Background Write-up: MAC Forgery Assignment

1. What is a MAC and its Purpose in Data Integrity/Authentication?

A Message Authentication Code (MAC) is a cryptographic mechanism used to ensure the integrity and authenticity of a message. It is generated by combining a secret key with the message using a hash function or a similar cryptographic algorithm. The MAC is appended to the message, allowing the recipient to verify that the message has not been altered (integrity) and that it originates from a legitimate sender who possesses the secret key (authentication). By comparing the received MAC with a locally computed MAC using the same key and message, the recipient can confirm the message's validity. MACs are widely used in secure communication protocols, such as TLS and VPNs, to prevent tampering and impersonation attacks.

2. How Does a Length Extension Attack Work in Hash Functions Like MD5/SHA1?

A length extension attack exploits a vulnerability in certain hash functions, such as MD5 and SHA1, due to their Merkle-Damgård construction. These hash functions process input data in fixed-size blocks and maintain an internal state that is updated as each block is processed. When a hash is computed, the final state is output as the hash value. In a length extension attack, an attacker who knows a valid hash (e.g., `hash(secret || message)`) and the length of the secret can append additional data to the message and compute a new valid hash without knowing the secret. This is possible because the attacker can initialize the hash function with the known hash value (as the internal state) and continue hashing the appended data, effectively extending the original message. For example, if the original message is `amount=100&to=alice`, the attacker can append `&admin=true` and generate a new valid hash, making it appear as though the extended message was signed with the secret key.

3. Why is `MAC = hash(secret || message)` Insecure?

The construction `MAC = hash(secret || message)` is insecure because hash functions like MD5 and SHA1 are vulnerable to length extension attacks. In this construction, the secret key is simply concatenated with the message, and the result is hashed. An attacker who intercepts a valid `(message, MAC)` pair can exploit the Merkle-Damgård structure of the hash function to append arbitrary data to the message and compute a new valid MAC without knowing the secret key. This is because the hash function's internal state can be reused to continue hashing from the point of the original hash. For instance, if an attacker intercepts `MAC = hash(secret || "amount=100&to=alice")`, they can append `&admin=true` and calculate a new MAC that the server will accept as valid. This vulnerability allows attackers to forge messages, compromising both integrity

and authentication. Secure MAC constructions, such as HMAC, mitigate this issue by using a more robust keying mechanism that prevents such attacks.