Deliverable 1: Background Write-up

Title: Demonstrating and Mitigating a MAC Forgery Attack via Length Extension

1. Introduction

Message Authentication Codes (MACs) are widely used to verify the integrity and authenticity of messages.

However, if MACs are implemented using insecure hash constructions such as MD5(key || message),

they are vulnerable to length extension attacks. This document explains such an attack, demonstrates it using code,

and proposes a secure mitigation using HMAC.

2. Problem Statement

We explore the vulnerability in systems that generate MACs using a secret-prefix method:

MAC = hash(secret_key || message)

If a user can obtain the hash of an original message, they may be able to compute the hash of a new message:

MAC = hash(secret_key || message || attacker_data)

...without knowing the secret key.

3. Why This Works

Hash functions like MD5 and SHA1 process input in fixed-size blocks and retain internal state across chunks.

If the attacker knows the hash output and original message length, they can:

- Guess the secret key length.

- Calculate the proper MD padding of key || message.

- Append new data and compute the new MAC using the internal state derived from the original MAC.

4. Threat Model

The attacker knows:

- The message.
- The MAC of the message.
- The hash function used (MD5).

The attacker does not know:

- The secret key (SECRET_KEY).

The attacker wants to:

- Append arbitrary data.
- Forge a new valid MAC for the extended message.

5. Real-World Relevance

This attack has real-world implications where integrity and authentication are critical. Systems that implement

insecure MACs using simple hash constructions are susceptible to message tampering without detection, undermining

both data integrity and authenticity.

6. Learning Goals

- Understand the internals of MD5 and padding logic.
- Implement a realistic attack to forge a valid MAC.
- Analyze how HMAC mitigates this vulnerability.