ПОРЕДИЦА „ЗАЩИТА НА ИНФОРМАЦИЯТА“

Веселин Целков

ВЪВЕДЕНИЕ В КРИПТОГРАФИЯТА И СИГУРНОСТТА НА ДАННИТЕ

ЗА БУКВИТЕ О ПИСМЕНЕХЪ

SECNET

Сигурна защита
на информацията

Ние предлагаме:

- Разработване, внедряване, поддържане и развитие на интегрирани системи за защита на информацията в компютърни системи или мрежи, базирани на оригинални български хардуерни и софтуерни криптографски решения
- Доставка и оборудване, изпълнено по TEMPEST стандартите на НАТО - SDIP-27 за офиса, АИС или мрежи, както и за използване в ектсремални условия на околната среда
- Консултации, проектиране и изграждане на защитени компютърни системи или мрежи
- Консултации при подготовката за акредитиране на АИС или мрежи

Адрес: ул. "Панорама София" 6, Бизнес Център Ричхил, бл. Б, ет.2
Телефон: +359 2 978 11 51 | email: office@secnet.technology

ПОРЕДИЦА „ЗАЩИТА НА ИНФОРМАЦИЯТА“

ВЪВЕДЕНИЕ В КРИПТОГРАФИЯТА И СИГУРНОСТТА НА ДАННИТЕ

Веселин Целков

Академично издателство „За буквите – О писменехъ“

София, 2021

ISBN 978-619-185-460-8

В ПАМЕТ НА МОЯ СЪСТУДЕНТ, КОЛЕГА И ПРИЯТЕЛ

ВЛАДИМИР КОЖУХАРОВ

СЪЗДАТЕЛ И РЪКОВОДИТЕЛ НА НАЦИОНАЛНОТО ЗВЕНО ПО КРИПТОГРАФСКА СИГУРНОСТ В РЕПУБЛИКА БЪЛГАРИЯ

СЪДЪРЖАНИЕ

Доц. д-р Мая Божилова

CONTENTS

CHAPTER 3. CLASSICAL CRYPTOGRAPHY

3.1. Substitution-based methods. One letter encryption

3.1.1. Atbash

3.1.2. ROT1 (k)

3.1.3. Polybius Square

3.1.4. Caesar Cipher

3.2. Substitution-based methods. Very alphanumeric encryption

3.2.1. Playfair Cipher

3.2.2. Hill Cipher

3.2.3. Double Playfair Cipher

3.2.4. Four Matrices Cipher

3.3. Substitution-based methods. Poly-Alphabetic Ciphers

3.3.1. Vigenère Cipher

3.3.2. Jefferson's disk

3.3.3. Alberti's disk

3.4. Transposition-based methods

3.4.1. Early Spartan Cipher

3.4.2. Rail-Fence Cipher

3.4.3. Route Method

3.4.4. Row - Column Cipher

3.4.5. Repeated Row - Column Cipher

3.5. Rotor Machines

CHAPTER 4. MODERN CRYPTOGRAPHY

4.1. Modern ciphers. General

4.2. Modern block ciphers. Feistel Ciphers

4.3. Modern block ciphers. Data encryption standard (DES)

4.4. DES development and encryption of large messages with DES

4.4.1. Development of DES

4.4.2. Encryption of large messages with DES

ПРЕДГОВОР

ВЛАДИМИР ХРИСТОВ КОЖУХАРОВ

11.03.1955 г. – 30.04.2018 г.

СЪЗДАТЕЛ И РЪКОВОДИТЕЛ НА НАЦИОНАЛНОТО ЗВЕНО ПО КРИПТОГРАФСКА СИГУРНОСТ В РЕПУБЛИКА БЪЛГАРИЯ

Роден е във Велико Търново, където завършва средно образование през 1973 г. във Втора гимназия с отличен успех и златен медал.

Завършва висше образование в Софийския университет „Св. Климент Охридски“, факултет по математика и механика, с отличен успех, магистърска степен по „Изследване на операциите“.

Започва работа септември 1980 г. в Комитета по ЕССИ (Комитет по единна система за социална информация при Министерския съвет, сега НСИ – Национален статистически институт), където работи до януари 1986 г. и стига до длъжността директор на дирекция.

С постъпването си в МВР, Четвърти самостоятелен отдел на Държавна сигурност (по-късно Служба „Защита на средствата за връзка“ в МВР и Дирекция „Защита на средствата за връзка“ в ДАНС) през 1986 г., Владимир Кожухаров насочва изцяло своите математически и компютърни способности в областта на приложната криптография.

След промените през 1989 г. в България остава в пълна изолация по отношение на техниката и технологията за защита на информацията – Съветският съюз се разпада и отказва по-дългосрочно сътрудничество, а дотогава именно там са произвеждани голямата част от криптографската техника и ключовите материали за НРБ (Народна Република България). В

същото време страната е в черните списъци на Запада по отношение на всички чувствителни технологии.

Именно в този период, Владимир Кожухаров успява да разработи почти сам технология за производство на собствени ключови материали за съветската техника, а по-късно в колектив – и напълно независими, български софтуерни и хардуерни средства за криптографска защита на информацията, включително до най-високо ниво. Тези му усилия и постижения имат отношение не толкова за напредъка на специалните служби в областта на математиката, криптографията, информационните технологии и електрониката, колкото до факта, че за първи път в следосвобожденската история на България е постигнат реален суверенитет на страната по отношение на опазване на държавната тайна. До този момент средствата за защита на информацията, използвани в България, са били зависими или изцяло произвеждани и подслушвани в годините от Русия, Франция, Германия и Съветския съюз.

В периода на сближаване на Република България със западните демокрации, възниква и необходимостта от разработка на нормативна уредба, синхронизирана с тази на НАТО и Европейския съюз.

Като ръководител на структурното звено, което изпълнява функциите на Национален орган по криптографската сигурност и Орган по сигурността на комуникационните и информационните системи, Владимир Кожухаров има решаваща роля, особено при разработката на процедурите, залегнали в Наредбата за криптографската сигурност на класифицираната информация и Наредбата за задължителните общи условия за сигурност на автоматизираните информационни системи и мрежи (от 28.02.2020 г. – Наредба за сигурността на комуникационните и информационните системи).

В качеството си на изявен специалист и визионер, Владимир Кожухаров многократно участва в работата на органите по криптографска и компютърна сигурност на НАТО и ЕС в Брюксел.

След като е пенсиониран от ДАНС през 2010 г., Владимир Кожухаров е привлечен като специалист от българска фирма, разработваща криптографски средства. Негова е основната заслуга за успешното приключване на първия проект на фирмата в областта на криптографската защита на информацията.

Владимир Христов Кожухаров отдава по-голямата част от живота си на криптографската и компютърната сигурност – както в ролята си на специалист, така и като основоположник на нормативната уредба в тези области.

В основата на цялостната му дейност е мечтата му да бъде постигнат реален суверенитет на Република България по отношение на опазването на държавната тайна.

ПОКЛОН!

март 2021 г.
гр. София

Д-р Виолета Дъчева
Директор на дирекция,
Държавна агенция „Национална сигурност“

ПРЕДИСЛОВИЕ

Утвърдилата се вече сред академичните среди поредица от сериозни научни публикации на тема „Защита на информацията“ се разраства с ново заглавие, изследващо проблемите на криптографията. Книгата „ВЪВЕДЕНИЕ В КРИПТОГРАФИЯТА И СИГУРНОСТТА НА ДАННИТЕ“ представя систематизирано знание в предметната област.

Информационната революция създаде област на сигурността, в която поради относително ниските разходи, могат да участват както държавни, така и други организации. Много автори подчертават факта, че самата природа на информационните технологии е такава, че широк кръг от участници могат да провеждат операции с национално значение. Към днешна дата, няма държавна структура или частна организация, която да няма присъствие в интернет пространството. Това присъствие е застрашено от възможността за провеждане на кибер атаки срещу ресурсите или услугите на съответната организация. Информацията и информационните въздействия се явяват един от основните начини за стимулиране развитието на съвременното общество. Безспорно е, че в съвременния век на „информационните цунамита“ е необходимо да са постави въпросът за начините за защита на информацията.

Актуалността на темата на представения труд се обуславя от необходимостта да се изследват информационните технологии, кибер пространството, и в частност алгоритмите, системите и процесите за защита на информацията, чрез използването на криптографски средства.

В своето изложение проф. д.т.н. Веселин Целков по своя неповторим начин запознава читателите с основите на криптографията, като последователно разкрива същността и взаимовръзките, започвайки с основните постулати на защита на информацията – цели и принципи при изграждане на системи за сигурност на КИС. Продължавайки своето представяне, авторът ни запознава с различните модели, приложими в

предметната област, а именно „Модел на заплахите за сигурността“ и „Модел на услугите за сигурност“. В следващата част на книгата – „Въведение в криптографията и сигурността на данните“, на читателя е предоставена възможността да се запознае с основите на криптографската наука чрез разглеждане на въпросите за типовете криптографски алгоритми и моделите за управление на криптографски ключове. В тази част авторът представя пред обществото своето виждане за „…Единно решение CSS за криптографска система за защита на информацията в корпоративни информационни системи…“.

Глава 3 на книгата е посветена на класическата криптография. В нея авторът умело води и запознава читателя през различните етапи на развитието на криптографската наука, започвайки от алгоритми с еднобуквено криптиране, продължавайки с „Методи базирани на заместванията. Много буквено криптиране“ и „Методи базирани на заместванията. Много азбучни шифри“. Специално внимание е отделено на методите базирани на размесвания и на популярните роторни шифрови машини.

В следващата глава на изложението, читателите имат възможност да се запознаят и да изучат основите на съвременната криптография и свързаните с нея различни видове криптографски алгоритми и подходи. Авторът на книгата последователно представя своя начин на виждане и интерпретиране на съвременни криптографски схеми и алгоритми като: Схема на Фейстел; Data encryption standard (DES) и неговите разновидности; Advanced Encryption Standard (AES).

Обърнато е внимание и на криптографските алгоритми с публичен ключ, като са разгледани: RSA; криптосхема на Рабин; Ел–Гамал (ElGamal); Дифи–Хелман(Diffie–Hellman); алгоритъм за цифров подпис (Digital Signature Algorithm, DSA) и схема на Feige–Fiat–Shamir. Елиптичните криви,

като един от сравнително най-новите инструменти използвани за криптографска защита на информацията, са представени в края на тази част.

Книгата може да се използва както от студенти изучаващи проблемите и практическите решения за криптографска защита на информацията, така и от докторанти и от всички желаещи да се запознаят по-подробно с основите на защитата на информацията, криптографията като наука, различните криптографски схеми и алгоритми, начините на тяхното дефиниране, формиране на задачите за защита и приложение в практиката.

март 2021 г.
гр. София

Доц. д-р Николай Стоянов
Заместник директор на Институт по отбрана,
Министерство на отбраната

FOREWORD

The series of serious scientific publications on “Information Security”, which has already established itself among academics, is expanding with a new title exploring the issues of cryptography. The book “INTRODUCTION TO CRYPTOGRAPHY AND DATA SECURITY” presents a systematic knowledge in the subject area.

The information revolution has created an area of security in which, due to relatively low costs, both government and other organizations can participate. Many authors emphasize the fact that the very nature of information technologies is such that a wide range of participants can carry out operations of national importance. To date, there has been no government agency or private organization that does not reside on the Internet. This presence is threatened by the possibility of cyberattacks against the resources or services of the organization. Information and information impacts are one of the main ways to stimulate the development of modern society. Undoubtedly, in the modern age of the “information tsunamis” it is necessary to raise the question of ways of protecting information.

The relevance of the topic of the presented work has been determined by the need to study information technologies, cyberspace, and in particular algorithms, systems and processes for information protection through the use of cryptographic means.

In his exposition, Prof. DSc Veselin Tselkov in his unique way acquaints readers with the basics of cryptography, consistently revealing the nature and relationships, starting with the basic postulates of information protection - goals and principles in building security systems of CIS. The author then introduces us to the various models applicable in the subject area, namely the “Security Threat Model” and the “Security Services Model”. In the next part of the book –

“Introduction to cryptography and data security”, the reader is given the opportunity to learn the basics of cryptographic science by ways of addressing the issues of types of cryptographic algorithms and models for managing cryptographic keys. In this part, the author presents to the readership his vision for “… A unified CSS solution for cryptographic system for information protection in corporate information systems…”.

Chapter 3 of the book is devoted to classical cryptography, where the author skillfully guides and introduces the reader through the various stages of development of cryptographic science, starting with algorithms with one-letter encryption, continuing with “Substitution-based methods. Multi-letter encryption” and Substitution-based methods. Poly-Alphabetic Ciphers. Special attention is paid to methods based on mixing and popular rotors cipher machines.

In the next chapter of the exposition, readers have the opportunity to acquaint themselves with and learn the basics of modern cryptography and related different types of cryptographic algorithms and approaches. The author of the book consistently presents his way of seeing and interpreting modern cryptographic schemes and algorithms such as: Feistel Scheme; Data Encryption Standard (DES) and its variants; Advanced Encryption Standard (AES).

Attention is also paid to public key cryptographic algorithms, such as: RSA; Rabin's cryptogram; El-Gamal; Diffie-Hellman; Digital Signature Algorithm (DSA) and Feige-Fiat-Shamir scheme. Elliptic curves, as one of the relatively new tools used to cryptographically protect information, are presented at the end of this section.

The book can be used by students learning about the problems and practical solutions for cryptographic information protection, as well as by doctoral students and by anyone wishing to know more about the basics of information security,

cryptography as a science, various cryptographic schemes and algorithms, the ways of their definition, formation of the tasks for protection and application in practice.

March 2021
Sofia

Assoc. Prof. Dr. Nikolay Stoyanov
Deputy Director of the Defense Institute,
Ministry of Defense

ЗА АВТОРА

ПРОФ. Д.Т.Н. ВЕСЕЛИН ЦЕНОВ ЦЕЛКОВ

Веселин Ценов Целков е завършил Факултета по математика и механика на Софийски университет „Св. Климент Охридски“, магистър със специализация по „Математическа логика“.

В. Целков е защитил докторантура (кандидат на техническите науки) на тема „Управление на информацията в мрежи от ЕИМ“ (специалност „Системно програмиране“) през 1990 г. През 1996 г. Веселин Целков е избран за старши научен сътрудник II степен с научна специалност „Информатика“ (Информационна сигурност). През ноември 2008 г. защитава дисертация за научна степен „доктор на техническите науки“ (Информатика, Информационна сигурност). През август 2010 г. му е присъдено научно звание „професор“ (Автоматизирани системи за обработка на информацията и управление). Създател, ръководител и сертифициран инструктор е на първата в София регионална Cisco академия във Военна Академия „Г. С. Раковски“ (2001 г).

Специализирал е „Управление на критичната информационна инфраструктура“ в George Marshal European Centre for Security Studies и Комуникационни и информационни технологии (Cisco Academy) в Технически университет, Букурещ, Румъния.

Веселин Целков е преподавател по множество дисциплини свързани с информационната сигурност в Университета по библиотекознание и информационни технологии, СУ „Св. Климент Охридски“ и ПУ „Паисий Хилендарски“.

Автор е на над 250 монографии, научни публикации и изследвания в областта на сигурността, защитата на информацията и информационните технологии, от които над 80 в чужбина или на международни форуми. Ръководител е на научния колектив издаващ поредицата „ЗАЩИТА НА

ИНФОРМАЦИЯТА“. Известни са над 200 цитирания в страната и чужбина. Научен ръководител е на студенти, специализанти и докторанти в областта на информационните технологии и информационната сигурност.

Веселин Целков е полковник от резерва на Българската армия.

От 2003 г. до 2007 г. Веселин Целков е член на Държавната комисия по сигурността на информацията. На 19.12.2007 г. е избран от Народното събрания за член на Комисията за защита на личните данни, а от 15.04.2014 г. е преизбран за втори мандат за член на Комисията за защита на личните данни.

Приятел и член е на Атлантическия клуб от основаването му. Член на AFCEA – Sofia. Бил е и в Управителния съвет на AFCEA – Sofia.

ВЪВЕДЕНИЕ

Поредицата от сериозни научни публикации в България на тема „Защита на информацията“ успешно се продължава от книгата ***„Въведение в криптографията и сигурността на данните“***. В изложението основно е представено учебното съдържание на едноименния курс от магистърската програма ***„Защита на информацията в компютърните системи и мрежи“*** на Факултета по математика и информатика на Софийския университет „Св. Климент Охридски“. Представени са част от резултатите, получени от проучванията, изследванията и разработките извършени във Военния научно-изследователски институт на Генералния щаб и Института за перспективни изследвания за отбраната на Министерството на отбраната в периода 1990 г. – 2003 г. Отразен е опитът на автора като представител на Република България в подкомитета (*SC/4*) за информационна сигурност (безопасност) на НАТО, участник в работната група на НАТО, разработила архитектурата за информационна сигурност, и член на техническия комитет за управление на проекта „Съвместимост на защитените комуникации“. В книгата е синтезиран и представен опитът на автора, придобит като ръководител на колектива и автор на книгите от поредицата „Защита на информацията“ на Академичното издателство „За буквите – О писменехъ“.

Книгата е структурирана в четири глави, приложения, литература, въведение, заключение и представяне на автора.

В първа глава са представени основите на сигурността на данните. Формулирани са ***целта и принципите*** при изграждане на системата за сигурност на комуникационно-информационните системи. Дефинирани са основните направления за осигуряване на сигурността на данните – ***поверителност, цялостност, наличност, отчетност и гарантираност***. В модела на заплахите за сигурността са дадени определенията за ***атака и инцидент*** и са показани четирите категории атаки в модела – ***подслушване, прекъсване, модифициране и изфабрикуване***. Извършена е класификация

на атаките по различни критерии и е направен анализ на атаките в измеренията – ***атакуващи, средства, достъп, резултати и цели***. Услугите за сигурност са представени в зависимост от тяхното предназначение – ***поддръжка, предотвратяване (превенция) и възстановяване***. Показана е връзката между направленията и услугите за сигурност.

Втора глава е въведение в науката криптография. Представени са основните понятия – ***подател и получател, съобщение и криптиране, алгоритми и шифри, криптографски ключове, симетрични и асиметрични криптографски алгоритми***. Показани са използването на асиметричните криптографски алгоритми и съдържанието и структурата на понятията цифров подпис и инфраструктура на публичните ключове. Дефинирана е криптографска система за защита на информацията. Моделите за управление на криптографските ключове са базирани на критериите – ***групи, нива и време***. Моделите по време са разделени на ***статичен, динамичен и хибриден модел***. Описано е оригинално, ***единно решение за защита на информацията в корпоративни информационни системи***, представен е ***пример за интеграция и разпределение на криптографски ключове*** и ***архитектура на защитена система за обмен на съобщения за специални нужди.***

Трета глава е посветена на класическата криптография. Следвайки историческата хронология и класификацията на криптографските методи по начин на трансформация на откритото съобщение в шифрирано, броя на символите и броя на използваните азбуки, последователно са разгледани:

- ***Методи базирани на заместванията. Едно буквено криптиране***. Представени са алгоритмите – Атбаш (иврит), ROT1(k), Квадрат Полибий и Шифър на Цезар;
- ***Методи базирани на заместванията. Много буквено криптиране***. Представени са алгоритмите – Шифър на Плейфеър (Playfair),

Шифър на Хил (Hill), Шифър двоен Плейфеър и Шифър Четири матрици (квадрата);

- ***Методи базирани на заместванията. Много азбучни шифри***. Представени са алгоритмите – Шифър на Вижинер, Диск на Джеферсън и Диск на Алберти;
- ***Методи базирани на разместванията.*** Представени са алгоритмите – Ранен спартански шифър (*scytale*), Шифър RAIL FENCE (релсова ограда), Метод на маршрута, Шифър РЕД – СТЪЛБ и Шифър ДВОЕН РЕД – СТЪЛБ;
- ***Роторни машини***. Този подход е обединение на заместителни и разместителни шифри. Представено е описанието, структурата и използването им.

Основите на съвременната криптография са представени в ***Глава 4***. Дефиниран е основният принцип на съвременната криптография ***„Моделът за сигурност не се базира на неизвестността на алгоритъма. Допуска се, че алгоритъмът е известен на противника. Сигурността зависи от сигурността на ключа“***. Представени са различни гледни точки за разделяне на криптографските алгоритми – безусловна и условна криптография, известни и тайни алгоритми, блокови и поточни шифри. Съвременните блокови шифри са представени от схемата на ***Фейстел и Data Encryption Standard (DES)***. Разгледани са развитието на DES и начините за криптиране на големи съобщения с DES. Описан е най-използваният блоков криптографски алгоритъм – ***Advanced Encryption Standard (AES)***. ***Криптографските алгоритми с публичен ключ*** са разгледани в историческа хронология, развитие и използване. Обхванати са – задачата за раницата, криптографското решение на Merkle–Hellman, криптосхемите RSA, Рабин, Ел–Гамал (*ElGamal*), Дифи–Хелман(*Diffie–Hellman*), алгоритъмът за цифров подпис (*Digital Signature Algorithm, DSA*)

и схемата на Feige–Fiat–Shamir. ***Множество от криптографски системи и алгоритми върху елиптични криви*** са представени в края на главата.

В Приложение 1. е представена шифровата машина ***ЕНИГМА***. Описана е архитектурата на машината и механизмите за настройка на параметрите и функционирането и. Математическите основи и проблеми са част от представянето.

В Приложение 2. е представен ***Secure Hash Algorithm (SHA),*** който е имплементиран в стандарта ***Secure Hash Standard (SHS)*** като стандарт за цифров подпис (*Digital Signature Standard*). Включени са историческото развитие, сравнение на различните версии на алгоритъма и алгоритъм за функционирането му.

Забележка. В съдържанието са имплементирани фигури и изображения, свободно достъпни в Интернет, които са публикувани без изменения и корекции.

Авторът благодари на ръководството на УниБИТ за създадените условия и възможности за научни изследвания и публикации.

Специална благодарност на ***Ректора на УниБИТ проф. д.н. Ирена Петева***, на ***Председателя на Общото събрание проф. д.ик.н. Стоян Денчев*** и на ***Академично издателство „За буквите – О писменехъ“*** за отделеното време и ценните препоръки.

Книги, разширяващи знанието в предметната област, могат да бъдат изданията от научната серия ***„Защита на информацията“***:

- Целков, В. Управление на риска за сигурността на комуникационно-информационните системи. София: За буквите – О писменехь, 2020, 128 с., ISBN 978-619-185-445-5-pdf.
- Целков, В., Д. Петков, Г. Средков и Пл. Георгиев. Защита на данните. Принципи и практики. Второ преработено издание. София: За буквите – О писменехь, 2020, 283 с., ISBN 978-619-185-441-7.
- Целков, В., С. Денчев и Ир. Петева. Сигурност на информационните ресурси. София: За буквите – О писменехь, 2020, 268 с., ISBN 978-619-185-432-5
- Целков, В., Д. Петков, Г. Средков и Пл. Георгиев. Защита на данните. Принципи и практики. София: За буквите – О писменехь, 2019, 308 с., ISBN 978-619-185-364-9.
- Целков, В. и О. Исмаилов. Сигурност на информацията. София: За буквите – О писменехь, 2016, 216 с., ISBN 978-619- 185-253-6.
- Целков, В., Н. Стоянов и О. Исмаилов. Управление на риска, тестване и оценка на мрежовата и информационната сигурност. София: За буквите – О писменехь, 2016, 334 с., ISBN 978-619-185-159-1.
- Целков, В., Н. Стоянов и О. Исмаилов. Международни стандарти и добри практики за защита на информацията. София: За буквите – О писменехь, 2010, 231 с., ISBN 978-954-8887-68-7.
- Целков, В. и Н. Стоянов. Защитени криптографски приложения в компютърните системи и мрежи. София: Нова звезда, 2009, 170 с., ISBN 978-954-8933-20-9.

- Целков, В. Модели на защитени взаимодействия в компютърни системи и мрежи. София: За буквите – О писменехь, 2008, 322 с., ISBN 987-954-8887-45-8.

INTRODUCTION

The series of serious scientific publications in Bulgaria on the topic of "Information Security" is successfully continued by the book "Introduction to Cryptography and Data Security". The exposition mainly presents the syllabus of the eponymous course of the master's program "Information Protection in Computer Systems and Networks" of the Faculty of Mathematics and Informatics at Sofia University "St. Kliment Ohridski". Some of the results obtained from the research, studies and development carried out at the Military Research Institute of the General Headquarters and the Institute for Advanced Studies for Defense of the Ministry of Defense in the period 1990 – 2003 have been presented. The author's experience as a representative of the Republic of Bulgaria in the subcommittee (SC / 4) for information security (assurance) of NATO, participant in the NATO working group, which developed the architecture for information security, and member of the technical management committee of the project "Compatibility of secure communications" is reflected. The book synthesizes and presents the author's experience gained as a team leader and author of the books in the series "Information Protection" of the Academic Publisher "Za Bukvite – O pismeneh".

The book is structured in four chapters, appendices, literature, introduction, conclusion and presentation of the author.

The first chapter presents the basics of data security. The purpose and principles in building the security system of communication and information systems have been formulated. The main directions for ensuring data security have been defined – confidentiality, integrity, availability, accountability and security. The security threat model defines the notions of attack and incident and shows the four categories of attacks in the model – phone tapping, interrupting, modifying, and fabricating. The attacks are classified according to different criteria and analysis has been made of the attacks in the dimensions - attackers, means, access, results and goals. Security services are presented depending on

their purpose – maintenance, prevention and recovery. The connection between the directions and the security services has been shown.

The second chapter is an introduction to the science of cryptography. The basic concepts – sender and receiver, message and encryption, algorithms and ciphers, cryptographic keys, symmetric and asymmetric cryptographic algorithms have been presented. The use of asymmetric cryptographic algorithms and the content and structure of the concepts digital signature and public key infrastructure are shown. A cryptographic system for information protection has been defined. Cryptographic key management models are based on criteria - groups, levels and time. The time models are divided into static, dynamic and hybrid. An original, unified solution for information protection in corporate information systems has been described, an example of integration and distribution of cryptographic keys and architecture of a secure system for messaging for special needs has been presented.

The third chapter is devoted to classical cryptography. Following the historical chronology and classification of cryptographic methods in the way of transformation of the open message into encrypted one, the number of symbols and the number of used alphabets are considered in turn:

• Substitution-based methods. One letter encryption. The algorithms are presented – Atbash (Hebrew), ROT1 (k), Polybius Square and Caesar's Code;

• Substitution-based methods. Very alphanumeric encryption. The algorithms are presented – Playfair Code, Hill Code, Double Playfair Code and Four Matrix Code (squares);

• Substitution-based methods. Poly-Alphabetic Ciphers. The algorithms are presented - Vigenere cipher, Jefferson disk and Alberti disk;

• Displacement-based methods. The algorithms are - Early Spartan cipher (scytale), Code RAIL FENCE (rail fence), Method of the route, Code ROW - PILLAR and Code DUAL ROW - PILLAR;

• Rotary machines. This approach is a combination of substitution and placement codes. Their description, structure and use have been presented.

The basics of modern cryptography have been presented in **Chapter 4**. The basic principle of modern cryptography has been defined: "The security model is not based on the uncertainty of the algorithm. It is assumed that the algorithm is known to the opponent. Security depends on the security of the key. Different points of view for division of cryptographic algorithms are presented - unconditional and conditional cryptography, known and secret algorithms, block and stream ciphers. Modern block ciphers are represented by the Feistel scheme and the Data Encryption Standard (DES). The development of DES and ways to encrypt large messages with DES are discussed. The most used block cryptographic algorithm – Advanced Encryption Standard (AES) has been described. Public key cryptographic algorithms are discussed in historical chronology, development and use. Covered are the backpack task, the Merkle – Hellman cryptographic solution, the RSA, Rabin, ElGamal, Diffie – Hellman cryptocurrencies, the Diffie – Hellman algorithm, the Digital Signature Algorithm (DSA), and the Feige scheme. – Fiat – Shamir. Many cryptographic systems and algorithms on elliptic curves have been presented at the end of the chapter.

Appendix 1 presents the ENIGMA cipher machine. The architecture of the machine and the mechanisms for setting its parameters and functioning have been described. The mathematical bases and problems are part of the presentation.

Appendix 2 presents the Secure Hash Algorithm (SHA), which is implemented in the Secure Hash Standard (SHS) as a Digital Signature Standard. Historical development, comparison of different versions of the algorithm and algorithm for its operation are included.

Note. The content uses figures and images freely available on the Internet, which are published without changes and corrections.

The author thanks the management of ULSIT for the created conditions and opportunities for research and publications.

Special thanks to the Rector of ULSIT Prof. DSc Irena Peteva, and the Chairman of the General Assembly Prof. DSc Stoyan Denchev, and the team of Academic Publisher "Za bukvite – O pismeneh" for the time and valuable recommendations.

ГЛАВА 1. СИГУРНОСТ НА ДАННИТЕ

В главата са представени основите на сигурността на данните. Формулирани са целта и принципите при изграждане на системата за сигурност на комуникационно-информационните системи. Дефинирани са основните направления за осигуряване на сигурността на данните – поверителност, цялостност, наличност, отчетност и гарантираност. В модела на заплахите за сигурността са дадени определенията за атака и инциденти и са показани четирите категории атаки в модела – подслушване, прекъсване, модифициране и изфабрикуване. Извършена е класификация на атаките по различни критерии и е направен анализ на атаките в измеренията – атакуващи, средства, достъп, резултати и цели. Услугите за сигурност са представени в зависимост от тяхното предназначение – поддръжка, предотвратяване (превенция) и възстановяване. Показана е връзката между направленията и услугите за сигурност.

1.1. ЦЕЛ И ПРИНЦИПИ ЗА ИЗГРАЖДАНЕ НА СИСТЕМАТА ЗА СИГУРНОСТ НА КОМУНИКАЦИОННО-ИНФОРМАЦИОННИТЕ СИСТЕМИ

Сигурността на данните в комуникационно-информационните системи се осигурява от системата за сигурност.

Цел

Целта на системата за сигурност на комуникационно-информационните системи е да осигури на организацията възможността да реализира своите мисия и бизнес задачи посредством внедряването на комуникационно-информационни технологии при отделяне на особено

внимание на свързаните с тях рискове за организацията, партньорите и потребителите.

Основни принципи за изграждане на системата за сигурност

Основните принципи за изграждане на системата за информационна сигурност, които трябва да се имат предвид при управлението на риска, тестването за сигурност и анализ на резултатите и оценката са както следва:

- Простота;
- Защита при провал;
- Продължаващо посредничество;
- Отворен проект;
- Разделяне на привилегиите;
- Психологическо приемане;
- Слоеста отбрана;
- Записване при компрометиране.

Простота

Механизмите за сигурност (и информационните системи като цяло) трябва да бъдат възможно най-прости. Сложността е в основата на много от проблемите на сигурността.

Защита при провал

Ако се появи проблем, системата трябва да преустанови функционирането си по сигурен начин. Това означава, че ако се появи проблем с функционирането на системата, механизмите за сигурност трябва да продължат да работят. По-добре е да се губят функционалности, отколкото да се губи сигурност.

Продължаващо посредничество

Вместо да се разрешава директен достъп до информация, трябва да се използват посредници, които да осигурят изпълнение на политиката за контрол на достъпа. Типичните примери включват разрешения към системните файлове, прокси сървъри и пощенски шлюзове.

Отворен проект

Сигурността на системата не трябва да зависи от тайната на приложението и изпълнението му или от тайната на компонентите й. „***Сигурност чрез неизвестност***“ не е приложима.

Разделение на привилегиите

Функциите в привилегирован режим трябва да бъдат разделени, където това е възможно. Концепцията за разделение на привилегиите може да се прилага, както за системите, така и за потребителите. В случаите на системни администратори и потребители, ролите трябва да бъдат разделени, ако е възможно. Например, ако ресурсите позволяват, ролята на системен администратор, трябва да бъде отделена от тази, на администратора по сигурността.

Психологическо приемане

Потребителите трябва да се убедят в необходимостта от сигурност. Това може да се осигури чрез обучение и образование. В допълнение, механизмите за сигурност, трябва да представят на потребителите разумни опции, които ще им дадат възможността, те да ги използват ежедневно. Ако потребителите намират механизмите за сигурност твърде тромави, ще търсят начин да ги преодолеят или да ги компрометират. Пример за това е използването на случайни пароли, които са много силни, но е трудно да се

запомнят. В този случай потребителите могат да ги запишат или да се опитват да ги преодолеят.

Слоеста отбрана

Организациите трябва да разберат, че всеки един механизъм за сигурност сам по себе си е недостатъчен. Механизмите за сигурност трябва да се изграждат на различни нива, така че компрометирането на един механизъм за сигурност да е недостатъчен, за да се компрометира системата.

Записване при компрометиране

Когато системите и/или мрежите са компрометирани, трябва да бъдат създадени регистри или дневници със записи за събитието на компрометиране. Тази информация може да съдейства за повишаване на сигурността на мрежата след компрометирането и да помогне при определяне на методите и уязвимостите, използвани от хакера (нарушителя). Регистрите могат да се използват, за да се осигури подобряването на сигурността на системата/мрежата в бъдеще. В допълнение, това може да помогне на организациите при идентифицирането и преследването на нарушителите.

1.2. ОСНОВНИ НАПРАВЛЕНИЯ ЗА ОСИГУРЯВАНЕ НА СИГУРНОСТТА НА ДАННИТЕ

Основните направления за осигуряване на сигурността на информацията (данните) са:

- Поверителност/тайна (*Confidentiality*);
- Цялостност (*Integrity*);
- Наличност/достъпност (*Availability*);
- Отчетност (*Accountability*);
- Гарантираност (*Assurance*).

Поверителност

Изискването за поверителност се отнася за данните в системата и системната информация. Тайната е изискване за не разкриване на данни (класифицирана информация, лични данни, чувствителни системни и корпоративни данни и др.) на неоторизирани потребители. Тайната на данните се прилага в процеса на тяхното съхранение, обработване и предаване.

Цялостност

Цялостността има две части:

- ***Цялостност на данните*** – свойството на данните да не могат да бъдат променени/подменени по неоторизиран начин в процеса на тяхното съхранение, обработване и предаване;
- ***Цялостност на системата*** – свойството на системата да изпълнява функциите си така, както са проектирани и без възможност за неоторизирани манипулации на функциите.

Наличност

Наличността е изискване за осигуряване на коректна работа на системата и за гарантиране, че нейните услуги няма да бъдат блокирани за оторизираните потребители. Наличността осигурява защитата против:

- Умишлени или случайни опити за:
  - Неоторизирано изтриване на данни;
  - Друг начин за реализация на отказ от обслужване или данни (*Denial of Service or Data*).
- Опити за използване на системата или данните за неоторизирани (нерегламентирани) действия.

Отчетност

Отчетността е изискване за това, че действията на всяка единица могат да бъдат проследени и идентифицирани. Отчетността обикновено е изискване на политиката за сигурност и непосредствено поддържа:

- Неотхвърляемост/невъзможност за отхвърляне/отричане (*Non-Repudiation*);
- Възпиране (*Deterrence*);
- Изолиране на грешките (*Fault Isolation*);
- Откриване и предотвратяване на проникване (*Intrusion Detection and Prevention*);
- Възстановяване след откази (*After-Action Recovery*);
- Законни действия (*Legal Action).*

Гарантираност

Изискването за гарантираност осигурява изпълнението, адекватно и коректно, на изискванията във всички четири останали направления за сигурност. Гарантираността е осигуряване, че мерките за сигурност

(технически и оперативни) работят както са предвидени за защита на системата и обработваната информация. Отнася се за всички направления и осигурява, че:

- Изискваните функционалности съществуват и са коректно реализирани;
- Съществува достатъчна защита против неумишлени грешки (човешки, технически и програмни);
- Съществува достатъчно противодействие против умишлено проникване или заобикаляне.

Зависимости между направленията

Петте направления за сигурност са взаимно зависими. Реализацията на изискванията на едно направление без зависимост от останалите е трудно постижимо. Зависимостите са както следва:

- Тайната е зависима от цялостността;
- Цялостността е зависима от тайната;
- Наличността и отчетността са зависими от тайната и цялостността;
- Всички направления са взаимно зависими от гарантираността.

1.3. МОДЕЛ НА ЗАПЛАХИТЕ ЗА СИГУРНОСТТА

В модела на заплахите за сигурността основно място заемат определенията за атака и инциденти и четирите категории атаки – прекъсване, подслушване, модифициране и изфабрикуване. Класификацията на атаките е извършена по различни критерии и е направен анализ на атаките в измеренията – атакуващи, средства, достъп, резултати и цели.

Атаки и инцидент

Едни от най-често използваните понятия свързани със сигурността са понятията:

- Атака;
- Инцидент.

Атака

Атака е един единичен опит за неоторизиран достъп или опит за неоторизирано използване, независимо от това успешно или не.

Инцидент

Един инцидент включва една група от атаки, която може да бъде отделена от други инциденти поради характера и степента на атаките (напр. компютри, техники и време).

Модел на заплахите за сигурността

Моделът дефинира четири категории на атаки, както следва (***Фигура 1.1***.):

- ***Прекъсване*** – едно от ценните (жизнените) качества на системата е разрушено или е невъзможно използването му;

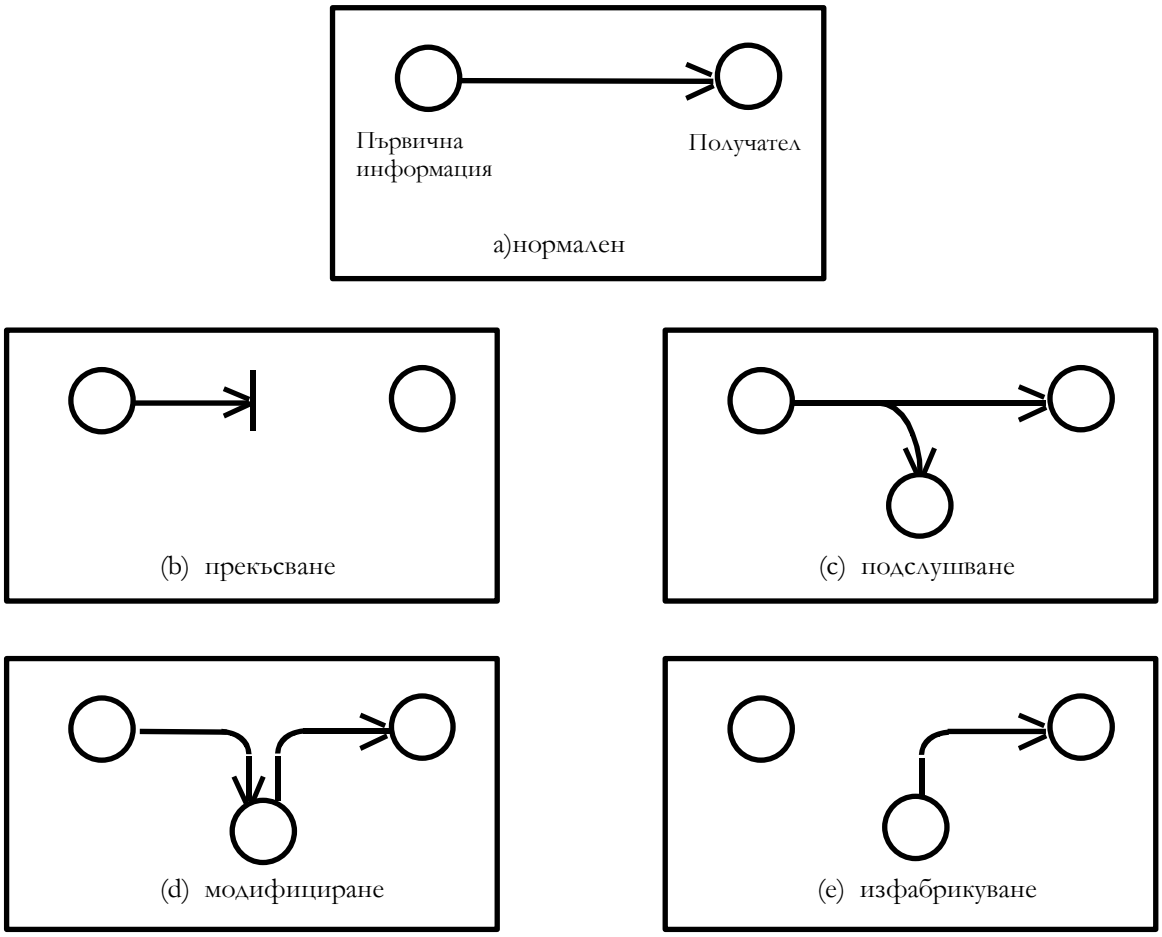
- ***Подслушване*** – едно неоторизирано действие дава достъп до системата;
- ***Модифициране***;
- ***Изфабрикуване***.

Подслушването е една пасивна атака, докато прекъсването, модифицирането и изфабрикуването са активни атаки.

Типове атаки в компютърните системи и мрежи

Типовете атаки могат да се обобщят в седем категории:

- ***Кражба на пароли*** – методи за получаване на други потребителски пароли;
- ***Грешки и черни врати (bugs and backdoors)*** – получаване (използване) на преимущества посредством използване на системни грешки;
- ***Непълноти в механизмите за автентификация*** – използване на дефектите (противоречивост и непълнота) на механизмите за автентификация;
- ***Грешки (провали) в някои протоколи*** – протоколи с грешки в проектирането и реализацията;
- ***Социален инженеринг*** – придобиване на информация до която нямате достъп;
- ***Изтичане на информация*** – използване на системи като системата за подписване (*finger*) или системата за именуване (DNS), за информация необходима на администратора или необходима за функционирането на мрежата, но която може да се използва за мрежови атаки;
- ***Отказ от обслужване*** – опити за лишаване на потребителите от услугите на тяхната компютърна система.



Фигура 1.1. Модел на заплахите за сигурността

Категоризация на атаките в зависимост от резултатите

Един друг вариант на категоризация на атаките е в зависимост от постигнатите от атаката резултати. Пример на такава класификация е както следва:

- Разрушение;
- Изтичане;
- Отказ (блокировка).

Емпиричен списък

Една емпирична категоризация обхваща осем категории. Основното и преимущество е, че съществуват атаки, които логически не могат да попаднат в точно една от трите изброени по-горе категории и които удовлетворяват тази категоризация. Тази категоризация е както следва:

- Външна кражба на информация;
- Външна злоупотреба с (на) ресурсите;
- Представяне (преструване) (записване и използване на мрежовия трафик);
- Вредителски програми (инсталиране на злонамерени програми);
- Повторение на автентификацията или авторизацията (разбиване на пароли);
- Злоупотреба с авторизацията (фалшифициране на записи);
- Злоупотреба с бездействие (лошо администриране);
- Индиректна злоупотреба (използвайки други системи за създаване на зловредни програми).

Уязвимости на компютърните системи и мрежи

Най-често атакуващите използват предимствата от компютърните и мрежови уязвимости. Уязвимостите могат да се използват както следва:

- ***Чрез софтуерните (хардуерни) грешки (bugs).*** Типичен пример са Unix системите, които са в основата на Internet и които имат много проблеми в send mail програмата, която често се използва за получаване на неоторизиран достъп до Host компютъра;
- ***Използвайки грешките, възникнали при проектирането на системите***;
- ***Грешките възникнали при конфигурирането на системите***. Грешките при конфигурирането включват такива проблеми за

сигурността, като системни правомощия с добре известни пароли, разрешение по подразбиране за използване на нови файлове и т.н.

Анализ на атаките

За пълния анализ на атаките е целесъобразно всяка една атака да се разглежда (класифицира) в следната последователност:

- Атакуващи;
- Средства;
- Достъп;
- Резултати;
- Цели.

Графичното представяне е както следва (***Фигура 1.2.***)*:*

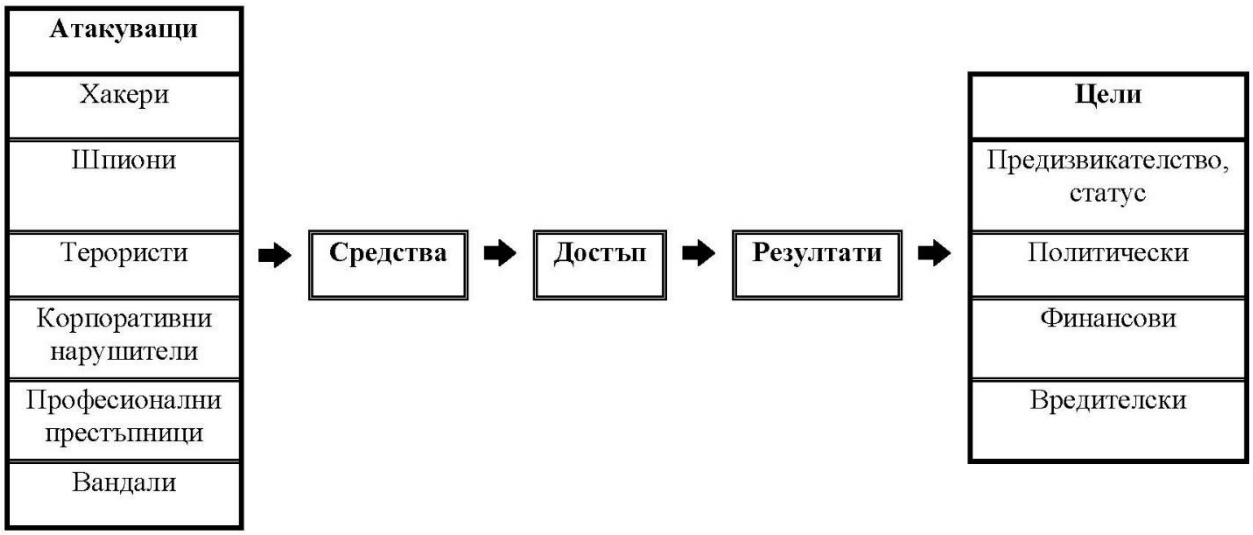
Фигура 1.2. Последователност за анализ на компютърни и мрежови атаки

Атакуващи

Атакуващите могат да се класифицират както следва:

- Хакери – за предизвикателствата и статус на получили достъп;
- Шпиони;
- Терористи;
- Корпоративни нарушители;
- Професионални престъпници;
- Вандали.

Графичното представяне на атакуващите и тяхната първична мотивация е както следва (***Фигура 1.3.***):



Фигура 1.3. Атакуващи и тяхната първична мотивация

Цели

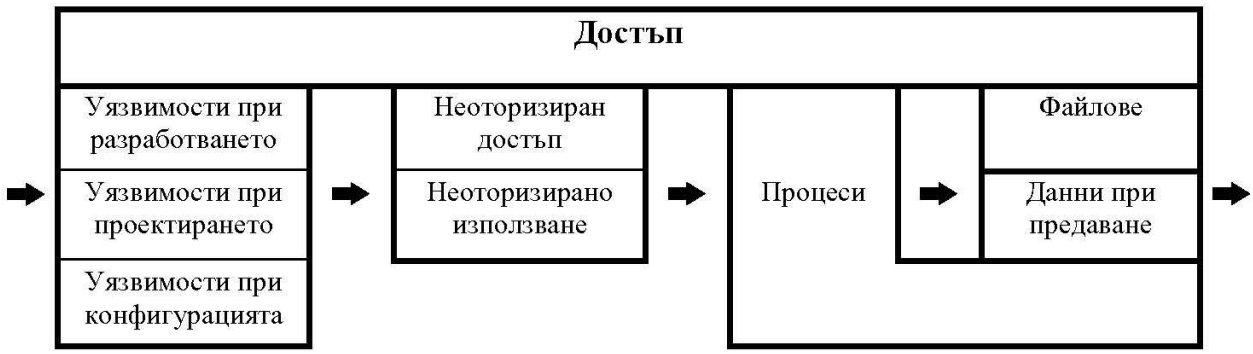
Целите могат да се класифицират както следва (***Фигура 1.3.***):

- Предизвикателство, статус;
- Политически;
- Финансови;
- Вредителски.

Достъп

Основното свързващо звено между атакуващите и техните цели е неоторизирания достъп или неоторизираното използване (***Фигура 1.4.***). Неоторизираният достъп или използване е към процесите или към файловете и данните, предавани по мрежата през процесите. Неоторизираният достъп и използване са един от начините (пътища) за атака. Не трябва да се пренебрегва и фактът, че са възможни и атаки при

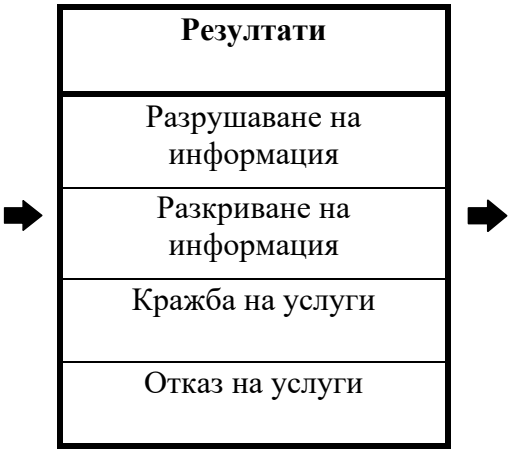
злоупотреба с правата за достъп. Не по-малко от 80% от проникванията в системите са от напълно оторизирани потребители, които са злоупотребили с правата си за достъп. Това е потенциално и един от най-големите проблеми при защита на мрежата.



Фигура 1.4. Достъп при атака

Резултати

Между придобиването на достъп и целите на атакуващите са резултатите от атаката (***Фигура 1.5.***). В тази точка от последователността на една атака, атакуващият получава достъп до желаните процеси, файлове и данни. В този момент той е свободен да използва този достъп за чувствителните файлове, да забрани услуги, да получи информация или да използва услуги.



Фигура 1.5. Резултати от атака

Средства

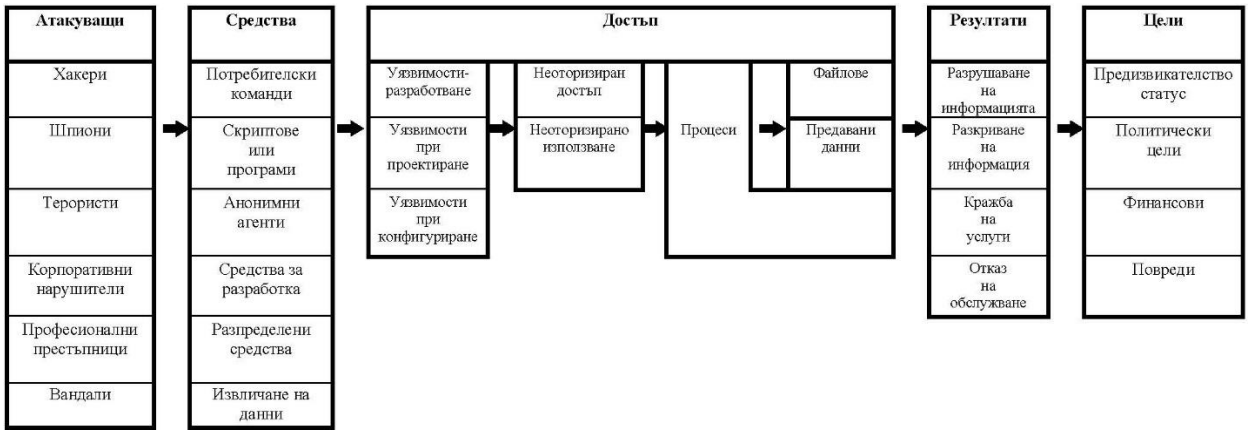
Средствата за атака могат да се разделят в следните категории:

Фигура. 1.6. Средства за атака

- ***Потребителски команди*** – въвеждани от командната линия или посредством потребителски графичен интерфейс;
- ***Скриптове или програми*** – стартирани от атакуващия и използващи уязвимостите на системите;
- ***Анонимни агенти*** – инсталират се програми или фрагменти от тях, които работят в последствие независимо от потребителя и използват уязвимостите;
- ***Средства за разработване*** – софтуерни пакети съдържащи скриптове, програми или анонимни агенти;
- ***Разпределени средства*** – средствата за атака се разпределят върху различни компютри;

- ***Извличане на данни*** – когато се подслушва електромагнитното излъчване от компютърните системи и мрежи чрез устройства, външни за мрежите.

Графичното представяне на пълния анализ на атаките (***Фигура 1.7.***) е както следва:



Фигура 1.7. Пълна схема на компютърните и мрежови атаки

1.4. МОДЕЛ НА УСЛУГИТЕ ЗА СИГУРНОСТ

Услугите за сигурност се класифицират, в зависимост от тяхното предназначение, в областите както следва:

- Поддръжка;
- Предотвратяване/превенция;
- Възстановяване.

Поддръжка

Тези услуги реализират най-общите и основни възможности за сигурността на информацията. Включват:

- ***Идентификация и именуване*** (*Identification and Naming*). Основният въпрос е коректното идентифициране на обектите и субектите на системата. Услугите осигуряват уникална идентификация на потребителите, процесите и ресурсите;
- ***Управление на криптографските ключове*** (*Cryptographic Key Management*). Криптографските ключове трябва на бъдат управлявани сигурно, когато се използват криптографски механизми в други услуги;
- ***Администриране на сигурността*** (*Security Administration*). Възможностите за сигурност трябва да бъдат администрирани, за да удовлетворят изискванията при специфични инсталации и при промяна на средата;
- ***Защита на системата*** (*System Protections*). Основните функционални възможности за сигурност трябва да бъдат приложени в системата. Примери за защита на системата са:
  - Повторно използване на обекти (*Object Reuse*);
  - Минимални привилегии (*Least Privilege*);
  - Разделяне на процесите (*Process Separation*);
  - Модулност (*Modularity*);

- Разделяна на нива (*Layering*);
- Минимизиране на необходимостите от защита (*Minimization of What Needs to be Trusted*).

Предотвратяване/превенция

Тези услуги се фокусират върху предотвратяване на настъпването на нарушения на сигурността. Включват се услугите:

- ***Защита на комуникациите*** (*Protected Communications*);
- ***Автентификация*** (*Authentication);*
- ***Оторизация/авторизация*** (*Authorization*);
- ***Неотхвърляемост*** (*Non-repudiation*);
- ***Неприкосновеност*** (*Transaction Privacy*).

Възстановяване

Тези услуги са свързани с откриване и възстановяване на нарушенията в системата за сигурност. Невъзможно е съществуването на перфектна система за превенция и това налага предприемането на мерки за възстановяването на системата. Включват услугите:

- ***Наблюдение*** (*Audit*). Наблюдението на събитията, свързани със сигурността на системата е ключов елемент в случаите, когато е установено нарушение на сигурността и е необходимо възстановяването на системата;
- ***Откриване на проникванията и ограничаване на влиянието*** (*Intrusion detection and containment*);
- ***Доказателство за пълнота*** (*Proof of Wholeness*);
- ***Връщане/възстановяване на „сигурно“ състояние*** (*Restore „Secure“ State*).

1.5. НАПРАВЛЕНИЯ И УСЛУГИ ЗА СИГУРНОСТ

Реализацията на направленията за сигурност чрез услугите за сигурност е както следва.

Поверителност

Изискванията за поверителност са свързани със следните услуги за сигурност:

- Оторизация;
- Контрол на достъпа;
- Защита на комуникациите;
- Неприкосновеност;
- Идентификация и именуване;
- Управление на криптографските ключове.

Цялостност

Изискванията за цялостност са свързани със същите услуги за сигурност, както и наличността:

- Оторизация;
- Контрол на достъпа;
- Защита на комуникациите;
- Откриване на прониквания и ограничаване на влиянието;
- Доказателство за пълнота;
- Връщане в „сигурно“ състояние;
- Идентификация и именуване;
- Управление на криптографските ключове.

Наличност

Изискванията за наличност са свързани със следните услуги за сигурност:

- Оторизация;
- Контрол на достъпа;
- Защита на комуникациите;
- Откриване на проникванията и ограничаване на влиянието;
- Доказателство за пълнота;
- Връщане в „сигурно“ състояние;
- Идентификация и именуване;
- Управление на криптографските ключове.

Отчетност

Изискванията за отчетност са свързани със следните услуги за сигурност:

- Контрол на достъпа;
- Неотхвърляемост;
- Наблюдение;
- Идентификация и именуване;
- Управление на криптографските ключове.

Гарантираност

Изискванията за гарантираност са свързани със следните услуги за сигурност:

- Автентификация;
- Контрол на достъпа;
- Защита на комуникациите;
- Наблюдение;

- Откриване на проникванията и ограничаване на влиянието;
- Доказателство за пълнота;
- Връщане в „сигурно“ състояние;
- Администриране на сигурността;
- Защита на системата.

ГЛАВА 2. КРИПТОГРАФСКИ АЛГОРИТМИ

Глава втора е въведение в науката криптография. Представени са основните понятия – подател и получател, съобщение и криптиране, алгоритми и шифри, криптографски ключове, симетрични и асиметрични криптографски алгоритми. Показани са използването на асиметричните криптографски алгоритми и съдържанието и структурата на понятията цифров подпис и инфраструктура на публичните ключове. Дефинирана е криптографска система за защита на информацията. Моделите за управление на криптографските ключове са базирани по критериите – групи, нива и време. Моделите по време са разделени на статичен, динамичен и хибриден модел. Описано е едно оригинално, единно решение за защита на информацията в корпоративни информационни системи, представен е пример за интеграция и разпределение на криптографски ключове и архитектура на защитена система за обмен на съобщения за специални нужди.

2.1. КРИПТОГРАФСКИ АЛГОРИТМИ. ОСНОВНИ ПОНЯТИЯ

В многообразието на понятието информационна сигурност важно място заема защитата на информацията в компютърните системи. В основата на тази защита са криптографските алгоритми и механизми. Основно приложение криптографските алгоритми и механизми намират при:

- Процесите по идентификация и автентификация;
- Контрол на достъпа до ресурсите;
- Осигуряване на поверителност и цялостност на данните.

Основни понятия

Получател и подател

Предполагаме, че някой, когото ще наричаме подател, иска да изпрати съобщение на някой друг, когото ще наричаме получател.

Подателят иска да бъде сигурен, че при предаването на съобщението няма да му бъде въздействано по никакъв начин:

- Единствено получателят може да прочете съобщението ***(поверителност, идентификация);***
- Съобщението няма да бъде модифицирано или да бъде заменено с друго съобщение ***(цялостност)***.

Получателят иска да бъде сигурен, че:

- Съобщението е от подателя ***(идентификация);***
- На съобщението няма да му бъде въздействано по никакъв начин ***(цялостност)***.

Съобщения и криптиране

Съобщението се нарича явен текст. Процесът на прикриване на съдържанието на съобщението по какъвто и да е начин се нарича криптиране. Криптираното съобщение се нарича шифриран текст. Процесът на връщане обратно на шифрирания текст в явен текст се нарича декриптиране.

Алгоритми и шифри

Криптографският алгоритъм (шифър) е математическа функция (един от параметрите на която се нарича ключ), която се използва за криптиране и декриптиране. За да се криптира явен текст, към него се прилага

криптиращ алгоритъм. За да се декриптира шифриран текст, към него се прилага декриптиращ алгоритъм.

Означения

Ще използваме следните означения:

- M – открито съобщение;
- C – криптирано (шифрирано) съобщение;
- E – функция криптиране (шифриране);
- D – функция декриптиране(дешифриране);
- K – ключ.

Тогава:

- $E_K(M) = C$;
- $D_K(C) = M$.

Симетрични и асиметрични алгоритми

Криптографските алгоритми биват два вида:

- Симетрични
- Асиметрични.

Симетрични криптографски алгоритми

При симетричните алгоритми ключът за декриптиране може да бъде пресметнат от ключа за криптиране и обратно. В голямата част от тези алгоритми ключовете за криптиране и декриптиране са еднакви. Симетричните алгоритми (наричани също и алгоритми със секретни ключове) изискват получателя и подателя предварително да разполагат със съответните ключове и да се грижат за опазването им в тайна. Сигурността на симетричния алгоритъм се основава на ключа, защото ако ключът бъде разкрит, всеки би могъл да криптира и декриптира съобщения.

Симетричните алгоритми биват два вида: поточни и блокови. Поточните алгоритми обработват явния текст бит по бит, а блоковите – по групи битове. Групата битове се нарича блок. За удобство при компютърно приложение на алгоритмите най-често използваната големина на блока е 64 бита – достатъчно голяма, за да затрудни криптоанализа, и достатъчно малка за удобна работа.

В общия случай алгоритмите са базирани на субституции и пермутации. За осъществяването на взаимодействие (предаване/приемане на съобщения) е необходимо наличието на един и същ ключ в изпращача и приемача. При необходимост от взаимодействия с много участници е необходимо споделянето на един и същ ключ, което увеличава опасността от компрометирането му.

Фигура 2.1. Симетричен алгоритъм

Асиметрични криптографски алгоритми

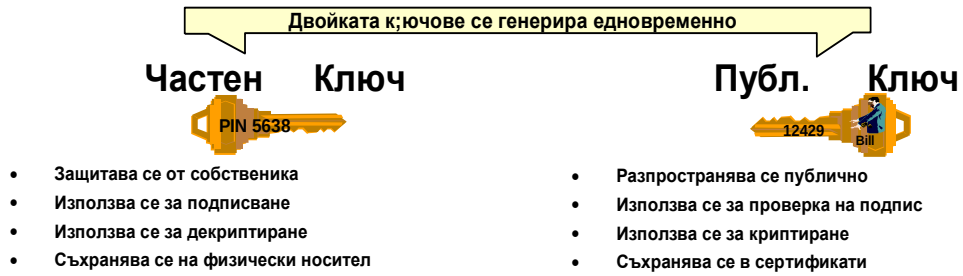
При асиметричните алгоритми (наричани още и алгоритми с публични ключове) криптирането и декриптирането се извършва с различни ключове – ***частен (личен) ключ*** и ***публичен ключ***. Освен това декриптиращият ключ не може да бъде (за обозримо време) пресметнат от криптиращия ключ.

Използване на ключовете

Използването на ключовете е както следва:

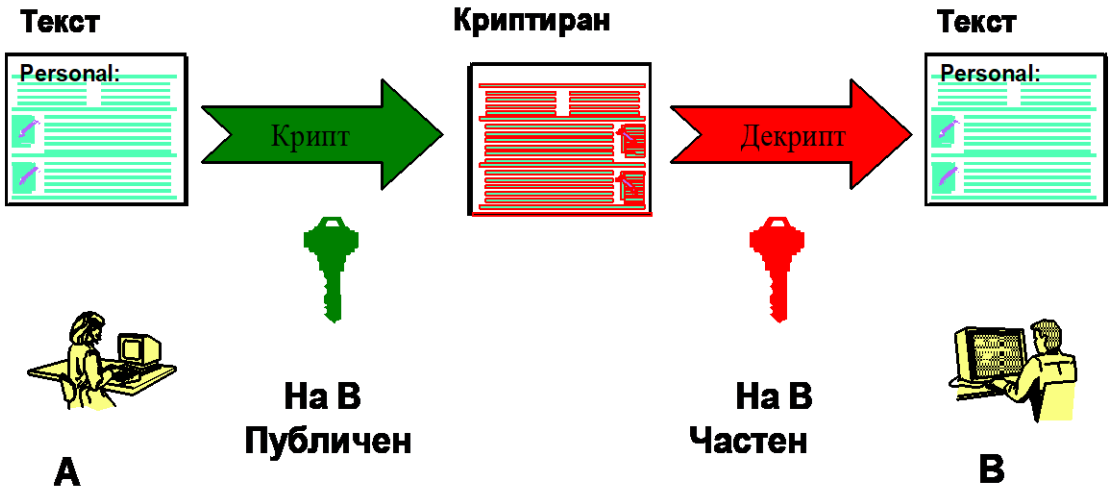
- Криптиращият ключ се нарича публичен ключ и той не се пази в тайна. Всеки може да го използва, за да криптира съобщение, но само който знае съответстващия му декриптиращ ключ може да декриптира съобщението.
- Декриптиращият ключ се нарича личен ключ. Съобщение може да се криптира и с личния ключ и да се декриптира с публичния ***(използва се за цифров подпис)***.

Информация за двата ключа на асиметричните криптографски алгоритми е показана на фигурата:



Фигура 2.2. Частен и публичен ключ на асиметричните криптографски алгоритми

За предаване на защитени съобщения страната А използва публичния ключ на страната В за криптиране, а страната В използва своя частен ключ за декриптиране на съобщението както е показано на ***Фигура 2.3***.



Фигура 2.3. Криптиране с асиметричен алгоритъм

2.2. АСИМЕТРИЧНИ КРИПТОГРАФСКИ АЛГОРИТМИ

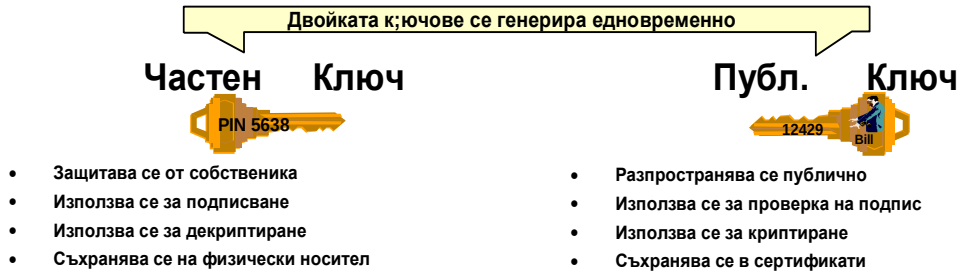
При асиметричните алгоритми (наричани още и алгоритми с публични ключове) криптирането и декриптирането се извършва с различни ключове – частен (личен) ключ и публичен ключ (означаваме с Kpr и Kpubl). Освен това декриптиращият ключ не може да бъде (за обозримо време) пресметнат от криптиращия ключ. Криптиращият ключ се нарича публичен ключ и той не се пази в тайна. Всеки може да го използва, за да криптира съобщение, но само който знае съответстващия му декриптиращ ключ може да декриптира съобщението. Декриптиращият ключ се нарича личен ключ. Понякога съобщението се криптира с личния ключ и се декриптира с публичния.

Личен и публичен ключ

Връзките между личния и публичния ключ са както следва:

- $E_{Kpubl}(M) = C$ и $D_{Kpr}(C) = M$;
- $E_{Kpr}(M) = C$ и $D_{Kpubl}(C) = M$.

Информация за връзките и използването на двата ключа е показана на фигурата:



Фигура 2.4. Връзки и използване на частен и публичен ключ

Цифрови сертификати

Едно от най-добрите и нови решения за реализирането на политиката за използване на криптографията с публични ключове и за сигурността в средата на Internet и WWW - технологиите са цифровите сертификати. Най-често с тях се решават следните основни въпроси:

- Идентификация и автентификация на потребителите в Web-сървер;
- Идентификация и автентификация на Web-сървер;
- Контрол на достъпа до ресурсите на Web-сървер;
- Защита на сесията;
- Защита на електронната поща.

Цифровите сертификати (наричани само сертификати или цифрови идентификатори) са файлове с определена структура. Върху данните от цифровите сертификати се изпълняват криптографски операции (базирани на публичните ключове), за осигуряване на по-високо ниво на сигурност при идентификацията и автентификацията, отколкото при използване на име и парола. Сертификатите за криптография с публичен ключ (X.509) формират основата за разпространение на публични ключове и установяване на самоличност (идентификация и автентификация).

Всеки цифров сертификат (X.509) съдържа информация за:

- дата на издаване;
- сериен номер;
- използван алгоритъм за електронен подпис;
- издател;
- срок на валидност (от ..., до ...);
- притежател;
- използван алгоритъм за работа с публични ключове;
- публичен ключ;

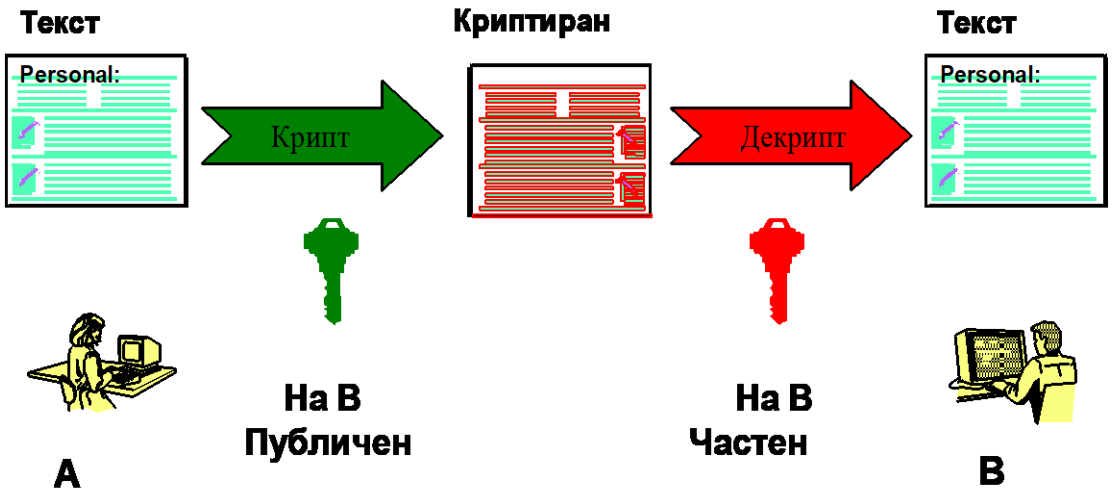
- идентификатор (тип на сертификата);
- предназначение (за SSL – сесия, за E-mail и т.н.);
- идентификатор на сертифициращ орган (*Certificate Authority*);
- алгоритъм за електронен подпис;
- електронен подпис.

Поверителност

Условие: Страната А иска да предаде съобщение на страната В, така че единствено В да може да прочете това съобщение.

Изпълнение: За удовлетворяване на условието:

- Страната А използва публичния ключ на страната В за криптиране
- Страната В използва своя частен ключ за декриптиране на криптираното съобщение както е показано на ***Фигура 2.5***



Фигура 2.5. Криптиране с асиметричен алгоритъм

Цифров подпис

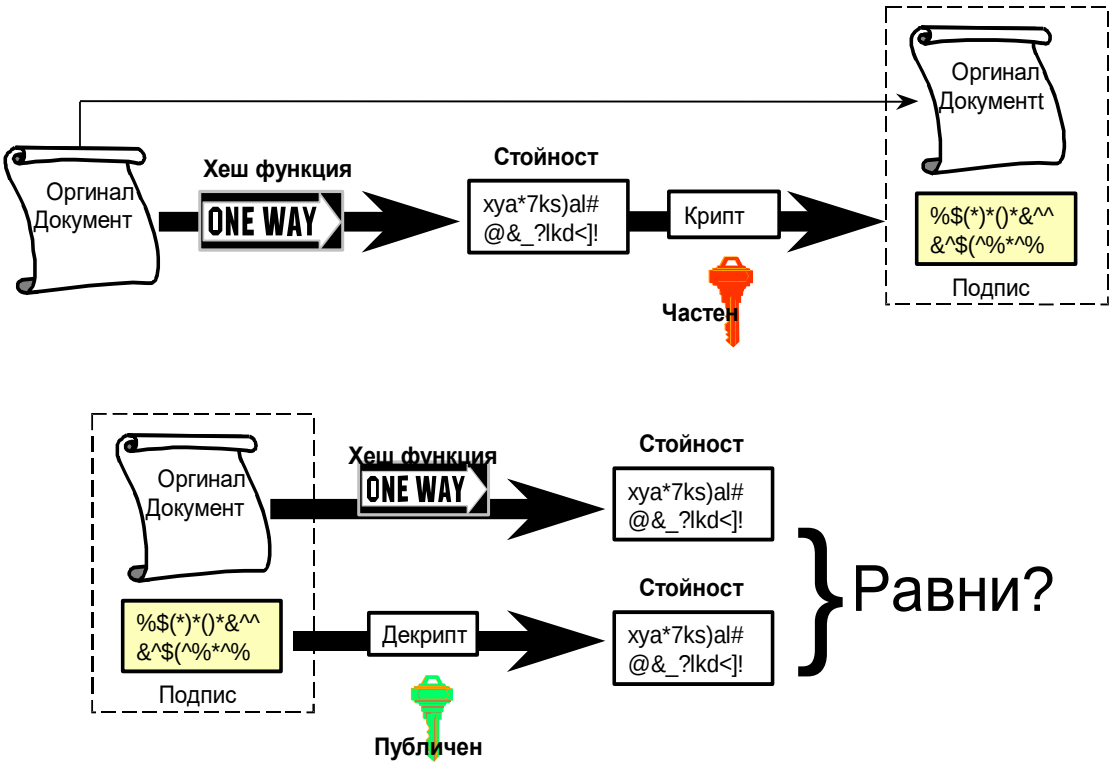
Условия: Страната А иска да предаде документ на страната В, така че да бъдат изпълнени условията:

- Да има гаранции, че документът е изпратен от А;
- Да има гаранции, че документът не е модифициран (променян).

Изпълнение: За удовлетворяване на условията:

- Страната А генерира код за удостоверяване на цялостността на документа (MAC, *message authentication code).* Най-често за тази цел се използва Хеш функция;
- Страната А използва своя частен ключ за криптиране на MAC;
- Страната А предава документа и шифрирания MAC;
- Страната В получава документа и шифрирания MAC;
- Страната В генерира код за удостоверяване на цялостността на документа – MAC1;
- Страната В използва публичния ключ на страната А за декриптиране на шифрирания MAC;
- Страната В сравнява MAC и MAC1:
  - Ако MAC = MAC1, то условието е изпълнено;
  - Ако MAC не = MAC1, то условието не е изпълнено (или А не е автор на документа, или документът е модифициран).

Механизмите на цифровия подпис и използването на криптографията с публични ключове са показани на ***Фигура 2.6***.



Фигура 2.6. Цифров подпис на документ

Обмен на защитени съобщения

Условия: Страната А иска да предаде документ на страната В, така че да бъдат изпълнени условията:

- Да има гаранции, че документът е изпратен от А;
- Да има гаранции, че документът не е модифициран (променян);
- Единствено В да може да прочете това съобщение.

Изпълнение. За удовлетворяване на условията са възможни решенията:

- Използване само на асиметрични криптографски алгоритми, като комбинация на механизмите за поверителност и цялостност. Това решение не се използва, тъй като асиметричните криптографски алгоритми са бавни, трудно се реализират хардуерно и са почти неприложими за криптиране на големи съобщения;

- Комбиниран метод – използване на симетрични и асиметрични алгоритми:
  - Симетричните се използват за криптиране на документа (с използване на сесиен ключ);
  - Асиметричните се използват за обмен на сесийния ключ и за подписване на документа.

Алгоритъмът за предаване и приемане на съобщения и за използването на различните криптографски техники е показан на ***Фигура 2.7.*** и ***Фигура 2.8.***

User A
--$K_{A's\ Public}$
--$K_{A's\ Private}$
--$K_{Message}$

--$K_{B's\ Public}$

- А подготвя съобщение за В и пресмята хеш
- А генерира ключ за съобщението, $K_{message}$, И го криптира

$$C_M = E_{K_{Message}}(M)$$

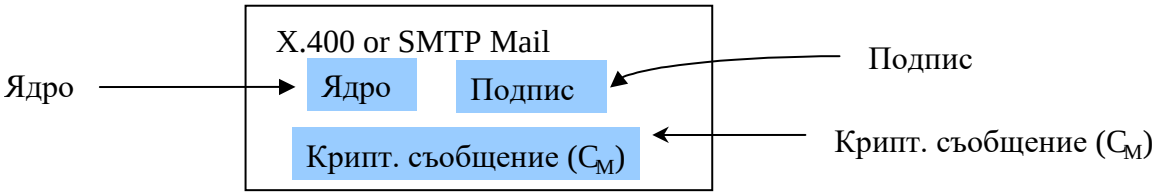
•А получава сертификата на В
•А извлича публичния ключ на В от сертификата и криптира ключа за съобщението с него

$$Ядро = E_{K_{B's\ Public}}(K_{message})$$

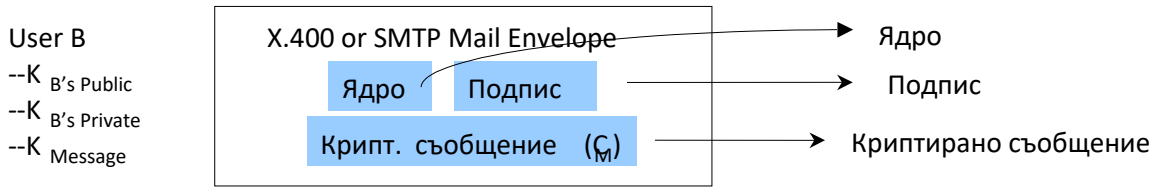
- А подписва съобщението с частния си ключ

$$Signature = E_{K_{A's\ Private}}(message\ hash, ...)$$

- А създава съобщение включващо ядро, криптирано съобщение, подпис и го изпраща на В



Фигура 2.7. Криптография с публични ключове – изпращане на съобщение



•В използва частния си ключ да декриптира ядрото и получава ключа за криптиране на съобщението, $K_{Message}$

$$K_{Message} = D_{K_{B's\ Private}}(\text{Token})$$

•В използва $K_{Message}$ да декриптира съобщението

$$\text{Message} = D_{K_{Message}}(C_M)$$

•В изисква и получава сертификата на А от Directory, извлича публичния ключ на А и го използва да декриптира подписа

$$(\text{Message Hash}, \ldots.) = D_{K_{A's\ Public}}(\text{Signature})$$

•В пресмята hash на декриптираното съобщение

•В сравнява hashes—ако те са еднакви разбира, че съобщението е коректно и че идва от А

Фигура 2.8. Криптография с публични ключове – приемане на съобщение

Защита на сесията

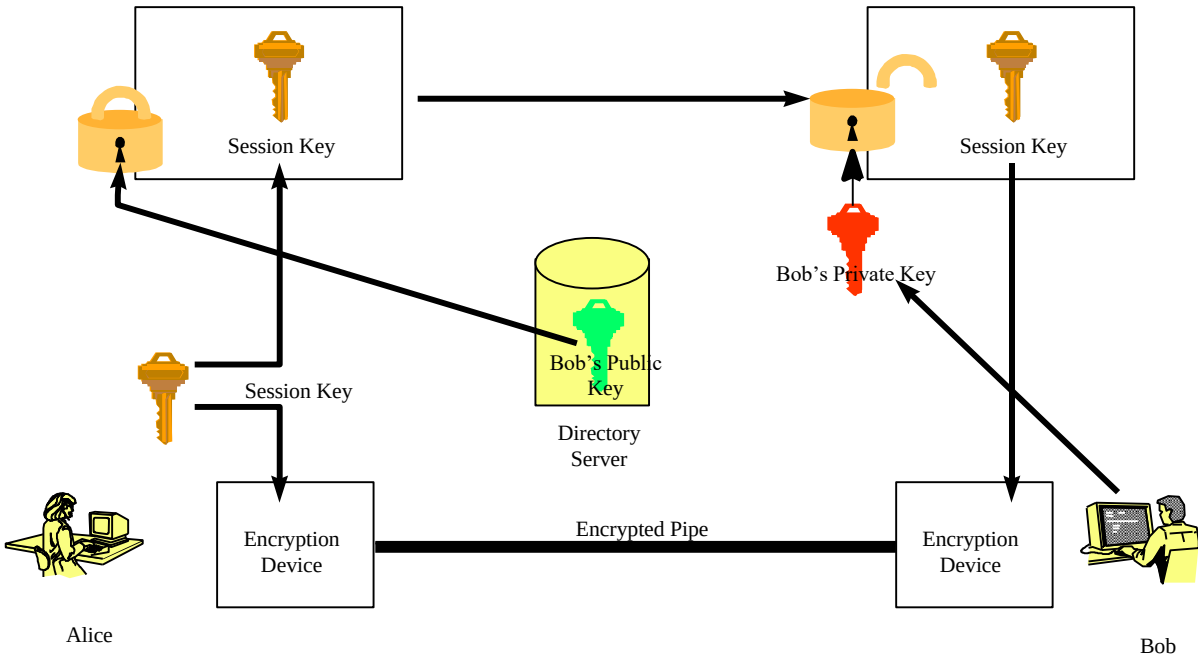
Условия: Страната А иска да изгради защитена комуникационна сесия със страната В.

Изпълнение Криптографията с публичен ключ може да се използва за защитена комуникационна сесия със страната В.

- Основният подход е страната А да използва софтуер и произволни характеристики на работна станция или сървър, за да генерира сесиен ключ и след това да го обмени със страната В, с която потребителят А иска да комуникира;
- Този обмен се извършва въз основа на използването на публичния ключ на страната В за криптиране на ключа на сесията;

- Публичният ключ на получателя се извлича от директорията и се използва за криптиране на сесийния ключ;
- Страната B получава криптирания сесиен ключ и го декриптира със своя личен ключ.
- След като и двете страни имат ключа на сесията, който е уникално споделен помежду им, те могат сигурно да комуникират.

Генерирането и обмена на сесиен ключ за защита на сесията с използването на криптография с публични ключове е показана на ***Фигура 2.9.***



Фигура 2.9. Криптография с публични ключове – генериране и обмен на сесиен ключ

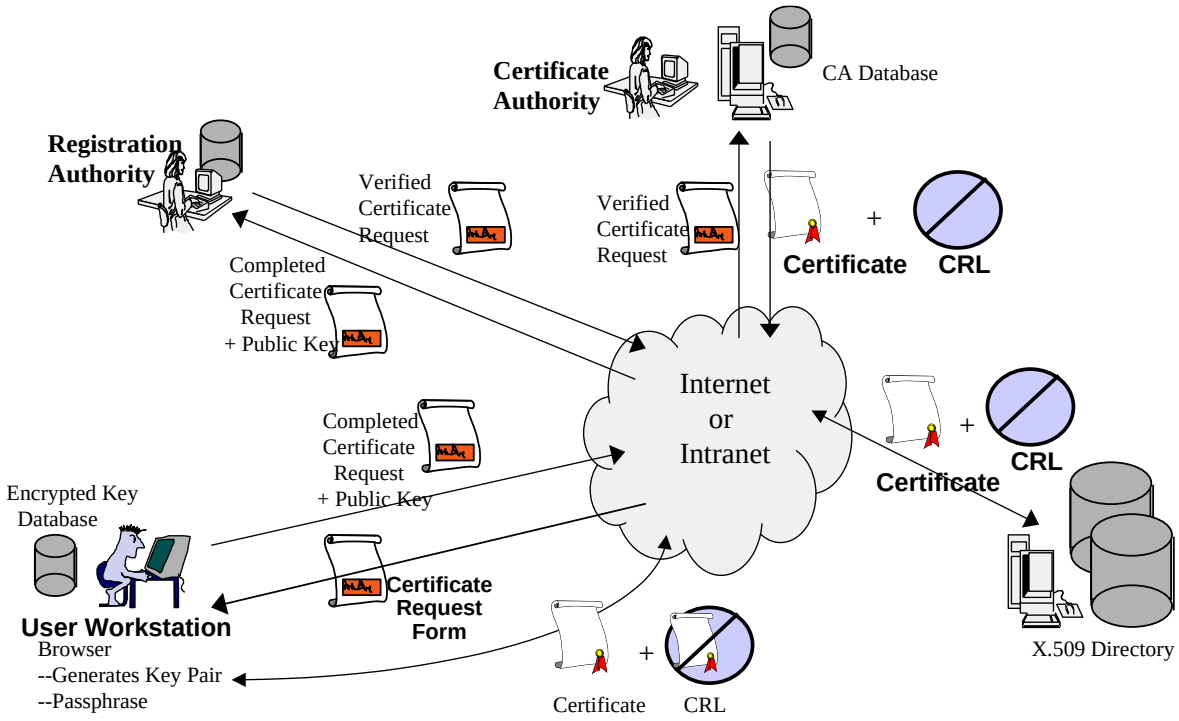
Пример за Инфраструктура на Публични Ключове (*PKI*)

За ефективно използване на криптографията с публични ключове и осигуряване услугите като строга идентификация и автентификация, цифров подпис и криптиране на съобщения е необходима Инфраструктура

на Публичните Ключове (*Public Key Infrastructure, PKI*). Една PKI се състои от пет елемента:

- Сертифициращ орган (*certificate authority*);
- Регистрационен орган (*registration authority*);
- Хранилище за сертификати (*certificate repository*);
- Притежатели на сертификати (*certificate holders*);
- Потребители (*system users, clients*).

Най-добрият начин да се представи функционирането на PKI е да се проследи стъпка по стъпка жизнения цикъл за издаване и използване на сертификатите на основата на типични комерсиални PKI продукти:



Фигура 2.10. Инфраструктура на Публичните Ключове (Public Key Infrastructure)

- Потребителят се свързва към Web Certificate Server за да получи формата на заявка за издаване на сертификат (*Certificate Request*

form). Той използва своя browser за генериране двойката публичен/частен ключ, попълва формата на заявка и я изпраща до Certificate Server. Потребителският browser съхранява двойката ключове в собствена локална база данни, защитена сигурно със зададена от потребителя ключова фраза.

- Registration Authority се свързва с Web Certificate Server и преглежда заявката. След проверка на информацията в заявката Registration Authority я автентифицира и я изпраща в Web Certificate Server.
- Certificate Authority достъпва до Web Certificate Server за преглед на заявката. Certificate Authority утвърждава (одобрява) заявката, генерира подписан сертификат, поставя го в Web Certificate Server и го въвежда в X.509 Directory.
- Потребителят се свързва към Web Certificate Server и получава (изтегля) своя сертификат, който Web Browser съхранява в собствената локална база данни.

Съществуват и много други технически аспекти свързани с имплементирането на PKI. Най важните от тях са:

- Как да се защити процеса заявка, утвърждаване (одобрение) и разпространение на сертификата;
- Как да се отмени (анулира) издаден сертификат;
- Как да се защитят директориите;
- Как да се организира крос сертификация (*cross certification*).

2.3. КРИПТОГРАФСКИ СИСТЕМИ ЗА ЗАЩИТА НА ДАННИТЕ

Определение: Криптографска система (криптосистема) ще наричаме криптографски алгоритъм, криптографските услуги и механизми плюс начина на прилагането им.

Ще бъдат разгледани въпросите:

- Криптографски услуги и механизми;
- Сигурност на криптографска система;
- Управление на криптографски ключове;
- Модели за управление на криптографските ключове.

Криптографски услуги и механизми

Разработването на архитектурата на сигурността на компютърните мрежи се базира на услугите за сигурност и механизмите, реализиращи тези услуги . Услугите и механизмите за сигурност са показани в ***Таблица 2.1.***

Таблица 2.1. Услуги и механизми на системата за сигурност на компютърните мрежи

Услуги	Механизми
Поверителност	• Криптиране • Контрол на достъпа
Цялостност	• Хеширане и цифров подпис • Контрол на достъпа
Наличност	• Криптиране • Цифров подпис • Контрол на достъпа
Автентичност	• Криптиране • Цифров подпис
Невъзможност за отказ	• Криптиране • Цифров подпис

Поверителност

Услугите за поверителност (конфиденциалност) осигуряват невъзможност за неоторизиран достъп до данните, съхранявани в работните станции и сървърите или предавани по мрежата. Използва механизмите за контрол на достъпа и криптиране.

Цялостност

Услугите за цялостност осигуряват невъзможността данните да бъдат променени или разрушени от неразрешени действия. Механизмите, които осигуряват това са хеширане, криптиране и цифров подпис. Хеширането е специална техника, основаваща се на пресмятането на контролна сума на данните посредством еднопосочна функция (*one-way function*) и добавянето и към данните. Цифровият подпис е специална криптографска техника.

Наличност

Услугите за наличност осигуряват, че необходимите ресурси са достъпни и използваеми за потребителите, които имат право на това.

Автентичност

Услугите за автентичност осигуряват, че вие сте това за което се представяте. Исторически в компютърните системи автентичността се осигурява от парола. Съществуват много техники за доказване на автентичност (идентификация и автентификация), но всички достоверни се базират на криптографските методи. Използва се също и цифровият подпис.

Невъзможност за отказ

Услугите, свързани с невъзможността за отказ са предназначени за осигуряване на взаимодействията в компютърните мрежи. Те не дават

възможност изпращача да отрече изпращането, а получателя – получаването на дадено съобщение. Основната техника е цифровият подпис.

Сигурност на криптосистема

Криптосистема ще наричаме криптографски алгоритъм плюс начина на прилагането му.

Разбиване на криптографска система ще наричаме възможността за декриптиране на всички съобщения, криптирани с алгоритъма, без да е наличен декриптиращия ключ.

Различните криптосистеми имат различни нива на сигурност в зависимост от степента на трудност на разбиването им. Всички алгоритми, с изключение на един, са теоретически разбиваеми при достатъчно време и изчислителна мощност.

Дължина на ключа

Сигурността на една симетрична криптосистема е функция от:

- Сложността (силата) на алгоритъма;
- Дължината на ключа.

Оценка на времето и цената за директна атака (Brute-Force Attack)

Ако допуснем, че директната атака (проверка на всички възможни ключове) е единствено ефективна (възможна) за разбиването на една криптосистема, естествено възникват въпросите:

- ***Колко време ще е необходимо?***
- ***Колко ще струва?***

Два са параметрите, определящи времето за успешна директната атака:

- Брой на ключове за проверка;
- Скорост (време) за една проверка.

Изборът на дължина на ключа за конкретно приложение зависи от времето, за което искаме данните да бъдат защитени и от стойността на данните.

Таблица 2.2. дава изисквания за минимална дължина на ключа (при симетричните алгоритми) за различни видове информация.

Таблица 2.2. Минимална дължина на ключа за различни данни

Вид информация	Време на секретност	Минимална дължина на ключа
Тактическа военна информация	няколко часа	56 бита
Информация за продукти, сделки и цени	няколко седмици	56 - 64 бита
Търговски тайни	десетилетия	64 бита
Ядрени тайни	40 години	128 бита
Досиета на шпиони	50 години	128 бита
Дипломатически преговори	65 години	128 бита

Ако нужните време и пари за разбиване на алгоритъма са повече от стойността на криптираните данни, тогава криптографската системата се счита сигурна. Компютрите стават много по-бързи и по-евтини. В същото време стойността на данните намалява с течение на времето.

Таблица 2.3. дава броя на паралелните процесори, необходими за успешна директна атака на 64-битов ключ за определен период от време при различни скорости на процесорите.

Таблица 2.3. Брой процесори, необходими за разбиване на алгоритъм с пълно изчерпване за определен период от време

БРОЙ КРИПТИРАНИЯ ЗА СЕКУНДА	БРОЙ ПРОЦЕСОРИ, НЕОБХОДИМИ ЗА РАЗБИВАНЕ НА 64-БИТОВ КЛЮЧ ЗА ВРЕМЕ ОТ:			
	1 ГОДИНА	1 МЕСЕЦ	1 СЕДМИЦА	1 ДЕН
1 милион	590 000	7 100 000	$3{,}1.10^7$	$2{,}1.10^8$
2 милиона	290 000	3 600 000	$1{,}5.10^7$	$1{,}1.10^8$
4 милиона	150 000	1 800 000	7 600 000	$5{,}3.10^7$
32 милиона	18 000	220 000	950 000	6 700 000
256 милиона	2 300	28 000	120 000	830 000

Таблици 2.4., 2.5. и 2.6. дават различни оценки на цената за разбиване на криптографски алгоритъм на основата на предполагаемата цена на компютърните системи в годините, дължината на ключа и необходимото време за разкриване на ключа при директна атака. Оценките за различните години са направени като средна стойност за машини с производителност, както следва:

- 1990 г. – чипове с капацитет 1 милион криптирания за секунда и 2 милиона криптирания за секунда;
- 1995 г. – 2 милиона и 4 милиона;
- 2000 г. – 4 милиона и 32 милиона;
- 2005 г. – 32 милиона и 128 милиона криптирания за секунда.

Управление на криптографските ключовете

В реалния свят управлението на ключовете за една криптосистема е най-важният въпрос за осигуряване на сигурността и.

Таблица 2.4. Цена за разбиване на алгоритъм с 56-битов ключ с пълно изчерпване за определено време (в хиляди щатски долари)

ГОД.	1 ГОДИНА	1 МЕСЕЦ	1 СЕДМИЦА	1 ДЕН
1990	130	1600	6700	47000
1995	64	780	3300	23000
2000	16	190	830	5800
2005	2	24	100	730

Таблица 2.5. Цена за разбиване на алгоритъм с 64-битов ключ с пълно изчерпване за определено време (в хиляди щатски долари)

ГОД.	1 ГОДИНА	1 МЕСЕЦ	1 СЕДМИЦА	1 ДЕН
1990	33000	400000	17000000	12000000
1995	16000	200000	850000	6000000
2000	4100	50000	210000	1500000
2005	510	6200	27000	190000

Таблица 2.6. Цена за разбиване на алгоритъм със 128-битов ключ с пълно изчерпване за определено време (в щатски долари)

ГОД.	1 ГОДИНА	1 МЕСЕЦ	1 СЕДМИЦА	1 ДЕН
1990	6.10^{26}	$7{,}3.10^{27}$	$3{,}2.10^{28}$	$2{,}2.10^{29}$
1995	3.10^{26}	$3{,}7.10^{27}$	$1{,}6.10^{28}$	$1{,}1.10^{29}$
2000	$7{,}5.10^{25}$	$9{,}2.10^{26}$	$3{,}9.10^{27}$	$2{,}8.10^{28}$
2005	$9{,}4.10^{24}$	$1{,}2.10^{26}$	$4{,}9.10^{26}$	$3{,}4.10^{27}$

Разработването на сигурни криптографски алгоритми и протоколи не е труден проблем, но реализирането на сигурно управление на ключовете е изключително важно, защото атаките за разбиване на криптографските системи почти винаги са насочени към слабостите в управлението на ключовете. В компютърните системи управлението на ключовете обикновено се осъществява от Център за Управление на Ключове (*Key Management Center, KMC*). Управлението на ключовете включва:

- Генериране;
- Разпределение;
- Използване;
- Съхранение;
- Време (период) за използване;
- Смяна;
- Унищожаване.

Генериране на ключ

Сигурността на един криптографски алгоритъм се основава на ключа. Ето защо генерирането на ключ е първият проблем за разрешаване. Често се използват някои лоши практики като:

- Редуциране (ограничаване) на ключовото пространство;
- Избор на лош (слаб) ключ.

Примери на използване на лош ключ

Примери на използване на лош ключ са:

- Имена, инициали, рождени дати и друга свързана с личността информация;
- Думи от различни съществуващи бази данни;
- Пермутации от известни думи;

- Комбинации с големи и малки букви в съществуващи думи;
- Думи от чужд език;
- Двойки думи (свързани).

Стандартът ANSI X9.17 определя един метод за генериране на “хубави“ (добри) ключове и включва използването на криптографски алгоритъм.

Разпределение на ключове

Сигурността на разпределение на ключовете се решава чрез използването на:

- Главен ключ (*master key*) или ключ за ключове (*E*);
- Криптоалгоритми с публични ключове (особено важно за големите компютърни мрежи).

Използване на ключове

Използването само на програмни средства за криптиране крие опасности. Наличието на многопотребителски и многозадачни операционни системи е предпоставка за разкриването на ключа. Използването на технически решения е за предпочитане, но е значително по-скъпо.

Съхранение на ключове

Възможни решения за съхранение на ключа са:

- Запомняне на ключа – неприложимо при реални системи;
- Съхраняване на ключа на отделен технически носител (електронен или магнитен), понякога на две или повече части;
- Съхраняване на ключа в криптиран вид;
- Комбинирани методи.

Време (период) за използване на ключове

Няма сигурна криптосистема с безкрайно дълъг период за използване на ключове. Винаги трябва да съществува политика, определяща периода за използване на различните ключове. Критериите за определяне на този период са различни:

- В зависимост от предназначението;
- В зависимост от честотата на използване;
- В зависимост от типа на криптоалгоритъма.

Унищожаване на ключове

Смяната на ключовете предполага и унищожаване на старите ключове. Методите за унищожаване трябва да гарантират невъзможността за повторна употреба на старите ключове. В компютърните мрежи могат да възникват проблеми поради:

- Възможността ключовете да бъдат лесно копирани и съхранявани на много различни места;
- Наличието на процеси, контролиращи паметта;
- Наличието на процеси, извършващи суопинг;
- Наличието на буфериране.

2.4. МОДЕЛИ ЗА УПРАВЛЕНИЕ НА КРИПТОГРАФСКИТЕ КЛЮЧОВЕ

Важен елемент от управлението на криптографските ключове е разпределението на ключовете. Възможни критерии за разпределение са:

- Критерий ***„време“***;
- Критерий ***„групи“***;
- Критерий ***„нива“***.

2.4.1. Разпределение на криптографските ключове по време

Разглежданата компютърна система се състои от краен брой потребители, работещи в мрежова среда и имащи достъп до краен брой криптографски ключове чрез мрежовата среда. Достъпът до криптографските ключове се управлява от Център управление на ключове (KMC).

Описатели

Всеки потребител и всеки криптографски ключ (ресурс) се характеризират с набор от описатели, от стойностите на които KMC определя правата за достъп на потребителя до ключа.

Моделите за разпределение и управление на ресурси са класифицирани по критерии „***време на определяне на достъпа до ресурса от потребителя“***. За потребителя във всеки модел са показани неговите възможности и примитиви. Моделите за разпределение и управление на ключовете класифицирани по критерии „***време на определяне на достъпа до ключа от потребителя***“ са:

- Статичен модел;
- Динамичен модел;
- Хибриден модел.

Статичен модел

При статичния модел за разпределение и управление на ключове, достъпът до всеки ключ от всеки потребител е определен предварително и не се променя във времето.

Дефинираме статичен модел за управление на ключове

SM = <U, R, Muk >, където:

- U = {U(i)| i=1,2,…,n} – крайно множество на потребителите;
- К = {К(k)| k=1,2,…,p} – крайно множество на ключовете;
- Мик = {Мик(i,j)| i=1,2,…,n и j=1,2,…,p} – е матрица, която описва връзката между потребители и ключове.

Muk(i,j) дефинира правомощията на i - тия потребител към j - тия ключ, като:

- ако Muk(i,j) = '1' - i - тия потребител притежава j - тия ключ;
- ако Muk (i,j) = '0' - i - тия потребител не притежава j - тия ключ;
- редът Muk (i,*) – определя списъка на ключовете на i - тия потребител;
- стълбът Muk (*,j) – определя списъка на потребителите притежаващи j - тия ключ.

Правомощията на потребителите са показани в ***Таблица 2.7.***

Функционални възможности на потребителите

Функционалните възможности и примитиви на потребителя са:

- списък на достъпните ключове, примитив ListKeys;
- избор на достъпен ключ, примитив SelectKey;
- използване на ключ, примитив UseKey;

- отказ, примитив Quit.

Таблица 2.7. Правомощия на потребителите

Групи / **Потребители**	**K(1)**	**K(2)**	**K(3)**	**. . .**	**K(p)**
U(1)	1	1			
U(2)		1	1		1
. . .					
U(n)	1				1

Функционални възможности на КМС

Функционалните възможности и примитиви на КМС са:

- дефиниране на потребители и ключове, примитив DefU&K;
- подготовка на таблица на достъпните ключове за всеки потребител, примитив MakeTbl;
- подготовка на носител с таблица на правомощия за всеки потребител и достъпните ключове, примитив MakeDisk.

Разширения на Статичния модел

Едно от възможните разширения на Статичния модел е разширяване на обхвата на взаимодействието между потребители и ключове, като се добавят и възможните действия, които потребителят може да извършва с този ключ – криптира или декриптира.

Muk(i,j), която дефинира правомощията на i - тия потребител към j - тия ключ, се разширява като:

- ако Muk(i,j) = '1' - i - тия потребител притежава j - тия ключ;
- ако Muk(i,j) = '0' - i - тия потребител не притежава j - тия ключ;
- ако Muk(i,j) = 'e' - i - тия потребител притежава j - тия ключ, с възможност само да криптира;
- ако Muk(i,j) = 'd' - i - тия потребител притежава j - тия ключ, с възможност само да декриптира.

Динамичен модел

При динамичния модел, правомощията за използването на криптографски ключ зависят от времето и се определят след поява на необходимостта от използването му.

Дефинираме динамичен модел за разпределени на ключовете

DM = <U, R>, където:

- U = {U(i)| i=1,2,…,n} – крайно множество на потребителите;
- K = {K(k)| k=1,2,…,p} – крайно множество на ключовете;
- i-тият потребител има достъп до j-тия ключ точно тогава, когато получи разрешение от KMC. При необходимост за използване на ключ i-тият потребител прави заявка към KMC за използването на ключ, получава отговор, анализира го и т.н.

Функционални възможности на потребителите

Функционалните възможности и примитиви на потребителя са:

- заявка за ключ, примитив GetKey;
- получаване на отговор, примитив RecieveAnswer;
- анализ, примитив AnalyseAnswer;
- използване на ключ, примитив UseKey;

- отказ, примитив Quit.

Функционални възможности на КМС

Функционалните възможности и примитиви на КМС са:

- дефиниране на потребители и ключове, примитив DefU&K;
- анализ на заявка за ключ, примитив TestGetKey;
- изпращане на отговор, примитив SendAnswer;

Хибриден модел

Хибридният модел е комбинация от статичния и динамичния модел. За всеки потребител и за най-често използваните ключове се използва статичния модел, а за всички останали – динамичния модел.

Дефинираме хибриден модел разпределение на ключове

HM = <U, R, Muk >, където:

- U = {U(i)| i=1,2,…,n} – крайно множество на потребителите;
- R = {R(k)| k=1,2,…,p} – крайно множество на ключовете;
- Muk = {Muk(i,j)| i=1,2,…,n и j=1,2,…,p} – е матрица, която описва връзката между потребители и ключове.

Muk(i,j) дефинира правомощията на i - тия потребител към j - тия ключ, като:

- ако Muk(i,j) = '1' - i - тия потребител притежава до j - тия ключ;
- ако Muk(i,j) = '0' - i - тия потребител не притежава j - тия ключ;
- ако Muk(i,j) = '*' - i-тият потребител има достъп до j-тия ключ точно тогава, когато получи разрешение от КМС. При необходимост за използване на ключа i-тият потребител прави заявка към КМС за използването на ключа, получава отговор, анализира го и т.н.;

- редът Muk(i,*) – определя таблицата на правомощията на i - тия потребител.

Правомощията на потребителите са показани в ***Таблица 2.8.***

Таблица 2.8. Таблица на правомощията

Ключове / **Потребители**	**K(1)**	**K(2)**	**K(3)**	. . .	**K(o)**
U(1)	1	*			
U(2)		1	1		1
. . .					
U(n)	1				*

Функционални възможности на потребителите

Функционалните възможности и примитиви на потребителя са:

- списък на достъпните ключове, примитив ListKeys;
- избор на достъпен ключ, примитив SelectKey;
- заявка за ключ, примитив GetKey;
- получаване на отговор, примитив RecieveAnswer;
- анализ, примитив AnalyseAnswer;
- използване на ключ, примитив UseKey;
- отказ, примитив Quit.

Функционални възможности на КМС

Функционалните възможности и примитиви КМС са:

- дефиниране на потребители и ключове, примитив DefU&K;

- подготовка на таблица на правомощия за всеки потребител, примитив MakeTbl;
- подготовка на носител с таблица на правомощия за всеки потребител и достъпните ключове, примитив MakeDisk.
- анализ на заявка за ресурс, примитив TestGetResource;
- изпращане на отговор, примитив SendAnswer.

2.4.2. Модел за разпределение на криптографски ключове по групи

Моделът за разпределение на криптографски ключове по групи е едно приложение на модела достъп до ресурси по групи и имплементира основното правило от сигурността на информацията ***„необходимост да се знае“***. Ще използваме и термина ресурс вместо криптографски ключ когато няма възможност от неясноти.

Модел на разпределение

Дефинираме модел за разпределение на криптографски ключове по групи

G = <U, R, G >, където:

- U = {U(i)| i=1,2,…,n} – крайно множество на потребителите;
- R = {R(k)| k=1,2,…,p} – крайно множество на криптографски ключове;
- G = {G(j)| j=1,2,…,m} – крайно множество на групите, като $G \subseteq U \cup R$. Т.е. всяка група се състои от потребители и криптографски ключове.

Принцип на достъп

Дефинираме принцип на достъп в модела, както следва:

- Принадлежността на потребителите в групите се описва с множеството UG = {Ug(i,j)| i=1,2,…,n и j=1,2,…,m}, като:
  - ако Ug(i,j)=1, то потребител U(i) принадлежи на група G(j);
  - ако Ug(i j)=0, то потребител U(i) не принадлежи на група G(j).
- Принадлежността на криптографските ключове в групите се описва с множеството RG = {Rg(k,j)| k=1,2,…,p и j=1,2,…,m }, като:
  - ако Rg(k,j)=1, то ключ R(k) принадлежи на група G(j);
  - ако Rg(k,j)=0, то ключ R(k) не принадлежи на група G(j).
- Всеки потребител има достъп само до тези криптографски ключове, с които принадлежат едновременно в някоя група.

На ***Таблица 2.9.*** е дадено разпределението на потребителите по групи.

Таблица 2.9. Разпределение на потребителите по групи

Групи / **Потребители**	**G(1)**	**G(2)**	**G(3)**	. . .	**G(m)**
U(1)	1	1			
U(2)		1	1		1
. . .					
U(n)	1				1

На ***Таблица 2.10.*** е дадено разпределението на ресурсите по групи.

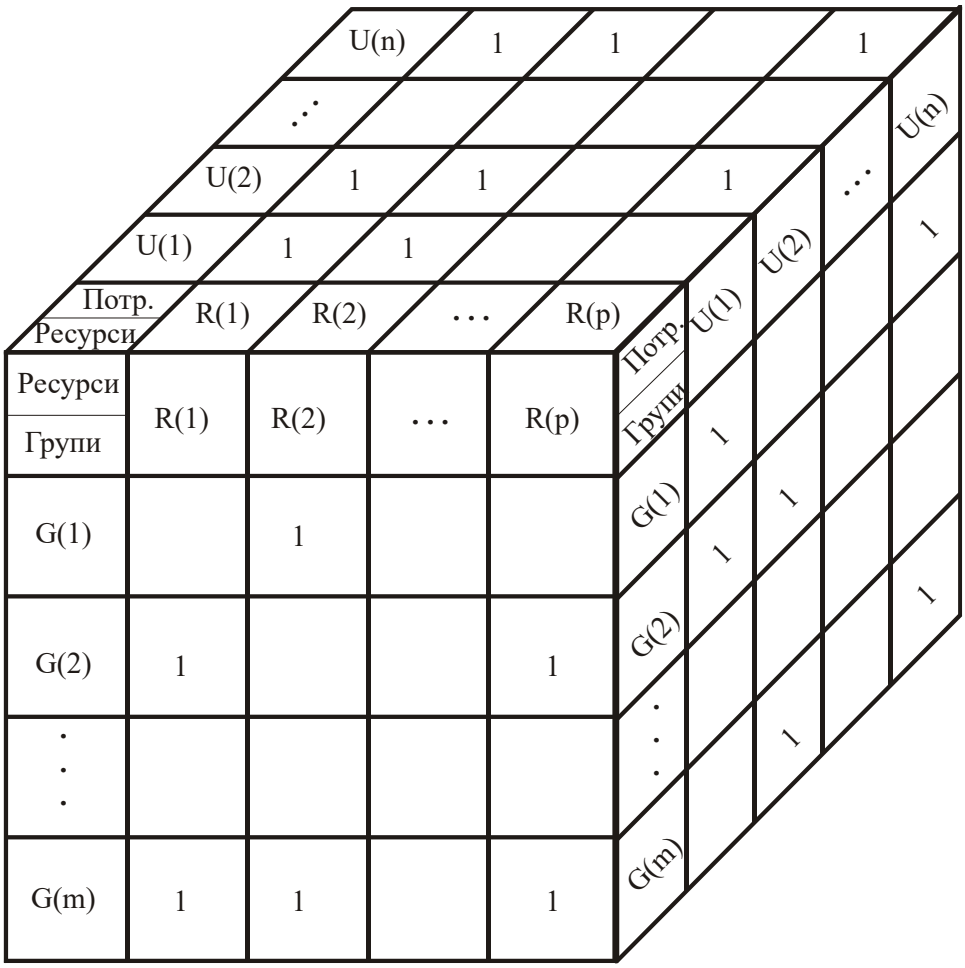
Таблица 2.10. Разпределение на ресурсите по групи

Групи / **Ресурси**	**G(1)**	**G(2)**	*G(3)*	**. . .**	**G(m)**
R(1)		1	1		1
R(2)	1		1		1
. **.** **.**					
R(p)		1			1

Функционални възможности на потребителите

Функционалните възможности на потребителя са:

- заявка за достъп до системата, примитив Ident;
- проверка на правомощия, примитив CheckAuthorities;
- списък на достъпните групи, примитив ListGroups;
- избор на група, примитив SelectGroup;
- списък на достъпните ресурси, примитив ListResources;
- избор на ресурс, примитив SelectResource;
- използване на ресурс, примитив UseResource;
- запис на извършените действия, примитив LogFile;
- изход от системата, примитив Quit.



Фигура 2.11. Взаимодействие между потребители и ресурси при разпределение по групи

2.4.3. Модел за достъп до криптографските ключове по нива

Моделът за разпределение на криптографски ключове по нива е едно приложение на модела достъп до ресурси по групи и имплементира основното правило от сигурността на информацията „***ниво на сигурност***“. Ще използваме и термина ресурс вместо криптографски ключ когато няма възможност от неясноти.

Модел на разпределение

Дефинираме модел за разпределение на криптографски ключове по нива **L = <U, R, L >**, където:

- U = {U(i)| i=1,2,…,n} – крайно множеството на потребителите;
- R = {R(k)| k=1,2,…,p} – крайно множество на криптографските ключове;
- L = {L(l)=l| l=1,2,…,q} – крайно множеството от естествени числа определящи нивата на достъп, като на всеки потребител и на всеки криптографски ключ се присвоява максимално ниво на достъп.

Принцип на достъп

Дефинираме принцип на достъп в модела, както следва:

- На всеки потребител U(i), $i \in \{1,2,\ldots,n\}$ се присъединява число $Lu(i) \in L$, определящо максималното му ниво на достъп.
- На всеки криптографски ключ R(k), $k \in \{1,2,\ldots,p\}$ се присъединява число $Lr(k) \in L$ определящо максималното му ниво на достъп.
- Всеки потребител може да взаимодейства (има достъп) само с тези криптографски ключове (ресурси), чието ниво на достъп е по-малко или равно на неговото;

Връзката между потребители и нива таблично е изразена на ***Таблица 2.11.***

Таблица 2.11. Връзка между потребители и нива

Нива / **Потребители**	**L(1)**	**L(2)**	**L(3)**	. . .	**L(q-1)**	**L(q)**
U(1)	**Lu(1)**					
U(2)		**Lu(2)**				
. . .						
U(n)						*Lu(n)*

Връзката между ресурси и нива таблично е изразена на ***Таблица 2.12.***

Таблица 2.12. Връзка между ресурси и нива

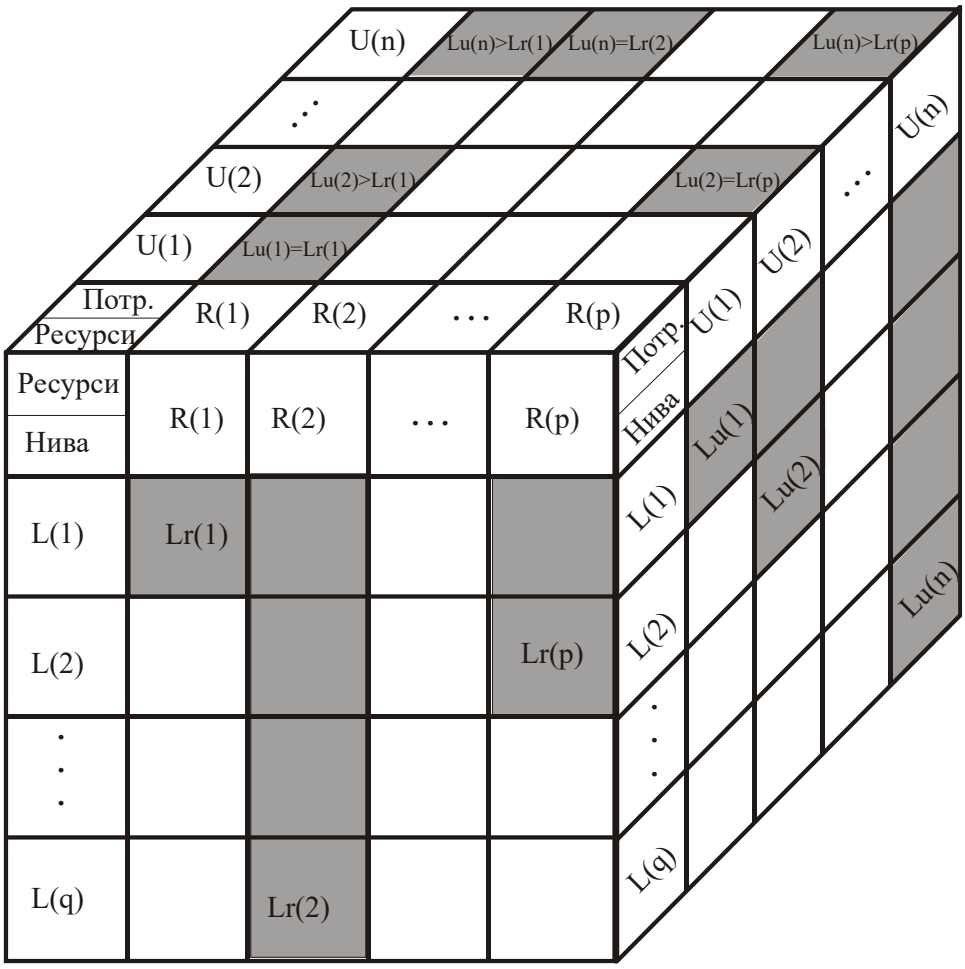
Нива / **Ресурси**	**L(1)**	**L(2)**	**L(3)**	. . .	**L(q-1)**	**L(q)**
R(1)	**Lr(1)**					
R(2)						**Lr(2)**
. . .						
R(p)		*Lr(p)*				

Функционални възможности на потребителите

Функционалните възможности на потребителя са:

- заявка за достъп до системата, примитив Ident;

- проверка на правомощия, примитив CheckAuthorities;
- идентификация на нивото, примитив IdentLevel;
- списък на достъпните ресурси, примитив ListResources;
- избор на ресурс, примитив SelectResource;
- използване на ресурс, примитив UseResource;
- запис на извършените действия, примитив LogFile;
- изход от системата, примитив Quit.



Фигура 2.12. Взаимодействие между потребители и ресурси при разпределение по нива

2.5. ЕДИННО РЕШЕНИЕ CSS ЗА КРИПТОГРАФСКА СИСТЕМА ЗА ЗАЩИТА НА ИНФОРМАЦИЯТА В КОРПОРАТИВНИ ИНФОРМАЦИОННИ СИСТЕМИ

CSS е единно решение за криптографска система за защита на информацията в корпоративна Интранет.

Функционални възможности на CSS

CSS е Интранет базирана система, използва симетрични и асиметрични алгоритми и осигурява:

- идентификация и автентификация на потребители и потребителски приложения;
- криптографска защита на данните на нива файл, сектор и блок данни в работната станция;
- подготовка и проверка на имитосума (на всички нива);
- изработване и проверка на електронен подпис;
- контрол на достъпа до криптографските функции;
- регистрация на извършваните дейности;
- потребителски интерфейс и възможност за вграждане в потребителски приложения.

Архитектура на CSS

CSS се състои от следните модули (***Фигура 2.13.***)

- криптографска машина – ***Crypto Machine (CM)***;
- криптографски програмен интерфейс – ***Crypto Application Program Interface (CAPI)***;
- локални крипто сървъри – ***Local Crypto Servers (LCS)***;
- глобален крипто сървър – ***Global Crypto Server (GCS)***;

- център за администриране и контрол – ***Security Administration and Control Center (SACC)***;
- център за разпределение и управление на криптографски ключове – ***Crypto Keys Distribution and Management Center (CKDMC)***;
- защитени приложения – ***Security Applications***.

Описание на модулите

Crypto Machine

Модулът ***Crypto Machine*** функционира във всеки криптографски процесор и осигурява контрол на достъпа и криптографските функции ***(Crypto Application Program Interface, CAPI).***

Crypto Application Program Interface

Модулът ***Crypto Application Program Interface*** е вграден в модула ***Crypto Machine*** и осигурява връзката му с потребителските приложения, използващи криптографски функции.

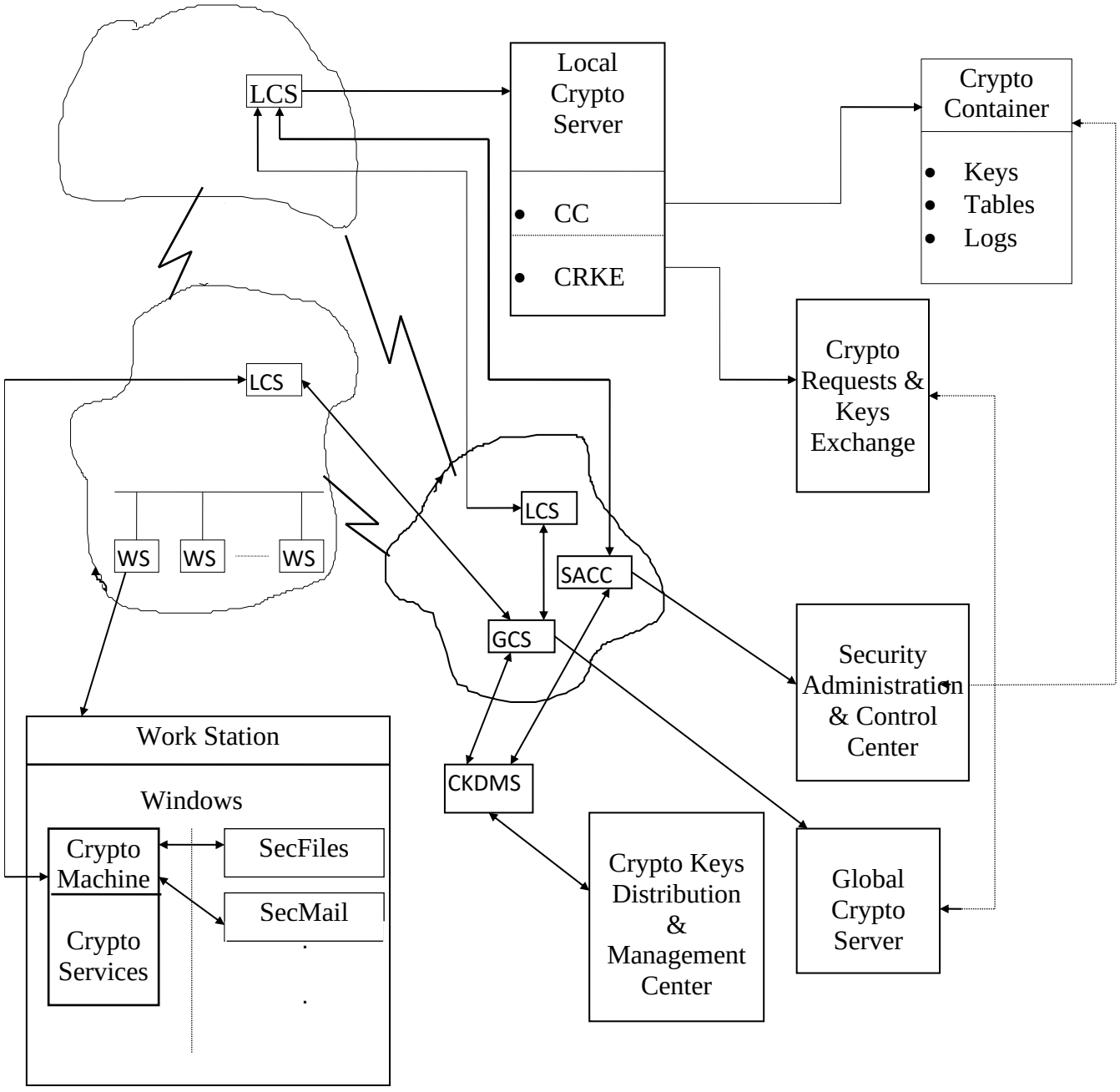
Local Crypto Server

Модулът ***Local Crypto Server*** функционира във всеки възел на системата (един или няколко за всеки възел в зависимост от функционалните приложения). Състои се от:

- ***Crypto Container (CC).*** В него се съхраняват криптографските ключове, системни таблици и дневници на потребителите на този възел (функционална група);
- ***Crypto Requests and Keys Exchange (CRKE).*** Реализира взаимодействията при заявки и обмен на криптографски ключове.

Global Crypto Server

Модулът ***Global Crypto Server*** осигурява изпълнението на заявките за криптографски ключове и последващото им разпределение. Той е един за цялата корпоративна Интранет и има връзка с всички ***Local Crypto Servers*** посредством ***CRKE***.



Фигура 2.13. Архитектура на CSS

Crypto Keys Distribution and Management Center

Модулът ***Crypto Keys Distribution and Management Center*** осигурява разпределението и управлението на ключовете и паролите, подготовката на системни, ключови и паролни носители. Взаимодейства с GCS.

Security Administration and Control Center

Модулът ***Security Administration and Control Center*** осигурява администрирането и контрола на извършените дейности със системата CSS. Поддържа връзки с всички LCS, обобщава и анализира информацията от криптографските дневници.

Security Applications

CSS е отворена система за създаване на приложения за защита на информацията. Типични приложения са за защита на диск, директория и файл, за защита на електронната поща, за защита на електронни страници, за защита на клипборд, за защита на данните в Бази Данни. Базирани на стандартите на Майкрософт, всички приложения на CSS се интегрират естествено в продуктите на Майкрософт (Офис напр.)

Информационни структури

Описанието на CSS включва описание на възлите, работните станции (потребителите), приложенията и групите ключове за всяко приложение. За всяка работна станция се описват приложенията, които ще използват криптографските функции на модула ***Crypto Machine*** и достъпните (в тази работна станция) групи ключове от това приложение.

Основните информационни структури на CSS са:

- Описателите на възела са име (*Node.Name*) и идентификатор (*Node.Id*).

- Описателите на работната станция са име (*WS.Name*) и идентификатор (WS.Id).
- Описатели на приложението са име (Appl.Name) и идентификатор (*Appl.Id*).
- Основните описатели на групата са име (*Group.Name*) и идентификатор (*Group.Id*).

За всяка работна станции (потребител) се подготвят:

- списък на приложенията;
- списъци на достъпните (в тази работна станция) групи ключове (за всяко приложение отделен списък).

Елементи на Системното и Ключовото Пространство

За всяка работна станция и сървър се дефинират следните информационни елементи:

- CryptoMachine.GFG – конфигурационен елемент за криптографския процесор;
- PasswTbl – таблица с паролите;
- ApplTbl – таблица с приложенията за този процесор;
- за всяко приложение ApplGrTbl – таблица с групите на това приложение.

Осигуряване на CSS

В зависимост от назначението си, осигуряването на системата CSS се разделя както следва. Осигуряване за:

- работна станция;
- отговорник по безопасност във възел;
- център за разпределение и управление на ключове;

- център за администриране и наблюдение.

Осигуряване за работна станция

Осигуряването за работната станция включва следните модули:

- Crypto Machine;
- Security Aplications.

Всяка крипто машина от даден възел записва извършените от нея действия в дневник. За всяка услуга се отбелязват:

- IP адрес на работната станция;
- име на защитено приложение, използвало крипто услугата;
- време във формат дд.мм.гг чч:мм:сс;
- използван ключ;
- криптографска услуга (действие на CryptoMachine).

Данните за действията се наблюдават от Центърът за администриране и безопасност за всички възли и работни станции или от Отговорникът по безопасност за крипто машините в един възел.

Осигуряване за отговорник по безопасност във възел

Осигуряването за отговорник по безопасност във възел се състои от следните модули::

- LCSConfig – за конфигуриране на локалния крипто сървър (Local Crypto Serve, LCS);
- LCSActual – поддържат актуално състояние на системните, ключови и паролни носители на средствата за криптографска защита;
- LCSInstall и CMInstall – инсталират осигуряването, както върху LCS на възела, така и върху работните станции;

- LCSConfig и CMConfig – конфигурират програмното осигуряване, както върху LCS на възела, така и върху работните станции;
- CSSView – следят и регистрират извършваните дейности.

Осигуряване за център за разпределение и управление на ключове

Програмното осигуряване за център за разпределение и управление на ключове дава възможност за:

- дефиниране на архитектурата – възли, сървъри, работни станции и описателите им;
- дефиниране на защитените приложения и съответните за всяко приложение ключове;
- дефиниране на релациите – работни станции, приложения и ключове;
- осигуряване на работните станции, сървърите и възлите със системни и ключови носители;
- наблюдение и контрол.

Осигуряване за център за администриране и наблюдение

Осигуряване за център за администриране и наблюдение дава възможност за:

- администриране;
- наблюдение.

Възможни са следните режими на работа:

- наблюдение на всички действия;
- наблюдение на действия в зависимост от различните условия (филтри), на базата на отбелязаните в базата данни събития.

За всяка услуга в базата данни са отбелязват:

- IP адрес на работната станция;
- име на защитено приложение, използвало крипто услугата;
- време във формат дд.мм.гг чч:мм:сс;
- криптографски ключ;
- криптографска услуга;
- код на изпълнение на услугата.

2.6. АРХИТЕКТУРА НА ЗАЩИТЕНА СИСТЕМА ЗА ОБМЕН НА СЪОБЩЕНИЯ ЗА СПЕЦИАЛНИ НУЖДИ

Представеното единно решение на криптографска система за защита на информацията в корпоративна Интранет е основа за изграждане на цялостна система от защитени приложения за работна станция и обхващаща защита на диск, файл, документи, електронна поща, електронни страници, взаимодействие с ORACLE и интегрирани услуги. За всяко приложение са дефинирани предназначението, функционалните възможности, криптографските основи, архитектурата и описанието на основните модули на архитектурата, технологията на функциониране и интеграцията с другите защитени приложения. Списъкът от защитените приложения е както следва:

- Приложение „Защитен диск“;
- Приложение „Защита на файл“;
- Приложение „Защитени документи“;
- Приложение „Защитена електронна поща“;
- Приложение „Защитени електронни страници“;
- Приложение „Защитено взаимодействие с ORACLE“;
- Приложение „Защитени интегрирани услуги“.

Приложение „ЗАЩИТА НА ФАЙЛ“

Предназначение на приложението

Приложението „Защита на файл“ е предназначено да осигури защитата на чувствителна информация на ниво файл в персоналните компютри и компютърните мрежи.

Функционални възможности

Приложението „Защита на файл“ дава възможност за:

- криптиране на файл;

- декриптиране на файл;
- прекриптира файл;
- разглежда файл;
- информация за файл;
- дневник на извършените със системата действия;

Криптографски основи

Приложението „Защита на файл“ е едно приложение на Програмната система за криптографска защита на информацията „CSS“. Използва симетрична криптосхема.

Дава възможности за:

- Криптиране и декриптиране на файл;
- Изработване и проверка на имитосума;
- Идентификация на потребителя.

В този смисъл е осигурено функционирането на всички елементи на електронния подпис, осъществени със симетричен криптографски алгоритъм.

Архитектура на Приложението „Защита на файл“

Приложението „Защита на файл“ се състои от единствен Модул за крайния потребител – CS_file.

Технология на функциониране на „Защита на файл“

Технологията на функциониране на Приложението „Защита на файл“ обхваща:

- Предварителна подготовка;
- Работа с CS_file.

Предварителна подготовка

Предварителната подготовка за работа на „Защита на файл“ включва:

- Дефиниране на потребителите;
- Дефиниране на групите криптографски ключове;
- Подготовка, разпространение и инсталиране на системни и ключови носители.

Работа с CS_file

След стартиране на CS_file се извършва идентификация на приложението в Крипто машината. При успешна идентификация се извежда се информация за дата и часа на последната успешна идентификация.

Криптиране

За криптиране на даден файл е необходимо избор на ключ от списъка на достъпните ключове и изходящ файл, ако искаме да запазим оригиналния файл. Последователността на работа е както следва:

- избор на файл за криптиране;
- избор на ключ;
- запазване или не на името на файла (и избор на ново име), като системата прави проверка за съвпадение имената на входящия и изходящия файл.

Прекриптиране

За прекриптиране на даден файл е необходимо избрания файл да е криптиран от системата с достъпен ключ и да се избере нов ключ, с който да се прекриптира файла. По желание може да се запази стария файл.

Декриптиране

Декриптиране на файл става след избор на криптиран със системата файл и достъпен ключ, с който е криптиран файла. По желание може да се запази оригиналния файл. Процеса на декриптиране е последователното изпълнение на следните действия:

- избор на файл за декриптиране;
- запазване или не на името на файла (и избор на ново име).

Информация

Функцията информация извежда справка за:

- състоянието на всеки файл (криптиран, не криптиран);
- име на ключа;
- коректност на контролната сума.

Разглежда

Функцията разглежда предоставя възможност за разглеждане на екрана на съдържанието на криптиран файл без да се декриптира. Повторно натискане на същия бутон чисти екрана. Изборът на файл е аналогично с предходните функции.

Дневник

CS_file поддържа дневник за извършваните действия (събития). Чрез тази команда се генерира справка от базата със събития на системата. В дневника се записват следните събития:

- влизане в системата (InSys);
- излизане от системата (OutSys);
- криптиране (Crypt);
- прекриптиране (PreCrypt);

- декриптиране (DeCrypt);
- информация (State);
- разглежда (FileView).

Събитията се оформят в таблица със следните колони: дата и час, действие, име на файл, имена нов файл, ключ, и статус. Статусът е кодът за грешка, който се връща от Крипто машината. Всички тези колони описват състоянието на едно събитие с приложението CS_file. Генерираната справка може да се отпечата или да се запази в отчет-файл или текстов файл.

Приложение „ЗАЩИТЕНА ЕЛЕКТРОННА ПОЩА“

Предназначение на приложението

Приложението „Защитена електронна поща“ е предназначено да осигури предаването на защитена информация в компютърните мрежи използвайки съществуващите компоненти на електронната поща.

Използва:

- Функционалните възможности на стандартна електронна поща (MS Outlook);
- Криптографските възможности на Програмната система за криптографска защита на информацията „CSS“.

С увеличаване на възможностите на Internet/Intranet услугите и все по-честото им използване в корпоративните информационни системи, проблемът за сигурността на информацията става все по-актуален. Една от най-често използваните услуги е електронната поща на базата на MS Outlook. Предимствата, обуславящи нейното използване са:

- осъществява сравнително бърза комуникация между потребителите;

- лесна за използване;
- сравнително евтина;
- масово разпространена.

Освен изброените по-горе предимства, тази услуга има и редица недостатъци, като:

- най-често е атакувана от вируси, бомби, червеи, троянски коне и т.н.;
- лесно се подправя;
- дава възможност на автора да отрече изпратено от него съобщение;
- дава възможност на получателя да отрече получено от него съобщение;
- всяко съобщение може сравнително лесно да се прочете, промени и унищожи от трето неоторизирано лице.

Функционални възможности

Приложението „Защитена електронна поща“ дава възможност за създаването на различни варианти на защитена електронна поща в компютърните мрежи.

Дава възможности за:

- Създаването на единна „Защитена поща“ за цялата компютърна мрежа;
- Създаването на „Защитена поща“ в рамките на един възел;
- Създаването на „Защитена поща“ в рамките на едно (или група) управления.

Приложението защитава информацията във всичките елементи на електронната поща, като:

- Субект;

- Текст (основно съобщение);
- Присъединени файлове (документи, таблици, образи и т.н., без ограничения).

Приложение „ЗАЩИТЕНИ ЕЛЕКТРОННИ СТРАНИЦИ“

Предназначение на приложението „Защитени електронни страници“

Приложението „Защитени електронни страници“ е предназначено да осигури създаването и защитен достъп до web-страници, съдържащи „секретна“ и „стр. секретна“ информация в компютърните мрежи, като осигурява контрол на достъпа и защитена комуникация, използвайки съществуващите компоненти за създаване и поддържане на web-страници и Internet Explorer.

Използва

- Функционалните възможности на MS Internet Explorer;
- Всички стандартни средства за създаване и поддържане на web-страници;
- Криптографските възможности на Програмната система за криптографска защита на информацията „CSS“.

Предимства

С увеличаване на възможностите на Internet/Intranet една от най-често използваните услуги е публикуването на информацията в електронни страници. Предимствата, обуславящи нейното използване са:

- осъществява бърза и актуална информация за потребителите;
- лесна за използване от крайния потребител;

- икономична за поддържане;
- масово разпространена.

Слабости

Освен изброените по-горе предимства, тази услуга има и редица недостатъци, като:

- най-често е атакувана от недоброжелатели, злоумишленици, хакери и т.н.;
- всяко страница може сравнително лесно да се прочете, промени и унищожи от неоторизиран потребител;
- липсва на защитена комуникация;
- слабости при идентификацията и автентификацията на сървъра за публикуване на електронната страница;
- слабости при идентификацията и автентификацията на потребителите в сървъра за публикуване.

Функционални възможности

Приложението „Защитени електронни страници“ дава възможност за създаването и поддържането на защитени *web*-страници в компютърните мрежи.

Дава възможности за:

- Създаването и поддържане на защитени *web*-страници;
- Идентификацията и автентификацията на сървъра за публикуване;
- Идентификацията и автентификацията на потребителите в сървъра за публикуване;
- Защитена комуникация между крайния потребител и сървъра за публикуване.

Приложението защитава информацията във всичките елементи на електронната страница като:

- Съдържание;
- Параметри;
- Свързани файлове (документи, таблици, образи и т.н., без ограничения).

2.7. ПРИМЕР ЗА ИНТЕГРАЦИЯ И РАЗПРЕДЕЛЕНИЕ НА КЛЮЧОВЕТЕ ПРИ ЗАЩИТЕНИТЕ ПРИЛОЖЕНИЯ

Компанията „Анализи и Прогнози“

Компанията „Анализи и Прогнози“ извършва стратегически изследвания в областта на информационните технологии и тяхното приложение в системите за сигурност и отбрана. Много от изследванията са финансирани от Правителството и „специалните служби“. Ето защо въпросът за защитата на информацията е ключов за успешно изпълнение на изследванията. Компанията има йерархична структура е изградена както следва:

- Ръководство;
- Дирекции.

Ръководство

- Управител, означаваме с U;
- Управителен съвет.

Управителен съвет

Управителният съвет се състои от трима човека. Означаване с:

- US1;
- US2;
- US3.

Дирекции

Компанията има четири дирекции:

- Дирекция „Анализи“;
- Дирекция „Прогнози“;

- Дирекция „Управление на ресурсите“;
- Дирекция „Вътрешни дела“.

Всяка дирекция има директор и служители.

Дирекция „Анализи“, означаваме с D1

- Директор, означаваме с D1r
- Служители
  - D11
  - D12
  - D13

Дирекция „Прогнози“, означаваме с D2

- Директор, означаваме с D2r
- Служители
  - D21
  - D22
  - D23

Дирекция „Управление на ресурсите“, означаваме с D3

- Директор, означаваме с D3r
- Служители
  - D31
  - D32
  - D33

Дирекция „Вътрешни дела“, означаваме с D4

- Директор, означаваме с D4r
- Служители

- D41
- D42
- D43

Отговорности

Отговорностите в компанията „Анализи и Прогнози“ са разпределени както следва:

- Управителят контролира и ръководи Управителния съвет и всички дирекции;
- US1 е заместник Управител;
- US2 контролира работата на дирекциите „Анализи“ и „Прогнози“;
- US3 контролира работата на дирекции „Управление на ресурсите“ и „Вътрешни дела“;
- Всеки директор на дирекция контролира и ръководи служителите от тази дирекция.

Функционални групи

Изградени са и две функционални групи, като във всяка функционална група участват служители от различните дирекции в зависимост от поставената задача на групата и компетенциите на съответните служители. Групите са:

- Стратегии за развитие;
- Сигурност.

Стратегии за развитие

- Ръководител – US1
- Членове
  - D1r

- D11
- D2r
- D21
- D31
- D41

Сигурност

- Ръководител – D4r
- Членове
  - D12
  - D22
  - D32
  - D42

Условия за защита на информацията

Защитата на информацията в компанията се осъществява с криптографски средства на ниво файл. Необходимо е да бъдат спазени следните условия:

- Всеки служител трябва да има възможност за защита на собствените си файлове;
- Служителите от компанията трябва да имат възможност за достъп до общи файлове в рамките на компанията;
- Ръководството трябва да има възможност за достъп до общи файлове в рамките на Ръководството;
- Ръководството и директорите трябва да имат възможност за достъп до общи файлове в рамките на ръководството и директорите;
- Служителите от всяка дирекция трябва да имат възможност за достъп до общи файлове в рамките на дирекцията;

- Участниците от всяка функционална група трябва да имат възможност за достъп до общи файлове в рамките на групата;
- Управителят трябва да има възможност за четене на всички файлове;
- Всеки член на Управителния съвет трябва да има възможност за четене на всички файлове на служителите от дирекциите за които отговаря;
- Всеки Директор трябва да има възможност за четене на всички файлове на служителите от дирекцията за която отговаря;
- Управителят трябва да има възможност за създаване на файлове, които да бъдат четени от всички служители на компанията;
- Всеки Директор трябва да има възможност за създаване на файлове, които да бъдат четени от всички служители на дирекцията;
- Всеки ръководител на функционална група трябва да има възможност за създаване на файлове, които да бъдат четени от всички членове на групата.

Криптографски ключове

Използваните криптографски ключове могат да бъдат разделени, като:

- Персонални;
- Дирекции;
- Групови;
- Циркулярни.

Персонални криптографски ключове

- Включват ключовете U, US1, …, D43
- Назначение за защита на персоналните файлове на съответните служители от компанията

Дирекции

- Включват ключовете D_1, D_2, D_3, D_4, D_{1a}, D_{2a}, D_{3a} *и* D_{4a}
- Назначение:
  - Ключовете D_1, D_2, D_3 и D_4 са за защита на общите файлове в рамките на съответната дирекция;
  - Ключовете D_{1a}, D_{2a}, D_{3a} и D_{4a} са защита на файлове създадени от директора и четени от всички служители от съответната дирекция.

Групови криптографски ключове

- Включват ключовете *R, RD,* F_1, F_2, F_{1a}, *и* F_{2a},
- Назначение:
  - Ключът *R* е за защита на общите файлове в рамките на Ръководството;
  - Ключът *RD* е за защита на общите файлове в рамките на Ръководството и директорите;
  - Ключовете F_1 и F_2 за защита на общите файлове в рамките на съответната група;
  - Ключовете F_{1a} и F_{2a} за защита на файлове създадени от ръководителя на съответната група и четени от всички служители на съответната група.

Циркулярни

- Включват ключовете C_{all} и U_{all};
- Назначение:
  - Ключът C_{all} е за защита на общите файлове в рамките на компанията;
  - Ключът U_{all} е за защита на файлове създадени от Управителя и четени от всички служители в рамките на компанията.

Разпределение на криптографските ключове

Разпределението на криптографските ключове е показано в ***Таблица 2.13.1***, ***Таблица 2.13.2*** и ***Таблица 2.13.3***, като са изпълнени:

- Всеки ред съответства на потребител;
- Всеки стълб съответства на ключ;
- Съдържанието на клетката (потребител, ключ) е от множеството {*E, D, B*}, където означенията са както следва:
  - *E* – потребителят има права за криптиране с този ключ;
  - *D* – потребителят има права за декриптиране с този ключ;
  - *B* – потребителят има права за криптиране и декриптиране с този ключ;
  - ‘ ‘ – потребителят няма достъп (не притежава) този ключ.

Таблица 2.13.1. Разпределение на криптографските ключове

Ключове *Потребители*	*U*	*US_1*	*US_2*	*US_3*	*D_{1r}*	*D_{11}*	*D_{12}*	*D_{13}*	*D_{2r}*	*D_{21}*	*D_{22}*	*D_{23}*
U	*B*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*
US_1		*B*										
US_2			*B*		*D*	*D*	*D*	*D*	*D*	*D*	*D*	*D*
US_3				*B*								
D_{1r}					*B*	*D*	*D*	*DR*				
D_{11}						*B*						
D_{12}							*B*					
D_{13}								*B*				
D_{2r}									*B*	*D*	*D*	*D*
D_{21}										*B*		
D_{22}											*B*	
D_{23}												*B*
D_{3r}												
D_{31}												
D_{32}												
D_{33}												
D_{4r}												
D_{41}												
D_{42}												
D_{43}												

Таблица 2.13.2. Разпределение на криптографските ключове

Ключове / Потребители	D_{3r}	D_{31}	D_{32}	D_{33}	D_{4r}	D_{41}	D_{42}	D_{43}	R	RD	D_1	D_2
U	D	D	D	D	D	D	D	D	D	D	D	D
US_1									B	B		
US_2									B	B	D	D
US_3	D	D	D	D	D	D	D	D	B	B		
D_{1r}										B	B	
D_{11}											B	
D_{12}											B	
D_{13}											B	
D_{2r}										B		B
D_{21}												B
D_{22}												B
D_{23}												B
D_{3r}	B	D	D	D						B		
D_{31}		B										
D_{32}			B									
D_{33}				B								
D_{4r}					B	D	D	D		B		
D_{41}						B						
D_{42}							B					
D_{43}								B				

Таблица 2.13.3. Разпределение на криптографските ключове

Ключове / Потребители	D_3	D_4	F_1	F_2	C_{all}	U_{all}	F_{1a}	F_{2a}	D_{1a}	D_{2a}	D_{3a}	D_{4a}
U	*D*	*D*	*D*	*D*	*B*	*B*	*D*	*D*	*D*	*D*	*D*	*D*
US_1			*B*		*B*	*D*	*B*					
US_2					*B*	*D*			*D*	*D*		
US_3	*D*	*D*			*B*	*D*					*D*	*D*
D_{1r}			*B*		*B*	*D*	*D*		*B*			
D_{11}			*B*		*B*	*D*	*D*		*D*			
D_{12}				*B*	*B*	*D*		*D*	*D*			
D_{13}					*B*	*D*			*D*			
D_{2r}			*B*		*B*	*D*	*D*			*B*		
D_{21}			*B*		*B*	*D*	*D*			*D*		
D_{22}				*B*	*B*	*D*		*D*		*D*		
D_{23}					*B*	*D*				*D*		
D_{3r}	*B*				*B*	*D*					*B*	
D_{31}	*B*		*B*		*B*	*D*	*D*				*D*	
D_{32}	*B*			*B*	*B*	*D*		*D*			*D*	
D_{33}	*B*				*B*	*D*					*D*	
D_{4r}		*B*		*B*	*B*	*D*		*B*				*B*
D_{41}		*B*	*B*		*B*	*D*	*D*					*D*
D_{42}		*B*		*B*	*B*	*D*		*D*				*D*
D_{43}		*B*			*B*	*D*						*D*

ТВЪРДЕНИЕ: Така дефинираните криптографски ключове и разпределението им удовлетворяват условията за защита на информацията в компанията „Анализи и Прогнози“.

ДОКАЗАТЕЛСТВО: С последователна проверка на условията.

2.7. ЗАЩИТЕНА СИСТЕМА ЗА РАБОТА И ОБМЕН НА СЪОБЩЕНИЯ ЗА СПЕЦИАЛНИ НУЖДИ

При изграждането на автоматизирани информационни системи широко се използват компютърни мрежи от тип „звезда“, в които един централен възел кореспондира с множество възли, на които работят различни потребители. Естествено възниква въпросът за защитата на съобщенията, обменяна между възлите и за организирането на различни нива на достъп на потребителите до информацията. Този проблем е актуален както за компютърните мрежи (въпреки предоставените от мрежовата среда възможности), така и за предаване на съобщения, организирани по различни комуникационни канали.

Среда и функциониране на защитената система

Разглежданата защитена система за работа и обмен на съобщения е от тип „звезда“ и се състои от:

- абонатски пунктове (П1, П2,..., Пm);
- център (Ц);
- връзки на центъра с всички абонатски пунктове (направления).

Абонатски пункт

Абонатският пункт се състои от:

- оператор;
- потребители. Всеки потребител се характеризира с:
  - име (*U_Name*);
  - личен идентификационен номер (*U_Id*)
  - ниво на достъп (*U_Level*);
- ръководител;

- групи – обединение на потребители по различни функционални признаци. Всяка група се характеризира с:
  - име (*GR_Name*);
  - списък на потребителите от тази група (*GR_U_List*).

Център

Центърът се състои от:

- диспечер (Д);
- отдели (О1, О2,...,Оn).

Отдел

Отделът се състои от:

- отговорници;
- ръководител.

Направление

Обменът на съобщения се извършва по направления при спазване на ограничителни условия.

Направление

Направлението е логическата връзка между центъра и даден абонатски пункт за защитен обмен на съобщения.

Ограничителни условия

Направленията се изграждат при следните ограничителни условия:

- всеки абонатски пункт кореспондира с точно един отдел (т.е за всяко направление отговаря точно един отдел);
- един отдел може да кореспондира с няколко абонатски пункта, като всяко направление се обслужва от точно един отговорник от този отдел.

Направленията в защитената система се дефинират посредством таблица на направленията (***фиг. 1***.), удовлетворяваща условията:

- всеки ред съответства на отдел;
- всеки стълб съответства на абонатски пункт;
- всяко направление се отбелязва със *;
- във всеки стълб има точно една *.

Таблица 2.14. Таблица на направленията

АП Отдел	$П_1$	$П_2$	…	$П_m$
$О_1$	*			
$О_2$				
…				
$О_n$		*		

Обменът на съобщения по дадено направление може да се извърши по два начина:

- отдел – абонатски пункт;
- отдел – диспечер – абонатски пункт.

Съобщение

Съобщението в защитената система за работа и обмен на съобщения е структурирана информация и се състои от:

- адресна част;
- анотация;
- текст.

Адресна част

Адресната част на съобщението се състои от следните полета:

- Съобщение: име на съобщението и тип. Типът на съобщението определя начина за работа с него;
- Получател: абонатски пункт (център) и име на потребителя;
- Подател: абонатски пункт (център), име на потребителя и дата;
- Утвърдил: име и дата;
- Срочност: дата.

Анотация

Анотацията на съобщението се състои от:

- Съдържание: свободен текст, резюмиращ съдържанието на съобщението;
- Ниво: определящо нивото на достъп до съдържанието на анотацията.

Текст

Текстът на съобщението се състои от:

- Текст: може да бъде свободен или структуриран. В зависимост от това съобщенията се делят на:
  - свободни;
  - формализирани.
- Ниво: определящо нивото на достъп до съдържанието на текста.

Ниво на достъп

За анотацията и текста нивото може да бъде:

- All – всички потребители имат достъп до съдържанието на съответната част от съобщението;

- >N – всички потребители, за които U_Level>N, имат достъп;
- N – всички потребители, за които U_Level=N, имат достъп;
- GR_Name – всички потребители, които участвуват в групата с име GR_Name, имат достъп;
- U_Name – само потребителят с име U_Name има достъп.

При наличие на повече от едно съобщение за предаване по дадено направление те (съобщенията) се обединяват в пакет. Структурата на пакета е както следва:

- Брой на съобщенията в пакета (N_messages);
- Последователност от N_messages на брой съобщения.

Работа и достъп до съобщение

Работата и обменът в абонатския пункт се осъществяват при спазване на следните правила, регламентиращи достъпа до съобщенията:

- за съобщенията, получавани в абонатския пункт, анотацията и текстът съдържат и права за достъп до тях;
- операторът винаги има достъп до адресната част и до анотацията и/или текста, ако в тях са му дадени права за това;
- всеки потребител винаги има достъп до адресната част и до анотацията и/или текста, ако в тях са му дадени права за това (нивото му на достъп е по-голямо или равно на указаното в съответната част на съобщението);
- ръководителят винаги има достъп до цялото съобщение.

Работата и обменът в отдела се осъществяват при спазване на следните правила, регламентиращи достъпа до съобщенията:

- за съобщенията, получавани в отдела, анотацията и текстът съдържат и права за достъп до тях;

- отговорникът винаги има достъп до адресната част и до анотацията и/или текста, ако в тях са му дадени права за това;
- ръководителят винаги има достъп до цялото съобщение.

Диспечерът в центъра винаги има достъп до всички съобщение.

Механизми за защита на съобщенията и пакетите

Защитата на информацията в системата се осъществява на три нива:

- на ниво магнитен носител;
- на ниво съобщение, съхранявано върху магнитен носител;
- на ниво съобщение и пакет, предавани по каналите за връзка.

Защитата на информацията в системата се постига с криптографски средства при използване на следните схеми:

- криптосхема ШС:
  - предназначение – за шифриране на данни;
  - ключ;
  - алгоритъм – симетричен (могат да се използват различни съществуващи алгоритми);
- криптосхема ИС:
  - предназначение – за създаване на имитосума;
  - ключ;
  - алгоритъм – симетричен (могат да се използват различни съществуващи алгоритми).

Схема за криптиране на съобщение

Защитената система се базира на следната схема за защита на съобщенията. Криптирането на всяка част от съобщението (адресна, анотация и текст) се извършва в следната последователност:

- частта се криптира по схемата ШС;

- с резултата се изработва имитосума по ИС.

Декриптирането на частите на получения съобщение се извършва в следната последователност:

- с шифрираната част се изработва имитосума;
- шифрираната част се декриптира.

Схема за криптиране на пакет

Криптирането на пакетите се извършва, в следната последователност:

- пакетът се криптира по схемата ШС;
- с резултата се изработва имитосума по ИС.

Декриптирането на получения пакет се извършва в следната последователност:

- с получения пакет се изработва имитосума по ИС;
- полученият пакет се декриптира по схемата ШС.

Организирането на нивата на секретност на съобщенията, правата на достъп на потребителите и идентификацията им се постига с използването на криптографските схеми и различни ключове и пароли. Всеки потребител има индивидуална парола и достъп до част от ключовете на системата.

Ключовете в системата се разделят на:

- за абонатски пункт (за магнитните носители в абонатския пункт);
- индивидуални (за всеки потребител);
- за направление;
- групови (за потребителите, включени в групата);
- циркулярен (за всички потребители от всички абонатски пунктове и центъра).

Функционални възможности

Защитената система за работа и обмен на съобщения дава възможности за:

- дефиниране на абонатски пункт. Всеки пункт се идентифицира с име и идентификационен номер. При дефинирането на пункта се описват с име и идентификационен номер оператор, потребители и ръководител;
- дефиниране на отдел. Всеки отдел се идентифицира с име и идентификационен номер. При дефинирането на отдел се описват с име и идентификационен номер отговорници и ръководител;
- дефиниране на направления. Всяко направление се идентифицира с име и идентификатор. При дефинирането на направления се описват абонатски пункт, отдел и отговорник за направлението в отдела;
- генериране на Криптографските ключове;
- дефиниране на индивидуални пароли.

На потребителите на защитената система се предоставят следните възможности:

- създаване и редактиране на съобщение – за целта се използва специализиран редактор, съобщенията могат да се въвеждат от клавиатурата или от произволен файл, съхраняват се винаги криптирани със съответния ключ;
- пакетиране на съобщения;
- разпакетиране на пакет;
- работа с получени съобщения – получените съобщения носят в себе си информация за правата за достъп и за дейностите, които

могат да бъдат извършени с тях – декриптиране, разпечатване, само разглеждане на екран;

- смяна на индивидуалните пароли;
- информация за извършените дейности, неправомерните опити за достъп до системата и повреди;
- избор на маршрут за обмен и получател;
- изпращане и получаване на пакети.

В осигуряването на отдела и диспечера се включва и таблица на направленията със списък на достъпните направления. Таблицата на направленията съдържа информация за достъпните абонатски пунктове. Таблицата на диспечера съдържа всички абонатски пунктове, на отдела – само тези абонатски пунктове, чиито направления обслужва, а в абонатските пунктове такава таблица липсва.

В осигуряването на абонатския пункт се включва таблица на потребителите, съдържаща информация за оператора, всички потребители и ръководителя.

Технологични схеми за взаимодействие при работа и обмен на съобщения

Технологичните схеми за подготовка и обмен на съобщения между потребителите в абонатския пункт и центъра се определят от следните фактори:

- кой е инициаторът за подготовката и обмена на съобщението – потребителят или центърът;
- дали е необходимо да се даде отговор след получаването и обработката на съобщението.

Основните технологични схеми на взаимодействия са, както следва:

- Инициатор АП, без отговор;
- Инициатор АП, с отговор;
- Инициатор Център, без отговор;
- Инициатор Център, с отговор.

Инициатор АП, без отговор

Основните стъпки на взаимодействието са:

- абонатският пункт подготвя съобщението;
- абонатският пункт предава съобщението;
- центърът приема съобщението;
- центърът обработва съобщението.

Инициатор АП, с отговор

Основните стъпки на взаимодействието са:

- абонатският пункт подготвя съобщението;
- абонатският пункт предава съобщението;
- центърът приема съобщението;
- центърът обработва съобщението, подготвя и предава отговор;
- абонатският пункт приема отговора;
- абонатският пункт обработва отговора.

Инициатор Център, без отговор

Основните стъпки на взаимодействието са:

- центърът подготвя заявка за получаване на съобщение;
- центърът предава заявка за получаване на съобщение;
- абонатският пункт приема заявката и подготвя съобщението;
- абонатският пункт предава съобщението;

- центърът приема съобщението, подготвя и предава отговор;
- центърът обработва съобщението.

Инициатор Център, с отговор

Основните стъпки на взаимодействието са:

- центърът подготвя заявка за получаване на съобщение;
- центърът предава заявка за получаване на съобщение;
- абонатският пункт приема заявката и подготвя съобщението;
- абонатският пункт предава съобщението;
- центърът приема съобщението;
- центърът обработва съобщението;
- центърът подготвя и предава отговор;
- абонатският пункт приема отговора;
- абонатският пункт обработва отговора.

ГЛАВА 3. КЛАСИЧЕСКА КРИПТОГРАФИЯ

Трета глава е посветена на класическата криптография. Следвайки историческата хронология и класификацията на криптографските методи по начин на трансформация на откритото съобщение в шифрирано, броя на символите и броя на използваните азбуки, последователно са разгледани:

- ***Методи базирани на заместванията. Едно буквено криптиране***. Представени са алгоритмите – Атбаш (иврит), POT1(k), Квадрат Полибий и шифър на Цезар;
- ***Методи базирани на заместванията. Много буквено криптиране***. Представени са алгоритмите – Шифър на Плейфеър (Playfair), Шифър на Хил (Hill), Шифър Двоен Плейфеър и Шифър Четири матрици (квадрата);
- ***Методи базирани на заместванията. Много азбучни шифри***. Представени са алгоритмите – Шифър на Вижинер, Диск на Джеферсън и Диск на Алберти;
- ***Методи базирани на разместванията.*** Представени са алгоритмите – Ранен спартански шифър (scytale), Шифър Rail Fence (релсова ограда), Метод на маршрута, Шифър Ред – стълб и Шифър Двоен ред – стълб;
- ***Роторни машини***. Този подход е обединение на заместителни и разместителни шифри. Представено е описанието, структурата и използването им.

Класически модел за криптиране/декриптиране

В основите на класическия модел за криптиране/декриптиране са:

- Крайна азбука – за откритото и криптирано съобщение;

- Открит текст $X=[X_1, X_2,...,X_m]$, с дължина m – m-те елемента са букви в крайната азбука;
- Секретен ключ $K=[K_1, K_2,...,K_p]$, с дължина p;
- Криптиран текст $Y=[Y_1, Y_2,...,Y_n]$, с дължина n;

- Със съобщението X и ключът K криптиращият алгоритъм получава криптираното съобщение – $Y=E_K(X)$;
- Получателят може да декриптира съобщението Y и да получи откритото съобщение X – $X=D_K(Y)$.

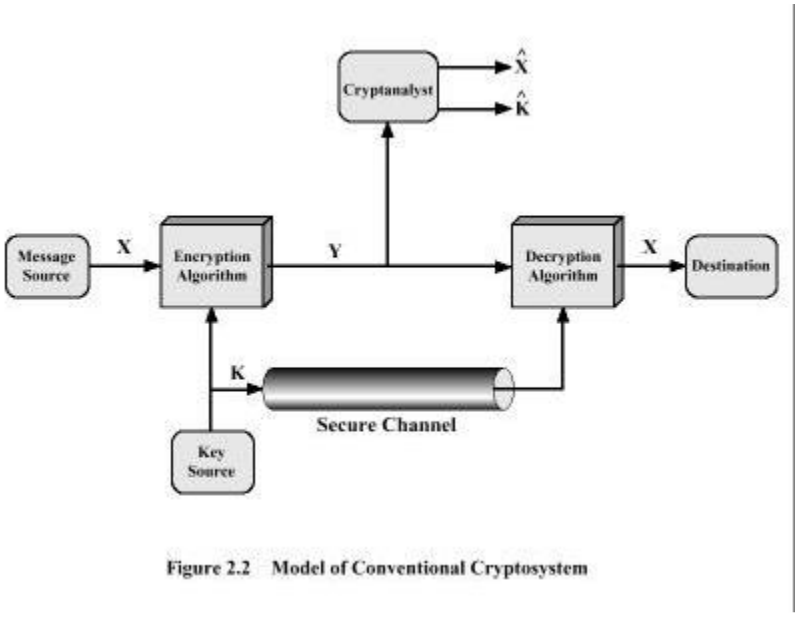


Figure 2.2 Model of Conventional Cryptosystem

Фигура 3.1. Криптиране и декриптиране на съобщение

Азбука

Ще разглеждаме българската азбука:

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я

Правоъгълна таблица – 5x6

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Класификация на методи за криптиране

Методите за криптиране могат да бъдат класифицирани по критериите:

- Тип на криптиращия алгоритъм;
- Използвания брой азбуки;
- Брой обработвани едновременно букви от откритото съобщение.

Основни методи могат да се представят като:

- **Заместване (субституция):**
  - Брой азбуки:
    - Една азбука:
    - Много азбуки:
  - Брой букви:
    - Една буква;
    - Много букви:
- **Разместване (пермутация).**

3.1. МЕТОДИ БАЗИРАНИ НА ЗАМЕСТВАНИЯ. ЕДНО БУКВЕНО КРИПТИРАНЕ

В методите базиране на заместване и използващи само една азбука ще бъдат разгледани:

- Атбаш;
- ROT1(k);
- Квадрат Полибий;
- Шифър на Цезар.

3.1.1. АТБАШ

Криптографски метод в който първата буква от азбуката се заменя с последната, втората с предпоследната и т.н.

Математическо представяне:

Ако n е броят на буквите в азбуката, тогава на мястото на I-тата буква от азбуката се поставя n – I + 1, или с други думи заместването е както следва: I се замества с n – I + 1

ПРИМЕР

АТБАШ шифрира **ВЕСЕЛИН** в **ЪШМШТХР**.

3.1.2. ROT1(K)

Този шифър е познат на много деца. Ключът към шифъра е много прост – всяка буква на азбуката се заменя със следващата подред, А се заменя с Б, Б с В и така нататък. ROT1 буквално означава ***„да се завърти на 1 буква напред в азбуката“***.

ПРИМЕР

РОТ 1 шифрира **ВЕСЕЛИН** в **ГЖТЖМЙО**.

Този шифър е предназначен за развлечение, тъй като е много лесен за разбиране и ползване и съответно за дешифриране, дори ако ключът се използва в обратна посока.

3.1.3. Квадрат на ПОЛИБИЙ

Криптографията с използване на Квадрата на Полибий е предложен от Полибий (гръцки историк, войник, държавник, III век пр. н. е.). Този тип кодиране се прилага за първи път за гръцката азбука, но по-късно е удължен до други езици. В основата е замяната на всяка буква от съобщението с друга и алгоритъмът е известен и като една от най-старите системи за кодиране.

Реализация

Разглеждаме:

- Българската азбука – 30 букви;
- Правоъгълна таблица = 5 х 6;
- Във всяка клетка по една буква.

Получаваме правоъгълната таблица, както следва:

	1	**2**	**3**	**4**	**5**	**6**
1	А	Б	В	Г	Д	Е
2	Ж	З	И	Й	К	Л
3	М	Н	О	П	Р	С
4	Т	У	Ф	Х	Ц	Ч
5	Ш	Щ	Ь	Ъ	Ю	Я

Варианти

Възможни са вариантите:

- Българската азбука – 30 букви;
- За някои методи се използва правоъгълна таблица с размери 5x6, а за други квадратна – 5x5;
- Някой от двойки букви записваме в една клетка, както следва:
  - И/Й;
  - Ч/Ц;
  - Ш/Щ;
  - Ь/Ъ;
  - Ю/Я.

Получаваме квадратната таблица, както следва:

	1	**2**	**3**	**4**	**5**
1	А	Б	В	Г	Д
2	Е	Ж	З	И/Й	К
3	Л	М	Н	О	П
4	Р	С	Т	У	Ф
5	Х	Ч/Ц	Ш/Щ	Ь/Ъ	Ю/Я

Методи за шифриране по Полибий

Думата за шифриране е **ВЕСЕЛИН**

Метод 1.

Шифриране. Всяка буква от откритото съобщение се замества със следващата (***надолу в стълба от таблицата***). Ако е последна – първата.

Резултат

ВЕСЕЛИН

ЗЛЧЛРОТ

Дешифриране: Всяка буква от шифрираното съобщение се замества със предходната (***нагоре в стълба от таблицата***). Ако е първа – последната.

Вариант: Да се заменят по ред.

Метод 2.

Шифриране. Последователно се извършват действията:

- Буквите от таблицата се идентифицират с координата (ред, стълб);
- За всяка буква координатите и (ред, стълб) се записват в колона съответно в два реда;

ПРИМЕР

Открит текст: **ВЕСЕЛИН**

В	**Е**	**С**	**Е**	**Л**	**И**	**Н**
1	**2**	**4**	**2**	**3**	**2**	**3**
3	**1**	**2**	**1**	**1**	**4**	**3**

- Координатите на буквите от шифрираното съобщение се получават след интерпретиране (четене) на двойки координати по хоризонтал:

  1,2 4,2 3,2 3,3 1,2 1,1 4,3

  От таблицата се получава **БСМНБАТ**

Резултат

*Открит текст***: ВЕСЕЛИН**

Шифриран: **БСМНБАТ**

Дешифриране: По обратния ред.

Метод 3.

След записа на цифрите – циклично завъртане на 1 стъпка или повече стъпки, само на първия ред, само на втория ред или и на двата реда.

Нови методи

Могат да се генерират множество нови методи, като:

- Използва се цикличен запис (с определено отместване) на азбуката в таблицата;
- Използване на ключ за попълване на буквите в таблицата (например ЦЕЛКОВ);
- Случайна permутация на азбуката.

Използване на Полибий за предаване на сигнали в древността

Използват се светлинни сигнали, последователно предаващи координатите на съответните букви.

ПРИМЕР

За буквата В последователно се подават един и три светлинни сигнали (координати (1, 3), първи ред, трети стълб).

3.1.4. Шифър на ЦЕЗАР

Исторически бележки

Първият шифър, добил популярност, е заместителният шифър, използван от Юлий Цезар около 58 г. пр. Хр. Сега се нарича ***„шифърът на***

Цезар“. Цезар сменял мястото на всяка буква във военните си заповеди, за да ги накара да изглеждат безсмислени, ако попаднат в ръцете на врага.

Описателен пример

Представете си, че Алис и Боб са решили да общуват чрез шифъра на Цезар. Първо, те ще трябва предварително да се договорят за отместване, например с 3, така че, за да зашифрова своето съобщение, Алис ще трябва да премести всяка буква с 3 позиции в първоначалното си съобщение, така че А да стане Г, Б да стане Д, В да стане Е и така нататък. Така нечетимо или закодирано, съобщението е изпратено на Боб по открит канал. Тогава Боб просто премества всяка буква с 3 позиции назад, за да прочете първоначалното съобщение.

Звучи невероятно, но този прост шифър е бил използван от военачалници в продължение на стотици години след Цезар. ***„Воювал съм и съм побеждавал. Но все още не съм успял да подчиня човешкия дух, който е несломим.“***

Обаче, една ключалка е само толкова силна, колкото е най-слабата ѝ точка. Разбивач на ключалки ще търси за механични дефекти – или ако не намери такива, ще опита да извлече информация – за да изведе правилната комбинация. Процесите по разбиване на ключалка и този за разбиване на код са много сходни. Слабото място на шифъра на Цезар било публикувано 800 години по-късно от арабски математик на име ***Ал-Кинди***. Той разбил шифъра на Цезар, като използвал методика, основана на важно свойство на езика, на който е написано едно съобщение. Ако прегледаш текст от произволна книга и преброиш честотата на срещане на всяка буква, ще откриеш умерена последователност Това се нарича „***честотен анализ***“. И бил тежък удар по сигурността на шифъра на Цезар.

Описание на шифъра на Цезар

Шифърът на Цезар използва:

Азбука

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я

***Ключ* – k –** число между 1 и броя на буквите в азбуката **(**за българския език 30**, 1<= к<30)**

Шифриране: Всяка буква се замества с друга по правилото, ако I е поредния номер на буквата в азбуката, тя се замества с буква с пореден номер I + k (mod 30)

ПРИМЕР: ако k = 3, на практика всяка буква се замества с буква от азбуката циклично отместена с 3.

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я

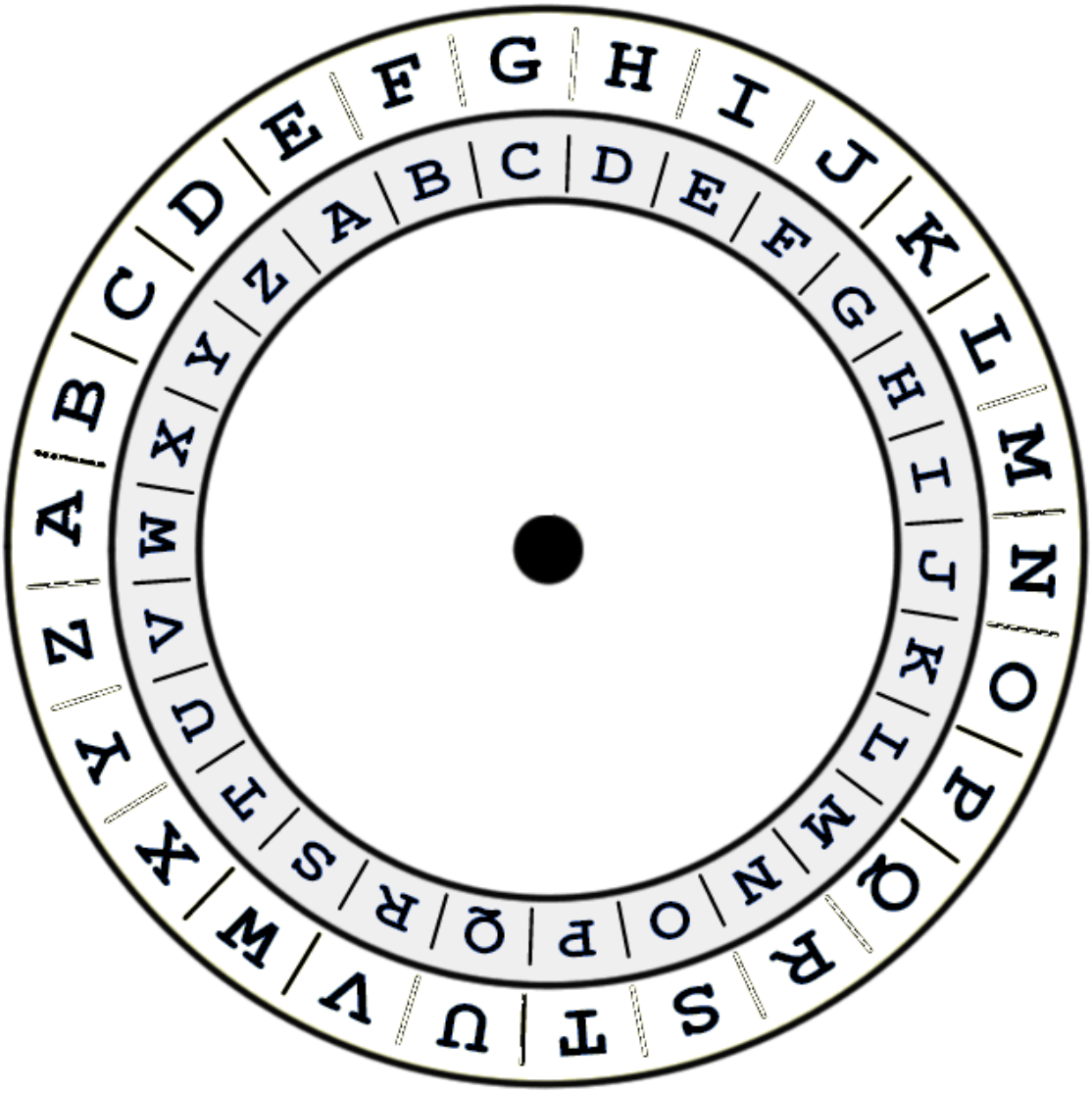
Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я А Б В

Тогава откритото съобщение:

Лекциите по въведение в криптографията и сигурността на данните са всеки понеделник

се шифрира по ЦЕЗАР с ключ 3 и се получава шифрираното съобщение:

Оинщллхии тс еяезирли е нултхсжугчлвнг л флжцурсфххг рг згррлхи фг ефинл тсризиорлн



Фигура 3.2. Механична крипто машина „ЦЕЗАР“

Механичен пример

- Два концентрични и въртящи се кръга в краищата на които последователно са написани буквите на съответната азбука;
- Ключът (отместването) се настройва с въртене на вътрешния кръг;
- Шифриране:
  - Буквите от откритото съобщение се избират от външния кръг;
  - Буквите от шифрираното съобщение са определят съответно от вътрешния кръг;
- Дешифриране – очевидно.

3.2. МЕТОДИ БАЗИРАНИ НА ЗАМЕСТВАНИЯ. МНОГО БУКВЕНО КРИПТИРАНЕ

Като шифри базирани на разместванията, използващи една азбука и обработващи повече от една буква в процеса на криптиране ще бъдат разгледани:

- Шифър на Плейфеър (Playfair);
- Шифър на Хил (Hill);
- Шифър двоен Плейфеър;
- Шифър Четири матрици (квадрата).

3.2.1. Шифър на ПЛЕЙФЕЪР (PLAYFAIR)

Шифърът на Плейфеър е заместителен шифър, при който две букви от откритото съобщение се заместват с две букви в шифрираното съобщение. Шифърът на Плейфеър използва:

Азбука

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я

Правоъгълна таблица – 5x6

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Характеристики на шифъра на ПЛЕЙФЕЪР

- Метод на много буквено криптиране - криптира всяка двойка букви от съобщението на всяка стъпка.
- Използва ключ, за генериране на криптиращата таблица.

Правила за генериране на криптираща таблица

- Използваме правоъгълна таблица – 5x6.
- В редовете последователно попълваме не повтарящите се букви от ключа.
- Допълваме таблицата с останалите букви от азбуката.

ПРИМЕР ЗА КРИПТИРАЩА ТАБЛИЦА

- Ключ – ЦЕЛКОВ

Криптираща таблица с ключ ЦЕЛКОВ

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Правила за криптиране

- Повтарящи се букви в открития текст се разделят с буква разделител – например с буквата ь
- Открития текст се разделя на двойки букви (при необходимост се допълва с буквата ь

- Когато двете букви от откритият текст попадат в един и същи ред на таблицата, тогава те се заменят с двете букви, които се намират от дясната им страна на реда в таблицата – като цикличен буфер
- Когато двете букви от откритият текст попадат в един и същи стълб на таблицата, тогава те се заменят с двете букви, които се намират под тях в стълба на таблицата – като цикличен буфер
- В противен случай, всека двойка букви от откритият текст се заменя с друга двойка, която се определя от сечението на реда на първата буква и стълба на втората и сечението на стълба на първата буква и реда на втората.

ПРИМЕР ЗА КРИПТИРАНЕ

- Ключ – ЦЕЛКОВ
- Текст – „Лекциите са в понеделник“

Стъпка 1: Повтарящи се букви в откритият текст се разделят с буква разделител – например с буквата ь

- ***Текст:*** Лекциите са в понеделник
- ***Резултат:*** Лекциьите са в понеделник
- ***По двойки:*** ЛЕ КЦ Иь ИТ ЕС АВ ПО НЕ ДЕ ЛН ИК

Стъпка 2: Заместваме по двойки

Ще разгледаме двойките:

- ***ЛЕ***
- ***Иь***
- ***ПО***

ЛЕ

- ***ЛЕ*** – в един и същи ред
- Резултат ***ЛК***

Ц	Е	Л	К	О	В
А	Б	Г	Д	Ж	З
И	Й	М	Н	П	Р
С	Т	У	Ф	Х	Ч
Ш	Щ	Ь	Ъ	Ю	Я

Иь

- ***Иь*** – в различни редове и стълбове
- Резултат ***МШ***

Ц	Е	Л	К	О	В
А	Б	Г	Д	Ж	З
И	Й	М	Н	П	Р
С	Т	У	Ф	Х	Ч
Ш	Щ	Ь	Ъ	Ю	Я

ПО

- ***ПО*** – в един и същи стълб
- Резултат ***ХЖ***

Ц	Е	Л	К	О	В
А	Б	Г	Д	Ж	З
И	Й	М	Н	П	Р
С	Т	У	Ф	Х	Ч
Ш	Щ	Ь	Ъ	Ю	Я

Правила за декриптиране

- Шифрираният текст се разделя на двойки
- Когато двете букви от шифрирания текст попадат в един и същи ред на таблицата, тогава те се заменят с двете букви, които се намират от лявата им страна на реда в таблицата – като цикличен буфер
- Когато двете букви от открития текст попадат в един и същи стълб на таблицата, тогава те се заменят с двете букви, които се намират над тях в стълба на таблицата – като цикличен буфер
- В противен случай, всека двойка букви от открития текст се заменя с друга двойка, която се определя от сечението на реда на първата буква и стълба на втората и сечението на стълба на първата буква и реда на втората.
- Отстранява се очевидната буква разделител ь

Атака

- Плейфеър шифърът работи в пространството 30 x 30 = 900 (26 x 26 = 676 за английски) диаграми (двойки букви), а не в 30 (26) букви
- Графиката на честотния анализ на Playfair шифъра е много по-плоска (заравнена) от графиката на едно азбучен шифър
- Използването на Playfair шифър прави атаките на базата на честотния анализ значително по-сложно – но той все още е обект (податлив) на атаки на базата на честотен анализ

3.2.2.Шифър на ХИЛ (HILL)

Характеристики на алгоритъма

- Криптографски алгоритъм, при който на всяка стъпка се криптират определен брой букви **m** от откритото съобщение
- Криптиращият ключ е квадратна матрица **K**
- Размери на матрицата **m** x **m**
- Елементите (коефициентите) на матрицата са цели числа
- За матрицата **K** съществува обратна матрица $\mathbf{K^{-1}}$ така, че

$\mathbf{K \times K^{-1}=E}$

Криптиране

- Умножаване на матрицата **K** с **m** мерен вектор от откритото съобщение за получаване на m мерен вектор от шифрираното съобщение
- Умножението е по модул „***брой на буквите в азбуката***“

Декриптиране

- Умножаване на матрицата $\mathbf{K^{-1}}$ с **m** мерен вектор от шифрираното съобщение за получаване на **m** мерен вектор от откритото съобщение
- Умножението е по модул ***„брой на буквите в азбуката“***

ПРИМЕР

- **m=3**
- Матрицата **K:**

$$\begin{matrix} \mathbf{K_{11}} & \mathbf{K_{12}} & \mathbf{K_{13}} \\ \mathbf{K_{21}} & \mathbf{K_{22}} & \mathbf{K_{23}} \\ \mathbf{K_{31}} & \mathbf{K_{32}} & \mathbf{K_{33}} \end{matrix}$$

- Открито съобщение P_1 P_2 P_3
- Криптиране

$$C_1 = K_{11} P_1 + K_{12} P_2 + K_{13} P_3$$

$$C_2 = K_{21} P_1 + K_{22} P_2 + K_{23} P_3$$

$$C_3 = K_{31} P_1 + K_{3}2 P_2 + K_{33} P_3$$

- Декриптиране

$$K \times K^{-1} = E$$

3.2.3. Шифър ДВОЕН ПЛЕЙФЕЪР

Характеристики на шифъра ДВОЕН ПЛЕЙФЕЪР

- Метод на много буквено криптиране – криптира всяка двойка букви от съобщението на всяка стъпка
- Използва две криптиращи таблици
- Използва два ключа, за генериране на двете криптиращи таблици

Правила за генериране на двете криптиращи таблици

- Използваме два ключа
- Използваме две правоъгълни таблици – 5x6 (***ПЪРВА*** и ***ВТОРА***)
- В празните таблици ***ПЪРВА*** и ***ВТОРА*** попълваме съответно не повтарящите се букви от двата ключа
- Допълваме таблиците с останалите букви от азбуката

Забележка: Възможни са различни правила за попълване на криптиращите таблици.

ПРИМЕР

- В таблица **ПЪРВА** по редовете последователно попълваме не повтарящите се букви от първия ключ.
- В таблица **ВТОРА**, започвайки от горе дясно по часовниковата стрелка, последователно попълваме не повтарящите се букви от първия ключ.

ПРИМЕР

- Ключ 1 – **ЦЕЛКОВ**
- Ключ 2 – **ВЕСЕЛИН**

Криптираща таблица ПЪРВА с ключ ЦЕЛКОВ

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ ВЕСЕЛИН

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

Правила за криптиране

- Повтарящи се букви в открития текст се разделят с буква разделител – например с буквата ь
- Открития текст се разделя на двойки букви
- За всяка двойка букви с първите букви от откритата и закритата двойка се използва таблица ***ПЪРВА***
- За всяка двойка букви с вторите букви от откритата и закритата двойка се използва таблица ***ВТОРА***
- Когато двете букви от открития текст попадат в един и същи ред на съответната таблица, тогава те се заменят с двете букви, които се намират от дясната им страна на реда в съответната таблица – като цикличен буфер
- Когато двете букви от открития текст попадат в един и същи стълб на съответната таблицата, тогава те се заменят с двете букви, които се намират под тях в стълба на съответната таблица – като цикличен буфер
- В противен случай, всека двойка букви от открития текст се заменя с друга двойка, която се определя от сечението на реда на първата буква и стълба на втората и сечението на стълба на първата буква и реда на втората в съответната таблица (за първата буква – таблица ***ПЪРВА***, за втората – таблица ***ВТОРА***)

ПРИМЕР

- Ключ 1 – **ЦЕЛКОВ**
- Ключ 1 – **ВЕСЕЛИН**
- Текст – „Лекциите са в понеделник“

Стъпка 1: Повтарящи се букви в откритият текст се разделят с буква разделител – например с буквата ь

- ***Текст:*** Лекциите са в понеделник
- ***Резултат:*** Лекциьите са в понеделник
- ***По двойки:*** ЛЕ КЦ Иь ИТ ЕС АВ ПО НЕ ДЕ ЛН ИК

Стъпка 2: Заместваме по двойки

Ще разгледаме двойките:

- ***ЛЕ***
- ***ДЕ***

ЛЕ

- ***ЛЕ*** – са в различни редови и различни стълбове
- Резултат ***ВК***

Криптираща таблица ПЪРВА с ключ **ЦЕЛКОВ**

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ **ВЕСЕЛИН**

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

ДЕ

- ***ДЕ*** – са в един и същи ред
- Резултат ***ЖЗ***

Криптираща таблица ПЪРВА с ключ **ЦЕЛКОВ**

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ **ВЕСЕЛИН**

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

Правила за декриптиране

- Шифрирания текст се разделя на двойки букви (при необходимост се допълва с буквата ь
- За всяка двойка букви с първите букви от откритата и закритата двойка се използва таблица ***ПЪРВА***
- За всяка двойка букви с вторите букви от откритата и закритата двойка се използва таблица ***ВТОРА***
- Когато двете букви от шифрирания текст попадат в един и същи ред на съответната таблица, тогава те се заменят с двете букви,

които се намират от лявата им страна на реда в съответната таблица – като цикличен буфер

- Когато двете букви от шифрирания текст попадат в един и същи стълб на съответната таблицата, тогава те се заменят с двете букви, които се намират над тях в стълба на съответната таблица – като цикличен буфер
- В противен случай, всека двойка букви от шифрирания текст се заменя с друга двойка, която се определя от сечението на реда на първата буква и стълба на втората и сечението на стълба на първата буква и реда на втората в съответната таблица (за първата буква – таблица ***ПЪРВА***, за втората – таблица ***ВТОРА***)
- Отстранява се очевидната буква разделител ь

3.2.4. Шифър ЧЕТИРИ МАТРИЦИ (КВАДРАТА)

Характеристики на шифъра ЧЕТИРИ МАТРИЦИ

- Метод на много буквено криптиране – криптира всяка двойка букви от съобщението на всяка стъпка
- Използва четири таблици, от които:
  - Две открити таблици – ***ПЪРВА_О*** и ***ВТОРА_О;***
  - Две криптиращи таблици.
- Използва два ключа, за генериране на двете криптиращи таблици

Правила за генериране на двете криптиращи таблици

- Използваме два ключа
- Използваме две правоъгълни таблици – 5x6 (***ПЪРВА*** и ***ВТОРА***)

- В празните таблици ***ПЪРВА*** и ***ВТОРА*** попълваме съответно не повтарящите се букви от двата ключа
- Допълваме таблиците с останалите букви от азбуката

Забележка: Възможни са различни правила за попълване на криптиращите таблици.

ПРИМЕР

- Ключ 1 – **ЦЕЛКОВ**
- Ключ 2 – **ВЕСЕЛИН**
- В таблица ***ПЪРВА*** по редовете последователно попълваме не повтарящите се букви от първия ключ
- В таблица ***ВТОРА*** започвайки от горе дясно по часовниковата стрелка последователно попълваме не повтарящите се букви от първия ключ

Криптираща таблица ПЪРВА с ключ **ЦЕЛКОВ**

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ ВЕСЕЛИН

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

Правила за криптиране

- Повтарящи се букви в откритият текст се разделят с буква разделител – например с буквата ь
- Откритият текст се разделя на двойки букви (при необходимост се допълва с буквата ь
- За всяка двойка букви с първите букви от откритата и закритата двойка се използват съответно таблиците ***ПЪРВА_О*** и ***ПЪРВА***
- За всяка двойка букви с вторите букви от откритата и закритата двойка се използват съответно таблиците ***ВТОРА_О*** и ***ВТОРА***
- ***Забележка:***
  - Първата буква от откритата двойка е в ***ПЪРВА_О***
  - Втората буква от откритата двойка е във ***ВТОРА_О***
  - Първата буква от шифрираната двойка е в ***ПЪРВА***
  - Втората буква от шифрираната двойка е във ***ВТОРА***
- Когато двете букви от откритият текст попадат в един и същи ред на съответната таблица, тогава те се заменят с двете букви, които се намират от дясната им страна на реда в съответната таблица – като цикличен буфер
- Когато двете букви от откритият текст попадат в един и същи стълб на съответната таблицата, тогава те се заменят с двете букви, които се намират под тях в стълба на съответната таблица – като цикличен буфер
- В противен случай, всека двойка букви от откритият текст се заменя с друга двойка, която се определя от сечението на реда на първата буква и стълба на втората и сечението на стълба на първата буква и реда на втората в съответната таблица (за първата буква – таблица ***ПЪРВА***, за втората – таблица ***ВТОРА***)

ПРИМЕР

- Ключ 1 – **ЦЕЛКОВ**
- Ключ 1 – **ВЕСЕЛИН**
- Текст – „Лекциите са в понеделник“

Стъпка 1: Повтарящи се букви в открития текст се разделят с буква разделител – например с буквата ь

- ***Текст:*** Лекциите са в понеделник
- ***Резултат:*** Лекциьите са в понеделник
- ***По двойки:*** ЛЕ КЦ Иь ИТ ЕС АВ ПО НЕ ДЕ ЛН ИК

Стъпка 2: Заместваме по двойки

Ще разгледаме двойките:

- ***ЛЕ***
- ***Иь***
- ***АВ***

ЛЕ

- ***ЛЕ*** – са в различни редове и различни стълбове
- Резултат ***ЖЕ***

Иь

- ***Иь*** – са в един и същи стълб
- Резултат ***МГ***

АВ

- ***АВ*** – са в един и същи ред
- Резултат ***ЕД***

Открита таблица ПЪРВА_О

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ПЪРВА с ключ ЦЕЛКОВ

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ ВЕСЕЛИН

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

Открита таблица ВТОРА_О

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Открита таблица ПЪРВА_О

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ПЪРВА с ключ ЦЕЛКОВ

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ ВЕСЕЛИН

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

Открита таблица ВТОРА_О

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Открита таблица ПЪРВА_О

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ПЪРВА с ключ ЦЕЛКОВ

Ц	**Е**	**Л**	**К**	**О**	**В**
А	**Б**	**Г**	**Д**	**Ж**	**З**
И	**Й**	**М**	**Н**	**П**	**Р**
С	**Т**	**У**	**Ф**	**Х**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Криптираща таблица ВТОРА с ключ ВЕСЕЛИН

А	**Б**	**Г**	**Д**	**Ж**	**В**
З	**Й**	**К**	**М**	**О**	**Е**
П	**Р**	**Т**	**У**	**Ф**	**С**
Х	**Ц**	**Ч**	**Ш**	**Щ**	**Л**
Ь	**Ъ**	**Ю**	**Я**	**Н**	**И**

Открита таблица ВТОРА_О

А	**Б**	**В**	**Г**	**Д**	**Е**
Ж	**З**	**И**	**Й**	**К**	**Л**
М	**Н**	**О**	**П**	**Р**	**С**
Т	**У**	**Ф**	**Х**	**Ц**	**Ч**
Ш	**Щ**	**Ь**	**Ъ**	**Ю**	**Я**

Правила за декриптиране

- Шифрирания текст се разделя на двойки букви
- За всяка двойка букви с първите букви от откритата и закритата двойка се използват съответно таблиците ***ПЪРВА_О*** и ***ПЪРВА***
- За всяка двойка букви с вторите букви от откритата и закритата двойка се използват съответно таблиците ***ВТОРА_О*** и ***ВТОРА***
- ***Забележка:***
  - Първата буква от шифрираната двойка е в ***ПЪРВА***
  - Втората буква от шифрираната двойка е във ***ВТОРА***
  - Първата буква от откритата двойка е в ***ПЪРВА_О***
  - Втората буква от откритата двойка е във ***ВТОРА_О***
- Когато двете букви от откритията текст попадат в един и същи ред на съответната таблица, тогава те се заменят с двете букви, които се намират от дясната им страна на реда в съответната таблица – като цикличен буфер
- Когато двете букви от откритията текст попадат в един и същи стълб на съответната таблицата, тогава те се заменят с двете букви, които се намират под тях в стълба на съответната таблица – като цикличен буфер
- В противен случай, всека двойка букви от откритията текст се заменя с друга двойка, която се определя от сечението на реда на първата буква и стълба на втората и сечението на стълба на първата буква и реда на втората в съответната таблица (за първата буква – таблица ***ПЪРВА***, за втората – таблица ***ВТОРА***)
- Отстранява се очевидната буква разделител ь

3.3. МЕТОДИ БАЗИРАНИ НА ЗАМЕСТВАНИЯ. МНОГО АЗБУЧНИ ШИФРИ

Ще бъдат представени методите базирани на заместванията и използващи повече от една азбуки, както следва:

- Шифър на Вижинер;
- Диск на Джеферсън;
- Диск на Алберти.

3.3.1. Шифър на ВИЖИНЕР

Шифърът на Вижинер е много азбучен шифър. Методът шифрира откритото съобщение като се използва поредица от различни шифри на Цезар, свързани с буквите на ключова дума.

Макар да носи името на французина Блез дьо Вижинер шифърът е описан първо от Джовани Батиста Беласо в 1553 г. През 19 век обаче този подход получава името на Вижинер, с което е известен и днес.

Характеристики на шифъра на ВИЖИНЕР

- Метод на едно буквено криптиране – на всяка стъпка криптира само една буква от съобщението
- Метод на много азбучен шифър. Използва всички възможни азбуки от шифъра на Цезар
- Използва буквите на ключ, за да определи коя от азбуките на Цезар ще бъде използване за съответната буква на откритото съобщение

Елементи на шифъра на ВИЖИНЕР

- Ключ – последователност от букви (име, дума, изречение, последователен текст от книга и т.н.)

- Таблица на Вижинер – квадратна таблица (30 х 30 за български език), в която на всеки ред са записани азбуките от шифрите на Цезар, съответно с отместване 0, 1, 2, … , 29.

	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я
А	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я
Б	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А
В	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б
Г	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В
Д	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г
Е	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д
Ж	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е
З	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж
И	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З
Й	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И
К	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й
Л	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К
М	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л
Н	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М
О	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н
П	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О
Р	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
С	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р
Т	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С
У	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т
Ф	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У
Х	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф
Ц	Ц	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х
Ч	Ч	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц
Ш	Ш	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч
Щ	Щ	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш
Ь	Ь	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ
Ъ	Ъ	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь
Ю	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ
Я	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ь	Ъ	Ю

Таблица на Вижинер

Правила за криптиране

- Последователно буквите на ключа се записват под съответната буква вот открития текст. При изчерпване на буквите на ключа процедурата се повтаря до изчерпване на цялото открито съобщение
- Всяка от буквите на ключа определя с коя от азбуките на Цезар (с кой шифриращ ред) ще се шифрира съответната буква на откритото съобщение
- Всяка от буквите на откритото съобщение се замества със съответната буква от определения шифриращ ред

ПРИМЕР

- Ключ 1 – **ЦЕЛКОВ**
- Текст – „Лекциите са в понеделник“

Стъпка 1: Буквите от ключа се записват под буквите на откритото съобщение

Л Е К Ц И И Т Е С А В П О Н Е Д Е Л Н И К
Ц Е Л К О В Ц Е Л К О В Ц Е Л К О В Ц Е Л

Стъпка 2: Определят се шифриращите редове (с начало буквите на ключа)

А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я
Ц Ц Ч Ш Щ Ь Ъ Ю Я А Б В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х
Е Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я А Б В Г Д
Л Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я А Б В Г Д Е Ж З И Й К
К К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я А Б В Г Д Е Ж З И Й
О О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я А Б В Г Д Е Ж З И Й К Л М Н
В В Г Д Е Ж З И Й К Л М Н О П Р С Т У Ф Х Ц Ч Ш Щ Ь Ъ Ю Я А Б

Стъпка 3: Всяка от буквите на откритото съобщение се замества със съответната буква от определения шифриращ ред

Л Е К Ц И И Т Е С А В П О Н Е Д Е Л Н И К
Ц Е Л К О В Ц Е Л К О В Ц Е Л К О В Ц Е Л

Шифрирано съобщение
Г К Х В Ц К К К Ъ Ш Р З Ж Т Р О У Н Е Н Х

Правила за декриптиране

- Последователно буквите на ключа се записват под съответната буква от шифрирания текст. При изчерпване на буквите на ключа процедурата се повтаря до изчерпване на цялото шифрирано съобщение
- Всяка от буквите на ключа определя с коя от азбуките на Цезар (с кой шифриращ ред) е шифрирана съответната буква на откритото съобщение
- Всяка от буквите на шифрираното съобщение от определения шифриращ ред се замества със съответната буква от реда започващ с А

При все че е лесен за разбиране и реализация, шифърът устоява на опитите за разбиването му в продължение на три века и затова си спечелва прозвището неразбиваем шифър Първият общ метод за разбиване на шифъра е предложен от Фридрих Касиски през 1863 г.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB	AC	AD	AE
	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я
А	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я
Б	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А
В	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б
Г	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В
Д	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г
Е	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д
Ж	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е
З	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж
И	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З
Й	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И
К	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й
Л	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К
М	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л
Н	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М
О	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н
П	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О
Р	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
С	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р
Т	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С
У	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т
Ф	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У
Х	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф
Ц	Ц	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х
Ч	Ч	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц
Ш	Ш	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч
Щ	Щ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш
Ъ	Ъ	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ
Ь	Ь	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ
Ю	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь
Я	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ь	Ю

Вариант на таблицата на Вижинер

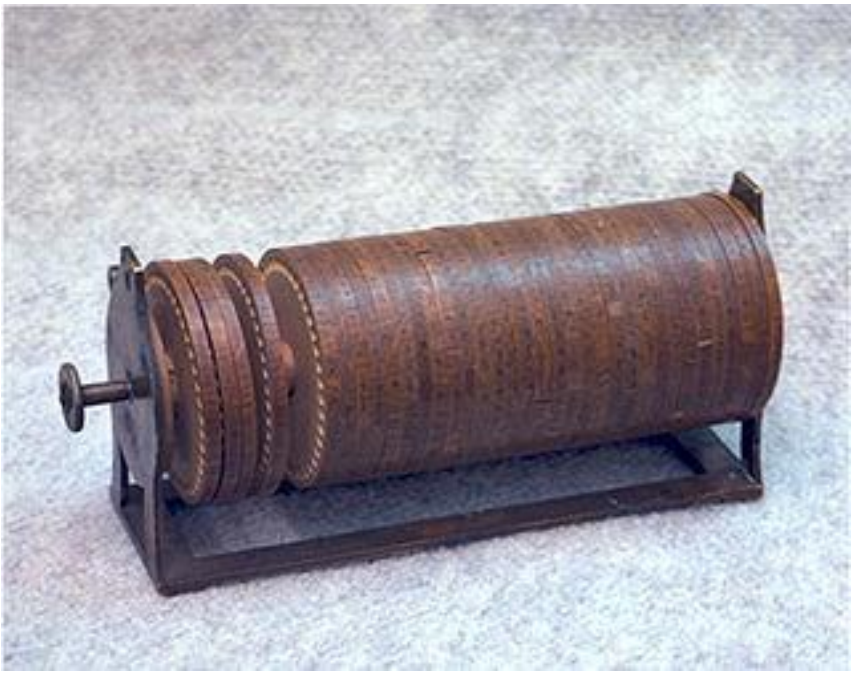
3.3.2. Диск на ДЖЕФЕРСЪН

Дискът на Джеферсън или колело шифър, както го е нарекъл Томас Джеферсън, известен също като *Bazeries Cylinder*, е шифрова система, използваща набор от колела или дискове, всяка с 26 букви от азбуката, разположени около ръба им. Редът на буквите е различен за всеки диск и обикновено се разбърква по някакъв случаен начин. Всеки диск е маркиран с уникален номер. Отвор в центъра на дисковете им позволява да бъдат подредени на ос. Дисковете са подвижни и могат да бъдат монтирани на оста в произволен ред. Редът на дисковете е шифров ключ и изпращачът и

получателят трябва да подредят дисковете в същия предварително определен ред. Устройството на Джеферсън е имало 36 диска.

След като дисковете бъдат поставени на оста в уговорения ред, подателят завърта всеки диск нагоре и надолу, докато в един ред се изписва желаното съобщение. Тогава подателят може да копира всеки ред текст на дисковете, различен от този, който съдържа съобщението за свободен текст. Получателят просто трябва да подреди дисковете в договорения ред, да завърти дисковете, така че да изписват криптираното съобщение на един ред, и след това да огледа редовете, докато не види съобщението открит текст, т.е. реда, който съдържателно изречение. Има изключително малък шанс да има две четливи съобщения, но това може да бъде проверено бързо от кодиращия.

За първи път изобретен от Томас Джеферсън през 1795 г., този шифър не е станал добре известен. Аналогична система се използва от армията на САЩ от 1923 до 1942 г. като М-94.



Фигура 3.3. Диск на Джеферсън (Bazeries Cylinder)

Тази система не се счита за защитена срещу съвременното разбиване на кода, ако се използва за криптиране на повече от един ред текст с една и съща подредба на дискове (т.е. използване на един и същ ключ).

3.3.3. Диск на АЛБЕРТИ

Дискът на ***Alberti Cipher***, описан от Леон Батиста Алберти в своя трактат De Cifris, въплъщава първия пример за много азбучно заместване със смесени азбуки и променлив период. Това устройство, наречено ***Formula***, е съставено от два концентрични диска, прикрепени от общ щифт, който може да се върти единия спрямо другия.

Фигура 3.4. Диск на Алберти (Alberti Cipher)

По-големият се нарича ***Stabilis*** (неподвижен), по-малкият се нарича ***Mobilis*** (подвижен). Обиколката на всеки диск е разделена на 24 равни клетки. Външният пръстен съдържа една главна азбука за свободен текст, а вътрешният пръстен има малка смесена азбука за шифрирания текст. Външният пръстен също включва числата от 1 до 4 за шифриране на кодова книга, съдържаща 336 фрази с присвоени цифрови стойности.

Правила за криптиране

- Неподвижният диск се използва за буквите от откритото съобщение
- Подвижният диск се използва за буквите от криптираното съобщение
- Съответствието между буквите на неподвижния и подвижния се определя от ***подвижен индекс***

Подвижен индекс

Да предположим, че сме избрали да използваме буквата ***k*** като индексна буква в подвижния диск. В момента на писане ще позиционираме двата диска например съпоставяне на индексната буква с главна буква ***B***, с което се определя съответствието на всички други малки букви (от подвижния диск), с главните букви над тях (от неподвижния диск). Шифрираното съобщение започва с главна буква ***B***, след което следват малките букви, съответстващи на главните букви от откритото съобщение. След шифрирането на определен брой букви може да се промени позиционирането на индекса ***к***, например под буквата ***R***. Шифрираното съобщение продължава с главната буква ***R*** и т.н.

Забележка: Появата на главна буква в шифрираното съобщение е указание за промяна на позиционирането на индекса.

Възможно е за индекс да се използват и главните букви от неподвижния диск. Процедира се по аналогичен начин.

Правила за декриптиране

Правилата за декриптиране са аналогични на правилата за криптиране.

- Позиционира се ***подвижният индекс***
- Подвижният диск се използва за буквите от криптираното съобщение
- Неподвижният диск се използва за буквите от откритото съобщение

3.4. МЕТОДИ БАЗИРАНИ НА РАЗМЕСТВАНИЯ

Характеристики на методите базирани на разместванията

- Метод на едно или много буквено криптиране, най-често еднобуквено (едно символно)
- Криптирането се осъществява чрез разместване на местата на буквите в откритото съобщение за получаване на шифрираното съобщение
- Ключът е правилото за разместване на буквите
- Шифрите базирани на разместванията се класифицират като блокови шифри

Математически модел

- Разглеждаме откритото съобщение **M (m_1, m_2, … , m_N),** където:
  - **N** е дължината на съобщението **M** (брой букви в съобщението / блока);
  - **m_1, m_2, … , m_N** са буквите на съобщението **M.**
- **(1,2, … ,N)** е основната пермутация на числата от **1** до **N**
- **P (p_1, p_2, … ,p_N)** в пермутация на числата от **1** до **N**. Използва се за ***ключ за шифриране (шифрираща пермутация)***
- **Криптиране:** Шифрираното съобщение **C (c_1, c_2, … ,c_N)** се получава по правилото:
  - **c_i = m_{pi},** на **i**-то място в шифрираното съобщение се записва буквата която е на **p_i**-то място в откритото съобщение;
  - т.е. **c_1 = m_{p1}, c_2 = m_{p2}**, … , **c_N = m_{pN}.**
- **Декриптиране:** Откритото съобщение **M (m_1, m_2, … , m_N)**се получава по правилото:

- $\mathbf{m_{pi}} = \mathbf{c_i}$, на $\mathbf{p_i}$-то място в откритото съобщение се записва буквата която е на **i** -то място в шифрираното съобщение;
- т.е. $\mathbf{m_{p1}} = \mathbf{c_1}$, $\mathbf{m_{p2}} = \mathbf{c_2}$, … , $\mathbf{m_{pN}} = \mathbf{c_N}$

Забележка: Когато откритото съобщение е с голяма дължина, то се разделя на блокове е дължина **N**.

ПРИМЕР

- Открито съобщение **M (ВЕСЕЛИН)**
- Ключ (криптираща пермутация) **P (7, 5, 4, 3, 1, 6, 2)**
- Шифрирано съобщение **C (НЛЕСВИЕ**)

3.4.1. Ранен СПАРТАНСКИ ШИФЪР (SCYTALE)

В криптографията scytale (skɪtəli, skytale, древногръцки: σκυτάλη skutálē, също σκύταλον, палка, цилиндър) е инструмент, използван за извършване на транспозиционен шифър, състоящ се от цилиндър с лента от пергамент, навита около цилиндър, върху което по редове се написва съобщението. Твърди се, че древните гърци и в частност спартанците са използвали този шифър за комуникация по време на военни кампании.

Фигура 3.5. Ранен спартански шифър

Получателят използва пръчка със същия диаметър, върху която се навиват пергаментът, за да прочете съобщението. Той има предимството да бъде бърз и да не е склонен към грешки – необходимо свойство, когато е на бойното поле.

3.4.2. Шифър RAIL FENCE (РЕЛСОВА ОГРАДА)

Характеристики на шифъра Rail Fence

- Шифърът Релсова ограда е разместителен шифър
- Използва N на брой въображаеми релси, подредени отгоре надолу ***(ключ на шифъра***)
- Буквите на откритото съобщение се записват върху релсите по правилото:
  - Последователно върху релсите отгоре надолу,
  - Запазвайки позицията
- Шифрираното съобщение се получава след прочитането на буквите по релсите, отгоре надолу
- Дешифрирането – по обратен ред:
  - Шифрираното се записва последователно по релсите спазвайки отместването определено от **N**
  - Прочита се откритото съобщение

ПРИМЕР

- Открито съобщение **ЛЕКЦИИТЕ СА В ПОНЕДЕЛНИК**
- При ключ **3** (брой на релсите **3**)

- Записваме откритото съобщение върху въображаемите релси

Л И С О Е К
Е Ц И Е А П Н Д Л И
К Т В Е Н

- Шифрираното съобщение **ЛИСОЕКЕЦИЕАПНДЛИКТВЕН**

- При ключ **2** (брой на релсите **2**)
- Записваме откритото съобщение върху въображаемите релси

Л К И Т С В О Е Е Н К
Е Ц И Е А П Н Д Л И

- Шифрираното съобщение **ЛКИТСВОЕЕНКЕЦИЕАПНДЛИ**

3.4.3. Метод на МАРШРУТА

Характеристики на шифъра Метод на маршрута

- Шифърът метод на маршрута е разместителен шифър
- Използва предварително определена мрежа от позиции
- Буквите на откритото съобщение се записват в позициите на мрежата
- Шифрираното съобщение се получава при прочитане на буквите от мрежата в предварително определен ред
- Забележка. Мрежата от позиции и редът на четене определят ***ключ на шифъра***

3.4.4. Шифър РЕД – СТЪЛБ

Характеристики на шифъра ред – стълб

- Шифърът ***ред*** – ***стълб*** е разместителен шифър
- Използва се правоъгълна таблица с фиксиран брой колони, а броят на стълбовете зависи от дължината на съобщението
- Буквите на откритото съобщение се по редове в таблицата
- Шифрираното съобщение се получава при прочитане на буквите от таблицата по колони, в предварително определен ред
- ***Забележка***. Редът на прочитане на колоните се определят от ***ключ на шифъра***

ПРИМЕР

- Таблица с шест колони
- Ключ – **ЦЕЛКОВ**
- Открито съобщение **ЛЕКЦИИТЕ СА В ПОНЕДЕЛНИК**

Последователност на четене на колоните с ключ **ЦЕЛКОВ**

Ц	Е	Л	К	О	В
6	2	4	3	5	1

Откритото съобщение в таблицата

Ц	Е	Л	К	О	В
6	2	4	3	5	1
Л	Е	К	Ц	И	И
Т	Е	С	А	В	П
О	Н	Е	Д	Е	Л
Н	И	К			

- Шифрираното съобщение **ИПЛЕЕНИЦАДКСЕКИВЕЛТОН**
- Дешифрирането се извършва, както следва:
  - Подготвена е празната таблица с ключа и последователността на колоните
  - От дължината на ключа и дължината на съобщението се определя кои колони са пълни и кои не (В примера 21 и 6, първите три колони са с по четири букви, а вторите три с по три)
  - Попълват се буквите от шифрираното съобщение по колони, като се съобразява с броя на буквите във всяка колона
  - Откритото съобщение се получава след четенето по редове

3.4.5. Шифър ДВОЕН РЕД – СТЪЛБ

Характеристики на шифъра двоен ред – стълб

- Шифърът **двоен ред – стълб** е двукратно прилагане на шифърът **ред – стълб**
- Възможни са решенията с:
  - Използване на един и същи ключ
  - Използване на два различни ключове

ПРИМЕР С ЕДИН КЛЮЧ

- Таблица с шест колони
- Ключ **– ЦЕЛКОВ**
- Открито съобщение **ЛЕКЦИИТЕ СА В ПОНЕДЕЛНИК**

Последователност на четене на колоните с ключ ЦЕЛКОВ

Ц	Е	Л	К	О	В
6	2	4	3	5	1

Откритото съобщение в таблицата (първа стъпка)

Ц	Е	Л	К	О	В
6	2	4	3	5	1
Л	Е	К	Ц	И	И
Т	Е	С	А	В	П
О	Н	Е	Д	Е	Л
Н	И	К			

- Резултат **ИПЛЕЕНИЦАДКСЕКИВЕЛТОН**

Шифрираното съобщение в таблицата (втора стъпка)

Ц	Е	Л	К	О	В
6	2	4	3	5	1
И	П	Л	Е	Е	Н
И	Ц	А	Д	К	С
Е	К	И	В	Е	Л
Т	О	Н			

- Шифрираното съобщение **НСЛПЦКОЕДВЛАИНЕКЕИИЕТ**

ПРИМЕР С ДВА КЛЮЧА

- Таблица с шест колони
- Ключове – **ЦЕЛКОВ** и **ВЕСЕЛИН**
- Открито съобщение **ЛЕКЦИИТЕ СА В ПОНЕДЕЛНИК**

Последователност на четене на колоните с ключ ЦЕЛКОВ

Ц	Е	Л	К	О	В
6	2	4	3	5	1

Откритото съобщение в таблицата (първа стъпка)

Ц	Е	Л	К	О	В
6	2	4	3	5	1
Л	Е	К	Ц	И	И
Т	Е	С	А	В	П
О	Н	Е	Д	Е	Л
Н	И	К			

- Резултат **ИПЛЕЕНИЦАДКСЕКИВЕЛТОН**
- Ключът **ВЕСЕЛИН** се редуцира **ВЕСЛИН** (без повтарящи се букви)

Последователност на четене на колоните с ключ **ВЕСЕЛИН/ ВЕСЛИН**

В	Е	С	Л	И	Н
1	2	6	4	3	5

Шифрираното съобщение в таблицата (втора стъпка)

В	Е	С	Л	И	Н
1	2	6	4	3	5
И	П	Л	Е	Е	Н
И	Ц	А	Д	К	С
Е	К	И	В	Е	Л
Т	О	Н			

- Шифрираното съобщение **ИИЕТПЦКОЕКЕЕДБХСЛЛАИН**

3.5. РОТОРНИ МАШИНИ

Описание

- Ще разглеждаме азбука от 26 букви
- Роторна машина е набор от L независимо въртящи се цилиндри, всеки от които има 26 входни щифта и 26 изходни щифта, с вътрешно окабеляване, което свързва всеки входен щифт към един уникален изходен щифт
- Всеки цилиндър изпълнява фиксирана пермутация между неговите 26 входни щифта и 26 изходни щифта
- 26-те завъртания на цилиндъра изпълняват 26 пермутации
- Комплектът L цилиндри реализира 26L пермутации
- Може да се установи връзка между входните и изходните щифтове на съседните цилиндри, както следва:
  - Изходните щифтове на (i-1)-вия цилиндър са входните щифтове на i-тия цилиндър;
  - Изходните щифтове на i-тия цилиндър са входните щифтове на (i+1)-вия цилиндър.
- Ключът на роторната машина се състои от относителното разположение на L-те цилиндри

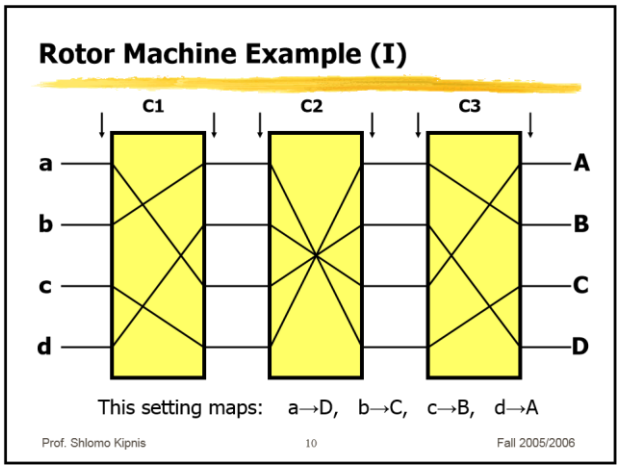
Характеристика

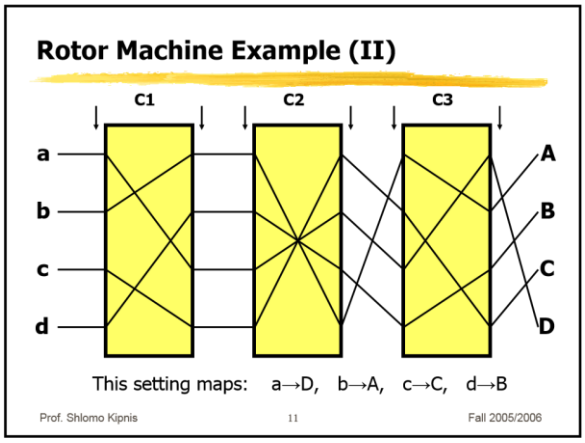
- Роторните машини съчетават принципите на методите на заместване и методи на разместване
- Роторните машини произвеждат шифри, които са много трудни да се разбиване
- Роторни машини през Втората световна война:
  - „Енигма“, използвана от немците (**Приложение 1.)**;

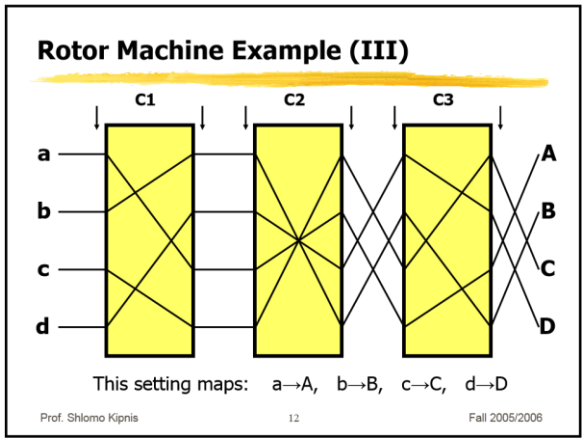
- „Пурпурен/Лилав“, използвана от японците.

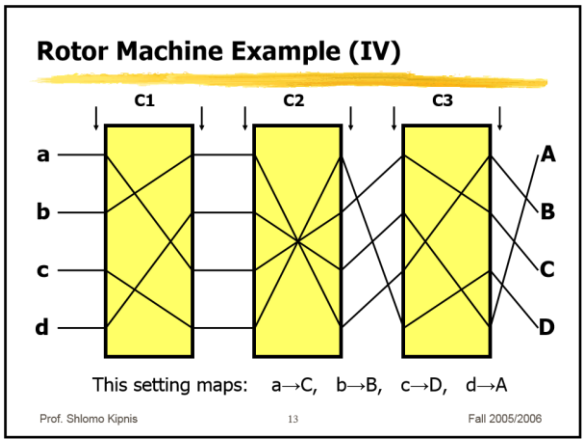
- Разбиването на двете роторни машини от съюзниците е било важен фактор за изхода на войната.

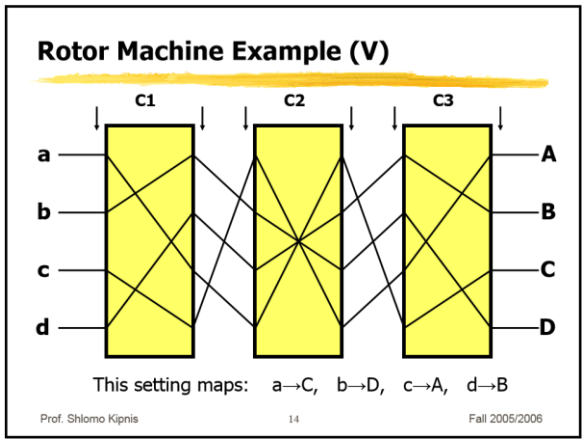
ПРИМЕРИ











ГЛАВА 4. СЪВРЕМЕННА КРИПТОГРАФИЯ

Основите на съвременната криптография са представени в Глава 4. Дефиниран е основният принцип на съвременната криптография ***„Моделът за сигурност не се базира на неизвестността на алгоритъма. Допуска се, че алгоритъмът е известен на противника. Сигурността зависи от сигурността на ключа“***. Представени са различни гледни точки за разделяне на криптографските алгоритми – безусловна и условна криптография, известни и тайни алгоритми, блокови и поточни шифри. Съвременните блокови шифри са представени от схемата на Фейстел и Data Encryption Standard (DES). Разгледани са развитието на DES и начините за криптиране на големи съобщения с DES. Описан е най-използваният блоков криптографски алгоритъм – Advanced Encryption Standard (AES). Криптографските алгоритми с публичен ключ са разгледани в историческа хронология, развитие и използване. Обхванати са – задачата за раницата, криптографското решение на Merkle–Hellman, криптосхемите RSA, Рабин, Ел–Гамал (*ElGamal*), Дифи–Хелман(*Diffie–Hellman*), алгоритъмът за цифров подпис (*Digital Signature Algorithm*, DSA) и схемата на Feige–Fiat–Shamir. Множество от криптографски системи и алгоритми върху елиптични криви са представени в края на главата.

4.1. СЪВРЕМЕННИ ШИФРИ. ОБЩИ ПОЛОЖЕНИЯ

Съвременните криптографски алгоритми (шифри) могат да бъдат класифицирани по различни критерии, като:

- Необходими условия за разбиване на шифъра;
- Поверителност на алгоритмите;
- Броят на едновременно трансформираните символи от откритото съобщение.

Безусловна и условна криптография

Два са възможните подходи за създаването и използването на модерните криптографски алгоритми. Най-общо те се базират на:

- Безусловна криптография;
- Условна криптография.

Безусловна криптография

Основните характеристики на безусловната криптография са:

- Криптографски схеми, които са 100% сигурни;
- Противникът не може да разбие схемата при условия, че:
  - неограничен текст е достъпен;
  - разполага с неограничено време;
  - разполага с неограничени компютърни ресурси;
- ***Сигурността на схемата се основава на математическо доказателство***, че противникът няма достатъчно информация за разбиване на схемата;
- Съществуват схеми за безусловна криптография;
- За съжаление схемите за безусловна криптография не са удобни за използване в практиката.

Условна криптография

Криптографските схеми удовлетворяващи изискванията на условната криптография включват:

- Криптографски схеми, за които не съществува математическо доказателство, че са 100% сигурни;
- Противникът може да разбие схемата при условия, че:
  - неограничен текст е достъпен;

    - разполага с неограничено време;
    - разполага с неограничени компютърни ресурси;
- ***Сигурността на схемата се основава на допускането, че противникът няма достатъчно ресурси*** (текст, време, компютри, пари и т.н.) за разбиване на схемата;
- Всички модерни криптографски схеми за условно сигурни.

Тайни и известни алгоритми

Криптографските алгоритми могат да се разделят на:

- Тайни алгоритми;
- Известни алгоритми.

Тайни алгоритми

Тайните алгоритми се характеризират с:

- Използват се в затворени системи (военни, разузнаване, служби за сигурност и др.);
- Анализирани са само вътре в системата;
- Трудно се опазват;
- В крайна сметка не са много сигурни (могат да бъдат разбити).

Известни алгоритми

Известните алгоритми се характеризират с:

- Имат търговска стойност;
- Силата на алгоритъма се базира на ключ:
    - По лесен за опазване;
    - Може да бъде динамично променян;
- Публично са анализирани.

Основен принцип

Основен принцип на модерната криптография (*Kerckhoff's Principle*) се формулира както следва: ***Моделът за сигурност не се базира на неизвестността на алгоритъма. Допуска се, че алгоритъмът е известен на противника. Сигурността зависи от сигурността на ключа.***

Избор на ключ

За осигуряване на сигурността на ключа е необходимо изпълнение на условията:

- Ключът трябва да се избира от голямо множество възможни ключове:
  - Да се намали вероятността от разгадаване (откриване) на секретния ключ;
  - Да се увеличи необходимото за проверка време, ако атакуващия опитва всички възможни ключове.
- Дължина на ключа:
  - 40 bits (2^{40} възможности за ключ) използва се в 1980-те години в Интернет приложения;
  - 56 bits (2^{56} възможности за ключ) използва се в DES; добър през 1980-те, но не е сигурен днес;
  - 64 bits (2^{64} възможности за ключ) използват се днес;
- 128 bits (2^{128} възможности за ключ) се разглежда като най-малката дължина на ключа (в битове) за използване в модерните алгоритми днес.

Изисквания към криптографския алгоритъм

Изискванията към криптографския алгоритъм могат да бъдат разглеждани от две гледни точки – от страна на легитимния потребител и от страна на не легитимния потребител. Могат да се обобщят като:

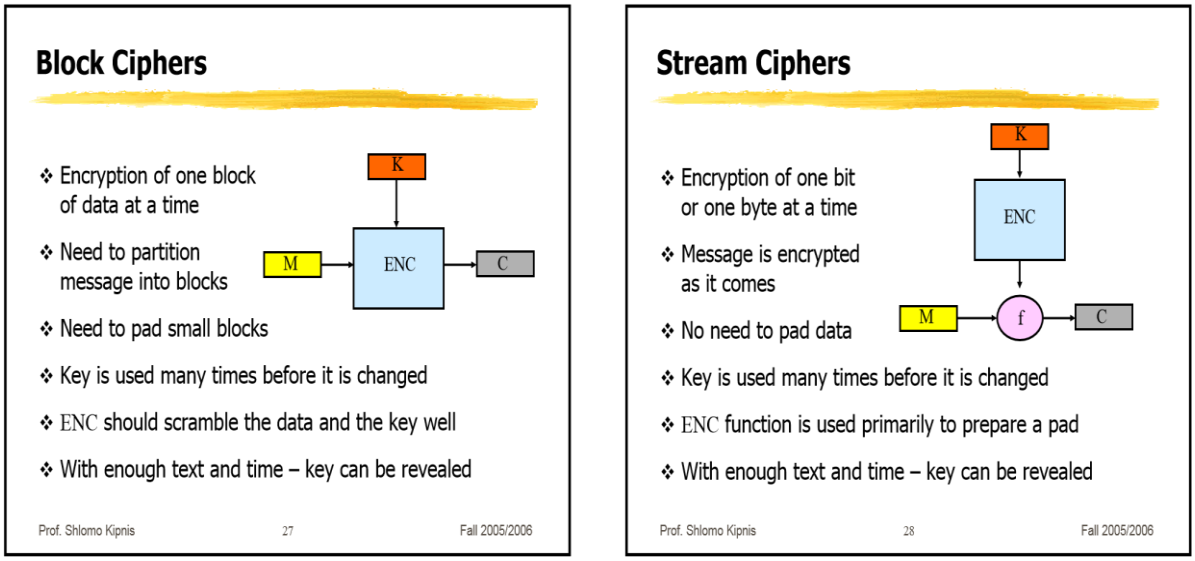
- За легитимния потребител:
  - Лесно криптиране/декриптиране при наличие (притежание на ключа).
- За не легитимния потребител (атакуващия):
  - Трудно криптиране/декриптиране при не наличие (без притежание на ключа);
  - Трудно е да се възстанови ключа;
  - Трудно е да се получи каквато и да е информация от открития текст;
  - Всичко това и при условие, че е налично голямо количество шифриран текст.
- Алгоритъмът не е таен.

Блокови и поточни алгоритми

Има два основни начини за криптиране на открито съобщение:

- ***Блокови алгоритми*** – откритото съобщение се разделя на части (блокове) с фиксирана дължина (обикновено 64/128 бита) и всеки блок се криптира;
- ***Поточни алгоритми*** – откритото съобщение се криптира на малки групи символи (1 или 8 бита), последователно, с контрол на стъпките на криптиране.

Приложението на блоковите и поточните алгоритми (***Фигура 4.1.***) е както следва:



Фигура 4.1. Блокови и поточни алгоритми

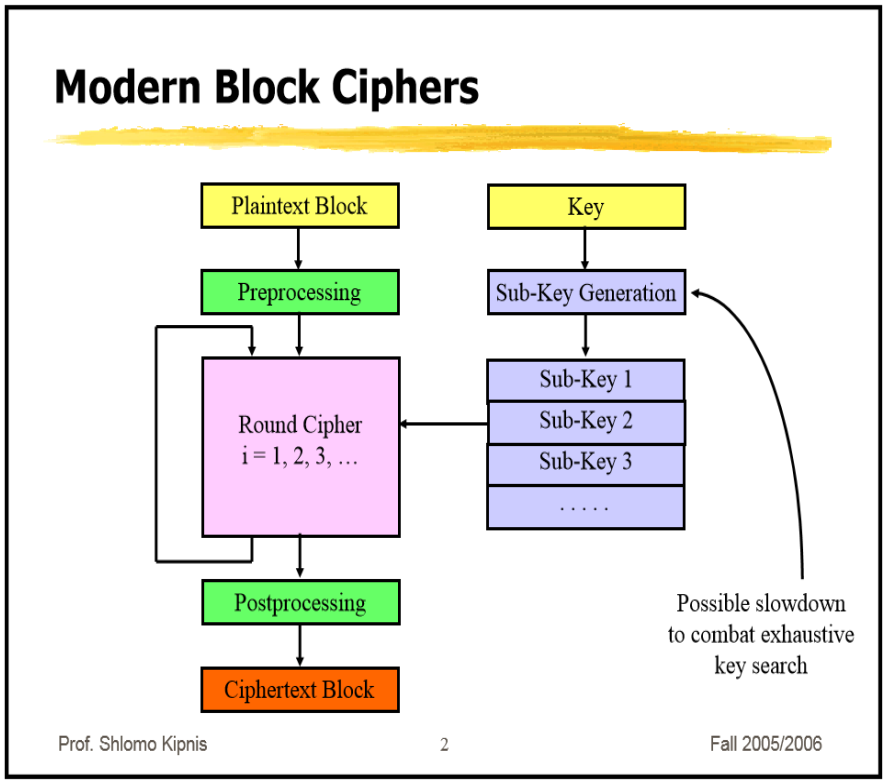
4.2. СЪВРЕМЕННИ БЛОКОВИ ШИФРИ. СХЕМА НА ФЕЙСТЕЛ

В съвременните блокови шифри откритото съобщение се разделя на части (блокове) с фиксирана дължина и всеки блок се криптира. Получените блокове се наричат входни и служат за вход на алгоритъма. Ако дължината на входния блок е по-малък от размера на този, който е избран за алгоритъма то той се удължава по някакъв начин. Обикновено дължината на блока е степен на 2, например, 64 бита или 128 бита.

Основна схема

Основната схема на съвременните блокови шифри (***Фигура 4.2.***) се базира на следните стъпки:

- Входящ блок (не криптиран);
- Предварителна обработка на входящия блок;
- Циклична обработка:
  - Фиксиран брой цикли;
  - Входящ блок:
    - За първия цикъл входящ блок е резултатът от предварителната обработка;
    - За всеки следващ – резултатът от предходния цикъл;
  - Изходящ блок – резултатът от последния цикъл;
  - За всеки цикъл – различен ключ (резултат от генератор на ключове);
- Последваща обработка на изходящия блок на цикличната обработка;
- Изходящ блок – шифриран.



Фигура 4.2. Основна схема на блоков шифър

СХЕМА НА ФЕЙСТЕЛ

Схемата на Фейстел (***Фигура 4.3.)*** е в основата на множество блокови шифри.

Алгоритъм за шифриране

Ще представим действието на алгоритъма само върху един блок M, за повече блокове – аналогично.

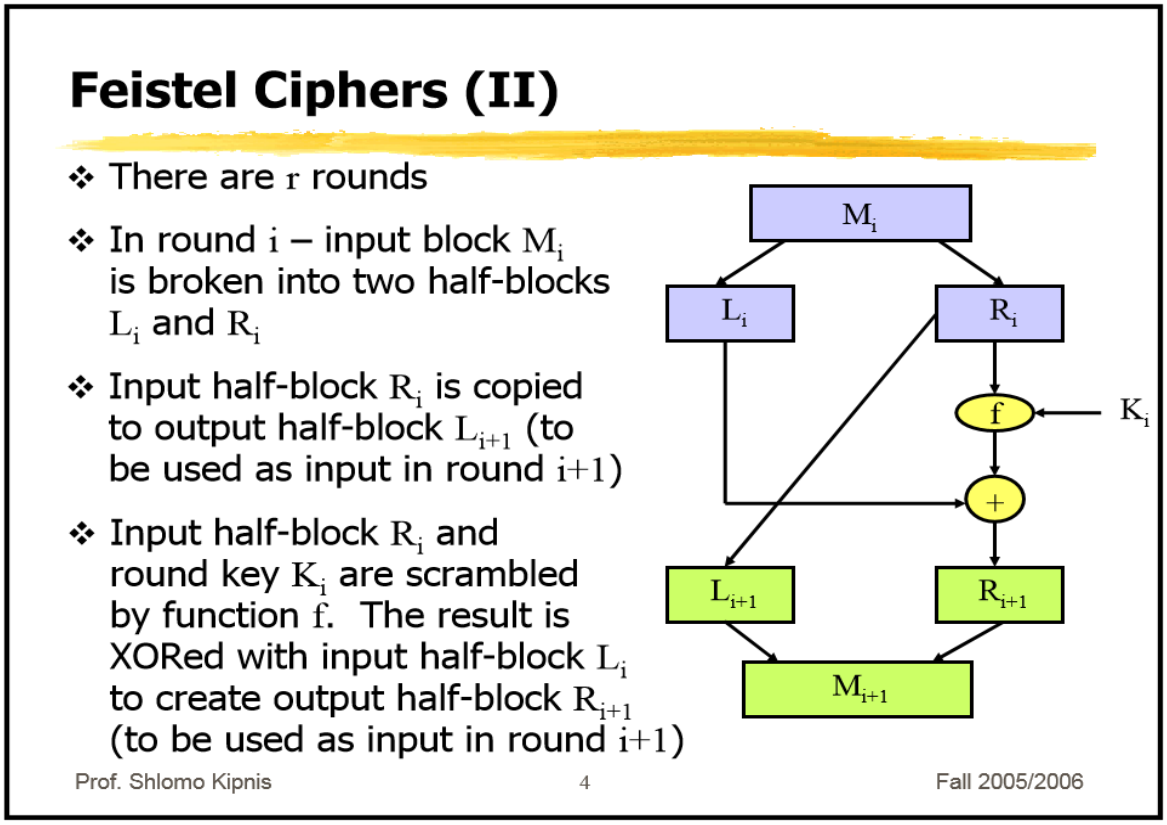
Начало:

- Входящият блок M разделя на две части:
  - Лява – L_0;
  - Дясна – R_0;
  - $M = (L_0, R_0)$;
- Циклична обработка:
  - $M_0 = M$;

- N – брой на стъпките;
- За i = 0, … , N – 1, стъпката на цикъла е както следва:
  - $L_{i+1} = R_i$;
  - $R_{i+1} = L_i$ XOR $f(R_i, K_i)$;
  - $M_{i+1} = (L_{i+1}, R_{i+1})$;

- C = (RN, LN).

Край.

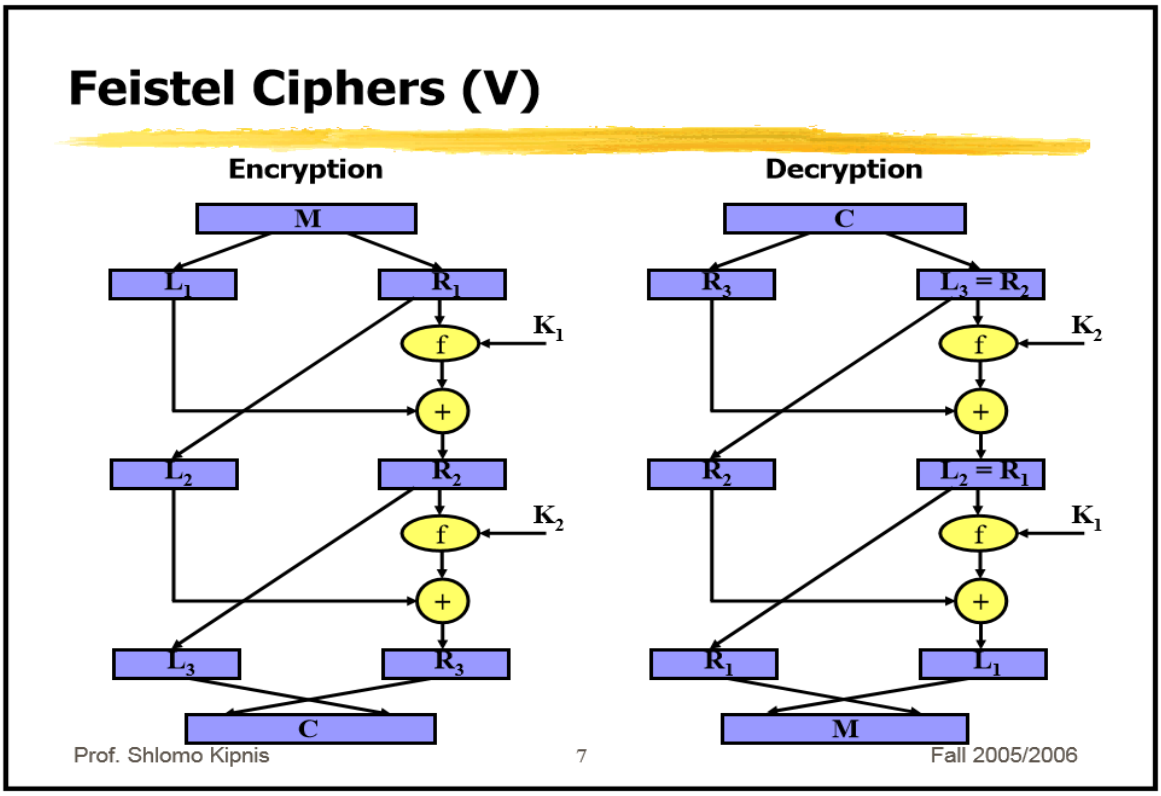


Фигура 4.3. Схема на Фейстел

Алгоритъм за дешифриране

За дешифриране се прилага същият алгоритъм, като ключовата поредица се използва в обратен ред:

- ако $K_1, K_2, \dots, K_N$ е ключовата поредица за шифриране;
- тогава $K_N, K_{N-1}, \dots, K_1$ е ключовата поредица за шифриране;



Фигура 4.4. Криптиране и декриптиране по схемата на Фейстел

Особености в схемата на Фейстел

Схемата на Фейстел не специфицира:

- Размера на блока;
- Дължината на ключа;
- Брой на циклите;
- Генераторът за генериране на ключовата поредица;
- Функцията f.

4.3. СЪВРЕМЕННИ БЛОКОВИ ШИФРИ. DATA ENCRYPTION STANDARD (DES)

История на DES

Основните моменти от историята на DES са:

- През 1973 г. Националното бюро за стандарти (*National Bureau of Standards, NBS*) излиза с Искане за предложения (*Request for Proposals, RFP*) за търговски стандарт за криптиране;
- IBM предлага своя силен алгоритъм на Луцифер (разработен от *Feistel* и други);
- Агенцията за национална сигурност (*National Security Agency, NSA*) изисква да се намали силата на (да отслаби силата на) Луцифер (чрез съкращаване на ключа);
- NSA също прави промени в алгоритъма на Луцифер на IBM;
- Стандартът за шифриране на данни (*Data Encryption Standard, DES*) е приет през 1976 г.

Критерии за проектиране на криптографския алгоритъм

NBS поставя следните критерии за проектиране:

- Алгоритъмът трябва да осигури високо ниво на сигурност;
- Алгоритъмът трябва да бъде напълно определен;
- Сигурността на алгоритъма трябва да бъде в сигурността на ключа;
- Алгоритъмът трябва да бъде достъпен за всички потребители;
- Алгоритъмът трябва да се приспособява за използване в различни приложения;
- Алгоритъмът трябва да осигурява възможност за ефективно хардуерно изпълнение;
- Алгоритъмът трябва да е ефективно използваем;

- Алгоритъмът трябва да може да бъде валидиран;
- Алгоритъмът трябва да бъде с опциите за експорт.

Описание на DES

В алгоритъма на DES се специфицират:

- Дължина на блока – 64 бита;
- Дължина на ключа – 56 бита (от 64 битов буфер, т.е. само 56 се използват в даден момент);
- Фиксирана първоначална пермутация на входящия буфер – 64 бита;
- 16 циклични ключа (48 бита) получени от основния 56 битов ключ;
- Схема за генерирана на цикличните ключове;
- 16 цикъла, всеки от които се състои от разбъркването (функция f) на 64 битов блок с цикличния ключ (48 – бита);
- Функция f (ще бъде описана по-късно);
- Фиксирана крайна пермутация, обратна на първоначалната.

Алгоритъм на DES

Вход – Открит текст, блок M (64 бита)

Стъпка 1. Инициализираща пермутация (*Init-Perm*)

Стъпка 2. Цикъл. За всяка стъпка от цикъла:

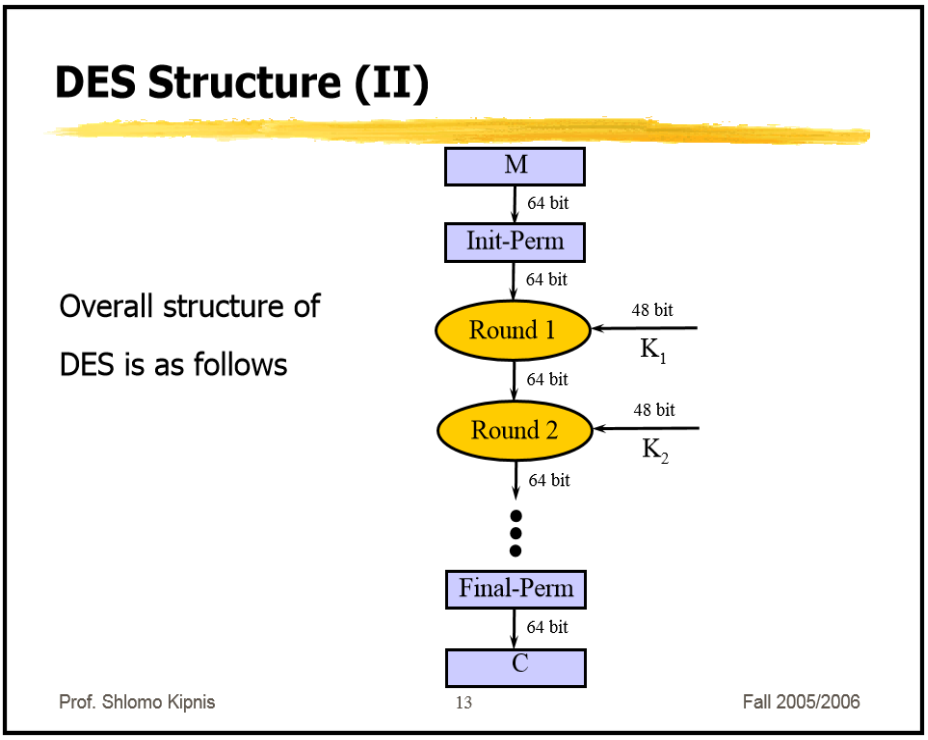
- Входящ блок – 64 бита;
- Изходящ блок – 64 бита;
- Ключ – 48 бита.

Стъпка 3. Заключителна пермутация (*Final-Perm*)

Изход – Шифриран текст, блок C(64 бита)

Структура на DES

Структурата на DES е показана на ***Фигура 4.5.***



Фигура 4.5. Структура на алгоритъм DES

Стъпка 1

Инициализираща пермутация (входящ блок – 64 бита, изходящ блок – 64 бита):

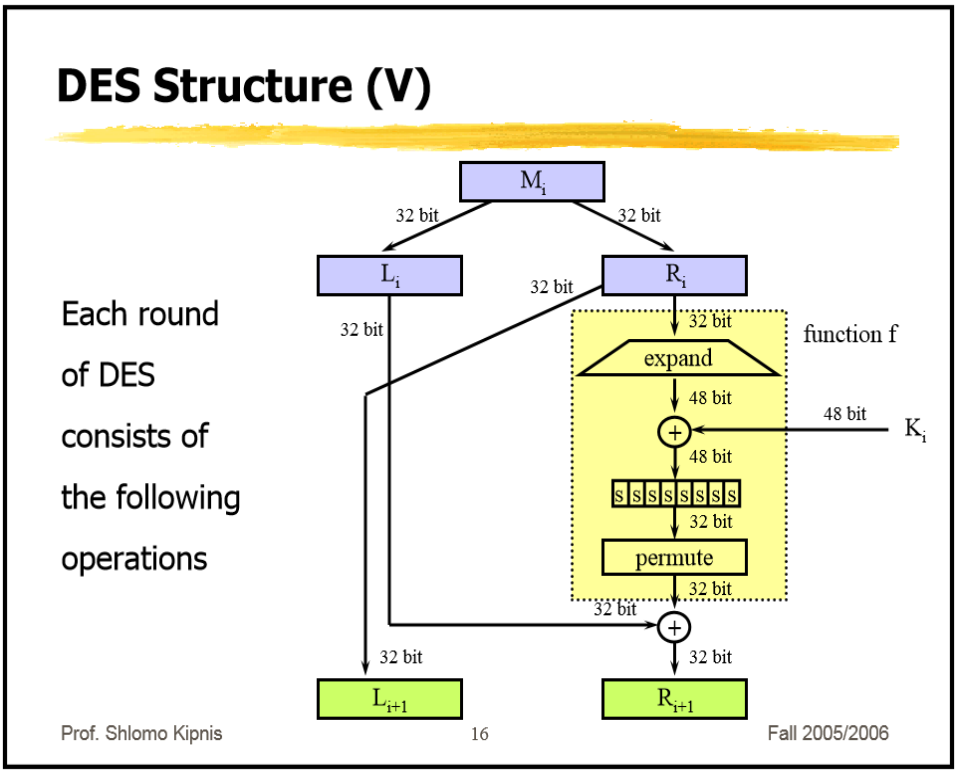
58	**50**	**42**	**34**	**26**	**18**	**10**	**2**
60	**52**	**44**	**36**	**28**	**20**	**12**	**4**
62	**54**	**46**	**38**	**30**	**22**	**14**	**6**
64	**56**	**48**	**40**	**32**	**24**	**16**	**8**
57	**49**	**41**	**33**	**25**	**17**	**9**	**1**
59	**51**	**43**	**35**	**27**	**19**	**11**	**3**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

Фигура 4.6. Init-Perm

Стъпка 2

За всяка стъпка (i = 1, 2, …, 16) от цикъла се изпълняват действията (за i+1 -та стъпка) (***Фигура 4.7.***):

- Входящият блок M_i (64 бита) се разделя на два полублока, ляв L_i (32 бита) и десен блок R_i (32 бита);
- Десният полублок R_i се копира директно в левия полублок L_{i+1};
- Десният полублок R_i се разширява до 48 бита (Expand Function) и се сумира (XOR) със стъпков Ключ K_i (48 бита);
- S-блок (осем S-кутии (S-Boxes) – всяка обработва по 6 бита (от получените 48 бита по горе) и генерира 4 бита (общ резултат 32 бита);
- Резултатът от 32 бита са подлага на Вътрешна пермутация (Internal Permute**)** и се сумира (XOR) с левия полублок L_i за получаването на десния полублок R_{i+1}.



Фигура 4.7. Един цикъл на DES

Разширяване на 32 бита до 48 бита (Expand Function – входни 32 бита, изходни 48 бита)

Разширяването на входните 32 бита до 48 бита (*Expand Function*) до изходните 48 бита се осъществява по правилото:

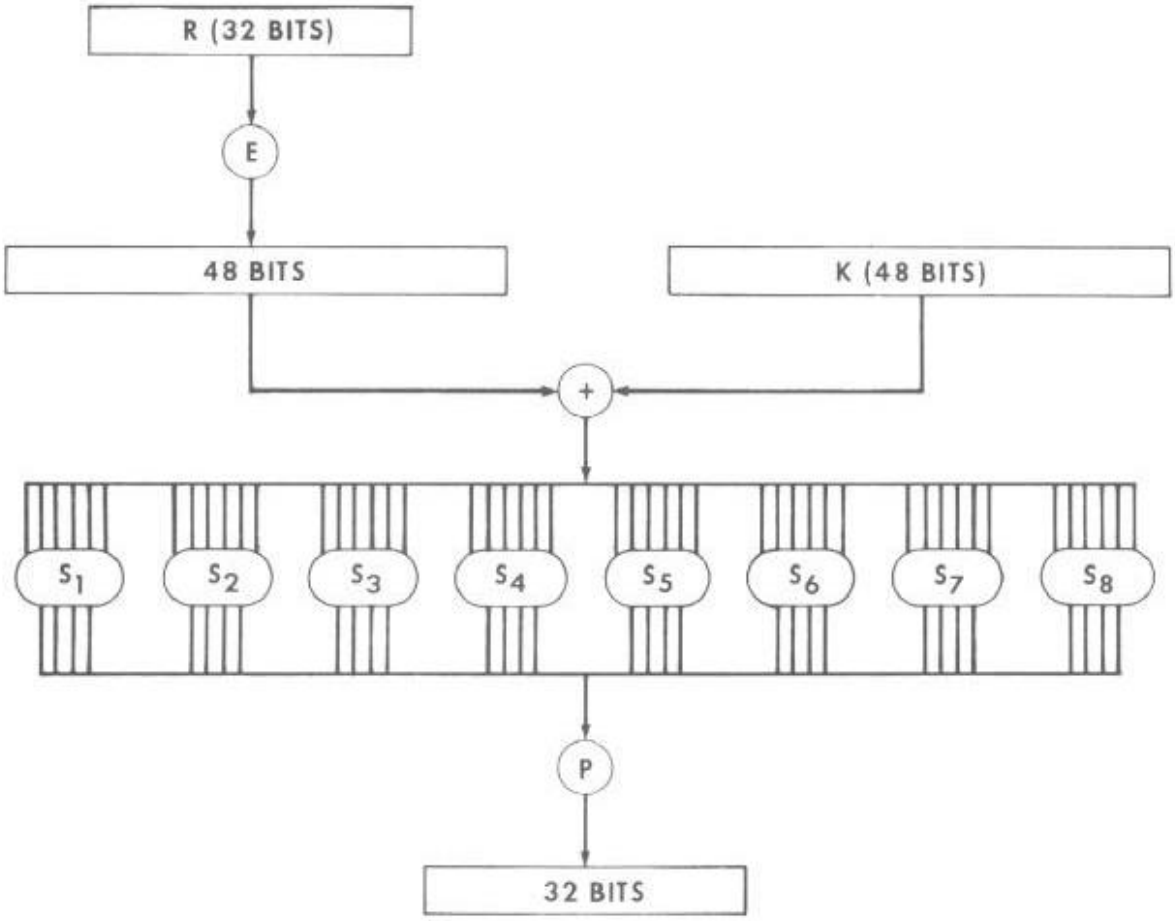
32	**1**	**2**	**3**	**4**	**5**
4	**5**	**6**	**7**	**8**	**9**
8	**9**	**10**	**11**	**12**	**13**
12	**13**	**14**	**15**	**16**	**17**
16	**17**	**18**	**19**	**20**	**21**
20	**21**	**22**	**23**	**24**	**25**
24	**25**	**26**	**27**	**28**	**29**
28	**29**	**30**	**31**	**32**	**1**

Фигура 4.8. Expand Function

Функцията f от схемата на Фейстел

Функцията f от схемата на Фейстел се състои от следните действия:

- Разширение (E);
- XOR;
- S-блок;
- Пермутация (P).



Фигура 4.9. Функцията f от схемата на Фейстел

S-блок

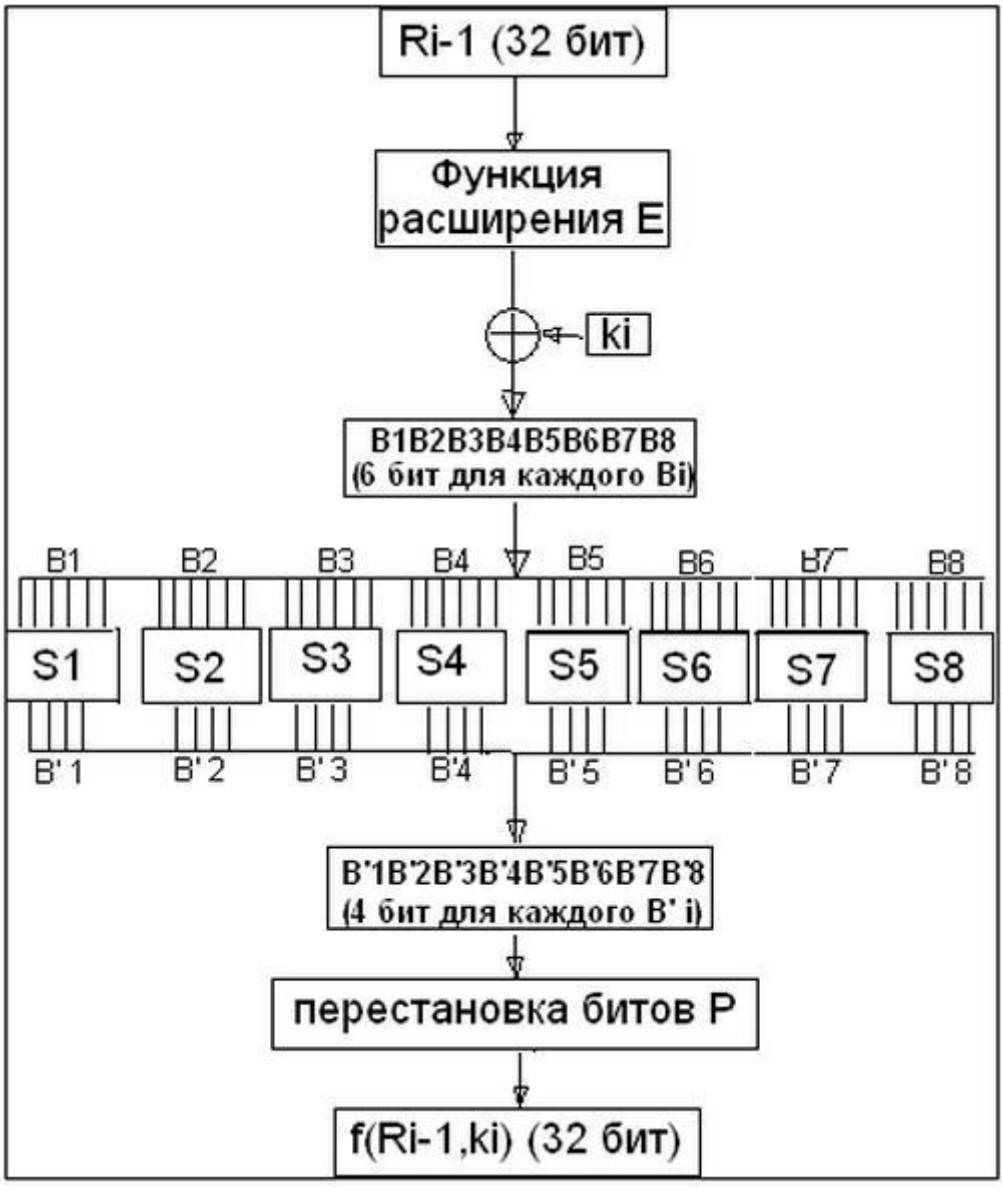
Ред	Стълб																S_i
	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0	
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S_2
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	S_3
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S_4
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S_5
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3	
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S_6
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6	
3	4	3	2	12	9	5	15	0	11	14	1	7	6	0	8	13	
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	S_7
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S_8
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

Фигура 4.10. S-блок

S-кутия (S-box, 6-входни и 4 изходни бита)

Структура

- Правоъгълна таблица от числа от 0 до 15 (4 бита)
- Четири реда – от 0 до 3 (2 бита)
- Шестнадесет стълба – от 0 до 15 (4 бита)



Фигура 4.11. Функциониране на S-блок

Функциониране

- Първи и шести входен бит определят ред (между 0 и 3 в таблицата **S-Box**)
- Втори, трети, четвърти и пети бит определят колона (между 0 и 15 в таблицата **S-Box)**
- Стойността на всеки от четирите изходни бита е между 0 и 15 и се определя от стойността на таблицата **S-Box** с определените вече ред и стълб

Вътрешна пермутация (Internal Permute – входни 32 бита, изходни 32 бита)

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**

Фигура 4.12. Internal Permute

Стъпка 3

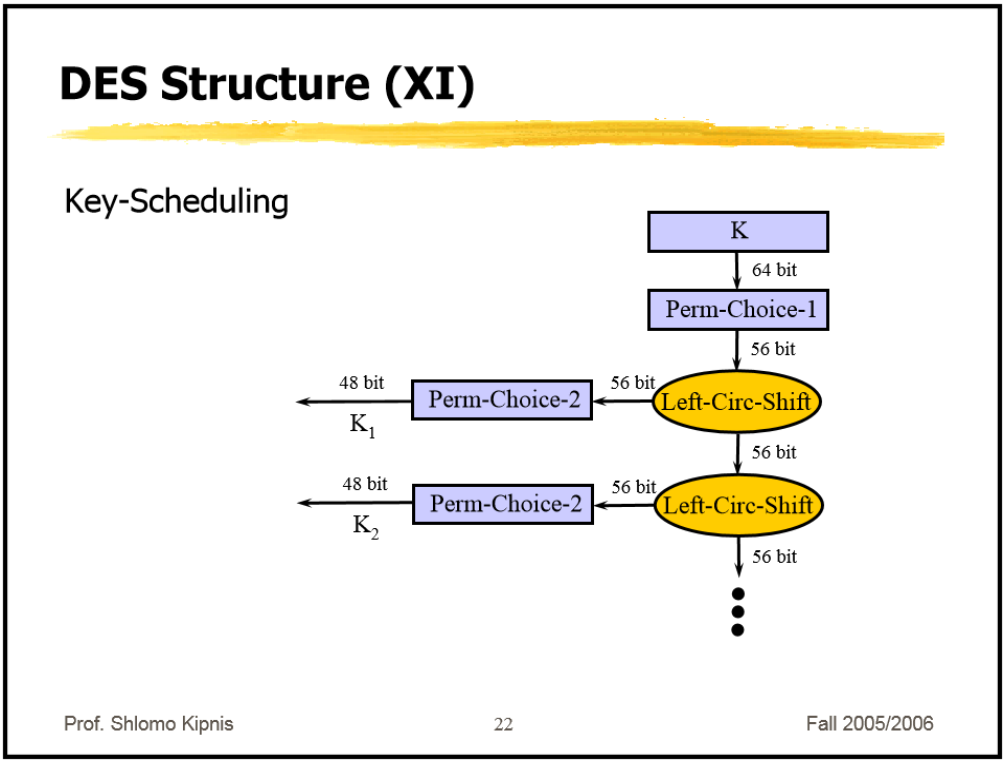
Заключителна пермутация (входящ блок – 64 бита, изходящ блок – 64 бита)

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

Фигура 4.13. Final-Perm

Генериране на ключовата поредица на DES

Алгоритъмът за генериране на ключовата поредица на DES е както следва:



Фигура 4.14. Алгоритъм за генериране на ключовата поредица на DES

Вход – блок K (64 бита)

Стъпка 1.

- Пермутация за избор 1(*Perm-Choice-1*)
- Изход блок 56 бита

Стъпка 2. – Цикъл за генериране на ключовата поредица

За всяка стъпка от цикъла:

- Входящ блок – 56 бита;
- Циклично изместване на ляво (*Left-Circ-Shift*);*
- Пермутация за избор 2 (*Perm-Choice-2*);
- Изходящ блок – Ключ – 48 бита.

Край.

Пермутация за избор 1 (*Perm-Choice-1)*

- Правоъгълна таблица от числа съдържаща 56 числа от 1 до 64
- Осем реда
- Седем стълба

Вход: Последователност от 64 бита

Изход: Последователност от 56 бита, определени в съответствие с Пермутация за избор 1.

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**

Фигура 4.15. Perm-Choice-1

Пермутация за избор 2 (*Perm-Choice-2*)

- Правоъгълна таблица от числа съдържаща 48 числа от 1 до 56
- Шест реда
- Осем стълба

Вход: Последователност от 56 бита

Изход: Последователност от 48 бита, определени в съответствие с Пермутация за избор 2.

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**

Фигура 4.16. Perm-Choice-2

Таблица на отместванията при Циклично изместване на ляво (*)

Цикъл:	1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
Отместване:	1 1 2 2 2 2 2 2 1 2 2 2 2 2 1

4.4. РАЗВИТИЕ НА DES И КРИПТИРАНЕ НА ГОЛЕМИ СЪОБЩЕНИЯ

От началото на 90-те – DES се счита за не достатъчно сигурен за техническа и търговска употреба.

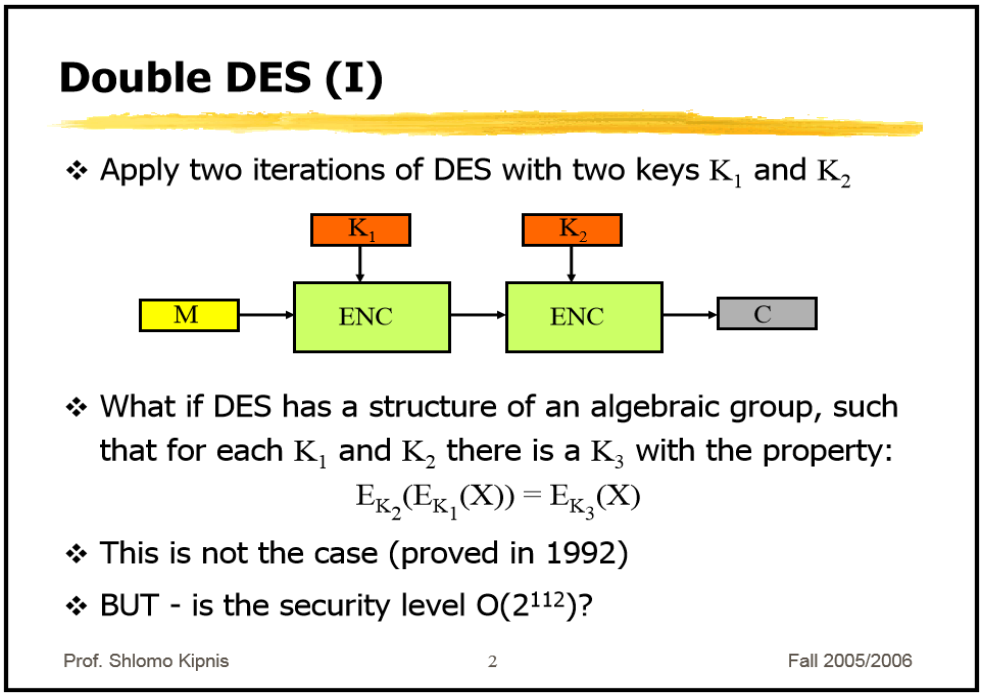
4.4.1. Развитие на DES

За подобряване на сигурността са възможни няколко подхода:

- Усилване на DES – Двоен DES (2-DES);
- Усилване на DES – Троен DES (3-DES);
- Усилване на DES – DES-X;
- Други алгоритми.

Двоен DES (2-DES)

Двойният DES се състои в прилагането на последователно на две криптирания с два различни ключа K_1 и K_2, както е показано:



Фигура 4.17. Алгоритъм на двоен DES

Троен DES (3-DES)

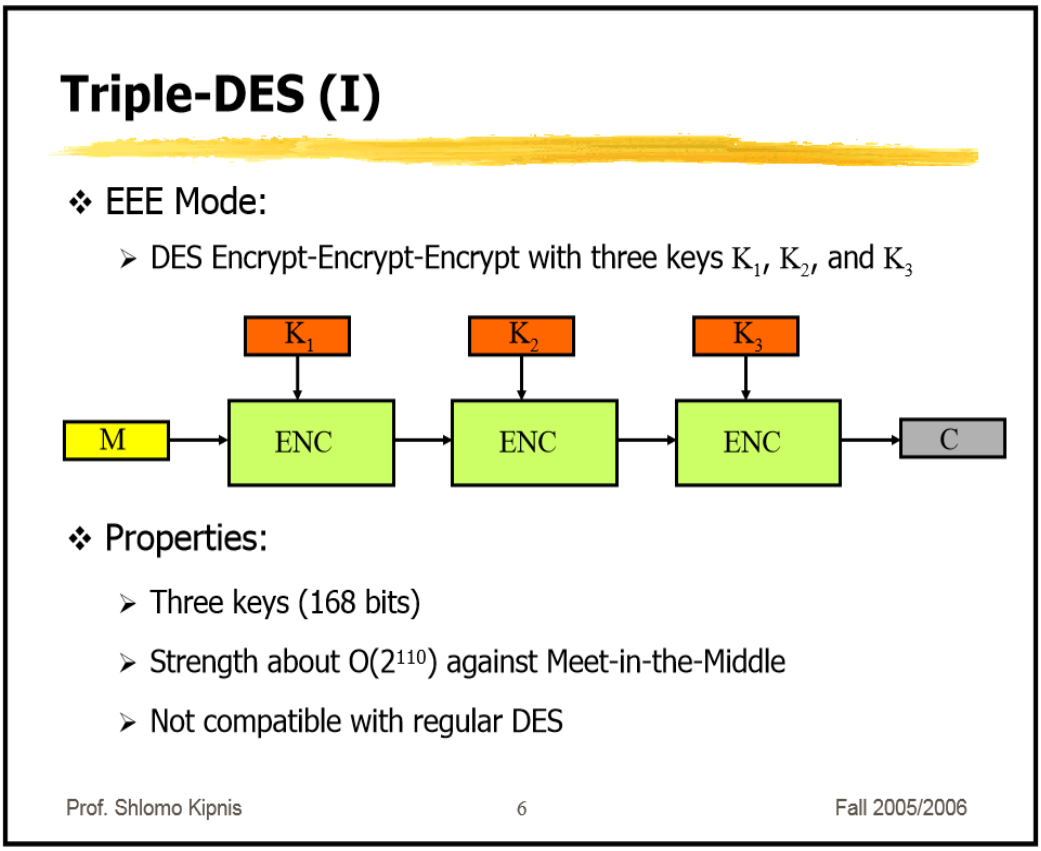
Тройният DES се състои в прилагането на последователно на две три криптирания/ декриптирания с два или три различни ключа K_1, K_2, K_3.

Съществуват режимите:

- EEE – три последователни криптирания;
- EDE – последователно криптиране, декриптиране, криптиране.

Режим EEE

Режим EEE е с три последователни криптирания с три различни ключа K_1, K_2, K_3.

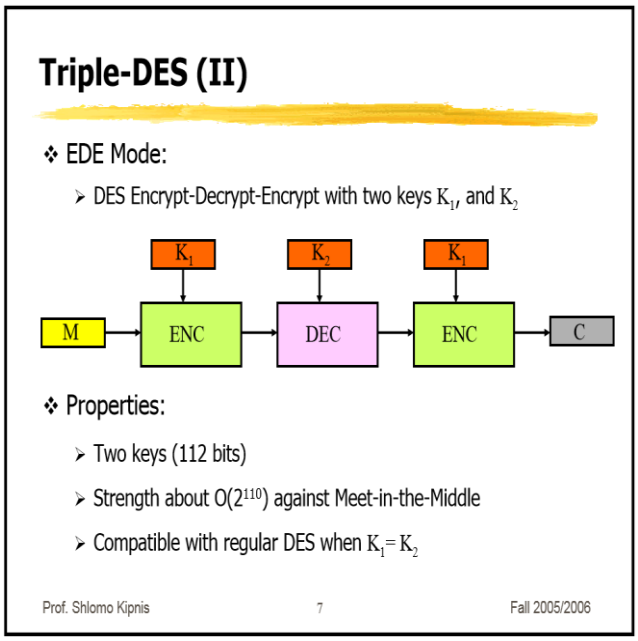


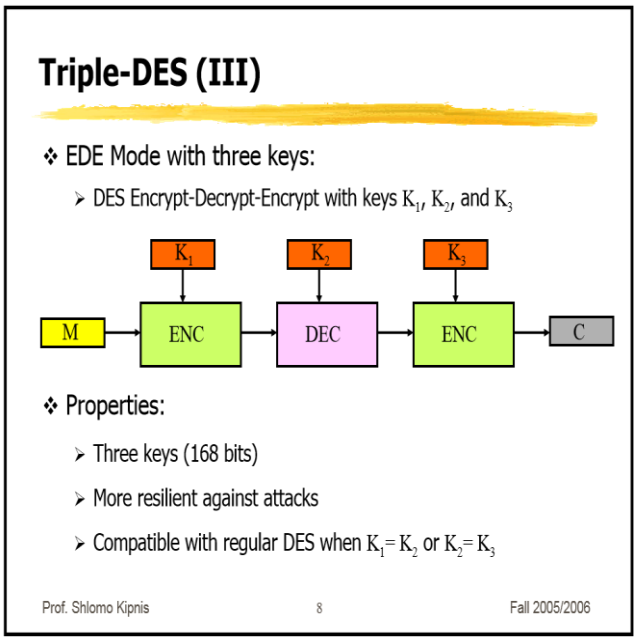
Фигура 4.18. Режим EEE с три различни ключа K_1, K_2 и K_3

Режим EDE – криптиране, декриптиране, криптиране

Възможни са вариантите с:

- Два ключа – K_1 и K_2;
- Три ключа – K_1, K_2 и K_3.



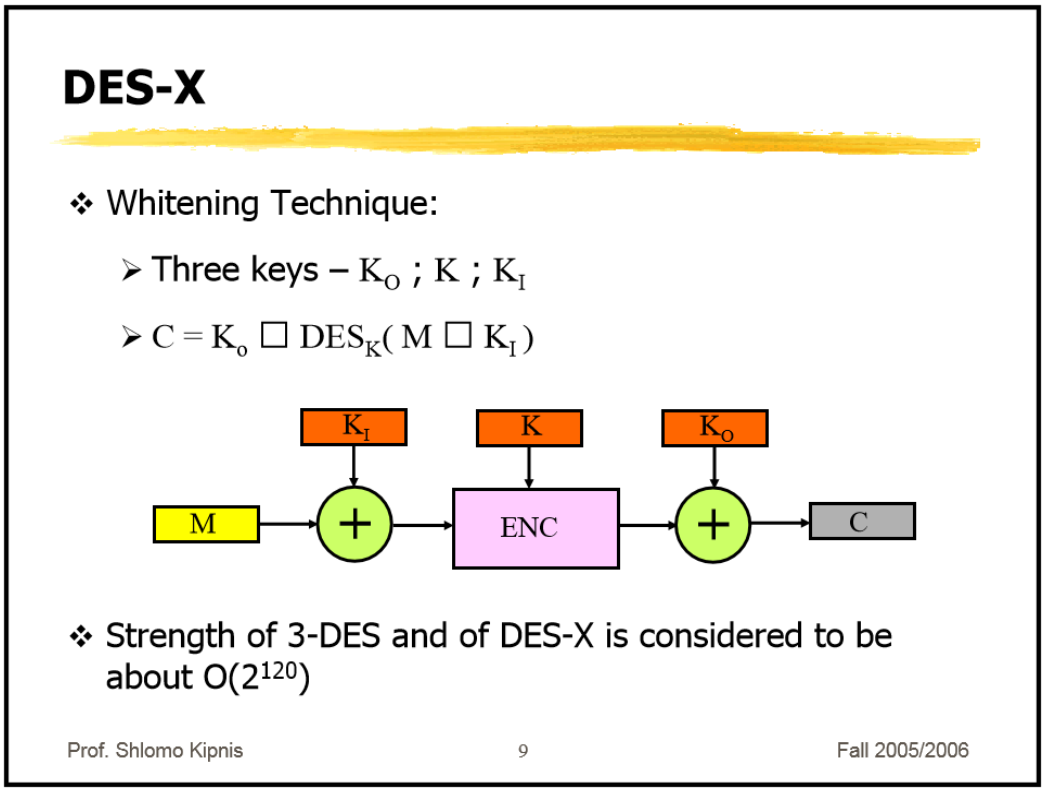


Фигура 4.19. Режим EDE с два или три ключа

DES-X

Описание:

- Режимът DES-X работи с три различни ключа K_0, K и K_1;
- Последователно се извършват действията XOR, криптиране и XOR;
- Използването на ключовете е както е показано:



Фигура 4.20. Криптиране с DES-X

Други алгоритми

Други популярни и използвани алгоритми са:

- IDEA (1990);
- RC5 (1990).

4.4.2. Криптиране на съобщение с голяма дължина с DES

Нека M е открито съобщение с дължина по-голяма от един блок на DES. Криптирането на M с DES се осъществява като откритото съобщение M се разделя на L на брой блока $M=M_1, M_2, \ldots, M_L$. Криптираното съобщение C се състои от L на брой криптирани блока $C = C_1, C_2, \ldots, C_L$. Получаването на криптираното съобщение C може да се осъществи в няколко режима, както следва:

- Режим електронна кодова книга (***ECB – Electronic Code Book**);*
- Режим на използване на брояч **(*CTR – Counter Mode*);**

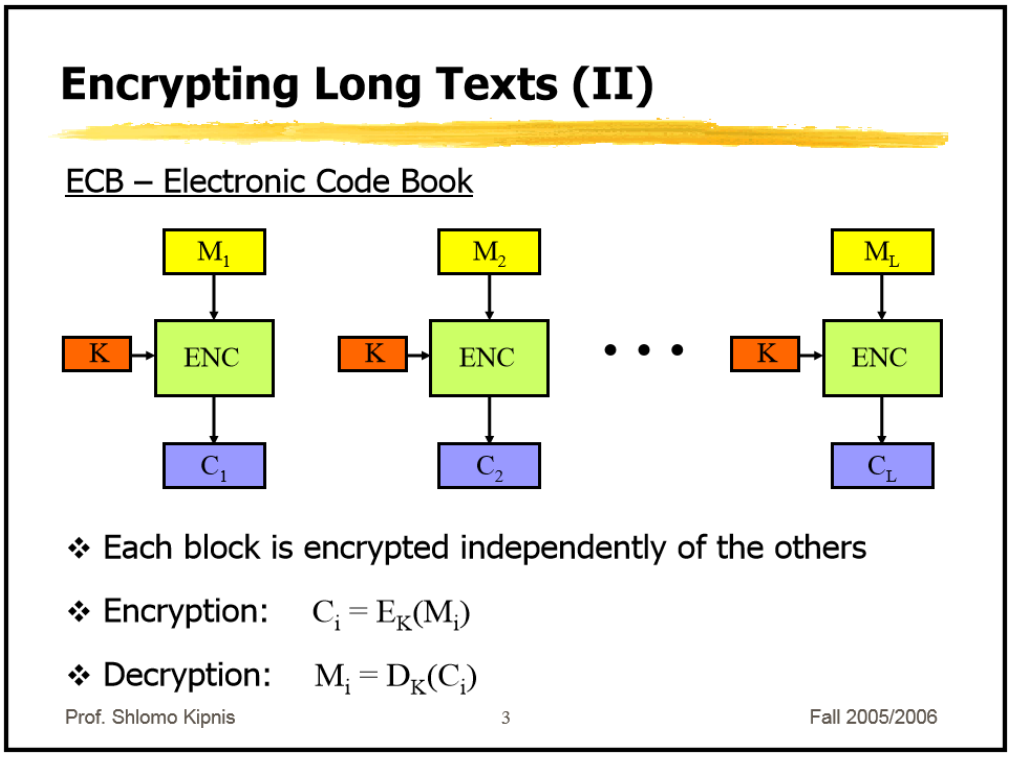
- Режим на свързване на блокове (***CBC – Cipher Block Chaining***);
- Режим на обратна връзка по изход (***OFB – Output Feed Back*);**
- Режим на обратна връзка по криптиран текст (***CFB – Cipher Feed Back***).

ECB – Electronic Code Book

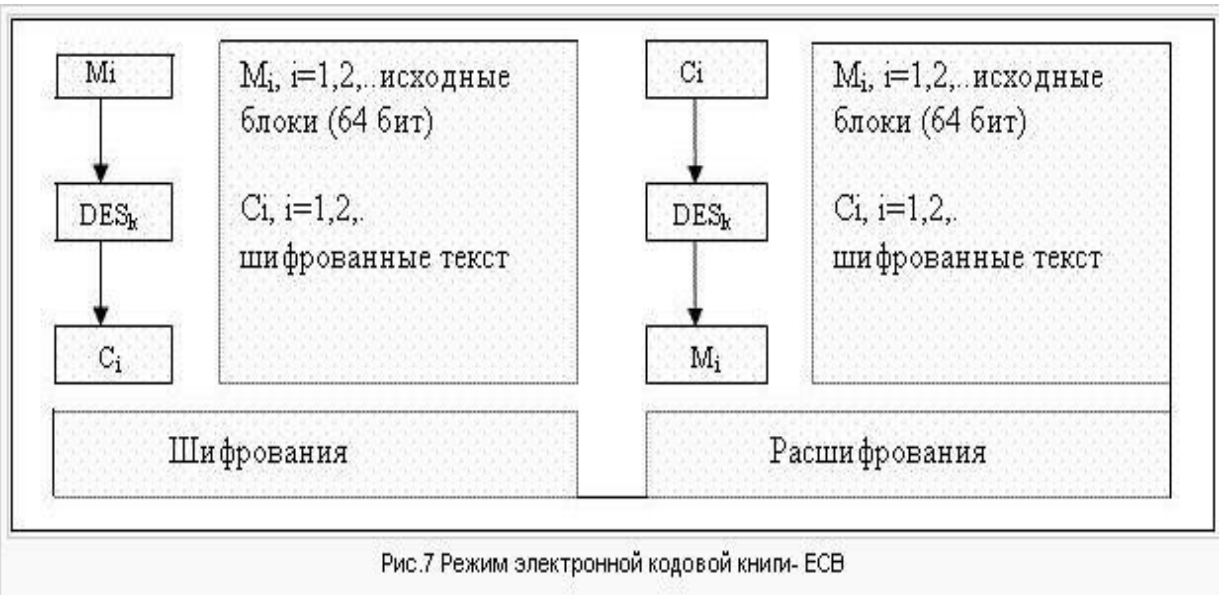
Режимът на електронна кодова книга (ECB) е обичайното използване на DES като блоков шифър, като всеки блок от входния текст се криптира сам за себе си.

Предимства: Този режим позволява избягването на последователност (свързаност) от грешки, ако е допусната такава в криптирането на един блок. Позволява паралелна реализация.

Слабости: Всеки блок (всяко съобщение), всеки път ще се криптира по един и същи начин, което е много сериозна слабост.



Фигура 4.21. Electronic Code Book



Фигура 4.21. Криптиране и декриптиране в режим ECB

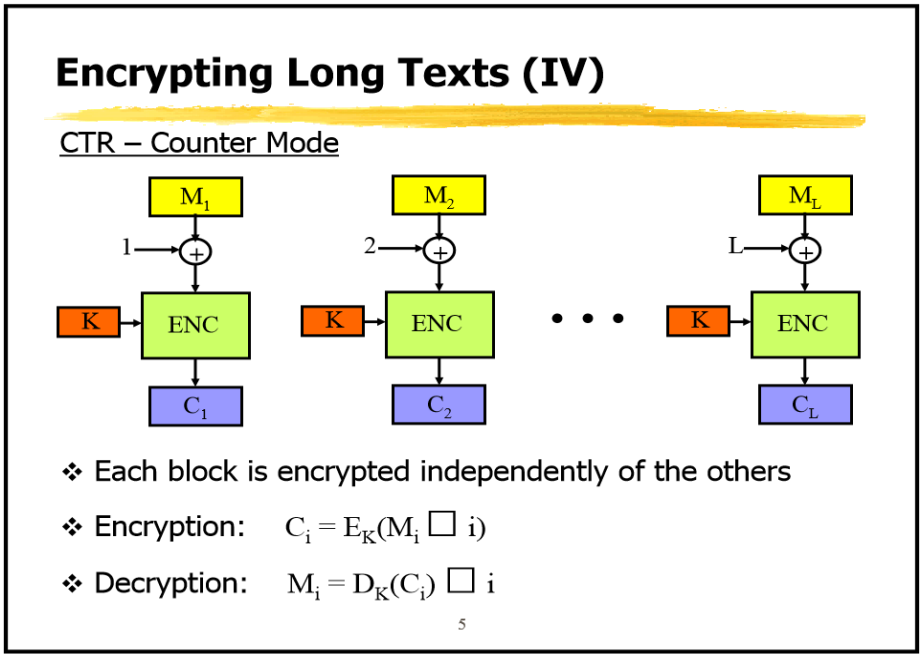
CTR – Counter Mode

Режимът използване на брояч прилича на режима на електронна кодова книга, като всеки блок от входния текст преди да се криптира се сумира (XOR) с поредния си номер (брояч).

Предимства: Този режим позволява избягването на последователност (свързаност) от грешки, ако е допусната такава в криптирането на един блок. Позволява паралелна реализация.

Слабости: Всеки блок (всяко съобщение), всеки път ще се криптира по един и същи начин, което е много сериозна слабост.

Забележка: Вместо пореден номер може да се използва поредица от битове без повторения на дълги групи в нея (може да бъде публична). Криптиращите блокове се генерират чрез тази редица и тайния ключ. Този метод е паралелен, сумирането може да се извърши предварително. Декриптирането може да се прави в произволен ред.



Фигура 4.22. Counter Mode

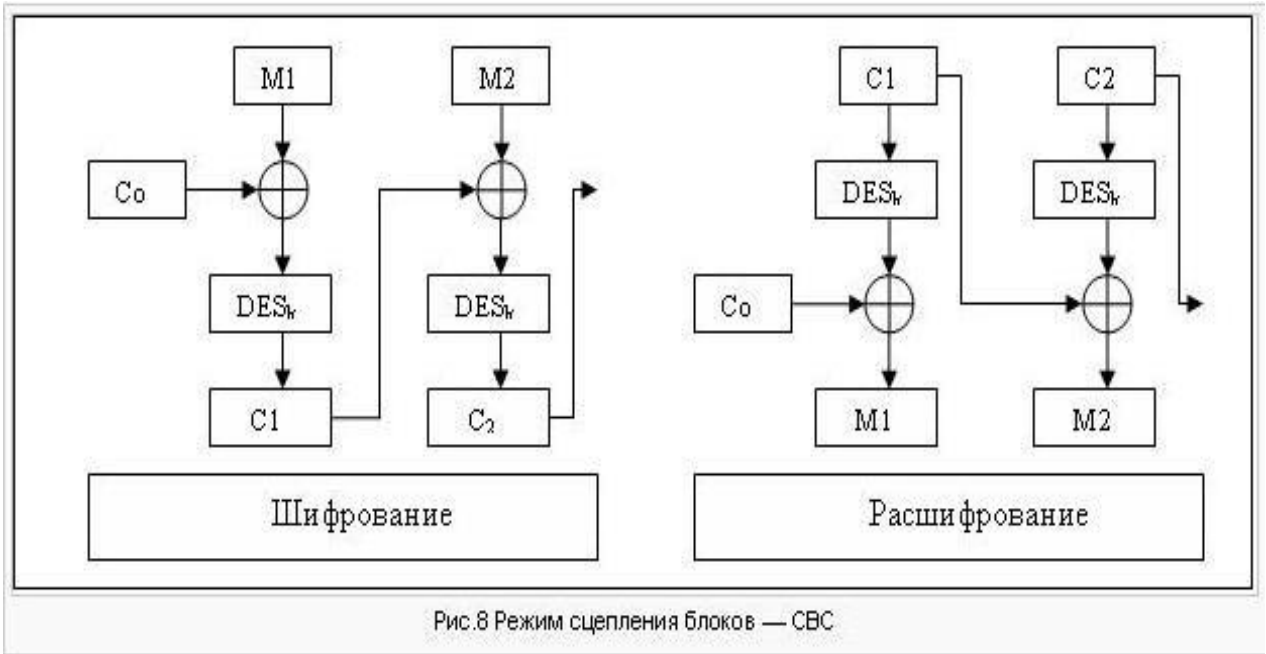
CBC – Cipher Block Chaining

Режимът на свързване на блоковете използва криптирането с DES, като:

- Всеки криптиран блок се сумира (XOR) със следващия открит блок;
- Нулевият криптиран блок C_0 (**инициализиращ вектор, *Initial Vector*, IV**) е случайно генериран и се променя за всяко съобщение.

Предимства: Всеки следващ открит блок се криптира в зависимост от предходния криптиран блок. Това позволява еднаквите блокове (съобщения) да се криптират по различен начин.

Слабости: Декриптирането на блок зависи от всички криптирани преди него блокове и съответно грешка или размяната на блокове води до грешно декриптиране. Методът е последователен, което не му позволява да се реализира като паралелно решение.



Фигура 4.23. Криптиране и декриптиране в режим CBC

OFB – Output Feed Back

Режимът на обратна връзка по изход използва криптирането с DES, като:

- Нулевият блок Z_0 (**инициализиращ вектор, Initial Vector (IV)**) е случайно генериран и се променя за всяко съобщение.
- Z_0 се използва за:
  - Криптиране, сумиране (XOR) с първия блок от откритото съобщение и получаване на първия блок на криптираното съобщение;
  - Начало на верига от блокове, като всеки блок се получава от предходния чрез криптиране, сумира се със съответния блок на откритото съобщение (XOR) и се получава съответния криптиран блок.

Предимства: Всеки следващ шифриран блок е резултат от сумирането (XOR) на съответния открит блок с предходния криптиран блок

от веригата с начало Z_0. Това позволява еднаквите блокове (съобщения) да се криптират по различен начин.

Слабости: Декриптирането на блок зависи от всички криптирани преди него блокове и съответно грешка или размяната на блокове води до грешно декриптиране. Методът е последователен, което не му позволява да се реализира като паралелно решение.

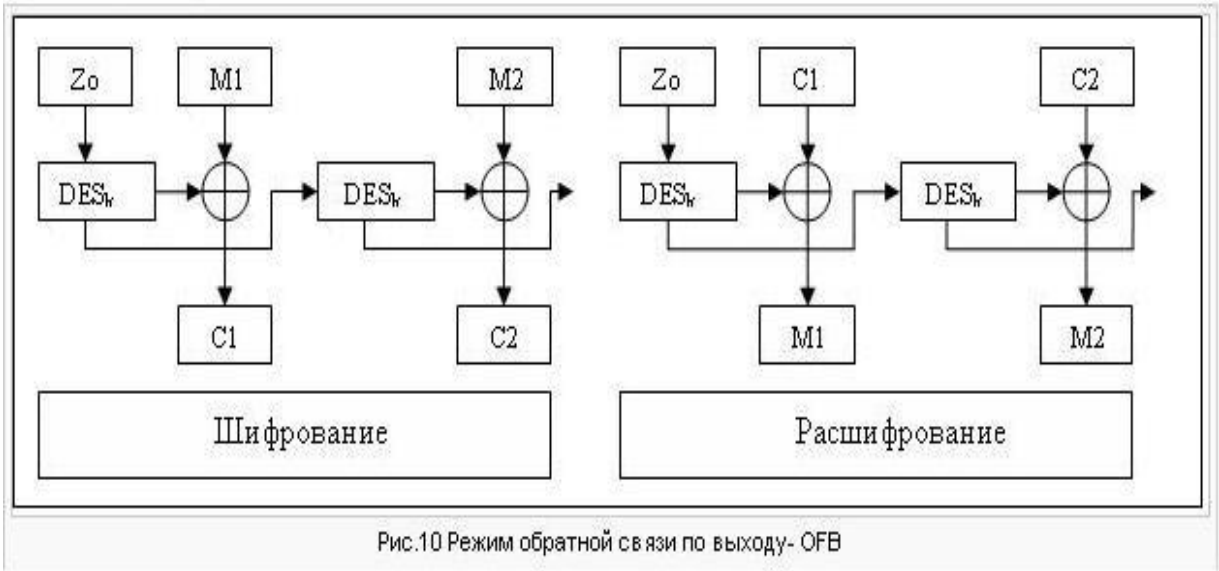


Рис.10 Режим обратной связи по выходу- OFB

Фигура 4.24. Криптиране и декриптиране в режим OFB

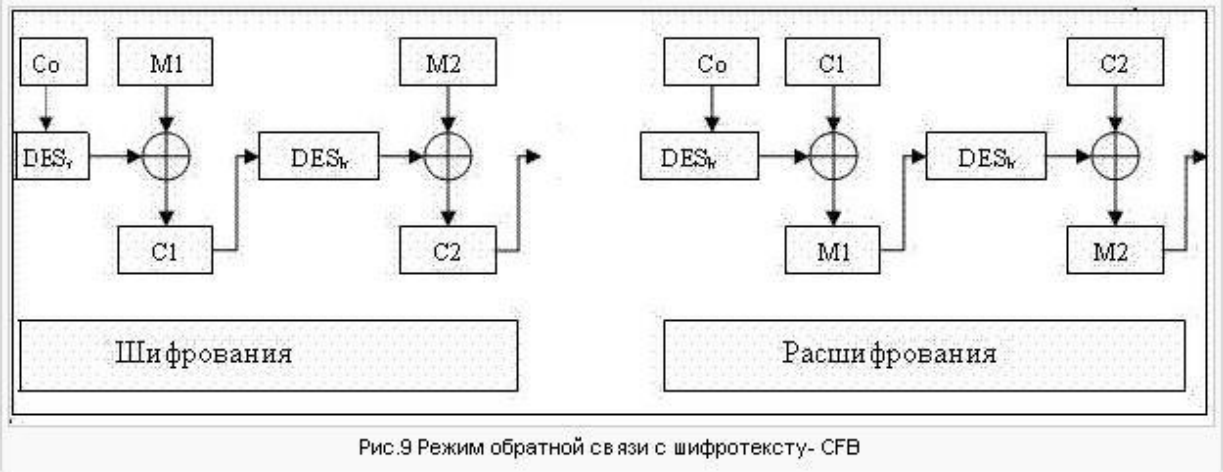
CFB – Cipher Feed Back

Режимът на обратна връзка по криптиран текст използва криптирането с DES, като:

- Всеки криптиран блок се криптира и след това се сумира (XOR) със следващия открит блок. Резултатът се криптира за получаване на съответния криптиран блок.;
- Нулевият блок C_0 (**инициализиращ вектор, *Initial Vector,* IV**) е случайно генериран и се променя за всяко съобщение. Той се криптира и се сумира (XOR) с първия блок от откритото съобщение.

Предимства: Всеки следващ открит блок се сумира (XOR) с предходния криптиран блок. Това позволява еднаквите блокове (съобщения) да се криптират по различен начин.

Слабости: Декриптирането на блок зависи от всички криптирани преди него блокове и съответно грешка или размяната на блокове води до грешно декриптиране. Методът е последователен, което не му позволява да се реализира като паралелно решение.



Фигура 4.25. Криптиране и декриптиране в режим CFB

Крипто анализ на DES

Без преувеличение може да се каже, че DES е предизвикало появата на една приложна наука, наречена „Криптографски анализ“. От самото начало на появата на DES се правят опити да се пробие (счупи) алгоритъма и е извършена много по обем научно-изследователска работа по неговия анализ. Всичко това е довело до появата на такива области на математиката като:

- Линеен крипто анализ – изследване и идентифициране на връзката между открит текст и криптиран текст;

- Диференциален крипто анализ – изследване и анализ на зависимости между няколко отворени текстове и техните криптирани версии;
- Крипто анализ по свързани ключове – изследване на зависимостите между криптирани текстове, получени на първичен (основен) ключ и ключовете, свързани по някакъв начин с основния.

Първите резултати

DES издържа 20 години в световен мащаб на крипто анализ и атаки, но остава силен шифър. Първите резултати:

- Бихам и Шамир, учени от Израел, показаха през 1991 г., използвайки диференциален крипто анализ, че DES може да бъде атакуван, в който ключът е бил изчислен, при условие че нападателят е имал определен брой специално подбрани двойки отворени и криптирани съобщения.
- Японският учен Мицуру Мацуи през 1993 г. показа, че ключът може да бъде изчислен с помощта на линейния крипто анализ. За да се направи това, просто трябва да се знае определено количество открит текст и съответното криптирано съобщение.

4.5. СЪВРЕМЕННИ БЛОКОВИ ШИФРИ. ADVANCED ENCRYPTION STANDARD (AES)

История на AES

Advanced Encryption Standard (AES) е стандарт за криптиране на данни, приет през 2001 г. от NIST (*National Institute of Standards and Technology*).

През 1997 г. NIST започват разработката на нов стандарт за криптографски алгоритъм. Стандартът трябвало да бъде по-добър от тогава използвания стандарт DES. DES вече бил твърде уязвим на атаки с изчерпване на случаите (*brute-force attacks*). Новият алгоритъм трябвало да бъде:

- Способен да защитава класифицирана държавна информация;
- Лесен за имплементиране в софтуер и хардуер;
- Да предоставя добра защита срещу различни техники за атака.

Приемането му е предшествано от 5-годишен процес на изследване и разработване в резултат на което са били предложени 15 различни конкурентни проекта (алгоритъма). През 1999 г. са избрани 5 алгоритъма, които да продължат участие в конкурса:

- MARS (представен от IBM);
- RC6 (представен от *RSA Security*);
- Rjindael (*Vincent Rijmen and Joan Daemen*);
- Serpent (*R. Anderson, E. Biham & L. Knudsen*);
- Twofish (представен от екип учени от *Counterpane Internet Security*).

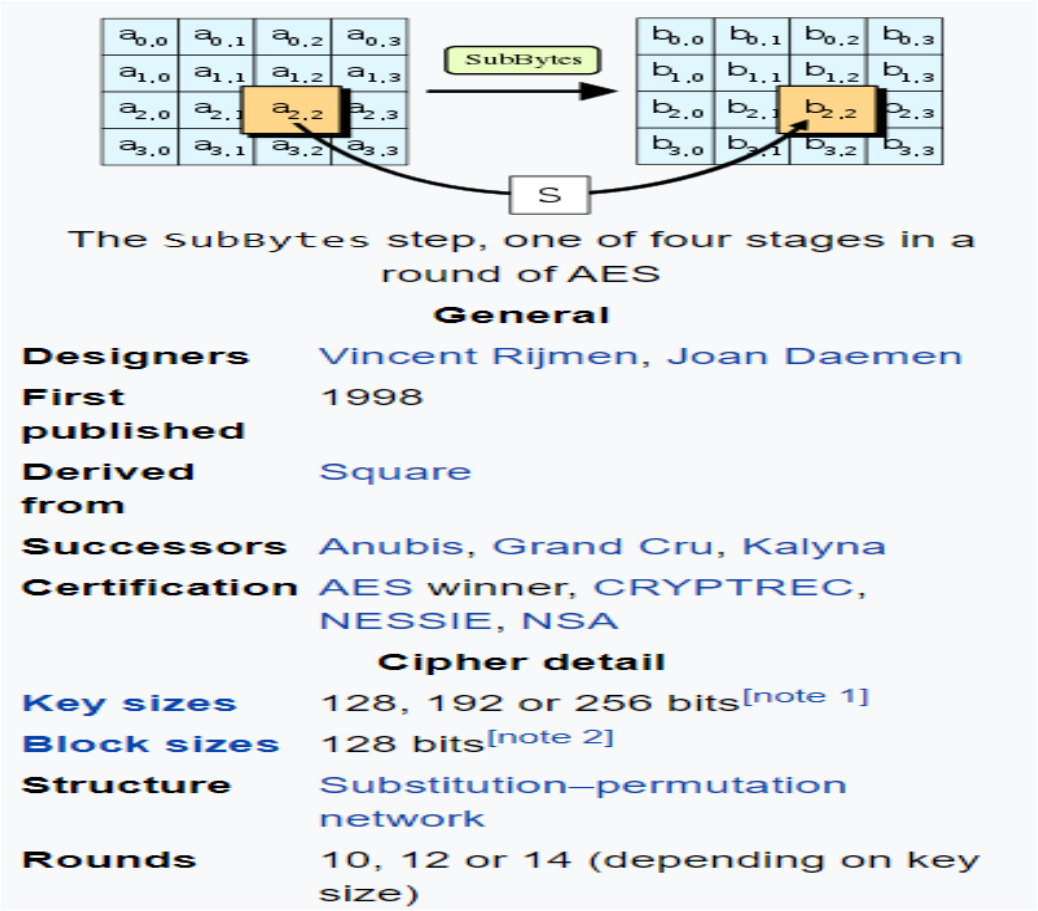
Алгоритмите са били реализирани на езиците за програмиране ANSI C и Java и били тествани за:

- Надеждност;
- Скорост при криптиране и декриптиране;
- Устойчивост на атаки в хардуера и софтуера;
- Време за подготвяне на ключовете;
- Сигурност на алгоритъма.

След оценка и преразглеждане на всички проекти през октомври 2000 г. е избран шифърът Райндаал (Rijndael), като най-подходящ за стандарт. Името му е комбинация между имената на създателите му, двамата белгийци – Винсент Раймен (*Vincent Rijmen*) и Жоан Дамен (*Joan Daemen*).

Райндаал представлява семейство от шифриращи алгоритми с различни ключове и размери на блока. При разработването на AES, NIST избират три от алгоритмите Райндаал – всички с дължина на блока 128 бита, но с различна дължина на ключа – 128, 192 и 256 бита.

В САЩ алгоритъмът AES е представен официално (*FIPS 197*) на 26 ноември 2001 г. Той е много по-сигурен от своите предшественици DES и 3DES, с по-дълги ключове. Позволява по-бързо криптиране, което го прави изключително подходящ за софтуерни приложения, фърмуер и хардуер. Използва се в много протоколи (*SSL – Secure Sockets layer* и *TLS – Transport Layer Security*) и може да бъде открит в повечето модерни приложения и устройства, които използват криптиране.

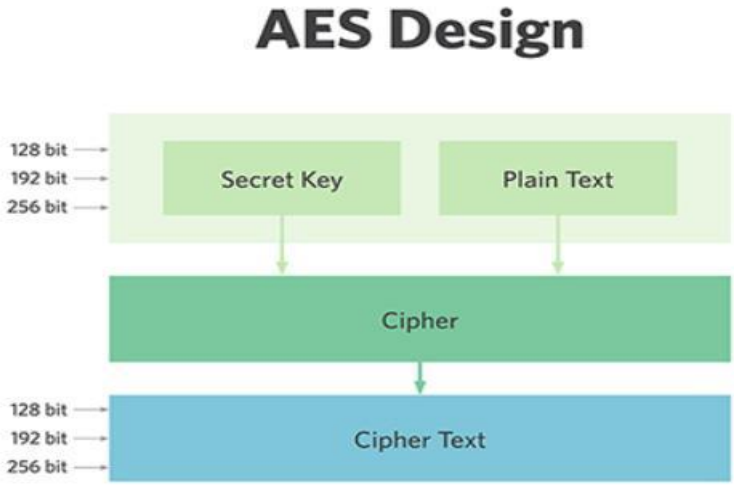


Фигура 4.26. Общо представяне на AES

Характеристика на AES

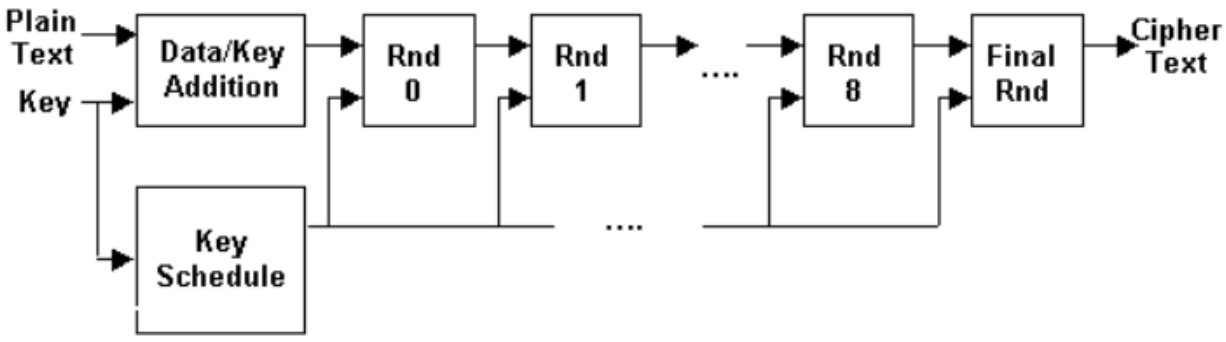
Основните характеристики на криптографския алгоритъм AES са:

- AES е симетричен алгоритъм с блоково криптиране;
- Блоковете са с фиксирана дължина от 128 бита (16 байта). AES работи върху матрица от байтове от 4 x 4 колони и редове, наречен състояние (*state*). Върху тази матрица се извършват криптиращите трансформации на алгоритъма Rijndael;
- Дължината на ключа варира между 128бита, 192 бита и 256 бита;
- Циклите са съответно:
  - 10 цикъла за 128-битов ключ;
  - 12 – за 192-битов ключ ;
  - 14 – за 256-битов ключ.



Фигура 4.27. Характеристика на AES

Цикълът съдържа няколко стъпки за обработване, които включват заместване, разместване и смесване на входния открит текст (открито съобщение) до получаване на криптирания текст (криптирано съобщение). Набор от обратни стъпки се прилага за преобразуване на криптирания текст обратно в оригиналния явен текст, използвайки същия ключ за криптиране. За разлика от своя предшественик DES, ***AES не използва схемата на Feistel***.

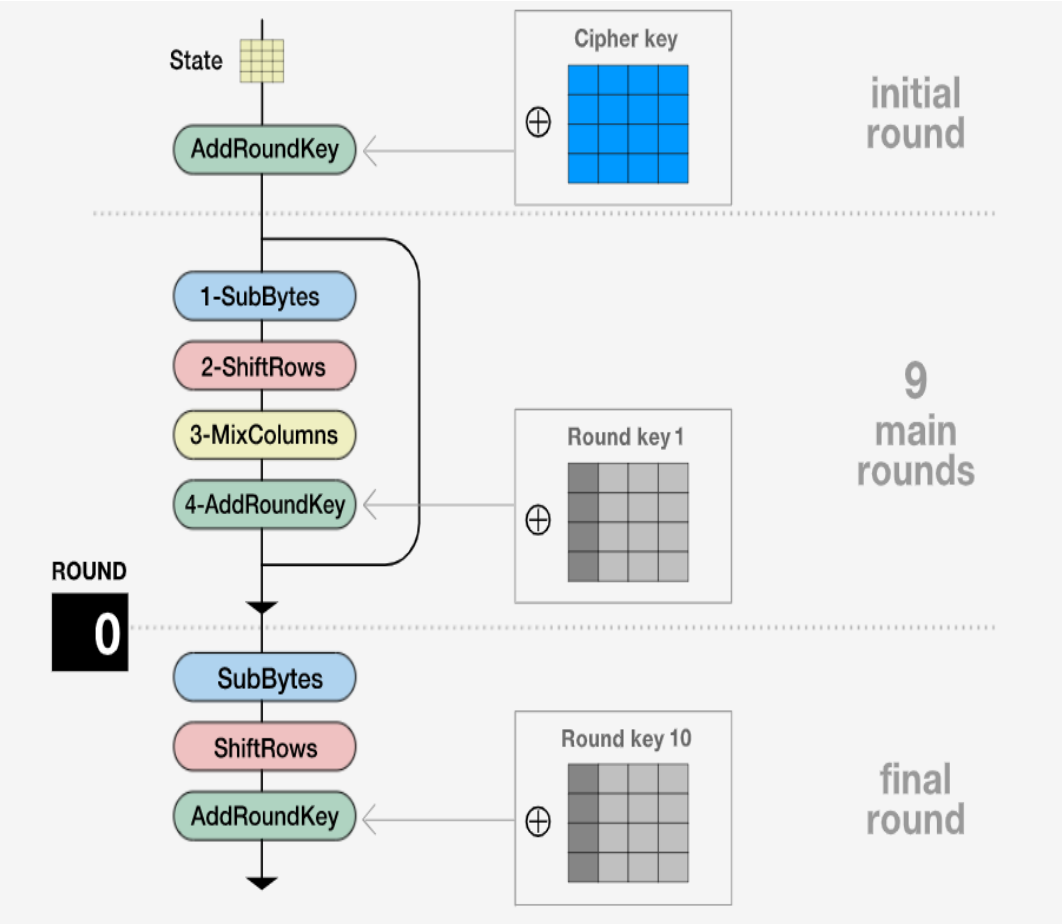


Фигура 4.28. Схема на работа за AES с 128-битов ключ

Структура на AES

Алгоритъмът на AES е структуриран в следните стъпки:

- ***Инициализираща*** – всеки байт на състоянието се сумира по битово (XOR) с байт на ключа за съответната стъпка;
- ***Основна*** – изпълнение на 9 (за 128 битов ключ) , 11 (за 192 битов ключ) или 13 (за 256 битов ключ) кръга от цикличната обработка;
- ***Заключителна*** – последен кръг от цикличната обработка.



Фигура 4.29. Структура на AES за 128 битов ключ (10 цикъла)

Описание на процедурите на алгоритъма AES

Алгоритъмът AES използва следните процедури:

- ***KeyExpansion*** – ключовете във всеки кръг се извличат от криптиращия ключ, използвайки схема на ключовете на Rijndael (*Rijndael's key schedule*).

- Инициализираща стъпка:
  - **AddRoundKey** – всеки байт на състоянието се комбинира с байт на ключа от съответния кръг, използвайки по битово XOR.

- Основна стъпка (9, 11 или 13 кръга):
  - **SubBytes** – нелинейна стъпка на заместване, при която всеки байт се заменя с друг според S-таблицата (*S-box*).
  - **ShiftRows** – стъпка на транспониране, при която последните три реда на състоянието се изместват циклично с определен брой позиции.
  - **MixColumns** – линейна операция на смесване, която работи върху колоните на състоянието, комбинирайки четирите байта във всяка колона.
  - **AddRoundKey**.

- Заключителна стъпка (общо 10, 12 или 14 кръг):
  - SubBytes.
  - ShiftRows.
  - AddRoundKey.

```
PROCEDURE Cipher128(in[16]:byte, out[16]:byte, w[44]:word)
BEGIN
  state[4,4]:byte
  state = in
  AddRoundKey(state, w[0, 3])
  FOR round = 1 to 10
    SubBytes(state)
    ShiftRows(state)
    MixColumns(state)
    AddRoundKey(state, w[round*4, (round+1)*4-1])
  END FOR
  SubBytes(state)
  ShiftRows(state)
  AddRoundKey(state, w[40, 43])
  out = state
END PROCEDURE
```

Фигура 4.30. Изходен текст за реализация на AES за 128 битов ключ (10 цикъла)

Процедура KeyExpansion (Разширение на ключа)

Ключовете във всеки кръг се извличат от криптиращия ключ, използвайки схема на ключовете на Rijndael (*Rijndael's key schedule*).

Round constants (Rcon)

Стъпковата константа rcon_i се използва в i-тата стъпка на процедурата ***KeyExpansion*** и е 32-битова дума

$$rcon_i = \begin{bmatrix} rc_i & 00_{16} & 00_{16} & 00_{16} \end{bmatrix}$$

където rc_i е осем битово число (стойност), дефинирано както следва:

$$rc_i = \begin{cases} 1 & \text{if } i = 1 \\ 2 \cdot rc_{i-1} & \text{if } i > 1 \text{ and } rc_{i-1} < 80_{16} \\ (2 \cdot rc_{i-1}) \oplus 11\text{B}_{16} & \text{if } i > 1 \text{ and } rc_{i-1} \geq 80_{16} \end{cases}$$

Забележка. Действията са в 16-тична бройна система и сумирането е XOR.

Стойност на rc_i

i	1	2	3	4	5	6	7	8	9	10
rc_i	01	02	04	08	10	20	40	80	1B	36

Фигура 4.31. Стойност на rc_i в шестнадесетична бройна система

Означения

- N е дължината на ключа в 32 - битови думи: 4 думи за AES-128, 6 думи за AES-192 и 8 думи за AES-256;
- K_0, K_1, ... K_{N-1} са 32 - битовите думи на основния (оригиналния) ключ;
- R е броят на необходимите за всяка стъпка ключове: 11 ключа за AES-128, 13 ключа за AES-192 и 15 ключа за AES-256;
- W_0, W_1, ... W_{4R-1} са 32 - битовите думи на разширения ключ.

Дефиниране на процедури

- RotWord – циклично отместване на ляво с един байт

$$\mathrm{RotWord}([\,b_0 \quad b_1 \quad b_2 \quad b_3\,]) = [\,b_1 \quad b_2 \quad b_3 \quad b_0\,]$$

- SubWord – всеки байт се замества с помощта AES S-box

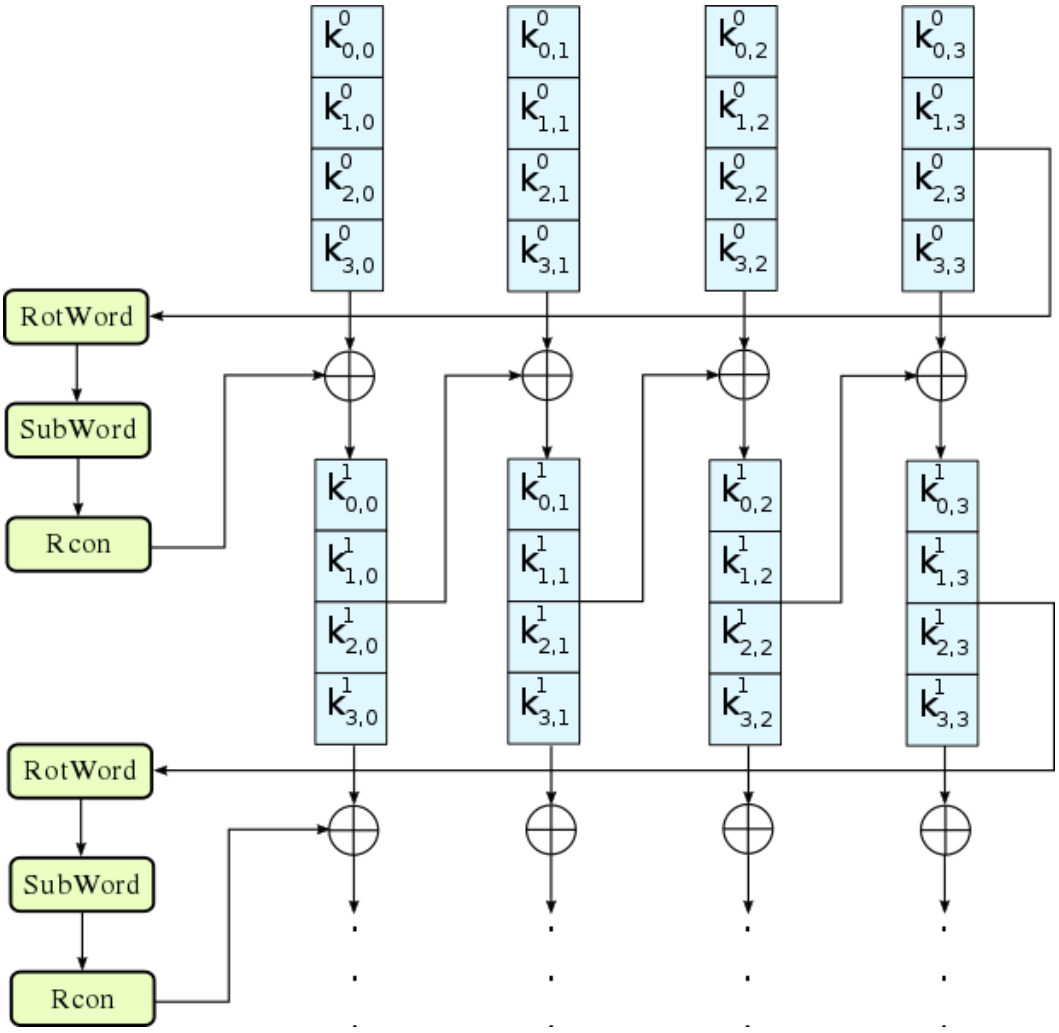
$$\mathrm{SubWord}([\,b_0 \quad b_1 \quad b_2 \quad b_3\,]) = [\,\mathrm{S}(b_0) \quad \mathrm{S}(b_1) \quad \mathrm{S}(b_2) \quad \mathrm{S}(b_3)\,]$$

Разширен ключ

Тогава W_0, W_1, ... W_{4R-1} 32-битовите думи на разширения ключ се получават по правилото:

$$W_i = \begin{cases} K_i & \text{if } i < N \\ W_{i-N} \oplus \mathrm{SubWord}(\mathrm{RotWord}(W_{i-1})) \oplus rcon_{i/N} & \text{if } i \geq N \text{ and } i \equiv 0 \pmod{N} \\ W_{i-N} \oplus \mathrm{SubWord}(W_{i-1}) & \text{if } i \geq N, N > 6, \text{and } i \equiv 4 \pmod{N} \\ W_{i-N} \oplus W_{i-1} & \text{otherwise.} \end{cases}$$

Графичното представяне на алгоритъма за разширение на ключа е както следва:

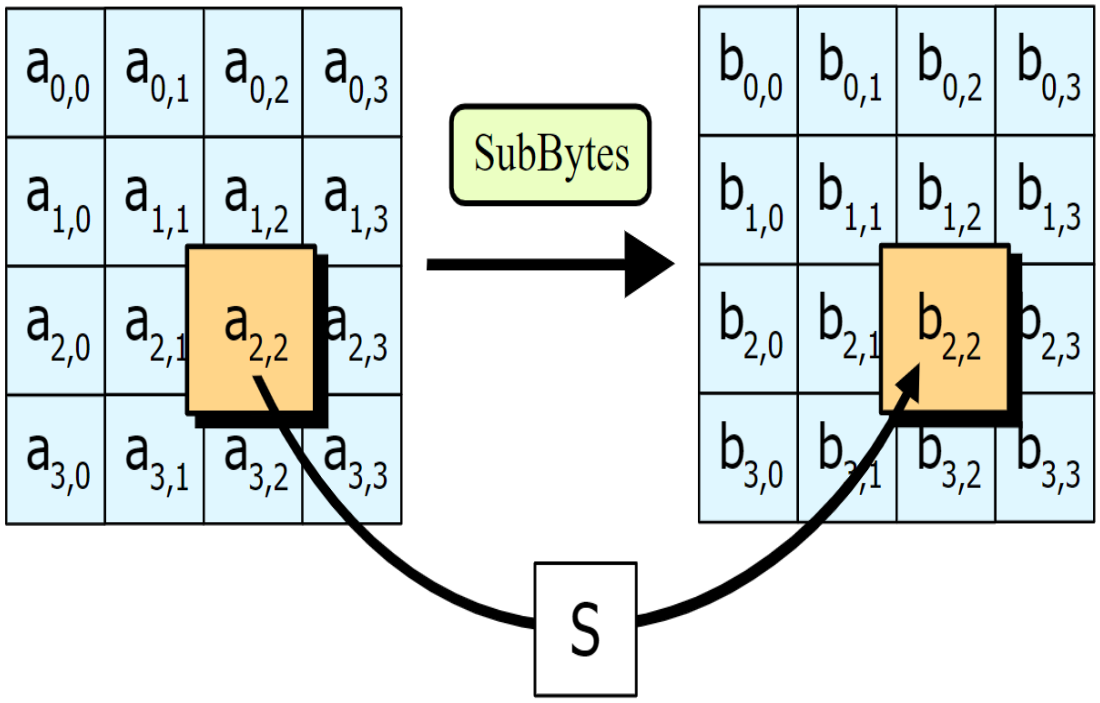


Фигура 4.32. Алгоритъм за разширение на ключа за AES-128

Процедура SubBytes

В процедурата ***SubBytes*** всеки байт a(i,j) от матрицата на състоянието се заменя с S(a(i,j)), използвайки 8-битова таблица за заместване S-кутия (S-box). Тази операция осигурява нелинейността в шифъра. Използваната S-кутия е избрана така, че да има добри нелинейни свойства. За да се избегнат атаки, базирани на прости алгебрични свойства, S-кутията е конструирана чрез комбиниране на обратната функция с необратима афинна

трансформация. S-таблицата също е избрана, за да се избегнат всякакви фиксирани точки, т.е. S(a(i,j)) != a(i,j), а също и всякакви противоположни фиксирани точки, т.е. S(a(i,j)) ⊕ a(i,j) != FF16. При извършване на декриптиране се използва процедурата ***InvSubBytes*** (обратната на ***SubBytes***), която изисква първо да се извърши обратната на афинната трансформация и след това да се намери мултипликативната инверсия.



Фигура 4.33. Процедура SubBytes

AES S-Box

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

The column is determined by the least significant nibble, and the row by the most significant nibble. For example, the value `0x9a` is converted into `0xb8`.

Фигура 4.34. AES S-Box

S-Box

Ако приемем означенията, че

- a(i,j) от матрицата на състоянието се заменя с S(a(i,j));
- a(i,j) = (b_0 b_1 b_2 b_3 b_4 b_5 b_6 b_7);
- S(a(i,j)) =(s_0 s_1 s_2 s_3 s_4 s_5 s_6 s_7),

тогава правилото за заместване се определя по формулата:

$$\begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Определяне на S(a(i,j)) от a(i,j)

Inverse S-box

При дешифрирането се използва обратната S-кутия (Inverse S-box). Тогава правилото за заместване се определя по формулата:

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \\ s_4 \\ s_5 \\ s_6 \\ s_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Определяне на a(i,j) от S(a(i,j))

Inverse S-box

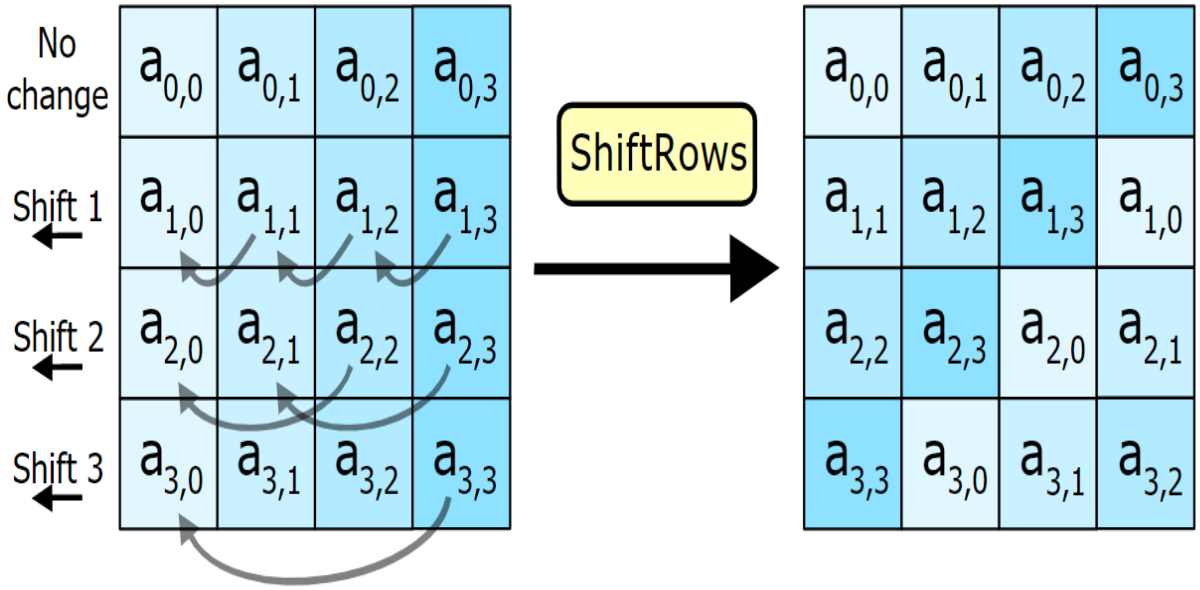
	00	**01**	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**0a**	**0b**	**0c**	**0d**	**0e**	**0f**
00	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
10	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
20	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
30	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d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

Фигура 4.35. Inverse AES S-Box

Процедура ShiftRows

Процедурата ***ShiftRows*** се изпълнява върху редовете на матрицата на състоянието, като циклично се изместват байтовете във всеки ред с определено отместване. При AES първият ред се оставя непроменен. Всеки

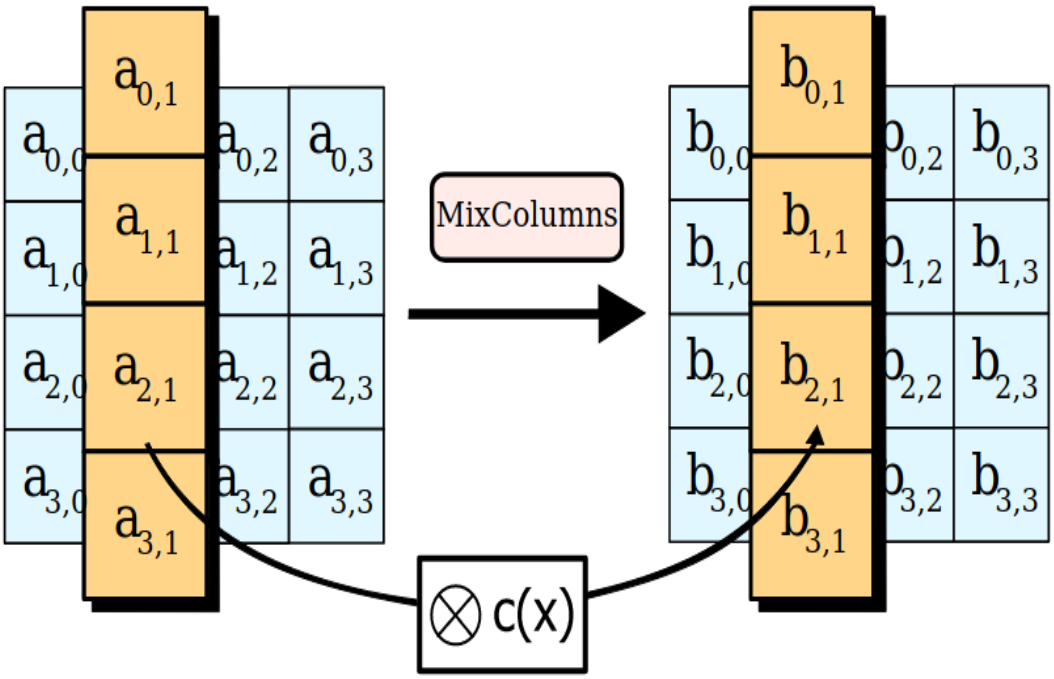
байт от втория ред се премества един вляво. По подобен начин, третият и четвъртият ред се изместват съответно с две и три позиции. По този начин всяка колона на изходното състояние на стъпката ***ShiftRows*** се състои от байтове от всяка колона на състоянието на входа. Значението на тази стъпка е да се избегне колоните да бъдат криптирани поотделно, в този случай AES се разпада на четири независими блокови шифъра.



Фигура 4.36. Процедура ShiftRows

Процедура MixColumns

При процедурата ***MixColumns***, четирите байта на всяка колона на състоянието се комбинират, като се използва обратима линейна трансформация. Функцията ***MixColumns*** приема четири байта като вход и извежда четири байта, където всеки входен байт влияе на всички четири изходни байта. Заедно с ***ShiftRows, MixColumns*** осигурява дифузия (разбъркване) в шифъра.



Фигура 4.37. Процедура MixColumns

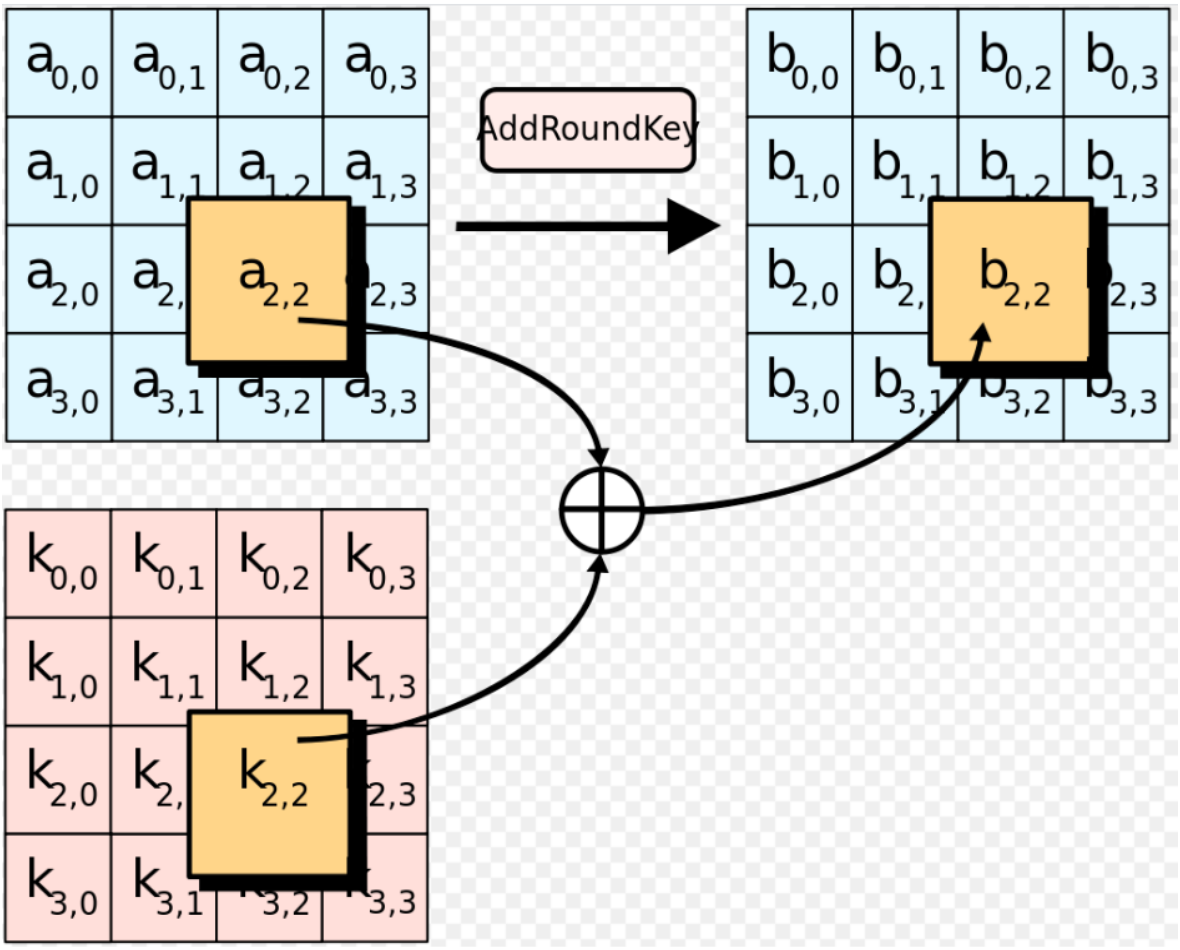
По време на тази операция всяка колона се трансформира с помощта на фиксирана матрица (матрицата, умножена по колона, дава нова стойност на колоната в състоянието).

$$\begin{bmatrix} b_{0,j} \\ b_{1,j} \\ b_{2,j} \\ b_{3,j} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \begin{bmatrix} a_{0,j} \\ a_{1,j} \\ a_{2,j} \\ a_{3,j} \end{bmatrix} \qquad 0 \leq j \leq 3$$

Трансформация в процедурата MixColumns

Процедура AddRoundKey

В процедурата ***AddRoundKey*** ключът за съответната стъпка се комбинира със състоянието. Всеки байт на ключа за съответната стъпка се сумира със съответния байт на състоянието, използвайки по битово XOR.



Фигура 4.38. Процедура AddRoundKey

Декриптиране

При декриптиране се използват обратните процедури:

- **InvSubBytes;**
- **InvShiftRows;**
- **InvMixColumns;**

и процедурата **AddRoundKey.**

Сигурност на AES

През юни 2003 г. правителството на САЩ информира, че AES би могъл да се използва за защита и на класифицирана информация:

„Дизайнът и силата на всички ключови дължини на алгоритъма AES (т.е., 128, 192 и 256) са достатъчни за защита на класифицирана информация до ниво секретно. Строго секретната информация ще изисква използването на 192 или 256 дължина на ключа. Внедряването на AES в продукти, предназначени за защита на националните системи за сигурност и/или информация, трябва да стане, едва когато те бъдат преразгледани и сертифицирани от NSA и след това да се говори за тяхното придобиване и използване.“

Публикуване

ADVANCED ENCRYPTION STANDARD (AES) е публикуван като **Federal Information Processing Standards, Publication 197 (*FIPS 197*)** от **26 ноември 2001 г.**

4.6. КРИПТОГРАФСКИ АЛГОРИТМИ С ПУБЛИЧЕН КЛЮЧ

Исторически бележки

Концепцията за криптография с публичен ключ е измислена от Дифи и Хелман (*Whitfield Diffie и Martin Hellman*) и независимо от тях от Ралф Меркъл (*Ralf Merkle*). Техният принос в криптографията е идеята, че ключовете могат да бъдат по двойки – ключ за криптиране и ключ за декриптиране и че не е възможно да се генерира единия ключ от другия. Дифи и Хелман представят тази концепция за първи път през 1976 г. на Национална компютърна конференция. Няколко месеца по-късно техният основен доклад „Нови насоки в Криптографията“ е публикуван. Поради трудности в процеса на публикуване, първият резултат на Меркъл се появил едва през 1978 г.

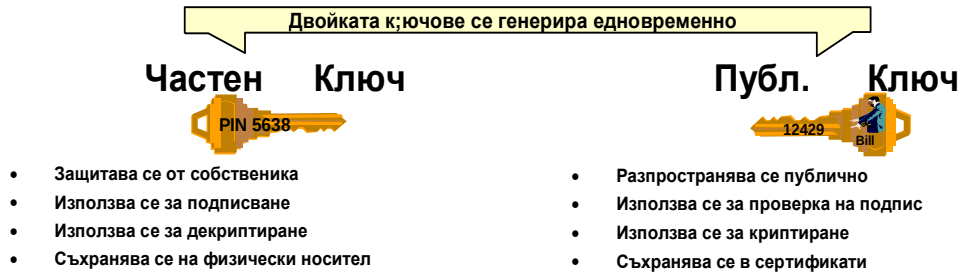
При алгоритмите с публични ключове криптирането и декриптирането се извършва с различни ключове – личен (частен) ключ и публичен ключ (означаваме с K_{pr} и K_{publ}). Освен това декриптиращият ключ не може да бъде (за обозримо време) пресметнат от криптиращия ключ. Криптиращият ключ се нарича публичен ключ и той не се пази в тайна. Всеки може да го използва, за да криптира съобщение (M), но само който знае съответстващия му личен ключ може да декриптира съобщението. Декриптиращият ключ се нарича личен ключ. Понякога съобщението се криптира с личния ключ и се декриптира с публичния.

Личен и публичен ключ

Връзките между личния и публичния ключ са както следва:

- $E_{Kpubl}(M) = C$ и $D_{Kpr}(C) = M$;
- $E_{Kpr}(M) = C$ и $D_{Kpubl}(C) = M$.

Информация за връзките и използването на двата ключа е показана на фигурата:



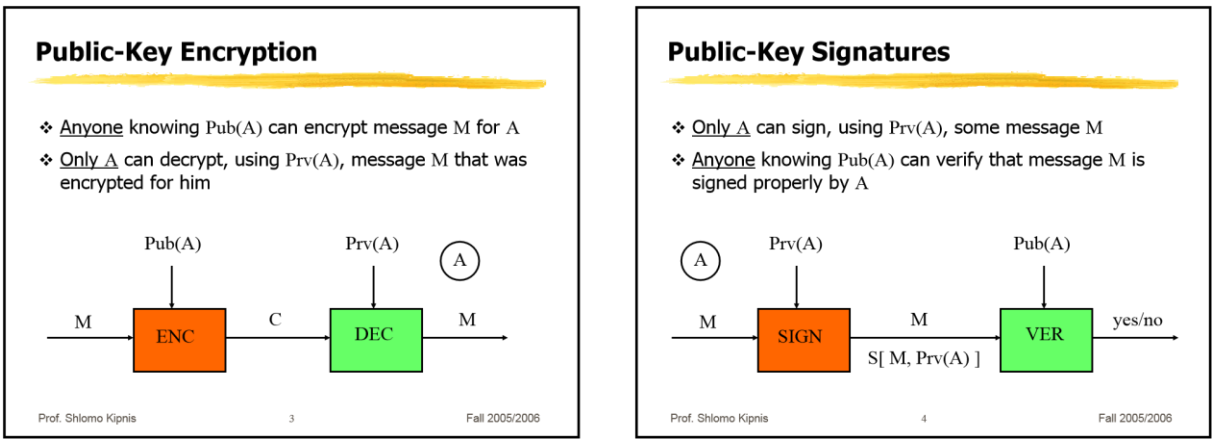
Фигура 4.39. Връзки и използване на двойката асиметрични ключове

От 1976 г. са предложени множество алгоритми за криптография с публичен ключ. Много от тях са несигурни. От тези, които все още се считат за сигурни, много са непрактични. Или имат твърде голям ключ, или криптираният текст е много по-голям от откритият текст. Само няколко алгоритъма са едновременно сигурни и практични. Тези алгоритми обикновено се базират на:

- Разлагане големи числа на прости множители;
- Изчисляване логаритъм в крайно поле;
- Изчисляване на корен на алгебрично уравнение.

Сигурните и практични алгоритми с публичен ключ могат да се използват с различи цели:

- Криптиране и декриптиране;
- Цифров подпис;
- Разпространение на криптографски ключове;
- Идентификация на страните в процеса на взаимодействие.



Фигура 4.40. Криптиране и цифров подпис с алгоритми с публичен ключ

От алгоритмите с публичен ключ, някои са подходящи само за разпространение на ключове. Други са подходящи за криптиране и като за разпределение на ключове. Други са полезни само за цифрови подписи. Само три алгоритъма работят добре както за криптиране, така и за цифрови подписи:

- RSA;
- ElGamal;
- Rabin.

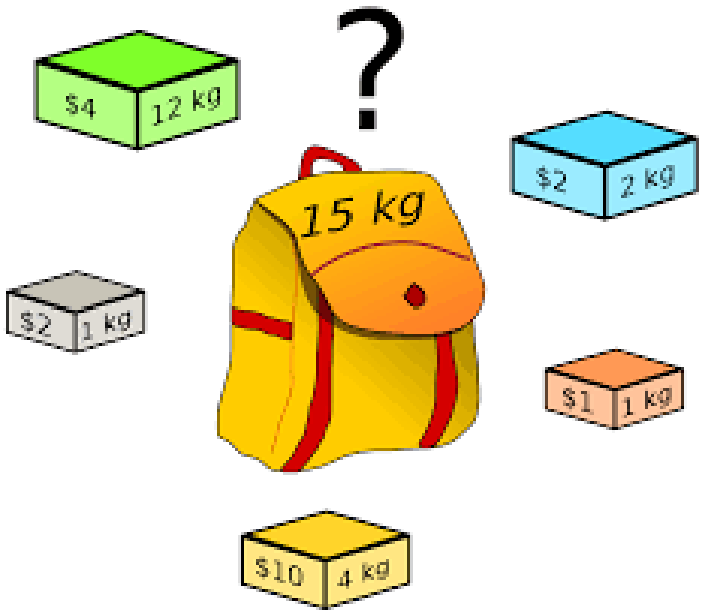
Всички тези алгоритми са бавни. Те криптират и декриптират данни много по-бавно от симетричните алгоритми и затова най-често се използват в хибридните криптосистеми, които използват:

- Симетричен алгоритъм с произволен сесиен ключ за криптиране / декриптиране на съобщението;
- Алгоритъм с публичен ключ за установяване и разпределение на произволния сесиен ключ.

4.6.1. Задачата за раницата (*the knapsack problem*)

Проблемът с раницата е проблем при комбинаторната оптимизация: Задачата е като се има набор от артикули, всеки с тегло и стойност, да се определи броя на всеки артикул, който да включите в колекцията, така че

общото тегло да е по-малко или равно на дадената граница на колекцията и общата стойност е възможно най-голяма. Той получава името си от проблема, пред който е изправен някой, който е ограничен от раница с фиксиран размер и трябва да го напълни с най-ценните предмети. Проблемът често възниква при разпределението на ресурсите.



Фигура 4.41. Задачата за раницата

Задачата за раницата се изучава повече от век, като ранните изследвания са от далечната 1897 г. Името „проблем с раницата“ датира от ранните трудове на математика Тобиас Данциг (1884 – 1956), и се отнася до обичайния проблем с опаковането на най-ценните или полезни предмети без претоварване на багажа. Решаването на задачата за раницата се появява в реални процеси на вземане на решения в най-различни области, като например намиране на най-добрия начин за намаляване на суровините, избор на инвестиции и портфейли, избор и обезпечения на и генериране на ключове за Merkle–Hellman и други криптосистеми базирани на задачата за раницата.

Първият алгоритъм за криптиране с публичен ключ е алгоритъмът за раници, разработен от Ралф Меркъл и Мартин Хелман. Може да се използва само за криптиране, въпреки че по-късно Ади Шамир адаптира системата за цифрови подписи.

Формулировка на задачата за раницата

Задачата за раницата се формулира лесно. Като се има предвид множество предмети, всеки с различно тегло, дали е възможно да се поставят някои от тези предмети в раницата, така че раницата да тежи определено тегло?

Формално представяне

Даден набор от стойности $\mathbf{M_1, M_2, ..., M_n}$ и сума **S**, да се намерят числата (стойностите на $\mathbf{b_i, i = 1, 2, …, n}$)) така, че:

$$\mathbf{S = b_1M_1 + b_2M_2 + ... + b_nM_n}$$

Стойностите на b_i могат да бъдат нула или единица. Единица означава, че предметът е в раницата, а нула показва, че предметът не е в раницата.

Например предметите могат да имат тежести 1, 5, 6, 11, 14 и 20. Можете да опаковате раница, която тежи 22 използвайки тежести 5, 6 и 11. Не бихте могли да опаковате раница с тегло 24.

На практика съществуват две задачи за раницата:

- Лесна задача;
- Трудна задача.

Лесна задача за раницата

Ако списъкът с тегла е ***супер увеличаваща (нарастваща) се последователност***, тогава възникналият проблем с раницата е лесен за решаване. ***Супер нарастващата последователност*** е последователност, в която всеки член е по-голям от сумата на всички предишни членове.

Например {1, 3, 6, 13, 27, 52} е една ***супер увеличаваща се последователнос***т, но {1, 3, 4, 9, 15, 25} не е.

Решението за супер нарастваща раница е лесно да се намери.

- Вземете общото тегло и го сравнете с най-голямото число в последователността:
  - Ако общото тегло е по-малко от това число, то не е в раницата;
  - Ако общото тегло е по-голямо или равно на числото, то най-голямото число е в раницата.
- Намалете теглото на раницата със стойността и преминете към следващото най-голямо число в последователността.
- Повторете стъпките, докато приключите.
- Накрая:
  - Ако общото тегло е доведено до нула, тогава има решение;
  - Ако общото тегло не е, задачата няма решение.

ПРИМЕР

Нека общото тегло на раницата е **70** и последователността от тежести е от {**2, 3, 6, 13, 27, 52**} (**супер увеличаваща се последователност**). Алгоритъмът за решение на лесната задача за раницата е:

- Най-голямото тегло 52 е по-малко от 70, така че 52 е в раницата;
- Изваждаме 52 от 70, остават 18;
- Следващото тегло 27 е по-голямо от 18, така че 27 не е в раницата;
- Следващото тегло 13, е по-малко от 18, така че 13 е в раницата;
- Изваждаме 13 от 18, остават 5;
- Следващото тегло 6 е по-голямо от 5, така че 6 не е в раницата;
- Продължаването на този процес ще покаже, че и 2 и 3 са в раницата и останалото тегло е доведено до 0, което показва, че е намерено решението.

Трудна задача за раницата

Когато последователността от тежести ***не е супер увеличаваща се (нормални последователност)*** задачата за раницата е трудна задача, тъй като няма известен бърз алгоритъм за решение. Единственият известен начин да се определи кои предмети са в раницата е да се тестват методично всички възможности, докато не се натъкнете на правилното решение. Най-бързите алгоритми, като се вземат предвид различните евристики, растат експоненциално с броя на възможните тегла в раницата. Ако се добави един елемент към последователността от тежести, това ще отнеме два пъти повече време за намиране на решението. Това е много по-трудно отколкото при супер нарастващата последователност, където, ако добавите още една тежест към последователността, това просто отнема само една нова операция за намиране на решението.

4.6.2. Криптографско решение на Merkle–Hellman

Идеята на алгоритъма за раници използвана от Merkle–Hellman е да се кодира откритото съобщение като решение на множество (поредица) от задачи за раницата. Блок от открит текст, равен по дължина на броя елементи в множеството ще избере елементите в раницата (битове за свободен текст, съответстващи на стойностите $\mathbf{b_i}$) и криптираният текст ще бъде получената сума.

От примера по-горе за открития текст 70 криптираният текст е 110101

Алгоритъм на Merkle–Hellman

Алгоритъмът Merkle–Hellman (раничен алгоритъм) се основава на следните параметри:

- Частният ключ е супер увеличаваща са последователност от тегла за решаване на задачата за раницата;

- Публичният ключ е последователност от тегла с нормална последователност за решаване на задачата за раницата със същото решение.

Merkle и Hellman са разработили техника за конвертиране на супер увеличаващата се последователност в нормална последователност за задачата за раницата. Решението се базира на модулната аритметика.

ПРИМЕР

- Разглеждаме супер увеличаваща последователност, например {2, 3, 6, 13, 27, 52};
- Избираме числата n и m, така че:
  - m трябва да бъде число по-голямо от сумата на всички числа в последователността, например 105;
  - n не трябва да има общи делители с m, например 31;
- Умножаваме всички стойности с число n по mod m;
- Пресмятането на нормалната последователност на задачата за раницата тогава ще бъде както следва:
  - 2 * 31 mod 105 = 62
  - 3 * 31 mod 105 = 93
  - 6 * 31 mod 105 = 81
  - 13 * 31 mod 105 = 88
  - 27 * 31 mod 105 = 102
  - 52 * 31 mod 105 = 37
- Нормалната последователност е {62, 93, 81, 88, 102, 37}.

Криптиране

- Откритото съобщение (в двоичен вид) се разделя на блокове с дължина равна на броя на елементите

- За всеки блок се пресмята теглото на раницата
- Шифрираният текст е последователност от получените тегла на раниците

ПРИМЕР

- Открито съобщение 0110001101011101110
- Разделено на блокове
  - 011000
  - 110101
  - 101110
- Пресмятаме теглото на раницата за всеки блок
  - За 011000 теглото е 93 + 81 = 174
  - За 110101 теглото е 62 + 93 + 88 + 37 = 280
  - За 101110 теглото е 62 + 81 + 88 + 102 = 333
- Криптираното съобщение е 174,280,333

Декриптиране

Получателят:

- Знае частния ключ – супер увеличаваща последователност, например {2, 3, 6, 13, 27, 52};
- Знае числата n и m, 31 и 105;
- Пресмята се n^{-1} така, че $n(n^{-1}) = 1 \pmod{m}$, $n^{-1}= 61$;
- Знае криптираното съобщение 174,280,333;
- Пресмята се умножението на всички стойности на криптираното съобщение с n^{-1} по (mod m)
  - 174 * 61 mod 105 = 9 = 3 + 6, което е 011000
  - 280 * 61 mod 105 = 70 = 2 + 3 + 13 + 52, което е 110101
  - 333 * 61 mod 105 = 48 = 2 + 6 + 13 + 27, което е 101110

- Декриптираното съобщение е 011000 110101 101110.

4.6.3. RSA (Rivest, Shamir, Adleman)

Скоро след раничния алгоритъм на Merkle е създаден първият пълноправен алгоритъм RSA с публичен ключ, който може да се използва за криптиране и цифрови подписи. От всички алгоритми с публичен ключ предложени през годините, RSA е най-лесният за разбиране и прилагане. Той е и най-популярния. Наименованието идва от фамилните имена на създателите му Ronald L. Rivest, Adi Shamir и Leonard Adleman. Патентован е през 1983 г., освободен е от патент през 2000 г.

Алгоритъмът се базира на трудността да се разложи едно естествено число на n прости множители.

Алгоритъмът е намерил широко приложение в много системи и стандарти (в операционните системи на Майкрософт, Епъл и др., SSL, S-HHTP, S-MIME, S/WAN, STT и PCT). В реализирани системи, базирани на този алгоритъм, се ползват ключове с дължина 512, 768, 1024, 2048 бита, но е препоръчително използването минимум на 1024 – битови ключа, за да се постигне дълъг срок на надеждност – от няколко месеца до години.

Лицензът за технологията на RSA е придобит от около 350 компании. Приблизителният брой на машините за кодиране на базата на RSA е около 300 милиона, правейки го досега най-широко използвания алгоритъм за криптиране в света.

Алгоритъм на RSA

Алгоритъмът на RSA обхваща следните елементи:

- Генериране на двойка ключове;
- Криптиране;
- Декриптиране.

Генериране на двойка ключове

Алгоритъмът на RSA за генериране на двойка ключове обхваща следните стъпки:

- Избират се две различни прости числа **p** и **q**. За по-добра сигурност те трябва да бъдат случайни и с еднакъв брой цифри (представени в двоична бройна система);
- Изчислява се **n=pq**. Това е модулът на частния и публичния ключ;
- Изчислява се функцията на Ойлер: **φ(n) = (p-1)(q-1**);
- Избира се цяло число **e**, така че **1 < e < φ(n)**, и освен това **e** и **φ(n)** да нямат общи делители. ***Експонента на публичния ключ е*** **e**;
- Изчислява се **d**, което удовлетворява уравнението **de≡1 (mod φ(n))**. Това означава, че **ed – 1** може да бъде разделено без остатък на **(p-1)(q-1)**. ***Експонента на частния ключ е*** **d**.

Двойката криптографски ключове

Двойката криптографски ключове, публичен и личен са:

- $\mathbf{K_{publ} = (n, e)}$ - публичният ключ е съставен от модула **n** и от публичната експонента **e**;
- $\mathbf{K_{pr} = (n, d)}$ - частният ключ съдържа модула **n** и частната експонента **d**.

Криптиране

Алгоритъмът на RSA за криптиране обхваща следните стъпки:

- Получателят предава публичния си ключ $\mathbf{K_{publ} = (n, e)}$ на изпращача и съхранява частния ключ $\mathbf{K_{pr} = (n, d)}$;
- Изпращачът трябва да изпрати съобщението **M;**

- Първоначално M е превърнато в число **m, 0<m< n,** като се използва предварително договорен за целта протокол. Ако е дадено **m**, оригиналното съобщение **M** може да бъде възстановено, като се обърне схемата;
- Криптиране – изчислява се шифрираното съобщение **c**, където: $\mathbf{c \equiv m^e \pmod n}$

Декриптиране

Алгоритъмът на RSA за декриптиране обхваща следните стъпки:

- Получателят възстановява **m** от **c**, като използва ключовата експонента d от $\mathbf{m \equiv c^d \pmod n;}$
- От полученото **m**, оригиналното съобщение **M** може да бъде възстановено, като се обърне схемата.

Доказателство:

- $\mathbf{c^d = (m^e)^d = m^{ed} \pmod n}$
- тъй като $\mathbf{ed = 1 + k\varphi(n)}$
  $\mathbf{m^{ed} = m^{1 + k\,\varphi(n)} = m(m^k)^{\varphi(n)} = m \pmod n}$

Последното съждение следва от Теорема на Ойлер където **m** е взаимно просто с **n**. С това се доказва, че сме получили първоначалното съобщение:

$\mathbf{c^d = m \pmod n}$

Пример

- Нека **p = 61 и q = 53**
- Изчисляваме **n = 61 x 53 = 3233**
- Изчисляваме функцията на Ойлер $\mathbf{\varphi(3233) = (61 - 1)(53 - 1) = 3120}$

- Избираме **e = 17**
- Изчисляваме **d = 2753**, да удовлетворява **d x e mod φ(n) = 1**
  2753 x 16 mod 3120 = 1
- Публичният ключ е $\mathbf{K_{publ}}$ **= (n = 3233, e = 17)**.
- За съобщението **m**, криптиращата функция е
  $\mathbf{c(m) = m^{17} \bmod 3233}$
- Частният ключ е $\mathbf{K_{pr}}$ **= (n = 3233, d = 2753)**
- За криптираното съобщение **c** декриптиращата функция е
  $\mathbf{m(c) = c^{2753} \bmod 3233}$
- ***Например***
  - За да криптираме **m = 65**, изчисляваме
    $\mathbf{c = 65^{17} \bmod 3233 = 2790}$
  - За да декриптираме **c = 2790**, изчисляваме
    $\mathbf{m = 2790^{2753} \bmod 3233 = 65}$

4.6.4. Криптосхема на Рабин

Схемата на Рабин (Rabin) се базира на трудността да се намерят квадратни корени на модула на съставно (сложно) число. Този проблем е еквивалентен на разделянето на множители. По-късно схемата е предефинирана и подобрена от Hugh Williams.

Схемата на Рабин се състои от стъпките:

- Избиране на две прости числа , **p** и **q** сравними с **3** по модул **4**. Това е личния ключ $\mathbf{K_{pr} = (p, q);}$
- Пресмятане на **n = pq**. Това е публичния ключ . $\mathbf{K_{publ} = (n);}$
- Криптиране на съобщението **M** (**M** трябва да бъде по малко от **n**), като се получава криптираното съобщение
  $\mathbf{C, C = M^2 \bmod n;}$
- Декриптирането на съобщението е както следва:

- Получателят притежава личния ключ $\mathbf{K_{pr} = (p, q)}$. Знае **p** и **q**;
- Използвайки Китайската теорема за остатъците се пресмятат:
  - $\mathbf{m_1 = C^{(p+1)/4} \bmod p}$
  - $\mathbf{m_2 = (p - C^{(p+1)/4}) \bmod p}$
  - $\mathbf{m_3 = C^{(q+1)/4} \bmod q}$
  - $\mathbf{m_4 = (q - C^{(q+1)/4}) \bmod q}$
- Избера се цяло число $\mathbf{a = q(q^{-1} \bmod p)}$
  и цяло число $\mathbf{b = p(p^{-1} \bmod q).}$ Възможни са случаите:
  - $\mathbf{M_1 = (am_1 + bm_3) \bmod n}$
  - $\mathbf{M_2 = (am_1 + bm_4) \bmod n}$
  - $\mathbf{M_3 = (am_2 + bm_3) \bmod n}$
  - $\mathbf{M_4 = (am_2 + bm_4) \bmod n}$

Един от тези четири резултата $\mathbf{M_1, M_2, M_3}$ или $\mathbf{M_4}$, е равен на **M**. Ако съобщението е текст на естествен език е лесно да се изберете правилния $\mathbf{M_i}$. От друга страна, ако съобщението е произволен битов поток (да речем, за ключ или цифров подпис), няма начин да се определи кой $\mathbf{M_i}$ е правилен. Един от начините за решаване този проблем е да добавите известен идентификатор към съобщението преди криптиране.

4.6.5. Криптосхема на Ел–Гамал (*ElGamal*)

Криптографската схема на ElGamal може да се използва както за цифрови подписи, така и за криптиране / декриптиране на данни. Сигурността на схемата се основава на трудността при изчисляване на дискретни логаритми в крайно поле.

Генериране на двойка ключове

Алгоритъмът на ElGamal за генериране на двойка ключове обхваща следните стъпки:

- Избиране на просто число **p**;

- Избиране на две произволни числа, **g** и **x**, така че и двете **g** и **x** са по-малки от **p**;
- Пресмятане на $\mathbf{y = g^x \bmod p}$.

Двойката криптографски ключове

Двойката криптографски ключове, публичен и личен са:

- $\mathbf{K_{publ} = (y, g, p)}$ – е публичният ключ на ElGamal;
- $\mathbf{K_{pr} = (x)}$ – е частният ключ на ElGamal.

Цифров подпис с ElGamal

Подписването на едно съобщение **M** е последователност от стъпките:

- Избор на случайно число **k**, взаимно просто с се **p – 1;**
- Пресмятане на $\mathbf{a = g^k \bmod p;}$
- Определяне на **b** от уравнението $\mathbf{M = (xa + kb) \bmod (p - 1)}$ (Използва се разширения алгоритъм на Евклид);
- Цифровият подпис е двойката **(a, b)**.
  Случайно число **k** трябва да бъде тайно и да се пази;
- За проверка на подписа се проверява равенството
  $\mathbf{y^a a^b \bmod p = g^M \bmod p.}$

Забележка. Всеки подпис и криптиране с ElGamal изисква използването на нова, случайно избрана, различна стойност на **k**.

ПРИМЕР

- Избираме **p = 11** и **g = 2**
- Избираме личен ключ **x = 8**
- Пресмятаме $\mathbf{y = g^x \bmod p = 2^8 \bmod 11 = 3}$
- Публичния ключ е **y = 3, g = 2, and p = 11**.
- За подписване на съобщението **M = 5**

- Избираме случайно число **k = 9,** взаимно просто с **p-1** **(gcd(9, 10) = 1)**
- Пресмятаме $\mathbf{a = g^k \bmod p = 2^9 \bmod 11 = 6}$
- Определяме **b** от уравнението **M = (ax + kb) mod (p - 1)** **5 = (8 * 6 + 9 * b) mod 10**
- Решението е **b = 3**
- Подписът е двойката **(a = 6, b = 3)**

- За проверка на подписа проверяваме равенството
  $\mathbf{y^a a^b \bmod p = g^M \bmod p}$
  $\mathbf{3^6\ 6^3 \bmod 11 = 2^5 \bmod 11}$

Криптиране с ElGamal

Една модификация на ElGamal може да се използва за криптиране и декриптиране на съобщения.

Криптиране

Криптирането на едно съобщение **M** е последователност от стъпките:

- Избор на случайно число **k**, взаимно просто **p – 1;**
- Пресмятане на $\mathbf{a = g^k \bmod p;}$
- Пресмятане на $\mathbf{b = y^k M \bmod p;}$
- Криптираното съобщение **C** е двойката **C = (a, b).**

Декриптиране

Декриптирането на криптираното съобщение **C = (a, b)** е:

- Пресмятане на откритото съобщение **M**
  $\mathbf{M = b/a^x \bmod p.}$
- *Доказателство:*
  от $\mathbf{a^x = g^{kx} \pmod p}$ и $\mathbf{b/a^x = y^k M/a^x = g^{xk} M/g^{xk} = M \pmod p.}$

4.6.6. Криптосхема Дифи–Хелман(*Diffie–Hellman*)

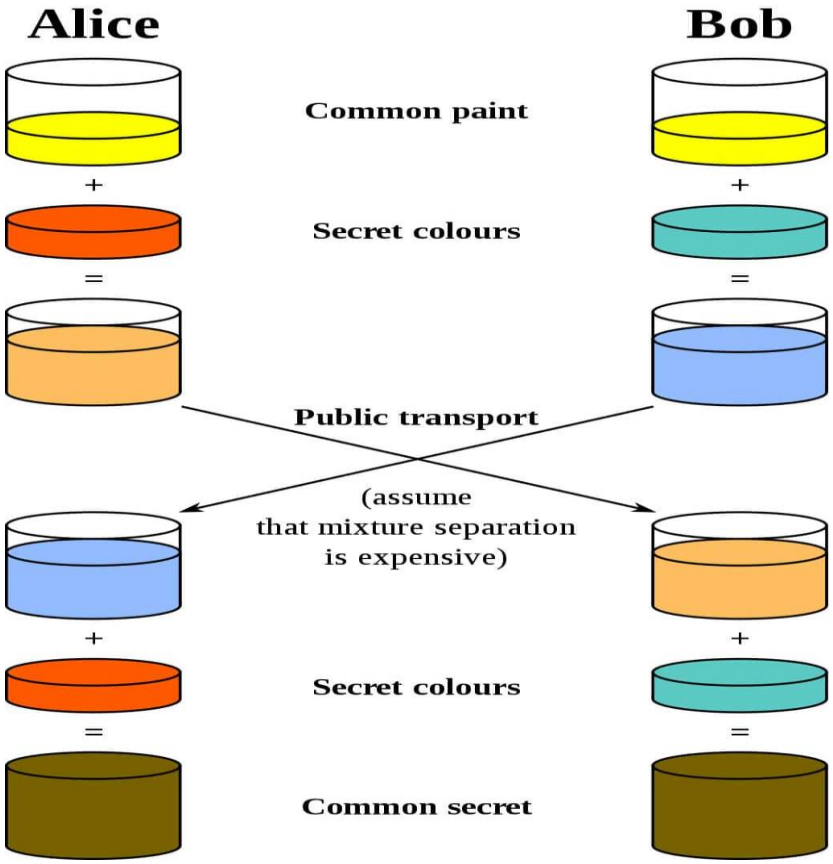
Diffie–Hellman е първият алгоритъм с публичен ключ, изобретен още през 1976 г. Сигурността на алгоритъма произтича от трудността при изчисляване на дискретни логаритми в крайно поле в сравнение с лекота на изчисляване на степенуването в същото поле. Diffie–Hellman може да се използва за разпределение на криптографски ключ, но не може да се използва за криптиране и декриптиране на данни.

Евристично представяне

Във взаимодействието участват две страни , Alice (Алиса) и Bob (Боб).

- Страните се споразумяват за случаен цвят с който да започнат (жълто);
- Двете страни имат свой собствен таен цвят. Нека за Алиса той е червен, докато за Боб цветът е леко зеленикаво синьо;
- Страните смесват тайния си цвят (червен за Алиса, зеленикаво син за Боб) с жълтия, който взаимно се съгласяват;
- В резултат Алиса получава портокалов микс, докато резултатът на Боб е по-дълбоко синьо;
- Страните изпращат резултата си на другата страна. Алиса получава по-дълбокото синьо, докато Боб получава боя с оранжев цвят (портокалов микс);
- Към получения цвят, страните добавят тайния си цвят:
    - Алиса взема по-дълбокото синьо и добавя тайната си червена боя;
    - Боб добавя тайното си зеленикаво-синьо към оранжевата смес;

- Резултатът – и двамата излизат с един и същи цвят, което в случая е отвратително кафяво. Този споделен цвят се интерпретира като общата им тайна.



Фигура 4.42. Установяване на общ таен цвят между Алиса и Боб

Протокол Diffie–Hellman

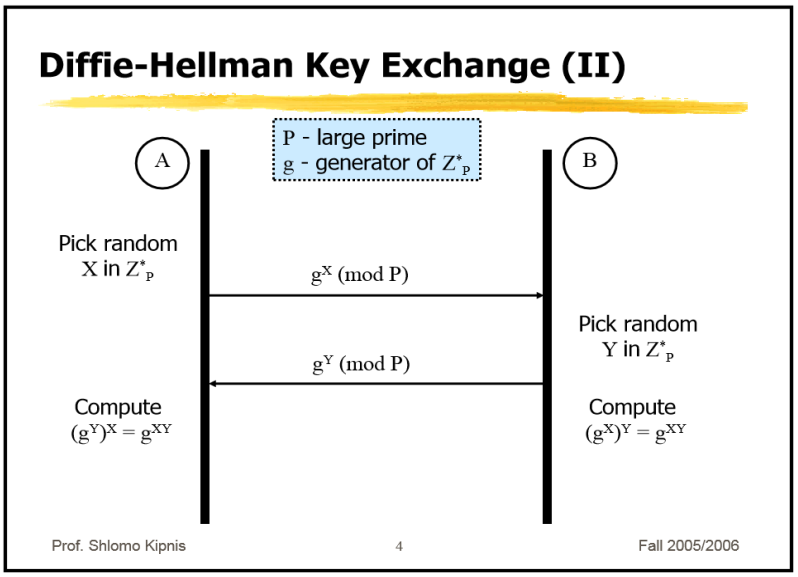
Diffie–Hellman може да се използва за установяване и разпределение криптографски ключ. Алис и Боб могат да използват този алгоритъм, за да генерират таен ключ, но алгоритъмът не може да се използва за криптиране и декриптиране на съобщения.

Протоколът за установяване на ключ е както следва:

- Алис и Боб се споразумяват за голямо просто число **P** и цяло число **g**, такова че **g е примитивно по mod P**. Тези две цели

числа не трябва да са тайни; Алис и Боб могат да се договорят за тях по несигурен канал;

- Алиса избира голямо цяло число **x** и изпраща на Боб
  $X = g^x \bmod P$;
- Боб избира голямо цяло число y и изпраща на Алиса
  $Y = g^y \bmod P$;
- Алиса пресмята **$k = Y^x \bmod P$;**
- Боб пресмята **$k' = X^y \bmod P$;**
- Ключът е **$k = k' = g^{xy} \bmod P$.**



Фигура 4.43. Протокол Diffie-Hellman

Diffie–Hellman с три и повече страни

Протоколът Diffie–Hellman за установяване (обмен) на криптографски ключове може да бъде разширен да работи за обмен между три и повече страни. Следващият пример показва генерирането на секретен ключ между три страни Алис (A), Боб (B) и Карол (K):

- Алиса избира голямо цяло число **x** и изпраща на Боб

$X = g^x \bmod P$;

- Боб избира голямо цяло число y и изпраща на Карол

  $Y = g^y \bmod P$;

- Карол избира голямо цяло число z и изпраща на Алиса

  $Z = g^z \bmod P$;

- Алиса изпраща на Боб

  $Z' = Z^x \bmod P$;

- Боб изпраща на Карол

  $X' = X^y \bmod P$;

- Карол изпраща на Алиса

  $Y' = Y^z \bmod P$;

- Алиса пресмята

  $k = Y'^x \bmod P$;

- Боб пресмята

  $k = Z'^y \bmod P$;

- Карол пресмята

  $k = X'^z \bmod P$.

Секретният ключ е **k, $k = g^{xyz} \bmod P$**. Никой, който подслушва комуникацията не може да пресметне секретния ключ.

4.6.7. Алгоритъм за цифров подпис (*Digital Signature Algorithm*, DSA)

През август 1991 г. Националният институт за стандарти и технологии (NIST) е предложил алгоритъм цифров за подпис (DSA) за използване в американския стандарт за цифров подпис (DSS). На 19 май 1994 г. стандартът е издаден. Този стандарт може да се използва при проектирането и изпълнението схеми за цифров подпис на базата на публичен ключ.

Описание на DSA

DSA е вариант на алгоритмите за подписване на Schnorr и ElGamal.

Алгоритъмът DSA Използва следните параметри:

- **p** = просто число с дължина **L** бита, когато **L** варира от 512 до 1024 и е кратно на 64 (В оригиналния стандарт размерът на **p** е фиксиран на 512 бита, но по-късно е променен от NIST ;
- **q** = 160-битов прост делител на **p – 1**;
- $\mathbf{g = h^{(p-1)/q} \bmod p}$, където **h** е произволно число, по-малко от **p–1** и такова, че $\mathbf{h^{(p-1)/q} \bmod p}$ е по-голямо от 1;
- **x** = число, по-малко от **q**;
- $\mathbf{y = g^x \bmod p}$.

Алгоритъмът също така използва еднопосочна хеш функция **H (m)**. Стандартът определя сигурен хеш алгоритъм (SHA), обект на разглеждане в Приложение 2.

Двойката криптографски ключове

Първите три параметъра **p**, **q** и **g** са публични и могат да бъдат общо достъпни в мрежа от потребители. Частният ключ е **x** и публичният ключ е **y**.

Двойката криптографски ключове, публичен и личен са:

- $\mathbf{K_{publ} = (x)}$ – е публичният ключ на DSA;
- $\mathbf{K_{pr} = (y)}$ – е частният ключ на DSA.

Подпис на съобщение

За да подпишете съобщение **m**:

- Алиса генерира произволно число **k**, по-малко от **q**;
- Алиса пресмята:

- $\mathbf{r = (g^k \bmod p) \bmod q;}$
- $\mathbf{s = (k^{-1}(H(m) + xr)) \bmod q.}$

Пресметнатите **r** и **s** са цифровия подпис на Алиса, които тя изпраща на Боб.

- Боб верифицира подписа пресмятайки:
  - $\mathbf{w = s^{-1} \bmod q;}$
  - $\mathbf{u_1 = (H(m) * w) \bmod q;}$
  - $\mathbf{u_2 = (rw) \bmod q;}$
  - $\mathbf{v = ((g^{u1} * y^{u2}) \bmod p) \bmod q.}$

Ако **v = r**, тогава подписът е верифициран.

Забележка. Математическото доказателство не е обект на разглеждане в тази книга.

4.6.8. Схема на Feige–Fiat–Shamir

Първата версия на схемата за удостоверяване и дигитален подпис е на Amos Fiat и Adi Shamir. По-късно Uriel Feige, Fiat и Shamir модифицират алгоритъма до доказателство за самоличност с нулеви знания. Това е най-известното доказателство за самоличност с нулеви знания.

На 9 юли 1986 г. тримата автори подават американска заявка за патент. Заради възможността за използване във военни системи на 6 януари 1987 г., три дни преди края на шестмесечния им период, Патентното ведомство наложило заповед за секретност (поверителност) по искане на армията. Основният аргумент е, че ***„... разкриването или публикуването на предмета ... ще бъде вредно за националната сигурност*** …“. Това е скандал. През втората половина на 1986 г. авторите представят работата си в конференции в цял Израел, Европа и САЩ. Авторите дори не са били американци граждани и цялата работа е извършена в Института Вайцман в

Израел. Вестта се е разпространила в академичната общност и пресата. В рамките на два дни заповедта за секретност е била отменена.

Опростена схема за идентификация на Feige–Fiat–Shamir

Страните в модела за идентификация са:

- Арбитър;
- Алиса;
- Боб.

Двойката криптографски ключове

Преди да издаде някакви частни ключове, Арбитърът избира случайно число **n**, който е резултат от умножението на два големи прости числа. В реалния живот n трябва да е с дължина поне 512 бита и вероятно по-близо до 1024 бита. Това **n** може да се споделя между групата доказващи.

- За генерирането на публичния ключ на Алиса се избира число **v**, където **v** е а квадратичен остатък по **mod n**. С други думи, **v** се избера така, че уравнението $\mathbf{x^2 = v \pmod n}$ да има решение и $\mathbf{v^{-1} \bmod n}$ съществува. Това **v** е публичният ключ на Алиса.
- Изчислява се най-малкото **s**, за които $\mathbf{s = sqrt\ (v^{-1}) \pmod n}$. Това **s** е частният ключ на Алиса.

Двойката криптографски ключове, публичен и личен са:

- $\mathbf{K_{publ} = (v)}$ – е публичният ключ на Алиса;
- $\mathbf{K_{pr} = (s)}$ – е частният ключ на Алиса.

Протокол за идентификация

Протоколът за идентификация се състои от следните стъпки:

- Алиса избира произволно число r, където **r** е по-малко от n. След това тя изчислява $\mathbf{x = r^2 \bmod n}$ и изпраща **x** към Боб;

- Боб изпраща на Алиса случаен бит **b**;
- Алиса анализира **b**:
  - Ако **b = 0**, тогава Алиса изпраща на Боб **r**;
  - Ако **b = 1**, тогава Алиса изпраща на Боб **y = r * s mod n**;
- Боб анализира:
  - Ако **b = 0,** Боб проверява, че $\mathbf{x = r^2 \bmod n}$, доказвайки, че Алиса знае **sqrt (x)**;
  - Ако **b = 1**, Боб проверява, че $\mathbf{x = y^2 * v \bmod n}$, доказвайки, че Алиса знае $\mathbf{sqrt\ (v^{-1})}$.

Това е един кръг, наречен акредитация, на протокола. Алиса и Боб повтарят този кръг **t** пъти, докато Боб се убеди, че Алиса знае **s**. (Или с други думи Алиса е притежателят на частния ключ $\mathbf{K_{pr} = (s)}$.

Това е протокол за изрязяване и избор. Ако Алиса не знае **s**, тя може да избере **r**, така че да заблуди Боб, ако той й изпрати 0, или тя може изберете **r** така, че да може да заблуди Боб, ако той й изпрати 1. Тя обаче не може да направи и двете. Коефициентите за нейното заблуждение Боб за един кръг са 50 процента. При **t** на брой кръга (повторения) шансовете Алиса да заблуди Боб **t** пъти са **1 на** $\mathbf{2^t}$.

Схема за идентификация на Feige–Fiat–Shamir

В свои публикации Feige, Fiat и Shamir показват как може да увеличат броят акредитации в един кръг и намаляване на взаимодействията на Алиса и Боб.

Двойката криптографски ключове

Числото **n** е произведението на две големи прости числа.

За генериране публичния и частния ключове:

- Избират се **k** различни числа: $\mathbf{v_1, v_2, ..., v_k}$, където всеки $\mathbf{v_i}$ е квадратичен остатък по **mod n**. С други думи, $\mathbf{v_i}$ се изберат така,

че уравнението $\mathbf{x^2 = v_i \bmod n}$ да има решение и $\mathbf{v_i^{-1} \bmod n}$ съществува. Тогава $\mathbf{v_1, v_2, ..., v_k}$, е публичния ключ на Алиса;

- Изчислява се най-малкото $\mathbf{s_i}$ така, че $\mathbf{s_i = sqrt(v_i^{-1}) \bmod n}$.Тогава $\mathbf{s_1, s_2, ..., s_k}$, е частният ключ на Алиса.

Двойката криптографски ключове, публичен и личен са:

- $\mathbf{K_{publ} = (v_1, v_2, ..., v_k)}$ – е публичният ключ на Алиса;
- $\mathbf{K_{pr} = (s_1, s_2, ..., s_k)}$ – е частният ключ на Алиса.

Протокол за идентификация

Протоколът за идентификация се състои от следните стъпки:

- Алиса избира произволно **r**, като **r** е по-малко от **n**. След това тя изчислява $\mathbf{x = r^2 \bmod n}$ и изпраща **x** към Боб;
- Боб изпраща на Алиса произволен двоичен низ с дължина на **k**-бита: $\mathbf{b_1, b_2, ..., b_k;}$
- Алиса изчислява $\mathbf{y = r * (s_1^{b1} * s_2^{b2} * ... * s_k^{bk}) \bmod n}$. (умножават се стойности на $\mathbf{s_i}$, които съответстват на $\mathbf{b_i = 1}$):
    - Ако първият бит на Боб е 1, тогава $\mathbf{s_1}$ е част от продукта;
    - Ако първият бит на Боб е 0, след това $\mathbf{s_1}$ не е част от продукта;
    - и така до **k**;
- Алиса изпраща **y** на Виктор;
- Виктор проверява, че $\mathbf{x = y^2 * (v_1^{b1} * v_2^{b2} * ... * v_k^{bk}) \bmod n}$. (умножението се извършва за стойностите $\mathbf{v_i}$ на базата на произволен двоичен низ:
    - Ако първият му бит е 1, тогава $\mathbf{v_1}$ е част от продукта;
    - Ако първият му бит е 0, тогава $\mathbf{v_1}$ не е част от продукта;
    - и така до **k.**

Алиса и Боб повтарят този протокол **t** пъти, докато Боб се убеди, че Алиса знае $\mathbf{s_1, s_2, ..., s_k}$.(Или с други думи Алиса е притежателят на частния ключ $\mathbf{K_{pr} = (s_1, s_2, ..., s_k)}$.

Шансът Алиса да заблуди Боб е **1** на $\mathbf{2^{kt}}$. Авторите препоръчват шанс **1 на** $\mathbf{2^{20}}$ за измама и предлага числата **k = 5** и **t = 4**.

4.7. ЕЛИПТИЧНИ КРИВИ (ELLIPTIC CURVES)

Исторически бележки

Елиптичните криви са уравнения от вида $\mathbf{y^2 = p(x)}$, където **p(x)** е кубичен полином без повтарящи се корени. Те се появяват за първи път в трудовете на Диофант – 200 – 300 г. пр. хр. Той не познава аналитичната геометрия, модерната алгебра и определено няма представа за елиптичните криви. Но за първи път елиптичните криви се появяват в неговата книга „Аритметика“. Проблемът описан от Диофант е следния: ***„да се раздели дадено число на две числа така че тяхното произведение да е куб минус неговата страна“***. И уравнението, което е получил е $\mathbf{Y\,(a - Y) = X^3 - X}$, което всъщност е прикрита елиптична крива. Диофант решава уравнението по следния начин:

- Нека $\mathbf{a = 3}$ и извадим **9** от двете страни на уравнението, получава се $\mathbf{6Y - Y^2 - 9 = X^3 - X - 9}$. Заместваме **Y** с **y+3 и X** с **–x** и се получава $\mathbf{y^2 = x^3 - x + 9}$;
- Диофант решава проблема за a = 6 като замества $\mathbf{X = 3Y - 1}$ игнорирайки факта, че има двоен корен, получава $\mathbf{y = 26/27}$ и $\mathbf{x=17/9}$. Следователно двете числа са $\mathbf{y = 26/27}$ и $\mathbf{a - y = 136/27}$. И произведението на тези две числа е $\mathbf{(17/9)^3 - (17/9)}$.

Истинската значимост на постигнатото от Диофант в тази част от книгата му отнема 1500 години за да се разбере и приложи.

Елиптичните криви се засягат слабо през 8-ми век и Фибоначи ги прави известни през 18-ти век. Той се сблъсква с проблема да намери рационално число **r** такова, че $\mathbf{r^2 - 5}$ и $\mathbf{r^2 + 5}$ да са рационални квадрати. Фибоначи нарича положителното число **n** конгруентно ако $\mathbf{r^2 - n}$, $\mathbf{r^2}$, $\mathbf{r^2 + n}$ са всички ненулеви квадрати за някое рационално число **r**. Връзката с елиптичните криви е, че ако **n** е конгруентно число, тогава произведението

на трите ненулеви рационални квадрати $\mathbf{r^2 - n}$, $\mathbf{r^2}$, $\mathbf{r^2 + n}$ е също рационален квадрат. Ако положим $r^2 = x$ получаваме равенството $y^2 = x(x-n)(x+n)$, което представлява елиптична крива. Също така ако **n** е конгруентно число, то елиптичната крива съдържа ненулева рационално точка.

Френският математик Бахит (Bachet) превежда на латински книгата на Диофант „Аритметика“ и я публикува през 1621 г. Ферма се сдобива с копие през 1630 г. В своите разработки Ферма има няколко референции към проблеми свързани с елиптични криви. В частност неговото предположение, че единствените числа удовлетворяващи равенството $\mathbf{y^2 = x^3 - 2}$ са $\mathbf{(x, y) = (3, 5)}$ или $\mathbf{(3, -5)}$ и че единствените числа, които удовлетворяват равенството $\mathbf{y^2 = x^3 - 4}$ са $(x, y) = \mathbf{(2,2), (2, -2), (5, 11), (5, -11).}$

Ойлер използва копие от работата на Ферма и разширява обхвата на теорията на числата като тя получава статут на легитимен дял от математиката. Ойлер също така работи върху конгруентните числа и получава много резултати свързани с елиптичните интеграли.

През 1670-те Нютон използва наскоро разработените инструменти на аналитичната геометрия за да класифицира кубичните криви. Правейки това той изяснява мистерията около проблемът от „Аритметика“ на Диофант и теоремата на Бахит относно рационалните решения на елиптична крива.

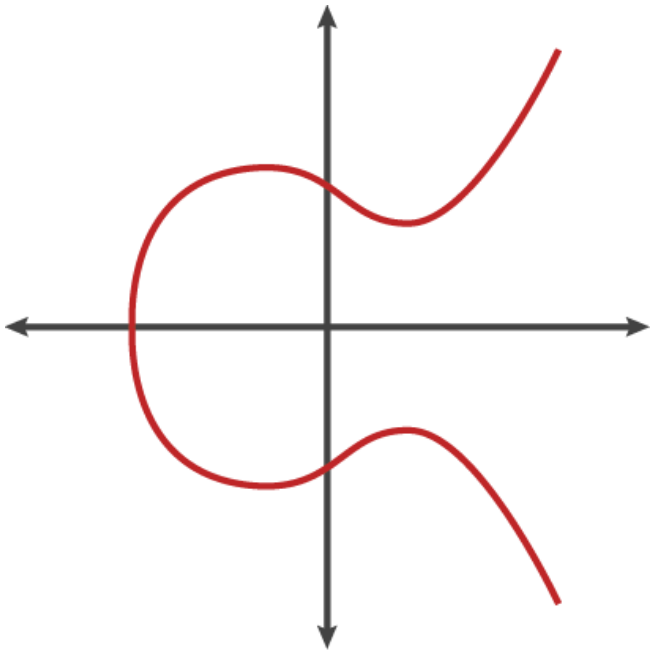
През 19-ти век Якоби и Вайършрас свързват тези открития с елиптичните интеграли и елиптичните функции. През 1901 г. Хенри Пойнкеър (*Poincare*) обединява и генерализира тези разработки в алгебрични криви.

Името ***„елиптичен“*** е дадено поради фактът, че тези криви възникват при изучаването на проблема за намирането на дължина на дъгата на елипса. Ако вземем интегралът, който дава дължината на дъгата на елипса и направим елементарна субституция, подинтегралният израз ще съдържа квадратен корен от кубичен полином, което е наречено елиптична крива.

Елиптични криви – една по-добра „*trapdoor*“ функция

След откриването на RSA и Diffie–Hellman, учените изследват различни криптографски решения базирани на математиката в търсене на алгоритми различни от разлагането, които биха могли да служат за добра ***„trapdoor“*** функция. През 1985 г. Нийл Коблиц и Виктор Милър предлагат алгоритъм базиран на елиптичните криви.

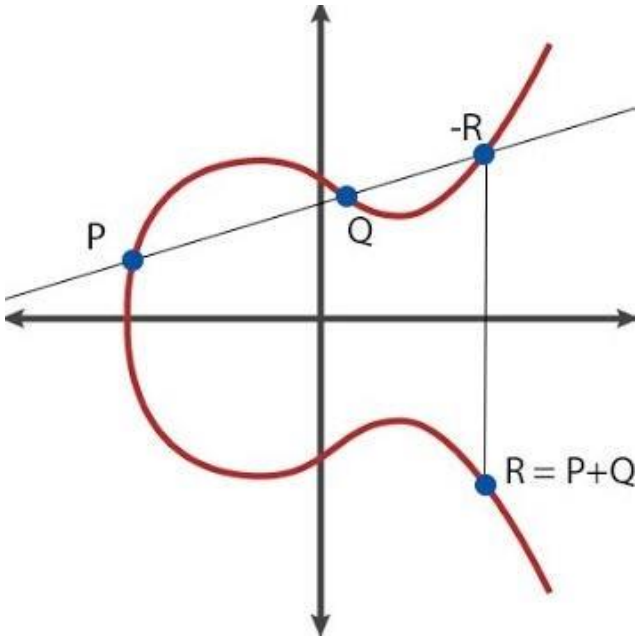
Елиптичната крива е множество от точки, които удовлетворяват конкретно математическо равенство от вида $\mathbf{y^2 = x^3 + ax + b}$. Графиката изглежда по следния начин:



Ако разгледаме по-подробно графиката, елиптичната крива има някой важни свойства. Едно от тях е хоризонталната симетрия. По интересното свойство е че всяка невертикална линия пресича графиката в най-много три точки.

Криптографията с елиптични криви използва „метода на Диофант“, където множество от точки се използват за получаването на нови точки. Аритметиката, използвана в метода на Диофант позволява събиране на

точки, удвояване на точки, добавяне на вертикални точки. Двете математически функции, които представляват основен интерес на криптографията с елиптични крива са събиране на точки и удвояване на точки. И двете функции използват геометрия в координатна система, за да изчислят наклона на линия срещу кривата (m) и да намерят решение за третата точка x от кривата. Решението се получава като заместим формулата за права линия **y = mx + c** в уравнението на кривата разгледано спрямо **x**. Така се получава **x3 = m² - (x1 + x2)**, където **x1** е абсцисата на точката **P**, а **x2** е абсцисата на точката **Q**.



Намирането на третата точка R става като се отрази пресечната точка с кривата **- R**.

Криптографска система с елиптични криви може да бъде дефинирана като се избере просто число за максимум, равенство определящо кривата, и публична точка върху кривата. Частният ключ е число (priv), а публичният е публичната точка към, която е приложена операцията описана по-горе priv пъти.

Дискретен логаритъм на елиптична крива (*Elliptic curve discrete logarithm, ECDL*) е сложният проблем, който стои в основата на криптографията с елиптични криви. Въпреки вече няколко десетилетия на изучаване все още не е открит алгоритъм, който да решава проблема и да има подобрение спрямо първоначалния подход. С други думи за разлика от проблемът за разлагането на числа, в този случай базирайки се на знанията за математиката, които имаме към момента няма начин, по който да се намали разликата в нарастването на двете операции. Това означава, че за числа с еднакъв размер, решаването на дискретен логаритъм на елиптична крива е значително по-трудно от разлагането. Тъй като по сложен проблем означава по-силна криптографска система, следва, че криптографските системи с елиптични криви са по-трудни за разбиване от RSA и Diffie–Hellman.

За да визуализира колко по-трудно е за разбиване Ариен Ленстра (*Arjen Lenstra*) е разработил концепция за „Глобална сигурност“. Посредством нея може да се изчисли колко енергия е необходима за разбиването на криптографски алгоритъм и да сравни това с енергията необходима за завиране на определено количество вода. Според тази мярка счупването на 228 битов RSA ключ изисква по-малко енергия отколкото трябва за завирането на една чаена лъжица вода. От друга страна счупването на 228 битов ключ за алгоритъм с елиптични криви изисква достатъчно енергия, за да заври цялата вода на земята. За същото ниво на сигурност с RSA е необходим ключ с дължина 2380 бита.

С елиптични криви могат да се използват по-малки ключове за постигане на същото ниво на сигурност. Малките ключове са важни особено в свят, в който все повече криптографски услуги се извършват върху устройства с ограничени ресурси.

Математически бележки

От математическа гледна точка е целесъобразно да се отбележи, че:

- Елиптичната крива образува алгебрична група, която се конструира спрямо геометричните операции;
- Елиптичните криви могат да съдържат точки с координати от всяко поле, примерно Fp, Q, R, C;
- Елиптичните криви с точки от Fp са крайни групи;
- Най-добрият известен алгоритъм за решаване на ECDLP има сложност върху Fp по време O(√p). Поради тази причина дължината на ключа трябва да е два пъти по-голяма от нивото на сигурност, което искаме да постигнем;
- В общия случай най-добрия алгоритъм за решаване на ECDLP има експоненциална сложност;
- Има изискване дискриминантата на уравнението на кривата да е ненулева;
- При конструирането на групата се добавя допълнителна точка O, за стойности в безкрайността;
- Теорема на Пойнкеър – нека K е поле и елиптична крива E е дадена с уравнението E: $y^2 = x^3 + Ax + B$, A и B принадлежат на K. Нека $E(K) = \{(x,y) \in E : x, y \in K\} \cup \{O\}$. Следователно E(K) е подгрупа на групата от всички точки от E. Това свойство е важно, защото дава възможност да се използват твърдения доказани за E върху E(K);
- Върху крайно поле групата от точки E(Fp) е циклична или е произведение на циклични групи;
- Групата E на всички точки от елиптична крива е абелева група;
- Има приблизително 2p различни елиптични криви дефинирани върху Fp;

- От практическа гледна точка компютрите работят по-ефективно с полета с характеристика 2 отколкото с такива, имащи характеристика голямо просто число. Поради тази причина криптографските често използват елиптични криви дефинирани върху поле Fq имащо q=2k елемента.

4.8. КРИПТОГРАФСКИ СИСТЕМИ И АЛГОРИТМИ ВЪРХУ ЕЛИПТИЧНИ КРИВИ

Криптографията с елиптични криви предоставя няколко групи системи базирани на математиката на елиптични криви върху крайни полета:

- Дигитални подписи: ECDSA (за класически криви) и EdDSA (за кръстосани криви на Едуардс);
- Хибридни системи: ECIES, EEECC;
- Системи за договаряне на ключове: ECDH, X25519 и FHMQV.

4.8.1. ECDSA (*Elliptic Curve Digital Signature Algorithm*)

Генериране на подпис

Да предположим, че Алис иска да изпрати подписано съобщение на Боб. Първоначално, те трябва да договорят параметрите на кривата. В допълнение към равенството на кривата и полето е необходима точка **G** от прост ред върху кривата и число **n** самия ред. Знаем, че **n** задължително е просто число.

Алис създава двойка ключове състояща се от частен ключ – число $\mathbf{d_A}$, случайно избрано в интервала **[1, n - 1]** и публична точка върху кривата $\mathbf{Q_A = d_A \times G}$. Операторът **x** се използва за означаване на умножение със скалар в елиптична крива. За да подпише съобщение **m**, Алис изпълнява следните стъпки:

- Изчислява **e = HASH(m),** където **HASH** е хеш функция, примерно SHA-2;
- Нека **z** съдържа $\mathbf{L_n}$ най-леви бита на e, където $\mathbf{L_n}$ е дължината в битове на реда на групата **n** (така **z** може да е по-голямо число от **n**, но не и по-дълго);
- Избира се случайно число **k** в интервала **[1, n-1];**

- Изчислява точката от кривата **(x1, y1) = k x G;**
- Изчислява **r = x_1 mod n**. Ако **r = 0**, се връща на стъпка 3;
- Изчислява **s = k^{-1} (z + rd_A) mod n**. Ако **s = 0** се връща на стъпка 3;
- Подписът е двойката **(r, s).** Също така **и (r, -s mod n)** е валиден подпис.

Важно е да се отбележи, че **k** не само трябва да е тайна, а и също за всеки подпис трябва да се избира различно **k**. В противен случай равенството от стъпка 6 може да бъде решено за d_A и така да се разкрие частния ключ. Използвайки подобна уязвимост, хакери успяват да извлекат подписващия ключ на PlayStation 3. Друга уязвимост, която често се използва за разкриване на частните ключове, е слаб генератор на случайни числа.

Алгоритъм за верификация на подписи

За да провери подписа на Алис, Боб има нужда от копие на нейната публична точка от кривата Q_A. Боб може да потвърди че Q_A е валидна точка по следния начин:

- Проверява, че Q_A не е равна на идентичния елемент **O** и нейните координати са валидни;
- Проверява, че Q_A лежи на кривата;
- Проверява, че **n x Q_A = O.**

След това Боб следва следните стъпки:

- Проверява, че **r** и **s** са числа в интервала **[1, n-1]**. Ако не са, подписът е невалиден;
- Изчислява **e = HASH(m)** където **HASH** е същата функция използвана в генерирането на подписа;
- Нека **z** са най-левите L_n бита;
- Изчислява **u_1 = zs-$^{-1}$ mod n** и **u_2 = rs^{-1} mod n;**

- Изчислява точка от кривата $(x_1, y_1) = u_1 \times G + u_2 \times Q_A$. Ако $(x_1, y_1) = O$ подписът е невалиден;
- Подписът е валиден ако **r** и x_1 дават един и същ остатък **по mod n.** В противен случай е невалиден.

Сигурност на подписа

През декември 2010 г., хакерската група „ailOverflow“ обявява, че е успяла да разкрие частния ключ използван за подписване на игровите конзоли. Въпреки това атаката е била възможна, защото Sony не са имплементирали правилно алгоритъма – k вместо случайно число е било статично.

На 29 март 2011 г. двама изследователи публикуват документ, в който демонстрират, че е възможно да се получи частния ключ за TLS от сървър използващ Open SSL и ECDSA върху двоично поле с помощта на тайминг атака. Уязвимостта е отстранена в следващи версии на OpenSSL.

През август 2013 г. е разкрита грешка (пропуск) в имплементацията на класът SecureRandom в Java, което води до колизии при избора на случайното число k. Това дава възможност на атакуващите да намерят частен ключ, с който да получат достъп до биткойни също както техните собственици имат.

В заключение въпреки добрите си криптографски характеристики алгоритъмът е труден за имплементиране, което понякога води до пропуски в реализацията, които могат да бъдат атакувани.

4.8.2. Размяна на ключове с ECDH

ECDH (*Elliptic Curve Diffie–Hellman Key Exchange*) е система за договаряне, която позволява двете страни да договорят споделен ключ използвайки незащитен канал. Алгоритъмът е много подобен на

класическия DHKE(*Diffie–Hellman Key Exchange*) с разликата, че се използва умножение на точки от елиптична крива.

Нека Алис иска да договори споделен ключ с Боб. И двамата разполагат с двойка публичен / частен ключ подходящи за криптография с елиптични криви. Нека бележим частните ключове с **d** (случайно избрани числа в интервала **[1, n-1]),** а публичните с **Q** (където **Q = d . G**, което е резултат от добавянето на G към себе си d пъти). Двойката ключове на Алис е **(d_A, Q_A),** а на Боб е **(d_B, Q_B).** Всяка от страните трябва да знае публичния ключ на другата страна преди изпълнението на протокола.

Протоколът за установяване на ключ е:

- Алис намира точка **$(x_k, y_k) = d_A . Q_B$**;
- Боб намира точка **$(x_k, y_k) = d_B . Q_A$**;
- Споделения ключ е **x_k.**

Полученият споделен ключ е еднакъв защото:

$d_A . Q_B = d_A . d_B .G = d_B . d_A . G = d_B . Q_A$.

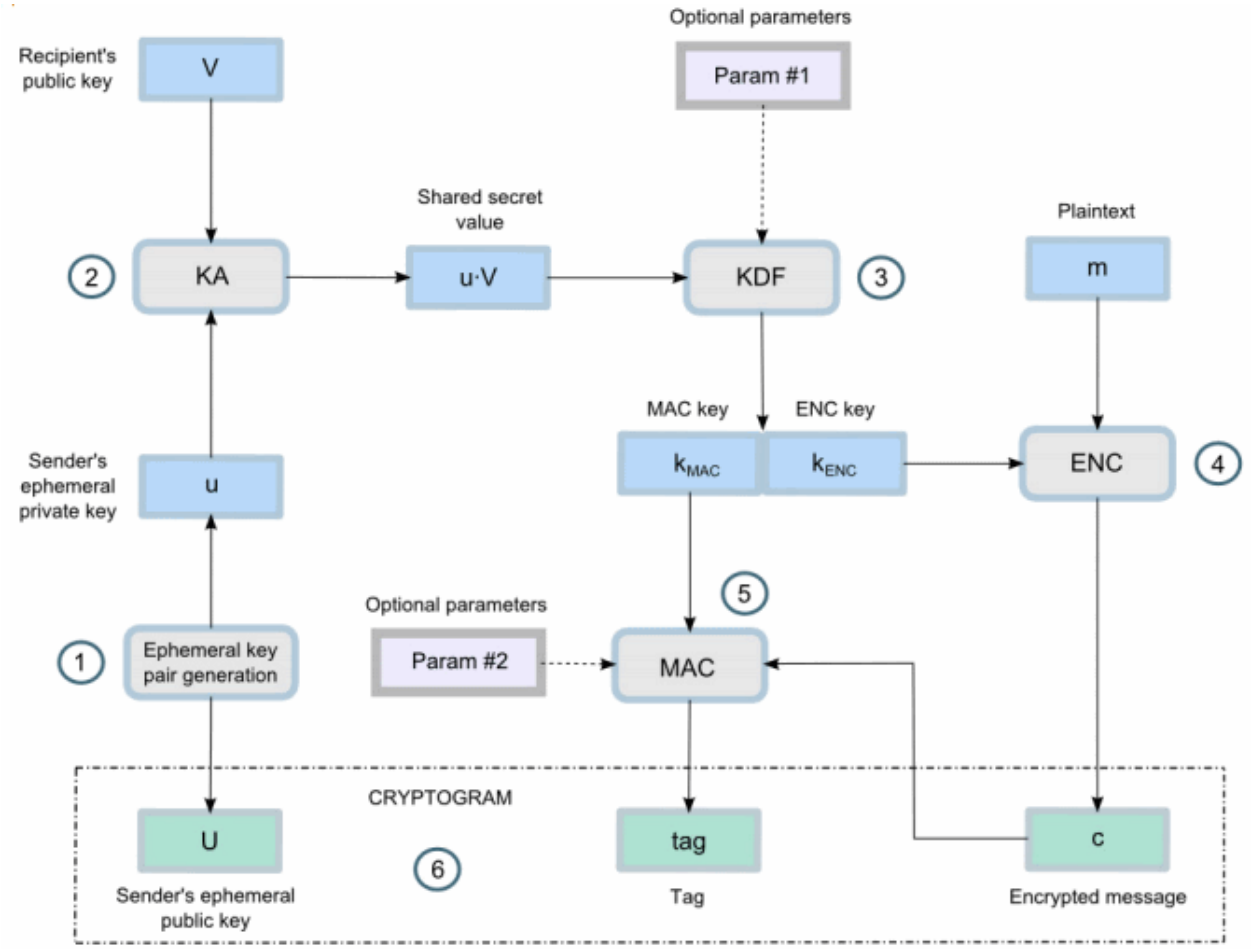
Единствената информация, която Алис разкрива е нейният публичен ключ, така че никой освен Алис не може да определи нейния частен ключ освен ако не реши проблемът за намиране на дискретен логаритъм на елиптична функция. Аналогично и за частния ключ на Боб.

4.8.3. Хибриден алгоритъм ECIES

ECIES е криптографска система с публичен ключ, която използва KDF(*key derivation function*) за да извлече отделен MAC ключ и симетричен криптиращ ключ от общия ключ на ECDH. Стандартът ECIES комбинира ECC базирани асиметрични криптографии със симетрични шифри, за да предостави криптиране на данните с ЕС частен ключ и тяхното декриптиране с ЕС публичен ключ. Криптографската система ECIES използва ECC криптография + KDF функция + алгоритъм за симетрично

криптиране + MAC алгоритъм. На входа се подават публичният ключ на получателя и съобщението като текст. Изходът съдържа кратковременния публичен ключ на изпращача, криптираното съобщение и тагът за автентификация (MAC code). Процесът на декриптиране приема резултата от криптирането и частния ключ на получателя като връща оригиналния текст или грешка ако има проблем с автентификацията например.

Криптографската система ECIES е работна рамка, а не конкретен алгоритъм. Може да се имплементира като се използват различни алгоритми примерно secp256k1 или P-521 елиптични криви, PBkDF2 или Scrypt за KDA функцията, AES-CTR или AES-GCM а симетричния шифър и системите за автентификация HMAC-SHA512 за MAC алгоритъма.



Фигура 4.44. Работна рамка на ECIES

NSA Suite B

NSA Suite B е набор от криптографски алгоритми разпространяван от Националната агенция са сигурността (NSA) като част от програмата й за модернизация в криптографията. Използва се както за некласифицирана информация така и за повечето от класифицираната информация. В случаите когато Suite B не е приложим се използва Suite A, като участващите в него алгоритми и технологии не са напълно известни. Suite B е анонсиран на 16 февруари 2005 година. Съдържа следните компоненти:

- Advanced Encryption Standard (AES) с големина на ключовете 128 и 256 бита. Използва се заедно с Counter Mode (CTR) за канали с малка ширина на трафика или с Galois/Counter Mode (GCM) в останалите случаи;
- Elliptic Curve Digital Signature Algorithm (ECDSA) – за дигитални подписи;
- Elliptic Curve Diffie–Hellman (ECDH) – за договаряне на ключове;
- Secure Hash Algorithm 2 (SHA-256 и SHA-384) – когато е необходимо използването на хеш функция.

През август 2015, NSA обяви, че планира миграция към нов набор от алгоритми в близко бъдеще, който да е устойчив на атаки с квантови компютри. Очаква се новият стандарт да бъде публикуван около 2024 година.

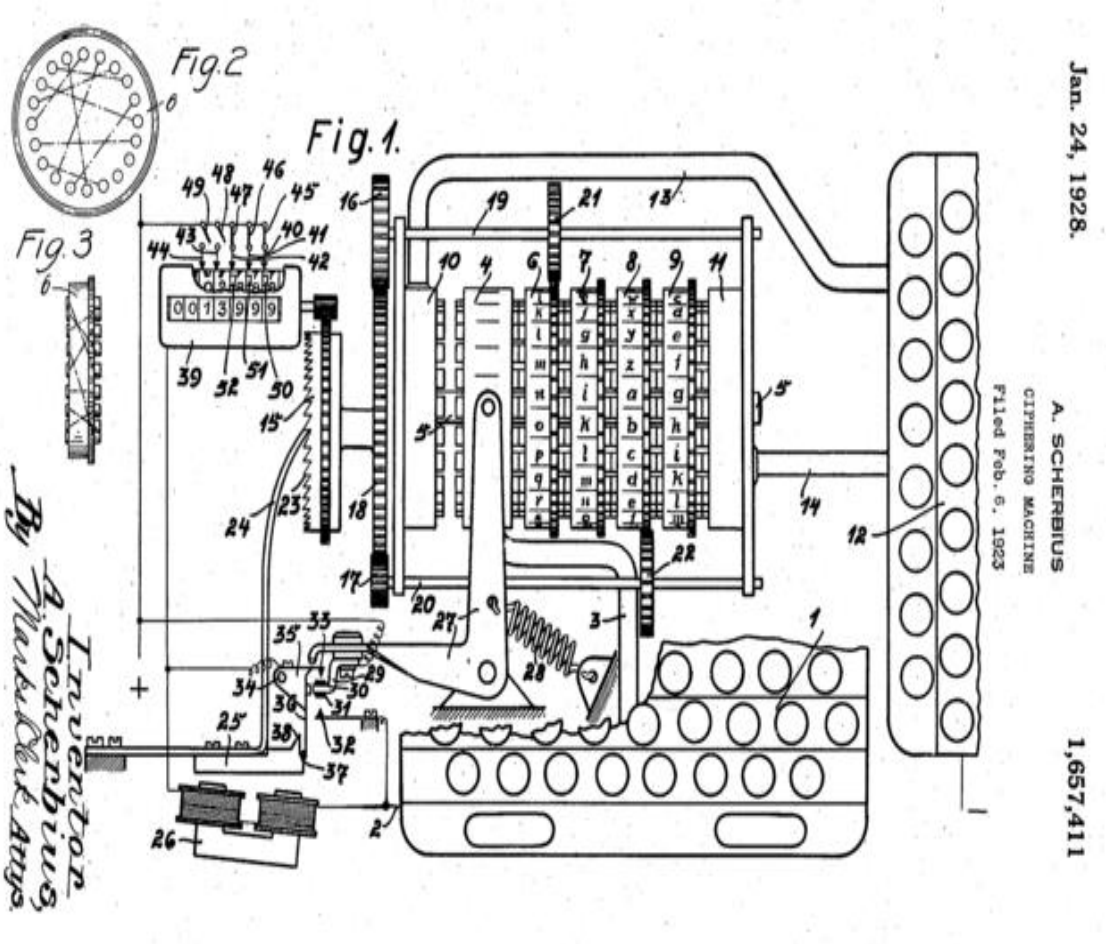
ПРИЛОЖЕНИЕ 1. ШИФРОВА МАШИНА ЕНИГМА

В Приложението е описана шифровата машина ЕНИГМА.

1. Увод

Машините Енигма са били серия от електро-механични шифриращи машини разработени и използвани в началото и средата на XX век за търговски цели, както и във военните и държавни служби и учреждения на много страни, но най – широко разпространение получава в Националсоциалистическа Германия по време на Втората световна война. Затова под „Енигма“ най - често се разбира немският военен модел Енигма на Вермахта (*Wehrmacht Enigma*).

Думата „Енигма“ идва от латинската дума aínigma и означава „загадка“. За изобретател на Енигма се счита немския електроинженер д-р Артур Шербиус, който на 23 февруари 1928 година получава първи патент за шифриращата машина, а за производството на машината на 9 юли 1923 година. На ***Фигура .1*** можем да видим как е изглеждал патентът.



Фигура 1. Патентът, който Артур Шербиус получава за Енигма

На ***Изображение 1.*** можем да видим една истинска Енигма.

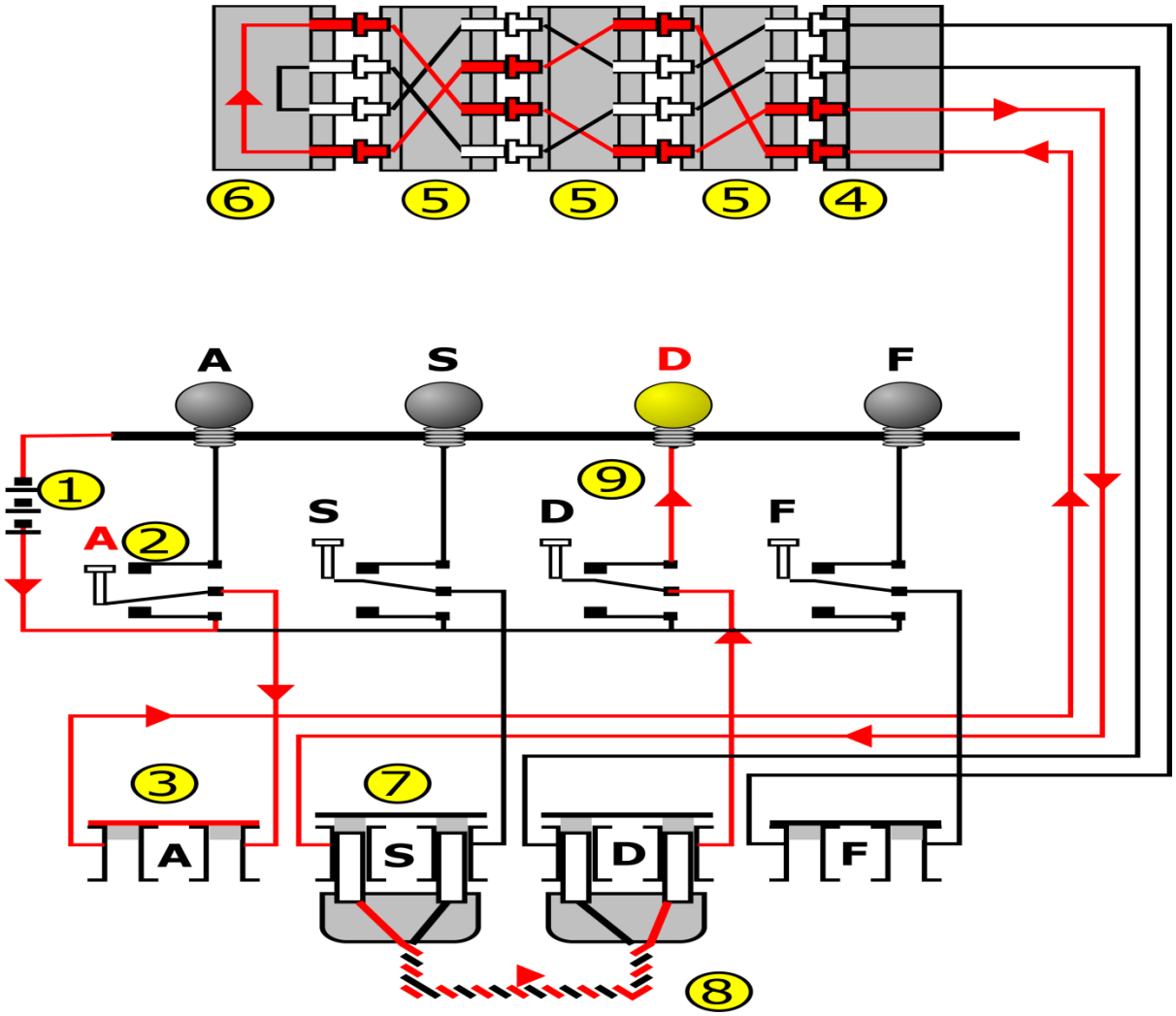
Изображение 1. Трироторна военна машина Енигма

Различните специалисти сочат различен брой машини произведени по време на Втората световна война: между 30 000 и 200 000 бройки. До края на войната се сменят различни модели и варианти на Енигма. Най-използваният обаче е т.н. Енигма-I („Енигма едно“), който от 1930 година е на въоръжение в Райхсвера, а по-късно и във Вермахта.

2. Описание на Енигма

Машината Енигма-I тежи около 10 килограма и е с размери 310 мм x 255 мм x 130 мм. На пръв поглед прилича на пишеща машина. Както и другите роторни машини, Енигма се състои от комбинация от механически и електрически системи. Механичната част включва клавиатура, набор от въртящи се дискове (ротори, *rotors*), които са разположени около ос (вал, *spindle*) и са долепени до него и механизъм от степени, който задвижва един или повече ротора при всяко натискане на клавиш. Също така има и рефлектор, който връща сигнала пак през роторите. Конкретният начин на работа може да е различен, но общият принцип е един и същ: при всяко натискане на клавиш най – десният ротор се измества с една позиция, а при определени условия се изместват и другите ротори. Движението на роторите води до различни криптографски преобразования при всяко следващо натискане на клавиш на клавиатурата.

Механичните части се движат, образувайки променяща се електрическа верига, тоест фактически шифрирането на буквите се осъществява електрически. При натискането на клавиш веригата се затваря, токът преминава през различните компоненти и накрая включва една от множеството лампи, изобразяваща буквата, която ще излезе на изхода. Например, при шифриране на съобщение, започващо с DIMITAR.. операторът първо натиска клавиш D святка например лампа C, т.е. буквата C става първата буква от криптирания текст. На ***Фигура 2***. е показана електрическата схема на машината, но опростена до азбука с 4 елемента = {A, S, D, F}.



Фигура 2. Опростена електрическа схема на Енигма

Функциониране на Енигма

Стъпките на работа са следните:

- Токът преминава от батерията (1) през превключвателя (2) в комутационния панел (3);
- Комутационният панел позволява сигналът да премине от клавиатурата (2) до неподвижното входно колело (4);
- След това токът преминава през щепсел (3), в конкретния пример не е използван, входното колело (4) и схемата на съединение на три (в армейския модел) или четири (във военноморския модел) ротора (5), откъдето влиза в рефлектора (6);

- Рефлекторът връща тока обратно по друг път през роторите (5) и входното колело (4);
- След това тока минава през щепсел S , съединен чрез кабел с щепсел D, през друг двупосочен превключвател (9), като захранва лампата.

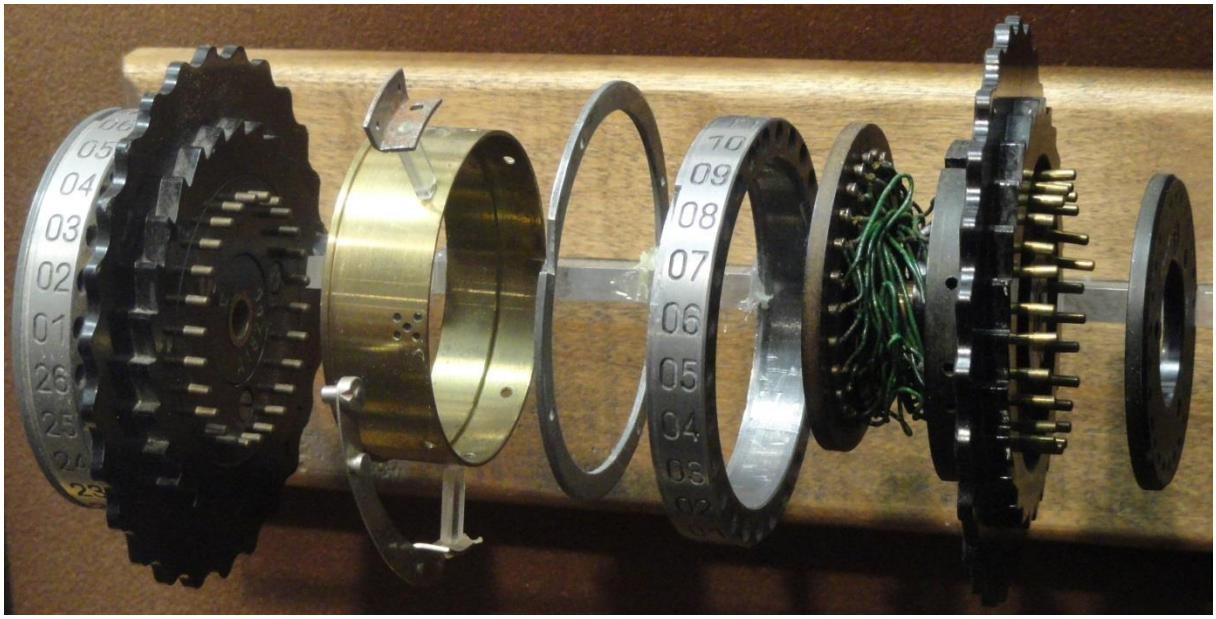
Това постоянно изменение на електрическата верига, през която минава електричеството позволява да се постигне много буквен шифър, който е сравнително високоустойчив за времето си. Ако операторът изпише „DIMITAR“, ще се получи нещо от сорта на „RPWHYRY“. Едно интересно наблюдение, което можем да направим е, че еднаквите букви I отиват в различни – P и H, а буквите D и A отиват в еднакви R. Точно поради тази причина, германците са смятали, че имат неразбиваема шифрираща машина, но малко по – нататък ще разгледаме какъв е бил проблемът с Енигма.

2.1 Ротори

Роторите са ядрото на Енигма. Всеки ротор представлява диск с диаметър около 10 сантиметра, изработен най – често от твърда гума с пружини контактори от едната страна на ротора, разположени в кръг. От другата страна има съответното количество плоски електрически контакти. Всяка двойка съответства на буква от азбуката, обикновено 26-те букви A – Z или просто числата 1 – 26. При допир контактите на съседни ротори затварят електрическата верига. Вътре в ротора всеки контактор е съединен с друг от плоските контакти. Редът на съединяване може да е произволен.

Роторът сам по себе си извършва най – простия тип криптиране – замяна на една буква с друга. Например контактът, отговарящ на буквата „E“ може да е свързан с контакта на буквата „T“ от другата страна на ротора и тогава всички букви „E“ в шифрирания текст биха били заменени с „T“,

но при Енигма са се използвали няколко взаимно свързани ротора (обикновено три или четири), при тяхното въртене се получава наистина надеждно шифриране. На ***Изображение 2.*** можем да видим един разглобен ротор изложен в Националния Криптографски музей в Америка.



Изображение 2. Разглобен ротор от машината Енигма

На ***Изображения 3. и 4.*** можем да видим и как изглеждат пластините.

Роторът може да заема една от 26 възможни позиции в машината. Той може да бъде завъртян до друга позиция ръчно посредством назъбения роторен палец, който се подава навън. За да може операторът винаги да определя положението на ротора, на венеца на всеки от тях в кръг са изписани буквите от азбуката и само една от тях се вижда през прозорчето. В първите модели Енигма азбучното колело било фиксирано, а в по-късните версии конструкцията била усложнена и то можело да се регулира. Всеки ротор имал една или няколко зъба, използвани за управление на движението на роторите. Във военните версии зъбите били разположени на азбучното колело.

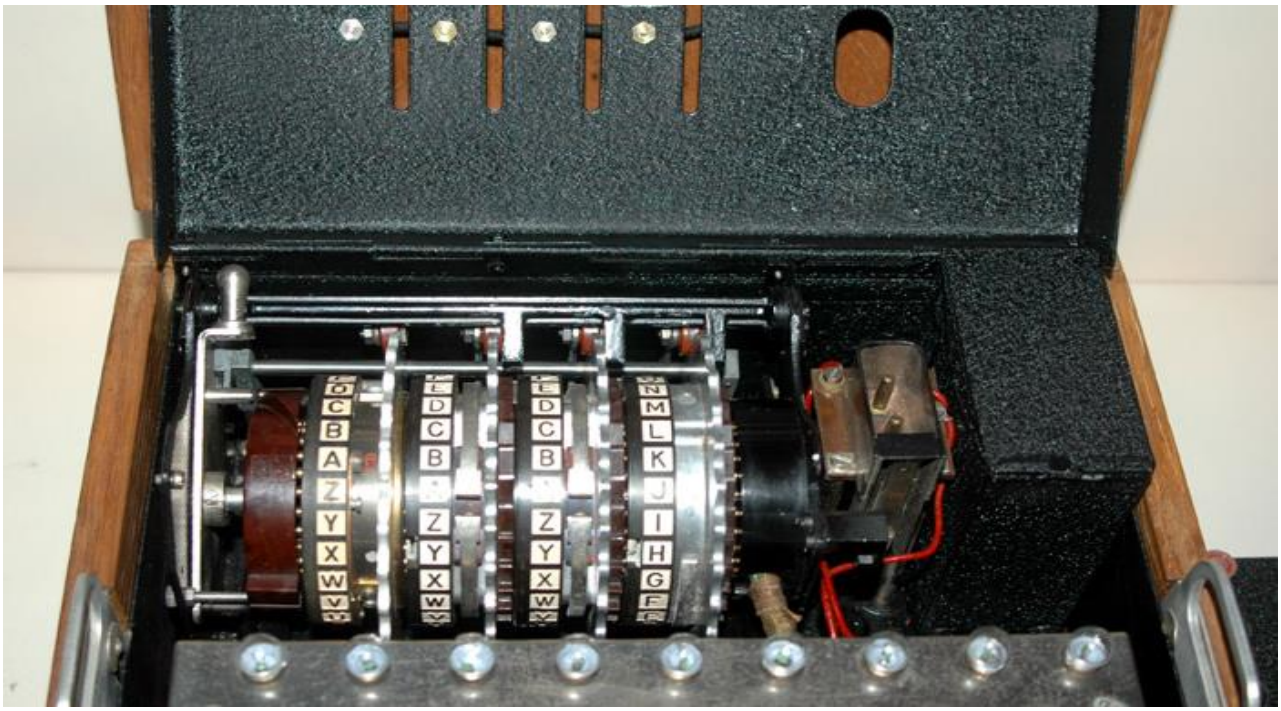
Изображение 3. Лявата страна на ротора на Енигма. Виждат се плоските електрически контакти

Изображение 4. Дясната страна на ротора на Енигма. Виждат се конекторите. Римското V показва проводника в ротора

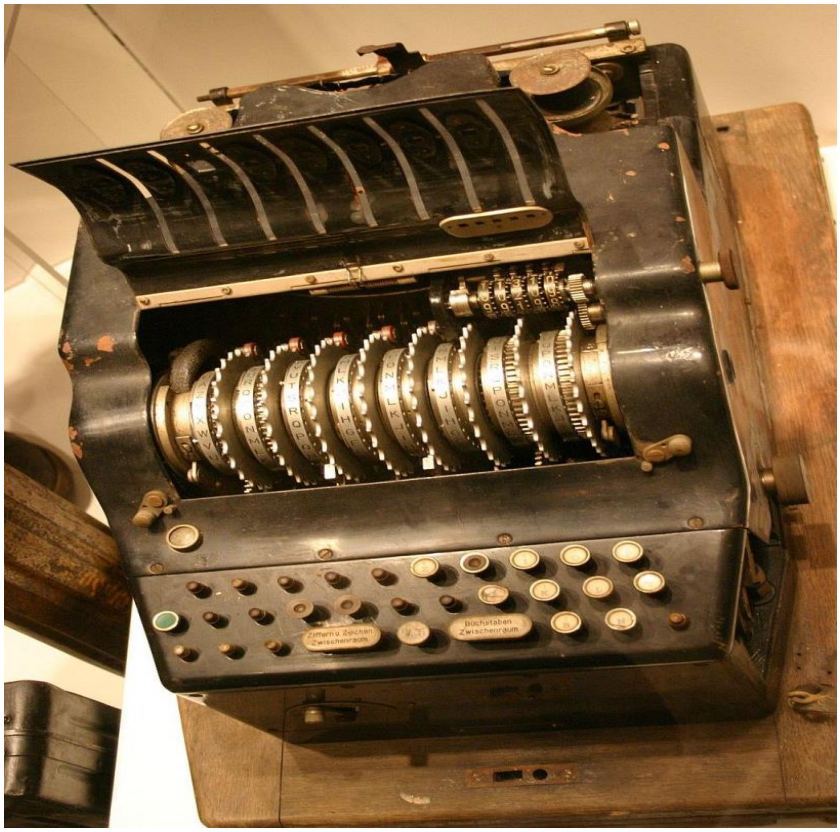
Военните модели Енигма се произвеждали с различен брой ротори. Първият модел имал едва три, от 15 декември 1938 година те станали пет, но само три от тях се използвали едновременно в машината. Тези ротори били маркирани с римските числа от I до V, и всеки от тях имал по един зъбец, разположен на различни места в азбучното колело. Военноморските модели винаги имали повече ротори, отколкото другите – по шест, седем или дори осем. Тези допълнителни ротори били маркирани с числата VI, VII и VIII, като всеки от тях имал различни електропроводници. Имали по два зъбеца около буквите 'N' и 'A', поради което се завъртали по-често.

Четири роторният военноморски модел Енигма-M4 имал един допълнителен ротор, въпреки че бил с размерите на три роторен. Това се постигало с по-тънък рефлектор. Имало два типа допълнителен ротор – Бета и Гама. Те не се движели в процеса на шифриране, но можели да бъдат

преместени ръчно на която и да е от 26 различни позиции. На ***Изображения 5. и 6.*** можем да видим Енигма със съответно 4 и 8 ротора.



Изображение 5. Машина Енигма с 4 ротора



Изображение 6. Машина Енигма с 8 ротора

2.2 Стъпково движение

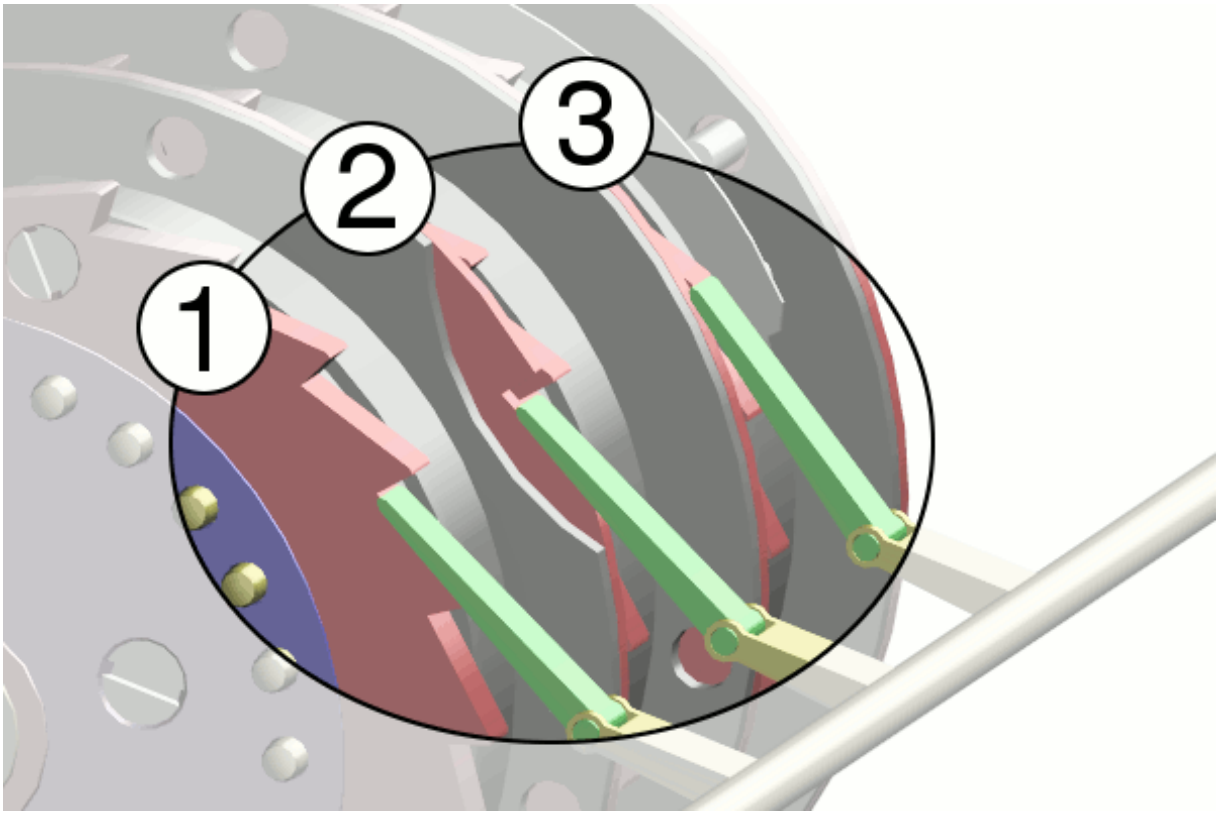
Всеки ротор бил прикрепен към зъбно колело с 26 зъба (зъбен запиращ механизъм), а групата езичета се зацепвала в зъбите на това колело. Езичетата се изместали напред едновременно с натискането на клавиш на машината. Ако езичето зацепвало зъб на запиращия механизъм, роторът се завъртал с една стъпка.

В армейския модел на Енигма всеки ротор бил прикрепен към регулируемо колело със зъбци. Петте базови ротора (I-V) имали по един зъбец, докато военноморският модел (VI-VIII) имал по два. В даден момент зъбецът попадал срещу езичето, позволявайки му да се зацепи за зъбния запиращ механизъм на следващия ротор при последващото натискане на клавиш. Когато обаче езичето не попадало в зъбец, то просто се плъзгало по повърхността на колелото, без да се зацепва за механизма. В система с един зъбец вторият ротор се изместал напред с една позиция за същото време, за което първият се изместал с 26 позиции. Аналогично третият се преместал с една позиция за времето, за което втория правел 26 стъпки. Специфична особеност на машината била, че вторият ротор също се местел, ако се преместал третия. Това означава, че втория ротор може да се премести два пъти при две последователни натискания на клавиши – така нареченото „двустъпково движение“, което водело до намаляване на периода.

Двустъпковото движение различава работата на роторите от тази на обикновения километражен брояч. Двойната стъпка действала по следния начин: първият ротор се завъртал, карайки втория също да направи една стъпка. И ако втория попаднел в необходимата позиция, третото езиче се зацепвало в третия запиращ зъбен механизъм. На следващата стъпка това

езиче бутало запъващия механизъм и го преместало, преместваики същевременно и третия ротор.

С три диска, имащи само по един зъбец на първия и втория диск, машината имала период на повторение 26 * 25 * 26 = 16 900 символа. Като правило за сигурност, съобщенията изпратени от германците били по-къси от 250 символа, поради което нямало риск да се повтори една и съща позиция на роторите при едно и също съобщение. В четири роторните военноморски модели нямало изменения в механизма. Четвъртият ротор не се въртял, но можел да бъде сложен ръчно на която и да е от наличните 26 позиции. На ***Фигура 3.*** можем да видим това движение.



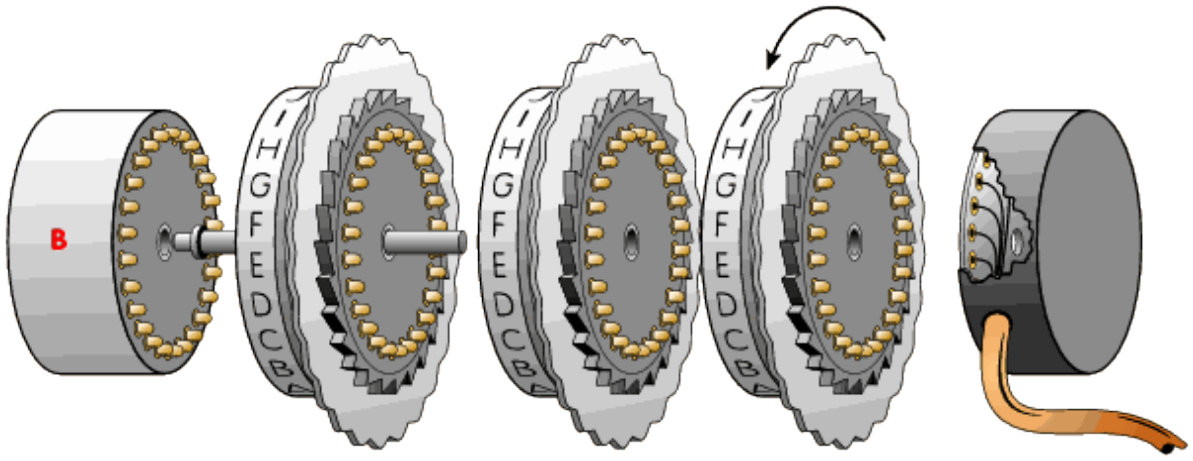
Фигура 3. Стъпково движение при Енигма

Стъпаловидно движение на роторите на Енигма. И трите езичета (обозначени със зелен цвят) се движат едновременно. За първия ротор (1), зъбният запиращ механизъм (в червено) е винаги застопорен и роторът се

завърта при всяко натискане на клавиша. В този случай зъбецът на първия ротор позволява на езичето да се зацепи и за втория ротор (2), и той ще се завърти при следващото натискане на клавиш. Третият ротор (3) не е зацепен, тъй като езичето му не е попаднало в зъбеца на втория и просто ще се плъзга по повърхността на диска.

2.3 Входно колело

Входното колело (на немски Eintrittswalze, съкратено ETW), или входният стартер, съединявало комутационния панел или клавиатурата и ламповия панел с роторите. Той е важна част от машината Енигма, въпреки че фиксираното свързване на проводниците имало сравнително малка роля в осигуряването на сигурността на криптограмите. На ***Фигура 4.*** най – вдясно се вижда тази част.



Фигура 4. Най – вдясно се намира входното колело, а най – вляво рефлекторът

2.4 Рефлектор

С изключение на ранните модели A и B, след последния ротор бил разположен *рефлекторът* (на немски Umkehrwalze, съкратено UKW),

патентован детайл, отличаващ машините Енигма от другите роторни машини, разработени по това време. Рефлекторът съединявал контактите на последния ротор по чифтове, комутирайки ток през роторите в обратно направление, но по друг маршрут. Наличието на рефлектор гарантирало, че преобразуването, което извършва Енигма е инволюция, т.е. дешифрирането е същото, което е и шифрирането. Наличието на рефлектора обаче правело невъзможно която и да е буква да бъде шифрирана чрез самата себе си. Това бил сериозен концептуален недостатък и основният проблем на Енигма. Ще го разгледаме по – подробно в следващите точки.

В търговския модел на Енигма-С рефлекторът можел да бъде поставян в две различни позиции, а в модел D – в 26 възможни позиции, но в самия процес на шифриране бил неподвижен. В модела, използван от Абвера (германското контраразузнаване), рефлекторът се движел по време на шифриране, както и останалите дискове.

В армейския и военновъздушния модели на Енигма рефлекторът бил фиксиран и не можел да се върти. Имало четири разновидности. Оригиналната версия била белязана с *A* и била заменена с *Umkehrwalze B* на 1 ноември 1937 година Третата, *Umkehrwalze C* била използвана за кратко през 1940 година, вероятно поради проблеми. Четвъртата, *Umkehrwalze D,* появила се на 2 януари 1944 година, позволявала на оператора на Енигма да управлява настройките на комутация вътре в рефлектора.

В ***Таблица .1*** можем да видим тогавашната строго секретна схема на запояване на жичките на роторите на Енигма-I.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
I	E	K	M	F	L	G	D	Q	V	Z	N	T	O	W	Y	H	X	U	S	P	A	I	B	R	C	J
II	A	J	D	K	S	I	R	U	X	B	L	H	W	T	M	C	Q	G	Z	N	P	Y	F	V	O	E
III	B	D	F	H	J	L	C	P	R	T	X	V	Z	N	Y	E	I	W	G	A	K	M	U	S	Q	O
IV	E	S	O	V	P	Z	J	A	Y	Q	U	I	R	H	X	L	N	F	T	G	K	D	C	M	W	B
V	V	Z	B	R	G	I	T	Y	U	P	S	D	N	H	L	X	A	W	M	J	Q	O	F	E	C	K

Таблица 1. Начин на запояване на жиците на петте ротора на Енигма.

На ***Таблица 2.*** са показани и двата рефлектора B и C и коя буква в коя се отразява.

UKW B	**A –>**	**Y**	**B –>**	**R**	**C –>**	**U**	**D ->**	**H**	**E –>**	**Q**	**F –>**	**S**	**G –>**	**L**	**I –> P**		**J –> X**		**K –>**	**N**	**M –>**	**O**	**T –>**	**Z**	**V –>**	**W**
UKW C	A –>	F	B –>	V	C –>	P	D –>	J	E –>	I	G –>	O	H –>	Y	K –>	R	L –>	Z	M –>	X	N –>	W	Q –>	T	S –>	U

Таблица 2. Преобразование на буквите от рефлекторите B и C в Енигма

2.5 Комутационен панел

Комутационният панел (на немски Steckerbrett) позволява на оператора да променя свързването на проводниците. За първи път се появява в немските армейски версии през 1930 година и скоро успешно е приложен и във военноморските версии. Комутационният панел има огромен принос в усложняването на шифроването на машината. Според експерти-криптографи, усложняването е дори по-сериозно, отколкото с въвеждането на допълнителен ротор. При Енигма, без комутационен панел, разбивачът на кодове би се справил на практика ръчно, но след добавянето на панела, това можело да стане само с конструирането на специални машини.

Кабелът, разположен на комутационния панел, свързвал буквите две по две – например E и Q. Ефектът бил в размяната на тези букви преди и след преминаването на сигнала през ротора. Например, когато операторът натискал E, сигналът се отправял в Q и едва след това във входния ротор. Същевременно можело да се ползват няколко такива двойки (до 13, но обикновено се ползвали само 10 и се оставяли 3 двойки празни).

Всяка буква на комутационния панел имала две гнезда. Включването на щепсела разделяло горното гнездо от клавиатурата и долното гнездо (към входния ротор) на тази буква. Щепселът на другия край на кабела се слагал в гнездото на друга буква, като по този начин ги превключвал една към друга.

3. Процедури за настройка и използване на Енигма

В германските въоръжени сили средствата за свръзка били разделени в отделни мрежи, като всяка от тях имала собствени настройки за кодиране за машините Енигма. На всяка единица, работеща в мрежата за определен период от време били в сила определени настройки, които след това трябвало да бъдат променяни. За да бъдело съобщението правилно шифрирано и дешифрирано, машините на изпращащия и приемащия трябвало да бъдат настроени еднакво. Идентични трябвало да бъдат: изборът на ротори, началните позиции на роторите и връзките в комутационния панел. Тези настройки се уточнявали предварително и се записвали в специални шифрови книги. За по-късните версии, допълнителна настройка била и позицията на променящия се рефлектор.

Първоначално криптографският ключ на Енигма включвал следните параметри:

- Разположение на роторите (*Walzenlage*): избор на роторите и тяхното подреждане;

- Първоначални позиции на роторите: избрани от оператора, различни за всяко съобщение;
- Настройка на колелата (*Ringstellung*): избор на позиция на азбучното колело, съвпадаща с роторната схема;
- Настройка на щепселите (*Steckerverbindungen*): свързване на щепселите на комутационния панел.

Повечето от ключовете се съхранявали само за ограничен период от време, обичайно за денонощие, но за всяко ново съобщение се задавали нови начални позиции на роторите. Това се налагало, защото в случай на твърде много съобщения – надхвърлящи като брой символи периода на повторение на кодовите комбинации и изпратени с идентични настройки, криптоаналитикът получавал възможност чрез използване на честотен анализ да намери шифъра на съобщенията. Тези начални позиции се изпращали заедно с криптограмата, преди шифрирания текст. Този принцип се нарича „индикационна процедура“. Именно слабостта на подобни индикационни процедури довела до първите успешни случаи на разбиване на кода на Енигма.

Ранните индикационни процедури били използвани от полските криптоаналитици за разбиване на кода. При тези процедури операторът настройвал машината в съответствие със списъка настройки, които съдържат главните първоначални стартови позиции на роторите. Да предположим, че главната ключова дума е АОН. Операторът завъртал роторите ръчно, докато думата АОН не се изобрази на прозорчетата. След това операторът избирал свой собствен ключ за новото съобщение. Нека този ключ е думата EIN. Тази дума се въвеждала два пъти за да се избегнат грешки при предаването. Като резултат, след двойното въвеждане на АОН в криптограмата се изобразява думата XHTLOA, която предхождала тялото на основното съобщение. И накрая, операторът отново завъртал роторите в

съответствие с избрания ключ, в случая AOH, след което въвеждал основния текст на съобщението.

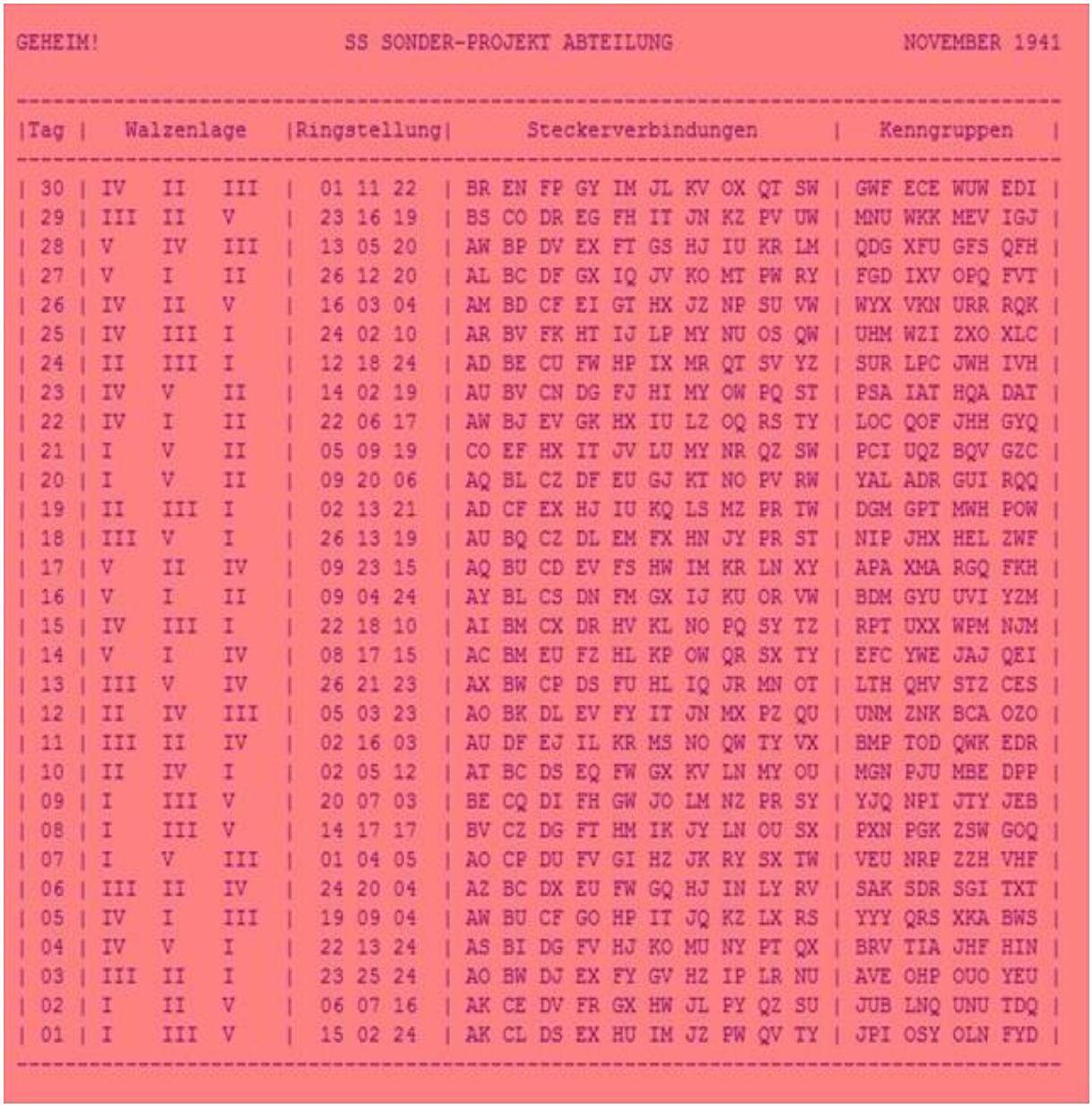
При получаване на това шифрирано съобщение, цялата операция се изпълнявала в обратен ред. Операторът-получател въвеждал в машината първоначалните настройки и отпечатвал първите шест букви от съобщението (в случая XHTLOA). Тогава лампичките примигвали в последователност EINEIN, тоест разбирал, че ключовата дума е EIN и че трябва да завърти роторите така, че да изобразят позиция EIN. След това отпечатвал основния текст, разшифровайки съобщението.

Този метод имал два недостатъка. Първият бил в използването на главни ключови настройки. Впоследствие това било изменено и операторът избирал собствени начални позиции и ги изпращал в не зашифрован вид. Вторият проблем бил в повторяемостта на избраната от оператора-шифровчик дума-индикатор, което било съществен пробив в сигурността. Ключът на съобщението се шифровал два пъти, в резултат на което се появявало закономерно сходство между първия и четвъртия, втория и петия и третия и шестия символи. Този недостатък позволил на полските дешифровчици да разбият кода на Енигма още през 1932 година. От 1940 година обаче немците променили процедурата, за да повишат сигурността.

По време на Втората световна война немските оператори използвали шифрова книга само за настройка на роторите и колелата. За всяко съобщение операторът избирал случайна стартова позиция, например WZA, и случаен ключ на съобщението, примерно SXT. Той завъртал роторите на стартова позиция WZA и шифровал ключа на съобщението SXT. Получавало се примерно UHL. След това операторът слагал думата SXT като начална позиция на роторите и шифровал съобщението. Изпращал стартовата позиция WZA, шифровия ключ UHL и основния текст. Получателят от своя страна слагал роторите на стартова позиция в съответствие с първата триграма WZA и разшифровал втората триграма

UHL, за да получи ключа на съобщението – SXT. Този ключ бил използван за разшифроване на текста. Така всеки път главния ключ се оказвал различен и бил премахнат недостатъка, присъщ на процедурата с двойно шифроване на ключа.

„Таблицата с ключове“ (***Изображение 7)*** представяла в табличен вид валидните за целия месец дневни ключове, които били сменяни в полунощ. Обратното подреждане позволявало вече използвания ключ да бъде изрязан и унищожен.



GEHEIM! SS SONDER-PROJEKT ABTEILUNG NOVEMBER 1941

Tag	Walzenlage	Ringstellung	Steckerverbindungen	Kenngruppen
30	IV II III	01 11 22	BR EN FP GY IM JL KV OX QT SW	GWF ECE WUW EDI
29	III II V	23 16 19	BS CO DR EG FH IT JN KZ PV UW	MNU WKK MEV IGJ
28	V IV III	13 05 20	AW BP DV EX FT GS HJ IU KR LM	QDG XFU GFS QFH
27	V I II	26 12 20	AL BC DF GX IQ JV KO MT PW RY	FGD IXV OPQ FVT
26	IV II V	16 03 04	AM BD CF EI GT HX JZ NP SU VW	WYX VKN URR RQK
25	IV III I	24 02 10	AR BV FK HT IJ LP MY NU OS QW	UHM WZI ZXO XLC
24	II III I	12 18 24	AD BE CU FW HP IX MR QT SV YZ	SUR LPC JWH IVH
23	IV V II	14 02 19	AU BV CN DG FJ HI MY OW PQ ST	PSA IAT HQA DAT
22	IV I II	22 06 17	AW BJ EV GK HX IU LZ OQ RS TY	LOC QOF JHH GYQ
21	I V II	05 09 19	CO EF HX IT JV LU MY NR QZ SW	PCI UQZ BQV GZC
20	I V II	09 20 06	AQ BL CZ DF EU GJ KT NO PV RW	YAL ADR GUI RQQ
19	II III I	02 13 21	AD CF EX HJ IU KQ LS MZ PR TW	DGM GPT MWH POW
18	III V I	26 13 19	AU BQ CZ DL EM FX HN JY PR ST	NIP JHX HEL ZWF
17	V II IV	09 23 15	AQ BU CD EV FS HW IM KR LN XY	APA XMA RGQ FKH
16	V I II	09 04 24	AY BL CS DN FM GX IJ KU OR VW	BDM GYU UVI YZM
15	IV III I	22 18 10	AI BM CX DR HV KL NO PQ SY TZ	RPT UXX WPM NJM
14	V I IV	08 17 15	AC BM EU FZ HL KP OW QR SX TY	EFC YWE JAJ QEI
13	III V IV	26 21 23	AX BW CP DS FU HL IQ JR MN OT	LIH QHV STZ CES
12	II IV III	05 03 23	AO BK DL EV FY IT JN MX PZ QU	UNM ZNK BCA OZO
11	III II IV	02 16 03	AU DF EJ IL KR MS NO QW TY VX	BMP TOD QWK EDR
10	II IV I	02 05 12	AT BC DS EQ FW GX KV LN MY OU	MGN PJU MBE DPP
09	I III V	20 07 03	BE CQ DI FH GW JO LM NZ PR SY	YJQ NPI JTY JEB
08	I III V	14 17 17	BV CZ DG FT HM IK JY LN OU SX	PXN PGK ZSW GOQ
07	I V III	01 04 05	AO CP DU FV GI HZ JK RY SX TW	VEU NRP ZZH VHF
06	III II IV	24 20 04	AZ BC DX EU FW GQ HJ IN LY RV	SAK SDR SGI TXT
05	IV I III	19 09 04	AW BU CF GO HP IT JQ KZ LX RS	YYY QRS XKA BWS
04	IV V I	22 13 24	AS BI DG FV HJ KO MU NY PT QX	BRV TIA JHF HIN
03	III II I	23 25 24	AO BW DJ EX FY GV HZ IP LR NU	AVE OHP OUO YEU
02	I II V	06 07 16	AK CE DV FR GX HW JL PY QZ SU	JUB LNQ UNU TDQ
01	I III V	15 02 24	AK CL DS EX HU IM JZ PW QV TY	JPI OSY OLN FYD

Изображение 7. Месечна таблица с ключове за Енигма

В нея можем да видим колони за ден, разположение на роторите, първоначални позиции на колелата на роторите, настройка на щепселите и групи, по които да се идентифицира ключа.

Армейската версия на Енигма използвала само 26 букви. Другите символи се заменяли с редки комбинации от букви. Интервалът се пропускал или се заменял с X. Символът „X“ най-често се използвал за точка или за край на съобщението. В отделните подразделения се използвали специални символи. В шифрованите съобщения на армията запетайката се заменяла с ZZ, а въпросителния знак с FRAGE или FRAQ. В шифровките, използвани от военноморските сили, запетаята се заменяла с Y, а въпросителния знак – с комбинацията UD. Комбинацията от символи CH, например в думите „ACHT“ (осем), „RICHTUNG“ (направление), се заменяла със символа Q („AQT“, „RIQTUNG“). Две, три или четири нули се заменяли съответно с думите „CENTA“, „MILLE“ и „MYRIA“. Вермахта (армията) и Луфтвафе (ВВС) изпращали съобщенията си в групи от по пет символа. Военноморските сили, използващи четирироторни машини, изпращали съобщенията си в групи от по четири символа. Често употребяваните думи и имена се изменяли по най-различни начини. Например, думата „Minensuchboot“ (минотърсач) можело да бъде изписана като „MINENSUCHBOOT“, „MINBOOT“, „MMMBOOT“ или „MMM354“. За да се затрудни криптоанализа, отделните съобщения не съдържали повече от 250 символа. По-дългите текстове били разделяни на части и за всяка част се използвал отделен ключ.

4. Математическо описание

4.1 Начин на работа

Преобразуването на всяка буква от Енигма може да бъде определено математически като резултат от пермутация. Да разгледаме трироторния армейски модел. Нека с $\boldsymbol{P}$ обозначим комутационния панел, с $\boldsymbol{U}$ – рефлектора, а $\boldsymbol{L}$, $\boldsymbol{M}$, $\boldsymbol{R}$ да обозначават съответно действията на левия, средния и десния ротори. Тогава шифрирането $\boldsymbol{E}$ с Енигма може да бъде изразено като:

$$E = PRMLUL^{-1}M^{-1}R^{-1}P^{-1}$$

След всяко натискане на клавиш, роторът се движи, водейки до трансформация. Например, ако десният ротор $\boldsymbol{R}$ се завърти $\boldsymbol{i}$ позиции, се получава трансформация $\boldsymbol{\rho R \rho^{-1}}$, където $\boldsymbol{\rho}$ е циклична пермутация, преминаваща от A към B, B към C и така нататък. Нещата се случват аналогично за средния и левия ротор. Нека означим с $\boldsymbol{j}$ и $\boldsymbol{k}$ даден брой завъртания на $\boldsymbol{M}$ и $\boldsymbol{L}$. Тогава функцията на шифриране може да се представи по следния начин:

$$E = P\left(\rho^{i}R\rho^{-i}\right)\left(\rho^{j}M\rho^{-j}\right)\left(\rho^{k}L\rho^{-k}\right)U\left(\rho^{k}L^{-1}\rho^{-k}\right)\left(\rho^{j}M^{-1}\rho^{-j}\right)\left(\rho^{i}R^{-1}\rho^{-i}\right)P^{-1}$$

4.2 По колко начина може да бъде настроена Енигма ?

Ще разгледаме по колко начина може да бъде настроена Енигма и по – точно Енигма I, която се е използвала от армията.

Имаме 5 ротора и от тях можем да изберем 3. За първия ротор имаме 5 варианта, за втория 4 и за последния 3. Следователно имаме 5 * 4 * 3 = 60.

Имаме 26 начални позиции за първия ротор, 26 за втория и 26 за третия. Следователно получаваме 26 * 26 * 26 = 26^3 = 17 576.

Но най – голямото усложнение идва от комутационният панел. Да разгледаме по колко начина можем да свържем 26 букви в 10 двойки на комутационния панел. Имаме 26 букви в азбуката. По колко начина можем да ги подредим ? Отговора е 26 * 25 * 24 … * 1 = 26!, но не искаме всички комбинации от 26 – те букви, а искаме 10 двойки. Остават 6 букви, следователно можем да разделим на 6! и формулата става следната:

$$\frac{26!}{6!}$$

Имаме 10 двойки, но не ни интересува реда, в който са подредени, следователно трябва да разделим на 10! И сега формулата е следната:

$$\frac{26!}{6! * 10!}$$

Последното нещо, което не трябва да забравяме е, че двойките A – B и B – A са едно и също нещо и затова трябва да разделим на 2. Имаме 10 двойки, тоест 10 пъти ще разделим на 2 и в формулата придобива следния вид:

$$\frac{26!}{6! * 10! * 2^{10}} = 150\,738\,274\,937\,250$$

Общо за начините, по които може да бъде подредена началната конфигурация на Енигма получаваме:

60 * 17 576 * 150 738 274 937 250 = 158 962 555 217 826 360 000 (158 квинтилиона 962 квадрилиона 555 трилиона 217 милиарда 826 милиона 360 хиляди).

5. Проблемът при Енигма

Както споменахме по – горе, когато описвахме частите съставящи Енигма, ако натиснем даден клавиш, например D, той винаги ще се шифрира

в някоя друга буква, но никога няма да бъде D. Това е така поради начинът, по който е устроен рефлекторът.

Това е била нишка, която разбивачите на Енигма са използвали. Всяка сутрин в 06:00 сутринта германците са изпращали доклад за времето. На немски доклад за времето е „WETTERBERICHT“. Сега ще опишем процедурата, която са използвали Алан Тюринг и колегите му в Блечли парк.

Да вземем за пример част от текст шифриран с Енигма и под него да наложим текста WETTERBERICHT и да се опитаме да намерим къде тази фраза стои в шифрирания текст, като имаме предвид, че дадена буква не може да бъде себе си:

```
... J X A T Q B G G Y W C R Y B G D T ...
    W E T T E R B E R I C H T
```

Забелязваме, че тук T се е шифрирала в T, което е невъзможно при Енигма. Следователно отместваме с една позиция напред.

```
... J X A T Q B G G Y W C R Y B G D T ...
      W E T T E R B E R I C H T
```

Тук забелязваме същия проблем, затова пак отместваме с едно напред.

```
... J X A T Q B G G Y W C R Y B G D T ...
        W E T T E R B E R I C H T
```

Тук нямаме съвпадения, така че това може да е правилната позиция. Нека да видим какво ще стане ако отместим с още едно напред.

... J X A T Q B G G Y W C R Y B G D T ...

W E T T E R B E R I C H T

Тук буквите R съвпадат.

... J X A T Q B G G Y W C R Y B G D T ...

W E T T E R B E R I C H T

Тук буквите T съвпадат.

Следователно можем да започнем да разбиваме шифъра от третата позиция.

Както вече знаем Енигма работи по следния (опростен) начин:

Натискаме клавиш -> електрическия сигнал отива при комутационния панел -> минава през роторите -> минава през рефлектора -> отново минава през роторите -> пак минава през комутационния панел -> светва лампа

Една много опростена диаграма е следната:

Диаграма 1. Опростена схема на работата на Енигма

Да не забравяме че работим със следния текст.

… J X A T Q B G G Y W C R Y B G D T …

W E T T E R B E R I C H T

Тук виждаме че например T се е шифрирало в E. Ще използваме това за да се опитаме да разберем как са били поставени щепселите. Тоест диаграмата изглежда така.

Диаграма 2. Опростена схема на работата на Енигма комбинирана с текст

Сега ще направим едно предположение: ***T*** е свързано с ***A*** с щепселите. Сега ние знаем как роторите са свързани избираме си позиция и пробваме да видим какво ще стане. ***A*** влиза при роторите и да кажем, че излиза ***P***. Но знаем, че крайния изход е буквата ***E***. Следователно можем да направим заключението, че ***P*** е свързано с ***E*** в комутационния панел. Диаграмата изглежда така:

Диаграма 3. Схема на работата на Енигма комбинирана с текст

Продължаваме работата в същия дух и правим други заключения.

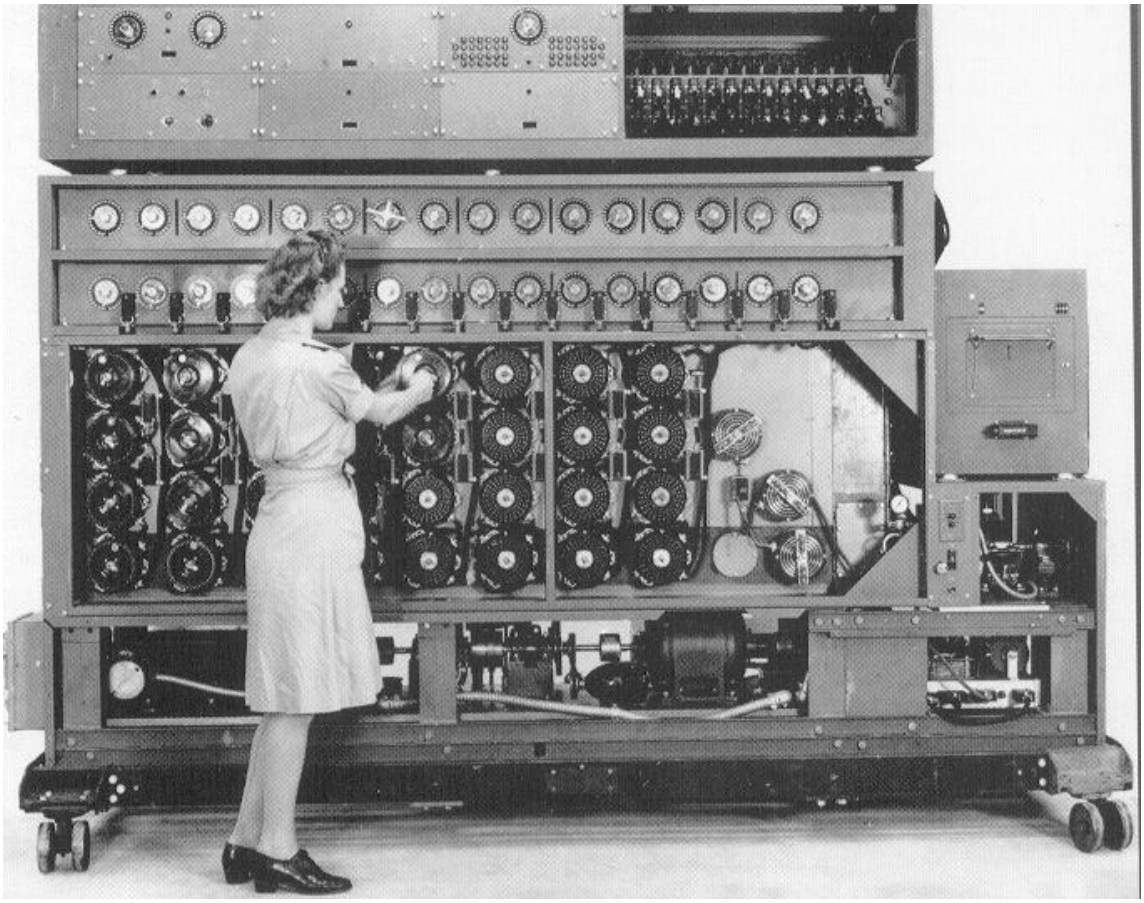
1. T -> A -> P -> E – следователно P е свързано с E в комутационния панел
2. T -> A -> K -> Q – следователно K е свързано с Q в комутационния панел
3. T -> A -> X -> B – следователно X е свързано с B в комутационния панел
4. T -> A -> T -> G – следователно T е свързано с G в комутационния панел

Току що открихме противоречие. В заключение 4 се казва, че T е свързано с G в комутационния панел, но това няма как да е вярно, защото ние сме допуснали, че T е свързано с A в комутационния панел. Сега може да започнем с друго предположение, например: T и B са свързани. Трябва да проверим всички 26 опции (25-те опции T да е свързано с друга буква и опцията T да не е свързана с никоя буква). Ако след всяка една опция стигнем до противоречие, това означава, че роторите са настроени грешно и преместваме позицията на роторите. Това е ужасно много работа и имайки предвид, че всеки ден позициите на роторите се сменят, няма как цялото това нещо да се изчисли от човек за 24 часа. Алан Тюринг се е сетил за 2 гениални неща:

1. Ако нашето предположение, че T и A са свързани е грешно, то всяко заключение направено от това предположение също е грешно и не трябва да ги проверяваме;
2. Това нещо може да се провери мигновено с електрическа верига.

Алан Тюринг построява машината Bombe, която да проверява точно тези неща. Тя прилага електрически заряд на T и A и проверява какво се

случва. Това, което забелязваме, че тази машина е построена отзад – напред. С тази процедура тя можела да мине през всички комбинации за около 20 минути. На ***Изображение 8.*** можем да видим как изглежда тази машина.



Изображение 8. – Машината Bombe

ПРИЛОЖЕНИЕ 2. SECURE HASH ALGORITHM (SHA)

Въведение

Secure Hash Algorithm (SHA), който се имплементира в стандарта ***Secure Hash Standard (SHS)*** като правителствен стандарт, е разработен от NIST и е публикуван като стандарт за обработка на федерална информация FIPS 180 през 1993 година, който определя алгоритъмът да се използва с като стандарт за цифровия подпис (*Digital Signature Standard*). Стандартът специфицира алгоритъма, който е необходим за осигуряване на защита на алгоритъма за цифров подпис DSA (*Digital Signature Algorithm*). Когато едно съобщение, по-малко от 264 бита, се явява входен текст, SHA генерира 160-битов изход, наречен извлечение.

Подобрената версия излиза през 1995 година като FIPS 180-1, което всъщност се отнася за SHA-1. Така споменатият алгоритъм се използва в стандарта Secure Hash Standard. SHA е сигурен алгоритъм, защото е проектиран така, че да бъде изчислимо невъзможно да се възстанови оригиналното съобщение, или да се намерят две различни съобщения, които дават един и същ хеш резултат. SHA се основава на принципи, подобни на тези които Ronald Rivest е използвал при създаването на алгоритъма MD4. Въпреки това SHA дава 160 бита хеш, който е по-дълъг от този при MD5. SHA е базиран върху хеш функцията на MD4 и моделът на действие много се доближава до този на MD4.

SHA-1 дава като резултат хеш стойност от 160 бита. През 2002 година NIST представя нова версия на стандарта, FIPS 180-2, който дефинира три нови разновидности на SHA, съответно с дължина на хеш стойностите 256, 384 и 512 бита, известни като SHA-256, SHA-384, SHA-512. Тези алгоритми имат същата структура в основата си и използват еднаква модулна аритметика и логически бинарни операции като SHA-1. През 2010 NIST

обявява намерението си да внесе в сила нова версия на SHA. По това време изследователски екип описва атака, в която две различни съобщения са открити да дават еднаква хеш функция с използване на 2^{69} операции, доста по-малко от 2^{80} операции, считани преди това за нужни за откриване на колизии със SHA-1 хеш. Този резултат ускорява прехода към другите версии на алгоритъма.

Таблица 1. Сравнение на параметри на различни SHA версии

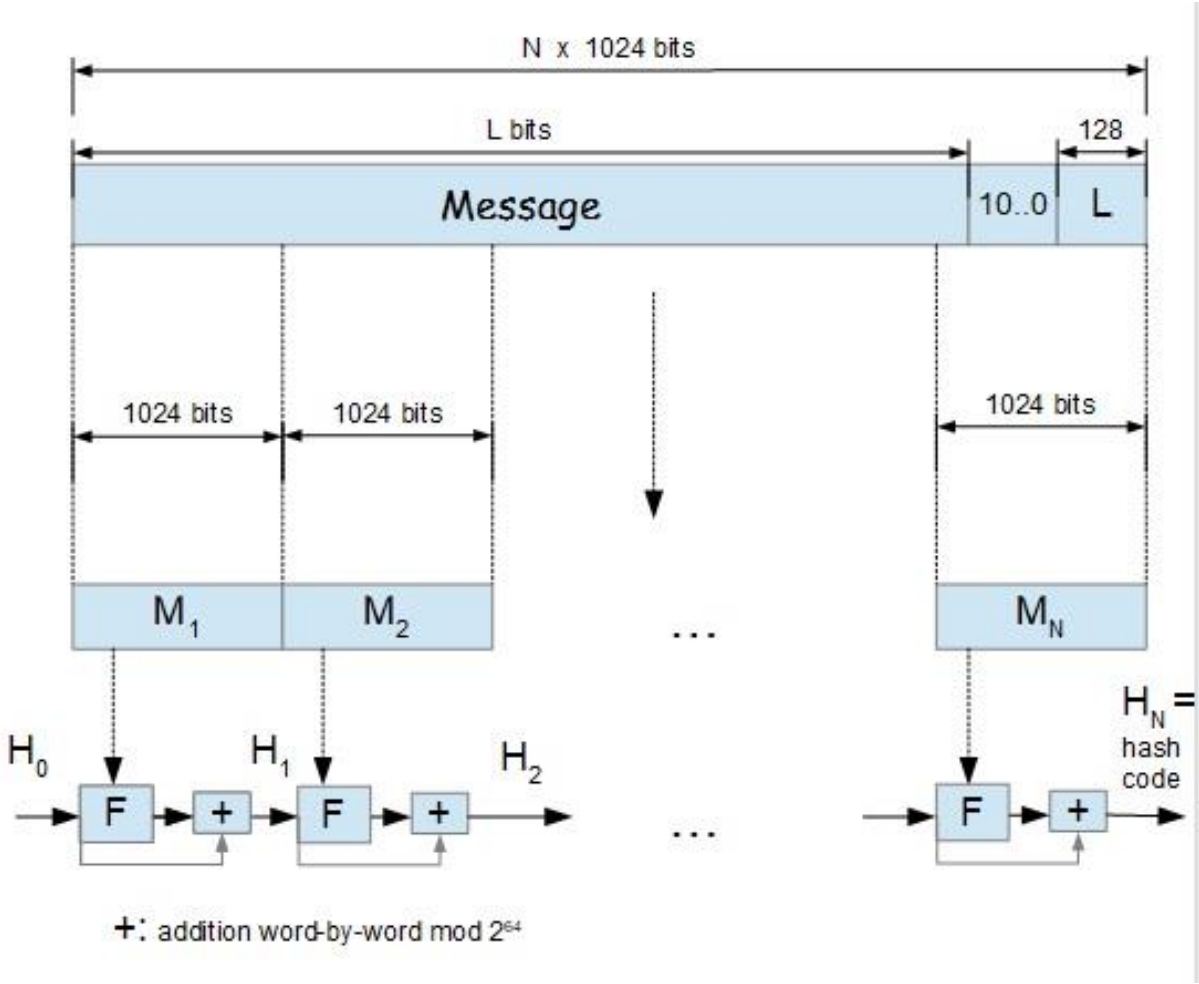
	SHA-1	SHA-256	SHA-384	SHA-512
Големина на хеш извлечение	160	256	384	512
Големина на съобщението	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$
Големина на блока	512	512	1024	1024
Големина на думата	32	32	64	64
Брой стъпки	80	64	80	80
Сигурност	80	128	192	256

Всички стойности са измерени в битове. Сигурността се отнася до факта, че т.нар. „***birthday attack***“ върху извлечението на съобщението с големина *n* продуцира колизия в работен фактор приблизително $<2^{n/2}$. В следващите редове ще се разгледа стъпките при шифриране с SHA-512

алгоритъм, като се има предвид, че логиката при другите версии е еднаква. Разликите, както се вижда и от таблицата, идват от параметрите.

Описание на SHA-512

Алгоритъмът взима като вход съобщение с дължина не по-голяма от 2^{128} бита (при SHA-384 и SHA-512) и произвежда изход от 512 бита извлечение. Входът се обработва като блокове от по 1024 бита. ***Фигура 1.*** изобразява най-общо съобщение, от което излиза хешираното съобщение. SHA е базиран на тази схема, която е известна като ***Merkel Damgard Scheme.***



Фигура 1. Генериране на извлечение чрез SHA-512

Процесът се състои от следните стъпки:

- ***Стъпка 1: Прибавяне на допълнителни допълващи битове***. Съобщението се допълва така, че дължината му е сравнима с 896 модул 1024. Винаги се добавят битове дори когато съобщението е с необходимата дължина. Поради това разширението е със стойности в диапазона от 1 до 1024 бита. То се състои от единичен 1 бит, следват от необходимия брой 0 битове.

- ***Стъпка 2***: ***Прибавяне на дължина.*** Блок от 128 бита се добавя към съобщението. Блокът се възприема като без знакова 128-битова целочислена стойност (най-значимият байт първо) и съдържа дължината на оригиналното съобщение преди разширението.

Резултатът от първите две стъпки дава съобщение от 1024 бита дължина. Както се вижда от фигурата разширеното съобщение е представено като последователност от 1024-битови блокове $M_1, M_2, \dots, M_N$, като крайната дължина на съобщението става N x 1024 бита.

- ***Стъпка 3: Инициализиране на хеш буфер.*** Буфер от 512 бита се използва за съхранение на междинни и крайни резултати от хеш функцията. Буферът се представя като осем 64-битови регистъра – a, b, c, d, e, f, g, h. Тези регистри се инициализират както следва с 64 бита целочислени числа (шестнадесетични стойности):

  a=6A09E667F3BCC908, b=BB67AE8584CAA73B,
  c=3C6EF372FE94F82B, d=A54FF53A5F1D36F1,
  e=510E527FADE682D1, f=9B05688C2B3E6C1F,
  g=1F83D9ABFB41BD6B, h=5BE0CDI9137E2179.

Стойностите се съхраняват с първи най-старшия бит формат, който е най-значимият байт от дума в най-лява позиция. Тези думи са извлечени, като се вземат първите 64 бита от дробните части на квадратните корени на първите осем прости числа.

- ***Стъпка 4: Обработка на съобщението като 1024-битови (128-дума) блокове***. Сърцето на алгоритъма е модул, който се състои от осемдесет обработки. Този модул е обозначен като F във фигурата по-горе. Логиката на алгоритъма е представена в следващата фигура. Всеки етап от обработки взима като вход стойността на 512-буфер a b c d e f g h , и обновява съдържанието на буферите. Като входни данни за първата обработка буферът взима стойността на междинния хеш резултат H_{i-1}. Всеки етап от обработката t използва стойността 64 бита W_t, извлечена от съответния 1024-битов блок, който е в процес на обработка (M_i). Тези стойности са получени, използвайки схема на съобщението, описана последователно. Всеки етап на обработката също включва допълнителна константа K_t, където $0 < t < 79$ индикира един от осемдесетте етапа на обработката. Тези думи представляват първите 64 бита от дробните части на кубични корени на първите осемдесет прости числа. Константите осигуряват случаен набор от 64-битови образци, които трябва да елиминират всякакви еднаквости и съответствия във входните данни. Резултатът от последния етап се прибавя към входа на първия етап H_{i-1}, което дава H_i. Добавянето се извършва независимо на всеки отделен етап за всяка от осемте думи в буфера с всяка от съответстващите й думи H_{i-1}, използвайки добавяне по модул 2^{64}.

- ***Стъпка 5: Резултатът***. След като всички 1024-битове блокове бъдат обработени, изходът на N-тата обработка е крайното извлечение от 512 бита. Поведението на алгоритъма може да бъде обобщено по следния начин:

$$H_0 = IV$$

$$H_i = SUM_{64}(H_{i-1}, abcdefgh_i)$$

$$MD = H_N$$

където IV е първоначалната стойност на буфера, дефинирана в стъпка 3, $abcdefgh_i$ е резултатът от последния етап на обработката на i-тия блок, N е броят на блокове в съобщението, включително разширението и полетата за дължина, SUM_{64} е прибавяне по модул от 2^{64}, извършвано поотделно за всяка дума от двойките входове, MD е крайното съобщение (извлечението).

В следващите редове се представени формули, които показват подробно логиката на всяка една от осемдесетте стъпки при обработка на един блок от 512 бита. Всеки един етап се дефинира със следните уравнения:

$$T_1 = h + Ch(e, f, g) + \sum_1^{512} e + W_t + K_t$$

$$T_2 = \sum_0^{512} a + Maj(a, b, c)$$

$$a = T_1 + T_2$$

$$b = a$$

$$c = b$$

$$d = c$$

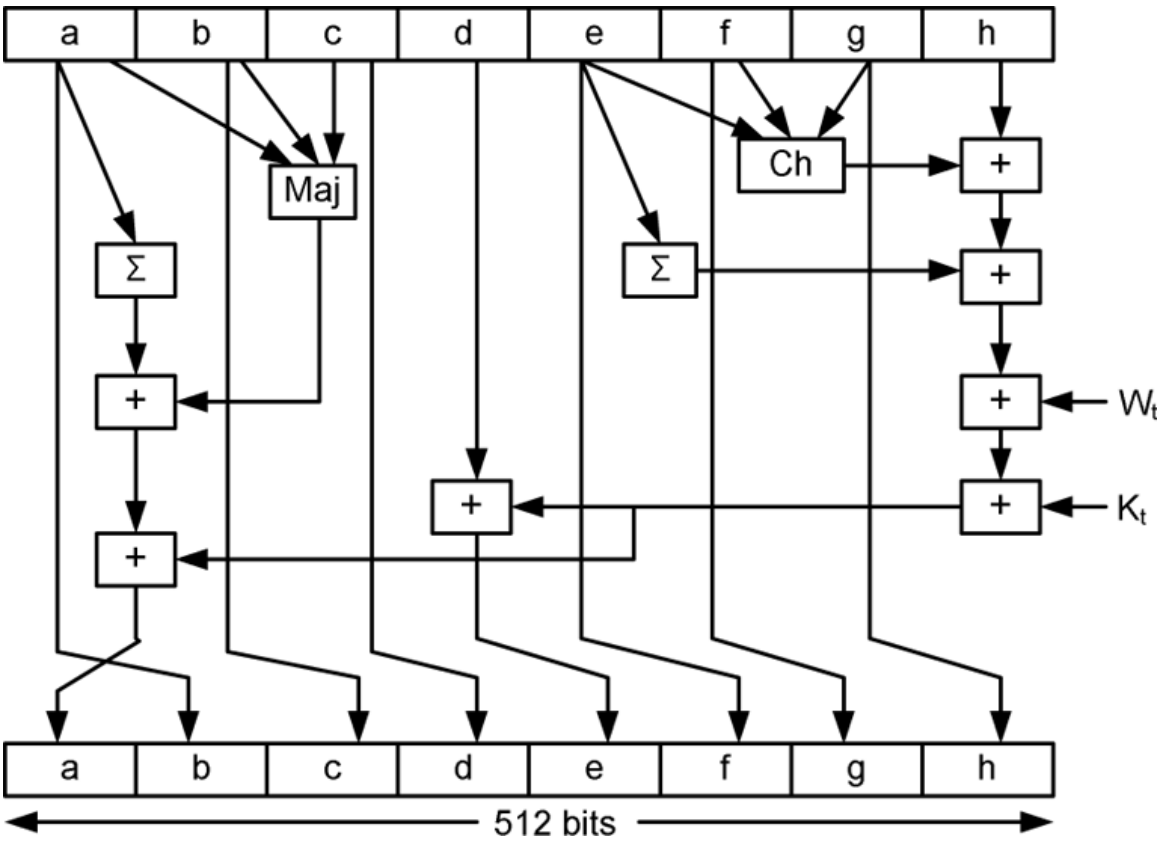
$$e = d + T_1$$

$$f = e$$

$$g = f$$

$$h = g$$

- където t е номерът на етапа на обработката със стойности между 0 и 79,
- $Ch(e,f,g) = (e\ and\ f) + (not\ e\ and\ g)$ е функция с условие „Ако e, то f, иначе g“,
- $Maj(a,b,c) = (a\ and\ b) + (a\ and\ c) + (b\ and\ c)$ е функция, която е вярна само ако два или трита й аргумента са истина.
- $\sum_1^{512} e = ROTR^{14}(e) + ROTR^{18}(e) + ROTR^{41}(e)$,
- $\sum_0^{512} a = ROTR^{28}(a) + ROTR^{34}(a) + ROTR^{39}(a)$,където $ROTR^n(x)$ е ротация вдясно на аргумент от 64 бита с n бита. W_t е дума от 64 бита, която е извлечена от входното съобщение. K_t е допълнителна константа с големина 64 бита. Знакът плюс се използва за илюстриране на събиране по модул 2^{64}.



Фигура 2. Елементарните операции при един етап на обработката

Единственото необяснено, което остана да се изясни, е как стойностите W_t се извличат от цялото съобщение, състоящо се от 1024 бита. Първите шестнадесет стойности на W_t се извличат директно от шестнадесетте думи на настоящия блок. Останалите стойности се определят от следния израз:

$$W_t = \sigma_1^{512}(W_{t-2}) + W_{t-7} + \sigma_0^{512}(W_{t-15}) + W_{t-16}$$, където:

$$\sigma_0^{512}(x) = ROTR^1(x) + ROTR^8(x) + SHR^7(x)$$

$$\sigma_1^{512}(x) = ROTR^{19}(x) + ROTR^{61}(x) + SHR^6(x)$$

$ROTR^n(x)$- кръгова ротация надясно на 64-битовия аргумент с n бита. $SHR^n(x)$ е изместване вляво на 64-битовия аргумент с n позиции, като се добавят нули отдясно. От така описания алгоритъм следва, че първите шестнадесет стъпки стойността на W_t е еднаква на съответстващата дума в блока съобщение. За останалите 64 стъпки стойността на W_t се състои от кръгово изместване с един бит от четири стойности на W_t, между които се извършва операция „изключващо или“, като две от тези стойности са обекти едновременно на ротация и изместване. Този подход внася голяма доза резервираност и независимост на съобщенията, които са компресирани. Това допълнително усложнява задачата за намиране на различен блок съобщение, който да съответства на същата компресирана функция, която се получава като резултат.

ПРИМЕР

ASCII символите „abc“ са еквивалентни на следният 24-битов бинарен низ: 01100001 01100010 01100011, който е равен на 616263 в шестнадесетична система. Дължината е 24 бита, които има шестнадесетична стойност от 18.

Така съобщението от 1024 бита става:

6162638000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000
0000000000000000	0000000000000000	0000000000000000
0000000000000018		
$W_0 = 6162638000000000, W_1 = W_2 = \cdots = W_{14} = 0000000000000000,$ $W_{15} = 0000000000000018.$		

Таблица 2. Инициализиране на буферните константи при обработка от 80 стъпки

Стойности на буферни константи	Първоначална	След първа стъпка	След втора стъпка
A	6a09e667f3bcc908	F6afceb8bcfcddf5	1320f8c9fb972cc0
B	Bb67ae8584caa73b	6a09e667f3bcc908	F6afceb8bcfcddf5
C	3c6ef372fe94f92b	Bb67ae8584caa73b	6a09e667f3bcc908
D	A54ff53a5f1d36f1	3c6ef372fe94f92b	Bb67ae8584caa73b
E	510e527fade682d1	**58cb02347ab51f91**	**C3d4ebfd48605ffa**
F	9b05688c2b3e6c1f	510e527fade682d1	58cb02347ab51f91
G	1f83d9abfb41bd6b	9b05688c2b3e6c1f	510e527fade682d1
H	5be0cd19137e2179	1f83d9abfb41bd6b	9b05688c2b3e6c1f

Този процес продължава така осемдесет стъпки(етапи), като на всяка стъпка стойностите на константите се обновяват. Стойностите се изчисляват според формулите за буферни константи, които бяха описани. Начинът, по който се взимат стойностите при първоначалната инициализация на буферните константи, е следният. Например, стойността на последната h константа се извлича, като се изчисли корен квадратен от осмото просто число, което е 19. Корен квадратен от 19 = 4.35889894354. Това число се конвертира в двоичната му стойност със 64 бита в дробната част и така получаваме:

$$100.0101\ 1011\ 1110 \ldots 1001_2 = 4.5be0cd19137e2179_{16}$$

Съответно взимаме стойността на дробната част $5be0cd19137e2179$. При изчисляване на осемдесетата стойност обаче се вземе дробната част на кубичния корен на осемдесетото просто число, което е 409.

След последния етап на обработката се получава следния резултат:

73a54f399fa4b1b2	10d9c4c4295599f6	d67806db8b148677
654ef9abec389ca9	d08446aa79693ed7	9bb4d39778c07f9e
25c96a7768fb2aa3	ceb9fc3691ce8326	

Калкулират се съответните хеш стойности, като всяка една е сбор от резултатната и първоначалната за съответния регистър.

$H_{1,7} = 5\text{be0cd19137e2179} + \text{ceb9fc3691ce8326} = 2a9ac94fa54ca49f$

$H_{1,6} = 1\text{f83d9abfb41bd6b} + 25\text{c96a7768fb2aa3} = 454\text{d4423643ce80e}$

$H_{1,5} = 9\text{b05688c2b3e6c1f} + 9\text{bb4d39778c07f9e} = 36\text{ba3c23a3feebbd}$

$H_{1,4} = 510\text{e527fade682d1} + \text{d08446aa79693ed7} = 2192992\text{a274fc1a8}$

$H_{1,3} = \text{A54ff53a5f1d36f1} + 654\text{ef9abec389ca9} = 0\text{a9eeee64b55d39a}$

$H_{1,2} = 3\text{c6ef372fe94f92b} + \text{d67806db8b148677} = 12\text{e6fa4e89a97ea2}$

$H_{1,1} = \text{Bb67ae8584caa73b} + 10\text{d9c4c4295599f6} = \text{cc417349ae204131}$

$H_{1,0} = 6\text{a09e667f3bcc908} + 73\text{a54f399fa4b1b2} = \text{ddaf35a193617aba}$

Оттук крайното хеширано съобщение от 512 бита е:

73a54f399fa4b1b2 cc417349ae204131 12e6fa4e89a97ea2 0a9eeee64b55d39a

2192992a274fc1a8 36ba3c23a3feebbd 454d4423643ce80e $2a9ac94fa54ca49f$

Хеш функцията SHA-2 се реализира в голяма част от широко използвани приложения и протоколи като TLS, SSL, PGP, SSH.

Стандартът SHA-3 – хеш функция, основана на пермутация

Стандартът от семейството на SHA-3 се основава на алгоритъма *Keccak,* алгоритъм, който *NIST* избират като победител на състезанието *SHA-3 Cryptographic Hash Algorithm Competition.* Семейството на SHA-3 се състои от четири криптографски хеш функции и две разшируеми функции. Тези шест функции споделят структура, наречена *sponge construction* и съответно функциите се наричат *sponge functions.* Четирите хеш функции са съответно SHA3-224, SHA3-256, SHA3-384, SHA3-512, като както при предходните версии числото в името означава дължината в битове на хеш резултата (извлечението). Това позволява хеш функциите на SHA-3 да бъдат реализирани като алтернативи на тези на SHA-2 и обратно.

Extendable Output Functions (XOF) са функции, засягащи бит низове, чийто резултат може да бъде разширен до всяка желана дължина. Двете *XOF* функции се наричат *SHAKE128* и *SHAKE256.* Числата „128“ и „256“ определят ниво на сигурност, които функциите могат да поддържат. Тези две функции са първите *XOF* функции, които са стандартизирани. Шестте SHA-3 функции поддържат устойчивост на колизии, *preimage* атаки. Когато някое приложение изисква хеш функция с нестандартна дължина на хеша,

XOF е естествена алтернатива за конструкции, включващи множество извиквания на хеш функция и/или съкращаване на битовете на резултатния хеш.

ЗАКЛЮЧЕНИЕ

В настоящата книга е отразен дългогодишният опит на автора в разработването и внедряването на системи за защита на информацията в компютърните системи и мрежи на Българската армия и Генералния щаб. Съдържанието отразява и част от пътят извървян от автора в изследванията му в предметната област ***„сигурност и защита на информацията“.***

Съдържанието на книгата е обобщение на учебния материал от лекциите на няколко университетски курса водени в Софийския Университет „Св. Климент Охридски“ и Нов български университет.

За основа на част от материалите са послужили оригиналните издания на съответните стандарти, критерии и ръководства. Друга част – съдържа оригинални авторски изследвания и обобщения.

Книгата е подходящо четиво за широк кръг читатели. Тя би могла да бъде използвана като учебно пособие и методическо ръководство и да задоволи интереса на:

- Специалисти по сигурност на информационните технологии;
- Потребители, производители, изследователи и оценители на средства за мрежова и информационна сигурност;
- Проектанти и ръководители на корпоративни компютърни мрежи;
- Студенти, докторанти и научни работници в областта на информационната сигурност;
- Читатели с желание да повишат общата си култура;
- И много други.

Специална благодарност на ***д-р Виолета Дъчева*** за предговора и представянето на Владо Кожухаров и на рецензентите ***доц. д-р Николай Стоянов*** и ***доц. д-р Мая Божилова*** за отделеното време и ценните препоръки.

Авторът благодари и на всички колеги и приятели, които спомогнаха и не попречиха тази книга да бъде издадена.

СПИСЪК НА ФИГУРИТЕ И ТАБЛИЦИТЕ В ОСНОВНИЯ ТЕКСТ

Списъкът на фигурите и таблиците в основния текст е представен по глави по долу.

ФИГУРИ

ГЛАВА 1. СИГУРНОСТ НА ДАННИТЕ

ГЛАВА 2. КРИПТОГРАФСКИ АЛГОРИТМИ

ГЛАВА 3. КЛАСИЧЕСКА КРИПТОГРАФИЯ

ГЛАВА 4. СЪВРЕМЕННА КРИПТОГРАФИЯ

ТАБЛИЦИ

ГЛАВА 2. КРИПТОГРАФСКИ АЛГОРИТМИ

ЛИТЕРАТУРА

1. Божилова, М. и Н. Стоянов. Криптография, базирана на решетки. Криптосистема NTRU, Научна сесия 2013, Национален военен университет Васил Левски, Факултет „Артилерия, ПВО и КИС“. Шумен, Българи, ISSN 1313-7433.
2. Денчев, С. Информация и сигурност. София: За буквите – О писменехь, 2019, ISBN:978-619-185-369-4 – pdf.
3. Стоянов, Н. и М. Божилова. Нови направления в развитието на криптографията. – В: Новите предизвикателства пред системите за информационна сигурност. Шумен, 2015, с. 9 – 23. ISBN 978-954-9681-65-9.
4. Стоянов Н., Г. Велев и М. Божилова. Компютърни мрежи и комуникации. София: За буквите – О писменехь, 2014, ISBN 978-619-185-148-5.
5. Целков, В. и О. Исмаилов. Сигурност на информацията. София: За буквите – О писменехь, 2016, ISBN 978-619- 185-253- 6.
6. Целков, В., Н. Стоянов и О. Исмаилов. Международни стандарти и добри практики за защита на информацията. София: За буквите – О писменехь, 2010, ISBN 978-954-8887-67-0.
7. Целков, В. и Н. Стоянов. Защитени криптографски приложения в компютърните системи и мрежи. София: Нова звезда, 2009, ISBN 978-954-20-9.
8. Целков, В., Н. Стоянов и О. Исмаилов. Управление на риска, тестване и оценка на мрежовата и информационна сигурност. София: За буквите – О писменехь, 2016, ISBN 978-619-185-159-1.
9. Целков, В. Управление на риска за сигурността на комуникационно-информационните системи. София: За буквите – О писменехь, 2020, ISBN 978-619-185-445-5- pdf.

10. Целков, В., С. Денчев и Ир. Петева. Сигурност на информационните ресурси. София: За буквите – О писменехь, 2020, ISBN 978-619-185-432-5.

11. Adams, Carlisle and Steve Lloyd. Understanding Public-Key Infrastructure: Concepts, Standards, and Deployment Considerations, New Riders Publishing, 1999.

12. Anderson R. Introduction to Security. University of Cambridge, 1999.

13. Doraswamy, Naganand and Dan Harkins, IPSec: The New Security Standard for the Internet, Intranets, and Virtual Private Networks, Prentice Hall, 1999.

14. Douglas R. Stinson. Cryptography: Theory and Practice, Second Edition, Chapman and Hall/CRC, 2002.

15. Farley M. and etc. LAN Times Guide to Security and Data Integrity, McGraw-Hill, Inc., 1996.

16. Kaufman, Charlie, Radia Perlman, and Mike Speciner. Network Security: Private Communication in a Public World, Second Edition, Prentice Hall, 2002.

17. Menezes J., Vanstone S. A., and Oorschot P. C. V. Handbook of Applied Cryptography, 1st ed., ser. Computer Sciences Applied Mathematics Engineering. CRC Press, Inc., 1996, http://www.cacr.math.uwaterloo.ca/hac/.

18. Norton P., Stockman M. "Peter Norton's Network Security Fundamentals", Indianapolis, SAMS, 1999.

19. Russell D. and G. Ganemi, Computer Security Basics, O'Reily & Associates, Inc., 1991.

20. Stallings, W. Cryptography and Network Security: Principles and Practice, Third Edition, Prentice Hall, 2003.

21. Rescorla, Eric. SSL and TLS: Designing and Building Secure Systems, Addison-Wesley, 2001.

22. Schneier, Bruce. E-Mail Security: How to Keep Your Electronic Messages Private, John Wiley and Sons, 1995.

23. Schneier Br. Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition, John Wiley and Sons, 1996.

24. Stoianov N., M. Bozhilova and G. Velev. Towards security requirements of the SPIDER project, Proceedings Scientific Conference with International Participation on Cyber security in the Information Society, Shumen, Bulgaria, April 19 – 20, 2017, ISBN 978-954-9681-82-6, pp. 25 – 31.

25. Stoianov N. and M. Bozhilova. A study of Lattice-based Cryptography, 5th International Scientific Conference on Defensive Technologies, OTEH 2012, Belgrade, Serbia, Proceedings, ISBN 978-86-81123-58-4, pp. 465 – 469, http://www.vti.mod.gov.rs/oteh12/elementi/rad/5-07.html;

26. IT Baseline Protection Manual, Federal Agency for Security in Information Technology, BSI, 2000.

27. Tselkov, V. Cryptographic Software Solution for Information Protection in a Corporate Intranet, Information&Security, vol. 4, 2000, pp. 97 – 104, (ISSN 1311-1493).

28. Tselkov, V. and N. Stoianov, A model of software cryptography system for data protection in distribution information system, South Eastern Europe Conference on Regional Security through Data Protection, Belgrade, December, 2003, www.dataprotection2003.info/speakers/Veselin_Tselkov/presentation.pdf

29. Tselkov V. and N. Stoianov. A Formal Model of Cryptographic Systems for Protection of Information in Computer Systems and Networks,

Proceedings of Bulgarian Cryptography Days – BulCrypt 2012, Sofia, 2012, pp. 15 – 29.

30. http://www.nsa.gov
31. http://www.nist.gov
32. http://www.commoncriteriaportal.org
33. http://www.itu.int
34. http://www.rsa.com
35. http://www.iwar.org.uk
36. http://www.cnss.gov
37. http://www.isaca.org

ПОСЛЕСЛОВ

Съвременното развитие на науката и технологиите направиха комуникационните мрежи и информационните системи неразделна част от живота на обществото. Непрекъснато увеличаващата им се роля в ежедневната дейност, промени начина на работа, на обучение, на търговия, на комуникации и взаимодействия. Разпространението на Интернет, бързо нарастващият брой на приложенията и потребителите в Световната мрежа, създадоха условия за увеличаване на заплахите за информационната сигурност, и повишаване на риска от кибер атаки. Евентуалните щети след реализирани атаки могат да не са само икономически и финансови, а да са свързани и с човешки жертви. В този контекст, е необходимо задълбочаване на изследванията, непрекъснато развитие на знанията и търсене на нови по-ефективни решения за сигурността на данните. Основно средство за защита на данните и осигуряване на тяхната цялост са криптографските алгоритми. Освен това, сигурността на информацията може да бъде постигната само с адекватно обучение и подготовка на висококвалифицирани експерти по приложението на криптографските услуги и механизми в защитата на информацията в компютърните системи и мрежи.

Книгата „Въведение в криптографията и сигурността на данните“ е увод в криптографията – алгоритми, услуги и механизми, и възможности за приложението им. Книгата е структурирана в четири глави, в които авторът последователно въвежда отделните аспекти от осигуряване на сигурността на данните в компютърните системи и мрежи, основните понятия от областта на криптографията, както и разглежда конкретни класически и съвременни криптографски алгоритми.

Настоящата книга е с принос в решаването на две задачи – от една страна, тя доразвива съществуващите знания за сигурността на данните, а от друга страна предоставя инструмент за обучение в областта на криптографията и приложението и в осигуряване на сигурността на данните.

Познавам лично проф. д.т.н. Веселин Целков от дългогодишната ни успешна съвместна работа в научно-изследователските структури на Министерството на отбраната и Генералния щаб в полза на националната сигурност и повишаване на отбранителните способности на Република България. Авторът е един от най-изявените специалисти в областта на информационната сигурност, с опит, както на национално, така и на международно ниво. Това е поредната книга от серията „Защита информацията“, с която той потвърждава експертното знание в областта на криптографията и приложения и доказва за пореден път уменията си за изграждане на стройна система от знания в областта на мрежовата и информационната сигурност.

Приложимостта на знанието представено в книгата „Въведение в криптографията и сигурността на данните“ е безспорна. Книгата е интересно четиво за любознателния читател и професионалиста, подходящ учебник за обучение и удачно методическо средство за студенти, специализанти и докторанти, наръчник за потребители, разработчици и оценители.

март 2021 г.
гр. София

Доц. д-р Мая Божилова
Институт по отбрана,
Министерство на отбраната

AFTERWORD

Modern developments in science and technology have made communication networks and information systems an integral part of society. Their constantly increasing role in daily activities has changed the way people work, study, trade, communicate and interact. The proliferation of the Internet, the rapidly growing number of applications and users on the World Wide Web, have created the conditions for increasing threats to information security and increasing the risk of cyber-attacks. Possible damage after attacks can be not only economic and financial, but also related to human casualties. In this context, it is necessary to deepen research, continuous development of knowledge and search for new and more effective solutions for data security. Cryptographic algorithms are the main means of data protection and ensuring their integrity. In addition, information security can only be achieved through adequate training and preparation of highly qualified experts in the application of cryptographic services and mechanisms in the protection of information in computer systems and networks.

The book "Introduction to Cryptography and Data Security" is an introduction to cryptography – algorithms, services and mechanisms, and opportunities for their application. The book is structured in four chapters, in which the author consistently introduces the various aspects of data security in computer systems and networks, the basic concepts in the field of cryptography, as well as considers specific classical and modern cryptographic algorithms.

I know personally Prof. DSc Vesselin Tselkov from our long-term successful joint work in the research structures of the Ministry of Defense and the General Headquarters in favor of national security and increasing the defense capabilities of the Republic of Bulgaria. The author is one of the most prominent specialists in the field of information security, with experience both nationally and internationally. This is another book in the "Information Security" series, whereby he confirms his expertise in the field of cryptography and applications

and proves once again his skills for building a coherent system of knowledge in the field of network and information security.

The applicability of the knowledge presented in the book "Introduction to Cryptography and Data Security" is indisputable. The book is an interesting read for the curious reader and the professional, a suitable textbook for training and a suitable methodological tool for students, postgraduates and doctoral students, a guide for users, developers and evaluators.

March 2021
Sofia

Assoc. Prof. Dr. Maya Bozhilova
Institute of Defense, Ministry of Defense

ABOUT THE AUTHOR

PROF. DSc VESELIN TSENOV TSELKOV

Veselin Tsenov Tselkov graduated from the Faculty of Mathematics and Mechanics at Sofia University “St. Kliment Ohridski”, as a Master with a specialization in Mathematical Logic.

V. Tselkov defended his doctoral dissertation (candidate of technical sciences) on the topic “Information Management in EIM Networks” (specialty “System Programming”) in 1990. In 1996 Veselin Tselkov was elected a senior research associate 2nd degree in the scientific specialty "Informatics" (Information Security). In November 2008 he defended his dissertation for the degree of Doctor of Technical Sciences (Informatics, Information Security). In August 2010, he was awarded the scientific title of "Professor" (Automated Information Processing and Management Systems). He is the founder, leader and certified instructor of the first in Sofia regional Cisco Academy in the Military Academy “G. S. Rakovski” (2001).

He specializes in Critical Information Infrastructure Management at the George Marshal European Center for Security Studies and Communication and Information Technology (Cisco Academy) at the Technical University, Bucharest, Romania.

Vesselin Tselkov is a lecturer in many courses related to information security at the University of Library Studies and Information Technologies, Sofia University “St. Kliment Ohridski” and Plovdiv University “Paisiy Hilendarski”.

He is the author of over 250 monographs, scientific publications and research in the field of security, information protection and information technology, of which over 80 abroad or in international forums. He is the head of the research team publishing the series "INFORMATION PROTECTION". More than 200 citations of his works are known in the country and abroad. He is a

research supervisor of students, postgraduates and doctoral students in the field of information technologies and information security.

Vesselin Tselkov is a colonel in the reserve of the Bulgarian Army.

From 2003 to 2007 Veselin Tselkov was a member of the State Commission on Information Security. On 19 December 2007 he was elected by the National Assembly as a member of the Commission for Personal Data Protection, and on 15th April 2014 he was re-elected for a second term as a member of the Commission for Personal Data Protection.

He has been a friend and member of the Atlantic Club since its founding. Member of AFCEA - Sofia. He was also on the Board of AFCEA - Sofia.

Веселин Целков

ВЪВЕДЕНИЕ В КРИПТОГРАФИЯТА И СИГУРНОСТТА НА ДАННИТЕ

Българска
Първо издание

Предговор
д-р Виолета Дъчева

Научни рецензенти
доц. д-р Николай Стоянов
доц. д-р Мая Божилова

Редактор и коректор
д-р Георги Средков

Графичен дизайн на корица
Цветелин Цонев

Формат 64/90/16

Академично издателство „За буквите – О писменехь“

ISBN 978-619-185-460-8

ICO.

Bulgaria

Information
Consulting
Office

Вашите консултанти в защитата на личните данни и в подготовката за постигане на съответствие с Регламент 2016/679

Тел: +359 2 417 1105
Моб: +359 887 348 071
Моб: +359 877 551 111
Office@icobg.eu

гр. София - 1404
бул."България" № 109
бизнес сграда "Вертиго", ет. 1, офис 45
www.icobg.eu

- ***Оценка на съответствието на защитата на личните данни с изискванията на Регламента за защита на личните данни;***

- ***Оценка на въздействието и управление на риска;***

- ***Защита на данните, политика за неприкосновеност и прилагане на основни принципи за защита на личните данни;***

- ***Длъжностно лице по защита на данните;***

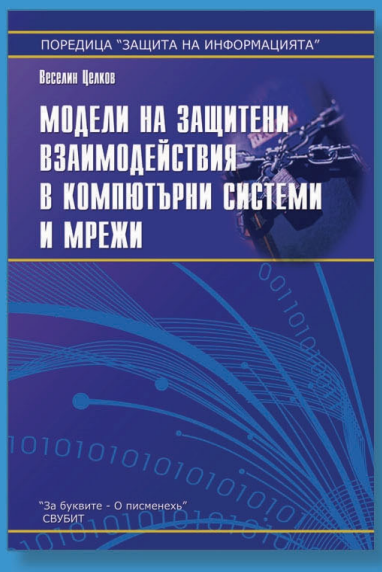
- ***Обучения и тренинги;***

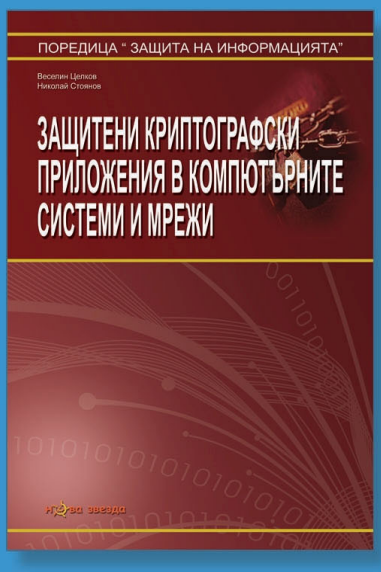
- ***Управление на нарушенията;***

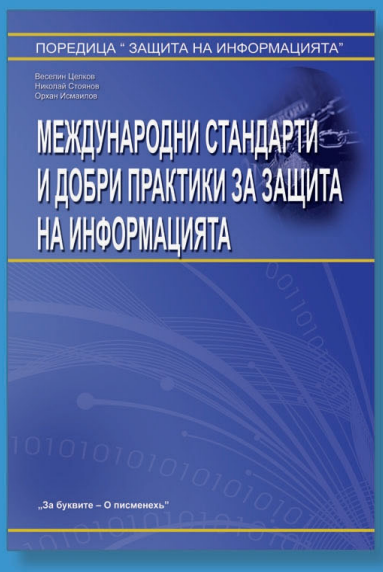
- ***Политика и процедури за работа със съвместни администратори, обработващи, граждани, контрагенти и КЗЛД;***

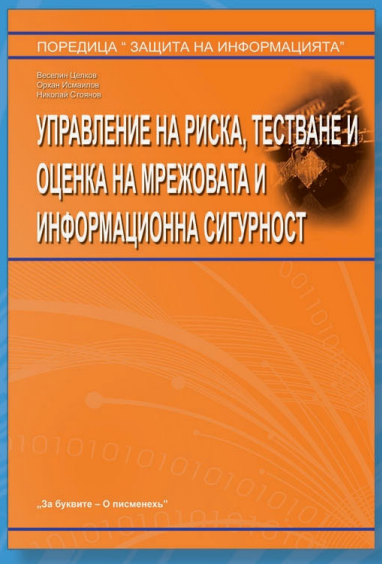
- ***Преносимост, трансфер и публикуване на лични данни;***

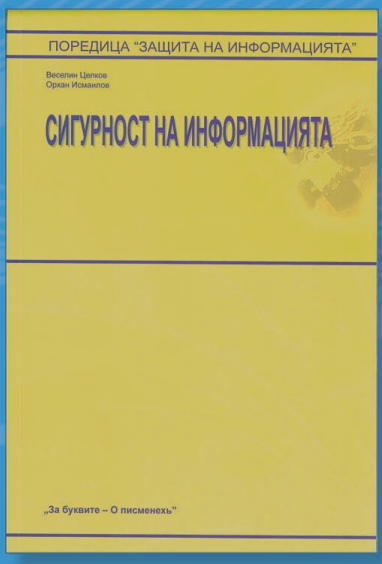
- ***Информационна сигурност.***











9 786191 854608