# ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
# TRƯỜNG ĐẠI HỌC BÁCH KHOA

## Khoa Khoa Học Và Kỹ Thuật Máy Tính

**Write-up Applied Cryptography**


**Name: Dương Bá Khang**

**Student ID: 2311403**


Thành phố Hồ Chí Minh, tháng 9 năm 2025

# MỤC LỤC

# CHƯƠNG 1

# HASH FUNCTION

## 1.1 Jack's Birthday Hash

**Challenge:**

Today is Jack's birthday, so he has designed his own cryptographic hash as a way to celebrate.

Reading up on the key components of hash functions, he's a little worried about the security of the JACK11 hash.

Given any input data, JACK11 has been designed to produce a deterministic bit array of length 11, which is sensitive to small changes using the avalanche effect.

Using JACK11, his secret has the hash value: JACK(secret) = 01011001101.

Given no other data of the JACK11 hash algorithm, how many unique secrets would you expect to hash to have (on average) a 50% chance of a collision with Jack's secret?

**Solve:**

Hash Value Output (11 bits contains 0 and 1) $= 2^{11} = 2048$

So here A is the chance that we have a collision with Jack's secret.

$A = \frac{1}{2048}$ which the opposite is $\bar{A} = \frac{2047}{2048}$

So the chance of not colliding with Jack's secret is $\bar{A}$.

With 50% chance of collision, we have:

$P(A) = 1 - P(\bar{A}) = 0.5$

$\Rightarrow P(\bar{A}) = 0.5$

$P(\bar{A}) = (\bar{A})^n = (\frac{2047}{2048})^n = 0.5$

$\Rightarrow n * ln(\frac{2047}{2048}) = ln(0.5)$

$\Rightarrow n = \frac{ln(0.5)}{ln(\frac{2047}{2048})} \approx 1419.7$

So we need at least 1420 unique secrets to have a 50% chance of a collision with Jack's secret.

## 1.2 Jack's Birthday Confusion

**Challenge:**

The last computation has made Jack a little worried about the safety of his hash, and after doing some more research it seems there's a bigger problem.

Given no other data of the JACK11 hash algorithm, how many unique secrets would you expect to hash to have (on average) a 75% chance of a collision between two distinct secrets?

**Solve:**

$P(n) = 1-$ (prob that n hashes are unique)

$\Rightarrow P(n) = 1 - (\frac{H}{H} * \frac{H-1}{H} * \frac{H-2}{H} * ... * \frac{H-n+1}{H}) \Rightarrow \Pi_{k=0}^{n-1}(1 - \frac{k}{H})$

$\Rightarrow ln(\Pi_{k=0}^{n-1}(1 - \frac{k}{H})) = \sum_{k=0}^{n-1} ln(1 - \frac{k}{H})$ (1)

Where H is the number of hash value output $= 2^{11} = 2048$

Talk about Taylor series, we have: $ln(1 - x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - ...$

$\Rightarrow ln(1 - x) \approx -x$ when x « 1

$\Rightarrow 1 - x \approx e^{-x}$

So with the (1), we have: $\sum_{k=0}^{n-1} ln(1 - \frac{k}{H}) \approx -\sum_{k=0}^{n-1} \frac{k}{H} = -\frac{n(n-1)}{2H}$ (2)

We take the exponential of both sides of (2) to get: $\Pi_{k=0}^{n-1}(1 - \frac{k}{H}) \approx e^{-\frac{n(n-1)}{2H}}$ (3)

But for the large n but still n « H, we can approximate n(n-1) $\approx$ n$^2$.

(3) becomes: $e^{-\frac{n^2}{2H}}$

$\Rightarrow P(n) \approx 1 - e^{-\frac{n^2}{2H}}$

$\Rightarrow -\frac{n^2}{2H} = ln(1 - p) \Rightarrow n(p) \approx \sqrt{-2H * ln(1 - p)}$

$\Rightarrow n(0.75) \approx \sqrt{-2 * 2048 * ln(1 - 0.75)} \approx 76$

So there is 76 unique secrets to have a 75% chance of a collision between two distinct secrets.

# CHƯƠNG 2

# READING BOOK

## 2.1 Chapter 1

Kerckhoffs's Principle: must not be required to be secret, and it must be able to fall into the enemy's hands without causing inconvenience

Definition: $k \leftarrow \{0,1\}^\lambda$ means to sample k uniformly from the set of $\lambda$-bit strings.

EAVESDROP algorithm: randomized algorithm that takes as input a ciphertext c and outputs a bit b.

## 2.2 Chapter 2

Encryption syntax: A symmetric key encryption scheme (SKE) is a tuple of three algorithms (Gen, Enc, Dec) such that:

- KeyGen: a randomized algorithm that outputs a key $k \in K$.
- Enc: a (possibly randomized) algorithm that takes a key $k \in K$ and plaintext $m \in M$ as input, and outputs a ciphertext $c \in C$.
- Dec: a deterministic algorithm that takes a key $k \in K$ and ciphertext $c \in C$ as input, and outputs a plaintext $m \in M$.
- With $K$ is the key space, $M$ is the message space, and $C$ is the ciphertext space.

SKE correctness: For every key $k \in K$ and message $m \in M$, if $c \leftarrow Enc(k,m)$, then $Dec(k,c) = m$.

$\sum$ means is just a package name for "the encryption scheme," and the dot notation ($\sum .KeyGen, \sum .Enc, \sum .Dec$) means the specific algorithm belonging to that scheme