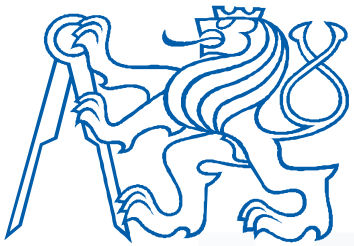
České vysoké učení technické v Praze
Fakulta elektrotechnická
Katedra telekomunikační techniky

A7B32KBE 5. přednáška

Režimy činnosti blokových šifer

Ing. Tomáš Vaněk, Ph.D. tomas.vanek@fel.cvut.cz



Šifrování více bloků OT

- Jak šifrovat více bloků OT?
- Používat různé pro každý blok nový klíč ?
  - ale to je (přinejlepším) stejně neefektivní jako one-time pad !
- Šifrovat každý blok nezávisle na ostatních?
- Šifrovat v závislosti na předchozích blocích např. nějaké zřetězení bloků dohromady?
- Co s neúplnými bloky (když délka OT není celistvým násobkem délky bloku)?

Co to je „režim činnosti“?

Režim činnosti blokové šifry (block cipher mode of operation), zkráceně režim činnosti je algoritmus využívající symetrické blokové šifry tak, aby zajistil některé cíle informační bezpečnosti např. utajení nebo autentizaci.

Režimy činnosti pro zajištění utajení

- FIPS 81 - ECB, CBC, CFB a OFB
  - vydán v 70.letech (DES)
  - definice **základních** režimů
- NIST SP800-38A
- SP800-38A obsahuje všechny režimy z FIPS 81 a
  - režim CTR
  - rozšíření CBC o „cipher-text stealing“ (CTS)
    - pro aplikace, s různými délkami zpráv, které jsou nesoudělné s velikostí bloku
    - nezvětšuje velikost ŠT
    - nevýhoda - složitější
    - změna proti CBC pouze v posledních dvou blocích
    - poslední dva bloky se odesílají v opačném pořadí

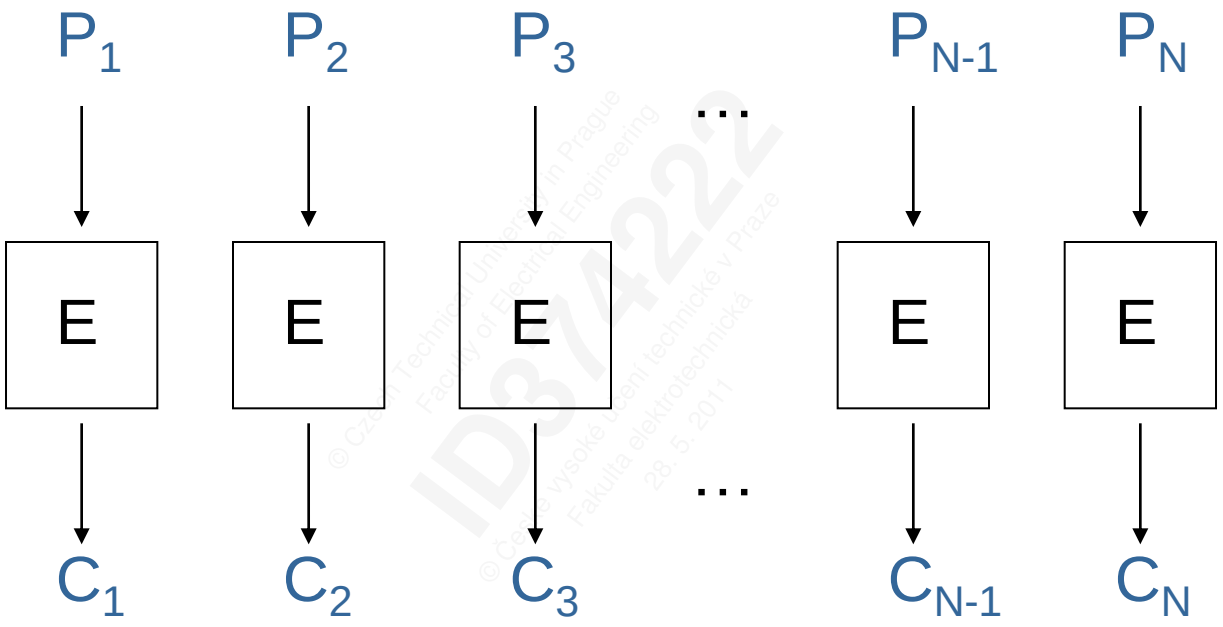
Režimy činnosti pro zajištění autentizace

- NIST SP800-38B
- slouží pouze k autentizaci zpráv
- CMAC (viz. přednáška o autentizačních protokolech)
- OMAC - One Key CBC-MAC
- XCBC

Kombinované režimy činnosti

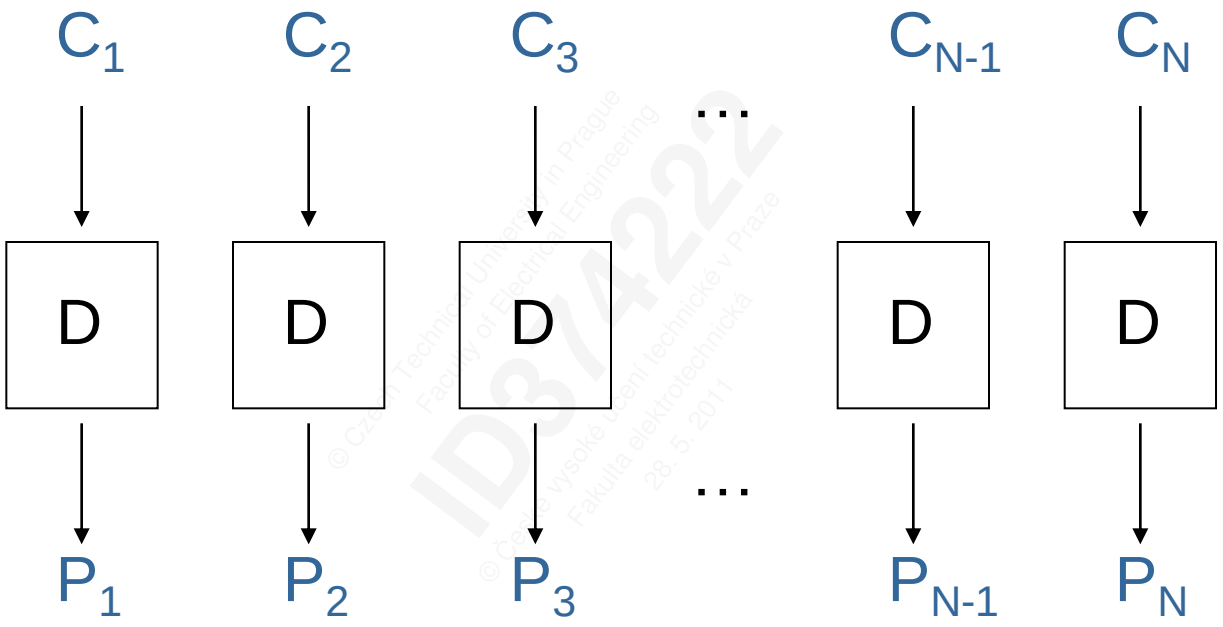
- utajení + autentizace
- CCM (**C**ounter **C**BC-**M**AC)
  - AES-CCM (viz. 802.11i)
  - RFC 3610
  - NIST SP800-38C
- GCM (**G**alois **C**ounter **M**ode)
  - AES-GCM
  - vysoká propustnost u HW implementace
  - NIST SP800-38D

Režim ECB - šifrování



$C_i = E(P_i)$ pro i=1..N

Režim ECB - dešifrování



$P_i = D(C_i)$ pro i=1..N

Režim ECB

- Zapisujeme: $C=E(P,K)$
- OT: $P_0, P_1, \ldots, P_m, \ldots$ ŠT: $C_0, C_1, \ldots, C_m, \ldots$
- Obvyklý způsob použití blokové šifry:

Šifrování

$C_0=E(P_0,K)$,
$C_1=E(P_1,K)$,
$C_2=E(P_2,K)$,…

Dešifrování

$P_0=D(C_0,K)$,
$P_1=D(C_1,K)$,
$P_2=D(C_2,K)$,…

- Pro daný klíč K, to je elektronická verze kódové knihy.
- Pro každý klíč *K* existuje jiná kódová kniha!

Špatné vlastnosti ECB

- Pro blok délky 64 bitů má ECB velikost 2^{64} položek.

- Nechť $P_i = P_j$
- Pak $C_i = C_j$ a útočník ví, že $P_i = P_j$.
- To dává útočníkovi určitou informaci , přestože nezná P_i nebo P_j .
- Je možné realizovat útok typu „cut-and-paste”.
- Tabulku šifer je možné začít konstruovat i bez znalosti klíče.
- Ve většině situací mají části zprávy tendenci se opakovat. (začátky zpráv obsahují záhlaví, která mohou být velice podobné, podobně i konce zpráv)

Špatné vlastnosti ECB

Zprávy mohou mít parametry které mohou nabývat pouze několika hodnot.

Útočník může modifikovat obsah zprávy bez toho, aby byla modifikace zachycena na přijímači straně bez znalosti klíče. (dokonce bez znalosti algoritmu).

Výhody ECB:

Kazdy blok je šifrován nezávisle.

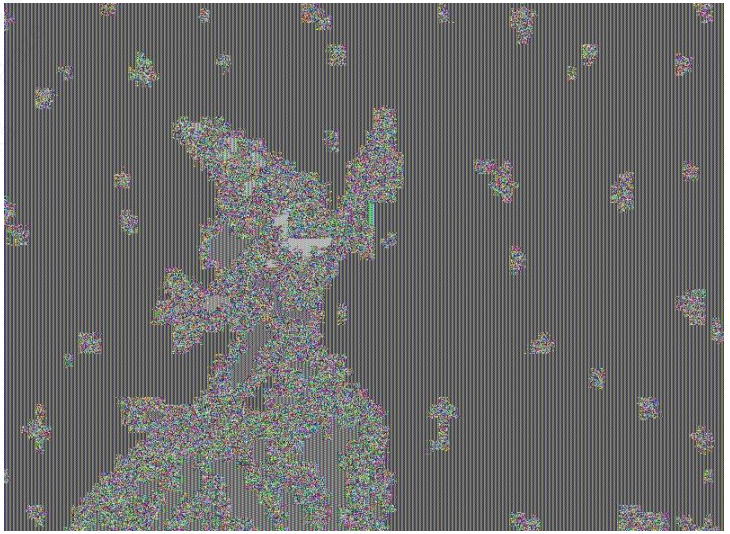
Jednotlivé bloky je možné šifrovat v libovolném pořadí.

Možnost paralelizace při šifrování i dešifrování.

Špatné vlastnosti ECB

- Obrázek Mickey Mouse

- Zašifrovaný obrázek
- režim ECB (algoritmus TEA)



Jak je to možné ?

Stejný blok OT $\Rightarrow$ stejný blok ŠT

ECB – příklad zneužití

- *Phantasy Star Online: Blue Burst*
- online MMORPG hra používající algoritmus Blowfish v režimu ECB.
- Hráči mohli ilegálně zvyšovat zkušenost opakovaným posíláním paketu obsahujícím zprávu "monster killed" zašifrovanou pomocí algoritmu Blowfish v režimu ECB.

ECB – ukázka Cut-and-Paste útoku

- Nechť OT je:

  `Alice svádí Toma. Klára svádí Jana.`

- Pak (při délce bloku 64-bitů a 8-bitovém ASCII kódování)

  P_0="Alice sv", P_1="ádí Toma.",

  P_2="Klára sv", P_3="ádí Jana. "

- ŠT: C_0, C_1, C_2, C_3

Útočník prohodí bloky C_1 a C_3, takže příjemce obdrží zprávu $ŠT' = C_0, C_3, C_2, C_1$

- Po dešifrování:

  `Alice svádí Jana. Klára svádí Toma.`

Šíření chyb

- Reakce jednotlivých módů na chyby při přenosu dat
- Chyby mohou být dvojího druhu:
  - záměna bitu za jiný
  - vložení nebo ztráta bitu
- Všechny blokové šifry obecně reagují na chybu tak, že jednobitová chyby zašifrovaného textu po dešifrovaní zničí minimálně jeden celý blok otevřeného textu.
- Módy které mezi sebou spojují jednotlivé bloky mohou mít důsledky na šíření takovéto chyby.
- ECB reaguje na změnu bitu ztrátou celého bloku a na vložení nebo ztrátu bitu ztrátou bloku v němž se chyba vyskytla a všech následujících.

Zarovnání OT (padding)

Většinou zprávy nekončí přesně na hranicích bloku.
Poslední blok je kratší. Je potřeba ho doplnit.

Nejjednodušší (a nejhloupější): doplnit poslední blok pravidelnou strukturou - nulami, jedničkami, nebo alternaci 0,1 do konce bloku.

Složitější způsob vyplňování:

- pokud je vyžadováno, aby ŠT byl stejně dlouhý jako OT
- poslední kompletní blok se po zašifrování se zašifruje ještě jednou. Pak se z něj vezme prvních *n* bitů a provede XOR s n-bity krátkého bloku.

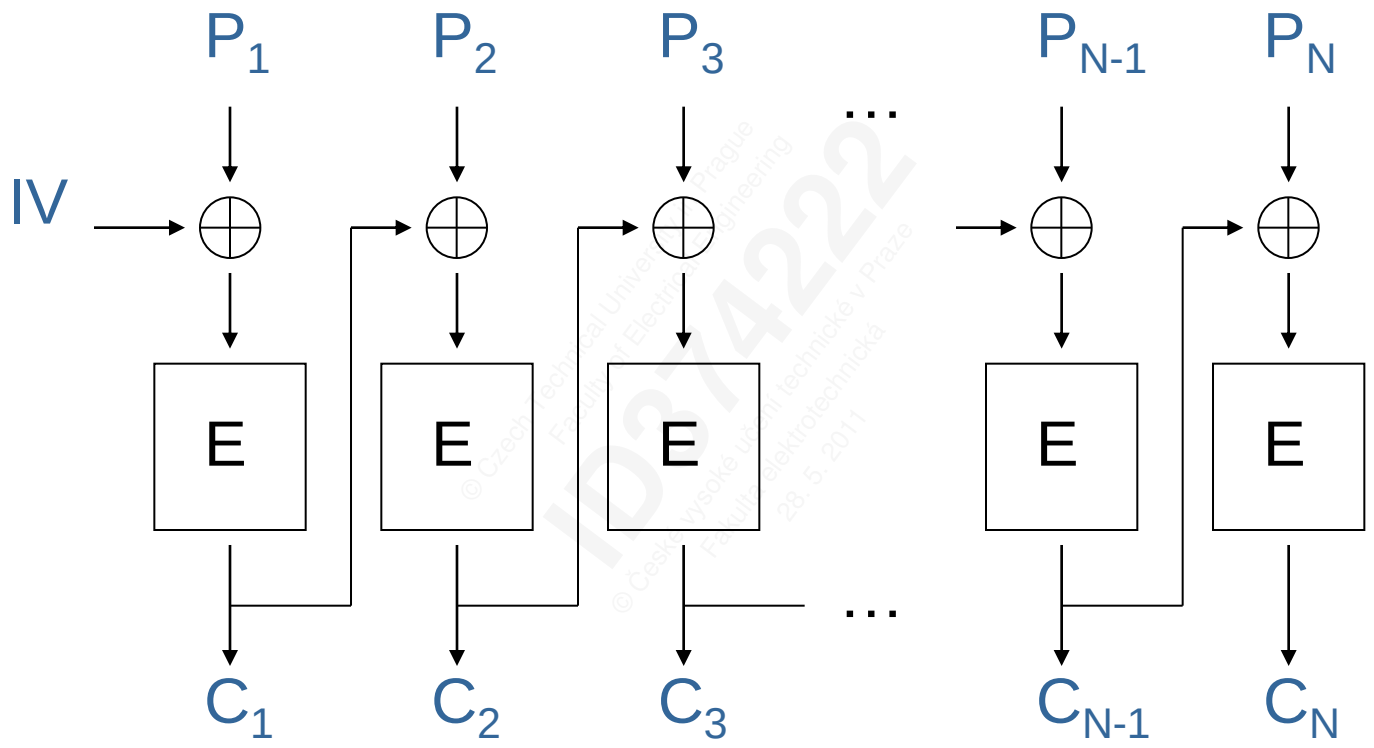
Doporučené vyplňování OT podle B. Schneiera

- nechť OT je otevřený text a ℓ(OT) je délka OT [B], a *b* je velikost bloku v [B]
- doporučený způsob doplňování č.1
  - za OT připojit jeden byte s hodnotou 128 $(10000000)_{b,}$
  - a poté za OT+1 přidat tolik nulových bajtů, až celková délka OT+výplně bude násobkem *b*
- doporučený způsob doplňování č.2
  - určit počet požadovaných výplňových bajtů *n*, kde $1 \leq n \geq b$ a $n+\ell(P)$ je násobek b
  - vyplnit OT n bajty, kde každý bude obsahovat hodnotu n
- lze použít libovolný jiný způsob, ale **musí** být **reversibilní**

CBC – Cipher Block Chaining

- CBC používá zpětnou vazbu, při níž je výsledek šifrovaní předchozího bloku zařazen do šifrovaní současného bloku.
- U CBC je OT nejprve XORován se ŠT předchozího bloku a teprve pak je zašifrován pro výstup. Blok ŠT je jednak odeslán a jednak zaznamenán pro použití na dalším bloku.
- Dešifrovaní je analogické. Zašifrovaný text je jednak dešifrován a jednak zaznamenán v registru. Poté co je následující blok dešifrován je provedeno na něm XOR s uloženým blokem.

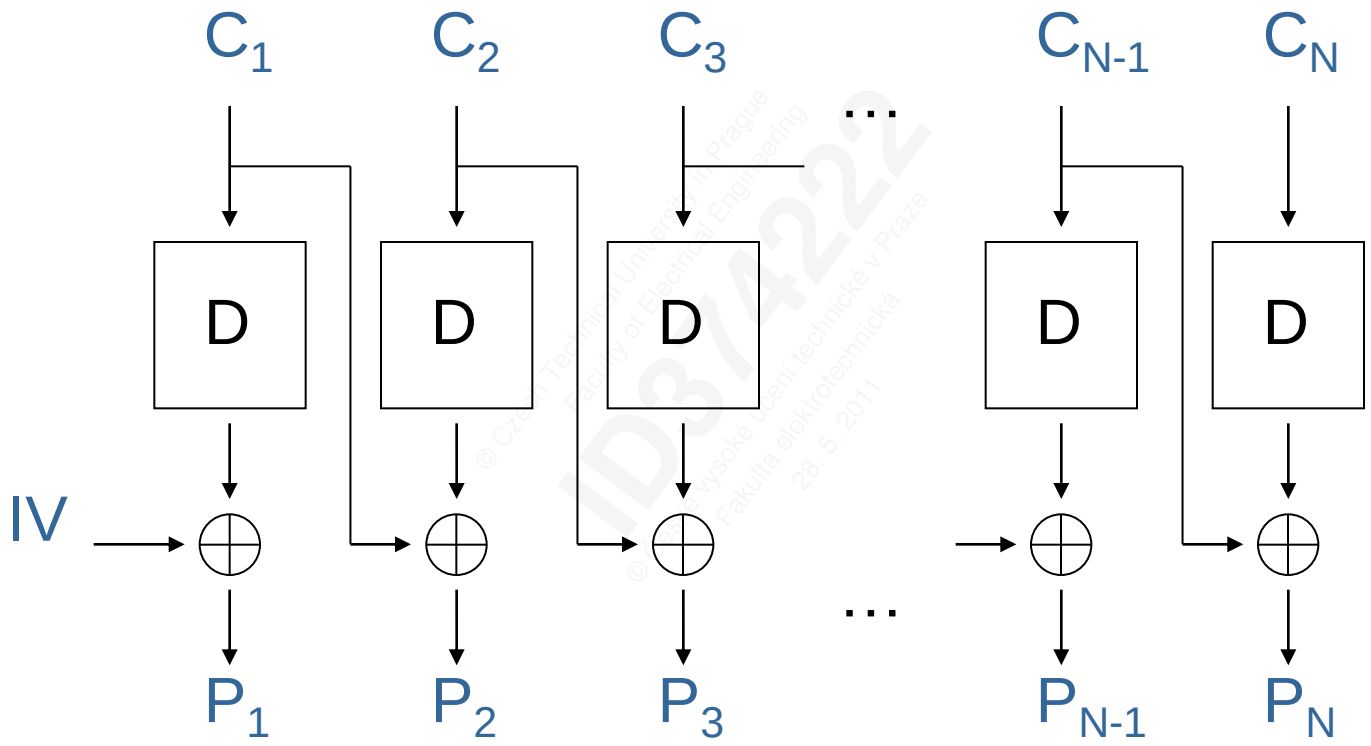
Režim CBC - šifrování



$C_1 = E(P_1 \oplus IV)$

$C_i = E(P_i \oplus C_{i-1})$ pro i=2..N

Režim CBC - dešifrování



$P_1 = D(C_1 \oplus IV)$

$P_i = D(C_i \oplus C_{i-1})$ pro i=2..N

Režim CBC

- bloky jsou „zřetězeny“ dohromady
- na začátku CBC je nutné použít inicializační vektor (IV)
- stejné bloky OT se šifrují na různé bloky ŠT
- cut-and-paste is stále možný, ale je složitější a zůstanou po něm „artefakty“
- pokud je během přenosu blok C_1 změněn na G , pak

  $P_1 \neq C_0 \oplus D(G,K)$, $P_2 \neq G \oplus D(C_2,K)$ ale , $P_3 = C_2 \oplus D(C_3,K)$, $P_4 = C_3 \oplus D(C_4,K)$, …
- dojde tedy k automatické obnově OT !

Šifrování

$C_0 = E(IV \oplus P_0,K)$,

$C_1 = E(C_0 \oplus P_1,K)$,

$C_2 = E(C_1 \oplus P_2,K)$,…

Dešifrování

$P_0 = IV \oplus D(C_0,K)$,

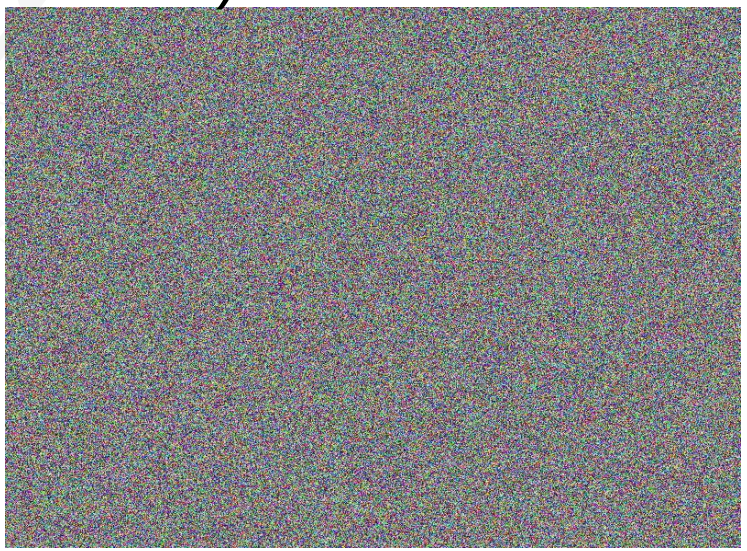
$P_1 = C_0 \oplus D(C_1,K)$,

$P_2 = C_1 \oplus D(C_2,K)$,…

Dobrá vlastnost CBC oproti ECB

- Obrázek Mickey Mouse

- Zašifrovaný obrázek
- režim CBC (algoritmus TEA)



Jak je to možné ?

Stejný blok OT $\Rightarrow$ různý blok ŠT

CBC - inicializační vektor (IV)

- Identické bloky OT se šifrují na různé ŠT pokud je předchozí blok OT_{i-1} různý od OT_i
- v prvním bloku je nutné přidat náhodná data nazývaná inicializační vektor (IV).
- IV zajistí aby dvě zprávy (OT_1 a OT_2) byly zašifrovány pokaždé jinak i pokud $OT_1=OT_2$
- IV není třeba držet v tajnosti ale **musí** být **změněn v každé zprávě**
  - **stejný IV a zároveň stejný OT = stejná situace jako u ECB**

CBC - šíření chyb

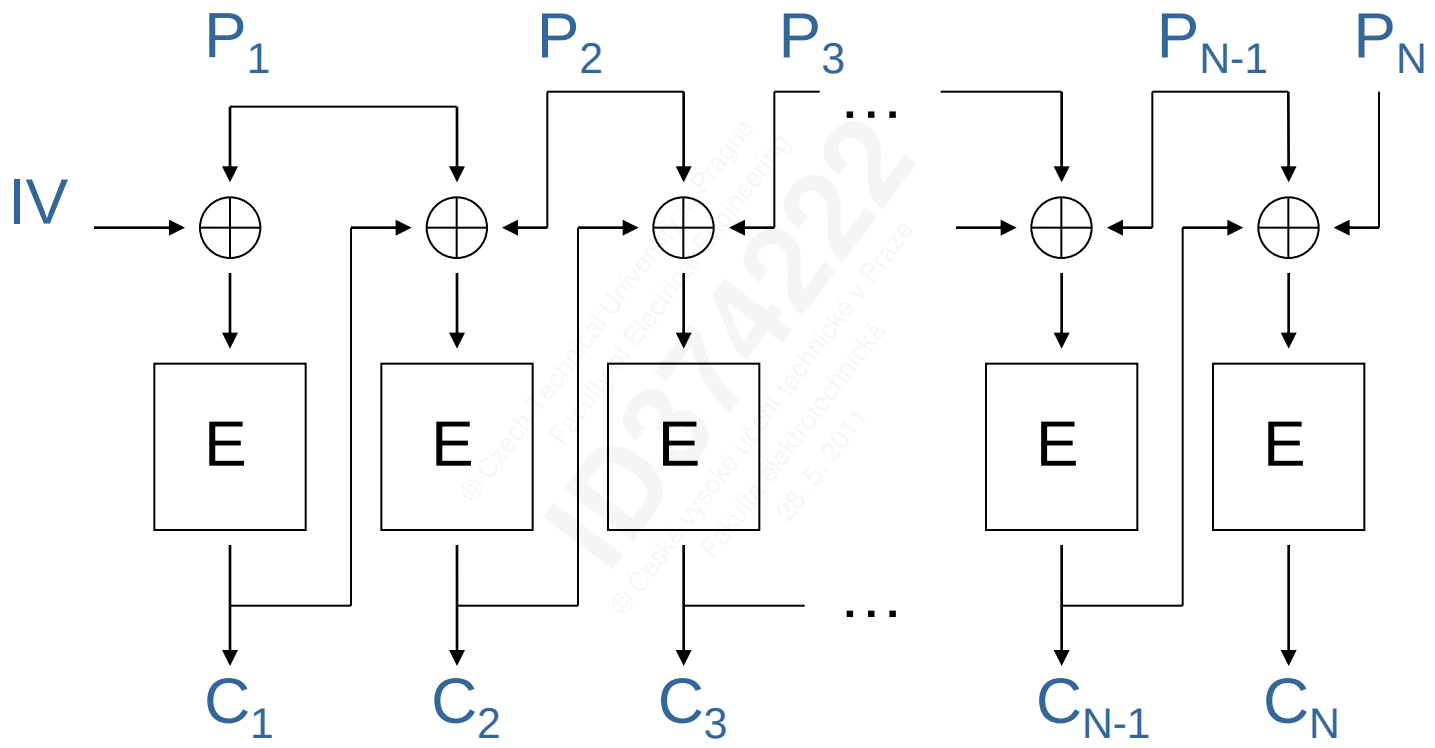
- změna jednoho bitu ŠT způsobí změnu jednoho bloku OT plus jednoho bitu v následujícím bloku OT ve stejné pozici v jaké se chyba vyskytla v ŠT.
- další bloky již nejsou ovlivněny
- CBC je samoobnovující (self-recovering) pro 1 bitovou změnu
- ztráta nebo vložení bitu vede ke ztrátě všech následujících bloků

Vlastnosti při niž se malá chyba v ŠT rozšíří na velkou chybu v OT, se říká rozšíření chyby (error-extension).

CBC - shrnutí

- vstup šifrovače (IV) - náhodný text
- šifrování neparalelizovatelné
- dešifrování paralelizovatelné, v libovolném pořadí
- snadná softwarové implementace
- vhodné pro šifrování souborů
- bezpečnější než ECB
- nejpoužívanější „klasický“ režim

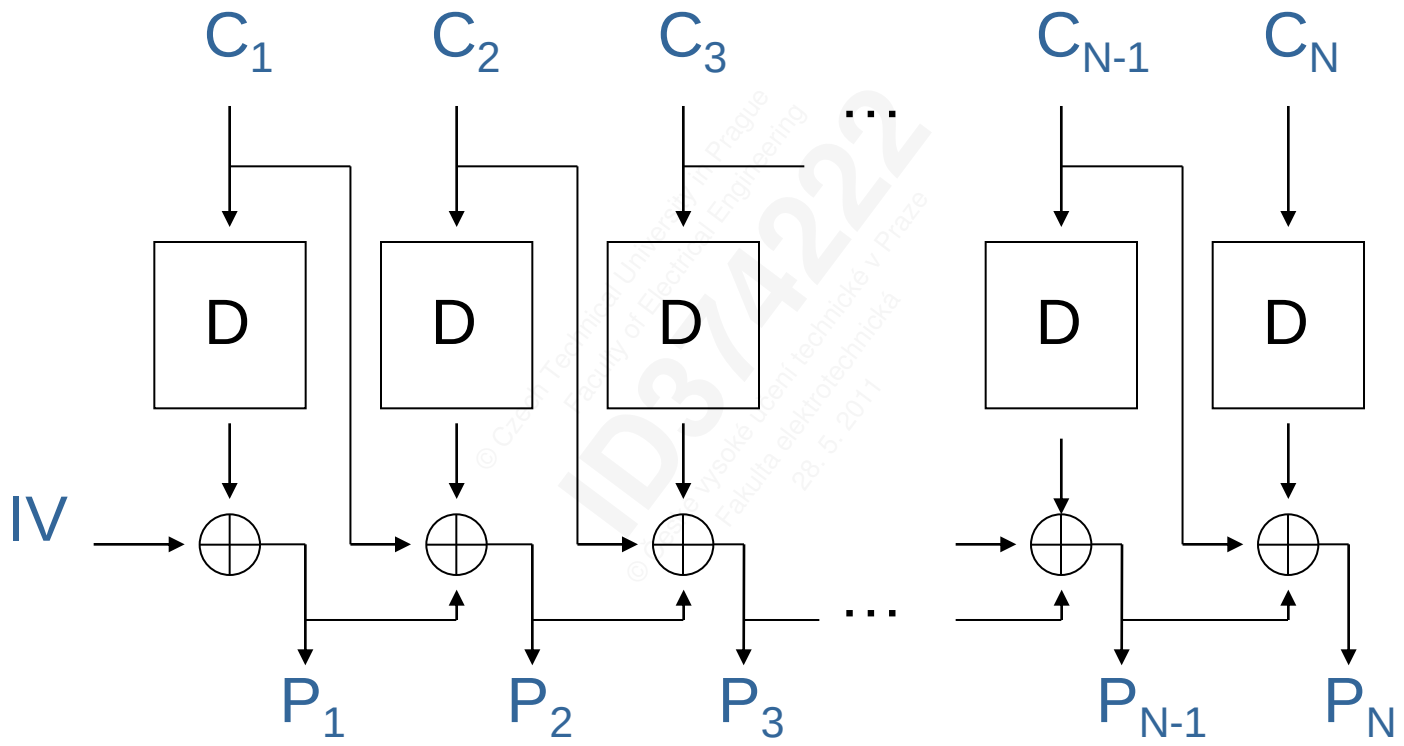
Režim PCBC - šifrování



$C_1 = E(P_1 \oplus IV)$

$C_i = E(P_i \oplus P_{i-1} \oplus C_{i-1})$ pro i=2..N

Režim PCBC - dešifrování



$P_1 = D(C_1) \oplus IV$

$P_i = D(C_i) \oplus C_{i-1} \oplus P_{i-1}$ pro i=2..N

PCBC - info

- PCBC – Propagating CBC (nebo Plaintext CBC)
- používaný u protokolu Kerberos v4
- není standardizován ani příliš rozšířen
- hlavním rozdílem oproti CBC je schopnost rozšíření (propagace) jednobitové chyby v bloku ŠT
- toto umožňuje zjistit chyby vnesené do komunikačního řetězce během přenosu a odmítnutí celého OT (i v případě modifikace pouze jednoho bitu)
- chyba se rozšíří na všechny následující bloky OT

Další režimy – CFB,OFB,CTR

- v režimech ECB, CBC a PCBC lze šifrovat pouze celé bloky

- v režimech CFB, OFB, CTR je možné šifrovat data i po menších částech, protože tyto transformují blokovou šifru na proudovou tím, že šifrovací transformaci E_k používají ke generování proudu klíče

CFB – Ciphertext Feedback Mode

- používá pouze šifrovací transformaci -> efektivní HW implementace
- není nutné přenášet celý blok ŠT do vstupního bloku, stačí pouze část *n* bitů → označuje se jako *n*-bitový CFB režim
- vstupní blok má nejvýše 2^N možných stavů, kde N je délka vstupního bloku v bitech. Po nejvýše 2^N blocích se tak začne proud klíče opakovat.
- vstup algoritmu = náhodný text (IV)
- šifrování neparalelizovatelné
- dešifrování paralelizovatelné, v libovolném pořadí
- šifrování znakových terminálů (8-bitový CFB)
- nejčastější varianta 8-bitový CFB, ale lze i jiné (1-bitový, 16 bitový…)

CFB – Ciphertext Feedback Mode

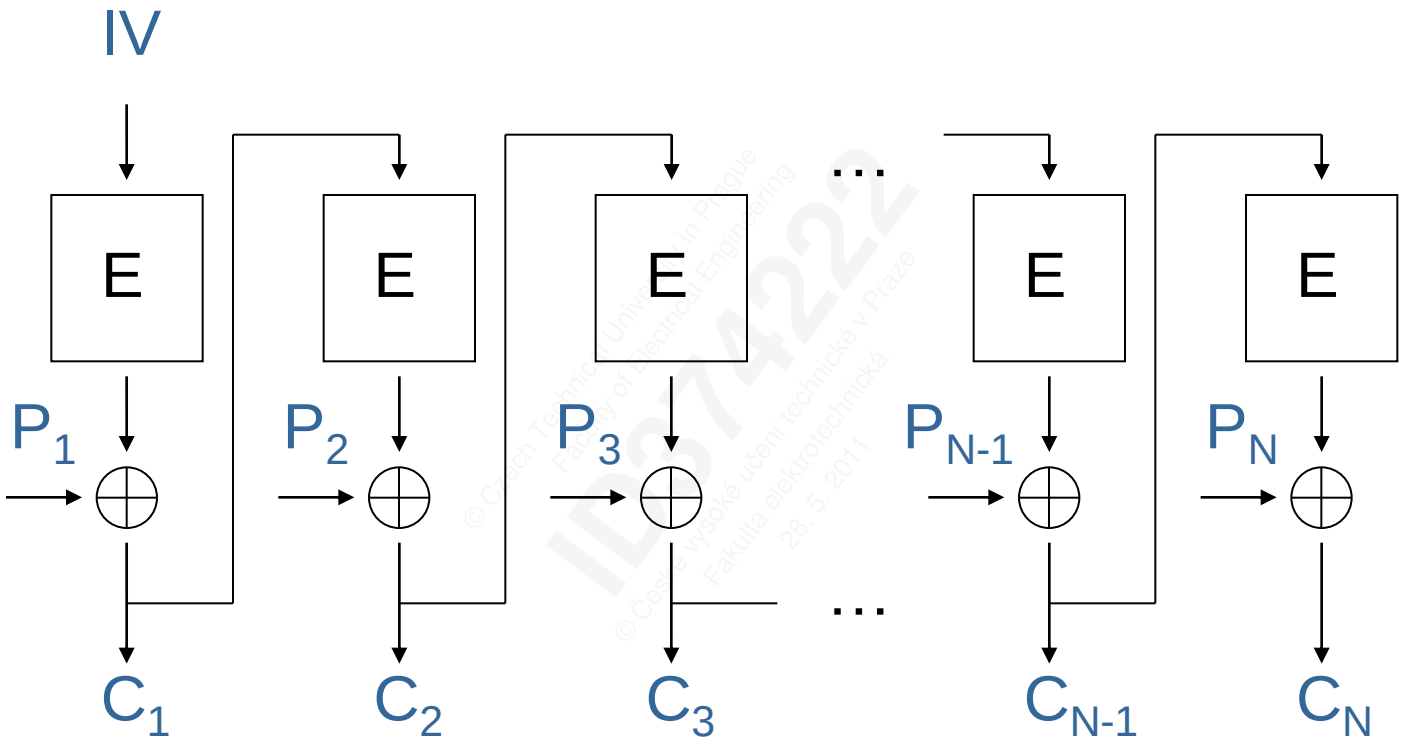
Šifrování:

- zvolí se náhodný IV (stejně jako u CBC)
- IV je zašifrován a jeho nejlevější byte je použit pro XOR s prvním bytem OT, získáme první byte ŠT .
- byte ŠT je odeslán komunikačním kanálem a po příchodu dalšího bytu OT se pokračuje v šifrování (druhý byte zašifrovaného IV XOR druhý byte OT)

Dešifrování:

- zvolí se náhodný IV (stejně jako u CBC)
- IV je zašifrován a jeho nejlevější byte je použit pro XOR s prvním bytem ŠT, získáme první byte OT
- pro inicializační vektor platí totéž, co CBC

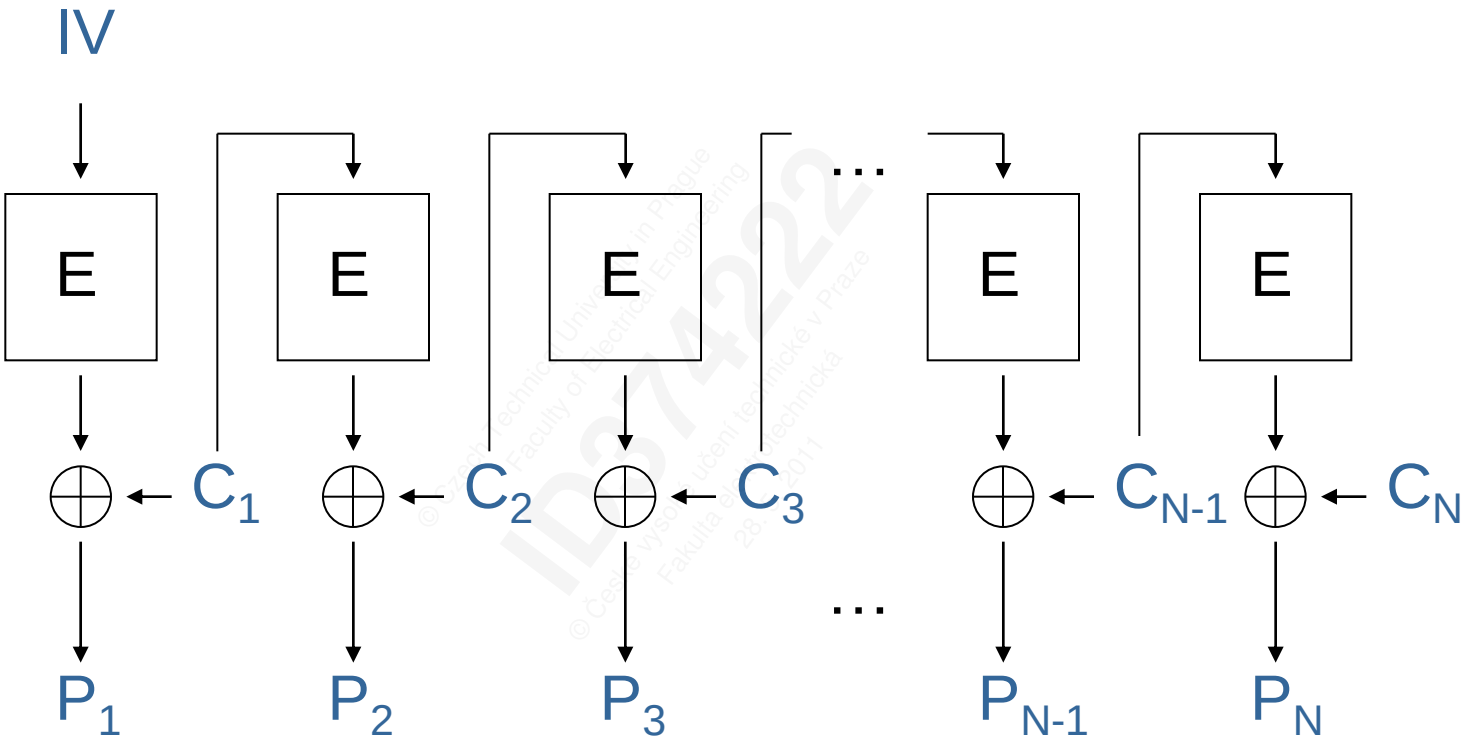
Režim CFB - šifrování



$C_1 = P_1 \oplus \mathrm{E}(IV)$

$C_i = P_i \oplus \mathrm{E}(C_{i-1})$ pro i=2..N

Režim CFB - šifrování



$P_1 = C_1 \oplus \mathrm{E}(IV)$

$P_i = C_i \oplus \mathrm{E}(C_{i-1})$ pro i=2..N

Režim CFB – šíření chyb

- CFB režim je samosynchronizující
- k obnovení správného OT stačí pouze dva po sobě jdoucí nenarušené bloky ŠT, resp. neporušené příslušné *n* bitové části ve dvou po sobě jdoucích blocích ŠT

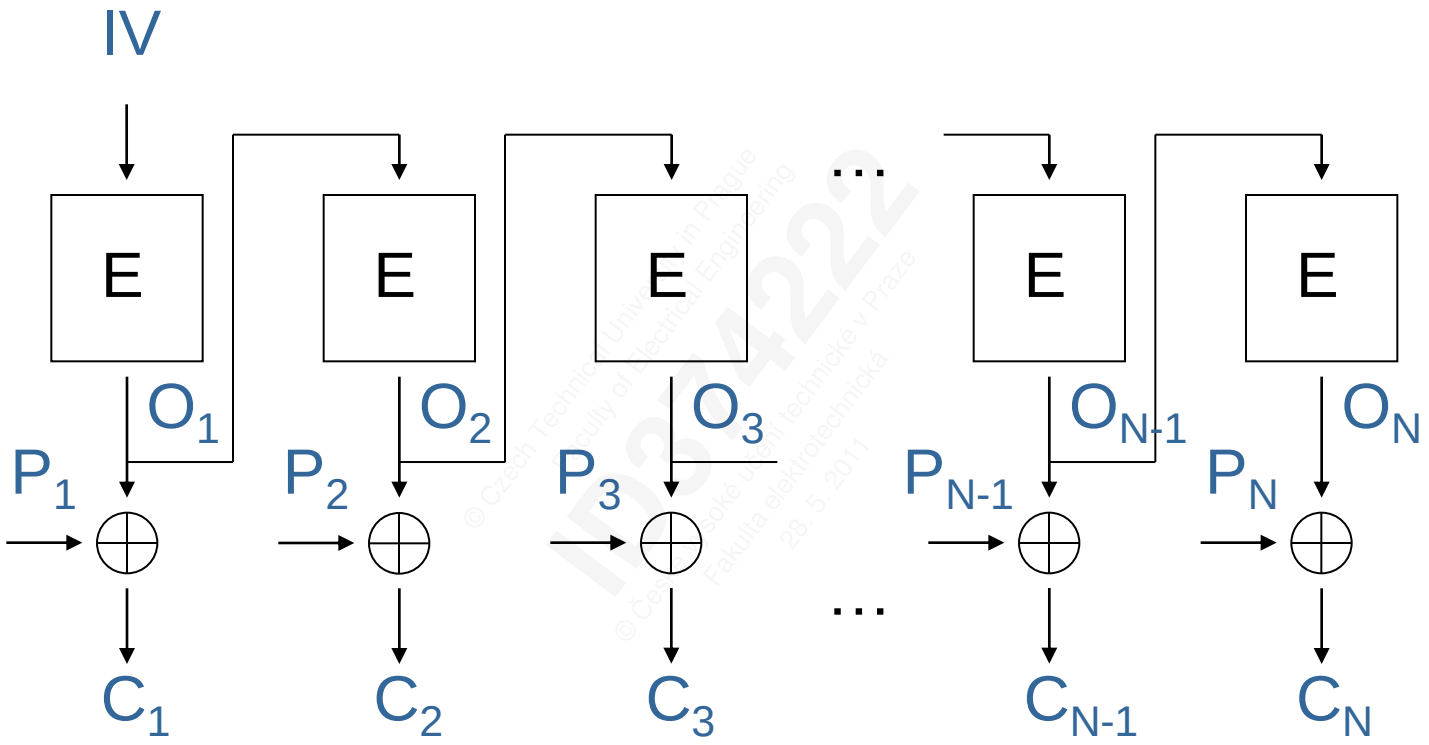
Šíření chyb u 8bitového CFB:

- jednobitová chyba v ŠT způsobí nejprve jednobitovou chybu v bloku OT
- chybný blok ŠT vstoupí do blokové šifry a následujících 8 bytů bude dešifrováno chybně
- další byty již budou dešifrovány dobře.
- CFB je samosynchronizující po 8B

Režim OFB (Output Feedback Mode)

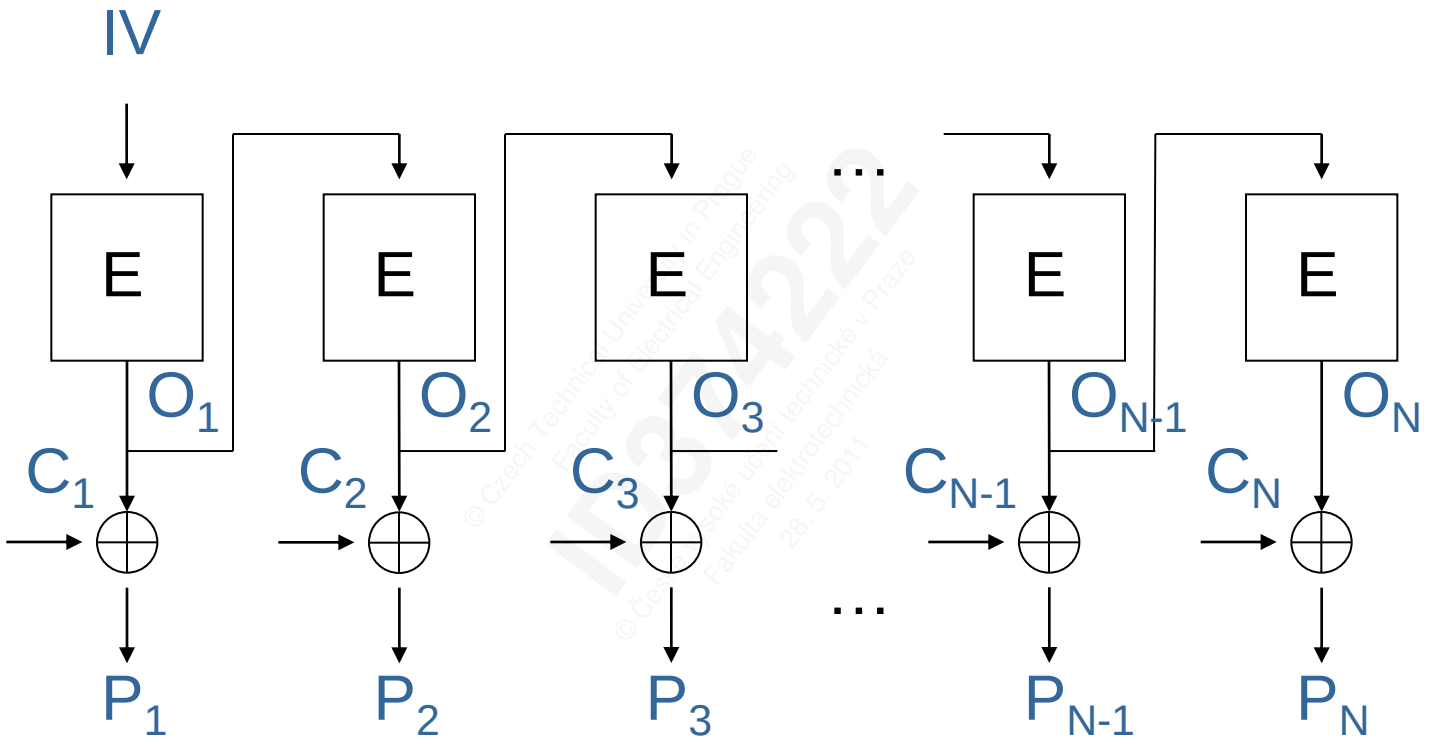
- používá pouze šifrovací transformaci -> efektivní HW implementace
- z výstupního bloku není nutné přenášet celý blok do vstupního bloku, stačí pouze část, nějakých *n* bitů.
- vstup algoritmu = náhodný text
- šifrování a dešifrování neparalelizovatelné
- vhodný pro vysokorychlostní systémy s nepřípustným šířením chyb
- 1-bitová chyba v šifře ovlivní jeden bit po dekódování
- OFB je čistě synchronní proudová šifra, proud klíče není ovlivňován ani otevřeným ani šifrovým textem
- vstupní blok má nejvýše 2^N možných stavů, kde N je délka vstupního bloku v bitech. Po nejvýše 2^N blocích se tak začne proud klíče opakovat

Režim OFB - šifrování



$C_1 = P_1 \oplus E(IV)$ $C_i = P_i \oplus O_i$ $O_i = E(O_{i-1})$ pro i=2..N

Režim OFB - dešifrování



$P_1 = C_1 \oplus E(IV)$ $P_i = C_i \oplus O_i$ $O_i = E(O_{i-1})$ pro i=2..N

Režim CTR (Counter - čítač)

- používá blokovou šifru jako proudovou, podobný OFB
- v proudu klíče se nemusí používat celý výstupní blok, ale jenom nějaká jeho část
- CTR se používá v případě potřeby náhodného přístupu (mohu šifrovat/dešifrovat libovolný blok bez ohledu na ostatní)
- při šifrování dvou různých zpráv se stejným klíčem se nesmí vygenerovat stejná část proudu klíče

Šifrování

$C_0 = P_0 \oplus E(IV, K)$,

$C_1 = P_1 \oplus E(IV+1, K)$,

$C_2 = P_2 \oplus E(IV+2, K)$,…

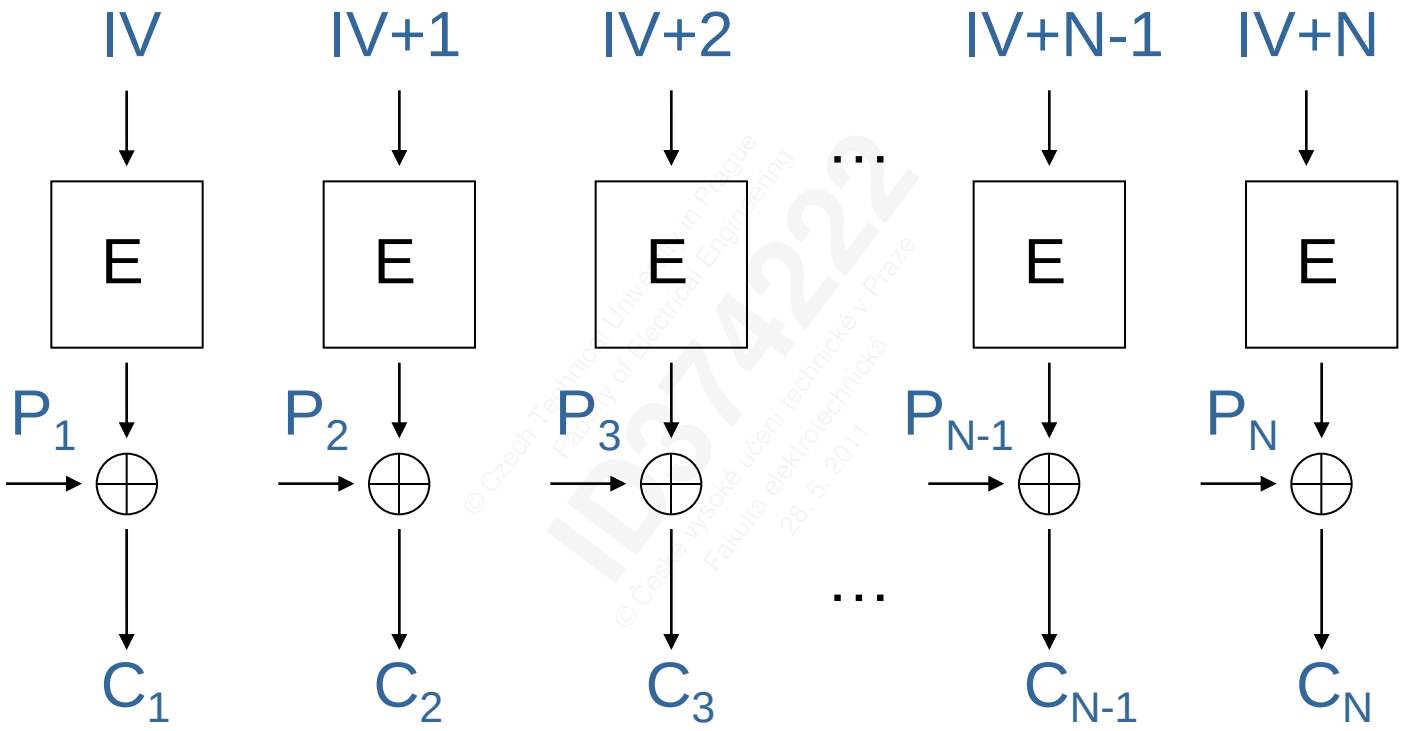
Dešifrování

$P_0 = C_0 \oplus E(IV, K)$,

$P_1 = C_1 \oplus E(IV+1, K)$,

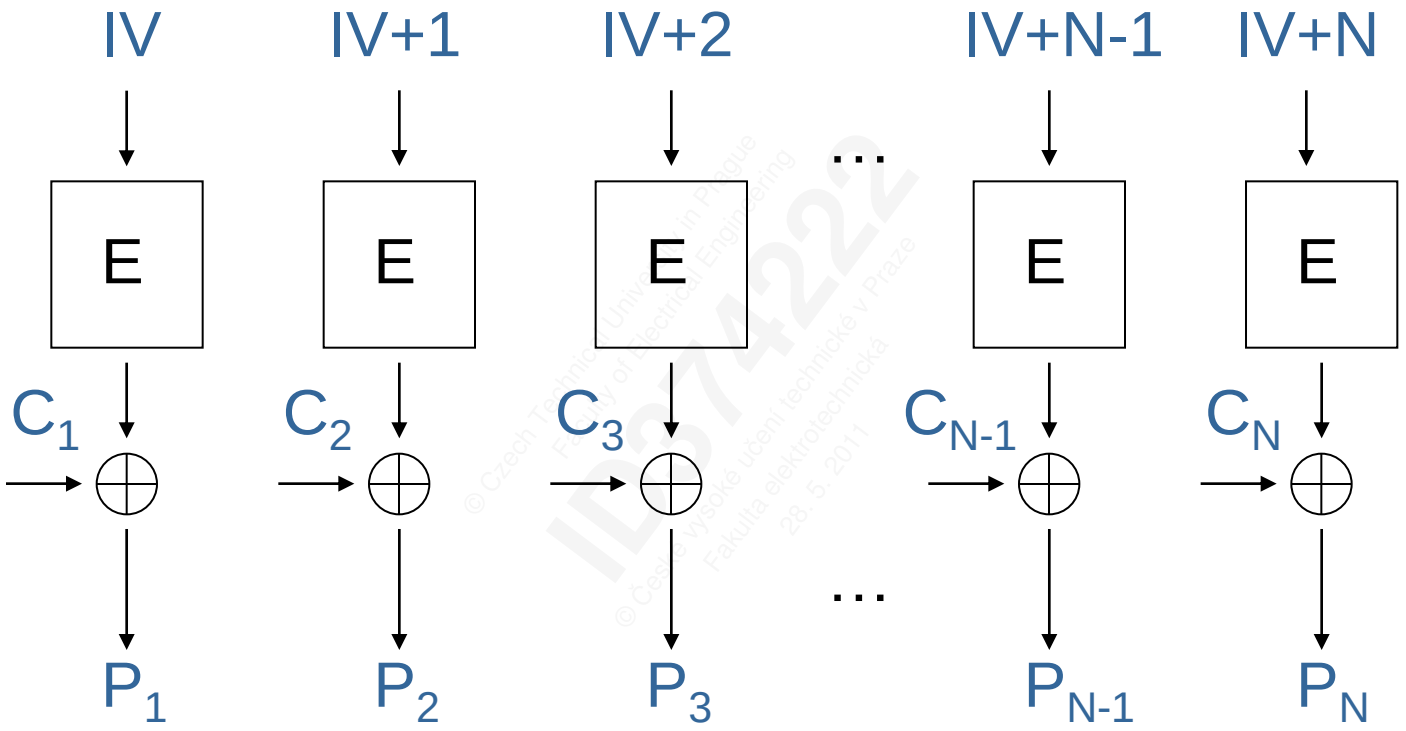
$P_2 = C_2 \oplus E(IV+2, K)$,…

Režim CTR - šifrování



$C_i = P_i \oplus E(IV+i)$ pro i=0..N

Režim CTR - dešifrování



$P_i = C_i \oplus E(IV+i)$ pro i=0..N

Solení (salting) IV

- podobné jako bílení klíčů u blokových šifer
- lze použít tam, kde se posílá IV
  - CBC, OFB, CFB a CTR
- zajistí utajení IV
  - není nutné
- *IV* se pošle v otevřeném tvaru, ale nepoužije se k šifrování, ale je použit k vygenerování jiného *IV'*
- *IV'* je použit v algoritmu
- hodnota *IV'* se nepřenáší komunikačním kanálem

Útok na CFB, OFB a CTR

- změna *ŠT* v CFB režimu z *ŠT* na *ŠT* ⊕ *X* se do *OT* projeví jako *OT* ⊕ *X*
- analogická změna se stane u OFB/CTR, když proud klíče *O* změním na *O* ⊕ *d*
- útok funguje i na synchronní proudové šifry

OT	… a celková cena činí 1.000.000€ bez DPH
proud_klíče	…sdmnfriwerxcvjkqweruiytrbhzcjkqrizujlowet
ŠT	…jvneurlpzndhtrpqznab**w**eopgrmfjqwhjuernx

9 ⊕ w = j

ŠT	…jvneurlpzndhtrpqznab**j**abpgrmfjqwhjuernx
proud_klíče	…sdmnfriwerxcvjkqweruiytrbhzcjkqrizujlowet
OT	… a celková cena činí 1.900.000€ bez DPH

ŠT ⊕X ⊕ proud_klíče = (OT ⊕X ⊕ proud_klíče) ⊕ proud_klíče =OT ⊕X

Speciální režimy činnosti

- CBC (nejpoužívanější režim) má několik bezpečnostních problémů, kvůli kterým se je jeho použití někdy nevhodné
  - typicky šifrování disků
- např. pokud je IV generován předvídatelně, může útočník vygenerovat soubor se speciálním obsahem, který vyruší vliv IV
- po uložení tohoto souboru na disk je na něm čitelný „otisk“, tj. lze detekovat, že na disku je tento soubor přítomen (i když je zašifrován!)

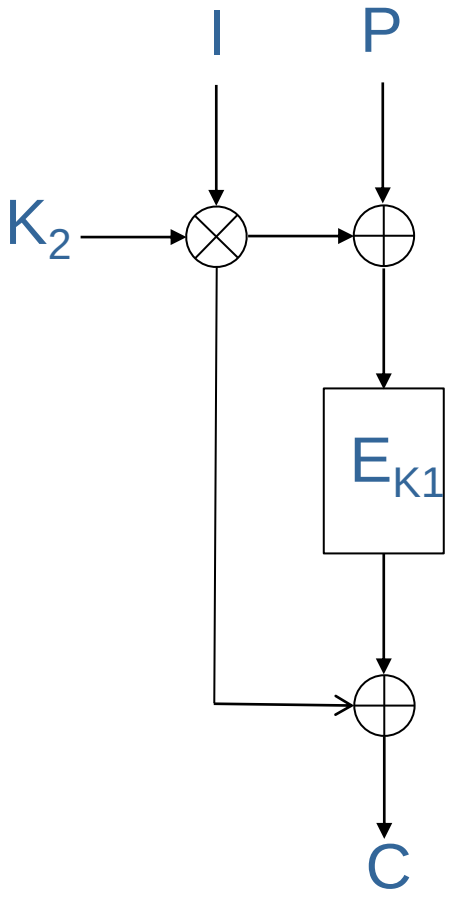
- aby to nebylo možné, je nutné generovat IV nepředvídatelně – pomocí nějakého šifrovacího algoritmu nebo hashovací funkce

Speciální režimy činnosti - LRW

- využívané při šifrování pevných disků

LRW - Liskov, Rivest, Wagner

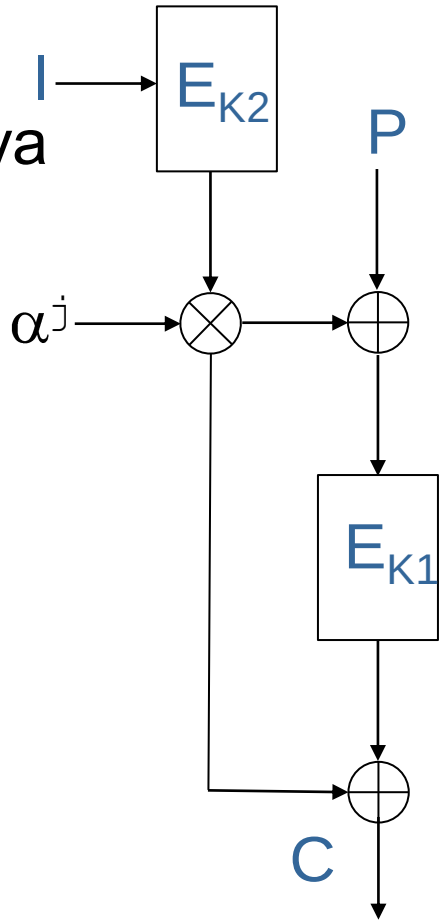
- $C=E_{K1}[P \oplus X] \oplus X$, kde $X=K_2 \otimes I$
- dva klíče
  - K_1 – délka závisí na algoritmu
  - K_2 – pevná délka
- operace se odehrávají v poli GF(2^{128})
  - sčítání = XOR
- dnes spíše nahrazen režimem XTS
- používal ho Truecrypt
- podporuje ho dmcrypt, FreeOTFE,



Speciální režimy činnosti - XEX

XEX - Xor-Encryption-Xor

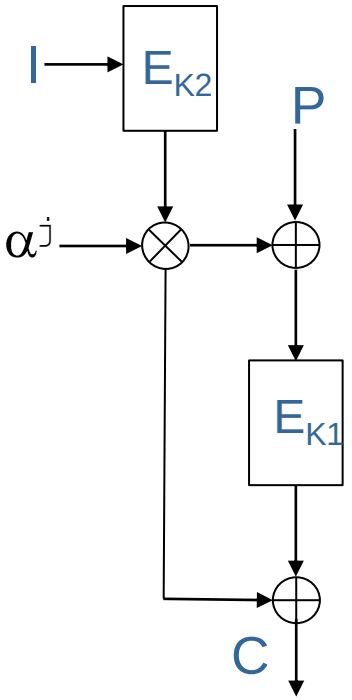
- $C=E_{K1}[P \oplus X] \oplus X$, kde $X=E_{K2}(I) \otimes \alpha^j$
- $K=K_1||K_2$ – **pouze jeden klíč**, rozdělený na dva
- efektivní pro šifrování po sobě jdoucích bloků
- I - číslo sektoru
- *j* - číslo bloku v sektoru
- α generátor pole $GF(2^{128})$



Speciální režimy činnosti - XTS

XTS - **X**EX-**T**BC-CT**S**

- XEX-based Tweaked CodeBook mode with CipherText Stealing
- CTS (CipherText Stealing) – metoda doplnění bloku pokud sektor není dělitelný velikostí bloku šifrovacího algoritmu
- velmi podobný XEX
- $C=E_{K1}[P \oplus X] \oplus X$, kde $X=E_{K2}(I) \otimes \alpha^j$
- I - číslo sektoru
- *j* - číslo bloku v sektoru
- α generátor pole $F(2^{128})$
- **dva nezávislé klíče K_1, K_2**
- 12/2007 IEEE P1619
- leden 2010 - NIST SP800-38E – XTS-AES
- Bestcrypto, Truecrypt, dm-crypt, FreeOTFE



Speciální režimy činnosti - ESSIV

ESSIV - Encrypted Salt-Sector Initialization Vector

- $IV_{sektor} = E_{hash(K)}(sektor)$
- pseudonáhodné generování IV pro šifrování jednotlivých sektorů
- FreeOTFE, dm-crypt
- používá se v kombinaci s CBC (CBC-ESSIV)

Další režimy činnosti

IACBC - Integrity Aware CBC
IAPM - Integrity-Aware Parallelizable Mode
XCBC - Extended Cipher Block Chaining
OCB - Offset Codebook Mode
ABC - Accumulated Block Chaining
DCTR - Dual Counter Mode
CWC - Carter-Wegman Counter Mode
WHF - Whiting-Housley-Ferguson Mode
…

Dotazy