CS 465
Computer Security

Block Cipher Modes, Authenticated Encryption Modes, and Padding

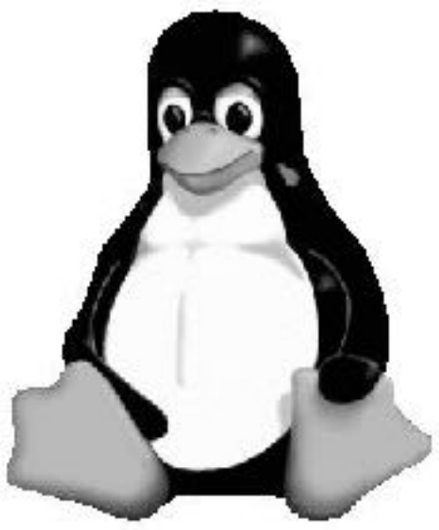
Last Updated: Sep 19, 2017

Block Cipher Modes

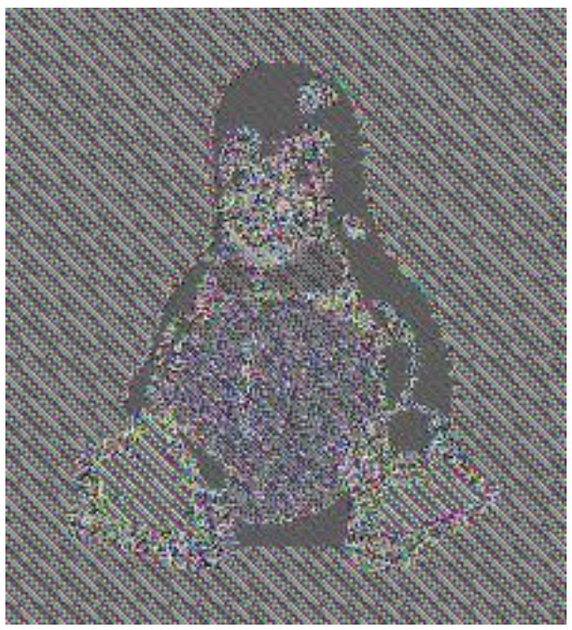
ECB Mode

- Electronic Code Book
- Divide the plaintext into fixed-size blocks
- Encrypt/Decrypt each block independently
- There is a weakness with this approach

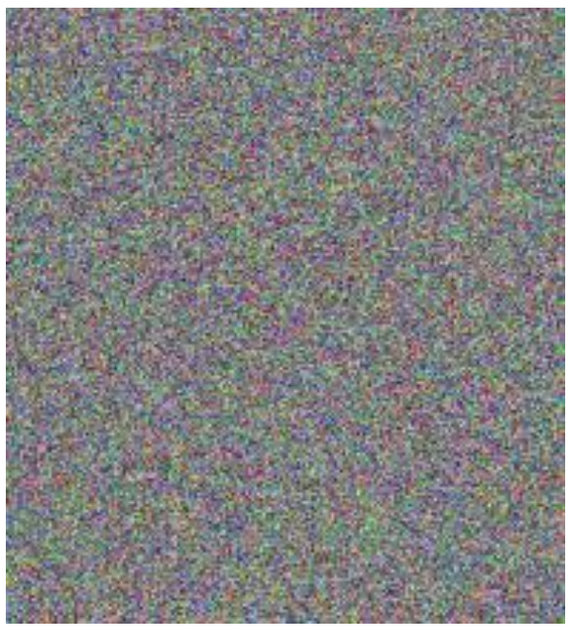
“Plain-Tux”

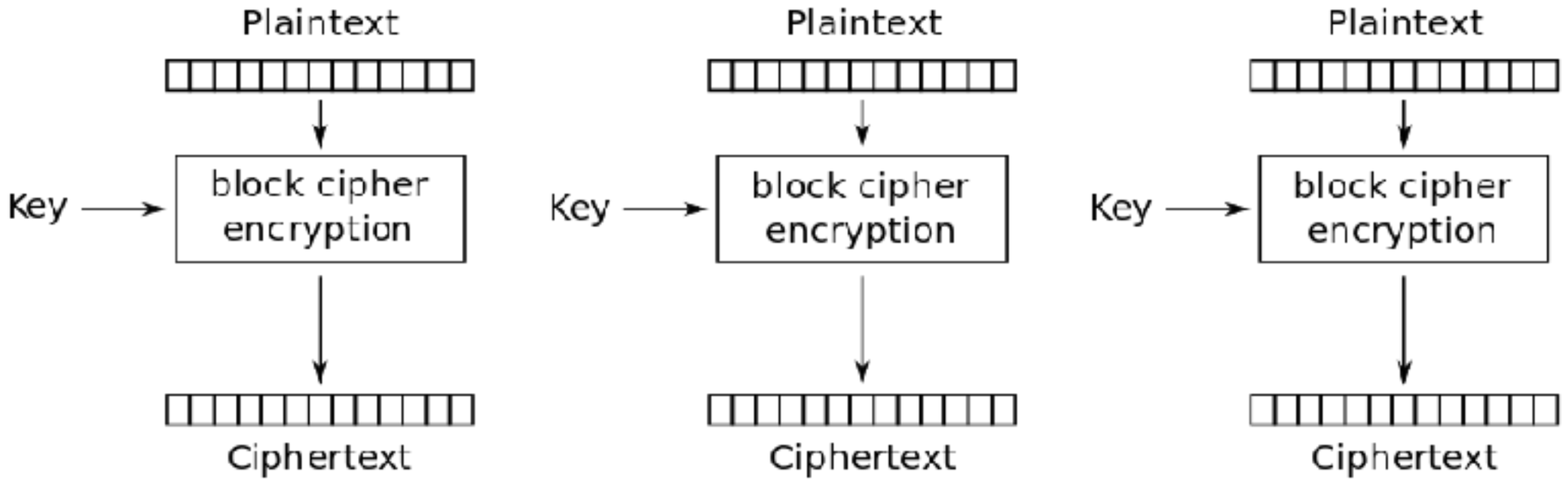


“Cipher-Tux”

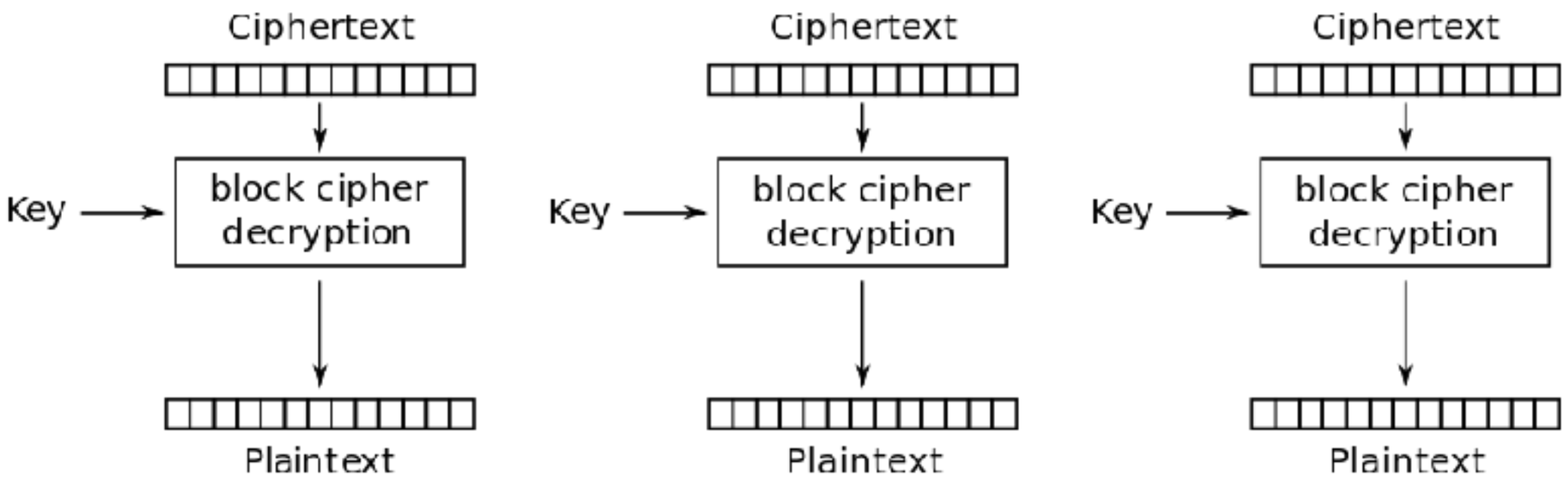


“Cipher-Tux2”





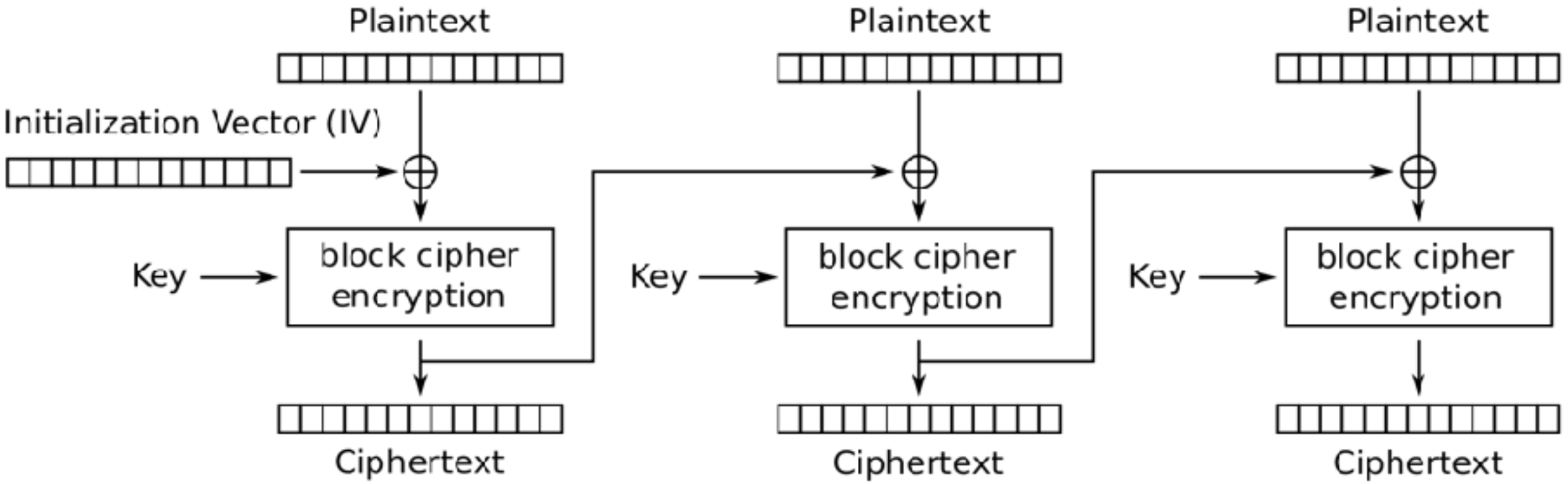
Electronic Codebook (ECB) mode encryption



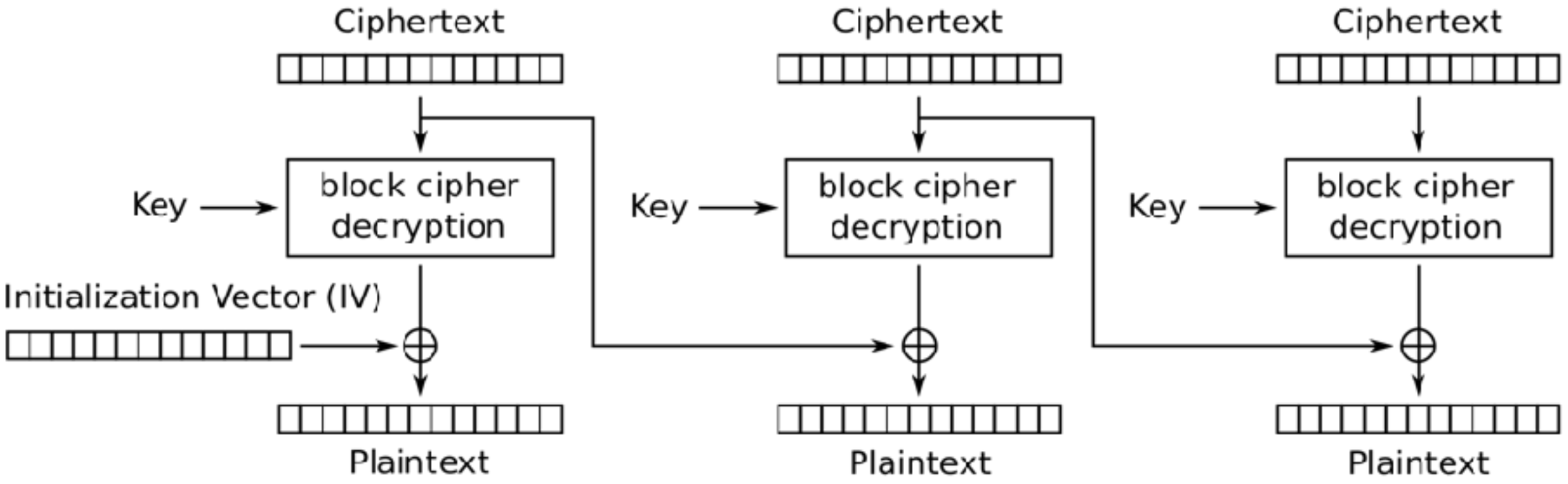
Electronic Codebook (ECB) mode decryption

CBC Mode

- Cipher Block Chaining
- Overcomes the problem with ECB
- XOR the plaintext with the prior ciphertext block
- First block must use an Initialization Vector because there is no prior block



Cipher Block Chaining (CBC) mode encryption



Cipher Block Chaining (CBC) mode decryption

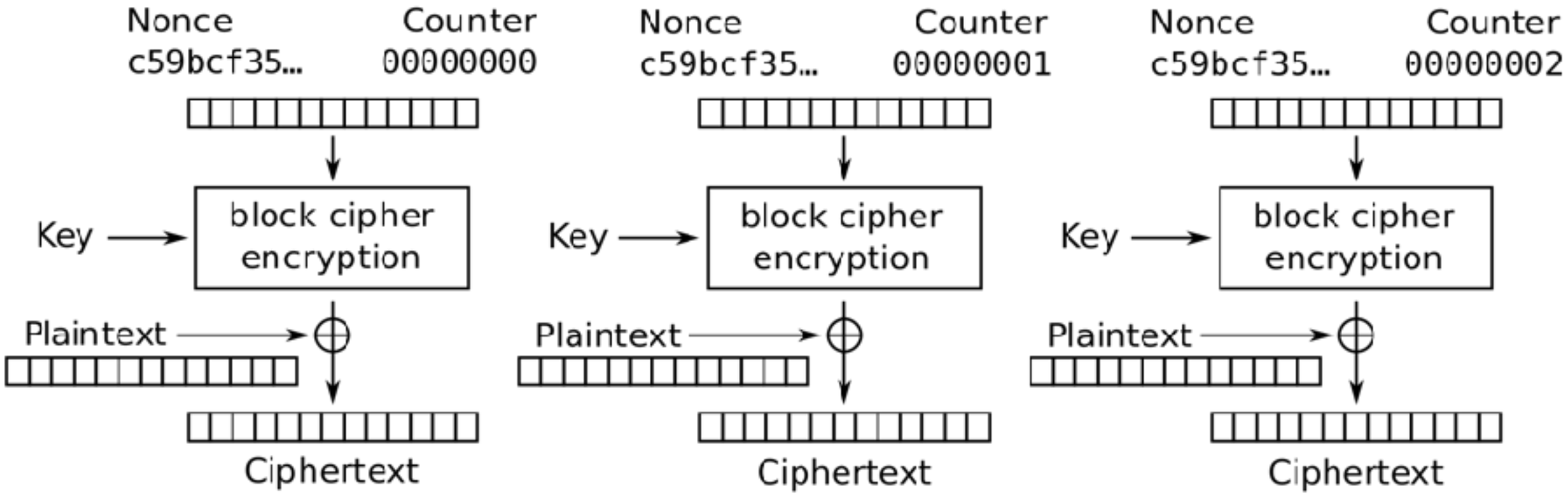
Initialization Vector (IV)

- Must be known to both the sender and recipient
- Must be random and unpredictable (not drawn from a distribution)
- May be public
  - Sometimes encrypted anyway using ECB
- Most importantly, an IV should never be reused with the same key. Why?
  - What happens if we have two plaintext message with the same prefix? How will the ciphertexts compare?

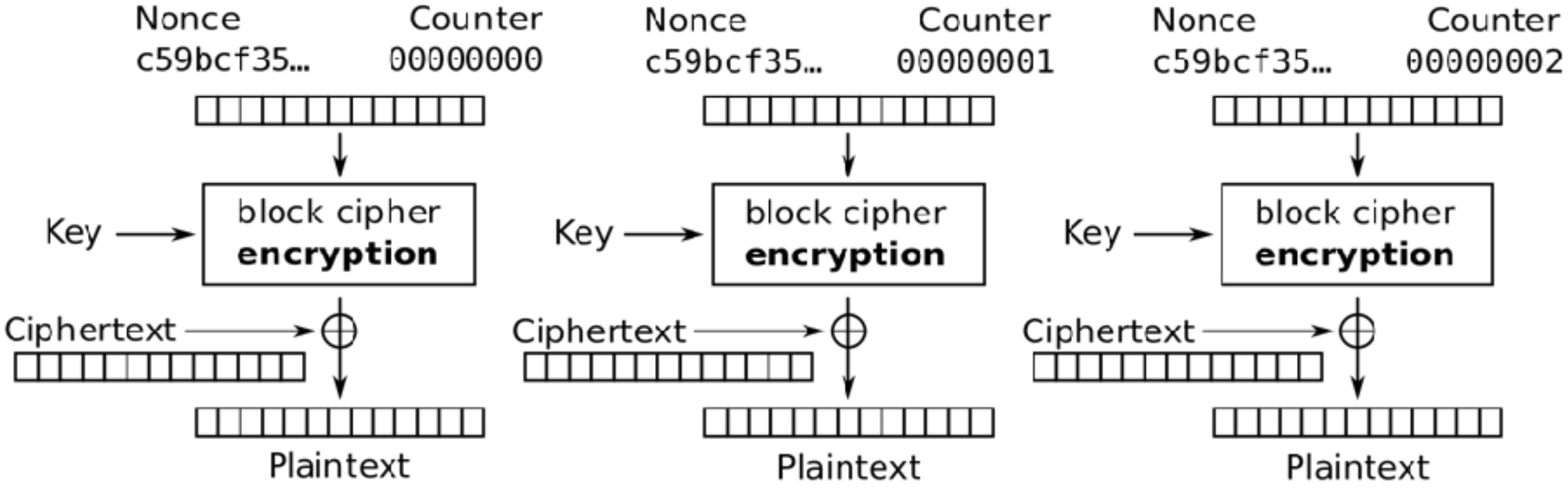
Block Cipher as a Stream Cipher

- The following modes create a stream cipher from a block cipher. How is it done?

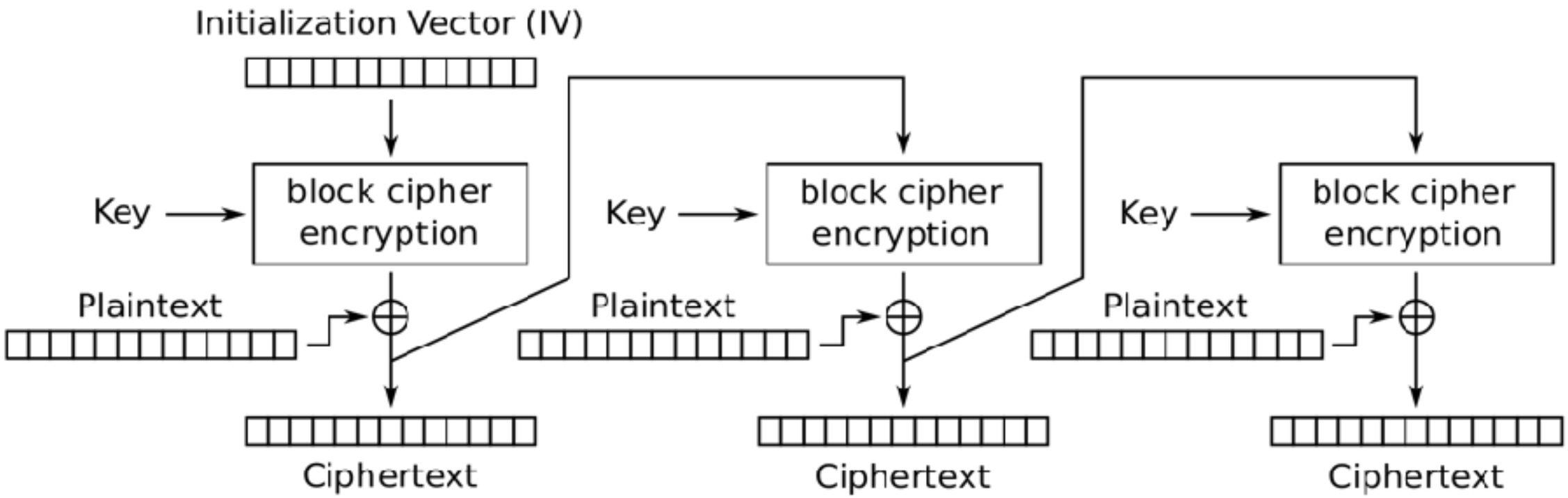
- Three modes
  - Counter Mode (CTR)
  - Cipher Feedback Mode (CFB)
  - Output Feedback Mode (OFB)



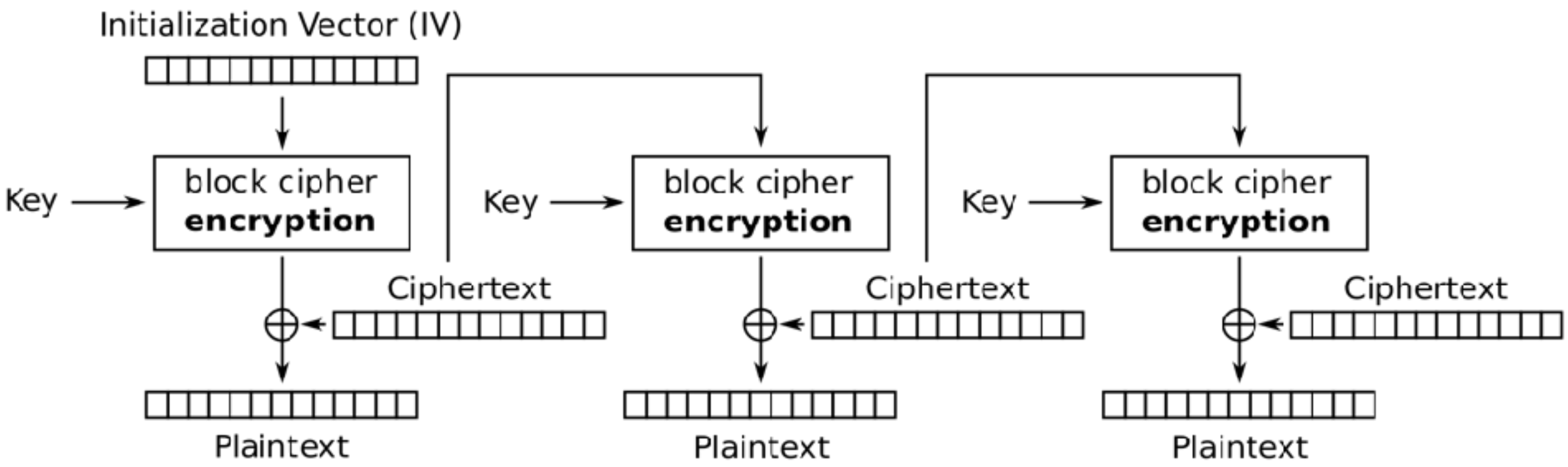
Counter (CTR) mode encryption



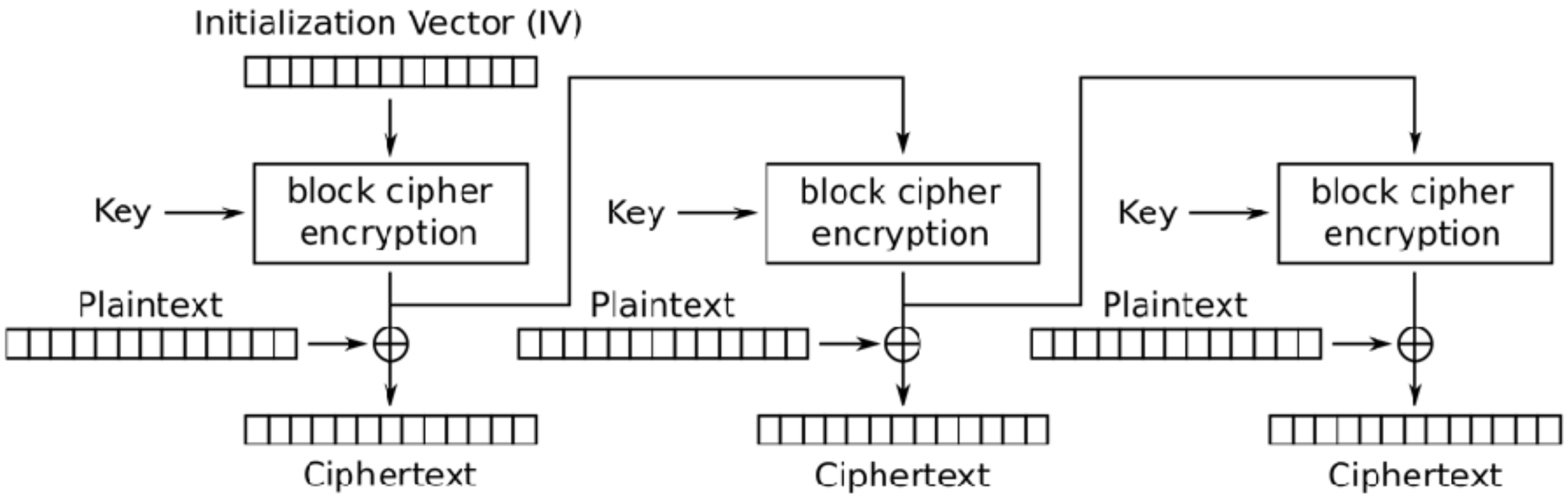
Counter (CTR) mode decryption



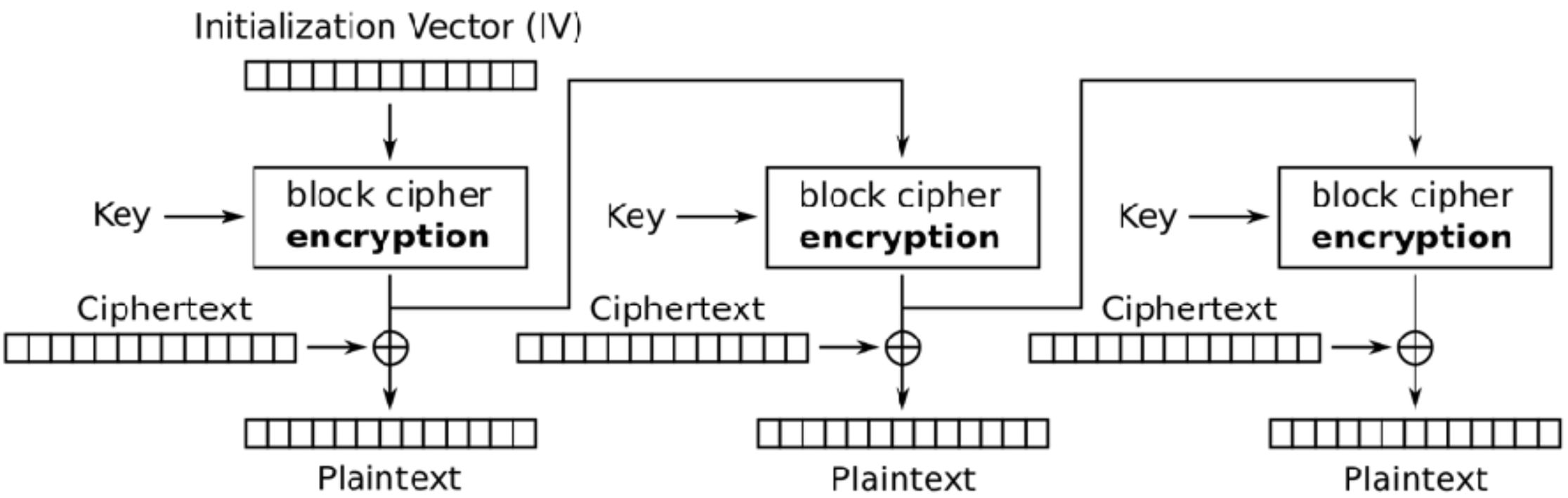
Cipher Feedback (CFB) mode encryption



Cipher Feedback (CFB) mode decryption



Output Feedback (OFB) mode encryption



Output Feedback (OFB) mode decryption

Homework 3

- Comparison of these modes
- Summative – we will grade for correctness
- Contrived scenario to illustrate why the IV should not be reused

Summary

- ECB
  - Simple
  - Parallel encryption/decryption
  - Reveals patterns in the plaintext – should not use
- CBC
  - Conceals plaintext patterns
  - Requires sequential encryption
  - Parallel decryption

Summary

- Block cipher as stream cipher
  - No need for padding
  - Only have to implement encrypt function
- CTR
  - Parallel encryption/decryption
  - Preprocessing able to generate the keystream in advance

Summary

- CFB
  - Parallel decryption
- OFB
  - Preprocessing able to generate the keystream in advance

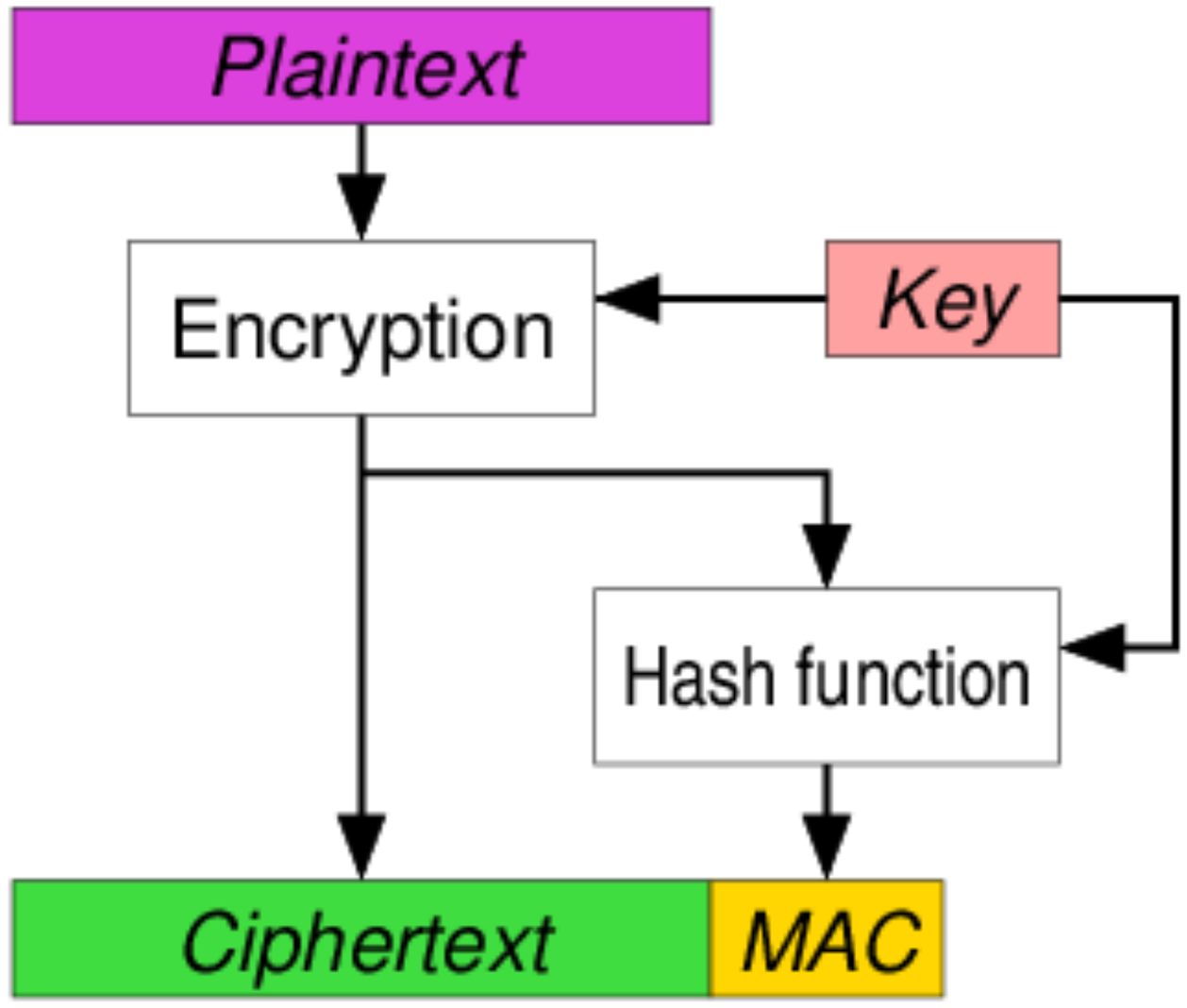
Authenticated Encryption Modes

Motivation

- Symmetric encryption offers confidentiality, but not integrity and authenticity
  - HW 3 – bit flipping attacks
- Securely combining separate confidentiality and authentication block cipher operation modes can be error prone and difficult
- Authenticated Encryption with Associated Data (AEAD) — confidentiality, integrity, and authentication with associated data (e.g. message header)
  - EAX Mode
    https://en.wikipedia.org/wiki/EAX_mode
  - GCM (Galois Counter Mode)
    https://en.wikipedia.org/wiki/Galois/Counter_Mode

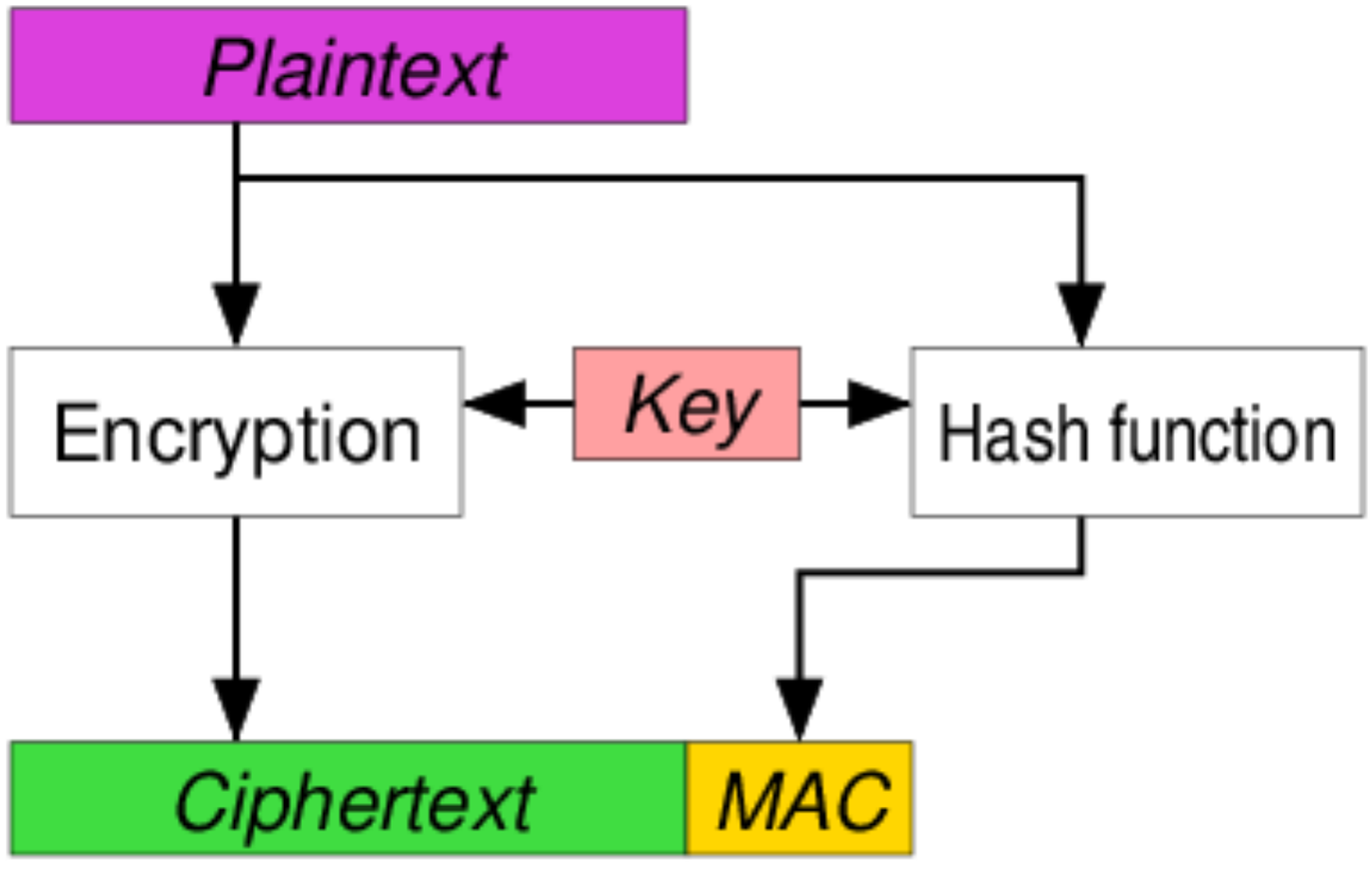
Encrypt-then-MAC (EtM)

- IPSec
- TLS/DTLS
- SSHv2



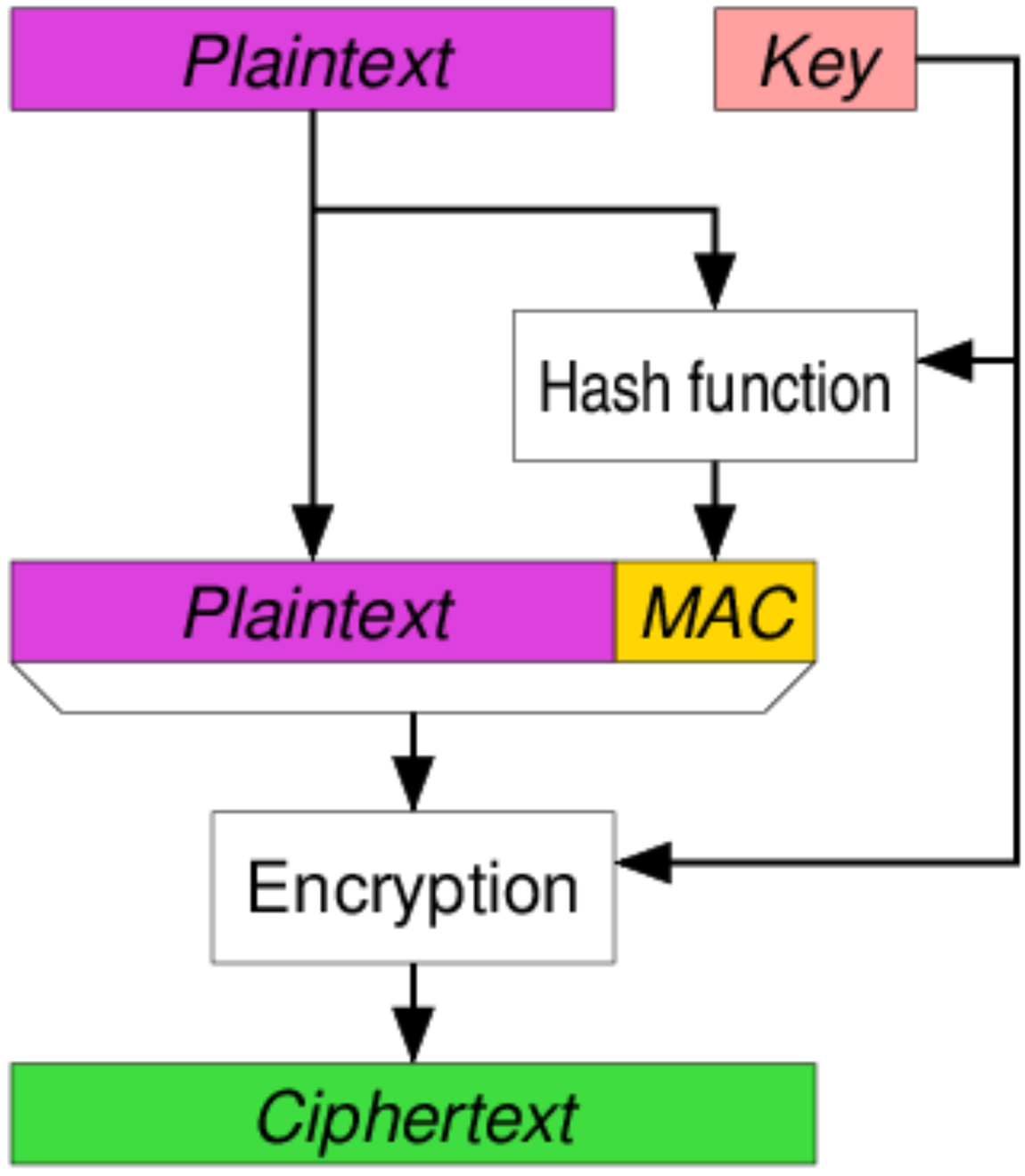
Encrypt-and-MAC (E&M)

- SSH



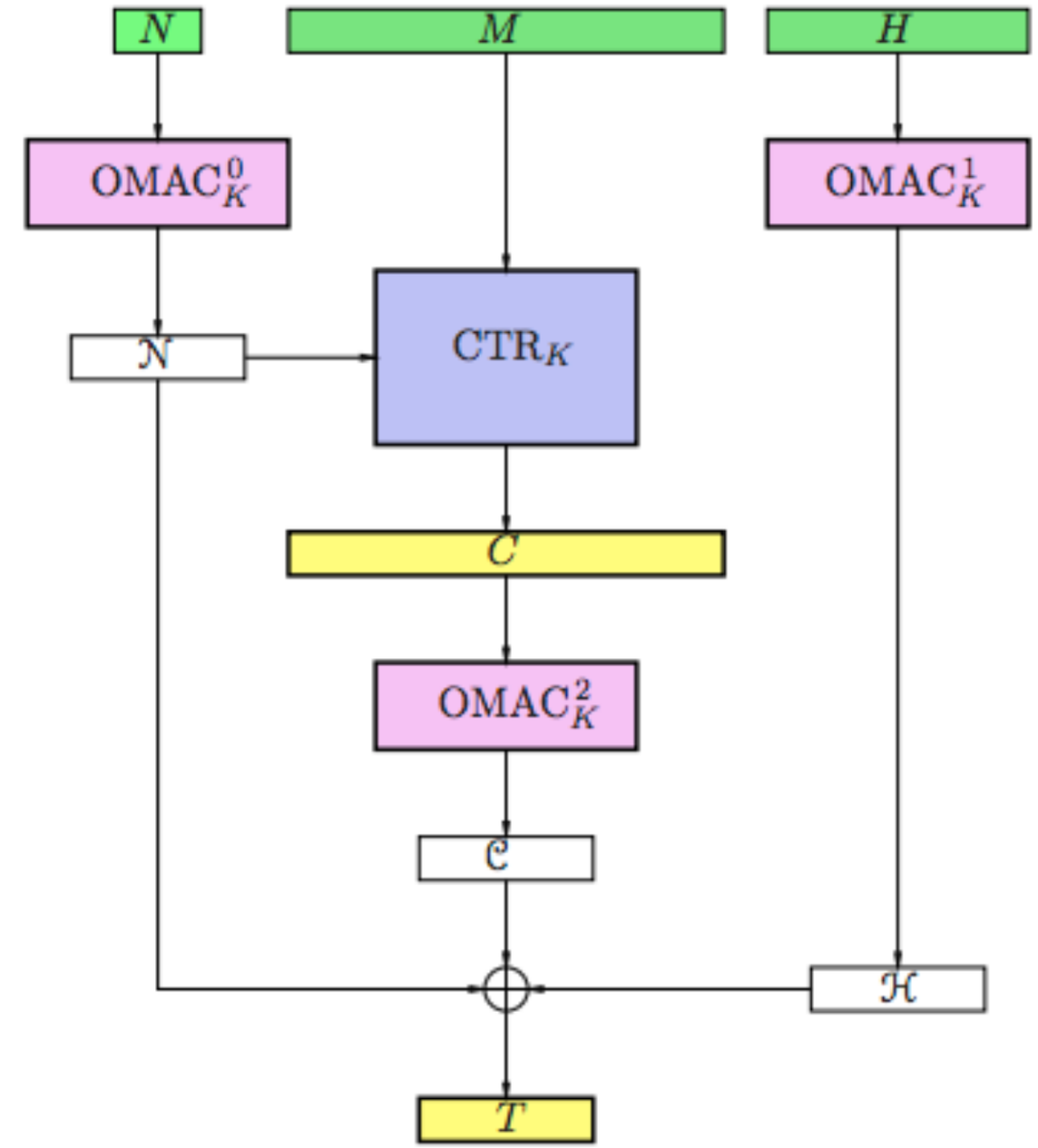
MAC-then-Encrypt (MtE)

- SSL/TLS



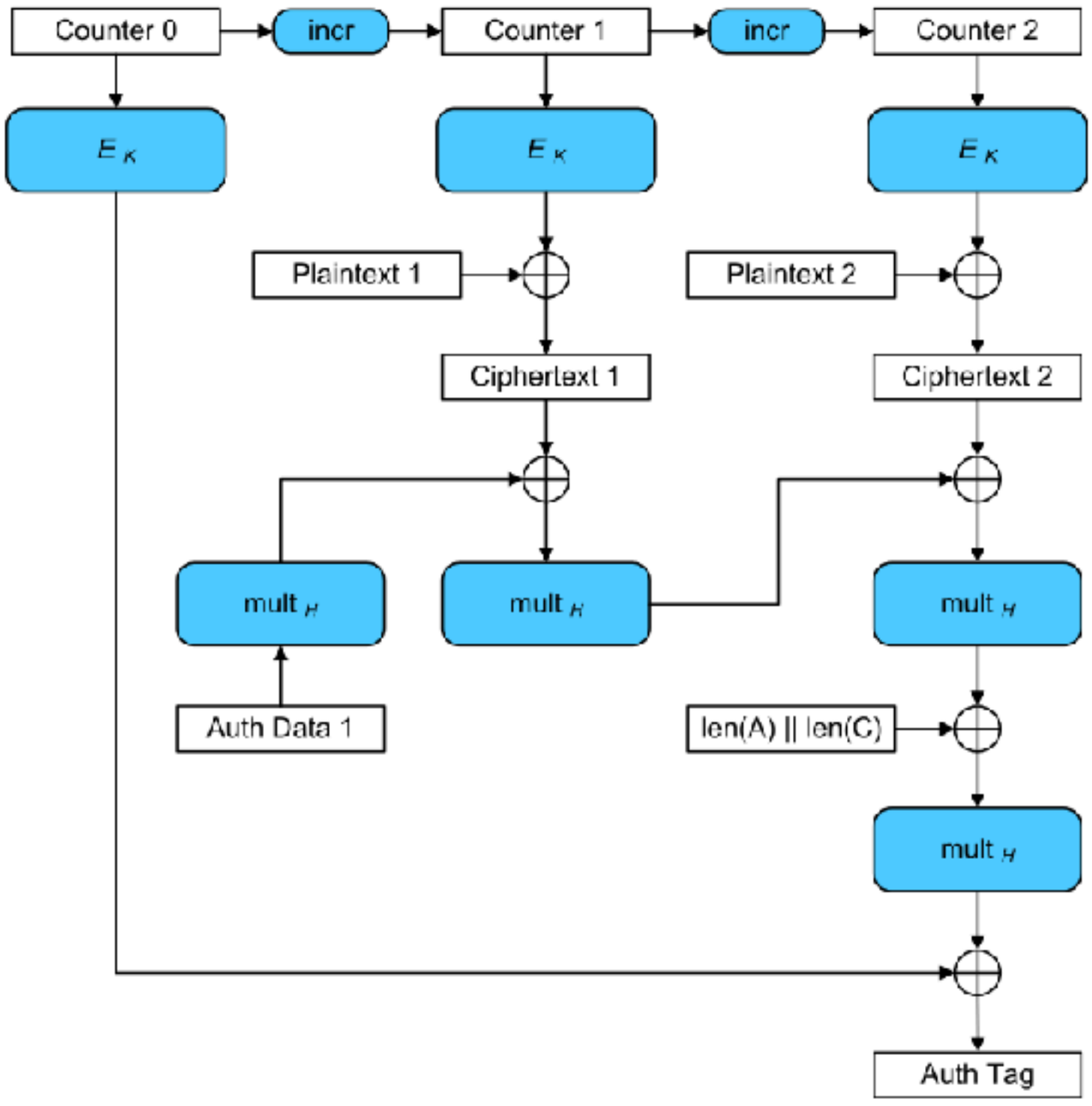
EAX Mode

- two-pass: encryption and then authentication in separate operatio
- Message M, Nonce N, Header H
- OMAC - one-key MAC
- CTR - encryption mode
- When receiving H, C, T, run C through $OMAC^2_K$ and XOR with OMAC off N, H to get T'
  - compare T' to T



GCM Mode

- TLS, SSH, IPSec, OpenVPN
- $mult_H$ is finite field multiplication by hash key H using GF(2^{128})
- fast, patent-free, on-line (don't need to know message length in advance), can be parallelized
- security depends on choosing a unique initialization vector for every encryption performed with the same key



Padding

Block Ciphers & Padding

- Block ciphers require that the plaintext be a multiple of the block size (ECB and CBC modes)
- Padding is used to make sure that all blocks are “full”
- Both sides need to know the padding scheme

Padding Schemes

- Pad with bytes all of the same value as the number of padding bytes
- Pad with 0x80 followed by 0x00 bytes
- Pad with 0x00 characters, followed by a byte equal to the number of padding bytes
- Pad with 0x00 characters or spaces
  - Assuming these values don't appear at the end of the actual data
- Short one-block messages in ECB mode will all encrypt the same with the same key – use random padding

See http://www.di-mgt.com.au/cryptopad.html

Other Uses for Padding?

- Disguise identical messages
  - Identical messages encrypted with the same key will always produce the same ciphertext (assumes no IV, such as ECB mode)
- Disguise message length
  - Pad the message with a random number of bytes to create a random-sized messages
  - All messages are padded to a preset length
- When is padding not required?
  - When the plaintext is always a multiple of the block size and both sides agree not to include padding