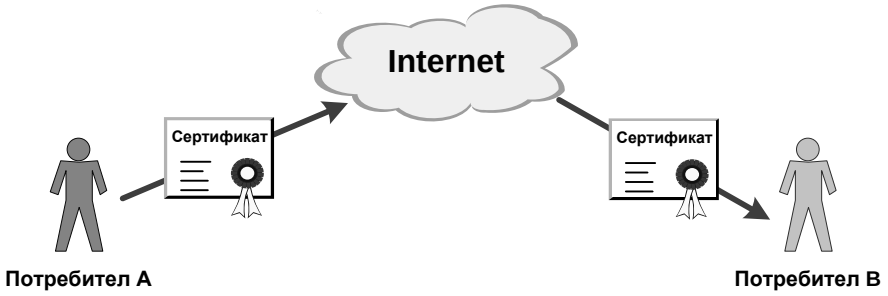
Десета глава

ВЪВЕДЕНИЕ В ПРИЛОЖНАТА КРИПТОГРАФИЯ

10.1. ОБЩИ СВЕДЕНИЯ[1] И ДЕФИНИЦИИ

Основната функция на криптографията е да защити данните срещу неоторизиран достъп или неоторизирано подменяне. Защитните си функции криптографията осъществява чрез идентификация на различните участници в един комуникационен процес посредством обмяна на *сертификати*[2]. Всеки сертификат еднозначно верифицира личността на неговия притежател или идентичността на използващия го обект. На Фиг. 10.1. е показана схема на обмен на сертификати.



Фиг. 10.1. Схема за обмен на сертификат

Потребител А изпраща през Internet-мрежата своя сертификат до потребител В. Потребител В проверява сертификата на А и, ако установи валидността му, приема, че действително комуникира с А.

Криптирането (*encryption*) е *метод за изменение на съобщение или документ по такъв начин, че неговото съдържание да стане неразпознаваемо, за всеки, който не познава метода за изменение*. **Декриптирането** (*decryption*) е обратна операция на криптирането и представлява *трансформиране на криптираните данни в явни.*

Криптирането и декриптирането са обекти на науката **криптография** (*cryptography*).

Правилото за преобразуване на данни в неразбираеми последователности от символи и тяхното правилно обратно възстановяване се нарича **криптографски алгоритъм** (*cryptographic algorithm*). **Явен текст** (*plain text*) се нарича *оригиналното съобщение преди да бъде обработено с определен криптографски алгоритъм*. **Криптиран текст** (*cipher text*) се нарича *получената символна поредица след криптирането на явния текст*.

---

[1] Основни принципи и детайлни примери на български език могат да се намерят в [31,55].

[2] Дигитални документи за удостоверяване на самоличност

Правилото за обмен на данни и използване на криптографския алгоритъм се нарича **криптографски протокол** *(cryptographic protocol).* Съвкупността от криптографски алгоритъм и криптографски протокол се обозначава с понятието **криптографска система** *(cryptosystem).*

Криптографски ключ (*key*) се нарича множество от числа или символи което се използва за криптиране или декриптириране на съобщенията [53]. Ключът е **секретен**, ако по криптографската система е задължително запазването му в тайна (*secret key*). Ако е известен, се нарича **публичен ключ** (public key).

Криптоанализът *(cryptanalysis)* е наука, занимаваща се с разработката на методи и средства за разкриване (на тайната) на криптографските системи и за оценка на тяхната сигурност. **Криптология** *(cryptology)* е обобщена дисциплина, включваща криптографията и криптоанализа.

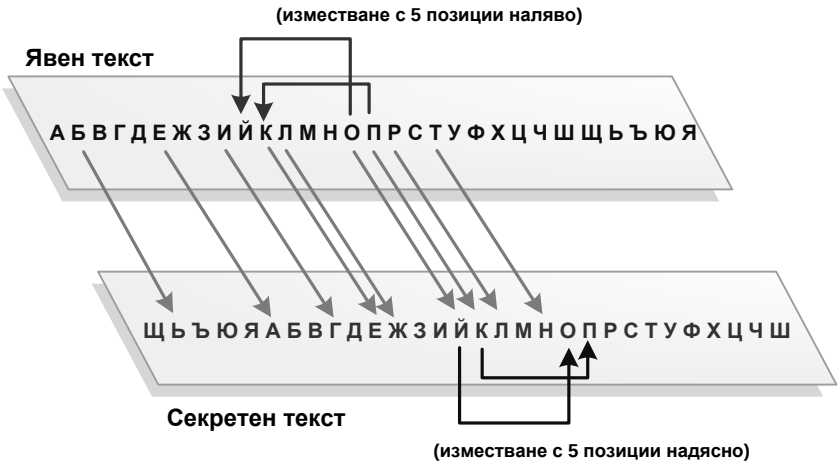
10.2. СЪЩНОСТ НА КРИПТИРАНЕТО

За реализиране на криптографски системи се използват две математически операции - **субституция и транспозиция.**

Субституция се нарича операция, извършваща еднозначно и обратимо съпоставяне на символ (блок от символи) в дадено съобщение с друг символ (блок от символи).

Криптографските алгоритми, работещи чрез субституции се наричат *субституционни алгоритми.*

Примерна схема [56] на субституция е дадена на Фиг. 10.1.



Явен текст (Message) = **П О Т Р Е Б И Т Е Л**

Криптиране на текста = **К Й Н Л А Ь Г Н А Ж**

Декриптиране на текста = **П О Т Р Е Б И Т Е Л**

Фиг. 10.1 Субституция

В представената примерна схема за съставяне на съобщенията е използвана българската азбука. На всяка буква се съпоставя пореден номер. Получава се подреждане, дадено в Таблица 10.1.

Таблица 10.1

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

Ако се преномерират буквите е възможно да се получи подредба, дадена в Таблица 10.2.

Таблица 10. 2:

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

При тази подредба може да се извърши субституция, по следния алгоритъм:

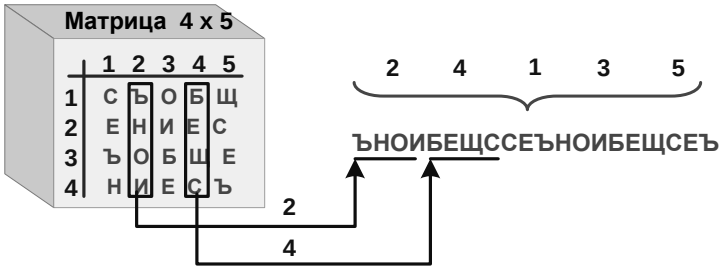
1. Взима се поредната буква от явния текст на съобщението.
2. Определя се нейния номер от таблица 10.1.
3. За този номер се взима съответната буква от таблица 10.2.
4. Изпраща се новата буква.

Пример: Трябва да се криптира думата ПОТРЕБИТЕЛ. В таблица 10.1 номерата на буквите са съответно 16,14,18,17,5,2,9,19,6,12. На тези номера в таблица 10.2 съответстват буквите КЙНЛАЬГНАЖ

Начинът на подреждане на азбуката във втората таблица (преномерирането) в случая е **ключът** на субституционния криптографски алгоритъм.

Транспозиция се нарича еднозначно и обратимо преобразуване на блок от краен брой октети[3] чрез пренареждане (пермутиране) на октетите, извършвано по определен криптографски алгоритъм и с определен ключ.

Криптографските алгоритми, работещи чрез транспозиции, се наричат *транспозиционни алгоритми.* При транспозиционните алгоритми символите в открития текст се запазват, т.е. не се маскират, но се променя тяхното местоположение в шифротекста. Пример за транспозиционен алгоритъм [56] е зареждане на една матрица по редове с открит текст с последващо получаване на криптиран текст посредством четене по колони е представен на фиг. 10.2.



Фиг. 10.2. Транспозиция

10.3 КЛАСИФИКАЦИЯ НА КРИПТОГРАФСКИТЕ АЛГОРИТМИ

Криптографските алгоритми се отличават с голямо разнообразие, което затруднява структурирането им в единна класификация. Използват се различни режими на тяхната реализация, включително многократно шифриране с една или няколко независими шифриращи процедури.

Целесъобразно е да се определят и да се разглеждат самостоятелно следните два **криптографски обекта:**

- *базов криптографски алгоритъм (базов шифър);*
- *режим на реализация на криптографски алгоритъм.*

Всеки конкретен криптографски алгоритъм се характеризира с използваните в него един или няколко базови криптографски алгоритми и с избраните режими за тяхната реализация.

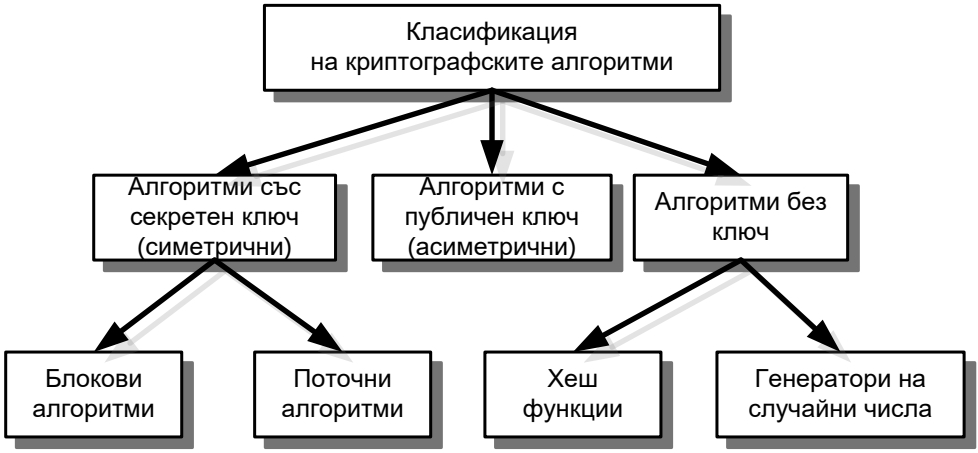
---

[3] Поредица от осем символа

Базовите криптографски алгоритми могат да се класифицират по различни признаци.

На Фиг. 10.3 алгоритмите са класифицирани по начина на използване на ключа. Първият тип алгоритми използват два ключа – секретен (частен) и публичен. Те се наричат **алгоритми с публичен ключ** (public key algorithms). Секретният ключ е известен само от неговият собственик, докато публичният ключ е достъпен за всички желаещи да изпратят криптирано съобщение на притежателя на секретния ключ. Чрез секретния ключ притежателят може да доказва своята самоличност. В този случай той ползва секретния ключ като **признак за идентификация.**

Алгоритмите с публичен ключ се използват за шифриране на малък обем данни.



Фиг. 10.3 Класификация на базови криптографските алгоритми

Пример: Всяка SIM карта притежава такъв ключ, с който се идентифицира пред обслужващата я мрежа.

Понеже ключовете, ползвани за криптиране и декриптиране на един документ, са различни, тези алгоритми са известни и под името **асиметрични алгоритми (**asymmetric algorithms**)**.

Вторият вид алгоритми са тези, които използват един ключ както за криптиране, така и за декриптиране. Този ключ е секретен и алгоритмите се наричат **алгоритми със секретен ключ** (secret key algorithms). Поради това, че за криптиране и декриптиране се използва един и същ ключ, той се нарича **симетричен**, а алгоритмите, които го ползват – **симетрични алгоритми**.

Съществуват *два основни типа* криптографски алгоритми (шифри) със секретен ключ: **блокови** и **поточни** криптографски алгоритми.

Блоковите алгоритми оперират върху блокове от данни с фиксиран или променлив размер, а **поточни** - последователно върху поток от символи или думи с произволна дължина.

Поточните алгоритми могат да бъдат класифицирани в две основни групи:

- *с независим ключов поток* и
- *със зависим ключов поток.*

В зависимост от начина на преобразуване на откритият текст блоковите алгоритми със секретен ключ могат да се разделят допълнително на *субституционни, транспозиционни* или *комбинирани.*

10.4 КРИПТОАНАЛИЗ

Криптоанализът има за цел декриптиране на части или целия секретен текст, без предварително да е известен ключа, с който е реализирано криптирането. **Основната цел** на криптоанализа е *откриването на секретния ключ*.

Криптоанализът ползва следните форми на атаки [55,47,51,55](методи):

1. **Атака единствено върху секретния текст.** Провежда се само върху криптиран текст, без да е достъпен явния, от който е произлязъл. За да се проведе успешна атака, е необходимо да се получи голям обем от криптираните съобщения, за да може да се търси в тях информация[4] за ключа.
2. **Атака върху част от секретен текст и кореспондиращ с него явен текст.** Поради по-богатата изходна информация и възможността за по-малко хипотези за ключа, провеждането на тази атака дава по-добри шансове за откриване на секретния ключ.
3. **Атака с избор на явен текст.** Прилага се в случаите, когато се разполага с криптиращото устройство, но още не е ясно как работи (анализ на черна кутия). Целта е да се разкрие тайната му. При тази атака е възможно предварително да се избере явен текст и да се криптира. След това се търсят зависимости за определяне на ключа посредством съпоставяне на явния текст с криптирания.
4. **Адаптивна атака с избор на явен текст.** Тя е специален случай на описаната в предходната точка атака. На основата на резултати от анализи на предишни криптирани поредици, при нея може да се променя динамично подлагания на криптиране явен текст.
5. **Адаптивна атака с избран криптиран текст.** Тя предполага, че провеждащият криптоанализа има на разположение хардуера за декриптиране, но не може да получи директно ключа за декриптиране. За да го открие, анализаторът използва части от прихванат криптиран текст и го подава за декриптиране. Върху получените резултати се строят работни хипотези за възможните стойности на ключа.

---

[4] Търсенето се извършва чрез статистически анализи и хипотези.

10.5. ОСНОВНИ ПРИНЦИПИ НА КРИПТОГРАФИЯТА

Основен принцип на криптографията е *съхраняването на секретността на ключовете на криптиращите алгоритми и разработването на сигурни методи за тяхната обмяна*.

Криптографията, като наука се развива в две направления, свързани с подобряване на сигурността на криптиращите системи:

1. *Математическа разработка на нови алгоритми за криптиране;*
2. *Намиране на нови ключове с увеличена дължина и сложност.*

Основни постулати на сигурните криптографски системи са:

- сигурността на системата не зависи от секретността на криптографския алгоритъм, а от секретността на използвания ключ за криптиране/декриптиране на данните;
- сигурната (силната) криптосистема има голямо пространство на ключовете;
- сигурната криптосистема произвежда криптиран текст, който има напълно случайно статистическо разпределение на символите в него и който не подлежи на статистически анализ;
- цената за „разбиване" на алгоритмите за криптиране трябва да е много по-висока от стойността на защитената информация;
- времето, необходимо за разбиване на алгоритмите за криптиране, трябва да е по-дълго от времето, за което криптираната информация представлява интерес.

За постигане на „строгост" на криптографските алгоритми съществена роля имат *ключовете за криптиране* и техните дължини. Един примерен ключ [56] за криптиране с дължина 56 бита има следния вид:

01101011100110011101010100111000111000100100010010101001

Дължината на криптиращия ключ се определя така, че да е защитена от атака с пълно изброяване (комбиниране, brute-force attack) на възможните стойности. Ето някой дължини на ключове и броя на възможните опити за тяхното отгатване по метода на пълното изброяване:

32 Bit key length ---> 2^{32} = 4.300.000.000
40 Bit key length ---> 2^{40} = 1.100.000.000.000
56 Bit key length ---> 2^{56} = 72.000.000.000.000.000
128 Bit key length ---> 2^{128}

34.000.000.000.000.000.000.000.000.000.000.000.000

За да се разработят добри криптиращи алгоритми е необходимо усилието на много хора, както по синтеза на конкретен алгоритъм, така и за криптоанализ, доказващ неговата сила. Това е една основна причина **силните алгоритми** за криптиране да се получават в рамките на **широка публичност и достъпност за криптоанализ**.

Както бе отбелязано в т. 10.3.1, алгоритмите за криптиране могат да се разделят по отношение на използваните ключове на два типа:

- симетрични;
- асиметрични.

10.6 СИМЕТРИЧНИ АЛГОРИТМИ ЗА КРИПТИРАНЕ

10.6.1 Определения и класификация

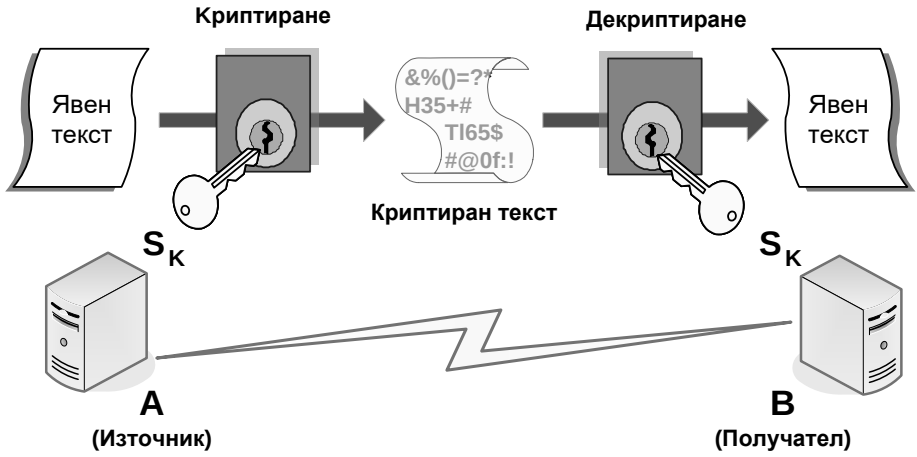
Симетричен алгоритъм за криптиране е всеки алгоритъм, който използва един и същи ключ в процеса на криптиране и декриптиране на информацията.

На Фиг. 10.4. е показана работата на такъв алгоритъм. Явният текст се криптира с ключ Sk, притежаван от получател А. Полученият криптиран текст се предава по линията до получател В. След приемането му, В го декриптира с помощта на същия ключ Sk.

Симетричните алгоритми, както вече бе казано в т.10.3.1, могат да се разделят на **поточни** и **блокови**..

Блоковите алгоритми използват следните техники за реализиране на функциите си:

- субституция;
- транспозиция;
- модулно събиране и умножение;
- линейни трансформации.



Фиг.10.4 Обща схема на симетричен алгоритъм

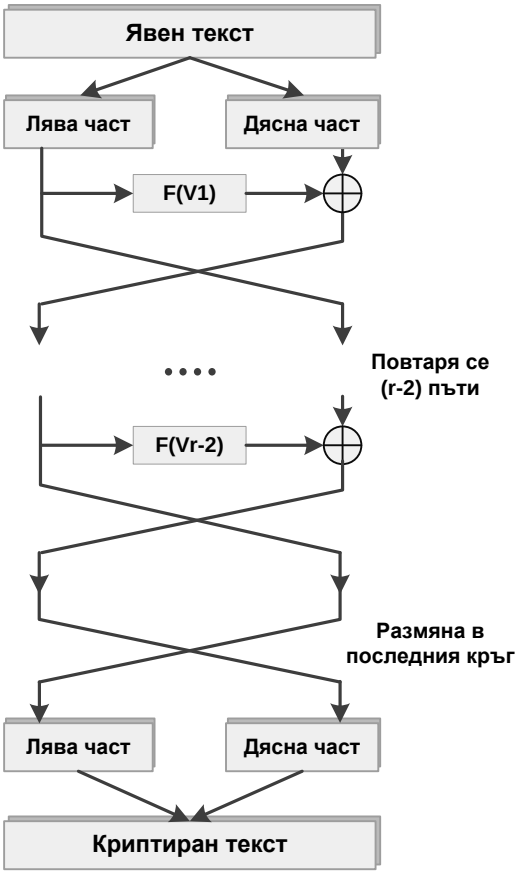
Основният принцип за криптиране в блоковите шифри се базира **на веригата на Фейстел.** Блокът с явен текст предварително се разделя на две части L (лява част) и R (дясна част). Тези две части се обработват в два

паралелни клона на алгоритъма. На всеки рунд[5] се криптира половината от подадения за обработка входен блок.

Както се вижда от Фиг. 10.4 в края на всеки рунд се извършва размяна на местата на лявата и дясната част от обработвания блок. Величината V_i се образува чрез прилагане на набор от математически операции (т.нар. ключова спецификация) върху използвания секретен ключ и представявава секретен ключ за криптиране на данните в поредния цикъл (нар. още рундов или циклов подключ). Функция F се нарича образуваща и се използва за обработване на блока от данни с изчисления ключ за съответния рунд. Броят на рундовете *K* е от 8 до 32. Увеличението на рундовете значително повишава устойчивостта на всеки блоков шифър срещу криптоанализ. При намиране на слабо място в алгоритъма в повечето случаи е достатъчно да се увеличи количеството на кръговете с 4-8, без да е нужно да се пренаписва самия алгоритъм.

Дадената схема е обратима и симетрична. Използваните операции XOR (сума по модул 2), са обратими при повторението си и инверсията на последния обмен на клона прави възможно декодирането на блока с веригата на Фейстел, но с инверсен порядък на параметъра V_i. Вижда се, че за обратимостта на веригата на Фейстел няма значение дали броя на кръговете е четен или нечетен.

[5] Поредица от повтарящи се операции за криптиране, прилагани неколкократно върху блок от битове.



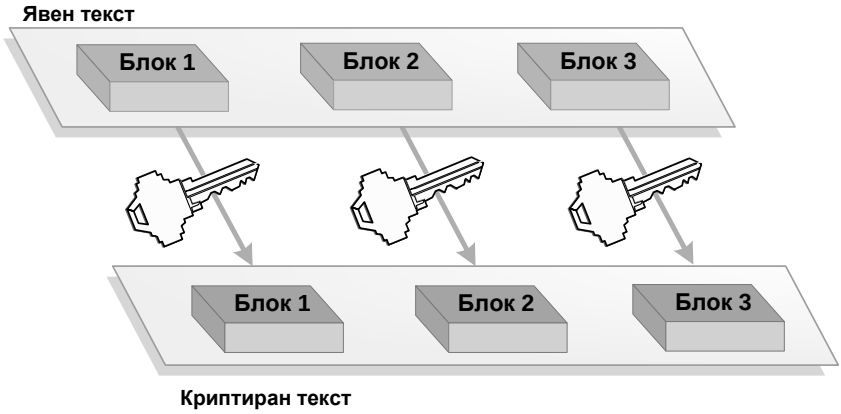
Фиг. 10.4. Верига на Фейстел

10.6.2. Режими на работа на блоковите криптографски системи

Блоковите криптографски системи се използват в различни режими на работа. Те са дефинирани в стандарти FIPS 81 (1980 December 2), ANSI X3.106-1983. Според този стандарт съществуват четири режима на работа:

1. *Електронна кодова книга* (Electronic Code Book - ECB),
2. *Блокова верига* (Cipher Block Chaining - CBC),
3. *Криприране с обратна връзка по вход* (K-bit Cipher FeedBack -CFB)
4. *Криптиране с обратна връзка по изход* (K-bit Output FeedBack - OFB).

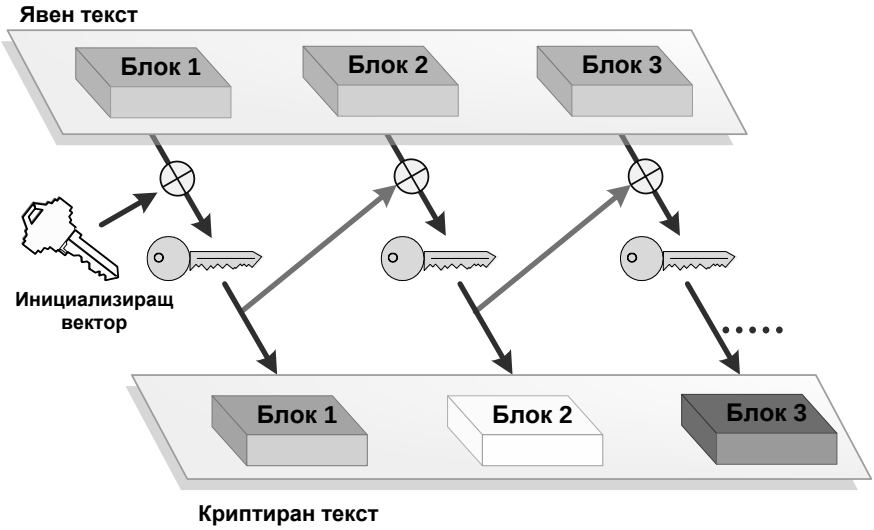
Електронна кодова книга (Electronic Code Block,ECB). Всеки блок от явния текст се криптира независимо от останалите с помощта на ключ, с дължина, равна на дължината на блока.



Фиг. 10.5. Режим „Електронна кодова книга”

Идентичните блокове явен текст се криптират по един и същ начин. Това е недостатък на този режим, защото може да се използва като основа за криптоанализ.

Режим на блокова верига (Chiper Blok Chaining – CBC). Всеки блок от явния текст се криптира с побитово сумиране по модул две (XOR) с получения в предишната операция криптиран блок (Фиг 10.6). За да стартира процедурата, се използва инициализиращ (начален) блок от битове. Инициализиращият вектор участва в криптирането на първия блок данни. Резултатът от това криптиране се използва за криптирането на втория блок и се предава по линията. При прихващане на част от съобщението в този режим не е възможно да се декриптира даден блок, защото не е известен текста, който е участвал в неговото криптиране.



Фиг. 10.6. Режим на работа в блокова верига.

10.6.3 Кратко описание на някои симетрични алгоритми

DES (Data Encryption Standard), съгласно Federal Information Processing Standard (FIPS) 46-1, е алгоритъм за криптиране на данни (DEA), дефиниран в стандарт ANSI X9.32. Разработен е от IBM. DEA, или както по често е наричан DES, е един от най-широко разпространените алгоритми за криптиране и най-често подлаган на криптоанализ. Сертифициран е през 1998. Алгоритъмът е базиран върху верига на Фейстел с 64 битови входящи блокове от данни. Използва ключ с дължина 64 бита. Всеки блок от криптираните данни се обработва в 16 рунда. От 1993 г. алгоритъмът се счита за компрометиран[6] и не е целесъобразно да се използва. В момента DES е заместен от алгоритъм AES (Advanced Encription Standart).

Triple-DES. В процеса на търсене на заместник на алгоритъм DES е разработен алгоритъм Triple-DES. Той криптира всеки входен блок от данни с три различни ключа. Начините, по които се извършва криптирането са: DES-EEE3: криптиране, чрез последователно прилагане на три различни ключа за криптиране. DES-EDE3: Изпълнение на последователни три операции от тип криптиране-декриптиране-криптиране (encryption - decryption - encryption), като за всяка от тях е използван различен ключ.

Blowfish е симетричен блоков алгоритъм, който се използва за заместване на DES или IDEA. Има променлива дължина на ключа - от 32 до 448

---

[6] Предложен е алгоритъм за разбиването му, които дава резултати за време, което се счита за критично по отношение на сигурността на алгоритъма.

бита. Разработен е през 1993 г. от Bruce Schneier като свободна и бърза алтернатива на съществуващите до момента алгоритми. Blowfish не е патентован и не подлежи на лицензни такси.

IDEA (International Data Encryption Algorithm) е разработен от Ascom Tech AG (Switzerland) и публикуван през 1993 г. Има входен 64 битов блок, който се криптира с 128 - битов ключ. Процесът на криптиране изисква 8 комплексни рунда. Структурата на алгоритъма е много добра както за хардуерна, така и за софтуерна реализация.

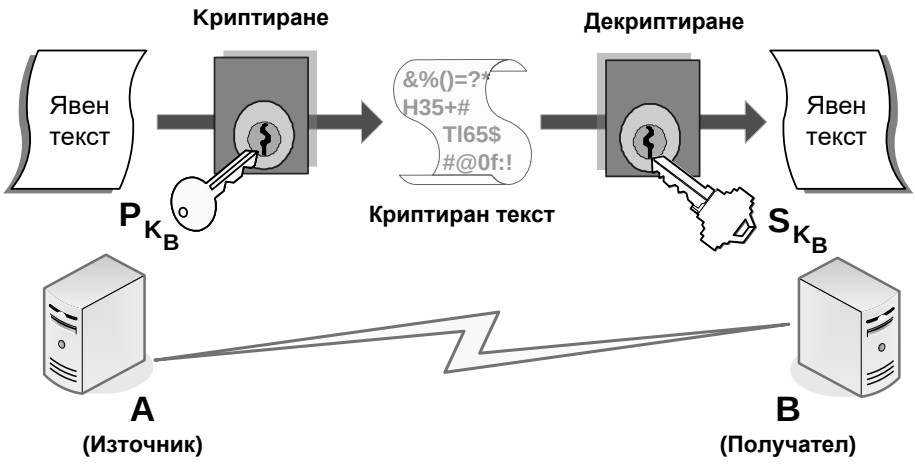
AEC (Advanced Encription Standart) е съвременният заместник на алгоритъм DES. Данните за криптиране се обработват под формата на 128 битови блокове. Ключовете с които се извършва криптирането, могат да са с дължина 128, 192 и 256 бита.

10.7 АСИМЕТРИЧНИ АЛГОРИТМИ

10.7.1 Общи сведения

Асиметричните алгоритми за криптиране (**Public-Key cryptography**) се използват различни ключове за криптиране и декриптиране. – криптиране с публично известен (**Public Key**) и декриптиране със секретен (**Private Key**) ключ.

Конфиденциалността на данните е свързана с възможността за криптиране на изпращаното съобщение (message) с публичния ключ и невъзможността да бъде декриптирано със същия ключ. Криптирано съобщение може да се прочете единствено от притежателя на секретния ключ.



Фиг. 10.7. Схема на асиметричен криптографски алгоритъм

В показаната схема на Фиг.10.7 **източникът А** криптира явния текст с публичния ключ (Pk_b) на **получателя В** и го изпраща по линията. **В** декриптира полученото съобщение с помощта на собствения си секретен ключ Sk_b.

Публичният ключ на **B** е достъпен за всеки, който иска да му изпрати криптирано съобщение[7]. Тази възможност е основно предимство на асиметричните системи пред симетричните. Липсата на необходимост за предаване на ключ за криптиране по засекретен канал, дава възможност за реализиране на публични криптиращи услуги. Свободната достъпност на публичния ключ, от своя страна, се явява и недостатък по отношение на автентификацията (установяването на самоличността) на изпращащия съобщението, защото може да се ползва от всеки. Този проблем налага допълнително организиране на процедура по автентификация.

Автентификацията на изпращащия може да се реализира чрез прилагане на публичния и секретния ключ в обратен ред. За криптиране на съобщението изпращащата страна използва собствения си секретен ключ. Получената криптограма се криптира повторно с публичния ключ на получателя. Криптираното съобщение се изпраща по открит комуникационен канал. Получателят на съобщението го декриптира, като прилага собствения си секретен ключ. След това декриптира повторно с публичния ключ на изпращащия и така получава оригиналния явен текст. Публичният ключ на изпращащия е достъпен за свободно ползване.

Схемата на работа на асиметричните алгоритми ще бъде илюстрирана с алгоритъм RSA.

10.7.2 RSA[8] (Rivest, Shamir, Aldeman)

RSA е класически асиметричен алгоритъм, чиято сигурност се определя от факторизацията на големи прости числа. Сигурността на алгоритъма се базира на следните предпоставки:

- много е трудно да се подложи на криптоанализ основата на алгоритъма - факторизацията на големите числа;
- сигурността се основава на притежаването на секретен ключ единствено от едно лице - неговия собственик;
- секретният ключ не се предава никога по какъвто и да е било канал.

Името RSA произлиза от първите букви на създателите на алгоритъма – математиците от *Масачузетският Технологичен Институт* Ron **R**ivest, Adi **S**hamir, и Leonard **A**dleman. В момента RSA се използва като част от SWIFT (*Society for Worldwide Interbank Financial Telecommunications*) – организация за световна комуникация между банки и финансови организации като стандарт за

---

[7] След като веднъж се криптира дадено съобщение с публичния ключ, то не може да се декриптира с него.

[8] Подробности за алгоритъма и приложенията му са достъпни на адрес: http://www.rsasecurity.com/

международен финансов трансфер и е възприет от ANSI X9.31 стандарта за банковата индустрия. Използва се от американските и австралийските банки.

Друго приложение на RSA е в Internet-браузърите, както и в устройствата за мрежова комуникация, смарт-картите и методите за електронно разплащане. Виртуални частни мрежи (Virtual Private Networks VPNs).

Алгоритъм RSA работи по следния начин:

1. Генерират се две големи произволни (и различни) прости числа '**p**' и '**q**'.
2. Изчислява се **n = p.q** и **ϕ(n) = (p-1).(q-1).**
3. Избира се произволно цяло число '***e***', $1 < e < \phi$, така че най-големия общ делител на числата '*e*' и 'ϕ' да е 1.
4. Намира се[9] цяло число '**d**', такова, че **e.d ≡ 1 (mod ϕ)**.
5. Публичният ключ на лицето А е (**n , e**), а частния – **(n,d)** .

Целите числа '**e**' и '**d**' при генерацията на ключове в RSA се наричат *криптиращи и декриптиращи експоненти,* а 'n' се нарича *модул (modulus).*

Пример (с много малки дължини на числата):

1. Лицето А избира простите числа $p = 2357$, $q = 2551$ и изчислява
   $n = p.q = 6012707$
   $\phi = (p-1)(q-1) = 6007800$.
2. После *А* избира $e = 3674911$ и намира $d = 422191$ така, че $e.d \equiv 1 \pmod{\phi}$.
3. *Публичният ключ на А е двойката (n = 6012707; e= 3674911), а частният (тайният) му ключ е d = 422191;*

Криптиране: *За да се криптира съобщение m = 5234673, потребител Б изчислява числото* $c = m^e \bmod n = 5234673^{3674911} \bmod 60127070 = 3650502$, *което се изпраща на А.*

Декриптиране: *За да се декриптира съобщението, А изчислява* $c^d \bmod n = 3650502^{422191} \bmod 6012707 = 5234673$.

Кои числа ще образуват публичния ключ се избира от създателя на числата. Следователно могат да се публикуват и числата **(n,d)** като публичен ключ. Тази обратимост дава възможност да се реализира схема за автентификация на изпращащия съобщението (**автентификация на изпращащия**).

Основното предимство на асиметричните алгоритми е възможността за разработване на лесна система за управление на обмена на ключове. След като веднъж се генерират секретните ключове, е необходимо те да бъдат пазени само от едно лице - техния собственик, с което се повишава сигурността на алгоритъма по отношение на атака, насочена към обмена на ключове.

Недостатъците на асиметричните алгоритми са свързани с по-бавната скорост на работа. Например за един и същ обем данни, при една и съща опитна конфигурация са получени следните резултати:

---

[9] За целта се използват математически операции, които са извън рамката на настоящата книга.

- при хардуерна реализация: IDEA[10] е от 1000 до 10 000 пъти по-бърз от RSA;
- при софтуерна реализация IDEA е 100 пъти по-бърз от RSA.

В таблица 10.3 е дадено сравнение на симетрични и асиметрични алгоритми [51,56].

Таблица 10.3

	Симетрични алгоритми	**Асиметрични алгоритми**
Предимства	лесна генерация на ключа; добра производителност за големи обеми от данни;	не изисква обмен на секретен ключ; използва се една единствена двойка ключове;
Недостатъци	необходимост от обмен на секретни ключове; необходимост от уникален ключов за всяка двойка партньори;	лоша производителност; трудна генерация на ключовете;
Използват се за:	криптиране/декриптиране на големи масиви от данни; осигурява конфиденциалност на данните;	секретен обмен на ключове за симетрични алгоритми; цифрови подписи интегритет на данните;

10.8 ХЕШ ФУНКЦИИ

10.8.1 Определения

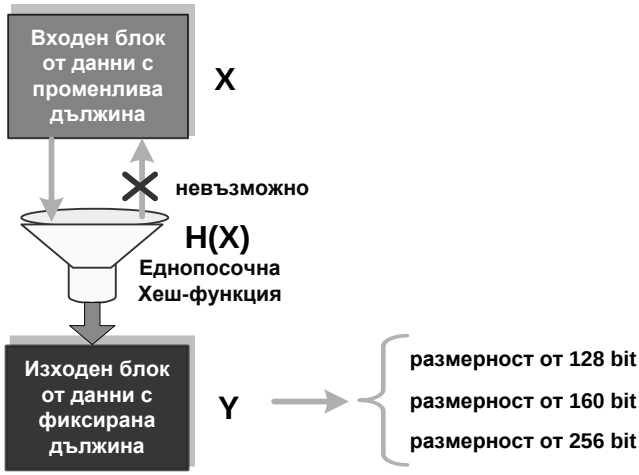
Еднопосочна функция (One-Way Function) - се нарича всяка функция, чието изчисление в права посока е лесно, но е много трудно да се изчисли нейната обратна (реверсна) функция.

Хеш функция (One-Way Hash Function) се нарича *всяка еднопосочна функция, която взима променлив по дължина блок от входни данни и създава блок от данни с фиксирана дължина*. След прилагането на хеш функцията от изходния блок по никакъв начин не може да се изчисли стойността на данните във входния блок. За два блока от данни, които се различават само по един бит, получените хеш-стойности също е трябва да се различават. Постигането на това свойство означава, че няма два различни текста с една и съща хеш-стойност.

Следователно хеш функцията е **свободна от колизии** (collision-free).

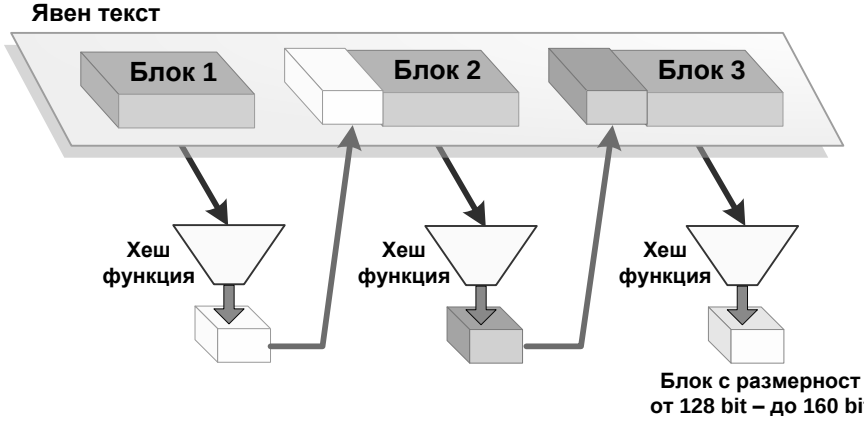
Еднозначността и еднопосочността на хеш функцията дават възможност тя да се използва при формиране на т.нар. цифров подпис (сигнатура). Начин за формиране на цифрова сигнатура е представен на Фиг. 10.8.

[10] Вид система за симетрично криптиране.



Фиг 10.8. Реализиране на хеш функция

Всеки файл се разделя на поредица от блокове, върху които последователно се прилага избраната хеш функция. Всеки блок от данни може да се раглежда като матрица с определна размерност. Елементи на матрицата са отделните битове на обработвания блок. Матрицата се образува чрез сумиране на редовете или колоните посредством математическа операция – сума по модул 2, докато се получи матрица (блок) с фиксирана дължина (128, 160, 256 бита). Тази операция се извършва за всички блокове от файла. Получените на всяка стъпка блокове с хеш-стойности (изходният блок с фиксирана дължина) се добавят към съответния входен блок. Принципът за формиране на хеш стойност (сигнатура), за файл от три блока е представена на Фиг. 10. 9.



Фиг. 10.9 Получаване на хеш стойност за три блока с данни

10.8.2 Съвременни алгоритми за формиране на Хеш-стойности

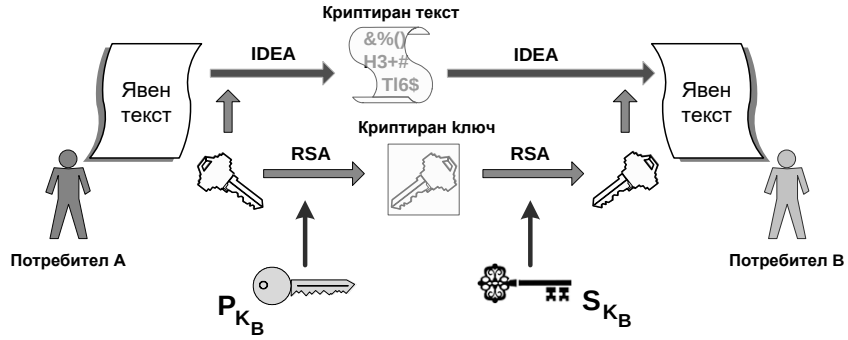
SHA-1 е стандарт за хеш-функция описана във FIPS 180-1. Използва се съвместно с Digital Signature Algorithm (DSA) съгласно спецификацията на стандарт Digital Signature Standard (DSS) за изработване на електронно подписани документи

MD4 и MD5 са алгоритми за създаване на хеш стойност. с дължина съобщение е 128 бита. Двата алгоритъма са описани в RFC 1320 и RFC 1321.

MD4 е разработен през 1990 г. **а MD5** е разработен през 1991 г. Той е подобрение на MD4, но е малко по-бавен от него. Важно е да се отбележи, че в оригиналната версия на RFC, описваща MD5, има грешки. Коректният алгоритъм се нарича MD5a..

10.9 ИЗГРАЖДАНЕ НА КРИПТОГРАФСКИ СИСТЕМИ

За да се изгради силна система за криптиране, е необходимо да се комбинират симетрични и асиметрични алгоритми. Реализирането на цялостна криптографска система ще да бъде обяснено чрез няколко схеми.



Фиг. 10.10. Принципна схема за изграждане на криптографска система

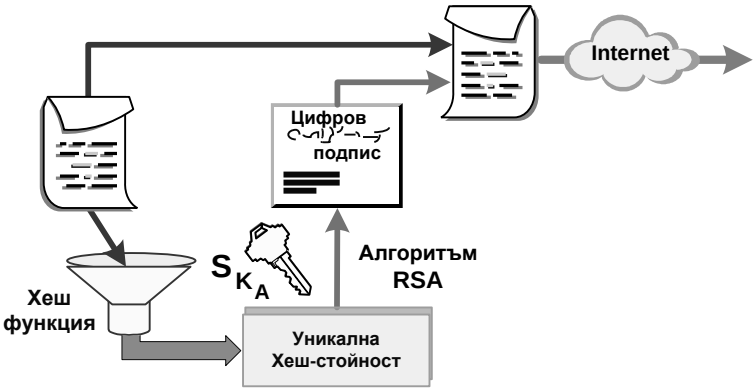
На Фиг. 10.10 е дадена принципна схема за реализация на криптографска система. В случая потребител А иска да изпрати криптиран текст до потребител **В**. За целта, А трябва да използва симетричен криптиращ алгоритъм**.** В този случай това е IDEA. Потребител А криптира явния текст с този алгоритъм като използва избран от него **секретен ключ**. Криптираният текст се изпраща на **В**. За да декриптира текста, **В** трябва да разполага с използвания от **А** секретния ключ. Потребител А изпраща секретния ключ по незащитения канал, като предварително го криптира с публичния ключ (за RSA алгоритъм) на **В**. Потребител **В** получава този криптиран секретен симетричен ключ и го декриптира със своя секретен ключ. След това, чрез декриптирания ключ за симетричния алгоритъм**, В** декриптира и получения текст от **А**.

В разгледаната схема съществуват два проблема:

- потребител **B** не може да идентифицира еднозначно A,
- потребител **B** не може да разбере, дали документът не е подменен при преминаването му през мрежата.

За да се извърши идентификацията на A, е необходимо да се създаде **цифров подпис** за документите, които той изпраща. Цифровият подпис на A удостоверява неговата идентичност и гарантира интегритета на изпратените от него данни.

Пълният процес на създаване на цифров подпис е представен на Фиг. 10.11[56].



Фиг. 10.11 Изготвяне на цифров подпис

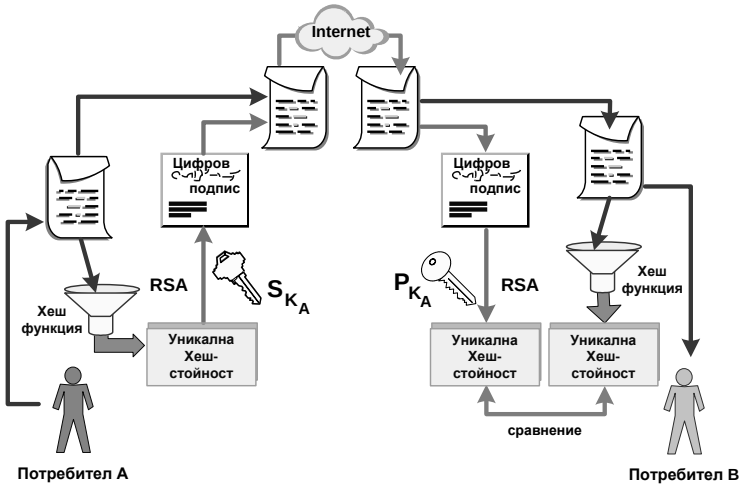
Цифровият подпис се получава като:

- Изпращащият (**A**) изготвя електронни документи и ги обработва с хеш функция. На този етап се получава уникален блок от битове, еднозначно описващ електронния документ.
- **A** криптира получената стойност след прилагането на хеш функцията с алгоритъм RSA като използва собствения си секретен ключ (Sk_a). След това изпраща съобщението като добавя и цифровия подпис.

След получаване на съобщението получателят трябва да верифицира два негови компонента:

- **интегритета на данните** - тоест, дали по време на преминаването по публичния канал, не са били подменени и
- **автентичността** (дали наистина са изпратени от **A**).

За да извърши проверка за самоличността на **A**, **B** получава от публична база данни публичния ключ на **A** за алгоритъма RSA и с него декриптира получения от **A** криптиран цифров подпис. При декриптирането се получава и хеш-стойността, изчислена от **A** за изпратения документ. Потребител **B** декриптира получения от **A** текст. За да провери интегритета, **B** изчислява хеш-стойността на получения открит текст. Ако изчислената стойност съвпадне с получената, то това е гаранция, че съобщението не е било променяно, и че потребител **A** е използвал своя секретен ключ за алгоритъм RSA. Последното идентифицира еднозначно потребител A.



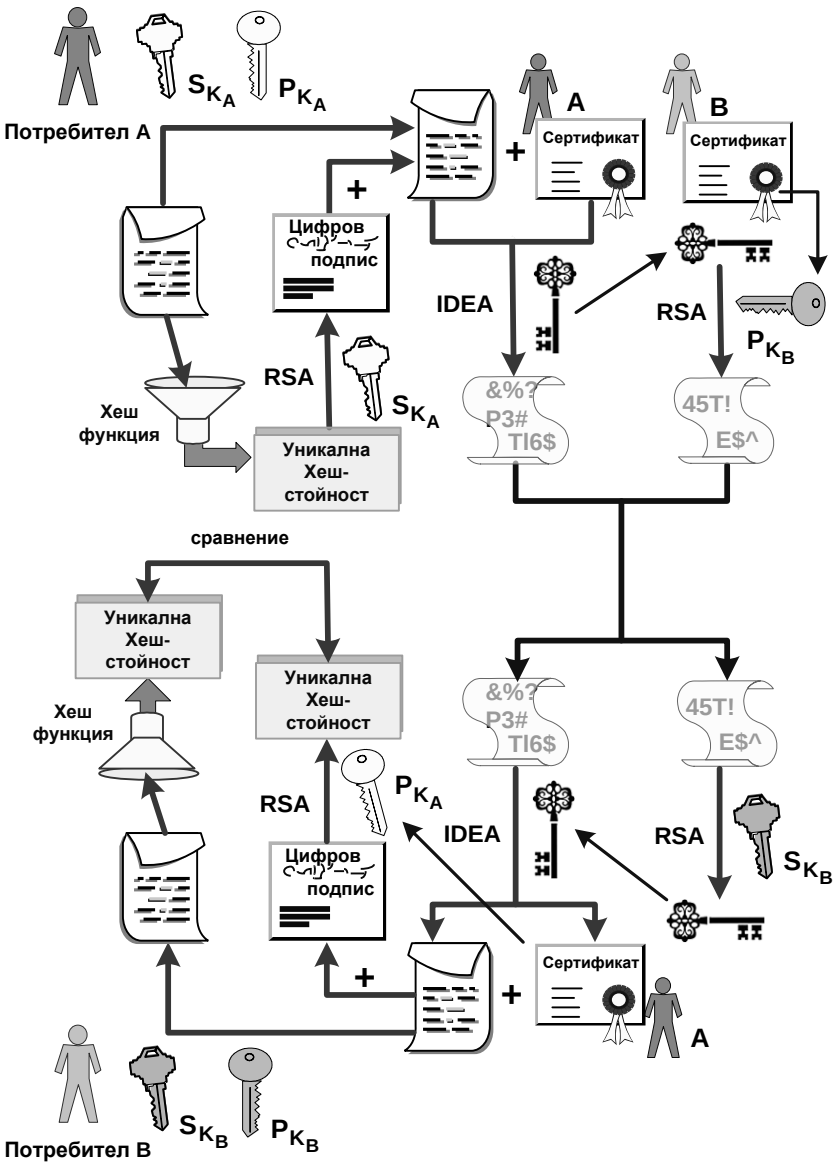
Фиг. 10.12 Обмен на криптирано съобщение с проверка на автентификация

Таблица 10.4.

	Процедури на функциониране на криптографска система
1	Потребител А използва хеш функция за получаване на уникален цифров подпис на документа, когото иска да изпрати.
2	Потребител А криптира получения цифров подпис със секретния си ключ и така формира дигитална сигнатура на документа.
3	Потребител А генерира случаен ключ за използвания симетричен алгоритъм и с него криптира данните, получената дигитална сигнатура и копие от сертификата си, съдържащ публичния ключ на А за RSA.
4	Ключът за симетричния алгоритъм А криптира с публичния ключ на В и го изпраща по линията, заедно с криптираното съобщение и криптираната сигнатура и копие от сертификата си.
5	Потребител В получава криптираните данни от А. След като използва личния си секретен ключ за RSA, В декриптира изпратения ключ за симетричния алгоритъм.
6	С получения симетричен ключ потребител В декриптира данните изпратени от потребител А, както и сертификатите и дигиталната сигнатура.
7	От сертификата[11] на А, потребител В получава неговия публичен ключ за RSA. С получения публичен ключ В декриптира изпратената криптирана дигитална сигнатура, която съдържа стойността от хеш функцията, приложена от потребител А.
8	Потребител В изчислява хеш функция за получения от А документ. Изчислената стойност се сравнява с получената хеш стойност от А.
9	Ако сравнението на двете стойности е успешно, т.е. те са равни, се счита че данните са автентични и е запазен техния интегритет и конфиденциалност.

[11] Разгледан е в следващата глава.

Пълният процес на изготвяне на криптирани данни - изпращане, получаване и верификация на всички компоненти на криптографската система, е даден на Фиг. 10.13. [51,56]и е обяснен в таблица 10.4:



Фиг. 10.13 Пълна криптографска система.

КОНТРОЛНИ ВЪПРОСИ:

1. Опитайте се да създадете собствена субституционна криптосистема, която да работи върху българската азбука, и с нея криптирайте и декриптирайте явния текст „сигурност и надеждност в комуникациите”.
2. Опитайте се да създадете собствена транспозиционна криптосистема и с нея криптирайте и дектирпитрайте явния текст „сигурност и надеждност в комуникациите”.
3. Напишете на SDL (или друг език за моделиране на алгоритми) процедурата за създаване на хеш-стойност.
4. Опитайте се да създадете криптосистема, работеща с RSA алгоритъм, като за прости числа използвате 3 и 5. За да решите тази задача е необходимо да определите връзката между **e и d** експериментално.
5. Предложете схема за симетрично криптиране базирана върху обработване на 8 битови числа с един 8 битов ключ. Какви са слабите места на тази схема според вас? Опишете ги.
6. Как според вас се реализира схемата (софтуерно и хардуерно) показана на Фиг. 10.13, при използване на Internet-мрежата?
7. Открийте в кои RFC документи са разгледани MD5 и MD4. Проучете намерените документи за да придобиете представа за тези методи. Опитайте да намерите продукти, които ги ползват и ги опишете.
8. Опитайте се да създадете модел на криптографска система за криптиране на видеопоток. Ще настъпят ли промени във Фиг. 10.13?
9. Кои данни във фирма, предлагаща комуникационни услуги, според вас трябва да се криптират?
10. Проучете какви алгоритми са използват за криптиране в GSM комуникациите и как се прилагат.
11. В някои фирмени сайтове, от които може да се изтегля (download) свободно софтуер, за всяка програма в дадена допълнителна контролна стойност за всяка програма. Най-често това е MD5. За какво се използва тази стойност?