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
- What about the first block?
Cipher Block Chaining (CBC) mode encryption

Cipher Block Chaining (CBC) mode decryption
Figure 2.7 Cipher Block Chaining (CBC) Mode
Initialization Vector (IV)

- Must be known to both the sender and recipient
- Ideally both IV and key should protected, but the IV may be public
- Common approach: encrypt IV using ECB and send it with the encrypted data
- Most importantly, an IV should never be reused with the same key. Why?
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
Summary

- **ECB**
  - Simple
  - Don’t have to create/manage an IV
  - 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
  o No need for padding
  o Only have to implement encrypt function

• CTR
  o Preprocessing able to generate the keystream in advance
  o Parallel encryption/decryption

• CFB
  o Parallel decryption

• OFB
  o Preprocessing able to generate the keystream in advance
Authenticated Encryption Modes
• Symmetric encryption offers confidentiality, but not integrity and authenticity
  o HW 3 – bit flipping attacks
• More recent block cipher modes offer both
• Authenticated Encryption with Associated Data (AEAD)
  o EAX mode
    https://en.wikipedia.org/wiki/EAX_mode
  o GCM (Galois Counter Mode)
    https://en.wikipedia.org/wiki/Galois/Counter_Mode
**EAX Mode**

**Algorithm EAX.Encrypt\(N^H\) (M)**

10. \(N \leftarrow \text{OMAC}^0_K(N)\)
11. \(\mathcal{E} \leftarrow \text{OMAC}^1_K(H)\)
12. \(C \leftarrow \text{CTR}^N_K(M)\)
13. \(\mathcal{C} \leftarrow \text{OMAC}^2_K(C)\)
14. \(\text{Tag} \leftarrow N \oplus \mathcal{E} \oplus \mathcal{H}\)
15. \(T \leftarrow \text{Tag} [\text{first} \, \tau \, \text{bits}]\)
16. **return** \(CT \leftarrow C \| T\)

**Algorithm EAX.Decrypt\(N^H\) (CT)**

20. **if** \(|CT| < \tau** then return **INVALID**
21. Let \(C \| T \leftarrow CT\) where \(|T| = \tau\)
22. \(N \leftarrow \text{OMAC}^0_K(N)\)
23. \(\mathcal{H} \leftarrow \text{OMAC}^1_K(H)\)
24. \(\mathcal{C} \leftarrow \text{OMAC}^2_K(C)\)
25. \(\text{Tag}' \leftarrow N \oplus \mathcal{E} \oplus \mathcal{H}\)
26. \(T' \leftarrow \text{Tag}' [\text{first} \, \tau \, \text{bits}]\)
27. **if** \(T \neq T'** then return **INVALID**
28. \(M \leftarrow C^{\tau}_K(C)\)
29. **return** \(M\)
GCM Mode

Counter 0 → incr → Counter 1 → incr → Counter 2

$E_K$

Plaintext 1 → $E_K$

Ciphertext 1 → $E_K$

Plaintext 2 → $E_K$

Ciphertext 2

$\text{mult}_H$

Auth Data 1

$\text{mult}_H$

$\text{len(A)} \parallel \text{len(C)}$

$\text{mult}_H$

Auth Tag
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 characters
• Pad with 0x00 characters
  o Last byte is equal to the number of padding bytes
  o Pad with spaces
• Pad with 0x00 characters or spaces
  o 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

Other Uses for Padding?

• **Disguise identical messages**
  - Identical messages encrypted with the same key will always produce the same ciphertext

• **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