CTF Challenge: "Message Integrity Violator"

The following is the high-level scheme how SHA-1 computes the hash. Blocks with “C” represent compression functions that take two imputs: A message block of length 64 bytes and the 20 byte output of the previous compression function (or 0x67452301efcdab8998badcfe10325476c3d2e1f0 if it is the first compression function).

“c” is the message to be hashed, including the secret key.

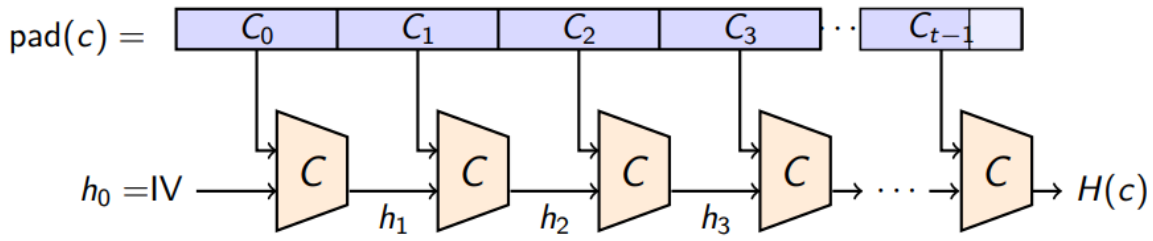

Image credit: *Taken from the "Introduction to Cryptography" course.*

Note that in this challenge, the server prepends the secret key to the received data and computes the hash of (secret key + received data) and then validates it against the received mac. The computation of the hash follows the specification found in https://www.ietf.org/rfc/rfc3174.txt. **Note that adding the padding is part of the implementation of SHA1.**

Objective: Forge a valid (message, hash) pair that differs from the original to reveal the flag.

Challenge Description

The admin portal at https://miv.chall.trojanc.tf/verify uses a **signed URL parameter** for authentication. The URL looks like:

https://miv.chall.trojanc.tf/verify?data=<url_encoded_data>&mac=<hex_mac>

You’ve intercepted a valid URL:

https://miv.chall.trojanc.tf/verify?data=user%3Dguest%26role%3Dmember&mac=49fed001fe4c84a9468711fa33784a1b6b5d3588

Your task:

1. Submit a **new URL** with a valid data and mac pair.
2. The data must be **different** from the original (user=guest&role=member).
3. The mac must match the server’s verification logic.

Server Behavior

1. **MAC Verification**:
   The server computes the MAC as:

SHA1(secret_key + data)

It checks if the provided mac matches this computed value.

2. **Flag Condition**:
   The server reveals the flag if:
   - The mac is valid.
   - The data is different from the original.

Important

1. The secret key is **32 bytes long**.
2. The hash algorithm is **SHA-1**.