ივანე ჯავახიშვილის სახელობის თბილისის სახელმწიფო უნივერსიტეტი

ზუსტ და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტი
კომპიუტერულ მეცნიერებათა დეპარტამენტი

ზურაბ ქოჩლაძე

ლექციების კონსპექტი საგანში
"ინფორმაციული უსაფრთხოების ტექნოლოგიები"

2012

ლექცია 1.

ინფორმაციის უსაფრთხოების პრობლემა და დაცვის კრიპტოგრაფიული ტექნოლოგიები.

1.1. თანამედროვე ინფორმაციული ტექნოლოგიები. მეოცე საუკუნის ბოლოს მსოფლიოში დაიწყო ციფრული ინფორმაციის ბუმი. თანამედროვე კომპიუტერული ტექნიკის სწრაფი განვითარების, ელექტრონული საკომუნიკაციო არხების, გლობალური და კორპორატიული კომპიუტერული ქსელების შექმნისა და განვითარების შედეგად დღეს ფაქტობრივად ინფორმაციის ძირითადი მატარებელი გახდა ელექტრონული სიგნალი. ციფრული სახით წარმოდგენილი ინფორმაცია თავისუფლად მოძრაობს მთელს მსოფლიოში და შესაძლებელია ამ ინფორმაციის გადაცემა მსოფლიოს ნებისმიერი წერტილიდან ნებისმიერში წამების განმავლობაში. თანამედროვე კომპიუტერული სისტემები და ინფორმაციული ტექნოლოგიები, რომლებიც საშუალებას გვაძლევენ სწრაფად დავამუშავოთ უზარმაზარი ინფორმაციული ნაკადები, ფართოდ ინერგება მართვის ნებისმიერ დონეზე, დაწყებული ფირმიდან, დამთავრებული სახელმწიფოთი.

საინფორმაციო გამოთვლით სისტემებსა და ღია კომპიუტერულ ქსელებში ინფორმაციის დამუშავების და გადაცემის პროცესები აუცილებლად მოითხოვენ ინფორმაციის დაცვას არასანქცირებული წვდომისაგან, რომლის შედეგიც შეიძლება იყოს კონფიდენციალური ინფორმაციის გაჟონვა, გადაცემული ინფორმაციის დამახინჯება და შესაძლოა მისი განადგურებაც. რა თქმა უნდა ინფორმაციის დაცვა არასანქცირებული წვდომისა თუ მისი განადგურებისაგან, ყოველთვის იყო აქტუალური პრობლემა, მაგრამ ის ტექნოლოგიები, რომლებიც გამოიყენებოდა ამ მიზნით მაშინ, როდესაც ინფორმაციის მატარებელი იყო ქაღალდი, დღეს უკვე აღარ აღმოჩნდა საკმარისი. ამის მიზეზი პირველ რიგში არის ის, რომ ციფრული სახით წარმოდგენილი ინფორმაციის მიტაცება, განადგურება, თუ წინასწარი განზრახვით დამახინჯება, აღმოჩნდა გაცილებით იოლი ვიდრე ქაღალდზე შენახული ინფორმაციის. ამასთან ერთად, თანამედროვე კომპიუტერული ტექნოლოგიების პირობებში ინფორმაციის დაცვის ძველი ტექნოლოგიების გამოყენებამ აზრი დაკარგა.

1.2. ინფორმაციული უსაფრთხოების ცნება. ინფორმაციული უსაფრთხოების ცნება სხვადასხვა კონტექსტში შეიძლება სხვადასხვა რამეს ნიშნავდეს. მაგალითად, სახელმწიფო დონეზე მას აქვს ფართო გაგება და იგულისხმება თუ როგორაა დაცული ინფორმაციის სფეროში ნაციონალური ინტერესები, რომლებიც შედგება პიროვნების, საზოგადოებისა და სახელმწიფოს დაბალანსებული ინტერესებისაგან.

ჩვენ ლექციათა კურსში ძირითად ყურადღებას დავუთმობთ ინფორმაციის უსაფრთხოებას მისი დამუშავების, შენახვისა და გადაცემის დროს. ამიტომ ამ შემთხვევაში ინფორმაციული უსაფრთხოების დაცვის ქვეშ ვიგულისხმებთ იმ ტექნოლოგიების ერთობლიობას, რომლებიც არ აძლევენ საშუალებას არასერთიფიცირებულ მომხმარებელს განახორციელოს წვდომა ინფორმაციასთან. ინფორმაციის დაცვა - ესაა ღონისძიებათა მთელი კომპლექსი, რომელიც მიმართულია ინფორმაციული უსაფრთხოების უზრუნველსაყოფად. აქედან გამომდინარე, პირველ რიგში უნდა დაზუსტდეს ინფორმაციულ პროცესში მონაწილე სუბიექტები და მათი ის ინტერესები, რომლებიც დაკავშირებულია ინფორმაციული სისტემების გამოყენებასთან. განისაზღვროს მათი უფლება-მოვალეობები და პროცედურები, რომელთა საშუალებითაც ამ სუბიექტებს შეუძლიათ განახორციელონ ურთიერთქმედა ინფორმაციულ სისტემასთან.

ყოველივე აქედან გამომდინარე შეიძლება დავასკვნათ, რომ

- სხვადასხვა სისტემებისა თუ სუბიექტებისათვის პრობლემები, დაკავშირებული ინფორმაციულ უსაფრთხოებასთან, შეიძლება იყოს სხვადასხვა. ამიტომ მათ მიერ განხორციელებული ღონისძიებებიც შესაძლებელია განსხვავდებოდეს ერთმანეთისაგან.
- ინფორმაციული უსაფრთხოება არაა მარტო თავის დაცვა არასანქცირებული წვდომისაგან. სუბიექტი შეიძლება დაზარალდეს მაშინაც, როდესაც საინფორმაციო სისტემაში მოხდება რაიმე შეფერხება.

- ინფორმაციული უსაფრთხოება დამოკიდებულია არა მარტო კომპიუტერების გამართულ და უსაფრთხო მუშაობაზე, არამედ აგრეთვე იმ ინფრასტრუქტურაზეც, რომელშიც მუშაობენ ეს კომპიუტერები (ელექტროკვების წყარო, წყალმომარაგება, კონდიციონერები და ა.შ.) ყველაზე უფრო მნიშვნელოვნად კი ის დამოკიდებულია იმ ადამიანებზე, რომლებიც ემსახურებიან უსაფრთხოების დაცვას. როგორც გვიჩვენებენ ისტორიული მაგალითები, ადამიანი ყოველთვის წარმოადგენს ამ სისტემის ყველაზე სუსტ რგოლს.

ამგვარად, ჩვენ ლექციათა კურსში ინფორმაციული უსაფრთხოების ქვეშ შემდეგში ყოველთვის ვიგულისხმებთ ინფორმაციის დაცულობას შემთხვევითი ან წინასწარგანზრახული ზემოქმედებისაგან, რომლის შედეგადაც ზარალდებიან ინფორმაციული ურთიერთობის სუბიექტები. ძალიან ხშირად ინფორმაციული უსაფრთხოება ესმით გამარტივებულად და დაყავთ მხოლოდ პროგრამის უსაფრთხოებაზე. ამბობენ, რომ "ეს პროგრამა გარანტირებულად დაიცავს თქვენს კომპიუტერს, ან ეს სისტემა დაიცავს ელექტრონულ კომერციას". ასეთ შემთხვევაში პირველ რიგში აუცილებლად უნდა დავაზუსტოთ, თუ რისგან დაგვიცავს ეს პროგრამა (ან სისტემა)? დაგვიცავს ის შეიარაღებული თავდასხმისაგან საინფორმაციო სისტემაზე ან ვიდოკამერისაგან, რომელიც დამიზნებულია მონიტორზე? ალბათ არა (ყოველ შემთხვევაში კრიპტოგფარიული ალგორითმები ნამდვილად არ იძლევიან ამის შესაძლებლობას) და არა იმიტომ, რომ ეს სისტემა არ ვარგა, არამედ იმიტომ, რომ ის შექმნილია სხვა ტიპის შეტევებისაგან თავის დასაცავად, ამიტომ ის უნდა გამოვიყენოთ მხოლოდ დანიშნულების მიხედვით. სწორედ ამიტომ ვამბობთ, რომ ინფორმაციული უსაფრთხოების პრობლემა წარმოადგენს კომპლექსურ პრობლემას და მისი გადაწყვეტა დაკავშირებულია სხვადასხვა სახის (იურიდიული, ორგანიზაციული, ფიზიკური, ტექნიკური და ტექნოლოგიური) საკითხების გადაწყვეტასთან. ამიტომ, როდესაც ვამბობთ, რომ ესა თუ ის საინფორმაციო სისტემა უსაფრთხოა, კარგად უნდა გვესმოდეს რა ტიპის შეტევებისაგანაა ის უსაფრთხო.

<u>1.3. ვირტუალური სამყაროს შესაძლებლობები.</u> შეტევები, რომლთა განხორციელებაც შესაძლებელია კიბერსივრცეში, არაფრით არ განსხვავდება იმ შეტევებისაგან, რომლებიც ყოველდღიურად ხდება რეალურ სამყაროში. თუ ძარცვავენ ფიზიკურ ბანკებს, გაძარცვავენ ციფრულ ბანკებსაც, თუ პაპარაცები ცდილობენ შეიჭრან პოპულარული ადამიანების პირად ცხოვრებაში ფოტოკამერის საშუალებით, იგივეს აკეთებენ ხაკერები, რომლებიც ცდილობენ მიიტაცონ ეს ინფორმაცია ქსელიდან. კანონდარღვევები კიბერსივრცეში ერთი ერთზე იმეორებენ კანინდარღვევებს რეალურ სამყაროში, მაგრამ კიბერსივრცეში ყველაფერი სახეშეცვლილია. მათი განხორციელება ფიზიკურ სამყაროსთან შედარებით უფრო ადვილია, მასშტაბები გაცილებით დიდია და დამნაშავეთა აღმოჩენა და დასჯა გაცილებით ძნელი. ამის მიზეზია შეტევათა ავტომატიზაცია, ქმედება შორიდან და ტექნიკური ხერხების გავრცელება.

გასული საუკუნის სამოციან წლებში ამერიკაში გავრცელებული იყო წვრილმანი თაღლითობა, რაც გამოიხატებოდა იმაში, რომ ხულიგნები ახერხებდნენ უფასოდ დაერეკათ ტელეფონ-ავტომატიდან ზოგიერთ რაიონებში. რა თქმა უნდა სატელფონი კომპანიები უკმაყოფილონი იყვნენ ამით, მაგრამ რა ზარალი შეიძლებოდა მიეყენებინა ამ ქმედებას კომპანიისათვის, რომლის შემოსავლები რამდენიმე მილიარდს აღწევდა, თუ ერთი სატელეფონო ზარის განხორციელება ღირდა ათი ცენტი, ხოლო ზარის თვითღირებულება თითქმის ნოლის ტოლი იყო. წარმოიდგინეთ რამდენჯერ უნდა დაერეკათ ასეთი ხერხით, რომ კომპანიის ზარალი ოდნავ მაინც საგრძნობი ყოფილიყო. კიბერსივრცეში კი ყველაფერი სხვანაირადაა. კომპიუტერებს გააჩნიათ საგრძნობი უპირატესობა ასეთი ერთფეროვანი, მოსაწყენი ოპერაციების მრავალჯერადი გამეორების დროს. თქვენ შეგიძლიათ შექმნათ პროგრამა, რომელიც ბანკის მიერ გადარიცხული ყოველი ანგარიშიდან თქვენს ანგარიშზე გადარიცხავს მხოლოდ ერთ ცენტს და შემდეგ აღარ დაგჭირდებათ ყოველ წუთს მისი ხელახალი გაშვება. სანამ ბანკი შეამჩნევს დაკარგულ თანხებს, თქვენს ანგარიშზე უკვე დაგროვდება საკმაოდ სოლიდური თანხა, საჭირო იქნება მხოლოდ დროულად შეწყვიტოთ პროგრამის მუშაობა (არ უნდა დაგბლიოთ სიხარბემ). ასეთი რამის გაკეთება კომპიუ-ტერის გარეშე წარმოუდგენელია.

დავუშვათ თაღლითმა მოიგონა მსხვილი აფიორის სქემა, რომლის შედეგადაც ის მიიღებს ძალიან დიდ თანხას, მაგრამ ალბათობა იმისა, რომ ეს აფიორა მოიტანს შედეგს არის ერთი ასიათასიდან. სანამ თაღლითი შეძლებს ასეთი შანსის მოძებნას რეალურ სამყაროში, მას შეიძლება შიმშილისაგან სული გასძვრეს, მაშინ, როდესაც კიბერსივრცეში მას შეუძლია მხოლოდ დააპროგრამოს თავისი კომპიუტერი ასეთი შანსის მოძებნაზე და დარწმუნებული იყავით, კომპიუტერს არ დასჭირდება ერთი დღეც კი, რომ აღმოაჩინოს ამ სქემის განხორციელების რამდენიმე ათეული

ვარიანტი. თუ თაღლითი შეძლებს, რომ ძებნაში ჩართოს სხვა კომპიუტერებიც, დარწმუნებული იყავით, რომ სულ რაღაც ერთი საათის განმავლობაში მას ექნება ათასობით ვარიანტი.

პროცესების ავტომატიზაცია, რომელიც შეუძლებელია რეალურ სამყაროში, მაგრამ ადვილად ხორციელდება კიბერსივრცეში, შესაძლებელს ხდის ისეთ შეტევებსაც კი, რომელთა განხორციელება რეალურ სამყაროში შეუძლებელი იყო. ესაა პირველი არსებითი განსხვავება რეალურ სამყაროსა და კიბერსივრცეში განხორციელებულ შეტევებს შორის.

პროცესების ასეთი ავტომატიზაცია კიბერსივრცეში, ძალიან უწყობს ხელს აგრეთვე კონფიდენციალობის დარღვევას. დავუშვათ, რომელიმე კომპანია ატარებს მარკეტინგულ კვლევას, რომლის მიზანიცაა დაადგინოს იმ მდიდარი მშობლების ვინაობა, რომლებსაც უყვართ თავიაანთი შვილები, ხოლო თავის მხრივ შვილები აგროვებენ მარკებს პინგვინების გამოსახულებებით. ინტერნეტის გარეშე ასეთი ინფორმაციის შეგროვება ძალიან ძნელია. კომპანიას მოუწევდა დაექირავებინა რამდენიმე ათასი თანამშრომელი, რომლებსაც მოუწევდათ კარდაკარ სიარული ასეთი ინფორმაციის შესაგროვებლად. კომპიუტერულ ქსელში კი უმარტივესად შეიძლება ამის გაგება. ამისათვის საჭიროა საფოსტო მარკების მარკეტინგული ბაზიდან ამოკრიფოთ ადამიანები გარკვეული წლიური შემოსავლით, დაადგინოთ მათი დაბადების თარიღები, შეადაროთ მონაცემთა ბაზას იმ ადამიანებისა, რომლებიც იწერენ **rec.collecting.stamps**-ს და შეადაროთ **Amazon.cim**-ის იმ მყიდველთა სიას, რომლებიც ყიდულობენ წიგნებს პინგვინების შესახებ. ინტერნეტში არსებობს საშუალება შეაგროვოთ მონაცემები ნებისმიერ ადამიანზე, რომელიც ერთხელ მაინც მოხვდა მომხმარებელთა ქსელში. ქაღალდზე შენახული ინფორმაცია, მაშინაც კი როდესაც ის ყველასთვის ხელმისაწვდომია, ძნელია მოაგროვო და შეუსაბამო ერთმანეთს, მაშინ, როდესაც ქსელში განთავსებული ინფორმაცია ხელმისაწვდომია ნებისმიერი წერტილიდან. ზოგიერთ შემთხვევაში ასეთი ინფორმაცია მოპოვებულია არაკანონიერი გზით და ამისათვის კერძო პირებს ხშირად დევნიან სასამართლოს გზით, როდესაც ისინი ცდილობენ წაიკითხონ პოლიციის საიდუმლო ფაილები. სხვა შემთხვევებში ასეთ ქმედებას უწოდებენ მონაცემთა მოპოვებას. მაგალითად, საკრედიტო ბარათების კომპანიები ფლობენ უამრავ ინფორმაციას ყველა მომხმარებელზე. ამ ინფორმაციას კრებენ, ახარისხებენ და შემდეგ ყიდიან ნებისმიერ პირზე ან კომპანიაზე, რომელიც კი გამოთქვამს სურვილს ასეთი ინფორმაციის შეძენაზე.

როგორც ხშირად ამბობენ სპეციალისტები თუ უბრალო მომხმარებლები, ინტერნეტს არ გააჩნია საზღვრები და შეზღუდვები. ერთნაირად იოლია (ან ზოგჯერ ძნელია) დაუკავშირდეთ აბონენტს, რომელიც ზის თქვენსავე ოთახში და აბონენტს, რომელიც თქვენგან დაშორებულია რამდენიმე ათასი კილომეტრით. ეს წარმოშობს კიდევ ახალ სიძნელეებს კომპიუტერული დანაშაულებებთან ბრძოლაში. იმისათვის, რომ შეუტიოთ **Bank of America**-ს, სულაც არაა აუცილებელი იჯდეთ ნიუ-იორკში, იგივე შეტევა თქვენ შეგიძლიათ განახორციელოთ თბილისიდანაც. ეს რა თქმა უნდა ართულებს უსაფრთხოების პრობლემას, თუ ადრე საკმარისი იყო თავის დაცვა მხოლოდ ადგილობრივი (ქვეყნის მასშტაბით) დამნაშავეებისაგან, ეხლა საჭირო ხდება გავითვალისწინოთ მთელი მსოფლიოს დამნაშავეთა სამყარო. ეს ართულებს როგორც დამნაშავეების ძებნას, ასევე შემდგომ მათ დასჯას, რადგანაც სხვადასხვა ქვეყნების კანონმდებლობა ხშირ შემთხვევაში სხვადასხვანაირად აფასებს დანაშაულს, ჩადენილს ინტერნეტის საშუალებით. თუმცა უნდა აღინიშნოს, რომ დღეისათვის ამ დანაშაულებთა არეალი ისე გაფართოვდა, რომ სულ უფრო მეტი ქვეყნები ცდილობენ გამოიმუშაონ ერთნაირი კანონები და უფრო მჭიდროდ ითანამშრომლონ, რათა როგორმე შეებრძოლონ ამ დანაშაულებებს.

მესამე თვისება, რომლითაც კიბერსივრცე განასხვავებდა რეალური სამყაროსაგან, არის წარმატებული გამოცდილების სწრაფი გავრცელების საშუალება. საკაბელო ტელევიზიების მფლობელებს მაინც და მაინც არ შეაწუხებთ, თუ რომელიმე კარგი სპეციალისტი შექმნის დეშიფრატორს და შეძლებს უყუროს მათ გადაცემებს. მაგრამ რა მოხდება მაშინ, თუ ეს სპეციალისტი დეშიფრატორის აღწერას და სქემას გაავრცელებს ინტერნეტის საშუალებით? მაშინ საკაბელო ტელევიზიას მოუწევს შექმნას ახალი, უფრო რთული დაშიფრვის სქემა, რაც არც თუ ისე ადვილია და არც ისე იაფი დაჯდება. რა თქმა უნდა ასეთი რამ შესაძლებელია რეალურ სამყაროშიც, მაგრამ აქაც საქმე გვაქვს მასშტაბებთან. მოვიყვანოთ მაგალითი. მაშინ, როდესაც ირანს მართავდა შახი, რომელიც აშშ დიდი მეგობარი იყო, აშშ მთავრობამ ირანს მიყიდა ღრმა ბეჭდვის დაზგები ირანული ფულის დასაბეჭდად. ისლამური რევოლუციის შემდეგ ირანის მთავრობაში გადაწყვიტეს, რომ რიალების ბეჭდვას, სჯობია ამერიკული ასდოლარიანების ბეჭდვა, მით უმეტეს, რომ ხარისხი იმდენად მაღალი იყო, რომ ყალბი ასდოლარიანების გარჩევა ნამდვილისაგან თითქმის შეუძლებელი იყო. (ამიტომ შეიცვალა აშშ ფულის დიზაინი და დაცულობის ხარისხი). სანამ ფედერალური გამოძიების ბიუროში თავს იმტვრევდნენ რა გაეკეთებინათ, ფინანსთა სამინისტროში სწრაფად დათვალეს, თუ

რა რაოდენობის ყალბი ფულის დაბეჭდვა შეეძლო ირანს წლის განმავლობაში და დაასკვნეს, რომ ყალბი ფულის ეს მასა გავლენას ვერ იქონიებდა დოლარის სტაბილურობაზე და ამიტომ არ წარმოადგენდა სერიოზულ პრობლემას. მაგრამ თუ ამ ყალბი ფულის ბეჭდვაში გამოყენებული ირანი შეძლებდა ელექტრონული საშუალებების გამოყენებას, ყველაფერი სხვანაირად იქნებოდა. ელექტრონული ყალბი ფულის მჭრელი განათავსებს პროგრამას რომელიმე ვებ-გვერდზე და ალბათ ყველასათვის გასაგებია რაც მოყვება ამას. ინტერნეტი წარმოადგენს ყველაზე უფრო სრულყოფილ სივრცეს ასეთი შეტევების გასავრცელებლად. მხოლოდ პირველი თავდამსხმელი უნდა იყოს გამომგონებელი და დახარჯოს დრო და რესურსები პროგრამის შესაქმნელად. ამის შემდეგ ის განათავსებს ამ პროგრამას ისეთ ვებ-გვერდზე, რომ ის ადვილად შეამჩნიონ და შემდეგ ყველაფერი მოხდება ავტომატურად. ყველა დანარჩენი დამნაშავისაგან მოითხოვება ერთადერთი რამ, გაუშვას პროგრამა როგორც ბოთლიდან გამოშვებული ჯინი, მაგრამ ჯინისაგან განსხვავებით შემდგომი პროცესები უკვე აღარ დაემორჩილება თვით ავტორსაც და იარსებებს სანამ არ იპოვნიან მის საწინააღმდეგო საშუალებას. ამის ყველაზე თვალსაჩინო მაგალითია ვირუსები, რომლებიც მოდებულია მთელ ინტერნეტში.

<u>1.4. ინფორმაციული უსაფრთხოების ამოცანები.</u> ინფორმაციული უსაფრთხოების ძირითად ამოცანებს წარმოადგენს

- ინფორმაციის კონფიდენციალურობის უზრუნველყოფა;
- ინფორმაციის მთლიანობის უზრუნველყოფა;
- ინფორმაციის ნამდვილობის უზრუნველყოფა;
- ინფორმაციასთან წვდომის ოპერატიულობის უზრუნველყოფა;
- ელექტრონული დოკუმენტის სახით წარმოდგენილი ინფორმაციის იურუდიული მნიშვნელობის უზრუნველყოფა;
- კლიენტის ქმედებათა თვალთვალის შეუძლებლობის უზრუნველყოფა.

კონფიდენციალობა არის ინფრომაციის თვისება ხელმისაწვდომი იყოს მხოლოდ გარკვეული ჯგუფის მომხმარებლებისათვის.

მთლიანობის ქვეშ იგულისხმება ინფორმაციის თვისება შეინარჩუნოს თავისი სტრუქტურა და შინაარსი გადაცემის ან შენახვის პროცესში.

ნამდვილობა ინფორმაციის თვისებაა, რომელიც გამოიხატება იმაში, რომ ზუსტად შეგვიძლია დავადგინოთ ვინ არის ინფორმაციის წყარო, ან ვისგან მივიღეთ ინფორმაცია.

ოპერატულობა ნიშნავს, რომ ინფორმაცია მომხმარებილისთვის ხელმისაწვდომია დროის იმ მომენტში, რომელშიც მომხმარებელი ითხოვს ინფორმაციას.

იურიდიული მნიშვნელობა ნიშნავს, რომ ელექტრონული სახით წარმოდგენილ დოკუმენტს გააჩნია იურიდიული ძალა. ამისათვის, სუბიექტები, რომლებიც მონაწილეობენ ასეთი სახის ინფორმაციის გაცვლაში, წინასწარ თანხმდებიან ერთმანეთში, თუ რა სახის ფორმით უნდა იყოს წარმოდგენილი დოკუმენტი და რა ატრიბუტები უნდა ახლდეს მას, რათა ჩაითვალოს იურიდიული ძალის მქონე დოკუმენტად. ასეთ შეთანხმებას შეიძლება ჰქონდეს კანონის სახე, როგორც ესაა მიღებული მრავალ ქვეყანაში. თუ ქვეყანაში არსებობს კანონი ელექტრონული ხელმოწერის შესახებ, მაშინ იურიდიული ძალა ექნება მხოლოდ იმ დოკუმენტს, რომელიც შესრულებულია ამ კანონის სრული დაცვით. ასეთი დოკუმენტები ძირითადად გამოიყენება ელექტრონული ანგარიშსწორებისა და მართვის პროცესებში.

თვალთვალის შეუძლებლობა ნიშნავს, რომ კლიენტს სისტემაში შეუძლია ქმედებების შესრულება სხვებისაგან შეუმჩნევლად. ეს ამოცანა აქტუალური გახდა ელექტრონული ანგარიშსწორებისა და **Internet-bancingis** ცნებების გამო. მაგალითად, ელექტრონული ანგარიშსწორების დროს კლიენტმა უნდა წარმოადგინოს ინფორმაცია, რომელიც ახდენს მის იდენტიფიკაციას, ამიტომ ცხადია, რომ ასეთი ინფორმაცია არ შეიძლება გახდეს ცნობილი სხვა სუბიექტებისათვის.

<u>1.5. ინფორმაციული უსაფრთხოების ტექნოლოგიები.</u> ინფორმაციული უსაფრთხოების საფუძველს წარმოადგენს ინფორმაციის დაცვის კრიპტოგრაფიული მეთოდები და საშუალებები, მაგრამ აუცილებლად უნდა აღვნიშნოთ, რომ დაცვა იქნება ყველაზე საიმედო მხოლოდ კომპლექსური მიდგომის დროს, ანუ როდესაც კრიპტოგრაფიულ საშუალებებთან ერთად გამოყენებული იქნება უსაფრთხოების დაცვის ორგანიზაციული და ტექნიკური საშუალებებიც.

კრიპტოგრაფიული მეთოდების საფუძველს წარმოადგენს ინფორმაციის კრიპტოგრაფიული გარდაქმნა, რომელიც ხორციელდება გარკვეული მათემატიკური გარდაქმნებით და რომლის მიზანია გამორიცხოს ამ ინფორმაციასთან არასერთიფიცირებული მომხმარებლის წვდომა და აგრეთვე ასეთი მომხმარებლის მიერ ინფორმაციის შეცვლა.

კრიპტოგრაფიული მეთოდების გამოყენება უზრუნველყოფს ინფორმაციული უსაფრთხოების დაცვის ისეთი ამოცანების გადაჭრას, როგორიცაა

- ღია ქსელში გადაცემული ინფორმაციული ტრაფიკის დაშიფრვა;
- ტრაფიკის იმიტოდაცვა (ანუ გადაცემული ინფორმაციის დაცვა შეცვლისაგან);
- გადაცემული ან მონაცემთა ბაზაში შენახული მონაცემების დაშიფრვა;
- ინფორმაციული ურთიერთობის სუბიექტების აუთენტიფიკაცია;
- გადაცემული დოკუმენტისათვის იურიდიული ძალის მინიჭება ელექტრონულ-ციფრული ხელმოწერის საშუალებით.

საკონტროლო კითხვები:

1. განმარტეთ რას ნიშნავს ინფორმაციული უსაფრთხოება.
2. რამ გამოიწვია ინფორმაციული უსაფრთხოების ახალი მეთოდების განვითარების აუცილებლობა?
3. რას ნიშნავს, რომ ინფორმაციული უსაფრთხოების დაცვა წარმოადგენს კომპლექსურ პრობლემას?
4. როგორ გაიზარდა საფრთხეები ვირტუალურ სამყაროში?
5. ჩამოთვალეთ ინფორმაციული უსაფრთხოების ამოცანები.
6. რა როლი ენიჭება კრიპტოგრაფიას ინფორმაციული უსაფრთხოების დაცვაში?

უსაფრთხოების სამი ძირითადი შემადგენელი ნაწილი: კონფიდენციალურობა, მთლიანობა და აუთენტიფიკაცია

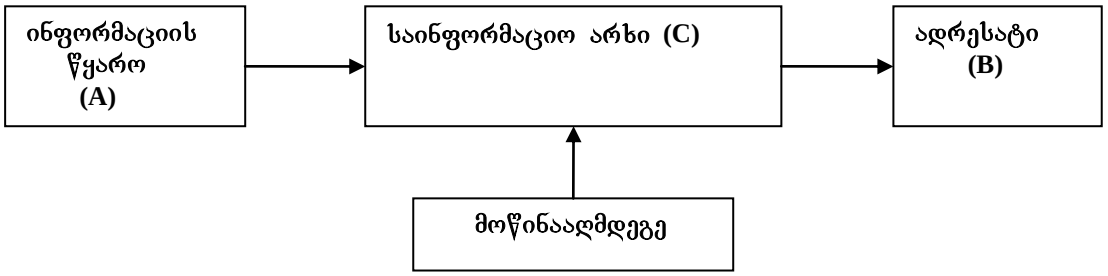
2.1 კონფიდენციალურობა. როგორ გადავცეთ რაიმე შეტყობინება ფიზიკურად დაშორებულ აბონენტს ისე, რომ მის გარდა ვერავინ შეძლოს ამ ინფორმაციის წაკითხვა? ეს პრობლემა დადგა კაცობრიობის წინაშე მას შემდეგ, რაც მან ისწავლა ინფორმაციის ჩაწერა ინფორმაციის რაიმე (თიხის ფირფიტა, გასანთლული ფიცარი, პაპირუსი, ქაღალდი და ა.შ.) ფიზიკურ მატარებელზე. დღეს ამ პრობლემას აყალიბებენ ცოტა უფრო სხვანაირად: როგორ დავიცვათ ინფორმაციის ***კონფიდენციალურობა*** საკომუნიკაციო არხებში მოძრაობის დროს? მიუხედავად ამისა შინაარსი არ შეცვლილა, ისევაც ლაპარაკია იმაზე, რომ ინფორმაცია გადავცეთ ფიზიკურად დაშორებულ აბონენტს ისე, რომ სხვამ ვერავინ შეძლოს მისი წაკითხვა.

ინფორმაციის გადაცემის პროცესი შეიძლება წარმოვადგინოთ შემდეგი სქემის სახით (იხ. სურ. 2.1): გვაქვს ***ინფორმაციის წყარო*** **(A),** რომელიც გამოიმუშავებს ინფორმაციას, ***საინფორმაციო არხი*** **(C),** რომელსაც შემდეგში შემოკლებით ვუწოდებთ ***არხს*** და რომლის საშუალებითაც ხდება ინფორმაციის გადაცემა ერთი ადგილიდან მეორეზე და ***ადრესატი*** **(B),** ვისთვისაც განკუთვნილია ეს ინფორმაცია. ამავე დროს არსებობს ***მოწინააღმდეგე (მტერი),*** რომელიც ცდილობს ამ ინფორმაციის ხელში ჩაგდებას.

თუ გამოვრიცხავთ იმ შემთხვევებს, როდესაც ან ინფორმაციის წყარო, ან ადრესატი თვითონ უწყობენ ხელს მოწინააღმდეგეს ინფორმაციის მოპოვებაში, ან მათზე ხდება ძალადობა ინფორმაციის გამოძალვის მიზნით, მაშინ მოწინააღმდეგეს რჩება ერთადერთი საშუალება - შეაღწიოს გადამცემ არხში და მოიპოვოს ეს ინფორმაცია არხიდან. ამ პროცესს ეწოდება ***ინფორმაციის მიტაცება*** (ამ ტერმინს დღეს ხშირად უწოდებენ ***არასანქცირებულ წვდომას***). აქედან გამომდინარე, კონფიდენციალური ინფორმაციის გადაცემის დროს ერთერთი უმთავრესი პრობლემაა

ინფორმაციის დაცვა არხში მიტაცებისაგან. ისტორიულად ცნობილია ამ პრობლემის გადაჭრის სამი შესაძლო გზა:

- ინფორმაციის გადასაცემად გამოვიყენოთ საიდუმლო, ანუ ***დახურული*** არხი, რომელშიც ვერ შეაღწევს მოწინააღმდეგე და ვერ შეძლებს ინფორმაციის მიტაცებას;
- ინფორმაციის გადასაცემად გამოვიყენოთ ჩვეულებრივი, ანუ ***ღია*** არხი, მაგრამ დავმალოთ თვით ინფორმაციის გადაცემის ფაქტი;
- ღია არხში ინფორმაციის გადაცემამდე ეს ინფორმაცია ისე გარდავქმნათ, რომ მოწინააღმდეგემ, მიტაცების შემთხვევაშიც, ვერ შეძლოს მისი წაკითხვა.



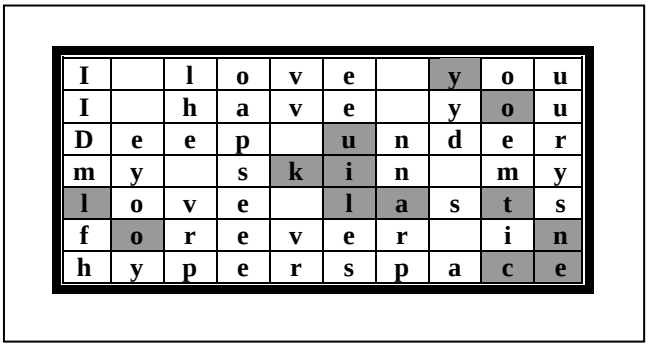
სურ. 2.1 ინფორმაციის გადაცემის ზოგადი სქემა

მრავალი საუკუნის განმავლობაში კაცობრიობა მეტ-ნაკლები წარმატებით იყენებს სამივე შესაძლებლობას. პირველი მიდგომის, დახურული არხის მაგალითია ფელდეგერული სამსახური, რომელიც დიდი ხანია არსებობს მრავალ სახელმწიფოში. რაც შეეხება დახურული ტექნიკური არხის შექმნას ორ, ან რამდენიმე ფიზიკურად დაშორებულ აბონენტს შორის, რომელშიც შესაძლებელი იქნება დიდი მოცულობის ინფორმაციის მრავალჯერადი გადაცემა, თუ გავითვალისწინებთ თანამედროვე მეცნიერებისა და ტექნიკის მიღწევებს, დღეს პრაქტიკულად თითქმის შეუძლებე-ლია.

ინფორმაციის გადაცემის ფაქტის დამალვის მეთოდებისა და საშუალებების შექმნით დაკავებულია სტეგანოგრაფია. ჯერ კიდევ ძველი ბერძენი ისტორიკოსი ჰეროდოტი თავის ‘‘ისტორიებში’’ მოგვითხრობს სტეგანოგრაფიის გამოყენების ორ ძალიან საინტერესო შემთხვევაზე. ჩვენს წელთაღრიცხვამდე მეხუთე საუკუნეში, სპარსეთის მეფე ქსერქსი ემზადებოდა რა სპარტისა და ათენის დასალაშქრად, რომლებმაც უარი განაცხადეს მისთვის ხარკის გადახდაზე, რამდენიმე წლის განმავლობაში აგროვებდა ჯარს. ქსერქს უნდოდა ესარგებლა მოულოდნელობის ეფექტით, ამიტომ მზადება მიმდინარეობდა ფარულად. ამ დროს სპარსეთში ცხოვრობდა დემარატი, ათენელი, რომელიც გაძევებული იყო სამშობლოდან. მიუხედავად ამისა თვლიდა რა ის თავის თავს ელინად და ხედავდა რა სპარსელთა მზადებას მისი სამშობლოს წინააღმდეგ, გადაწყვიტა გაეფრთხილებინა თავისი თანამემამულენი მოსალოდნელი საფრთხის შესახებ. პრობლემა მდგომარეობდა იმაში, თუ როგორ გაეგზავნა შეტყობინება ფარულად საბერძნეთში. წერილი, რომელსაც ის გააგზავნიდა შეიძლებოდა ხელში ჩავარდნოდა სპარსელებს, რომლებიც აკონტროლებდნენ გზებს და საზღვრებს. დემარატმა ამოირჩია ალბათ ერთადერთი სწორი გზა. მან ჩამოფხიკა სანთელი გასანთლული ფიცრიდან, (ასეთ ფიცრებს იყენებდნენ იმ დროს საწერად) და პირდაპირ სუფთა ფიცარზე დაწერა თავისი წერილი. ამის შემდეგ მან კვლავ წაუსვა ფიცარს სანთელი და ასეთი სახით გააგზავნა სპარტაში. სპარსელებმა დაუბრკოლებლივ გაატარეს სუფთა საწერი ფიცრები. როგორც ჰეროდოტი აღწერს, სპარტას დედოფალი მიხვდა, რომ არ შეიძლებოდა შემთხვევით ყოფილიყო გამოგზავნილი ორი ცარიელი საწერი დაფა და ბრძანა ჩამოეფხიკათ სანთელი ამ ფიცრებიდან. ასე გახდა ცნობილი ბერძნებისათვის ქსერქსის განზრახვა. შეიტყვეს რა სპარსელების განზრახვა, ბერძნები სათანადოდ მოემზადნენ და 480 წლის 28 სექტემბერს სარონიკოს ყურეში მთლიანად გაანადგურეს სპარსელთა ფლოტი. ასე გადაწყვიტა ბრძოლის ბედი ერთმა საიდუმლოდ გაგზავნილმა წერილმა.

მეორე შემთხვევა ეხება წერლის, რომელიც ბერძენმა ხისტაესმა გაუგზავნა მილეტის მბრძანებელ არისტაგორისს. იმისათვის, რომ მტრებს არ აღმოეჩინათ წერილი, ხისტაესმა ის დააწერა შიკრიკს გადაპარსულ თავზე და შიკრიკი გაგზავნა დანიშნულების ადგილზე მხოლოდ მას შემდეგ, რაც მას წამოეზარდა თმები. დანიშნულების ადგილზე მისვლის შემდეგ შიკრიკმა კვლავ გადაიპარსა თმები და ისე წარსდგა არისტაგორის წინაშე.

სტეგანოგრაფიის მეთოდები თანდათანობით ვითარდებოდა და სულ უფრო მეტად უკავშირდებოდა მეცნიერების მიღწევებს. შუა საუკუნეებიდან ცნობილია კარდანოს ცხაური და სიმპატიკური მელანი, რომლებსაც აქტიურად იყენებდნენ საიდუმლო მიმოწერის დროს. ჯიროლამო კარდანო იყო შუა საუკუნეების ცნობილი მეცნიერი, რომელმაც თავისი წვლილი შეიტანა როგორც სტეგანოგრაფიის, ასევე კრიპტოგრაფიის განვითარებაში. 1550 წელს, ინფორმაციის გადაცემის პროცესის დასამალად მან შემოიტანა ცხაური, რომელმაც შემდეგ მიიღო კარდანოს ცხაურის სახელი. კარდანოს ცხაური წარმოადგენს მუყაოს ნაჭერს შემთხვევითად ამოჭრილი ნახვრეტებით, რომლებშიც შეიძლება ჩავწეროთ როგორც თითო ასო, ასევე მარცვლები და მთელი სიტყვებიც კი. მუყაოს ნაჭრის მოცილების შემდეგ ღია ადგილები შეივსება ისე, რომ წერილმა მიიღოს ბუნებრივი სახე. ადრესატი ფლობს ასეთივე ცხაურს, რომლის საშუალებითაც ის კითხულობს დამალულ ინფორმაციას. ასეთ ცხაურს იყენებდა თავისი საიდუმლო მიმოწერისათვის ალექსანდრე დიუმას რომანებიდან ყველასათვის კარგად ცნობილი საფრანგეთის კარდინალი რიშელიე. რიშელიეს ცხაური წარმოადგენდა მართკუთხედის ფორმის მუყაოს ნაჭერს, რომელშიც შემთხვევითად იყო ამოჭრილი უჯრები (იხ. სურ. 2.2). "**I love you I have you. deep under my skin my love lasts forever in hyperspace**". აი ასეთი სასიყვარულო წერილი, სინამდვილეში შეიცავდა სასტიკ ბრძანებას, "**you kill at once**". შემდგომში ამ მეთოდმა მიიღო დასაცავი ინფორმაციის დიდ ინფორმაციაში "ჩაძირვის" სახელი. კომპიუტერული ტექნიკის განვითარებამ ახალი სტიმული მისცა ამ მეთოდს. დღეს შესაძლებელია მაგალითად, ტექსტური ინფორმაციის დამალვა გრაფიკულ ინფორმაციაში, ან იმიტაციური ფუნქციების გამოყენება, რომლებიც ისე გარდაქმნიან ინფორმაციას, რომ მისი სტატისტიკური პარამეტრები დაემგვანება რაიმე ცნობილი ტექსტის პარამეტრებს. კომპიუტერული სტეგანოგრაფია ვითარდება კომპიუტერული კრიპტოგრაფიის პარალელურად და ზოგჯერ მასთან მჭიდრო კავშირშიც.



სურ. 2.2. რიშელიეს ცხაური (გამუქებულია ამოჭრილი უჯრები)

მეოცე საუკუნეში სტეგანოგრაფიაში გაჩნდა ე.წ. "მიკროწერტილის" მეთოდი. ინფორმაცია ტექნიკური საშუალებებით ჩაიწერება ძალიან მცირე ზომის ინფორმაციის მატარებელზე და შემდეგ ეს მიკროწერტილი შეიძლება დაიმალოს ნებისმიერ ადგილას (საფოსტო მარკის ქვეშ, ადამიანის კბილში და ა.შ.)..

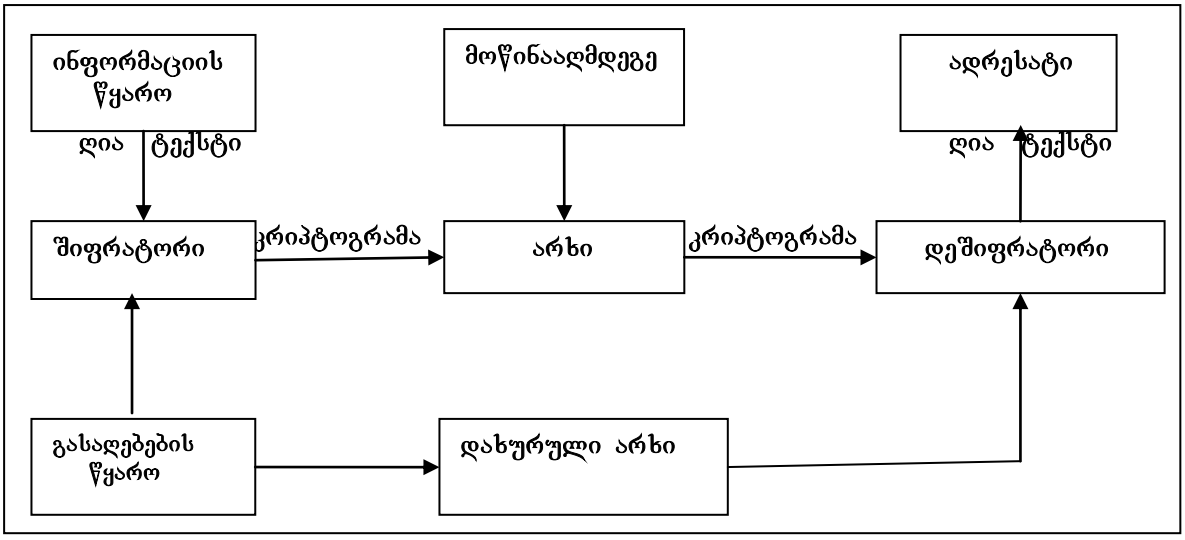
სტეგანოგრაფიის გამოყენებას აქვს ერთი სერიოზული ნაკლი. თუ მოწინააღმდეგემ აღმოაჩინა ინფორმაციის გადაცემის ფაქტი, ის ამ ინფორმაციას დაუბრკოლებლად წაიკითხავს. ცხადია, რომ თუ ეს შეტყობინება ჩაწერილი იქნება ისეთი სახით, რომ მოწინააღმდეგე ვერ შესძლებს მის წაკითხვას მაშინაც კი, როდესაც ის მას ხელში ჩაგდებს, ეს კიდევ უფრო საიმედოდ დაიცავს ინფორმაციას. ღია არხით ინფორმაციის გადაცემამდე მისი ისე გარდაქმნით, რომ მოწინააღმდეგემ მიტაცების შემთხვევაშიც ვერ შეძლოს ამ ინფორმაციის წაკითხვა, დაკავებულია ***კრიპტოგრაფია***, რომელიც წარმოადგენს ჩვენი შესწავლის საგანს.

ტერმინი "კრიპტოგრაფია" წარმოიშვა ორი ბერძნული სიტყვისაგან: "კრიპტოს" – საიდუმლოდ და "გრაფია" - წერა და ნიშნავს საიდუმლო დამწერლობას, რაც ზუსტად გამოხატავს ამ პროცესის შინაარსს. საუკუნეების განმავლობაში კრიპტოგრაფიას ჰყავდა ორი ძირითადი მომხმარებელი - სამხედროები და დიპლომატები, ამიტომაც კრიპტოგრაფიაში დამკვიდრდა მრავალი ტერმინი, რომელიც დაკავშირებულია ამ სფეროებთან (მტერი, შეტევა კრიპტოსისტემაზე, ინფორმაციის მიტაცება და ა.შ.).

თავისთავად ცხადია, რომ მოწინააღმდეგე, რომელმაც შეძლო კრიპტოგრამის მიტაცება, ცდილობს სხვადასხვა მეთოდების საშუალებით კრიპტოგრამიდან აღადგინოს ღია ტექსტი. ამ მეთოდების შექმნით და განვითარებით დაკავებულია ***კრიპტოანალიზი***. ორივე ერთად, კრიპტოგრაფია და კრიპტოანალიზი ქმნიან ***კრიპტოლოგიას***, მეცნიერებას, რომელიც შეისწავლის ინფორმაციის ისეთნაირად გარდაქმნის მეთოდებსა და საშუალებებს, რომლებიც უზრუნველყოფენ ინფორმაციის უსაფრთხო გადაცემას ფიზიკურად დაშორებულ აბონენტებს შორის.

კონფიდენციალურობის დაცვას უზრუნველყოფენ სიმეტრიული კრიპტოალგორითმები. კრიპტოსისტემას, რომელშიც დაშიფრვისა და დეშიფრაციის ალგორითმი და გასაღები ერთი და იგივეა, უწოდებენ სიმეტრიულ კრიპტოსისტემას. იმისათვის, რომ ასეთი კრიპტოსისტემის გამოყენებით გადავცეთ ინფორმაცია, საჭიროა წინასწარ შევათანხმოთ გასაღები, რომლითაც ისარგებლებენ აბონენტები. ამასთან, თუ გვინდა, რომ კრიპტოალგორითმი დიდი ხნის განმავლობაში ვერ გატეხოს მოწინააღმდეგემ, საჭიროა გარკვეული რაოდენობის ინფორმაციის გადაცემის შემდეგ (ეს დამოკიდებულია თვით ალგორითმზე) შეიცვალოს გასაღები. ორივე ამ შემთხვევაში აბონენტები ან პირადად უნდა შეხვდნენ ერთმანეთს, რაც ყოველთვის არ არის მოსახერხებელი, ან გამოიყენონ დახურული არხი გასაღების გასაცვლელად (იხ. სურ. 2.3).

გასაღებების გაცვლის პრობლემა წარმოადგენს სიმეტრიული კრიპტოალგორითმების ერთერთ ყველაზე უფრო რთულ პრობლემას. ჩვენ შემდეგ ვნახავთ თუ როგორ შეიძლება გადაიჭრას ეს პრობლემა თანამედროვე კრიპტოგრაფიაში. ჯერჯერობით კი შევნიშნოთ, რომ გასაღების გადაცემა დახურული არხით უფრო იოლია, ვიდრე თვით ინფორმაციის, რადგანაც გასაღების ზომა არის უფრო პატარა გადასაცემ ინფორმაციასთან შედარებით და გაცვლაც ხდება უფრო იშვიათად, ვიდრე ინფორმაციის გაცვლა. თუმცა თანამედროვე კრიპტოგრაფიაში, როდესაც კრიპტოსისტემას ჰყავს მრავალი მომხმარებელი და საჭიროა თითოეული წყვილისათვის განსხვავებული გასაღების შექმნა, ეს პრობლემა ხდება ერთერთი მთავარი და მოითხოვს გასაღებების მართვის სპეციალური სისტემის შექმნას.



სურ. 2.3 კონფიდენციალური ინფორმაციის გადაცემა ღია არხით სიმეტრიული კრიპტოგრაფიული ალგორითმის გამოყენებით.

2.2. ინფორმაციის მთლიანობის დაცვა. ძალიან ხშირად გვხვდება სიტუაციები, როდესაც ჩვენთვის არა იმდენად მთავარია ინფორმაციის კონფიდენციალურობის დაცვა, რამდენადაც იმის ცოდნა, მოაღწია თუ არა ჩვენამდე ინფორმაციამ შეუცვლელი სახით. მართლაც ბოლოს და ბოლოს ინტერნეტი შეიქმნა არა ინფორმაციის დასამალად, არამედ ინფორმაციის გასაცვლელად ადამიანებს შორის. ამიტომაც ამ შემთხვევაში მთავარია ინფორმაციის მთლიანობის პრობლემა, დამახინჯდა თუ არა ინფორმაცია (უნებლიედ თუ წინასწარი განზრახვით) ქსელში გადაცემის დროს და არა კონფიდენციალობის პრობლემა.

ასევე შესაძლებელია სიტუაცია, როდესაც შეტყობინებას თქვენი პარტნიორის სახელით აგზავნის სრულიად სხვა პირი, ანუ ხდება იმიტაცია. ინფორმაციის მთლიანობისა და იმიტაციისაგან თავის დასაცავად საჭიროა გადაიჭრას ინფორმაციისა და ავტორობის იდენტიფიკაციისა და აუთენტიფიკაციის პრობლემა, რომელიც შეიძლება სულაც არ იყოს დაკავშირებული კონფიდენციალობის პრობლემასთან.

ასევე ადვილი შესაძლებელია, რომ შეტყობინება გამოგიგზავნოთ ნამდვილად თქვენმა საქმიანმა პარტნიორმა, მაგრამ მეორე დღეს მან უარყოს ამ შეტყობინების ავტორობა. არც ეს მომენტი იქნება თქვენთვის სასიამოვნო, ამიტომ უნდა შეგეძლოთ დაუმტკიცოთ თქვენს პარტნიორს, რომ წერილი მის მიერ იყო გამოგზავნილი. ანუ შეტყობინების ავტორს ვერ უნდა შეეძლოს უარყოს თავისი ავტორობა.

ამ პრობლემის გადაჭრა სიმეტრიული კრიპტოგრაფიის საშუალებით არაეფექტურია, ამიტომ დღეს ასეთი ამოცანების გადასაჭრელად გამოიყენება ***ღია გასაღებიანი კრიპტოგრაფია.*** ღია გასაღებიან კრიპტოგრაფიაში გვაქვს ორი გასაღები, ერთი საიდუმლო (დეშიფრაციის გასაღები), რომელიც ცნობილია მხოლოდ ინფორმაციული ურთიერთობის ერთი სუბიექტისათვის და მეორე, ღია გასაღები (დაშიფრვის გასაღები), რომელიც ცნობილია ყველა დანარჩენი სუბიექტისათვის. ღია გასაღები გამოქვეყნებულია ქსელში და ნებისმიერ სუბიექტს შეუძლია დაშიფროს ამ გასაღებით ინფორმაცია. დაშიფრული ინფორმაციის დეშიფრაცია შესაძლებელია მხოლოდ საიდუმლო გასაღებით, ამიტომ მხოლოდ ამ გასაღების მფლობელს შეუძლია გაშიფროს ინფორმაცია.

ღია გასაღებიანი კრიპტოგრაფიის საფუძველს წარმოადგენს ე.წ. ***ცალმხრივ მიმართული ფუნქცია.*** განვიხილოთ ორი ნებისმიერი სიმრავლე X და Y და ბიექციური ფუნქცია $f : X \to Y$. f ფუნქცია იქნება ცალმხრივ მიმართული ფუნქცია, თუ ნებისმიერი $x \in X$-თვის ადვილია $y = f(x)$ მნიშვნელობის გამოთვლა, მაგრამ თითქმის ყველა $y \in E(f)$-თვის ისეთი $x \in X$-ის პოვნა, რომ შესრულდეს ტოლობა $y = f(x)$ არის რთული. გამოთვლათა სირთულის თეორიის დღევანდელი მდგომარეობა არ გვაძლევს საშუალებას, რომ უფრო ზუსტად განვსაზღვროთ ასეთი ფუნქციები და საერთოდ მკაცრად, მათემატიკურად დავამტკიცოთ მათი არსებობა. როგორც წესი, ადვილად გამოთვლის ქვეშ შემდგომში ვიგულისხმებთ, რომ არსებობს $y = f(x)$ ფუნქციის გამოთვლის პოლინომური ალგორითმი, ხოლო რთულად გამოთვლის ქვეშ კი – რომ ასეთი პოლინომური ალგორითმი უცნობია (დააკვირდით, რომ ჩვენ არ ვამბობთ, რომ ასეთი ალგორითმი არ არსებობს). "თითქმის ყველა" კი ნიშნავს, რომ შეიძლება არსებობდეს ძალიან მცირე რაოდენობა ისეთი y, რომლებისთვისაც ადვილი იქნება ვიპოვოთ შესაბამისი x, რომ შესრულდეს ტოლობა $y = f(x)$. განვიხილოთ ასეთი ფუნქციების მაგალითები. პირველ რიგში ასეთი ფუნქცია შეიძლება იყოს ერთი შეხედვით ისეთი მარტივი ფუნქცია, როგორიცაა მთელი რიცხვების გამრავლება რაიმე მოდულით. არსებობს ალგორითმი, რომელიც საშუალებას გვაძლევს პოლინომურ დროში ვიპოვოთ ნებისმიერი ორი მთელი რიცხვის ნამრავლი, მაშინაც კი, როდესაც ეს რიცხვები ძალიან დიდია (მათი ჩანაწერი ათობით სისტემაში შედგება რამდენიმე ასეული თანრიგისაგან), მაგრამ, შებრუნებული ამოცანა, მთელი რიცხვის დაშლა მარტივ თანამამრავლებად (ამ ამოცანას უწოდებენ რიცხვის ფაქტორიზაციას) დამოკიდებულია როგორც ამ რიცხვზე, ასევე რიცხვის ზომებ-ზე და მისი შესრულება პოლინომურ დროში ყოველთვის არ ხერხდება. რაც ნიშნავს სწორედ იმას, რომ არსებობენ მცირე რაოდენობის $y \in E(f)$, რომლებისთვისაც ადვილი იქნება ისეთი $x \in X$-ის პოვნა, რომ შესრულდეს ტოლობა $y = f(x)$, მაგრამ ძალიან დიდი რაოდენობის $y = f(x)$-თვის შესაბამისი x-ის მოძებნა პოლინომურ დროში შეუძლებელია. მაგალითად, დავშალოთ ორი მარტივი რიცხვის ნამრავლი, როდესაც თითოეული რიცხვი ოთხასთანრიგიანია ათობით წარმოდგე-ნაში, შეუძლებელია მაშინაც კი, როდესაც ვიყენებთ დღეს ცნობილ საუკეთესო ალგორითმებს და ყველაზე მძლავრ კომპიუტერულ სისტემას.

მეორე, კრიპტოგრაფიულად მნიშვნელოვანი ცალმხრივი ფუნქციის მაგალითია რიცხვის ხარისხში აყვანა მოდულით (როდესაც ხარისხის და მოდულის ფუძეები ცნობილია). დავუშვათ, n და a ორი ისეთი მთელი რიცხვია, რომ $1 < a < n$ და $Z_n = \{1,2,...,n-1\}$. მაშინ ხარისხში აყვანის ფუნქცია $f_{a,n} : Z \to Z_n$ მოდულით n, იქნება $f_{a,n}(m) = a^m \pmod n$. ერთი შეხედვით ამ ფუნქციის შესრულება, როცა a, n და m რამდენიმე ასეულ თანრიგს შეიცავს, არც ისე ადვილია და გამოთვლების რაოდენობა უნდა იზრდებოდეს ექპონენციალურად m-ის ზრდასთან ერთად, მაგრამ სინამდვილეში ეს ასე არაა. ამის დასადასტურებლად განვიხილოთ მაგალითი:

$$a^{25} \pmod n = ((((((a^2 \bmod n) \cdot a) \bmod n)^2 \bmod n)^2 \bmod n)^2 \cdot a) \bmod n$$

როგორც ვხედავთ a^{25} რიცხვის გამოსათვლელად საჭიროა მხოლოდ ოთხჯერ კვადრატში აყვანა და ორჯერ გამრავლება, ამასთან ყოველ ეტაპზე მოდულის გამოყენება n-ზე ნაკლებს ხდის ეტაპზე მიღებულ შედეგს და ამარტივებს გამოთვლის პროცესს. ეს მაგალითი ასევე გვიჩვენებს ახარისხების ფუნქციის ერთ მნიშვნელოვან თვისებას, რომ ნებისმიერ ხარისხში ახარისხება შეიძლება დავიყვანოთ კვადრატში ახარისხებაზე, რაც მნიშვნელოვანია ზოგიერთი კრიპტოგრაფიული ამოცანებისათვის.

შებრუნებულ ამოცანს, რომლის დროსაც მოცემულია a, n და x და უნდა ვიპოვოთ ისეთი მთელი m (თუ ის არსებობს), რომ $x = a^m \pmod n$, ეწოდება ***დისკრეტული ლოგარითმირების ამოცანა.*** განსხვავებით პირველი ამოცანისაგან, რომელიც შეიძლება ძალიან სწრაფად შესრულდეს, დღეისათვის ჩვენთვის უცნობია ისეთი ალგორითმი, რომლის საშუალებითაც პოლინომიალურ დროში შესაძლებელი იქნება ამ ამოცანის ამოხსნა დიდი რიცხვებისათვის.

კრიპტოგრაფიაში ცალმხრივ მიმართულ ფუნქციებზე უფრო მნიშვნელოვანია ცალმხრივ მიმართული ფუნქცია საიდუმლო პარამეტრით. რომელსაც კრიპტოგრაფიაში ხაფანგს უწოდებენ. ***ცალმხრივ მიმართული ხაფანგიანი*** ფუნქცია ეწოდება ისეთ $f_k : X \to Y$ ფუნქციას, რომლისთვისაც ისევე, როგორც ცალმხრივ მიმართული ფუნქციისთვის ცნობილი x-თვის ადვილია $y = f_k(x)$ გამოთვლა. რაც შეეხება მოცემული y-თვის ისეთი x-ის პოვნას, რომ დაკმაყოფილდეს ტოლობა $y = f_k(x)$, ეს უკვე დამოკიდებულია საიდუმლო პარამეტრის ცოდნაზე. ზოგადად ამ ამოცანისათვის ცნობილია არაპლინომური ალგორითმი, მაგრამ თუ ცნობილია პარამეტრი k, ამოცანა ამოიხსნება პოლინომური ალგორითმის საშუალებითაც. ასეთი ფუნქციის მაგალითი კვლავ შეიძლება იყოს ზემოთმოყვანილი მოდულით ახარისხების ფუნქცია, მაგრამ უკვე ფიქსირებული ხარისხის მაჩვენებლითა და მოდულის ფუძით $g_{m,n}(a) = a^m \pmod n$. ამ შემთხვევაში შებრუნებული ამოცანაა m-ური ხარისხის ფესვის ამოღება x-დან, ანუ თუ მოცემულია მთელი რიცხვები m, n და x, უნდა ვიპოვოთ ისეთი მთელი a რიცხვი, რომ $x = a^m \pmod n$. მაგალითად, 5 არის მეოთხე ხარისხის ფესვი 16-დან მოდულით 16, რადგანაც $5^4 \pmod{21} = 16$, მაგრამ ცხადია, რომ 2-იც ასევე მეოთხე ხარისხის ფესვია თექვსმეტიდან მოდულით 21, რადგანაც $2^4 \pmod{21} = 16$. განსხვავებით დისკრეტული ლოგარითმირების ამოცანისაგან, ამ შემთხვევაში არსებობს ეფექტური პოლინომური ალგორითმი, რომელიც საშუალებას გვაძლევს ვიპოვოთ a, მაგრამ ამისათვის საკმარისი არ არის მხოლოდ m და n რიცხვების ცოდნა, საჭიროა აგრეთვე ვიცოდეთ, თუ როგორ იშლება მარტივ მამრავლებად მოდულის ფუძე. სწორედ ესაა ამ ფუნქციის საიდუმლო პარამეტრი. ის ვინც იცის მოდულის ფუძის მარტივ მამრავლებად დაშლა, შეძლებს იპოვოს a რიცხვი, ვინც ამ საიდუმლოს არ ფლობს, მისთვის a რიცხვის მნიშვნელობის გაგება შეუძლებელია.

განსაკუთრებით მნიშვნელოვანია ამ ფუნქციის კერძო შემთხვევა, როდესაც ხარისხის მაჩვენებელი ტოლია ორის, ხოლო მოდულის ფუძე წარმოადგენს ბლუმის რიცხვს. ბლუმის რიცხვი ეწოდება $n = p \cdot q$ სახის შედგენილ რიცხვს, სადაც $p \equiv q \equiv 3 \pmod 4$.

აღვნიშნოთ $Z^* = \{1,2,...,n-1\}$ სიმრავლე იმ მთელი რიცხვებისა, რომლებიც ნაკლებია n-ზე და რომლებიც არ იყოფიან p-ზე და q-ზე. $QR_n \subset Z^*$ იყოს სიმრავლე იმ რიცხვებისა, რომლებიც წარმოადგენენ კვადრატულ ნაშთებს n ფუძით. მაგალითად, დავუშვათ, რომ $p = 19$ და $q = 23$, მაშინ $n = 19 \cdot 23 = 437$. $135 \in Z^*$, მაგრამ $133 = 19 \cdot 7 \notin Z^*$. 135 არ წარმოადგენს კვადრატულ ნაშთს, რადგანაც არ არსებობს ისეთი მთელი რიცხვი $1 < a < n$ და $a^2 \equiv 135 \pmod{437}$, მაშინ როდესაც $24^2 = 576 \equiv 139 \pmod{437}$, ამიტომ 139 კვადრატული ნაშთია. როგორც ვიცით, ამ $Z^* = \{1,2,...,n-1\}$ სიმრავლეში ელემენტების რაოდენობა ტოლია $(p-1)(q-1)$, ამათგან ზუსტად მეოთხედი წარმოადგენს კვადრატულ ნაშთს. თითოეულ კვადრატულ ნაშთს Z^* სიმრავლეში შეესაბამება ოთხი "კვადრატული ფესვი", რომელთაგან მხოლოდ ერთი წარმოადგენს პირველყოფილ ფესვს. მაგალითად, 139 შეესაბამება "კვადრატული ფესვები" $24,185,253,413$, რომელთაგანაც მხოლოდ 24 წარმოადგენს პირველყოფილ ფესვს.

კრიპტოგრაფიისათვის მნიშვნელოვანია ის ფაქტი, რომ პირველყოფილი ფესვის საპოვნელად აუცილებელია ვიცოდეთ n რიცხვის დაშლა მარტივ მამრავლებად. სწორედ ეს ფაქტი იქნება

ჩვენი ფუნქციის საიდუმლო პარამეტრი. ის ვინც იცის n რიცხვის დაშლა მარტივ მამრავლებად პოლინომიალურ დროში იპოვის პირველყოფილ ფესვს, ხოლო ვისაც ეს ინფორმაცია არა აქვს, მოუწევს ამოხსნას რიცხვის ფაქტორიზაციის ამოცანა, რომლის პოლინომურ დროში ამოხსნის ალგორითმი უცნობია.

როგორც ვხედავთ, ცალმხრივი ფუნქციები ძირითადად წარმოადგენენ რიცხვთა თეორიის ისეთ ამოცანებს, რომელთა ამოხსნის ალგორითმი არაპოლინომურია. ამიტომ მოწინააღმდეგისათვის შეუძლებელი ხდება ღია გასაღებიდან საიდუმლო გასაღების აღდგენა, რაც წარმოადგენს ასეთი კრიპტოსისტემების საიმედოობის საფუძველს.

2.3. აუთენტიფიკაცია. აუთენტიფიკაცია ინფორმაციული უსაფრთხოების დაცვაში გამოიყენება

- არასანქცირებული წვდომისაგან თავის აცილების;
- ინფორმაციის მთლიანობის და
- ინფორმაციული ურთიერთობის სუბიექტების ნამდვილობის დასადასტურებლად.

ინფორმაციული უსაფრთხოების ერთერთ ყველაზე უფრო მნიშვნელოვან პრობლემას წარმოადგენს არასანქცირებული წვდომისაგან საინფორმაციო სისტემების დაცვა. ამ მიზნის მისაღწევად აუცილებელია მოვახდინოთ სისტემაში შემოსვლის დროს მომხმარებლის იდენტიფიკაცია და აუთენტიფიკაცია. სისტემის მომხმარებლის იდენტიფიკაცია ნიშნავს მომხმარებლისათვის უნიკალური იდენტიფიკატორის მინიჭებას, ხოლო აუთენტიფიკაცია წარმოადგენს მომხმარებლის სისტემაში შემოსვლის დროს წარმოდგენილი იდენტიფიკატორის შემოწმებას და მისი ნამდვილობის დადასტურებას. იდენტიფიკატორი შეიძლება იყოს მომხმარებლის ბიომეტრიული პარამეტრები, ფოტოსურათი, მაგრამ დღეს ყველაზე გავრცელებულ სისტემას წარმოადგენს მომხმარებლის აუთენტიფიკაციის პაროლური სისტემა. ამის მიზეზია ის, რომ დაცვის პაროლური სისტემა არ მოითხოვს დამატებით აპარატურულ საშუალებებს და ის ძალიან მარტივი აღსაქმელია მომხმარებლისათვის.

ინფორმაციის მთლიანობის დასაცავად აგრეთვე გამოიყენება აუთენტიფიკაციის პროცედურა, ოღონდ ინფორმაციის მთლიანობის აუთენტიფიკაციის პროცედურა. წარმოვიდგინოთ ასეთი სიტუაცია: ინფორმაციული ურთიერთობის ორი სუბიექტი ინფორმაციის გასაცვლელად სარგებლობენ ღია არხით, რომელსაც მთლიანად აკონტროლებს მოწინააღმდეგეა და მას შეუძლია არა მარტო მიიტაცოს ინფორმაცია (რომელიც შეიძლება ღიაა), არამედ განახორციელოს აქტიური შეტევაც, რაც გამოიხატება იმაში, რომ მას აქვს საშუალება

- დაბლოკოს სუბიექტის მიერ გადაცემული ინფორმაცია და არ მიუშვას ის ადრესატამდე;
- შეუძლია მთლიანად ან ნაწილობრივ შეცვალოს გადაცემული ინფორმაცია;
- შეუძლია სუბიექტის სახელით გაუგზავნოს ადრესატს ინფორმაცია, რომელიც სუბიექტს არ გადაუცია.
- შეუძლია შეცვალოს გადაცემულ შეტყობინებათა მიმდევრობა;
- შეუძლია დააყოვნოს შეტყობინება არხში და გადასცეს მოგვიანებით.

პირველი ვარიანტი მოწინააღმდეგეს არ უნდა აწყობდეს, რადგანაც სუბიექტები პირველივე დაბლოკი გადაცემის შემდეგ აუცილებლად შეცვლიან არხს და მოწინააღმდეგე ვეღარ შეძლებს ხელი შეუშალოს მათ. მეოთხე და მეხუთე ვარიანტების წინააღმდეგ საკმარისია დროითი ჭდის გამოყენება, რათა აღადგინონ შეტყობინებათა ნორმალური მიმდევრობა და გადაცემის დრო. ამიტომ ყველაზე რთულ ვარიანტებად რჩება მხოლოდ მეორე და მესამე, რომლებიც წარმოადგენენ ინფორმაციის შეცვლისა და იმიტაციის შეტევებს. არსებობს ამ შეტევებისაგან თავის დაცვის სხვადასხვა საშუალებები, რომლებიც უზრუნველყოფენ დაცვის სხვადასხვა დონეს. ამისთვის თანამედროვე კრიპტოგრაფიაში შესაძლებელია გამოვიყენოთ სამი განსხვავებული მიდგომა, რომლებიც უზრუნველყოფენ უსაფრთხოების სხვადასხვა დონეს და შესაბამისად მოითხოვენ სხვადასხვა სირთულის ალგორითმებს პრაქტიკული რეალიზაციისათვის. ყველა მათგანის დამახასიათებელი თვისებაა, რომ სხვადასხვა პროცედურების საშუალებით გამოითვლება სპეციალური კოდი, რომელიც გადაიცემა ღია არხით ტექსტთან ერთად, რომელსაც ეწოდება აუთენტიფიკაციის კოდი და რომელიც ადასტურებს ინფორმაციის მთლიანობას.

პირველ რიგში უნდა აღვნიშნოთ ინფორმაციის თეორიაზე დაფუძნებული მიდგომა, რომელიც უზრუნველყოფს უპირობოდ მედეგ ინფორმაციის მთლიანობის აუთენტიფიკაციის პროცედურას ნებისმიერი შეტევის მიმართ, რა გამოთვლითი საშუალებებიც არ უნდა გააჩნდეს მოწინააღმდეგეს.

	მოწინააღმდეგის გამოთვლითი	უსაფრთხოების დონე	პრაქტიკული გამოყენება
თეორიულ ინფორმაციული	შემოუსაზღვრავი	დამტკიცებადად უპირობო	არაპრაქტიკულია
სირთულის თეორიაზე	პოლინომიალური	ასიმპტოტიკური (ვერ მტკიცდება)	არაპრაქტიკულია
სისტემაზე	ფიქსირებული	არ მტკიცდება	პრაქტიკულია

სურ. 2.4 სამი მიდგომის შედარება.

აუთენტიფიკაციის პროცედურის უსაფრთხოება შეიძლება ემყარებოდეს სირთულის თეორიას. ასეთ შემთხვევაში იგულისხმება, რომ პროცედურის უსაფრთხოება ტოლფასია რომელიმე **NP** კლასის ამოცანის სირთულის და მოწინააღმდეგეს არ გააჩნია საჭირო გამოთვლითი საშუალებები ასეთი ამოცანის ამოსახსნელად.

ყველაზე ხშირად კი გამოიყენება რომელიმე სიმეტრიულ კრიპტოგრაფიულ სისტემაზე დაფუძნებული აუთენტიფიკაციის პროცედურა, რომლის უსაფრთხოება გათვლილია იმაზე, რომ მოწინააღმდეგეს გააჩნია შეზღუდული გამოთვლითი საშუალებები, რაც არ აძლევს მას საშუალებას გატეხოს სისტემა, რომლის კრიპტომედეგობა მთლიანადაა განსაზღვრული გამოყენებული შიფრით. სურ. #2.4 მოყვანილია ამ მიდგომათა შედარება მათი უსაფრთხოებისა და პრაქტიკულად გამოყენების თვალსაზრისით.

საკონტროლო კითხვები:

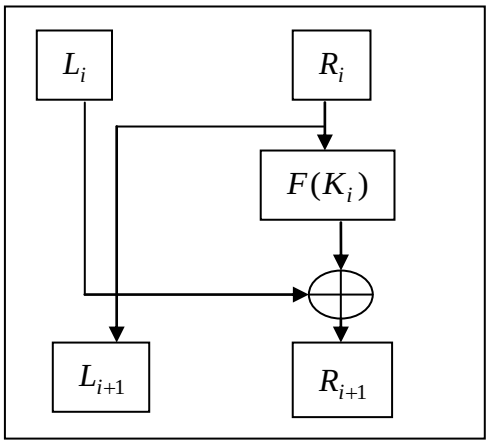
1. რას ნიშნავს ინფორმაციის კონფიდენციალურობა?
2. რა მიდგომები შეიძლება გამოვიყენოთ ინფორმაციის კონფიდენციალურობის დასაცავად?
3. რა არის სტეგანოგრაფია?
4. როგორ კრიპტოსისტემებს ეწოდებათ სიმეტრიული ალგორითმები?
5. რას ნიშნავს ინფორმაციის მთლიანობის დაცვა?
6. როგორი კრიპტოსისტემები გამოიყენება მთლიანობის დასაცავად?
7. როგორ ფუნქციებს ეწოდებათ ცალმხრივ მიმართული ფუნქციები?
8. როგორ ფინქციებს ეწოდებათ ცალმხრივ მიმართული ხაფანგიანი ფუნქციები?
9. რა პრობლემების გადასაჭრელად გამოიყენება აუთენტიფიკაციის პროცედურა?
10. რას ნიშნავს არასანქცირებული წვდომა?

სიმეტრიული კრიპტოგრაფიული ალგორითმები DES და RIJNDAEL

3.1. DES-ის შექმნის ისტორია. გასული საუკუნის სამოციანი წლების ბოლოს, ცნობილმა ამერიკულმა კომპანიამ **IBM**-მა დაიწყო ზრუნვა ისეთი სისტემების (მათ შორის კრიპტოგრა-ფიულის) შექმნაზე, რომლებიც უზრუნველყოფდნენ კონფიდენციალობის დაცვას ინფორმაციის კომპიუტერულ ქსელებში გადაცემის დროს. პირველ ეტაპზე ამ პროგრამას ხელმძღვანელობდა

ჰორსტ ფეისტელი, რომელიც სამოციან წლებში მუშაობდა კ. შენონთან ერთად და რომელიც იმ დროისათვის იყო ცნობილი კრიპტოგრაფი. რთული, კრიპტოგრაფიულად მედეგი სქემების შემუშავება, რომლებიც ამავე დროს აუცილებლად უნდა იყოს შებრუნებადიც, წინააღმდეგ შემთხვევაში მათ გამოყენებას აზრი არა აქვს, არც თუ ისე ადვილია. ამასთან, ასეთი რთული გარდაქმნები აუცილებლად შეიცავენ გარდაქმნის არაეფექტურ ალგორითმებს, რომლებიც მოქმედებენ რა დაშიფრვის სისწრაფეზე, სერიოზულად აფერხებენ ინფორმაციის გადაცემისა და დამუშავების პროცესებს.

ყოველივე ამის გათვალისწინებით ჰ. ფეისტელმა გადაწყვიტა აეგო დაშიფრვის ისეთი სქემა, რომელიც არ გამოიყენებდა ასეთ ალგორითმებს და მთლიანად დაეფუძნებოდა კ. შენონის მიერ ჩამოყალიბებულ კლასიკური კრიპტოგრაფიის ორ ცნობილ პრინციპს, გადანაცვლებებსა და დიფუზიას. მის მიერ შექმნილი სიმეტრიული კრიპტოსისტემის არქიტექტურა, რომელიც დღეს ცნობილია ფეისტელის სქემის სახელით, იმდენად მოხერხებული და მარტივი გამოდგა, რომ გასული საუკუნის ოთხმოც – ოთხმოცდაათიან წლებში შექმნილ კრიპტოსისტემათა აბსოლუტური უმრავლესობა იყენებდა სწორედ ამ სქემას. როგორც სქემიდან (იხ. სურ. 3.1) ჩანს, ჰ. ფეისტელის სქემაში დასაშიფრი ბლოკი იყოფა ორ ნაწილად: მარცხენა (**L**) და მარჯვენა (**R**) (ინდექსი მიუთითებს რაუნდის ნომერს, ანუ მერამდენედ ხდება ამ ოპერაციების ჩატარება დასაშიფრ ბლოკზე). მარჯვენა ნაწილი რაიმე ფუნქციით გარდაიქმნა და შემდეგ **XOR**-ით შეიკრიბება მარცხენა ნაწილთან. მიღებული შედეგი წარმოადგენს შემდეგი რაუნდის მარჯვენა ნაწილს, ხოლო შემდეგი რაუნდის მარცხენა ნაწილს წარმოადგენს წინა რაუნდის მარჯვენა ნაწილი, სანამ ის გარდაიქმნება გარდამქმნელი ფუნქციის საშუალებით.



სურ. 3.1. ჰ. ფეისტელის სქემა

F ფუნქცია როგორც წესი შეიცავს ისეთ მარტივ ოპერაციებს, როგორიცაა დასაშიფრი ინფორმაციის წაძვრის, გადანაცვლებისა და გასაღებთან XOR-ით შეკრების ოპერაციებს. მათემატიკურად ეს სქემა აღიწერება შემდეგი სახით:

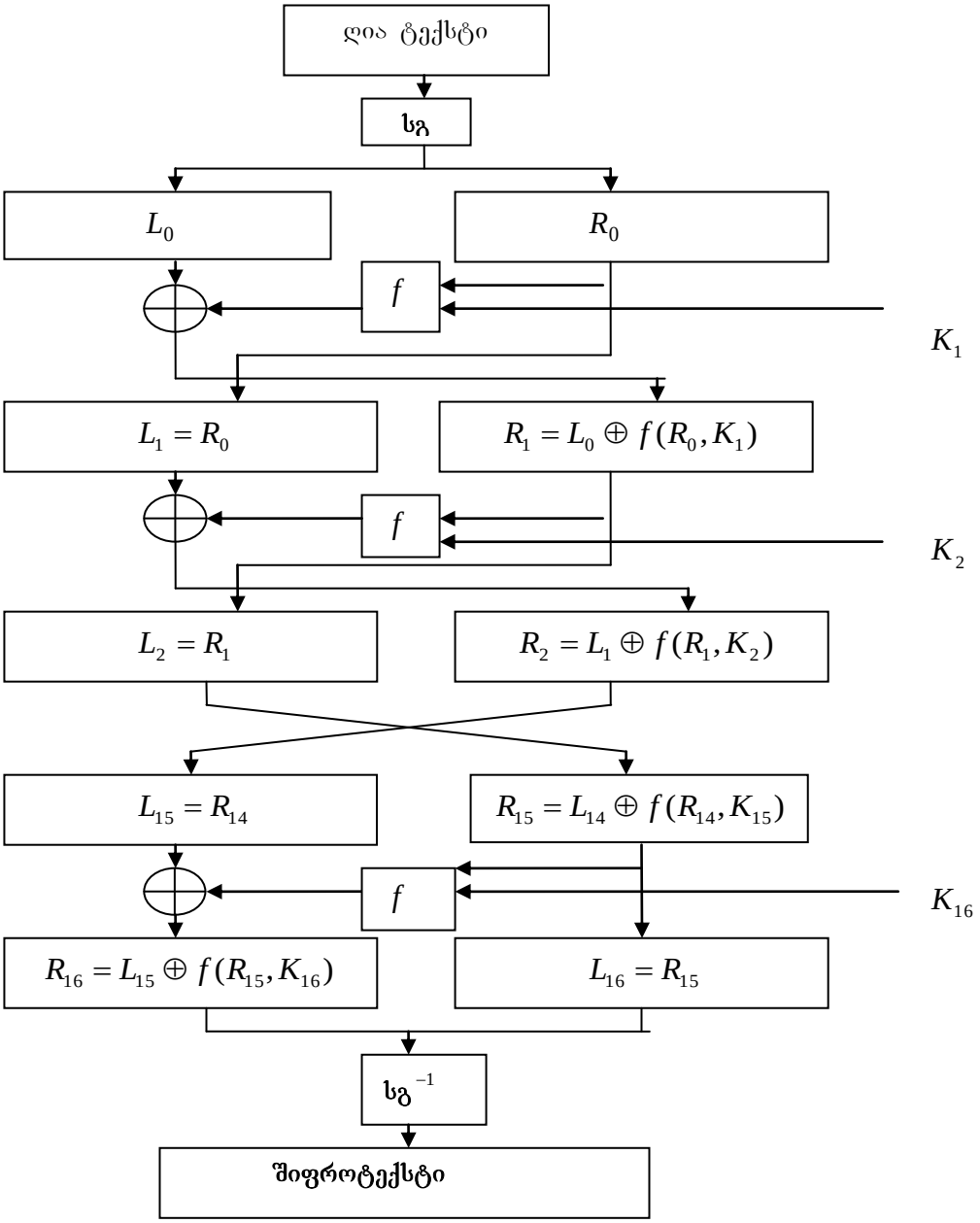
$$L_{i+1} = R_i,$$
$$R_{i+1} = L_i \oplus f_k(R_i)$$

ამ ფორმულებიდან კარგად ჩანს ჰ. ფეისტელის სქემის ძირითადი თვისება, რამაც განაპირობა ამ სქემის ასე ფართოდ გამოყენება, თითოეული რაუნდისათვის სქემა შებრუნებადია ისე, რომ საჭირო არ არის დამშიფრავი ფუნქციის შებრუნება. ანუ ყოველთვის შესაძლებელია L_i და R_i მოძებნა. მართლაც $R_i = L_{i+1}$

$$L_i = R_{i+1} \oplus f(L_{i+1})$$

ასეთი მარტივი სქემა საშუალებას იძლევა ძალიან სწრაფად დაიშიფროს (გაიშიფროს) ინფორმაცია. მედეგობა კრიპტოგრაფიული შეტევების მიმართ კი მიიღწევა ამ პროცედურების მრავალჯერადი განმეორებით, რის შედეგადაც გამოსასვლელი შიფროგრამის თითოეული ბიტი პრაქტიკულად დამოკიდებული ხდება შესასვლელი ტექსტის ყველა ბიტზე (ესაა სწორედ

კლასიკური კრიპტოგრაფიის ერთერთი ძირითადი პრინციპი). ასეთ ალგორითმებს უწოდებენ აგრეთვე იტერაციულ ალგორითმებსაც. სწორედ ამ პრინციპებზე შეიქმნა სამოცდათიანი წლების დასაწყისში **IBM**-ში პირველი დაშიფრვის ალგორითმი ‘‘ლუციფერი’’, რომლის აღწერაც გამოქვეყნდა (მანამდე სამუშაოები კრიპტოგრაფიაში წარმოადგენდა სახელმწიფო საიდუმლოებას). ეს იყო ბლოკური შიფრი, 128 ბიტიანი დასაშიფრი ბლოკით და გასაღებით.



სურ. 3.2. კრიპტოალგორითმ DES -ის სქემა

როდესაც სტანდარტებისა და ტექნოლოგიების ნაციონალურმა ინსტიტუტმა (**ANSI**), რომელსაც მაშინ ეწოდებოდა სტანდარტების ნაციონლური ბიურო (**NBS**), დაიწყო მუშაობა მონაცემთა დაშიფრვის სტანდარტზე, სწორედ ლუციფერი აღმოჩნდა ერთადერთი კანდიდატი, რომელიც შეიძლებოდა გამოეყენებინათ სტანდარტის შესაქმნელად. **NBS**-მა ალგორითმის კრიპტომედეგობის შესაფასებლად დახმარება სთხოვა ამერიკის ნაციონალური უსაფრთხოების ბიუროს, რომელსაც დიდი გამოცდილება ჰქონდა კრიპტოგრაფიული ალგორითმების შექმნასა და გამოყენებაში. ნაციონალური უსაფრთხოების ბიურომ შეიტანა რამდენიმე ცვლილება ალგორითმ-

ში, კერძოდ, დასაშიფრი ბლოკის სიგრძე შეამცირა სამოცდაოთხ ბიტამდე, გასაღების სიგრძე ორმოცდათექვსმეტ ბიტამდე და მთლიანად შეცვალა **S** ბლოკების სტრუქტურა. თუ ლუციფერში **S** ბლოკში შედიოდა ოთხი ბიტი და გამოდიოდა ოთხი ბიტი, აქ უკვე შედიოდა ექვსი ბიტი და გამოდიოდა ოთხი ბიტი, ამასთან ბლოკების სტრუქტურა წარმოადგენდა ცხრილებს, რომლის შედგენის მეთოდიკაც უცნობი იყო სპეციალისტებისათვის. ასეთმა ცვლილებებმა გამოიწვია სპეციალისტთა ნაწილის უნდობლობა. ისინი ფიქრობდნენ, რომ ეს ცვლილებები საშუალებას მისცემდა ნაციონალური უსაფრთხოების ბიუროს გაეკონტროლებინა კერძო სექტორის მიერ დაშიფრული ინფორმაცია. მიუხედავად ასეთი უნდობლობისა, ამ შესწორებების შემდეგ 1975 წელს ახალი ალგორითმი, რომელსაც ეწოდა **DES, NBS**-მა მიიღო სტანდარტად.

3.2 DES-ის (Data Encryption Standard) სტრუქტურა. ალგორითმის სქემა მოყვანილია სურ. 3.2. ალგორითმი წარმოადგენს სიმეტრიულ ბლოკურ შიფრს, რომელიც მუშაობს სამოცდაოთხ ბიტიან ტექსტთან და გამოიყენებს ორმოცდათექვსმეტ ბიტიან გასაღებს. ფორმალურად გასაღებიც შედგება სამოცდაოთხი ბიტისაგან, მაგრამ გასაღების გადაცემის დროს საიმედოობის გაზრდის მიზნით ყოველი მერვე ბიტი წარმოადგენს ლუწობა–კენტობის მაკონტროლებელ ბიტს, ამიტომ გასაღების გამოყენებამდე ეს ბიტები ამოიყრება გასაღებიდან. ალგორითმი შედგება თექვსმეტი რაუნდისაგან, ანუ ერთი და იგივე პროცედურები დასაშიფრ ბლოკზე მეორდება თექვსმეტჯერ. ასეთ ალგორით-მებს უწოდებენ იტერაციულ ალგორითმებს. განსხვავებულია მხოლოდ მეთექვსმეტე რაუნდის შესასვლელი. როგორც უკვე აღვნიშნეთ, ერთი და იგივე პროცედურის რამდენჯერმე (კერძოდ, თექვსმეტჯერ გამეორების მიზანია, რომ გამოსასვლელი შიფროტექსტის თითოეული ბიტი დამოკიდებული იყოს ღია ტექსტის ყველა ბიტზე. სანამ დასაშიფრი ტექსტი მოხვდება პირველ რაუნდში, ხდება მისი საწყისი გადანაცვლება, რომელიც (ისევე როგორც მიღებული შიფროტექსტის საბოლოო გადანაცვლება) არავითარ გავლენას არ ახდენს ალგორითმის მედეგობაზე. ეს გადანაცვლება სრულდება სპეციალური ცხრილის საშუალებით (ცხრილი 3.1) და როგორც ალგორითმის ავტორები ხსნიდნენ, სრულდება იმისათვის, რომ გაადვილდეს ღია და შიფრო ტექსტების ბაიტების შეტანა **DES**-ს იმდროინდელ მიკროსქემაში მისი აპარატურული რეალიზაციის დროს.

თუ გამოიყენება **DES**-ის პროგრამული რეალიზაცია, ეს პროცედურა შეიძლება ამოვიღოთ ალგორითმიდან, რადგანც მისი პროგრამული რეალიზაცია აპარატურულისაგან განსხვავებით საკმაოდ რთულია (მაგრამ უნდა გავითვალისწინოთ, რომ ამ შემთხვევაში ალგორითმი უკვე აღარ იქნება სტანდარტი).

გადანაცვლების ეს ცხრილი (ისევე, როგორც დანარჩენი ცხრილები ამ პარაგრაფში) უნდა წავიკითხოთ ასე: დასაშიფრი ტექსტის ბიტი გადადის პირველ ბიტში, – მეორეში და ასე შემდეგ.

ცხრილი 3.1. საწყისი გადანაცვლება

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

ცხრილი 3.2. გასაღების საწყისი გადანაცვლება

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

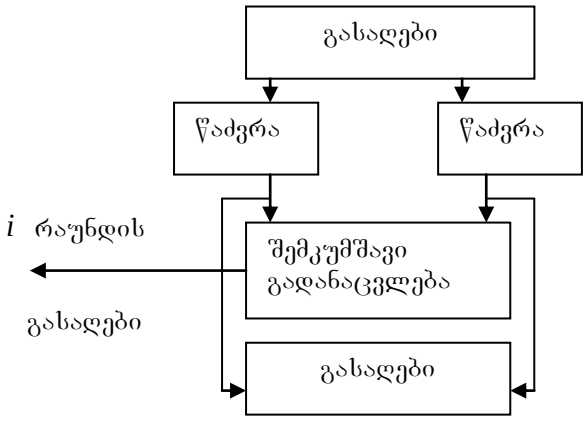
3.3. რაუნდის გასაღების გამომუშავება. სანამ დავიწყებდეთ უშუალოდ ალგორითმის ერთ რაუნდში მიმდინარე გარდაქმნების განხილვას, ჯერ ვნახოთ, როგორ გამომუშავდება გასაღები თითოეული რაუნდისათვის. როგორც ალგორითმის სქემიდან ჩანს (იხ. სურ. 3.2), თითოეული რაუნდისათვის გვაქვს ცალკე გასაღები (სულ თექვსმეტი). ცხადია, რომ თექვსმეტი დამოუკიდებელი გასაღების დამახსოვრება შეუძლებელია, ამიტომ ალგორითმს აქვს ერთი საწყისი გასაღები, რომლიდანაც შემდეგ გამომუშავდება გასაღებები თითოეული რაუნდისათვის. ეს გასაღები პირველ რაუნდში შესვლამდე გარდაიქმნება სპეციალური ცხრილის მიხედვით (იხ. ცხრილი 3.2).

ამის შემდეგ, ყოველ რაუნდში გასაღები გაიყოფა ორ ქვებლოკად, თითოეული ოცდარვა ბიტის ოდენობით და ხდება მათი ციკლური წაძვრა მარცხნივ (იხ. სურ. 3.3). წაძრული ბიტების

რაოდენობა დამოკიდებულია რაუნდის ნომერზე. კერძოდ რაუნდებისათვის, რომელთა ნომრებია 1,2,3,9 წაძვრა ხდება ერთი ბიტით, დანარჩენი რაუნდებისათვის კი – ბიტით.

წაძვრის შემდეგ ორმოცდათექვსმეტი ბიტიდან ამოირჩევა ორმოცდარვა ბიტიანი რაუნდის გასაღები, ამასთან ერთად კვლავ ხდება ბიტების მიმდევრობის შეცვლა, ამიტომ ამ პროცესს ეწოდება შემკუმშავი გადანაცვლება და სრულდება სპეციალური ცხრილის საშუალებით (ცხრილი 3.3).

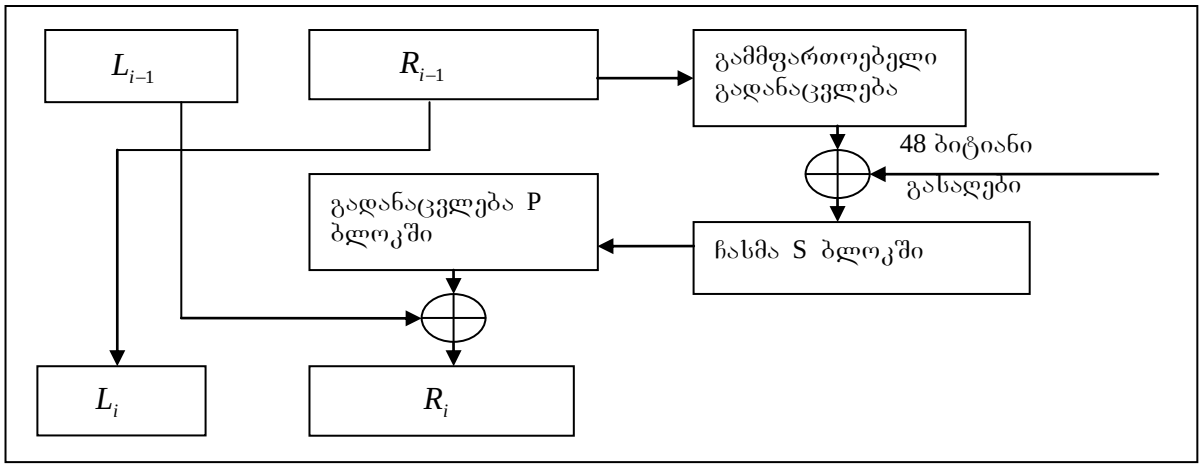
ასეთი ოპერაციები გასაღებზე უზრუნველყოფს ყოველ რაუნდში განსხვავებული გასაღების გამოყენებას. საწყისი გასაღების თითოეული ბიტი მონაწილეობს დაახლოებით თოთხმეტ რაუნდში თექვსმეტიდან, თუმცა გასაღების ყველა ბიტი ერთნაირი სიხშირით არ გამოიყენება.



სურ.3.3. რაუნდის გასაღების გამომუშავების სქემა

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

ცხრილი 3.3. გასაღების შემკუმშავი გარდაქმნა

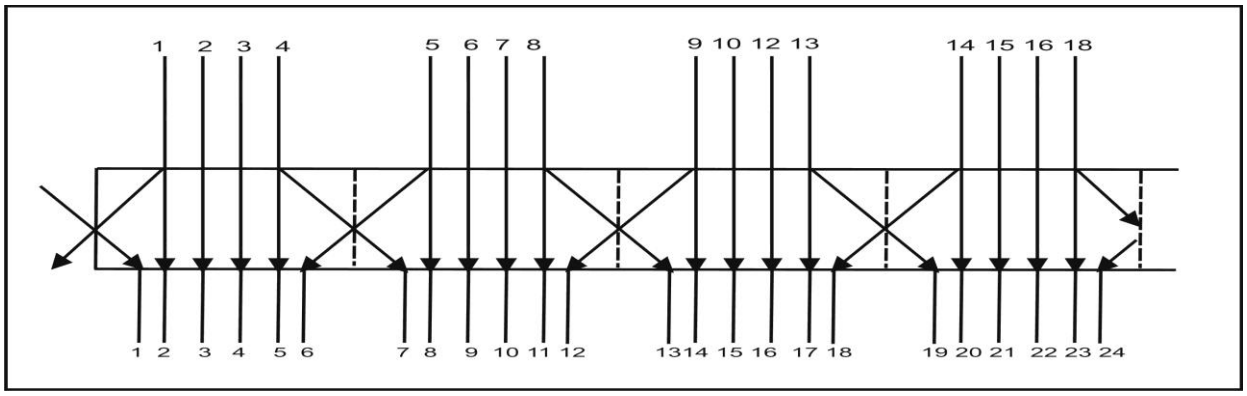


სურ. 3.4 DES-ის ერთი რაუნდის სქემა

3.4. ოპერაციები DES-ის ერთ რაუნდში. როგორც უკვე ვიცით, **DES**-ი აგებულია ფეისტელის სქემაზე, ამიტომ დასაშიფრი სამოცდაოთხბიტიანი ბლოკი იყოფა ორ ოცდათორმეტბიტიან მარჯვენა (**R**) და მარცხენა (**L**) ნახევრებად (იხ. სურ. 3.4). ამასთან მარჯვენა ნახევარი წარმოადგენს შემდეგ რაუნდში შესასვლელი ბლოკის მარცხენა ნახევარს (სანამ მასზე ჩატარდება რაუნდის ოპერაციები). ერთ რაუნდში მარჯვენა ნახევარზე ტარდება შემდეგი ოპერაციები: გამფართოებელი გადანაცვლება, **XOR**-ით შეკრება რაუნდის გასაღებთან, **S** ბლოკებში ჩასმა, გადანაცვლება **P** ბლოკში და **XOR**-ით შეკრება თავის მარცხენა ნახევართან. მიღებული შედეგი წარმოადგენს შემდეგი რაუნდის შესასვლელი ბლოკის მარჯვენა ნახევარს. გამონაკლის წარმოადგენს მეთექვსმეტე რაუნდი (იხ. სურ. 3.2).

რადგანაც ერთ რაუნდში გარდაქმნას განიცდის დასაშიფრი ბლოკის ნახევარი, იმისათვის, რომ მთლიანად ტექსტზე თანაბრად ჩატარდეს გარდაქმნები, საჭიროა გვქონდეს რაუნდების ლუწი რაოდენობა. განვიხილოთ რაუნდის თითოეული ოპერაცია ცალ-ცალკე.

3.5. დასაშიფრი ტექსტის გაფართოება. DES-ის რაუნდი იწყება გამფართოებელი გადანაცვლებით, ანუ ერთდროულად ხდება ღია ტექსტის ბიტების გადანაცვლება და ამასთან ოცდათორმეტი შესასვლელი ბიტიდან მიიღება ორმოცდარვა გამოსასვლელი ბიტი. ამისათვის შესასვლელი ბიტების ნახევარი (თექვსმეტი ბიტი) მეორდება (იხ.სურ. 3.5.). ამ ოპერაციის კრიპტოგრაფიული მიზანია, რაც შეიძლება სწრაფად, ე.წ. ზვავისებური ეფექტით მოხდეს გამოსასვლელი ბიტების დამოკიდებულება შესასვლელი ბიტების მაქსიმალურ რაოდენობაზე. ეს მიიღწევა იმით, რომ ღია ტექსტის ერთი ბიტი ფაქტობრივად მონაწილეობს ორი გამოსასვლელი ბიტის შექმნაში, რაც შემდეგ რაუნდში კიდევ უფრო სწრაფად გაზრდის ამ დამოკიდებულებას. ამასთან, მიუხედავად იმისა, რომ შესასვლელი ბლოკი უფრო პატარაა, ვიდრე გამოსასვლელი, გამფართოებელი გადანაცვლების სქემა ისეა შედგენილი, რომ დაცულია ურთიერთცალსახა შესაბამისობა შესასვლელ და გამოსასვლელ ბლოკებს შორის გამფართოებელი გადანაცვლება სრულდება სპეციალური ცხრილის საშუალებით (იხ. ცხრილი 3.4). როგორც ამ ცხრილიდან ჩანს, მეორდებიან 32,1,4,5,8,9,12,13,16,17,20,21,24,25,28,29 ბიტები.



სურ. 3.5. გამფართოებელი გადანაცვლების სქემა

ცხრილი 3.4. გამფართოებელი გადანაცვლება

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

3.6. შეკრება გასაღებთან. რაუნდში ქვეგასაღები გამოიყენება მხოლოდ ერთხელ. გამფართოებელი გადანაცვლებიდან გამოსული ტექსტის ორმოცდარვა ბიტი ოპერაცია **XOR**-ით იკრიბება რაუნდის ორმოცდარვაბიტიან ქვეგასაღებთან.

3.7. S ბლოკები. მას შემდეგ, რაც **XOR**-ით შეიკრიბება შეკუმშული გასაღები და დასაშიფრი ბლოკის გაფართოებული ნახევარი, ალგორითმში სრულდება ყველაზე მნიშვნელოვანი ოპერაცია, ჩასმა **S**

S1	ბ	ჟ	ო	კ	ი										
14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2	ბ	ჟ	ო	კ	ი										
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S3	ბ	ჟ	ო	კ	ი										
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S4	ბ	ჟ	ო	კ	ი										
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S5	ბ	ჟ	ო	კ	ი										
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S6	ბ	ჟ	ო	კ	ი										
12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S7	ბ	ჟ	ო	კ	ი										
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8	ბ	ჟ	ო	კ	ი										
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	3	6	11

სურ. 3.6. **S** ბლოკების ცხრილები

–ბლოკებში, რის შედეგადაც ორმოცდარვა ბიტისაგან კვლავ მიიღება ოცდათორმეტბიტიანი შიფროტექსტი. ეს ოპერაცია მნიშვნელოვანია იმით, რომ ის წარმოადგენს ერთადერთ არაწრფივ ოპერაციას. ყველა სხვა დანარჩენი ოპერაციები, იქნება ეს **XOR**-ით შეკრება, წაძვრა თუ

გადანაცვლება, წარმოადგენენ წრფივ ოპერაციებს, მაშინ როდესაც ჩასმა **S** ბლოკების საშუალებით წარმოადგენს არაწრფივ ოპერაციას.

S ბლოკების შედგენის მომენტი იყო ყველაზე უფრო გაუმჭვირვალე მომენტი ალგორითმის შექმნის პროცესში. რაღაც ცხრილები (იხ. სურ. 3.6) რომლებიც შესულ ექვს ბიტს ინფორმაციას შეუსაბამებენ გამოსასვლელ ოთხ ბიტს და რომლებიც ნაციონალური უსაფრთხოების სააგენტომ შეასწორა თავისი შეხედულების მიხედვით. ეს გახდა უნდობლობის საფუძველი მრავალი სპეციალისტისათვის, რომლებიც ფიქრობდნენ, რომ სააგენტო მოქმედებდა გამომდინარე თავისი ინტერესებიდან. მხოლოდ 1990 წელს, დიფერენციალური კრიპტოანალიზის გამოქვეყნების შემდეგ კორპორაცია **IBM** გაამჟღავნა **S** და **P** ბლოკების პროექტირების კრიტერიუმები. ეს კრიტერიუმებია:

- თითოეულ **S** ბლოკს გააჩნია ექვსი შესასვლელი და ოთხი გამოსასვლელი, რაც განპირობებულია იმით, რომ ესაა მაქსიმალური ზომა, რომლის განხორციელებაც შესაძლებელი იყო წელს ერთ მიკროსქემაში;
- **S** ბლოკის არცერთი გამოსასვლელი ბიტი არ უნდა იყოს ძალიან ახლოს შესასვლელი ბიტების წრფივ კომბინაციასთან;
- თუ დავაფიქსირებთ პირველ და მეექვსე ბიტს და ვცვლით დარჩენილ ოთხ ბიტს, ყველა გამოსასვლელი ოთხბიტაინი რიცხვი უნდა იყოს განსხვავებული;
- თუ **S** ბლოკის ორი შესასვლელი სტრიქონი განსხვავდება მხოლოდ ერთი ბიტით, გამოსასვლელი სტრიქონები უნდა განსხვავდებოდნენ ორი ბიტით მაინც;
- თუ **S** ბლოკის ორი შესასვლელი სტრიქონი განსხვავდებიან ერთმანეთისაგან მხოლო ორი ცენტრალური ბიტით, გამოსასვლელი სტრიქონები უნდა განსხვავდებოდნენ ორი ბიტით მაინც;
- თუ **S** ბლოკის ორი შესასვლელი სტრიქონი განსხვავდება მხოლოდ პირველი ორი ბიტით და ბოლო ორი ბიტი ერთნაირია, გამოსასვლელი სტრიქონები არ უნდა იყოს ერთნაირი;
- ნებისმიერი ორი შესასვლელი სტრიქონის არანულოვანი ექვსბიტიანი სხვაობისათვის მხოლოდ რვა წყვილს ოცდათორმეტიდან შეიძლება შეესაბამებოდეს გამოსასვლელების ერთაინი სხვაობა;
- იგივე პირობა უნდა სრულდებოდეს სამი აქტიური **S** ბლოკისათვისაც.

დღევანდელი კოპიუტერული ტექნიკის საშუალებით **S** ბლოკების დაპროექტება, რომლებიც დააკმაყოფილებენ ამ კრიტერიუმებს ადვილია, მაგრამ სამოცდაათიანი წლების დასაწყისში **S** ბლოკების დაპროექტებას სჭირდებოდა რამდენიმე თვე.

<u>3.8. S ბლოკების მუშაობა.</u> ალგორითმში ორმოცდარვა ბიტი, რომელიც მიიღება შეკრების შემდეგ, იყოფა რვა ქვებლოკად ექვს-ექვსი ბიტის ოდენობით და შედის რვა ცალ **S** ბლოკში შემდეგი ალგორითმით: შესასვლელი ბიტების პირველი და მეექვსე ბიტი ერთიანდება და აღინიშნავს სპეციალური ცხრილის სტრიქონის ნომერს, ხოლო ბიტები მეორედან მეხუთის ჩათვლით კი სვეტის ნომერს (დანომრვა იწყება ნოლიდან). ცხრილის უჯრაში მოთავსებულია რიცხვი ნოლიდან ხუთმეტის ჩათვლით, რომელიც (ჩაწერლი ორობით სისტემაში) წარმოადგენენ ბლოკის გამოსასვლელს. თითოეულ ბლოკს აქვს თავისი ცხრილი (იხ. სურ. 3.6) ამგვარად. პრაქტიკულად ბლოკში შედის ექვსი და გამოდის ოთხი ბიტი, შედეგად კვლავ ვღებულობთ ოცდათორმეტ ბიტს.

<u>3.9. გადანაცვლების P ბლოკი და შეკრება მეორე ნახევართან.</u> ოცდათორმეტი ბიტი, რომელიც გამოდის რვა **S** ბლოკიდან ოთხ-ოთხი ბიტის სახით, კვლავ ერთიანდება ოცდათორმეტ ბიტად, მაგრამ არა იმავე მიმდევრობით, როგორითაც ისინი გამოდიან **S** ბლოკებიდან. ხდება მათი გადანაცვლება **P** ბლოკში სპეციალური ცხრილის საშუალებით. გადანაცვლების ბლოკსაც გააჩნია თავისი კრიტერიუმები კერძოდ:

- რაუნდში **S** ბლოკებიდან გამოსული ოთხი ბიტი უნდა გადანაწილდეს ისე, რომ მათგან ორმა ბიტმა რაუნდში გავლენა მოახდინოს **S** ბლოკების შუა ბიტებზე დანარჩენმა ორმა კი – ბოლო ბიტებზე.
- თითოეული **S** ბლოკების ოთხი გამოსასვლელი ბიტი გავლენას ახდენენ ექვს სხვადასხვა **S** ბლოკზე და ამასთან არცერთ ბლოკზე არ მოქმედებს ორი ბიტი ერთდროულად;
- თუ **S** ბლოკის ერთი გამოსასვლელი ბიტი მოქმედებს სხვა **S** ბლოკის შუა ბიტებზე, მაშინ ამ სხვა **S** ბლოკის გამოსასვლელი ბიტი არ შეიძლება მოქმედებდეს პირველი **S** ბლოკის შუა ბიტებზე.

თექვსმეტი ასეთი რაუნდის გამეორების შემდეგ (მეთექვსმეტე რაუნდში შესასვლელი განსხვავებულია დანარჩენი რაუნდებისაგან) მიღებული შიფროტექსტი განიცდის საწყისი გადანაცვლების შებრუნებულ გადანაცვლებას და მზადაა გადასაცემად ან შესანახად. მიუხედავად იმისა, რომ ღია ტექსტი შიფროგრამად გადაქცევამდე განიცდის ასეთ მრავალ გარდაქმნას, ალგო-

რითმი ძალიან სწრაფია როგორც სპეციალური მიკროსქემებით განხორციელებისას, ასევე მისი პროგრამული სახით გამოყენებისას.

ცხრილი 3.5. გადანაცვლება **P** ბლოკში

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

3.10. DES-ის რამდენიმე საინტერესო თვისება. პირველ რიგში აღვნიშნოთ ის სუსტი მხარეები, რაც გააჩნია **DES**-ს. ესაა ***სუსტი და ნახევრად სუსტი გასაღებები.*** რადგანაც გასაღები ალგორითმში გამოიყენება ერთხელ, დასაშიფრ ბლოკთან ოპერაცია **XOR**-ით შესაკრებად, ცხადია, რომ გასაღებები, რომლებიც შედგებიან მხოლოდ ორმოცდათექვსმეტი ნოლისაგან, ან ამდენივე ერთიანისაგან, იქნება ***სუსტი გასაღები.*** რადგანაც რაუნდის გასაღების გამომუშავების დროს საწყისი გასაღები იყოფა ორ ნაწილად და შემდეგ ხდება მათზე ოპერაციები, ასევე სუსტი იქნება გასაღებები, რომლებიც შედგებიან ოცდარვა ერთიანისა და ოცდარვა ნოლისაგან, ან პირიქით. ანუ აშკარად სუსტი გასაღებების რაოდენობაა ოთხი.

გარდა ასეთი აშკარად სუსტი გასაღებებისა, არსებობენ კიდევ ***ნახევრად სუსტი გასაღებები***. საქმე იმაშია, რომ ზოგიერთ წყვილს გასაღებებისა ღია ტექსტი გადაყავთ ერთსა და იმავე დაშიფრულ ტექსტში. ანუ ერთი გასაღებით დაშიფრული ტექსტი შესაძლებელია გაიშიფრის მეორე გასაღებით. ეს არის შედეგი ალგორითმში გამოყენებული რაუნდის გასაღების გამომუშავების მეთოდისა. ეს გასაღებები თექვსმეტის ნაცვლად გამოიმუშავებენ მხოლოდ ორ გასაღებს. ასეთი გასაღებების რაოდენობა ტოლია თორმეტის. არსებობენ გასაღებები, რომლებიც გამოიმუშავებენ რაუნდის მხოლოდ ოთხ გასაღებს. ასეთ გასაღებებს უწოდებენ ***შესაძლო სუსტ გასაღებებს***. მათი რაოდენობა უდრის ორმოცდარვას. ამგვარად, ალგორითმს გააჩნია მხოლოდ სამოცდაოთხი სუსტი, ნახევრად სუსტი ან შესაძლო სუსტი გასაღები, მაშინ როდესაც გასაღებების რაოდენობა ტოლია 2^{56}-ის. გარდა ამისა, ალგორითმს გააჩნია ***კომპლანარობის*** თვისება, რაც გამოიხატება შემდეგში: თუ M არის ღია ტექსტი და $\overline{M}$ არის კომპლანარული ღია ტექსტი, ანუ ტექსტი, რომელშიც ერთიანები შეცვლილია ნოლებით და პირიქით, K არის გასაღები და $\overline{K}$ კი კომპლანარული გასაღებია, რომელშიც ასევეა ერთიანები შეცვლილია ნოლებით და პირიქით, მაშინ ალგორითმისათვის სამართლიანია პირობა $E = f_K(M) \Rightarrow \overline{E} = f_{\overline{K}}(\overline{M})$. ეს იმას ნიშნავს, რომ ასეთი გასაღებებისათვის ძალისმიერი შეტევის შემთხვევაში საჭიროა 50%-ით ნაკლები შეტევის განხორციელება, როგორც ამას **DES**-ის მოწინააღმდეგენი აცხადებენ, მაგარამ ეს სინამდვილეში ნიშნავს, რომ საჭირო იქნება 2^{56} გადარჩევის ნაცვლად "მხოლოდ" 2^{55} გადარჩევის ჩატარება, ამიტომ ეს ფაქტორი დიდად არ მოქმედებს ალგორითმის მედეგობაზე.

უფრო საინტერესო აღმოჩნდა **DES**-ის ის თვისება, რომ მისი გასაღებების სიმრავლე არ ქმნის ჯგუფს, ანუ, თუ თქვენ ჯერ ერთი გასაღებით დაშიფრავთ ღია ტექსტს და შემდეგ მიღებულ შიფროგრამას გადაშიფრავთ მეორე გასაღებით, არ მოიძებნება ისეთი მესამე გასაღები, რომელიც ღია ტექსტს გადაიყვანს პირდაპირ მეორე შიფროგრამაში. ეს უკვე ალგორითმის ძლიერი თვისებაა, რომელიც გამოიყენეს ოთხმოცდაათიან წლებში, როდესაც ეშინოდათ, რომ შესაძლებელი იყო გაეშიფრათ ალგორითმი. **DES**-ის მაგივრად დაიწყეს **3DES**-ის გამოყენება, რომელშიც ღია ტექსტი დაიშიფრება მიმდევრობით სამჯერ სამი სხვადასხვა გასაღების საშუალებით (ორმაგი **DES**-ის გამოყენება არ შეიძლება რადგანაც არსებობს შეტევა, რომლითაც ასეთ შემთხვევაში შეიძლება გავტეხოთ ალგორითმი).

შესაძლებელია მნიშვნელოვნად გაიზარდოს ალგორითმის მედეგობა ძალისმიერი შეტევის მიმართ, თუ რაუნდების გასაღებები გამომუშავდება არა ერთი საწყისი გასაღებიდან, არამედ აიღება თექვსმეტი დამოუკიდებელი გასაღები.

3.11. დეშიფრაცია. ალგორითმი წარმოადგენს აბსოლუტურად სიმეტრიულ ალგორითმს. როგორც უკვე აღვნიშნეთ, ამას განაპირობებს ჰ. ფეისტელის სქემა, ამიტომ ალგორითმის დაშიფრვისა და დეშიფრაციის და ფუნქციაც ერთმანეთისაგან არ განსხვავდება. რომელი ფუნქციითაც და რა მომდევრობითაც იშიფრება ღია ტექსტი, იგივე ფუნქციით და იგივე მიმდევრობით ხდება დეშიფრაციაც. ერთადერთი განსხვავება ეხება რაუნდის გასაღებებს. თუ დაშიფრვის რეჟიმში გასაღებები გამოიყენება მიმდევრობით $K_1, K_2, ..., K_{16}$, გაშიფრვის რეჟიმში უნდა გამოვიყენოთ გასაღებები $K_{16}, K_{15}, ..., K_1$ მიმდევრობით.

3.12. DES-ის ვარიანტები. დამოუკიდებელი ქვეგასაღებები. შესაძლებელია თითოეულ რაუნდში გამოვიყენოთ დამოუკიდებელი ქვეგასაღებები. ასეთ შემთხვევაში გასაღების სიგრძე იქნება ბიტი და ძალისმიერი შეტევისათვის უკვე საჭირო გახდება ვარიანტის გადარჩევა და რაც უფრო მთავარია შეუძლებელი გახდება შეტევა დაკავშირებული გასაღებების მეთოდით.

ალგორითმი DESX. ამ ვარიანტში ბლოკში შესასვლელი ღია ტექსტისა და გამოსასვლელი შიფროტექსტის შენიღბვის მიზნით გასაღებს ემატება 64 ბიტი, რომელიც **XOR** ოპერაციით იკრიბება ღია ტექსტთან პირველი ეტაპის წინ (ამას კრიპტოგრაფიაში უწოდებენ ტექსტის გათეთრებას). კიდევ 64 ბიტი გამოითვლება როგორც ცალმხრივ მიმართული ფუნქცია სრული, 120 ბიტიანი გასაღებისა და კვლავ **XOR** ოპერაციით იკრიბება მეთექვსმეტე რაუნდიდან გამოსულ შიფროტექტთან. გარდა იმისა, რომ ამ ცვლილებით იზრდება გამძლეობა ძალისმიერი შეტევის მიმართ, დიფერენციალური და წრფივი შეტევების დროს საჭირო ხდება უკვე შესაბამისად 2^{61} შერჩეული და 2^{60} ღია ტექსტი.

ალგორითმი CRYPT(3). ეს ვარიანტი გამოიყენება UNIX-ის სისტემებში როგორც ცალმხრივ მიმართული ფუნქცია პაროლების შესანახად, თუმცა ზოგჯერ გამოიყენება დასაშიფრადაც. განსხვავებაა ის, რომ ამ ვარიანტში გამოიყენება გასაღებზე დამოკიდებული გამფართოებელი გადანაცვლება, რომელიც უზრუნველყოფს გადანაცვლებას. ეს გაკეთებულია იმისათვის, რომ შეუძლებელი გახდეს **DES**-ის სერიული მიკროსქემების გამოყენება პაროლების გასატეხი მოწყობილობის შესაქმნელად.

განზოგადოებული ალგორითმი GDES. ესაა **DES**-თან შედარებით უფრო სწრაფი ალგორითმი, რომელშიც გაიზარდა ბლოკის ზომა, გამოთვლების მოცულობა კი იგივე დარჩა. დაშიფრვის ბლოკი შეიძლება იყოს ნებისმიერი ზომის (მაგრამ ფიქსირებული მოცემული რეალიზაციისათვის). ბლოკი იყოფა 32 ზომის q რაოდენობის ქვებლოკებად. ამ ალგორითმის მედეგობა კატასტროფულად შესუსტდა. იმისათვის, რომ გატყდეს GDES, რომლისთის რაუნდების რაოდენობაა თექვსმეტი და $q = 8$, საჭიროა მხოლოდ ექვსი შერჩეული ტექსტი და თუ ქვეგასაღებები იქნება დამოუკიდებლები – სულ 16 შერჩეული ტექსტი.

გასაღებზე დამოკიდებული S-ბლოკებიანი DES-ი. დიფერენციალური და წრფივი კრიპტოანალიზი შესაძლებელია წარმატებით გამოვიყენოთ, თუ ცნობილია **S**-ბლოკების სტრუქტურა და გაცილებით რთულდება, როდესაც ეს სტრუქტურა დამოკიდებულია გასაღებზე. არსებობს **DES**-ის ვარიანტი, რომელშიც დამატებულია გასაღებზე ორმოცდარვა ბიტი S-ბლოკების შესაქმნელად. ეს პროცედურა სრულდება შემდეგი ბიჯებით:

1. იცვლება S-ბლოკების მიმდევრობა: 24673158 .
1. ავიღოთ დამატებითი გასაღების თექვსმეტი ბიტი. თუ პირველი ბიტი ტოლია ერთის, შევუცვალოთ ადგილები პირველი S-ბლოკის პირველ ორ და ბოლო ორ სტრიქონებს, თუ მეორე ბიტი უდრის ერთს, შევიცვალოთ ადგილები პირველ და ბოლო რვა სვეტებს. გავიმეოროთ იგივე პროცედურა მეორე ბლოკისათვის მესამე და მეოთხე ბიტების მიხედვით. გავაკეთოთ იგივე დანარჩენი ბლოკებისთვის.
2. დარჩენილი ბიტიდან ავიღოთ ოთხი ბიტი და შევკრიბოთ XOR ოპერაციით პირველი ბლოკის ყველა ელემენტთან. გავიმეოროთ ეს ოპერაცია დარჩენილი ბიტებით ყველა S-ბლოკთან.

გატეხვის სირთულე ძალისმიერი შეტევისათვის გახდება 2^{102}, დიფერენციალური კრიპტოანალიზის მიმართ – 2^{51}, ხოლო წრფივი კრიპტოანალიზის მიმართ – 2^{53}.

3.13. აუცილებელი ხდება დაშიფრვის ახალი სტანდარტი. მიუხედავად იმისა, რომ თავიდან სპეციალისტთა ნაწილი უნდობლობას უცხადებდა **DES**-ს, აღმოჩნდა, რომ ყველა პარამეტრებით ის იყო ერთერთი საუკეთესო ალგორითმი. ამას ადასტურებს ის ფაქტიც, რომ მიუხედავად იმისა, რომ 1975 წელს ის სტანდარტად დაამტკიცეს მხოლოდ ხუთი წლის ვადით, სინამდვილეში ის იყო

სტანდარტი 2001 წლამდე. ამ ხნის განმავლობაში არაერთხელ იყო მცდელობა შეეცვალათ სტანდარტი, გამოცხადდა რამდენიმე კონკურსი, მაგრამ არცერთი კანდიდატი არ იყო DES-ზე უკეთესი. მხოლოდ 1999 წელს, როდესაც ალგორითმი მორალურად მოძველდა და აგრეთვე გაჩნდა საშიშროება, რომ ახალი ტექნიკური საშუალებებით შესაძლებელი იქნებოდა ალგორითმის გატეხვა ძალისმიერი შეტევის საფუძველზე, საკონკურსო კომისიამ შეარჩია ახალი სტანდარტის კანდიდატი, რომელიც 2001 წელს დამტკიცებული იქნა ახალ სტანდარტად.

3.14. AES (Advanced Encryption Standard) Rijndael-ს არითმეტიკა. განსხვავებით DES-ისაგან, **Rijndael**-ში ყველა ოპერაცია, რომელიც სრულდება დასაშიფრ ბლოკზე, ეფუძნება არითმეტიკას $FG(2^8)$ ველში, ამიტომ აქ ყველაფერი გასაგებია და ნებისმიერი ადამიანისათვის ადვილი შესამოწმებელია ალგორითმის სუსტი და ძლიერი მხარეები. ალგებრიდან ჩვენთვის ცნობილია, რომ ერთი და იგივე რიგის ყველა სასრული ველი იზომორფულია და ერთმანეთისაგან განსხვავ-დებიან მხოლოდ ელემენტების წარმოდგენით. **Rijndael**-ში მიღებულია კლასიკური, პოლინომური წარმოდგენა, სადაც რიცხვები წარმოიდგინება რვაბიტიანი (ბაიტიანი) ვექტორების სახით. რაიმე ბაიტი **b**, რომლიც შედგება $b_7, b_6, ...b_0$ ბიტებისაგან, წარმოადგენს პოლინომს კოეფიციენტებით {0,1} სიმრავლიდან, რომელსაც აქვს შემდეგი სახე:

$$b_7x^7 + b_6x^6 + ... + b_0$$

ამასთან, კომპიუტერში ინახება მხოლოდ ვექტორი, რომელიც შედგენილია ამ პოლინომის კოეფიციენტებისაგან და რომელიც აღიქმება როგორც თექვსმეტობითი რიცხვი. მაგალითად, ბაიტი, რომლის თექვსმეტობითი მნიშვნელობაა $(57)_{hex}$ და ორობითი მნიშვნელობა კი 01010111, შეესაბამება პოლინომს

$$x^6 + x^4 + x^2 + x + 1$$

ასეთი წარმოდგენის პირობებში ორი პოლინომის შეკრების ოპერაცია პრაქტიკულად წარმოადგენს მსგავსი წვერების კოეფიციენტების **XOR**- ით შეკრებას.

პოლინომურ წარმოდგენაში $FG(2^8)$ ველში რიცხვების გამრავლება შეესაბამება პოლინომების გამრავლებას რომელიმე მერვე ხარისხის დაუყვანადი ბინარული პოლინომის მოდულით. **Rijndael**-ში ასეთ პოლინიმად აღებულია

$$m(x) = x^8 + x^4 + x^3 + x + 1$$

გამრავლების მიმართ ერთეულოვანი ელემენტია $(01)_{hex}$. შეკრებისაგან განსხვავებით, გამრავლების დროს უკვე აღარ გვაქვს მარტივი ოპერაცია ბიტების დონეზე. იმისათვის, რომ ადვილად განხორციელდეს გამრავლების ოპერაცია, ავტორებმა გამოიყენეს თვისება, რომ ორობით სისტემაში რიცხვის ორზე გამრავლება ტოლფასია რიცხვის ერთი თანრიგით მარცხნივ წაძვრის და შემდეგ, თუ საჭიროა უნდა მოვახდინოთ XOR-ით შეკრება $(1B)_{hex}$ -თან (ეს რიცხვი პოლინომურ წარმოდგენაში ტოლია მოდულის ფუძის). $xtime(\ \ )$ აღნიშნულია ფუნქცია რომელიც ანხორციელებს გამრავლებას x -ზე ამ მეთოდით. მაშინ n -ჯერ ამ ფუნქციის გამოყენებით შესაძლებელია მივიღოთ x^n -ზე გამრავლების შედეგი და შემდეგ x -ის სხვადასხვა ხარისხების შეკრებით მივიღოთ ველის ნებისმიერი ელემენტი. მაგალითად, დავუშვათ, გვინდა გავმოვთვალოთ $(57)_{hex} \cdot (13)_{hex}$. ვიქცევით შემდეგნაირად, რადგანაც

$$(13)_{hex} = (01)_{hex} \oplus (02)_{hex} \oplus (10)_{hex}$$

ჯერ $xtime(\ \ )$ ფუნქციის საშუალებით გამოვთვალოთ შემდეგი სიდიდეები:

$$(57)_{hex} \cdot (02)_{hex} = xtime(57) = (AE)_{hex}$$
$$(57)_{hex} \cdot (04)_{hex} = xtime(AE) = (47)_{hex}$$
$$(57)_{hex} \cdot (08)_{hex} = xtime(47) = (8E)_{hex}$$
$$(57)_{hex} \cdot (10)_{hex} = xtime(8E) = (07)_{hex}$$

მაშინ

$$(57)_{hex} \cdot (13)_{hex} = (57)_{hex} \cdot ((01)_{hex} \oplus (02)_{hex} \oplus (10)_{hex}) = (57)_{hex} \oplus (AE)_{hex} \oplus (07)_{hex} = (FE)_{hex}$$

Rijndael-ში პოლინომი შეიძლება განიმარტოს აგრეთვე $F_{2^8}[X]$ რგოლიდან. ასეთ შემთხვევაში ოცდათორმეტ ბიტიან სიტყვას შეესაბამება მესამე ხარისხის პოლინომი, რომლის კოეფიციენტებსაც

წარმოადგენენ რვაბიტიანი რიცხვები, ამასთან კოეფიციენტების მინიჭება ხდება შებრუნებით. ანუ, თუ ოცდათორმეტბიტიანი სიტყვა დაყოფისას გვაძლევს ოთხ რიცხვს

$$a_0 \updownarrow\updownarrow a_1 \updownarrow\updownarrow a_2 \updownarrow\updownarrow a_3$$

მაშინ პოლინომს აქვს სახე:

$$a_3x^3 + a_2x^2 + a_1x + a_0$$

ალგორითმში შემოტანილი არითმეტიკა ემთხვევა არითმეტიკას $F_{2^8}[X]$ რგოლში მოდულით

$$M(x) = x^4 + 1 = (x+1)^4.$$

საინტერესოა, რომ $(x+1)^4$ არ არის დაუყვანადი პოლინომი და ამიტომ შესაძლებელია გვქონდეს ნოლის გამყოფები, ანუ პოლინომს არ გააჩნდეს შებრუნებული პოლინომი, მაგრამ ალგორითმში ისეა შერჩეული ფიქსირებული პოლინომი, რომ მას გააჩნია შებრუნებული პოლინომი ამ მოდულით. ასეთი მოდულის გამოყენება განაპირობა იმან, რომ

$$x^i \bmod (x+1)^4 = x^{i \bmod 4}$$

რაც ცხადია ამარტივებს მოდულის გამოთვლას.

ალგორითმში პოლინომის გამრავლება ამ ფიქსირებულ პოლინომზე და პოლინომის გამრავლება x ერთწევრზე შეიძლება მატრიცების საშუალებით, კერძოდ:

$$a(x) \otimes b(x) = d(x) \bmod (x+1)^4$$

სადაც $a(x)$ ფიქსირებული პოლინომია, მატრიცულად ტოლია

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_{43} \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_3 & a_0 & a_1 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

ხოლო $x \otimes b(x)$ შეესაბამება მატრიცული ნამრავლი

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 00 & 00 & 00 & 01 \\ 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

ეს განმარტებები გაგვიადვილებს ალგორითმის გაგებას.

3.15. Rijndael-ს არქიტექტურა. როგორც უკვე აღვნიშნეთ, 2001 წლიდან ამერიკის შეერთებული შტატების ნაციონალური სტანდარტების ინსტიტუტმა მრავალწლიანი განხილვის შემდეგ მიიღო დაშიფრვის ახალი ფედერალური სტანდარტი **Rijndael**-ი, რომელიც შექმნეს ბელგიელმა კრიპტოგრაფებმა იოან დამენმა და ვინსენტ რაიმანმა 1998 წელს (ალგორითმის სახელი წარმოადგენს ავტორთა გვარების კომბინაციას). ეს ალგორითმი განსხვავებით **DES**-საგან მიეკუთვნება იმ მცირერიცხოვან ბლოკურ ალგორითმებს, რომელიც არ იყენებენ ფეინსტელის სქემას. მას საფუძვლად უდევს ე.წ. "კვადრატის" არქიტექტურა. ეს სახელწოდება არქიტექტურამ მიიღო კრიპტოალგორითმ კვადრატისაგან, რომელიც 1996 წელს შექმნეს იმავე ავტორებმა ლარს კნუდსენთან ერთად. ალგორითმს შეუძლია იმუშაოს 128,192 და 256 ბიტიან ბლოკებთან (თუმცა სტანდარტს წარმოადგენს 128 ბიტიანი ბლოკი). გასაღების სიგრძე ასევე შეიძლება იყოს 128,192 და 256 ბიტის სიგრძის, ამასთან შესაძლებელია სხვადასხვა სიგრძის ბლოკებთან სხვადასხვა სიგრძის გასაღებების გამოყენება.

ვინაიდან ალგორითმში ფაქტობრივად სრულდება ორი მარტივი ოპერაცია – **XOR**-ით შეკრება და ბაიტების ინდექსირებული ამოღება მეხსიერებიდან, ალგორითმი შეიძლება ეფექტურად იყოს რეალიზებული ყველა ცნობილი პლატფორმისა და აპარატურისათვის. ალგორითმში რაუნდების რაოდენობა დამოკიდებულია ბლოკისა და გასაღების ზომებზე. ეს დამოკიდებულება მოცემულია სპეციალური ცხრილით (იხ. ცხრილი 3.6).

3.16. შებრუნებული ოპერაციების არსებობა. როგორც უკვე აღვნიშნეთ, განსხვავებით **DES**-ისაგან, **Rijndael**-ი არ არის აგებული ფეისტელის სქემის მიხედვით, მისი არქიტექტურა არის ე.წ. კვადრატი და მასში ყოველი გარდაქმნა ემორჩილება ზევით განხილულ არითმეტიკულ ოპერაციებს. ფეისტე-

ლის სქემა, როგორც ეს აღნიშნული იყო **DES**-ის განხილვის დროს, თვითონ არის შებრუნებადი სქემა და არა აქვს მნიშვნელობა როგორი სახის ფუნქცია გამოიყენება დასაშიფრად. რადგანაც კვადრატულ არქიტექტურას აღარ გააჩნია ასეთი თვისება, იმისათვის, რომ დეშიფრაციის პროცესი წარიმართოს დაშიფრვის სიმეტრიულად საჭიროა დაშიფრვის პროცესში გამოყენებულ ოპერაციებს გააჩნდეთ შებრუნებული ოპერაციები, პრაქტიკულად ეს იმას ნიშნავს, რომ დაშიფრვისა და დეშიფრაციის პროცედურები აღარ ემთხვევა ერთმანეთს.

ცხრილი 3.6. რაუნდების დამოკიდებულება გასაღების და ბლოკის ზომებზე.

გასაღების ზომა ბლოკის ზომა	128	192	256
128	10	12	14
192	12	12	14
256	14	14	14

3.17. მდგომარეობათა მატრიცა. **Rijndael**-ში ყველა გარდაქმნა სრულდება ბაიტურ მატრიცაზე, რომელსაც ეწოდება მდგომარეობა. ეს მატრიცა შეიცავს ოთხ სტრიქონს, ხოლო სვეტების რაოდენობა, რომელიც აღინიშნება Nb-თი, დამოკიდებულია ბლოკის სიგრძეზე და უდრის ბლოკის სიგრძე გაყოფილი 32-ზე, ამიტომ, **Rijndael**-ი შეიძლება მუშაობდეს 4×4, 4×6 და 4×8 ზომის მატრიცებთან. ჩვენ შემდგომში განვიხილავთ მხოლოდ 4×4 მატრიცას, რომელსაც აქვს შემდეგი სახე:

$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

სანამ დაიწყება მოქმედებები რაუნდში დასაშიფრი ბლოკი **XOR** ოპერაციით იკრიბება გასაღებთან და ისე შედის მდგომარეობათა ბლოკში. ამასთან მიღებული ტექსტი (დავუშვათ 128 ბიტი) მდგომარეობათა მატრიცას ავსებს მარცხნიდან მარჯვნივ და ზევიდან ქვევით. ანუ მიმდევრობით:

$a_{0,0}, a_{1,0}, a_{2,0}, \ldots, a_{3,3}$.

ყოველი რაუნდი, გარდა დამამთავრებელი რაუნდისა, შედგება ოთხი გარდაქმნისაგან:

```
Round (State,Roundkey)
 {
  ByteSub (State);
  ShiftRow (State);
  MixColumn (State);
  AddRoundKey (State, RoundKey);
                                  }
```

დამამთავრებელი რაუნდი განსხვავდება სხვა რაუნდებისაგან იმით, რომ მასში გამოტოვებულია გარდაქმნის ოპერაცია **Mixcolumn (state).**

3.18. ოპერაცია ByteSub. ეს გარდაქმნა არის არაწრფივი ბაიტური გარდაქმნა, რომელიც ყოველი ბაიტისათვის სრულდება დამოუკიდებლად. ჩანაცვლების ცხრილი (ანუ **S**–ბლოკი) შებრუნებადია და კონსტუირებულია ორი გარდაქმნის კომპოზიციის სახით:

1) აიღება ბაიტის მულიტიპლიკაციური შებრუნებული $GF(2^8)$ ველში. $'00'$ ელემენტი აისახება თავის თავში;

2) ამის შემდეგ გამოიყენება აფინური გარდაქმნა $GF(2)$ ველში, განსაზღვრული შემდეგი სახით:

$$\begin{bmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \times \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

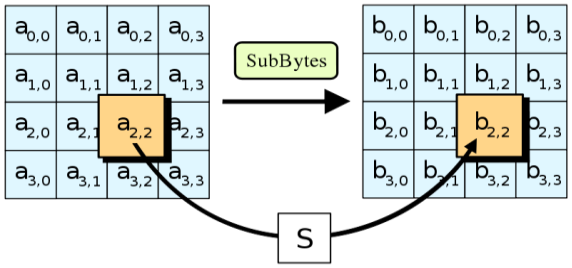
სხვანაირად ეს გარდაქმნა შეიძლება აღვწეროთ განტოლებით:

$$y_i = x_i \oplus x_{(i+4) \bmod 8} \oplus x_{(i+5) \bmod 8} \oplus x_{(i+6) \bmod 8} \oplus x_{(i+7) \bmod 8} \oplus c_i$$

სადაც x_i და y_i შესაბამისად i-ური ბიტის საწყისი და გარდაქმნილი მნიშვნელობებია $(i = 0,1,...,7)$, $c_0 = c_1 = c_5 = c_6 = 1$ და $c_2 = c_3 = c_4 = c_7 = 0$.

ამ გარდაქმნის შედეგად მიიღება მდგომარეობის ახალი მნიშვნელობა, რომელიც ნაჩვენებია სურ. 3.7.

ეს გარდაქმნა შესაძლებელია განხორციელებული იქნას სპეციალური ცხრილის საშუალებითაც, თუ ჩვენ წინასწარ გამოვთვლით ყველა შესაძლო ბაიტის გარდაქმნებს და შევინახავთ კომპიუტერში 16×16 მატრიცის სახით (იხ. ცხრილი 3.7). შევნიშნოთ, რომ ამ ცხრილში მოყვანილი ყველა რიცხვები ჩაწერილია თექვსმეტობითი ფუძით.



სურ. 3.7. ოპერაცია ByteSub-ის შედეგად მიღებული მდგომარეობა.

ცხრილი 3.7

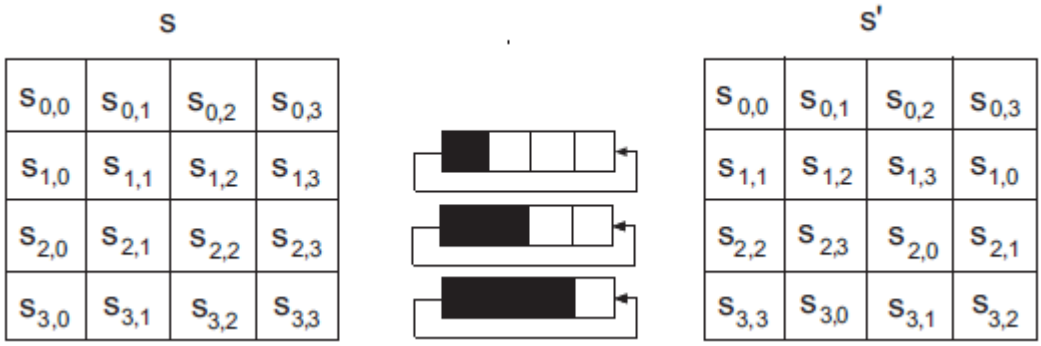
x	*y*															
	0	1	2	3	4	5	6	7	8	9	*A*	*B*	*C*	*D*	*E*	*F*
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	A0	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5°	A0	52	3B	D6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

F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

3.19. ShiftRow (State). შემდეგი გარდაქმნა, **SriftRow (State)** ახდენს მდგომარეობის მატრიცის სტრიქონების ციკლურ გადაადგილებას (წაძვრას მარცხნივ). ამასთან ეს წაძვრები სხვადასხვა სტრიქონებისათვის სრულდება განსხვავებულად და დამოკიდებულია მდგომარეობის მატრიცაში სვეტების რაოდენობაზე **(Nb)**. ეს დამოკიდებულება მოცემულია შემდეგ ცხრილში 3.8. სადაც **C0**, **C1, C2, C3** აღნიშნავენ შესაბამისად პირველ, მეორე, მესამე და მეოთხე სტრიქონებს. როგორც ამ ცხრილიდან ჩანს, პირველი სტრიქონის წაძვრა არ ხდება.
ცხრილი 3.8. სტრიქონების წაძვრის დამოკიდებულება სვეტების რაოდენობაზე

Nb	**CO**	**C1**	**C2**	**C3**
4	0	1	2	3
6	0	1	2	3
10	0	1	3	4

ამ ოპერაციის გრაფიკული სქემა მოცემულია სურ . 3.8



სურ . 3.8 ოპერაცია ShiftEow (State)-ის შედეგად მიღებული მდგომარეობა.

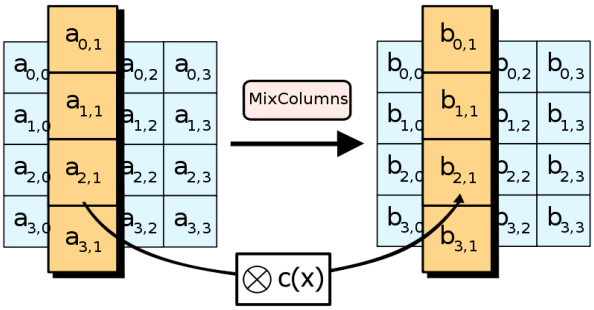
3.20. MixColumn (State). ამ გარდაქმნაში მდგომარეობის ყოველი სვეტი განიხილება როგორც პოლინიმი $GF(2^8)$ ველში და მრავლდება ფიქსირებულ

$$c(x) = ^{/} 03^{/} x^3 +^{/} 01^{/} x^2 +^{/} 01^{/} x +^{/} 02^{/}$$

პოლინომზე მოდულით $x^4 + 1.$ $c(x)$ პოლინომი თანამარტივია $x^4 + 1$ და ამიტომ გააჩნია შებრუნებული. ეს გამრავლება პრაქტიკულად სრულდება როგორც გამრავლება მატრიცაზე. თუ $b(x) = c(x) \otimes a(x)$ მაშინ მატრიცულ წარმოდგენაში გვექნება:

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \otimes \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

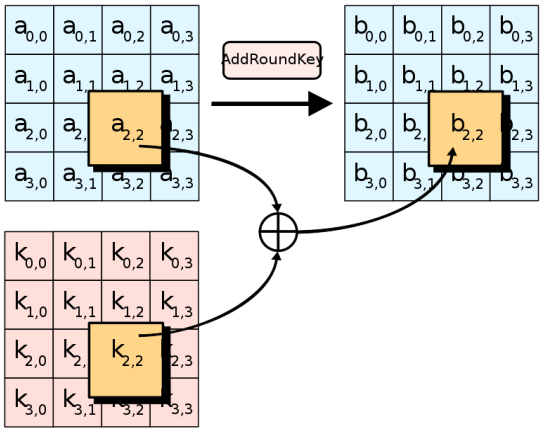
3.21. AggRoundKey. შემდეგი გარდაქმნაა **AddRounKey (State).** ამ დროს ხდება რაუნდის გასაღების **XOR**-ით შეკრება მიღებულ მდგომარეობასთან. გასაღებიც წარმოდგინება ისეთივე მატრიცის სახით, როგორც მდგომარეობა.



სურ. 3.9 ოპერაცია MixColumn (State)-ის შედეგად მიღებული მდგომარეობა.

3.21. AggRoundKey. შემდეგი გარდაქმნაა **AddRounKey (State).** ამ დროს ხდება რაუნდის გასაღების XOR-ით შეკრება მიღებულ მდგომარეობასთან. გასაღებიც წარმოდგინება ისეთივე მატრიცის სახით, როგორც მდგომარეობა.

3.22. რაუნდის გასაღების შექმნა. 128 ბიტიანი ტექსტის შემთხვევაში, როგორც ეს სურ. 2.4.-დანაც ჩანს, ერთ რაუნდში საჭიროა 128 ბიტიანი გასაღები, ანუ როგორც ამბობენ ოთხი 32 ბიტიანი სიტყვა. სულ საჭიროა ათი ასეთი ქვეგასაღები. ამისათვის, საწყისი გასაღები იყოფა ოთხ 32 ბიტიან სიტყვად k_0, k_1, k_2, k_3. აღვნიშნოთ i-ური რაუნდის გასაღები $[W_{4i}, W_{4i+1}, W_{4i+2}, W_{4i+3}]$. რაუნდის გასაღები გამომუშავდება შემდეგი ალგორითმით.



სურ .3.10 მდგომარეობის და გასაღების შეკრება XOR-ის საშუალებით.

$$W[0] = k[0]; W[1] = k[1]; W[2] = k[2]; W[3] = k[3]$$
$$for(i = 1; i \Leftarrow 10; i++)$$
$$(T = RotBytes(W[4 \times i - 1]);$$
$$T = SubBytes(T);$$
$$T = T \wedge RC[i]$$
$$W[4 \times i] = W[4 \wedge i - 4] \wedge T$$
$$W[4 \times i + 1] = W[4 \times i - 3] \wedge W[4 \times i]$$
$$W[4 \times i + 2] = W[4 \times i - 2] \wedge W[4 \times i + 1]$$
$$W[4 \times i + 3] = W[4 \times i - 1] \wedge W[4 \times i + 2]\}$$

სადაც **RotRyte (W)** აღნიშნავს მარცხნივ ერთ ბაიტზე ციკლურად წაძვრის პროცედურას, ისე, რომ შემავალი სიტყვისათვის **(a,b,c,d)** მიიღება გამომავალი სიტყვა **(b,c,d,a),** ხოლო **SubByte(W)** კი ზემოთაღწერილი პროცედურაა, რომელიც გამოიყენება სიტყვის ყოველი ბაიტისათვის. **RC[i]** არის რაუნდის კონსტანტა, რომელიც გამოითვლება შემდეგი ფორმულით:

$$RC(i) = x^i \bmod(x^8 = x^4 + x^3 + x + 1).$$

ცხადია, რომ გასაღების გაფართოება უნდა ჩატარდეს დაშიფრვის (ან დეშიფრაციის) პროცესის დაწყებამდე.

საკონტროლო კითხვები:

1. როგორ კრიპტოალგორითმებს ეწოდებათ ბლოკური შიფრები?
2. რას წარმოადგენს ჰ. ფეისტელის სქემა და რა განსაკუთრებული თვისება აქვს მას?
3. რომელი ალგორითმი დაედო საფუძვლად **DES**-ის შექმნას?
4. თქვენი აზრით რა ცვლილებებს იწვევს ბლოკის ზომის ზრდა სიმეტრიულ ალგორითმში?
5. აღწერეთ გარდაქმნები, რომლებიც ტარდება ტექსტზე **DES**-ის ერთ რაუნდში.
6. როგორ გამომუშავდება რაუნდული გასაღები **DES**-თვის?
7. რა პირობებს უნდა უნდა აკმაყოფილებდნენ **S** ბლოკები?
8. რა ეწოდება **AES RIJNDAEL**-ის არქიტექტურას?
9. ჩამოთვალეთ და მათემატიკურად აღწერეთ ოპერაციები, რომლებიც გამოიყენება **AES RIJNDAEL**-ის ერთ რაუნდში. როგორ სრულდება ეს ოპერაციები პროგრამულად?
10. არის თუ არა **AES RIJNDAEL**-ის არქიტექტურა სიმეტრიული? რა არის საჭირო იმისათვის, რომ შესაძლებელი იყოს დეშიფრაცია?

სიმეტრიული ბლოკური ალგორითმები IDEA და BLOW-FISH

4.1. ალგორითმის სახელწოდება. **IDEA** მიეკუთვნება იმ ცნობილი ალგორითმების მცირერიცხოვან კლასს, რომლებიც მოქმედებენ უკვე ორი ათეული წელია და ჯერ არ არსებობს ცნობები მათი გატეხვის შესახებ. ალგორითმის პირველი ვარიანტი შეიქმნა 1990 წელს კ. ლაისა და ჯ.მესის მიერ და ეწოდებოდა **PES (Proposed Emcryption Standard)** – დაშიფრვის შემოთავაზებული სტანდარტი. ერთი წლის შემდეგ (როცა გამოქვეყნდა დიფერენციალური კრიპტოანალიზი), ავტორებმა გააძლიერეს ეს ალგორითმი და უწოდეს მას **IPES (Improved Proposed Eбcryption Standard)** - დაშიფრვის გაუმჯობესებული შემოთავაზებული სტანდარტი. კიდევ ერთი წლის შემდეგ კვლავ შეუცვალეს სახელი და უწოდეს **IDEA (International Data Encryptionნ stabdard)** - მონაცემთა დაშიფრვის საერთაშორისო სტანდარტი. ალგორითმი დაპატენტებულია. ის გამოიყენება ყველასათვის ცნობილ კომპიუტერულ სისტემაში **PGP (Pretty Good Privacy),** რომელიც გამოიყენება ელექტრონული სახით წარმოდგენილი ინფორმაციის დაშიფრვისა და ელექტრონული ხელმოწერი-სათვის, არაა დაპატენტებული და შეუძლია გამოიყენოს ნებისმიერ მომხმარებელს.

4.2. ალგორითმის აღწერა. **IDEA** წარმოადგენს ბლოკურ შიფრს, რომელიც ისევე, როგორც **DES** მუშაობს 64 ბიტიან ბლოკთან, მაგრამ იყენებს 128 ბიტიან გასაღებს. როგორც ცნობილია, **DES** იყო შექმნილი ძირითადად სპეციალური სქემებით შესასრულებლად, მისგან განსხვავებით **IDEA** შეიქმ-ნა როგორც პროგრამული უზრუნველყოფა თექვსმეტბიტიანი მიკროპროცესორიანი კომპიუტერე-ბისათვის, რომლებიც წარმოადგენდნენ საუკეთესო არქიტერქტურის მქონე კომპიუტერებს ოთხმოცდაათიანი წლების დასაწყისში. ამ ალგორითმის არქიტექტურა წარმოადგენს ჰ. ფეისტელის სქემის მოდიფიკაციას და ამიტომაც ცხადია, რაღაცით წააგავს **DES**-ის არქიტექტურას, თუმცა გამოყენებული ფუნქციებით და კონსტრუქციით არსებითად განსხვავდება მისგან.

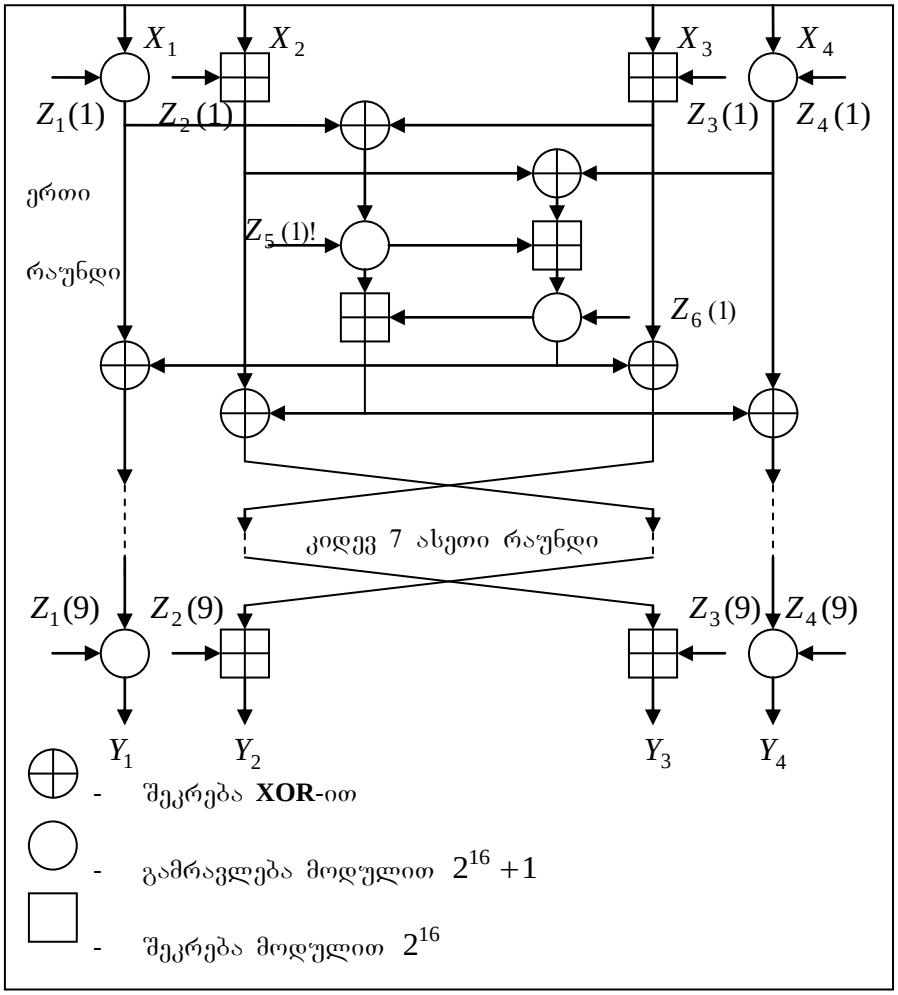
ძირითადი იდეა, რომელიც დაედო საფუძვლად **IDEA**-ს კონსტრუქციას, მდგომარეობს შეუთავსებად ოპერაციათა გარეთიანებაში სხვადასხვა ალგებრული ჯგუფებიდან. კერძოდ, ალგო-რითმში გამოიყენება სამი ალგებრული ჯგუფი, რომელთაგან თითოეული მათგანი ადვილად განხორციელებადია როგორც აპარატურულად, ასევე პროგრამულად და არცერთი ორი არაა თავსე-ბადი. ეს ჯგუფებია **XOR** ოპერაცია, შეკრება 2^{16} მოდულით და გამრავლება $2^{16}+1$ მოდულით.

(ეს ოპერაცია შეიძლება ჩაითვალოს **IDEA**-ს **S** ბლოკად). ოპერაციები შეუთავსებადია იმ თვალსაზრისით, რომ არცერთი წყვილისათვის არ კმაყოფილდება ასოციაციურობისა და დისტრიბუტულობის კანონი, ანუ

$$a*(b+c) \neq a*b+a*c$$

$$a+(b \oplus c) \neq (a+b) \oplus c$$

ამ ოპერაციების გამოყენება ართულებს **IDEA**-ს კრიპტოანალიზს **DES**-თან შედარებით და საშუალებას გვაძლევს არ გამოვიყენოთ **S** ბლოკები.



სურ. 4.1. ალგორითმ IDEA-ს სქემა

როგორც სურ. 4.1. ჩანს, დასაშიფრი სამოცდაოთხი ბიტი იყოფა ოთხ თექვსმეტბიტიან ქვებლოკებად X_1, X_2, X_3, X_4, რომლებზედაც თითოეულ რაუნდში ტარდება შემდეგი ოპერაციები:

1. X_1 ქვებლოკი გადამრავლდება Z_1 ქვეგასაღებზე $2^{16}+1$ მოდულით;
2. X_2 ქვებლოკი შეიკრიბება Z_2 ქვეგასაღებთან 2^{16} მოდულით;
3. X_3 ქვებლოკი შეიკრიბება Z_3 ქვეგასაღებთან 2^{16} მოდულით;
4. X_4 ქვებლოკი გადამრავლდება Z_4 ქვეგასაღებზე $2^{16}+1$ მოდულით;
5. სრულდება **XOR** ოპერაცია პირველი და მესამე ოპერაციების შედეგებზე;
6. სრულდება **XOR** ოპერაცია მეორე და მეოთხე ოპერაციების შედეგებზე;
7. მეხუთე ეტაპის შედეგი გადამრავლდება Z_5 ქვეგასაღებზე $2^{16}+1$ მოდულით;

8. მეექვსე და მეშვიდე ეტაპების შედეგები შეიკრიბება 2^{16} მოდულით;

9. მერვე ეტაპის შედეგი გადამრავლდება Z_6 ქვეგასაღებზე $2^{16}+1$ მოდულით;

10. მეშვიდე და მეცხრე ეტაპების შედეგები შეიკრიბება 2^{16} მოდულით;
11. სრულდება **XOR** ოპერაცია პირველი და მეცხრე ეტაპების შედეგებზე;
12. სრულდება **XOR** ოპერაცია მესამე და მეცხრე ეტაპების შედეგებზე;
13. სრულდება **XOR** ოპერაცია მეორე და მეათე ეტაპის შედეგებზე;
14. სრულდება **XOR** ოპერაცია მეოთხე და მეათე ეტაპების შედეგებზე;

11–14 ეტაპების შედეგად მიიღება ოთხი ქვებლოკი. შიდა ქვებლოკების ადგილების გაცვლით (მხოლოდ პირველ შვიდ რაუნდში) მიიღება შემდეგი რაუნდის შესასვლელი. რვა რაუნდის შემდეგ სრულდება კიდევ ერთი დამამთავრებელი გარდაქმნა:

1. $2^{16}+1$ მოდულით გადამრავლდება X_1 ქვებლოკი და Z_1 ქვეგასაღები;

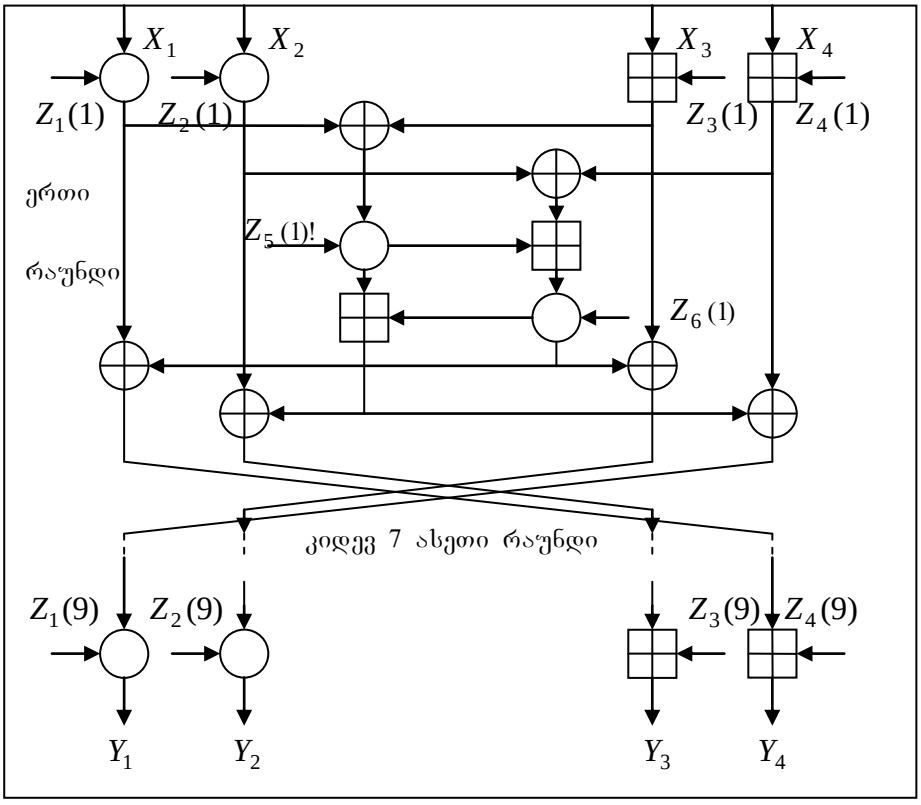
2. 2^{16} მოდულით შეიკრიბება X_2 ქვებლოკი და Z_2 ქვეგასაღები;

3. 2^{16} მოდულით შეიკრიბება X_3 ქვებლოკი და Z_3 ქვეგასაღები;

4. $2^{16}+1$ მოდულით გადამრავლდება X_4 ქვებლოკი და Z_4 ქვეგასაღები;

და ბოლოს ამ მიღებული ოთხი ქვებლოკის კონკატაციით მიიღება დაშიფრული ტექსტი.

4.3. გასაღებების გამომუშავება. ალგორითმში გამოიყენება სულ 52 ქვეგასაღები. ექვსი თითოეულ რაუნდში და კიდევ ოთხი დამამთავრებელ ეტაპზე. საწყის ეტაპზე 128 ბიტიანი გასაღები დაიყოფა რვა თექვსმეტბიტიან ქვეგასაღებად. ექვსი გამოიყენება პირველ ეტაპზე, ორი მეორე ეტაპზე. ამის შემდეგ გასაღები წაიძვრება მარცხნივ 25 ბიტზე და კვლავ დაიყოფა რვა ქვებლოკად. ასე გრძელდება ალგორითმის მიერ მუშაობის დამთავრებამდე.



სურ.4.2. ალგორითმ **PES**-ის სქემა.

4.4. დეშიფრაცია. დეშიფრაცია ხდება დაშიფრვის ანალოგიურად, საჭიროა მხოლოდ გასაღებების ინვენტირება და მცირეოდენი შეცვლა. ქვეგასაღებები, რომლებიც გამოიყენება დეშიფრაციის დროს წარმოადგენენ დაშიფრვის გასაღებების შებრუნებულ მნიშვნელობებს შეკრების ან გამრავლების

ოპერაციების მიმართ. **IDEA**-ში იგულისხმება, რომ ქვებლოკი, რომელიც შედგება მხოლოდ ნოლებისაგან ტოლია $2^{16} = -1$ გამრავლებისათვის $2^{16} + 1$, ამიტომ ნოლის შებრუნებული მნიშვნელობა გამრავლების მიმართ ასევე ნოლია. ამ გარდაქმნებს სჭირდება გარკვეული დრო, მაგრამ შესაძლებელია მათი წინასწარი ჩატარება ერთხელ თითოეული გასაღებისათვის, ამიტომ დეშიფრაციის დროს გვექნება უკვე გამზადებული ქვეგასაღებები. დაშიფრვისა და დეშიფრაციის ოპერაციები **IDEA**-ში სრულდება დაახლოებით ორჯერ უფრო ჩქარა, ვიდრე **DES**-ში.

4.5. IDEA-ს კრიპტოანალიზის შედეგები. IDEA-ს გასაღების სიგრძეა ბიტი, ამიტომ ძალისმიერი შეტევა ამ ალგორითმზე ჯერჯერობით შეუძლებელია.

როგორც უკვე აღვნიშნეთ, ავტორებმა ალგორითმის პირველი ვარიანტი **PES** დიფერენციალური კრიპტოანალიზის გამოქვეყნების შემდეგ გადააკეთეს ისე, რომ ის გამხდარიყო მედეგი ამ შეტევის მიმართ. მათ წამოაყენეს მარკოვის შიფრის იდეა და აჩვენეს, რომ მედეგობა დიფერენციალური კრიპტოანალიზის მიმართ შეიძლება შეფასდეს რაოდენობრივად. ამ კონცეპციიდან გამომდინარე მოახდინეს ალგორითმ **PES**-ის მოდიფიკაცია. სურ. 4.2. მოყვანილია ალგორითმ **PES**-ის სქემა. თუ შევადარებთ ამ სქემას **IDEA**-ს სქემას აღმოვაჩენთ, რომ ეს მოდიფიკაცია უმნიშვნელოა. **PES**-ში სიმეტრიულად განლაგებული გამრავლებისა და შეკრების ოპერაციები **IDEA**-ში გადანაცვლებულია და ასევე შეცვლილია ქვებლოკების გადანაცვლების სქემა.

ერთი შეხედვით ამ უმნიშვნელო გადანაცვლებებმა ალგორითმი გახადა მედეგი დიფერენციალური კრიპტოანალიზის მიმართ. ალგორითმის კრიპტომედეგობის ზრდას ხელს უწყობს აგრეთვე ორი თვისება: დაშიფრვა დამოკიდებულია გასაღებზე რთული და ჩახლართული სახით და გაბნევა (ანუ დიფუზია), გამოსასვლელი ტექსტის ყოველი ბიტი დამოკიდებულია შესასვლელი ტექსტის ყველა ბიტზე. ეს მიიღწევა **IDEA**-ს სქემაზე პუნქტირით შემოსაზღვრული ოპერაციების საშუალებით. ყოველივე აქედან გამომდინარე, არაა გასაკვირი, რომ ჯერჯერობით არ არსებობს წარმატებული შეტევა (ყოველ შემთხვევაში არ გამოქვეყნებულა) ალგორითმზე. ცნობილია ვ. მეიერის შეტევა "შუაში შეხვედრის მეთოდით", როდესაც შესაძლებელი გახდა 4,5 რაუნდიანი **IDEA**-ს გატეხვა (ალგორითმი არის 8,5ბიტიანი). ამისათვის საჭირო გახდა ყველა 2^{64} ბლოკის ცოდნა და ანალიზის სირთულე ტოლი იყო 2^{112} ოპერაციის, რაც თავისთავად მიუთითებს ასეთი შეტევის არაპრაქტიკულობაზე.

ალგორითმი **IDEA** მონაწილეობდა აშშ სტანდარტისათვის გამოცხადებულ კონკურსში და გავიდა ფინალურ ტურში ოთხ სხვა კანდიდატთან ერთად.

4.6. ალგორითმი BLOWFISH. ამ ალგორითმის ავტორია მთელ მსოფლიოში ერთერთი ყველაზე ცნობილი კრიპტოლოგი ბრიუს შნაიერი. როგორც აღნიშნავს ავტორი, მან ალგორითმი შექმნა 1994 წელს, როდესაც **DES**-ის გამოყენება უკვე არასაიმედო გახდა და ყველა სხვა ალგორითმი კი დაპატენტებული იყო. ბ. შნაიერის მიზანს წარმოადგენდა რომ ნებისმიერ ადამიანს ჰქონოდა საშუალება დაეშიფრა მისთვის საჭირო ინფორმაცია საიმედოდ, ამიტომ მას არ აუღია პატენტი, პირიქით, მან გამოაქვეყნა ალგორითმის პროგრამული უზრუნველყოფაც.

ალგორითმის შექმნის დროს ავტორისთვის მთავარი კრიტერიუმები იყო ა) სისწრაფე, ბ) კომპაქტურობა, გ) სიმარტივე და დ) კრიპტომედეგობის ვარირების საშუალება. ოთხივე ეს პარამეტრი ალგორითმში პრაქტიკულად მიღწეულია. ალგორითმის პროგრამული რეალიზაცია ოცდათორმეტ ბიტიან მიკროპროცესორებზე მონაცემებს შიფრავს ბაიტზე ოცდაექვსი ტაქტის სიჩქარით, პროგრამული რეალიზაცია იკავებს ხუთ კილობაიტ მეხსიერებას. ალგორითმში გამოყენებულია მხოლოდ ორი ოპერაცია, **XOR**-ით შეკრება და ოცდათორმეტ ბიტიანი ოპერანდის ამოღება ცხრილიდან. **Blowfish**-ის გასაღების სიგრძე იცვლება 32 ბიტიდან 448 ბიტამდე, რაც საშუალებას იძლევა შევარჩიოთ ალგორითმის სასურველი კრიპტომედეგობა.

4.7. $F(x)$ ფუნქციის გამოთვლა. ისევე როგორც სხვა მრავალ ალგორითმში, **Blowfish**-შიც გამოყენებულია ჰ. ფეისტელის სქემა. თექვსმეტრაუნდიანი ალგორითმი მუშაობს სამოცდაოთხ ბიტიან ბლოკთან ცვლადი სიგრძის გასაღებით, რომელიც შეიძლება იცვლებოდეს ბიტიდან ბიტამდე.

თითოეულ რაუნდში სრულდება გასაღებზე დამოკიდებული გადანაცვლება და გასაღებზე დამოკიდებული ჩანაცვლება. ამიტომ სანამ ალგორითმი დაიწყებს უშუალოდ მონაცემთა დაშიფრვას, ჯერ საჭიროა საწყისი საიდუმლო გასაღების საშუალებით თექვსმეტი რაუნდისათვის საჭირო $P_1 - P_{18}$ ოცდათორმეტბიტიანი ქვეგასაღებების და ჩანაცვლების ოთხი **S** ბლოკის მომზადება, რომელთა საშუალებითაც უნდა მოხდეს რაუნდული დაშიფრვა. ჩანაცვლების ცხრილების

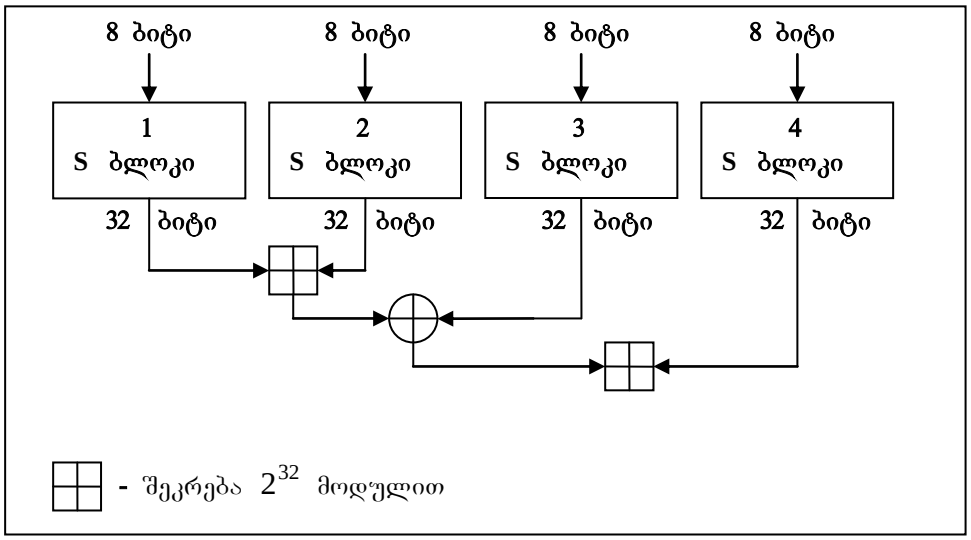
მიღების ალგორითმი მოყვანილია სურ. 4.3 შესასვლელი ოცდათორმეტი ბიტი გაიყოფა ოთხ რვაბიტიან ქვებლოკებად $[X_1, X_2, X_3, X_4]$ და თითოეული შედის ერთ S ბლოკში. ეს რვა ბიტი წარმოადგენს S_j ბლოკის ინდექსს. ბლოკის გამოსასვლელი კი არის ოცდათორმეტბიტიანი სტრიქონი, რომელიც მიიღება შემდეგნაირად. S_{1,X_1} და S_{2,X_2} იკრიბება მოდულით 2^{32}. მიღებული შედეგი იკრიბება **XOR**-ით S_{3,X_3}-თან და მიღებული შედეგი კვლავ იკრიბება მოდულით 2^{32} S_{4,X_4}-თან . ესაა $F(x)$-ის მნიშვნელობა.

$$F(x) = (((S_{1,X_1} + S_{2,X_2}) \bmod 2^{32} \oplus S_{3,X_3}) + S_{4,X_4}) \bmod 2^{32}$$

ამგვარად თითოეული ოცდათორმეტბიტიანი S ბლოკი შეიცავს 256 ელემენტს.

$$S_{1,0}, S_{1,1}, ..., S_{1,255}$$
$$S_{2,0}, S_{2,1}, ..., S_{2,255}$$
$$S_{3,0}, S_{3,1}, ..., S_{3,255}$$
$$S_{4,0}, S_{4,1}, ..., S_{4,255}$$

თითო რაუნდში გამოიყენება თითო ქვეგასაღები, ანუ თექვსმეტი რაუნდისთვის საჭიროა თექვსმეტი ქვეგასაღები. კიდევ ორი ქვეგასაღები საჭიროა დამათავრებელი გარდაქმნისათვის, ასე, რომ საერთო სიგრძე ქვეგასაღებებისა შეადგენს 576 ბიტს.



სურ.4.3 $F(x)$ ფუნქციის გამოთვლის სქემა

4.8. ქვეგასაღებებისა და S ბლოკების გამოთვლა. ქვეგასაღებებისა და ჩანაცვლების ოთხი **S** ბლოკის გამოსათვლელად, რომლებიც ასევე დამოკიდებულები არიან გასაღებზე უნდა ჩავატაროთ შემდეგი პროცედურები:

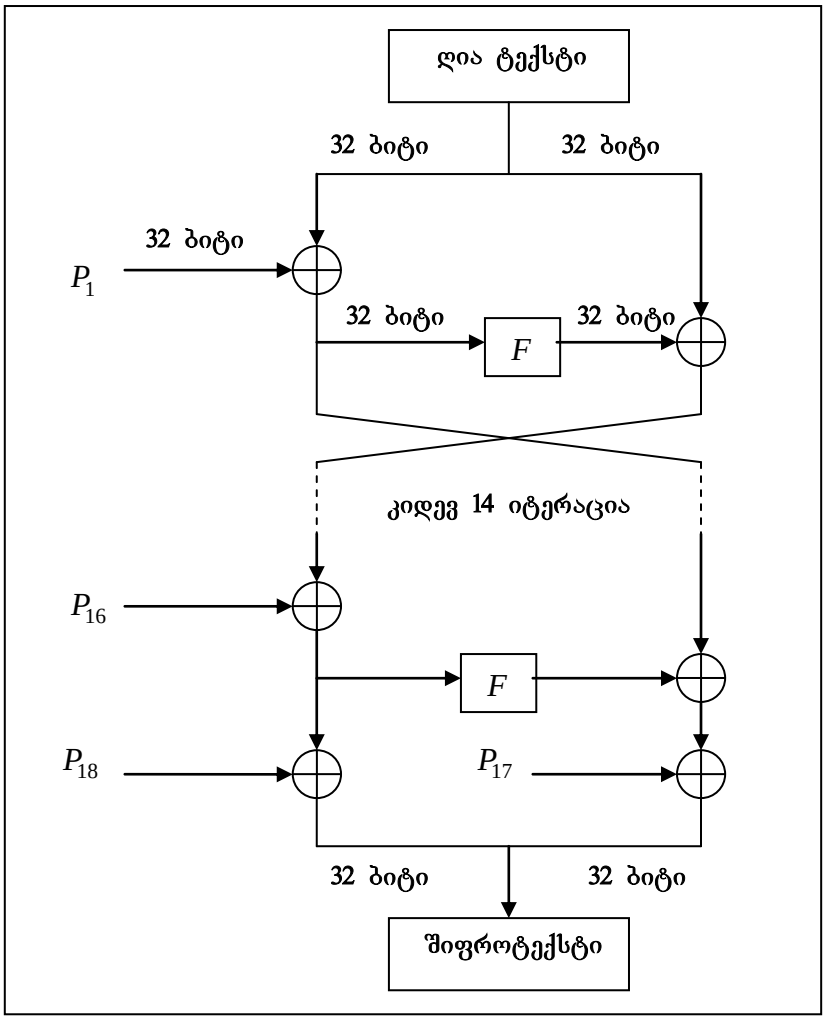
1. $P_1 - P_{18}$ ფიქსირებულ სტრიქონებში და ოთხივე **S** ბლოკში უნდა ჩაიწეროს ნებისმიერი საწყისი მიმდევრობა (ავტორი გვირჩევს ჩავწეროთ თექვსმეტობით სისტემაში წარმოდგენილი π რიცხვის მანტისა).
2. ამის შემდეგ საიდუმლო გასაღების პირველი ოცდათორმეტი ბიტი შეიკრიბება **XOR**-ით P_1 -ის ოცდათორმეტ ბიტთან, მეორე ოცდათორმეტი ბიტი -ის ოცდათორმეტ ბიტთან და ასე შემდეგ. თუ არ გვყოფნის საიდუმლო გასაღების სიგრძე, ხდება ბიტების ციკლური გამეორება.
3. ავიღოთ ნოლებისაგან შემდგარი ოცდათორმეტბიტიანი სტრიქონი და დავშიფროთ იმ ქვეგასაღებებით, რომლებიც მივიღეთ პირველ და მეორე ეტაპზე.

4. შევცვალოთ მესამე ეტაპზე მიღებული შიფროტექსტით P_1 და P_2 ქვეგასაღებები.
5. დავშიფროთ მესამე ეტაპზე მიღებული ტექსტი ახალი ქვეგასაღებების გამოყენებით.
6. შევცვალოთ მეხუთე ეტაპზე მიღებული ტექსტით P_3 და P_4 ქვეგასაღებები;
7. გავაგრძელოთ ეს პროცედურა სანამ არ შევცვლით ყველა ქვეგასაღებს და შემდეგ ოთხივე **S** ბლოკს ასეთი მიმდევრობით დაშიფრვის შედეგებით.

სულ დაგვჭირდება 521 იტერაცია, რათა მივიღოთ ყველა საჭირო ქვეგასაღები **S** ბლოკების მნიშვნელობათა ჩათვლით, რომელთა საერთო სიგრძეც იქნება 4168 ბაიტი. ცხადია, რომ ყველა ეს პროცედურა უნდა ჩატარდეს წინასწარ, სანამ დაიწყება გადასაცემი, ან შესანახი ღია ტექსტის დაშიფრვა და მიღებული ქვეგასაღებები შევინახოთ მეხსიერებაში.

4.9. ალგორითმის აღწერა. დაშიფრვა ხდება შემდეგი სქემით (იხ. სურ.4.4.). დასაშიფრი 64 ბიტი გაიყოფა ორ, მარცხენა და მარჯვენა ქვებლოკებად. მარცხენა ნახევარი (32 ბიტი) შეიკრიბება P_1 ქვეგასაღებთან **XOR** ოპერაციით და შედის $F(x)$-ს ბლოკში საიდანაც გამოსული 32 ბიტი ტექსტი შეიკრიბება მარჯვენა 32 ბიტთან. ამის შემდეგ მარჯვენა და მარცხენა ქვებლოკები გაცვლიან ადგილებს და იგივე პროცედურა გამეორდება კიდევ ხუთმეტჯერ. თექვსმეტი იტერაციის შედეგად მიღებული 32 ბიტიანი მარჯვენა და მარცხენა ნახევრები კიდევ ერთხელ შეიკრიბება P_{17} და P_{18} ქვეგასაღებებთან, რის შემდეგაც მოხდება მათი კონკატენცია და მიიღება 64 ბიტი დაშიფრული ტექსტი.

ალგორითმი **BLOWFISH** ასევე მონაწილეობდა კონკურსში და გავიდა ფინალურ ტურში.



სურ. 4.4. ალგორითმ **BLOWFISH**-ის სქემა.

4.10. ბლოკური დაშიფრვის რეჟიმები. ბლოკური დაშიფრვის რეჟიმი – ესაა ღია ტექსტის შიფროტექსტად გარდაქმნის ხერხი, ამასთან როგორც ღია ტექსტი, ასევე შიფროგრამა შეიძლება იყოს ნებისმიერი სიგრძის. უმრავლესობა დაშიფრვის რეჟიმებისა მოითხოვს, რომ ღია ტექსტის სიგრძე ჯერადი იყოს ბლოკის სიგრძისა. რადგანაც პრაქტიკაში იშვიათდა გვხვდება ასეთი ღია ტექსტები, საჭიროა მათი შევსება, ანუ გარკვეული რაოდენობის ბიტების დამატება, რათა ტექსტის ზომა გახდეს ბლოკის სიგრძის ჯერადი. არსებობს ღია ტექსტის შევსების სხვადასხვა მეთოდები, ერთადერთი პირობა, რომელსაც ეს მეთოდები უნდა აკმაყოფილებდნენ, მდგომარეობს იმაში, რომ ტექსტი დამატებულ ბიტებთან უნდა იყოს შებრუნებადი, ანუ შესაძლებელი იყოს დამატებული ბიტებით მიღებული შეტყობინებიდან ცალსახად აღვადგინოთ საწყისი ღია ტექსტი. ძალიან ხშირად იქცევიან სწორედ ამ პირობის გათვალისწინების საწინააღმდეგოდ და ღია ტექსტს P -ს უმატებენ მხოლოდ ნოლებს, მაგრამ P და $P \| 0$ (|| აღნიშნავს კონკატაციის ოპერაციას) ტექსტები ცხადია ვერ იქნება ერთნაირად შებრუნებადი, რაც იმას ნიშნავს, რომ იკარგება სწორედ ცალსახობა.

ერთი შეხედვით იმ შემთხვევაში, როდესაც ღია ტექსტის სიგრძე ჯერადია ბლოკის სიგრძის, დამატების პროცედურა აღარ უნდა ზრდიდეს ტექსტის ზომებს, მაგრამ ეს ასე არ არის. პრაქტიკულად დამატების ყველა წესი მინიმუმ ერთი ბაიტით ზრდის ტექსტის ზომას.

ვნახოთ როგორ შეიძლება მოვახდინოთ ღია ტექსტის შევსება საჭირო სიგრძემდე. პირველი ხერხით, თუ მოცემული გვაქვს ღია ტექსტი P , რომლის სიგრძეა $l = L(P)$ და b არის გამოყენებული ბლოკური შიფრის ბლოკის სიგრძე. დავუმატოთ ტექსტს ერთი ბაიტი, რომლის მნიშვნელობაა 128 და შემდეგ იმდენი ნულოვანი ბაიტი, რამდენიც საჭიროა იმისათვის, რომ ტექსტის სიგრძე გახდეს ბლოკის სიგრძის ჯერადი. შესაძლებელია მოვიქცეთ შემდეგნაირად, დავითვალოთ, რამდენი ბაიტის დამატებაა საჭირო და აღვნიშნოთ ეს რაოდენობა n -ით, ამასთან $1 \le n \le b$ და $n + L(M)$ ჯერადია b -სი. შევავსოთ ტექსტი n ბაიტით, რომელთა მნიშვნელობა ტოლია n -ს. ორივე ხერხი იძლევა კარგ შედეგებს.

როდესაც მოხდება დაშიფრული ტექსტის დეშიფრაცია, ცხადია დამატებული ბაიტები უნდა ჩამოეჭრას ღია ტექსტს, მაგრამ სანამ ჩამოვაჭრით ამ ბაიტებს, აუცილებელია დავრწმუნდეთ, რომ ტექსტის შევსება ჩატარებული იყო სწორედ. უნდა შემოწმდეს თითოეული დამატებული ბაიტის მნიშვნელობა და შეცდომის აღმოჩენის შემთხვევაში ეს შეცდომა უნდა დამუშავდეს ისე, როგორც აუთენტიფიკაციის შეცდომა.

4.11. დაშიფრვის რეჟიმები. ბლოკური დაშიფრვის დროს დაშიფრვის ერთი და იგივე ალგორითმი შეიძლება გამოყენებული იქნას სხვადასხვა წესით. ამ წესს ეწოდება დაშიფრვის რეჟიმი. კრიპტოგრაფიულ რეჟიმში გაერთიანებულია დაშიფრვის ალგორითმი, რომელიმე ტიპის უკუკავშირი და რამდენიმე მარტივი ოპერაცია. ოპერაციათა სიმარტივე გამოწვეულია იმით, რომ სისტემის კრიპტომედეგობა განისაზღვრება არა დაშიფრვის რეჟიმით, არამედ დაშიფრვის ალგორითმით. დაშიფრვის რეჟიმის მიზანია მაქსიმალურად შეუსაბამოს დაშიფრვის ერთი და იგივე ალგორითმი სხვადასხვა მოთხოვნებს, რომლებიც შეიძლება წარმოიშვას ამა თუ იმ კონკრეტულ შემთხვევაში. დაშიფრვის რეჟიმის არჩევის დროს ხელმძღვანელობენ სამი ძირითადი მოსაზრებით, რამდენად უსაფრთხოა, ეფექტურია და მდგრადია რეჟიმი. უსაფრთხოების თვალსაზრისით, რეჟიმს უნდა შეეძლოს დამალოს ღია ტექსტის სტრუქტურა. რეჟიმში შეუძლებელი უნდა იყოს შიფროტექსტში შეცდომების შეტანით ღია ტექსტის მანიპულირება და უკიდურეს შემთხვევაში, რეჟიმი არ უნდა ახდენდეს გამოყენებული კრიპტოალგორითმის კომპრომეტაციას. რეჟიმი აგრეთვე მდგრადი უნდა იყოს სისტემაში წარმოქმნილი ხარვეზების მიმართ. დეშიფრაციის პროცესში შესაძლებელი უნდა იყოს შიფროტექსტში დაშვებული შეცდომების გასწორება და თვით პროცესი უნდა იყოს გამძლე შიფროტექსტში ბიტების ჩამატების ან დაკარგვის მიმართ.

აშშ-ში პირველი ბლოკური დაშიფრვის სტანდარტის – **DES**-ის შემოღებიდან ძალიან მალე, მიღებული იქნა მეორე ფედერალური სტანდარტი, რომელშიც რეკომენდირებული იყო **DES**-ის ექსპლუატაციის ოთხი მეთოდი. ამ მეთოდების გამოიყენება შესაძლებელია ნებისმიერი ბლოკური ალგორითმისთვისაც. ძალიან მალე მცირეოდენი ცვლილებებით ეს რეჟიმები გადაიქცა საერთაშორისო სტანდარტად. ეს რეჟიმებია:

_ **ECB (Electromic codebook)** – ელექტრონული კოდური წიგნის რეჟიმი;

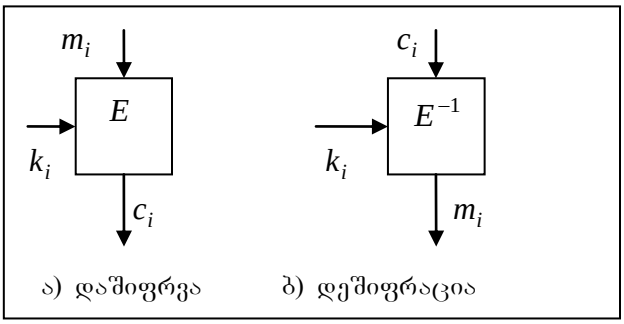
_ **CBC (Cipher Block Chaining)** – შიფროტექსტის ბლოკების გადაბმის რეჟიმი;

_ **OFB (Output-Feedmack)** _ გამოსასვლელით უკუკავშირის რეჟიმი;

_ **CFB (Cipher-Feedback)** – შიფროტექსტით უკუკავშირის რეჟიმი;

4.12. ელექტრონული კოდური წიგნის (ECB) რეჟიმი. ეს რეჟიმი ძალიან მარტივია შესასრულებლად, მაგრამ თითქმის არ არის დაცული შეტევებისაგან, რომლის დროსაც ხდება ტექსტის ამოღება, ან ტექსტის ჩამატება. მის დადებით მხარედ უნდა ჩაითვალოს, რომ შეცდომა, რომელიც შეიძლება დაშვებული იყოს შიფროტექსტის ერთ ბიტში, გავლენას ახდენს მხოლოდ ამ ბლოკის გაშიფრვაზე.

ამ რეჟიმში დასაშიფრი ტექსტი M, დაიყოფა n სიგრძის ბლოკებად $m_1, m_2, ...m_k$ და თითოეული მათგანი დაიშიფრება და სხვა ბლოკებისაგან დამოუკიდებლად $c_i = E(k, m_i)$ (იხ. სურ. 4.5). რადგანაც ერთი და იგივე გასაღების დროს ღია ტექსტის ერთი და იგივე ბლოკი გადავა კრიპტოგრამის ერთნაირ ბლოკში, თეორიულად შესაძლებელია შევქმნათ ე. წ. კოდური წიგნი, რომელშიც მითითებული იქნება თუ ღია ტექსტის რომელ ბლოკს კრიპტოგრამის რა ბლოკი შეესაბამება მოცემული გასაღებისათვის და დაშიფრვის პროცესი დავიყვანოთ ბლოკების ჩანაცვლების პროცესსზე, რაც მნიშვნელოვნად შეამცირებს დაშიფრვისა და გაშიფრვის დროს. მაგრამ ასეთ წიგნში ჩანაწერების რაოდენობა იქნება 2^n, სადაც n არის ბლოკის სიგრძე. რადგანაც კრიპტოგრაფიაში თითქმის ყოველთვის $n \geq 64$, ცხადია, რომ პრაქტიკულად ასეთი წიგნის შედგენა შეუძლებელია, მით უმეტეს, თუ გავითვალისწინებთ, რომ თითოეული გასაღებებისათვის მოგვიხდებოდა თითო ასეთი წიგნის შექმნა.



სურ. 4.5 **ECB** რეჟიმი.

ECB რეჟიმში შესაძლებელია ბლოკების დაშიფრვა და გაშიფრვა ნებისმიერი მიმდევრობით, შესაძლებელია ჯერ დავშიფროთ ღია ტექსტის შუა რამდენიმე ბლოკი და მხოლოდ შემდეგ საწყისი ბლოკები. ეს განსაკუთრებით მნიშვნელოვანია მაგალითად მონაცემთა ბაზების დაშიფრვა – დეშიფრაციის დროს, სადაც გვაქვს წვდომა ნებისმიერი მიმდევრობით. თუ მონაცემთა ბაზა დაშიფრულია **ECB** რეჟიმში, მაშინ შესაძლებელია ნებისმიერი ჩანაწერის დამატება, ან წაშლა ისე, რომ არ შევეხოთ დანარჩენ ჩანაწერებს (იმ პირობით, რომ თითოეული ჩანაწერი იკავებს ბლოკების მთელ რაოდენობას). შესაძლებელია დაშიფრვის პროცესის განპარალელებაც, რაც საშუალებას მოგვცემს დავაჩქაროთ დაშიფრვისა და გაშიფრვის პროცესები.

4.13. შეტევა ECB რეჟიმზე. ამ რეჟიმის სუსტ მხარეს წარმოადგენს ის, რომ ღია ტექსტის სტრუქტურა არ იმალება, და ამიტომ, ეს რეჟიმი უძლურია წინ აღუდგეს შეტევას ტექსტის ამოღებით ან ჩამატებით. ამ შეტევის გასარკვევად განვიხილოთ ასეთი მაგალითი, დავუშვათ, ღია ტექსტია "გადაუხადეთ ანის ასი ევრო. არ გადაუხადოთ ბექას ორასი ევრო". დავუშვათ, **ECB** რეჟიმით დაშიფრვის დროს თითოეული სიტყვა იშიფრება ერთი ბლოკის საშუალებით და კრიპტოგრამას აქვს შემდეგი სახე: "ლელას უყვარს წითელი ვარდები. მე არ მიყვარს ყვითელი ვარდები". თუ მოწინააღმდეგეს აქვს საშუალება ჩაერიოს გადაცემის პროცესში და იცის კრიპტოგრამის შინაარსი, მას შეუძლია ეს კრიპტოგრამა მარტივად გარდაქმნას ისე, რომ შეცდომაში შეიყვანოს ადრესატი. მაგალითად, საკმარისია მესამე და მერვე ბლოკებს გაუცვალოს ადგილები და მივიღებთ: "ლელას უყვარს ყვითელი ვარდები. მე არ მიყვარს წითელი ვარდები", რაც გაიშიფრება ასე: "გადაუხადეთ ანის ორასი ევრო. არ გადაუხადით ბექას ასი ევრო". მოწინააღმდეგეს შეუძლია საერთოდ ჩაამატოს ბლოკები, ან ამოაგდოს ზოგიერთი ბლოკი და სრულიად შეცვალოს გადაცემული ტექსტი.

მოწინააღმდეგეს შეუძლია მაშინაც ჩაერიოს გადაცემის პროცესში, როდესაც მისთვის უცნობია დაშიფრვის გასაღები ან ალგორითმი. განვიხილოთ ასეთი მაგალითი. დავუშვათ, ორი ბანკი

შეთანხმდა რაიმე პროცედურაზე, რომლის საშუალებითაც ხდება ფულის გადარიცხვის პროგრამული უზურნველყოფა და ეს პროცედურა ასეთია:

დროითი ჭდე 1 ბლოკი
1 ბანკი 1.5 ბლოკი
2 ბანკი 1,5 ბლოკი
კლიენტის სახელი 6 ბლოკი
კლიენტის ანგარიშიდ ნომერი 2 ბლოკი
თანხა 1 ბლოკი

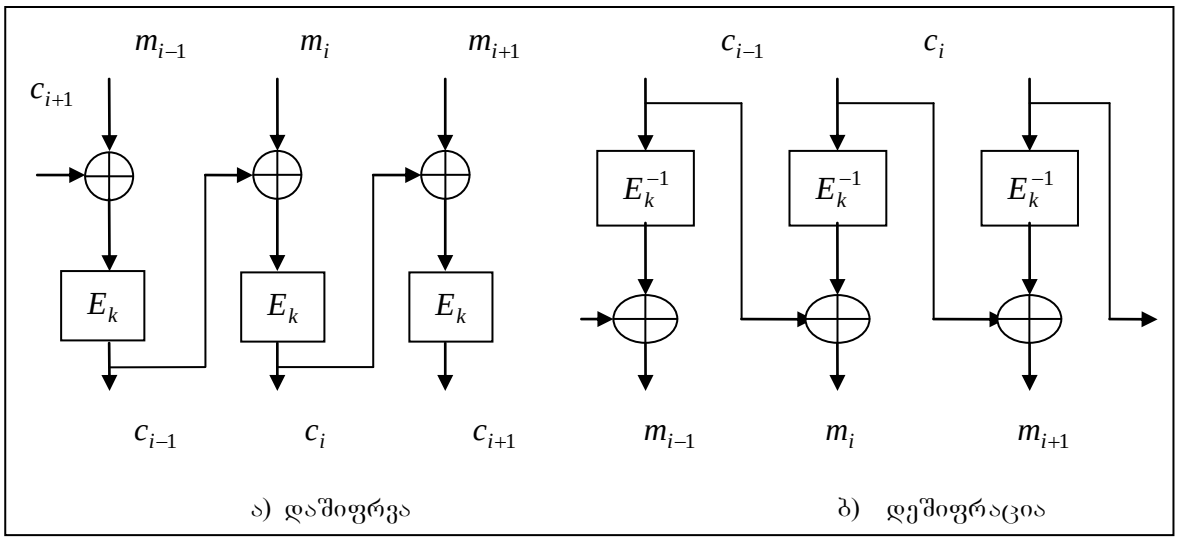
დაშიფრვა ხდება რაიმე სიმეტრიული ალგორითმით **ECB** რეჟიმში. მოწინააღმდეგე, რომელიც ცდილობს მოატყუოს ბანკი, კომპიუტერული ქსელიდან მიიტაცებს ამ ბანკებს შორის გაცვლილ ყველა დაშიფრულ შეტყობინებას. შემდეგ ის გახსნის ბანკებში ანგარიშებს და რამდენჯერმე გადარიცხავს ერთსა და იმავე თანხას ერთი ბანკიდან მეორეში. ერთნაირი ჩანაწერების საშუალებით აღმოაჩენს კრიპტოგრამებს, რომლებიც შეესაბამებიან მის გადარიცხვებს, შემდეგ გადარიცხავს სხვადასხვა თანხებს და ამით დაადგენს თუ რომელი ბლოკები შეესაბამება გადარიცხულ თანხას. ყველაფერი უკვე მზადაა. მოწინააღმდეგე გადარიცხავს რაიმე თანხას ერთი ბანკიდან მეორეში, მიიტაცებს კრიპტოგრამას და შეცვლის მას სხვა კრიპტიგრამით, რომელშიც თანხის რაოდენობა შეცვლილია. ეს თაღლითობა აღმოჩენილი იქნება, როდესაც ორივე ბანკი შეადარებს ერთმანეთთან გადარიცხვებს. საუკეთესო შემთხვევაში ეს მოხდება დღის ბოლოს. ამ დროისათვის კი თაღლითი უკვე გაქცეული იქნება ისეთ ქვეყანაში, რომელიც არ ახორციელებს დამნაშავის ექსტრადირებას ამ ქვეყანაში.

ყოველივე აქედან გამომდინარე, **ECB** რეჟიმის გამოყენება არ არის რეკომენდირებული, როდესაც ღია ტექსტი შეიცავს ერთ ბლოკზე მეტ ინფორმაციას, ან როდესაც გასაღები არ არის ერთჯერადი. შესაძლებელია უსაფრთხოების გაზრდა, თუკი თითოეულ ბლოკში ჩავრთავთ მაკონტროლებელ ბიტს.

თუ შიფროგრამის გადაცემის დროს დაშვებული იქნება შეცდომა, შეიცვლება ერთი ან რამდენიმე ბიტი, ან დაიკარგება ბიტი, შეცდომით მოხდება მხოლოდ იმ ბლოკის გაშიფრვა, რომელშიც დაშვებული იქნა შეცდომა.

4.14. შიფროტექსტის ბლოკების გადაბმის (CBC) რეჟიმი. რეჟიმი ისეა ორგანიზებული, რომ გაუძლოს შეტევას ტექსტის ამოღებით ან ჩამატებით. მათემატიკურად ეს რეჟიმი შეიძლება აღვწეროთ ასე:

$$c_i = E_k(m_i \oplus c_{i-1}) \quad \text{და} \quad m_i = c_{i-1} \oplus E_k^{-1}(c_i).$$



სურ. 4.6 **CBC** რეჟიმი.

ამ რეჟიმში შიფროტექსტის ერთი შეცდომითი ბიტი გავლენას ახდენს არა მარტო მთელი ბლოკის გაშიფრვაზე, არამედ აფუჭებს ერთ ბიტს ღია ტექსტის შემდეგ ბლოკშიც, რაც ადვილი

აღმოსაჩენია და მიუთითებს იმაზე, რომ ადგილი ჰქონდა შეტევას ტექსტის ამოღებით ან ჩამატებით. სამაგიეროდ, ამ რეჟიმში თავიდანაა აცილებული ის სუსტი მხარე, რომელიც გააჩნია **ECB** რეჟიმს (მხედველობაში გვაქვს რეჟიმის სისუტე ბლოკების ჩამატებისა და გამოკლების მიმართ). ეს ხდება ბლოკურ რეჟიმში უკუკავშირის მექანიზმის დამატებით (იხ. ნახ. 4.6), რომელიც უზრუნველყოფს კავშირს უკვე დაშიფრულ ბლოკსა და მის შემდეგ დასაშიფრ ბლოკს შორის. ამ მიზნით უკვე დაშიფრული ბლოკი ოპერაცია **XOR**-ის საშუალებით იკრიბება ღია ტექსტის შემდეგ ბლოკთან და მხოლოდ ამის შემდეგ ხდება მისი დაშიფრვა.

გასათვალისწინებელია, რომ CBC რეჟიმში მუშაობის დროს ღია ტექსტის ერთნაირი ბლოკები გადაიქცევა შიფროგრამის სხვადასხვა ბლოკებად მხოლოდ იმ შემთხვევაში, თუ განსხვავდებოდა ღია ტექსტის წინა ბლოკები. აქედან გამომდინარე, ორი ერთნაირი დასაწყისის მქონე ღია ტექსტი დაიშიფრება ერთნაირად მანამდე, სანამ არ შეგვხვდება მათ შორის განსხვავება. ასეთი რამ კი ხშირადაა მოსალოდნელი, როდესაც საქმე გვაქვს სტანდარტული შეტყობინებების გაგზავნასთან (მაგალითად ბანკებს შორის ფულადი გზავნილების გადაცემა). ეს შეიძლება კარგად გამოიყენოს კრიპტოანალიტიკოსმა, ამიტომ პირველი ბლოკის დაშიფრვის დროს როგორც წესი, იყენებენ რაიმე მონაცემებს, რომლებსაც უწოდებენ ***ინიციალიზაციის ვექტორს*** (Initialization Vector, IV). თუ ჩვენ ერთი და იგივე საწყისი ბლოკების მქონე შეტყობინებისათვის გამოვიყენებთ სხვადასხვა ინიციალიზაციის ვექტორს, მაშინ მივიღებთ სხვადასხვანაირად დაშიფრულ ბლოკებს. მართალია ეს ვექტორი არ წარმოადგენს გასაღებს, მაგრამ მისი არჩევის წესი გავლენას ახდენს სისტემის კრიპტომედეგობაზე, ამიტომ აუცილებელია გათვალისწინებული იყოს ეს მომენტი.

დაშიფრვის მათემატიკური ფორმულებიდან ადვილი შესამჩნევია, რომ ამ რეჟიმში ხდება ღია ტექსტის სტრუქტურის დამალვა მისი წინა ბლოკის შიფროტექსტთან შეკრების გზით, თუმცა ეს ხდება მაშინ, როდესაც გადასაცემი ბლოკების რაოდენობა მცირეა. თუ გადასაცემი ტექსტი არის ძალიან დიდი, მაშინ მხოლოდ ერთი შეკრება არ აღმოჩნდება საკმარისი გადაცემულ შიფროგრამაში ღია ტექსტის სტრუქტურის დასამალად და საჭირო გახდება სხვა დამატებითი ღონისძიებების გამოყენება. ამიტომ უნდა ვეცადოთ, რომ ერთი და იგივე გასაღებით და ერთი და იგივე ინიცილიზაციის ვექტორით არ გადავცეთ ძალიან დიდი მოცულობის ღია ტექსტი.

4.15. შეცდომების გამრავლება. როგორც ვნახეთ, ამ რეჟიმში m_{i+1} ბლოკის დაშიფრვაზე გავლენას ახდენს m_i ბლოკი, ამიტომ m_i ბლოკში დაშვებული თუნდაც ერთი შეცდომა (ბიტის შეცვლა), გავლენას მოახდენს ყველა დანარჩენი ბლოკის დაშიფრვაზე, მაგრამ დეშიფრაციის დროს მოხდება ცალსახად აღდგენა და შეცდომა დარჩება იქ სადაც ის იყო დაშვებული. მართლაც, დავუშვათ, დაშიფრვის დროს შეცდომა იქნა დაშვებული m_i ბლოკში და ყველა შემდეგი ბლოკებიც დაიშიფრა შეცდომით. როგორც ვიცით, დაშიფრვა ხდება ფორმულით $c_i = E_k(m_i \oplus c_{i-1})$, ხოლო დეშიფრაცია კი $m_i = c_{i-1} \oplus E_k^{-1}(c_i)$, ამიტომ გაშიფრვის შემდეგ ყოველი შეცდომითი ბლოკისათვის მივიღებთ:

$m_i^{/} = c_{i-1} \oplus E_k^{-1}(c_{i-1}) = c_{i-1} \oplus E_k^{-1}(E_k(m_i \oplus c_{i-1})) = c_{i-1} \oplus m_i \oplus c_{i-1} = m_i$, ანუ $m_i^{/} = m_i$.

დეშიფრაციის შემდეგ დაგვრჩება სწორედ ის ერთი შეცდომა, რომელიც დაშვებული იყო დაშიფრვის დროს.

სულ სხვანაირად გვაქვს საქმე მაშინ, როდესაც შეცდომა დაშვებულია შიფროგრამის გადაცემის დროს. დავუშვათ, გადაცემის დროს შიფროგრამის c_i ბლოკში ერთი ბიტი ინფორმაცია დამახინჯდა (მაგრამ არ მომხდარა ბიტის დაკარგვა, ან ჩამატება). მაშინ ეს ბლოკი გავლენას მოახდენს აგრეთვე c_{i+1} ბლოკის დეშიფრაციაზეც, ამასთან თვით c_i ბლოკის აღდგენა შეუძლებელი იქნება, ხოლო c_{i+1} ბლოკის გაშიფრვისას ღია ტექსტში გვექნება შეცდომა მხოლოდ ერთ ბიტში, იმ ადგილას, რომელიც შეესაბამება c_i დაშვებული შეცდომის ადგილს. აქედან გამომდინარე, შეგვიძლია ვთქვათ, რომ **CBC** რეჟიმი არის თვითსინქრონიზებადი (იხ. 4.17. ნაკადური შიფრები) ბლოკების დონზე. **CBC** რეჟიმი აბსოლუტურად უძლურია შიფროგრამაში ბიტების ჩამატების ან დაკრგვის მიმართ. თუ გადაცემის პროცესში დაიკარგა, ან ჩაემატა თუნდაც ერთი ბიტი, ყველა დანარჩენი ბლოკის მდგომარეობა წაიძვრება და დეშიფრაციის შემდეგ მივიღებთ რაღაც აბდაუბდას, ამიტომ, აუცილებელია, რომ თვით კრიპტოსისტემა, რომელიც გამოიყენება ამ რეჟიმში, რაღაცნაირად უნდა უზრუნველყოფდეს ბლოკების მთლიანობას.

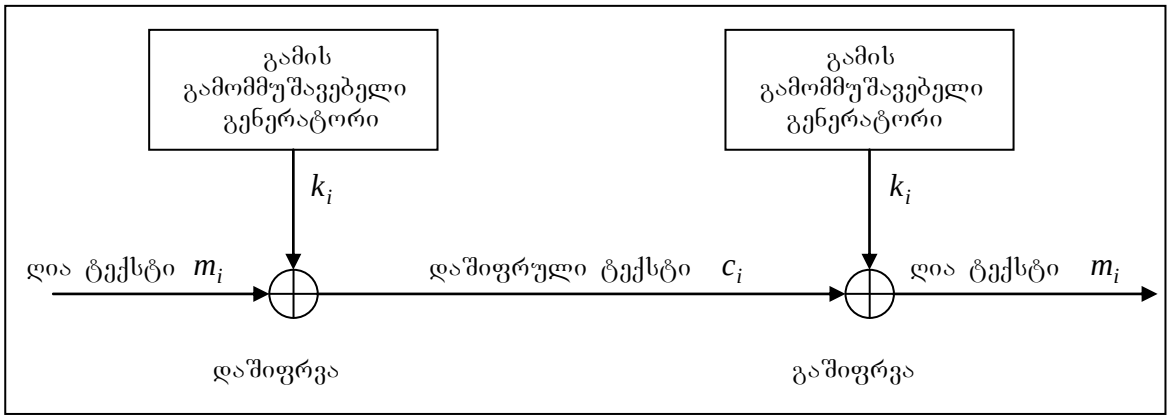
4.16. ინიციალიზაციის ვექტორის ამორჩევა. ჩვენ ზევით აღვნიშნეთ, რომ ინიციალიზაციის ვექტორი არ წარმოადგენს გასაღებს მაგრამ მისი არჩევა გავლენას ახდენს სისტემის კრიპტომედეგობაზე. ვნახოთ ეხლა, როგორ უნდა ავირჩიოთ ეს ვექტორი.

შესაძლებელია ინიციალიზაციის ვექტორად ავირჩიოთ ფიქსირებული ვექტორი, ანუ ყველა შეტყობინებისათვის გამოვიყენოთ ერთი და იგივე ვექტორი, მაგრამ ასეთ შემთხვევაში ის ვერ შეასრულებს თავის დანიშნულებას, სხვადასხვა შეტყობინებების მსგავსი საწყისი ბლოკები დაიშიფრება ერთნაირად, რაც გაუადვილებს მოქმედებას კრიპტოანალიტიკოსს.

ზოგჯერ ინიცილაიზაციის ვექტორად ირჩევენ შეტყობინებათა მთვლელს, ანუ პირველ შეტყობინებას შეუსაბამებენ ნოლს, მეორეს – ერთს და ასე შემდეგ, მაგრამ არც ასეთი არჩევანია ძალიან კარგი, რადგანაც, როდესაც საწყისს ბლოკებს შორის არის პატარა განსხვავება, შესაძლებელია ასეთმა მარტივმა ინიციალიზაციის ვექტორმა ეს განსხვავება **XOR**-ით შეკრების დროს მოსპოს და ჩვენ კვლავ მივიღოთ ერთნაირად დაშიფრული საწყისი ბლოკები.

ყველაზე კარგი იქნება, თუ ინიციალიზაციის ვექტორად ავიღებთ შემთხვევით სიდიდეს, მაგრამ ასეთი ვექტორის გამოყენება დაკავშირებულია გარკვეულ პრობლემებთან, რომელთაგან მთავარია, რომ სისტემაში უნდა გვქონდეს მაღალი ხარისხის შემთხვევით რიცხვთა გენერატორი, რაც მოითხოვს დამატებით რთული სამუშაოების ჩატარებას. გარდა ამისა, ეს შემთხვევითი სიდიდე უნდა გადაეცეს ადრესატს, რაც ზრდის ბლოკების რაოდენობას ერთით. ეს კი არაა რეკომენდირებული მოკლე შეტყობინებების გადაცემის დროს.

დღეისათვის ყველაზე საუკეთესოდ მიჩნეულია შემდეგი პროცედურა. პირველ რიგში შეტყობინებას, რომელიც უნდა დაიშიფროს მოცემული გასაღებით მიანიჭებენ უნიკალურ რიცხვს. ამას უწოდებენ შემთხვევას (ინგლისურად **nonce – number used once**). ამ შემთხვევის მთავარი თვისებაა მისი უნიკალურობა. არ შეიძლება ერთი და იგივე რიცხვის ორჯერ გამოყენება ერთ გასაღებთან. ინიციალიზაციის ვექტორი მიიღება ამ რიცხვის დაშიფრვით.



სურ. 4.7 ნაკადური შიფრი

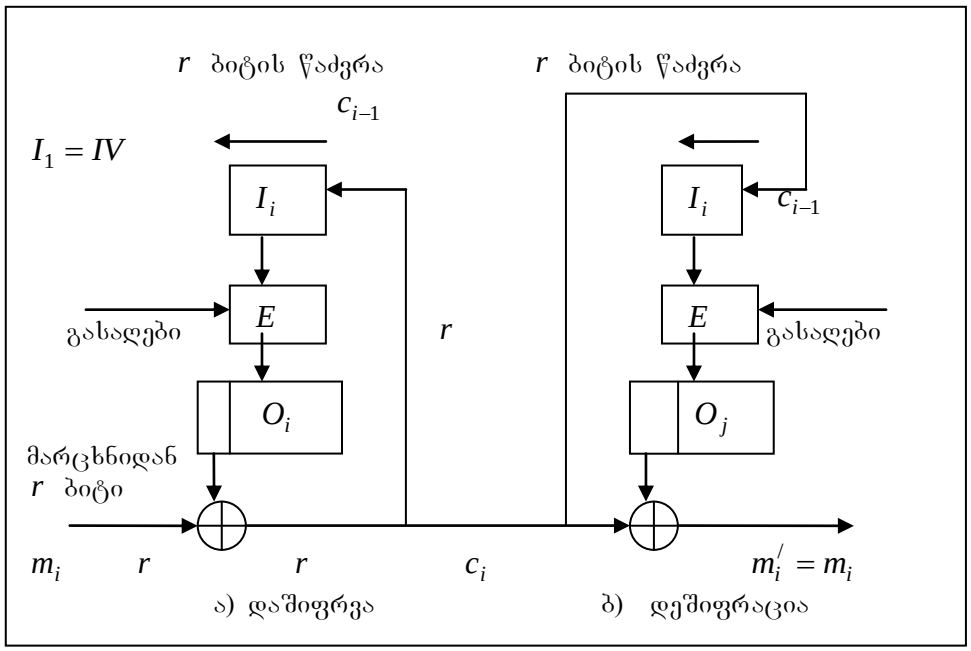
4.17. ნაკადური შიფრები. იმისათვის, რომ განვიხილოთ დაშიფრვის დანარჩენი რეჟიმები, ჩვენ დაგვჭირდება ნაკადური შიფრის ცნება. განსხვავებით ბლოკური შიფრებისაგან ნაკადურ შიფრებში ხდება ღია ტექსტის თითოეული ბიტის (ან ბაიტის) გარდაქმნა შიფროტექსტად მაშინვე, როდესაც ის შემოდის ალგორითმში. ასეთი შიფრის რეალიზაცია მოცემულია სურ. 4.7

გენერატორი, რომელსაც ზოგჯერ უწოდებენ მორბენალი გასაღების გენერატორს (**running key generator)** გამოიმუშავებს ბიტების ნაკადს $k_1, k_2, ..., k_i, ...$, რომელსაც უწოდებენ გამას. ეს გამა და ღია ტექსტის $m_1, m_2, ..., m_i, ...$ ბიტები იკრიბება ოპერაცია **XOR**-ით და მიიღება შიფროტექსტი: $c_i = m_i \oplus k_i$. დეშიფრაციის დროს კვლავ იკრიბება უკვე შიფროტექსტი და გამა და მიიღება ღია ტექსტი. ასეთი შიფრის საიმედოობა მთლიანად დამოკიდებულია გენერატორის მიერ გამომუშავებულ გამაზე, რაც უფრო ახლოს იქნება ეს გამა ნამდვილად შემთხვევით მიმდევრობასთან, მით უფრო გაუჭირდება კრიპტოანალიტიკოსს მისი გატეხვა (ამ საკითხებს ჩვენ განვიხილავთ ცოტა მოგვიანებით). ეხლა კი გვაინტერესებს ასეთი შიფრების ერთი მნიშვნელოვანი თვისება, ისინი ძალიან მგრძნობიარენი არიან ყოველი შეცდომის მიმართ. საკმარისია გადაცემის დროს დაიკარგოს

ან ჩაემატოს ერთი ბიტი ინფორმაცია, რომ გაშიფრვა გახდება შეუძლებელი, ამიტომ აუცილებელია, გადამცემი და მიმღები გენერატორების ***სინქრონიზაცია***, რაც წარმოადგენს საკმაოდ რთულ პრობლემას. არსებობს ამ პრობლემის გადაჭრის ორი გზა. პირველი ესაა ისეთი სპეციალური პროცედურების გათვალისწინება რეჟიმში, რომლებიც საშუალებას მოგვცემენ მოვახდინოთ დამშიფრავი და გამშიფრავი გენერატორების სინქრონიზაცია. ასეთ შიფრებს უწოდებენ ***სინქრონიზებად ნაკადურ შიფრებს.*** ასეთი შიფრების უპირატესობა მდგომარეობს იმაში, რომ არ ხდება შეცდომების გავრცელება, შეცდომა ერთ ბიტში (ბიტის შეცვლა და არა დაკარგვა, რაც უფრო ხშირადაა მოსალოდნელი) არ გამოიწვევს შეცდომას მომდევნო ბიტებში.

ამ პრობლემის გადაჭრა შეიძლება ***თვითსინქრონიზებადი ნაკადური შიფრის*** საშუალებითაც. ამ დროს გამის თითოეული ბიტი წარმოადგენს რაიმე წინა ბიტების ფუნქციას. რადგანაც გენერატორის მდგომარეობა მთლიანად დამოკიდებულია წინა ბიტზე, გამშიფრავი გენერატორი მიიღებს რა n ბიტს ავტომატურად მოვა სინქრონიზაციაში დამშიფრავ გენერატორთან. ასეთი გენერატორების შემთხვევაში შეცდომა ერთ ბიტში გამოიწვევს შეცდომის გავრცელებას n ბიტზე, რის შემდეგ გენერატორები კვლავ მოვლენ სინქრონიზაციაში.

4.18. შიფროტექსტით უკუკავშირის (CFB) რეჟიმი. ბლოკური შიფრებისათვისაც შესაძლებელია განვახორციელოთ თვითსინქრონიზებადი ნაკადური რეჟიმი. ასეთ რეჟიმს წარმოადგენს სწორედ **CFB** რეჟიმი. **CBC** რეჟიმში (ისევე, როგორც **ECB** რეჟიმში) დაშიფრული ინფორმაციის გადაცემა ხდება მხოლოდ მას შემდეგ, რაც დაიშიფრება მთელი ბლოკი ღია ტექსტისა (ბლოკის სიგრძე n-ის ტოლია), მაგრამ ზოგიერთ გამოყენებებში მოთხოვება, რომ ღია ტექსტის $r < n$ ბიტი ინფორმაცია დაიშიფროს ერთად და გადაცემული იქნას დაუყოვნებლივ. (ყველაზე ხშირად $r = 1$ ან $r = 8$). ასეთ შემთხვევებში შეიძლება გამოვიყენოთ შიფროტექსტით უკუკავშირის რეჟიმი (**CFB**), რომელიც ძირითადად გათვლილია ნაკადური შიფრებისათვის, თუმცა შესაძლებელია გამოვიყენოთ ბლოკური შიფრისთვისაც. (იხ. ნახ. 4.8). ამით საშუალება გვეძლევა აღარ ჩავატაროთ ღია ტექსტის შევსება მაშინ, როდესაც ღია ტექსტი არ არის ბლოკის სიგრძის ჯერადი.



სურ. 4.8 CFB რეჟიმი.

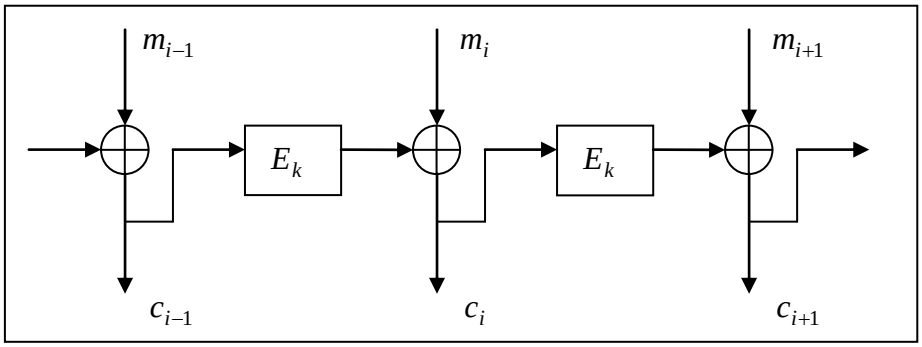
ამ რეჟიმში ხდება გარკვეული, $r < n$ (n არის ბლოკის სიგრძე) რაოდენობის ბიტების დაშიფრვა ისე, რომ არ დაველოდოთ მთელი ბლოკის ჩატვირთვას, თუმცა სხვათა შორის, თეორიულად, შესაძლებელია, რომ r მეტი იყოს ბლოკის სიგრძეზეც, თუმცა, როგორც წესი $r < n$. ისევე როგორც წინა რეჟიმში, აქაც გვაქვს ინიციალიზაციის ვექტორი, რომლის r ბიტიც იშიფრება დაშიფრვის ალგორითმით და შემდეგ ოპერაცია XOR-ის საშუალებით იკრიბება ღია ტექსტის პირველ r ბიტთან, რის შედეგადაც წარმოიშვება შიფროტექსტის პირველი r ბიტი. ეს შიფროტექსტი

გადაეცემა ადრესატს და ამავე დროს ის გადაეწოდება რეგისტრს, რომელშიც თავიდან მოთავსებულია ინიციალიზაციის ვექტორი. მოხდება წაძვრა მარცხნივ r ბიტზე, რის შედეგადაც დაშიფრული ტექსტი დაიკავებს მარჯვენა r ბიტს, ხოლო ინიციალიზაციის მარცხენა r ბიტი გადაიყრება. ზუსტად ასევე იშიფრება ღია ტექსტის შემდეგი r ბიტი და ასე გრძელდება მანამ, სანამ არ დაიშიფრება და გადაიცემა მთელი ტექსტი. გაშიფრვა ხდება შებრუნებული მიმდევრობით, ამასთან, მიაქციეთ ყურადღება, რომ ორივე შემთხვევაში ბლოკური ალგორითმი გამოიყენება მხოლოდ დაშიფრვის რეჟიმში, ამიტომ ეს ალგორითმი უნდა იყოს აუცილებლად სიმეტრიული. ასიმეტრიული ალგორითმის გამოყენება ამ რეჟიმში შეუძლებელია.

ისევე როგორც **CBC** რეჟიმში, აქაც, თუ ორი ღია ტექსტის საწყისი ბლოკები ემთხვევა ერთმანეთს, დაშიფრული ტექსტის ბლოკებიც დაემთხვევა ერთმანეთს, ამიტომ საჭიროა გამოყენებული იყოს სხვადასხვა ინიციალიზაციის ვექტორი. ინიციალიზაციის ვექტორი ამ რეჟიმშიც შეიძლება იყოს ღია (ანუ გადაცემის დროს არ დაიშიფროს), მაგრამ აუცილებლად უნდა იყოს უნიკალური. ესაა მოთხოვნა და არა რეკომენდაცია, როგორც ეს იყო CBC რეჟიმში. ბლოკები აქაც გადაბმულია ერთმანეთზე, ამიტომ r ბიტი სიგრძის ბლოკი c_j დამოკიდებულია შესაბამისად ღია ტექსტის m_j და წინა ბლოკებზე. შესაბამისად, დაშიფრული ბლოკების მიმდევრობის შეცვლა გავლენას მოახდენს გაშიფრვაზე. იმისათვის, რომ კორექტულად გაიშიფროს შიფროგრამის ერთი ბლოკი, საჭიროა ზუსტად $\lceil n/r \rceil$ ცალი შიფროგრამის ბლოკი იყოს დაშიფრული კორექტულად.

ერთი ან მეტ ბიტში შეცდომა r-ბიტიან შიფროტექსტის c_j ბლოკში გამოიწვევს გაშიფრვის შეცდომებს ამ და მომდევნო $\lceil n/r \rceil$ დაშიფრულ ბლოკებში, ანუ სანამ არასწორი ბიტები იქნებიან რეგისტრში. დეშიფრაციის შედეგად მიღებულ $m_j^{/}$ ბლოკში იქნება მხოლოდ ერთი შეცდომა, რომელიც შეესაბამება შეცდომას c_j ბლოკში, დანარჩენი ბლოკების დეშიფრაცია კი იქნება შეუძლებელი, რადგანაც შეცდომების რაოდენობა იქნება 50%. აქედან გამომდინარე, უნდა ვივარაუდოთ, რომ მოწინააღმდეგეს აქტიური შეტევის დროს გაუჩნდება სურვილი, შეცვალოს შიფროგრამის თუნდაც ერთი ბიტი ინფორმაცია. მას შემდეგ რაც ეს ბიტები დატოვებენ რეგისტრს, შეცდომები გასწორდება, რადგანაც **CFB** რეჟიმიც წარმოადგენს თვითსინქრონიზებად შიფრს.

თუ $r = n$, მაშინ დაშიფრვის პროცესს **CFB** რეჟიმში ექნება უფრო მარტივი სახე (იხ. სურ. 4.9). ამ სქემიდან კარგად ჩანს, რომ **CFB** რეჟიმში, ისევე, როგორც **CBC** რეჟიმში, ხდება ბლოკების გადაბმა.



სურ. 4.9 დაშიფრვა CFB რეჟიმში, როდესაც $r = n$

4.19. გამოსასვლელით უკუკავშირის (OFB) რეჟიმი. პირველ რიგში შევნიშნოთ, რომ ამ რეჟიმში ბლოკური შიფრით არ იშიფრება უშუალოდ ღია ტექსტი. ბლოკური შიფრი გამოიყენება გამის გამოსამუშავებლად, რომელიც არაა დამოკიდებული არც ღია და არც დაშიფრულ ტექსტზე (იხ. ნახ. 4.10). ასეთ უკუკავშირს ზოგჯერ უწოდებენ შინაგან უკუკავშირს. როგორც **CFB** რეჟიმის დროს, აქაც ბლოკური შიფრი ტრანსფორმირდება ნაკადურში. ამასთან, ცალკეული შეცდომა კრიპტოგრამაში გავლენას ახდენს არა მარტო ამ ბლოკის გაშიფრვაზე, არამედ შემდეგი ბლოკის გაშიფრვაზეც, როგორც ეს არის **CBC** რეჟიმში.

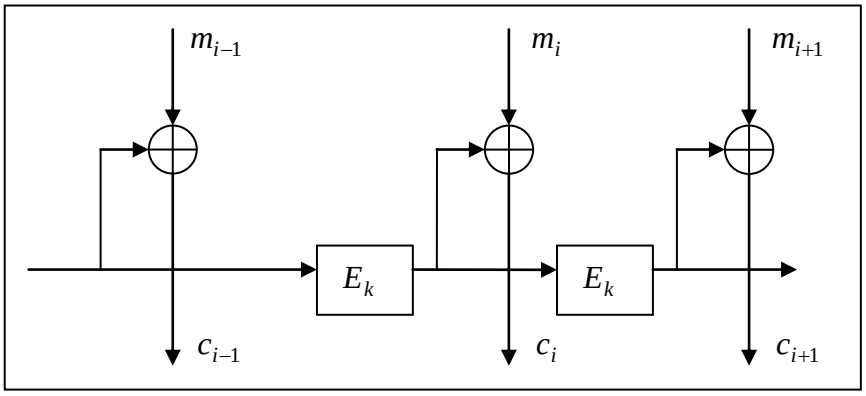
გამოსასვლელით უკუკავშირის რეჟიმი შეიძლება გამოვიყენოთ მაშინ, როდესაც შეცდომების გავრცელება უნდა იყოს გამორიცხული. ეს რეჟიმი წარმოადგენს ბლოკური შიფრის გამოყენებას სინქრონულ ნაკადურ შიფრად.

OFB რეჟიმში დაშიფრვის ალგორითმის მიერ გამომუშავებული გამა, რომელიც შემდეგ იკრიბება ღია ტექსტთან, გარკვეული პერიოდის შემდეგ იწყებს გამეორებას, რაც ასუსტებს რეჟიმის კრიპტომედეგობას. თუ უკუკავშირი ხორციელდება n ბიტით, სადაც n არის ბლოკის სიგრძე, მაშინ ამ პერიოდის სიგრძე იქნება $2^n - 1$. თუ $n = 64$, ეს ძალიან დიდი რიცხვია, მაგრამ თუ უკუკავშირი ხორციელდება r ბიტით $(r < n)$, მაშინ ეს რიცხვი მცირდება დაახლოებით $2^{n/2} - 1$-მდე, რაც უკვე აღარ არის საკმარისი. სწორედ ამიტომ საერთაშორისო სტანდარტი მოითხოვს, რომ უკუკავშირი ხორციელდებოდეს ბლოკის სიგრძის ვექტორით, განსხვავებით ამერიკული სტანდარტისაგან, რომელშიც დასაშვებია, რომ ამ ვექტორის სიგრძე ნაკლები იყოს ბლოკის სიგრძეზე.

მათემატიკურად n ბიტიანი უკუკავშირის შემთხვევაში დაშიფრვისა და გაშიფრვის პროცედურებს აქვთ შემდეგი სახე:

$$c_i = m_i \oplus s_i; \qquad s_i = E_k(s_{i-1})$$
$$m_i = c_i \oplus s_i; \qquad s_i = E_k(s_{i-1})$$

სადაც s_i არის გამის i-ური ელემენტი. ამ ფორმულებიდან ჩანს **OFB** რეჟიმის კიდევ ერთი უპირატესობა, როგორც დასაშიფრად, ასევე დეშიფრაციისათვის გამოიყენება მხოლოდ დაშიფრვის პროცედურა, რაც საშუალებას გვაძლევს აღარ ავაგოთ ბლოკური შიფრის დეშიფრაციის პროცედურა. ეს ამარტივებს ალგორითმს.



სურ. 4.10 **OFB** რეჟიმი

რადგანც **OFB** რეჟიმი ფაქტობრივად არის სინქრონული რეჟიმი, მასში არ ხდება შეცდომების გავრცელება. გაშიფრვის შემდეგ თქვენ მიიღებთ ზუსტად იმდენ შეცდომას და იმ ბიტებში, რამდენიც წარმოიშვა შიფროტექსტის გადაცემის დროს. მაგრამ თუ დაირღვა სინქრონიზაცია, გაშიფრის შემდეგ მიიღება ყოველგვარ აზრს მოკლებული ტექსტი. ეს რეჟიმი, ისევე როგორც მთლიანად სინქრონული ნაკადური შიფრები ძალიან მგრძნობიარეა ჩამატებით შეტევის მიმართ.

4.20. CTR - მთვლელის რეჟიმი. მიუხედავად იმისა, რომ **CTR** რეჟიმი უკვე ძალიან დიდი ხანია რაც გამოიყენება, ის არ ყოფილა სტანდარტიზებული **DES**-ის ოფიციალურ რეჟიმად, ამის გამო ის ნაკლებად განიხილებოდა ლიტერატურაში, მაგრამ დღეს ის უკვე სტანდარტიზებულია **NIST**-ის მიერ. ისევე, როგორც **OFB** და **CFB** რეჟიმები, **CTR** რეჟიმიც ეფუძნება ნაკადური შიფრის გამოყენებას და განისაზღვრება შემდეგი განტოლებებით:

$$k_i = E(K, nonce \| i) \quad i = 1,2,...k$$
$$c_i = m_i \oplus k$$

ამ რეჟიმშიც აუცილებელია ინიციალიზაციის ვექტორი, რომელიც უნდა განისაზღვროს $nonce$ ფუნქციის საშუალებით. კრიპტოგრაფიულ სისტემათა უმრავლესობაში, იმისათვის რომ

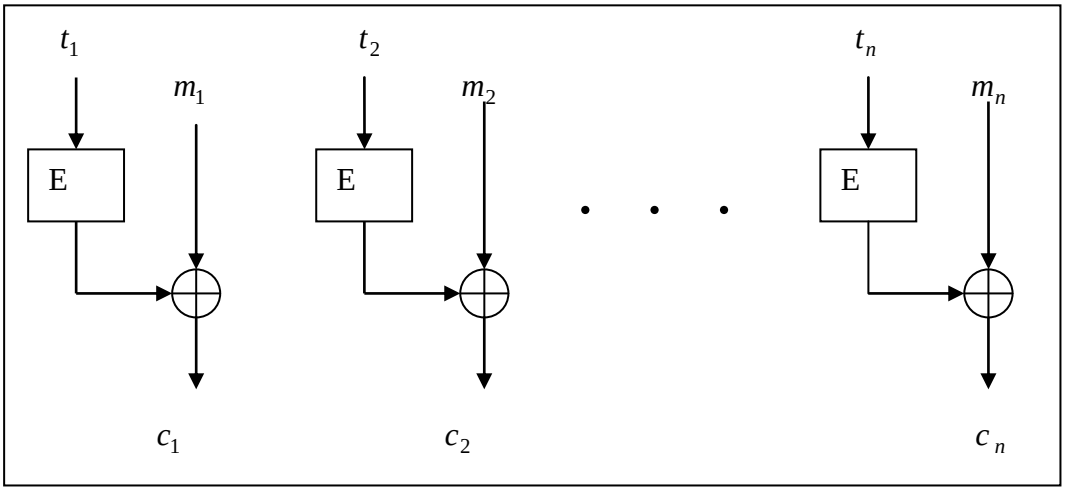
უზრუნველყოფილი იყოს ამ ვექტორის უნიკალურობა, გამოიყენება შეტყობინების ნომერი და რაიმე დამატებითი ინფორმაცია.

დასაშიფრი გამის გენერირებისათვის გამოიყენება ძალიან მარტივი მეთოდი. ხდება *nonce* ფუნქციის მნიშვნელობისა და მთვლელის კონკატენცია, რომელიც შემდეგ იშიფრება გამის ერთი ბლოკის მისაღებად. ეს მოითხოვს, რომ *nonce* ფუნქციის და მთვლელის მნიშვნელობები არ აღემატებოდეს ბლოკის ზომას, რაც თანამედროვე 128 ბიტიანი ბლოკებისათვის არ წარმოადგენს პრობლემას. ტიპური 128 ბიტიანი ბლოკი შეიძლება შედგებოდეს შეტყობინების 48 ბიტიანი ნომრის, დამატებითი ინფორმაციის თექვსმეტი ბიტისა და მთვლელის 64 ბიტიანი მნიშვნელობისაგან.

აუცილებელია, რომ **CTR** რეჟიმშიც გარანტირებული იყოს რომ "გასაღები + *nonce*" თითოეული კონკრეტული მნიშვნელობა გამოყენებული იქნება მხოლოდ ერთხელ. თვით მეთოდი ძალიან მარტივი გამოსაყენებელია. ის მოითხოვს მხოლოდ ბლოკური შიფრის დაშიფრვის პროცედურის რეალიზაციას, რადგანაც დაშიფრვისა და დეშიფრირების ფუნქციები CTR რეჟიმში ერთნაირია.

ყველაზე მნიშვნელოვანია, რომ რადგანაც გამის ნებისმიერი ბლოკი შეიძლება გამოვთვალოთ მყისიერად, **CTR** რეჟიმში ადვილად ხორციელდება წვდომა ღია ტექსტის ნებისმიერ ბლოკთან, ამიტომ შესაძლებელია დაშიფრვის პროცესისი განპარალელება.

კიდევ ერთი მნიშვნელოვანი თვისებაა, რომ ამ რეჟიმის უსაფრთხოება მთლიანადაა დამოკიდებული ბლოკური შიფრის საიმედოობაზე, ამიტომ თუ არ არსებობს წარმატებული შეტევა ბლოკურ შიფრზე, შეუძლებელი იქნება შეტევა **CTR** რეჟიმზეც.



სურ. 4.11 დაშიფრვა CTR რეჟიმში

საკონტროლო კითხვები:

1. რა ძირითად იდეას ეფუძნება ალგორითმი **IDEA,** რაც გვაძლევს საშუალებას არ გამოვიყენოთ **S** ბლოკები?
2. როგორ ხდება რაუნდული გასაღების გამომუშავება **IDEA**-ში?
3. აღწერეთ **IDEA**-ს ერთ რაუნდში მიმდინარე ოპერაციები.
4. აღწერეთ **BLOWFISH**-ის ფუნქცია და **S** ბლოკები.
5. რამდენი გასაღებია საჭირო **BLOWFISH**-ში?
6. როგორი იყო ავტორის იდეა გასაღებების საწყისი მონაცემებით შევსების შესახებ?
7. როგორ შეიძლება ვარეგულიროთ ალგორითმ **BLOWFISH**-ის მედეგობა?
8. რას ნიშნავს ბლოკური დაშიფრვის რეჟიმი და რა ძირითადი რეჟიმები იცით?
9. აღწერეთ **ECB** რეჟიმი. რა უპირატესობა აქვს ამ რეჟიმს სხვებისაგან განსხვავებით და რაში მდგომარეობს მისი სისუსტე?
10. აღწერეთ **CBC** რეჟიმი. რა მოხდება ამ რეჟიმში, თუ დაშიფრვის დროს რომელიმე

ბლოკში დაშვებული იქნება ერთი შეცდომა? თუ შეცდომა დაშვებული იქნება გადაცემის დროს (ერთი ბიტი გადაცემული იქნა შეცდომით)? თუ დაკარგა ან ჩაემატა ერთი ბიტი დაშიფრული ტექსტის გადაცემის დროს?

11. რას წარმოადგენს ინიციალიზაციის ვექტორი და რისთვის გამოიყენება ის? საიდუმლოა თუ არა ინიციალიზაციის ვექტორი?
12. აღწერეთ **CFB, OFB** და **CTR** რეჟიმები. რა განსხვავებაა მათ შორის? როდის შეიძლება მათი გამოყენება?
13. თქვენი აზრით, რომელი რეჟიმებია ყველაზე უფრო მოსახერხებელი და საიმედო?

სიმეტრიული კრიპტოალგორითმების კრიპტოანალიზის მეთოდები

5.1. შეტევის ტიპები. როგორც ვნახეთ, თანამედროვე ბლოკური შიფრებისათვის დამახასიათებელია მრავალჯერადი იტერაცია, რის შემდეგაც დაშიფრული ტექსტის თითოეული ბიტი დამოკიდებული ხდება თითქმის ყველა შესასვლელ ბიტზე. ამით ღია ტექსტის სტრუქტურა შიფროგრამაში კარგად იმალება. ამიტომ შეტევას მხოლოდ შიფროტექსტის საფუძველზე ფაქტობრივად აზრი ეკარგება და წარმატებული შეტევა შესაძლებელია მხოლოდ ღია ტექსტის, ან შერჩეული ღია ტექსტის საფუძველზე. სწორედ ასეთია ჩვენს მიერ ქვემოთ განხილული შეტევის მეთოდები.

თავის მხრივ კრიპტოსისტემის მედეგობა, რომელსაც ჩვენ ვიყენებთ ინფორმაციის კონფიდენციალობის დასაცავად, დამოკიდებულია სამ ძირითად ფაქტორზე:

- კრიპტოალგორითმის მედეგობა;
- გასაღების მედეგობა;
- იმ პროტოკოლის მედეგობა, რომლითაც ხდება კავშირის დამყარება ორ აბონენტს შორის.

თუ ჩვენს მიერ გამოყენებული კრიპტოალგორითმი შესაძლებელია გატყდეს, ანუ არსებობს შეტევა, რომელიც კრიპტოანალიტიკოს საშუალებას აძლევს გახსნას გასაღები უშუალოდ ალგორითმზე შეტევით, ცხადია, ასეთი კრიპტოალგორითმის გამოყენება არ შეიძლება. თუ ალგორითმი, რომლის საშუალებითაც ხდება გასაღებების გენერირება მოცემული კრიპტოალგორითმისათვის, უფრო სუსტია, ვიდრე თვით კრიპტოალგორითმი, მაშინ კრიპტოანალიტიკოსი შეეცდება გატეხოს ეს ალგორითმი და გამოთვალოს გასაღები და არა კრიპტოალგორითმი. იგივე ეხება გასაღებების განაწილების, გადაცემის, გამოცვლისა და განადგურების პროტოკოლებს. თუ კონფიდენციალურობის დასაცავად მისი ქსელში გადაცემის დროს გამოიყენება პროტოკოლი, რომლის მედეგობაც ნაკლებია შიფრის მედეგობაზე, ცხადია მოწინააღმდეგე შეტევას განახორციელებს პროტოკოლზე და არა შიფრზე. საბოლოოდ შეგვიძლია დავასკვნათ, რომ კრიპტოსისტემის მედეგობა ტოლია ამ სამი ფაქტორიდან ყველაზე სუსტის კრიპტომედეგობის. ამიტომ აუცილებელია, რომ ეს კრიპტომედეგობები იყოს ერთმანეთთან ახლოს.

5.2. კრიპტომედეგობის რაოდენობრივი შეფასების კრიტერიუმები. კრიპტომედეგობა შეტევების მიმართ რაოდენობრივად შეიძლება შევაფასოთ იმ რესურსებით, რომლებიც საჭიროა წარმატებული შეტევისათვის. ეს რესურსებია

- ინფორმაციის რაოდენობა, რომელიც აუცილებელია შეტევისათვის (მაგალითად დაშიფრული, ან ღია ტექსტების რაოდენობა, ან შერჩეული ღია ტექსტების რაოდენობა ან ტექსტთა წყვილების რაოდენობა).
- დრო, რომელიც აუცილებელია შეტევის განსახორციელებლად. როგორც წესი იზომება ალგორითმის ტესტური დაშიფრვის ოპერაციათა რაოდენობით, რომელიც სხვა პირობების შესრულების შემთხვევაში, საჭიროა გასაღების გამოსათვლელად.
- მეხსიერება, რომელიც აუცილებელია შეტევის დროს გამოყენებული ინფორმაციის შესანახად.

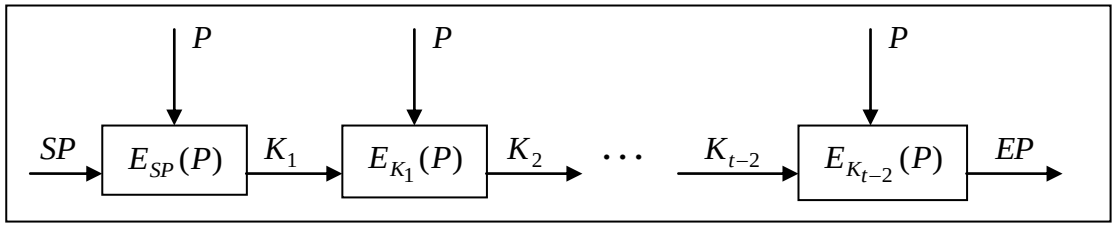
ასეთი სამი სიდიდით შეიძლება დავახასიათოთ ნებისმიერი შეტევა, განხორციელებული ნებისმიერ კრიპტოსისტემაზე. მაშინ, საუკეთესო შეტევა, რომელიც მოითხოვს მინიმალურ რესურსებს, განსაზღვრავს რამდენად კრიპტომედეგია მოცემული ალგორითმი.

5.3. შეტევა "დროისა და მეხსიერების კომპრომისით". ჩვენ აღარ განვიხილავთ ძალისმიერ შეტევას, რადგანაც უკვე ვიცით, რომ ნებისმიერი კრიპტოგრაფიული ალგორითმი შესაძლებელია გავტეხოთ ამ მეთოდით. ამისათვის საჭიროა ჩვენ გვქონდეს შიფროტექსტი და ვიცოდეთ დაშიფრვის (და შესაბამისი დეშიფრაციის) ალგორითმი და შესაძლო გასაღებების სიმრავლე. მაშინ ჩვენ შეგვიძლია თანმიმდევრულად გამოვცადოთ ყველა შესაძლო გასაღებები დაშიფრულ ტექსტზე, სანამ არ ვიპოვით ნამდვილ გასაღებს. ამ პროცედურის ჩატარებას თითქმის არ სჭირდება მეხსიერება, რადგანაც მცდარ გასაღებებს ჩვენ არ ვინახავთ, მაგრამ სჭირდება ძალიან დიდი დრო, ასე მაგალითად, თუ ამ შეტევას ვიყენებთ **DES**-ის წინააღმდეგ, ყველაზე უარეს შემთხვევაში დაგვჭირდება 2^{55}-ჯერ ჩავატაროთ ასეთი გამოთვლები, რათა ვიპოვით ნამდვილი გასაღები. დრო, რომელიც საჭიროა ასეთი რაოდენობის გამოთვლებისათვის იმდენად დიდია, რომ ვერავინ შეძლო ამ მეთოდით **DES**-ის გატეხვა (თუმცა უნდა აღინიშნოს, რომ დღეს ეს რიცხვი უკვე აღარაა მიუღწევად).

არსებობს მეორე გზაც – შეტევა შერჩეული ღია ტექსტის საფუძველზე. დავუშვათ ეკას, რომელსაც სურს ამ ალგორითმის გატეხვა და რომელიც უტევს შერჩეული ღია ტექსტის საფუძველზე, გააჩნია იმდენი დრო, რომ მას შეუძლია წინასწარ, სანამ ანი გადასცემს ბექას დაშიფრულ ტექსტს, ჩაატაროს ნებისმიერი გამოთვლები. მაშინ მას შეუძლია, P ფიქსირებული ღია ტექსტის ერთი ბლოკისათვის ყველა შესაძლო გასაღებებით გამოთვალოს შესაბამისი $C_K = E_K(P)$ შიფროტექსტი და შეადგინოს ცხრილი წყვილებისა (C_K, K), რომელსაც მოაწესრიგებს პირველი კოორდინატის მიხედვით. ასეთი ცხრილის არსებობის შემთხვევაში ეკას აღარ გაუჭირდება ანისა და ბექას მიერ შერჩეული ღია გასაღების გამოცნობა. ამისათვის საკმარისია ეკამ გაუგზავნოს ანის (ან ბექას) შერჩეული ღია ტექსტი და მიიღოს შესაბამისი შიფროტექსტი, ის თავის ცხრილში ადვილად იპოვნის იმ გასაღებს, რომელსაც იყენებენ ანი და ბექა. ამ შემთხვევაში ეკას აღარ ექნება დროის პრობლემა შიფროტექსტის გაშიფრვისას, მაგრამ გაუჩნდება მეხსიერების პრობლემა, რადგანაც იმ ცხრილის შენახვას, რომელსაც ეკა შეადგენს მოითხოვს ძალიან დიდ მეხსიერებას. ამასთან რეალურ შემთხვევებში გაუჩნდება დროის პრობლემა წინასწარი გამოთვლების დროს.

"დროისა და მეხსიერების კომპრომისით" შეტევის იდეა მდგომარეობს სწორედ იმაში, რომ მოხდეს რაღაც კომპრომისი ამ ორ მიდგომას, შორის და შევადგინოთ არა სრული ცხრილი, არამედ ისეთი ზომის ცხრილი, რომელიც დაიკავებს გარკვეული რაოდენობის მეხსიერებას და წინასწარი გამოთვლების დროც იქნება რეალური.

ასეთი ცხრილის შედგენის დროს ეკამ უნდა გაითვალისწინოს ოთხი პარამეტრი, რომელიც ზღუდავს მის შესაძლებლობებს: დრო, რომელიც საჭიროა ასეთი ცხრილის წინასწარ გამოსათვლელად, ცხრილის ზომები, რომელიც განსაზღვრავს საჭირო მეხსიერებას, წარმატების ალბათობა (რა ალბათობით გატეხავს შიფრს მოცემული ზომების ცხრილის შემთხვევაში) და დრო, რომელიც დასჭირდება ცხრილში გასაღების მოძებნას.



სურ. 5.1 დაშიფრვათა ჯაჭვი

5.4. ჰელმანის ალგორითმი (იდეალური ვარიანტი). სპეციალურად DES-ის გასატეხად ასეთი შეტევის ორიგინალური მეთოდი შემოგვთავაზა მარტინ ჰელმანმა 1980 წელს. იმისათვის, რომ კარგად გავიგოთ ეს მეთოდი, განვიხილოთ ჯერ იდეალური შემთხვევა. დავუშვათ გვაქვს კრიპოტოსისტემა, რომელშიც გასაღების და ბლოკის სიგრძეები ემთხვევა ერთმანეთს (დავუშვათ,

ორივე უდრის სამოცდაოთხ ბიტს). მაშინ ეკა პირველ რიგში დააფიქსირეს ღია ტექსტის ბლოკს (P), რომლითაც მან უნდა შეუტიოს კრიპტოსისტემას. ამის შემდეგ შემთხვევითად შეარჩევს 64-ბიტიან სტრიქონს, რომელსაც ის უწოდებს "საწყის წერტილს" (SP), დააფიქსირებს რამე t მთელ დადებით რიცხვს და შერჩეული SP-სა და P-ს საშუალებით ააგებს გასაღებების ***ჯაჭვს***:

$$K_0 = SP,\ K_1 = E(P, SP), K_2 = E(P, K_1),\ \ldots,\ K_{t-1} = E(P, K_{t-2}).$$

ასეთი ჯაჭვის აგება არ მოითხოვს დაშიფრვის ალგორითმის ცოდნას. ერთადერთი, რაც ეკას სჭირედება, ესაა ბლოკის და გასაღების სიგრძეების ცოდნა, რაც შეეხება დამშიფრავ ფუნქციას, მის როლში ეკას შეუძლია გამოიყენოს ნებისმიერი ფუნქცია, რომელიც 64-ბიტიან სტრიქონს ასახავს 64-ბიტიან სტრიქონში.

ეკა შეარჩევს m ცალ ასეთ "საწყის წერტილებს" და მიიღებს m ცალ ჯაჭვს. იდეალურ შემთხვევაში ყველა ეს ჯაჭვი იქნება პარალელური (ანუ ერთი და იგივე გასაღებები არ იქნება სხვადასხვა ჯაჭვებში. თუ $m = 2^{32}$ და $t = 2^{32}$, მაშინ ეკას ექნება აგებული ყველა შესაძლო გასაღები. ეკა თავის ცხრილში შეინახავს მხოლოდ ჯაჭვის საწყის (SP_j) და (EP_j) საბოლოო რგოლებს, ამიტომ მას დასჭირდება მეხსიერება $2m$ სიტყვის შესანახად, სადაც თითოეული სიტყვა არის 64-ბიტიანი სტრიქონი. მიღებული T ცხრილი, რომელიც წარმოადგენს ორ სვეტს, მოწესრიგებულს პირველი კოორდინატით, გამოიყენება ყოველი შეტევის დროს, ამიტომ დრო, რომელიც დაიხარჯება მის შედგენაზე, თანაბრად უნდა განაწილდეს შეტევების რაოდენობაზე. ამ ცხრილის შედგენის შემდეგ თითოეული შეტევა ხორციელდება შემდეგი სცენარით: ეკა აიღებს იმ P ღია ტექსტს, რომლითაც გამოთვალა ცხრილი და გაუგზავნის ანის ან ბექას (ან ორივეს ერთად). მიღებული C შიფროგრამით მან უნდა მოძებნოს ისეთი გასაღები K, რომ $C = E(P, K)$. ამისათვის ის იწყებს ჯაჭვის აგებას მიღებული შიფროტექსტიდან

$$C = X_0,\ X_1 = E(P, X_0),\ X_2 = E(P, X_1), \ldots$$

მაქსიმუმ t-მდე. ამასთან, ყოველი X_i-ის გამოთვლის შემდეგ ეკა ადარებს X_i-ს ცხრილის მეორე სვეტის ელემენტებთან (EP_j). რადგანაც ჩვენ ვიხილავთ იდეალურ შემთხვევას და ამასთან ცხრილი მოიცავს ყველა 2^{64} გასაღებს, ამიტომ C აუცილებლად წარმოადგენს რომელიმე ჯაჭვის რგოლს და არაუმეტეს t ბიჯისა ეკა იპოვის ამ ჯაჭვს და C-ს ამ ჯაჭვზე, რაც ტოლფასია იმის, რომ ეკა დაადგენს C-ს j და i კოორდინატებს. მაშინ ეკა აიღებს ცხრილდან (SP_j) და დაიწყებს P-ს დაშიფრვას

$$Y_0 = SP_j,\ Y_1 = E(P, Y_0), \ldots,\ Y_{t-j} = X_0 = E(P, Y_{t-j-1}).$$

რადგანაც

$$C = X_0 = E(P, K) \Rightarrow K = Y_{t-j-1}$$

ვნახოთ რა დაუჯდება ეკას ასეთი შეტევა. წინასწარი გამოთვლები მოითხოვს $tm = 2^{64}$ გამოთვლებს. კონკრეტული ერთი შეტევის დროს C-ს მიხედვით (EP_j)-ს პოვნა საშუალოდ მოითხოვს 2^{31} გამოთვლებს. გამოთვლების ასეთივე რაოდენობაა საჭირო საწყისი (SP_j) რგოლიდან გასაღების საპოვნელად, ასე, რომ კონკრეტული შეტევისათვის საჭიროა 2^{32} გამოთვლა. თუ ეკა აპირებს მხოლოდ ერთხელ გამოიცნოს გასაღები, მაშინ ცხადია, მას ურჩევნია გამოიყენოს ძალისმიერი შეტევა, რომლისთვისაც საჭიროა 2^{63} გამოთვლა, მაგრამ თუ მას სურს, რომ ყოველთვის შეძლოს ანისა და ბექას შორის საუბრების მოსმენა, მაშინ ცხადია უმჯობესია ამ ალგორითმის გამოყენება. ამასთან, ეკას ყოველთვის შეუძლია შეამციროს წინასწარი გამოთვლების რაოდენობა m-ისა და t-ს შემცირებით. ასეთ შემთხვევაში ცხადია აღარ ექნება ასპროცენტიანი გამოცნობის საშუალება, მაგრამ მას შეუძლია აირჩიოს მისთვის სასურველი ალბათობა, რომელიც ტოლი იქნება ზუსტად იმ ფართობისა, სრული ცხრილს რა ნაწილსაც შექმნის ეკა.

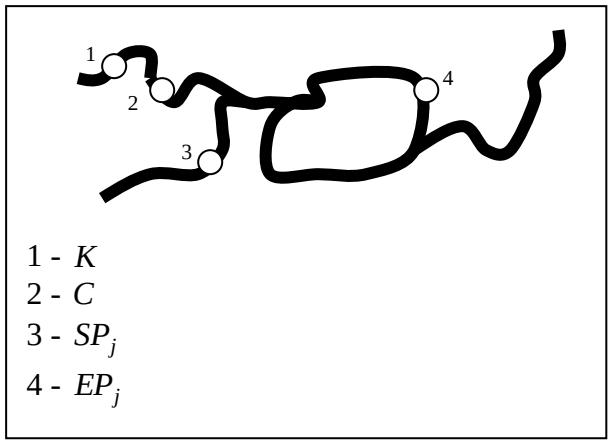
5.5. არაიდეალური ვარიანტი. პირველი, რითაც რეალური სისტემები განსხვავდებიან ზევით აღწერილი იდეალური ვარიანტისაგან, არის დაშიფრვის ჯაჭვები. იმის მაგივრად, რომ ეს ჯაჭვები იყოს ისე ლამაზად განცალკევებული გასაღებების სივრცეში, როგორც ეს იყო ჩვენს იდეალურ ვარიანტში, რეალობაში ისინი ხშირად იკვეთებიან, ერწყმიან ერთმანეთს, ან სულაც ქმნიან მარყუჟებს (იხ. სურ .5.2).

წინასწარი გამოთვლების დროს ასეთი ჯაჭვები გამოიწვევენ დროის უქმად დაკარგვას, რადგანაც ჯაჭვების რაღაც ნაწილები და გასაღებები გამეორდება. გარდა ამისა შეტევის დროს ასეთმა ჯაჭვებმა შეიძლება აგვაცდინოს მიზანს და ვერ შევძლოთ გასაღების გამოთვლა. ამის მაგალითი მოყვანილია სურ. 5.2. დავუშვათ, ეკამ მიიღო რა C, დაიწყო შეტევა და გამოთვალა შესაბამისი EP_j. ამის შემდეგ ის იწყებს გასაღების გამოთვლას EP_j-ს შესაბამისი SP_j-დან. ცხადია, ის ამას ვერ შეძლებს, რადგანაც C არ მდებარეობს $SP_j \leftrightarrow EP_j$ ჯაჭვზე.

თუ ეკა შეძლებს მოიცილოს ასეთი მარყუჟები და გადაბმები, მაშინ ის შეძლებს ჯერ ერთი შეამციროს წინასწარი გამოთვლების მოცულობა და მეორეც თავიდან აიცილოს შეცდომები შეტევის დროს. ამ მიზნით ეკა იყენებს "შემთხვევით ფუნქციას" F-ს რომელსაც იყენებს ჯაჭვის გამოთვლის დროს:

$$K_0 = SP,\ \ K_1 = F(E(P, SP)),\ K_2 = F(E(P, K_1)),\ \ldots,\ K_{t-1} = EP = F(E(P, K_{t-2})).$$

როდესაც $n = k$, ასეთი "შემთხვევითი" ფუნქცია შეიძლება იყოს გადანაცვლება. ასეთი შემთხვევითი ფუნქცია საშუალებას აძლევს ეკას გაყოს ერთმანეთისაგან ორი შეერთებული ჯაჭვი და დაიყვანოს ისინი მხოლოდ ერთ წერტილში გადაკვეთმადე. ეს ნიშნავს, რომ ეკას შეუძლია გამოიყენოს იდეალური შემთხვევისათვის აღწერილი ძებნის პრინციპი რეალურ შემთხვევებში, როდესაც გასაღების და შიფროტექსტის ბიტური სტრიქონები ერთმანეთის ტოლია.



სურ. 5.2 ჯაჭვები, რომლებიც ქმნიან მარყუჟებს და ერწყმიან ერთმანეთს

მაგრამ არც ეს ვარიანტი არ არის მთლად რეალური, რადგანაც უმრავლესობა ბლოკურ შიფრებში გასაღების და ბლოკის სიგრძეები ერთმანეთს არ ემთხვევა, რაც გამორიცხავს ღია ტექსტის დაშიფრვას შიფროტექსტის საშუალებით. მაგალითად, **DES**-ში, რომლის გასატეხადაც მ. ჰელმანმა შექმნა ეს მეთოდი, შიფროტექსტის სიგრძე მეტია გასაღების სიგრძეზე, **AES WIJNDEAL**-ში კი შეიძლება სამივე შემთხვევა. ამიტომ ეკას, გარდა შემთხვევითი ფუნქციისა, დასჭირდება აგრეთვე R შემკუმშავი ფუნქცია, თუ ბლოკის სიგრძე მეტია გასაღების სიგრძეზე, ან S გამჭიმავი ფუნქცია, თუ ბლოკის სიგრძე ნაკლებია გასაღების სიგრძეზე. ამ ფუნქციების საშუალებით ეკა შესძლებს დაიყვანოს ბლოკის სიგრძე გასაღების სიგრძეზე

$$R(C_K) = R(E_K(P)) \ \text{ და } \ S(C_K) = S(E_K(P)).$$

ეს ფუნქცია შეიძლება იყოს ძალიან მარტივი. მაგალითად **DES**-თვის შემკუმშავი ფუნქცია შეიძლება იყოს გარკვეული პოზიციებიდან ექვსი ბიტის ამოყრის ფუნქცია. ასეთი გარდაქმნა აუცილებელია, მაგრამ ის კიდევ უფრო ართულებს გასაღების მოძებნის პროცედურას, რადგანაც, როდესაც ეკა იპოვის თავისი ცხრილის საშუალებით K გასაღებს, ის არ შეიძლება იყოს დარწმუნებული, რომ ნაპოვნია ზუსტი გასაღები. მართლაც, დავუშვათ, ეკამ ალგორითმ **DES**-ის შემთხვევაში ზემოთ აღწერილი მეთოდით იპოვა K გასაღები. მაგრამ რადგანაც R-ის განსაზღვრის არეა 2^{64}, ხოლო მნიშვნელობათა სიმრავლე 2^{56}, საშუალოდ იარსებებს კიდევ ვარიანტი $2^8 = 256$ შიფროტქსტებისა, რომლებიც შეიძლებოდა გადმოსულიყო იგივე მნიშვნელობაში. ამიტომ ეკას დასჭირდება

კიდევ სხვა მეთოდი, რათა შეამოწმოს გამოთვლილი გასაღები მართლაც ნამდვილი გასაღებია, თუ არა.

ალგორითმს, რომლის საშუალებითაც ეკას შეუძლია ჩაატაროს წინასწარი გამოთვლები, აქვს შემდეგი სახე:

შესასვლელი: დაშიფრვის ალგორითმი , ფიქსირებული ღია ტექსტი

პარამეტრები: l, n, t

წინასწარი გამოთვლები

1: $s = 1$-დან l -მდე გააკეთე

2: ამოირჩიე შემთხვევითად შემკუმშავი ფუნქცია R_s და განსაზღვრე

$f_s : k \mapsto R_s(C_k(m))$

3: $i = 1$ -დან n -მდე **გააკეთე**

4: ამოირჩიე შემთხვევითად $k^/$

5: $k \leftarrow k^/$

6: $j = 1$დან t -მდე **გააკეთე**

7: გამოთვალე $k \leftarrow f_s(k)$

8: დაამთავრე

9: შეიტანე $(k, k^/)$ ცხრილში T_s

10: **დაამთავრე**

11: **დაამთავრე**

ამ ალგორითმში დამატებულია კიდევ ერთი პარამეტრი l, რომელიც აღნიშნავს ცხრილების რაოდენობას, ანუ ეკას შეუძლია გამოთვალოს l ცალი პატარ-პატარა ცხრილები. ეს აძლევს მას საშუალებს დაფაროს გასაღებების სივრცის ის ნაწილები, რომელიც მას მიაჩნია უფრო მეტი ალბათობით წარმატების მომტანად. რა თქმა უნდა ეს ხდება მაშინ, როდესაც ეკა არ ადგენს მთელი სივრცის დამფარავ ცხრილს.

5.6. ორმხრივი ძებნის მეთოდი ("შუაში შეხვედრის" ალგორითმი). ორმხრივი ძებნის ალგორითმი მიეკუთვნება კრიპტოანალიზის კლასიკურ მეთოდებს, რომელიც ხშირად გვხვდება სხვადასხვა კონსტრუქციებში. ამ შეტევის იდეა მდგომარეობს იმაში, რომ ეკა იწყებს შეტევას ორი მხრიდან, (შეტევა ეკუთვნის ღია ტექსტით შეტევის ტიპს) ცნობილი ღია ტექსტიდან და ამ ტექსტის შესაბამისი შიფროგრამიდან. გამოიყენებს რა ან ყველა შესაძლო გასაღებს, ან გასაღებების იმ ნაწილს, რომელიც მისი აზრით ყველაზე ალბათურია რომ იყო გამოყენებული, ყოველ ნაბიჯზე ეკა აგებს შესაძლო ვარიანტებს როგორც ღია ტექსტიდან, ასევე შიფროგრამიდან და ინახავს მათ შესაბამისად S_1 და S_2 სიმრავლეებში. ასევე ყოველი ბიჯის შემდეგ ეკა ადარებს ელემენტებს ერთმანეთთან ამ სიმრავლეებიდან. თუ რომელიმე ბიჯზე წყვილი ელემენტებისა პირველი და მეორე სიმრავლეებიდან დაემთხვევა ერთმანეთს, ეს ნიშნავს, რომ ეკა იპოვნის გასაღებს. თუ დამთხვევა არ მოხდა და ბიჯების რაოდენობა გადასცდა ნახევარს, მაშინ ის გასაღებების სიმრავლე, რომელიც აირჩია ეკამ, არ შეიცავს ნამდვილ გასაღებს. როგორც აღვნიშნეთ ეს იდეა გამოყენება სხვადასხვა კონსტრუქციებში. ერთერთი ალგორითმი ასეთი შეტევისა გამოიყენება, როდესაც გასაღებების სიმრავლე წარმოადგენს ჯგუფს მათი კომპოზიციის მიმართ, ანუ, თუ ნებისმიერი k_i გასაღებისათვის მოიძებნება ისეთი წყვილი (k_j, k_l) გასაღებებისა, რომ $f(k_i, m) = f(k_j, f(k_l, m))$ მაშინ ამ შეტევის ალგორითმი ასეთია:

ბიჯი 1: $\boldsymbol{j = 1,2,...\sqrt{|K|}}$

შეასრულე შემდეგი:

1. ამოირჩიე შემთხვევითი გასაღები $\boldsymbol{k_j}$

2. დაშიფრე ღია $\boldsymbol{m}$ ტექსტი გასაღებით $\boldsymbol{k_j}$

$\boldsymbol{z_j := f(m, k_j)}$

ბიჯი 2: **მოახდინე ბაზის სორტირება პარამეტრით** $\boldsymbol{z_i}$

ბიჯი 3: $\boldsymbol{l = 1,2,...\sqrt{|K|}}$

შეასრულე შემდეგი:

1. ამოირჩიე შემთხვევითი გასაღები k_l

2. გაშიფრე შიფროგრამა გასაღებით k_l

$$w_l = f^{-1}(e, k_l)$$

ბიჯი 4: **მიღებული შედეგი შეადარე ბაზას** (z_j, k_j) **, თუ**

$$w_l = z_j$$ **მაშინ**

გამოიტანე (k_j, k_l).

დასასრული.

ალგორითმის სირთულე იქნება $O(\sqrt{|K|} \cdot \log|K|)$, სადაც მამრავლი $\log|K|$ დაკავშირებულია ბაზის სორტირებასთან. ძალისმიერი შეტევის სირთულე იქნებოდა $O(|K|)$. თუ გვექნება 2^{64} გასაღები, მაშინ სრული გადარჩევის სირთულე იქნება $O(2^{64})$, ხოლო შეტევისა შუაში შეხვედრის ალგორითმით კი - $O(\sqrt{2^{64}} \cdot \log 2^{64}) \approx O(2^{32} \cdot 2^6) = O(2^{38})$.

ეს შეტევა შეიძლება გამოვიყენოთ მაშინაც, როდესაც ხდება ორმაგი დაშიფრვა სხვადასხვა გასაღებებით. ალგორითმი არის ალბათური, თუმცა არსებობს მისი დეტერმინირებული ვარიანტიც, რომელიც ცნობილია როგორც "გიგანტური ნაბიჯი – პატარა ნაბიჯი" ალგორითმის სახით.

5.7. დიფერენციალური კრიპტოანალიზი. 1990 წელს ელი ბიჰამმა და ადი შამირმა სპეციალურად **DES**-ის გასატეხად შეიმუშავეს შერჩეული ტექსტით შეტევის ახალი მეთოდი, რომელსაც უწოდეს დიფერენციალური კრიპტოანალიზი. სანამ აღვწერდეთ უშუალოდ ამ მეთოდს, მოვიყვანოთ ერთი საინტერესო ფაქტი. როდესაც ე. ბიჰამმა და ა. შამირმა გამოიკვლიეს ამ ახალი მეთოდით **DES**-ი, აღმოჩნდა, რომ თექვსმეტრაუნდიანი იტერაცია, რომელიც გამოიყენება **DES**-ში, არის ის ზღვარი, რომლის შემდეგაც დიფერენციალური ანალიზის გამოყენებას აზრი ეკარგება, რადგანაც ის მოითხოვს ისეთივე რაოდენობის გადარჩევას, როგორსაც ძალისმიერი შეტევა. ეს საინტერესო ფაქტი მარტივად ახსნა **DES**-ის ერთერთმა ავტორმა, დონ კოპერსმიტმა, რომელმაც განაცხადა, რომ ალგორითმის დაპროექტების დროს მათთვის ცნობილი იყო ამ ტიპის შეტევის არსებობის შესახებ, მაგრამ სახელმწიფო ინტერესებიდან გამომდინარე, ისინი არ საუბრობდნენ ამის შესახებ.

დიფერენციალური კრიპტოანალიზი წარმოადგენს ღია ტექსტით შეტევის სტატისტიკურ მეთოდს, რომელიც გამოიყენებს სხვაობას (დიფერენციალს) ორ ტექსტს შორის, რათა გამოთვალოს შესაძლო გასაღებების სიმრავლე და მათი შესაბამისი ალბათობები და აღმოაჩინოს ამ გასაღებებს შორის ყველაზე უფრო მოსალოდნელი გასაღები.

5.8. დიფერენციალური შეტევის ტერმინები. სანამ დავიწყებთ უშუალოდ ამ შეტევის განხილვას, შემოვიტანოთ ის ტერმინები და აღნიშვნები, რომლებიც გამოიყენება დიფერენციალურ კრიპტოანალიზში. f-ის ქვეშ დიფერენციალურ კრიპტოანალიზში იგულისხმება ყველა ის გარდაქმნა, რომელსაც განიცდის ტექსტი ერთ რაუნდში. წყვილი ღია ტექსტებისა, რომლებიც გამოიყენება შეტევის დროს აღვნიშნოთ შესაბამისად M-ით და M^*-ით. ამასთან, შევნიშნოთ, რომ **DES**-ის ამ მეთოდით კრიპტოანალიზის დროს ღია ტექსტის ქვეშ იგულისხმება ტექსტი, რომელიც მიიღება საწყისი (IP) გადანაცვლების შემდეგ, რადგანაც, როგორც ალგორითმის აღწერის დროსაც აღვნიშნეთ, ეს გარდაქმნა არ წარმოადგენს რაიმე მნიშვნელოვანს კრიპტოგრაფიული თვალსაზრისით. **DES**-ის და მისი მსგავსი ალგორითმებისათვის სხვაობა ტექსტებს შორის ეწოდება ამ ტექსტების ოპერაცია **XOR**-ით შეკრების შედეგს რომელსაც უწოდებენ ღია ტექსტი **XOR**-ით. ის აღინიშნება $M' = M \oplus M^*$. ღია ტექსტის მარცხენა და მარჯვენა ნაწილები აღინიშნება შესაბამისად M_L-ით და M_R-ით. $M = M_L \parallel M_R$ (|| არის კონაქტაციის, ანუ გადაბმის ნიშანი). ღია ტექსტების შესაბამისი შიფროტექსტები აღინიშნება E-თი და E^*-ით (აქ შიფროგრამების ქვეშ იგულისხმება ტექსტი, რომელიც გამოდის მარტივი გადანაცვლების **P** ბლოკიდან (მხდეველობაში არ მიიღება საწყისი გადანცვლების შებრუნებული გადანაცვლება). ანალოგიურად ღია ტექსტებისა, "შიფროტექსტი **XOR**-ით", ესაა $E' = E \oplus E^*$ სიდიდე, რომელიც აღნიშნავს მიღებული შიფროტექსტების სხვაობას **XOR**-ით. ანალოგიურად ღია ტექსტებისა, შიფროგრამისთვისაც გვექნება, რომ $E = E_L \parallel E_R$. გამფართოებელი გადანაცვლება აღინიშნება $Ex(X)$, მარტივი გადანაცვლება $P(X)$ და გადანაცვლება S ბლოკებში Si-ით. ორი ტექსტის სხვაობას, რომელიც შედის რომელიმე ბლოკში, უწოდე-

ბენ **XOR**-ით შესასვლელს, შესაბამისად, ბლოკიდან გამოსულ სხვაობას, **XOR**-ით გამოსასვლელს. რიცხვების მარტივად ჩაწერის მიზნით (განსაკუთრებით ცხრილებში) გამოიყენება თექვსმეტობითი ფუძე. რიცხვს, რომელიც ჩაწერილია თექვსმეტობით წარმოდგენაში, ინდექსად უზის ასო x, მაგალითად, $1D_x$. $S1_I$ -ით აღნიშნულია პირველ ბლოკში შემავალი სხვაობა $S1_O$ -ით კი – გამოსული სხვაობა. ასევე აღინიშნება სხვა ბლოკებისთვისაც შესასვლელი და გამოსასვლელი სხვაობები.

5.9. დიფერენციალი (სხვაობა) იცვლება მხოლოდ S ბლოკში. იმისათვის, რომ გაადვილდეს დიფერენციალური კრიპტოანალიზის მათემატიკური ანალიზი, იგულისხმება, რომ რაუნდის გასაღებები წარმოადგენენ დამოუკიდებელ გასაღებებს, ანუ არ არიან ნაწარმოები რაიმე საწყისი გასაღებიდან. როგორც ექსპერიმენტებმა აჩვენა, შეტევა ორივე შემთხვევაში აღწევს ერთი და იგივე შედეგს, მაგრამ იმ ალბათობების თეორიული ანალიზი, რომლებიც გამოითვლება დიფერენციალურ კრიპტოანალიზში, დამოკიდებული გასაღებების შემთხვევაში გაცილებით რთულია.

როგორც უკვე აღვნიშნეთ, დიფერენციალური კრიპტოანალიზის მიზანია გააანალიზოს სხვაობის ქცევა ტექსტების f ფუნქციით გარდაქმნის დროს და ამის საფუძველზე დაადგინოს გასაღების მნიშვნელობა. პირველი გარდაქმნა, რომელსაც ტექსტი განიცდის, არის გამფართოებელი გადანაცვლება, რომლის შედეგადაც ოცდათორმეტი ბიტისაგან მიიღება ორმოცდარვა ბიტი. თუ მოცემულია **XOR**-ით შესასვლელის მნიშვნელობა ამ ბლოკის შესასვლელზე, **XOR**-ით გამოსასვლელის მნიშვნელობის დადგენა შესაძლებელია შემდეგი ფორმულის საშუალებით:

$$. \; Ex(X) \oplus Ex(X^*) = Ex(X + X^*)$$

XOR-ით გამოსასვლელის მნიშვნელობა არ იცვლება გასაღებთან შეკრების დროსაც. მართლაც

$$(X \oplus K) \oplus (X^* \oplus K) = X \oplus X^*$$

ბლოკიდან **XOR**-ით გამოსასვლელის მნიშვნელობა მარტივი გადანაცვლების ბლოკში იცვლება შემდეგი ფორმულით:

$$. \; P(X) \oplus P(X^*) = P(X \oplus X^*)$$

f ფუნქციიდან **XOR**-ით გამოსასვლელის მნიშვნელობა წრფივადაა დამოკიდებული **XOR** ოპერაციაზე სხვადასხვა რაუნდების შეერთების დროს:

$$(X \oplus Y) \oplus (X^* \oplus Y^*) = (X \oplus X^*) \oplus (Y \oplus Y^*)$$

რაც გვაძლევს გარანტიას, რომ ცალსახად გვეცოდინება **XOR**-ით გამოსასვლელის მნიშვნელობა. ერთადერთი რჩება S ბლოკები, რომლებისთვისაც **XOR**-ით შესასვლელის მნიშვნელობის ცოდნა არ გვაძლევს გარანტიას, რომ ცალსახად გვეცოდინება **XOR**-ით გამოსასვლელის მნიშვნელობა. ნებისმიერ S ბლოკში შედის ექვსი ბიტი ინფორმაცია, ამიტომ **XOR**-ით შესასვლელის მნიშვნელობები ტოლია სამოცდაოთხის. შესაბამისად, ბლოკიდან გამოდის ოთხი ბიტი ინფორმაცია, რის გამოც **XOR**-ით გამოსასვლელის მნიშვნელობათა სიმრავლე შეიცავს თექვსმეტ ელემენტს. აქედან გამოდის, რომ შესასვლელების ზოგიერთ სხვადასხვა მნიშვნელობებს აუცილებლად უნდა შეესაბამებოდეს გამოსასვლელების ერთი და იგივე მნიშვნელობა. საშუალოდ, ოთხ შესასვლელს ერთი მნიშვნელობა. მაგრამ ეს მნიშვნელობები არ არიან თანაბრად განაწილებულები (იხ. სურ.5.3).

პირველ რიგში, შევნიშნოთ, რომ განსაკუთრებული შემთხვევაა, როდესაც **XOR**-ით შესასვლელის მნიშვნელობა ტოლია ნოლის. ასეთ შემთხვევაში გამოსასვლელიც იქნება ნოლის ტოლი.

XOR-ით შესასვლელის თითქმის ყველა მნიშვნელობებისათვის, არსებობს გამოსასვლელის ისეთი მნიშვნელობები, რომლებიც ამ შესასვლელიდან არ მიიღება.

ამ ცხრილიდან მაგალითად, ჩვენ ვხედავთ, რომ შესასვლელს შეესაბამება შესაძლო თექვსმეტიდან მხოლოდ რვა გამოსასვლელი მნიშვნელობა, რომლებიც არათანაბრად არიან განაწილებული, ამიტომ ჩვენ შეგვიძლია ვამტკიცოთ, რომ, თუ ბლოკის შესასვლელზე, გვაქვს რიცხვი 1_x, გამოსასვლელზე ალბათობით $3/32$ გვექნება 3_x.

ასეთი არათანაბარი განაწილება გამოწვეულია S ბლოკების აგების იმ პრინციპებით, რომლებიც ჩამოთვლილი იყო ალგორითმის განხილვის დროს.

შესას ვლე ლი	გამოსასვლელი სხვაობა															
	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	A_x	B_x	C_x	D_x	E_x	F_x
0_x	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

1_x	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
2_x	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
3_x	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
4_x	0	0	0	6	0	10	10	6	0	4	6	4	2	8	6	2
5_x	4	8	6	2	2	4	4	2	0	4	4	0	12	2	4	6
6_x	0	4	2	4	8	2	6	2	8	4	4	2	4	2	0	12
7_x	2	4	10	4	0	4	8	4	2	4	8	2	2	2	4	4
8_x	0	0	0	12	0	8	8	4	0	6	2	8	8	2	2	4
9_x	10	2	4	0	2	4	6	0	2	2	8	0	10	0	2	12
A_x	0	8	6	2	2	8	6	0	6	4	6	0	4	0	2	10
B_x	2	4	0	10	2	2	4	0	2	6	2	6	6	4	2	12
C_x	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
D_x	6	6	4	8	4	8	2	6	0	6	4	6	0	2	0	2
E_x	0	4	8	8	6	6	4	0	6	6	4	0	0	4	0	8
F_x	2	0	2	4	4	6	4	2	4	8	2	2	2	6	8	8
10_x	0	0	0	0	0	0	2	14	0	6	6	12	4	6	8	6
11_x	6	8	2	4	6	4	8	6	4	0	6	6	0	4	0	0
12_x	0	8	4	2	6	6	4	6	6	4	2	6	6	0	4	0
13_x	2	4	4	6	2	0	4	6	2	0	6	8	4	6	4	6
14_x	0	8	8	0	10	0	4	2	8	2	2	4	4	8	4	0
15_x	0	4	6	4	2	2	4	10	6	2	0	10	0	4	6	4
16_x	0	8	10	8	0	2	2	6	10	2	0	2	0	6	2	6
17_x	4	4	6	0	10	6	0	2	4	4	4	6	6	6	2	0
18_x	0	6	6	0	8	4	2	2	2	4	6	8	6	6	2	2
19_x	2	6	2	4	0	8	4	6	10	4	0	4	2	8	4	0
$1A_x$	0	6	4	0	4	6	6	6	6	2	2	0	4	4	6	8
$1B_x$	4	4	2	4	10	6	6	4	6	2	2	4	2	2	4	2
$1C_x$	0	10	10	6	6	0	0	12	6	4	0	0	2	4	4	0
$1D_x$	4	2	4	0	8	0	0	2	10	0	2	6	6	6	14	0
$1E_x$	0	2	6	0	14	2	0	0	6	4	10	8	2	2	6	2
$1F_x$	2	4	10	6	2	2	2	8	6	8	0	0	0	4	6	4
20_x	0	0	0	10	0	12	8	2	0	6	4	4	4	2	0	2
21_x	0	4	2	4	4	8	10	0	4	4	10	0	4	0	2	8
22_x	10	4	6	2	2	8	2	2	2	2	6	0	4	0	4	10
23_x	0	4	4	8	0	2	6	0	6	6	2	10	2	4	0	10
24_x	12	0	0	2	2	2	2	0	14	14	2	0	2	6	2	4
25_x	6	4	4	12	4	4	4	10	2	2	2	0	4	2	2	2
26_x	0	0	4	10	10	10	2	4	0	4	6	4	4	4	2	0
27_x	10	4	2	0	2	4	2	0	4	8	0	4	8	8	4	4

28_x	12	2	2	8	2	6	12	0	0	2	6	0	4	0	6	2
29_x	4	2	2	10	0	2	4	0	0	14	10	2	4	6	0	4
$2A_x$	4	2	4	6	0	2	8	2	2	14	2	6	2	6	2	2
$2B_x$	12	2	2	2	4	6	6	2	0	2	6	2	6	0	8	4
$2C_x$	4	2	2	4	0	2	10	4	2	2	4	8	8	4	2	6
$2D_x$	6	2	6	2	8	4	4	4	2	4	6	0	8	2	0	6
$2E$	6	6	2	2	0	2	4	6	4	0	6	2	12	2	6	4
$2F_x$	2	2	2	2	2	6	8	8	2	4	4	6	8	2	4	2
30_x	0	4	6	0	12	6	2	2	8	2	4	4	6	2	2	4
31_x	4	8	2	10	2	2	2	2	6	0	0	2	2	4	10	8
32_x	4	2	6	4	4	2	2	4	6	6	4	8	2	2	8	0
33_x	4	4	6	2	10	8	4	2	4	0	2	2	4	6	2	4
34_x	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
35_x	2	2	4	0	8	0	0	0	14	4	6	8	0	2	14	0
36_x	2	6	2	2	8	0	2	2	4	2	6	8	6	4	10	0
37_x	2	2	12	4	2	4	4	10	4	4	2	6	0	2	2	4
38_x	0	6	2	2	2	0	2	2	4	6	4	4	4	6	10	10
39_x	6	2	2	4	12	6	4	8	4	0	2	4	2	4	4	0
$3A_x$	6	4	6	4	6	8	0	6	2	2	6	2	2	6	4	0
$3B_x$	2	6	4	0	0	2	4	6	4	6	8	6	4	4	6	2
$3C_x$	0	10	4	0	12	0	4	2	6	0	4	12	4	4	2	0
$3D_x$	0	8	6	2	2	6	0	8	4	4	0	4	0	12	4	4
$3E_x$	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
$3F_x$	4	8	4	2	4	0	2	4	4	2	4	8	8	6	2	2

სურ. 5.3 XOR-ით შესასვლელის მნიშვნელობებსა და XOR-ით გამოსასვლელი მნიშვნელობების განაწილება $S1$ ბლოკისათვის. შესასვლელი და გამოსასვლელი მნიშვნელობები ჩაწერილია თექვსმეტობითი ფუძით, ხოლო მათი რაოდენობები ათობითით.

5.10. შესაძლო გასაღებების დადგენა. სხვაობათა განაწილების ცხრილი, რომელიც რა თქმა უნდა შედგენილი უნდა იყოს რვავე ბლოკისათვის, საშუალებას გვაძლევს მოვძებნოთ ის შესაძლო შესასვლელი და გამოსასვლელი წყვილები ტექსტებისა, რომლებიც იძლევიან მოცემულ **XOR**-ით შესასვლელსა და **XOR**-ით გამოსასვლელს.

ამის საჩვენებლად განვიხილოთ მარტივი მაგალითი: ავიღოთ ჩანაწერი $S1$ ბლოკის ცხრილიდან $34_x \to 4_x$ (ორობითი ფუძით იქნება $110100 \to 0100$). ამ შესასვლელსა და გამოსასვლელს აქვს მხოლოდ ორი მნიშვნელობა, ამიტომ, ამ სხვაობას შეიძლება იძლეოდეს მხოლოდ ორი წყვილი ტექსტებისა და ამასთან ისინი იქნებიან ურთიერთშებრუნებულნი, ანუ თუ პირველი წყვილია $(S1_I, S1_I^*)$, მაშინ მეორე წყვილი იქნება $(S1_I^*, S1_I)$. იმისათვის, რომ მოვძებნოთ ეს წყვილები, უნდა შევადგინოთ ცხრილი, რომელშიც ყველა იმ **XOR**-ით გამოსასვლელი მნიშვნელობებისათვის, რომლებიც ფიქსირებული **XOR**-ით შესასვლელი მნიშვნელობისათვის გვაქვს, ჩამოთვლილი იქნება შესაძლო შესასვლელი მნიშვნელობები. ასეთი ცხრილი **XOR**-ით შესასვლელისათვის 34_x მოცემულია სურ. 5.4

გამოსასვლელი **XOR**-ით $(S1_O^{/})$	შესასვლელი **XOR**-ით $S1_I^{/} = 34_x$ შესაძლო შესასვლელები $(S1_I)$
1_x	$3_x, F_x, 1E_x, 1F_x, 2A_x, 2B_x, 37_x, 3B_x,$
2_x	$4_x, 5_x, E_x, 11_x, 12_x, 14_x, 1A_x, 1B_x, 20_x, 25_x, 26_x, 2E_x, 2F_x, 30_x, 31_x, 3A_x$
3_x	$1_x, 2_x, 15_x, 21_x, 35_x, 36_x$
4_x	$13_x, 27_x$
7_x	$0_x, 8_x, D_x, 17_x, 18_x, 1D_x, 23_x, 29_x, 2C_x, 34_x, 39_x, 3C_x$
8_x	$9_x, C_x, 19_x, 2D_x, 38_x, 3D_x$
D_x	$6_x, 10_x, 16_x, 1C_x, 22_x, 24_x, 28_x, 32_x$
F_x	$7_x, A_x, B_x, 33_x, 3E_x, 3F_x$

სურ. 5.4 $S1_I^{/} = 34_x$ შესასვლელისათვის და $(S1_O^{/})$ გამოსასვლელებისათვის ყველა შესაძლო შესასვლელების განაწილების ცხრილი.

ამ ცხრილიდან ვხედავთ, რომ $34_x \to 4_x$ შეესაბამება შესასვლელები 13_x და 27_x. თუ ჩვენ ამ რიცხვებს გადავიყვანთ ორობით სისტემაში და გამოვიყენებთ $S1$ ბლოკის ცხრილის, მაშინ ვნახავთ, რომ შესაბამისი გამოსასვლელები იქნება 6_x და 2_x. ვნახოთ ეხლა როგორ შეიძლება გამოვიყენოთ ეს ინფორმაცია გასაღების ბიტების მოსაძებნად. განვიხილოთ ახალი მაგალითი. დავუშვათ, ვიცით, რომ $S1_E = 1_x$, $S1_E = 35_x$ და $S1_O^{/} = D_x$ და გვინდა ვიპოვოთ $S1_K$ გასაღების მნიშვნელობა. **XOR**-ით შესასვლელი $S1_E^{/} = S1_I^{/} = 34_x$ $S1_K$ გასაღების გარეშე.

სურ. 5.4 მოცემული ცხრილიდან ვხედავთ, რომ ამ **XOR**-ით შესასვლელს (34_x) შეესაბამება რვა XOR-ით გამოსასვლელი, რაც იძლევა გასაღების რვა შესაძლო მნიშვნელობებს (რადგანაც $S_K = S_E \oplus S_I$). ეს შესაძლო გასაღებები მოცემულია ცხრილში 5.1

ცხრილი 5.1.

$S1$ ბლოკის შესასვლელები	შესაძლო გასაღებები
$6_x, 32_x$	$7_x, 33_x$
$10_x, 24_x$	$11_x, 25_x$
$16_x, 22_x$	$17_x, 23_x$
$1C_x, 28_x$	$1D_x, 29_x$

ცხრილი 5.2.

$S1$ ბლოკის შესასვლელები	შესაძლო გასაღებები
$1_x, 35_x$	$20_x, 14_x$
$2_x, 36_x$	$23_x, 17_x$
$15_x, 21_x$	$34_x, 0_x$

თითოეული სტრიქონი ამ ცხრილში აღწერს ორ წყვილს შესავლელი მნიშვნელობებსა ერთი და იგივე შესასვლელებით, მაგრამ სხვადასხვა მიმდევრობით. თითოეულ წყვილს შეესაბამება ერთი გასაღები, ამიტომ სტრიქონში გვხვდება ორი გასაღები. ამ რვა შესაძლო გასაღებიდან ერთი უნდა იყოს ნამდვილი გასაღები. იმისათვის, რომ დავადგინოთ რომელი, საჭიროა კიდევ დამატებითი წყვილი, რომელიც მოგვცემს $S1_K$ გასაღების დამატებით კანდიდატებს. დავუშვათ, ეს წყვილი იყოს $S1_E = 21_x, S1_E^{*} = 15_x$. **XOR**-ით გამოსასვლელი ამ შემთხვევაში იქნება $S1_O^{/} = 3_x$. შესაძლო ექვსი შესასვლელი და მათი შესაბამისი შესაძლო გასაღებები მოცემულია ცხრილში 5.2 ნამდვილი გასაღები უნდა იყოს ამ ცხრილშიც. ცხრილების შედარებით ადვილად დავასკვნით,

ნამდვილი გასაღები შეიძლება იყოს $23_x, 17_x$ წყვილიდან ერთერთი. მოცემული **XOR**-ით შესასვლელით (34_x) ნამდვილი გასაღების ამორჩევა შეუძლებელია, მაგრამ გარჩევა გახდება შესაძლებელი, თუ ჩვენ გამოვიყენებთ შესასვლელ წყვილს, რომლის **XOR**-ით შესასვლელით განსხვავებული იქნება 34_x.

როგორც ვხედავთ, ასეთი შეტევით ძალიან მარტივია **DES**-ის ერთი რაუნდის გახსნა, რადგანაც ჩვენ ვიცით S ბლოკებიდან გამოსული მნიშვნელობები (გამოითვლება შიფროტექსტიდან), მაგრამ როდესაც რაუნდების რაოდენობა იზრდება, რთულდება ანალიზიც, თუმცა სპეციალურად შერჩეული ღია ტექსტების წყვილებისა და მათი რაოდენობის გაზრდით, შესაძლებელია მივაღწიოთ სასურველ მიზანს, გავიგოთ დაშიფრვის გასაღების თუნდაც გარკვეული ნაწილი, რათა შემდეგ დარჩენილი ბიტები გავიგოთ ძალისმიერი შეტევის საფუძველზე. ამისათვის უნდა ავაგოთ მეორე, მესამე და ა.შ. ხარისხის დიფერენციალები.

შევნიშნოთ, რომ დიფერენციალური კრიპტოანალიზი ძირითადად მაინც გამოიყენება კრიპტოსისტემის მედეგობის თეორიულად შესამოწმებლად და არა სისტემის პრაქტიკულად გასატეხად. ამას ადასტურებს ავტორების მიერ ჩატარებული **DES**-ის ანალიზიც. იმისათვის, რომ გატეხოთ თექვსმეტრაუნდიანი **DES**-ი, თქვენ უნდა გქონდეთ ერთსა და იმავე გასაღებზე დაშიფრული ან 2^{47} შერჩეული ღია ტექსტი, ან 2^{55} ღია ტექსტი, რაც პრაქტიკულად გამორიცხულია.

5.10. წრფივი კრიპტოანალიზის ძირითადი იდეა. წრფივი კრიპტოანალიზის მეთოდი პირველად გამოაქვეყნა მ. მათსუიმ 1993 წელს. ეს მეთოდი, ისევე როგორც დიფერენციალური კრიპტოანალიზი, წარმოადგენს შეტევას ღია ტექსტის საფუძველზე და ისევე როგორც დიფერენციალური კრიპტოანალიზი, წარმოადგენს სტატისტიკურ მეთოდს. განსხვავება მათ შორის მდგომარეობს იმაში, რომ თუ დიფერენციალური კრიპტოანალიზი შეისწავლის არაწრფივი გარდაქმნის სტატისტიკას და ამის საფუძველზე ცდილობს აღმოაჩინოს გასაღები, წრფივი ანალიზი ცდილობს სტატისტიკის საფუძველზე მოახდინოს არაწრფივი ფუნქციის აპროქსიმაცია წრფივი ფუნქციების საშუალებით და ამ გზით მიაღწიოს დასახულ მიზანს. თუ განვიხილავთ ამ მიდგომას ბიტების დონეზე (რომელზედაც სინამდვილეში ხდება ოპერაციების შესრულება ნებისმიერ შიფრში), მაშინ შეგვიძლია ვთქვათ, რომ წრფივი ანალიზის მიზანია მოვახდინოთ შიფრის რაუნდების ოპერაციის აპროქსიმაცია წრფივი გამოსახულებით, სადაც წრფივი ფუნქციის ქვეშ იგულისხმება ოპერაცია **XOR.**

კონკრეტულად, მეთოდის არსი მდგომარობს ისეთი

$$X_{i_1} \oplus \ldots \oplus X_{i_n} \oplus Y_{j_1} \oplus \ldots \oplus Y_{j_m} = 0 \quad (5.1.)$$

წრფივი დამოკიდებულებების მოძებნაში, რომლებიც სრულდება $0{,}5$-ზე გაცილებით მეტ და გაცილებით პატარა ალბათობებით. (გაცილებით განსხვავდება კრიპტოგაფრიულად ნიშნავს, რომ შესაძლებელია სტატისტიკურად გავარჩიოთ ერთმანეთისაგან შემთხვევითი ბინარული მიმდევრობა მიმდევრობისაგან, რომელიც ემორჩილება მოცემულ წრფივ დამოკიდებულებას). (5.1.) გამოსახულებაში X_i, Y_j, შესაბამისად არიან ღია ტექსტისა და შიფროგრამის ბიტების გარკვეული ქვესიმრავლეები. (5.1.) ტოლობიდან ჭეშმარიტი გასაღები განისაზღვრება როგორც ყველაზე მაღალალბათური, რომლისთვისაც ყველაზე ხშირად სრულდება (5.1) პირობა.

რადგანაც (5.1.) სახის წრფივ განტოლებათა სისტემის ამოხსნა არ წარმოადგენს სირთულეს, პრაქტიკულად წრფივი კრიპტოანალიზის ამოცანა დაიყვანება ასეთი სახის ჯამების მოძებნაზე, რომლებიც სრულდება ყველაზე უფრო ხშირად და გამოთვლილი გასაღებიც იქნება ყველაზე უფრო ალბათური.

5.11. ბულის ფუნქციის არაწრფივობა. განვიხილოთ მაგალითი. ავიღოთ DES-ის S0 ბლოკის მეორე სტრიქონი და ვნახოთ რას უდრის გამოსასვლელის უმცირესი ბიტი. ეს იქნება ბულის ფუნქცია (აღვნიშნოთ ის f -ით), რომელიც დამოკიდებული იქნება ოთხ ბიტზე X_1, X_2, X_3, X_4 (ცვლადზე), $f(X_1, X_2, X_3, X_4)$.

განვიხილოთ აგრეთვე $g(x) = X_1 \oplus X_2 \oplus X_3 \oplus X_4$ აფინური ბულის ფუნქცია და შევადაროთ ისინი ერთმანეთს. შედეგები მოყვანილია ცხრილში 5.3.

როგორც ამ ცხრილიდან ჩანს, f და g ფუნქციების მნიშვნელობები მხოლოდ ოთხ პოზიციაში არ ემთხვევა ერთმანეთს, ამიტომ, შეგვიძლია ვათქვათ, რომ $NL(f) = 4$, სადაც NL აღნიშნავს ფუნქციის არაწრფივობას. g ფუნქცია წარმოადგენს f ფუნქციის საუკეთესო აფინურ აპროქ-

სიმაციას. ზოგადად შეიძლება არსებობდეს რამდენიმე საუკეთესო აფინური აპროქსიმაცია მაგალითად, f ფუნქციის მეორე საუკეთესო მააპროქსიმირებელი ფუნქციაა $g_1(x) = 1 \oplus X_1 \oplus X_2 \oplus X_3$.

ცხრილი 5.3

X_1	X_2	X_3	X_4	f	g	X_1	X_2	X_3	X_4	f	g
0	0	0	0	0	0	1	0	0	0	0	1
0	0	0	1	1	1	1	0	0	1	0	0
0	0	1	0	1	1	1	0	1	0	0	0
0	0	1	1	0	0	1	0	1	1	1	1
0	1	0	0	0	1	1	1	0	0	1	0
0	1	0	1	0	0	1	1	0	1	1	1
0	1	1	0	1	0	1	1	1	0	1	1
0	1	1	1	1	1	1	1	1	1	0	0

.

იმისათვის, რომ უზრუნველყოფილი იყოს შიფრის მედეგობა წრფივი კრიპტოანალიზის მიმართ, უნდა შევარჩიოთ ისეთი დამშიფრავი ფუნქცია, რომლისთვისაც (5.1.) ტოლობის შესრულების ალბათობა რაც შეიძლება ახლოს იქნება $0{,}5$. ეს კი მიიღწევა მხოლოდ მაშინ, თუ ფუნქცია იქნება ნამდვილად არაწრფივი.

განსაზღვრება 5.1. *განვსაზღვროთ ფუნქციის არაწრფივობა $NL(f)$ როგორც უმცირესი მანძილი ჰემინგის აზრით ცხრილის სახით მოცემულ ფუნქციასა და მააპროქსიმებელ ფუნქციებს შორის, ანუ არგუმენტთა იმ ნაკრებების რაოდენობა, რომლებზედაც f ფუნქციის მნიშვნელობა არ ემთხვევა მააპროქსიმებელი აფინური ფუნქციის მნიშვნელობას.*

თავისთავად ცხადია, რომ აფინური ფუნქციის არაწრფივობა ტოლია ნოლის. დაბალანსებული ბულის ფუნქციის წრფივობა კი იქნება აუცილებლად ლუწი რიცხვი. ზოგადად, სამართლიანია შემდეგი უტოლობა

$$0 \le NL(f) \le 2^{n-1} - 2^{(n-1)/2},$$

სადაც n არის ფუნქციის არგუმენტების რიცხვი.

5.12. მ. მათსუის ლემა. იმისათვის, რომ ჩვენ მოვახდინოთ S ბლოკის წრფივი აპროქსიმაცია, ანუ შევძლოთ საბოლოო კრიპტოგარდაქმნისათვის ვიპოვოთ (5.1) სახით მოცემული წრფივი ჯამები შესაბამისი დაშიფრვის ოპერატორების წრფივი ჯამებით, დაგვჭირდება შემთხვევითი ფუნქციის რამდენიმე თვისება. დავუშვათ, მოცემული გვაქვს ორი შემთხვევითი ბინარული ცვლადი X_1 და X_2. მაშინ მათ შორის შესაძლებელია უმარტივესი

$$X_1 \oplus X_2 = 0$$

წრფივი დამოკიდებულების განსაზღვრა, რომელიც ტოლფასია

$$X_1 = X_2 .$$

მეორეს მხრივ გამოსახულება

$$X_1 \oplus X_2 = 1$$

არის აფინური გამოსახულება და ტოლფასია

$$X_1 \ne X_2 .$$

დავუშვათ ეხლა მოცემული გვაქვს ალბათობების განაწილება ამ ცვლადებისათვის:

$$\Pr(X_1 = i) = \begin{cases} p_1; i = 0 \\ 1 - p_1; i = 1 \end{cases}$$

და

$$\Pr(X_2 = i) = \begin{cases} p_2; i = 0 \\ 1 - p_2; i = 1 \end{cases}$$

თუ ეს ორი ცვლადი დამოუკიდებელია, მაშინ გვექნება

$$\Pr(X_1 = i; X_2 = j) = \begin{cases} p_1 p_2; i = 0, j = 0 \\ p_1(1 - p_2); i = 0, j = 1 \\ (1 - p_1) p_2; i = 1, j = 0 \\ (1 - p_1)(1 - p_2); i = 1, j = 1 \end{cases}$$

საიდანაც გამოვა, რომ

$$\Pr(X_1 \oplus X_2 = 0) = p_1 p_2 + (1 - p_1)(1 - p_2) .$$

თუ ჩავთვლით, რომ $p_1 = 1/2 + \varepsilon_1$ და $p_2 = 1/2 + \varepsilon_2$, სადაც ε_1 და ε_2 ***წრფივობისკენ გადახრის*** ალბათობება და $-1/2 \le \varepsilon_1, \varepsilon_2 \le 1/2$, მაშინ

$$\Pr(X_1 \oplus X_2 = 0) = 1/2 + 2\varepsilon_1\varepsilon_2 = 1/2 + \varepsilon_{1,2}$$

სადაც $\varepsilon_{1,2} = 2\varepsilon_1\varepsilon_2$ არის $X_1 + X_2 = 0$ გამოსახულების წრფივობისკენ გადახრის ალბათობა. მათსუიმ დაამტკიცა შემდეგი ლემა.

<u>ლემა 5.1. (მათსუის ლემა).</u> *n დამუკიდებელი, ბინარული შემთხვევითი $X_1, X_2, ..., X_n$ ცვლადისათვის ადგილი აქვს ტოლობას*

$$\Pr(X_1 \oplus X_2 \oplus ... \oplus X_n = 0) = 1/2 + 2^{n-1} \prod_{i=1}^{n} \varepsilon_i ,$$

სადაც $\varepsilon_{1,2,...,n} = 2^{n-1} \prod_{i=1}^{n} \varepsilon_i$ არის $X_1 \oplus X_2 \oplus ... \oplus X_n = 0$ გამოსახულების წრფივობისკენ გადახრის ალბათობა.

შევნიშნოთ, რომ თუ თუნდაც ერთი $\varepsilon_i = 0$, ანუ p_i უდრის $1/2$, მაშინ ალბათობა ჯამისა .

$$\Pr(X_1 \oplus X_2 \oplus ... \oplus X_n = 0) = 1/2$$

ეს ლემა საშუალებას გვაძლევს წრფივ გამოსახულებათა კომბინაციით მივიღოთ ახალი წრფივი კომბინაციები და გამოვთვალოთ მათი ალბათობები. მაგალითად, დავუშვათ, მოცემული გვაქვს ბინარული ცვლადები X_1, X_2, X_3, X_4 და დავუშვათ ვიცით, რომ

$$\Pr(X_1 \oplus X_2 = 0) = 1/2 + \varepsilon_{1,2} \quad \text{და} \quad \Pr(X_2 \oplus X_3 = 0) = 1/2 + \varepsilon_{2,3} ,$$

მაშინ თუ ეს წრფივი კომბინაციები არიან დამოუკიდებელინი, შეგვიძლია დავწეროთ, რომ

$$\Pr(X_1 \oplus X_3 = 0) = \Pr([X_1 \oplus X_2] \oplus [X_2 \oplus X_3] = 0) = 1/2 + 2\varepsilon_{1,2}\varepsilon_{2,3} .$$

ჩასმის შიფრებში ჩასმის ყველა თანრიგების ბულის ფუნქციების არაწრფივობა წარმოადგენს შიფრის კრიპტოანალიზისადმი მედეგობის აუცილებელ, მაგრამ არასაკმარის პირობას. ჩასმების კლასი, რომელიც წარმოადგენეს სუსტ კლასს წრფივი კრიპტოანალიზის მიმართ განისაზღვრება შემდეგი თეორემით (მოგვყავს დამტკიცების გარეშე).

<u>თეორემა 5.1.</u> *ჩასმის ყველა შესასვლელისა და გამოსასვლელის წრფივი ჯამი სრულდება ალბათობით 0 ან 1 მაშინ და მხოლოდ მაშინ, როდესაც ჩასმა აფინურად ექვივალენტურია ჩასმის, რომელიც შეიცავს აფინურ ბულის ფუნქციას.*

<u>5.13. მარტივი კრიპტოსისტემის მაგალითი.</u> წრფივი კრიპტოანალიზის მიდგომა განვიხილოთ გამარტივებული სიმეტრიული კრიპტოსისტემის მაგალითზე. ასეთი სისტემა შემოთავაზებული იყო თვით ჰ. ფეისტელის მიერ, როდესაც ის აგებდა თავის ცნობილ სქემას.. დასაშიფრი ტექსტი იყოფა თექვსმეტბიტიან ბლოკებად და ისე შედის სისტემაში.სულ გვაქვს ოთხი რაუნდი. თითოეულ რაუნდში ჯერ სრულდება შემოსული ტექსტის XOR-ით შეკრება რაუნდის ქვეგასაღებთან. ამის შემდეგ ჩანაცვლების ოპერაცია ოთხი ერთნაირი S ბლოკის საშუალებით (S ბლოკში შედის ოთხი ბიტი და გამოდის ასევე ოთხი ბიტი). ჩანაცვლება ოთხივე ბლოკში ხდება ერთი და იგივე ცხრილის საშუალებით (იხ. ცხრილი 5.4.).

ცხრილი 5.4.

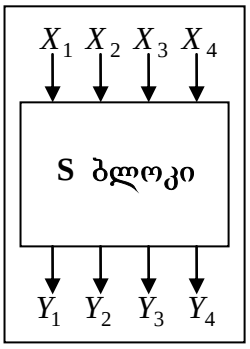
შესასვლელი	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
გამოსასვლელი	E	4	D	1	2	F	B	8	3	A	5	C	6	9	0	7

ცხრილში ჩაწერილია რიცხვები თექვსმეტობით სისტემაში. ჩანაცვლების ეს სტრიქონი წარმოადგენს **DES**-ის პირველი **S** ბლოკის პირველ სტრიქონს. **S** ბლოკებიდან გამოსული თექვსმეტი ბიტი (ისე, როგორც ეს ხდება **DES**-ში) გადანაცვლდება და მზადაა შემდეგ რაუნდში შესასვლელად. გადანაცვლება ხდება გადანაცვლების ცხრილის საშუალებით (იხ. ცხრილი 5.5).
ცხრილი 5.5

შესასვლელი	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
გამოსასვლელი	1	5	9	13	2	6	10	14	3	7	11	15	4	8	12	16

ამ გამარტივებულ ალგორითმში გვჭირდება რაუნდის ოთხი ქვეგასაღები. რათა არ გადვტვირთოთ ალგორითმი ზედმეტი დეტალებით, იგულისხმება, რომ ეს გასაღებები აიღება ერთმანეთისაგან დამოუკიდებლად.

5.14.S ბლოკის ანალიზი. იმისათვის, რომ შევძლოთ შეტევის დაწყება წრფივი კრიპტოანალიზის საშუალებით, ჩვენ უნდა ვიცოდეთ რამდენად მგრძნობიარეა წრფივი შეტევის მიმართ S ბლოკი. განვიხილოთ გამარტივებული მაგალითი. დავუშვათ, გვაქვს S ბლოკი, რომლის შესასვლელების და გამოსასვლელების რაოდენობაა ოთხი (იხ. სურ. 5.5).



სურ. 5.5 მარტივი S ბლოკი

დავუშვათ, წრფივი აპროქსიმაციის გამოსახულებას (5.1) აქვს შემდეგი სახე:

$$X_2 \oplus X_3 \oplus Y_1 \oplus Y_3 \oplus Y_4 = 0 \ ,$$

რაც ტოლფასია გამოსახულების

$$X_2 \oplus X_3 = Y_1 \oplus Y_3 \oplus Y_4 \ .$$

თუ განვიხილავთ თექვსმეტივე შესაძლო შესასვლელების და გამოსასვლელების კომბინაციას (იხ. ცხრილი 5.6), ვნახავთ, რომ ეს გამოსახულება იქნება სამართლიანი თორმეტ შემთხვევაში და აქედან გამომდინარე გადახრის ალბათობა იქნება $12/16 - 1/2 = 1/4$.
ამავე ცხრილიდან ჩანს, რომ

$$X_1 \oplus X_4 = Y_2$$

გამოსახულებისათვის ამ სიდიდის მნიშვნელობა ტოლია ნოლის, ხოლო

$$X_3 \oplus X_4 = Y_1 \oplus Y_4$$ -თვის კი - $-3/8$.

ცხრილი 5.6 **S** ბლოკის წრფივ აპროქსიმაციათა მაგალითი

X_1	X_2	X_3	X_4	Y_1	Y_2	Y_3	Y_4	$X_2 \oplus X_3$	$Y_1 \oplus Y_3 \oplus Y_4$	$X_1 \oplus X_4$	Y_2	$X_3 \oplus X_4$	$Y_1 \oplus Y_4$
0	0	0	0	1	1	1	0	0	0	0	1	0	1
0	0	0	1	0	1	0	0	0	0	1	1	1	0
0	0	1	0	1	1	0	1	1	0	0	1	1	0
0	0	1	1	0	0	0	1	1	1	1	0	0	1
0	1	0	0	0	0	1	0	1	1	0	0	0	0
0	1	0	1	1	1	1	1	1	1	1	1	1	0
0	1	1	0	1	0	1	1	0	1	0	0	1	0
0	1	1	1	1	0	0	0	0	1	1	0	0	1
1	0	0	0	0	0	1	1	0	0	1	0	0	1
1	0	0	1	1	0	1	0	0	0	0	0	1	1
1	0	1	0	0	1	1	0	1	1	1	1	1	0
1	0	1	1	1	1	0	0	1	1	0	1	0	1
1	1	0	0	0	1	0	1	1	1	1	1	0	1
1	1	0	1	1	0	0	1	1	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1	0	1	0
1	1	1	1	0	1	1	1	0	0	0	1	0	1

იმისათვის, რომ ვიპოვოთ ყველაზე უფრო კარგი წრფივი აპროქსიმაცია, საჭიროა ავაგოთ შესასვლელი და გამოსასვლელი ცვლადების ყველა შესაძლო წრფივი კომბინაციები, რომელთაგანაც შემდეგ ამოვირჩევთ საუკეთესოს. სრული ჩამონათვალი ჩვენი შიფრის **S** ბლოკის შესაძლო წრფივი აპროქსიმაციებისა გვაძლევს ცხრილს 5.7 ცხრილის შესასვლელი და გამოსასვლელი "ჯამები" წარმოადგენენ შესასვლელი და გამოსასვლელი ბიტების კომბინაციებს, ჩაწერილს თექვსმეტობით სისტემაში. მაგალითად, ცვლადების კომბინაცია

$$X_2 \oplus X_3 = 0 \cdot X_1 \oplus 1 \cdot X_2 \oplus 1 \cdot X_3 \oplus 0 \cdot X_4 \ .$$

აქ კოეფიციენტები გამოსახავენ **S** ბლოკში შესასვლელი ბიტების მნიშვნელობებს, რომელთა ჯამი, ჩაწერილი თექვსმეტობით სისტემაში ტოლია ექვსის. ასეთი სახით შესასვლელი და გამოსასვლელი "ჯამების" კომბინაციები გვაძლევენ ყველა შესაძლო წრფივ კომბინაციებს. 5.6. ცხრილიდან დაითვლება, თუ რამდენჯერ იყო მართალი ეს წრფივი აპროქსიმაცია (ეს ტოლობა) და 5.7. ცხრილში ამ კომბინაციების გადაკვეთაზე ზის რიცხვი, რომელიც უდრის ამ რიცხვს მინუს რვა. მაშინ ამ ცხრილიდან ადვილი დასათვლელია გადახრის კოეფიციენტი. მაგალითად წრფივი ტოლობის

$$X_3 \oplus X_4 = Y_1 \oplus Y_4$$

(თექვსმეტობითში შესასვლელი "ჯამი" – 3 და გამოსასვლელი "ჯამი" – 9. გადახრის კოეფიციენტი იქნება $-6/16 = -3/8$ და ალბათობა იმისა, რომ ეს წრფივი ტოლობა იქნება სამართლიანი, იქნება $1/2 - 3/8 = 1/8$.

5.15. წრფივი აპროქსიმაციის აგება მთელი შიფრისათვის. მას შემდეგ, რაც შევადგენთ ასეთ ცხრილებს ყველა ბლოკებისათვის, ჩვენ უკვე შეგვიძლია გადავიდეთ წრფივი აპროქსიმაციის განტოლების შედგენაზე მთელი შიფრისათვის. ეს შეიძლება გავაკეთოთ S ბლოკების შესაბამისი წრფივი აპროქსიმაციების კონკატაციით. წრფივი აპროქსიმაციის კონსტრუირებით, რომელშიც ჩართული იქნება ღია ტექსტის ბიტები და წინა რაუნდის გამოსასვლელი ბიტები, უკვე შესაძლებელი იქნება შევუტიოთ შიფრს და მივიღოთ გასაღების ბიტების ქვესიმრავლე, რომლებიც გამოიყენება ბოლო რაუნდში.

განვიხილოთ ეს პროცედურა ჩვენი მარტივი კრიპტოსისტემისათვის. სურ. 5.6 ნაჩვენებია ის **S** ბლოკები, რომელთა შეტევითაც, ჩვენ ვაპირებთ გავტეხოთ სისტემა. ამ ბლოკებისათვის ვიყენებთ წრფივი აპროქსიმაციის შემდეგ ტოლობებს:

ცხრილი 5.7

	გამოსასვლელი ჯამი															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	-2	-2	0	0	-2	+6	+2	+2	0	0	+2	+2	0	0
2	0	0	-2	-2	0	0	-2	-2	0	0	+2	+2	0	0	-6	+2
3	0	0	0	0	0	0	0	0	+2	-6	-2	-2	+2	+2	-2	-2
4	0	+2	0	-2	-2	-4	-2	0	0	-2	0	+2	+2	-4	+2	0
5	0	-2	-2	0	-2	0	+4	+2	-2	0	-4	+2	0	-2	-2	0
6	0	+2	-2	+4	+2	0	0	+2	0	-2	+2	+4	-2	0	0	-2
7	0	-2	0	+2	+2	-4	+2	0	-2	0	+2	0	+4	+2	0	+2
8	0	0	0	0	0	0	0	0	-2	+2	+2	-2	+2	-2	-2	-6
9	0	0	-2	-2	0	0	-2	-2	-4	0	-2	+2	0	+4	+2	-2
A	0	+4	-2	+2	-4	0	+2	-2	+2	+2	0	0	+2	+2	0	0
B	0	+4	0	-4	+4	0	+4	0	0	0	0	0	0	0	0	0
C	0	-2	+4	-2	-2	0	+2	0	+2	0	+2	+ 4	0	+2	0	-2
D	0	+2	+2	0	-2	+4	0	+2	-4	-2	+2	0	+2	0	0	+2
E	0	+2	+2	0	-2	-4	0	+2	-2	0	0	-2	-4	+2	-2	0
F	0	-2	-4	-2	-2	0	+2	0	0	-2	+4	-2	-2	0	+2	0

$S_{12} : X_1 \oplus X_3 \oplus X_4 = Y_2$ ალბათობით $3/4$ და გადახრით $+1/4$;

$S_{22} : X_2 = Y_2 \oplus Y_4$ ალბათობით $3/4$ და გადახრით $-1/4$;

$S_{32} : X_2 = Y_2 \oplus Y_4$ ალბათობით $3/4$ და გადახრით $-1/4$;

$S_{34} : X_2 = Y_2 \oplus Y_4$ ალბათობით $3/4$ და გადახრით $-1/4$.

აღვნიშნოთ U_i-ით თექვსმეტბიტიანი ბლოკი ბიტების i-ური რაუნდის S ბლოკის შესასვლელზე და $U_{i,j}$-ით ამ ბლოკის j-ური ბიტი. ბიტები გადანომრილია მარცხნიდან მარჯვნივ. შესაბამისად V_i და $V_{i,j}$ იქნება გამოსასვლელი ბიტები. K_i იყოს ქვეგასაღების ბლოკი, რომელიც იკრიბება j-ური რაუნდის შესასვლელთან და გამონაკლისის სახით K_5 იყოს გასაღები, რომელიც იკრიბება მეოთხე რაუნდის გამოსასვლელთან. მაშინ $U_1 = P \oplus K_1$ სადაც P არის ღია ტექსტის თექვსმეტი ბიტი.

გამოვიყენოთ პირველი რაუნდის წრფივი აპროქსიმაცია და მივიღებთ, რომ

$$V_{1,6} = U_{1,5} \oplus U_{1,7} \oplus U_{1,8} = (P_5 \oplus K_{1,5}) \oplus (P_7 \oplus K_{1,7}) \oplus (P_8 \oplus K_{1,8}) \qquad (5.2.)$$

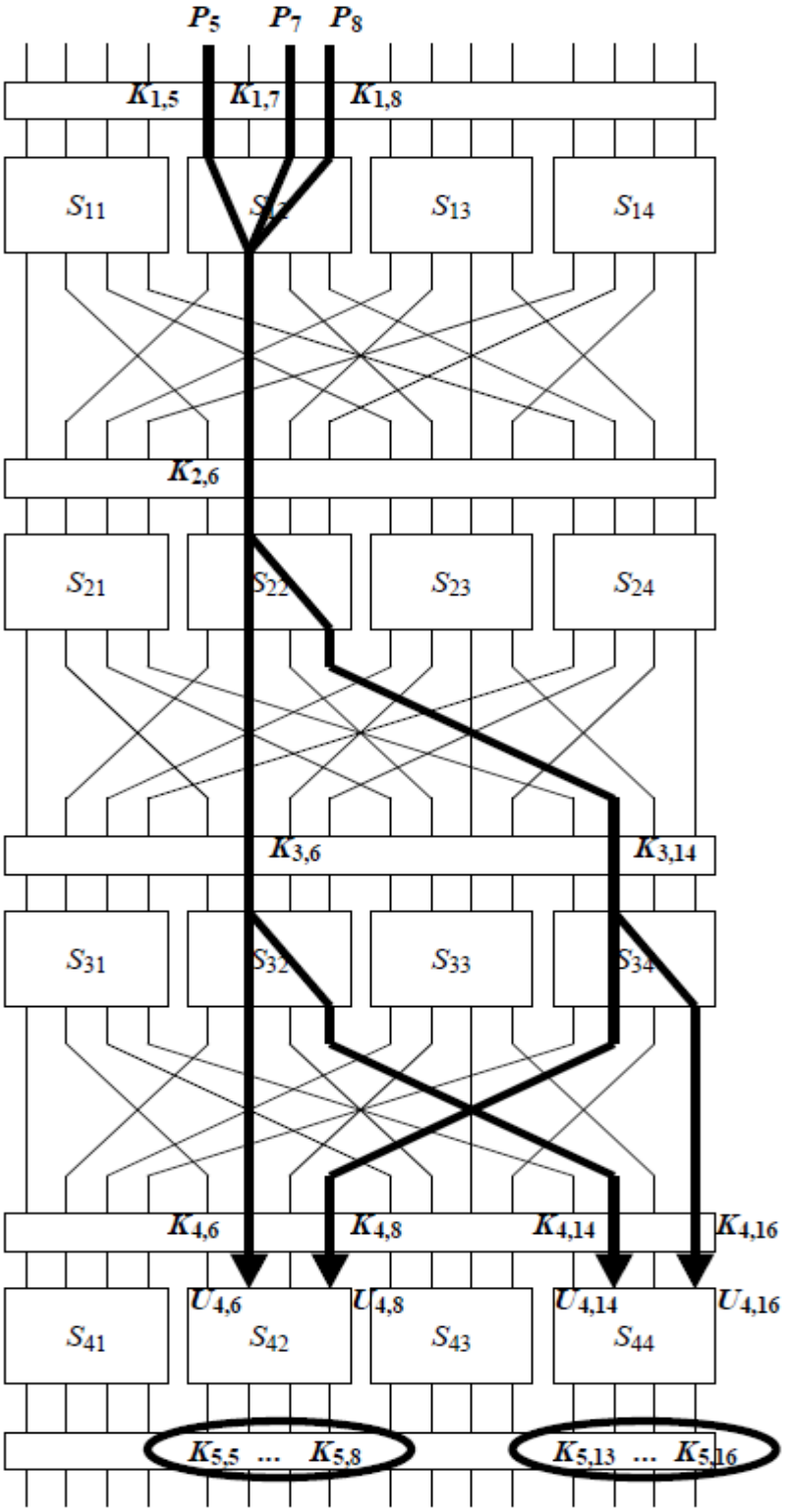
ალბათობით $3/4$. მეორე რაუნდის აპროქსიმაციიდან გვექნება

$$V_{2,6} \oplus V_{2,8} = U_{2,6}$$

ალბათობით $1/4$. რადგანაც $U_{2,6} = V_{1,6} \oplus K_{2,6}$ ალბათობით $1/4$, მივიღებთ, რომ

$$V_{2,6} \oplus V_{2,8} \oplus P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_8 \oplus K_{2,6} = 0 \qquad (5.3.)$$

ეს ტოლობა ლემის თანახმად, სამართლიანია ალბათობით $1/2 + 2(3/4 - 1/2)(1/4 - 1/2) = 3/8$ და გადახრით $-1/4$. ამ შემთხვევაში ჩვენ ვუშვებთ, რომ S ბლოკების აპროქსიმაციები არაა ერთმანეთზე დამოკიდებული, რაც რა თქმა უნდა არ შეესაბამება სინამდვილეს, მაგრამ პრაქტიკაში იძლევა საკმარისად კარგ შედეგებს შიფრების დიდი უმრავლესობისათვის.



სურ. 5.6 შეტევა წრფივი კრიპტოანალიზი საშუალებით.

მესამე რაუნდისათვის გვექნება:

$V_{3,6} \oplus V_{3,8} = U_{3,6}$ ალბათობით $1/4$ და

$V_{3,14} \oplus V_{3,16} = U_{3,24}$ ასევე ალბათობით $1/4$.

მაშინ, რადგანაც

$U_{3,6} = V_{2,6} \oplus K_{3,6}$ და $U_{3,14} = V_{2,8} \oplus K_{3,14}$

მივიღებთ, რომ

$$V_{3,6}+V_{3,8}\oplus V_{3,14}\oplus V_{3,16}\oplus P_5\oplus P_7\oplus P_8\oplus K_{1,5}\oplus K_{1,7}\oplus K_{1,8}\oplus K_{2,6}\oplus K_{3,6}\oplus K_{3,14}=0 \quad (5.4)$$

ალბათობით $5/8$ და გადახრით $1/8$.

(5.3.) და (5.4.) ტოლობებიდან მივიღებთ:

$$V_{3,6}\oplus V_{3,8}\oplus V_{3,14}\oplus V_{3,16}\oplus P_5\oplus P_7\oplus P_8\oplus K_{1,5}\oplus K_{1,7}\oplus K_{1,8}\oplus K_{2,6}\oplus K_{3,6}\oplus K_{3,14}=0.$$

თუ გავითვალისწინებთ, რომ

$$V_{3,6}=U_{4,6}\oplus K_{4,6}; V_{3,8}=U_{4,14}\oplus K_{4,14}; V_{3,14}=U_{4,8}\oplus K_{4,8}$$

და

$$V_{3,16}=U_{4,16}\oplus K_{4,6},$$

შეგვიძლია დავწეროთ

$$U_{4,6}\oplus U_{4,8}\oplus U_{4,14}\oplus U_{4,16}\oplus P_5\oplus P_7\oplus P_8\oplus \sum{}_K=0 . \quad (5.5.)$$

აქ $\sum{}_K=K_{1,5}\oplus K_{1.7}\oplus K_{1,8}\oplus K_{2,6}\oplus K_{3,6}\oplus K_{3,14}\oplus K_{4,6}\oplus K_{4,8}\oplus K_{4,14}\oplus K_{4,16}$.

(5.5.) გამოსახულების ალბათობა ლემის თანახმად გამოდის $15/32$, გადახრა კი – $-1/32$. გამოსახულება მოცემული გასაღებისათვის არის ფიქსირებულად ერთი ან ნოლი, ამიტომ შეგვიძლია ვთქვათ, რომ

$$U_{4,6}\oplus U_{4,8}\oplus U_{4,14}\oplus U_{4,16}\oplus P_5\oplus P_7\oplus P_8\oplus =0$$

უნდა სრულდებოდეს $15/32$ ან $17/32$ ალბათობით. ამგვარად, ჩვენ მივიღეთ სამი რაუნდის წრფივი აპროქსიმაციის გამოსახულება გადახრით .

5.16. გასაღების ბიტების გამოცნობა. მას შემდეგ რაც R რაუნდიანი შიფრისათვის ავაგეთ $R-1$ რაუნდის წრფივი აპროქსიმაცია შესაბამისი მაღალი ალბათობით წრფივი გადახრისაკენ, შეგვიძლია დავიწყოთ შეტევა ბოლო ქვეგასაღების ბიტების გამოსაცნობად. დაშიფრული ტექსტის **XOR**-ით შეკრება ბოლო ციკლის გასაღებთან გვაძლევს $U_{4,j}$. ამიტომ, ყველა შესაძლო გასაღებისა და მოცემული შიფროტექსტისათვის ვითვლით $U_{4,j}$ და შემდეგ ვამოწმებთ განტოლების (5.5)

$$U_{4,6}\oplus U_{4,8}\oplus U_{4,14}\oplus U_{4,16}\oplus P_5\oplus P_7\oplus P_8\oplus =0 \quad (5.5)$$

სამართლიანობას მოცემული შიფროგრამის შესაბამისი ღია ტექსტის საშუალებით. ვითვლით წრფივობისაკენ გადახრის ექსპერიმენტალურ მნიშვნელობას და ვადარებთ ჩვენს მიერ გამოთვლილ მნიშვნელობას. იმ გასაღებს, რომლისთვისაც ექსპერიმენტალურად გამოთვლილი წრფივობისაკენ გადხრის ალბათობა ყველაზე ახლოს იქნება ჩვენს მიერ გამოთვლილ ალბათობასთან, აქვს ყველაზე დიდი შანსი იყოს ჭეშმარიტი გასაღები.

ცხრილი 5.8

$K_{5,5}...K_{5,8}K_{5,13}...K_{5,16}$	გადახრის ალბათობა	$K_{5,5}...K_{5,8}K_{5,13}...K_{5,16}$	გადახრის ალბათობა
1C	0,0031	2A	0,0044
1D	0,0078	2B	0,0186
1E	0,0071	2C	0,0094
1F	0,0170	2D	0,0053
20	0,0025	2E	0,0062
21	0,0220	2F	0,0133
22	0,0211	30	0,0027
23	0,0064	31	0,0050
24	**0,0336**	32	0,0075
25	0,0106	33	0,0162
26	0,0096	34	0,0218
27	0,0074	35	0,0052
28	0,0224	36	0,0056
29	0,0054	37	0,0048

მაგალითისათვის განხილული იყო 10000 ღია და შესაბამისი შიფროტექსტი. წრფივობისაკენ გადახრის კოეფიციენტი (ცხრილი შემოკლებულად "გადახრის ალბათობა") გამოითვალებოდა შემდეგნაირად: დაითვლებოდა მოცემული გასაღებისთვის რამდენჯერ იყო სწორი შესამოწმებელი გამოსახულება, აკლდებოდა მას ხუთი ათასი და მიღებული შედეგი იყოფოდა ათი ათასზე. შედეგები მოყვანილია ცხრილში 5.8. როგორც ამ ცხრილიდან ჩანს, ყველაზე კარგი შედეგი მიიღება როდესაც გასაღები არის $[2,4]$ თექვსმეტობით წარმოდგენაში, რაც ბიტების დონეზე შეესაბამება გასაღებს.

როგორც მ. მაცუიმ აჩვენა, წრფივი კრიპტოანალიზით შეტევისათვის საჭირო ღია ტექსტებისა და შესაბამისი შიფროტექსტების წყვილების რაოდენობა შეიძლება დავადგინოთ მიახლოებითი ტოლობით $N \approx 1/\varepsilon^2$.

საკონტროლო კითხვები:

1. რაზეა დამოკიდებული ძალისმიერი შეტევისათვის საჭირო გამოთვლების რაოდენობა?
2. როგორი მეთოდებით შეგვიძლია შევამციროთ გადარჩევის რაოდენობა, რომელიც საჭიროა ძალისმიერი შეტევის დროს?
3. აღწერეთ შეტევა "დროისა და მეხსიერების კომპრომისით".
4. როდი იქნება ყველაზე ეფექტური "შუაში შეხვედრის ალგორითმი"?
5. როგორი ტიპის შეტევას მიეკუთვნება დიფერენციალური კრიპტოანალიზი?
6. რა სიდიდეს ეწოდება დიფერენციალი?
7. რომელი ოპერაციის დროს იცვლება დიფერენციალი DEშ-ში?
8. როგორ ხდება გასაღების ბიტების გამოცნობა?
9. რა იდეა უდევს საფუძვლად წრფივ კრიპტოანალიზს?
10. რას ეწოდება ბულის ფუნქციის არაწრფივობა?
11. როგორ ხდება მოცემული ფუნქციის წრფივი აპროქსიმაცია?
12. როგორ ხდება გასაღების ბიტების გამოცნობა წრფივ კრიპტოანალიზის დროს?

კომპიუტერულ ქსელებში გადაცემული ინფორმაციის მთლიანობის უზრუნველყოფის ტექნოლოგიები

6.1. თანამედროვე კრიპტოგრაფიის ძირითადი მიზნები. აქამდე ჩვენ მივყვებოდით რა კრიპტოგრაფიის ტრადიციულ გაგებას, კრიპტოსისტემებს ვიხილავდით და ვაფასებდით მხოლოდ იქიდან გამომდინარე, თუ როგორ იცავენ ეს სისტემები ინფორმაციის კონფიდენციალურობას, მაგრამ ცნობილია, რომ საინფორმაციო სისტემებზე შეტევა შეიძლება განხორციელდეს არა მარტო ინფორმაციის მიტაცების მიზნით. ძალიან ხშირად მოწინააღმდეგის მიზანია, შეცვალოს გადაცემული ინფორმაცია და ინფორმაციული ურთიერთობის ობიექტები შეიყვანოს შეცდომაში. შესაძლებელია ისეთი შემთხვევებიც, როდესაც მოწინააღმდეგე ცდილობს უბრალოდ არ დაუშვას ინფორმაციის გაცვლა ობიექტებს შორის და ამ მიზნით ანადგურებს გადაცემულ ინფორმაციას.

კომპიუტერული ქსელების განვითარებამ დაგვანახა, რომ თანამედროვე კრიპტოგრაფიის წინაშე დგას ოთხი ძირითადი ამოცანა:

- კონფიდენციალურობა;
- ინფორმაციის მთლიანობა;
- იდენტიფიკაცია და აუთენტიფიკაცია;
- ავტორობის უარყოფის შეუძლებლობა.

კონფიდენციალობის ქვეშ იგულისხმება ის, რასაც ადრე უწოდებდნენ საიდუმლოებას. ორივე შემთხვევაში იგულისხმება ინფორმაციის გადაცემა, ან შენახვა ისე, რომ უფლებამოსილი

პირების გარდა ვერავინ შეძლოს ამ ინფორმაციის წაკითხვა. საუკუნეების განმავლობაში სწორედ ეს პრობლემა წარმოადგენდა კრიპტოგრაფიის მთავარ, ძირითად ამოცანას. მიაჩნდათ, რომ დანარჩენი ამოცანები იყო დამოკიდებული ამ მთავარ ამოცანაზე და მათი გადაჭრა მოხდებოდა ამ ძირითადი ამოცანის გადაჭრასთან ერთად. ასე მაგალითად, 1986 წელს ცნობილი კრიპტოგრაფი, ახალი, ღია გასაღებიანი კრიპტოგრაფიის ერთერთი ფუძემდებელი, მ. ჰელმანი თავის ერთერთ სტატიაში წერდა: "აშკარაა, რომ კონფიდენციალობის და აუთენტიფიკაციის პრობლემები მჭიდროდაა დაკავშირებული ერთმანეთთან." მაგრამ საკომუნიკაციო ქსელების განვითარებამ და "მსოფლიო აბლაბუდას"- ინტერნეტის შექმნამ აშკარად დაგვანახა, რომ ეს ორი პრობლემა წარმოადგენს დამოუკიდებელ ამოცანებს და საჭიროა მათი გადაჭრისათვის სხვადასხვა მეთოდების გამოყენება.

ძალიან ხშირად გვხვდება სიტუაციები, როდესაც ჩვენთვის არა იმდენად მთავარია ინფორმაციის კონფიდენციალურობის დაცვა, რამდენადაც იმის ცოდნა, მოაღწია თუ არა ჩვენამდე ინფორმაციამ შეუცვლელი სახით. მართლაც ბოლოს და ბოლოს ინტერნეტი შეიქმნა არა ინფორმაციის დასამალად, არამედ ინფორმაციის გასაცვლელად ადამიანებს შორის. ამიტომაც ამ შემთხვევაში მთავარია ინფორმაციის მთლიანობის პრობლემა, დამახინჯდა თუ არა ინფორმაცია (უნებლიედ თუ წინასწარი განზრახვით) ქსელში გადაცემის დროს და არა კონფიდენციალობის პრობლემა.

ასევე შესაძლებელია სიტუაცია, როდესაც შეტყობინებას თქვენი პარტნიორის სახელით აგზავნის სრულიად სხვა პირი, ანუ ხდება იმიტაცია. ინფორმაციის მთლიანობისა და იმიტაციისაგან თავის დასაცავად საჭიროა გადაიჭრას ინფორმაციისა და ავტორობის იდენტიფიკაციისა და აუთენტიფიკაციის პრობლემა, რომელიც შეიძლება სულაც არ იყოს დაკავშირებული კონფიდენციალობის პრობლემასთან.

ასევე ადვილი შესაძლებელია, რომ შეტყობინება გამოგიგზავნოთ ნამდვილად თქვენმა საქმიანმა პარტნიორმა, მაგრამ მეორე დღეს მან უარყოს ამ შეტყობინების ავტორობა. არც ეს მომენტი იქნება თქვენთვის სასიამოვნო, ამიტომ უნდა შეგეძლოთ დაუმტკიცოთ თქვენს პარტნიორს, რომ წერილი მის მიერ იყო გამოგზავნილი. ანუ შეტყობინების ავტორს ვერ უნდა შეეძლოს უარყოს თავისი ავტორობა.

ძალიან ხშირად საჭიროა ამ ოთხი ძირითადი ამოცანისაგან შედგენილი კომბინაციების გადაწყვეტა. მაგალითად, მთლიანობის, აუთენტიფიკაციის და ვერ უარყოფის ერთდროულად გადაჭრა. ამიტომ თანამედროვე კრიპტოგრაფიაში ვითარდება მეთოდები, რომლებიც იძლევიან საშუალებას გადავწყვიტოთ ეს ამოცანები როგორც ცალ-ცალკე, ასევე კომბინირებულად.

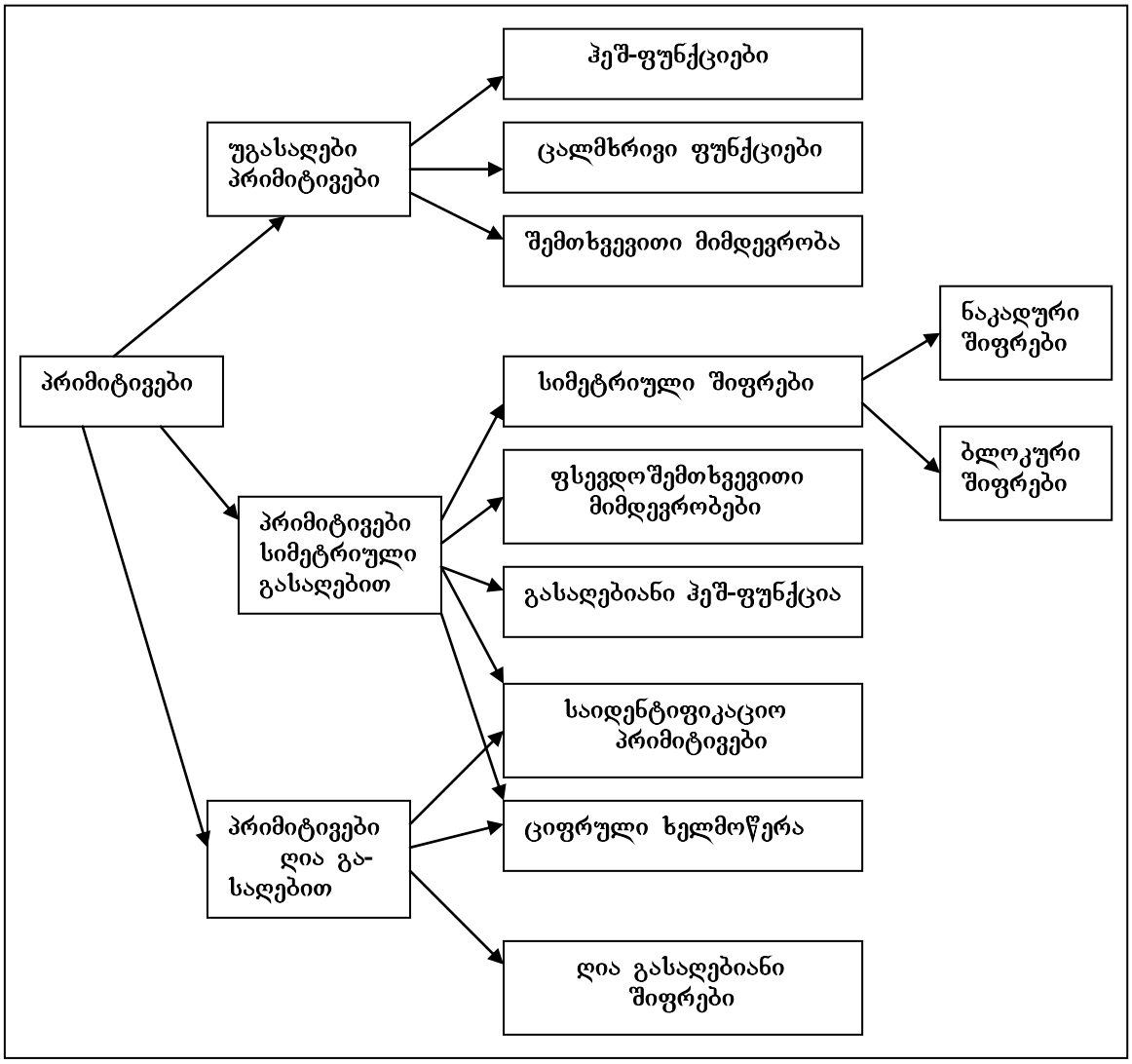
<u>6.2. კრიპტოგრაფიული პრიმიტივები.</u> იმისათვის, რომ გადავჭრათ კრიპტოგრაფიის წინაშე მდგომი ამოცანები, საჭიროა სხვადასხვა კრიპტოგრაფიული მეთოდების საშუალებით ავაგოთ სისტემა, რომელიც შეძლებს მოცემული კონკრეტული ამოცანის გადაჭრას. ამ კრიპტოგრაფიულ მეთოდებს უწოდებენ კრიპტოგრაფიულ ინსტრუმენტებს ანუ კრიპტოგრაფიულ პრიმიტივებს. სურ. 6.1. მოყვანილია კრიპტოგრაფიული პრიმიტივების იერარქიული სქემა. როგორც ამ სქემიდან ვხედავთ, ჩვენ ჯერჯერობით შევისწავლეთ მხოლოდ ერთი ნაწილი კრიპტოგრაფიული პრიმიტივებისა, ძირითადად სიმეტრიული გასაღებიანი პრიმიტივები.

პრიმიტივები, რომლებიც გამოიყენება კრიპტოგრაფიაში, უნდა აკმაყოფილებდნენ გარკვეულ კრიტერიუმებს, რათა უზრუველყოფილი იყოს ინფორმაციის დაცვის სასურველი დონე. მოკლედ ჩამოვთვალოთ ძირითადი კრიტერიუმები:

1. **უსაფრთხოების დონე.** მიუხედავად იმისა, რომ ეს ერთერთი ყველაზე მნიშვნელოვანი კრიტერიუმია, მისი რაოდენობრივი შეფასება რთულია. ყველზე ხშირად ის განისაზღვრება იმ ოპერაციების ტერმინებში, რომლებიც საჭიროა ობიექტის უსაფრთხოების გასატეხად. ჩვეულებრივ ესაა იმ სამუშაოთა მოცულობის ზედა საზღვარი, რომელიც უნდა ჩაატაროს მოწინააღმდეგემ კრიპტოგრაფიული პრიმიტივის გასატეხად.
2. **ფუნქციონალურობა**. შესაძლებელი უნდა იყოს მოცემული პრიმიტივის კომბინირება სხვა პრიმიტივებთან ობიექტის უსაფრთხოების დასაცავად. რომელი პრიმიტივების გამოყენებაა უკეთესი მოცემულ კონკრეტულ შემთხვევაში განისაზღვრება პრიმიტივების თვისებებიდან გამომდინარე.
3. **ოპერაციული მეთოდები.** ერთი და იგივე პრიმიტივები სხვადასხვა სიტუაციებში ამჟღავნებენ სხვადასხვა თვისებებს, ამიტომ პრიმიტივის ფუნქციონალურობა დამოკიდებულია ოპერაციათა რეჟიმზე.
4. **წარმატებულობა.** ეს კრიტერიუმი განისაზღვრება პრიმიტივის ეფექტიანობით მოცემულ კონკრეტულ რეჟიმში. მაგალითად, დაშიფრვის ალგორითმის წარმატებულობა შეიძლება

შეფასდეს მისი სიჩქარით, ანუ რამდენი ბიტი ინფორმაციის დაშიფრვა შეუძლია მას ერთ წამში მოცემულ რეჟიმში.

5. **ადვილი იმპლემენტაცია.** ეს კრიტერიუმი ფასდება იმ სირთულეების მიხედვით, რომლებიც წარმოიშვება პრიმიტივის პრაქტიკული იმპლემენტაციის დროს უსაფრთხოების სისტემაში. გათვალისწინებული უნდა იყოს სირთულეები როგორც ტექნიკური (**hardware**), ასევე პროგრამული (**software**) იმპლემენტაციების დროს.



სურ. 6.1 კრიპტოგრაფიული პრიმიტივების სისტემატიკა.

6.3.ინფორმაციის მთლიანობის დაცვის ტექნოლოგიები. მეორე თავში ჩვენ ჩამოვაყალიბეთ ქსელში გადაცემული ინფორმაციის მთლიანობის ამოცანა. ვნახოთ, როგორ შეიძლება გადავჭრათ ეს ამოცანა კრიპტოგრაფიული პრიმიტივების საშუალებით.

წარმოვიდგინოთ ასეთი სიტუაცია: ანი და ბექა ინფორმაციის გასაცვლელად სარგებლობენ ღია არხით, რომელსაც მთლიანად აკონტროლებს ეკა და მას შეუძლია არა მარტო მიიტაცოს ინფორმაცია (რომელიც შეიძლება ღიაა), არამედ განახორციელოს აქტიური შეტევაც, რაც გამოიხატება იმაში, რომ ეკას აქვს საშუალება

- დაბლოკოს ანის მიერ გადაცემული ინფორმაცია და არ მიუშვას ის ბექამდე;
- შეუძლია მთლიანად ან ნაწილობრივ შეცვალოს გადაცემული ინფორმაცია;
- შეუძლია ანის სახელით გაუგზავნოს ბექას ინფორმაცია, რომელიც ანის არ გადაუცია.
- შეუძლია შეცვალოს გადაცემულ შეტყობინებათა მიმდევრობა;
- შეუძლია დააყოვნოს შეტყობინება არხში და გადასცეს მოგვიანებით.

პირველი ვარიანტი ეკას არ უნდა აწყობდეს, რადგანაც ანი და ბექა პირველივე დაბლოკი გადაცემის შემდეგ აუცილებლად შეცვლიან არხს და ეკა ვეღარ შეძლებს ხელი შეუშალოს მათ. მეოთხე და მეხუთე ვარიანტების წინააღმდეგ საკმარისია ანიმ და ბექამ გამოიყენონ დროითი ჭდე, რათა აღადგინონ შეტყობინებათა ნორმალური მიმდევრობა და გადაცემის დრო. ამიტომ ყველაზე რთულ ვარიანტებად რჩება მხოლოდ მეორე და მესამე, რომლებიც წარმოადგენენ ინფორმაციის შეცვლისა და იმიტაციის შეტევებს. არსებობს ამ შეტევებისაგან თავის დაცვის სხვადასხვა საშუალებები, რომლებიც უზრუნველყოფენ დაცვის სხვადასხვა დონეს. ამისთვის, როგორც უკვე აღნიშნული იყო მეორე თავში, შესაძლებელია გამოვიყენოთ სამი განსხვავებული მიდგომა, რომლებიც უზრუნველყოფენ უსაფრთხოების სხვადასხვა დონეს და შესაბამისად მოითხოვენ სხვადასხვა სირთულის ალგორითმებს პრაქტიკული რეალიზაციისათვის. ყველა მათგანის დამახასიათებელი თვისებაა, რომ სხვადასხვა პროცედურების საშუალებით გამოითვლება სპეციალური სიდიდე, რომელიც გადაიცემა ღია არხით ტექსტთან ერთად. ამ სიდიდეს ეწოდება კოდი (ან ჭდე) და მან უნდა დაადასტუროს ინფორმაციის მთლიანობა.

6.4. A-კოდები. პირველ რიგში უნდა აღვნიშნოთ ინფორმაციის თეორიაზე დაფუძნებული მიდგომა, რომელიც უზრუნველყოფს ინფორმაციის მთლიანობის აუთენტიფიკაციის პროცედურას, რომელიც უპირობოდ მედეგია ნებისმიერი შეტევის მიმართ, რა გამოთვლითი საშუალებებიც არ უნდა გააჩნდეს მოწინააღმდეგეს.

აუთენტიფიკაციის პროცედურის უსაფრთხოება შეიძლება ემყარებოდეს სირთულის თეორიას. ასეთ შემთხვევაში იგულისხმება, რომ პროცედურის უსაფრთხოება ტოლფასია რომელიმე **NP** კლასის ამოცანის სირთულის და მოწინააღმდეგეს არ გააჩნია საჭირო გამოთვლითი საშუალებები ასეთი ამოცანის ამოსახსნელად.

ყველაზე ხშირად კი გამოიყენება რომელიმე სიმეტრიულ კრიპტოგრაფიულ სისტემაზე დაფუძნებული აუთენტიფიკაციის პროცედურა, რომლის უსაფრთხოება გათვლილია იმაზე, რომ მოწინააღმდეგეს გააჩნია შეზღუდული გამოთვლითი საშუალებები, რაც არ აძლევს მას საშუალებას გატეხოს სისტემა, რომლის კრიპტომედეგობა მთლიანადაა განსაზღვრული გამოყენებული შიფრით.

განსაზღვრა 6.1. *A-კოდი ეწოდება* (P, A, K, E) *ოთხეულს, რომლისთვისაც სრულდება შემდეგი პირობები:*

1. P *არის ღია ტექსტების სასრული სიმრავლე.*
2. A *არის შეტყობინებათა კოდების სასრული სიმრავლე.*
3. K *არის შესაძლო გასაღებების სასრული სიმრავლე.*
4. *თითოეული* k *გასაღებისათვის* $(k \in K)$ *არსებობს აუთენტიფიკაციის წესი* $e_k : P \to A$, *რომლის საშუალებითაც ყოველი შეტყობინებისათვის გამოითვლება უნიკალური კოდი.*

გადასაცემი ტექსტების (შეტყობინებათა) სიმრავლე ამ შემთხვევაში განისაზღვრება როგორც $M = P \times A$. იმისათვის, რომ ანიმ და ბექამ გამოიყენონ A-კოდი ინფორმაციის მთლიანობის აუთენტიფიკაციისათვის, საჭიროა მათ შეასრულონ შემდეგი სახის პროტოკოლი. პირველ რიგში ისინი დახურული არხის საშუალებით ირჩევენ საერთო გასაღებს $(k \in K)$. ეს ხდება ზუსტად ისე, როგორც სიმეტრიული ალგორითმების გამოყენების დროს. ამის შემდეგ, თუ ანის სურს ღია არხის საშუალებით გაუგზავნოს ბექას $m \in M$ შეტყობინება, რომელიც შეიცავს p ტექსტს, ანიმ უნდა გამოთვალოს გამოთვლის $a = e_k(p)$ აუთენტიფიკაციის კოდი $m = (p, a)$ წყვილი და გაუგზავნოს ბექას. თავის მხრივ, ბექა მიიღებს რა m' შეტყობინებას, გამოთვლის $a' = e_k(p)$ და შეადარებს მიღებულს. თუ შესრულდება პირობა $a' = a$, ეს ნიშნავს, რომ შეტყობინების აუთენტიფიკაცია მოხდა. თუ პირობა არ სრულდება, ეს იმას ნიშნავს, რომ ეკამ მოახდინა ან ინფორმაციის შეცვლა, ან იმიტაცია (აქ ჩვენ არ განვიხილავთ ტრივიალურ შემთხვევას, როდესაც გადაცემის დროს დაშვებული იქნა შეცდომა).

6.5. მოტყუების ალბათობები. როგორც ზევით აღვნიშნეთ, ღია არხს აკონტროლებს ეკა, რომლის მიზანია გაუგზავნოს ბექას ან შეცვლილი ინფორმაცია (ამას ეწოდება შეტევა ჩანაცვლებით), ან დაუკავშირდეს ანის სახელით და გადასცეს ინფორმაცია, რომელიც არ შეუქმნია ანის (ამას ეწოდება გადაცემის იმიტაცია ან უბრალოდ იმიტაცია). შევხედოთ ეხლა ამ ყველაფერს ეკას მხრიდან და ვნახოთ, რა შანსები აქვს მას რომ მოატყუოს ბექა, ანუ დაარწმუნოს ის, რომ ინფორმაცია

ნამდვილად ანიმ გამოაგზავნა. იმიტაციის დროს ეკა ქმნის $m' \in M$ შეტყობინებას ღია ტექსტისა და კოდის საშუალებით (ცხადია, რომ ეკასთვის აუთენტიფიკაციის პროტოკოლისა და ალგორითმის შესახებ ყველაფერი ცნობილია კონკრეტული გასაღების გარდა) და უგზავნის ბექას. არსებობს რაღაც ალბათობა იმისა, რომ ეკა შეძლებს ბექას მოტყუებას. აღვნიშნოთ ეს ალბათობები იმიტაციის შემთხვევაში $\Pr(i)$ და შეცვლის შემთხვევაში $\Pr(s)$-ით. ცხადია, რომ ამ ორი სტრატეგიიდან ეკა ამოირჩევს იმას, რომლის ალბათობაც იქნება უფრო მეტი. აღვნიშნოთ ეკას წარმატების ალბათობა $\Pr(d)$, რომელიც გამოითვლება როგორც $\Pr(d) = \max(\Pr(i), \Pr(s))$.

გასაღები	P		
	0	1	2
(0,0)	0	0	0
(0,1)	1	1	1
(0,2)	2	2	2
(1,0)	0	1	2
(1,1)	1	2	0
(1,2)	2	0	1
(2,0)	0	2	1
(2,1)	1	0	2
(2,2)	2	1	0

სურ. 6.2 აუთენტიფიკაციის მატრიცა

განვიხილოთ მარტივი მაგალითი. დავუშვათ, რომ სულ გვაქვს სამი ღია ტექსტი და სამი კოდი $P = A = \{0,1,2\}$, მაშინ გასაღებების სიმრავლე ტოლი იქნება

$$K = \{(0,0),(0,1),(0,2),(1,0),(1,1),(1,2),(2,0),(2,1),(2,2)\}.$$

ყოველი $(i, j) \in K$ და ყოველი $p \in P$ შეტყობინებისათვის განვსაზღვროთ კოდის გამოსათვლელი ფუნქცია შემდეგი სახით: $e_{ij}(p) = ip + j (\mathrm{mod}\, 3)$. მივიღებთ შემდეგი სახის აუთენტიფიკაციის მატრიცას (იხ. სურ. 6.2). ჯერჯერობით, ჩავთვალოთ, რომ გასაღებების ამორჩევა ხდება თანაბარალბათურად, ანუ $\Pr(k) = 1/9$ (ამ მაგალითში არ დავაზუსტებთ ღია ტექსტების განაწილების ფუნქციას $\Pr(P)$, რადგანაც ეს არაა არსებითი).

განვიხილოთ იმიტაციის შემთხვევა. ეკა ამოირჩევს სასურველ p-ს და ცდილობს გამოიცნოს შესაბამისი a. დავუშვათ, გასაღები, რომელსაც იყენებენ ანი და ბექა და რომელიც არ იცის ეკამ არის k_{12} და შეტყობინება, რომელიც ამოირჩია ეკამ $p = 2$. მაშინ აუთენტიფიკაციის კოდი იქნება $a = 1$. თუ დავუკვირდებით აუთენტიფიკაციის მატრიცას, ადვილად შევამჩნევთ, რომ ყოველ სვეტში გვაქვს მხოლოდ სამი ერთიანი ცხრა ელემენტიდან, ანუ აუთენტიფიკაციის კოდების მნიშვნელობები თანაბრადაა განაწილებული, რაც იმას ნიშნავს, რომ $\Pr(i) = 1/3$.

შედარებით უფრო რთულია $\Pr(s)$-ს ალბათობის გამოთვლა. დავუშვათ, ეკამ არხიდან მიიტაცა შეტყობინება $(0,0)$. ეს მას აძლევს გარკვეულ ინფორმაციას გასაღების შესახებ, კერძოდ, როგორც ცხრილიდან ჩანს, გასაღები შეიძლება იყოს მხოლოდ

$$K_0 = \{(0,0),(1,0),(2,0)\}$$

სიმრავლიდან. თუ ეხლა ეკა შეეცდება შეცვალოს შეტყობინება სხვა შეტყობინებით, დავუშვათ, $(1,1)$-ით, მაშინ ის მიაღწევს წარმატებას მხოლოდ იმ შემთხვევაში, როდესაც ანის და ბექას მიერ შერჩეული გასაღები იქნება $K_0 = (1,0)$. ალბათობა იმისა, რომ ეკა აირჩევს გასაღებს $K_0 = (1,0)$ კვლავ იქნება $1/3$. მოკლედ აღვწეროთ სქემა, რომლითაც გამოვთვალეთ ჩანაცვლების ალბათობა.

ეკამ მიიღო შეტყობინება (p,a). ამიტომ გასაღებების რაოდენობა შეიზღუდა ცხრიდან სამამდე, ამიტომ ნებისმიერი (p',a') შეტყობინებისათვის, რომელსაც ეკა გაუგზავნის ბექას, არსებობს მხოლოდ ერთი გასაღები დარჩენილი სამიდან, რომელიც მართლაც გვაძლევს $e_k(p')=a'$ და რამაც შეიძლება მოატყუოს ბექა.

გამოვთვალოთ ეხლა ეს ალბათობები ზოგად შემთხვევაში. დავიწყოთ $\Pr(i)$-ს გამოთვლით. თუ ანის და ბექას მიერ შერჩეული საიდუმლო გასაღებია k_0, მაშინ განვსაზღვროთ ნებისმიერი $p\in P$ და $a\in A$ წყვილისათვის ფუნქცია $f(p,a)$, როგორც ალბათობა იმისა, რომ მოხდება ბექას მიერ მიიღებული (p,a) შეტყობინების აუთენტიფიკაცია, ანუ ეკას მიერ გამოგზავნილი a კოდის მნიშვნელობა დაემთხვევა ბექას მიერ გამოთვლილი a' კოდის მნიშვნელობას. ადვილი დასანახია, რომ

$$f(p,a)=\Pr(a=e_{k_0}(p))=\sum_{\{k\in K:e_k(p)=a\}}\Pr(k)$$

ეს ნიშნავს, რომ ფუნქციის გამოსათვლელად აუთენტიფიკაციის მატრიცის p სვეტში უნდა შეირჩეს ის სტრიქონები, რომლებშიც დგას a კოდი და შეიკრიბოს შესაბამისი გასაღებების ალბათობები.

თავისთავად ცხადია, რომ ეკა ცდილობს გაზარდოს თავისი წარმატების შანსი, ამიტომ ის შეეცდება მიიღოს $f(p,a)$ ფუნქციის მაქსიმუმი. ამიტომ

$$\Pr(i)=\max\{f(p,a):p\in P,a\in A\}.$$

შევნიშნოთ, რომ $\Pr(i)$-ს გამოთვლაში არ გვჭირდება $\Pr(P)$ ალბათობების განაწილება.

$\Pr(s)$-ს გამოთვლა შედარებით უფრო რთულია და მასში მონაწილეობს $\Pr(P)$ განაწილებაც. ჯერ განვიხილოთ შემთხვევა, როდესაც ეკამ მიიტაცა ანის გადაცემული შეტყობინება (p,a) და შეცვალა ის შეტყობინებით (p',a'), სადაც $p\neq p'$. ასეთი შეცვლის შემთხვევაში, ეკა შეძლებს ბექას მოატყუებას თუ $a'=e_{k_0}(p')$. $g(p',a';p,a)$ ფუნქცია იყოს ალბათობა იმის, რომ ეკას მიერ შეტყობინების შეცვლა იქნება წარმატებული. მაშინ $g(p',a';p,a)$ ფუნქცია გამოითვლება შემდეგი სახით:

$$g(p',a';p,a)=\Pr(a'=e_{k_0}(p')/a=e_{k_0}(p))=$$

$$\frac{\Pr(a'=e_{k_0}(p')\wedge a=e_{k_0}(p))}{\Pr(a=e_{k_0}(p)}=$$

$$=\frac{\sum_{\{k\in K:a=e_{k_0}(p),a'=e_{k_0}(p')\}}\Pr(k)}{\sum_{\{k\in K:a=e_{k_0}(p)\}}\Pr(k)}=\frac{\sum_{\{k\in K:a=e_{k_0}(p),a'=e_{k_0}(p')\}}\Pr(k)}{f(p,a)}$$

ეკა ცდილობს მიაღწიოს ამ ფუნქციის მაქსიმუმს

$$\Pr(m,m')=\max\{g(p,a;p',a')\ :p\neq p',a'\in A\}$$

სადაც $\Pr(m,m')$ ალბათობა იმისა, რომ ეკა წარმატებით შეატყუებს ბექას m' შეტყობინებას, როდესაც გადაცემული იყო m შეტყობინება. შემოვიტანოთ $\Pr_M(p,a)$ შეტყობინებათა განაწილების ფუნქცია, რომელიც ტოლი იქნება

$$\Pr_M(p,a)=\Pr(P=p)\times\Pr_k(a/p)=$$
$$=\Pr(P=p)\times\sum_{\{k\in K:e_k(p)=a\}}\Pr_K(k)=$$
$$=\Pr(P=p)\times f(p,a)$$

მაშინ შეგვიძლია გამოვთვალოთ

$$\Pr(s) = \sum \Pr_M(p,a) \times \Pr(m,m') \ .$$

6.6. მოტყუების ალბათობების ქვედა საზღვარი. ამ გამოთვლებიდან აშკარა გახდა, რომ A-კოდების საიმედოობა იზომება მოტყუების (ეკას მიერ ბექას შეცდომაში შეყვანის) ალბათობით. აქედან გამომდინარე, უნდა მოხდეს ამ კოდების ისეთნაირად კონსტრუირება, რომ ეს ალბათობები იყოს ძალიან მცირე, მაგრამ ამავე დროს აუცილებლად გათვალისწინებული უნდა იყოს სხვა მოთხოვნებიც, რომლებსაც უნდა აკმაყოფილებდნენ A-კოდები. კერძოდ, ღია ტექსტების სიმრავლე უნდა იყოს საკმარისად დიდი, რათა შესაძლებელი იყოს სასურველი ინფორმაციის გადაცემა და შესაძლებელი უნდა იყოს თითოეულ ინფორმაციას მოვდოთ თავისი, უნიკალური კოდი. ამავე დროს გასაღებების სიმრავლე უნდა იყოს მინიმალური, რადგანაც გასაღების მნიშვნელობა გადაიცემა დახურული არხით, ხოლო თვითონ გასაღები იცვლება ყოველი გადაცემული შეტყობინების შემდეგ, როგორც ეს ხდება ერთჯერადი ბლოკნოტის გამოყენების დროს.

დავუშვათ, მოცემული გვაქვს A-კოდი - (P, A, K, E), სადაც $|A| = n$. დაუმტკიცებლად მოვიყვანოთ მოტყუების ალბათობების ქვედა საზღვრები: $\Pr(i) \geq 1/n$ ტოლობა შესრულდება მაშინ და მხოლოდ მაშინ, როდესაც

$$\sum_{\{k \in K : e_k(p) = a\}} \Pr_K(k) = 1/n$$

ნებისმიერი $p \in P$ და $a \in A$ ელემენტებისათვის.

ანალოგიურად, $\Pr(s) \geq 1/n$ და ტოლობა შესრულდება მაშინ და მხოლოდ მაშინ, როდესაც

$$\frac{\sum_{\{k \in K : e_k(p) = a, e_k(p') = a'\}} \Pr_K(k)}{\sum_{\{k \in K : e_k(p) = a\}} \Pr_K(k)} = \frac{1}{n}$$

ნებისმიერი $p, p' \in P, \ p \neq p'$ და $a, a' \in A$.

$\Pr(i) = \Pr(s) = 1/n$ მაშინ და მხოლოდ მაშინ, როდესაც

$$\sum_{\{k \in K : e_k(p) = a, e_k(p') = a'\}} \Pr_K(k) = 1/n^2 \ .$$

იგივე საზღვრები ენტროპიის გამოყენებით ჩაიწერება შემდეგი სახით:

$$\log \Pr(i) \geq H(K/M) - H(K)$$

$$\log \Pr(s) \geq H(K/M^2) - HK/M) \ .$$

მოვიყვანოთ ეხლა სრულყოფილი A-კოდის განმარტება.

განსაზღვრა 6.2. A-*კოდს ეწოდება აუთენტიფიკაციის სრულყოფილი კოდი, თუ სრულდება პირობა* $\Pr(i) = \Pr(s) = 1/n$.

6.7. აუთენტიფიკაციის კოდები და ორთოგონალური მასივი. შემოვიტანოთ ორთოგონალური მასივის ცნება.

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 0 & 2 \\ 2 & 1 & 9 \end{pmatrix}$$

სურ. 6.3 ორთოგონალური მასივი

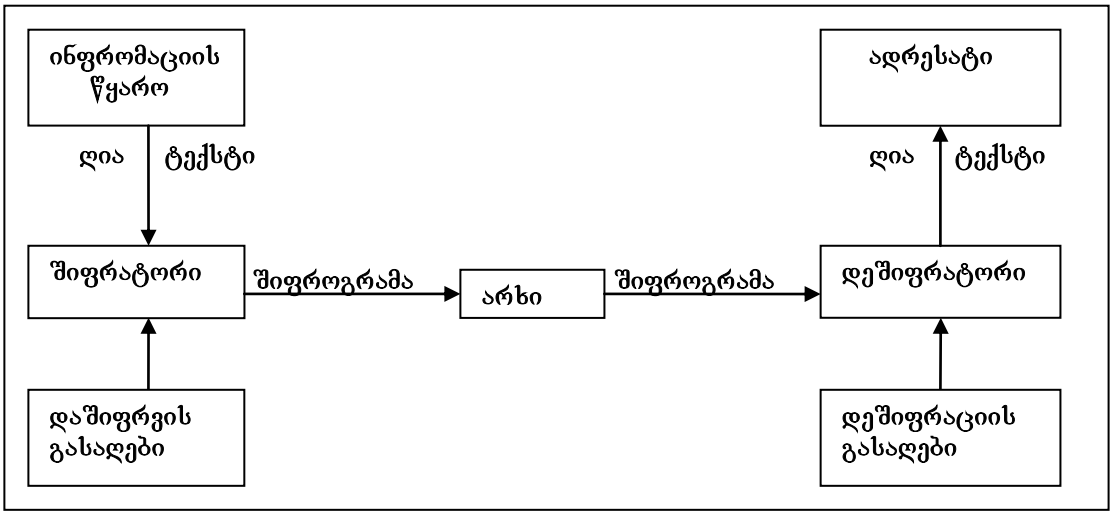
განსაზღვრა 6.3. *n სიმბოლოს ორთოგონალური მასივი $OA(n,k,\lambda)$ ეწოდება $\lambda n^2 \times k$ მასივს, რომელშიც ნებისმიერ ორ სვეტში ყველა შესაძლო n^2 წყვილი გვხვდება ზუსტად λ სტრიქონში.*

ზემოთ განხილულ მაგალითში აუთენტიფიკაციის მატრიციდან მიიღება ორთოგონალური მასივი (იხ.სურ. 6.3).

ორთოგონალური მასივები ექვივალენტურები არიან კომბინატორიკის თეორიაში კარგად შესწავლილი ურთიერთორთოგონალური ლათინური კვადრატების. მტკიცდება თეორემა იმის შესახებ, რომ თუ გვაქვს ორთოგონალური მასივი $OA(n,k,\lambda)$, მაშინ ყოველთვის არსებობს ისეთი აუტენტიფიკაციის კოდი (P,A,K,E), სადაც $|P|=k,\ \ |A|=n,\ \ |K|=\lambda n^2$ და $\Pr(i)=\Pr(s)=1/n$, ანუ A-კოდი არის სრულყოფილი კოდი. ამასთან უნდა აღვნიშნოთ, რომ მხოლოდ ასეთი კოდები შეიძლება იყოს სრულყოფილი კოდი ზემოთმოყვანილი განსაზღვრის შესაბამისად.

ასეთი სრულყოფილი კოდების პრაქტიკაში გამოყენების სირთულე დაკავშირებულია იმასთან, რომ თუ გვაქვს m ბიტის სიგრძის ღია ტექსტი, გასაღების სიგრძე არ უნდა იყოს ნაკლები $n=2m$ ბიტზე, რაც ძალიან რთულად განსახორციელებელია პრაქტიკაში.

6.8. ღია გასაღებიანი კრიპტოგრაფიული ალგორითმები. ღია გასაღებიანი კრიპტოგრაფიული ალგორითმების ფუნქციონირება და კრიპტომედეგობა მთლიანად ეფუძნება იმ **NP** სირთულის ამოცანას, რომელიც გამოყენებულია ცალმხრივ მიმართულ ფუნქციაში. ამ შემთხვევაში კრიპტოსისტემაში გვაქვს ორი გასაღები: ღია და დახურული. იმის და მიხედვით, თუ რას ემსახურება სისტემა, ღია გასაღები შეიძლება გამოიყენებოდეს ინფორმაციის დასაშიფრად და საიდუმლო კი დეშიფრაციისათვის ან პირიქით. მაგალითად, **RSA** სისტემაში, რომელიც ძირითადად გამოიყენება ინფორმაციის დასაშიფრად, ღია გასაღებით ხდება ინფორმაციის დაშიფრვა, ხოლო საიდუმლო გასაღებით კი დეშიფრაცია. ციფრული ხელმოწერის ალგორითმებში კი პირი-ქით - სადუმლო გასაღებით ხდება ტექსტის ხელმოწერა, ხოლო ღია გასაღებით კი ხელმოწერის შემოწმება. ღია გასაღები ნიშნავს, რომ ეს გასაღები ცნობილია სისტემის ყველა მომხმარებლისთვის და ყველას შეუძლია ისარგებლოს ამ გასაღებით. ასეთი სისტემების დიდ უპირატესობას სიმეტრიულ ერთგასაღებიან სისტემებთან შედარებით წარმოადგენს ის, რომ აღარ გვჭირდება დახურული არხი საიდუმლო გასაღების გადასაცემად, რაც ყოველთვის დაკავშირებული სირთულეებთან (იხ. სურ. 6.4).



სურ. 6.3 ინფორმაციის გადაცემის სქემა ღია გასაღებიანი კრიპტოალგორითმის საშუალებით

თავისთავად ცხადია, რომ ღია და საიდუმლო გასაღებებს შორის უნდა არსებობდეს რაღაც კავშირი, წინააღმდეგ შემთხვევაში შეუძლებელი იქნებოდა ინფორმაცია დაგვეშიფრა ერთი გასაღებით და დეშიფრაცია კი მოგვეხდინა მეორე გასაღებით. ეს კავშირი ხდება ჩვენთვის უკვე ცნობილი ცალმხრივ მიმართული ხაფანგიანი ფუნქციის საშუალებით. კერძოდ, საიდუმლო გასაღები წარმოადგენს ცალმხრივ მიმართული ხაფანგიანი ფუნქციის არგუმენტს, ხოლო ღია გასაღები კი ამ

ფუნქციის მნიშვნელობას. შესაბამისად, ღია გასაღებიანი ალგორითმის მედეგობა კრიპტოანალიტიკური შეტევების მიმართ მთლიანად განისაზღვრება იმ **NP** ამოცანის სირთულით, რომელიც საფუძვლად უდევს ცალმხრივ მიმართულ ხაფანგიან ფუნქციას. მართლაც, რადგანაც კრიპტოანალიტიკოსმა იცის დაშიფრვის ალგორითმი და ღია გასაღები, მას აღარ სჭირდება არხიდან ინფორმაციის მიტაცება. ალგორითმისა და გასაღების საშუალებით კრიპტოანალიტიკოსი თვითონ ქმნის დაშიფრულ ტექსტს და შეუძლია განახორციელოს ნებისმიერი შეტევა კრიპტოსისტემაზე, იქნება ეს შეტევა მხოლოდ შიფროგრამის, ღია ტექსტის, თუ შერჩეული ღია ტექსტის საშუალებით.

6.9. ღია გასაღებიანი კრიპტოსისტემების გამოყენება. სიმეტრიული კრიპტოალგორითმებისაგან განსხვავებით, ღია კრიპტოგრაფიულ ალგორითმებში გამოიყენება ისეთი ოპერაციები, რომელთა შესრულება მოითხოვს საკმაოდ დიდ დროს, ამიტომ ასეთი ალგორითმების გამოყენება ინფორმაციის კონფიდენციალობის დასაცავად არაა პრაქტიკული. ინფორმაციის დიდი მასივების დაშიფრვასა და დეშიფრაციას სჭირდება ძალიან დიდი დრო, მაგრამ ამ ალგორითმების გამოყენება ძალიან მოსახერხებელია ინფორმაციის მთლიანობის, აუთენტიფიკაციისა და ციფრული ხელმოწერის ამოცანების გადასაჭრლად. ძალიან მნიშვნელოვანია აგრეთვე მათი გამოყენება სიმეტრიული კრიპტოალგორითმებისათვის გასაღებების გადაცემის პრობლემის გადასაჭრელად. როგორც ვიცით, სიმეტრიული კრიპტოალგორითმების გამოყენების დროს ერთერთი ყველაზე უფრო სუსტი მომენტია გასაღებების განაწილება, რადგანაც ის მოითხოვს დახურული არხის არსებობას.

თუ ამ პრობლემის გადასაჭრელად გამოვიყენებთ ღია გასაღებიან სისტემას, მაშინ აღარ დაგვჭირდება დახურული არხი. გასაღები, რომლის გადაცემაც გვინდა ადრესატისათვის, დაიშიფრება ღია გასაღების საშუალებით (გასაღების სიგრძე არაა დიდი) და გადაიცემა ღია არხის საშუალებით. ადრესატი დაშიფრული გასაღების დეშიფრაციას მოახდენს მხოლოდ მისთვის ცნობილი საიდუმლო გასაღების საშუალებით. არსებობს ასევე ღია გასაღებიანი სისტემები, რომლებიც საშუალებას იძლევიან შევქმნათ საიდუმლო გასაღები ღია არხში ადრესატების კავშირის პროცესში ისე, რომ მიუხედავად იმისა, რომ მოწინააღმდეგე ისმენს მათ საუბარს, მას არ შეუძლია გაიგოს რა გასაღები შექმნეს ადრესატებმა. ასეთი სისტემა განსაკუთრებით მოსახერხებელია ერთჯერადი, სეანსური გასაღების შესაქმნელად.

საკონტროლო კითხვები:

1. რა ამოცანები დგას თანამედროვე კრიპტოგრაფიის წინაშე?
2. განმარტეთ ტერმინი "ინფორმაციის მთლიანობა".
3. რას უწოდებენ კრიპტოგრაფიულ პრიმიტივს?
4. რა კრიტერიუმებს უნდა აკმაყოფილებდნენ კრიპტოგრაფიული პრიმიტივები?
5. ჩამოაყალიბეთ A კოდის განმარტება.
6. რა ტიპის შეტევები შეიძლება განხორციელდეს ინფორმაციის მთლიანობის დასარღვევად?
7. რას ნიშნავს მოტყუების ალბათობები? როგორია მათი ქვედა საზღვარი?
8. განმარტეთ სრულყოფილი კოდი.
9. რა არის ორთოგონალური მასივი?
10. სრულყოფილი კოდის რა თვისება ართულებს მის გამოყენებას პრაქტიკაში?
11. როგორ კრიპტოსისტემას ეწოდება ღია გასაღებიანი ალგორითმი?
12. რა უპირატესობა აქვთ ასეთ სისტემებს სიმეტრიულ კრიპტოსისტემებთან შედარებით?
13. თანამედროვე კრიპტოგრაფიის რომელი ამოცანების გადასაჭრელად გამოიყენება ძირითადად ღია გასაღებიანი კრიპტოსისტემა?
14. რა პრინციპით ხდება ღია გასაღების მიღება დახურული გასაღებიდან?

.

ჰეშ-ფუნქციები

7.1. რა არის ჰეშ-ფუნქცია. ჰეშ-ფუნქციები გამოიყენება არა მარტო კრიპტოგრაფიაში, არამედ ინფორმატიკის სხვა დარგებშიც, კერძოდ ფართოდ გამოიყენება კოდირების თეორიაში. ჰეშ-ფუნქცია საშუალებას გვაძლევს შევუსაბამოთ ნებისმიერი სიგრძის რაიმე ინფორმაციას (რომელიც წარმოდგენილია როგორც ნოლების და ერთების მიმდევრობისაგან შედგენილი ნებისმიერი სიგრძის სტრიქონი), ფიქსირებული n სიგრძის სტრიქონი.

განსაზღვრა 7.1. *ფუნქციას, რომელიც ნებისმიერი სიგრძის ორობით მიმდევრობას შეუსაბამებს ფიქსირებული n სიგრძის ორობით მიმდევრობას, ეწოდებ ჰეშ-ფუნქცია.*

$$h = H(m)$$

როგორც ამ განმარტებიდან ჩანს, ფუნქციის მნიშვნელობათა სიმრავლე არ აღემატება 2^n-ს, მაშინ როდესაც მისი განსაზღვრის არე გაცილებით მეტია ამ რიცხვზე, რაც იმას ნიშნავს, რომ ორი განსხვავებული m_1 და m_2-თვის თეორიულად შეიძლება ადგილი ჰქონდეს ტოლობას:

$$H(m_1) = H(m_2)$$

ამ შემთხვევას უწოდებენ ***კოლიზიას***. ასეთი ფუნქციის გამოყენებას კრიპტოგრაფიაში აზრი არა აქვს. ამიტომ ***კრიპტოგრაფიული ჰეშ-ფუნქცია*** უნდა აკმაყოფილებდეს შემდეგ პირობებს:

1. *ჰეშ-ფუნქცია უნდა იყოს ცალხმრივ მიმართული ფუნქცია, ანუ ნებისმიერი m-თვის ადვილი უნდა იყოს h-ის გამოთვლა, მაგრამ არ უნდა არსებობდეს პოლინომური ალგორითმი, რომელიც მოგვცემს საშუალებას მოცემული h-თვის ვიპოვოთ ისეთი m, რომ დაკმაყოფილდეს ტოლობა $h = H(m)$.*
2. *თუ მოცემული გვაქვს ფიქსირებული m_1, მაშინ შეუძლებელი უნდა იყოს ისეთი m_2-ის მოძებნა, რომ შესრულდეს ტოლობა $h(m) = h(m^{/})$.*
3. *ნებისმიერად აღებული ორი m_1 და m_2 წყვილისათვის უნდა სრულდებოდეს პირობა*

$$h(m_1) \neq h(m_2)\ .$$

თუ სრულდება მეორე პირობა და არ სრულდება მესამე, მაშინ ამბობენ, რომ ფუნქცია **სუსტად თავისუფალია კოლიზიისაგან** და თუ სრულდება მესამე პირობაც, მაშინ ამბობენ, რომ ფუნქცია **ძლიერ თავისუფალია კოლიზიისაგან.**

რა შეიძლება მოხდეს თუ ეს ბოლო პირობა არ იქნება დაცული. განვიხილოთ ასეთი პრო-ტოკოლი. დავუშვათ, ეკას სურს მოატყუოს ბექა და მოაწერინოს მას ხელი ისეთ კონტრაქტზე, რომელიც გამოიწვევს ბექას გაკოტრებას. პროტოკოლის პროცედურა ასეთია:

1. ეკა ამზადებს კონტრაქტის ორ ვერსიას, ერთი სასარგებლოა ბექასათვის, მეორე კი არა.
2. ეკას შეაქვს მცირეოდენი ცვლილებები ორივე კონტრაქტში (ეს ცვლილებები შეიძლება იყოს ჰარის ჩამატება, ჰარის შეცვლა "ჰარი - ადგილის გამოტოვება – ჰარი" კომბინაციით და ასე შემდეგ) და ითვლის ყველა მიღებული ვარიანტების ჰეშ-ფუნქციებს, თუ ეკა ასეთ ცვლილებებს შეიტანს ყოველ ოცდამეთორმეტე სტრიქონში, მას შეუძლია მიიღოს 2^{32} განსხვავებული ვარიანტი თითოეული კონტრაქტისა.
3. ეკა ადარებს ჰეშ-ფუნქციის მნიშვნელობებს და ეძებს ისეთ წყვილს, რომელთა ჰეშ-ფუნქციის მნიშვნელობები ერთმანეთს ემთხვევა (თუ ჰეშ-ფუნქციის გამოსასვლელი მნიშვ-ნელობა მხოლოდ სამოცდაოთხ ბიტიანი სტრიქონია, ეკა შეძლებს ასეთი წყვილის პოვნას დაბადების დღის პარადოქსიდან გამომდინარე). მაშინ ის დაიტოვებს კონტრაქტების იმ წყვილს, რომელთა ჰეშ-ფუნქციათა მნიშვნელობები ერთმანეთს დაემთხვა.
4. ეკა მოაწერინებს ბექას ხელს იმ კონტრაქტზე, რომელიც სასარგებლოა ბექასათვის (ამისათვის ის გამოიყენებს პროტოკოლს, რომელშიც ხელმოწერა ხდება მხოლოდ ჰეშ-ფუნქციაზე).
5. ეკა შეცვლის ამ კონტრაქტს მეორე კონტრაქტით და ბექა ვერ შეძლებს დაამტკიცოს, რომ მას ხელი არ მოუწერია ამ კონტრაქტზე.

როგორც ვხედავთ, ეს შეტევა დაფუძნებულია ცნობილი "დაბადების დღის" პარადოქსზე და სწორედ ამ შეტევისაგან თავის დასაცავად მოითხოვება მესამე პირობა. ვნახოთ რას წარმოადგენს დაბადების დღის პარადოქსი.

7.2. დაბადების დღეების" პარადოქსი. დავუშვათ, გვაინტერესებს რამდენი ადამინი უნდა იყოს ჯგუფში, რომ ორი, შემთხვევითად აღებული ადამიანის დაბადების დღეები ერთმანეთს დაემთხვეს ალბათობით $p(n)$. ამ მაგალითში ჩვენ ვთვლით, რომ წელიწადი შედგება 365 დღისაგან (არ ვითვალისწინებთ ნაკიან წელს) და ასევე ვთვლით, რომ დაბადების დღეები თანაბრადაა განაწილები წელიწადში. ცხადია რეალურად ეს ასე არაა, მაგრამ ისიც ცხადია, რომ რაც უფრო არათანაბრი იქნება ეს განაწილება, მით უფრო გაიზრდება თანხვედრების ალბათობა.

ამ ალბათობის დასათვლელად ჯერ დავითვალოთ ალბათობა იმისა, რომ თუ ჯგუფში არის n ადამიანი, არცერთი მათგანის დაბადების დღეები არ დაემთხვევა ერთმანეთს $\overline{p}(n)$. თუ $n > 365$, მაშინ ცხადია ეს ალბათობა იქნება ნოლის ტოლი, მაგრამ თუ $n \le 365$, მაშინ შეგვიძლია ვიმსჯელოთ ასე: შემთხვევითად ავიღოთ ერთი ადამიანი და დავაფიქსიროთ მისი დაბადების დღე. ცხადია, ალბათობა იმისა, რომ ეს დაბადების დღე არ დაემთხვევა სხვა ადამიანების დაბადების დღეს იქნება ერთის ტოლი. ამის შემდეგ კვლავ შემთხვევითად ავიღოთ მეორე ადამიანი და დავაფიქსიროთ მისი დაბადების დღე. ამ ხდომილობის ალბათობა უკვე იქნება $(1-1/365)$-ის ტოლი. თუ გავაგრძელებთ ანალოგიურ მსჯელობას, მივიღებთ, რომ ბოლო მე-n ადამიანისათვის ალბათობა იქნება $(1-(n-1)/365)$-ს ტოლი. რადგანაც ყველა ეს ალბათობები ერთმანეთისაგან დამოუკიდებელია მივიღებთ, რომ

$$\overline{p}(n) = 1\cdot(1-1/365)\cdot(1-2/365)\cdot\ldots\cdot(1-(n-1)\Big/365) = \frac{365!}{(365-n)!365^n}\ .$$

მაშინ ალბათობა იმისა, რომ ორი, შემთხვევითად აღებული ადამიანის დაბადების დღეები დაემთხვევა ერთმანეთს ტოლია $p(n) = 1-\overline{p}(n)$. ეს ფუნქცია n-ის ზრდასთან ერთად ძალიან სწრაფად იზრდება, რაც ეწინააღმდეგება ჩვენს ინტუიციურ წარმოდგენას მის ქცევაზე და $n = 23$-თვის უკვე აჭარბებს $0{,}5$-ს ($p(23) = 0{,}503$). განსაკუთრებით საინტერესო და კრიპტოგრაფიულად მნიშვნელოვანია, თუ მიღებულ შედეგებს შევადარებთ ოდნავ სახეშეცვლილ ამოცანის შედეგებს. დავუშვათ დავაფიქსირეთ შემთხვევითად რომელიმე ადამიანის დაბადების დღე და გვაინტერესებს, რა არის ალბათობა იმისა, რომ n ადამიანისაგან შემდგარ ჯგუფში (ეს ადამიანი არ შედის ჯგუფში) რომელიმე მათგანის დაბადების დღე დაემთხვევა ამ ადამიანის დაბადების დღეს. ეს ალბათობა ტოლია

$$p(n) = 1-\frac{364^n}{365^n}.$$

ეს ფუნქცია განსხვავებით წინა ფუნქციისაგან იზრდება გაცილებით ნელა და მეტი ხდება $0{,}5$-ზე, როდესაც $n = 253$. ეს ხდება იმიტომ, რომ ჯგუფში ადამიანების დაბადების დღეები შეიძლება ემთხეოდეს ერთმანეთს, ეს კი ამცირებს ფუნქციის მნიშვნელობას.

7.3. კრიპტოგრაფიული ჰეშ-ფუნქცია. ვაჩვენოთ, რომ ზემოთმოყვანილ განმარტებაში მესამე პირობის შესრულება ავტომატურად იწვევს პირველი და მეორე მოთხოვნების შესრულებას. დავუშვათ, $n = 253$ არის ჰეშ-ფუნქცია და P და $H(P)$ სასრული სიმრავლეებია, რომლებისთვისაც სრულდება პირობა $|P| \ge 2|H(P)|$. დავუშვათ არსებობს h ის შებრუნებული ალგორითმი. მაშინ უნდა არსებობდეს ისეთი ალბათური ალგორითმი, რომელიც აღმოაჩენს კოლიზიას ალბათობით არანაკლებ $1/2$-სა. დავუშვათ, არსებობს ასეთი ალგორითმი, რომელიც ან იპოვნის კოლიზიას, ან ვერ გაგვცემს პასუხს არის თუ არა კოლიზია. გამოვთვალოთ ამ ალგორითმის წარმატების ალბათობა. ნებისმიერი $p \in P$-თვის განვსაზღვროთ $p \sim p_1$, თუ $h(p) = h(p_1)$. ადვილი შესამოწმებელია, რომ $\sim$ წარმოადგენს ეკვივალენტობის მიმართებას. აღვნიშნოთ ეს კლასები შემდეგი სახით: $[p] = \{p_1 \in P : p \sim p_1\}$. თითოეული ეკვივალენტობის $[p]$ კლასი შედგება $H(P)$ ელემენტების შებრუნებულებისაგან, ამიტომ ეკვივალენტური კლასების რაოდენობა იქნება მაქსიმუმ $|H(P)|$. აღვნიშნოთ ამ კლასების სიმრავლე C-თი. დავუშვათ, რომ p არის ელემენტი P-დან, რომელიც პირველ ბიჯზე აირჩია ალგორითმმა. მეორე ბიჯზე ის გამოთვლის $h(p)$-ს. ამ p-თვის არსებობს

$\lceil p \rceil$ ცალი შესაძლო p_1, რომლებშიც შეიძლება დაბრუნდეს $h(p)$ მესამე ბიჯზე. აქედან p_1-ის $\lceil p \rceil - 1$ ცალი მნიშვნელობა განსხვავებულია p-გან და მათი კვლავ ჰეშ-ფუნქციით გარდაქმნით მეოთხე ბიჯზე ალგორითმი იპოვის კოლიზიებს, რაც იქნება მისი წარმატება (შევნიშნოთ, რომ პირველ ბიჯზე ალგორითმმა არ იცის, თუ ეკვივალენტობის რომელი კლასის წარმომადგენელი იყო არჩეული კონკრეტულად). აქედან გამომდინარე კონკრეტული $p \in P$ წარმატების ალბათობა ტოლია $(\lceil p \rceil - 1)/\lceil p \rceil$-ის. რადგანაც ჩვენ გვაინტერესებს წარმატების გასაშუალოებული ალბათობის გამოთვლა p-ს ყველა მნიშვნელობებისათვის, გვექნება:

$$p(y) = \frac{1}{|P|}\sum_{p\in P}\frac{\lceil p \rceil - 1}{\lceil p \rceil} = \frac{1}{|P|}\sum_{c\in C}\sum_{p\in c}\frac{|C|-1}{|C|} = \frac{1}{|P|}\sum_{c\in C}(|C|-1) =$$

$$= \frac{1}{|P|}\left(\sum_{c\in C}|C| - \sum_{c\in C}1\right) \geq \frac{|P| - |H(P)|}{|P|} \geq \frac{|P| - |P|/2}{|P|} = \frac{1}{2}\ .$$

ამგვარად, ჩვენ ავაგეთ ალბათური ალგორითმი, რომელიც სულ მცირე, $1/2$ ალბათობით გამოიცნობს კოლიზიებს. აქედან გამოდის, რომ მესამე პირობა სრულიად საკმარისია, რადგანაც მისი შესრულება ავტომატურად ნიშნავს პირველი და მეორე პირობების შესრულებას. ისეთ ჰეშ-ფუნქციებს, რომლებიც აკმაყოფილებენ მესამე პირობას უწოდებენ ***კრიპტოგრაფიულ ჰეშ- ფუნქციებს.***

7.4. გასაღებიანი და უგასაღებო ჰეშ-ფუნქციები. ჰეშ-ფუნქციის გამოთვლის ალგორითმი არგუმენტად შეიძლება იყენებდეს მხოლოდ ღია ტექსტს. ასეთ შემთხვევაში მიღებული მნიშვნელობა შეუძლებელია უშუალოდ გამოვიყენოთ ინფორმაციის მთლიანობის დასაცავად, რადგანაც თუ ეკა გადაცემის დროს შეცვლის ტექსტს, ის აგრეთვე შეცვლის ჰეშ-ფუნქციის მნიშვნელობასაც და ბექა, რომელიც მიიღებს ამ ტექსტს, ვერ მიხვდება, რომ ტექსტი შეცვლილია. უგასაღებო ჰეშ-ფუნქციები ფართოდ გამოიყენება ისეთ კრიპტოსისტემებში, რომლებიც ამ ფუნქციის მნიშვნელობას შემდეგ კიდევ ამუშავებენ სხვა პარამეტრებთან ერთად ისე, რომ თუ ეკა შეცვლის მის მნიშვნელობას, ბექა ამას ადვილად მიხვდება. ერთერთი ასეთი სისტემაა ციფრული ხელმოწერის ალგორითმები. ჰეშ-ფუნქციის გამოთვლის როგორც უგასაღებო, ასევე გასაღებიანი ალგორითმები აუცილებლად უნდა აკმაყოფილებდნენ კრიპტოგრაფიული ჰეშ-ფუნქციის პირობებს, წინააღმდეგ შემთხვევაში შესაძლებელია მოწინააღმდეგემ შეძლოს სისტემის გატეხვა. ვნახოთ, რა ალგორითმები არსებობენ უგასაღებო კრიპტოგრაფიული ჰეშ- ფუნქციის გამოსათვლელად.

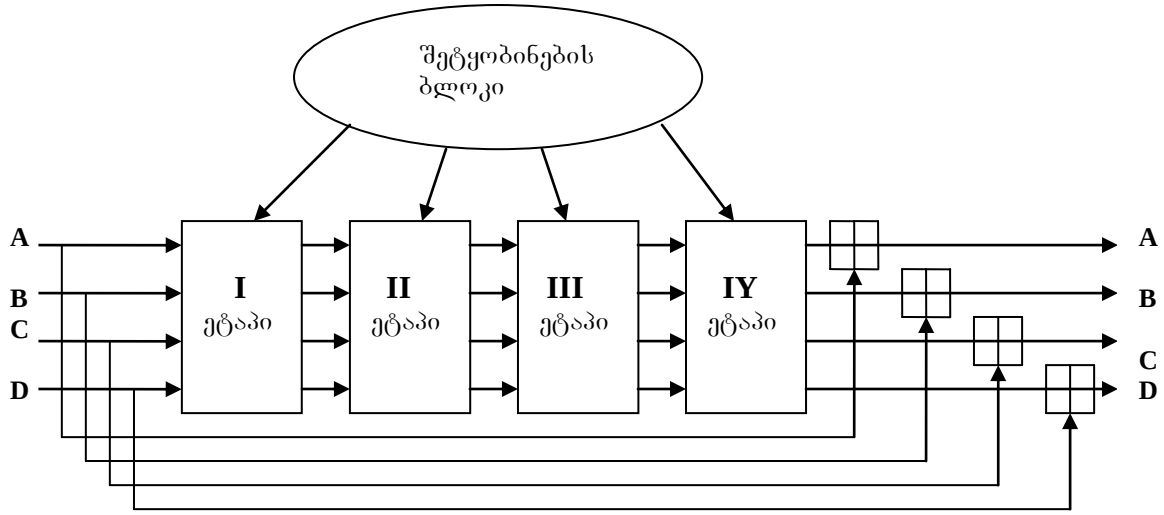
7.4. ალგორითმი MD5 (Message Digest). აშკარაა, რომ ჰეშ-ფუნქციის გამძლეობა ძალისმიერი და დაბადების დღის პარადოქსით შეტევების მიმართ პირდაპირაა დამოკიდებული ჰეშ-ფუნქციის მნიშვნელობის სიგრძეზე, რაც უფრო გრძელი იქნება გამოსასვლელი მნიშვნელობა, მით უფრო გართულდება შეტევა. დღეს არსებული ალგორითმები გამოიმუშავებენ 128, 160 და 256 ბიტის სიგრძის გამოსასვლელებს, რომლებიც უზრუნველყოფენ ჰეშ-ფუნქციის გამძლეობას აღნიშნული შეტევისადმი. მაგალითად, იმისათვის, რომ შეტევა დაბადების დღის პარადოქსით იყოს წარმატებული 160 ბიტიანი გამოსასვლელის შემთხვევაში, საჭიროა 2^{80} ჰეშირების ჩატარება. ჰეშ-ფუნქციის მნიშვნელობის გამოსათვლელად შეიძლება გამოვიყენოთ ნებისმიერი სიმეტრიული ალგორითმი მცირეოდენი მოდიფიკაციით, მაგრამ საიმედოობის თვალსაზრისით მაინც უმჯობესია სპეციალური ალგორითმის გამოყენება. დღემდე ყველაზე გავრცელებულ ალგორითმებს წარმოადგენენ **MD-5**, რომელიც შექმნილია რ. რაივესტის მიერ და **SHA-1**, რომელიც შეიქმნა აშშ სტანდარტებისა და ტექნოლოგიების ინსტიტუტში.

ალგორითმი **MD-5** შეტყობინების წინასწარი დამუშავების შემდეგ გადაამუშავებს მას 512 ბიტიან ბლოკებად, რომლებიც დაყოფილა თექვსმეტ 32 ბიტიან ქვებლოკებად.Aალგორითმის გამოსასვლელია ოთხი 32 ბიტიანი ქვებლოკი, რომლებიც ერთიანდებიან ერთ 128 ბიტიან ბლოკში. წინასწარი დამუშავების დროს ტექსტის სიგრძე იზრდება $512 \cdot k - 64$ ბიტამდე, ამისათვის მას ემატება ერთიანი და იმდენი ნოლი რამდენიც საჭიროა შესაბამისი სიგრძის მისაღებად. ამის შემდეგ მას ემატება 64 ბიტიანი რიცხვი, რომელიც აღნიშნავს შეტყობინების სიგრძეს (რეალურს და არა დამატების შედეგად მიღებულს). ეს ხდება იმისთვის, რომ ჯერ ერთი ტექსტის ზომა იყოს 512-ის ჯერადი, რაც აუცილებელია ალგორითმის შემდგომი მუშაობისათვის და მეორეც, ჩანაწერი ტექსტის სიგრძის შესახებ იძლევა გარანტიას, რომ სხვადასხვა შეტყობინებები არ დაემსგავსებიან

ერთმანეთს დამატების შემდეგ. ალგორითმში განისაზღვრება ოთხი ცვლადი, რომლებსაც უწოდებენ ***გადაბმის ცვლადებს*** და რომელთა მნიშვნელობები თექვსმეტობით სისტემაში ასეთია:

$$A=(01234567)_x; B=(89absdef)_x; C=(fedcba98)_x; D=(76543210)_x .$$

ალგორითმის მთავარი ციკლის სქემა მოყვანილია სურ. 7.1. ეს ციკლი გრძელდება სანამ არ ამოიწურება შეტყობინების ყველა 512 ბიტიანი ბლოკი. ციკლში შესვლის წინ ხდება ცვლადების კოპირება შესაბამისად a,b,c და d ცვლადებში და 512 ბიტიანი ტექსტი დაიყოფა 16 ოცდათორმეტბიტიან ქვებლოკებად. მთავარი ციკლი შედგება ოთხი, ერთმანეთის მსგავსი გარდაქმნისაგან, რომელსაც ეწოდება ეტაპი. თითოეულ ეტაპზე თექვსმეტჯერ (ერთი ქვებლოკისათვის ერთხელ) გამოიყენება სხვადასხვა ოპერაციები და თითოეული ასეთი ოპერაცია წარმოადგენს არაწრფივ ფუნქციას, რომელიც ატარებს ოპერაციებს სამ ცვლადზე მოცემული ოთხი ცვლადიდან (იხ. სურ. 7.2). შემდეგ მიღებულ შედეგს ემატება ჯერ მეოთხე ცვლადის მნიშვნელობა, შემდეგ ტექსტის ქვებლოკი (M_j) და შემდეგ მუდმივი რიცხვი (t_j).



სურ. 7.1 **MD-5** ალგორითმის მთავარი ციკლის სქემა.

ყოველ ეტაპზე გვაქვს განსხვავებული არაწრფივი ფუნქცია:

პირველზე – $F(X,Y,Z)=(X\wedge Y)\vee((\neg X\wedge Z)$;

მეორეზე – $G(X,Y,Z)=(X\wedge Z)\vee(Y\wedge(\neg Z))$:

მესამეზე – $H(X,Y,Z)=X\oplus Y\oplus Z$;

მეოთხეზე – $I(X,Y,Z)=Y\oplus(X\vee(\neg Z))$

თითოეულ ეტაპზე თექვსმეტჯერ სრულდება შემდეგი ოპერაციები:

$$a=b+((a+F(b,c,d)+M_j+t_j)<<<s)$$

$$a=b+((a+G(b,c,d)+M_j+t_j)<<<s)$$

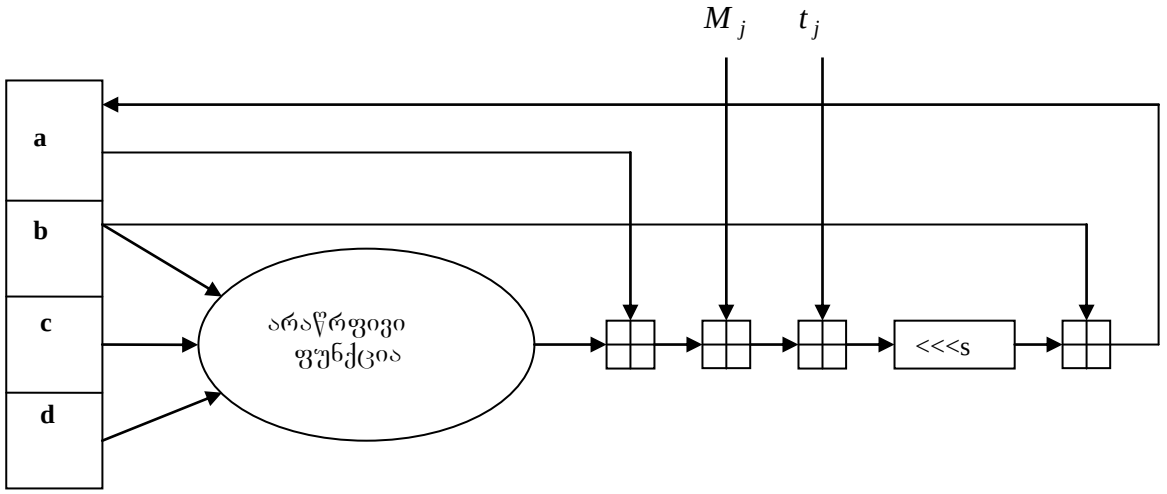
$$a=b+((a+H(b,c,d)+M_j+t_j)<<<s)$$

$$a=b+((a+I(b,c,d)+M_j+t_j)<<<s)$$

ამასთან ყოველი შესრულების დროს t_j ღებულობს განსხვავებულ მნიშვნელობებს. ოთხივე ეტაპის დამთავრების შემდეგ მიღებული შედეგი ოცდათორმეტის მოდულით კვლავ იკრიბება გადაბმის ცვლადებთან და ალგორითმი იწყებს ახალი ბლოკის დამუშავებს. საბოლოო გამოსასვლელს წარმოადგენს გადაბმის ცვლადების კონკატაცია.

7.5. უსაფრთხო ჰეშირების ალგორითმი (Secure Hadh Algorithm – SHA-1). უსაფრთხო ჰეშირების ალგორითმი შეიმუშავა აშშ სტანდარტებისა და ტექნოლოგიების ინსტიტუტმა (**NIST**) ნაციონალური უსაფრთხოების სააგენტოსთან (**NSA**) ერთად, ციფრული ხელმოწერის სტანდარტში გამო-

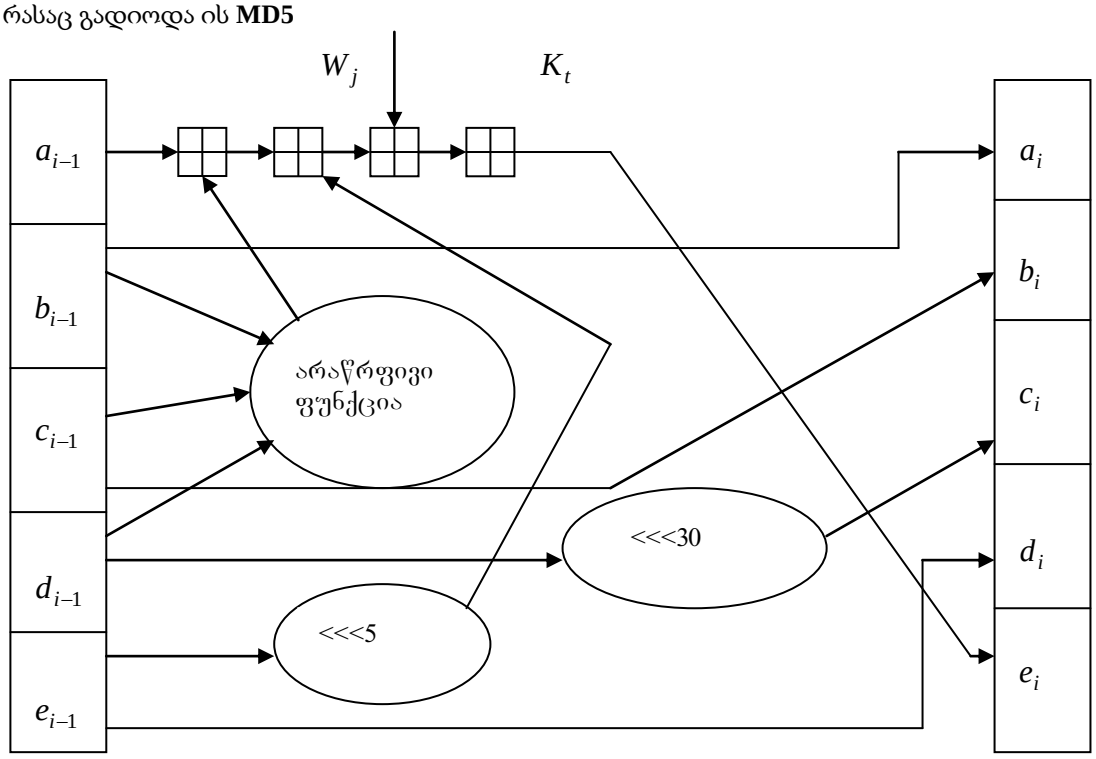
საყენებლად. როგორც თვითონვე აღნიშნავდნენ ეს ორგანიზაციები, **SHA** ალგორითმი აგებული იყო იგივე პრინციპეზე, რაც გამოიყენება **MD5**-ში.



სურ. 7.2 **MD**5 –ის ერთი ეტაპის სქემა.

აქ M_j არის 512 ბიტიანი ტექსტის ოცდათორმეტ ბიტიანი ქვებლოკი ($j = \overline{1,16}$), t_j - ცვლადია, $<<< s$ ნიშნავს წაძვრას მარცხნივ s თანრიგით და $\boxplus$ შეკრებას ოცდათორმეტის მოდულით

ძირითადი განსხვავება ამ ორ ალგორითმს შორის მდგომარეობს იმაში, რომ **SHA-1** გამოსასვლელზე იძლევა 160 ბიტიან მნიშვნელობას. აქედან გამომდინარე, არის ცვლილებები ალგორითმის მუშაობაშიც. კერძოდ, გადაბმის ოთხი ცვლადის ნაცვლად გამოიყენება ხუთი ცვლადი, რომელთა მნიშვნელობები ოდნავ განსხვავდება **MD5**-ში გამოყენებული მნიშვნელობებისაგან (მაგალითად, თუ **MD5**-ში $A = (01234567)_x$, **SHA-1**-ში გვაქვს $A = (67452301)_x$). ალგორითმი ამუშავებს შეტყობინებას 512-ბიტიან ბლოკებად. დამუშავების წინ შეტყობინება გადის იმავე პროცედურებს რასაც გადიოდა ის **MD5**



სურ. 7.3 **SHA-1**-ის ერთი ოპერაციის სქემა

-ში. დამუშავება ხდება ოთხ ეტაპად, მაგრამ იმისათვის, რომ მივიღოთ 160 ბიტიანი გამოსასვლელი, ეტაპზე ოპერაციების რაოდენობა გაზრდილია ოცამდე. ამ მიზნით შეტყობინების თექვსმეტი ოცდათორმეტბიტიანი ქვებლოკების რაოდენობა გაზრდილია ოთხმოც ქვებლოკამდე შემდეგი პროცედურის გამოყენებით:

$W_t = M_t$, როცა $t = \overline{0,15}$ და $W_t = (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) <<<1$, როცა $t = \overline{16,79}$.

განსხვავებით **MD5**-გან ამ ალგორითმში მეორე და მეოთხე ეტაპების ფუნქციები ერთი და იგივეა და თუ **MD5**-ში ყოველი ოპერაციისათვის გამოიყენება უნიკალური მუდმივი, **SHA-1**-ში გვაქვს ერთი მუდმივი ერთი ეტაპისათვის. **SHA-1**-ის ერთი ოპერაციის სქემა მოცემულია სურ. 7.3.

გამოთვლითი საშუალებების გაძლიერებასთან ერთად (იგულისხმება არა მარტო კომპიუტერები, არამედ სპეციალური ტექნიკაც), საშიშროება იმისა, რომ ძალისმიერი შეტევით შესაძლებელი გახდება ჰეშ-ფუნქციათა გამომთვლელი ალგორითმების გატეხვა, კრიპტოგრაფებს აიძულებთ გაზარდონ უსფრთხოება ასეთი შეტევის მიმართ გამოსასვლელი სიდიდის ბიტების რაოდენობის გაზრდის ხარჯზე. დღეისათვის უკვე არსებობს **SHA-256**, რომლის გამოსასვლელი ბიტების რაოდენობა ტოლი ორასორმოცდათექვსმეტის. ამ ალგორითმში შეტყობინების დამუშავება ხდება სამოცდაოთხ ეტაპად, მაგრამ თითოეული ეტაპი არის ერთოპერაციანი. **SHA-512**-ში, რომლის გამოსასვლელი ბიტების რაოდენობა ხუთასთორმეტია, შეტყობინების დამუშავება ხდება ოთხმოც ერთოპერაციან ეტაპად. არსებობს აგრეთვე **SHA-384**, რომელიც ანალოგიურია **SHA-512**-ის, ოღონდ გამოსასვლელი ბიტების რაოდენობა ტოლია სამასოთხმოცდაოთხის.

7.6. გასაღებიანი ჰეშ-ფუნქციის გამოთვლა. ლექციის პირველ ნაწილში ჩვენ განვიხილეთ უპირობოდ მედეგი აუთენტიფიკაციის კოდები და აღვნიშნეთ, რომ მათი პარაქტიკაში გამოყენება დაკავშირებულია გარკვეულ სირთულეებთან. გასაღებიანი ჰეშ-ფუნქციები, ე.წ. **MAC (Messade Authentication Codes)** შეიძლება გამოვთვალოთ ორი გზით. პირველი გულისხმობს ჰეშ-ფუნქციის გამოთვლას გასაღებთან ერთად შემდეგი ფორმულის საშუალებით:

$$h = H(k \parallel P)$$

სადაც k არის გასაღები, $\parallel$ აღნიშნავს კონკატაციას (გადაბმას), P არის გადასაცემი ტექსტი, H არის გამოთვლის ალგორითმი და h ჰეშ-ფუნქციის მნიშვნელობა. ასეთი სახით გამოთვლილ მნიშვნელობას ხშირად უწოდებენ **HMAC**-ს (**Hash Message Authentication Codes**). ასეთ შემთხვევაში ჰეშ-ფუნქციის მნიშვნელობა შეიძლება გამოვთვალოთ ნებისმიერი, დავუშვათ, იგივე **MD-5** ან **SHA-1** ალგორითმით.

MAC-ს გამოსათვლელად ჩვენ შეგვიძლია ვისარგებლოთ აგრეთვე რომელიმე ბლოკური დაშიფრვის ალგორითმითაც. ასეთი მიდგომის გამოყენებას გააჩნია ორი საკმარისად ძლიერი არგუმენტი:

1. საიმედო და პრქტიკულად ადვილად გამოყენებადი ჰეშ-ფუნქციების, ისევე როგორც ბლოკური შიფრების, შექმნა დაკავშირებულია დიდ დანახარჯებთან, როგორც ეკონიმიკური, ასევე ადამიანური რესურსების თვალსაზრისით. ამიტომ უკეთესი იქნება გამოვიყენოთ უკვე არსებული ინსტრუმენტები, ვიდრე შევქმნათ ახალი ალგორითმები.
2. გაცილებით მნიშვნელოვანია ალბათ ის, რომ კრიპტოგრაფიული მედეგობა, რომელიც გააჩნიათ არსებულ საიმედო ბლოკურ ალგორითმებს, ავტომატურად შეიძლება გადავიტანოთ მათი გამოყენებით შექმნილ ჰეშ-ფუნქციებზე.

ბლოკური შიფრებით **MAC**-ის გამოთვლის ყველაზე გავრცელებული სქემაა **CBC-MAC**-ი. თავიდან ამ სქემაში **MAC**-ის მნიშვნელობად იღებდნენ ინფორმაციის დაშიფრვის შედეგად მიღებულ ბოლო ბლოკს, მაგრამ ეს მიდგომა არ აღმოჩნდა უსაფრთხო.

შემდეგი იდეა მდგომარეობდა **CBC-MAC**-ის ბოლო ბლოკის კიდევ ერთხელ დაშიფრვაში სხვა გასაღებით. ამას უწოდეს "დაშიფრული **MAC**-ი (**EMAC**)". ეს მიდგომა უძლური აღმოჩნდა დაბადების დღის პარადოქსით შეტევის მიმართ.

მესამე მიდგომაა ამოვყაროთ ზოგიერთი ბიტები, რათა თავიდან ავიცილოთ შეტევა დაბადების დღის საშუალებით, ანუ შევამციროთ **MAC**-ი, პირველი ნაწილიდან ზოგიერთი ბიტის ამოგდებით. ასეთი სახით გამოიყენება ეს ალგორითმი **ISO/IEC 9797** სტანდარტში.

საკონტროლო კითხვები:

1. მოიყვანეთ ჰეშ-ფუნქციის განსაზღვრა.
2. რას ნიშნავს კოლიზია?

3. რა პირობებს უნდა აკმაყოფილებდეს კრიპტოგრაფიული ჰეშ-ფუნქცია?
4. რაში მდგომარეობს დაბადების დღის პარადოქსი?
5. აღწერეთ ალგორითმი **MD5.**
6. აღწერეთ ალგორითმი **SHA-1.**
7. სად გამოიყენება უგასაღებო ჰეშ-ფუნქცია?
8. რისთვის გამოიყენება გასაღებიანი ჰეშ-ფუნქცია?
9. რას ნიშნავს **HMAC** და როგორ შეიძლება გამოვთვალოთ ეს ფუნქცია?
10. რას ნიშნავს **CBC-MAC** და რა მეთოდებით გამოითვლება ის?
11. სად გამოიყენება **CBC-MAC**.

იდენტიფიკაცია და აუთენტიფიკაცია

1. კონფიდენციალურ ინფორმაციასთან წვდომა. ამ ლექციაში განვიხილავთ ინფორმაციის ავტომატიზებული დამუშავების სისტემაში კონფიდენციალური ინფორმაციის დაცვისა და მასთან წვდომის საკითხებს. ავტომატიზებული დამუშავების სისტემის ქვეშ ვიგულისხმებთ შემდეგი ობიექტების ერთობლიობას:

1. გამოთვლითი ტექნიკის საშუალებები;
2. პროგრამული უზრუნველყოფა;
3. კავშირის არხები;
4. სხვადასხვა მატარებლებზე განთავსებული ინფორმაცია;
5. სისტემის მომხმარებლები და მომსახურე პერსონალი.

სისტემის ინფორმაციული უსაფრთხოების (**იუ**) ქვეშ კი გვესმის სისტემის ისეთი მდგომარეობა, რომლის დროსაც: სისტემას შეუძლია წინააღმდეგობა გაუწიოს მადესტაბილიზირებელ შეტევებს და თვით სისტემის არსებობა და ფუნქციონირება არ წარმოადგენს საფრთხეს გარემოსა და ამ სისტემის ელემენტებისათვის.

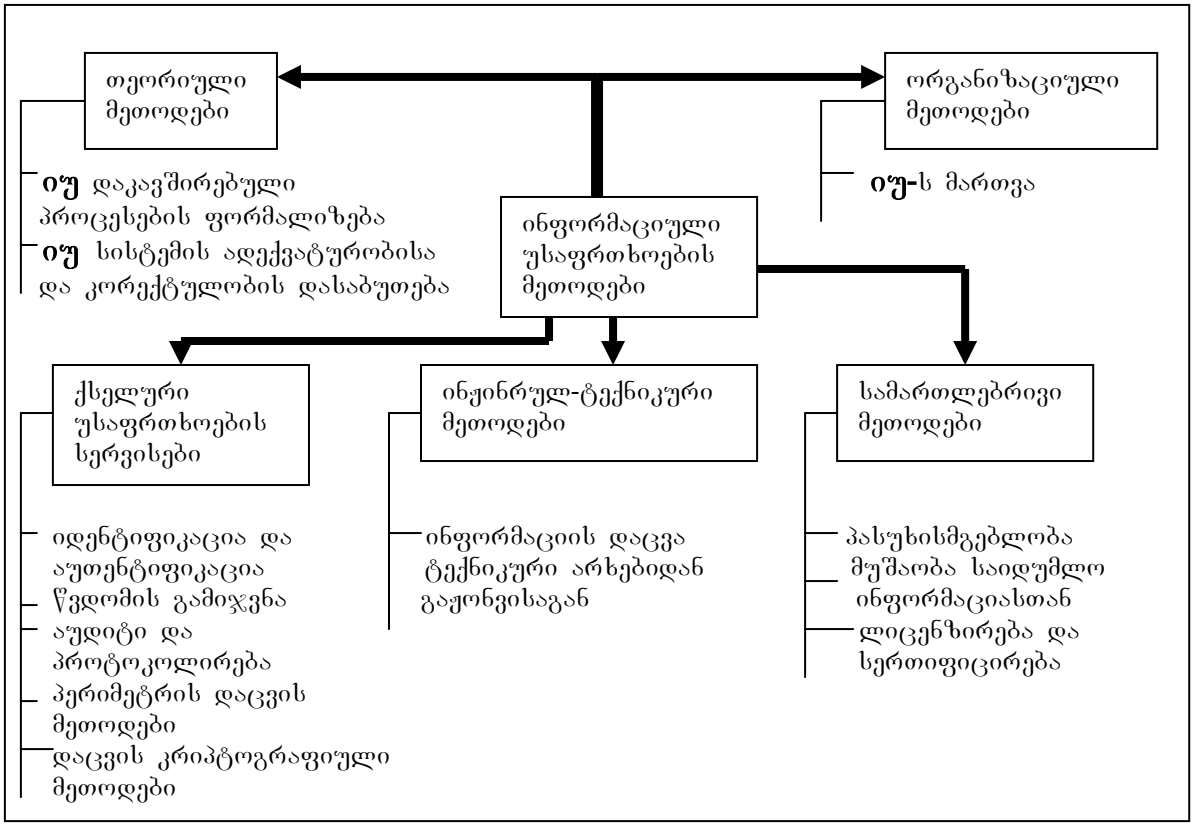
პრაქტიკულად, სისტემის ინფორმაცული უსაფრთხოება ნიშნავს რომ დაცულია ***ინფორმაციის კონფიდენციალურობა,*** ანუ ინფორმაციასთან წვდომა შეუძლიათ მიიღონ მხოლოდ ლეგალურმა მომხმარებლებმა, დაცულია ***ინფორმაციის მთლიანობა*** და უზრუნველყოფილია ***ინფორმაციასთან წვდომა,*** ანუ ლეგალური მომხმარებელი დაუბრკოლებლივ მიიღებს მისთვის სასურველ ინფორმაციას.

8.2. შეტევები ინფორმაციულ უსაფრთხოებაზე. ინფორმაციულ უსაფრთხოებაზე შეტევის ქვეშ ჩვენ ვგულისხმოთ ყველა იმ პოტენციურ ქმედებებს, პროცესებს ან მოვლენებს, რომელთა განხორციელებამაც შეიძლება გამოიწვის არალეგალიზებული მომხმარებლის წვდომა კონფიდენციალურ ინფორმაციასთან. იმის და მიხედვით, თუ რა ნიშნებით დავახასიათებთ ამ შეტევებს, შესაძლებელია მათი სხვადასხვანაირი კლასიფიკაცია. ერთერთი ყველზე ფართოდ გავრცელებულია შემდეგი:

1. ბუნებრივი და ხელოვნური შეტევები. ბუნებრივია შეტევები, რომლებიც წარმოიშვება ობიექტური ფიზიკური პროცესების, ან ბუნების სტიქიური მოვლენების შედეგად და რომელიც არაა დამოკიდებული ადამიანზე. მაგალითად წყალდიდობა, ხანძარი, მიწისძვრა და ა. შ.. ყველა შეტევა რომელიხ წარმოიშვება ადამიანის ქმედების შედეგად, წარმოადგენს ხელოვნურ შეტევას.
2. შეტევები უნებლიე და წინასწარ განზრახვით. უნებლიე შეტევები, შეიძლება განხორციელდეს სისტემაზე, მომსახურე პერსონალის დაუდევრობით, ან შეცდომის გამო. ყველა წინასწარ განზრახული შეტევა ხორციელდება ბოროტმომქმედის დაგეგმილი ქმედებების შედეგად (ეს ბოროტმომქმედი შეიძლება იყოს მომსახურე პერსონალიც).

3. შეტევის წყარო შეიძლება იყოს ბუნებრივი მოვლენა, ადამიანი, სანქცირებული აპარატულ-პროგრამული უზრუნველყოფა (მაგალითად, სისტემის რომელიმე უტილიტის არასწორად გამოყენება).
4. შეტევის წყაო შეიძლება მდებარეობდეს თვით სისტემის შიგნით (მაგალითად მოსამენი აპარატურა, ან ინფორმაციის ფიზიკური მატარებლის მოტაცება) და სისტემის გარეთ (მაგალითად არხებიდან ინფორმაციის მიტაცება, ეკრანის მიერ გამოსხივებული ელექტრომაგნიტური ტალღების მიტაცება).
5. სისტემაზე ზემოქმედების მიხედვით შეტევა შეიძლება იყოს პასიური და აქტიური. პასიური შეტევის დროს ხდება მხოლოდ ინფორმაციის მიტაცება, რაც გავლენას ვერ ახდენს სისტემის მუშაობაზე. აქტიური შეტევის დროს ირღვევა სისტემის მუშაობის ჩვეული რეჟიმი.

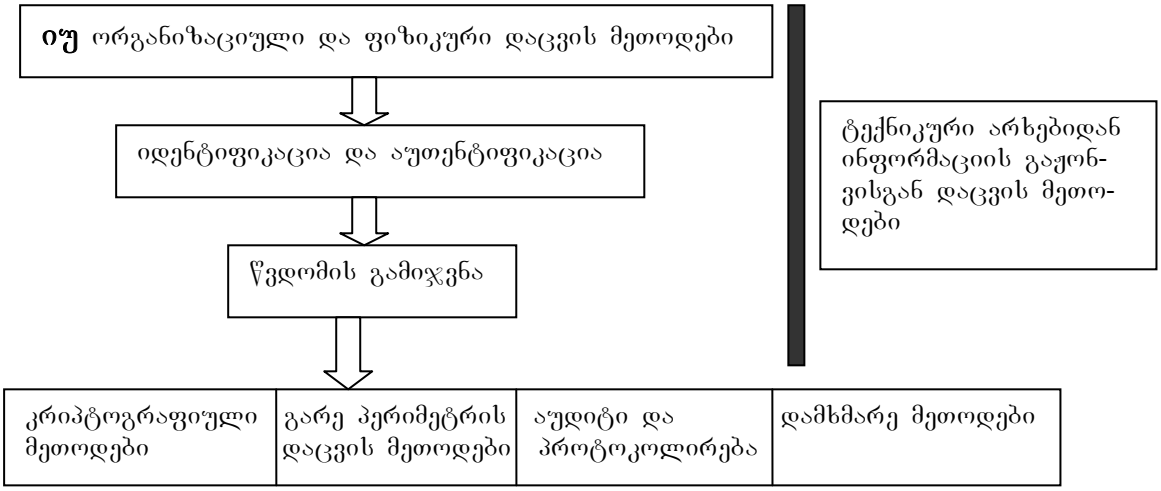
შეტევების კლასიფიკაცია აუცილებელი პროცედურაა, რომლითაც იწყება ინფორმაციული უსაფრთხოების დაცვის ქვესისტემის აგება. მხოლოდ მას შემდეგ, რაც მოხდება შეტევათა კლასიფიკაცია და განისაზღვრება ამა თუ იმ შეტევის შესაძლებელი ალბათობები, შეფასდება ის რისკები, რაც შეიძლება მოყვეს შეტევას, განისაზღვრება დაცვის ქვესისტემის ამოცანები და დაიწყება მუშაობა ქვესისტემის ასაგებად.



სურ. 8.1.ინფორმაციული უსაფრთხოების უზრუნველყოფის ძირითადი მეთოდები

8.3 კონფიდენციალურობაზე შეტევისაგან დაცვის სისტემის აგება. ამ შეტევისაგან თავის დაცვა მოითხოვს კომპლექსურ მიდგომას, რათა სისტემა, რომელსაც ავაგებთ, უზრუნველყოფდეს კონფიდენციალურობაზე ნებისმიერი შეტევისაგან დაცვას. კომპლექსური მიდგომის დროს გათვალისწინებული უნდა იქნეს შემდეგი პარამეტრები (იხ.სურ. 8.2).

როგორც ამ სქემიდან ჩანს, პირველ რიგში უნდა მოხდენს საინფორმაციო სისტემასთან ფიზიკური წვდომის ორგანიზება და გამოთვლითი საშუალებების ფიზიკური დაცვა. შემდეგ, ლოგიკური წვდომის ეტაპზე დაცვა ხორციელდება სხვადასხვა მექანიზმების საშუალებით. ამასთან პარალელურად უნდა ხორციელდებოდეს დაცვა ისეთი ტექნიკური საშუალებებით, რომლებიც გამორიცხავენ ინფორმაციის გაჟონვას ტექნიკური არხებიდან.



სურ. 8.2. კონფიდენციალურობის დაცვის ქვესისტემის აგების სქემა.

8.4. ორგანიზაციული და ფიზიკური დაცვის ზომები. ამ ეტაპზე აუცილებლად უნდა იქნას გათვალისწინებული შემდეგი ღონისძიებები:

- ავტომატიზებული სისტემის ელემენტებთან ფიზიკური წვდომის გამიჯვნა და კონტროლის მექანიზმების შექმნა;
- ფიზიკური დაცვის სამსახურის შექმნა;
- თანამშრომელთა გადაადგილებისათვის თვალყურის დევნის მექანიზმების განთავსება შენობაში (ვიდეოთვალი, პროქსიმიტი-კარტები და სხვა);
- თანამდებობრივი ინსტრუქციების და სამუშაო საათების რეგლამენტირების დოკუმენტების შექმნა და თანამშრომლებისათვის გაცნობა;
- კონფიდენციალური ინფორმაციის ფიზიკურ მატარებლებთან მუშაობის წესის რეგლამენტაცია.

უკვე ეს ღონისძიებები მათი კორექტული და ადექვატური გამოყენების შემთხვევაში წარმოადგენს უსაფრთხოების დაცვის საკმაოდ ეფექტურ მექანიზმს და აუცილებელია ნებისმიერი რეალური სისტემის სიცოცხლისუნარიანობისათვის.

8.5. იდენტიფიკაცია და აუთენტიფიკაცია. ***იდენტიფიკაციის*** ქვეშ იგულისხმება სისტემის მომხმარებლისთვის ისეთი უნიკალური ატრიბუტის მინიჭება, რომლის საშუალებითაც შესაძლებელი იქნება მისი გამორჩევა სხვა მომხმარებლებისაგან. ასეთ ატრიბუტს უწოდებენ ***იდენტიფიკატორს***. თავის მხრივ, ***აუთენტიფიკაცია*** წარმოადგენს მომხმარებლის მიერ წარმოდგენილი იდენტიფიკატორის შემოწმებას და ამ იდენტიფიკატორის ***ნამდვილობის დადასტურებას***. ამგვარად, შეიძლება ითქვას, რომ იდენტიფიკატორი საჭიროა იმისთვის, რომ გავიგოთ მომხმარებლის ვინაობა, ხოლო აუთენტიფიკაცია კი - დარწმუნდეთ, რომ მომხმარებელი ნამდვილად ისაა, ვისი სახელითაც შემოდის სისტემაში. იდენტიფიკაციისა და აუთენტიფიკაციის პროცესი აღიწერება შემდეგი სქემის სახით (იხ. სურ. 8.3.).

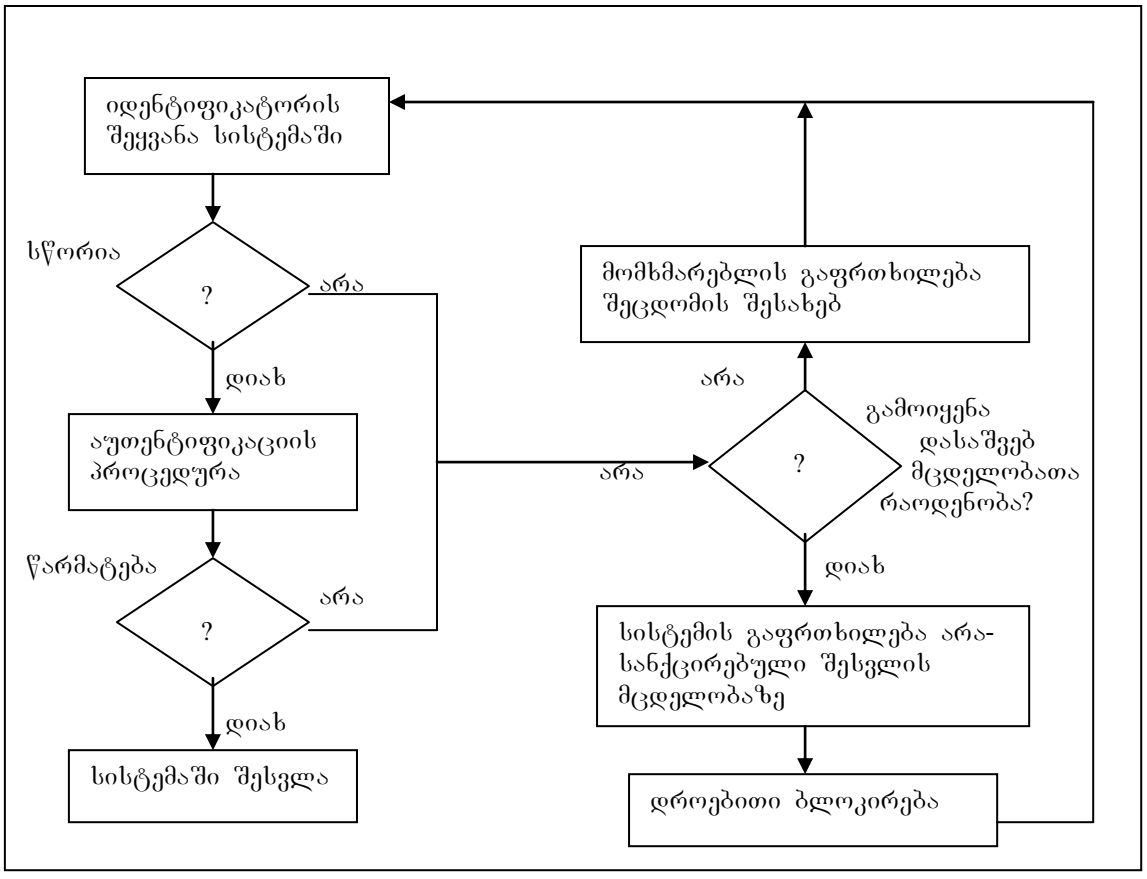
ეს სქემა ითვალისწინებს როგორც მომხმარებლის, ასევე ოპერატორის შესაძლო შეცდომებს, ამიტომ თუ შესვლათა მცდელობების რაოდენობა არ აღემატება წინასწარ დათქმულ რაოდენობას, მომხმარებელს შეუძლია ხელახლა სცადოს სისტემაში შესვლა. თუ შესვლათა რაოდენობა გადაჭარბებს დასაშვებ რაოდენობას, სისტემა დროებით ბლოკავს მომხმარებელს და ამავე დროს სისტემის ადმინისტრატორს აფრთხილებს არასანქცირებული შესვლის მცდელობის თაობაზე.

8.6. აუთენტიფიკაციის მეთოდები. აუტენტიფიკაციის დღეს ცნობილი ყველა მეთოდი შეიძლება დავყოთ ოთხ ძირითად ჯგუფად:

1. ***მეთოდები, დაფუძნებული რაიმე საიდუმლო ინფორმაციის ცოდნაზე.*** ასეთი მეთოდების კლასიკურ მაგალითს წარმოადგენს ***პაროლური სისტემა,*** როდესაც აუთენტიფიკაციის საშუალებას წარმოადგენს სიმბოლოთა რაღაც მიმდევრობა, რომელიც უნდა შეიყვანოს მომხმარებელმა სისტემაში.

2. ***მეთოდები, დაფუძნებული რაიმე უნიკალური ნივთის გამოყენებაზე.*** ასეთ ნივთად შესაძლებელია გამოყენებული იყოს მაგალითად სმარტ-ბარათი ან ელექტრონული გასაღები.
3. ***მეთოდები, დაფუძნებული ადამიანის ბიომეტრიულ პარამეტრებზე.*** პრაქტიკაში ყველაზე ხშირად ასეთ ბიომეტრიულ პარამეტრად გამოიყენება
    - თითის ანაბეჭდი;
    - თვალის გუგის ან ბადის ანაბეჭდი;
    - ხელის მტევნის სითბური სურათი;
    - ადამიანის სახის ფოტოგრაფია, ან სითბური სურათი;
    - ხელმოწერა;
    - ხმა.
4. ***მეთოდები დაფუძნებული რაიმე ინფორმაციაზე, რომელიც ასოცირდება მომხმარებელთან.*** ასეთი ინფორმაციის მაგალითს შეიძლება წარმპადგენდეს მომხმარებლის ბიოგრაფიის რაიმე ფაქტი, ან მაგალითად მისი კოორდინატები, განსაზღვრული GPშ-ის საშუალებით.

შესაძლებელია გამოყენებული იქნას ორი ან რამდენიმე მეთოდი ერთდროულადაც. ეს დამოკიდებულია იმაზე, თუ რამდენად მნიშვნელოვანია ის ინფორმაცის, რომელიც ინახება და მუშავდება მოცემულ საინფორმაციო სისტემაში.



სურ. 8.3. იდენტიფიკაციისა და აუთენტიფიკაციის ზოგადი სქემა.

8.7. პაროლური სისტემის თავისებურებები. დღეს ყველაზე ფართოდ გავრცელებულ მეთოდს წარმოადგენს პაროლური სისტემა, რაც განპირობებულია იმით, რომ ის არ მოითხოვს დამატებით დანახარჯებს, მომხმარებლისთვის მოსახერხებელი განსხვავებით მაგალითად თვალის ბადის ანაბეჭდის აღებისაგან და ტრადიციულია, ეს იყო პირველი მეთოდი, რომელიც გამოიყენეს მომხმარებლის იდენტიფიკაციისათვის.

ამ სისტემის გამოყენებას თან ახლავს ერთი პარადოქსი: რაც უფრო საიმედოა პაროლი, მით უფრო ძნელი გამოსყენებელია ის მომხმარებლისთვის. მართლაც ყველაზე საიმედო იქნება პარო-

ლი, რომელიც წარმოადგენს რაიმე სიმბოლოების შემთხვევით მიმდევრობას, მაგრამ ასეთი პაროლის დამახსოვრება ადამიანს პრაქტიკულად არ შეუძლია. ამიტომ მან ის უნდა ჩაწეროს სადმე, რაც შესაძლებელია გახდეს პაროლის გატეხვის მიზეზი. ამიტომ ადამიანი ცდილობს შექმნას ისეთი პაროლი, რომელსაც ის ადვილად დამიახსოვრებს. ეს კი უადვილებს მოწინააღმდეგეს მის გატეხვას. მოწინააღმდეგემ პაროლური სისტემის გასატეხად, ანუ პაროლის გასაგებად შეიძლება გამოიყენოს შეტევის შემდეგი მეთოდები:

1. ადამიანის ფაქტორის სისუსტეების გამოყენება. ეს მეთოდები გულისხმობენ თვალყურის დევნებას, მოსმენას, შანტაჟს, მუქარას და ასე შემდეგ.
2. პაროლის შერჩევის მეთოდები. ამ დროს შსეიძლება მოწინააღმდეგემ გამოიყენოს
    - სრული გადარჩევის მეთოდი;
    - შერჩევა ლექსიკონის დახმარებით. არსებობს ყველაზე ხშირად გამოყენებული
    - პაროლების ლექსიკონები, რომელთა გამოყენებას მნიშვნელოვნად ამცირებს სრულ გადარჩევას;
    - მომხმარებლის შესახებ ინფორმაციის გამოყენებით შერჩევის მეთოდი. ამ შემთხვევაში
3. გამოიყენება ისეთი ინფორმაცია მომხმარებლის შესახებ, როგორიცაა დაბადების დღე, შვილების, ან მეუღლის სახელები და ასე შემდეგ;

იდენტიფიკაციის პაროლური სისტემის რეალიზაციის დროს დაშვებული შეცდომები. ასეთ მეთოდებს მიეკუთვნება სისტემის ცალკეული კომპონენტების რეალიზაციის დროს დაშვებული შეცდომები.

8.8. რეკომენდაციები პაროლური სისტემის პრაქტიკულად ასაგებად. პაროლური სისტემის პრაქტიკულად აგების დროს პირველ რიგში აუცილებლად უნდა გათვალისწინებული იყოს საინფორმაციო სისტემის თავისებურებები და რისკების ჩატარებული ანალიზის შედეგები. თუმცა არსებობს ისეთი პრაქტიკული შედეგები, რომლებიც შეიძლება გამოვიყენოთ ნებისმიერი სისტემის აგების დროს. ასეთ რეკომენდაციებს მიეკუთვნება

- პაროლის მინიმალური სიგრძის დადგენა. აშკარაა, რომ ეს პარამეტრი გაურთულებს მოწინააღმდეგეს სრული გადარჩევის გზით პაროლის მოძებნას.

- პაროლის ალფაბეტის გაზრდა ალფაბეტში სხვადასხვა სიმბოლოების აუცილებლად გამოყენების გზით, რაც ასევე ართულება სრული გადარჩევის მეთოდით პაროლის მოძებნას.

- პაროლების შემოწმება და შერჩევა პაროლური ლექსიკონის დახმარებით. ეს გაურთულებს მოწინააღმდეგს ლექსიკონის დახმარებით პაროლის შსერჩევას.

- პაროლის გამოყენების მაქსიმალური დროის შერჩევა. ასეთი მაქსიმალური დროის დადგენა უზღუდავს მოწინააღმდეგეს დროს, რომლის განმავლობაშიც მან უნდა გატეხოს პაროლი.

- პაროლების ჟურნალის შექმნა. ეს საშუალებას იძლევა არ შევარჩიოთ პაროლად უკვე გამოყენებული პაროლები.

- პაროლის სისტემაში შეყვანის მცდელობების შეზღუდვა. ეს არ აძლევს მოწინააღმდეგეს საშუალებას, შეარჩიოს პაროლი ინტერაქტიურ რეჟიმში.

- მომხმარებლის იძულება, რათა მან შეცვალოს პაროლი სისტემაში პირველივე შესვლის დროს. თუ პირველად პაროლს ადგენს სისტემის ადმინისტრატორი ეს ნაბიჯი გამოიწვევს იმას, რომ ადმინისტრატორს არ ეცოდინება მომხმარებლის პაროლი.

- შეყოვნება არასწორი პაროლის შეყვანის დროს. ეს მექანიზმი ასევე არ აძლევს მოწინააღმდეგეს შეარჩიოს პაროლი ინტერაქტიურ რეჟიმში.

- აუკრძალოთ მომხმარებელს პაროლის შერჩევა და გამოვიყენოთ პაროლების ავტომატური გენერაციის მექანიზმი. ეს საუკეთესო საშუალებაა პაროლების მედეგობის გასაუმჯობესებლად, მაგრამ არ უნდა დაგვავიწყდეს, რომ ასეთ შემთხვევაში მომხმარებელს აუცილებლად გაუჩნდება პაროლის დამახსოვრების პრობლემა.

8.9. პაროლური სისტემის მედეგობის შეფასება. შევეცადოთ შევაფასოთ პაროლური სისტემის ძირითადი პარამეტრების ურთიერთკავშირი. შემოვიტანოთ აღნიშვნები:

A იყოს პაროლების ალფაბეტის სიმძლავრე;

L იყოს პაროლის სიგრძე; მაშინ $S = A^L$ იქნება შესაძლო პაროლების სიმრავლე.

V იყოს მოწინააღმდეგის მიერ პაროლების გადარჩევის სიჩქარე;

T იყოს პაროლის მოქმედების ვადა;

P იყოს ალბათობა იმისა, რომ მოწინააღმდეგე იპოვნის პაროლს მისი მოქმედების ვადის ამოწურვამდე. მაშინ ინტუიტურად შეგვიძლია ჩავთვალოთ, რომ ადგილი ექნება ტოლობას:

$$P = \frac{V \cdot T}{S}$$

ჩვენთვის ცნობილია, თუ რა ტექნიკური საშუალებები გააჩნია მოწინააღმდეგეს, ამიტომ შეგვიძლია ჩავთვალოთ, რომ ჩვენთვის ცნობილია პაროლების გადარჩევის სიჩქარე. აქედან გამომდინარე, ჩვენ შეგვიძლია განვსაზღვროთ პაროლის მოქმედების ვადა. ამ შემთხვევაში თუ ავიღებთ ჩვენთვის სასურველ P -ს მნიშვნელობას (სისტემის დაცულობის დონეს), შეგვიძლია დავადგინოთ შესაძლო პაროლების სიმრავლე, ანუ გამოვთვალოთ ალფაბეტში სიმბოლოების რაოდენობა და პაროლის სიგრძე.

8.10. პაროლების შენახვისა და ქსელში გადაცემის მეთოდები. პაროლები სისტემაში შეიძლება ინახებოდეს ღია სახით, დაშიფრული და გამოთვლილი ჰეშ-ფუნქციის მნიშვნელობის სახით. ცხადია, რომ ღია სახით შენახვა არაა გამართლებული, რადგანაც იშვიათად გჭირდება პაროლის ღია სახით გამოყენება. ამავე დროს ასეთი სახით შენახვა ქმნის იმის საშიშროებას, რომ მოწინააღმდეგემ თუ იპოვნა სად ინახება პაროლები, ის ერთდროულად მიიღებს ყველა მომხმარებლის პაროლს. ამიტომ პაროლების ღია სახით შენახვა სისტემაში არ შეიძლება.

პაროლები სისტემაში შეიძლება ინახებოდეს დაშიფრული სახით. ამისათვის შესაძლებელია გამოვიყენოთ დაშიფრვის რომელიმე ბლოკური ალგორითმი და დაშიფრვის გასაღები შეიძლება ინახებოდეს სისტემის რომელიმე ელემენტზე (რაც არაა საუკეთესო ვარიანტი), ან ინფორმაციის რომელიმე ფიზიკურ მატარებელზე (მაგალითად სმარტ-ბარათზე), ან ხდებოდეს გასაღების გენერირება სისტემის რომელიმე სხვა პარამეტრის საშუალებით (მაგალითად, ხდებოდეს გასაღების გენერირება ადმინისტრატორი პაროლიდან სისტემის ინიციალიზაციის დროს).

ყველაზე უფრო მოსახერხებელია სისტემაში ინახებოდეს პაროლების ჰეშ-ფუნქციები, რადგანაც თუნდაც რომ ჩაიგდოს ხელში მოწინააღმდეგემ ამ ჰეშ-ფუნქციათა მნიშვნელობები, ის მაინც ვერ შეძლებს პაროლის აღდგენას.

პაროლი ქსელში შეიძლება გადავცეთ ღია სახით, რაც ისევე, როგორც შენახვა, არაა რეკომენდირებული. პაროლი შეიძლება გადავცეთ ჰეშ-ფუნქციის მნიშვნელობის სახით, მაგრამ ასეთ შემთხვევაში ვერც მომხმარებელი ვერ შეძლებს პაროლის აღდგენას ჰეშ-ფუნქციის მნიშვნელობიდან, და პაროლი შეიძლება გადავცეთ ქსელში დაშიფრული სახით. ეს უკანასკნელი წარმოადგენს ყველაზე უფრო საიმედო გზას, რადგანაც მოწინააღმდეგე ვერ შეძლებს მის გაშიფრვას.

8.11 წვდომის გამიჯვნა. წვდომის გამიჯვნის ქვეშ ჩვენ გვესმის სუბიექტებისათვის უფლება-მოსილებათა განსაზღვრა, რათა შემდეგ ვაკონტროლოთ სისტემაში არსებული რესურსების სანქცირებული გამოყენების პროცესი. არსებობს წვდომის გამიჯვნის ორი მეთოდი ***დისკრეციული*** და ***მანდატური***.

დისკრეციული ეწოდება ისეთ გამიჯვნას, რომლის დროსაც გამიჯნულია კონკრეტული სუბიეტების წვდომა კონკრეტულ ობიექტებთან, ანუ ყველა სუბიეტისათვის ჩამოთვლილია ობიექტები, რომელთანაც სუბიექტს გააჩნია წვდომა. ასეთი გამიჯვნა შეიძლება განვახორციელოთ ***წვდომის მატრიცის*** საშუალებით. ასეთ შემთხვევაში იქმნება მატრიცა, რომლის სტრიქონებში ჩამოთვლილია სისტემის მომხმარებლები, ხოლო სვეტებში კი სისტემის რესურსები სტრიქონისა და სვეტების გადაკვეთაზე კი მითითებულია გააჩნია თუ არა მოცემულ სუბიექტს წვდომა მოცემულ რესურსთან.

წვდომის მანდატური გამიჯვნა როგორც წესი ხორციელდება როგორც წვდომის გამიჯვნა საიდუმლოების დონეებით. ამ დროს ხდება სისტემის ყველა რესურსის კლასიფიცირება საიდუმლოების დონეების მიხედვით, სუბიექტებს კი ენიჭებათ ის საიდუმლოების მაქსიმალური დონე, რომელთანაც მათ აქვთ წვდომის უფლება.

განსხვავება ამ ორ მეთოდს შორის მდგომარეობს შემდეგში: თუ წვდომის დისკრეციული გამიჯვნის დროს რესურსთან წვდომის უფლებას განსაზღვრავს ინფორმაციის მფლობელი, მანდატური წვდომის დროს საიდუმლოების დონეები მოიცემა გარედან და ინფორმაციის მფლობელს არ შეუძლია გავლენა იქონიოს ამ პროცესზე. ტერმინი "მანდატური" წარმოიშვა ინგლისური სიტყვიდან "**mandatiry**", რაც ნიშნავს აუცილებელს.

საკონტროლო კითხვები:

1. რას წარმოადგენს ინფორმაციის ავტომატიზებული დამუშავების სისტემა?

2. დაახასიათეთ ინფორმაციული უსაფრთხოების ძირითადი მეთოდები.
3. რა ტიპის შეტევები შეიძლება განხორციელდეს ინფორმაციულ უსაფრთხოებაზე?
4. რა პირობებს უნდა აკმაყოფილებდეს ინფორმაციის კონფიდენციალობაზე შეტევისაგან დაცვის ქვესისტემა?
5. რა ზომებს ითვალისწინებს ორგანიზაციული და ფიზიკური დაცვის ქვესისტემა? როგორია მისი ეფექტიანობა?
6. რას ნიშნავს მომხმარებლის იდენტიფიკაცია და აუთენტიფიკაცია? რისთვისაა საჭირო ეს პროცედურები?
7. აუთენტიფიკაციის რა მეთოდებია დღეს ცნობილი და რომელი მათგანი გამოიყენება ყველაზე უფრო ხშირად? რატომ?
8. რა თავისებურებებით ხასიათდება აუთენტიფიკაციის პაროლური სისტემა?
9. რა ზოგადი სახის რეკომენდაციები შეიძლება იყოს გათვალისწინებული პაროლური სისტემის აგების დროს?
10. როგორ შეიძლება შევაფასოთ პაროლური სისტემის მედეგობა?
11. პაროლების შენახვისა და გადაცემის რა მეთოდებია დღეს ცნობილი და რომელი ყველაზე საიმედო?

ზურგჩანთის ამოცანაზე დაფუძნებული ღია გასაღებიანი კრიპტოსისტემა

9.1. ზურგჩანთის ამოცანა. ინფორმაციის დაშიფრვის ერთერთი პირველი ღია გასაღებიანი სისტემა შემოგვთავაზეს მერკლმა და ჰელმანმა. ეს ალგორითმი დაფუძნებული იყო ალგორითმების თეორიაში კარგად ცნობილ ზურგჩანთის ამოცანაზე, რომელიც ზოგადი სახით შეიძლება ჩამოვაყალიბოთ შემდეგი სახით: მოცემული გვაქვს ნატურალურ რიცხვთა სიმრავლე $W = \{w_1, w_2, ..., w_n\}$ და კიდევ ერთი S ნატურალური რიცხვი. ამოცანა მდგომარეობს შემდეგში: შესაძლებელია თუ არა S წარმოვადგინოთ როგორც W-ს რაიმე ქვესიმრავლის ჯამი:

$$S = \sum_{i=1}^{n} x_i w_i \quad , \quad (9.1.)$$

სადაც x_i-ები ღებულობენ მნიშვნელობებს $\{0,1\}$ სიმრავლიდან (უფრო მეტი სიზუსტისათვის უნდა აღვნიშნოთ, რომ ალგორითმების თეორიაში ეს ამოცანა ცოტა უფრო განსხვავებული სახით განიხილება, ხოლო აქ ჩამოყალიბებული ამოცანა, რომელიც განიხილება კრიპტოგრაფიაში, არის ზოგადი ამოცანის კერძო შემთხვევა). სახელი ამოცანამ მიიღო შემდეგი ინტერპრეტაციის გამო. დავუშვათ, S არის ზურგჩანთის მაქსიმალური წონა, რომელიც შეიძლება ატაროს კონკრეტულმა პიროვნებამ, ხოლო W სიმრავლე იმ ნივთების წონებისა, რომლებიც გვინდა ჩავაწყოთ ამ ზურგჩანთაში. ჩვენი მიზანია, რაც შეიძლება მაქსიმალურად შევავსოთ ზურგჩანთა, ანუ გამოვთვალოთ x_i-ის ის მნიშვნელობები, რომლებისთვისაც შესრულდება (9.1.) ტოლობა. მაგალითად, თუ

$$W = \{43,129,215,473,903,302,561,1165,697,1523\} \text{ და } S = 3231.$$

მაშინ, $3231 = 129 + 473 + 903 + 561 + 1165$ და ჩვენ ვიპოვეთ ამ ამოცანის ამოხსნა. ამისათვის საჭირო გახდა ყველა შესაძლო მნიშვნელობების გადარჩევა. (W, S) წყვილს უწოდებენ შესასვლელს, ხოლო ამოხსნას W-ს ისეთ ქვესიმრავლეს, რომლის ელემენტების ჯამიც უდრის S. ზოგადად, ამოცანა მიეკუთვნება **NP** კლასს, თუმცა არსებობენ ამ ამოცანის ვარიანტები, რომლებიც არ მიეკუთვნებიან **NP** კლასს. ამას ჩვენ მალე ვნახავთ.

9.2. დაშიფრვის პროცედურა. W ვექტორის გამოყენებით M შეტყობინების დაშიფრვა ხდება შემდეგნაირად. M შეტყობინება (ჩაწერილი ორობითი სახით), უნდა დავყოთ ბლოკებად,

რომელთა სიგრძე ტოლია რიცხვების რაოდენობისა W სიმრავლეში და ჩავთვალოთ, რომ ისინი წარმოადგენენ x_i რიცხვების მნიშვნელობებს. შევკრიბოთ შესაბამისი w_i-ები და მიღებული ჯამი, ჩაწერლი ორობითი სახით, იქნება დაშიფრული ინფორმაცია. მაგალითად, თუ გასაღები იქნება ზემოთმოყვანილი W ვექტორი, ხოლო დასაშიფრ ინფორმაციას აქვს შემდეგი სახე:

$$0100110010\ \ 1011010010\ .$$

დასაშიფრი ინფორმაცია გაიყოფა ორ ბლოკად და დაშიფრვის პროცესს ექნება შემდეგი სახე:

$$S_1 = 0\cdot 43+1\cdot 129+0\cdot 215+0\cdot 473+1\cdot 903+1\cdot 302+0\cdot 561+0\cdot 1165+1\cdot 697+0\cdot 1523 = 2031$$

$$S_2 = 1\cdot 43+0\cdot 129+1\cdot 215+1\cdot 473+0\cdot 903+1\cdot 302+0\cdot 561+0\cdot 1156+1\cdot 697+0\cdot 1523 = 1730$$

და შესაბამისად მიღებული შიფროტექსტი იქნება $(2031,1730)$. ეხლა გასაგებია, რომ ზურგჩანთის ამოცანას კრიპტიგრაფიაში მართლაც ყოველთვის აქვს ამოხსნა.

როგორც ვხედავთ, W ვექტორი გამოიყენება როგორც დაშიფრვის გასაღები. ეხლა თუ ამ დაშიფრულ ტექსტს ანი გაუგზავნის ბექას, მაშინ ბექამ უნდა იცოდეს დაშიფრვის გასაღები, რათა შეძლოს შიფროგრამის დეშიფრაცია. ჩვენ მივიღეთ ჩვეულებრივ ერთგასაღებიან კრიპტოსისტემა, რომელშიც შიფროგრამის დეშიფრაციისათვის ბექას მოუწევს ზურგჩანთის **NP** სირთულის ამოცანის ამოხსნა. თუ ეკას ექნება შესაძლებლობა, რომ შეუტიოს ამ ალგორითმს შერჩეული ღია ტექსტის საფუძველზე, მას დასჭირდება ზუსტად n ცალი შერჩეული ტექსტი, რათა გაიგოს გასაღები, მაგრამ ამის შემდეგ მასაც, როგორც ბექას, მოუწევს **NP** სირთულის ამოცანის ამოხსნა.

ასეთ დაშიფრვას აქვს კიდევ ერთი სერიოზული ნაკლი. მოცემული W ვექტორისათვის ყველა კრიპტოტექსტს შეესაბამება მხოლოდ ერთი ღია ტექსტი, მაგრამ, თუ კრიპტოტექსტი იქნება

$$S = 515 \text{ და } W = \{14,28,56,82,90,132,197,284,341,455\},$$

მაშინ ამ კრიპტოტექსტს შეესაბამება ზუსტად სამი ღია ტექსტი

$$(1,1,0,0,0,1,0,0,1,0),\quad (0,1,1,0,0,1,0,0,1,0) \text{ და } (1,0,0,1,1,1,1,0,0,0).$$

თავისთავად ცხადია, რომ კრიპტოგრაფიაში ასეთი ვექტორების გამოყენებას აზრი არა აქვს, რადგანაც შეუძლებელი იქნება შიფროგრამის ცალსახა დეშიფრაცია. ამიტომ ვექტორი ისეთი უნდა იყოს, რომ ნებისმიერი S-თვის (W,S) ამოცანას უნდა გააჩნდეს ერთადერთი ამონახსნი. ზურგჩანთის ამოცანისათვის აღმოჩნდა, რომ გარკვეული კლასის W ვექტორებისათვის არა მარტო არსებობს ერთადერთი ამონახსნი, არამედ არსებობს ასეთი ამონახსნის პოვნის მარტივი, წრფივი სირთულის ალგორითმი. ასეთ ვექტორებს უწოდებენ ზეზრდად მიმდევრობას.

<u>განსაზღვრა 9.1</u>. *მიმდევრობას ეწოდება ზეზრდადი მიმდევრობა, თუ კმაყოფილდება პირობა:*

$$w_i > \sum_{j=1}^{i-1} w_j\ . \quad (9.2)$$

მაგალითად, მიმდევრობა $\{2,3,6,13,27,52\}$ წარმოადგენს ზეზრდად მიმდევრობას. ასეთი მიმდევრობისათვის ზურგჩანთის ამოცანის ამოხსნის ალგორითმი ძალიან მარტივია. უნდა ავიღოთ მიმდევრობის ყველაზე დიდი წევრი და შევადაროთ ზურგჩანთის წონას. თუ ეს წევრი ნაკლებია ან ტოლი ზურგჩანთის მოცულობაზე, მას დებენ ზურგჩანთაში. წინააღმდეგ შემთხვევაში ნივთი არ შეიძლება ჩავდოთ ზურგჩანთაში. თუ ნივთი ვერ ჩავდეთ ზურგჩანთაში, მაშინ გადავიდეთ შემდეგ ნივთზე. წინააღმდეგ შემთხვევაში გამოვაკლოთ ზურგჩანთის წონას ჩადებული ნივთის წონა და პროცედურა გავიმეოროთ შემდეგი ნივთისათვის. განვიხილოთ მაგალითი. ზემოთმოყვანილი ზეზრდადი მიმდევრობის შემთხვევაში, დავუშვათ ზურგჩანთის წონაა 70. რადგანაც ყველაზე დიდი ნივთის წონა ნაკლებია ზურგჩანთის წონაზე, ამ ნივთს დებენ ზურგჩანთაში. დაგვრჩება $70-52=18$. რადგანაც მეორე ნივთის წონა მეტია დარჩენილ წონაზე, მას არ დებენ ზურგჩანთაში. მესამე ნივთის წონა ნაკლებია ზურგჩანთის დარჩენილ წონაზე, ამიტომ მას დებენ ზურგჩანთაში. ამის შემდეგ ზურგჩანთაში დაგვრჩება $18-13=5$ თავისუფალი წონა, ამიტომ შემდეგ ნივთს არ დებენ ზურგჩანთაში ხოლო დარჩენილი ორი ნივთი ზუსტად შეავსებს ზურგჩანთას, მივიღეთ რომ ეს მიმდევრობა წარმოადგენს ამონახსნს. აშკარაა, რომ ასეთი მიმდევრობის შემთხვევაში ნებისმიერ წინასწარ დაფიქსირებულ S-ის მნიშვნელობას ყოველთვის შეესაბამება W-ს ერთარდერთი ქვესიმრავლე, რადგანაც, თუ ნივთი, რომელიც ეტევა ზურგჩანთაში, არ ჩავდეთ ზურგჩანთაში, მასზე პატარა მოცულობის ნივთები ვერ მოგვცემენ სასურველ შედეგს. ეს კი იმას ნიშნავს, რომ შეუძლებელი იქნება სხვადასხვა რიცხვების ჯამებმა მოგვცეს ერთი და იგივე რიცხვი, ანუ დეშიფრაციის პროცესი იქნება ყოველთვის ცალსახა.

9.3. მერკლი-ჰელმანის ალგორითმი. ცხადია ის ფაქტი, რომ ვიპოვეთ დეშიფრაციის მარტივი და სასურველი თვისებების მქონე პროცედურა, არაა საკმარისი იმისათვის, რომ სისტემას ვუწოდოთ ღია გასაღებიანი კრიპტოსისტემა. თუ ანი გამოიყენებს ასეთ მიმდევრობას გადასაცემი შეტყობინების დასაშიფრად, ჩვენ მივიღებთ ჩვეულებრივ სიმეტრიულ კრიპტოალგორითმს, რადგანაც იმისათვის, რომ ბექამ გაშიფროს ანის შეტყობინება, მან უნდა იცოდეს ის მიმდევრობა, რომელიც გამოიყენა ანიმ, მაგრამ თუ ამ მიმდევრობას გადავცემთ ღია არხით, მაშინ ეკაც ისევე ადვილად გაშიფრავს კრიპტოგრამას, როგორც ბექა. მერკლმა და ჰელმანმა ასეთი ზეზრდადი მიმდევრობა გამოიყენეს როგორც დახურული გასაღები, ხოლო ღია გასაღების მისაღებად ზეზრდადი მიმდევრობის თითოეული წევრზე ჩაატარეს მოდულით გამრავლების ოპერაცია. ასეთი გამრავლების შედეგად მიიღება ჩვეულებრივი მიმდევრობა, რომელიც იქნება კრიპტოსისტემის ღია გასაღები. ინფორმაცია დაიშიფრება ღია გასაღებით, მაგრამ მის გასაშიფრად საჭიროა დახურული გასაღები, რაც იმას ნიშნავს რომ სისტემის გასატეხად საჭიროა ვიცოდეთ, თუ როგორ მიიღება დახურულიდან ღია გასაღები, რათა შევძლოთ შებრუნებული პროცედურის ჩატარება. დავუშვათ, ანიმ დახურულ გასაღებად ამოირჩია ზეზრდადი მიმდევრობა

$$W = \{1,3,5,11,21,44,87,175,349,701,1399,2797\}.$$

ამის შემდეგ მან უნდა შეარჩიოს მამრავლი k და მოდულის ფუძე m პირობით $k < m$. ცხადია, რომ k და m ისე უნდა შეირჩეს, რომ

$$\text{უ.ს.გ.}(k,m) = 1 .$$

ეს მოგვცემს იმის გარანტიას, რომ იარსებობს k-ს შებრუნებული k^{-1} ისეთი, რომ $kk^{-1} \equiv 1 \bmod m$ და $1 \le k^{-1} \le m$. რაც შეეხება m-ს, ანიმ ის შეიძლება აირჩიოს პირობით, $m > \max w_i$ და გამოთვალოს

$$b_i \equiv kw_i \bmod m .$$

დავუშვათ, ანიმ აირჩია $k = 53$ და $m = 3001$. მაშინ W ზეზრდადი მიმდევრობიდან ის მიიღებს არაზეზრდად მიმდევრობას

$$B = \{53,159,265,583,1113,2332,1610,272,491,1141,2123,1192\}$$

ასეთ შემთხვევაში ამბობენ, რომ B მიიღება მოდულარული გამრავლებით W-დან (k,m)-ს მიმართ. ამის შემდეგ ანის შეუძლია ღია გასაღებად გამოაცხადოს B მიმდევრობა, რომლის საშუალებითაც ბექა დაშიფრავს შეტყობინებას და გამოუგზავნოს ანის. ანი მიიღებს რა კრიპტოგრამას, გარდაქმნის მას მისთვის ცნობილი k^{-1} და m-ის საშუალებით და გაშიფრავს მას.

პირობები $kk^{-1} \equiv 1 \bmod m$ და $1 \le k^{-1} \le m$

იძლევიან იმის გარანტიას, რომ შესაბამისად W-ც მიიღება მოდულარული გამრავლებით B-დან (k^{-1},m)-ის მიმართ (ცხადია, რომ ყოველი $b_i < m$, რადგანაც ისინი გამოითვლება m-ს მოდულით). ეს ნიშნავს იმას, რომ ეკას გაუჩნდება შანსი მოძებნოს ეს გარდაქმნა და მანაც გაშიფროს ბექას მიერ გადაცემული კრიპტოგრამა. თუ ანი მოდულის ფუძედ აირჩევს ისეთ m-ს, რომ

$$m > \sum_{i=1}^{n} w_i$$

და კვლავ იგივე პროცედურების საშუალებით გამოთვლის B მიმდევრობას, მაშინ ამბობენ, რომ B მიიღება W-გან ძლიერი მოდულარული გამრავლებით (k,m)-ის მიმართ. ამ შემთხვევაში უკვე აღარ შეიძლება ვამტკიცოთ, რომ W მიიღება B-გან ძლიერი მოდულარული გამრავლებით (k,m)-ის მიმართ. რადგანაც ამ შემთხვევაში უტოლობა

$$m > \sum_{i=1}^{n} b_i \qquad (9.3)$$

შეიძლება აღარ სრულდებოდეს, ეკას გაუჭირდება აღაგინოს ზეზრდადი მიმდევრობა ღია გასაღებიდან. აქედან გამომდინარე, ცხადია ანი B მიმდევრობის გამოსათვლელად ამოირჩევს ძლიერი მოდულარული გამრავლების მეთოდს.

დავუშვათ, ანიმ აირჩია $k = 1003$ და $m = 5597$. მაშინ ანი მიიღებს უკვე სხვა მიმდევრობას

$$B = \{1003,3009,5015,5436,4272,4933,3306,2018,3033,3478,3947,1294\},$$

სადაც აღარ სრულდება (9.3) უტოლობა. ჩაატარებს რა ასეთ გამოთვლებს ანი მიღებულ მიმდევრობას გამოაცხადებს ღია გასაღებად, დახურული გასაღები კი იქნება სამეული (W,k,m).

ვნახოთ როგორ მუშაობს ასეთი კრიპტოსისტემა. დავუშვათ, მოცემული გვაქვს ღია ტექსტი "ბათუმში წვიმს." როგორც ადრე ვაკეთებდით, ქართულ ასოებს შევუსაბამოთ რიცხვები ნოლიდან ოცდათორმეტის ჩათვლით, ოღონდ ეს რიცხვები წარმოვადგინოთ ორობით სისტემაში. ამისათვის დაგვჭირდება სულ ექვსი ბიტი. (იხ. ცხრილი 9.1.).
ცხრილი 9.1.

ჰარი	0	000000	ა	1	000001	ბ	2	000010	გ	3	000011	დ	4	000100
ე	5	000101	ვ	6	000110	ზ	7	000111	თ	8	001000	ი	9	001001
კ	10	001010	ლ	11	001011	მ	12	001100	ნ	13	001101	ო	14	001110
პ	15	001111	ჟ	16	010000	რ	17	010001	ს	18	010010	ტ	19	010011
უ	20	010100	ფ	21	010101	ქ	22	010110	ღ	23	010111	ყ	24	011000
შ	25	011001	ჩ	26	011010	ც	27	011011	ძ	28	011100	წ	29	011101
ჭ	30	011110	ხ	31	011111	ჯ	32	100000	ჰ	33	100001			

ამ ცხრილის მიხედვით ღია ტექსტს ორობით ჩანაწერში ექნება შემდეგი სახე:

000010 000001 001000 010100 001100 011001 001001 011101 000110 001001 001100 010010.

რადგანაც ჩვენი ღია გასაღები (B სიმრავლე) შეიცავს თორმეტ ელემენტს, ამიტომ კოდირებული ტექსტი უნდა დავყოთ თორმეტი ბიტის სიგრძის ბლოკებად (თუ საჭირო გახდება ბლოკის შესავსებად ბოლოში ემატება ნოლები). ჩავთვალოთ ეხლა, რომ კოდირებული ტექსტის თორმეტი ბიტი შეესაბამება x_i-ის მნიშვნელობებს და გამოვთვალოთ ჯამები (9.1.) ფორმულით, მივიღებთ

$$S_1 = 5566,\quad S_2 = 10511,\quad S_3 = 16796,\quad S_4 = 19771,\quad S_5 = 14035,\quad S_6 = 16416.$$

ეს ექვსი რიცხვი იქნება შიფროტექსტი, რომელსაც ბექა გაუფზავნის ანის.

9.4. დეშიფრაცია. იმისათვის, რომ ანიმ გაშიფროს ბექას მიერ გამოგზავნილი კრიპტოტექსტი, პირველ რიგში მას უნდა გააჩნდეს k-ს შებრუნებული k^{-1} რიცხვი. ევკლიდეს გაფართოებული ალგორითმით ადვილი გამოსათვლელია, რომ $k = 1003$ შებრუნებული მოდულით $m = 5597$ იქნება $k^{-1} = 5145$. ეკა თითოეულ ჯამზე ატარებს შემდეგ პროცედურას: ამრავლებს მას k^{-1} მოდულით m და მიღებულ რიცხვს წარმოიდგენს რა როგორც ზურგჩანთის ზომას, ამოხსნის ზურგჩანთის ამოცანას ზეზრდადი მიმდევრობის საშუალებით. გავშიფროთ ამ მეთოდით ანის მიერ მიღებული კრიპტოტექსტი.

$$S_1' = S_1 k^{-1} (\mathrm{mod}\, m) = 5566 \cdot 5145 (\mathrm{mod}\, 5597) = 28637070 (\mathrm{mod}\, 5597) = 2818.$$

რადგანაც W ზეზრდადი მიმდევრობის მეთორმეტე წევრი $2797 < 2818$, დეშიფრირებული ტექსტის მეთორმეტე ბიტი იქნება ერთის ტოლი. გამოვაკლოთ S_1' მეთორმეტე წევრის მნიშვნელობა: $S_1'' = 2818 - 2797 = 21$. W მიმდევრობის მეთერთმეტე, მეათე, მეცხრე, მერვე, მეშვიდე და მეექვსე წევრების მნიშვნელობები მეტია S_1''-ზე, ამიტომ დეშიფრირებული ტექსტის შესაბამისი ბიტები იქნება ნოლის ტოლი. მეხუთე წევრის მნიშვნელობა ტოლია 21, ამიტომ მეხუთე ბიტის მნიშვნელობა ტოლია ერთის. გამოვაკლოთ მეხუთე წევრის მნიშვნელობა S_1''-ს. მივიღებთ ნოლს, ამიტომ პირველი, მეორე, მესამე და მეოთხე ბიტების მნიშვნელობები იქნება ნოლი. საბოლოოდ მივიღეთ ღია ტექსტის პირველი ბლოკი 000010 000001.

$$S_2' = 811;\quad S_3' = 3337;\quad S_4' = 1917;\quad S_5' = 4178;\quad S_6' = 1590.$$

მეორე ჯამის დეშიფრაცია მოგვცემს: 001000 010100. ანალოგიურად გაიშიფრება დანარჩენი ჯამებიც.

ამგვარად, თუ მოცემული გვაქვს W ზეზრდადი მიმდევრობა და B მიმდევრობა, რომელიც მიიღება მიიღება W-გან ძლიერი მოდულარული (k, m) გამრავლებით, რომლებიც აკმაყოფილებენ ზემოთმოყვანილ პირობებს და ამასთან ერთად მოცემული გვაქვს რაიმე ნატურალური რიცხვი y და $S = k^{-1} y (\mathrm{mod}\, m)$, შეგვიძლია დავამტკიცოთ, რომ

- (W, S) შესასვლელისათვის არსებობს ზურგჩანთის ამოცანის ამოხსნის წრფივი ალგორითმი;

- თუ ამოხსნა არსებობს, ის ერთადერთია. (B, y) შესასვლელისათვის ამოცანას შეიძლება ჰქონდეს არაუმეტეს ერთი ამოხსნისა და
- თუ არსებობს ამოხსნა (B, y) შესასვლელისათვის, ის აუცილებლად დაემთხვევა (W, S) შესასვლელის ამოხსნას.

პრაქტიკული გამოყენებისათვის მიმდევრობაში უნდა იყოს 200-დან 300 ელემენტამდე, რომელთა ზომები იქნება 200-დან 400 ბიტამდე. უხეში ძალის გამოყენებით ასეთი კრიპტოსისტემის გატეხვა იქნება შეუძლებელი. იმ შემთხვევაშიც კი, თუ კომპიუტერი შეძლებს მილიონი ვარიანტის შემოწმებას წამში, ყველა შესაძლო ვარიანტის შემოწმებას დასჭირდება 10^{46} წელზე მეტი.

9.6. მერკლი-ჰელმანის ალგორითმის კრიპტოანალიზი. ამ სისტემის გატეხვას არ დასჭირდა არც სუპერკომპიუტერები, რომლებიც წამში შეასრულებდნენ მილონ ოპერაციას და არც რამდენიმე ათეული წელი. გამოქვეყნებიდან სულ რაღაც ორ წელიწადში ა. შამირმა გამოაქვეყნა ამ ალგორითმის გატეხვის პოლინომური ალგორითმი, რომელიც საშუალებას იძლევა აღვადგინოთ ღია გასაღებიდან საიდუმლო გასაღები.

სანამ განვიხილავდეთ ა. შამირის შეტევას, გავაკეთოთ ერთი შენიშვნა. განსხვავებით სიმეტრიულ ალგორითმებზე შეტევებისაგან, სადაც კრიპტოანალიტიკოსს შეტევის განსახორციელებლად აუცილებლად სჭირდება შიფროგრამების მიტაცება ღია არხიდან, ღია გასაღებიანი კრიპტოალგორითმების შემთხვევაში ეს თითქმის აღარ არის საჭირო. მართლაც, კრიპტოანალიტიკოსის ხელშია ალგორითმი და ღია გასაღები, ამიტომ მას შეუძლია თვითონ დაშიფროს ნებისმიერი ღია ტექსტი და შეუტიოს სისტემას. ეს ერთი შეხედვით აადვილებს კრიპტოანალიტიკოსის შრომას, მაგრამ რადგანაც ღია გასაღებიდან საიდუმლო გასაღების მისაღებად საჭიროა არაპოლინომური ამოცანის ამოხსნა, ეს გაადვილება სინამდვილეში მოჩვენებითია.

ა.შამირმა მერკლი-ჰელმანის ალგორითმი გატეხა ადვილად იმიტომ, რომ

- ჯერ ერთი, როგორც ზევით იყო აღნიშნული, ნებისმიერი ზეზრდადი მიმდევრობა, რომლიდანაც მიიღება მიმდევრობა მოდულით რაიმე რიცხვზე გამრავლების გზით გამოდგება საიდუმლო გასაღების როლში, ანუ აუცილებელი არაა ვიპოვოთ სწორედ ის ზეზრდადი მიმდევრობა, რომელიც ანიმ გამოიყენა საიდუმლო გასაღებად;
- მთავარი კი იმაში მდგომარეობს, რომ მერკლი-ჰელმანის ალგორითმში ღია გასაღები საიდუმლო გასაღებიდან სულაც არ მიიღება ცალმხრივი ფუნქციის საშუალებით, რადგანაც მოდულით რიცხვზე გამრავლების ოპერაცია არაა ცალმხრივი ფუნქცია. სწირედ ეს აჩვენა ა. შამირმა თავის სტატიაში, რომელშიც აღწერა ღია გასაღებიდან საიდუმლო გასაღების გამოთვლის პროცედურა.

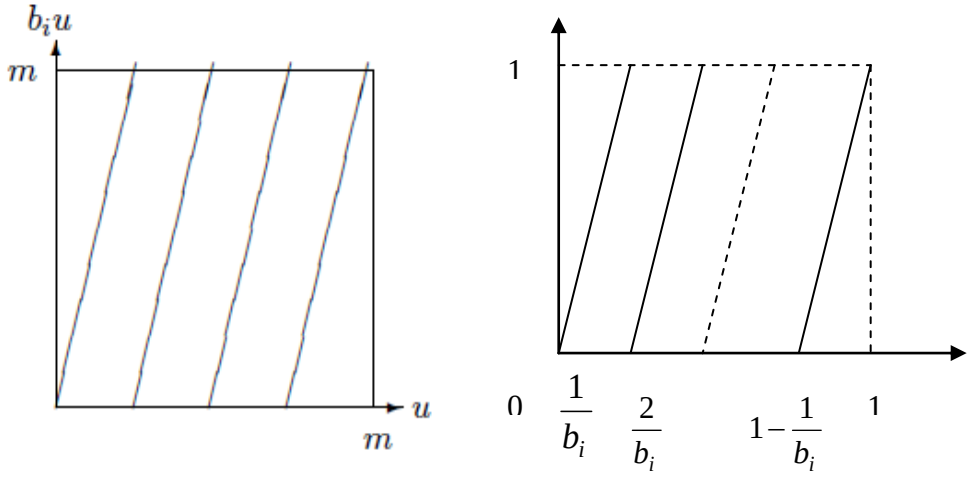
შევხედოთ ზურგჩანთის სისტემას ეკას თვალით. ეკამ იცის B მიმდევრობა, იცის, რომ ის მიღებულია ზეზრდადი W მიმდევრობიდან ძლიერი გამრავლებით რომელიღაც (k_0, m_0) წყვილის მიმართ, მაგრამ არ იცის არც k და არც m. სამაგიეროდ აქვს ბევრი დრო და შეუძლია წინასწარ, სანამ მიიტაცებს კრიტპოგრამას, მანამდე იფიქროს, როგორ გამოთვალოს ღია გასაღებიდან საიდუმლო გასაღები. როგორც უკვე აღვნიშნეთ, სისტემის გატეხვას ხელი შეუწყო თვით ზურგჩანთის ამოცანის ზემოთ მოყვანილმა მესამე თვისებამ, რომ თუ არსებობს ამოხსნა (B, y) შესასვლელისათვის, ის აუცილებლად დაემთხვევა (W, S) შესასვლელის ამოხსნას. ამ თვისების შედეგია ის, რომ ეკას არ სჭირდება იპოვოს ზუსტად (k_0, m_0), საკმარისია მან იპოვოს ნებისმიერი ისეთი (k, m) წყვილი, რომ $\overline{W} = (\overline{w_1}, ..., \overline{w_n})$ მიმდევრობა, რომელიც მიიღება B მიმდევრობიდან შემდეგი ფორმულით: $\overline{w_i} = b_i \cdot k^{-1} (\mathrm{mod}\, m)$ იყოს ზეზრდადი და აკმაყოფილებდეს პირობას $\sum_{i=1}^{n} \overline{w_i} < \overline{m}$.

ზურგჩანთის ამოცანაში გვაქვს ორი პარამეტრი, რომლებიც განსაზღვრავენ W ვექტორის ზომებს. ესაა ვექტორის კომპონენტების რაოდენობა n და თითოეული კომპონენტის სიდიდე, ამასთან, კომპონენტების რაოდენობით განისაზღვრება თითოეული კომპონენტის სიდიდე. ა. შამირმა დააფიქსირა რა კომპონენტების რაოდენობა n, შემოიღო პროპორციულობის კოეფიციენტი $d > 1$ და უცნობი მოდულის ზომა დააფიქსირა როგორც dn სიგრძის ორობითი რიცხვი. i-ური კომპონენტის ზომა კი გამოითვლება ფორმულით: $dn - 1 - n + i$. კერძოდ, ა. შამირის შეტევის დროს $d = 2, \quad n = 100$.

იმისათვის, რომ ვიპოვოთ რომელიმე (k,m) წყვილი, განვიხილოთ ფუნქცია $b_i \cdot u (\mathrm{mod}\, m)$, სადაც $u = k^{-1}$ და მისი გრაფიკი (იხ. სურ. 9.1.ა). i-ს ყველა მნიშვნელობისათვის ეს გრაფიკი შედგება წრფივი მონაკვეთებისაგან და წერტილები $u = pm/b_i$ წარმოადგენენ წყვეტის წერტილებს ($p = 1,2,...$). გავიხსენოთ, რომ $b_1 \cdot u (\mathrm{mod}\, m) = w_1$, სადაც u არის არა ცვლადი, არამედ მამრავლი, რომლის შებრუნებულ მნიშვნელობასაც ვეძებთ და რომ w_1 გაცილებით პატარა უნდა იყოს m-ზე, გამოვა, რომ u მნიშვნელობა ახლოს უნდა იყოს b_1 ფუნქციის მინიმუმთან.

ანალოგიური მსჯელობით გამოდის, რომ u მნიშვნელობა ახლოს უნდა იყოს b_2 ფუნქციის მინიმუმთანაც, რაც იმას ნიშნავს, რომ ამ ფუნქციების რომელიღაც მინიმუმები ახლოს უნდა იყოს ერთმანეთთან. თუ გავაგრძელებთ ასეთ მსჯელობას, აღმოჩნდება, რომ უნდა ვეძებოთ ისეთი პატარა ინტერვალი, რომელიც შეიცავს რამდენიმე b_i-ური ფუნქციების მინიმუმებს. ა. შამირმა ევრისტიკული გამოთვლებით აჩვენა, $d = 2$ კოეფიციენტისათვის საკმარისია პირველი ოთხი ფუნქცია. იმისათვის, რომ შევძლოთ ასეთი ინტერვალის მოძებნა, ერთი შეხედვით აუცილებელია მოდულის ფუძის ცოდნა, მაგრამ თუ დავუკვირდებით სურ. 9.1.ბ) დავინახავთ, რომ მოდულის ფუძეზე შეკვეცით ფუნქციის სახე არ იცვლება, იცვლება მხოლოდ ფუნქციის მასშტაბი.

ამ თვისების გამოყენებით ა. შამირმა შექმნა ზურგჩანთის სისტემაზე შეტევის ალგორითმი, რომელიც შედგება ორი ნაწილისაგან. პირველ ნაწილში ალგორითმი ლენსტრას მთელრიცხვა პროგრამირების საშუალებით ეძებს ისეთ p რიცხვებს, რომლებისთვისაც კმაყოფილდება პირობა, რომ u/m სიდიდე რამდენიმე b_i-თვის მდებარეობს ამ ფუნქციათა საერთო მინიმალურ ინტერვალში. გარკვეული რაოდენობის ასეთი წერტილების დაგროვების შემდეგ, ალგორითმი დიოფანტეს სწრაფი აპროქსიმაციის მეთოდით ეძებს (u,m) წყვილს, რომლის საშუალებითაც B ღია გასაღებიდან შესაძლებელი იქნება საიდუმლო გასაღების გამოთვლა.



ა) ბ)

სურ. 9.1. $b_i \cdot u (\mathrm{mod}\, m)$ ფუნქციის გრაფიკი

მას შემდეგ, რაც ა. შამირმა გატეხა მერკლი-ჰელმანის ალგორითმი, გამოქვეყნდა ზურგჩანთის ალგორითმის სხვადასხვა ახალი ვარიანტები, რომლებშიც ხდებოდა ორჯერ მოდულით გამრავლების და სხვა მსგავსი ოპერაციების საშუალებით საიდუმლო გასაღებიდან ღია გასაღების მიღება, მაგრამ ყველა მათგანი ტყდებოდა ადვილად, რადგანაც არცერთი ფუნქცია, რომლის საშუალებითაც საიდუმლო გასაღებიდან მიიღება ღია გასაღები არ იყო ცალმხრივ მიმართული ფუნქცია. ერთადერთი ვარიანტი ზურგჩანთიანი სისტემებისა, რომელიც ჯერაც არ არის გატეხილი, ღია გასაღების მისაღებად იყენებს მოდულით ახარისხების ფუნქციას, რომელიც ნამდვილად წარმოადგენს ცალმხრივ მიმართულ ფუნქციას.

საკონტროლო კითხვები:

1. ჩამოაყალიბეთ ზურგჩანთის ამოცანა იმ სახით, როგორც ის გამოიყენება კრიპტოგრაფიაში.
2. როგორ მიმდევრობას ეწოდება ზეზრდადი მიმდევრობა?
3. როგორ ხდება სიდუმლო გასაღებიდან ღია გასაღების მიღება?
4. არის თუ არა საიდუმლო გასაღებიდან ღია გასაღების მიღების პროცედურა ცალმხრივი ფუნქცია?
5. როგორ ხდება ინფორმაციის დაშიფრვა ჰელმან-მერკლის ალგორითმში?
6. აღწერეთ დეშიფრაციის პროცესი.
7. როგორ გატეხეს ჰელმან-მერკლის ალგორითმი?U

კრიპტოსისტემა RSA

10.1. ეილერის ფუნქცია. კრიპტოსისტემა **RSA** წარმოადგენს ღია გასაღებიან ალგორითმს, რომელიც შეიძლება გამოყენებული იქნას როგორც ინფორმაციის დასაშიფრად, ასევე ციფრული ხელმოწერის შესაქმნელად. ალგორითმში ღია და საიდუმლო გასაღებებს შორის დამოკიდებულება მოცემულია ცალმხრივი ფუნქციის საშუალებით, კერძოდ ეს ფუნქცია ჩვენს მიერ ზემოთ განხილული რიცხვის მარტივ მამრავლებად დაშლის ამოცანაა, რომელიც როგორც ვნახეთ, წარმოადგენს **NP** სირთულის ამოცანას.

კრიპტოსისტემის შესაქმნელად უნდა ავიღოთ ორი მარტივი რიცხვი (მაგალითად $p=17$ და $q=19$). გადავამრავლოთ ისინი ერთმანეთზე $n=17\cdot 19=323$. დავითვალოთ ამ რიცხვისათვის ეილერის ფუნქციის მნიშვნელობა. ეილერის ფუნქცია მიეკუთვნება არითმეტიკულ ფუნქციებს.

განსაზღვრა 10.1. *ფუნქციას ეწოდება **არითმეტიკული ფუნქცია**, თუ ის განსაზღვრულია მთელ ნატურალურ რიცხვთა სიმრავლეზე. ფუნქციის მნიშვნელობათა სიმრავლე შეიძლება იყოს ნებისმიერი.* $f(a)$ *იქნება მულტიპლიკაციური ფუნქცია, თუ*

$$f(a\cdot b)=f(a)\cdot f(b),$$

როდესაც უ.ს.გ. $(a,b)=1$ *და იქნება მულტიპლიკაციური ფუნქცია ფართო გაგებით, როდესაც*

$$f(a\cdot b)=f(a)\cdot f(b)$$

ნებისმიერი a *და* b *რიცხვებისათვის.*

ჩვენთვის საინტერესოა არითმეტიკული ფუნქცია, რომელსაც უწოდებენ ***ეილერის ფუნქციას***. $\varphi(n)$ ფუნქციის მნიშვნელობა მოცემული n-თვის არის იმ რიცხვების რაოდენობა, რომლებიც ნაკლებია n-ზე და ურთიერთმარტივია n-თან. მაგალითად $\varphi(7)=6;\quad \varphi(10)=4.$ აშკარაა, რომ მარტივი p რიცხვისათვის $\varphi(p)=p-1$. ეილერის ფუნქცია მულტიპლიკაციური ფუნქციაა, აქედან გამომდინარე, ადვილია $\varphi(n)$ ფუნქციის გამოსათვლელი ფორმულის მიღება შდგენილი რიცხვებისათვის. როგორც ვიცით, ნებისმიერი ნატურალური რიცხვი შეიძლება დავშალოთ მარტივი მამრავლების ხარისხებად. დავუშვათ, $n=p^{\alpha}\cdot q^{\beta}\cdot\ldots\cdot l^{k}$. როგორც ვნახეთ $\varphi(p)=p-1$, ადვილი გამოსათვლელია აგრეთვე $\varphi(p^{\alpha})$. რიცხვების მიმდევრობაში $1,2,\ldots,p^{\alpha}-1$ p –ზე იყოფა მხოლოდ რიცხვები p^{t}, სადაც t იცვლება ნოლიდან $p^{\alpha-1}$-მდე. მათი რაოდენობა უდრის $p^{\alpha-1}$. სულ გვაქვს p^{α} რიცხვი, ამიტომ

$$\varphi(p^{\alpha})=p^{\alpha}-p^{\alpha-1}=p^{\alpha}(1-1/p).$$

მაშინ

$$\varphi(n)=\varphi(p^{\alpha}q^{\beta}...l^{k})=\varphi(p^{\alpha})\varphi(q^{\beta})...\varphi(l^{k})=p^{\alpha}(1-1/p)q^{\beta}(1-1/q)...l^{k}(1-1/l)=$$
$$n(1-\frac{1}{p})(1-\frac{1}{q})...(1-\frac{1}{l}).$$

ეილერის ფუნქციას გააჩნია ერთი შესანიშნავი თვისება.

თეორემა 1 *ნებისმიერი მთელი დადებითი n რიცხვისათვის სრულდება პირობა*

$$\sum_{d|n}\varphi(d)=n\ .$$

დამტკიცება. დავუშვათ, $S_d=\{x\,|\,1\le x\le n,(x,n)=d\}$. აშკარაა, რომ ყოველი d -თვის, რომელიც ყოფს n -ს, სიმრავლე $S=\{1,2,...,n\}$ იყოფა თანაუკვეთ ქვესიმრავლეებად. მაშინ

$$S=\bigcup_{d|n}S_d\ ,$$

ამასთან, ყოველი $d\mid n$ რიცხვისათვის სრულდება პირობა $|S_d|=\varphi(n/d).$ აქედან გამოდის, რომ

$$\sum_{d|n}\varphi(n/d).$$

მაგრამ ნებისმიერი რიცხვისათვის სრულდება აგრეთვე პირობა, რომ $(n/d)\mid n$. აქედან მივიღებთ, რომ

$$\sum_{d|n}(\varphi(n/d)=\sum_{(n/d)|n}(\varphi(n/d)=\sum_{d|n}\varphi(d)\ .$$

მაგალითად. თუ $n=15$, მაშინ მისი გამყოფებია $1,3,5,15.$ მაშინ გვექნება

$$\varphi(1)+\varphi(3)+\varphi(5)+\varphi(15)=1+2+4+8=15.$$

ჩვენი მაგალითისათვის გვექნება $\varphi(323)=(17-1)(19-1)=16\cdot 18=288$.

10.2. საიდუმლო გასაღები. შევარჩიოთ ისეთი e რიცხვი, რომელიც ნაკლებია n-ზე და რომლისთვისაც სრულდება პირობა უ.ს.გ. $(e,\varphi(n))=1.$ ეს პირობა ნიშნავს, რომ e და $\varphi(n)$ რიცხვები ურთიერთმარტივი რიცხვებია. ეს მოთხოვნა აუცილებელია იმისათვის, რომ კრიპტოსისტემას გააჩნდეს ღია გასაღები. დავუშვათ, $e=13$ და ჩვენ გვინდა დავშიფროთ სიტყვა "წვიმს" (ძალიან მალე დავინახავთ, თუ რატომ ავიღეთ ასეთი მოკლე ტექსტი). პირველ რიგში გადავიყვანოთ ეს სიტყვა რაიმე რიცხვში. ამისათვის გადავნომროთ ქართული ასოები ერთიდან ოცდაცამეტამდე ანბანის მიხედვით, ოღონდ თითოეულ მათგანს შევუსაბამოთ ორთანრიგიანი რიცხვები (იხ. ცხრილი 10.1.):

ცხრილი 10.1.

ა	ბ	გ	დ	ე	ვ	ზ	თ	ი	კ	ლ	მ	ნ	ო	პ	ჟ	რ
01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17
ს	ტ	უ	ფ	ქ	ღ	ყ	შ	ჩ	ც	ძ	წ	ჭ	ხ	ჯ	ჰ	
18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	

მაშინ ჩვენს ტექსტს შეესაბამება რიცხვითი მნიშვნელობა 29 06 09 12 18.

დაშიფრვის პროცედურა მდგომარეობს ამ რიცხვის $e=13$ ხარისხში აყვანაში მოდულით n. ცხადია ასეთი დიდი რიცხვის აყვანა ასეთ მაღალ ხარისხში კალკულატორის საშუალებით შეუძლებელია, ამიტომ რიცხვი დავყოთ ბლოკებად. ბლოკების სიგრძე შეიძლება იყოს ნებისმიერი, ოღონდ გავითვალისწინოთ ორი პირობა, ბლოკი არ უნდა იწყებოდეს ნოლით და რიცხვი, რომელიც წარმოადგენს ბლოკს ნაკლები უნდა იყოს მოდულის სიდიდეზე (n =323-ზე). გვექნება შემდეგი სახის ბლოკები: 290 60 91 21 8.

დავშიფროთ პირველი ბლოკი. გვექნება $290^{13}(\mathrm{mod}\,323)=188$. აქ ჩვენ გამოვიყენეთ ცალმხრივი ფუნქციების დროს განხილული თვისება, რომელიც საშუალებას გვაძლევს შევამციროთ ახარისხებისათვის საჭირო ოპერაციების რაოდენობა, როდესაც რიცხვი აგვყავს ხარისხში რაიმე მოდულით. დავუშვათ, დავშიფრეთ დანარჩენი ბლოკებიც. როგორ უნდა მოხდეს დაშიფრული ტექსტის დეშიფრაცია. ამისათვის საჭიროა დეშიფრაციის გასაღები d , ისეთი, რომ კმაყოფილდებოდეს პირობა $ed\equiv 1(\mathrm{mod}\,\varphi(n))$. ეს იმას ნიშნავს, რომ დეშიფრაციის გასაღები წარმოადგენს დაშიფრვის გასაღების შებრუნებულს მოდულით $\varphi(n)$. იმისათვის, რომ ვიპოვოთ მოცემული რიცხვის შებრუნებული რაიმე მოდულით, უნდა ამოვხსნათ წრფივი შედარება

$$ax \equiv 1(\mathrm{mod}\, n)\ .$$

ამისათვის შეიძლება გამოვიყენოთ ეილერის თეორემა.

<u>თეორემა 2.</u> (ეილერის თეორემა). თუ **უ.ს.გ.** $(a,n)=1$, მაშინ ადგილი აქვს ტოლობას

$$a^{\varphi(n)} \equiv 1(\mathrm{mod}\, n)$$

<u>დამტკიცება.</u> დავამტკიცოთ ეს თეორემა. დავუშვათ, $r_1, r_2, ..., r_{\varphi(n)}$ წარმოადგენს ნაშთთა დაყვანილ სისტემას მოდულით n. მათი რაოდენობა იქნება $\varphi(n)$. რადგანაც უ.ს.გ. $(a,n)=1$, $ar_1, ar_2, ..., ar_{\varphi(n)}$ ასევე იქნება ნაშთთა სისტემა. დავამყაროთ მათ შორის ურთიერთცალსახა შესაბამისობა

$$\begin{cases} ar_1 \equiv r_\alpha (\mathrm{mod}\, n) \\ ar_2 \equiv r_\beta (\mathrm{mod}\, n) \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ ar_{\varphi(n)} \equiv r_\zeta (\mathrm{mod}\, n) \end{cases}$$

სადაც იგივე რიცხვებია, ოღონდ დალაგებული სხვა მიმდევრობით. გადავამრავლოთ შედარებები და რადგანაც თითოეული r_i ურთიერთმარტივია n -თან, შევკვეცოთ შედარების ორივე მხარე მათ ნამრავლზე. მივიღებთ ეილერის თეორემას.

ჩვენი წრფივი შედარების ამოსახსნელად შედარების ორივე მხარე გავამრავლოთ გამოსახულებაზე $a^{\varphi(n)-1}$, მივიღებთ,

$$a^{\varphi(n)} x \equiv a^{\varphi(n)-1} (\mathrm{mod}\, n).$$

თუ გამოვიყენებთ ეილერის ფორმულას, მივიღებთ

$$x \equiv \cdot a^{\varphi(n)-1} (\mathrm{mod}\, n)\ .$$

ამ ფორმულის გამოყენება არც ისე ადვილია, როდესაც დიდი რიცხვია. ამიტომ პრაქტიკაში უფრო ხშირად გამოიყენება ევკლიდეს გაფართოებული ალგორითმი. როგორც ვიცით, ევკლიდეს ალგორითმი გამოიყენება ორი რიცხვის უდიდესი საერთო გამყოფის მისაძებნად.

<u>10.3. ევკლიდეს გაფართოებული ალგორითმი.</u> ორი რიცხვის უდიდეს საერთო გამყოფს აქვს საინტერესო თვისებები, რომლებიც საშუალებას გვაძლევენ გამოვთვალოთ ეს რიცხვი. ადვილი საჩვენებელია, რომ

1. **უ.ს.გ.** $(a,a)=\{a,-a\}$
2. **უ.ს.გ.** $(a,0)=\{a,-a\}$
3. **უ.ს.გ.** $(a,b)=$ **უ.ს.გ.** (b,a)
4. **უ.ს.გ.** $(c\cdot a, c\cdot b) = c$ **უ.ს.გ.** (a,b).
5. **უ.ს.გ.** $(a,b)=$ **უ.ს.გ.** $(a-b,b)$
6. თუ **უ.ს.გ.** $(a,c)=\{1,-1\}$, მაშინ **უ.ს.გ.** $(a,c\cdot b)=$ **უ.ს.გ.** (a,b)
7. **უ.ს.გ.** $(a,b)=$**უ.ს.გ.** (b,r), სადაც r არის a რიცხვის b -ზე გაყოფისას მიღებული ნაშთი.
8. თუ $d=$**უ.ს.გ.** (a,b), მაშინ ყოველთვის მოიძებნება ისეთი $x, y \in Z$, რომ ადგილი აქვს ტოლობას $ax+by=d$.

სხვადასხვა თვისებების კომბინაციით, შეგვიძლია მივიღოთ უდიდესი სართო გამყოფის სხვადასხვა ალგორითმები. თუ გამოვიყენებთ (2) და (7) თვისებებს, მივიღებთ ცნობილ ევკლიდეს ალგორითმს, თუ გამოვიყენებთ (1), (4) და (5) თვისებებს, სადაც, მივიღებთ ბინარულ ალგორითმს, რომელიც გაცილებით სწრაფია ევკლიდეს ალგორითმთან შედარებით, როდესაც უდიდეს საერთო გამყოფს ვითვლით კომპიუტერის საშუალებით (ანუ ოროობით სისტემაში).

როგორც უკვე აღვნიშნეთ, ევკლიდეს ალგორითმი ეფუძნება უდიდესი საერთო გამყოფის (2) და (7) თვისებებს. (2) თვისება ტრივიალურია, როგორც ვიცით, ნოლი იყოფა ნებისმიერ რიცხვზე, გარდა ნოლისა, ამიტომ **უ.ს.გ.** $(a,0)=\{a,-a\}$. რაც შეეხება (7) თვისებას, ვაჩვენოთ, რომ **უ.სგ.** $(a,b)=$**უ.ს.გ** (b,r). ამისათვის საკმარისია ვაჩვენოთ რომ ამ რიცხვების საერთო გამყოფების სიმრავლე ერთმანეთს ემთხვევა. დავუშვათ, c არის a და b რიცხვების ნებისმიერი საერთი გამყოფი, რაც იმას ნიშნავს, რომ არსებობენ ისეთი $q_1, q_2 \in Z$ რიცხვები, რომ $a=c\cdot q_1$ და $b=c\cdot q_2$.

რადგანაც r არის b რიცხვის a –ზე გაყოფის შედეგად მიღებული ნაშთი, ის გამოისახება შემდეგი სახით:

$$r = b - a \cdot q$$

თუ ამ ტოლობაში ჩავსვავთ a -ს და b -ს მნიშვნელობებს, მივიღებთ:

$$r = c \cdot q_2 - c \cdot q_1 q = c(q_2 - q_1 q)$$

რაც იმას ნიშნავს, რომ c არის r -ის გამყოფიც. ანალოგიურად შეიძლება ვაჩვენოთ, რომ (b, r) რიცხვების ნებისმიერი საერთო გამყოფი გაყოფს აგრეთვე a -ს. ალგორითმი კი ძალიან მარტივია. დავუბრუნდეთ უდიდესი საერთო გამყოფის მერვე თვისებას @

$$ax + by = d \ .$$

ასეთი სახის ორუცნობიან განტოლებას, სადაც ვეძებთ ამოხსნას მთელ რიცხვებში, ეწოდება დიოფანტეს განტოლება. კრიპტოგრაფიაში განსაკუთრებული მნიშვნელობა აქვს იმ შემთხვევას, როდესაც $d = 1$, ანუ a და b რიცხვები ურთიერთმარტივია. ამ შემთხვევაში გვექნება, რომ

$$ax + by = 1$$

ეს განტოლება შეიძლება გამოვიყენოთ მოცემული რიცხვის შებრუნებული რიცხვის მოსაძებნად რაიმე მოდულით. თუ ჩვენ ვეძებთ b რიცხვის შებრუნებულს მოდულით a , ანუ ვხსნით განტოლებას

$$bx \equiv 1 (\mathrm{mod}\, a) \ ,$$

მოდულის განმატებიდან გამომდინარე შეგვიძლია ის შევცვალოთ სწორედ განტოლებითი

$$bx + ka = 1 \ .$$

იმისათვის, რომ ამოვხსნათ ეს განტოლება, კვლავ შეიძლება გამოვიყენოთ ევკლიდეს ალგორითმი, რომელსაც ეწოდება ევკლიდეს გაფართოებული ალგორითმი. ჯერ ევკლიდეს ალგორითმით მოვძებნოთ უდიდესი საერთო გამყოფი. შემდეგ მიღებული r_m ნაშთი (რომელიც წარმოადგენს **უ.ს.გ.** (a,b)). გამოვსახოთ r_{m-1} და r_{m-2} ნაშთებით. შემდეგე r_{m-1} ნაშთი გამოვსახოთ წინა ნაშთებით და ასე გავაგრძელოთ, სანამ ნაშთი არ გამოისახება a და b რიცხვების საშუალებით.

მაგალითად, დავუშვათ მოცემული გვაქვს რიცხვები 75 და 11. ეს რიცხვები ურთიერთმარტივია და ამიტომ მათი უდიდესი საერთო გამყოფი იქნება ერთი. გამოვიყენოთ ევკლიდეს ალგორითმი უდიდესი საერთო გამყოფის მოსაძებნად. გვექნება:

$$75 = 6 \times 11 + 9$$
$$11 = 1 \times 9 + 2$$
$$9 = 4 \times 2 + 1$$
$$2 = 1 \times 2 + 0$$

ევკლიდეს ალგორითმმა იპოვა **უ.ს.გ.** გამოვსახოთ მიღებული ნაშთები:

$$1 = 9 - 4 \times 2$$
$$2 = 11 - 1 \times 9$$
$$9 = 75 - 6 \times 11$$

საიდანაც საბოლოოდ მივიღებთ, რომ:

$$1 = 9 - 4 \times 2 = 9 - 4 \times (11 - 1 \times 9) = 5 \times 9 - 4 \times 11 = 5 \times (75 - 6 \times 11) - 4 \times 11 = 5 \times 75 - 34 \times 11$$

ამგვარად, ჩვენ გამოვსახეთ **უ.ს.გ.** $(75,11)$ ამ რიცხვების საშუალებით და შესაბამისად $x = 5$ და $y = -34$. ე.ი. თერთმეტის შებრუნებული სამოცდახუთმეტის მოდულით იქნება $-34 + 75 = 41$ (ნაშთი არ შეიძლება იყოს უარყოფითი სიდიდე). ამგვარად, მივიღეთ, რომ კოეფიციენტების მოსაძებნად (ანუ დიოფანტეს განტოლების ამოსახსნელად), საჭიროა თითოეულ ბიჯზე ნაშთი გამოვსახოთ იმ რიცხვებით, რომლებიც გვაძლევენ ამ ნაშთს და შემდეგ უდიდესი საერთო გამყოფი მარტივად წარმოიდგინება საწყისი რიცხვებით.

გამოვიყენოთ ეს ალგორითმი ჩვენს მაგალითში, გვექნება

$$d \cdot 13 \equiv 1 (\mathrm{mod}\, 288) \Rightarrow d \cdot 13 - k \cdot 288 = 1$$

$$288 = 22 \cdot 13 + 2 \qquad 2 = 1 \cdot 288 - 22 \cdot 13$$

$$13 = 6 \cdot 2 + 1 \qquad 1 = 1 \cdot 13 - 6 \cdot 2 = 1 \cdot 13 - 6 \cdot (1 \cdot 288 - 22 \cdot 13) = 133 \cdot 13 - 6 \cdot 288$$

საიდანაც გამოდის, რომ $d = 133$ (ეხლა ალბათ გასაგებია, რატომ ავირჩიეთ ასე პატარა ტექსტი).

10.4. დეშიფრაცია. ამგვარად, მივიღებთ, რომ $188^{133} (\mathrm{mod}\, 323) = 290$. თუ მოვახდენთ ყველა ბლოკების დეშიფრაციას, მივიღებთ საწყის რიცხვებს, რომელთა კონკატაციით და შემდეგ ასოებში გადაყვანით აღვადგენთ საწყის ტექსტს. ამ პროცედურიდან გასაგები ხდება თუ რატომ შევუსაბამეთ ასოებს ორთანრიგიანი რიცხვები. წინააღმდეგ შემთხვევაში გაუგებარი იქნებოდა მაგალითად რიცხვი 14 არის "ად" ასოების კომბინაცია, თუ ერთი ასო "ო".

დაგვრჩა დავამტკიცოთ, თუ როგორ მუშაობს ეს ალგორითმი, ანუ რატომ ხდება ის, რაც ხდება. დამტკიცება კი ძალიან მარტივია: დაშიფრვა ხდება ფორმულით:

$$c = m^e (\mathrm{mod}\, n)$$

სადაც c არის შიფროტექსტი, m არის ღია ტექსტი და e არის დაშიფრვის გასაღები. დეშიფრაცია ხდება შემდეგი ფორმულით:

$$m = c^d (\mathrm{mod}\, n) = m^{ed} (\mathrm{mod}\, n) \ .$$

თუ გავიხსენებთ, რომ $ed \equiv 1 (\mathrm{mod}\, \varphi(n)) \Rightarrow ed = k\varphi(n) + 1$ და ჩავსვავთ ამ მნიშვნელობას, გვექნება

$$m = m^{k\varphi(n)+1} (\mathrm{mod}\, n) = m \cdot m^{k\varphi(n)} (\mathrm{mod}\, n) = m(\mathrm{mod}\, n) \cdot m^{k\varphi(n)} (\mathrm{mod}\, n)$$

ეილერის ფორმულის თანახმად მეორე მამრავლი ტოლია ერთის. ამასთან, ბლოკები ისე იყო აღებული, რომ რიცხვის მნიშვნელობა ნაკლები იყოს მოდულის ფუძეზე, რაც ცალსახად აღადგენს ღია ტექსტს.

ჩვენ განვიხილეთ უმარტივესი მაგალითი და აქცა საჭირო გახდა რიცხვის 133 ხარისხში აყვანა, როდესაც მარტივი რიცხვები დიდია (200-300 თანრიგი ათობით სისტემაში) ცხადია, რომ გამოთვლის პროცედურა მოითხოვს ძალიან დიდ დროს. ამიტომ ამ ალგორითმის გამოყენება თითქმის შეუძლებელია დიდი ოდენობის ინფორმაციის დასაშიფრად (ის ათასჯერ უფრო ნელია ვიდრე იყო **DES**-ი). ალგორითმი ძალიან საიმედოა, მაგრამ აუცილებელია მისი ფრთხილი გამოყენება, რადგანაც არასწორი მოხმარების დროს ის შეიძლება გატყდეს. გატყდეს არა იმიტომ, რომ ვინმე მოახერხებს მოდულის ფუძის დაშლას მარტივ მამრავლებად, რაც წარმოადგენს ამ ალგორითმის საიმედოობის საფუძველს, არამედ სხვა მეთოდების საშუალებით იპოვის ღია ტექსტს. განვიხილოთ ერთი ასეთი შემთხვევა. დავუშვათ, ჩვენ ავაგეთ ალგორითმი და მომხმარებლების რაოდენობა დიდია, ანუ დიდი რაოდენობის მომხმარებლები იყენებენ ერთსა და იმავე ფუძეს, თუმცა გასაღებები აქვთ საკუთარი. აი რა შეიძლება მოხდეს ამ შემთხვევაში. დავუშვათ, ერთი და იგივე ღია ტექსტი დაიშიფრა ორი სხვადასხვა გასღებით, რომლებიც როგორც წესი არიან ურთიერთმარტივი რიცხვები. გვექნება:

$$c_1 = m^{e_1} (\mathrm{mod}\, n)$$

$$c_2 = m^{e_2} (\mathrm{mod}\, n)$$

მოწინააღმდეგემ იცის ორივე შიფროტექსტი, ორივე გასაღები (გასაღებები ღიაა) და მოდულის ფუძე. ვნახოთ რის გაკეთება შეუძლია მას. რადგანაც e_1 და e_2 გასაღებები ურთიერთმარტივი რიცხვებია, ევკლიდეს გაფართოებული ალგორითმით ის იპოვნის ისეთ ორ რიცხვს a და b-ს, რომ შესრულდება პირობა:

$$ae_1 + be_2 = 1,$$

ამ ორი რიცხვიდან ერთი აუცილებლად იქნება უარყოფითი. დავუშვათ უარყოფითია b, მაშინ ის კიდევ ერთხელ გამოიყენებს ევკლიდეს გაფართოებულ ალგორითმს და გამოთვლის c_2^{-1}-ს. ამის შემდეგ გამოთვლის $c_1^a \cdot (c_2^{-1})^{-b} = m(\mathrm{mod}\, n)$.

ასეთ შემთხვევაში არსებობს აგრეთვე შეტევა, როდესაც შესაძლებელია ფუძის დაშლა მარტივ მამრავლებად (ალგორითმი ალბათურია) და დეტერმინირებული ალგორითმი, რომლის საშუალებითაც შეიძლება ვიპოვოთ საიდუმლო გასაღები ფუძის დაშლის გარეშე. ასეთი შეტევების არსებობა მიუთითებს იმაზე, რომ არ შეიძლება ერთი და იგივე ფუძე გამოვიყენოთ რამდენიმე მომხმარებლისათვის.

საკონტროლო კითხვები:

1. რა ცალსახა ფუნქცია გამოყენება RSA ალგორითმში?
2. როგორ გამოითვლება ღიოა გასაღები დახურული გასაღებიდან?

3. როგორ უნდა გამოვთვალოთ ეილერის ფუნქციის მნიშვნელობა $n = p \cdot q$ შსედგენილი რიცხვისათვის?
4. როგორ ხდება ინფორმაციის წარმოდგენა ბლოკების სახით?
5. როგორ ხდება დაშიფრვა **RSA** ალგორითმში? გაშიფრვა?
6. როგორ შეიძლება გავტეხოთ **RSA** ალგორითმი?

ციფრული ხელმოწერის ალგორითმები

10.1. ციფრული ხელმოწერის მიზნები. ღია გასაღებიანი კრიპტოსისტემების გამოყენება განსაკუთ--რებით მოსახერხებელია ე.წ. ციფრული ხელმოწერებისათვის. ციფრული ხელმოწერა ნიშნავს განსა-კუთრებული პროცედურების საშუალებით ციფრული სახით წარმოდგენილი ინფორმაციის ჭეშმა-რიტობის დასაბუთებას. ანალოგიურად იმისა, როდესაც ქაღალდზე დაწერილი ინფორმაციის ჭეშ-მარიტებას ადასტურებენ ხელმოწერისა და ბეჭდის საშუალებით. აქედან გამომდინარე, შეგვიძლია ჩამოვთვალოთ ის კრიტერიუმები, რომლებსაც უნდა აკმაყოფილებდეს ციფული ხელმოწერის ალ-გორითმი:

1. ციფრული ხელმოწერის საშუალებით ნებისმიერ პირს, რომელიც ფლობს ინფორმაციის ავტორის ღია გასაღებს, ცალსახად უნდა შეეძლოს გადმოცემული ინფორმაციის ავტორის დადგენა;
2. შეუძლებელი უნდა იყოს ხელმოწერის შემდეგ დოკუმენტში ცვლილებების შეტანა;
3. შეუძლებელი უნდა იყოს ხელმოწერის გაყალბება;
4. ინფორმაციის ავტორს არ უნდა შეეძლოს უარყოს თავისი ხელმოწერა.

ამასთან, ციფრული ხელმოწერის ალგორითმების დანიშნულება არაა ინფორმაციის კონფი-დენციალურობის უზრუნველყოფა. ციფრული ხელმოწერა შეიძლება გაკეთდეს როგორც შიფრო-გრამაზე, ასევე ღია ტექსტზე. თუ ქვეყანაში მიღებულია კანონი ციფრული ხელმოწერის შესახებ, მაშინ ციფრულ ხელმოწერას ისეთივე ძალა აქვს, როგორიც ქაღალდზე გაკეთებულ ხელმოწერას.

ციფრული ხელმოწერისათვის შეიძლება გამოვიყენოთ ზემოთ განხილული **RSA** ალგორით-მი, ოღონდ ამ შემთხვევაში ხელმოწერა კეთდება საიდუმლო გასაღებით და მოწმდება ღია გასა-ღების საშუალებით. ხელმოწერა შეიძლება გაკეთდეს როგორც ტექსტზე, ასევე მის ჰეშ-ფუნქციაზეც. ასეთ შემთხვევაში გამოითვლება ტექსტის ხელმოწერა და შემდეგ მასზე გაკეთდება ხელმოწერა. ამ მეთოდის სუსტი მხარეა ის, რომ თუ მოწინააღმდეგემ შეარჩია განსხვავებული ტექსტი, რომლის ჰეშ-ფუნქციაც ემთხვევა ხელმოწერილი ტექსტის ჰეშ-ფუნქციას, შეუძლია თქვენი ხელმოწერა გამოიყენოს საკუთარი მიზნებისათვის. 1991 წლამდე, სანამ ამერიკის სტანდარტების ნაციონალური ინსტიტუტი (**NIST**) მიიღებდა ახალ სტანდარტს ციფრული ხელმოწერის შესახებ, **RSA** გამოიყე-ნებოდა როგორც ხელმოწერის დე-ფაქტო საერთაშორისო სტანდარტი.

ხელმოწერის ახალი სტნდარტი **DSS (Digital Signatyre Standard)** ეფუძნება **DSA (Digital Signatyre Algorithm)**-ს. ალგორითმში გამოყენებული იყო ცალმხრივ მიმართული ფუნქცია

$$f(x) = a^x \pmod p .$$

ალგორითმში გამოყენებულია შემდეგი პარამეტრები:

p - მარტივი რიცხვი, რომლის სიგრძე ბიტებში ტოლია L, $L = 64 \cdot k$, $512 \le L \le 1024$.

q - მარტივი რიცხვი, რომლის სიგრძე 160 ბიტი და $q \mid (p-1)$.

g - რიცხვი, რომელიც უნდა ავიყვანოთ ხარისხში, აკმაყოფილებს შემდეგ თვისებას:

$g \equiv h^{(p-1)/q} \bmod p$, სადაც h ნებისმიერი და $(p-1)$-ზე ნაკლები რიცხვია, ისეთი, რომ $h^{(p-1)/q} \bmod p > 1$.

$x < q$ ნებისმიერი რიცხვია, რომელიც წარმოადგენს საიდუმლო გასაღებს.

$y = g^x \pmod p$ წარმოადგენს ღია გასაღებს.

ალგორითმში გამოიყენება აგრეთვე ღია ტექსტის ჰეშ-ფუნქცია $H(m)$, რომელიც გამოითვლება ჩვენს მიერ განხილული **SHA-1** ალგორითმის საშუალებით.

p, q, g პარამეტრები შეიძლება საერთო იყოს მომხმარებელთა ჯგუფისათვის.

10.2. ხელმოწერის პროცედურა. იმისათვის, რომ ანიმ მოაწეროს ხელი შეტყობინებას, ის ასრულებს შემდეგ პროცედურებს:

1. ანი დააგენერირებს შემთხვევით რიცხვს $k < q$.
2. ანი გამოთვლის $r = (g^k \bmod p) \bmod q$ და $s = (k^{-1}(H(m) + xr) \bmod q$. თუ რომელიმე ამ რიცხვთაგანი იქნება ნოლის ტოლი, მაშინ ანიმ უნდა გამოცვალოს k რიცხვი.
3. ხელმოწერის პარამეტრია სამასოცბიტიანი $r \| s$, რომელიც მიეწერება ტექსტს ბოლოში აუცილებლად ამ მიმდევრობით.

10.3. ხელმოწერის შემოწმების პროცედურა. ბექა მიიღებს რა ამ ხელმოწერას, მის შესამოწმებლად ატარებს შემდეგ პროცედურებას:

1. გამოთვლის
$$w = s^{-1} \bmod q$$
$$u_1 = (H(M) \cdot w) \bmod q$$
$$u_2 = (rw) \bmod q$$
$$v = ((g^{u_1} \cdot y^{u_2}) \bmod p) \bmod q$$
თუ რომელიმე ამ სიდიდეებისაგან მიიღო ნოლის ტოლი, ხელმოწერა არასწორია.
2. თუ $r = v$, მაშინ ხელმოწერა სწორია, წინააღმდეგ შემთხვევაში არა.

როგორც ვხედავთ, ხელმოწერის და ხელმოწერის შემოწმების პროცედურები აბსოლუტურად განსხვავდება ერთმანეთისაგან, ამიტომ ასეთ ალგორითმებს ხშირად უწოდებენ აგრეთვე ასიმეტრიულ ალგორითმებსაც.

10.4. ალგორითმის კორექტულობა. პირველ რიგში, რადგანაც
$$g \equiv h^{(p-1)/q} (\bmod\ p),$$
ფერმას მცირე თეორემის თანახმად (ესაა ეილერის თეორემის კერძო შემთხვევა, როდესაც მოდულის ფუძეა მარტივი რიცხვი) მივიღებთ, რომ
$$g^q \equiv h^{p-1} \equiv 1 (\bmod\ p). \quad (10.1)$$
რადგანაც $g > 1$, (10.1) ტოლობიდან გამოდის, რომ q ყოფილა g -ს მულტიპლიკატორული რიგი მოდულით p . ანი ითვლის s -ს $s = (k^{-1}(H(m) + xr) \bmod q$ ფორმულის საშუალებით. გავამრავლოთ ამ ტოლობის ორივე მხარე $k \cdot s^{-1}$ ნამრავლზე, მივიღებთ
$$k = (H(m) \cdot s^{-1} + r \cdot x \cdot s^{-1}) (\bmod q) = H(m) \cdot w (\bmod q) + x \cdot r \cdot w (\bmod q)$$
რადგანაც g -ს მულტიპლიკატორული რიგი მოდულით p არის q -ს ტოლი, თუ გამოვიყენებთ u_1, u_2 და y განსაზღვრებებს, მივიღებთ:
$$g^k = g^{u_1} \cdot y^{u_2} (\bmod\ p) \Rightarrow r = ((g^k \bmod p) \bmod q) = ((g^{u_1} \cdot y^{u_2} \bmod p) \bmod q) = v.$$

10.5. ელიფსური წირები. ამ ალგორითმის საიმედოობა მთლიანად დამოკიდებულია სასრულ ჯგუფში დისკრეტული ალგორითმის გამოთვლის შესაძლებლობაზე. მაშინ, როდესაც მიიღეს ციფრული ხელმოწერის სტანდარტად, ითვლებოდა, რომ ეს პრობლემა ეკუთვნის **NP** პრობლემათა კლასს და ამიტომ ალგორითმი იქნებოდა საიმედო. როგორც ვიცით, თანამედროვე სირთულეთა თეორიის ყველაზე უფრო ცნობილი პრობლემა მდგომარეობს სწორედ იმაში, რომ ვერ ხერხდება დავამტკიცოთ **P=NP**, თუ **P** ≠ **NP**. ამიტომ ხშირად, ამოცანა, რომელიც გვგონია, რომ ეკუთვნის **NP** კლასს, შეიძლება გადავიდეს **P** კლასში. ასე მოხდა ამ შემთხვევაშიც. მართალია ჯერ არავის დაუმტკიცებია, რომ დისკრეტული ლოგარითმის პრობლემა ეკუთვნის **P** კლასს, მაგრამ მუშაობა ამ მიმართულებით ისე სწრაფად განვითარდა, რომ **NIST**-ს მოუწია თავისი სტანდარტის გადახედვა. მან თავი დაიზღვია და შეცვალა სტანდარტში საიდუმლო გასაღებიდან ღია გასაღების მიღების ცალმხრივი ფუნქცია. ახალი სტანდარტი ითვალისწინებს ელიფსურ წირებზე დაყრდნობილ პროცედურას.

განსაზღვრა 10.1. *E ელიფსური წირი K ველზე ეწოდება წერტილთა სიმრავლეს $\{(x, y) \bigcup O\}$, სადაც (x, y) წერტილები აკმაყოფილებენ განტოლებას (10.2.), ხოლო O არის "წერტილი უსასრულობაში", რომელიც წარმოადგენს ერთეულოვან ელემენტს ელიფსურ წირზე განმარტებული წერტილების შეკრების ოპერაციისათვის.*

$$E: y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad (10.2.)$$

სადაც $a_1, a_2, a_3, a_4, a_6 \in K$ და $\Delta \neq 0$. Δ არის განტოლების (10.2.) დეტერმინანტი. პირობა $\Delta \neq 0$ განსაზღვრავს ელიფსური წირის "გლუვობას", ანუ იმას, რომ წირს არ ექნება ისეთი წერტილები, რომლებშიც ფუნქციის ორივე კერძო წარმოებული $dF/dx, dF/dy$ ერთდროულად გახდება ნოლის ტოლი. (10.2.) განტოლებას ეწოდება ვეიერშტრასის განტოლება. E განსაზღვრულია K ველზე ნიშნავს, რომ კოეფიციენტები $a_1, a_2.a_3, a_4, a_6$ ეკუთვნის K ველს. ამას ზოგჯერ აღნიშნავენ როგორც E/K.

ორ ელიფსურ წირს

$$E_1: y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6$$

და

$$E_1: y^2 + \overline{a_1}xy + \overline{a_3}y = x^3 + \overline{a_2}x^2 + \overline{a_4}x + \overline{a_6}$$

ეწოდებათ იზომორფული K ველზე, თუ არსებობენ ისეთი $u, r, s, t \in K, u \neq 0$, რომ ცვლადების შეცვლა

$$(x, y) \to (u^2x + r, u^3y + u^2sx + t)$$

გადაიყვანს E_1 წირს E_2 წირში. ასეთ გარდაქმნას ეწოდება ცვლადების დასაშვები გარდაქმნა. მაშასადამე, შესაძლებელია (10.2.) განტოლების გამარტივება ასეთი გარდაქმნით, ოღონდ აუცილებელია განვასხვავოთ ეს გარდაქმნა K ველის მახასიათებლის მიხედვით.

1. თუ K ველის მახასიათებელი $p > 3$, მაშინ ცვლადები გარდაიქმნება შემდეგი სახით:

$$(x, y) \to \left(\frac{x - 3a_1^2 - 12a_2}{36}, \frac{y - 3a_1x}{216} - \frac{a_1^3 + 4a_1a_2 - 12a_3}{24}\right)$$

და მიიღება მარტივი განტოლება

$$y^2 = x^3 + ax + b, \quad (10.3.)$$

სადაც $a, b \in K$ და დეტერმინანტი $\Delta = -16(4a^3 + 27b^{2)}$.

2. თუ ველის მახასიათებელი $(p = 2)$, მაშინ გვექნება ორი შემთხვევა იმის და მიხედვით, როგორია a_1. თუ $a_1 \neq 0$, მაშინ გარდაქმნას ექნება სახე:

$$(x, y) \to \left(a_1^2x + \frac{a_3}{a_1}, a_1^3y + \frac{a_1^2a_4 + a_3^2}{a_1^3}\right)$$

და მივიღებთ განტოლებას:

$$y^2 + xy = x^3 + ax^2 + b, \quad (10.4.)$$

სადაც $a, b \in K$ და $\Delta = b$.

თუ $a_1 = 0$, მაშინ გარდაქმნას აქვს სახე:

$$(x, y) \to (x + a_2, y)$$

და მიიღება განტოლება

$$y^2 + cy = x^3 + ax + b, \quad (10.5)$$

სადაც $a, b, c \in K$ და $\Delta = c^4$.

3. როდესაც ველის მახასიათებელი $(p = 3)$, კვლავ გვაქვს ორი შემთხვევა იმის და მიხედვით როგორ დამოკიდებულებაში არიან a_1 და a_2 კოეფიციენტები. თუ $a_1^2 \neq -a_2$ მაშინ ცვლადების გარდაქმნას აქვს შემდეგი სახე:

$$(x, y) \to \left(x + \frac{d_4}{d_2}, y + a_1x + a_1\frac{d_4}{d_2} + a_3\right),$$

სადაც $d_2 = a_1^2 + a_2$ და $d_4 = a_4 - a_1a_3$.

მაშინ განტოლება მიიღებს სახეს

$$y^2 = x^3 + ax^2 + b,$$

სადაც $a,b,\in K$ და $\Delta = -a^3 b$.

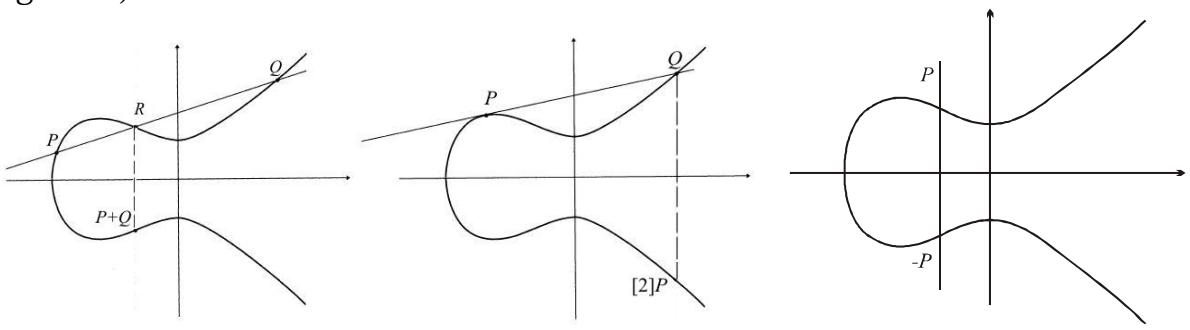
თუ $a_1^2 = -a_2$, მაშინ ცვლადების გარდაქმნას აქვს შემდეგი სახე: $(x,y) \to (x, y + a_1 x + a_3)$, განტოლება მიიღებს სახეს

$$y^2 = x^3 + ax + b \ ,$$

სადაც $a,b,\in K$ და $\Delta = -a^3$.

10.6. წერტილების შეკრება ელიფსურ წირზე. დავუშვათ, E არის ელიფსური წირი K ველზე. არსებობს წესი (ე.წ. ქორდისა და მხების წესი), რომელიც საშუალებას გვაძლევს ამ წირის ორი წერტილიდან მივიღოთ წირის მესამე წერტილი. თუ ამ წესს განვმარტავთ როგორც შეკრების ოპერაციას $E(K)$ წერტილების სიმრავლეზე, მაშინ იქნება ***ჯგუფი***, რომლის ნულოვან ელემენტსაც წარმოადგენს O წერტილი. სწორედ ეს ჯგუფი გამოიყენება ელიფსურ წირებზე დაფუძნებულ კრიპტოგრაფიულ სისტემებში.

განვმარტოთ ორი წერტილის შეკრების ოპერაცია ელიფსურ წირზე გეომეტრიულად (იხ. სურ. 10.1.).



ა) $P+Q$ ბ) $P+P$ გ) $P+(-P)=O$

სურ. 10.1 ელიფსურ წირზე ორი წერტილის ჯამის მოძებნის გეომეტრიული წესი.

1. იმისათვის, რომ შევკრიბოთ ელიფსური წირის ორი განსხვავებული P და Q წერტილები, საჭიროა ამ წერტილებზე გავატაროთ მკვეთი, რომელიც ელიფსურ წირს გადაკვეთს მესამე წერტილშიც. ამ მესამე წერტილის სიმეტრიული წერტილი x ღერძის მიმართ წარმოადგენს P და Q წერტილების ჯამს $P+Q$ (სურ. 10.1. ა)).
2. თუ ელიფსური წირის რომელიმე წერტილს გვინდა მივუმატოთ იგივე წერტილი, მაშინ ამ წერტილში უნდა გავატაროთ მხები და მხებისა და წირის გადაკვეთის შედეგად მიღებული წერტილის სიმეტრიული წერტილი x ღერძის მიმართ იქნება ამ წერტილს თავის თავთან შეკრების შედეგი. (სურ. 10.1. ბ)).
3. P წერტილის მოპირდაპირე ($-P$) წერტილი ეწოდება წერტილს, რომელიც სიმეტრიულია P წერტილის აბსცითა ღერძის მიმართ, ანუ თუ $P(x,y)$, მაშინ $-P(x,-y)$. ამ წერტილების ჯამი არის სწორედ უსასრულობის წერტილი $O=-O$, რომელიც არ ეკუთვნის ელიფსურ წერტილს და წარმოადგენს ერთეულოვან წერტილს (იხ. სურ. 10.1. გ)).

განვიხილოთ ისეთი ველი, რომლის მახასიათებელიც არ უდრის 2-ს ან 3-ს. როგორც ვნახეთ, ასეთ შემთხვევაში გარდაქმნის გზით შეგვიძლია მივიღოთ მარტივი სახის განტოლება

$$y^2 = x^3 + ax + b, \quad (10.3)$$

სადაც $a,b \in K$ და $\Delta = -16(4a^3 + 27b^{2)}$.

ამ შემთხვევაში, გლუვობის პირობა, რომ $\Delta \neq O$, დაიყვანება იმაზე, რომ (10.3) განტოლების მარცხენა მხარეს მდგომ კუბურ განტოლებას არ ჰქონდეს ჯერადი ფესვები. ვნახოთ, როგორ შეიძლება გამოვთვალოთ შეკრების შედეგად მიღებული წერტილების კოორდინატები.

ორი განსხვავებული წერტილების შეკრება. მოცემულია $P(x_1, y_1)$ და $Q(x_2, y_2)$. უნდა მოვძებნოთ მესამე (x_3, y_3) წერტილის კოორდინატები. როგორც ვნახეთ, გეომეტრიულად ამ წერტილების შესაკრებად უნდა გავატაროთ წრფე ამ ორ წერტილზე. დავუშვათ, ამ წრფის განტოლებაა

$y=\alpha x+\beta$ (ეს წრფე არ იქნება x ღერძის პერპენდიკულარი, როგორც ეს გვაქვს მესამე შემთხვევაში), მაშინ შეგვიძლია დავწეროთ, რომ

$$\alpha=\frac{y_2-y_1}{x_2-x_1} \qquad \text{და} \quad \beta=y_1-\alpha x_1 .$$

ამ წრფის წერტილი $(x,\alpha x+\beta)$ მაშინ და მხოლოდ მაშინ შეიძლება ეკუთვნოდეს წირს, თუ შესრულდება პირობა

$$(\alpha x+\beta)^2=x^3+ax+b \ ,$$

საიდანაც მივიღებთ, რომ რადგანაც განტოლებას

$$x^3-(\alpha x+\beta)^2+ax+b=0$$

აქვს მხოლოდ სამი განსხვავებული ამონახსნი, ეს წრფე მხოლოდ სამჯერ კვეთს E წირს, ამასთან ორი გადაკვეთა იქნება $P(x_1,y_1)$ და $Q(x_2,y_2)$ წერტილებში, ანუ x_1,x_2 წარმოადგენენ განტოლების ამონახსნებს. მაშინ, ვიეტის თეორემის თანახმად, $x_3=\alpha^2-x_1-x_2$. მარტივი გამოთვლებით მივიღებთ, რომ

$$x_3=\left(\frac{y_2-y_1}{x_2-x_1}\right)^2-x_1-x_2 \quad \text{და} \quad y_3=\left(\frac{y_2-y_1}{x_2-x_1}\right)(x_1-x_3)-y_1. \quad (10.6.)$$

ერთი და იგივე წერტილის შეკრება. როდესაც $P=Q$, როგორც ნახაზიდან ჩანს, გვექნება წირის მხები P წერტილში. დავუშვათ, რომ α ეხლა არის წარმოებული dy/dx წერტილში P. არაცხადი ფუნქციის წარმოებული, რომელიც მოცემულია განტოლებით (10.6), მოგვცემს, რომ

$$\alpha=\frac{3x_1^2+a}{2y_1}$$

და თუ ჩვენ ამ მნიშვნელობას ჩავსვავთ (10.6.) ტოლობებში მივიღებთ:

$$x_3=\left(\frac{3x_1^2+a}{2y_1}\right)^2-2x_1 \quad \text{და} \quad y_3=\left(\frac{3x_1^2+a}{2y_1}\right)(x_1-x_3)-y_1.$$

მოპირდაპირე წერტილების შეკრება. როდესაც გვინდა ორი მოპირდაპირე წერტილების $P(x,y)$ და $P(x,-y)$ შეკრება, მათი ჯამი იქნება უსასრულობის წერტილი, რომელსაც ცხადია კოორდინატები არ გააჩნია.

10.7. ჰაუფის რიგი. დავუშვათ, E არის ელიფსური წირი, განსაზღვრული $F(q)$ ველზე. აღვნიშნოთ $E(F_q)$-თი ის წერტილები, რომლებიც ერთდროულად ეკუთვნიან ელიფსურ წირს და $F(q)$ ველს. მაშინ ამ წერტილების სიმრავლე $\left|E(F_q)\right|$ იქნება წირის რიგი $F(q)$ ველზე. არსებობს ჰასეს თეორემა, რომლითაც მტკიცდება, რომ ეს სიმრავლე მოთავსებულია

$$[E(F_q)+1-2\sqrt{q},E(F_q)+1+2\sqrt{q}]$$

ინტერვალში. სხვანაირად ჰასეს თეორემა შეიძლება ჩამოვაყალიბოთ შემდეგნაირად: თუ E არის ელიფსური წირი, განსაზღვრული $F(q)$ ველზე, მაშინ $\left|E(F_q)\right|=q+1-t$ სადაც $|t|\le 2\sqrt{q}$. t-ს ეწოდება E ***წირის კვალი*** $F(q)$ ველზე. მტკიცდება, რომ ნებისმიერი მარტივი p რიცხვისა და მთელი t-სთვის, რომლებიც აკმაყოფილებენ პირობას $|t|\le 2\sqrt{p}$ არსებობს E ელიფსური წირი, განსაზღვრული $F(q)$ ველზე, რომლისთვისაც $\left|E(F_q)\right|=p+1-t$.

თუ E ელიფსური წირი განსაზღვრულია $F(q)$ ველზე, მაშინ ის განსაზღვრული იქნება ამ ველის ნებისმიერ გაფართოებაზე. შესაძლებელია ვაჩვენოთ, რომ ასეთ შემთხვევაში სამართლიანია შემდეგი თეორემა. თუ E ელიფსური წირი განსაზღვრულია $F(q)$ ველზე და

$$\left|E(F_q)\right|=q+1-t \ ,$$

მაშინ

$$\left|E(F_{q^n})\right| = q+1-V_n\ ,$$

სადაც $\{V_n\}$ არის შემდეგი სახის მიმდევრობა: $V_0 = 2, V_1 = t$ და $V_n = V_1 V_{n-1} - qV_{n-2}, n \geq 2$.

განვიხილოთ მაგალითი. დავუშვათ, მოცემული გვაქვს E ელიფსური წირი

$$y^2 = x^3 + x + 6 \quad (10.7.)$$

განსაზღვრული F_{11} ველზე. დავადგინოთ წერტილები, რომლებიც ეკუთვნიან E წირს ამ ველზე. ამისათვის ყველა $x \in F_{11}$ წერტილებისათვის უნდა გამოვთვალოთ $z = x^3 + x + 6$ და მიღებული მნიშვნელობებისათვის ამოვხსნათ (10.7.) განტოლება y-ის მიმართ. როდესაც $x = 0$, მივიღებთ, რომ $x^3 + x + 6 = 6$. შევამოწმოთ ეილერის კრიტერიუმით, არის თუ არა ეს რიცხვი კვადრატული ნაშთი. ამისათვის ჩავატაროთ გამოთვლები:

$$6^{(11-1)/2} (\mathrm{mod} 11) \equiv 7776 (\mathrm{mod} 11) \equiv 10 (\mathrm{mod} 11) \equiv -1 (\mathrm{mod} 11)\ .$$

ესე იგი $x = 0$ არ ეკუთვნის E წირს ამ ველზე. დავუშვათ, ეხლა $x = 2$, მაშინ $x^3 + x + 6 (\mathrm{mod} 11) = 8 + 2 + 6 (\mathrm{mod} 11) = 5$. გამოვიყენოთ ეილერის კრიტერიუმი, გვექნება $3125 (\mathrm{mod} 11) \equiv 1$, რაც იმას ნიშნავს, რომ წერტილი $x = 2$ ეკუთვნის E წირს ამ ველზე. ვიპოვოთ შესაბამისი კვადრატული ფესვები. როგორც ბლუმის რიცხვების განხილვის დროს ვნახეთ, როდესაც სრულდება პირობა $q \equiv 3 (\mathrm{mod} 4)$, (10.7.) განტოლების ფესვები შეიძლება გამოითვალოს ფორმულით

$$y = \pm z^{(q+1)/4} (\mathrm{mod}\, q)\ .$$

ამიტომ მივიღებთ, რომ $y = \pm 5^3 (\mathrm{mod} 11) = \pm 125 (\mathrm{mod} 11) = \pm 4$. მივიღეთ E წირის ორი წერტილი $(2,4)$ და $(2,7)$. ამ გამოთვლების შედეგები ნაჩვენებია ცხრილში (10.1).

როგორც ამ ცხრილიდან ჩანს, მივიღეთ თორმეტი წერტილი. ესე იგი E წირს F_{11} ველზე სულ ექნება ცამეტი წერტილი (ერთეულოვანი წერტილის დამატებით). შევკრიბოთ ორი წერტილი, დავუშვათ, $(2,4)$ და ზემოთ მოყვანილი ფორმულების თანახმად გვექნება, რომ

ცხრილი 10.1.

x	$x^3 + x + 6 (\mathrm{mod} 11)$	კვადრატული ნაშთია თუ არა	y
0	6	არა	
1	8	არა	
2	5	დიახ	4,7
3	3	დიახ	5,6
4	8	არა	
5	4	დიახ	2,9
6	8	არა	
7	4	დიახ	2,9
8	9	დიახ	3,8
9	7	არა	
10	4	დიახ	2,9

$$x_3 = (y_2 - y_1)^2 \times (x_2 - x_1)^{-2} - x_1 - x_2 (\mathrm{mod} 11) = 5^2 (\mathrm{mod} 11) \times (3^{-1})^2 (\mathrm{mod} 11) - 2 - 5 (\mathrm{mod} 11) =$$
$$3 \times 5 - 7 (\mathrm{mod} 11) = 8.$$

$$y_3 = ((y_2 - y_1)(x_2 - x_1)^{-1}(x_1 - x_3) - y_1)(\mathrm{mod} 11) = (5 \times 4 \times (-6) \,\mathrm{mod} 11 - 4)(\mathrm{mod} 11) =$$
$$(9 \times 5 (\mathrm{mod} 11) - 4)(\mathrm{mod} 11) = 1 - 4 (\mathrm{mod} 11) = 8.$$

მივიღეთ წერტილი $(8,8)$.

გამოვთვალოთ ეხლა წერტილი $2P$, თუ $P(2,7)$. ამ შემთხვევაში გვექნება

$x_3 = ((3x_1^2+1)^2(2y_1)^{-2} - 2x_1)(\text{mod}11) = 5$ $y_3 = ((3x_1^2+1)(2y_1)^{-1}(x_1 - x_3) - y_1)(\text{mod}11) = 2$.

თუ გავაგრძელებთ ამ გამოთვლებს, მივიღებთ, რომ

$3P = (8,3); 4P = (10,2); 5P = (3,6); 6p = (7,9); 7P = (7.2); 8P = (3,5); 9P = (10,9); 10P = (8,8)$

$11P = (5.9); 12P = (2,4),$

რაც იმაზე მიუთითებს, რომ $P(2,7)$ ყოფილა F_{11}^* ციკლური ჯგუფის წარმომქმნელი ელემენტი.

ცხადია პრაქტიკულად საინტერესო q-თვის E ელიფსური წირების $|E(F_q)|$ წერტილების სიმრავლის დათვლა ხელით შეუძლებელია, მაგრამ არსებობს პოლინომიალური ალგორითმი **SEA(Scoohcoof-Elkies-Atkin),** რომელიც წარმოადგენს დღეისათვის საუკეთესო ალგორითმს ნებისმიერი ელიფსური წირის წერტილების გამოსათვლელად მარტივ და ოპტიმალურად გაფრთოებულ ველებზე.

ელიფსური წირების აფინურ სივრცეში წარმოდგენის დროს ჯგუფური კანონის საშუალებით არითმეტიკული ოპერაციების ჩატარებისას აუცილებელია გაყოფის (სასრული ველების შემთხვევაში ინვერსიის) ოპერაცია რაც დაკავშირებულია გარკვეულ სიძნელეებთან. კერძოდ, ინვერსიის ოპერაციისათვის აუცილებელი ხდება ევკლიდეს გაფართოებული ალგორითმის გამოყენება, რომელსაც მართალია აქვს იგივე სირთულე, რაც გამრავლების ალორითმს, მაგარამ მაინც, როგორც წესი, მისი რეალიზაცია არასაკმარისად ეფექტურია.

იმისათვის, რომ თავიდან ავიცილოთ ინვერსიის ოპრაცია და გამოთვლები ჩავატაროთ უფრო ეფექტურად, გამოიყენება ელიფსური წირის წერტილების წარმოდგენა პროექციული კოორდინატების საშუალებით. $P^2(K)$ პროექციული სიბრტყე K ველზე განისაზღვრება როგორც ისეთი სამეულების (X,Y,Z) სიმრავლე, რომელშიც სამივე ელემენტი ერთდროულად არ უდრის ნოლს და $X,Y,Z \in K$ ველს. ამ სამეულებისათვის შემოტანილია ექვივალენტობის მიმართება შემდეგი სახით: $(X,Y,Z) \sim (\lambda^c X, \lambda^d Y, \lambda Z)$ ნებისმიერი $\lambda \in K$. მაგალითად, $P^2(F_{11})$-ის შემთხვევაში წერტილები $(4,3,2)$ და $(9,4,10)$ არიან ექვივალენტურები ($\lambda = 5, c = d = 1$). ექვივალენტობის კლასი აღინიშნება როგორც $(X:Y:Z)$ და წარმოადგენს

$$(X:Y:Z) = \{\lambda^c X, \lambda^d Y, \lambda Z : \lambda \in K^*\}$$

სიმრავლეს. $(X:Y:Z)$-ს ეწოდება ***პროექციული წერტილი,*** ხოლო (X,Y,Z) არის მისი წარმომადგენელი. პროექციული წერტილების სიმრავლე K ველზე აღინიშნება $P(K)$-თი.

თუ $(X',Y',Z') \in (X:Y:Z)$ მაშინ $(X',Y',Z') = (X:Y:Z)$,

ანუ ექვივალენტობის კლასის ნებისმიერი წარმომადგენელი შეიძლება იყოს ამ კლასიის წარმომადგენელი. პრაქტიკულად ეს იმას ნიშნავს, რომ თუ $Z \neq 0$, მაშინ $(X/Z^c, Y/Z^d, 1)$ არის $(X:Y:Z)$ პროექციული წერტილის წარმომადგენელი $Z = 1$ კოორდინატით. რაც იმას ნიშნავს, რომ ჩვენ გვაქვს ურთიერთცალსახა შესაბამისობა პროექციულ

$$P(K)^* = \{(X:Y:Z) : X < Y < Z \in K, Z \neq 0\}$$

და აფინურ $A(K) = \{(x,y) : x, y \in K\}$ წერტილებს შორის.

$P(K)^0 = \{X:Y:Z), X,Y,Z \in K, Z = 0\}$ პროექციული წერტილების სიმრავლეს ეწოდება წრფე უსასრულობაში და მისი წერტილები არ შეესაბამება არცერთ აფინურ წერტილს.

ვეიერშტრასის (10.1) განტოლების პროექციული ფორმა მიიღება x და y აფინური კოორდინატების პროექციული კოორდინატებით ჩანცვლების გზით. მაგალითად, თუ

$$y^2 = x^3 + ax + b$$

განტოლებაში გამოვიყენებთ ჩანაცვლებას $x \to X/Z^2$ და $y \to Y/Z^3$ (ამ კოორდინატებს უწოდებენ ***იაკობიანის პროექციულ კოორდინატებს***), მივიღებთ

$$Y^2 = X^3 + aXZ^4 + bZ^6$$

განტოლებას.

წერტილების ***შეკრება*** $(X_3,Y_3,Z_3) = (X_1,Y_1,Z_1) + (X_2,Y_2,Z_2)$ ხდება შემდეგი ფორმულების გამოყენებით $\lambda_1 = X_1 \cdot Z_2^2$ $\lambda_2 = X_2 \cdot Z_1^2$ $\lambda_3 = \lambda_1 - \lambda_2$

$$\lambda_4 = Y_1 \cdot Z_2^3 \qquad \lambda_5 = Y_2 \cdot Z_1^3 \qquad \lambda_6 = \lambda_4 - \lambda_5$$
$$\lambda_7 = \lambda_1 + \lambda_2 \qquad \lambda_8 = \lambda_4 + \lambda_5 \qquad Z_3 = \lambda_3 Z_1 Z_2$$
$$X_3 = \lambda_6^2 - \lambda_7 \lambda_3^2 \qquad \lambda_9 = \lambda_7 \lambda_3^2 - 2X_3 \qquad Y_3 = (\lambda_9 \lambda_6 - \lambda_8 \lambda_3^2)/2 .$$

რაც შეეხება წერტილის ***გაორმაგებას***, ეს ხდება შემდეგი გზით: $(X_3, Y_3, Z_3) = 2(X_1, Y_1, Z_1)$ და გამოიყენება ფორმულები:

$$\lambda_1 = 3X_1^2 + aZ_1^4 \qquad \lambda_2 = 4X_1 Y_1^2 \qquad \lambda_3 = 8Y_1^4$$
$$Z_3 = 2Y_1 Z_1 \qquad X_3 = \lambda_1^2 - 2\lambda_2 \qquad Y_3 = \lambda_1(\lambda_2 - X_3) - \lambda_3$$

როგორც ვხედავთ, აქ მხოლოდ ერთხელ გვხვდება გაყოფა და ისიც მუდმივზე (2) , ამიტომ მოცემული ველისათვის ჩვენ შეგვიძლია წინასწარ გამოვთვალოთ მისი შებრუნებული და დავზოგოთ გამოთვლებისათვის საჭირო დრო.

ციფრული ხელმოწერის ახალ სტანდარტში (და ცხადია ახალ ალგორითმშიც) საიდუმლო გასაღებად აიღება რომელიმე წერტილი და ღია გასაღების წარმოადგენს ამ წერტილის თავის თავთან რამდენჯერმე შეკრებით მიღებული წერტილი.

ახალ სტანდარტს ეწოდება ციფრული ხელმოწერის ალგორითმი ელიფსური წირებით (**Elliptic Curve Digital Signature Slgorithm - ECDSA**) და ძალაშია 2001 წლიდან.

საკონტროლო კითხვები:

1. რა კრიტერიუმებს უნდა აკმაყოფილებდეს ციფრული ხელმოწერის ალგორითმი?
2. კარგად დაუკვირდით ციგრული ხელმოწერის სტანდარტს და დაადგინეთ ხელმოწერა ხდება გადასაცემ ტექსტზე, თუ ამ ტექსტის ჰეშ-ფუნქციაზე?
3. აღწერეთ ხელმოწერის და შემოწმების პროცედურები ციფრული ხელმოწერის სტანდარტში.
4. აჩვენეთ, რატომ სრულდება პირობა $r = \nu$ როდესაც ხელმოწერა კორექტულია?
5. რატომ გახდა საჭირო ელიფსური წირების გამოყენება ციფრული ხელმოწერის ალგორითმში?
6. რა პროცედურა შეიცვალა ალგორითმში ელიფსური წირების გამოყენების შემდეგ?
7. როგორი სახის წირებს ეწოდებათ ელიფსური წირები?
8. როგორ ხდება წერტილების შეკრება ელიფსურ წირებზე?
9. რომელი წერტილი ასრულებს ერთეულოვანი ელემენტის როლს ელიფსური წერტილების ჯგუფში შეკრების ოპერაციის მიმართ?