ლექცია 4. ღია გასაღები

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* *Y* . *f* ფუნქცია იქნება ცალმხრივ მიმართული ფუნქცია, თუ ნებისმიერი *x**X* -თვის ადვილია *y* *f* (*x*) მნიშვნელობის გამოთვლა, მაგრამ თითქმის ყველა *y**E*( *f* ) -თვის ისეთი *x**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 ზომებზე და მისი შესრულება პოლინომურ დროში ყოველთვის არ ხერხდება. რაც ნიშნავს სწორედ იმას, რომ არსებობენ მცირე რაოდენობის *y**E*( *f* ) , რომლებისთვისაც ადვილი იქნება ისეთი

*x**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 წარმოდგენაში, 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).

მაგრამ აქ შევეხოთ მოდულებით არითმეტიკას:

თუ ორი რიცხვი a და b p რიცხვზე გაყოფისას გვაძლევს ერთი და იმავე ნაშთს, ვიტყვით, რომ a და b რიცხვები შეედარებიან ერთმანეთს p მოდულით. შედარება ჩაიწერება შემდეგნაირად *a=b(mod p*). ორი რიცხვის შედარებადობის ექვივალენტური განმარტებაა მათი სხვაობის უნაშთო გაყოფა p -ზე.

შედარებადობის ძირითადი თვისებები:

1 . თუ *a=c(mod p*) და *b=c(mod p*) მაშინ *a=b(mod p*).

2. შედარებებზე ისეთნაირი მოქმედებების შესრულება შეიძლება, რომელიც შესრულდება ტოლობებზე: თუ *a1=b1(mod p*) და *a2=b2(mod p*) მაშინ (*a1+ a2 )= (b1+ b2) (mod p*) ; (*a1- a2 )= (b1- b2) (mod p*) და *a1d=b1d(mod p*) სადაც *d* ნებისმიერი მთელი რიცხვია.

დავუშვათ, *n* და *a* ორი ისეთი მთელი რიცხვია, რომ 1*a* *n* და *Z* {1,2,...,*n* 1} *n* . მაშინ ხარისხში აყვანის ფუნქცია *f a, n*: *Z* *Z n*, მოდულით *n* , იქნება *fa, n*(*m*) *a* *m*(mod *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*(mod *n*) ((((((*a2* mod *n*) *a*)mod *n*)2 mod *n*)2 mod *n*)2 *a*)mod *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*(mod *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* *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 *y*  *f k*(*x*)

ტოლობას, 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 n, m*(*a*) *a* *m*(mod *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*(mod *n*) .

მაგალითად, 5 არის მეოთხე ხარისხის ფესვი 16-დან მოდულით 16, რადგანაც 5 4 (mod 21) 16, მაგრამ ცხადია, რომ 2 -იც ასევე მეოთხე ხარისხის ფესვია თექვსმეტიდან მოდულით 21, რადგანაც 24 (mod 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* *q* სახის შედგენილ რიცხვს, სადაც *p* *q* 3(mod 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**Z*იყოს სიმრავლე იმ რიცხვებისა, რომლებიც წარმოადგენენ კვადრატულ ნაშთებს *n* ფუძით. მაგალითად, დავუშვათ, რომ *p* 19 და *q* 23, მაშინ *n* 1923 437.

135*Z*, მაგრამ 133 197*Z*. 135 არ წარმოადგენს კვადრატულ ნაშთს, რადგანაც არ არსებობს ისეთი მთელი რიცხვი 1*a* *n* და *a* 2135(mod 437) , მაშინ როდესაც 24 2576 139(mod 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

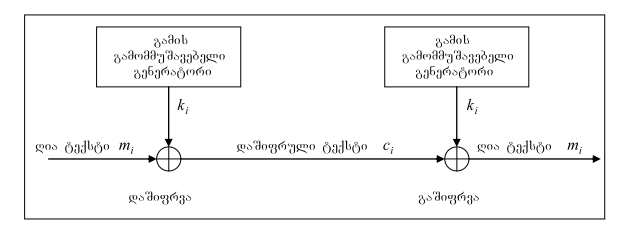
ფესვები” 24,185,253,413, რომელთაგანაც მხოლოდ 24 წარმოადგენს პირველყოფილ ფესვს.

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.

**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 მოცმეულია სურ. 1



სურათი 1. ნაკადური შიფრი

გენერატორი, რომელსაც ზოგჯერ უწოდებენ მორბენალი გასაღების გენერატორს (**running key generator)** გამოიმუშავებს ბიტების ნაკადს , რომელსაც უწოდებენ გამას. ეს გამა და ღია ტექსტის ბიტები იკრიბება ოპერაცია **XOR-**ით და მიიღება შიფროტექსტი: . დეშიფრაციის დროს კვლავ იკრიბება უკვე შიფროტექსტი და გამა და მიიღება ღია ტექსტი. ასეთი შიფრის საიმედოობა მთლიანად დამოკიდებულია გენერატორის მიერ გამომუშავე- ბულ 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 ბიტიში გამოიწვევს შეცდომის გავრცელებას *n* ბიტზე, რის

შემდეგ გენერატორები კვლავ მოვლენ სინქრონიზაციაში.

**სიმეტრიული კრიპტოალგორითმების კრიპტოანალიზის მეთოდები**

**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 ალგო-რითმზე 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 კონფიდენცია- ლურობის დასაცავად მისი ქსელში გადაცემის დროს გამოიყენება პროტოკოლი, რომლის მედე-გობაც ნაკლებია შიფრის მედეგობაზე, ცხადია მოწინააღმდეგე შეტევას განახორიციელებს პროტო-კოლზე და არა შიფრზე. საბოლოოდ შეგვიძლია დავასკვნათ, რომ კრიპტოსისტემის მედეგობა ტოლია ამ სამი ფაქტორიდან ყველაზე სუსტის კრიპტომედეგობის. ამიტომ აუცილებელია, რომ ეს კრიპტომედეგობები იყოს ერთმანეთთან ახლოს.

**კრიპტომედეგობის რაოდენობრივი შეფასების კრიტერიუმები.** კრიპტომედეგობა შეტევების მიმართ რაოდენობრივად შეიძლება შევაფასოთ იმ რესურსებით, რომლებიც საჭიროა წარმატე-ბული 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 შესანა-ხად.

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

**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 ვიპო- ვით 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 255-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 ევას, რომელსაც სურს ამ ალგორითმის გატეხვა და რომელიც უტევს შერჩეული ღია ტექსტის საფუძველ ზე, გააჩნია იმდენი დრო, რომ მას შეუძლია წინასწარ, სანამ ალისა გადასცემს ბობს დაშიფრულ ტექსტს, ჩაატაროს ნებისმიერი გამოთვლები. მაშინ მას შეუძლია, P ფიქსირებული ღია ტექსტის ერთი ბლოკისათვის ყველა შესაძლო გასაღებებით გამოთვალოს შესაბამისი Ck =Ek (P) შიფროტექსტი და შეადგინოს ცხრილი წყვილებისა(CK ,K), რომელსაც მოაწესრიგებს პირველი კოორ-დინატის მიხედვით. ასეთი ცხრილის არსებობის შემთხვევაში ევას აღარ გაუჭირდება ალისას და ბობის მიერ შერჩეული ღია გასაღების გამოცნობა. ამისათვის საკმარისია ევამ გაუგზავნოს ალისას (ან ბობს) შერჩეული ღია ტექსტი და მიიღოს შესაბამისი შიფროტექსტი, ის თავის ცხრილში ადვილად იპოვნის იმ გასაღებს, რომელსაც იყენებენ ალისა და ბობი. ამ შემთხვევაში ევას აღარ ექნება დროის პრობლემა შიფროტექსტის გაშიფრვისას, მაგრამ გაუჩნდება მეხსიერების პრობლემა, რადგანაც იმ ცხრილის შენახვას, რომელსაც ევა 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 ევამ 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.