# Kriptografi Modern

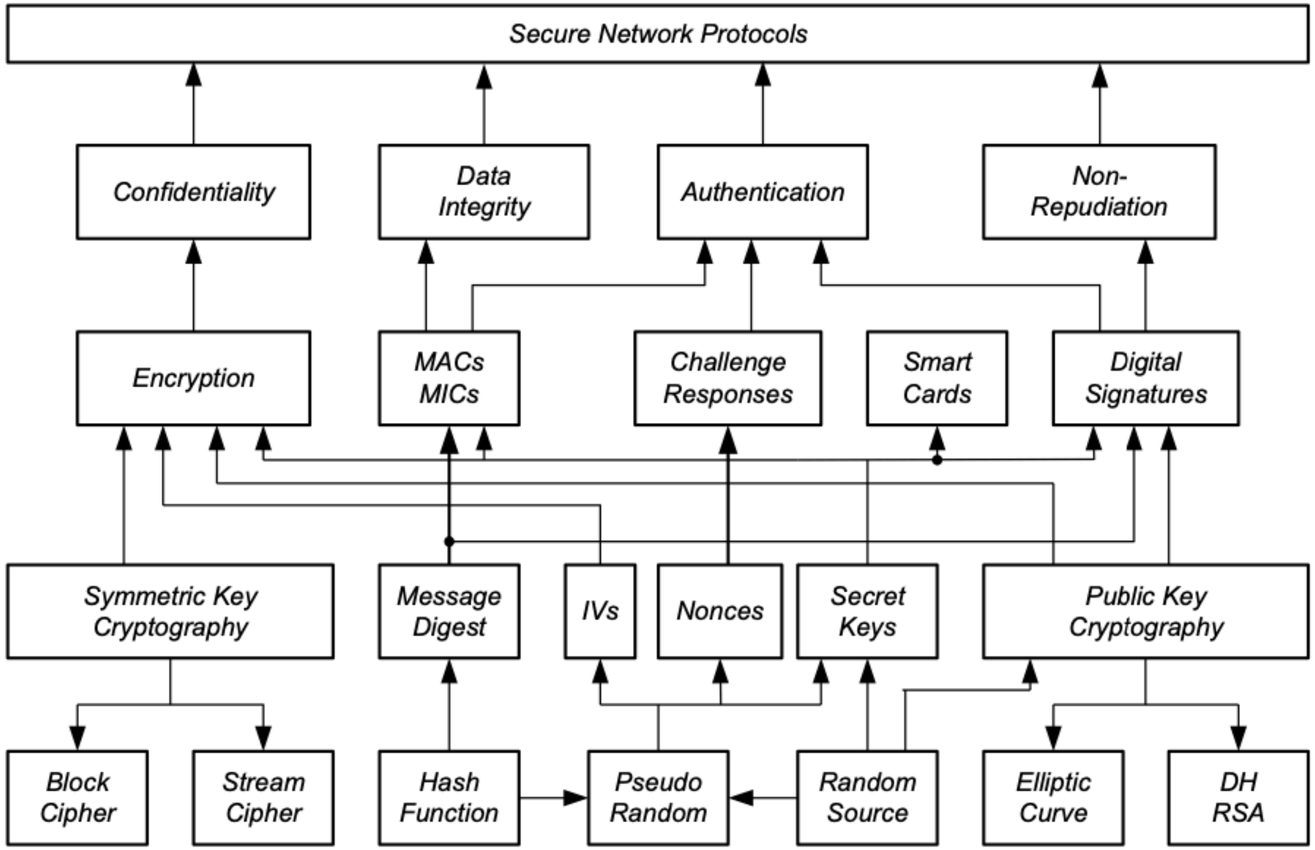
## Bahan Kuliah Keamanan Data

Sevi **Nurafni**

**Fakultas Sains dan Teknologi**
**Universitas Koperasi Indonesia 2025**

- Beroperasi dalam mode bit atau byte
    - Kunci, plainteks, chiperteks, diproses dalam rangkaian bit/byte
    - Operasi bit **xor** paling banyak digunakan
- Tetap menggunakan teknik pada algoritma klasik: substitusi dan transposisi, tetapi lebih kompleks
- Perkembangan algoritma kripstografi modern didorong oleh penggunaan komputer digital untuk keamanan pesan
- Komputer digital merepresentasikan data dalam bine

# Diagram Blok Kriptografi Modern



# Rangkaian Bit

- Pesan dalam bentuk rangkaian bit dipecah menjadi beberapa blok
- Contoh: Plainteks `100111010110`

  Bila dibagi menjadi blok 4-bit

  `1001  1101  0110`

  Maka setiap blok menyatakan 0 sampai 15

  `9  13  6`

# Rangkaian Bit

Bila plainteks dibagi menjadi blok 3-bit:

```
100 111   010   110
```

Maka setiap blok menyampaikan 0 sampai 7:

```
4     7     2     6
```

# Rangkaian Bit

Padding bits: bit-bit tambahan jika ukuan blok terakhir tidak mencukupi panjang blok

Contoh: Plainteks 100111010110

Jika dibagi menjadi blok 5-bit

`10011  10101  `**`000`**`10`

Padding bits mengakibatkan ukuran cipherteks sedikit lebih besar daripada ukuran plainteks semula

# Representasi dalam Heksadesimal

Pada beberapa algoritma kriptografi, pesan dinyatakan dalam kode Hex:

0000 = 0  0001=1  0010=2  0011=3

0100 = 4  0101=5  0110=6  0111=7

1000 = 8  1001=9  1010=A  1011=B

1100 = C  1101=D  1110=E  1111=F

Contoh: Plainteks `100111010110`

Bila dibagi menjadi blok 4-bit

`1001  1101  0110`

Dalam notasi Hex adalah `9 D 6`

# Operasi *XOR*

- Paling banyak digunakan di dalam cipher modern
- Notasi: $\oplus$
- Operasi

$$0 \oplus 0 = 0 \quad 0 \oplus 1 = 1$$

$$1 \oplus 0 = 1 \quad 1 \oplus 1 = 0$$

- Operasi XOR = penjumlahan modulo 2:

$$0 \oplus 0 = 0 \leftrightarrow 0 + 0 \pmod 2 = 0$$

$$0 \oplus 1 = 1 \leftrightarrow 0 + 1 \pmod 2 = 1$$

$$1 \oplus 0 = 1 \leftrightarrow 1 + 0 \pmod 2 = 1$$

$$1 \oplus 1 = 0 \leftrightarrow 1 + 1 \pmod 2 = 0$$

# Operasi *XOR*

- Hukum-hukum yang terkait dengan operator XOR:
  - $a \oplus a = 0$
  - $a \oplus b = b \oplus a$
  - $a \oplus (b \oplus c) = (a \oplus b) \oplus c$

# Operasi *XOR Bitwise*

- Jika dua rangkaian dioperasikan dengan XOR, maka operasinya dilakukan dengan meng-XOR-kan setiap bit yang berkoresponden dari kedua rangkaian bit tersebut.
- Contoh: $10011 \oplus 11001 = 01010$
- Hasilnya diperoleh sebagai berikut

$$
\begin{array}{ccccccccccc}
1 & & 0 & & 0 & & 1 & & 1 & & \\
1 & & 1 & & 0 & & 0 & & 1 & \oplus & \\
\hline
1 \oplus 1 & & 0 \oplus 1 & & 0 \oplus 0 & & 1 \oplus 0 & & 1 \oplus 1 & &
\end{array}
$$

$$
\begin{array}{ccccccccc}
0 & & 1 & & 0 & & 1 & & 0
\end{array}
$$

# Cipher dengan *XOR*

- Sama seperti Vigenere Cipher, tetapi dalam mode bit
- Setiap bit plainteks di-XOR-kan dengan setiap bit kunci

Enkripsi: $C = P \oplus K$

Dekripsi: $P = C \oplus K$

| | | | | |
|---|---|---|---|---|
| Contoh: | plainteks | 01100101 | | (karakter ‘e’) |
| | kunci | 00110101 | $\oplus$ | (karakter ‘5’) |
| | cipherteks | 01010000 | | (karakter ‘P’) |
| | kunci | 00110101 | $\oplus$ | (karakter ‘5’) |

# Cipher dengan *XOR*

- Jika panjang bit-bit kunci lebih pendek daripada panjang bit-bit pesan, maka bit-bit kunci diulang penggunaannya secara periodik

Contoh:

Plainteks : 10010010101110101010001110001

Kunci : 11011011011011011011011011011

Cipherteks: 01001001110101110001010101010

```python
import base64

def xor_encrypt_decrypt(text, key):
    result = bytearray()
    key_length = len(key)

    for i in range(len(text)):
        result.append(ord(text[i]) ^ ord(key[i % key_length]))

    return result

# Input dari user
text = input("Masukkan teks: ")
key = input("Masukkan kunci: ")

# Enkripsi (Hasil dalam Base64)
cipher_bytes = xor_encrypt_decrypt(text, key)
cipher_b64 = base64.b64encode(cipher_bytes).decode()
print("Hasil Enkripsi (Base64):", cipher_b64)

# Dekripsi
decoded_bytes = base64.b64decode(cipher_b64)
plaintext = xor_encrypt_decrypt(decoded_bytes.decode(), key)
print("Hasil Dekripsi:", plaintext.decode())
```

```
Masukkan teks: aku lagi makan burger
Masukkan kunci: kripto
Hasil Enkripsi (Base64): ChkcUBgODBtJHRUEChxJEgEdDBcb
Hasil Dekripsi: aku lagi makan burger
```

- Sayangnya, algoritma XOR sederhana tidak aman karena cipherteksnya mudah dipecahkan. Panjang kuncinya dapat ditemukan dengan Metode Kasiski.

# Kategori cipher berbasis bit

- Stream Cipher
  - Beroperasi pada bit tunggal
  - Enkripsi/dekripsi bit per bit

- Block Cipher
  - Beroperasi pada blok bit
    (Contoh: 64-bit/blok = 8 karakter/blok)
  - Enkripsi/dekripsi blok per blok

# SELAMAT BELAJAR