COMP 8920: Cryptography Winter 2025

Lecture 2 — Jan 9, 2025

Prof. Curtis Bright *Scribe: Aaron Barnoff*

1 Overview

This lecture covers material from Chapters 1.4 and 2.1.

2 Three Levels of Security

1. **Computational Security:** The best-known algorithms and most powerful computers require an impractical amount of time to break the cryptosystem.
2. **Provable Security:** Breaking the system can be reduced to solving a well-known, difficult mathematical problem (e.g., factoring large numbers).
    - **Reduction:** "A reduces to B" ($A \leq B$ or $A \propto B$) means that given an instance of problem A, we can transform it into an instance of problem B such that solving B provides a solution to A.
    - Example: RSA decryption reduces to factoring, meaning that breaking RSA is at most as difficult as factoring large numbers.
    - However, it is uncertain whether plaintext recovery is possible without factoring (e.g., exploiting the exponent).
3. **Unconditional Security:** The system remains secure even against an adversary with unlimited computational power.
    - Example: The one-time pad, where the key is a random string at least as long as the plaintext.
    - Encryption and decryption are performed using bitwise XOR with the key: $C = M \oplus K$, $M = C \oplus K$.
    - Assumption: The key is never reused.

Practical Considerations: Modern cryptosystems have various weaknesses:

- **Side-channel attacks:** Exploiting unintended information leakage.
    - Example: Timing attacks, where measuring encryption time can reveal cryptographic keys.
- **Best practices:** Use well-tested cryptographic libraries.
    - Ensure cryptographic operations run in constant time to prevent timing leaks.

3 Cryptosystems

A cryptosystem is defined as a tuple:

$$(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$$

where:

- $\mathcal{P}$: Finite set of plaintexts.
- $\mathcal{C}$: Finite set of ciphertexts.
- $\mathcal{K}$: Finite keyspace.
- For each $k \in \mathcal{K}$:
  - $e_k : \mathcal{P} \to \mathcal{C}$ is the encryption function, where $e_k \in \mathcal{E}$.
  - $d_k : \mathcal{C} \to \mathcal{P}$ is the decryption function, where $d_k \in \mathcal{D}$.
  - Satisfies: $d_k(e_k(x)) = x$ for all $x \in \mathcal{P}$.

Encryption Process: Suppose Alice wants to send a message $x = x_1 x_2 \ldots x_n$ to Bob:

- Alice applies the encryption function e_k to each x_i, giving $y_i = e_k(x_i)$.
- The ciphertext is $y = y_1 y_2 \ldots y_n$.
- Bob decrypts each y_i using $d_k(y_i)$.
- Encryption function e_k must be injective: $e_k(x) = e_k(y) \Rightarrow x = y$.
- If $\mathcal{C} = \mathcal{P}$, then e_k and d_k are bijections, meaning they permute the elements of $\mathcal{P}$ and $\mathcal{C}$.

4 Mathematical Background

Modular Arithmetic: For $a \in \mathbb{Z}$ and $m \in \mathbb{Z}^+$, define:

$$a \bmod m = \text{smallest nonnegative remainder of } a \text{ divided by } m.$$

Thus, $a \bmod m \in \{0, 1, \ldots, m-1\} = \mathbb{Z}_m$.

Properties of $\mathbb{Z}_m$:
$\mathbb{Z}_m$ forms a ring, meaning it supports addition and multiplication with:

- **Closure:** $\forall a, b \in \mathbb{Z}_m, \quad a + b \in \mathbb{Z}_m, \quad a \cdot b \in \mathbb{Z}_m.$
- **Commutativity:** $\forall a, b \in \mathbb{Z}_m, \quad a + b = b + a, \quad a \cdot b = b \cdot a.$
- **Associativity:** $\forall a, b, c \in \mathbb{Z}_m, \quad (a + b) + c = a + (b + c), \quad (a \cdot b) \cdot c = a \cdot (b \cdot c).$
- **Identity:** $\forall a \in \mathbb{Z}_m, \quad 0 + a = a + 0 = a, \quad 1 \cdot a = a \cdot 1 = a.$
- **Distributivity:** $\forall a, b, c \in \mathbb{Z}_m, \quad a(b + c) = a \cdot b + a \cdot c.$
- **Additive inverses:** $\forall a \in \mathbb{Z}_m, \quad \exists (m - a)$ such that $a + (m - a) = 0$.

Multiplicative Inverses: An element $a \in \mathbb{Z}_m$ only has a multiplicative inverse, a^{-1}, when $\gcd(a, m) = 1$. That is, when a is coprime to m.

- For example, 2 does not have an inverse in $\mathbb{Z}_4$ because $\gcd(2, 4) = 2 \neq 1$.
- But 3 has an inverse in $\mathbb{Z}_{10}$ because $\gcd(3, 10) = 1$, and we find that $3^{-1} = 7$ since $3 \times 7 \equiv 1 \pmod{10}$.

Fields: If m is prime, then every nonzero element of $\mathbb{Z}_m$ has a multiplicative inverse, and $\mathbb{Z}_m$ forms a field, meaning that it supports division.

5 Classical Ciphers

5.1 Caesar (Shift) Cipher

For simplicity, we use the mapping:

$$A = 0, B = 1, \ldots, Z = 25 \quad (\mathbb{Z}_{26}).$$

Define the plaintext space, ciphertext space, and keyspace as:

$$\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}.$$

For a given key $k \in \mathcal{K}$, define the encryption and decryption functions as:

$$e_k(x) = x + k \pmod{26}, \quad d_k(y) = y - k \pmod{26}.$$

Since e_k and d_k are inverses, encrypting a message and then decrypting it restores the original plaintext.

Example: Suppose we encrypt "CAT" using $k = 11$:

$$e_k(C) = N, \quad e_k(A) = L, \quad e_k(T) = E.$$

Thus, the ciphertext is "NLE".

Security Considerations: Eve, does not know k, so she cannot directly invert e_k . However, she can brute-force all possible keys, since $|\mathcal{K}| = 26$.

This makes the Caesar cipher insecure against modern cryptanalysis.

5.2 Substitution Cipher

The Substitution Cipher is a more complex scheme, where each letter is replaced by another letter.

Definition: The plaintext and ciphertext spaces remain:

$$\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}.$$

However, the keyspace consists of all permutations of $\mathbb{Z}_{26}$:

$$\mathcal{K} = \text{Perm}(\mathbb{Z}_{26}).$$

For a permutation (key) $\pi \in \mathcal{K}$ the encryption and decryption functions are:

$$e_\pi(x) = \pi(x), \quad d_\pi(y) = \pi^{-1}(y).$$

Security Considerations: Since the keyspace has size:

$$|\mathcal{K}| = 26! \approx 4 \times 10^{26},$$

it is infeasible to brute force all possible keys. However, frequency analysis can still break this cipher.

5.3 Affine Cipher

The Affine Cipher is a generalization of the Caesar cipher that applies a linear transformation.

Encryption Function: Define encryption as:

$$e_k(x) = ax + b \pmod{26}, \quad \text{where } a, b \in \mathbb{Z}_{26}.$$

Decryption Function: To decrypt, solve for x in:

$$y \equiv ax + b \pmod{26}.$$

Using modular inverses:

$$x \equiv a^{-1}(y - b) \pmod{26}.$$

Since a^{-1} must exist, a must be coprime with 26, which means that $\gcd(a, 26) = 1$.

Thus, we define the keyspace:

$$\mathcal{K} = \{(a, b) \in \mathbb{Z}_{26}^2 \mid \gcd(a, 26) = 1\} = \mathbb{Z}_{26}^* \times \mathbb{Z}_{26}.$$

Here, $\mathbb{Z}_{26}^*$ is the set of multiplicative invertible elements of $\mathbb{Z}_{26}$.

Example: Encrypting $H = 7$ with $k = (7, 3)$:

$$e_k(7) \equiv 7 \cdot 7 + 3 \equiv 52 \equiv 0 \pmod{26} = A.$$

To decrypt:

$$d_k(0) \equiv 7^{-1}(0 - 3) \equiv 15 \cdot 23 \equiv 345 \equiv 7 \pmod{26} = H.$$

Efficiency: The Euclidean algorithm can efficiently compute modular inverses for decryption.