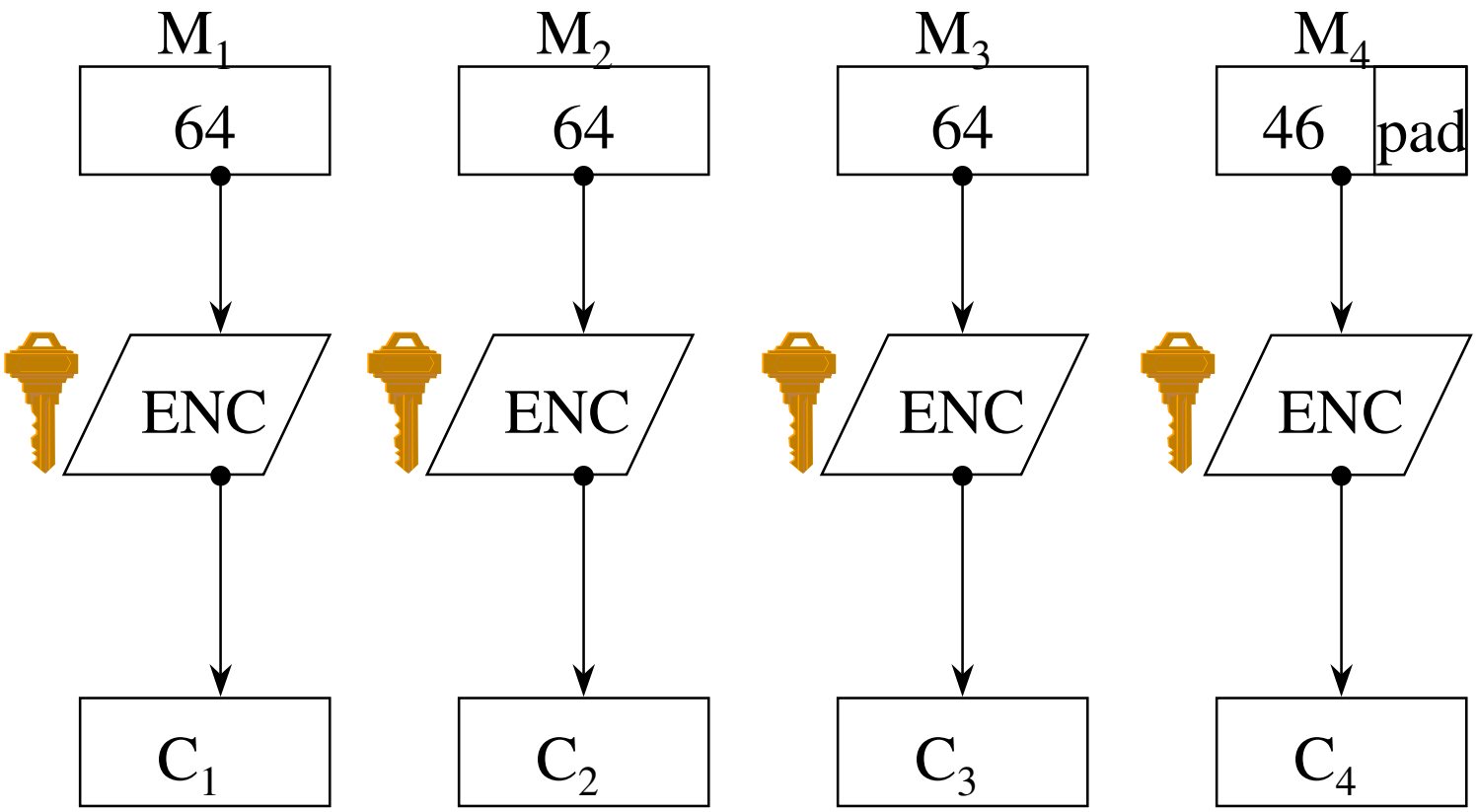
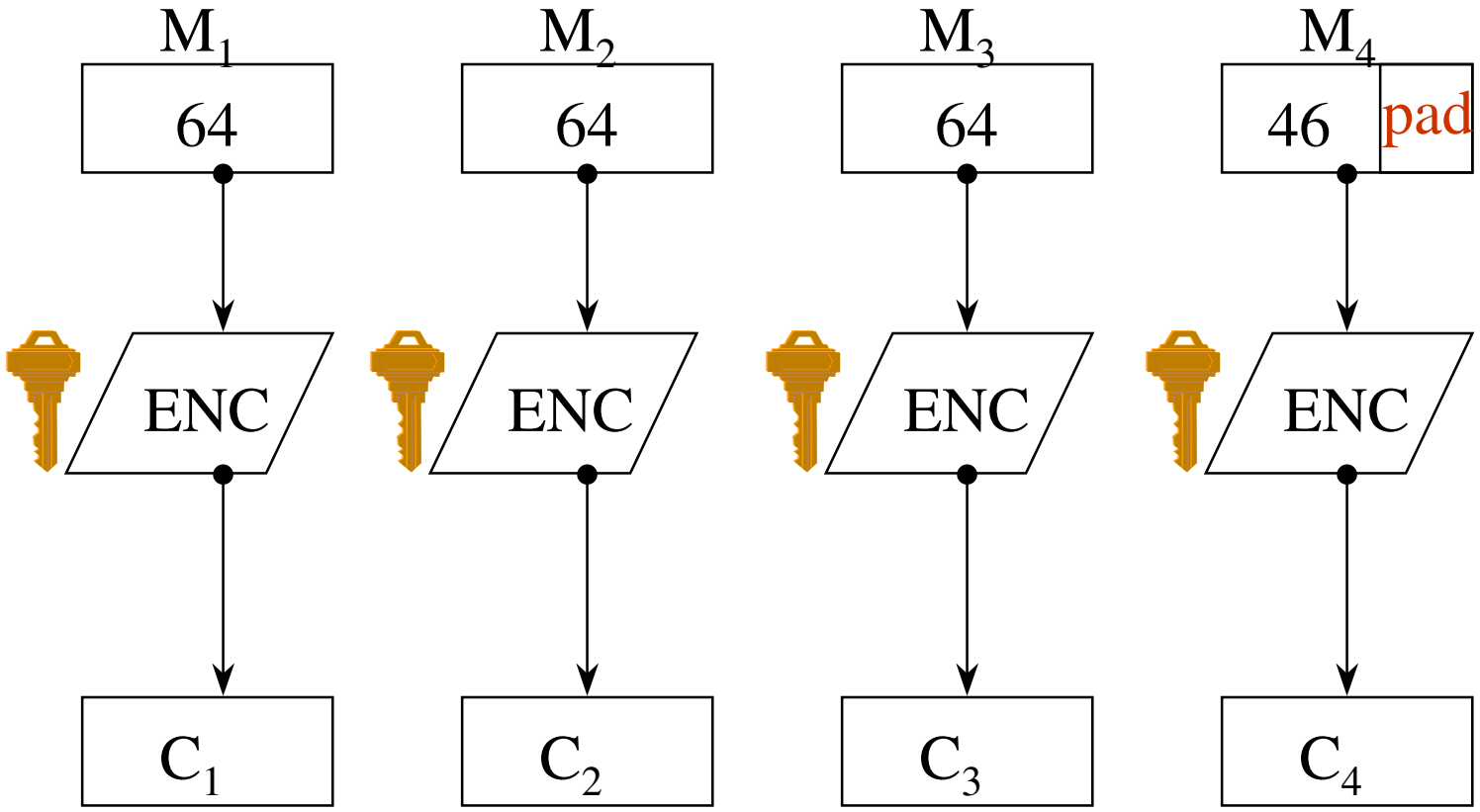
Modalità Operative dei Cifrari a Blocchi

- ECB (Electronic Code Book)
- CBC (Cipher Block Chaining Mode)
- OFB (Output Feedback Mode)
- CFB (Cipher Feedback Mode)

Electronic Code Book (ECB)



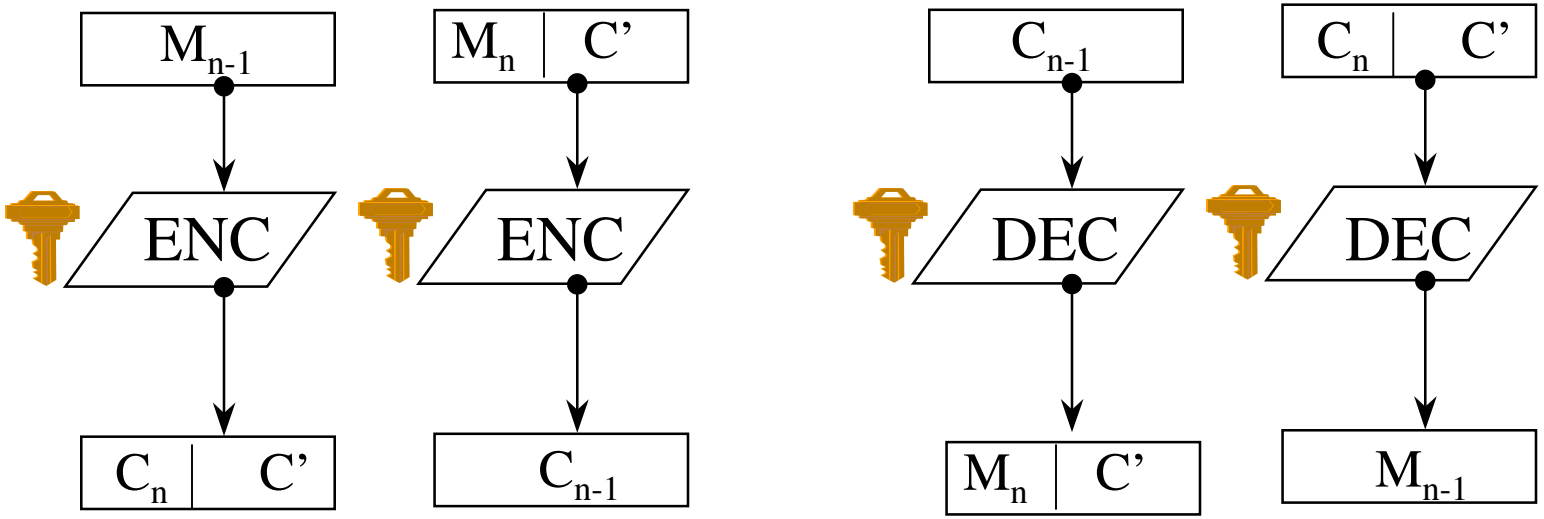
ECB Problem #1



$(M_1 == M_3) => (C_1 == C_3)$

Ciphertext Stealing: ovvero omettere il Pad

- Si può usare un trucco intelligente, denominato **Ciphertext stealing**, per evitare di avere pad.
  I due blocchi finali del messaggio sono trattati in modo speciale:



Un Problema dell'ECB

- Se i messaggi sono molto ridondanti, o contengono sequenze comuni di bits, la cifratura di questi blocchi corrisponderà sempre allo stesso plaintext, che potrebbe quindi essere riconosciuto da un attaccante a prescindere dalla forza del cifrario.

- Supponiamo che le risposte "yes" / "no" siano frequentemente inviate, se queste sono sempre riconoscibili, allora, successivamente, l'attaccante sarà sempre in grado di riconoscere queste risposte.

Altro Problema dell'ECB: Block Replay

- Supponiamo che i messaggi siano formattati in modo standard, allora un attaccante potrebbe fare il replay di vecchi blocchi.

- Supponiamo che una banca usi inviare ad un'altra banca notifiche del bonifico usando 2 blocchi, di cui il primo contiene il numero del conto a cui fare il bonifico, ed il secondo contiene la cifra del bonifico. Un attaccante memorizza questi messaggi e potrebbe modificare o l'importo o il numero di conto dei blocchi inviati in precedenza.

- L'attaccante potrebbe anche non sapere in che modo il messaggio è stato modificato, comunque, l'autenticità del messaggio è stata violata.

Riepilogo sull'ECB

- Sicurezza:
  - Le sequenze di plaintext non sono segrete.
  - L'input al cifrario a blocchi non è reso casuale; è lo stesso del plaintext.
  - E' facile manipolare il plaintext; i blocchi possono essere eliminati, ripetuti, o scambiati.

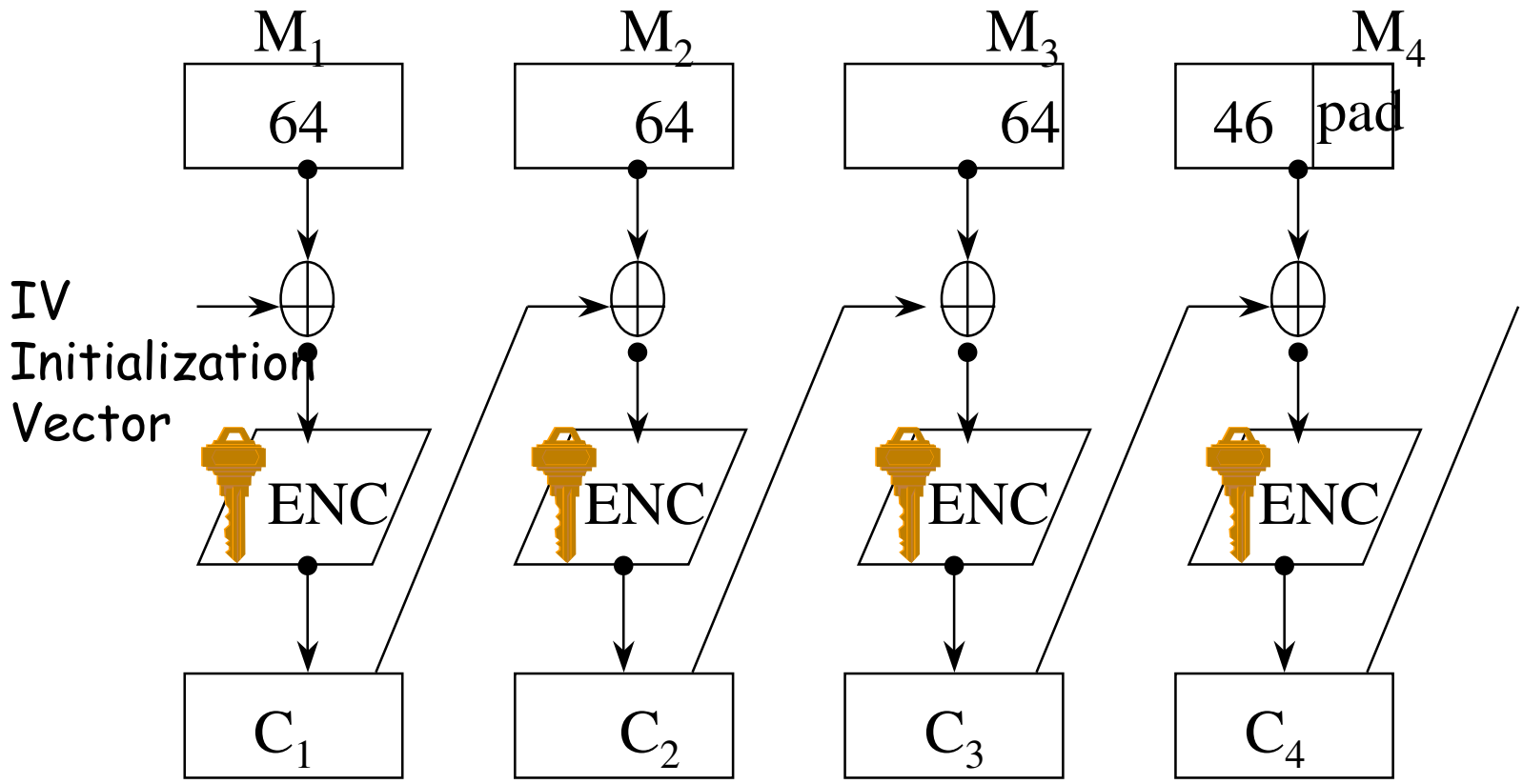
Riepilogo sull'ECB (cont).

- Efficienza:
  - + Il tempo di esecuzione.
  - - Il ciphertext è più lungo del plaintext di un blocco, che corrisponde al padding.
  - - Non è possibile fare alcun preprocessamento.
  - + L'elaborazione può essere fatta in parallelo.
- Fault-tolerance:
  - - Un errore nel ciphertext condiziona un blocco intero del plaintext.

Cipher Block Chaining (CBC)

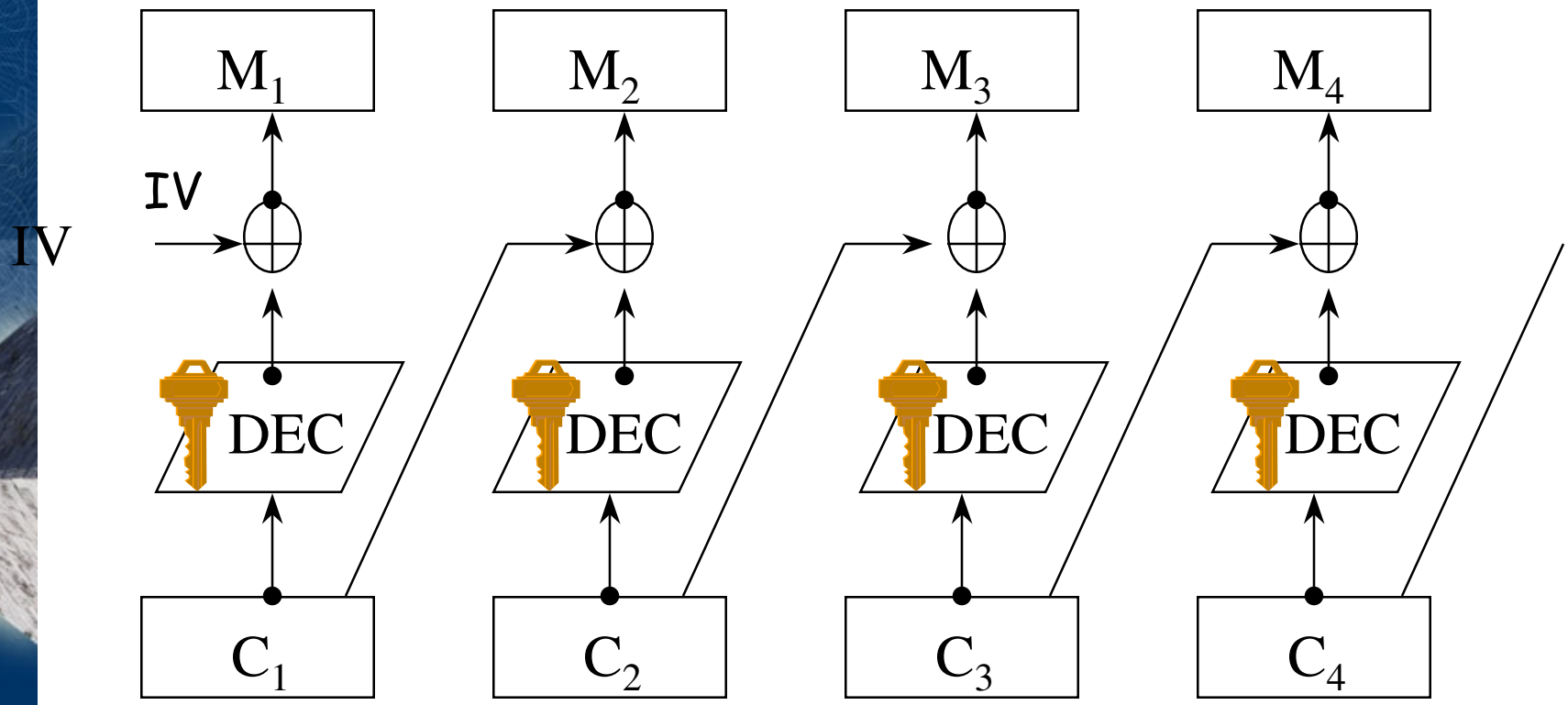
- Una possibile soluzione ai problemi dell'ECB è quella di concatenare l' output di un blocco al successivo. Viene usato un blocco iniziale (denominato Initialization Vector IV) per iniziare il concatenamento
- In questo modo, la cifratura di messaggi uguali può avere risultati diversi. Ciò previene completamente il replay attack.

Cipher Block Chaining (CBC)



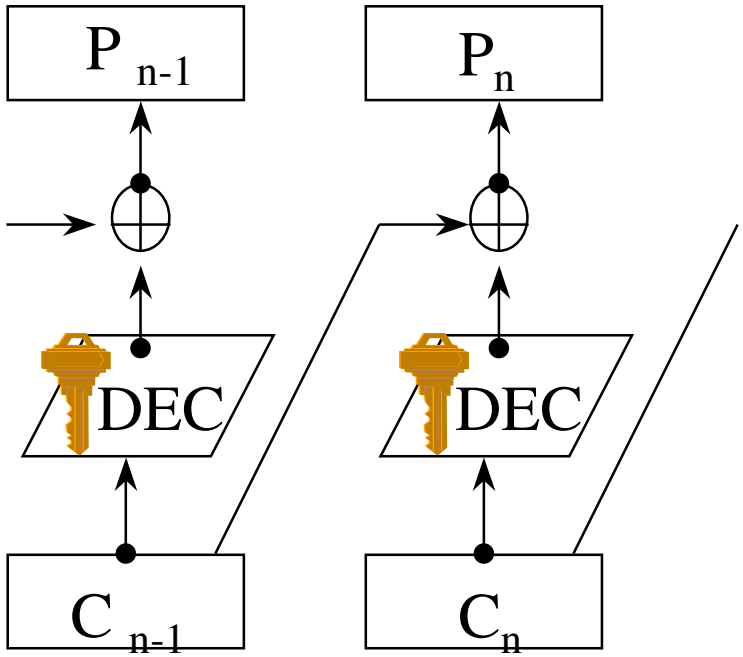
(M_1 == M_3) difficilmente produce (C_1 == C_3)

CBC Decryption



Punti di debolezza dell'CBC

- La perdita di sincronizzazione dell'ultimo blocco altera tutto il resto
- E' possibile creare modifiche desiderate nel blocco decifrato P_n sacrificando il blocco P_{n-1}



Riepilogo sull'CBC

- Sicurezza:
  - + Le sequenze di plaintext sono segrete grazie all'XOR con il precedente blocco di ciphertext.
  - + L'input al cifrario a blocchi è reso casuale grazie all'XOR con il precedente blocco di ciphertext.
  - +/- Il plaintext è più difficile da manipolare; i blocchi possono essere eliminati dall'inizio e dalla fine del messaggio, i bits del primo blocco possono essere modificati, e le ripetizioni favoriscono delle modifiche controllate.

Riepilogo sull'CBC

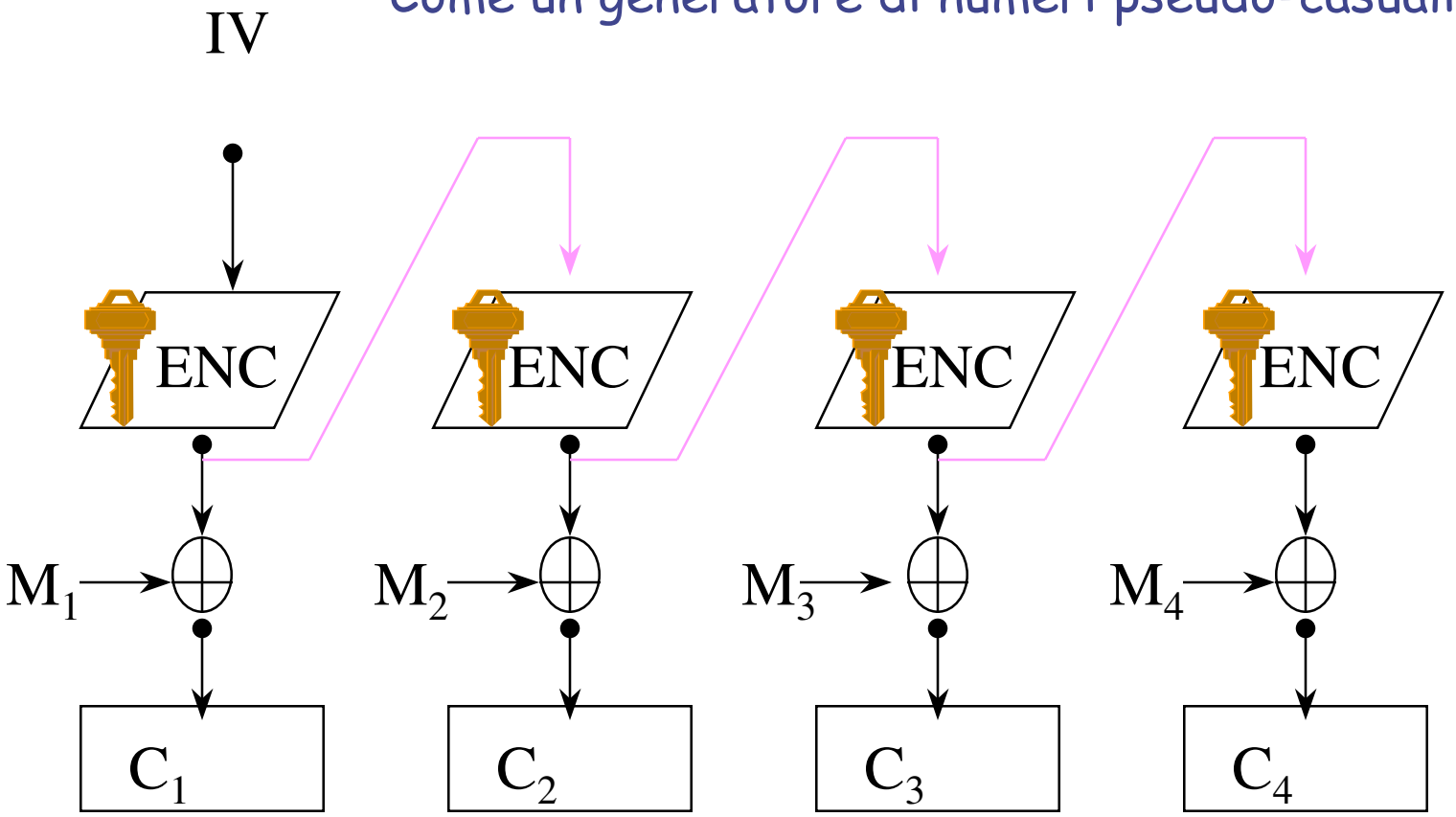
- Sicurezza:
  - Per iniziare occorre un **Initial Value** (IV) che deve essere noto sia al mittente che al ricevente
    - Comunque, se IV viene spedito in chiaro, un attaccante può cambiare i bits del primo blocco, e cambiare lo stesso IV
    - Quindi o IV deve essere un valore fisso (come in EFTPOS) oppure deve essere spedito cifrato in modalità ECB prima del resto del messaggio

Riepilogo sull'CBC (cont)

- Efficienza:
  - + Il tempo di esecuzione è lo stesso di quello del cifrario a blocchi.
  - - Il ciphertext è più lungo del plaintext di un blocco, senza consierare l'IV.
  - - Fault-tolerance:
    - - Un errore nel ciphertext condiziona sia un blocco intero del plaintext che il bit corrispondente nel blocco successivo.

Output Feedback Mode (OFB)

Come un generatore di numeri pseudo-casuali



Caratteristiche dell'OFB

- Vantaggi
  - Permette di pre-calcolare sequenze pseudo-random (One-Time Pad); l'XOR può essere implementato molto efficientemente
  - Non ci sono problemi di propagazione degli errori come per CBC
  - Permette la cifratura/decifratura contemporanea grazie al calcolo bit per bit (diversamente dai blocchi fissi)

Caratteristiche dell'OFB

- Svantaggi
  - È sempre una modalità a flusso, ma dipende dall'utilizzo, se il feedback dell'errore è un problema, o se la cifratura richiede che sia fatto prima che il messaggio sia disponibile
  - Di poco dissimile dal CFB, ma il feedback è dato dal risultato del cifrario a blocchi ed è indipendente dal messaggio, una variante del cifrario di Vernam
  - C'è bisogno ancora di un IV

Caratteristiche dell'OFB

- Svantaggi
  - Il mittente ed il ricevente devono sempre essere sincronizzati, ed è necessario qualche metodo per assicurare la sincronizzazione
  - Anche se inizialmente specificato dagli standards per avere un feedback di m bits variabile, è stato successivamente dimostrato che si dovrebbe usare solo l'**OFB** a **64-bit**. (e comunque questo è anche l'utilizzo più efficiente)

Riepilogo dell' OFB

- Sicurezza:
  + Le sequenze di plaintext sono segrete .
  + L'input al cifrario a blocchi è reso casuale.
  + E' possibile cifrare più di un messaggio con la stessa chiave, a condizione di utilizzare diversi IVs.
  - E' molto facile manipolare il plaintext; qualunque modifica al ciphertext condiziona direttamente il plaintext.

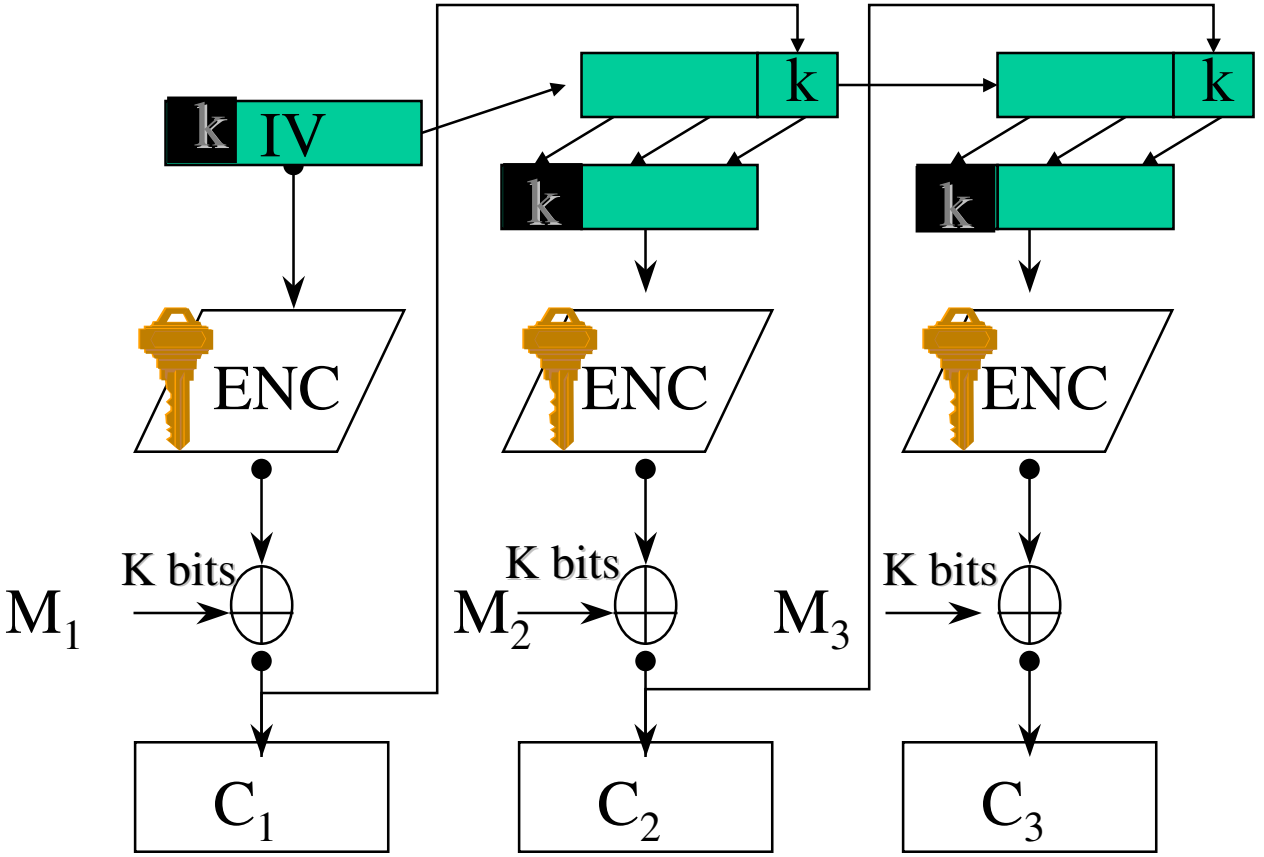
Riepilogo dell' OFB(cont.)

- Efficienza
  - Il tempo di esecuzione è molto meno di quello del cifrario a blocchi.
  - Il Ciphertext ha la stessa lunghezza del plaintext, senza considerare l'IV.
  -+ E' possibile fare elaborazioni prima di vedere il messaggio.
  + L'elaborazione dell'OFB non può essere fatta in parallelo
- Fault-tolerance:
  + Un errore del ciphertext condiziona solo il bit corrispondente del plaintext.
  - Gli errori di sincronizzazione non sono individuabili.

Cipher Feedback Mode

- Trasforma il l'algoritno di enc., ad es. il DES, in un **cifrario a flusso**
- **Elimina** la necessità di inserire bit di **padding** nel messaggio
- L'esecuzione è **real time**
- Ogni carattere può essere **cifrato** e **immediatamente trasmesso**

Cipher Feedback Mode (CFB) generalizzato a k bit



Caratteristiche dell' CFB

- Vantaggio rispetto a CBC.
  - Con k=8, gli errori su un byte del ciphertext condizionano solo gli 8 bytes seguenti.

- Svantaggio rispetto a OFB.
  - Non può essere più calcolata una sequenza casuale in anticipo.

Riepilogo del CFB

- Sicurezza:
  - +Le sequenze di plaintext sono segrete .
  - + L'input al cifrario a blocchi è reso casuale.
  - + E' possibile cifrare più di un messaggio con la stessa chiave, a condizione di utilizzare diversi IVs.
  - +/- E' difficile manipolare il plaintext; I blocchi possono essere eliminati dall'inizio e dalla fine del messaggio, i bits del primo blocco possono essere modificati, e le ripetizioni permettono qualche modifica controllata.

Riepilogo del CFB(cont).

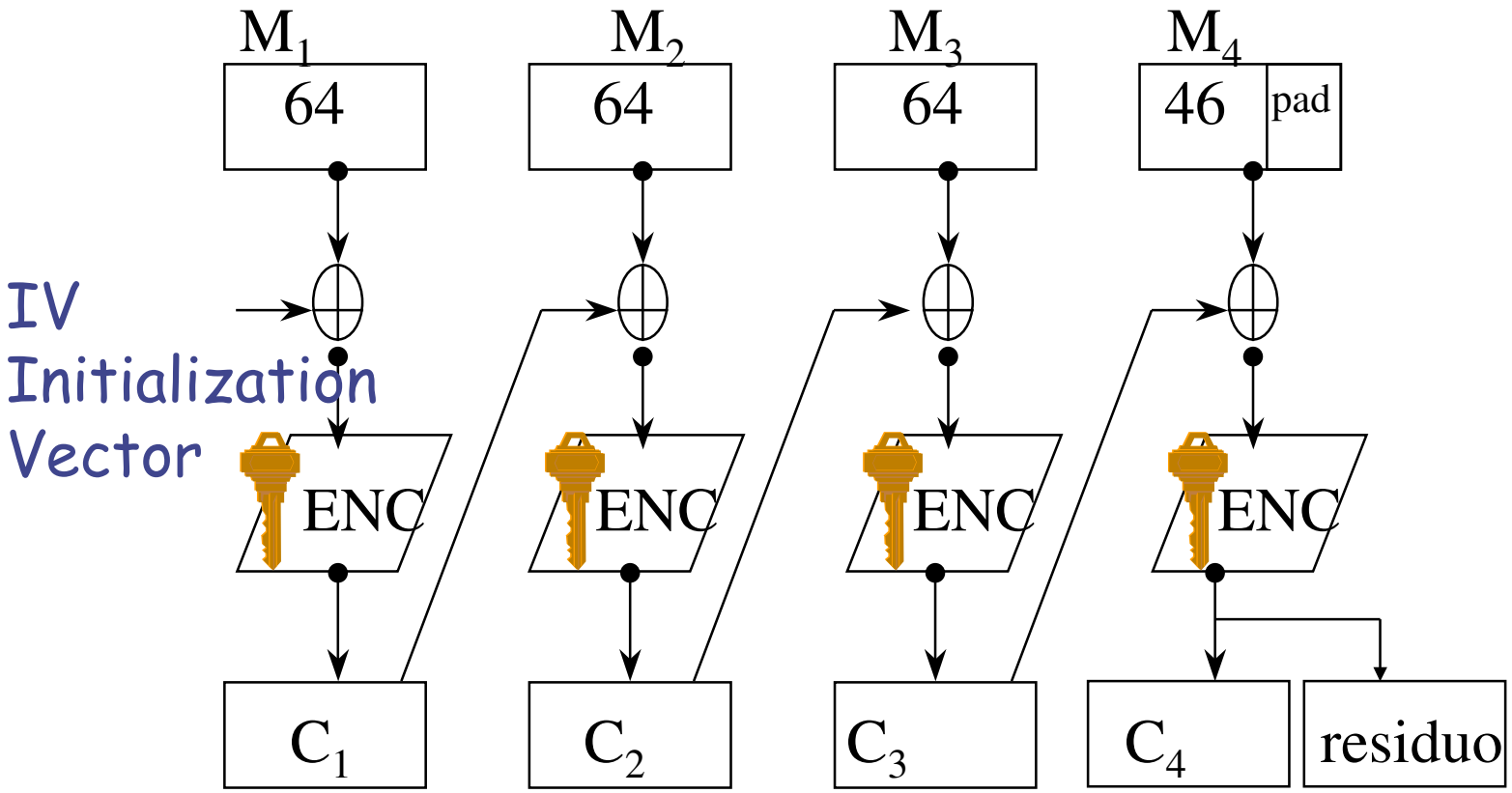
- Efficienza:
  - Il tempo di esecuzione è molto meno di quello del cifrario a blocchi.
  + L'input al cifrario a blocchi è reso casuale
  +/- La cifratura non può essere fatta in parallelo, mentre la decifratura si ed ha una proprietà di accesso casuale.
  - Qualche pre-processamento può essere fatto prima che un blocco sia disponibile; il blocco del ciphertext precedente può essere cifrato.
- Fault-tolerance:
  -Un errore del ciphertext condiziona il bit corrispondente del plaintext e l'intero blocco successivo.
  + Gli errori di sincronizzazione sui blocchi interi sono ripristinabili. CFB ripristina bits modificati dall'aggiunta o perdita di bits singoli

Generare MICs (Message Integrity Codes)

- Si Spedisce solo il blocco finale dell'CBC (residuo CBS)
- Si Spedisce il plaintext

- Qualunque modifica al plaintext modifica anche il residuo CBC
- Assicura l'integrità

CBC con Residuo



Assicurare sia la Confidenzialità che l'Integrità

- Confidentialità+Integrità: usa chiavi segrete separate (ma possono essere collegate) per la cifratura e per MIC (due passi di cifratura)

- CBC (messaggio|hash)

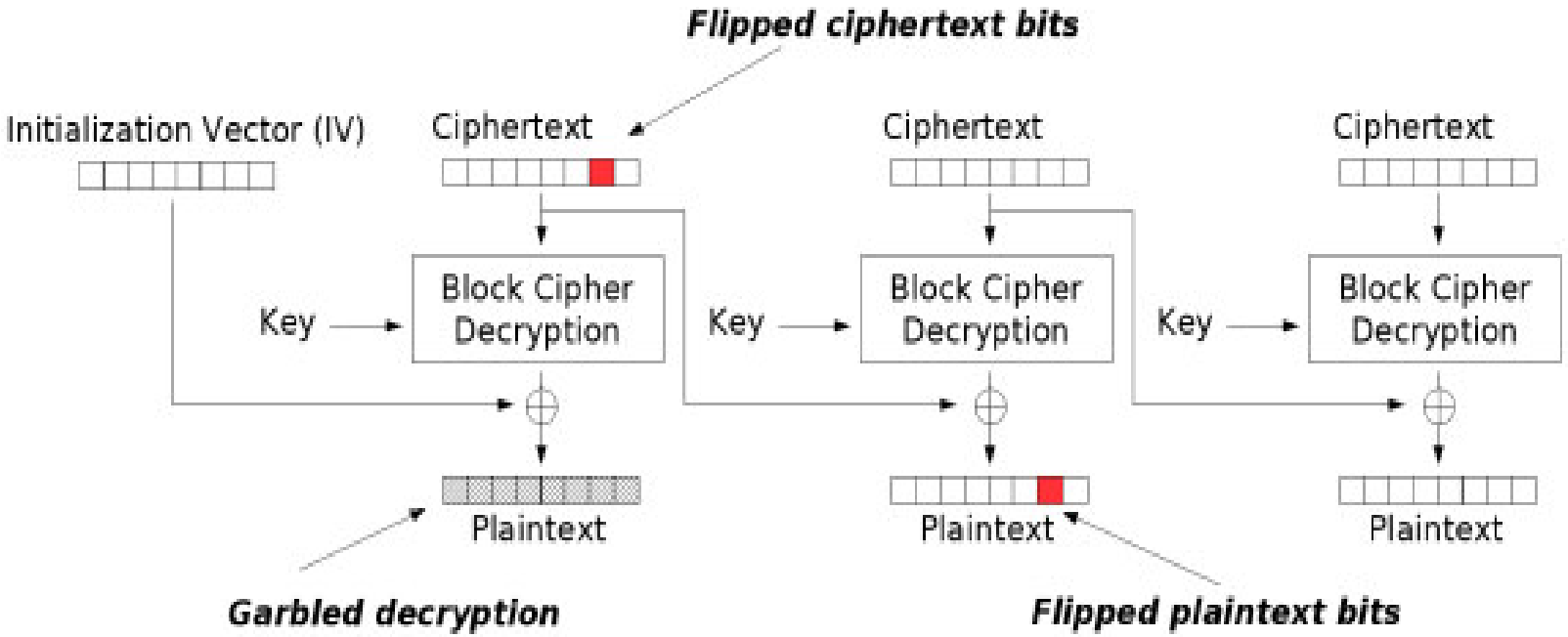
Integrità e Propagazione dell'errore

- I modi operazionali dei cifrari a blocchi fin qui presentati non forniscono nessuna protezione per l'integrità.
- Un attaccante che non conosce la chiave può comunque essere in grado di cambiare il flusso dei dati in qualche modo vantaggioso per lui.
- E' importante fornire la proprietà di integrità per la sicurezza.
- Per le operazioni sicure, l'IV ed il ciphertext ottenuti con questi modi dovrebbero essere autenticati con un MAC sicuro, che possa essere controllato prima della decifratura.

Integrità e Propagazione dell'errore

- Per quanto riguarda la "propagazione degli errori", si noti che ad un errore in un blocco nel ciphertext trasmesso, dovrebbe corrispondere ad un errore di un blocco nel plaintext ricostruito nel caso di cifratura secondo il modo ECB, mentre secondo il modo CBC questo errore corrisponderebbe a due blocchi:

Integrità e Propagazione dell'errore



Propagazione dell'errore in modalità CBC

Sommario

- Le modalità di funzionamento dei cifrari sono qui per restare;
- La modalità da utilizzare per un'applicazione specifica è da selezionare in base ai vincoli di sicurezza, affidabilità, efficienza richiesti all'applicazione;
- ECB e CBC basilari