Symetrické šifry

- blokové a prúdové šifry, základný princíp
- Feistelova štruktúra (DES, TDES, XTEA)
- algoritmus AES, história a opis štruktúry
- modulárna aritmetika, konečné polia, princíp a využitie (S-box)
- módy blokových šifier (ECB, CBC, OFB, ..., CTR, ..., GCM)
- iné používané šifry (DES, TDES, XTEA, ChaCha20, ...)

Primárny zdroj informácii k dnešnej prednáške:
[1] prof. Ing. Dušan Levický, CSc.
APLIKOVANÁ KRYPTOGRAFIA
od utajenia správ ku kybernetickej bezpečnosti
Elfa, Košice, 2018 (str.62-107)

Základné techniky na zníženie redundancie v otvorenom texte sú

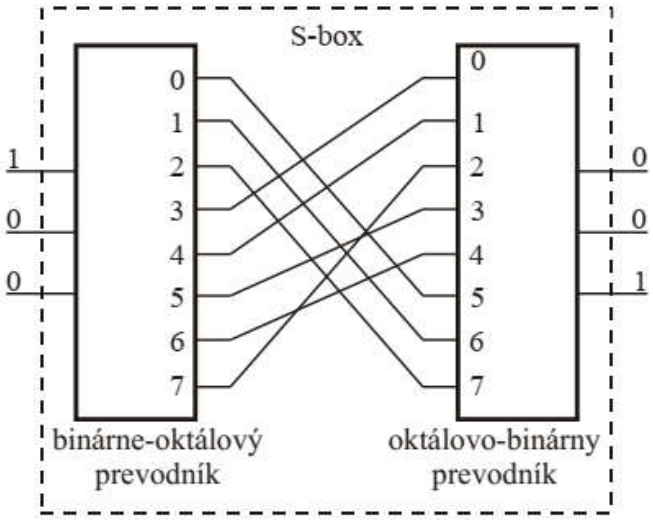
• konfúzia (confusion)
• difúzia (diffusion).

Konfúzia prerušuje vzťahy medzi zašifrovaným a otvoreným textom, čo komplikuje kryptoanalýzu zašifrovaného textu. Najjednoduchší spôsob realizácie konfúzie spočíva v substitúcii. Ako príklad môže slúžiť jednoduchá Cézarova šifra, v ktorej sú rovnaké znaky otvoreného textu nahradené vždy rovnakými znakmi zašifrovaného textu.

Difúzia rozprestiera redundanciu otvoreného textu tak, že ju rozptyľuje po celej dĺžke zašifrovaného textu. Najjednoduchší spôsob realizácie difúzie je **transpozícia**, resp. **permutácia**, ktoré možno dosiahnuť napr. transpozičnými šiframi.

Moderné šifry využívajú **viacnásobnú** konfúziu a difúziu (napr. vo forme viacerých iterácií – **rund**), ktoré umožňuje dosiahnuť rovnomernejšie rozptýlenie redundancie po celej dĺžke zašifrovaného textu.

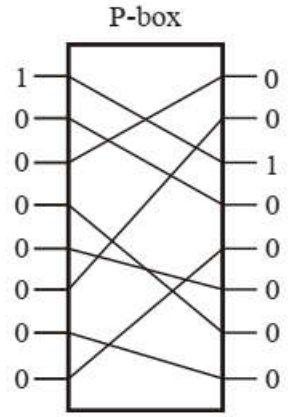
Substitúcia (substitution) je z pohľadu Shanonovej koncepcie konfúzie **nelineárna transformácia**. Ak uvažujeme, že n vstupných bitov reprezentuje jeden z 2^n vstupných znakov, potom substitúcia je operácia transformácie jedného znaku na iný znak z danej množiny znakov. V binárnom tvare ide o substitúciu jednotlivých bitov vstupného znaku (kódu) inými bitmi, čím vzniká výstupný znak (výstupný kód). Blok na realizáciu substitúcie sa označuje ako **S-box**.



Vstup	000	001	010	011	100	101	110	111
Výstup	101	110	111	000	001	011	100	010

Obr. 4.2 Príklad substitúcie pre n=3

Permutácia (permutation) je zmena poradia znakov, v binárnom vyjadrení zmena poradia, resp. usporiadania bitov. Príklad realizácie permutácie pomocou P-boxu s 8 vedeniami je uvedený na Obr. 4.3. Technika založená na permutácii nezvyšuje bezpečnosť, čo možno demonštrovať jednoduchým postupom. Ak na vstup P-boxu privedieme postupnosť tzv. trikových správ obsahujúcich iba jednu 1 (ostatné bity sú 0), potom jednoduchým overením výstupov možno odhaliť danú permutáciu, resp. prepojenia vstupov s výstupmi. Takto možno ilustrovať tézu, že bezpečnosť kryptografického systému nemožno postaviť na technických prostriedkoch, resp. na jeho architektúre.



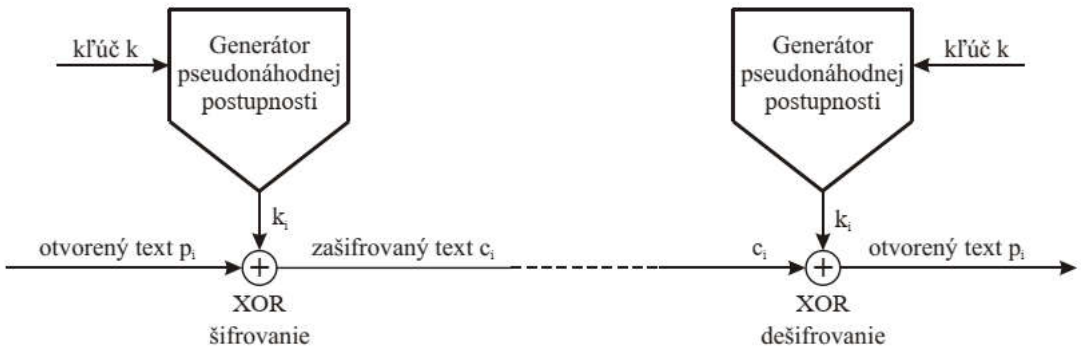
Obr. 4.3 Príklad P–boxu

Súčasné kryptografické systémy využívajú **kombinované architektúry**, ktorých súčasťou sú S-boxy a P-boxy, ako bloky realizujúce substitúciu a permutáciu. **Bezpečnosť** kryptografických systémov **s kombinovanou architektúrou je väčšia** ako bezpečnosť jednotlivých blokov.

Z pohľadu **realizácie šifrovania** možno moderné šifry rozdeliť do dvoch základných skupín a to na:

• **prúdové šifry** (stream ciphers)
• **blokové šifry** (block ciphers).

Prúdové šifry transformujú každý symbol otvoreného textu pi hneď na zodpovedajúci symbol zašifrovaného textu c_i. Najjednoduchšia implementácia binárnej prúdovej šifry je znázornená na Obr. 4.4.



Obr. 4.4 Princíp prúdovej šifry

Bezpečnosť binárnej prúdovej šifry závisí od generátora pseudonáhodnej postupnosti. Z tohto pohľadu možno uvažovať o dvoch extrémoch:

1. ak generátor pseudonáhodnej postupnosti produkuje postupnosť symbolov 0, c_i=p_i XOR 0=p_i, t. j. zašifrovaný text je rovný otvorenému textu a šifrovanie je neúčinné

2. ak generátor generuje **ideálnu náhodnú** (nie pseudonáhodnú) postupnosť, potom sa získa ideálna prúdová šifra s perfektnou bezpečnosťou, resp. binárna Vernamova šifra. Pri jednorazovom použití náhodnej postupnosti ide o binárnu šifru s jednorazovým slovníkom.

V reálnych podmienkach sa pracuje s **pseudonáhodnými postupnosťami**, preto bezpečnosť prúdových šifier leží medzi týmito hranicami.

Pozri prácu G. Vernama (https://en.wikipedia.org/wiki/Gilbert_Vernam) a US patent z roku 1919 (https://patents.google.com/patent/US1310719)

Výhody prúdových šifier sú najmä:

• veľká rýchlosť šifrovania
• malé šírenie chýb.

Veľká rýchlosť šifrovania vyplýva najmä z toho, že každý symbol otvoreného textu je šifrovaný nezávisle na ostatných symboloch otvoreného textu, teda symbol otvoreného textu môže byť šifrovaný hneď po jeho prečítaní šifrovacím zariadením. Doba šifrovania každého symbolu otvoreného textu je daná len technickými parametrami šifrovacieho zariadenia.

Malé šírenie chýb je dané podstatou prúdovej šifry. Pretože každý symbol otvoreného textu je kódovaný samostatne, chyba v šifrovacom procese ovplyvní iba tento symbol a nenaruší šifrovanie ďalšieho symbolu.

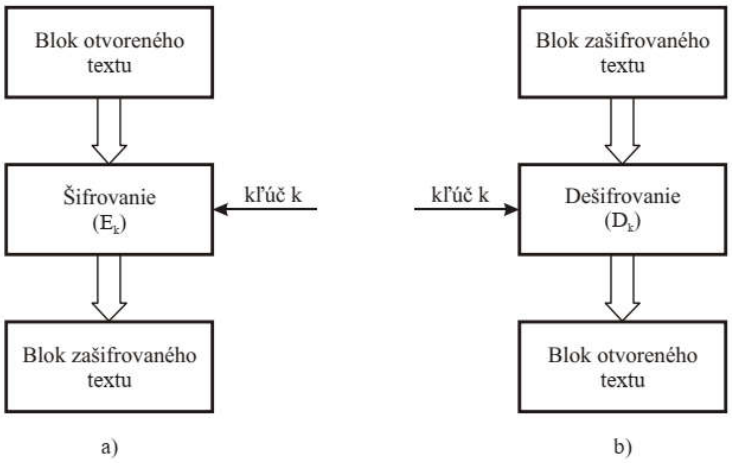
Prúdové šifry majú však tieto nevýhody:

• nízka úroveň difúzie
• malá odolnosť voči narušeniu integrity šifrovanej správy.

Nízka úroveň difúzie vyplýva z toho, že celá informácia symbolu otvoreného textu je transformovaná do jedného symbolu zašifrovaného textu. Zašifrovaný text teda vykazuje rovnaké frekvenčné a štatistické vlastnosti ako otvorený text, čo uľahčuje kryptoanalýzu.

Malá odolnosť voči narušeniu integrity šifrovanej správy vyplýva tiež z princípu prúdovej šifry, kde sa šifrovanie realizuje po symboloch otvoreného textu. Ak je šifra prelomená, nepovolaná osoba môže ľubovoľne modifikovať vysielané správy bez toho, aby to na prijímacej strane bolo možné detegovať.

Blokové šifry transformujú skupinu symbolov otvoreného textu na skupinu symbolov zašifrovaného textu, teda šifrovanie sa realizuje po blokoch otvoreného textu. Veľkosť bloku obvykle nie je podstatná. Princíp blokovej šifry je uvedený na Obr. 4.5.



Obr. 4.5 a) proces šifrovania blokovou šifrou, b) proces dešifrovania blokovou šifrou

V moderných blokových šifrách je **dĺžka bloku** otvoreného textu **64**, resp. **128 bitov**, pričom blok zašifrovaného textu má rovnakú veľkosť.

K výhodám blokových šifier patria najmä:

• vysoká úroveň difúzie
• dobrá imunita voči narušeniu integrity.

Vysoká úroveň difúzie vyplýva z toho, že informačný obsah otvoreného textu difunduje do niekoľkých symbolov zašifrovaného textu.

Dobrá imunita voči narušeniu integrity vyplýva z toho, že pri šifrovaní po blokoch nie je možné implementovať naviac ani jeden symbol. Týmto zásahom by sa zmenila dĺžka bloku a dešifrovanie by toto narušenie odhalilo.

Blokové šifry majú však aj nevýhody ktoré spočívajú najmä v:

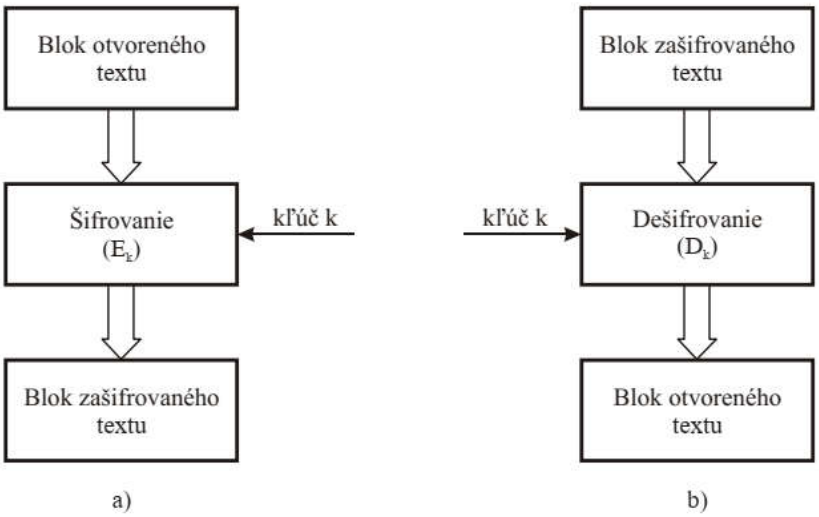
• oneskorení, ktoré vzniká pri šifrovaní
• šírení chýb.

Oneskorenie pri šifrovaní blokovými šiframi vyplýva z toho, že proces šifrovania sa môže začať až keď je vytvorený úplný blok symbolov otvoreného textu.

Šírenie chýb pri šifrovaní blokovými šiframi je spôsobené tým, že chyba v jednom znaku otvoreného textu sa premietne do celkového zašifrovaného textu.

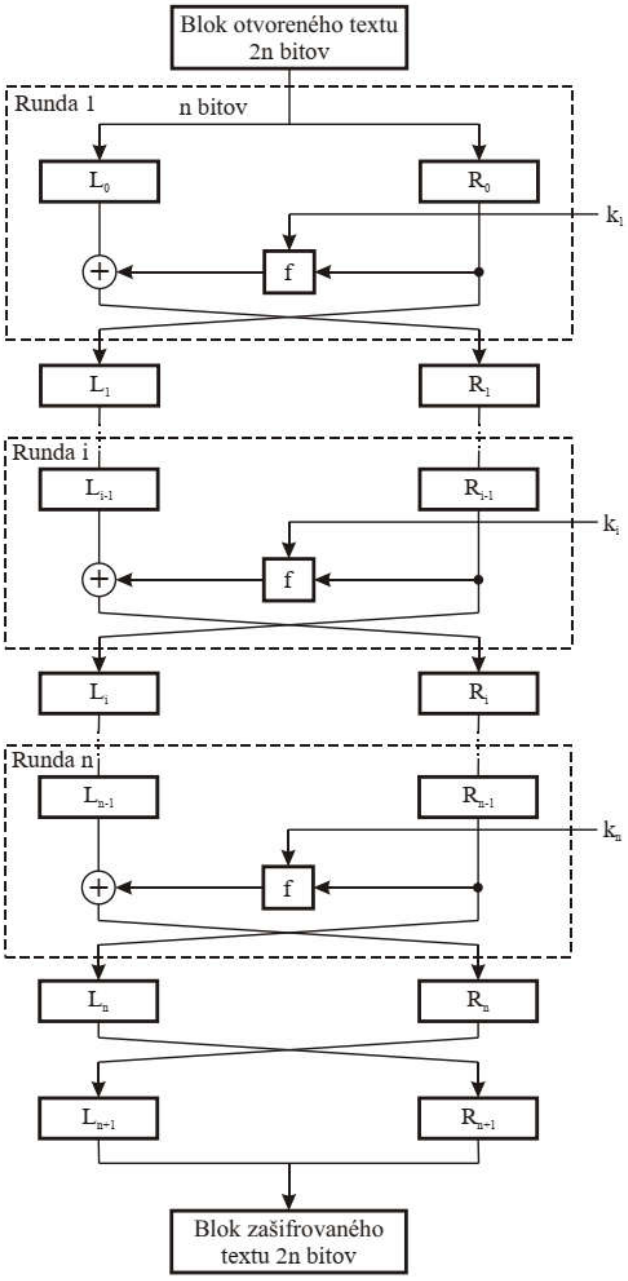
Symetrické blokové šifry

V moderných blokových šifrách je dĺžka bloku otvoreného textu **64, resp. 128 bitov**, pričom blok zašifrovaného textu má rovnakú veľkosť. Princíp blokovej šifry je uvedený na Obr.4.5.



Obr. 4.5 a) proces šifrovania blokovou šifrou, b) proces dešifrovania blokovou šifrou

Feistelova šifra (štruktúra)
Veľa algoritmov symetrických blokových šifier používa štruktúru, ktorá sa označuje ako Feistelova bloková šifra (1)



Obr. 5.1 Feistelova bloková šifra

(1) **Horst Feistel** – nemecký emigrant, ktorý prišiel do USA v roku 1934, autor šifrovacieho systému **LUCIFER**, ktorý po úprave bol r. 1976 prijatý ako **štandard DES**

Feistelova bloková šifra využíva striedavo **substitúcie** a **permutácie**, ktoré realizujú funkcie **konfúzie** a **difúzie**.

Konkrétna realizácia blokovej šifry na báze štruktúry Feistelovej šifry je daná voľbou týchto parametrov:
• **veľkosť bloku**
• **dĺžka kľúča**
• **počet rúnd**
• algoritmus **generovania kľúčov rúnd**
• **zložitosť operácií** v runde.

Veľkosť bloku ovplyvňuje bezpečnosť šifry. Väčší rozmer bloku zvyšuje bezpečnosť, ale znižuje rýchlosť šifrovania a dešifrovania. Typická veľkosť bloku je 64 bitov (štandard DES), ale v súčasnosti je 128 bitov (štandard AES(1),). Je potrebné však poznamenať, že tento parameter je tiež závislý na použitých technických prostriedkoch a na efektívnosti implementácie.

Dĺžka kľúča tiež ovplyvňuje bezpečnosť, takže väčšia dĺžka kľúča zvyšuje bezpečnosť, ale znižuje rýchlosť šifrovania a dešifrovania. Typická dĺžka kľúča je 64 bitov, ale novšie šifry používajú väčšiu dĺžku kľúča.

Počet rúnd znamená počet opakovaní rovnakej postupnosti operácií, pretože každá runda má rovnakú štruktúru. Počet rúnd je v rôznych šifrách variabilný a volí sa ako kompromis medzi rýchlosťou a stupňom bezpečnosti šifry.

(1) AES však **nevyužíva** Feistelovu štruktúru.

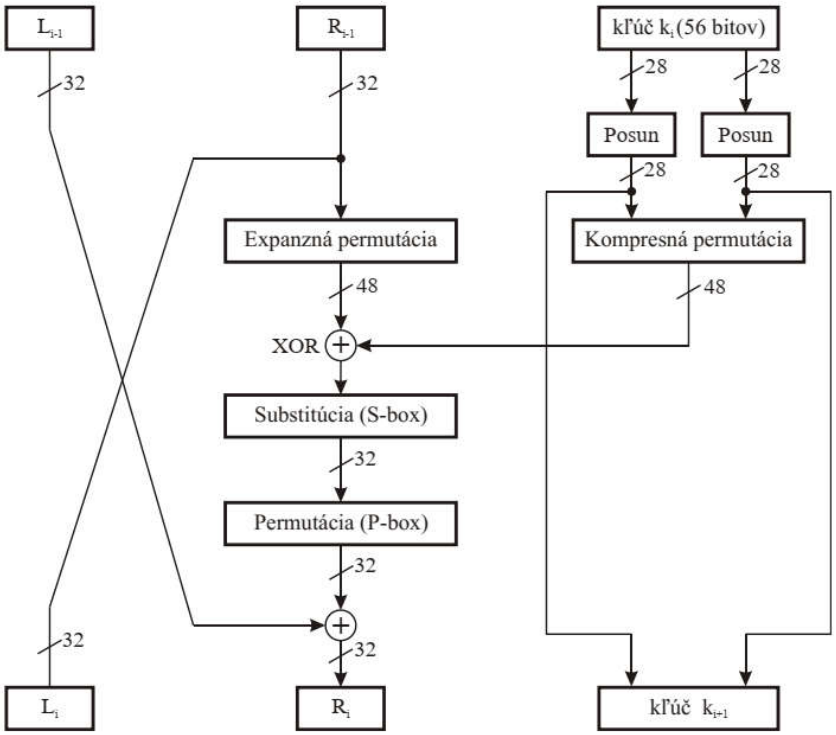
Šifrovací štandard DES

Šifrovací štandard **DES** (**Data Encryption Standard**) sa vyvinul z algoritmu, na ktorom pracovala firma IBM a ktorý dostal označenie LUCIFER. Uvedený algoritmus bol navrhnutý v rámci projektu, ktorý riešila výskumná skupina pod vedením Horsta Feistela. **LUCIFER** je Feistelova bloková šifra, ktorá pracuje so **64-bitovými blokmi dát** a **kľúč má dĺžku 128 bitov**. Šifrovací štandard DES, ktorý sa vyvinul z tohto algoritmu, používa 64-bitové bloky dát a **kľúč má dĺžku 56 bitov**.

Uvedený štandard bol prijatý už **v roku 1977** americkými inštitúciami pre bezpečnosť NBS (National Bureau of Standards), neskoršie **NIST** (National Institute of Standard and Technology). Šifrovací algoritmus mal označenie **DEA** (**Data Encryption Algorithm**).

Pôvodne bol šifrovací štandard DES určený pre aplikácie vo finančníctve. V roku 1999 bola prijatá inštitúciou NIST nová verzia štandardu DES, ktorá sa označuje **TDEA**. Využíva trojnásobné šifrovanie pomocou algoritmu DES a používa dva, resp. **tri rôzne kľúče**.

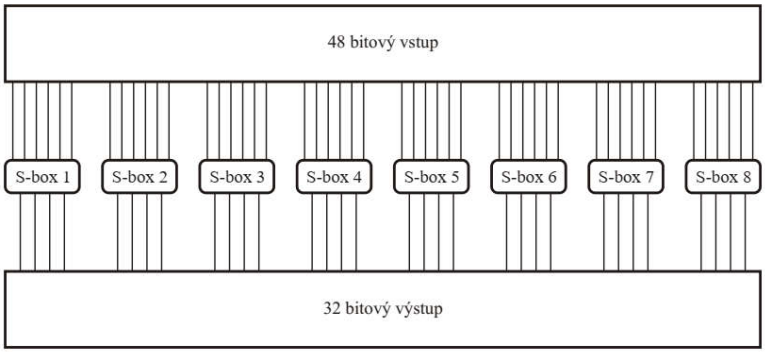
Runda DES



Obr. 5.3 Štruktúra rundy DES

Substitúcie a permutácie definované v štandarde **formou tabuliek**.

DES používa **8 rôznych** S-boxov.



Obr. 5.5 Substitúcia pomocou S–boxov

Bezpečnosť algoritmu DES
Bezpečnosť algoritmu DES bola dlho predmetom úvah, ktoré sa sústreďovali najmä na dva základné aspekty. Sú to:

• dĺžka kľúča
• vlastnosti algoritmu.

Dĺžka kľúča vyvolávala určité pochybnosti najmä preto, lebo pôvodný návrh počítal s dĺžkou kľúča 112 bitov. Jeho skrátenie na **56 bitov NSA prehlásila za vyhovujúce**. Aj keď táto dĺžka zabezpečuje možnosť voľby 2^56 rôznych kľúčov, v tejto množine existujú tzv. slabé, poloslabé a potenciálne slabé kľúče.

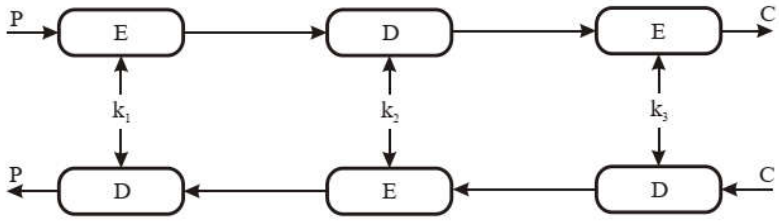
Iné úvahy o bezpečnosti DES sa sústreďovali na analýzu S-boxov a počet rúnd. Metodika návrhu S–boxov nebola nikdy zverejnená a to vyvolávalo rôzne podozrenia, že boli zámerne upravené tak, aby to umožnilo ľahšiu kryptoanalýzu pre autorov ich návrhu (NSA), resp., že úprava S-boxov umožnila vytvoriť tajný vstup do algoritmu. Uvedené úvahy sa však nepotvrdili(1). Napriek tomu, po rôznych skúsenostiach o možnosti prelomenia algoritmu DES a možnosti skonštruovať stroje na prelomenie algoritmu DES, bol tento algoritmus **v júli 1998 vyhlásený za nevyhovujúci** z hľadiska bezpečnosti.

Od zavedenia algoritmu DES sa vyvinuli viaceré modifikácie DES, z ktorých najvýznamnejšou je algoritmus **trojnásobný DES** (3DES, **TDEA**).

(1) Neskôr sa ukázalo, že návh S-boxov algoritmu DES bol realizovaný z ohľadom na odolnosť voči **diferenciálnej kryptoanalýze**, ktorá bola vo vedeckej komunite známa až od 90-tych rokov.

Trojnásobný DES s troma kľúčmi (**3TDEA**) využíva postupnosť EDE Ecrypt-Decrypt-Encrypt (Obr. 5.25) a proces šifrovania možno vyjadriť v tvare

$$C = E_{k_3}\left(D_{k_2}\left(E_{k_1}(P)\right)\right)$$



Obr. 5.25 Trojnásobný DES s troma kľúčmi

Kompatibilitu s algoritmom DES možno dosiahnuť, ak k3=k2=k1. **Dĺžka kľúča** 3TDEA je **168 bitov** (tri 56-bitové DES kľúče). V súčasnosti sú známe útoky na tento algoritmus, ktoré sa označujú ako **útoky zo stredu** (**meet-in-the-middle attacks**). Uvedené útoky výrazne znižujú kryptografickú bezpečnosť 3TDEA, ktorý má **efektívnu dĺžka kľúča len 112 bitov**. Algoritmus TDEA využívajú viaceré aplikácie v sieti internet, napr. PGP a S/MIME.

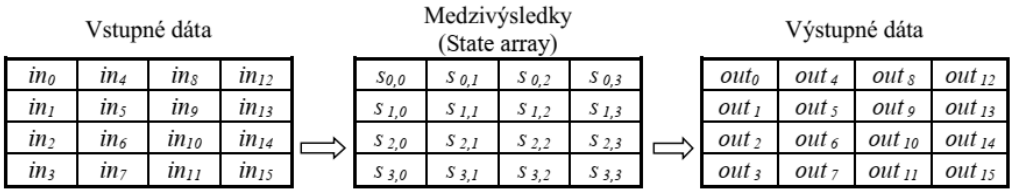
Algoritmus AES (Advanced Encryption Standard)
Vzhľadom na to, že dlhodobé používanie šifrovacieho algoritmu DES ukázalo možnosti jeho prelomenia, vyhlásil **NIST v roku 1997** verejnú súťaž na výber nového symetrického šifrovacieho algoritmu na šifrovanie dát, ktorý dostal označenie **AES (Advanced Encryption Standard)**. Cieľom bolo vybrať algoritmus, ktorého bezpečnosť by bola väčšia ako bezpečnosť 3DES, ktorý by umožňoval šifrovať **bloky o dĺžke 128 bitov** a pracovať so **128, 192 a 256-bitovými kľúčmi**.

V prvom kole sa vybralo **15 šifrovacích algoritmov**, pričom do užšieho výberu sa dostalo **5 algoritmov** (MARS, RC6, Rijndael, Serpent, Twofish). Výsledné hodnotenie NIST bolo publikované v podobe **štandardu (FIPS PUB 197)** v **novembri 2001**, pričom základom AES sa stal **algoritmus Rijndael**, ktorý do súťaže prihlásili Belgičania Joan Daemen a Vincent Rijmen.

Štruktúra dát v algoritme AES

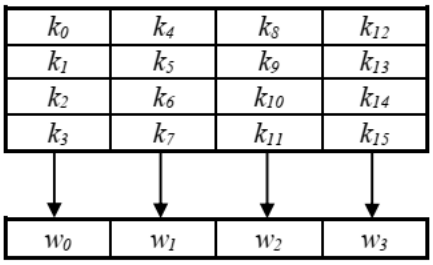
Na rozdiel od algoritmu DES (ktorý využíval **tabuľkovú reprezentáciu** použitých **transformácií** bez jasnej matematickej interpretácie) využíva algoritmus AES využíva na opis využitých **transformácií operácie v konečnom poli (GF - Galois Field)**.

AES pracuje výlučne s **veľkosťou bloku 128 bitov**, pričom **dĺžka kľúča** môže byť **128, 192 alebo 256 bitov**. V štandarde FIPS PUB 197 má vstupný 128-bitový blok na šifrovanie, resp. dešifrovanie organizáciu **v tvare štvorcovej matice bajtov**. Pre blok 128 bitov je to **16 bajtov**, organizovaných v **matici 4 x 4**. Tento blok dát sa v procese šifrovania, resp. dešifrovania transformuje na **dvojrozmerný blok medzivýsledkov** (**state array**), ktorý sa priebežne modifikuje v každom kroku šifrovania, resp. dešifrovania. Po ukončení jedného kroku sa blok medzivýsledkov prepíše do dvojrozmerného bloku výstupných dát, ktorý má tiež formu matice bajtov s rozmerom 4 x 4. **Štyri bajty každého stĺpca** v bloku State array vytvárajú **32-bitové slovo**.



Obr. 5.6 Štruktúra operácií a dát v algoritme AES

Kľúč, počet rúnd a rundové kľúče
Kľúč v algoritme AES možno vyjadriť vo forme matice bajtov so 4 riadkami, pričom počet stĺpcov Nk závisí od dĺžky kľúča. Pre kľúče s dĺžkou 128 bitov je Nk=4, pre kľúč s dĺžkou 192 bitov je Nk=6 a pre kľúč s dĺžkou 256 bitov je Nk=8. Jednotlivé stĺpce matice vytvárajú slová wi s dĺžkou 32 bitov. Príklad uvedenej štruktúry je pre dĺžku kľúča 128 bitov uvedený na Obr. 5.7.



Obr. 5.7 Štruktúra kľúča pre dĺžku 128 bitov

Prvé štyri bajty kľúča k0,k1,k2,k3 vytvárajú **slovo** w0, ďalšie bajty k4,k5,k6,k7 vytvárajú slovo w1 atď. Z uvedenej štruktúry sa generujú **operáciou expanzie kľúča** ďalšie **bajty rundových kľúčov**.

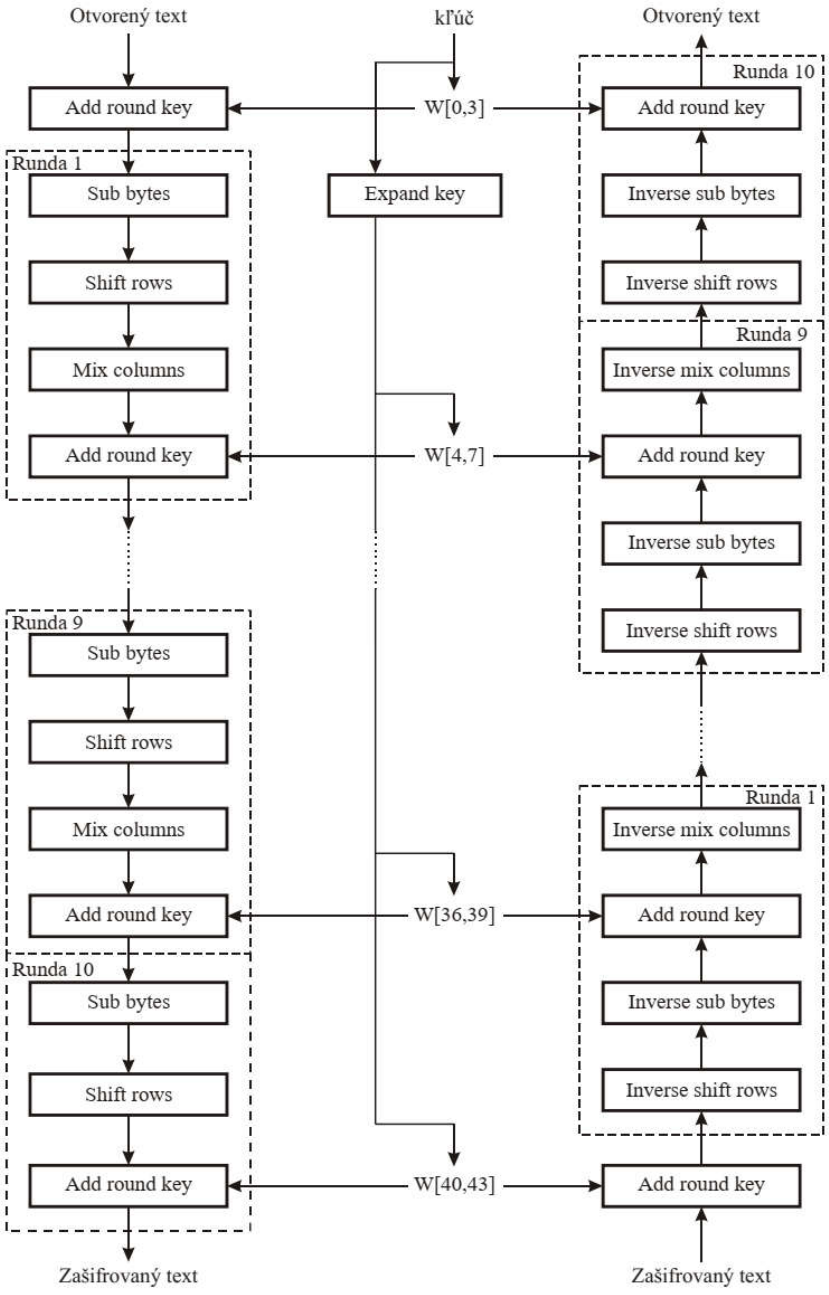
Algoritmus AES je **iteračný algoritmus** a každá iterácia sa označuje ako **runda**, podobne ako v algoritme DES. **Počet rúnd Nr** závisí od zvolenej **dĺžky kľúča**, čo vyjadruje Tab. 5.10.

Tab. 5.10 Závislosť počtu rúnd od dĺžky kľúča

N_k	N_r
4	10
6	12
8	14

Ak je zvolená dĺžka kľúča 128 bitov (**AES128**), t. j. Nk=4, je počet rúnd Nr rovný 10. Ďalej bude opisovaný algoritmus AES128.

Štruktúra algoritmu AES



Obr. 5.8 Štruktúra algoritmu šifrovania a dešifrovania v AES

Štruktúra algoritmu AES je pomerne jednoduchá. Proces šifrovania a dešifrovania začína operáciou, ktorá sa označuje ako **Add round key** a súčasne sa realizuje operácia **Expand key**, ktorá realizuje rozšírenie kľúča. Po týchto operáciách nasleduje **deväť rúnd**, pričom každá obsahuje **štyri operácie** (1) a to:

• **Substitute bytes** – využíva S-box na substitúciu bajtov v bloku State array
• **Shift rows** – permutácia riadkov cyklickým posunom
• **Mix columns** – substitúcia s využitím aritmetiky nad GF(2^8)
• **Add round key** – operácia XOR bitov aktuálneho bloku s časťou rozšíreného kľúča.

Desiata runda pri šifrovaní a dešifrovaní obsahuje iba tri operácie, pretože je v nej **vynechaná operácia Mix columns**. Je potrebné tiež poznamenať, že zo štyroch uvedených operácií **iba operácia Add round key využíva kľúč**, preto šifrovanie a dešifrovanie začína touto operáciou. Všetky **ostatné operácie sú reverzibilné** bez znalosti kľúča a **realizujú konfúziu, difúziu**, ale samotné nezaručujú bezpečnosť. Operácia **Add round key** je vlastne aplikáciou Vernamovej šifry s využitím **operácie XOR** medzi bitmi aktuálneho bloku a časti kľúča. Táto operácia **je inverzná**, ak sa použije **ten istý kľúč**, čo znamená, že **rozšírený kľúč sa musí použiť v obrátenom poradí**.

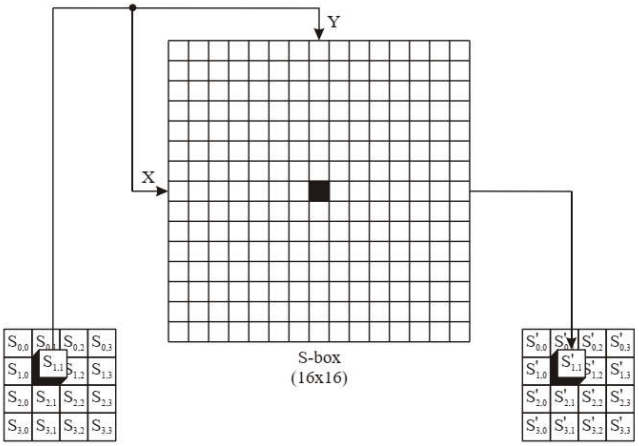
Posledná runda v šifrovaní, resp. dešifrovaní pozostáva len z troch operácií, čo je dôsledok štruktúry AES a zabezpečuje reverzibilnosť šifry.

Je potrebné tiež poznamenať, že algoritmus AES používa **aritmetiku v konečnom poli** (2) GF (2^8) s ireducibilným polynómom m(x)=x^8+x^4+x^3+x+1.

(1) Na cvičení bude realizovaná **vizualizácia** týchto operácií v nástroji CrypTool
(2) **Aritmetika v GF(256)** bude podrobnejšie precvičená v nasledujúcom cvičení

Operácia Substitute bytes

Priama substitúcia bajtov využíva S-box, ktorý má **tvar matice 16 x 16**, čo umožňuje transformovať všetky možné hodnoty 8-bitových hodnôt (bajtov). Princíp priamej substitúcie Substitute bytes je uvedený na Obr. 5.9.

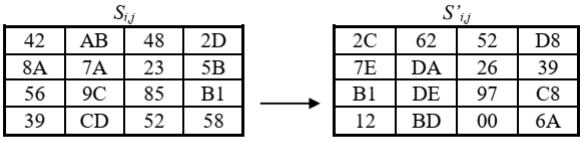


Obr. 5.9 Priama substitúcia bajtov

Tab. 5.11 Tabuľky S–boxov v AES, a) Priama substitúcia, b) Inverzná substitúcia

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

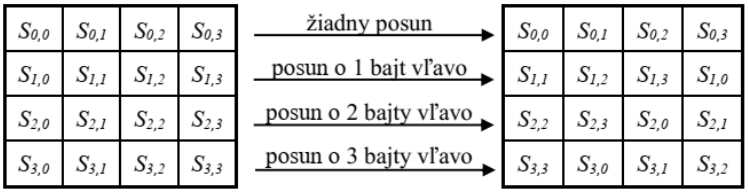
a) Priama substitúcia



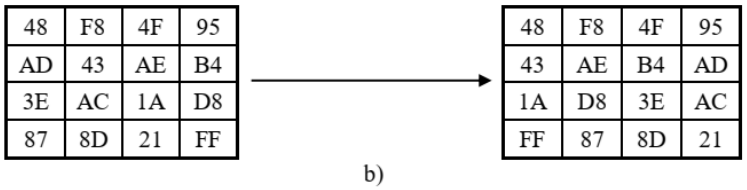
Obr. 5.10 Použitie priamej substitúcie

Operácia Shift Row Transformation

Priama operácia Shift Row Transformation, ktorá sa skrátene označuje Shift Rows je znázornená na Obr. 5.11 a aplikuje sa na **riadky podľa bajtov bloku State array**. Prvý riadok tohto bloku ostáva bez zmeny. Druhý riadok bloku bajtov State array sa posunie o jeden bajt vľavo, tretí riadok o dva bajty vľavo a tretí riadok sa posunie o tri bajty vľavo.



a)



b)

Obr. 5.11 Operácia Shift Rows

Operácia Mix Column Transformation
Priama operácia Mix Column Transformation, ktorá sa označuje Mix Columns **transformuje individuálne každý stĺpec** bloku State array. Každý bajt stĺpca sa transformuje na novú hodnotu, ktorá je funkciou všetkých štyroch bajtov stĺpca.

Uvedenú transformáciu možno vyjadriť ako súčin matíc v tvare

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \cdot \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} \tag{5.5}$$

Operácie v (5.5) sú realizované **v poli GF(2^8)** modulo m(x)=x^8+x^4+x^3+x+1 a príklad transformácie na nasledujúcom obrázku bude overený na cvičení.

F2	4D	97	87
4C	90	EC	6E
E7	4A	C3	46
8C	D8	95	A6

→

40	A3	4C	47
D4	70	9F	37
E4	3A	42	94
A5	A6	BC	ED

Obr. 5.12 Príklad operácie Mix Columns

$$(\{02\}\cdot\{F2\})\oplus(\{03\}\cdot\{4C\})\oplus\{E7\}\oplus\{8C\}=\{40\}$$
$$\{F2\}\oplus(\{02\}\cdot\{4C\})\oplus(\{03\}\cdot\{E7\})\oplus\{8C\}=\{D4\}$$
$$\{F2\}\oplus\{4C\}\oplus(\{02\}\cdot\{E7\})\oplus(\{03\}\cdot\{8C\})=\{E4\}$$
$$(\{03\}\cdot\{F2\})\oplus\{4C\}\oplus\{E7\}\oplus(\{02\}\cdot\{8C\})=\{A5\}$$

Operácia Add Round Key

Priama operácia Add Round Key realizuje **súčet XOR 128 bitov** bloku **State array** so **128 bitmi kľúča príslušnej rundy**, ktorá sa vykonáva po stĺpcoch. Príklad tejto operácie je uvedený na Obr. 5.13.

State			
32	AB	D4	31
80	4E	B8	85
A8	5B	85	63
61	43	72	BD

⊕

Round key			
61	8B	AC	9B
6A	93	D1	5C
DC	00	3A	E4
32	32	63	BC

=

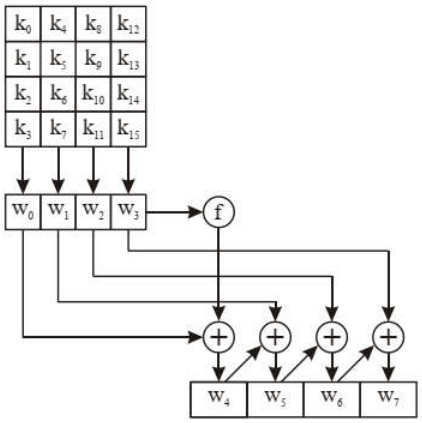
53	20	78	AA
EA	DD	69	D9
74	5B	BF	87
53	71	11	01

Obr. 5.13 Príklad operácie Add Round Key

Inverzná operácia InvAdd Round Key je identická s priamou operáciou Add Round Key, pretože operácia **XOR je inverzná**. Obe operácie sú veľmi jednoduché a **ovplyvňujú všetky bity** bloku State array.

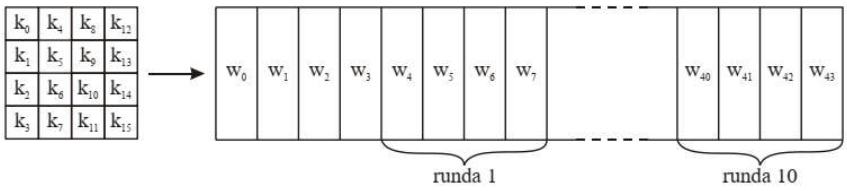
Operácia Key Expansion

Táto operácia realizuje **algoritmus expanzie kľúča**, ktorý má dĺžku 16 bajtov a vyjadruje sa formou štyroch slov s dĺžkou štyri bajty. Pri počte rúnd 10 je potrebná expanzia na 44 slov. Proces expanzie kľúča pre prvý rundový kľúč je znázornený na Obr. 5.14 (transformácia f, ktorá je závislá na čísle rundy, je podrobne opísaná napr. v [1]).



Obr. 5.14 Expanzia kľúča pre prvý rundový kľúč

Proces expanzie kľúča pre počet rúnd 10 je znázornený na Obr. 5.15.



Obr. 5.15 Expanzia kľúča pre 10 rúnd

Ďalšie symetrické šifry

Existuje viacero symetrických šifier, ktoré sa v praxi používajú resp. používali. Sú to napr.
blokové šifry:
RC5, Skipjack, Blowfish, Twofish, SIMON, Fantomas, Present, ...

prúdové šifry:
RC4, Trivium, Salsa20, ChaCha20, ...

a mnoho ďalších.

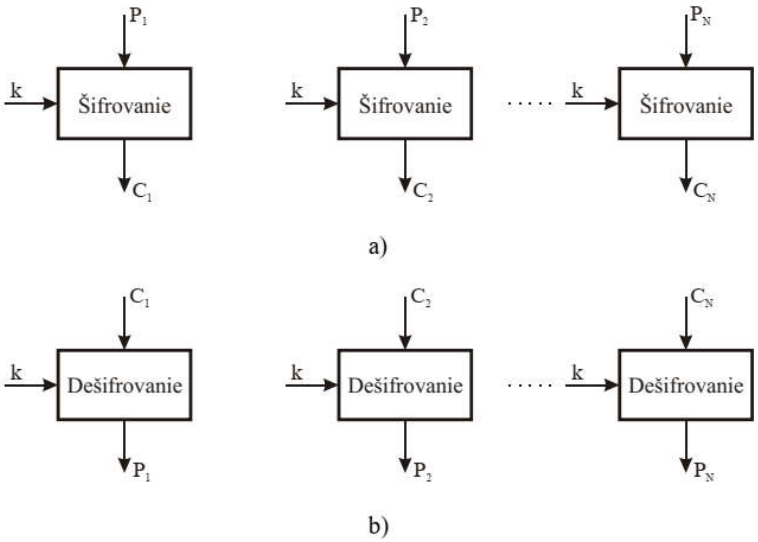
V súčasnosti má dominantné postavenie AES (najčastejšie AES128), čo sa prejavuje napr. aj integráciou **kryptografických koprocesorov na báze AES** do špecializovaných komunikačných procesorov alebo procesorov (mikrokontrolérov) napr. pre IoT aplikácie a tiež zahrnutím **špecializovaných inštrukcií** do procesorov Intel a AMD.

Alternatívne symetrické šifry sú zaujímavé (a aktívne vyvíjané) aj pre aplikácie tzv. ľahkej kryptografie (**lightweight cryptography**), ktorá využíva napr. mikroprocesory bez kryptografických koprocesorov alebo zákaznícke riešenia (na báze ASIC obvodov) s cieľom **znížiť** napr. **cenu**, **energetickú náročnosť** a pod.

Režimy blokových šifier

Elektronická kódová kniha (ECB – Electronic Codebook)

Najjednoduchší režim blokových šifier je režim, ktorý sa označuje ako **elektronická kódová kniha,** resp. **režim E*CB*** (Electronic Codebook - ECB). V tomto režime sa otvorený text rozdelí na bloky a každý blok otvoreného textu sa priamo transformuje na blok zašifrovaného textu. Jednotlivé bloky sa šifrujú postupne a nezávisle tým istým kľúčom. Analogicky prebieha aj proces dešifrovania.

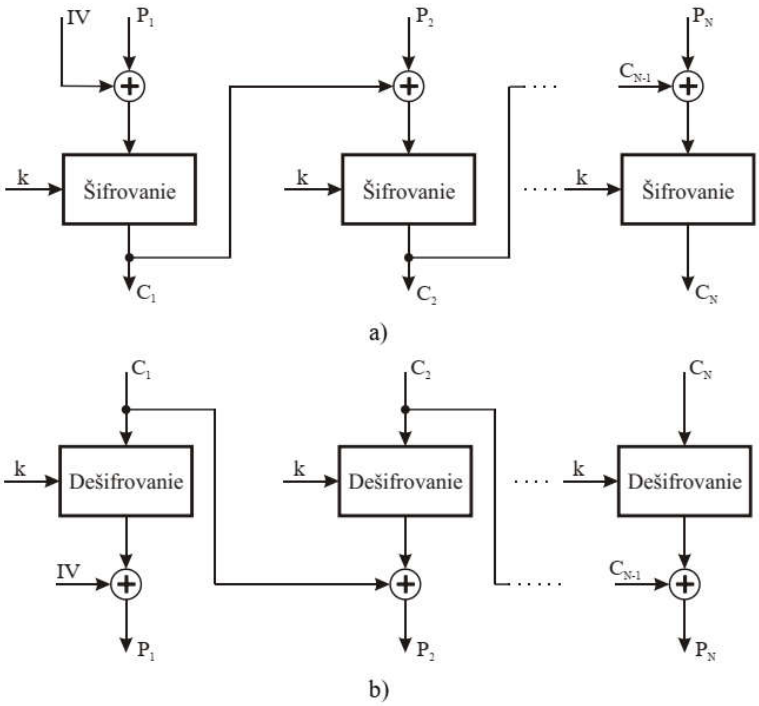


Obr. 5.16 Režim ECB a – šifrovanie, b – dešifrovanie

Významnou vlastnosťou režimu ECB je to, že ten istý blok otvoreného textu sa transformuje šifrovaním vždy na rovnaký blok zašifrovaného textu, čo pri dlhých správach môže uľahčiť kryptoanalýzu.

ECB využíva na dešifrovanie blok **Dešifrovanie**.

Zreťazenie zašifrovaného textu (CBC – Cipher Block Chaining)
Zvýšenie kryptografickej bezpečnosti režimu ECB možno dosiahnuť technikou, ktorá umožňuje šifrovaním toho istého bloku otvoreného textu získať rôzne bloky zašifrovaného textu, ak sa daný blok otvoreného textu opakuje.
Uvedenú požiadavku spĺňuje režim zreťazenia zašifrovaného textu (CBC), ktorého štruktúra je uvedená na *Obr. 5.17*. CBC využíva spätnú väzbu, čím sa dosiahne, že výsledok šifrovania aktuálneho bloku otvoreného textu závisí aj od bloku zašifrovaného textu v predošlom kroku.



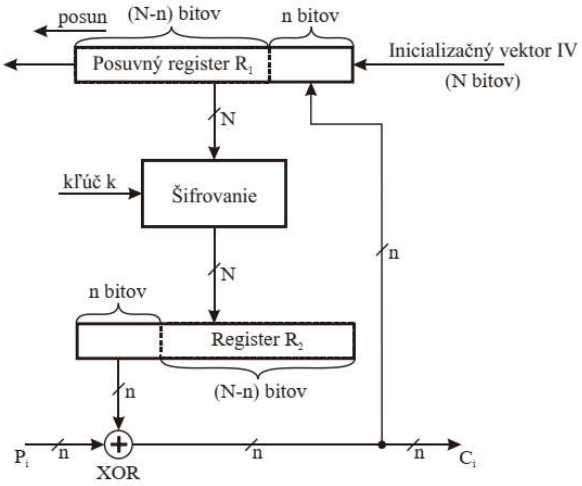
Obr. 5.17 Režim CBC a) šifrovanie b) dešifrovanie

Pri šifrovaní prvého bloku otvoreného textu P_1 ešte nie je vytvorený blok zašifrovaného textu (C_0), preto sa definuje tzv. **inicializačný vektor** IV=C_0. Úlohou IV je „**znáhodniť**“ výstup šifrovania tak, aby aj **rovnaké bloky dát** boli **šifrované odlišne**. IV **nemusí** byť **utajovaný**, **nesmie** sa však používať **opakovane**.

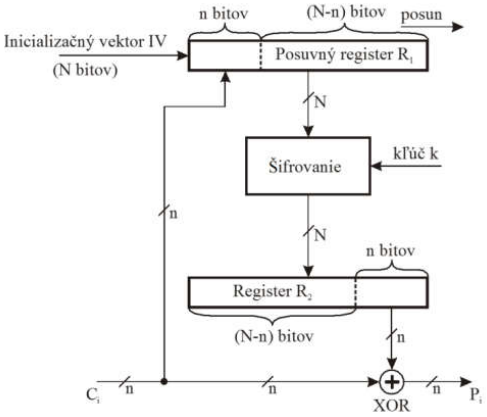
CBC využíva na dešifrovanie blok **Dešifrovanie**.

Režim CBC možno charakterizovať ako režim so **spätnou väzbou** zašifrovaného textu pri šifrovaní a **doprednou väzbou** zašifrovaného textu pri dešifrovaní, čo má vplyv na šírenie chýb.

Spätná väzba zo zašifrovaného textu (CFB – Cipher Feedback)
Režim CFB je určený na šifrovanie dát v **subblokoch, ktoré sú menšie než pôvodný blok** dát. Napr. blok dát o dĺžke 64 bitov sa šifruje po 8 bitoch (8-bitový režim CFB), resp. 8-bitový blok sa šifruje po jednom bite (1-bitový režim CFB). Šifrovanie po jednom bite umožňuje **realizovať konverzie** blokovej šifry (napr. DES, AES) na prúdovú šifru.



Obr. 5.18 Režim CFB (šifrovanie)



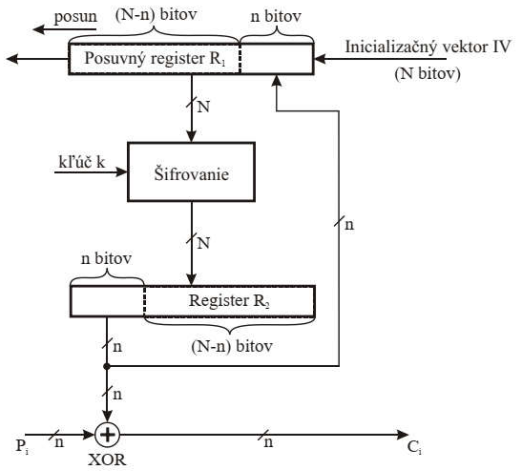
Obr. 5.19 Režim CFB (dešifrovanie)

CFB využíva na dešifrovanie blok **Šifrovanie!**

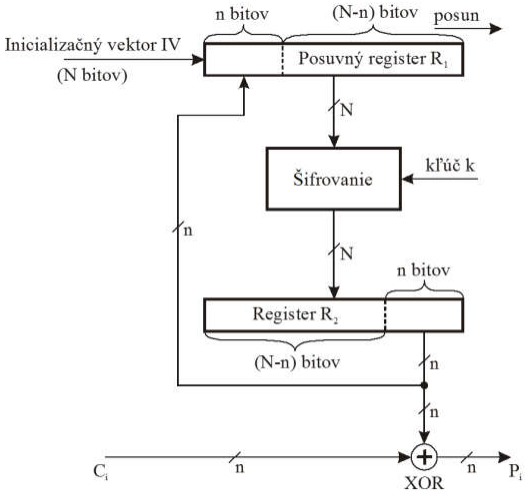
Režim CFB tiež využíva **spätnú väzbou**, čo má vplyv na šírenie chýb. Šírenie je závislé na hodnotách parametrov *N* a *n*.

Spätná väzba z výstupu (OFB – Output Feedback)

Režim OFB je veľmi podobný režimu CFB. Rozdiel spočíva v tom, že do *n* nižších bitov posuvného registra R_1 sa subblok *n* bitov, ktorý sa vyberie z *n*–vyšších bitov registra R_2, zapisuje ešte pred realizáciou operácie XOR. Spätná väzba **nie je závislá na vstupných** dátach a môže byť **vypočítaná a uložená vopred**.



Obr. 5.20 Režim OFB (šifrovanie)

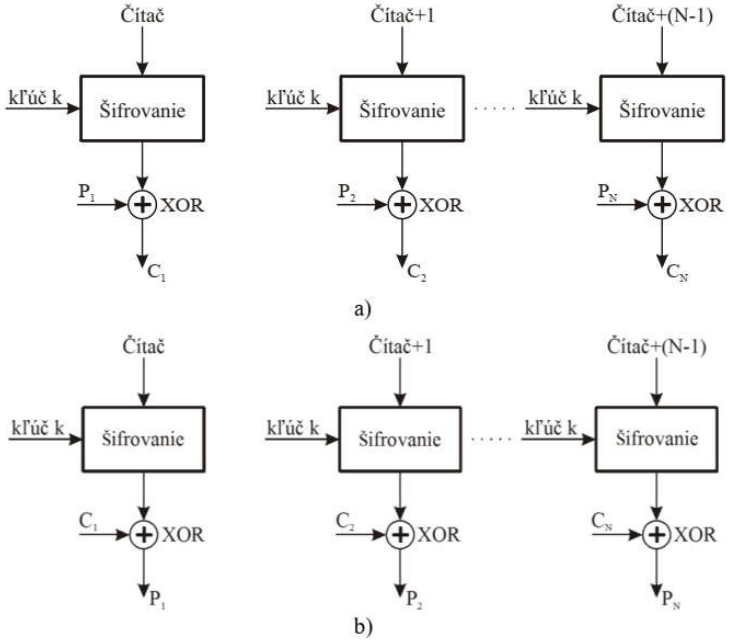


Obr. 5.21 Režim OFB (dešifrovanie)

OFB využíva na dešifrovanie blok **Šifrovanie!**

Výhodou režimu OFB v porovnaní s režimom CFB je to, že v tomto režime jednoduchá chyba v C_i ovplyvní len jeden bit v dešifrovanom bloku P_i a teda **nedochádza k šíreniu chyby** do ďalších dešifrovaných blokov P_i.

Čítačový režim (CTR - Counter Mode)
Režim CTR je založený na použití čítača, ktorý má rovnaký počet bitov ako je veľkosť bloku otvoreného textu. Čítač sa nastaví na určitú hodnotu, ktorá sa potom inkrementuje pre každý šifrovaný blok otvoreného textu. Výstup čítača sa podrobí operácii XOR s blokom otvoreného textu, čím sa získa blok zašifrovaného textu. Na dešifrovanie sa používa rovnaký čítač s rovnakou predvoľbou. Jednotlivé bloky otvoreného textu sa získajú operáciou XOR výstupu čítača a príslušného bloku zašifrovaného textu.



Obr. 5.22 Čítačový režim a) šifrovanie, b) dešifrovanie

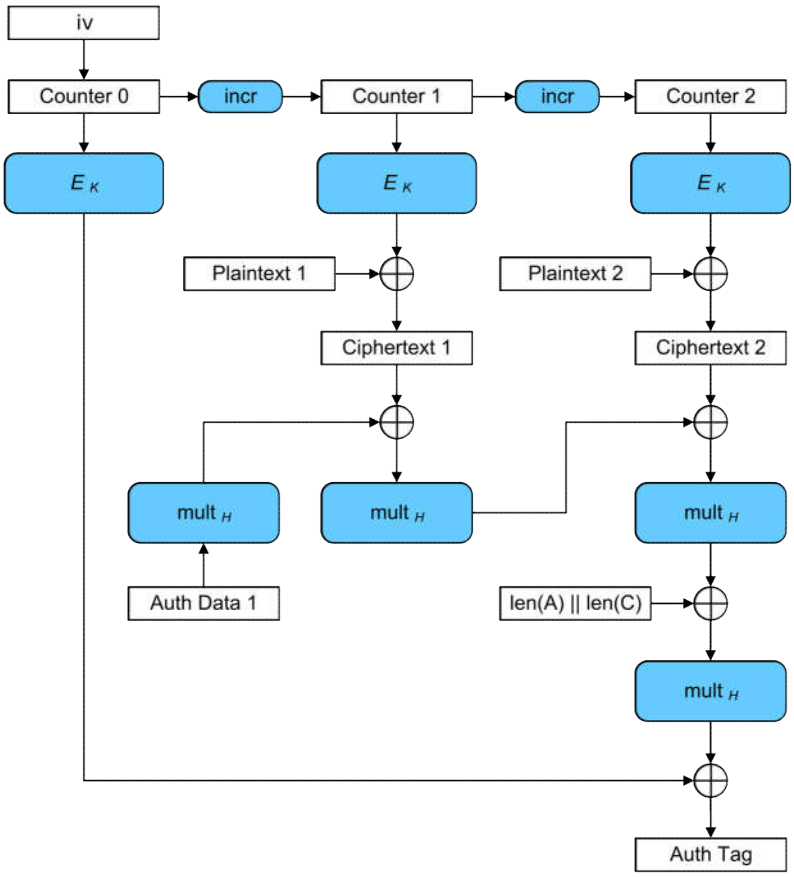
CTR využíva na dešifrovanie blok **Šifrovanie!**

Výhody režimu CTR spočívajú najmä v **jednoduchej hardvérovej a softvérovej implementácii**, pričom bezpečnosť tohto režimu je na úrovni predošlých režimov blokových šifier.

CTR umožňuje **veľmi jednoduché** spustenie dešifrovania **od zvolenej pozície** (napr. uloženého súboru).

Ďalšie módy
V praxi sa používajú aj ďalšie módy blokových šifier. Najznámejším je
Galoisov/ čítačový režim (GCM - Galois/Counter Mode)
ktorý realizuje okrem šifrovania dát (správy) aj výpočet **autentizačného kódu správy** (Message Authentication Code).

Základný princíp GCM je zobrazený na nasledujúcom obrázku (mult $_H$ reprezentuje operáciu násobenia v príslušnom GF (Galois Field)- konečnom poli, pre **AES128** to je **GF(2^128)**).



(zdroj: https://en.wikipedia.org/wiki/Galois/Counter_Mode)

Tejto problematike bude podrobnejšie venovaná prednáška v 7. týždni a tiež v predmete BIKS v inžinierskom štúdiu.