Symetrické šifry

Šifrovací štandard AES

Katedra matematiky a teoretickej informatiky, FEI TU

Symetrické šifry - opakovanie

- Čo viete o **symetrických šifrách**?
- Uveďte hlavné znaky **Feistelovej blokovej šifry**.
- Stručne opíšte šifrovací štandard **DES**.

Šifrovací štandard AES

- dlhodobé používanie šifrovacieho algoritmu DES ukázalo možnosti jeho prelomenia $\Rightarrow$ v roku **1997** NIST vyhlásil verejnú súťaž na výber nového symetrického šifrovacieho algoritmu, ktorý dostal označenie **AES (Advanced Encryption Standard)**
- **cieľ** - vybrať algoritmus, ktorého bezpečnosť by bola väčšia ako bezpečnosť 3DES, a ktorý by umožňoval **šifrovať bloky o dĺžke 128 bitov** a pracovať so **128, 192 a 256-bitovými kľúčmi**
- v novembri **2001** bol publikovaný nový štandard - základom AES sa stal algoritmus **Rijndael** , ktorý do súťaže prihlásili Belgičania **Joan Daemen** a **Vincent Rijmen**

Šifrovací štandard AES

Tri kategórie požiadaviek na nový štandard AES formulované NIST:

1. **bezpečnosť** - množstvo úsilia, resp. práce na prelomenie algoritmu kryptoanalýzou; minimálna dĺžka kľúča pre AES bola zvolená 128 bitov $\Rightarrow$ metóda totálnych skúšok s využitím súčasných, resp. predpokladaných technológií nebola uvažovaná
2. **náklady (cena)** - vychádzali z predpokladu, že tento algoritmus má pokrývať širokú oblasť aplikácii a má mať vysokú výpočtovú efektívnosť tak, aby bol použiteľný aj pre aplikácie s vysokými prenosovými rýchlosťami, napr. v širokopásmových sieťach
3. **implementácia** - zahrňuje širokú škálu parametrov ako sú napr. flexibilita, jednoduchosť a možnosť použitia v rôznych hardvérových a softvérových prostriedkoch

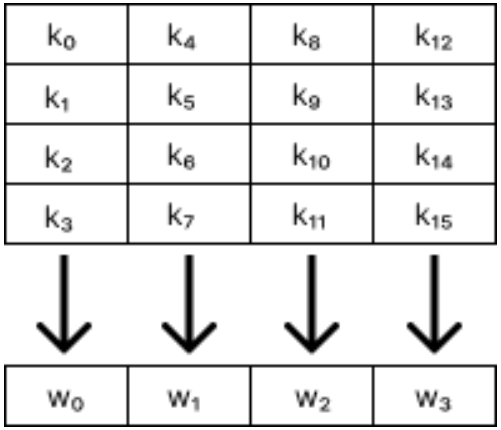
Opis štandardu AES

- algoritmus AES je založený na **symetrickej blokovej šifre**, ktorá používa rovnaký kľúč na šifrovanie a dešifrovanie
- hoci algoritmus Rijndael umožňoval režimy šifrovania 128, 192 a 256-bitových blokov dát, algoritmus AES pracuje výlučne s **veľkosťou bloku 128 bitov**, pričom **dĺžka kľúča** môže byť **128, 192 alebo 256 bitov**
- **vstupný 128-bitový blok** na šifrovanie, resp. dešifrovanie má organizáciu v tvare štvorcovej matice bajtov (16 bajtov v matici 4x4) - tento blok dát sa v procese šifrovania, resp. dešifrovania transformuje na dvojrozmerný **blok medzivýsledkov**, ktorý sa priebežne modifikuje v každom kroku šifrovania, resp. dešifrovania

Štruktúra operácií a dát v algoritme AES

1. **vstupné dáta** - organizované v matici vstupných bajtov s rozmerom 4x4, pričom jednotlivé bajty vstupných dát sú označované symbolmi in_0, in_1, ..., in_{15}
2. **blok medzivýsledkov** - pozostáva zo 16 bajtov označených symbolmi $S_{i,j}$ pre $0 \leq i, j \leq 3 \Rightarrow$ 4 bajty každého stĺpca v bloku State array vytvárajú 32-bitové slovo
3. **výstupné dáta** - tiež organizované v matici výstupných bajtov s rozmerom 4x4, pričom bajty výstupných dát majú označenie out_0, out_1, ..., out_{15}

Kľúč v algoritme AES



- kľúč možno vyjadriť vo forme matice bajtov so **4 riadkami**, pričom **počet stĺpcov** N_k, závisí od dĺžky kľúča
- pre kľúče s dĺžkou 128 bitov je $\mathbf{N_k = 4}$, pre kľúč s dĺžkou 192 bitov je $\mathbf{N_k = 6}$ a pre kľúč s dĺžkou 256 bitov je $\mathbf{N_k = 8}$
- jednotlivé stĺpce matice vytvárajú **slová** $\mathbf{w_i}$ s dĺžkou 32 bitov
- z uvedenej štruktúry sa generujú **operáciou expanzie kľúča** ďalšie bajty rundových kľúčov

Runda v algoritmu AES

- AES je **iteračný algoritmus** a každá iterácia sa označuje ako runda
- **počet rúnd N**, závisí od zvolenej dĺžky kľúča, čo vyjadruje tabuľka:

N_k	N_r
4	10
6	12
8	14

- ak je zvolená **dĺžka kľúča 128 bitov**, t. j $N_k = 4$, je počet rúnd N_r rovný 10, ak je zvolená dĺžka kľúča **192 bitov**, t. j. $N_k = 6$, je $N_r = 12$ a pre kľúč s dĺžkou **256 bitov** $N_k = 8$ je počet rúnd $N_r = 14$

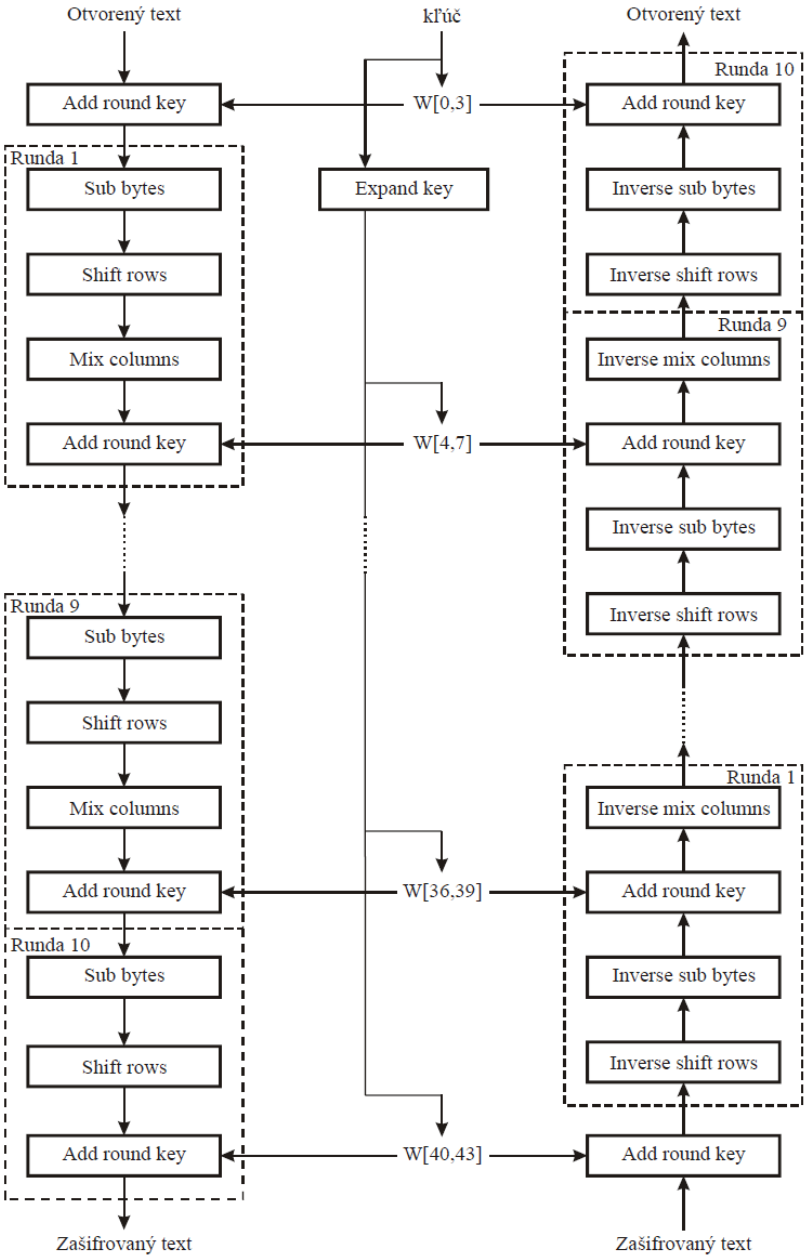
Štruktúra šifrovania a dešifrovania v AES

- uvedená štruktúra **nemá charakter Feistelovej šifry**, ktorá používa polovicu bloku dát na modifikáciu inej polovice bloku dát a uvedené polovice sa potom vymenili
- algoritmus AES **spracováva celý blok dát paralelne v každej runde** s využitím substitúcií a permutácií
- 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 nasledujú rundy, pričom každá s výnimkou poslednej obsahuje **štyri operácie**

Štruktúra šifrovania a dešifrovania v AES

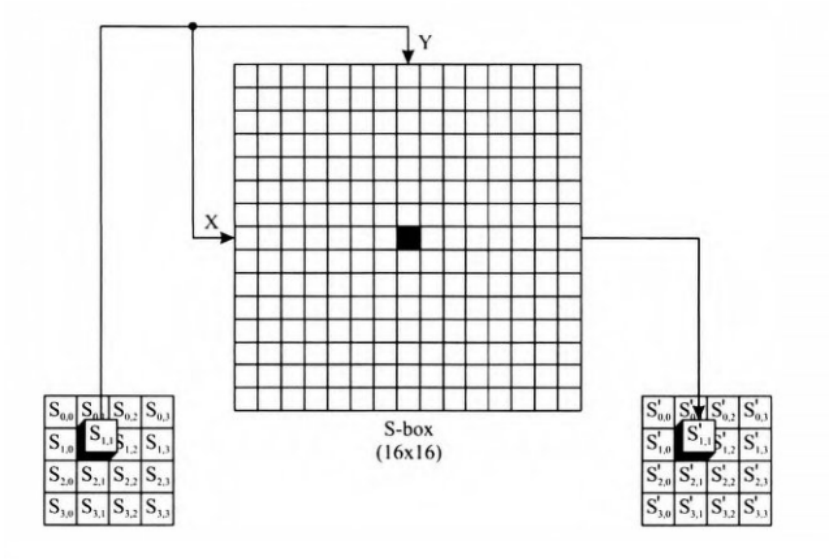
- operácie v rámci rundy:
  1. **Substitute bytes** - využíva S-box v bloku *State array*
  2. **Shift rows** - permutácia riadkov cyklickým posunom
  3. **Mix columns** - substitúcia s využitím aritmetiky nad GF(2^8)
  4. **Add round key** - operácia XOR bitov aktuálneho bloku s časťou rozšíreného kľúča
- iba operácia *Add round key* využíva kľúč $\Rightarrow$ **šifrovanie a dešifrovanie začína touto operáciou** - ide vlastne o aplikáciu Vernamovej šifry
- všetky ostatné operácie sú **reverzibilné bez znalosti kľúča** a realizujú konfúziu, difúziu, ale samotné nezaručujú bezpečnosť
- **posledná runda v šifrovaní, resp. dešifrovaní** - pozostáva len z 3 operácií (vynechaný *Mix columns*), čo je dôsledok štruktúry AES a zabezpečuje reverzibilnosť šifry
- algoritmus AES používa **aritmetiku v konečnom poli** GF(2^8) s ireducibilným polynómom $m(x) = x^8 + x^4 + x^3 + x + 1$

Štruktúra šifrovania a dešifrovania v AES



Operácia Substitute bytes

- transformácia, ktorá má **priamy aj inverzný tvar** a realizuje priamu, resp. inverznú substitúciu bajtov



- každý bajt bloku *State array* je transformovaný na novú hodnotu takto: 4 vyššie bity aktuálneho bajtu vytvárajú **adresu riadku** matice S-boxu a 4 nižšie bity vytvárajú **adresu stĺpca** matice $\Rightarrow$ nová hodnota bajtu leží v tabuľke S-boxu na tejto adrese

Operácia Substitute bytes

- tabuľka S–boxu v AES pre **priamu substitúciu**:

		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

- S–box, resp. inverzný IS–box boli navrhnuté tak, aby boli **odolné voči známym kryptografickým útokom**
- autori algoritmu AES navrhli S–box tak, že má **nízku koreláciu medzi vstupnými a výstupnými bitmi** a **výstup nemôže byť vyjadrený jednoduchou matematickou funkciou vstupu**

Priama operácia Substitute bytes

S–box pre priamu substitúciu je vyplnený tak, že pre každý bajt:

1. sa vypočíta **multiplikatívne inverzný prvok** v $GF(2^8)$ modulo $m(x)$, pričom hodnota $\{00\}$ sa transformuje sama na seba
2. ak označíme jednotlivé bity multiplikatívne inverzného prvku symbolmi $(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$ a bity výsledného prvku symbolmi $(b'_7, b'_6, b'_5, b'_4, b'_3, b'_2, b'_1, b'_0)$, potom priamu substitúciu možno vyjadriť v tvare

$$\mathbf{b'_i = b_i \oplus b_{(i+4) mod 8} \oplus b_{(i+5) mod 8} \oplus b_{(i+6) mod 8} \oplus b_{(i+7) mod 8} \oplus c_i}$$

kde c_i je i–tý bit bajtu $c = c_7c_6c_5c_4c_3c_2c_1c_0 = 01100011$ (konštanta c bola zvolená tak, že S–box nemá tzv. pevné body, pre ktoré platí $S - box(a) = a$)

Inverzná operácia Inverse substitute bytes

Inverzný S–box je konštruovaný:

1. aplikovaním inverznej transformácie na vstupný bajt $(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$:

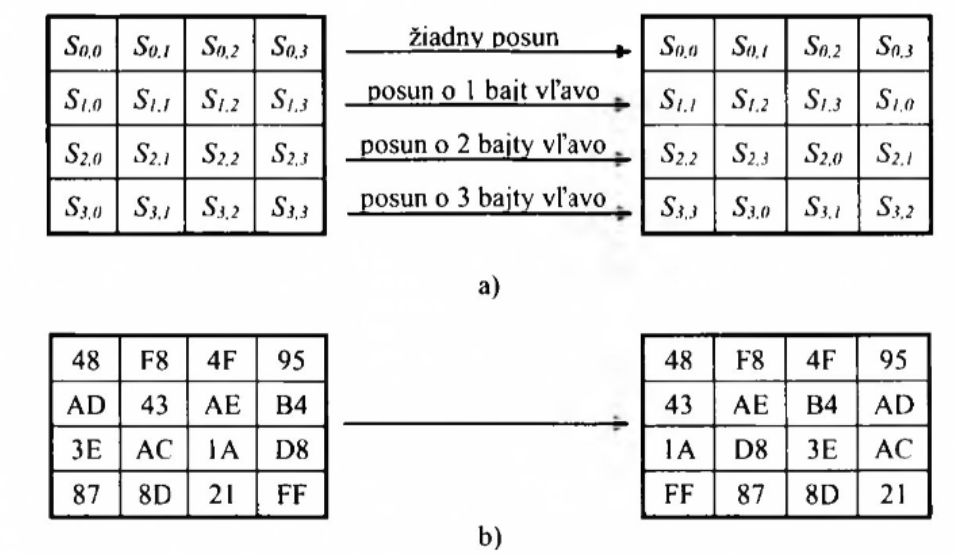
$$\mathbf{b'_i = b_i \oplus b_{(i+2)mod8} \oplus b_{(i+5)mod8} \oplus b_{(i+7)mod8} \oplus d_i}$$

   kde d je konštanta $d = \{05\} = (00000101)_2$ (zvolená tak, že IS–box nemá tzv. inverzné pevné body, pre ktoré platí $IS - box[S - box(a)] = \bar{a}$, kde $\bar{a}$ prvý doplnok k a)

2. následným výpočtom multiplikatívne inverzného prvku v $GF(2^8)$ modulo $m(x)$ k transformovanému bajtu $(b'_7, b'_6, b'_5, b'_4, b'_3, b'_2, b'_1, b'_0)$

Operácia Shift Row

- **priama operácia *Shift Row Transformation*** - aplikuje sa na riadky podľa bajtov bloku *State array* nasledovne:



- **inverzná transformácia *Shift Row Transformation*** (označuje sa *InvShift Rows*) - vykonáva cyklický posun riadkov bloku bajtov *State array* o rovnaký počet bajtov, ale v opačnom smere

Operácia Shift Row

- operácia *Shift Row Transformation* má oveľa väčší význam ako by sa zdalo
- je to preto, že blok *State array* možno z hľadiska vstupu a výstupu šifrovania chápať ako 4 stĺpce, z ktorých každý obsahuje 4 bajty
- ako bude ukázané neskoršie, aj kľúč rundy sa aplikuje po stĺpcoch
- operáciou *Shift Rows* sa vlastne premiestňujú bajty bloku *State array* z jedného stĺpca do druhého tak, že **štyri bajty jedného stĺpca sú distribuované do štyroch rôznych stĺpcov**

Priama operácia Mix Columns

- označuje sa *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**:

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

Inverzná operácia Mix Columns

- označuje sa *InvMix Columns* a je definovaná v maticovom tvare rovnicou:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \cdot \begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix} = \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}$$

- **dôkaz inverznosti** transformácie *InvMix Columns* možno vyjadriť rovnicou:

$$\begin{bmatrix} 0E & 0B & 0D & 09 \\ 09 & 0E & 0B & 0D \\ 0D & 09 & 0E & 0B \\ 0B & 0D & 09 & 0E \end{bmatrix} \cdot \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \cdot \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}$$

$$\begin{bmatrix} 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \\ 00 & 00 & 00 & 01 \end{bmatrix} \cdot \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix}$$

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
- **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*
- príklad operácie *Add Round Key*:

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

$\oplus$

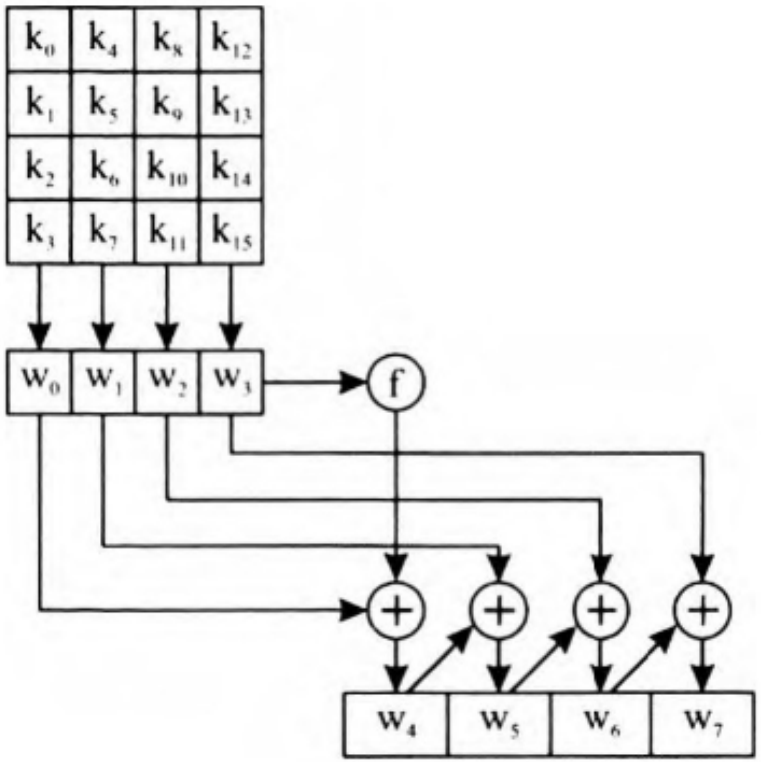
Round key			
61	8B	AC	9B
6A	93	D1	5C
DC	00	3A	E4
32	32	63	BC

$=$

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

Operácia Key Expansion

- realizuje **algoritmus expanzie kľúča**, ktorý má dĺžku 16 bajtov a vyjadruje sa formou 4 slov s dĺžkou 4 bajty
- pri počte rúnd 10 je potrebná expanzia na 44 slov
- **proces expanzie kľúča** pre prvý rundový kľúč:



Operácia Key Expansion

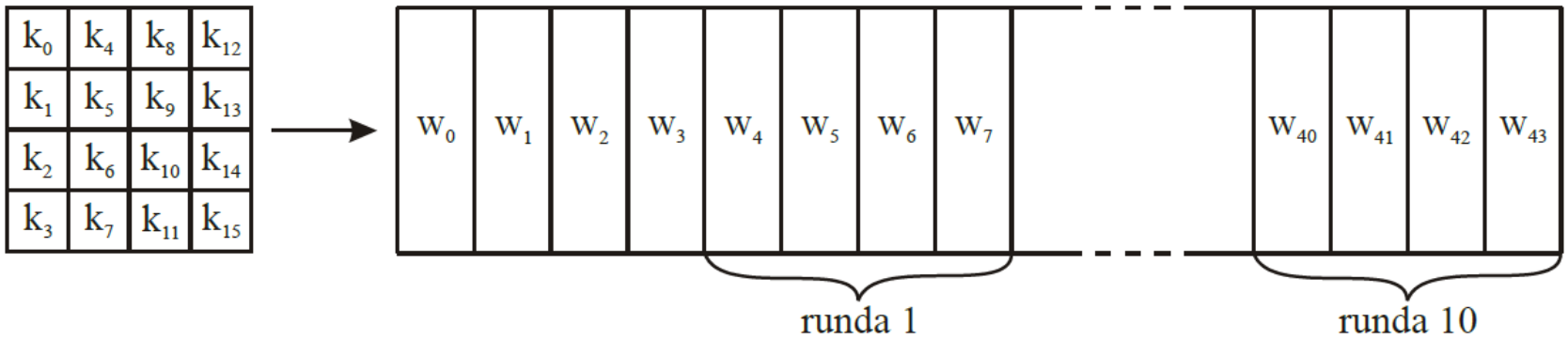
- prvé štyri bajty kľúča k_0, k_1, k_2, k_3 vytvárajú slovo w_0, ďalšie bajty k_4, k_5, k_6, k_7 vytvárajú slovo w_1 atď. - z takto získaných slov w_0, w_1 w_2 a w_3 sa generujú slová w_4, w_5, w_6 a w_7
- zatiaľ čo slová w_5, w_6 a w_7 sa vytvárajú jednoduchou operáciou XOR, slovo w_4 sa vytvára zložitejšie, pomocou funkcie f - aplikuje sa na slovo w_3 a pozostáva z 3 operácií:
  1. **operácia Rot Word** realizuje cyklický posun bajtov slova w_3 o jednu pozíciu vľavo
  2. **operácia Sub Word** realizuje substitúciu posunutých bajtov podľa S-boxu
  3. výsledok operácie Rot Word a Sub Word sa podrobí **operácii XOR s konštantou** $RC(j) = [B_j, 0, 0, 0]$, ktorej hodnota je definovaná pre každú rundu:

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

Operácia Key Expansion

- uvedený postup možno **cyklicky opakovať**, t.j. pre rundu 2 je vstupným kľúčom kľúč vytvorený v runde 1, teda w_4, w_5, w_6, w_7 a uvedeným postupom sa vygeneruje kľúč pre rundu 3 atď.

- **proces expanzie kľúča pre počet rúnd 10**:



Výhody štandardu AES

- ide o **najrobustnejší** bezpečnostný protokol (možno ho implementovať v hardvéri aj softvéri)
- na šifrovanie používa kľúče väčšej dĺžky $\Rightarrow$ je odolnejší voči hackerom (**zatiaľ bezpečný**)
- najbežnejší bezpečnostný protokol **používaný pre širokú škálu aplikácií** (bezdrôtová komunikácia, finančné transakcie, e-business, ukladanie šifrovaných dát, atď.)
- jedno z **najrozšírenejších komerčných a open source** riešení používaných po celom svete

Neýhody štandardu AES

- používa **príliš jednoduchú algebraickú štruktúru**
- každý blok je vždy **zašifrovaný rovnakým spôsobom**
- **AES v čítačovom režime** (*Counter Mode – CTR*) je zložité implementovať do softvéru, ak sa berie do úvahy výkon aj bezpečnosť
- **oslabená bezpečnosť vzhľadom na útoky cez postranný kanál**, kedy sa útočí na samotnú implementáciu šifry na hardvérových alebo softvérových systémoch (existuje niekoľko známych útokov na rôzne implementácie AES)

Ďakujem za pozornosť