Blokk kódoló üzemmódok

Smidla József

Rendszer- és Számítástudományi Tanszék
Pannon Egyetem

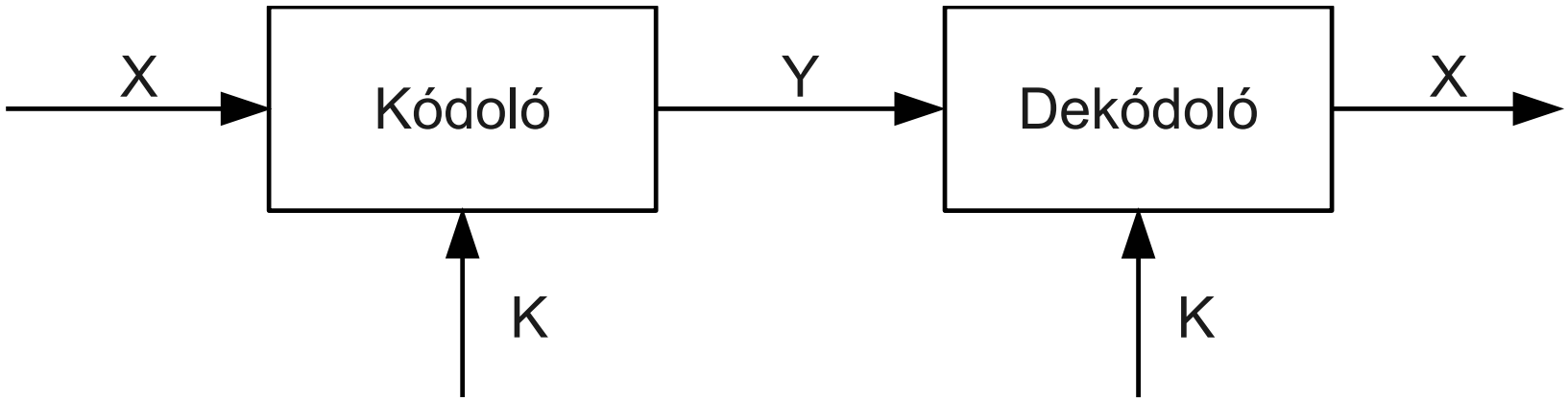
2012. 3. 6.

Tartalom

Elektronikus kódkönyv

Electronic Code Book mode (EBC)

- X: kódolatlan szöveg
- Y: kódolt szöveg
- K: kulcs
- $Y = E_k(X)$
- $X = D_k(Y)$
- Bemenet feldarabolása blokkokra
- Blokkokat egymástól függetlenül kódoljuk ugyanazzal a kulccsal



Banki átutalás

Az A bankból pénzt utalunk B bankba, az utalást az alábbi üzenettel végezzük el:

1	2	3	4	5
Küldő bank (A)	Küldő számla	Fogadó bank (B)	Fogadó számla	Pénz összeg

Tegyük fel, hogy

- A blokkok egyenként 16 bájtosak, és AES-el kódoljuk őket
- A bankok egy hétig minden üzenethez ugyanazt a 256 bites AES kulcsot használják
- Mi viszont piszok gyorsan meg szeretnénk gazdagodni

Az egyes blokkokat nem tudjuk visszafejteni, ám erre nem is lesz szükség...

Banki átutalás

1. Bankszámlát nyitunk A és B banknál

Banki átutalás

1. Bankszámlát nyitunk A és B banknál
2. Többször 1 $-os összegeket utalunk az A bankbeli számlánkról a B-be

Banki átutalás

1. Bankszámlát nyitunk A és B banknál
2. Többször 1 $-os összegeket utalunk az A bankbeli számlánkról a B-be
3. Észrevesszük, hogy 5 blokk van

Banki átutalás

1. Bankszámlát nyitunk A és B banknál
2. Többször 1 $-os összegeket utalunk az A bankbeli számlánkról a B-be
3. Észrevesszük, hogy 5 blokk van
4. Lementjük az 1., 3. és 4. blokk tartalmát

Banki átutalás

1. Bankszámlát nyitunk A és B banknál
2. Többször 1 $-os összegeket utalunk az A bankbeli számlánkról a B-be
3. Észrevesszük, hogy 5 blokk van
4. Lementjük az 1., 3. és 4. blokk tartalmát
5. Elfogunk minden más üzenetet

Banki átutalás

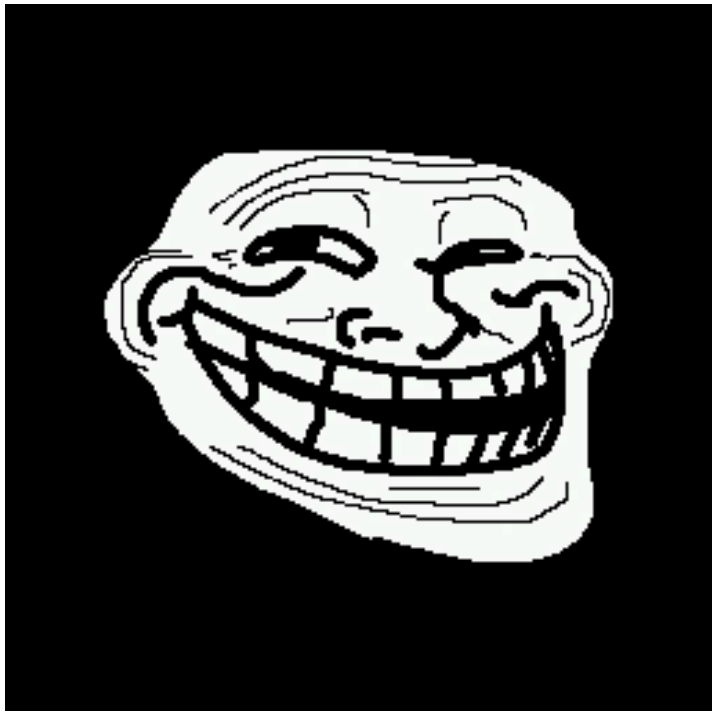
1. Bankszámlát nyitunk A és B banknál
2. Többször 1 $-os összegeket utalunk az A bankbeli számlánkról a B-be
3. Észrevesszük, hogy 5 blokk van
4. Lementjük az 1., 3. és 4. blokk tartalmát
5. Elfogunk minden más üzenetet
6. Ha az elfogott üzenet 1. és 3. blokkja megegyezik azzal, amit mi eltároltunk, akkor kicseréljük a 4. blokkot a lementett blokkra, majd az üzenetet tovább küldjük

Banki átutalás

1. Bankszámlát nyitunk A és B banknál
2. Többször 1 $-os összegeket utalunk az A bankbeli számlánkról a B-be
3. Észrevesszük, hogy 5 blokk van
4. Lementjük az 1., 3. és 4. blokk tartalmát
5. Elfogunk minden más üzenetet
6. Ha az elfogott üzenet 1. és 3. blokkja megegyezik azzal, amit mi eltároltunk, akkor kicseréljük a 4. blokkot a lementett blokkra, majd az üzenetet tovább küldjük
7. Ha elég gazdagok leszünk, meneküljünk egy olyan országba, ahonnan nem adnak ki minket

Kép kódolása EBC módban

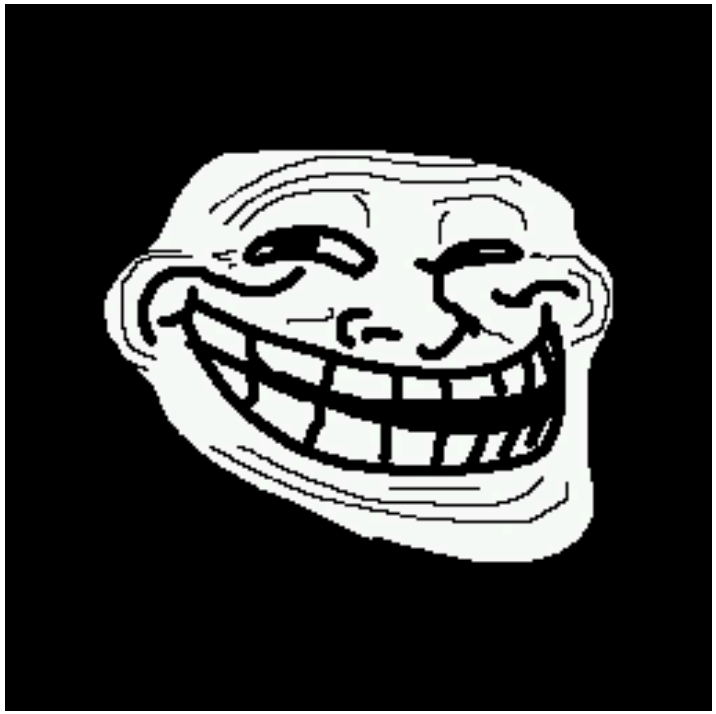
Eredeti kép:



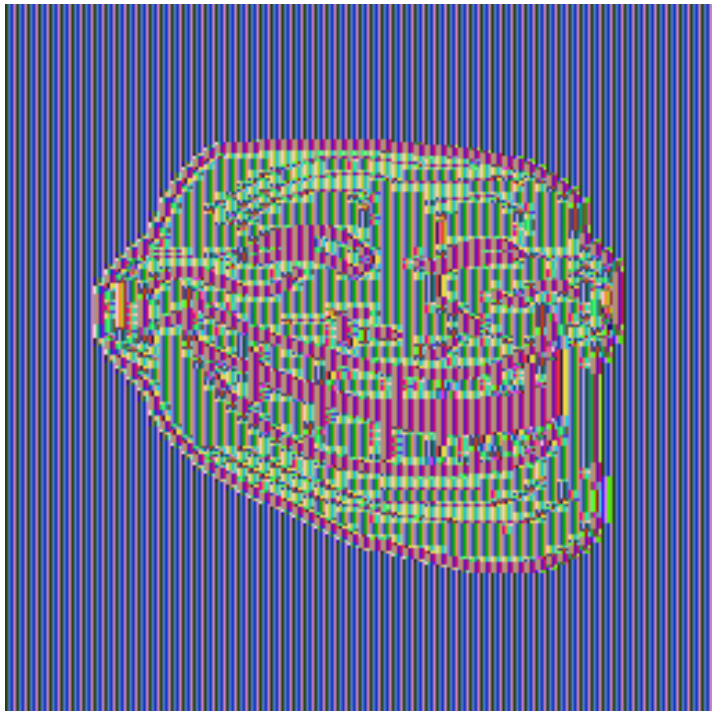
AES-el kódolt kép:

Kép kódolása EBC módban

Eredeti kép:



AES-el kódolt kép:



Előnyök és hátrányok

Előnyök

- Egyszerű
- Nagy adatmennyiség kódolása egyszerűen párhuzamosítható
- Ha az üzenet átvitele során egy blokk sérül, akkor a dekódoláskor is csak egy blokkot kapunk vissza hibásan

Előnyök és hátrányok

Előnyök

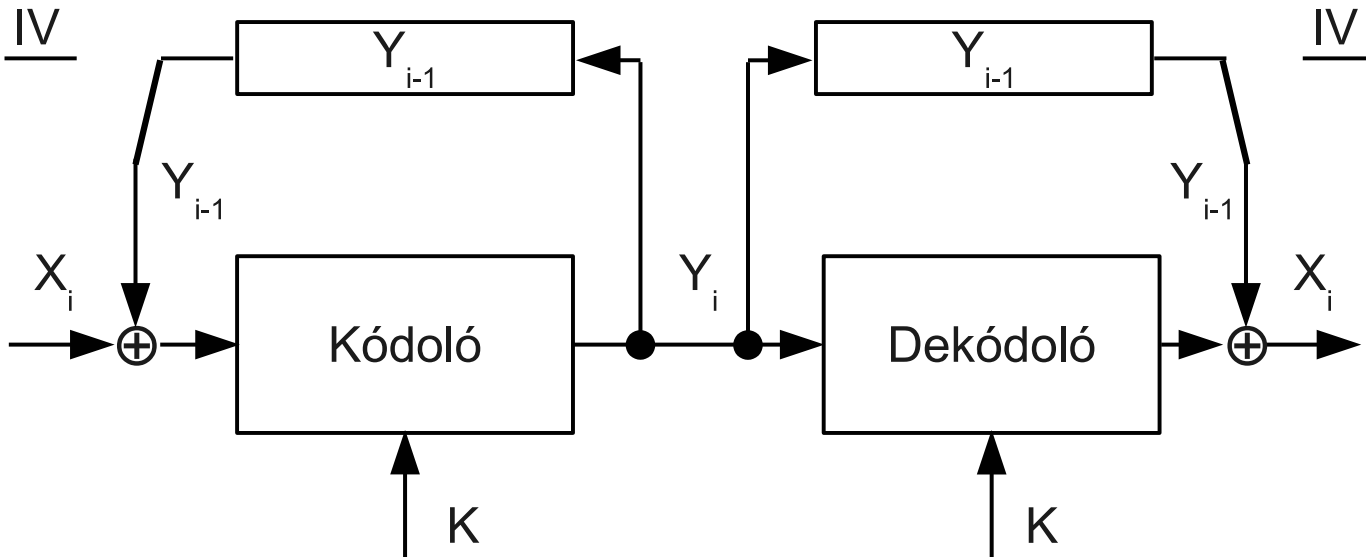
- Egyszerű
- Nagy adatmennyiség kódolása egyszerűen párhuzamosítható
- Ha az üzenet átvitele során egy blokk sérül, akkor a dekódoláskor is csak egy blokkot kapunk vissza hibásan

Hátrányok

- DETERMINISZTIKUS!
- Lényegében egy behelyettesítő kódolót kapunk, amiről tudjuk, hogy gyenge

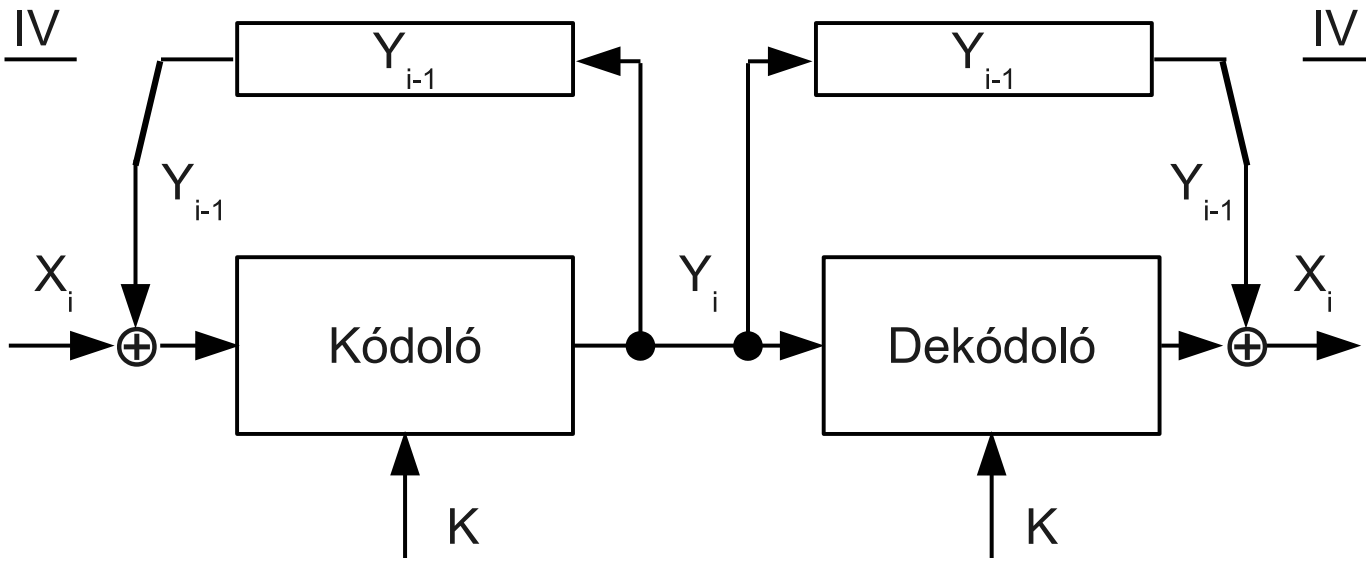
Titkosított blokkok láncolása

Cipher Block Chaining mode (CBC)



Titkosított blokkok láncolása

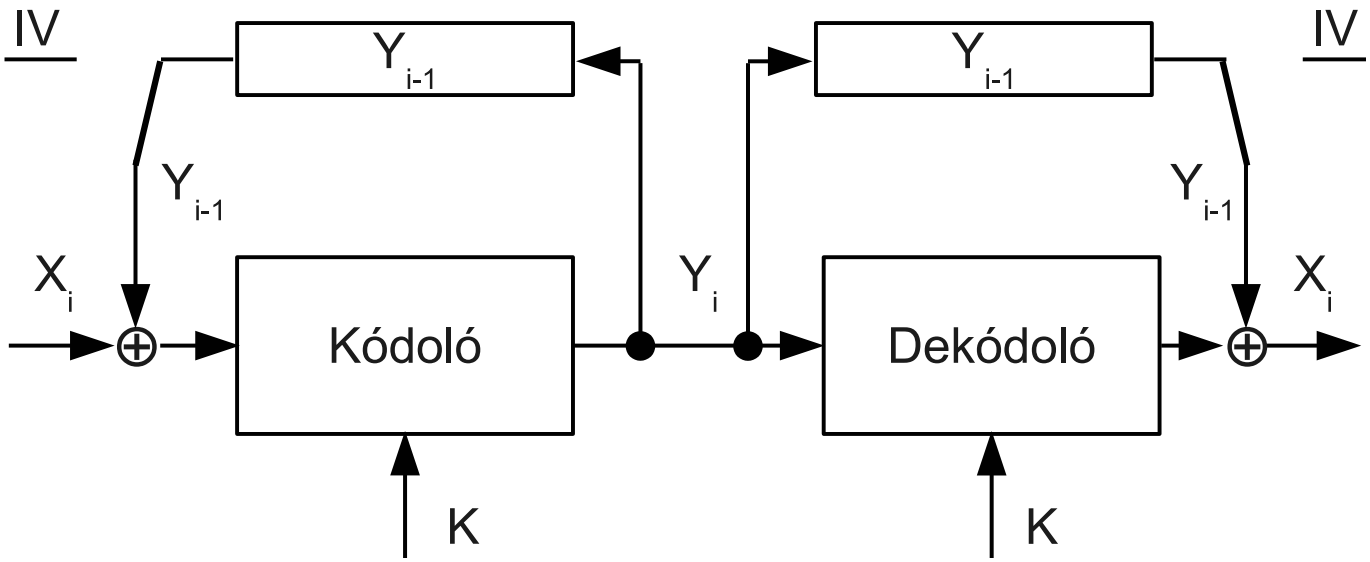
Cipher Block Chaining mode (CBC)



- IV: Inicializáló vektor, mindkét oldalon azonos, véletlenszerűen adjuk meg

Titkosított blokkok láncolása

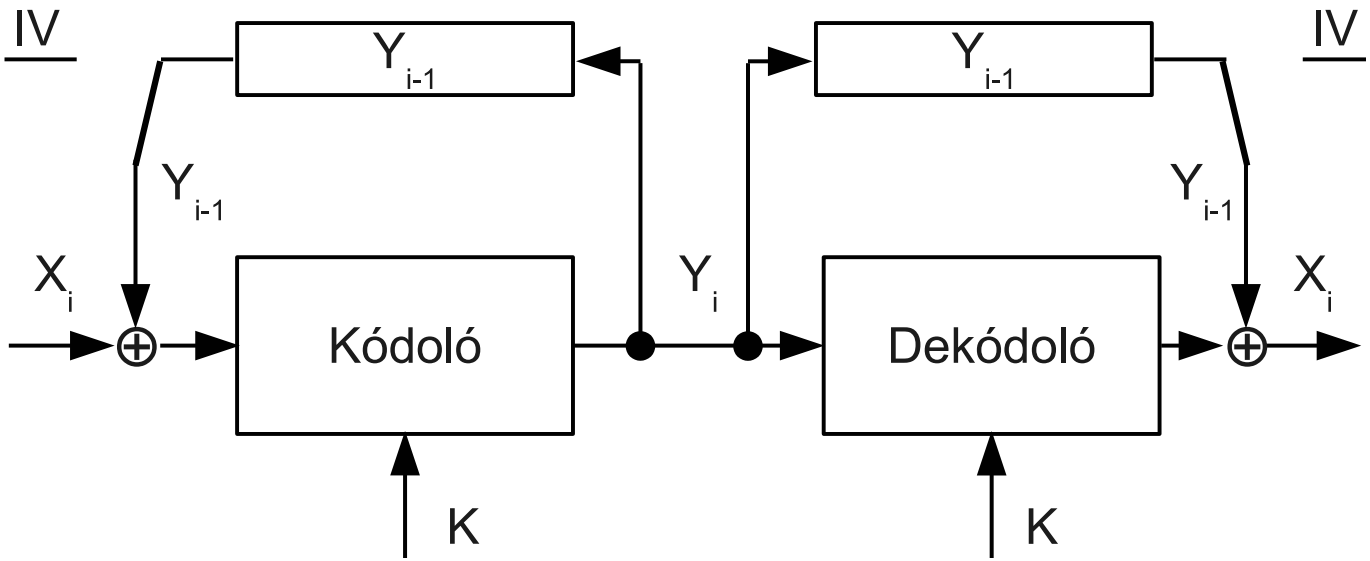
Cipher Block Chaining mode (CBC)



- IV: Inicializáló vektor, mindkét oldalon azonos, véletlenszerűen adjuk meg
- Első blokk: $Y_1 = E_k(IV \oplus X_1) \xrightarrow{\text{dekódolás}} X_1 = D_k(Y_1) \oplus IV$

Titkosított blokkok láncolása

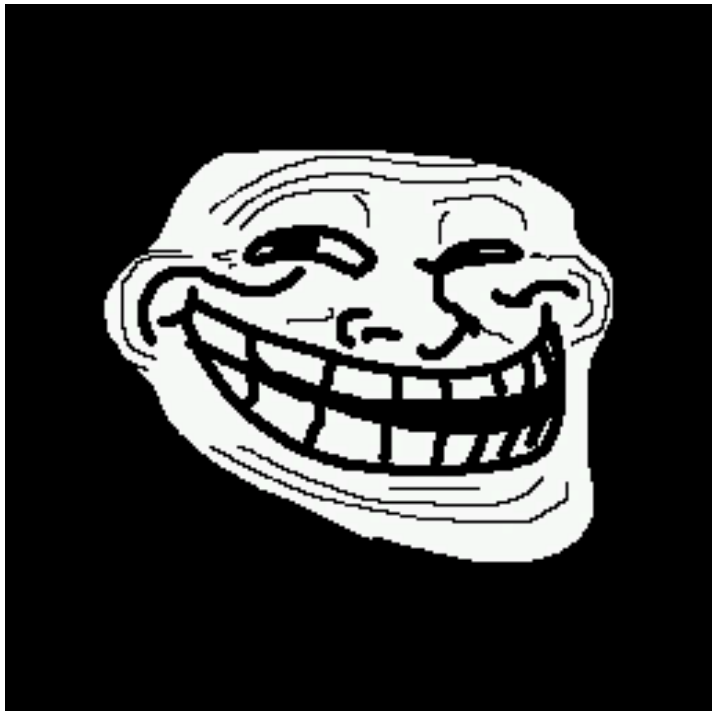
Cipher Block Chaining mode (CBC)



- IV: Inicializáló vektor, mindkét oldalon azonos, véletlenszerűen adjuk meg
- Első blokk: $Y_1 = E_k(IV \oplus X_1) \xrightarrow{\text{dekódolás}} X_1 = D_k(Y_1) \oplus IV$
- Következő blokkok:
  $Y_i = E_k(Y_{i-1} \oplus X_i) \xrightarrow{\text{dekódolás}} X_i = D_k(Y_i) \oplus Y_{i-1}$

Kép kódolása CBC módban

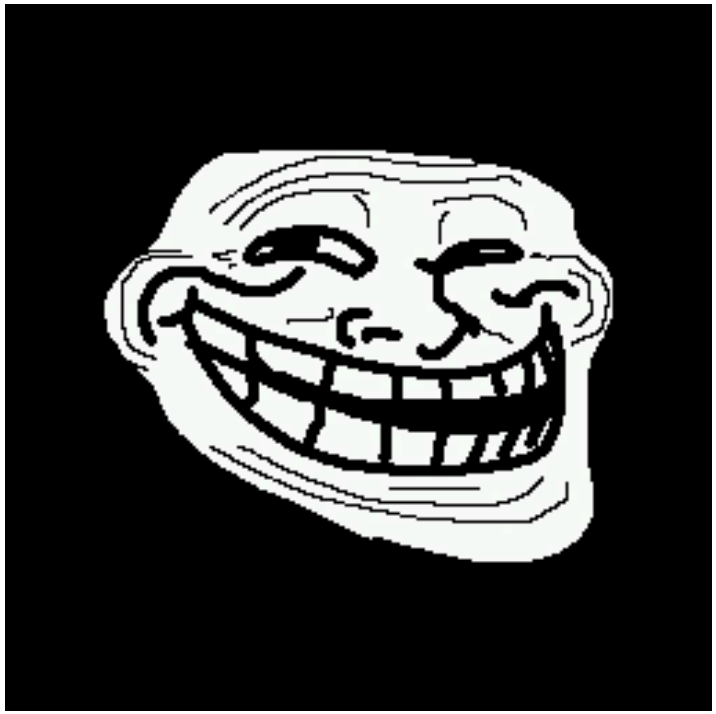
Eredeti kép:



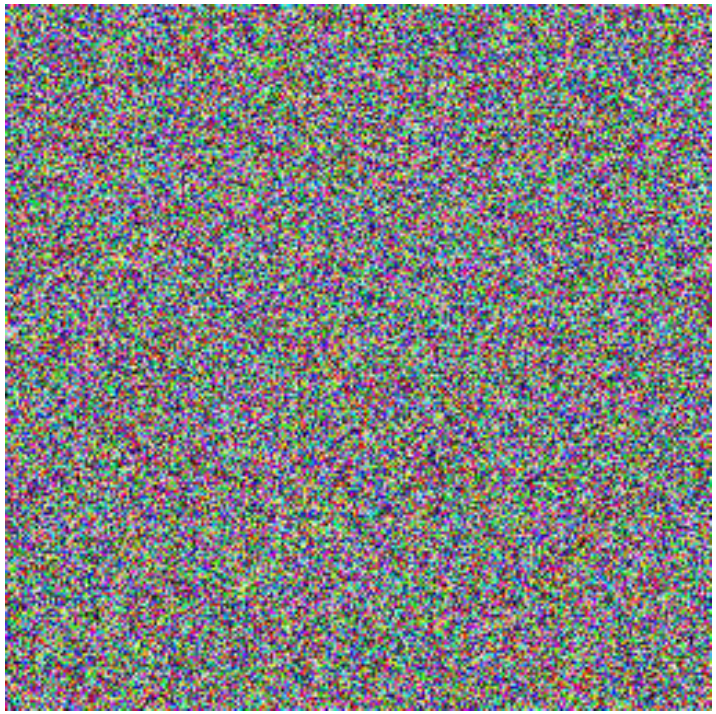
AES-el kódolt kép:

Kép kódolása CBC módban

Eredeti kép:



AES-el kódolt kép:



Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben

Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben
- Ha ugyanazt az IV-t használják, akkor:

Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben
- Ha ugyanazt az IV-t használják, akkor:
  1. Elfogjuk az üzeneteket

Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben
- Ha ugyanazt az IV-t használják, akkor:
  1. Elfogjuk az üzeneteket
  2. Kicseréljük a 4. blokkot (azaz az azonosítót), majd az üzenetet tovább küldjük

Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben
- Ha ugyanazt az IV-t használják, akkor:
  1. Elfogjuk az üzeneteket
  2. Kicseréljük a 4. blokkot (azaz az azonosítót), majd az üzenetet tovább küldjük
  3. A vevő oldalon visszafejtik az üzenetet

Titkosított blokk láncolása

- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben
- Ha ugyanazt az IV-t használják, akkor:
  1. Elfogjuk az üzeneteket
  2. Kicseréljük a 4. blokkot (azaz az azonosítót), majd az üzenetet tovább küldjük
  3. A vevő oldalon visszafejtik az üzenetet
  4. A 4. és 5. blokk véletlenszerű értéket vesz fel

Titkosított blokk láncolása

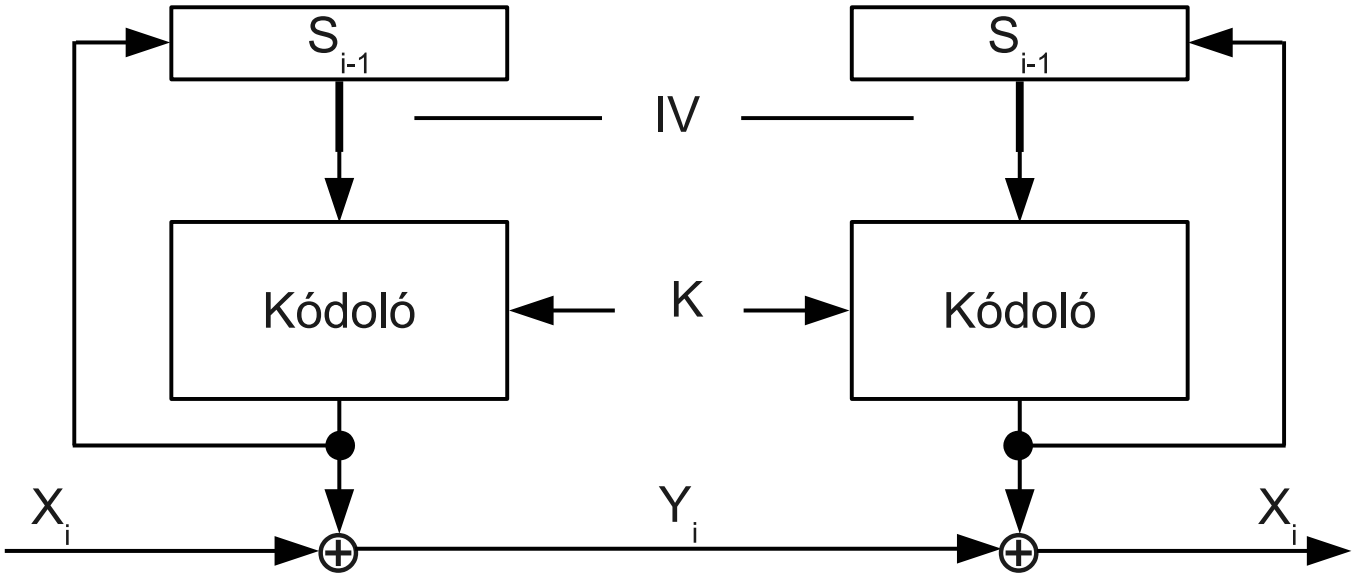
- A kommunikáló feleknek a kulcs mellett IV-ben is meg kell egyezniük
- IV-t szimmetrikus kódolóval is meg lehet osztani egymással
- A módszer nemdeterminisztikus kimenetet eredményez

Bankos példa

- Ha minden üzenetnél más IV-t használnak, nem fedezhetünk fel mintákat a kódolt üzenetekben
- Ha ugyanazt az IV-t használják, akkor:
  1. Elfogjuk az üzeneteket
  2. Kicseréljük a 4. blokkot (azaz az azonosítót), majd az üzenetet tovább küldjük
  3. A vevő oldalon visszafejtik az üzenetet
  4. A 4. és 5. blokk véletlenszerű értéket vesz fel
  5. Valószínűleg senki nem fog véletlenszerű mennyiségű pénzt kapni a számlájára

Kimenet visszacsatolása

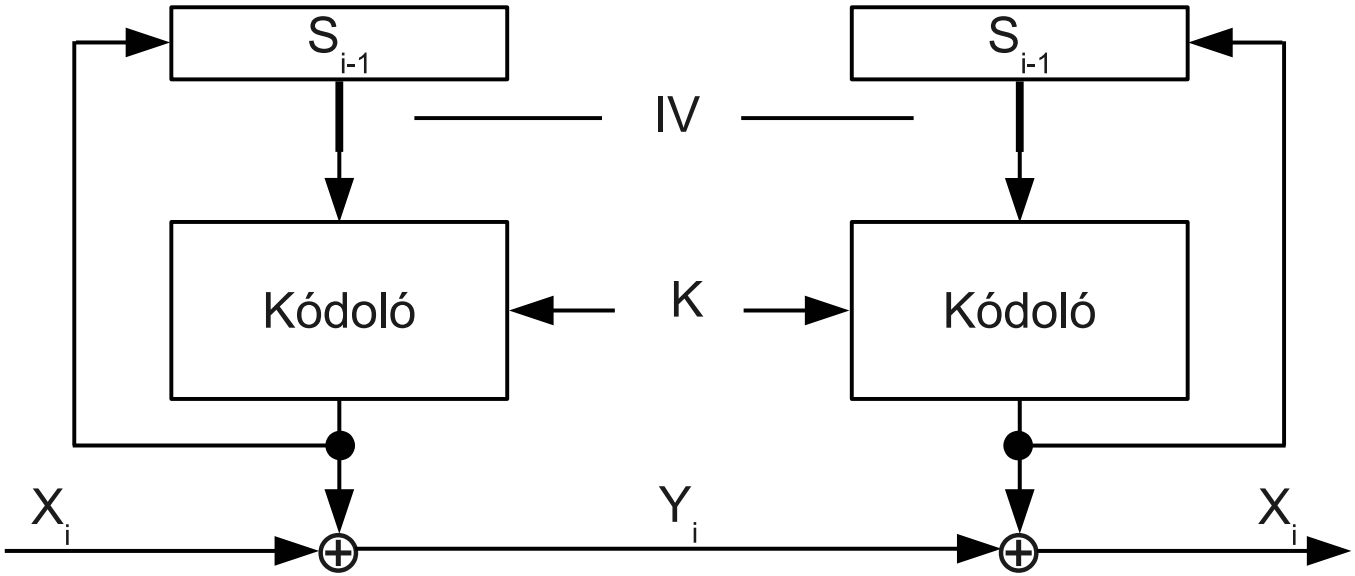
Output Feedback mode (OFB)



- Inicializáló vektor használata

Kimenet visszacsatolása

Output Feedback mode (OFB)

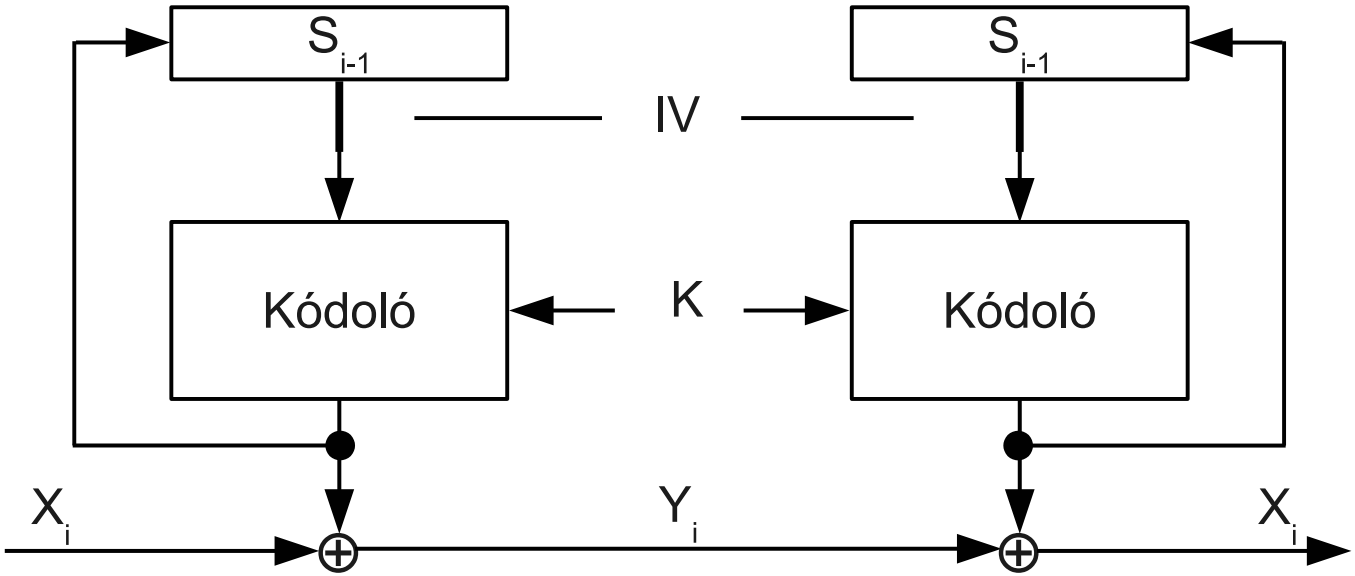


- Inicializáló vektor használata
- Első blokk:

$$S_1 = E_k(IV), Y_1 = X_1 \oplus S_1 \xrightarrow{\text{dekódolás}} S_1 = E_k(IV), X_1 = Y_1 \oplus S_1$$

Kimenet visszacsatolása

Output Feedback mode (OFB)



- Inicializáló vektor használata
- Első blokk:

$$S_1 = E_k(IV), Y_1 = X_1 \oplus S_1 \xrightarrow{\text{dekódolás}} S_1 = E_k(IV), X_1 = Y_1 \oplus S_1$$

- Következő blokkok:

$$S_i = E_k(S_{i-1}), Y_i = X_i \oplus S_i \xrightarrow{\text{dekódolás}} S_i = E_k(S_{i-1}), X_i = Y_i \oplus S_i$$

Kimenet visszacsatolása (OFB)

- A kódoló kimenete független a titkosítandó szövegtől
- Lényegében egy folyam titkosítót kapunk
- A kulcs folyamot akár előre is kiszámolhatjuk
- Ha az üzenet átvitelekor 1 bit hiba történik, akkor a vevő oldalon is csak 1 bit hiba keletkezik
- A vevő oldalon is kódolót kell használni, hogy ugyanazt a kulcsfolyamot állítsuk elő, mint az adó oldalon

Kimenet visszacsatolása (OFB)

- A kódoló kimenete független a titkosítandó szövegtől
- Lényegében egy folyam titkosítót kapunk
- A kulcs folyamot akár előre is kiszámolhatjuk
- Ha az üzenet átvitelekor 1 bit hiba történik, akkor a vevő oldalon is csak 1 bit hiba keletkezik
- A vevő oldalon is kódolót kell használni, hogy ugyanazt a kulcsfolyamot állítsuk elő, mint az adó oldalon

Ne ismételjük a kulcs folyamot!

1. Titkosítsuk P_1 és P_2 szöveget ugyanazon K kulcsfolyammal: $C_1 = P_1 \oplus K, C_2 = P_2 \oplus K$

Kimenet visszacsatolása (OFB)

- A kódoló kimenete független a titkosítandó szövegtől
- Lényegében egy folyam titkosítót kapunk
- A kulcs folyamot akár előre is kiszámolhatjuk
- Ha az üzenet átvitelekor 1 bit hiba történik, akkor a vevő oldalon is csak 1 bit hiba keletkezik
- A vevő oldalon is kódolót kell használni, hogy ugyanazt a kulcsfolyamot állítsuk elő, mint az adó oldalon

Ne ismételjük a kulcs folyamot!

1. Titkosítsuk P_1 és P_2 szöveget ugyanazon K kulcsfolyammal: $C_1 = P_1 \oplus K, C_2 = P_2 \oplus K$
2. A támadó elfogja C_1-et és C_2-t, majd XOR-olja őket: $C_1 \oplus C_2 = (P_1 \oplus K) \oplus (P_2 \oplus K) = P_1 \oplus P_2$

Kimenet visszacsatolása (OFB)

- A kódoló kimenete független a titkosítandó szövegtől
- Lényegében egy folyam titkosítót kapunk
- A kulcs folyamot akár előre is kiszámolhatjuk
- Ha az üzenet átvitelekor 1 bit hiba történik, akkor a vevő oldalon is csak 1 bit hiba keletkezik
- A vevő oldalon is kódolót kell használni, hogy ugyanazt a kulcsfolyamot állítsuk elő, mint az adó oldalon

Ne ismételjük a kulcs folyamot!

1. Titkosítsuk P_1 és P_2 szöveget ugyanazon K kulcsfolyammal: $C_1 = P_1 \oplus K, C_2 = P_2 \oplus K$
2. A támadó elfogja C_1-et és C_2-t, majd XOR-olja őket: $C_1 \oplus C_2 = (P_1 \oplus K) \oplus (P_2 \oplus K) = P_1 \oplus P_2$
3. Ha ismerjük P_1-et vagy P_2-t, akkor a másik egy XOR művelettel megkapható

Kimenet visszacsatolása (OFB)

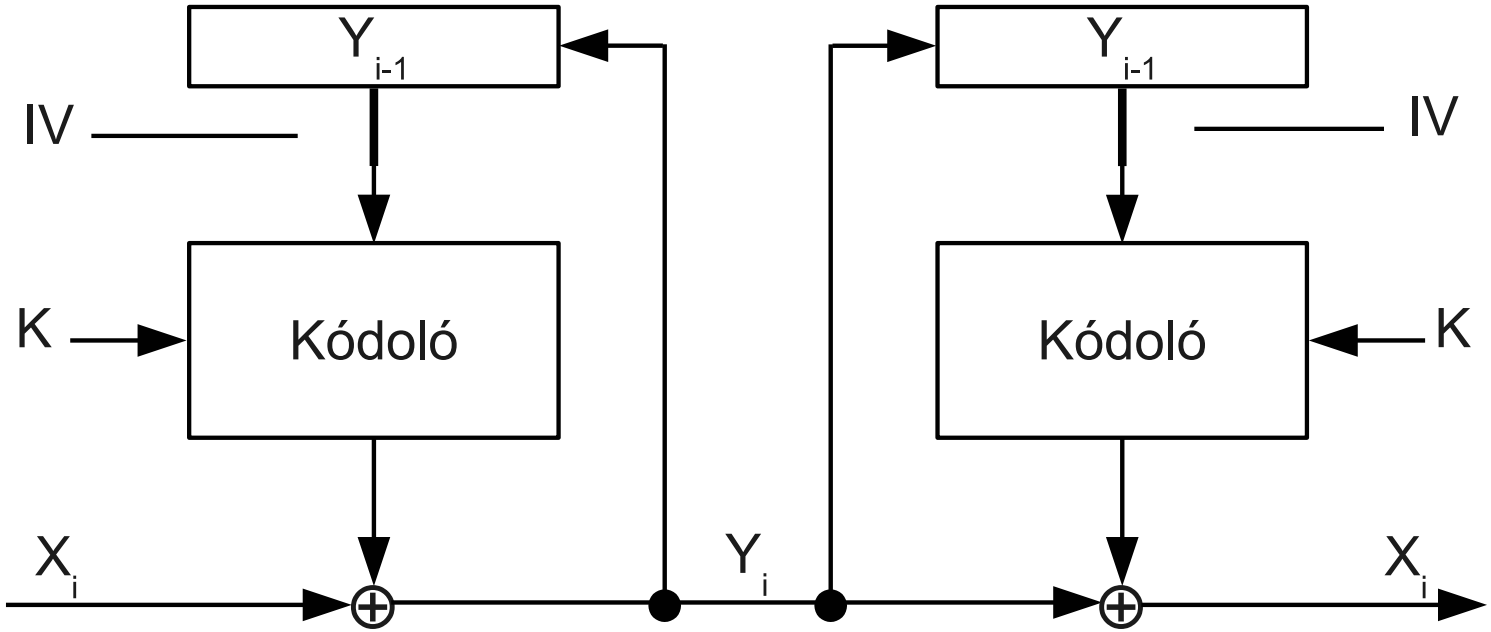
- A kódoló kimenete független a titkosítandó szövegtől
- Lényegében egy folyam titkosítót kapunk
- A kulcs folyamot akár előre is kiszámolhatjuk
- Ha az üzenet átvitelekor 1 bit hiba történik, akkor a vevő oldalon is csak 1 bit hiba keletkezik
- A vevő oldalon is kódolót kell használni, hogy ugyanazt a kulcsfolyamot állítsuk elő, mint az adó oldalon

Ne ismételjük a kulcs folyamot!

1. Titkosítsuk P_1 és P_2 szöveget ugyanazon K kulcsfolyammal: $C_1 = P_1 \oplus K, C_2 = P_2 \oplus K$
2. A támadó elfogja C_1-et és C_2-t, majd XOR-olja őket: $C_1 \oplus C_2 = (P_1 \oplus K) \oplus (P_2 \oplus K) = P_1 \oplus P_2$
3. Ha ismerjük P_1-et vagy P_2-t, akkor a másik egy XOR művelettel megkapható
4. Egyébként $P_1 \oplus P_2$ gyakoriságelemzéssel egyszerűen megfejthető

Kódolt üzenet visszacsatolása

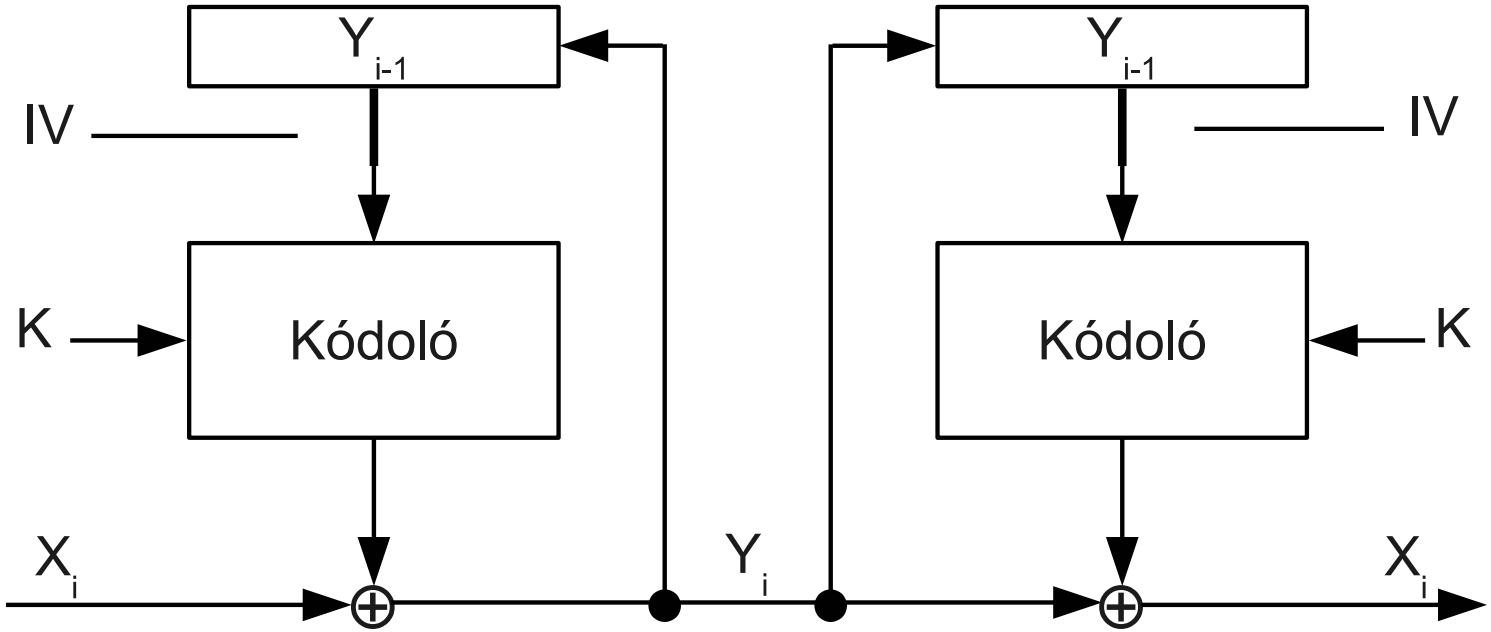
Cipher Feedback mode (CFB) Output Feedback mode (OFB)



- Inicializáló vektor használata

Kódolt üzenet visszacsatolása

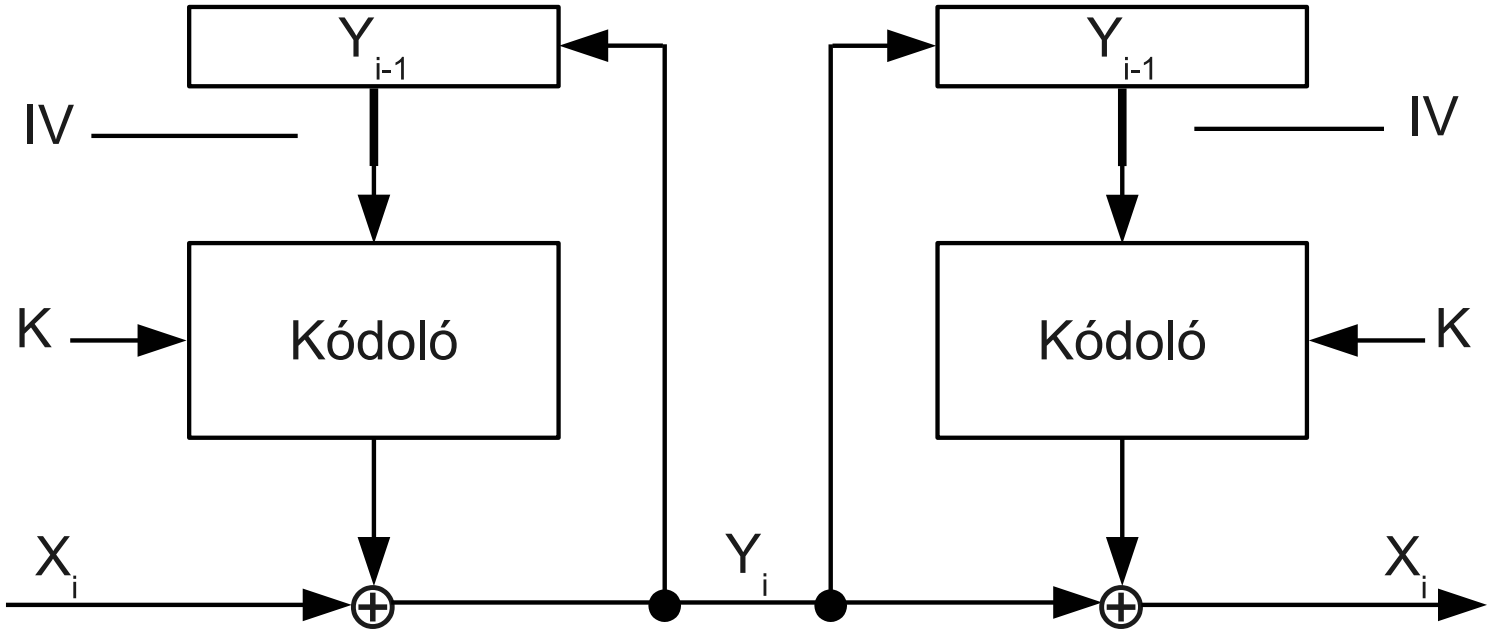
Cipher Feedback mode (CFB) Output Feedback mode (OFB)



- Inicializáló vektor használata
- Első blokk: $Y_1 = X_1 \oplus E_k(IV) \xrightarrow{\text{dekódolás}} X_1 = Y_1 \oplus E_k(IV)$

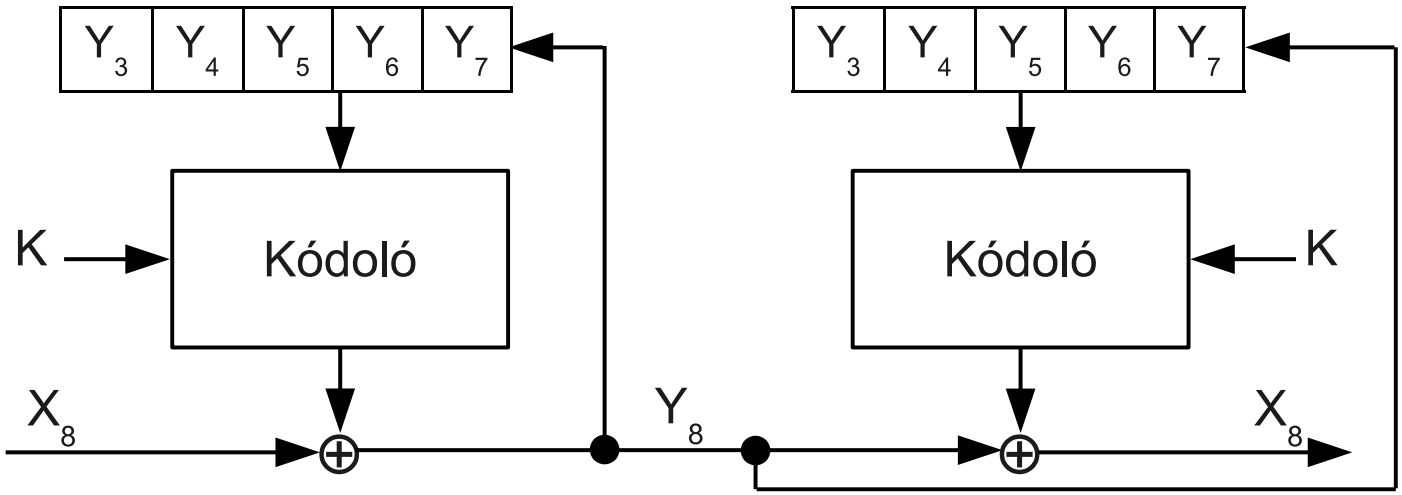
Kódolt üzenet visszacsatolása

Cipher Feedback mode (CFB) Output Feedback mode (OFB)



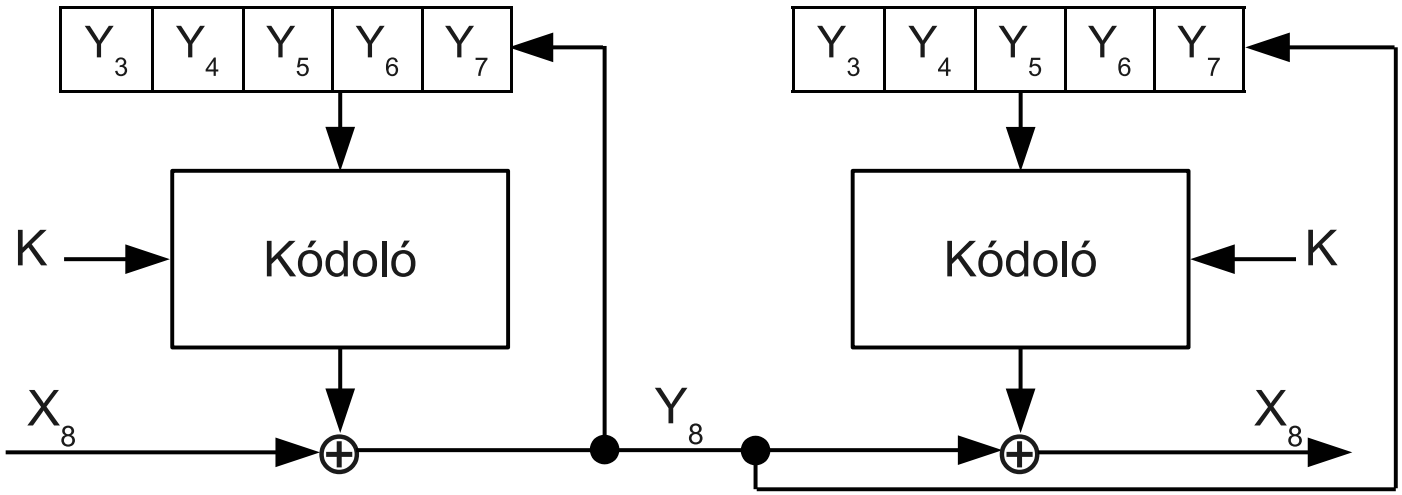
- Inicializáló vektor használata
- Első blokk: $Y_1 = X_1 \oplus E_k(IV) \xrightarrow{\text{dekódolás}} X_1 = Y_1 \oplus E_k(IV)$
- Következő blokkok:
  $Y_i = X_i \oplus E_k(Y_{i-1}) \xrightarrow{\text{dekódolás}} X_i = Y_i \oplus E_k(Y_{i-1})$

Kódolt üzenet visszacsatolása



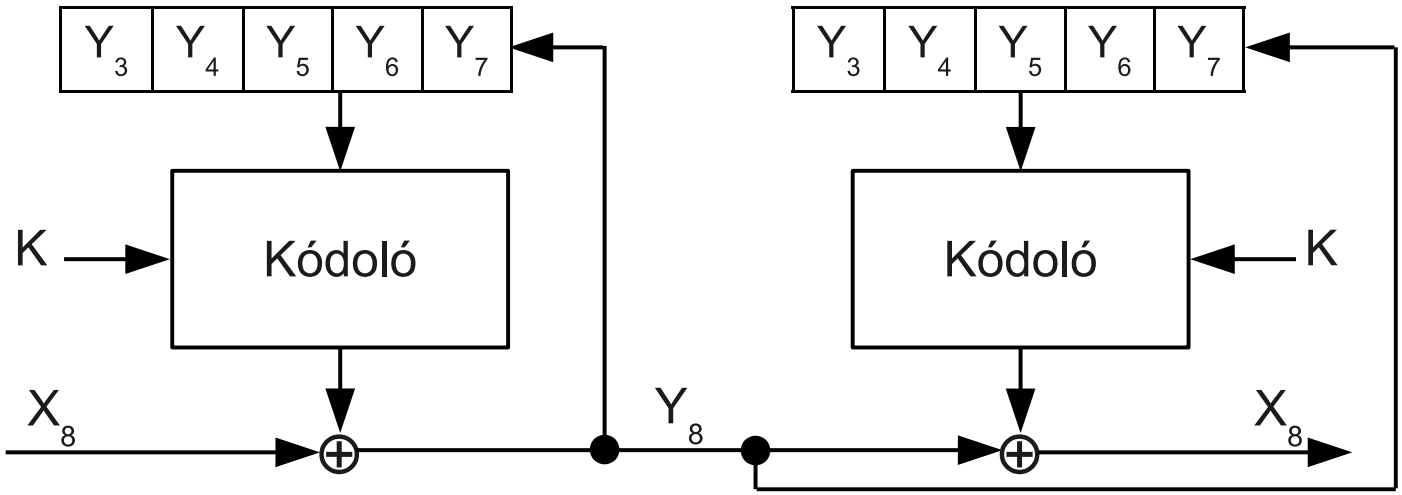
1. Tegyük fel, hogy muszáj a blokkméretnél kisebb méretű üzeneteket is elküldeni

Kódolt üzenet visszacsatolása



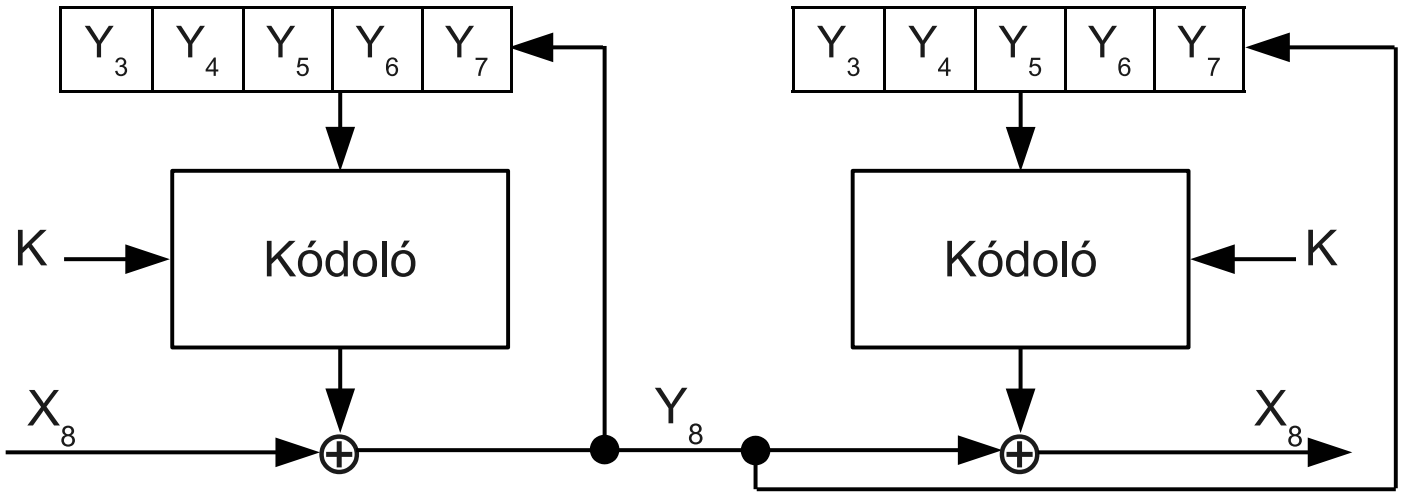
1. Tegyük fel, hogy muszáj a blokkméretnél kisebb méretű üzeneteket is elküldeni
2. Kiegészítsük blokkméretre 0 karakterekkel? $\rightarrow$ Feleslegesen megnő a továbbítandó üzenet mérete

Kódolt üzenet visszacsatolása



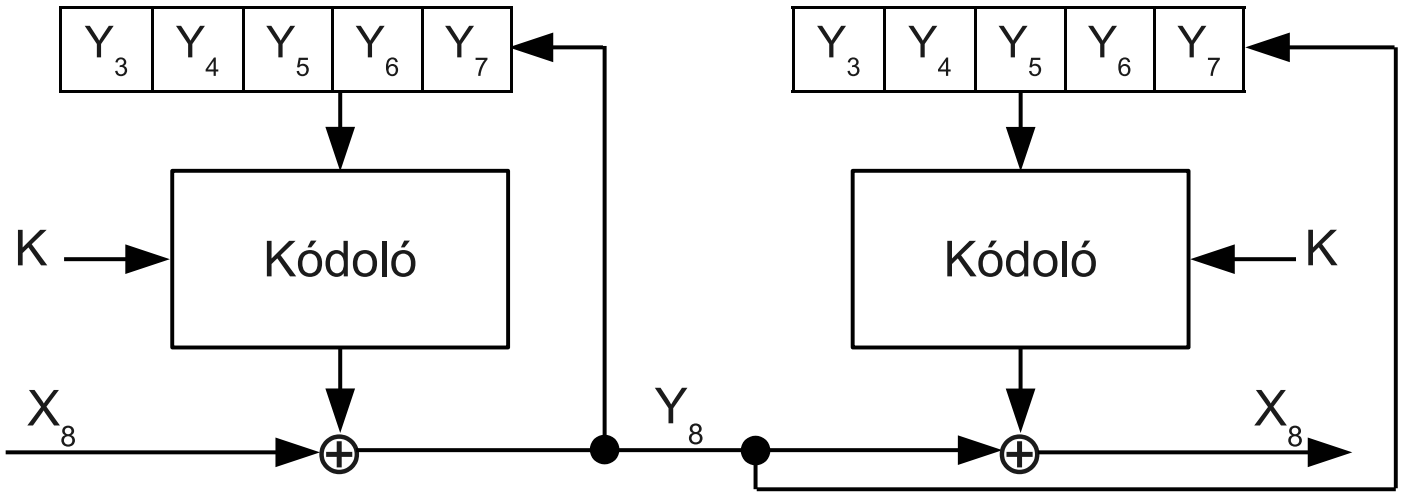
1. Tegyük fel, hogy muszáj a blokkméretnél kisebb méretű üzeneteket is elküldeni
2. Kiegészítsük blokkméretre 0 karakterekkel? $\rightarrow$ Feleslegesen megnő a továbbítandó üzenet mérete
3. X_i legyen a blokkméretnél rövidebb

Kódolt üzenet visszacsatolása



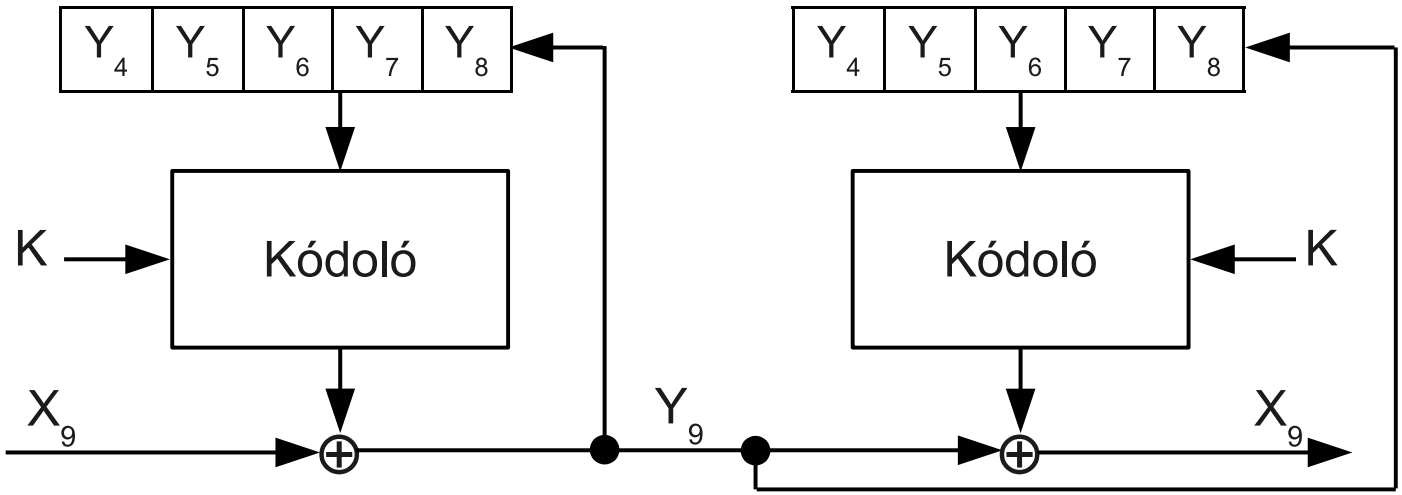
1. Tegyük fel, hogy muszáj a blokkméretnél kisebb méretű üzeneteket is elküldeni
2. Kiegészítsük blokkméretre 0 karakterekkel? $\rightarrow$ Feleslegesen megnő a továbbítandó üzenet mérete
3. X_i legyen a blokkméretnél rövidebb
4. A kódoló bemenetét egy shift-regiszterben tároljuk, ez kezdetben IV-vel egyezik meg

Kódolt üzenet visszacsatolása



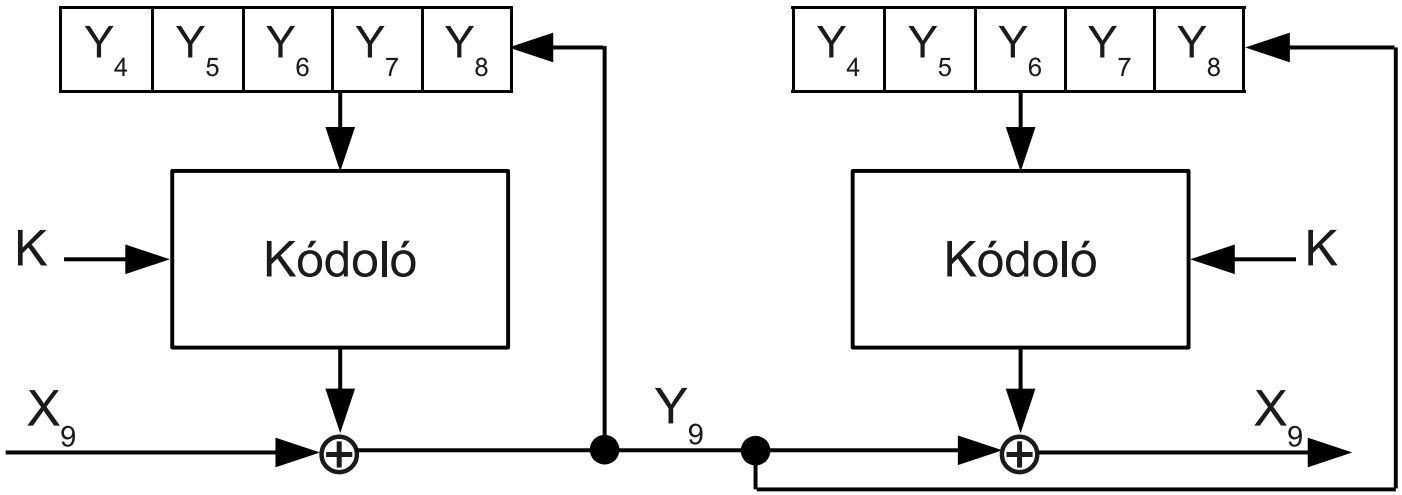
1. Tegyük fel, hogy muszáj a blokkméretnél kisebb méretű üzeneteket is elküldeni
2. Kiegészítsük blokkméretre 0 karakterekkel? $\rightarrow$ Feleslegesen megnő a továbbítandó üzenet mérete
3. X_i legyen a blokkméretnél rövidebb
4. A kódoló bemenetét egy shift-regiszterben tároljuk, ez kezdetben IV-vel egyezik meg
5. A kódoló kimenetének bal oldalát XOR-oljuk X_i-vel

Kódolt üzenet visszacsatolása



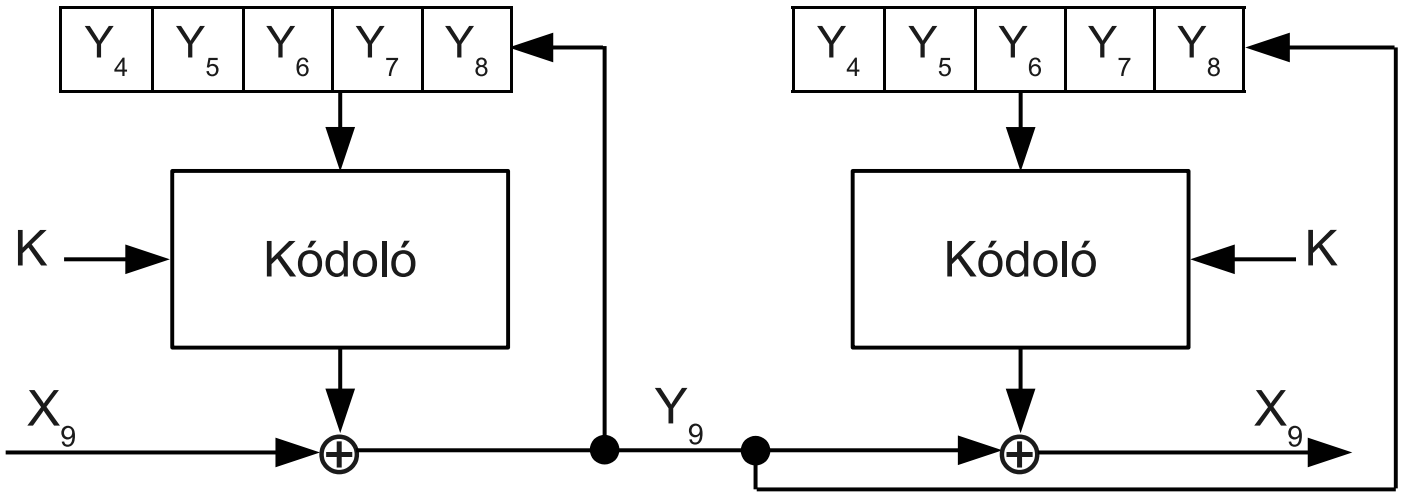
6. A kapott Y_i hossza X_i-vel egyezik, ezt visszatöltjük a shift-regiszterbe

Kódolt üzenet visszacsatolása



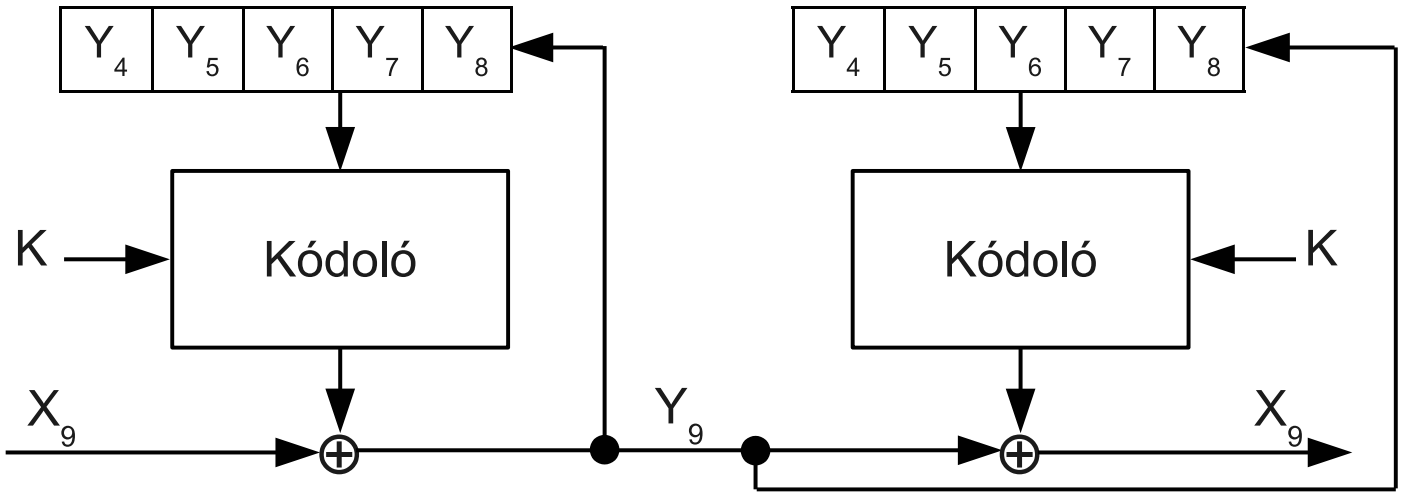
6. A kapott Y_i hossza X_i-vel egyezik, ezt visszatöltjük a shift-regiszterbe
7. A vevő oldalon a kapott Y_i-t dekódoljuk a kódoló kimenetének bal oldalával, megkapjuk X_i-t

Kódolt üzenet visszacsatolása



6. A kapott Y_i hossza X_i-vel egyezik, ezt visszatöltjük a shift-regiszterbe
7. A vevő oldalon a kapott Y_i-t dekódoljuk a kódoló kimenetének bal oldalával, megkapjuk X_i-t
8. Y_i-t itt is betöltjük a shift-regiszterbe

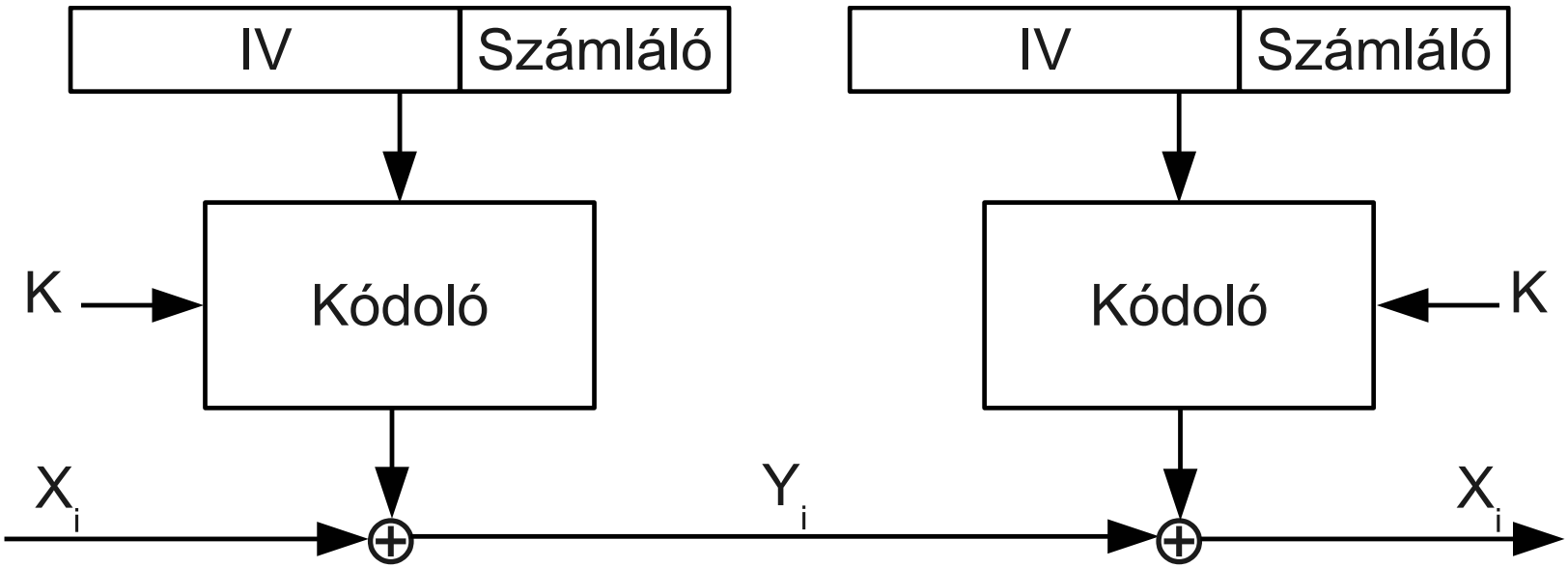
Kódolt üzenet visszacsatolása



6. A kapott Y_i hossza X_i-vel egyezik, ezt visszatöltjük a shift-regiszterbe
7. A vevő oldalon a kapott Y_i-t dekódoljuk a kódoló kimenetének bal oldalával, megkapjuk X_i-t
8. Y_i-t itt is betöltjük a shift-regiszterbe
9. Ha Y_i az átvitel során sérül, akkor amíg a shift-regiszterben van, addig hibásan dekódolt X_i-ket kapunk

Számláló mód

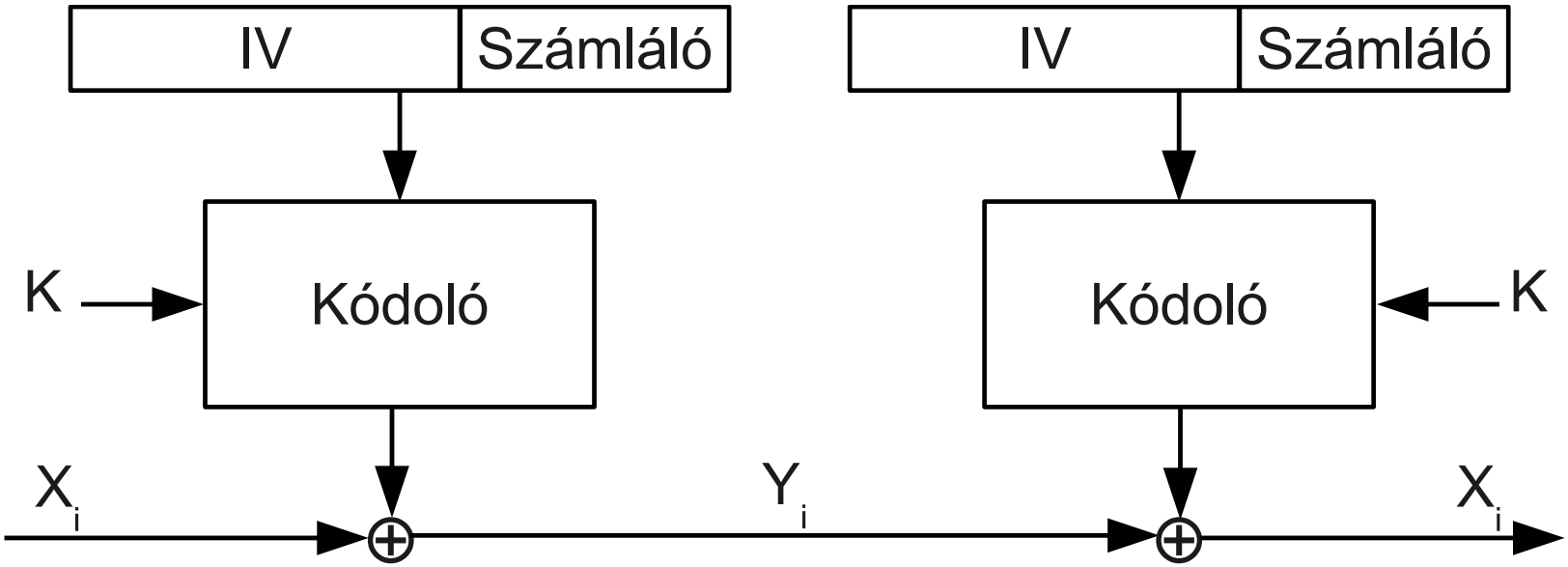
Counter mode (CTR)



- Az inicializáló vektor mellett van 1-1 számlálónk a kódolónál (CTR_e) és a dekódolónál (CTR_d)

Számláló mód

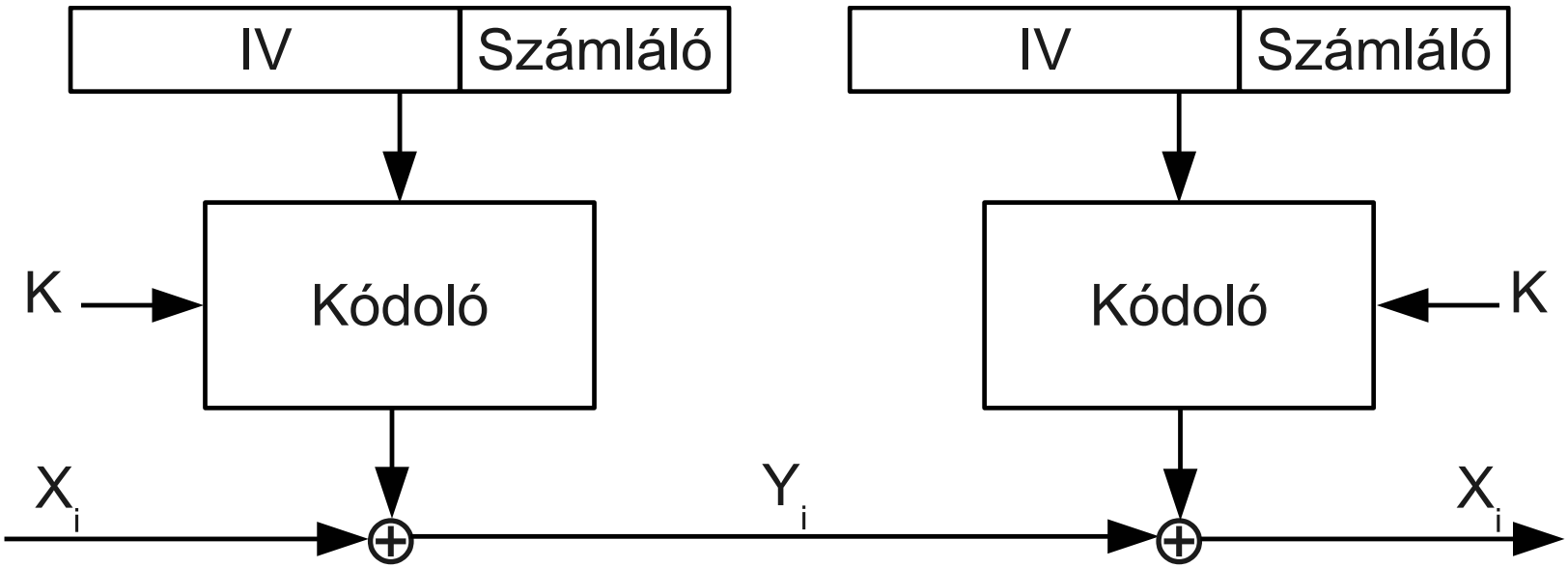
Counter mode (CTR)



- Az inicializáló vektor mellett van 1-1 számlálónk a kódolónál (CTR_e) és a dekódolónál (CTR_d)
- Kezdetben $CTR_e = CTR_d$

Számláló mód

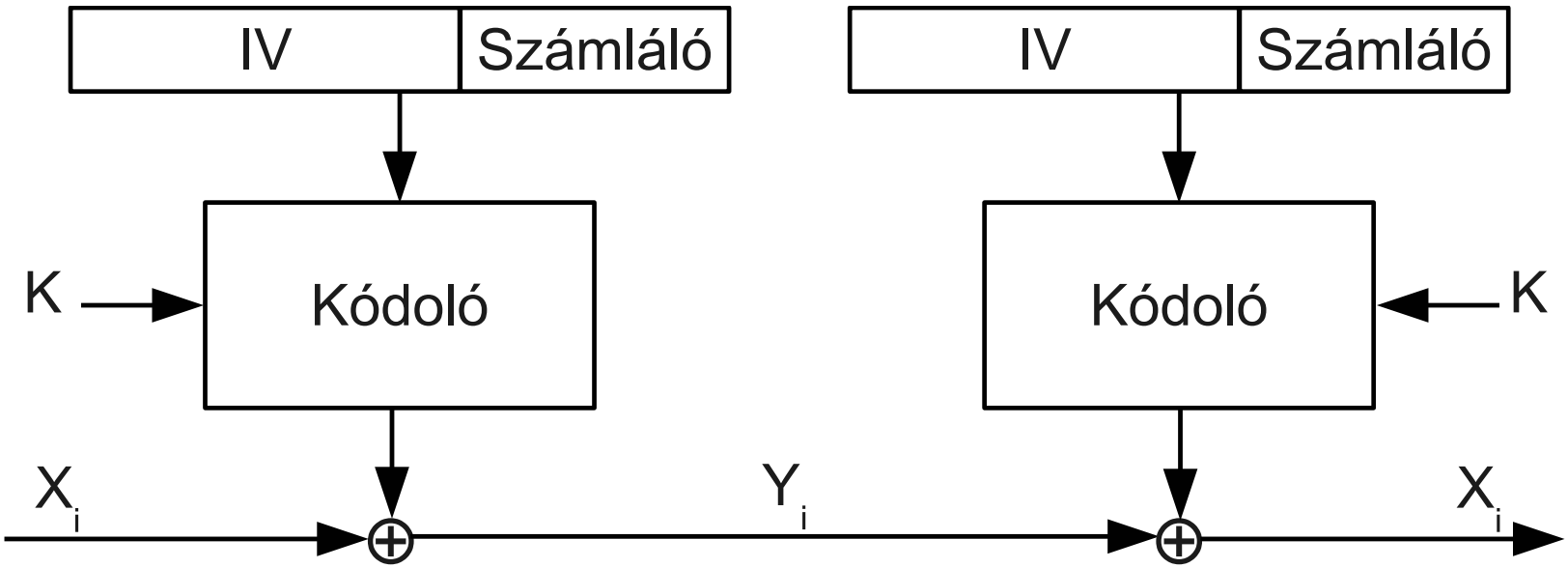
Counter mode (CTR)



- Az inicializáló vektor mellett van 1-1 számlálónk a kódolónál (CTR_e) és a dekódolónál (CTR_d)
- Kezdetben $CTR_e = CTR_d$
- Kódolás: $Y_i = X_i \oplus E_k(IV|CTR_e)$, majd CTR_e-t 1-el növeljük

Számláló mód

Counter mode (CTR)



- Az inicializáló vektor mellett van 1-1 számlálónk a kódolónál (CTR_e) és a dekódolónál (CTR_d)
- Kezdetben $CTR_e = CTR_d$
- Kódolás: $Y_i = X_i \oplus E_k(IV|CTR_e)$, majd CTR_e-t 1-el növeljük
- Dekódolás: $X_i = Y_i \oplus E_k(IV|CTR_d)$, majd CTR_d-t 1-el növeljük

Számláló mód

- Tegyük fel, hogy egy fájlt titkosítunk
- Szeretnénk közvetlen módon elérni a fájl egy adott rekordját
- A CBC, OFB és CFB módokban az i kódolt blokkra hatással van az összes megelőző blokk, vagy kulcs folyam
- CBC, OFB és CFB esetén ezért a fájl elejéről kell olvasni, míg el nem jutunk a kívánt pozícióig
- ECB viszont gyenge
- A CTR módban az i. kódolt blokkra nincs hatással a többi blokk
- CTR módban közvetlenül dekódolhatjuk az i. blokkot
- Ha két fájlt azonos kulccsal és IV-vel kódolunk, ugyanaz a probléma lép fel, mint az OFB-nél