Kryptografické systémy
hašovacie funkcie, MDC, MAC

doc. RNDr. Jozef Jirásek, PhD.

ZS 2023

Vyžitie hašovacích funkcií na zachovanie integrity

- integrita (celistvosť) správy – či nebola urobená zmena počas prenosu (uloženia) - šifrovacie systémy zmeny neriešia ...
- správa ľubovoľnej dĺžky => charakteristický odtlačok pevnej dĺžky (*fingerprint, message digest*)
- hašovacia funkia (*hash*) – obyčajne zobrazenie $h: \{0,1\}^* \rightarrow \{0,1\}^n$
- kryptografická hašovacia funkcia (aby ju bolo možné využiť na zabezpečenie integrity správ)
  MDC – modification detection code
  - jednocestná (*one-way*) funkcia $x \mapsto h(x)$ ťažko invertovateľná
  - odolná voči nájdeniu druhého vzoru (*second preimage resistance*)
    pre dané x je ťažké nájsť iné x‘, aby h(x) = h(x‘)
  - odolná voči kolíziám (*collision resistance*)
    je ťažké nájsť rôzne x, x‘, aby h(x) = h(x‘)

MDC

MDC – modification detection code $h: \{0,1\}^* \rightarrow \{0,1\}^n$

- jednocestná (*one-way*) hašovacia funkcia $x \mapsto h(x)$
- odolná voči nájdeniu druhého vzoru (*second preimage resistance*) - pre dané x je ťažké nájsť iné x‘, aby h(x) = h(x‘)
- odolná voči kolíziám (*CRF - collision resistance - strong*) je ťažké nájsť rôzne x, x‘, aby h(x) = h(x‘) (CRF ⇒ s. p.)

Pre prijatú správu x s MDC H overenie integrity: h(x)=?H
(pri zmene x by musel útočník zmeniť aj H)

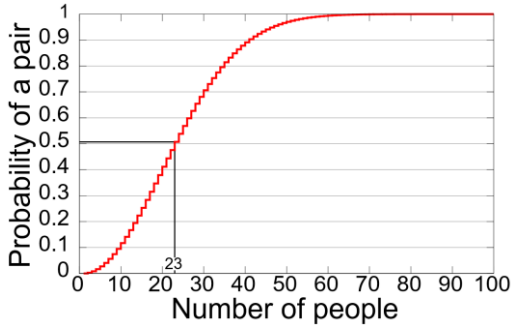
útok na nájdenie vzoru – 2^n možností (šanca na ~2^{n-1} pokus)

útok na nájdenie druhého vzoru – testovanie 2^n hodnôt pre x‘

útok na nájdenie kolízie – „narodeninový paradox“

Narodeninový útok

- aká je pravdepodobnosť, že v miestnosti majú dvaja narodeniny v ten istý deň v roku ?
- pravdepodobnosť, že majú všetci v rôznych dňoch :
  pre dvoch $365 \cdot 364 / 365^2 = 0{,}997$
  pre troch $365 \cdot 364 \cdot 363 / 365^3 = 0{,}997 * 0{,}995 = 0{,}992$
  pre k $365 \cdot 364 \cdot \ldots \cdot (365\text{-}k\text{+}1) / 365^k = \frac{365!}{365^k \cdot (365-k)!}$
- rastie pravdepodobnosť, že aspoň dvaja majú tieto narodeniny rovnako (pre k = 366 - istota)
- kedy bude $(1 - \frac{365!}{365^k \cdot (365-k)!}) > 0{,}5$?
- pre k = 23 dostaneme 50,73 % pravdep. (pre k = 50 už 97 %)



Útoky, využívajúce kolízie

- aby $P(\exists\, x, x' \in \{x_1, x_2, \ldots, x_k\};\ h(x) = h(x')) > p$

  potrebujeme $k \geq \sqrt{|h(X)|} \cdot \sqrt{2\, ln\frac{1}{1-p}}$

  pre $|h(X)| = 2^n$ na útok s pravdep. 50 % $k \approx 1{,}177 \cdot 2^{n/2}$

  90 % $k \approx 2{,}146 \cdot 2^{n/2}$

  99 % $k \approx 3{,}035 \cdot 2^{n/2}$

- pripraviť dve správy, ktoré bude možné navzájom zameniť (budú mať rovnaký odtlačok) :
  v každej správe nájsť n/2 miest, ktoré je možné upraviť bez zmeny významu textu (medzery, synonymá, nezobrazované znaky ...) – z nich vyrobiť $2^{n/2}$ správ (a odtlačkov) každého druhu. S veľkou pravdepodobnosťou sa medzi nimi nájde dvojica rôzneho druhu s rovnakým odtlačkom

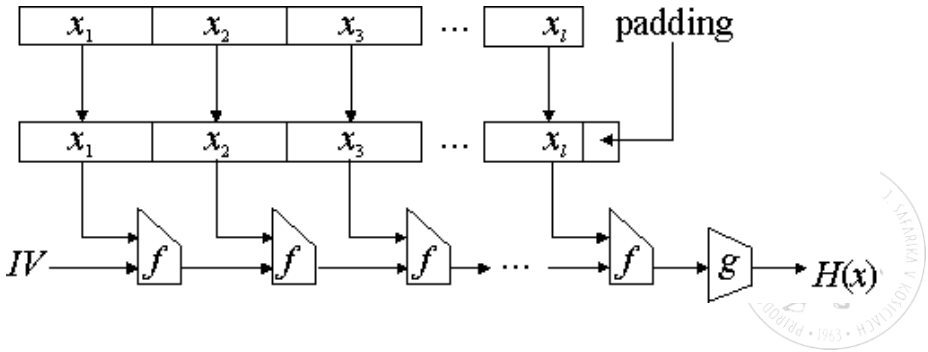
Merkle-Damgårdova konštrukcia

blokové schémy

- rozdelenie správy na bloky
- štart s IV a sekvenčné spracovanie blokov
- výplň posledného

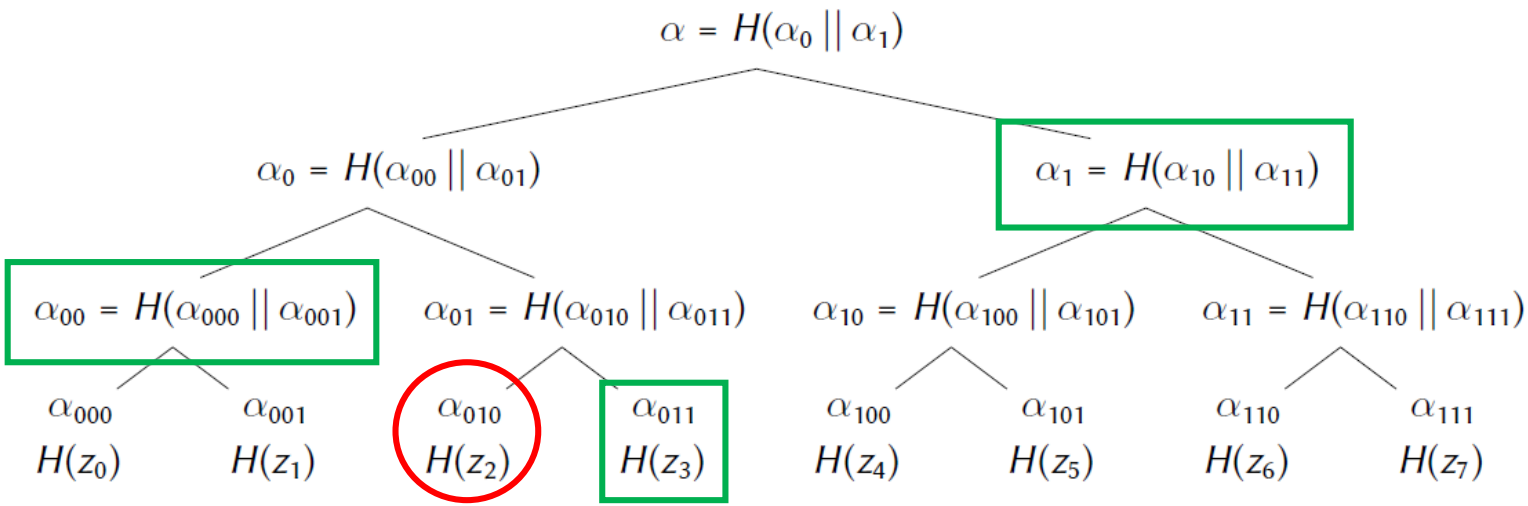
Merkle-Damgårdova konštrukcia (zachová odolnosť voči kolíziám)
$f : \{0,1\}^{n+r} \rightarrow \{0,1\}^n$ $x = x_1 x_2 \ldots x_k$ bloky veľkosti r bitov, posledný s výplňou 100... (resp. dĺžka)

$h_0 = IV$
$h_i = f(h_{i-1} | x_i)$ pre $i = 1,2,\ldots,k$
$H(x) = g(h_k) = f(h_k \mid |x|)$



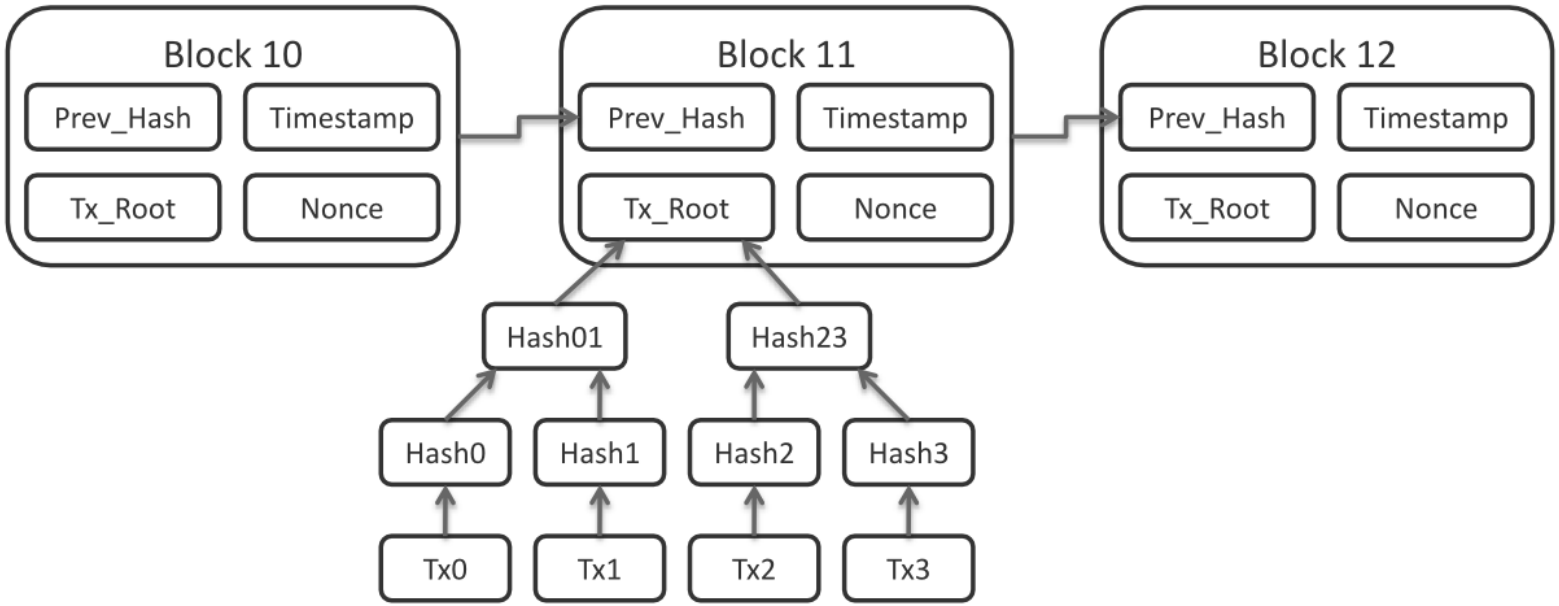
Merkleho hašovacie stromy

- Merkle 1979 usporiadanie hašov do stromu (Btrfs, Bitcoin,..)
- vstup pre i = 0, ..., 2^n-1 správy z_i s hašom $H(z_i)$
- H(a || b) – haš ľavého a pravého potomka v binárnom strome koreň α – spoločný haš na overenie integrity
- overenie integrity správy z_i pomocou $H(z_i)$ a postupnosti hodnôt súrodeneckých vrcholov na ceste ku koreňu (dĺžky n)



Blockchain – zreťazenie Merkleho stromov

Blockchain ledger

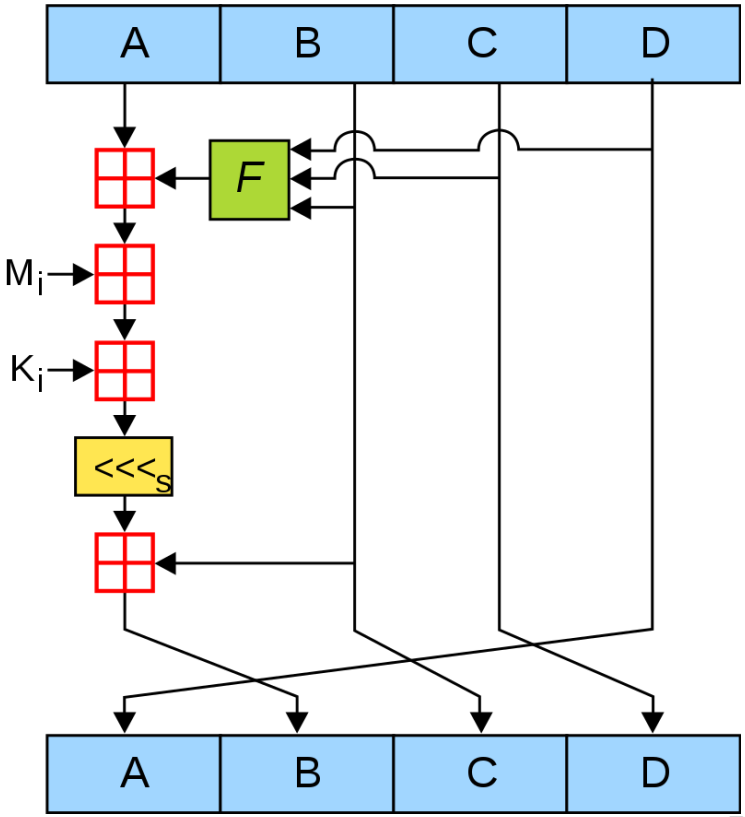


MD5

(Rivest 1991)

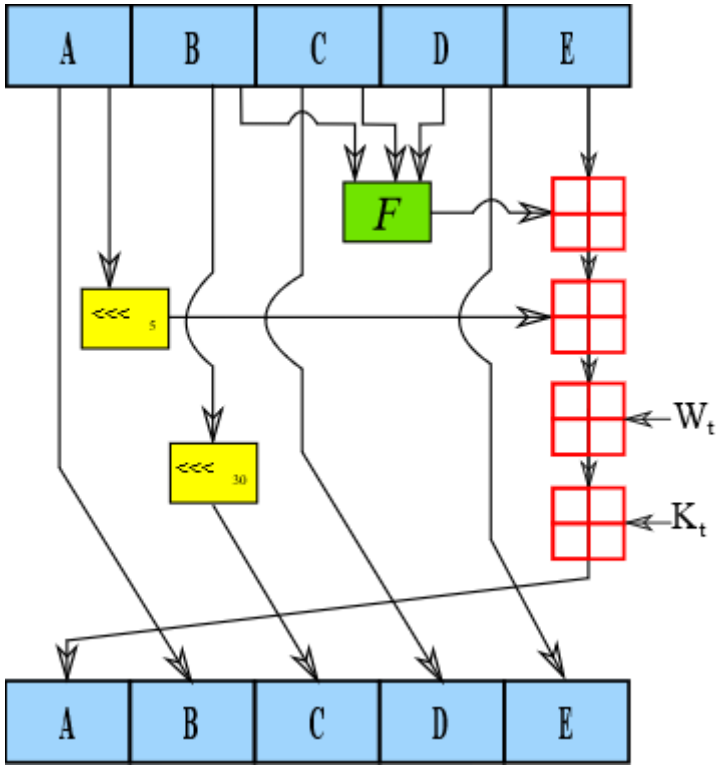
- 128 bitová MDC funkcia
- 512 bitové bloky
- výplň $100..0(|m|)_{64}$
- 4 rundy po 16 operáciách
- $F(B,C,D)=(B \wedge C) \vee (\neg B \wedge C)$ ...
  (4 rôzne pre každú rundu)
- M_i časť správy (32 bit)
  K_i konštanty (pre rundu)
- $\boxplus$ sčítanie mod 32

2008 - známe CRF kolízie



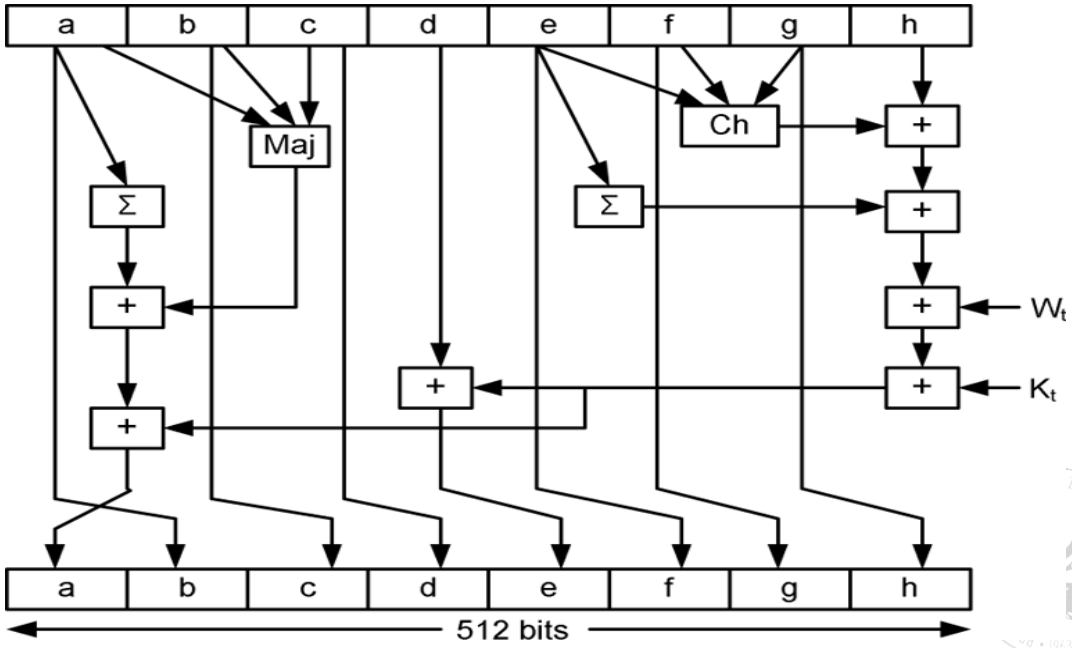
SHA-1

- Secure Hash Algorithm 1 (NSA 1995 pre DSS)
- 160 bitov pre 512 bitové bloky
- padding ako MD5 (max 2^{64} b)
- 4 rundy po 20 operáciách
- 32 bitové súčty
- 2.2017 - známe kolízne útoky (2^{63} výpočtov ~ 110 rokov GPU)



SHA-2

- (NSA 2001) SHA-224, SHA-256, SHA-384, SHA-512
- bloky 512 resp. 1024 bitov, výpočty 8x 32 B resp. 8x 64 B
- $Maj(a,b,c)=(a \wedge b) \oplus (a \wedge c) \oplus (b \wedge c)$ $Ch(e,f,g)=(e \wedge f) \oplus (\neg e \wedge g)$
- 64 (80) rúnd
- bitcoin (SHA-256)
  SHA(SHA(Bl,N))
  zač. 80 bitov je 0

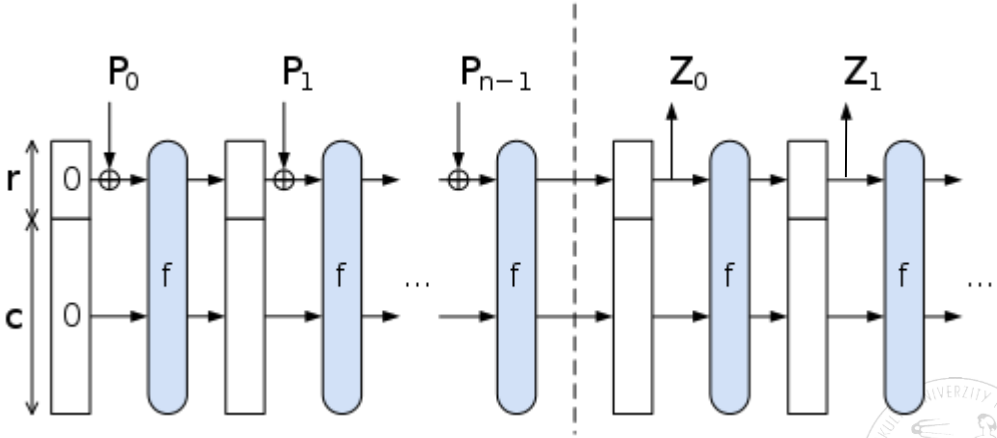


SHA-3 Secure Hash Algorithm

- 2006 NIST – súťaž o nový štandard (224,256,384,512,var)
- 2012-2015 Keccak (Bertoni, Daemen, Peeters, van Assche)
- NIST FIPS 202 (2015):
  - SHA3-224, SHA3-256, SHA3-384, SHA3-512, stream, AE, AEAD
  - SHAKE128, SHAKE256 (XOF – extendable output function)
- 1600 vnútorných stavových bitov – 5x5 64 b registrov (môže byť aj menší)
- 24 rúnd – v každej 5 operácií – θ parita, ρ rotácia, π permutácia, χ kombinácia riadkov (nelineárna), ι xor s LFSR (láme symetriu)
- špongiovitá (sponge) konštrukcia – nepoužíva MD schému

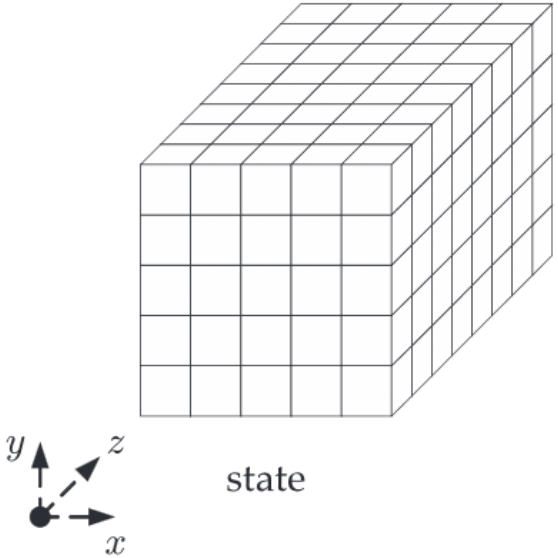
SHA-3 špongiovitá konštrukcia

- c – kapacita (2x počet bitov hašu – stupeň bepečnosti)
- P – vstup, rozdelený na bloky P_0, P_1, ..., P_{n-1} veľkosti r (rate) = 1600 – c, padding (x|p10*1) v poslednom bloku
- absorbcia – xor P_i blok k registru a výpočet f (24 rúnd Keccaku)
- vyžmýkane (squeeze) výsledku po r bitoch – hlavne pre SHAKE režimy
- proti útokom na zväčšovanie počtu blokov správy



SHA 3

- $b = 1600$ bitový stav môže byť vnímaný ako 3 dimenzionálne pole
- kocka 5 x 5 x 64, v každej jednotkovej kocke je 1 bit



SHA 3 – funkcia Θ

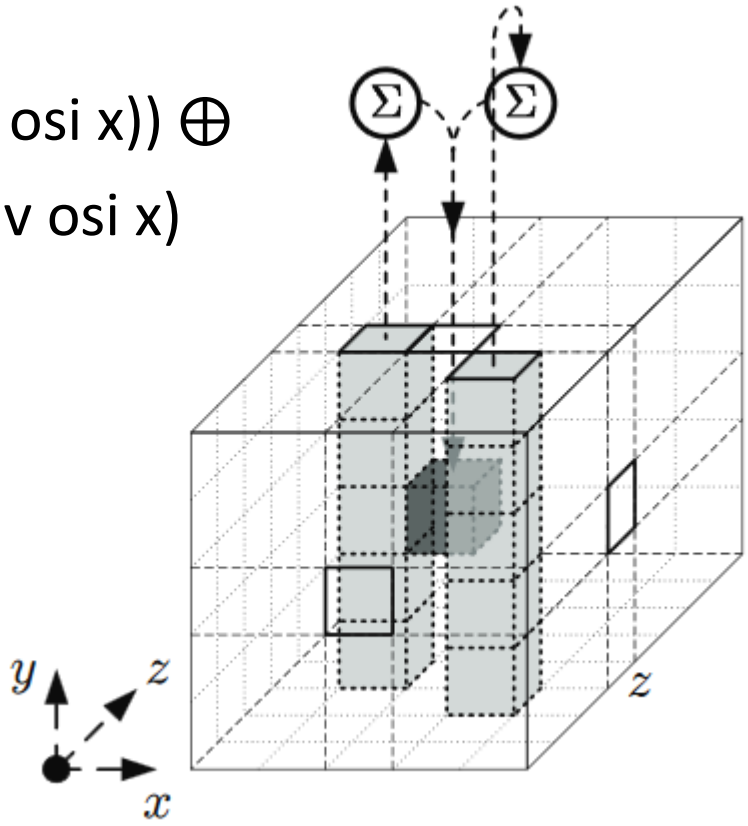
- každý bit (z 1600) je nahradený podľa susedných 10 bitov:

(pôvodný bit) ⊕

(parita 5 bitového stĺpca „vľavo“ (v osi x)) ⊕

(parita 5 bitového stĺpca „vpravo“ (v osi x) a „dopredu“ (v osi z)

- na okrajoch - rotácia modulo 5



Zdroj: http://www.crypto-textbook.com/

SHA 3 – funkcia Θ

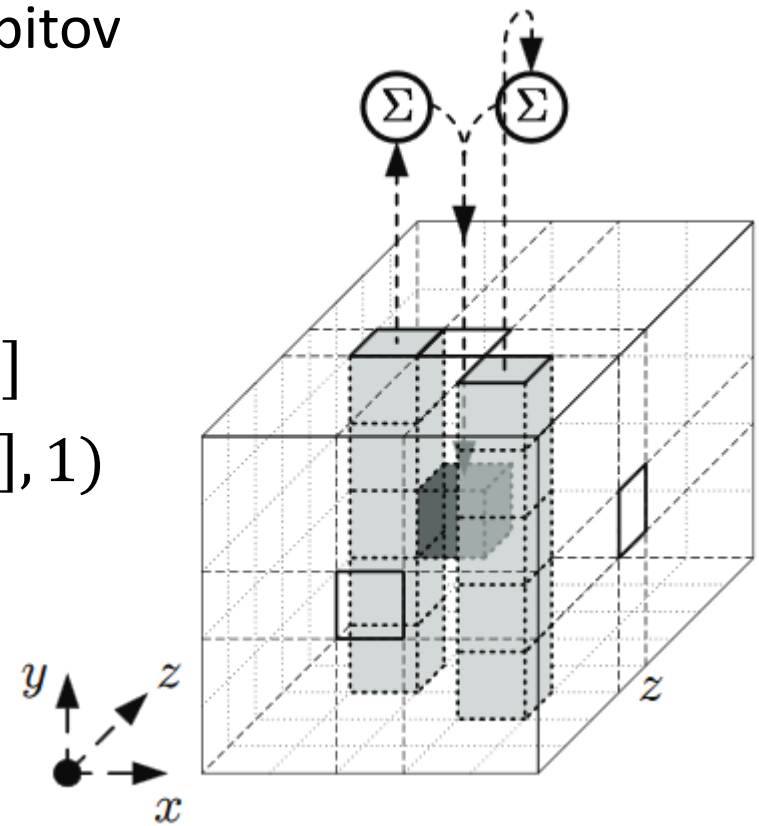
iný pohľad:

- operácie na 25 slovách dĺžky 64 bitov
  (súradnica z)
- $A[x, y];\ x, y = 0, 1, 2, 3, 4$

$$C[x] = A[x, 0] \oplus A[x, 1] \oplus$$
$$\oplus A[x, 2] \oplus A[x, 3] \oplus A[x, 4]$$

$$D[x] = C[x - 1] \oplus rot_z(C[x + 1], 1)$$

$$A[x, y] = A[x, y] \oplus D[x]$$

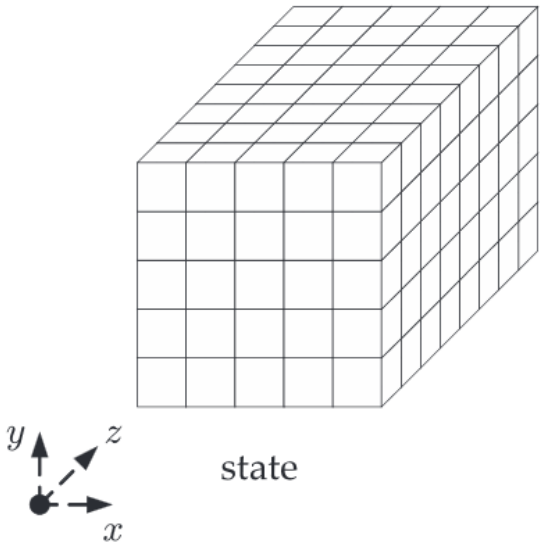


Zdroj: http://www.crypto-textbook.com/

SHA 3 – funkcia ρ

- $TEMP[x,y] = rot_z(A[x,y], r[x,y])$
  rotácia v smere z podľa tabuľky
- $r[x,y] =$

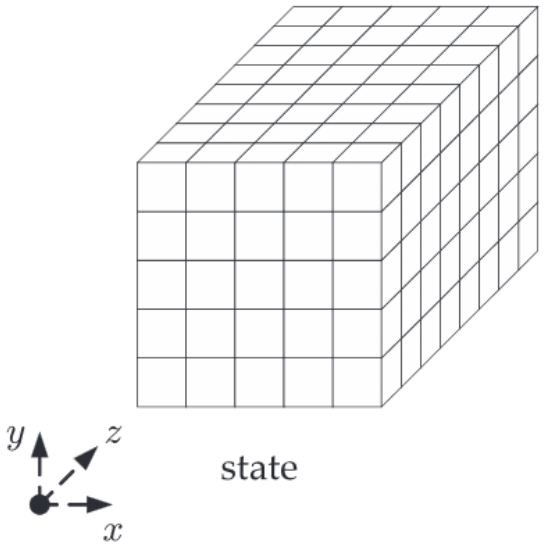
	x=3	x=4	x=0	x=1	x=2
y=2	25	39	3	10	43
y=1	55	20	36	44	6
y=0	28	27	0	1	62
y=4	56	14	18	2	61
y=3	21	8	41	45	15



SHA 3 – funkcia π

- permutácia zarotovaných slov
- $\mathrm{B}[y, 2x + 3y] = rot_z(A[x, y], r[x, y])$

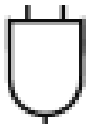
$$x, y = 0, 1, 2, 3, 4$$

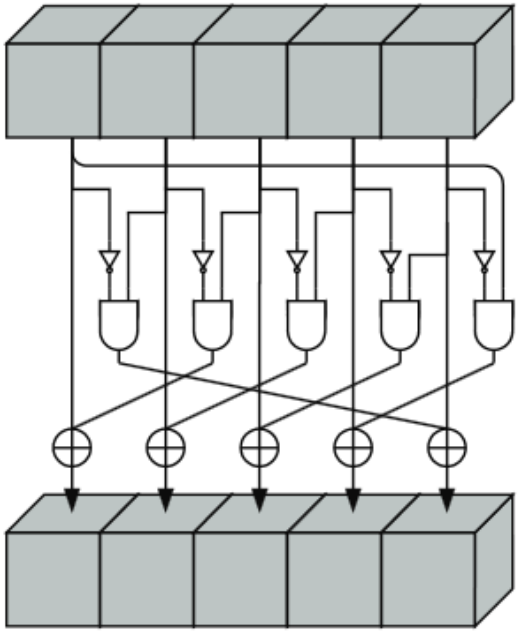


SHA 3 – funkcia χ

- $A[x,y] = B[x,y] \oplus (\bar{B}[x+1,y] \wedge B[x+2,y])$

($\bar{B}[x+1,y]$ je vektor negovaných bitov vektora $B[x+1,y]$)

- nelineárna operácia
- negácia
- and 



Zdroj: http://www.crypto-textbook.com/

SHA 3 – funkcia ι

- pripočítanie konštanty z tabuľky (dá sa použiť LFSR) k slovu $A[0,0]$
- $A[0,0] = A[0,0] \oplus \mathrm{RC}[i]$ kde i je číslo rundy

RC[0]	= 0x0000000000000001	RC[12]	= 0x000000008000808B
RC[1]	= 0x0000000000008082	RC[13]	= 0x800000000000008B
RC[2]	= 0x800000000000808A	RC[14]	= 0x8000000000008089
RC[3]	= 0x8000000080008000	RC[15]	= 0x8000000000008003
RC[4]	= 0x000000000000808B	RC[16]	= 0x8000000000008002
RC[5]	= 0x0000000080000001	RC[17]	= 0x8000000000000080
RC[6]	= 0x8000000080008081	RC[18]	= 0x000000000000800A
RC[7]	= 0x8000000000008009	RC[19]	= 0x800000008000000A
RC[8]	= 0x000000000000008A	RC[20]	= 0x8000000080008081
RC[9]	= 0x0000000000000088	RC[21]	= 0x8000000000008080
RC[10]	= 0x0000000080008009	RC[22]	= 0x0000000080000001
RC[11]	= 0x000000008000000A	RC[23]	= 0x8000000080008008

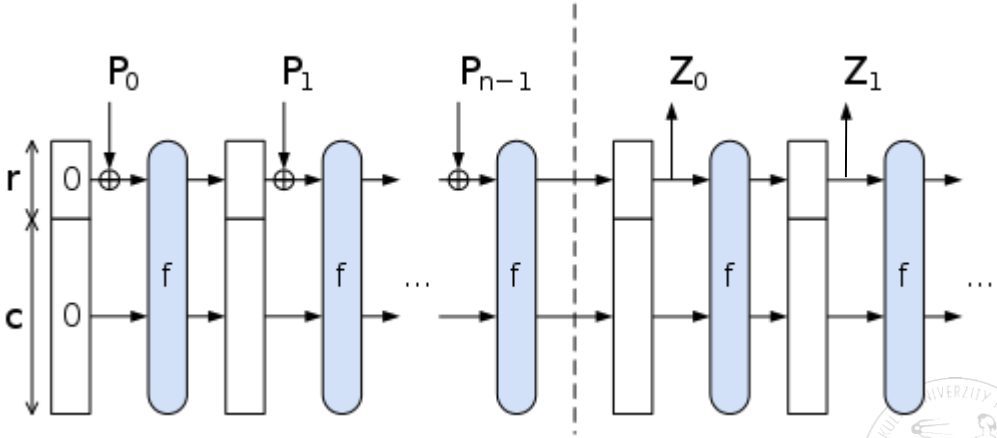
Zdroj: http://www.crypto-textbook.com/

SHA 3 – runda (Θ ρ π χ ι)

- $C[x] = A[x,0] \oplus A[x,1] \oplus$
  $\oplus A[x,2] \oplus A[x,3] \oplus A[x,4] \quad x = 0,1,2,3,4$
- $\mathrm{B}[y, 2x+3y] = rot_z(A[x,y] \oplus C[x-1] \oplus$
  $\oplus rot_z(C[x+1],1),\ r[x,y])$
  $x, y = 0,1,2,3,4$
- $A[x,y] = B[x,y] \oplus (\bar{B}[x+1,y] \wedge B[x+2,y])$
- $A[0,0] = A[0,0] \oplus \mathrm{RC[i]}$

SHA-3 špongiovitá konštrukcia

- c – kapacita (2x počet bitov hašu – stupeň bepečnosti)
- P – vstup, rozdelený na bloky P_0, P_1, ..., P_{n-1} veľkosti r (rate) = 1600 – c, padding (x|p10*1) v poslednom bloku
- absorbcia – xor P_i blok k registru a výpočet f (24 rúnd Keccaku)
- vyžmýkane (squeeze) výsledku po r bitoch – hlavne pre SHAKE režimy
- proti útokom na zväčšovanie počtu blokov správy



Ekvivalentné dĺžky kľúčov

AES	RSA	DLP	ECC	MDC
48	480	480	96	96
56	640	640	112	112
64	816	816	128	128
80	1248	1248	160	160
112	2432	2432	224	224
128	3248	3248	256	256
160	5312	5312	320	320
192	7936	7936	384	384
256	15424	15424	512	512

MAC – message authentication code

- autentifikačné kódy správ
- zabepečenie integrity a pôvodnosti (autenticity) správy
- vyžaduje kľúč, ktorým je možné kontrolovať pôvodnosť (správu môže zmeniť len ten, kto pozná kľúč)
- narodeninový útok sa nedá použiť – MAC môže byť kratšia ako MDC
- (kryptografický kontrolný súčet)
- neumožňuje jednoznačne určiť pôvodcu (kľúč poznajú viacerí)
- kľúč + hašovacia funkcia alebo kľúč + symetrická šifra

MAC z hašovacej funkcie

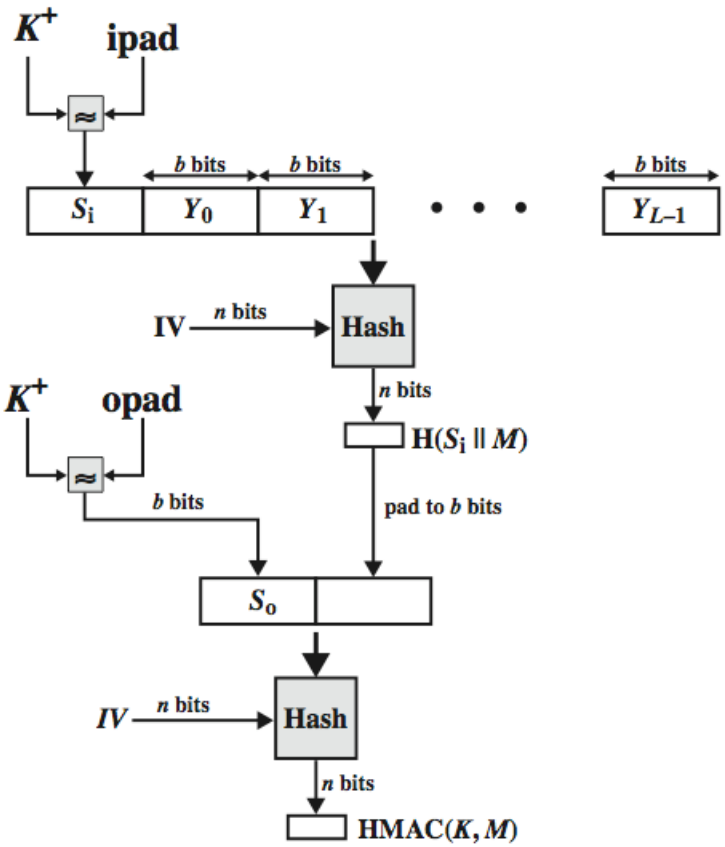
- $MAC_k(m) = h(k|m)$
  ak sa použije MD schéma – možno rozšíriť správu
- $MAC_k(m) = h(m|k)$
  ak sa nájdu kolízie v h je ohrozená bezpečnosť kľúča
- ďalšie možnosti ?

- kombinácia $MAC_k(m) = E_k(h(m))$
  kvôli veľkosti blokovej šifry môže byť ohrozená bezpečnosť h
- KMAC – Keccak MAC vyrobený špongiovou technológiou
  $h(k|m)$

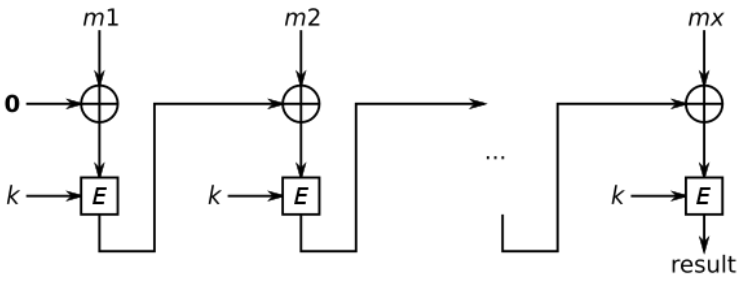
HMAC

- $HMAC_k(m) = h(k \oplus opad \mid h(k \oplus ipad \mid m))$
- ipad = 0x363636…
- opad = 0x5c5c5c…

- TLS, SSH, IPSec …



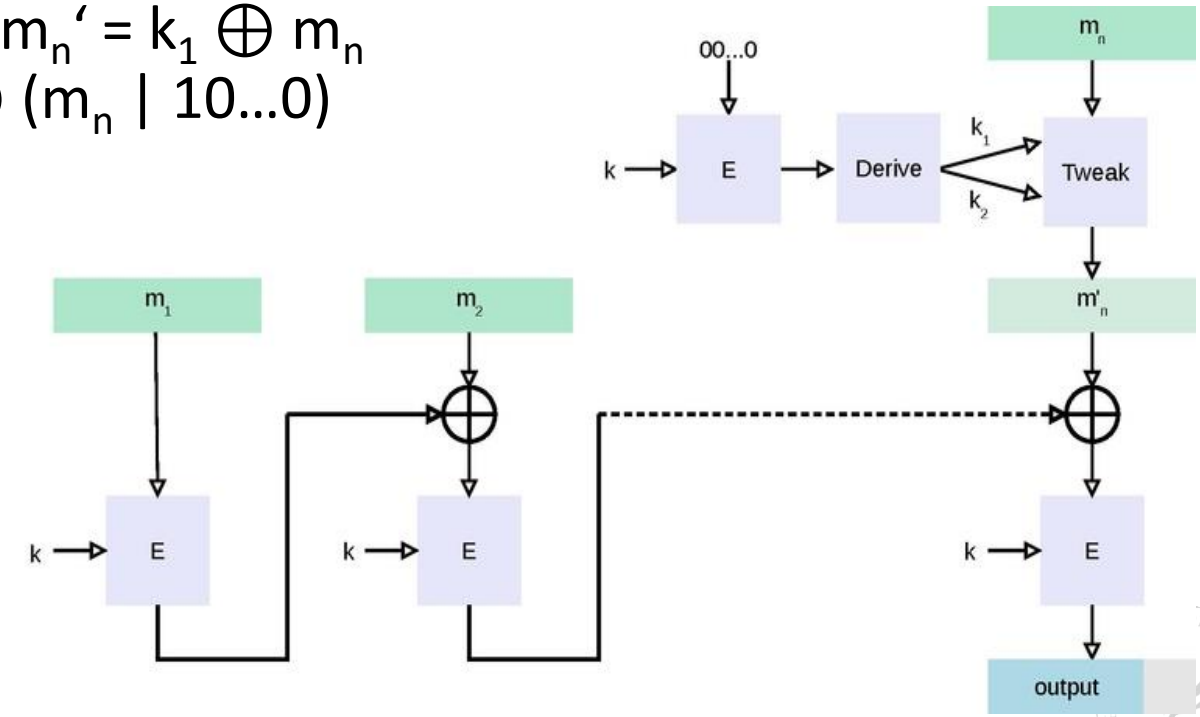
CBC-MAC



- konštrukcia MAC funkcie pomocou blokovej šifry
- ako v CBC režime – výsledkom je len posledný blok (meniť IV ?)
- pre vstupy rôznej dĺžky:
  $h = MAC_k(m) = E_k(m)$ $h' = MAC_k(m') = E_k(m')$
  ak poznám aj haš dvoch blokov (m| x)
  $MAC_k(m|\ x) = E_k(E_k(m) \oplus x)$, potom pre $MAC_k(m'|\ h \oplus h' \oplus x) =$
  $= E_k(E_k(m') \oplus E_k(m) \oplus E_k(m') \oplus x) = E_k(E_k(m) \oplus x) = MAC_k(m|\ x)$
- teda môžeme vyrobiť MAC pre inú správu $(m'|\ h \oplus h' \oplus x)$
- výsledok je vhodné upraviť ďalším kľúčom
  $CBC\text{-}MAC\text{-}ELB(m,(k1,k2)) = E_{k2}(CBC\text{-}MAC_{k1}(m))$

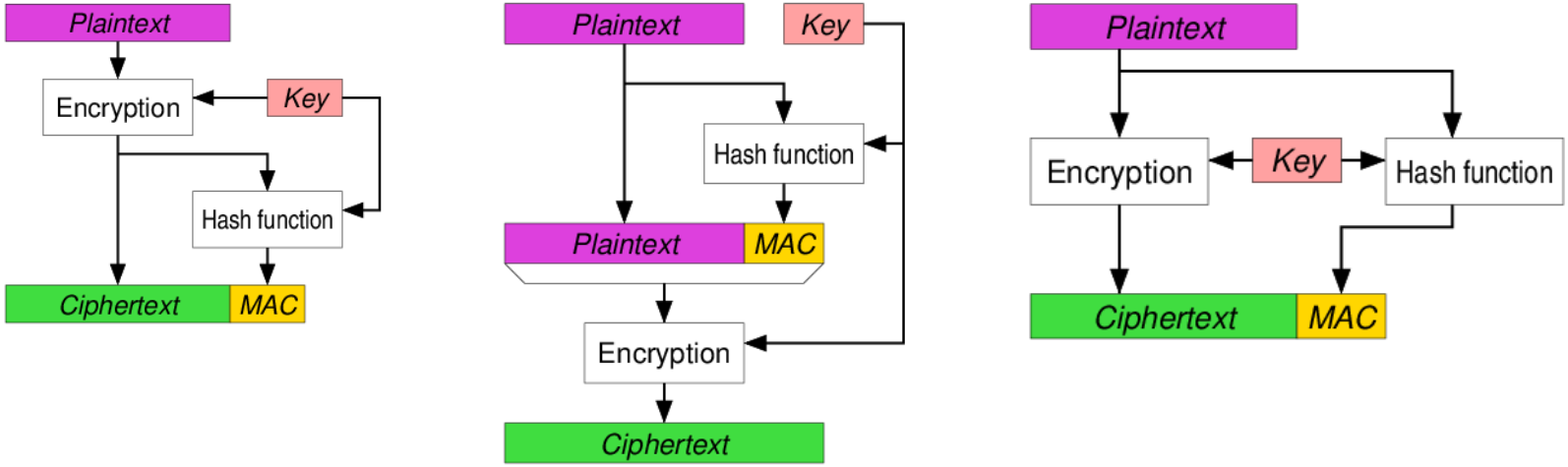
OMAC – one-key MAC

- podľa CMAC, XCBC
- z kľúča k sa vyrobia dva kľúče k_1 a k_2
- ak m_n je úplný, $m_n' = k_1 \oplus m_n$
  inak $m_n' = k_2 \oplus (m_n \mid 10...0)$



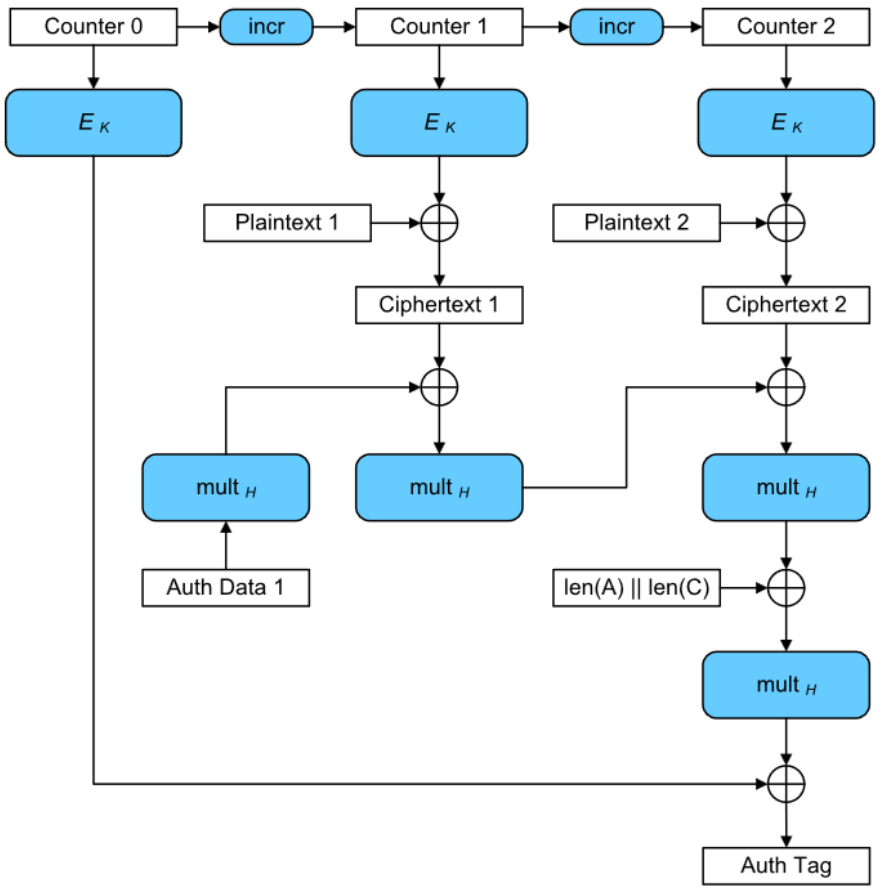
AE – authenticated encryption

- výhody spojenia dôvernosti, integrity a pôvodnosti správy
- prístupy EtM (IPSec), MtE (TLS) a E&M (SSH)



- CAESAR – competition for authenticated encryption
  ECRYPT 2007 - finále 15.12.2017 – Ascon, ACORN, AEGIS-128

GCM Galois Counter Mode (authenticated encryption)



šifrovanie (utajenie)

- $CTR_0 = IV$,
  $CTR_n = CTR_{n-1} + 1$
- $C_i = E_K(CTR_i) \oplus P_i$

autentifikácia správy

- $H = E_K(0) \quad g_0 = AD \otimes H$

$\otimes$ Galois field multiplication
$GF(2^{128})$ modulo $x^{128} + x^7 + x^2 + x + 1$
{10...010000111}

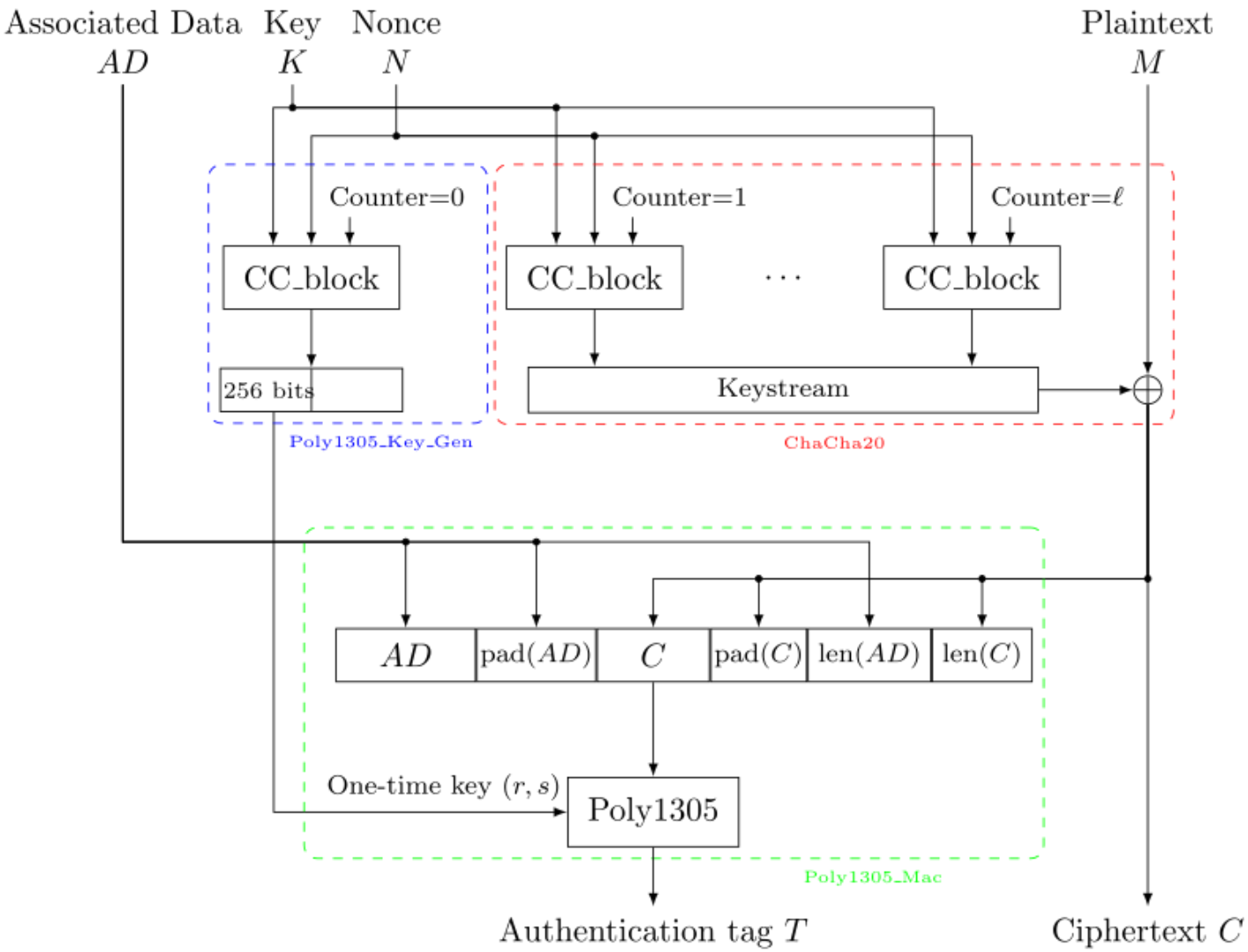
- $g_i = (g_{i-1} \oplus C_i) \otimes H$
- $g_{n+1} = g_n \oplus (len(A) \mid len(C))$
- $T = (g_{n+1} \otimes H) \oplus E_K(CTR_0)$

Poly 1305 (Bernstein 2005)

$Poly1305_{r,s}(m)$ - 128 bitov
r, s – kľúče po 128 b (16 B) ; $m = m_1 m_2 \ldots m_n$ po 128 b
(r redukované o 22 bitov)

- $T_0 = 0$
- $T_i = ((T_{i-1} + m_i + 2^{128}) \cdot r) \bmod (2^{130} - 5)$ aritmetika 5 x 26 b
- $Poly1305_{r,s}(m) := T = (T_n + s) \bmod 2^{128}$

ChaCha20 + Poly1305 (Bernstein)

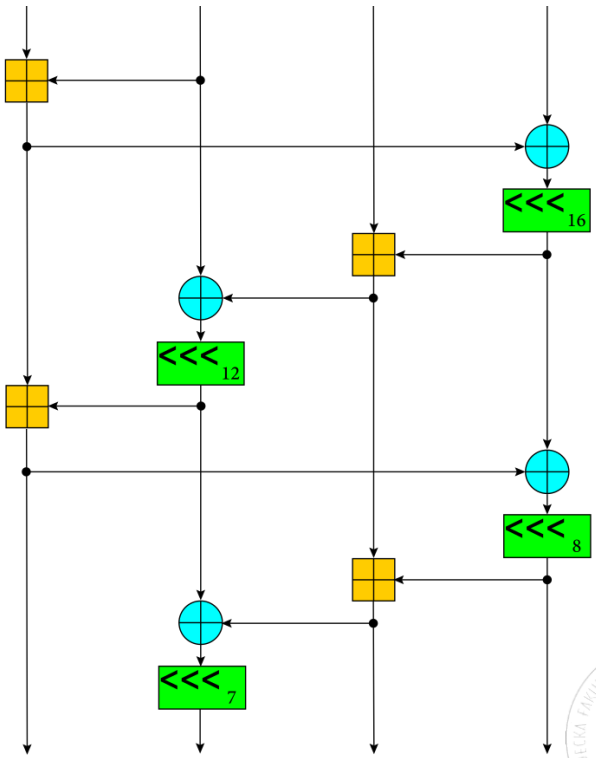


Ďakujem za pozornosť.

jozef.jirasek@upjs.sk

ChaCha20

- Bernstein 2008 – namiesto RC4 v TLS (RFC 7539) ARM
- kľúč 256 b, IV 64 b, cnt 64 b
  (resp. IV 96 b, cnt 32 b)
- do 512 b bloku

„expa“	„nd 3“	„2-by“	„te k“
K	K	K	K
K	K	K	K
cnt	cnt	IV	IV

- 10x 8 transf. (4 stĺpce + 4 diagonály)
  spolu 20x prepis celého bloku
  možno paralelizovať
- kvôli cnt len 2^{32} blokov (256 GB)