# Applied Cryptography

**Week #5 Extra**

Bernardo Portela and Rogério Reis

2025/2026

**Important**

- Your answers must **always** be accompanied by a justification. Presenting the final result (e.g. the result of a calculation) without the rationale that laid to said result will result in a grade of 0.
- Submit your answers via e-mail to *bernardo.portela@fc.up.pt*, with adequate identification of the group and its members.

## Q1: Collision resistant Hash Functions

Consider $H : M \to T$ a collision resistant hash function that takes messages of any size $m \in M = \{0,1\}^*$ and produces outputs with 64 bit length $t \in T = \{0,1\}^{64}$.

1. $H' = (H(m) \,||\, H(m) \,||\, H(m))$
2. $H' = H(m \,||\, m \,||\, m)$
3. $H' = H(64)$
4. $H' = H(m||64)$
5. $H' = H(m)[0 \ldots 10]$ // truncate the output to 10 bits
6. $H' = H(m[0 \ldots |m|\text{-}2])$ // hash $m$ without its last two bits
7. $H' = H(m) \,||\, H(m \oplus 1^{|m|}))$
8. $H' = H(m)$ if $m = 0^{64} \wedge m = 1^{64}$, $H(m \oplus 1^{|m|})$ otherwise

**Question:** Which of the proposed hash constructions $H'$ are also collision resistant? For those who are not, explain how one can construct an attacker against said hash function: i.e. how one can find two values $m_0, m_1$ such that $H(m_0) = h(m_1)$

## Q2: Rho method to find Hash collisions

As described in [1], the Rho method is an algorithm for finding collisions that, unlike the naive birthday attack, requires only a small amount of memory. To find collision in hash function $H(m)$, it works as follows.

1. Given a hash function with $n$-bit values, pick some random hash value $h_1$ and define $h'_1 = h_1$.
2. Compute $h_2 = H(h_1)$ and $h'_2 = H(H(h'_1))$. In the first case, we apply the hash function once. In the second, we apply it twice.
3. Iterate the process and compute $h_{i+1} = H(h_i)$ and $h'_{i+1} = H(H(h'_i))$, until you reach a $i$ such that $h'_{i+1} = h_{i+1}$
4. If this is the case, then you have found a loop within the possible hash values. How can we find the collision now? Check out this proof.

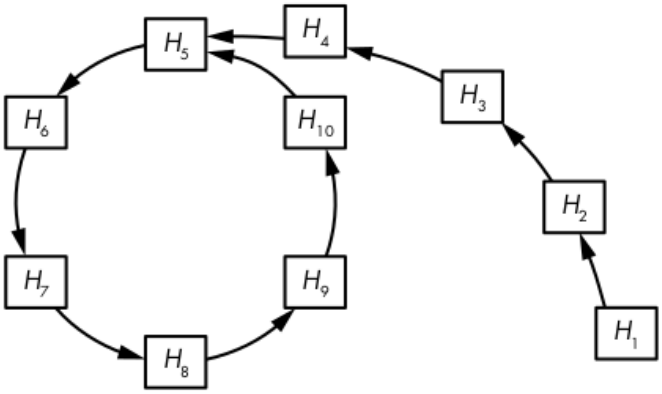


Figure 1: Rho Method

Complete the code in `rho_exercise.py` to do this.

- You must complete function `rho`, which is parametrized by an initial value
- Function `H` computes hashes truncated as necessary.
- You can adjust the global parameter during testing, but the goal is to find a collision in $L = 5$.

Also include a succinct analysis of how long it takes to find these collisions, both in cycle iterations and real time. How does this scale with `L`?

## Q3: Weak ciphers

The code in `ciphersuite_fsr.py` contains a very poorly implemented "stream cipher''.

1. Consider the IND-CPA security experiment. How many calls to the encryption oracle do you have to do to succeed?
2. Describe how one can construct an attacker against the IND-CPA experiment running this encryption scheme.

[1] Jean-Philippe Aumasson; Serious Cryptography: A Practical Introduction to Modern Encryption, No Startch Press, 2017. ISBN: 9781593278267