1d88ecf8b

8-раунд:

f2278ca0a16ea99fa9ff50994d427df4

9-раунд:

095074e19067bf36844758036fa9323e

10-раунд:

99a206d891786bb18bab410d2b1d5b97

11-раунд:

f444a3c10e6187ffa3bc04ea27f5fef7

12-раунд:

d438237c28fee5c096d60554917b96aa

13-раунд:

edb451ae4cc83cf1c9642845de232d04

14-раунд:

c855fab7a75a8d810c1bb2a9616b59ef

15-раунд:

afa4a76b673348f0f81d34497fe5f7b1

16-раунд:

666ca3f30afd3974f9cee31c02af8b2d

шифрмәтін:
666ca3f30afd3974f9cee31c02af8b2d
№№2-12 мысалдарда жазылу реті осы ретпен анықталған.
2):
904b9e1bd6eaa64db9a9c168a5e5f92d
904b9e1bd6eaa64db9a9c168a5e5f92c
2-мысалда берілген ашық мәтіндердің айырымы:
00000000000000000000000000000001
=>:
ShiferText:
435feefbd4452ec612fd21f0cd30e1a7
3):
81754b8c671be306adee86fc52174dcd
81754b8c671be306adee86fc52174ccd
=
00000000000000000000000000000100
=>:
ShiferText:
ef776d88aa59938fc296e078a8226370
4):
81754b8c671be306adee86fc52174dcd
81754b8c671be306adee86fc52164dcd
=
00000000000000000000000000010000
=>:
ShiferText:
398e122e1568d70a04ed4dce690ed8c9
5):
81754b8c671be306adee86fc52174dcd
81754b8c671be306adee86fc53174dcd
=
00000000000000000000000001000000
=>:
ShiferText:
06c4c8a0e0ad9a96eda8dcf96cff7c05
6):
81754b8c671be306adee86fc52174dcd
81754b8c671be306adee86fd52174dcd
=
00000000000000000000000100000000
=>:
ShiferText:
df94357db2c7f26f26af009713c40ef9
7):
81754b8c671be306adee86fc52174dcd

81754b8c671be306adef86fc52174dcd
=
00000000000000000001000000000000
=>:
ShiferText:
8c090e504ba40fbd447fa87ab580c5c7
8):
81754b8c671be306adee86fc52174dcd
81754b8c671be307adee86fc52174dcd
=
00000000000000010000000000000000
=>:
ShiferText:
938081a2a82b0126eee62714bdc47cf6
9):
81754b8c671be306adee86fc52174dcd
81754b8c671ae306adee86fc52174dcd
=
00000000001000000000000000000000
=>:
ShiferText:
f68a4d52863abb87e78345572a4be0b2
10):
81754b8c671be306adee86fc52174dcd
81754b8d671be306adee86fc52174dcd
=
00000010000000000000000000000000
=>:
ShiferText:
42e0511273c8ce72bb09e85a76969b04
11):
81754b8c671be306adee86fc52174dcd
81744b8c671be306adee86fc52174dcd
=
00010000000000000000000000000000
=>:
ShiferText:
51e4c13f5a1d8d3112b184d3244bd4ea

3.6 Бөлім бойынша қорытынды

Құрылған EMCipher алгоритмінің криптоберіктілігін тексеру үшін үшінші бөлімде алгоритмнің бағдарламасы жасалынды және алгоритмге криптоталдаулар жүргізілді. Алгоритмнің статистикалық қауіпсіздігі графикалық және бағалау тестері арқылы зерттелініп жақсы нәтижелер алынды. Сонымен қатар алгоритмнің биттік шашырау критериилері бойынша әрбір раунд бойынша зерттелініп, биттік шашыраудың орташа мәндері есептелінді, нәтижесінде биттік шашырау критериилерін қанағаттандыратыны анықталды.

EMCipher алгоритміне қолданылған S-блок алмастыру кестесі қазіргі криптоталдау әдістерінің ең танымал әдістері сызықты және дифференциалды әдістерін бойынша сараптамадан өткізіліп нәтижесі заманауи қолданыстағы шетелдік симметриялық блокты шифрлау алгоритмдермен тең нәтиже көрсететіндігі анықталды. Сонымен қатар алгоритмге дифференциалды криптоталдау жүргізіліп жақсы нәтижелер алынды.

ҚОРТЫНДЫ

Заманауи криптографиялық әдістер, оның ішінде итеративті блоктық шифрлар жылдамдығы жоғары ақпарат тарату желілерінде қауіпсіз ақпарат алмасуды қамтамасыз ететін, сұранысқа ие құралдардың бірі болып табылады. Ақпараттық технологияларды кеңінен қолдану және есептеу қуатының қарқынды дамуы белгілі шифрлардың криптоталдауына қауіп тудырады.

Мәліметтерді криптографиялық қорғау құралдарын құруға бағытталған зерттеулер көбінесе мемлекеттік құпияларға байланысты, сондықтан шетелдік дайын шешімдерді қолдану қауіпсіз емес. Ақпаратты криптографиялық қорғау құралдарын құру, оның ішінде шифрлау алгоритмдерін құру бойынша зерттеулер жүргізу өзекті және қажетті болып табылады.

Диссертатцияда ақпаратты криптографиялық қорғаудың жаңа блоктық шифрлау алгоритімі жасалынды. Онда осы алгоритмнің құрама бөлігі болып табылатын жаңа ЕМ (exponentiation modul) түрлендіру әдісі сипатталған. Ұсынылып отырған ЕМ түрлендіру әдісі Галуа өрісіндегі дәрежеге шығару операциясы негізінде жасалғандықтан ондағы үлкен көлемдегі позициялық санау жүйесіндегі разрядттарды таңдап алған жұмыс негіздерінің қалдығы ретінде өрнектеп кіші көлемдегі разрядтардың алыну және таңдап алған жұмыс негіздерінің индекс кестесін құру. Ол өз кезегінде шифрлау жылдамдағын арттырып, қателерді жылдам тауып жөндеуге мүмкіндік береді. Ұсынылып отырған алгоритмде жаңа S-блок алу әдісі жасалынып, алынған S-блок алмастыру кестесінің криптотұрақтылығы сызықты және дифференциалды криптоталдау әдістерімен тексеріліп жақсы нәтиже алынды және заманауи алгоритмдермен тең нәтиже көрсететіні анықталды. Сонымен қатар раундттық кілттерді жасау алгоритмі де ұсынылып ондағы RNS түрлендіру әдісі сипатталды. Жасалынып отырған алгоритмге биттік шашырау және статистикалық бағалау тесттері және дифференциалды криптоталдау әдістері арқылы криптотұрақтылығы тексеріліп нәтижелері ұсынылды. Ұсынылып отырған нәтижелердің криптографиялық талаптарды қанағаттандыратыны анықталды. Бұл жұмыстың ғылыми және практикалық маңыздылығы алынған нәтижелер Қазақстандағы ақпаратты криптографиялық қорғау құралдарын жасауға және құруға қолдануға болады. Ұсынылған шифрлау алгоритмі тиісті деңгейде ақпараттың қауіпсіздігі мен құпиялылығын қамтамасыз етуге мүмкіндік береді.

ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1. Концепция кибербезопасности («Киберщит Казахстана»): утв. постановлением Правительства Республики Казахстан от 30 июня 2017 года №407.

2. Девянин П.Н., Девянин П.Н., Михальский О.О., Правиков Д.И. Теоретические основы компьютерной безопасности. – М.: Горячая линия, 2000. – 416 с.

3. Авдошин С.М., Савельева А.А. Криптографические методы защиты информационных систем // Известия АИН им. А.М. Прохорова. Бизнес-информатика. – 2006. – С.91-99.

4. Петров В.А., Пискарев А.С., Шеин А.В. Информационная безопасность. Защита информации от несанкционированного доступа в автоматизированных системах. – М.: МИФИ, 1995.

5. Фомичев В.М. Симметричные криптосхемы. Краткий обзор основ криптологии для шифрсистем с открытым ключом. – М.: МИФИ, 1995.

6. Нечаев В.И. Элементы криптографии. Основы теории защиты информации. – М.: Высшая школа, 1999.

7. Зубов А.Ю. Криптографические методы защиты информации. Совершенные шифры. – М.: Гелиос АРВ, 2005.

8. Хоффман Л.Д. Современные методы защиты информации / Под ред. В.А. Герасименко. – М.: Сов. радио, 1980. – 264 с.

9. Брюс Ш. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. – М.: Издательство ТРИУМФ, 2002. – 816 с.: ил.

10. Алферов А.П., Зубов А.Ю., Кузьмин А.С. и др. Основы криптографии. – М.: Гелиос АРВ, 2001. – 122 с.

11. Асамбаев А.Ж. Криптография негіздері. Оқу құралы. – Павлодар, 2012. – 173 бет.

12. Тұрым А.Ш., Мұстафина Б.М. Ақпарат қорғау және қауіпсіздендіру негіздері. – Алматы: Алматы энергетика және байланыс институты, 2002.

13. Бабенко Л.К., Ищукова Е.А. Современные алгоритмы блочного шифрования и методы их анализа. – М.: Гелиос АРВ, 2006. – 376 с.

14. Романец Ю.В., Тимофеев П.А., Шаньгин В.Ф. Защита информации в компьютерных системах и сетях. – М.: РАДИО И СВЯЗЬ, 1999.

15. Алферов А.П., Зубов А.Ю., Кузьмин А.С., Черемушкин А.В. Основы криптографии: Учебное пособие. – М.: Гелиос АРВ, 2002. – 480 с.

16. Рожков А.В., Ниссенбаум О.В. Теоретико-числовые методы в криптографии. – Тюмень, 2007. – 175 с.

17. Гундарь К.Ю., Гундарь А.Ю., Янишевский Д.А., Защита информации в компьютерных системах. – Киев: Корншчук, 2000. –154 с.

18. Диффи У., Хеллман М.Э. Защищенность и имитостойкость: Введение в критпографию. – ТИИЭР, 1976. – Т.67, № 3. – С. 71-109.

19. Домарев В.В. Защита информации и безопасность компьютерных систем. – Киев: Diasoft, 1999. – 480 с.

20. Иванов М.А. Криптографические методы защиты информации в компьютерных системах и сетях. – М.: Кудиц-Образ, 2001. – 368 с.

21. Столлингс В. Криптография и защита сетей: принципы и практика. – М.: «Вильяме», 2002. – 672 с.

22. ГОСТ 28147-89. Защита криптографическая. Алгоритм криптографического преобразования. – М.: Изд-во стандартов, 1996.

23. ISO/IEC9797. Data cryptographic techniques dataintegrity mechanism using a cryptographic check function employing a block cipher algorithm, 1989.

24. M.Bellare P., Rogaway. Optimal symmetric encryption. In Advances in Cryptology // Eurocrypt'94. – 1995. – Vol.950. – P. 92-111.

25. Варфоломеев А.А., Варфоломеев А.А., Жуков А.Е. Блочные криптосистемы. Основные свойства и методы анализа стойкости. – М.: МИФИ, 1998. – 247 с.

26. Смарт Н. Криптография. – М.: Техносфера, 2005. – 528 с.

27. Фергюссон Н., Шнайер Б. Практическая криптография. – М.: «Вильямс», 2005. – 424 с.

28. Диффи У., Хеллмэн М.Э. Защищенность и имитостойкость: введение в криптографию. // ТИИЭР. – 1979 г. – № 3. – Т.67. – С. 71-109.

29. Зензин О.С., Иванов М.А. Стандарт криптографической защиты - AES. Конечные поля. – М.: КУДИЦ-ОБРАЗ, 2002. – 176 с.

30. Молдовян А. А. Криптография: скоростные шифры. – СПб.: БХВ-Петербург, 2002. – 496 с.

31. Акушский И.Я., Юдицкий Д.И. Машинная арифметика в остаточных классах. – Издательство «Советское радио» Москва, 1968. – 438 с.

32. Синьков М.В., Губарени Н.М. Непозиционные представления многомерных числовых систем. – Киев, Наукова думка, 1977. – 149 с.

33. Червяков Н.И., Сахнюк П.А., Шапошников А.В., Макоха А.Н., Червяков Н.И. Нейрокомпьютеры в остаточных классах. – М.: Радиотехника, 2003. – 272 с.

34. Амербаев В.М., Бияшев Р.Г., Нысанбаева С.Е. Применение непозиционных систем счисления при криптографической защите // Изв. Нац. акад. наук Респ. Казахстан. Сер. физ.-мат. – 2005. – № 3. – С. 84–89.

35. Biyashev R. G. Development and investigation of methods of the overall increase in reliability in data exchange systems of distributed ACSs // Doctoral Dissertation in Technical Sciences. – Moscow, 1985 (in Russian).

36. Хомпыш А. Позициялық емес санау жүйесін қолданылуы // «Көліктегі инновациялық технологиялар: білім, ғылым, тәжірибе» атты XLI Халықар. ғыл.-практ. конф. мат. – Қазақстан, Алматы, 2017. – Т.3. – 64-66 б.

37. Капалова Н.А., Хомпыш А. Позициялық емес санау жүйесін қолданып, Эль-Гамаль шифрлау алгоритмінің модификациясын құру // Қ.И. Сәтбаев атындағы Қазақ Ұлттық техникалық зерттеу университетінің, Хабаршысы. – Алматы, 2017. – №4 (122). – 506-510 б.

38. Biyashev R., Nyssanbayeva S., Kapalova N., Haumen A. Modied symmetric block encryption-decryption algorithm based on modular arithmetic // Proceedings of the International Conference on Wireless Communications, Network

Security and Signal Processing (WCNSSP2016). – Chiang Mai, Thailand. – 2016. – P. 263-265.

39. Biyashev R. G., Nyssanbayeva S. E. Algorithm for Creation a Digital Signature with Error Detection and Correction // Cybernetics and Systems Analysis. – 2012. – Vol.48. – No 4. – P. 489-497.

40. Biyashev R., Nyssanbayeva S., Kapalova N. The Key Exchange Algorithm on Basis of Modular Arithmetic // Proceedings of International Conference on Electrical, Control and Automation Engineering (ECAE2013), Hong Kong – Lancaster. – U.S.A.:DEStech Publications, 2013. – P.16-21.

41. Chengqing Li, Yun Zhang, Eric Yong Xie. When an attacker meets a cipher-image in 2018: A year in revi. // Journal of Information Security and Applications. – 2019. – Vol.48. 102361. – http://dx.doi.org/10.1016/j.jisa.2019. 102 361.

42. Ullah A., Jamal S.S., Shah T. A. novel scheme for image encryption using substitution box and chaotic system // Nonlinear Dyn. – 2018. –Vol.91(1). – P. 359-3707. – doi: 10.1007/s11071- 017- 3874- 6.

43. Liu L., Hao S., Lin J., Wang Z., Hu X., Miao S. Image block encryption algorithm based on chaotic maps // IET Signal Process. – 2018. – Vol.12(1). – P. 22-30. – doi:10.1049/iet-spr.2016.0584.

44. Xiong Y., He A., Quan C. Security analysis of a double-image encryption technique based on an asymmetric algorithm // J Opt Soc Am A. – 2018. – Vol.35(2). – P. 320-326. – doi:10.1364/JOSAA.35.000320.

45. Li C., Lin D., Lü J. Cryptanalyzing an image-scrambling encryption algorithm of pixel bits // IEEE Multimedia. – 2017. – Vol.3. – P.64–71. – doi: 10.1109/MMUL.2017.3051512.

46. Li C., Lin D., Lü J., Hao F. Cryptanalyzing an image encryption algorithm based on autoblocking and electrocardiography // IEEE Multimed. – 2018. – Vol.25(4). – P.46–56. – doi: 10.1109/MMUL.2018.2873472.

47. Xiong Y., He A., Quan C. Cryptanalysis of an optical cryptosystem based on phase-truncated fourier transform and nonlinear operations // Opt Commun. – 2018. – Vol 428. –P. 120–130. – doi: 10.1016/j.optcom.2018.07.058.

48. Хомпыш А. Эль-Гамаль шифрлау алгоритмінің мобильдік қосымшасын құру // «Есептеуіш технологиялар және информатиканың заманауи мәселелері» атты ғылыми конференция материалдары. – Алматы, 2017. – 281-284 б.

49. Хомпыш А. Позициялық емес санау жүйесін негізінде құрылған Эль-Гамаль шифрлау алгоритмін мәліметтер алмасу желісінде пайдалану // II Халықаралық «Информатика және қолданбалы математика» ғылыми конференция материалдары. – Алматы, 2017. – 157-161 б.

50. Kapalova N., Haumen A. The model of encryption algorithm based on non-positional polynomial notations and constructed on an SP-network // Open Engineering. – 2018. – Vol.8, Issue 1. – P. 140-146.

51. Biyashev R. G., Smolartz A., Algazy K.T., Khompysh A. Encryption algorithm ‘’Qamal NPNS’’ based on a nonpositional polynomial notation // Journal

of Mathematics, Mechanics and Computer Science, «Хабаршы» ҚазҰУ. – Алматы, 2020. –Vol. №1 (105). – P. 198-207.

52. Kapalova N., Khompysh A., Müslüm A., Algazy K. A block encryption algorithm based on exponentiation transform. // Cogent engineering. – 2020. – Vol.7(178829). – ISSN 2331-1916. – P. 1-12.

53. Чипига А.А., Калмыков И.А., Барильская А.В., Кихтенко О.А. Криптографическая защита данных в информационных технологиях на базе непозиционных полиномиальных систем // Известия ЮФУ. Технические науки. – Таганрог, 2009. – С.210-220.

54. Капалова Н.А., Хомпыш А., Алгазы К. Модуль бойынша дәрежеге шығару негізінде ақпаратты криптографиялық қорғау алгоритмінің модификациясы // М.Тынышбаев атындағы Қазақ көлік және коммуникациялар академиясының, Хабаршысы. – Алматы, 2018. – №4 (107). – 247-253б.

55. Чипига А.А., Калмыков И.А., Хайватов А.Б., Сагдеев А.К. Применение расширенных полей Галуа для повышения информационной скрытности передачи данных // Успехи современного естествознания. – № 5. – 2007. – С.103-105.

56. Чипига А.А., Калмыков И.А., Барильская А.В., Кихтенко О.А., Гахов В.Р. Реализация процедуры обратной нелинейному шифрованию с использованием индексного представления для поля Галуа // Материалы электронной заочной конференции Российской Академии Естествознания «Прикладные исследования и разработки по приоритетным направлениям науки и техники». – 15-20 ноября 2009.

57. Preneel B. Analysis and Design of Cryptographic Hash Functions, // Ph.D. dissertation. – Katholieke Universiteit Leuven. – Jan 1993.

58. Хомпыш А. Модуль бойынша дәрежеге шығару операциясы негізінде ақпаратты криптографиялық қорғау алгоритмін бағдарламалық жүзеге асыру // III Халықаралық «Информатика және қолданбалы математика» ғылыми конференция материалдары. – Алматы, 2018. – 167-171 б.

59. Кнут Д.Э. Искусство программирования: пер с англ. 3-е изд. Т. 2: Получисленные алгоритмы. – М.: Издат. Дом «Вильямс», 2000. – 828 с. [Knuth D.E. The art of computer programming. – In 3 V.3rd ed. Reading: Addison-Wesley Publ. Co., 1997].

60. Капалова Н.А., Хомпыш А., Алгазы К. ЕМ түрлендіру әдісі негізінде жасалған блокты шифрлеу алгоритміне жүргізілген бағалау тесттері // IV Халықаралық "Информатика және қолданбалы математика" ғылыми конференциясы. – Казахстан, Алматы, 2019. – 2. – 580-587 б.

61. Капалова Н.А., Хомпыш А., Алгазы К. Исследование разработанного алгоритма на основе преобразования ЕМ по критерию «лавинного эффекта» // М.Тынышбаев атындағы Қазақ көлік және коммуникациялар академиясының, Хабаршысы. –Алматы, 2020. – №3 (114). – 284-292б.

62. Бияшев Р.Г., Алгазы К.Т., Хомпыш А. Исследование разработанных алгоритмов по критерию «лавинного эффекта» // Матер. межд. науч.-практ. конф. «Актуальные проблемы информационной безопасности в Казахстане АПИБК-2020». – Алматы, 2020. – С.107-119.

63. Хомпыш А. Криптостойкости S-блоков в алгоритме шифрования на основе ЕМ // «Наука XXI века: новый подход»: Матер. XXIII молодёжной межд. науч.-практ. конф. студентов, аспирантов и молодых учёных. – г. Санкт-Петербург, 2019. – С. 15-19.

64. Kapalova N., Dyusenbayev D. Security analysis of an encryption scheme based on nonpositional polynomial notations // Open Engineering. – 2016. –№6. – P. 250-258.

65. Дюсенбаев Д.С., Сақан Қ.С., Алгазы К., Хомпыш А. «MODNPSS14» шифрлау алгоритміне криптографиялық талдау // М.Тынышбаев атындағы Қазақ көлік және коммуникациялар академиясының, Хабаршысы. – Алматы, 2019. – №3 (110). – 235-243 б.

66. Бияшев Р.Г., Капалова Н.А., Алгазы К.Т., Дюсенбаев Д.С., Хомпыш А. Криптоанализ генератора псевдослучайных последовательностей и ее модификация // Вестник Казахского национального исследовательского технического университета имени К.И. Сатпаева. – Алматы, 2019. – №3 (133). - С.179-185.

ҚОСЫМША А

```
#pragma once
#include<cstdlib>
namespace project {
	using namespace System;
	using namespace System::ComponentModel;
	using namespace System::Collections;
	using namespace System::Windows::Forms;
	using namespace System::Data;
	using namespace System::Drawing;
	using namespace System::IO;
	using namespace System::Text;
	public ref class MyForm : public System::Windows::Forms::Form
	{	public:
	 MyForm(void)
	 {	InitializeComponent();
				}
	protected:
		MyForm()
	 {		if (components)
		{	delete components; }  }
	protected:
	private: System::Windows::Forms::Button^  button6;
	private: System::Windows::Forms::Button^  button4;
	private: System::Windows::Forms::OpenFileDialog^  openFileDialog2;
	private: System::Windows::Forms::Button^  button5;
	private: System::Windows::Forms::TextBox^  textBox5;
	private: System::Windows::Forms::Label^  label4;
	private: System::Windows::Forms::TextBox^  textBox4;
	private: System::Windows::Forms::Label^  label3;
	private: System::Windows::Forms::SaveFileDialog^  saveFileDialog1;
	private: System::Windows::Forms::OpenFileDialog^  openFileDialog1;
	private: System::Windows::Forms::TextBox^  textBox3;
	private: System::Windows::Forms::Button^  button3;
	private: System::Windows::Forms::Button^  button1;
	private: System::Windows::Forms::Label^  label1;
	private: System::Windows::Forms::TextBox^  textBox1;
	private: System::Windows::Forms::Label^  label5;
	private: System::Windows::Forms::Label^  label2;
#pragma endregion
		long long Dare2(int x)
		{	long long tmp = 1;
			for (int i = 1; i <= x; i++)
			tmp += tmp;
```

```
        return tmp;         }
    long long JolBit_San(String^ sts)
    {   int k = sts->Length;
        long long sum = 0;
        for (int i = 0; i < k; i++)
        {   if (sts[i] == '1')
        {   sum += Dare2(k - 1 - i);        }           }
        return sum;  }
    int DarezheKorset(long long x)
    {   int n = 0;
        while (x)
        x >>= 1, ++n;
        return n;           }
    long long KaldikBit(long long x, long long p)
    {   long long res = p;
        while (DarezheKorset(x) >= DarezheKorset(p))
        {   x ^= p << (DarezheKorset(x) - DarezheKorset(p));
        }   return x;   }
    long long KobituBit(long long x, long long y, long long p)
    {   long long res = 0;
        while (y)
        {   if (y & 1)
            res = KaldikBit(res ^ x, p), --y;
            else
            x = KaldikBit(x <<= 1, p), y >>= 1;
        } return res;       }
    long long BoluBit(long long x, long long p)
    {   long long res = 0;
        while (DarezheKorset(x) >= DarezheKorset(p))
        {   int k = DarezheKorset(x) - DarezheKorset(p);
            x ^= p << k;
            res ^= 1 << k;
        }  return res;      }
    long long DarezheBit(long long x, long long y, long long p)
    {   long long res = 1;
        while (y)
    {   if (y & 1)
        res = KobituBit(res, x, p), --y;
        else
        x = KobituBit(x, x, p), y >>= 1;
        }   return res;   }
long long KeriSandar(long long a, long long b, long long p, long long& x, long
long& y)
    {   if (a == 0){
        x = 0; y = 1;
```

```
                return a;               }
                long long x1, y1;
                long long d = KeriSandar((b%a), a, p, x1, y1);
                x = y1 - (((b / a)* x1) % p);
                if (x < 0)
                {               x += p;                 }
                y = x1;
                return x;               }
        long long KeriBitdar(long long x, long long p)
        {       long long y, z;
                return KeriSandar(x, p, p, y, z);
        }       long long EYOB2(long long x, long long y)
        {       if (y == 0)
                return x;
                else
                return EYOB2(y, KaldikBit(x, y));       }
        long long EYOB(long long x, long long y)
        {       if (y == 0)
                return x;
                else
                return EYOB(y, x % y);
        }       StringBuilder^ OpenFile()               {
                StringBuilder^ str = gcnew StringBuilder();
                openFileDialog2->Filter = "All files (*.*)|*.*";if
(openFileDialog2->ShowDialog() == System::Windows::Forms::DialogResult::OK)
                {       System::IO::BinaryReader^ sr = gcnew
System::IO::BinaryReader(openFileDialog2->OpenFile(),
System::Text::Encoding::ASCII);
                String^ bit;
                String^ bat = "00000000";
                while (sr->PeekChar() != -1)
                {               bit = Convert::ToString(sr->ReadByte(), 2);
                        bit = bat->Substring(0, 8 - bit->Length) + bit;
                        str->Append(bit);
                }       sr->Close();                    }
                return str;             }
        void SaveFile(StringBuilder^ txt)
{ saveFileDialog1->Filter = L"All Files|*.*";
        if (this->saveFileDialog1->ShowDialog() ==
System::Windows::Forms::DialogResult::OK
                        && saveFileDialog1->FileName->Length > 0)
{ System::IO::BinaryWriter^ sr = gcnew System::IO::BinaryWriter(saveFileDialog1-
>OpenFile(), System::Text::Encoding::ASCII);
                int dl = txt->Length % 8;
                if (dl > 0)                             {
```

```
                for (int i = dl; i < 8; i++)
                txt->Append(0);                         }
                String^ str;
                for (int i = 0; i < txt->Length; i += 8)
                {       str = txt->ToString(i, 8);
                        long long san = JolBit_San(str);
                        unsigned char Char = Convert::ToByte(san);
                        sr->Write(Char);
                }       sr->Close();        }               }
        long long cryptoEM2(int *rkey, long long blok)
{ int ind[5][256] = { { 1, 2, 4, 8, 16, 32, 64, 128, 135, 137, 149, 173, 221, 61, 122,
244, 111, 222, 59, 118, 236, 95, 190, 251, 113, 226, 67, 134, 139, 145, 165, 205, 29,
58, 116, 232, 87, 174, 219, 49, 98, 196, 15, 30, 60, 120, 240, 103, 206, 27, 54, 108,
216, 55, 110, 220, 63, 126, 252, 127, 254, 123, 246, 107, 214, 43, 86, 172, 223, 57,
114, 228, 79, 158, 187, 241, 101, 202, 19, 38, 76, 152, 183, 233, 85, 170, 211, 33, 66,
132, 143, 153, 181, 237, 93, 186, 243, 97, 194, 3, 6, 12, 24, 48, 96, 192, 7, 14, 28, 56,
112, 224, 71, 142, 155, 177, 229, 77, 154, 179, 225, 69, 138, 147, 161, 197, 13, 26,
52, 104, 208, 39, 78, 156, 191, 249, 117, 234, 83, 166, 203, 17, 34, 68, 136, 151, 169,
213, 45, 90, 180, 239, 89, 178, 227, 65, 130, 131, 129, 133, 141, 157, 189, 253, 125,
250, 115, 230, 75, 150, 171, 209, 37, 74, 148, 175, 217, 53, 106, 212, 47, 94, 188,
255, 121, 242, 99, 198, 11, 22, 44, 88, 176, 231, 73, 146, 163, 193, 5, 10, 20, 40, 80,
160, 199, 9, 18, 36, 72, 144, 167, 201, 21, 42, 84, 168, 215, 41, 82, 164, 207, 25, 50,
100, 200, 23, 46, 92, 184, 247, 105, 210, 35, 70, 140, 159, 185, 245, 109, 218, 51,
102, 204, 31, 62, 124, 248, 119, 238, 91, 182, 235, 81, 162, 195 },
{ 1, 2, 4, 8, 16, 32, 64, 111, 49, 98, 43, 86, 67, 105, 61, 122, 27, 54, 108, 55, 110, 51,
102, 35, 70, 99, 41, 82, 75, 121, 29, 58, 116, 7, 14, 28, 56, 112, 15, 30, 60, 120, 31,
62, 124, 23, 46, 92, 87, 65, 109, 53, 106, 59, 118, 3, 6, 12, 24, 48, 96, 47, 94, 83, 73,
125, 21, 42, 84, 71, 97, 45, 90, 91, 89, 93, 85, 69, 101, 37, 74, 123, 25, 50, 100, 39,
78, 115, 9, 18, 36, 72, 127, 17, 34, 68, 103, 33, 66, 107, 57, 114, 11, 22, 44, 88, 95,
81, 77, 117, 5, 10, 20, 40, 80, 79, 113, 13, 26, 52, 104, 63, 126, 19, 38, 76, 119 },
{ 1, 2, 4, 8, 16, 32, 64, 119, 25, 50, 100, 63, 126, 11, 22, 44, 88, 71, 121, 5, 10, 20,
40, 80, 87, 89, 69, 125, 13, 26, 52, 104, 39, 78, 107, 33, 66, 115, 17, 34, 68, 127, 9,
18, 36, 72, 103, 57, 114, 19, 38, 76, 111, 41, 82, 83, 81, 85, 93, 77, 109, 45, 90, 67,
113, 21, 42, 84, 95, 73, 101, 61, 122, 3, 6, 12, 24, 48, 96, 55, 110, 43, 86, 91, 65, 117,
29, 58, 116, 31, 62, 124, 15, 30, 60, 120, 7, 14, 28, 56, 112, 23, 46, 92, 79, 105, 37,
74, 99, 49, 98, 51, 102, 59, 118, 27, 54, 108, 47, 94, 75, 97, 53, 106, 35, 70, 123 },
{ 1, 2, 4, 8, 16, 27, 13, 26, 15, 30, 7, 14, 28, 3, 6, 12, 24, 11, 22, 23, 21, 17, 25, 9, 18,
31, 5, 10, 20, 19, 29 },
{ 1, 2, 4, 8, 16, 15, 30, 19, 9, 18, 11, 22, 3, 6, 12, 24, 31, 17, 13, 26, 27, 25, 29, 21, 5,
10, 20, 7, 14, 28, 23 } };
int ind1[5][256] = { { 0, 1, 99, 2, 198, 100, 106, 3, 205, 199, 188, 101, 126, 107, 42,
4, 141, 206, 78, 200, 212, 189, 225, 102, 221, 127, 49, 108, 32, 43, 243, 5, 87, 142,
232, 207, 172, 79, 131, 201, 217, 213, 65, 190, 148, 226, 180, 103, 39, 222, 240, 128,
177, 50, 53, 109, 69, 33, 18, 44, 13, 244, 56, 6, 155, 88, 26, 143, 121, 233, 112, 208,
194, 173, 168, 80, 117, 132, 72, 202, 252, 218, 138, 214, 84, 66, 36, 191, 152, 149,
```

```
249, 227, 94, 181, 21, 104, 97, 40, 186, 223, 76, 241, 47, 129, 230, 178, 63, 51, 238,
54, 16, 110, 24, 70, 166, 34, 136, 19, 247, 45, 184, 14, 61, 245, 164, 57, 59, 7, 158,
156, 157, 89, 159, 27, 8, 144, 9, 122, 28, 234, 160, 113, 90, 209, 29, 195, 123, 174,
10, 169, 145, 81, 91, 118, 114, 133, 161, 73, 235, 203, 124, 253, 196, 219, 30, 139,
210, 215, 146, 85, 170, 67, 11, 37, 175, 192, 115, 153, 119, 150, 92, 250, 82, 228,
236, 95, 74, 182, 162, 22, 134, 105, 197, 98, 254, 41, 125, 187, 204, 224, 211, 77,
140, 242, 31, 48, 220, 130, 171, 231, 86, 179, 147, 64, 216, 52, 176, 239, 38, 55, 12,
17, 68, 111, 120, 25, 154, 71, 116, 167, 193, 35, 83, 137, 251, 20, 93, 248, 151, 46,
75, 185, 96, 15, 237, 62, 229, 246, 135, 165, 23, 58, 163, 60, 183 },
{ 0, 1, 55, 2, 110, 56, 33, 3, 88, 111, 102, 57, 117, 34, 38, 4, 93, 89, 123, 112, 66,
103, 45, 58, 82, 118, 16, 35, 30, 39, 42, 5, 97, 94, 23, 90, 79, 124, 85, 113, 26, 67, 10,
104, 71, 46, 61, 59, 8, 83, 21, 119, 51, 17, 19, 36, 100, 31, 53, 40, 14, 43, 121, 6, 49,
98, 12, 95, 77, 24, 69, 91, 64, 80, 28, 125, 108, 86, 115, 114, 107, 27, 63, 68, 76, 11,
48, 105, 74, 72, 73, 47, 75, 62, 106, 60, 70, 9, 25, 84, 78, 22, 96, 120, 13, 52, 99, 18,
50, 20, 7, 37, 116, 101, 87, 32, 109, 54, 126, 41, 29, 15, 81, 44, 65, 122, 92 },
{ 0, 1, 73, 2, 19, 74, 96, 3, 42, 20, 13, 75, 28, 97, 92, 4, 38, 43, 49, 21, 65, 14, 101,
76, 8, 29, 115, 98, 86, 93, 89, 5, 35, 39, 124, 44, 106, 50, 32, 22, 53, 66, 81, 15, 61,
102, 118, 77, 109, 9, 111, 30, 122, 116, 79, 99, 47, 87, 113, 94, 71, 90, 11, 6, 84, 36,
63, 40, 26, 125, 17, 45, 69, 107, 120, 51, 59, 33, 104, 23, 56, 54, 55, 67, 57, 82, 24,
16, 25, 62, 83, 103, 58, 119, 68, 78, 121, 110, 108, 10, 70, 112, 46, 31, 105, 123, 34,
117, 60, 80, 52, 100, 64, 48, 37, 88, 85, 114, 7, 95, 18, 72, 126, 91, 27, 12, 41 },
{ 0, 1, 13, 2, 26, 14, 10, 3, 23, 27, 17, 15, 6, 11, 8, 4, 21, 24, 29, 28, 20, 18, 19, 16,
22, 7, 5, 12, 30, 9, 25 },
{ 0, 1, 12, 2, 24, 13, 27, 3, 8, 25, 10, 14, 18, 28, 5, 4, 17, 9, 7, 26, 23, 11, 30, 15, 21,
19, 20, 29, 22, 6, 16 } };
                int pkey[5] = { 391, 239, 247, 59, 47 };
                int pbit[5] = { 8, 7, 7, 5, 5 };
                int ilr[5] = { 255, 127, 127, 31, 31 };
                int sblok[5];
                sblok[0] = (int)(blok >> 24);
                sblok[1] = (int)((blok >> 17) & 0x007f);
                sblok[2] = (int)((blok >> 10) & 0x00007f);
                sblok[3] = (int)((blok >> 5) & 0x000001f);
                sblok[4] = (int)(blok & 0x0000001f);
                long long deri, sum;
                int prim, indk;
                for (int i = 0; i < 5; i++)
                {   if (sblok[i] > 0)
                  {     indk = ind1[i][sblok[i] - 1];
                        indk = (indk * rkey[i]) % ilr[i];
                        deri = ind[i][indk];
                }       else{
                                deri = 0;
                }                               if (i == 0)
                {               sum = deri;                             }
```

```
                        else    {
                                sum = (sum << pbit[i]) ^ deri;
                    }           }       return sum;
            long long zikl(long long san, int kadam)
    {       return ((san << kadam) & 0x00000000ffffffff) ^ (san >> (32 - kadam));
            }       long long zikl1(long long san, int kadam)
    {       return ((san << (32 - kadam)) & 0x00000000ffffffff) ^ (san >> kadam);
            }       long long iSblok(long long engiz)
{ int S[256] = { 1, 27, 52, 94, 212, 101, 19, 236, 143, 198, 146, 168, 116, 201,11,
245, 141, 240, 250, 20, 173, 3, 45, 92, 226, 13, 175, 53, 69, 224, 59,199, 137, 156,
42, 29, 110, 230, 97, 127, 60, 134, 5, 119, 228, 87, 23, 128,95, 207, 81, 77, 56, 234,
213, 126, 39, 178, 91, 163, 129, 68, 251, 15, 153,93, 249, 57, 241, 225, 32, 243, 215,
72, 79, 14, 130, 105, 167, 237, 148, 242,204, 124, 17, 218, 231, 122, 75, 98, 82, 96,
100, 8, 216, 209, 18, 247, 187, 152,70, 205, 103, 37, 132, 51, 31, 88, 142, 221, 166,
246, 160, 172, 24, 25, 2, 54,104, 188, 217, 202, 38, 169, 111, 253, 85, 33, 232, 227,
22, 155, 107, 145, 133,40, 43, 6, 90, 184, 181, 26, 47, 106, 138, 177, 118, 255, 99,
73, 84, 58, 220,189, 194, 254, 120, 125, 10, 238, 185, 174, 46, 113, 190, 239, 162,
154, 112,165, 219, 252, 78, 21, 182, 55, 115, 136, 135, 30, 67, 186, 131, 114, 147,
179,64, 151, 223, 144, 158, 28, 117, 210, 63, 171, 89, 149, 233, 248, 34, 197, 191,
244, 150, 196, 164, 192, 200, 16, 193, 211, 36, 159, 7, 65, 140, 235, 206, 74,121,
102, 62, 176, 109, 203, 61, 157, 49, 41, 48, 50, 4, 108, 208, 9, 195, 229,76, 35, 222,
139, 170, 66, 161, 183, 44, 71, 214, 83, 123, 80, 86, 12, 180, 0 };
            int sb1 = (int)(engiz >> 24);
            int sb2 = (int)((engiz >> 16) & 0x00ff);
            int sb3 = (int)((engiz >> 8) & 0x0000ff);
            int sb4 = (int)(engiz & 0x000000ff);
            long long birgu;
            birgu = (long long)S[sb1] << 24;
            birgu ^= (long long)S[sb2] << 16;
            birgu ^= (long long)S[sb3] << 8;
            birgu ^= (long long)S[sb4];
            return birgu;
            }       long long iSblok1(long long engiz)    {
int S1[256] = { 255, 0, 116, 21, 232, 42, 137, 214, 93, 235, 158, 14, 253, 25, 75, 63,
209, 84, 96, 6, 19, 173, 130, 46, 114, 115, 141, 1, 191, 35, 179, 106, 70, 127, 200,
239, 212, 103, 122, 56, 135, 229, 34, 136, 246, 22, 162, 142, 230, 228, 231, 105, 2,
27, 117, 175, 52, 67, 151, 30, 40, 226, 222, 194, 186, 215, 243, 180, 61, 28, 100,
247, 73, 149, 219, 88, 238, 51, 172, 74, 251, 50, 90, 249, 150, 126, 252, 45, 107,
196, 138, 58, 23, 65, 3, 48, 91, 38, 89, 148, 92, 5, 221, 102, 118, 77, 143, 132,233,
224, 36, 124, 168, 163, 183, 176, 12, 192, 146, 43, 156, 220, 87, 250, 83, 157,55, 39,
47, 60, 76, 182, 104, 134, 41, 178, 177, 32, 144, 241, 216, 16, 108, 8, 189,133, 10,
184, 80, 197, 204, 187, 99, 64, 167, 131, 33, 227, 190, 213, 112, 244, 166,59, 206,
169, 110, 78, 11, 123, 242, 195, 113, 20, 161, 26, 223, 145, 57, 185, 254,140, 174,
245, 139, 160, 181, 98, 119, 153, 164, 202, 207, 210, 154, 236, 205, 201,9, 31, 208,
13, 121, 225, 82, 101, 218, 49, 234, 95, 193, 211, 4, 54, 248, 72, 94,120, 85, 170,
```

```
152, 109, 240, 188, 29, 69, 24, 129, 44, 237, 37, 86, 128, 198, 53,217, 7, 79, 159,
165, 17, 68, 81, 71, 203, 15, 111, 97, 199, 66, 18, 62, 171, 125,155, 147 };
                int sb1 = (int)(engiz >> 24);
                int sb2 = (int)((engiz >> 16) & 0x00ff);
                int sb3 = (int)((engiz >> 8) & 0x0000ff);
                int sb4 = (int)(engiz & 0x000000ff);
                long long birgu;
                birgu = (long long)S1[sb1] << 24;
                birgu ^= (long long)S1[sb2] << 16;
                birgu ^= (long long)S1[sb3] << 8;
                birgu ^= (long long)S1[sb4];
                return birgu;
            }       void cipherkeyround()           {
                    String^ key = "";
                    String^ ky = "";
                    String^ key_1 = "";
                    long long jai, jaiX;
                    long long x, y;
                    String^ strkey = textBox1->Text;
                    long long tkey[5];
                    for (int j = 256; j < 768; j += 32)
                    {       ky = strkey->Substring(j, 32);
                            int orin = 0;
                    for (int i = 0; i < 5; i++)
          { jai = Dare2(textBox3->Lines[i]->ToString()->Length - 1) - 1;
          tkey[i] = textBox3->Lines[i]->ToString()->Length - 1;
                    x = JolBit_San(ky->Substring(orin, tkey[i]));
                    while (EYOB(x, jai) != 1)
                    x = rand() % (jai - 1) + 2;
                    y = KeriBitdar(x, jai);
                    orin += tkey[i];
                    key += Convert::ToString(x) + "\r\n";
                    key_1 += Convert::ToString(y) + "\r\n";     }       }
                    textBox4->Text = key;
                    textBox5->Text = key_1;
            }       String^ Int32ToStr(long long* san)
            {       String^ rtxt = "";
                    String^ stxt;
                    String^ itxt;
                    for (int i = 0; i < 4; i++)
            {   stxt = Convert::ToString(san[i], 2);
                    itxt = "";
                    for (int j = stxt->Length; j < 32; j++)
                    {               itxt = itxt + "0";
                            }       rtxt = rtxt + itxt + stxt;
```

```
            }               return rtxt;
        }       StringBuilder^ CryptoX(StringBuilder^ str)
        {       String^ rstr1;
                String^ rstr = "";
                String^ rst;
                String^ strkey = textBox1->Text;
                long long tkey[8];
                int orin = 0;
                for (int i = 0; i < 8; i++)
                {       tkey[i] = JolBit_San(strkey->Substring(orin, 32));
                        orin += 32;
                }       int KeyJabik[16][5];
                for (int i = 0; i < 16; i++)
                {       for (int j = 0; j < 5; j++)
KeyJabik[i][j] = Convert::ToInt32(textBox4->Lines[i * 5 + j]->ToString());
                }       StringBuilder^ txt = gcnew StringBuilder();
                txt->Clear();
                int kaldik = 128 - (str->Length % 128);
                rstr1 = Convert::ToString(str->Length, 2);
                for (int i = rstr1->Length; i < 32; i++)
                txt->Append(0);
                txt->Append(rstr1);
                if ((str->Length % 128) != 0)
                for (int i = 0; i < kaldik; i++)
                {               str->Append(0);
                }               else            str;
                long long rblok[4];
                long long iblok[4];
                for (int i = 0; i < str->Length; i += 128)
                {       rst = str->ToString(i, 128);
                        {               for (int j = 0; j < 4; j++)
        {               rblok[j] = JolBit_San(rst->Substring((j * 32), 32)) ^ tkey[j];
        }       }       for (int i = 0; i < 16; i++)
                {       iblok[0] = cryptoEM2(KeyJabik[i], rblok[0]);
                        iblok[1] = cryptoEM2(KeyJabik[i], rblok[1]);
                        iblok[2] = iSblok(rblok[2]);
                        iblok[3] = iSblok(rblok[3]);
                        rblok[0] = zikl(iblok[1], 5) ^ iblok[3];
                        rblok[1] = zikl(iblok[0], 5) ^ iblok[2];
                        rblok[2] = zikl(iblok[0], 3);
                        rblok[3] = zikl(iblok[1], 3);
        }               for (int j = 0; j < 4; j++)
        {               rblok[j] ^= tkey[j + 4];
        }               rstr = Int32ToStr(rblok);
                        txt->Append(rstr);
```

```
}               return txt;    }
StringBuilder^ DecryptoX(StringBuilder^ str)
{       String^ rstr;
        String^ strkey = textBox1->Text;
        long long tkey[8];
        int orin = 0;
        for (int i = 0; i < 8; i++)
{       tkey[i] = JolBit_San(strkey->Substring(orin, 32));
        orin += 32;
}       int Jabik[16][5];
        for (int i = 0; i < 16; i++)
{       for (int j = 0; j < 5; j++)
Jabik[i][j] = Convert::ToInt32(textBox5->Lines[i * 5 + j]->ToString());
}       rstr = str->ToString(0, 32);
        long long dl = JolBit_San(rstr);
        StringBuilder^ txt = gcnew StringBuilder();
        txt->Clear();
        long long rblok[4];
        long long iblok[4];
        for (int i = 32; i < str->Length; i += 128)
{       rstr = str->ToString(i, 128);
        for (int j = 0; j < 4; j++)
{       rblok[j] = JolBit_San(rstr->Substring((j * 32), 32)) ^ tkey[j + 4];
}       for (int k = 15; k >= 0; k--)
{       iblok[0] = zikl1(rblok[2], 3);
        iblok[1] = zikl1(rblok[3], 3);
        iblok[2] = zikl(iblok[0], 5) ^ rblok[1];
        iblok[3] = zikl(iblok[1], 5) ^ rblok[0];
        rblok[0] = cryptoEM2(Jabik[k], iblok[0]);
        rblok[1] = cryptoEM2(Jabik[k], iblok[1]);
        rblok[2] = iSblok1(iblok[2]);
        rblok[3] = iSblok1(iblok[3]);
        }
        for (int j = 0; j < 4; j++)
        {       rblok[j] ^= tkey[j];
        }       rstr = Int32ToStr(rblok);
                txt->Append(rstr);
        }       str->Clear();
        str->Append(txt->ToString(0, dl));
        return str;             }
private: System::Void button1_Click_1(System::Object^  sender,
System::EventArgs^  e) { if (openFileDialog1->ShowDialog() ==
System::Windows::Forms::DialogResult::OK)
                {       String^ strkey;
```

```
FileStream^ skey = gcnew System::IO::FileStream(openFileDialog1->FileName,
System::IO::FileMode::Open, System::IO::FileAccess::Read);
System::IO::StreamReader^ binkey = gcnew System::IO::StreamReader(skey);
                        while ((strkey = binkey->ReadLine()) != nullptr)
            {    textBox1->Text += strkey;        }        } }
           private: System::Void button3_Click_1(System::Object^  sender,
System::EventArgs^  e) { if (openFileDialog1->ShowDialog() ==
System::Windows::Forms::DialogResult::OK)
                        {        String^ strkey;
                                 FileStream^ skey = gcnew
System::IO::FileStream(openFileDialog1->FileName, System::IO::FileMode::Open,
System::IO::FileAccess::Read);
System::IO::StreamReader^ binkey = gcnew System::IO::StreamReader(skey);
                                 while ((strkey = binkey->ReadLine()) != nullptr)
            {    textBox3->Text += strkey + "\r\n";     }  }    }
           private: System::Void button4_Click_1(System::Object^  sender,
System::EventArgs^  e) {          cipherkeyround();          }
private: System::Void button5_Click_1(System::Object^  sender,
System::EventArgs^  e) {  StringBuilder^ str = gcnew StringBuilder();
                                 StringBuilder^ txt = gcnew StringBuilder();
                                 str->Clear();
                                 txt->Clear();
                                 str->Append(OpenFile());
                                 txt->Append(CryptoX(str));
                SaveFile(txt);       }
private: System::Void button6_Click_1(System::Object^  sender,
System::EventArgs^  e) { StringBuilder^ str = gcnew StringBuilder();
                                 StringBuilder^ txt = gcnew StringBuilder();
                                 str->Clear();
                                 txt->Clear();
                                 str->Append(OpenFile());
                                 txt->Append(DecryptoX(str));
                                 SaveFile(txt);  }  };        }
```

ҚОСЫМША Ә

ЭЕМ-ге арналған бағдарламалық авторлық құқық куәлігі.

ҚАЗАҚСТАН РЕСПУБЛИКАСЫ

РЕСПУБЛИКА КАЗАХСТАН

АВТОРЛЫҚ ҚҰҚЫҚПЕН ҚОРҒАЛАТЫН ОБЪЕКТІЛЕРГЕ ҚҰҚЫҚТАРДЫҢ МЕМЛЕКЕТТІК ТІЗІЛІМГЕ МӘЛІМЕТТЕРДІ ЕНГІЗУ ТУРАЛЫ

КУӘЛІК

2019 жылғы «24» қыркүйек № 5450

Автордың (лардың) жөні, аты, әкесінің аты (егер ол жеке басын куәландыратын құжатта көрсетілсе):
ХОМПЫШ АРДАБЕК , КАПАЛОВА НУРСУЛУ АЛДАЖАРОВНА

Авторлық құқық объектісі: ЭЕМ-ге арналған бағдарлама

Объектінің атауы: Программа шифрования файлов «CryptoEM v1.0.1»

Объектіні жасаған күні: 16.09.2019

Подписано ЭЦП

Оспанов Е.К.

ҚОСЫМША Б

Кесте Қ.1 – EMCipher алгоритмнің 1 раундтағы биттік шашырау нәтижесі

	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,09	17	0,02	33	0,06	49	0,08	65	0,02	81	0,05	97	0,04	113	0,04
2	0,06	18	0,06	34	0,06	50	0,08	66	0,02	82	0,02	98	0,02	114	0,02
3	0,05	19	0,02	35	0,08	51	0,08	67	0,02	83	0,03	99	0,03	115	0,03
4	0,05	20	0,03	36	0,05	52	0,08	68	0,02	84	0,03	100	0,02	116	0,02
5	0,09	21	0,05	37	0,06	53	0,03	69	0,02	85	0,04	101	0,02	117	0,03
6	0,08	22	0,06	38	0,08	54	0,05	70	0,03	86	0,03	102	0,03	118	0,02
7	0,05	23	0,02	39	0,05	55	0,02	71	0,05	87	0,02	103	0,04	119	0,02
8	0,08	24	0,05	40	0,09	56	0,05	72	0,03	88	0,03	104	0,01	120	0,02
9	0,06	25	0,06	41	0,03	57	0,05	73	0,02	89	0,04	105	0,03	121	0,05
10	0,06	26	0,02	42	0,09	58	0,03	74	0,02	90	0,03	106	0,02	122	0,02
11	0,02	27	0,05	43	0,03	59	0,05	75	0,02	91	0,03	107	0,04	123	0,03
12	0,09	28	0,05	44	0,05	60	0,03	76	0,04	92	0,04	108	0,03	124	0,02
13	0,06	29	0,02	45	0,06	61	0,03	77	0,05	93	0,05	109	0,03	125	0,02
14	0,08	30	0,05	46	0,06	62	0,03	78	0,03	94	0,05	110	0,03	126	0,02
15	0,03	31	0,05	47	0,08	63	0,03	79	0,03	95	0,02	111	0,02	127	0,03
16	0,06	32	0,06	48	0,06	64	0,03	80	0,02	96	0,03	112	0,02	128	0,05

Кесте Қ.2 – EMCipher алгоритмнің 2 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,13	17	0,09	33	0,20	49	0,15	65	0,05	81	0,11	97	0,09	113	0,02
2	0,13	18	0,16	34	0,15	50	0,19	66	0,09	82	0,13	98	0,05	114	0,06
3	0,09	19	0,10	35	0,15	51	0,17	67	0,06	83	0,11	99	0,06	115	0,11
4	0,10	20	0,20	36	0,19	52	0,20	68	0,08	84	0,08	100	0,09	116	0,08
5	0,17	21	0,13	37	0,15	53	0,20	69	0,08	85	0,08	101	0,06	117	0,06
6	0,19	22	0,11	38	0,20	54	0,26	70	0,03	86	0,11	102	0,06	118	0,05
7	0,17	23	0,10	39	0,12	55	0,09	71	0,03	87	0,06	103	0,11	119	0,06
8	0,17	24	0,07	40	0,16	56	0,05	72	0,03	88	0,03	104	0,05	120	0,08
9	0,19	25	0,10	41	0,11	57	0,08	73	0,09	89	0,05	105	0,09	121	0,11
10	0,27	26	0,07	42	0,22	58	0,11	74	0,13	90	0,09	106	0,03	122	0,11
11	0,13	27	0,08	43	0,22	59	0,06	75	0,06	91	0,11	107	0,09	123	0,06
12	0,27	28	0,07	44	0,21	60	0,03	76	0,08	92	0,11	108	0,11	124	0,08
13	0,22	29	0,07	45	0,21	61	0,04	77	0,14	93	0,09	109	0,09	125	0,06
14	0,23	30	0,05	46	0,20	62	0,08	78	0,03	94	0,08	110	0,09	126	0,08
15	0,14	31	0,10	47	0,20	63	0,06	79	0,06	95	0,03	111	0,05	127	0,06
16	0,16	32	0,10	48	0,21	64	0,11	80	0,05	96	0,06	112	0,05	128	0,09

Кесте Қ.3 – EMCipher алгоритмнің 3 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,25	17	0,20	33	0,30	49	0,33	65	0,18	81	0,16	97	0,23	113	0,09
2	0,22	18	0,33	34	0,25	50	0,33	66	0,16	82	0,23	98	0,19	114	0,16
3	0,16	19	0,26	35	0,36	51	0,29	67	0,22	83	0,23	99	0,14	115	0,19
4	0,23	20	0,30	36	0,36	52	0,34	68	0,15	84	0,16	100	0,20	116	0,16
5	0,20	21	0,24	37	0,34	53	0,38	69	0,14	85	0,17	101	0,26	117	0,17
6	0,27	22	0,32	38	0,33	54	0,27	70	0,11	86	0,16	102	0,19	118	0,15
7	0,27	23	0,22	39	0,30	55	0,23	71	0,09	87	0,20	103	0,22	119	0,10
8	0,32	24	0,20	40	0,32	56	0,13	72	0,14	88	0,13	104	0,09	120	0,13
9	0,34	25	0,17	41	0,21	57	0,13	73	0,22	89	0,19	105	0,16	121	0,20
10	0,37	26	0,20	42	0,34	58	0,22	74	0,20	90	0,16	106	0,17	122	0,18
11	0,16	27	0,24	43	0,29	59	0,23	75	0,20	91	0,16	107	0,23	123	0,13
12	0,29	28	0,16	44	0,30	60	0,09	76	0,16	92	0,17	108	0,34	124	0,16
13	0,27	29	0,17	45	0,30	61	0,15	77	0,24	93	0,13	109	0,30	125	0,15
14	0,34	30	0,13	46	0,24	62	0,11	78	0,19	94	0,17	110	0,20	126	0,16
15	0,35	31	0,20	47	0,33	63	0,14	79	0,14	95	0,05	111	0,19	127	0,22
16	0,30	32	0,16	48	0,33	64	0,10	80	0,21	96	0,19	112	0,18	128	0,15

Кесте Қ.4 – EMCipher алгоритмнің 4 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,49	17	0,28	33	0,40	49	0,43	65	0,20	81	0,31	97	0,34	113	0,14
2	0,33	18	0,50	34	0,42	50	0,50	66	0,27	82	0,32	98	0,30	114	0,34
3	0,27	19	0,42	35	0,48	51	0,43	67	0,29	83	0,27	99	0,20	115	0,36
4	0,44	20	0,50	36	0,43	52	0,44	68	0,30	84	0,39	100	0,31	116	0,33
5	0,40	21	0,38	37	0,50	53	0,40	69	0,30	85	0,27	101	0,34	117	0,39
6	0,40	22	0,44	38	0,44	54	0,45	70	0,08	86	0,31	102	0,38	118	0,31
7	0,47	23	0,36	39	0,48	55	0,37	71	0,16	87	0,28	103	0,41	119	0,43
8	0,43	24	0,35	40	0,49	56	0,37	72	0,23	88	0,22	104	0,15	120	0,30
9	0,48	25	0,34	41	0,30	57	0,31	73	0,34	89	0,23	105	0,36	121	0,20
10	0,45	26	0,31	42	0,54	58	0,39	74	0,38	90	0,27	106	0,18	122	0,23
11	0,32	27	0,29	43	0,41	59	0,34	75	0,38	91	0,27	107	0,41	123	0,25
12	0,52	28	0,20	44	0,33	60	0,19	76	0,32	92	0,21	108	0,47	124	0,33
13	0,47	29	0,29	45	0,42	61	0,23	77	0,38	93	0,32	109	0,40	125	0,28
14	0,46	30	0,24	46	0,57	62	0,18	78	0,34	94	0,28	110	0,34	126	0,32
15	0,49	31	0,34	47	0,45	63	0,21	79	0,28	95	0,16	111	0,34	127	0,20
16	0,40	32	0,27	48	0,47	64	0,30	80	0,43	96	0,28	112	0,28	128	0,21

Кесте Қ.5 – EMCipher алгоритмнің 6 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,50	17	0,43	33	0,48	49	0,52	65	0,55	81	0,47	97	0,58	113	0,50
2	0,49	18	0,50	34	0,46	50	0,44	66	0,51	82	0,51	98	0,52	114	0,48
3	0,55	19	0,51	35	0,52	51	0,57	67	0,48	83	0,48	99	0,49	115	0,48
4	0,50	20	0,55	36	0,47	52	0,45	68	0,48	84	0,54	100	0,52	116	0,52
5	0,48	21	0,42	37	0,52	53	0,42	69	0,38	85	0,50	101	0,52	117	0,53
6	0,53	22	0,53	38	0,51	54	0,50	70	0,34	86	0,45	102	0,46	118	0,50
7	0,53	23	0,58	39	0,50	55	0,48	71	0,42	87	0,52	103	0,51	119	0,46
8	0,44	24	0,54	40	0,48	56	0,41	72	0,52	88	0,46	104	0,52	120	0,46
9	0,54	25	0,44	41	0,44	57	0,44	73	0,34	89	0,49	105	0,47	121	0,52
10	0,45	26	0,41	42	0,55	58	0,53	74	0,47	90	0,48	106	0,49	122	0,47
11	0,44	27	0,58	43	0,50	59	0,51	75	0,56	91	0,52	107	0,52	123	0,39
12	0,52	28	0,54	44	0,48	60	0,53	76	0,55	92	0,42	108	0,47	124	0,42
13	0,53	29	0,48	45	0,57	61	0,47	77	0,42	93	0,53	109	0,54	125	0,43
14	0,59	30	0,37	46	0,41	62	0,49	78	0,51	94	0,52	110	0,52	126	0,41
15	0,47	31	0,42	47	0,48	63	0,42	79	0,47	95	0,38	111	0,54	127	0,48
16	0,55	32	0,46	48	0,50	64	0,48	80	0,49	96	0,51	112	0,53	128	0,50

Кесте Қ.6 – EMCipher алгоритмнің 7 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,52	17	0,57	33	0,52	49	0,55	65	0,45	81	0,49	97	0,45	113	0,47
2	0,54	18	0,45	34	0,53	50	0,44	66	0,45	82	0,50	98	0,45	114	0,55
3	0,52	19	0,55	35	0,52	51	0,55	67	0,55	83	0,58	99	0,42	115	0,55
4	0,49	20	0,48	36	0,47	52	0,47	68	0,52	84	0,52	100	0,48	116	0,41
5	0,47	21	0,52	37	0,53	53	0,45	69	0,53	85	0,45	101	0,46	117	0,50
6	0,45	22	0,55	38	0,53	54	0,42	70	0,53	86	0,53	102	0,48	118	0,55
7	0,55	23	0,49	39	0,50	55	0,53	71	0,45	87	0,52	103	0,51	119	0,48
8	0,50	24	0,46	40	0,52	56	0,53	72	0,56	88	0,45	104	0,44	120	0,52
9	0,45	25	0,50	41	0,46	57	0,52	73	0,41	89	0,53	105	0,55	121	0,57
10	0,49	26	0,44	42	0,45	58	0,60	74	0,52	90	0,52	106	0,55	122	0,43
11	0,54	27	0,41	43	0,55	59	0,43	75	0,58	91	0,52	107	0,50	123	0,50
12	0,52	28	0,39	44	0,55	60	0,50	76	0,50	92	0,52	108	0,49	124	0,41
13	0,49	29	0,50	45	0,46	61	0,47	77	0,56	93	0,46	109	0,53	125	0,42
14	0,50	30	0,51	46	0,49	62	0,61	78	0,51	94	0,44	110	0,55	126	0,45
15	0,56	31	0,51	47	0,48	63	0,52	79	0,52	95	0,48	111	0,50	127	0,52
16	0,58	32	0,58	48	0,52	64	0,58	80	0,42	96	0,58	112	0,55	128	0,56

Кесте Қ.7 – EMCipher алгоритмнің 8 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,46	33	0,48	49	0,53	65	0,52	81	0,50	97	0,58	113	0,51
2	0,55	18	0,48	34	0,46	50	0,41	66	0,52	82	0,48	98	0,51	114	0,53
3	0,49	19	0,60	35	0,48	51	0,49	67	0,45	83	0,46	99	0,48	115	0,53
4	0,48	20	0,46	36	0,46	52	0,54	68	0,63	84	0,42	100	0,46	116	0,53
5	0,46	21	0,48	37	0,52	53	0,45	69	0,54	85	0,49	101	0,52	117	0,55
6	0,47	22	0,49	38	0,46	54	0,54	70	0,50	86	0,51	102	0,48	118	0,53
7	0,50	23	0,46	39	0,44	55	0,48	71	0,48	87	0,48	103	0,48	119	0,55
8	0,55	24	0,55	40	0,55	56	0,55	72	0,47	88	0,54	104	0,48	120	0,52
9	0,45	25	0,49	41	0,51	57	0,57	73	0,51	89	0,44	105	0,52	121	0,47
10	0,52	26	0,42	42	0,43	58	0,55	74	0,55	90	0,54	106	0,52	122	0,49
11	0,52	27	0,55	43	0,46	59	0,55	75	0,43	91	0,47	107	0,55	123	0,48
12	0,52	28	0,41	44	0,52	60	0,53	76	0,59	92	0,42	108	0,41	124	0,47
13	0,48	29	0,52	45	0,51	61	0,51	77	0,43	93	0,53	109	0,49	125	0,48
14	0,59	30	0,48	46	0,48	62	0,53	78	0,41	94	0,47	110	0,60	126	0,48
15	0,54	31	0,49	47	0,56	63	0,57	79	0,52	95	0,55	111	0,49	127	0,48
16	0,46	32	0,55	48	0,53	64	0,53	80	0,56	96	0,48	112	0,52	128	0,54

Кесте Қ.8 – EMCipher алгоритмнің 9 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,52	33	0,48	49	0,49	65	0,42	81	0,53	97	0,53	113	0,50
2	0,57	18	0,45	34	0,43	50	0,53	66	0,49	82	0,50	98	0,47	114	0,54
3	0,60	19	0,50	35	0,55	51	0,50	67	0,48	83	0,50	99	0,53	115	0,50
4	0,56	20	0,45	36	0,52	52	0,52	68	0,46	84	0,47	100	0,45	116	0,48
5	0,56	21	0,54	37	0,52	53	0,51	69	0,58	85	0,48	101	0,55	117	0,43
6	0,41	22	0,46	38	0,47	54	0,45	70	0,50	86	0,50	102	0,52	118	0,50
7	0,57	23	0,49	39	0,48	55	0,46	71	0,48	87	0,52	103	0,45	119	0,48
8	0,52	24	0,47	40	0,59	56	0,58	72	0,50	88	0,58	104	0,45	120	0,54
9	0,54	25	0,50	41	0,52	57	0,54	73	0,58	89	0,52	105	0,46	121	0,53
10	0,54	26	0,47	42	0,47	58	0,52	74	0,55	90	0,50	106	0,44	122	0,43
11	0,51	27	0,54	43	0,50	59	0,48	75	0,52	91	0,54	107	0,45	123	0,48
12	0,56	28	0,45	44	0,45	60	0,45	76	0,53	92	0,51	108	0,45	124	0,47
13	0,54	29	0,55	45	0,57	61	0,51	77	0,41	93	0,60	109	0,55	125	0,50
14	0,55	30	0,52	46	0,52	62	0,41	78	0,51	94	0,49	110	0,50	126	0,52
15	0,47	31	0,59	47	0,42	63	0,57	79	0,49	95	0,52	111	0,50	127	0,54
16	0,50	32	0,52	48	0,51	64	0,46	80	0,52	96	0,51	112	0,47	128	0,50

Кесте Қ.9 – EMCipher алгоритмнің 11 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,45	33	0,50	49	0,56	65	0,46	81	0,48	97	0,55	113	0,48
2	0,54	18	0,44	34	0,44	50	0,51	66	0,55	82	0,52	98	0,46	114	0,48
3	0,58	19	0,48	35	0,53	51	0,55	67	0,49	83	0,49	99	0,46	115	0,48
4	0,47	20	0,50	36	0,52	52	0,53	68	0,55	84	0,46	100	0,54	116	0,57
5	0,56	21	0,43	37	0,51	53	0,52	69	0,45	85	0,56	101	0,52	117	0,51
6	0,54	22	0,46	38	0,48	54	0,51	70	0,59	86	0,52	102	0,57	118	0,47
7	0,45	23	0,44	39	0,47	55	0,48	71	0,50	87	0,45	103	0,48	119	0,53
8	0,47	24	0,59	40	0,47	56	0,48	72	0,54	88	0,53	104	0,49	120	0,47
9	0,46	25	0,49	41	0,52	57	0,43	73	0,52	89	0,48	105	0,46	121	0,53
10	0,50	26	0,55	42	0,56	58	0,48	74	0,56	90	0,50	106	0,49	122	0,56
11	0,41	27	0,49	43	0,52	59	0,44	75	0,47	91	0,50	107	0,46	123	0,59
12	0,54	28	0,50	44	0,54	60	0,45	76	0,55	92	0,58	108	0,46	124	0,46
13	0,46	29	0,57	45	0,58	61	0,53	77	0,48	93	0,43	109	0,43	125	0,45
14	0,55	30	0,55	46	0,59	62	0,48	78	0,52	94	0,50	110	0,52	126	0,48
15	0,54	31	0,50	47	0,53	63	0,47	79	0,45	95	0,51	111	0,52	127	0,48
16	0,54	32	0,48	48	0,42	64	0,51	80	0,51	96	0,53	112	0,57	128	0,44

Кесте Қ.10 – EMCipher алгоритмнің 12 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,42	33	0,49	49	0,52	65	0,49	81	0,45	97	0,52	113	0,51
2	0,45	18	0,46	34	0,49	50	0,51	66	0,52	82	0,48	98	0,49	114	0,47
3	0,57	19	0,43	35	0,48	51	0,53	67	0,41	83	0,50	99	0,50	115	0,59
4	0,48	20	0,58	36	0,51	52	0,57	68	0,55	84	0,48	100	0,46	116	0,52
5	0,53	21	0,56	37	0,52	53	0,51	69	0,54	85	0,55	101	0,48	117	0,48
6	0,52	22	0,58	38	0,50	54	0,44	70	0,48	86	0,55	102	0,45	118	0,53
7	0,49	23	0,48	39	0,43	55	0,41	71	0,51	87	0,48	103	0,54	119	0,48
8	0,47	24	0,43	40	0,55	56	0,50	72	0,48	88	0,48	104	0,59	120	0,48
9	0,48	25	0,46	41	0,53	57	0,46	73	0,45	89	0,50	105	0,51	121	0,52
10	0,48	26	0,58	42	0,55	58	0,60	74	0,46	90	0,46	106	0,47	122	0,50
11	0,48	27	0,48	43	0,52	59	0,46	75	0,59	91	0,45	107	0,54	123	0,48
12	0,53	28	0,56	44	0,47	60	0,45	76	0,48	92	0,48	108	0,52	124	0,49
13	0,52	29	0,44	45	0,48	61	0,49	77	0,50	93	0,47	109	0,47	125	0,49
14	0,50	30	0,62	46	0,45	62	0,40	78	0,42	94	0,57	110	0,55	126	0,45
15	0,41	31	0,52	47	0,51	63	0,52	79	0,53	95	0,48	111	0,55	127	0,53
16	0,45	32	0,52	48	0,50	64	0,55	80	0,46	96	0,49	112	0,48	128	0,44

Кесте Қ.11 – EMCipher алгоритмнің 13 раундтағы биттік шашырау нәтижесі

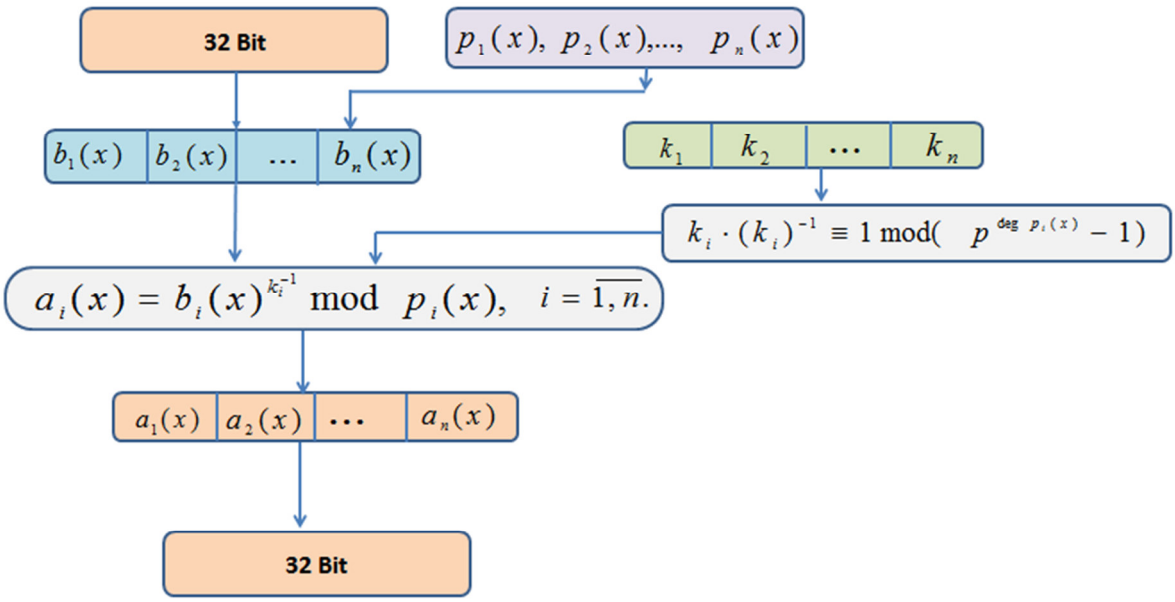
i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,42	33	0,50	49	0,53	65	0,52	81	0,46	97	0,51	113	0,58
2	0,51	18	0,51	34	0,46	50	0,50	66	0,47	82	0,50	98	0,53	114	0,47
3	0,53	19	0,50	35	0,48	51	0,52	67	0,48	83	0,49	99	0,56	115	0,46
4	0,58	20	0,47	36	0,50	52	0,52	68	0,53	84	0,56	100	0,55	116	0,51
5	0,51	21	0,55	37	0,48	53	0,52	69	0,59	85	0,52	101	0,49	117	0,49
6	0,45	22	0,49	38	0,49	54	0,52	70	0,47	86	0,46	102	0,51	118	0,52
7	0,52	23	0,43	39	0,55	55	0,52	71	0,50	87	0,48	103	0,52	119	0,51
8	0,50	24	0,52	40	0,47	56	0,41	72	0,45	88	0,44	104	0,51	120	0,39
9	0,52	25	0,52	41	0,43	57	0,57	73	0,49	89	0,52	105	0,48	121	0,52
10	0,41	26	0,47	42	0,50	58	0,55	74	0,43	90	0,43	106	0,43	122	0,49
11	0,63	27	0,46	43	0,47	59	0,45	75	0,48	91	0,48	107	0,48	123	0,51
12	0,47	28	0,47	44	0,49	60	0,53	76	0,44	92	0,49	108	0,47	124	0,43
13	0,45	29	0,53	45	0,45	61	0,58	77	0,53	93	0,48	109	0,49	125	0,51
14	0,57	30	0,57	46	0,47	62	0,50	78	0,48	94	0,55	110	0,59	126	0,46
15	0,57	31	0,51	47	0,47	63	0,52	79	0,46	95	0,52	111	0,51	127	0,48
16	0,52	32	0,54	48	0,46	64	0,52	80	0,47	96	0,52	112	0,59	128	0,52

Кесте Қ.12 – EMCipher алгоритмнің 14 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,49	33	0,48	49	0,55	65	0,53	81	0,55	97	0,47	113	0,49
2	0,50	18	0,53	34	0,59	50	0,49	66	0,59	82	0,59	98	0,52	114	0,56
3	0,44	19	0,52	35	0,46	51	0,53	67	0,42	83	0,57	99	0,48	115	0,48
4	0,51	20	0,46	36	0,53	52	0,43	68	0,43	84	0,52	100	0,45	116	0,53
5	0,59	21	0,50	37	0,50	53	0,56	69	0,56	85	0,45	101	0,49	117	0,48
6	0,59	22	0,41	38	0,59	54	0,47	70	0,45	86	0,52	102	0,50	118	0,55
7	0,49	23	0,45	39	0,47	55	0,45	71	0,48	87	0,57	103	0,45	119	0,52
8	0,52	24	0,52	40	0,50	56	0,48	72	0,57	88	0,49	104	0,46	120	0,48
9	0,50	25	0,47	41	0,48	57	0,60	73	0,45	89	0,59	105	0,53	121	0,52
10	0,46	26	0,48	42	0,47	58	0,48	74	0,52	90	0,53	106	0,49	122	0,53
11	0,48	27	0,48	43	0,50	59	0,52	75	0,47	91	0,49	107	0,45	123	0,57
12	0,50	28	0,54	44	0,51	60	0,57	76	0,58	92	0,50	108	0,58	124	0,56
13	0,47	29	0,55	45	0,53	61	0,49	77	0,54	93	0,48	109	0,53	125	0,48
14	0,52	30	0,48	46	0,44	62	0,50	78	0,48	94	0,63	110	0,55	126	0,54
15	0,52	31	0,52	47	0,56	63	0,41	79	0,48	95	0,57	111	0,55	127	0,49
16	0,50	32	0,52	48	0,52	64	0,51	80	0,48	96	0,45	112	0,49	128	0,52

Кесте Қ.13 – EMCipher алгоритмнің 15 раундтағы биттік шашырау нәтижесі

i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i	i	k_i
1	0,54	17	0,50	33	0,51	49	0,54	65	0,54	81	0,52	97	0,51	113	0,49
2	0,49	18	0,45	34	0,50	50	0,50	66	0,57	82	0,50	98	0,48	114	0,56
3	0,50	19	0,41	35	0,48	51	0,48	67	0,51	83	0,48	99	0,52	115	0,48
4	0,55	20	0,54	36	0,55	52	0,41	68	0,47	84	0,52	100	0,57	116	0,53
5	0,52	21	0,50	37	0,50	53	0,56	69	0,52	85	0,46	101	0,53	117	0,48
6	0,51	22	0,48	38	0,48	54	0,40	70	0,56	86	0,50	102	0,49	118	0,55
7	0,49	23	0,50	39	0,48	55	0,45	71	0,45	87	0,57	103	0,48	119	0,52
8	0,50	24	0,52	40	0,48	56	0,54	72	0,45	88	0,52	104	0,42	120	0,48
9	0,50	25	0,45	41	0,49	57	0,40	73	0,46	89	0,60	105	0,42	121	0,52
10	0,45	26	0,52	42	0,52	58	0,47	74	0,50	90	0,52	106	0,43	122	0,53
11	0,52	27	0,45	43	0,49	59	0,54	75	0,55	91	0,52	107	0,52	123	0,57
12	0,52	28	0,48	44	0,56	60	0,52	76	0,56	92	0,55	108	0,54	124	0,56
13	0,52	29	0,57	45	0,45	61	0,52	77	0,48	93	0,52	109	0,53	125	0,48
14	0,50	30	0,42	46	0,55	62	0,48	78	0,48	94	0,52	110	0,49	126	0,54
15	0,42	31	0,45	47	0,49	63	0,48	79	0,52	95	0,50	111	0,53	127	0,49
16	0,47	32	0,52	48	0,48	64	0,51	80	0,52	96	0,45	112	0,45	128	0,52



Сурет Қ.1- EMCipher түрлендіру әдісінің кері

Кесте Қ.14 – Дәрежені есептеу жылдамдығын салыстыру кестесі

Файлдың түрі	**Өлшемі**	**Дәрежені есептеу (модуль бойынша)**	**Индекс бойынша есептеу**	**Компьютерлердің сипаттамасы**
Pdf	13мб	5мин, 50 сек	35сек	Intel(R) Core(TM) i7-3770 CPU @3.40GHz 3.40GHz, ОЗУ 8,00ГБ, Windows 8 pro, 64bit
Pdf	13мб	7 мин	1 мин, 5 сек	Intel(R) Core(TM) i7-8700 CPU @3.20GHz 3.19GHz, ОЗУ 32,00ГБ, Windows 10 pro, 64bit
Pdf	15,8мб	7 мин, 30сек	55 сек	Intel(R) Core(TM) i7-3770 CPU @3.40GHz 3.40GHz, ОЗУ 8,00ГБ, Windows 8 pro, 64bit
rar	10.8мб	4мин,8сек	1 мин	Intel(R) Core(TM) i7-8700 CPU @3.20GHz 3.19GHz, ОЗУ 16,00ГБ, Windows 10 pro, 64bit

Кесте Қ.15 – Раундттық кілтті жасау алгоритміне қолданылатын S- блок алмастыру кестесі.

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	202	192	142	206	236	235	218	112	107	158	126	9	187	72	146	10
2	166	135	157	99	198	180	33	242	57	26	22	98	205	241	36	213
3	25	11	173	194	152	68	230	165	154	82	108	175	212	18	78	168
4	229	184	85	93	5	207	231	174	223	87	75	143	197	169	238	253
5	80	122	37	222	92	14	138	226	137	255	70	240	47	144	28	44
6	141	211	35	228	179	16	88	34	239	246	21	127	2	254	77	181
7	42	183	60	61	54	115	118	81	113	96	219	123	46	155	89	41
8	170	243	50	95	19	69	237	224	159	117	76	190	111	178	27	29
9	39	200	214	4	196	162	171	248	119	90	52	101	252	91	63	32
10	249	124	31	49	66	220	74	132	128	172	201	221	65	193	133	139
11	233	204	250	97	208	62	43	188	121	56	17	83	103	234	209	53
12	110	185	94	24	0	232	199	191	100	247	30	58	7	217	109	164
13	145	23	105	136	244	3	245	8	176	13	151	45	134	150	38	195
14	147	1	227	130	186	67	215	15	129	167	140	216	102	225	148	48
15	73	153	79	163	160	189	114	125	20	116	71	251	106	149	59	12
16	156	104	131	177	6	210	40	161	182	55	120	51	84	86	64	203

ҚОСЫМША В

УТВЕРЖДАЮ
Зам. генерального директора
Института информационных
и вычислительных технологий
PhD доктор
_____ Мамырбаев О.Ж.
«____» _____ 2020

АКТ
о внедрении результатов диссертационной работы
Хомпыш Ардабек

Экспертная комиссия «Института информационных и вычислительных технологий» (ИИВТ) Комитета науки МОН РК в состав:

председатель: д.т.н., ассоциированного профессора, заместитель генерального директора по связям с общественностью Калижанова А.У;

члены:

д.т.н., ассоциированный профессор, ГНС ИИВТ Нысанбаева С.Е.;
к.т.н., ВНС ИИВТ Капалова Н.А.;
к.т.н., доцент, ученый секретарь ИИВТ Юничева Н.Р.

составили настоящий акт о том, что результаты диссертационной работы «Разработка и исследование алгоритма защиты информации с использованием непозиционных систем счисления» МНС ЛИБ ИИВТ Хомпыш А. были получены при выполнении проектов РГП на ПХВ «ИИВТ» КН МОН РК, № гос. регистрации - 0118РК01064), источник финансирования Комитет науки МОН РК:

– программно-целевого финансировании (ПЦФ) КН МОН РК «Разработка программных и программно-аппаратных средств для криптографической защиты информации при ее передаче и хранении в инфокоммуникационных системах и сетях общего назначения» на 2018-2020 годы.

Результаты включены в отчёты проектов ПЦФ за 2019 - 2020 годы.

Краткое содержание внедренных результатов:

1. разработан алгоритм симметричного блочного шифрования с использованием метода преобразование ЕМ;
2. разработана таблица замены (S-блок), удовлетворяющая требованиям криптоанализа;
3. разработан алгоритм генерации раундовых ключей;
4. программно реализован алгоритм симметричного блочного шифрования с использованием метода преобразования ЕМ.

Материалы к настоящему акту были рассмотрены на Ученом Совете института (протокол № 9 от 21 июля 2020 год).

Председатель комиссии _____ Калижанова А.У.

Члены комиссии _____ Нысанбаева С.Е.

_____ Капалова Н.А.

_____ Юничева Н.Р.