SIGPwny

General SP2026 • 2025-02-01

Cryptography III

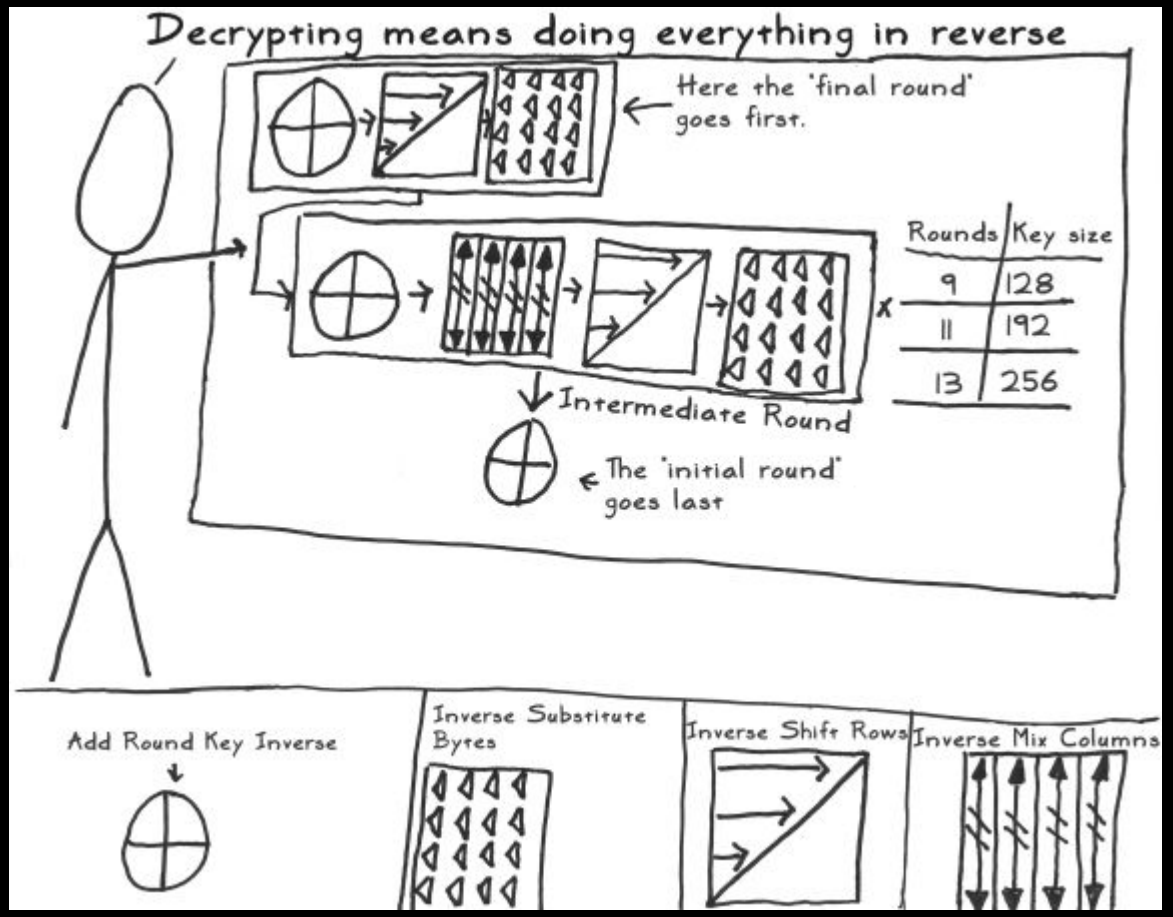
Ahmad Alkhalawi

Announcements

- There were some major SIGPwny infra changes last Friday!
  - Please let us know if any challenges or sites are no longer working.

ctf.sigpwny.com

sigpwny{5ub_5h1ft_4dd_r3p34t}



Symmetric Cryptography

- Symmetric Cryptography is a method where the same secret key is used for both encrypting plaintext and decrypting ciphertext
- To transport that key, we use an asymmetric algorithm like DHKE
- Symmetric algorithms are typically much faster

One-Time Pad

```
>>> plain = b"Test"
>>> key = bytes.fromhex("cafebabe")
>>> bytes([i ^ j for i, j in zip(key, plain)])
b'\x9e\x9b\xc9\xca'
```

- Achieves “perfect secrecy”! 🥳
- Requires a completely random bitstring the same length of your plaintext

Block Ciphers

- A deterministic encryption algorithm that operates on a plaintext of fixed length
- Typically, the plaintext is divided into “blocks” of 16 or 32 bytes instead of treated as a continuous stream of byte data
- An algorithm is used to transform each block into an enciphered block, and then the results are joined together to form a ciphertext

Block Ciphers

```
>>> plain = b"TestTest"
>>> key = bytes.fromhex("cafebabe")
>>> num_blocks = len(plain) // len(key)
>>> bytes([i ^ j for i, j in zip(key*num_blocks, plain)])
b'\x9e\x9b\xc9\xca\x9e\x9b\xc9\xca'
```

- One way to achieve this is to xor a fixed size key with each block
- But this is weak now

Diffusion and Confusion

- To give a description of what a secure block cipher looks like, the cryptographer Shannon introduced Diffusion and Confusion.
- Confusion is making the relation between the key and ciphertext as complicated as possible.
- Diffusion is spreading the influence of one plaintext symbol over many ciphertext symbols with the goal of hiding statistical properties of the plaintext.
- The previous XOR cipher has low diffusion and confusion

AES

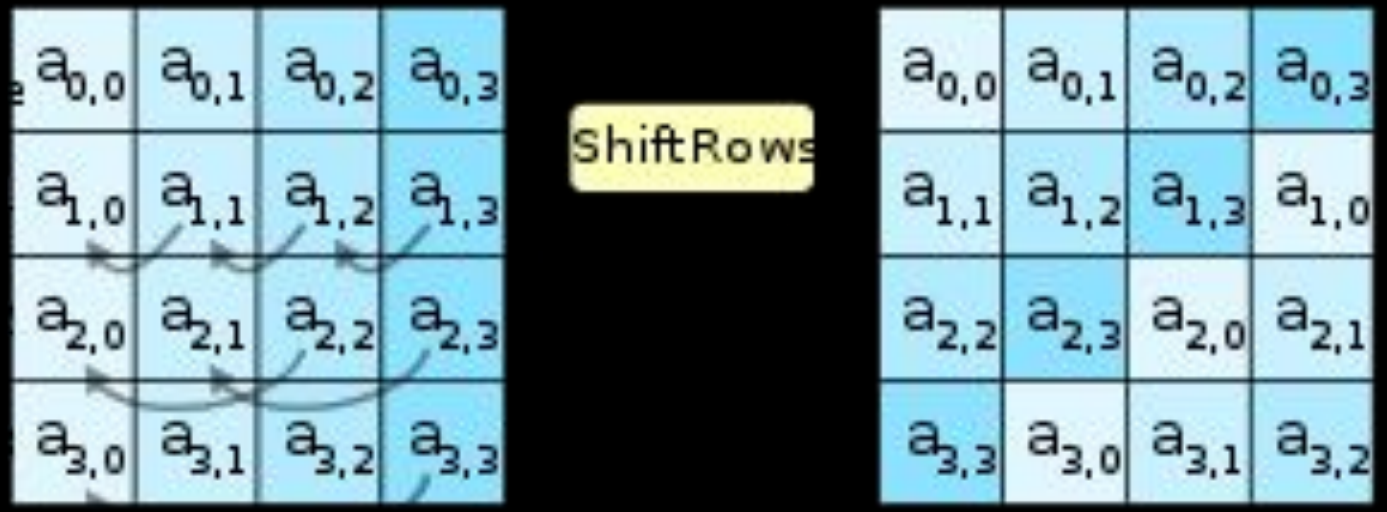
- Block cipher that operates on a fixed block length of 16 bytes (128 bits)

- There are a total of 3 different bit lengths for the keys: AES-128, AES-192, AES-256

- AES begins by splitting the key into round keys according the key schedule, representing each block as a 4x4 matrix, then performing the following operations:

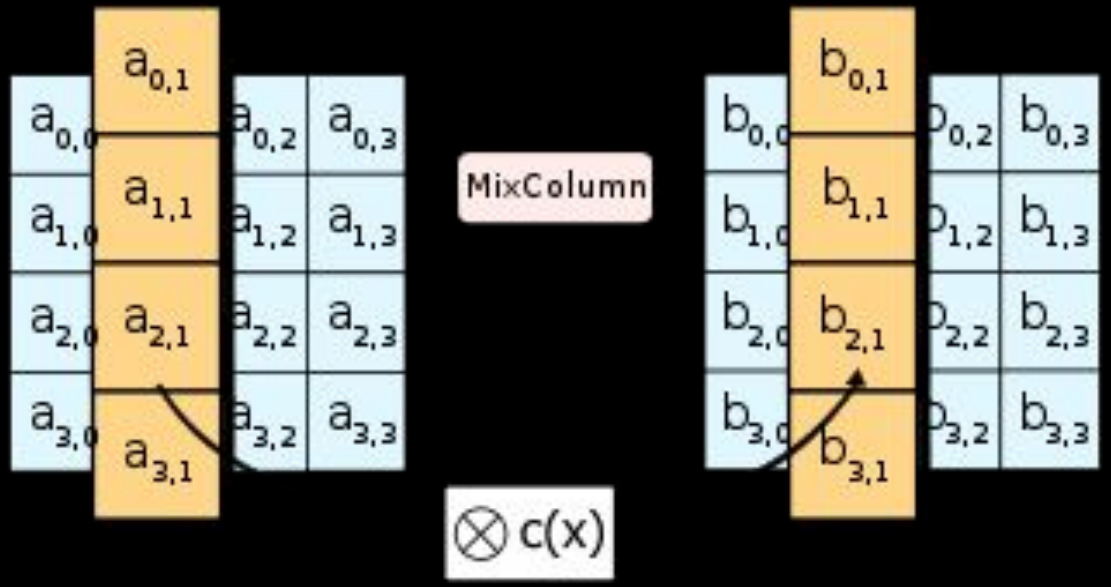
AES

- ShiftRows shifts the rows of the current block by a certain offset amount, providing diffusion in the vertical direction.



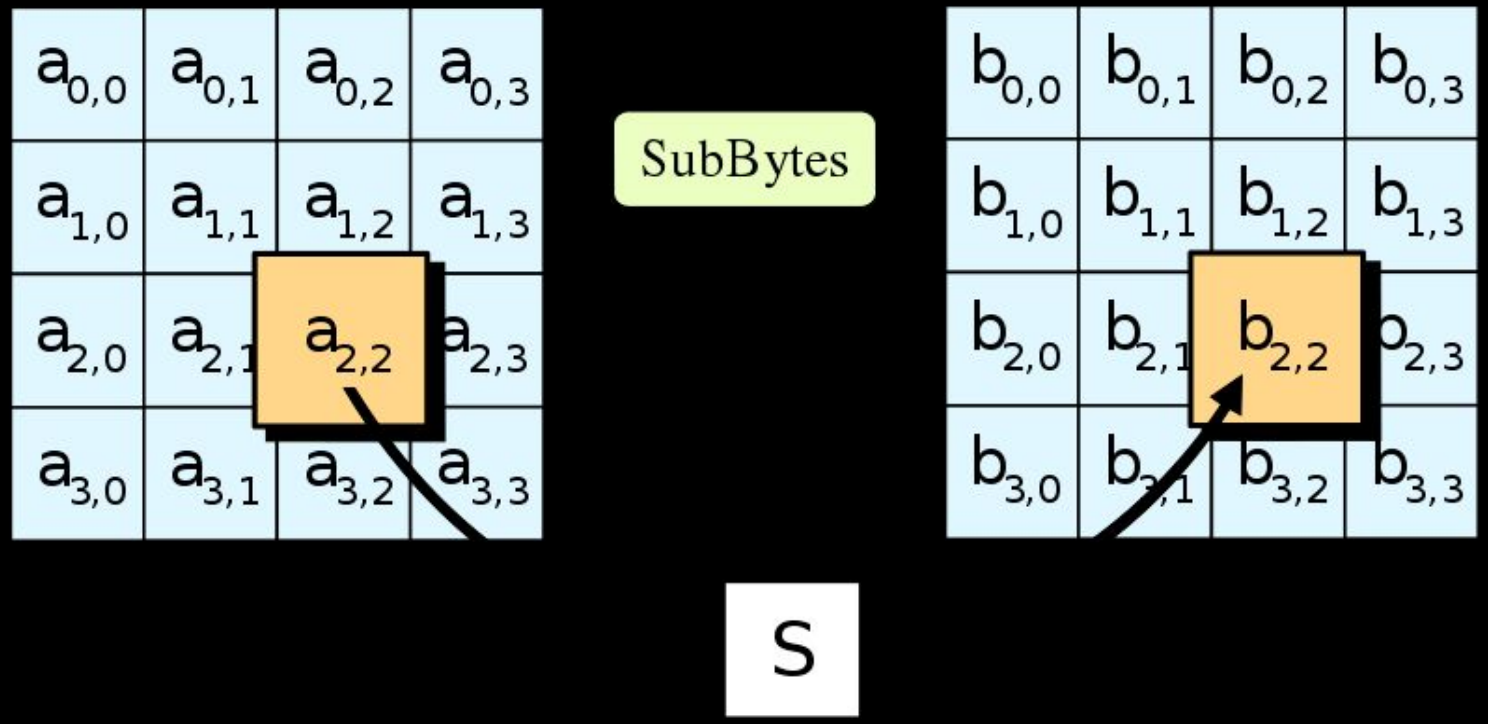
AES

- MixColumns applies a matrix multiplication operation to each column of the current block, providing diffusion in the horizontal direction.



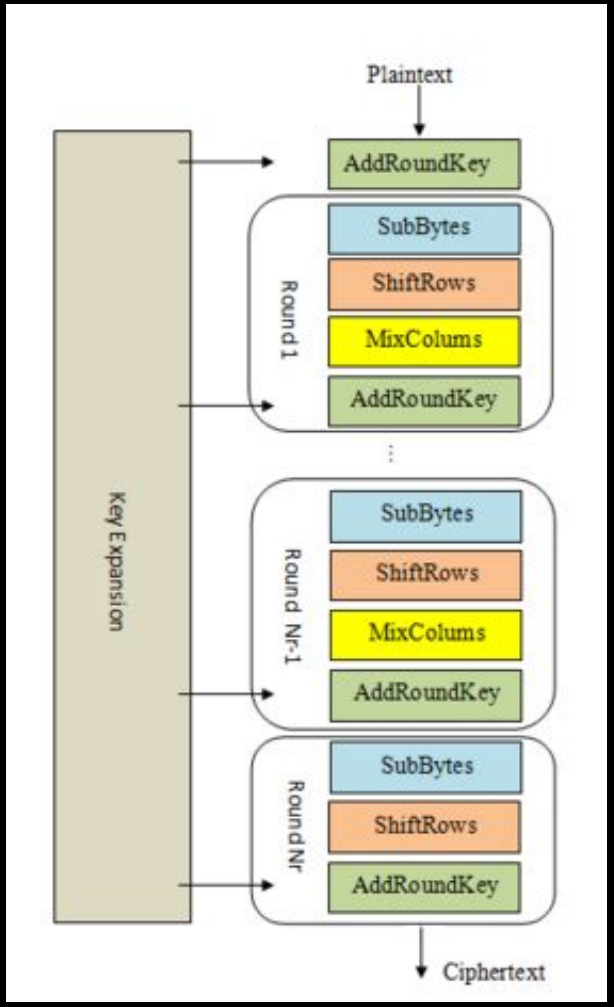
AES

- SubBytes uses a global substitution lookup table called the SBOX to substitute a set of bits in the current block, adding nonlinearity (confusion) to the encryption



AES

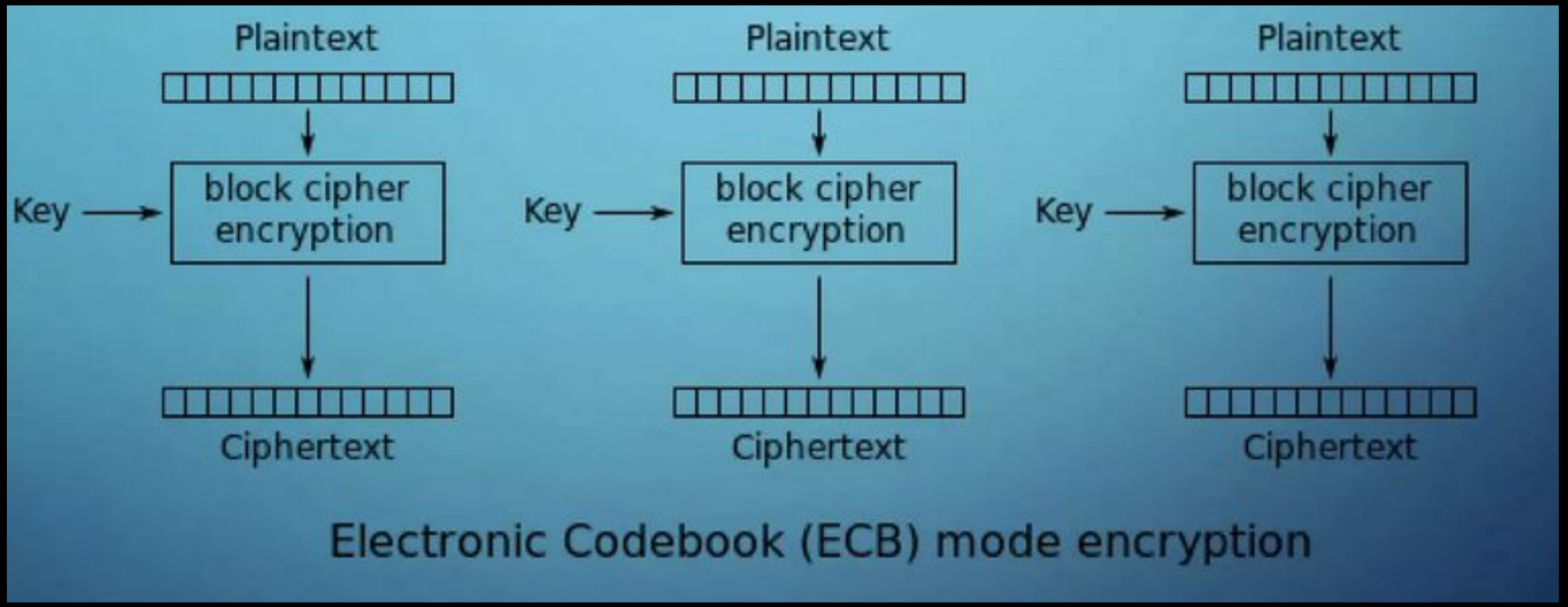
- AddRoundKey performs a bitwise XOR operation between the current block and a round key derived from the cipher's key schedule.



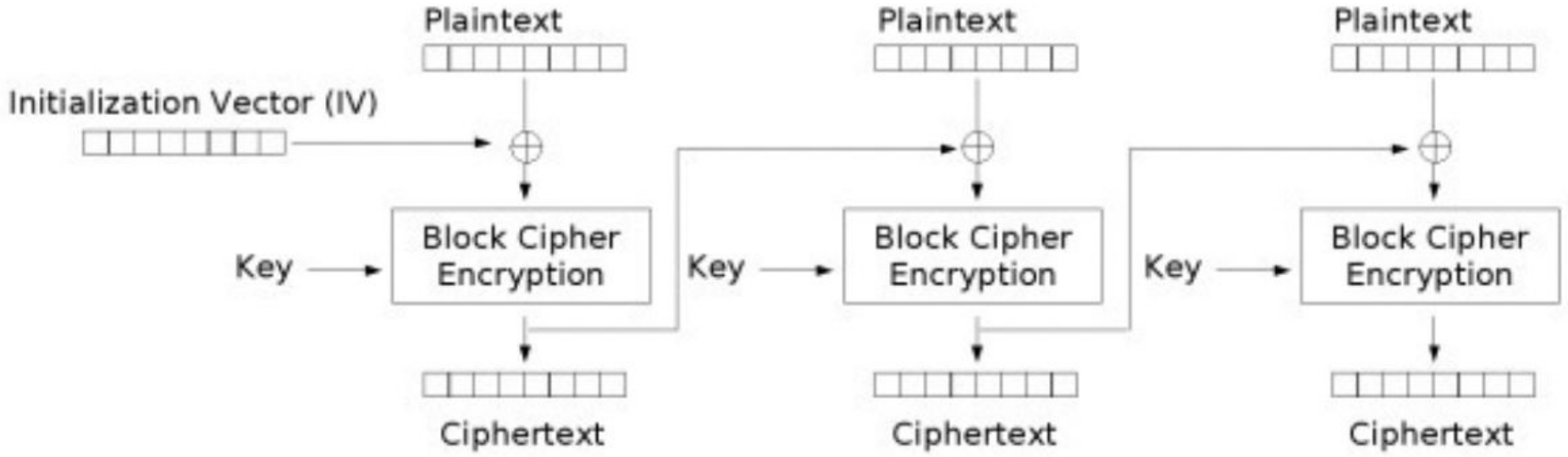
Cipher Modes

- Block ciphers like AES often have different *modes* of encryption governing how each block is encrypted: the algorithm itself is only good enough to encrypt one block of text, so we need to extend it to work on multiple blocks
- Example modes for AES: ECB, OFB, CTR, CBC, CFB

Cipher Modes - ECB

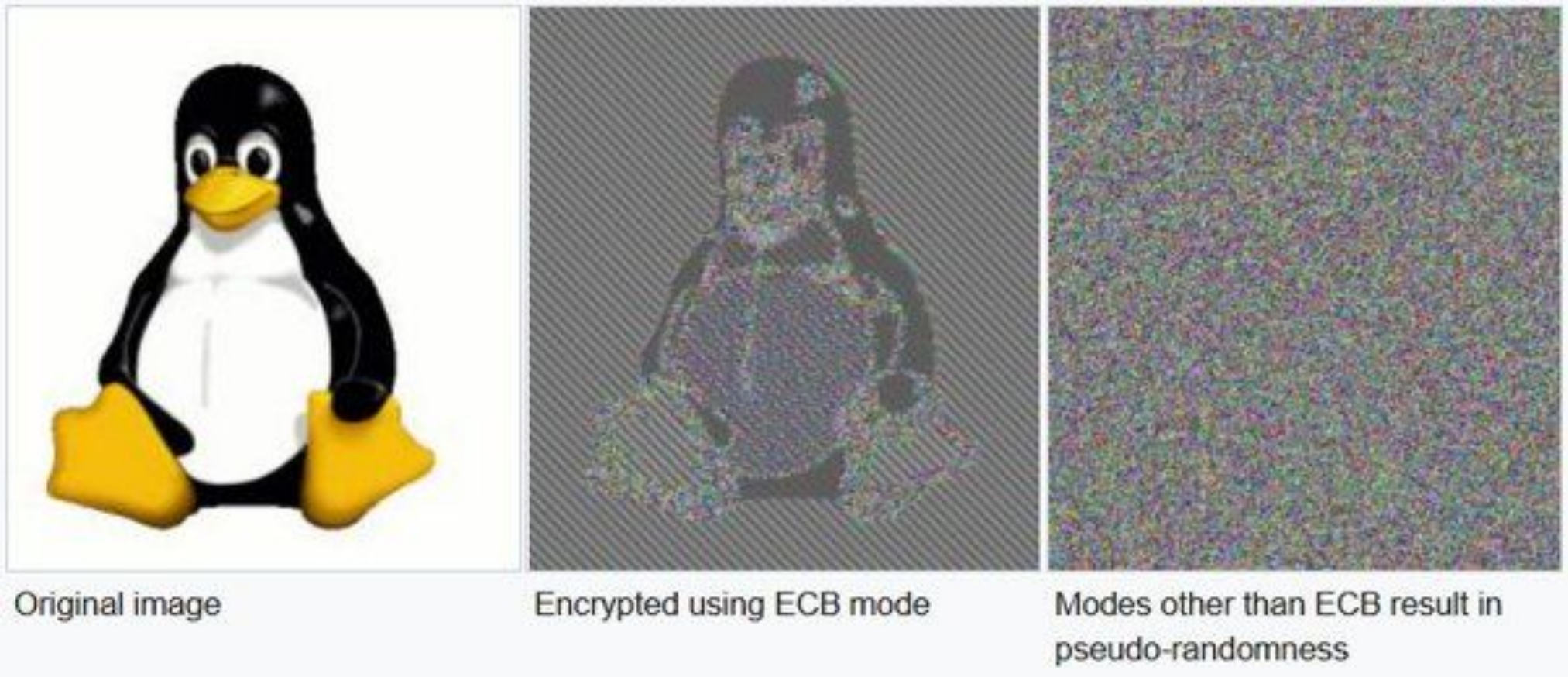


Cipher Modes - CBC



Cipher Block Chaining (CBC) mode encryption

Why we need modes



Original image

Encrypted using ECB mode

Modes other than ECB result in pseudo-randomness

Affine Transforms

- transformations upon a vector of the form $\boldsymbol{Ax + b}$
  - In the context of AES, this is represented by $\boldsymbol{y = A \cdot x \oplus b}$
- The transformation first mixes the bits linearly (via matrix A) and then shifts them (via vector b).
- Importantly, if A happens to be reversible, then it is possible to recover x from y: $\boldsymbol{x = A^{-1}(y \oplus b)}$
- Affine transformations are important as they allow efficient mixing
- However, being purely linear has its problems!!!

Linear Cryptanalysis

- Secure S-Boxes in block ciphers are designed to be resistant towards 2 kinds of cryptanalysis: linear and differential
  - If an S-Box is linear, the output bitvector ***y*** of the substitution can be expressed as the bitwise XOR-sum of some linear combination of the input bitvector ***x***
  - Basically, there exist some vector ***b*** and some matrix in GF(2) **A** such that the output bitvector
    - $y = A \cdot x \oplus b$
  - **If this is the case, then we can possibly represent the AES/DES encryption as an affine transformation!!!**

Pseudorandomness

- We say a sequence of symbols is **pseudorandom** if it seems to look completely random yet has been created by a deterministic, completely repeatable process

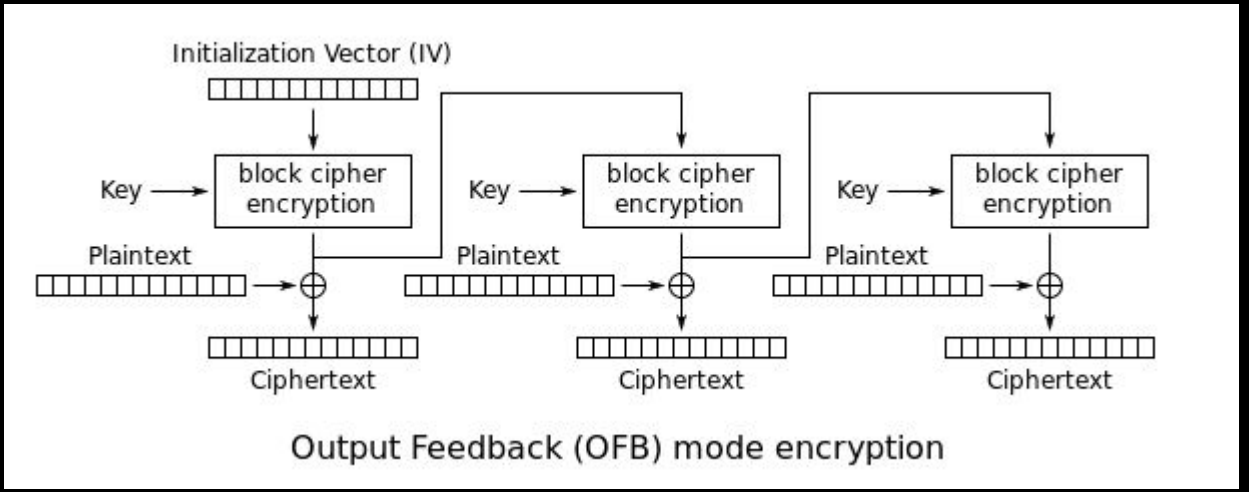
- A **PRG** $G : \{0,1\}^n \rightarrow \{0,1\}^{n+s}$ is a mapping such that it is very hard for any polynomial-time “guesser” to guess the output of the PRG given the input string

Stream Cipher

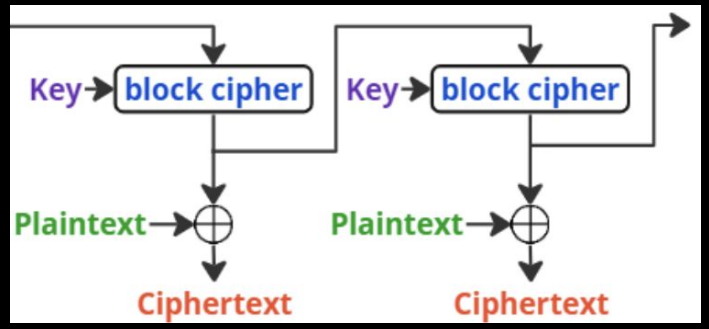
- A probabilistic encryption algorithm building on top of PRGs where the cryptographic key and algorithm are applied to each binary digit in an input (treated as a data stream)
- The key supplied as input into the PRG is known as the **keystream**
- General Example:
  - Calculate keystream with some random IV: $G(iv, k)$
  - Encrypt message (byte or bit level) $m \in \{0,1\}^{n+s}$: $c = (iv, G(iv, k) \oplus m)$
  - Decrypt with $m = G(k, iv) \oplus c$, discarding IV

Examples

- AES modes such as CTR, GCM, OFB, and CFB
- ChaCha20 : very popular used, low power stream cipher
- Rivest RC4: example of an insecure stream cipher, especially when you don’t discard beginning of keystream
- Chameleon, Fish, Helix
- many more



Bit Flip (OFB)



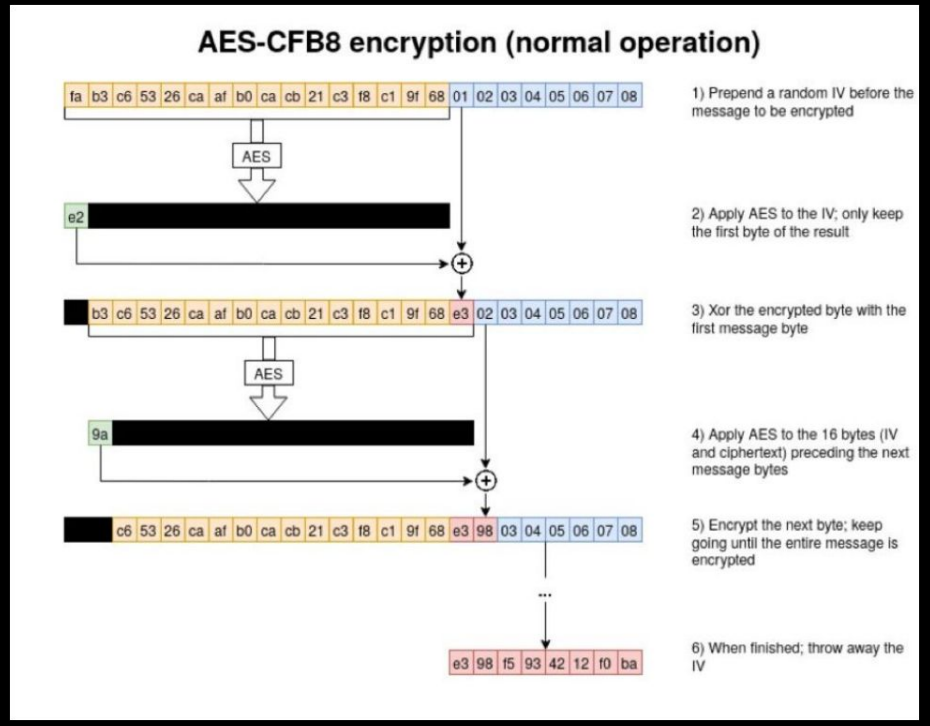
- **Each block of plaintext is XORed with the previous block of ciphertext before being encrypted**

- Thus, if an attacker modifies a bit in the ciphertext of one block, the corresponding bit in the decrypted plaintext of the next block will be flipped

- The alternative CFB mode will flip the bit in the same block like OFB, except it will also clobber the next block, making it easier to authenticate the decrypted message and detect the bit flip attack

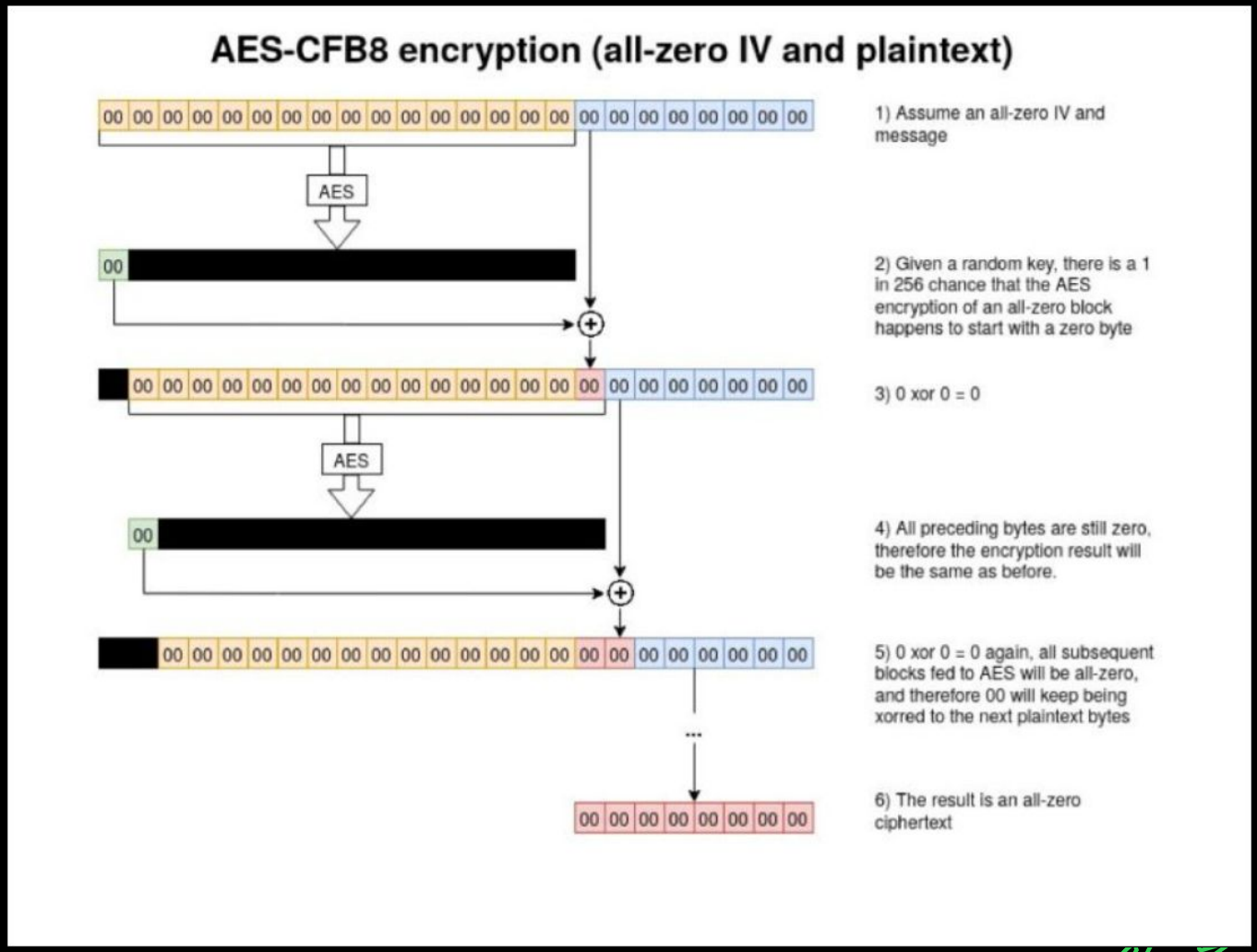
Zerologon

Zerologon is a vulnerability discovered in 2020 in the cryptography of Microsoft's Netlogon process that allows an attack against Microsoft Active Directory domain controllers.



Zerologon

The attack is based on Microsoft choosing all 0s as the IV. This means there is a 1/256 chance of the server computing all 0s from the cipher, which means you can predict what it will say and authenticate.



Next Meetings

2026-02-05 • **This Thursday**

- Java Rev
- Learn how to decompile and reverse engineer Java programs!

2026-02-08 • **Next Sunday**

- PWN III: ROP
- Learn how to bypass W^X protections with code reuse attacks!

ctf.sigpwny.com

`sigpwny{5ub_5h1ft_4dd_r3p34t}`

Meeting content can be found at sigpwny.com/meetings.

SIGPwny