Vivekanand Education Society's Institute of Technology

Department of AI&DS Engineering

Subject: Cryptography and System Security

Class: D11AD

Roll No: 46	**Name: Ashish Patil**
Practical No:4	**Title: Message Integrity using MD5 algorithm**
DOP:	**DOS:**
Grades:	**LOs Mapped:**
Signature:	

Title: Message Integrity using MD5 Algorithm
DOP: /2/24
DOS: /2/24

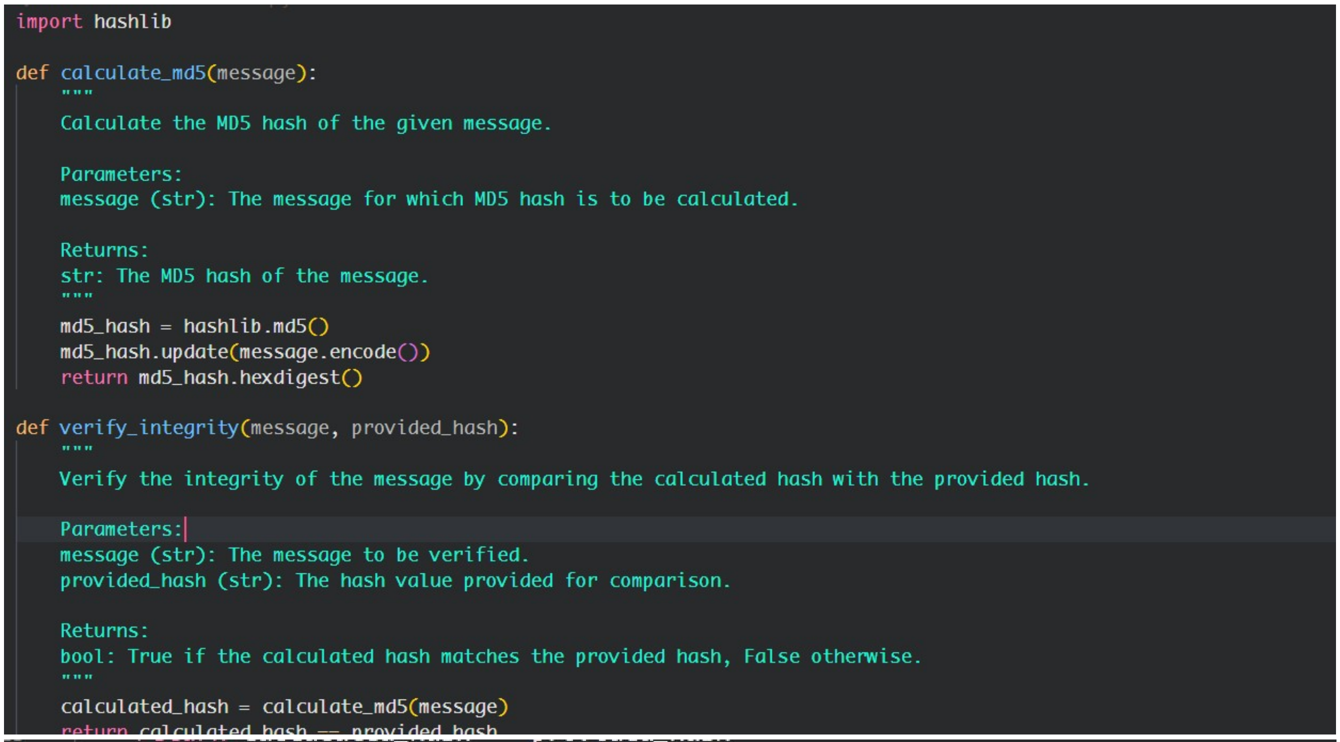
Aim: For varying message sizes, test integrity of message using MD-5, SHA-1, and analyze the performance of the two protocols. Use crypt APIs.

Theory:

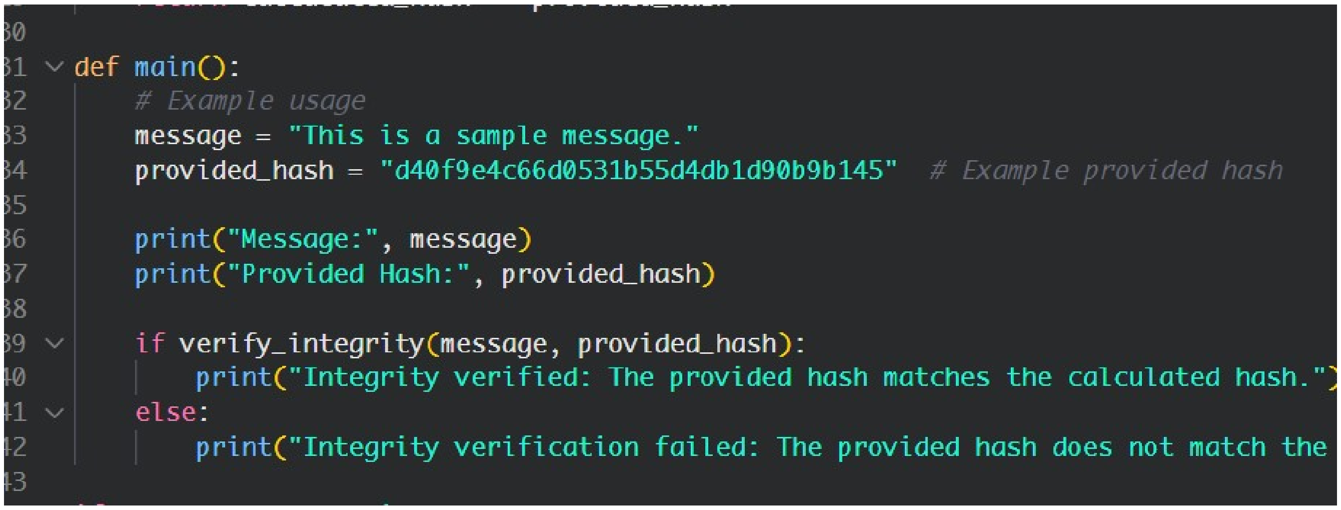
1.What is MD5 Algorithm.
2.Explain steps in MD5 Algorithm
3.SHA-1
4.How to Analyze the Performance of two protocols
5.How to use crypt APIs

Program:

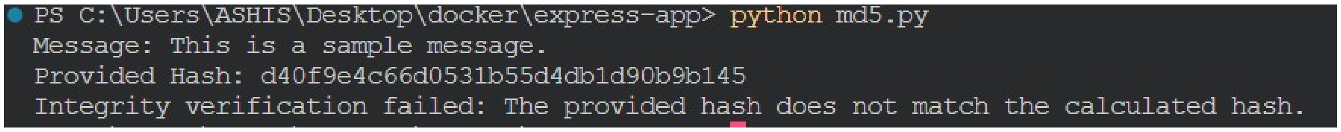
MD-5 algorithm:-



```python
import hashlib

def calculate_md5(message):
    """
    Calculate the MD5 hash of the given message.

    Parameters:
    message (str): The message for which MD5 hash is to be calculated.

    Returns:
    str: The MD5 hash of the message.
    """
    md5_hash = hashlib.md5()
    md5_hash.update(message.encode())
    return md5_hash.hexdigest()

def verify_integrity(message, provided_hash):
    """
    Verify the integrity of the message by comparing the calculated hash with the provided hash.

    Parameters:
    message (str): The message to be verified.
    provided_hash (str): The hash value provided for comparison.

    Returns:
    bool: True if the calculated hash matches the provided hash, False otherwise.
    """
    calculated_hash = calculate_md5(message)
    return calculated_hash == provided_hash
```

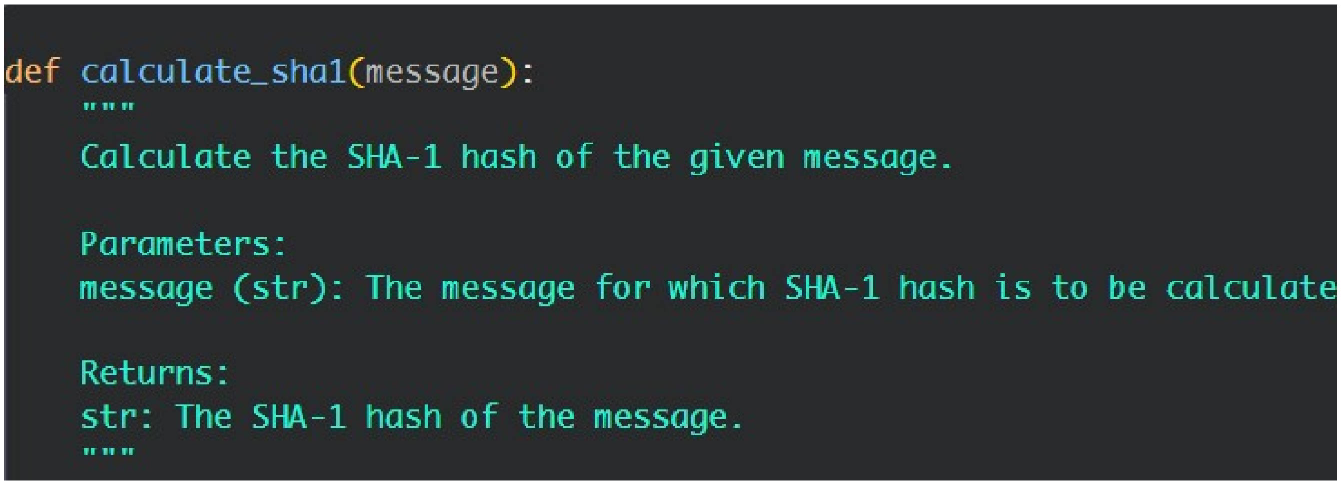


```python
def main():
    # Example usage
    message = "This is a sample message."
    provided_hash = "d40f9e4c66d0531b55d4db1d90b9b145"  # Example provided hash

    print("Message:", message)
    print("Provided Hash:", provided_hash)

    if verify_integrity(message, provided_hash):
        print("Integrity verified: The provided hash matches the calculated hash.")
    else:
        print("Integrity verification failed: The provided hash does not match the
```

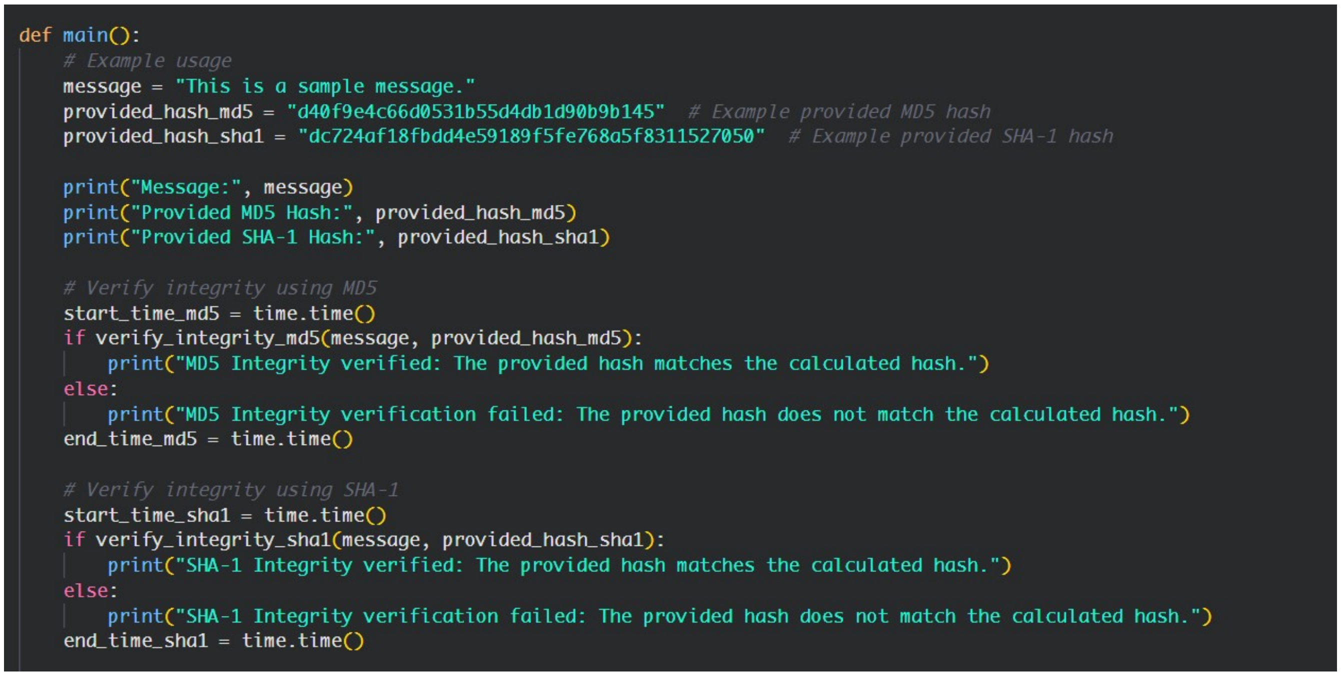


```
PS C:\Users\ASHIS\Desktop\docker\express-app> python md5.py
Message: This is a sample message.
Provided Hash: d40f9e4c66d0531b55d4db1d90b9b145
Integrity verification failed: The provided hash does not match the calculated hash.
```

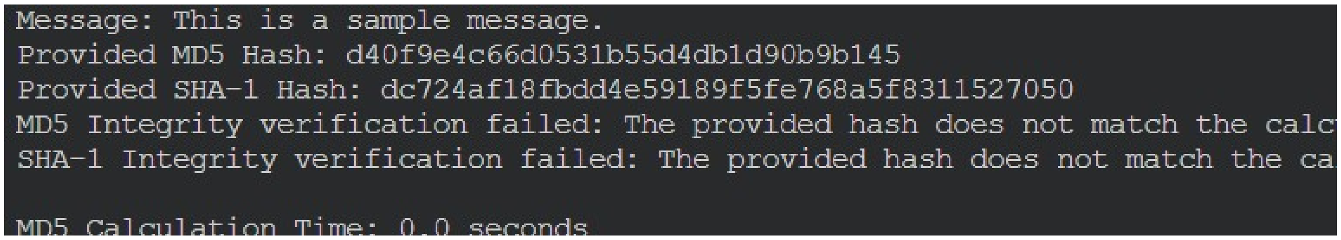
SHA-1 algorithm and comparison:



```
def calculate_sha1(message):
    """
    Calculate the SHA-1 hash of the given message.

    Parameters:
    message (str): The message for which SHA-1 hash is to be calculate

    Returns:
    str: The SHA-1 hash of the message.
    """
```



```
def main():
    # Example usage
    message = "This is a sample message."
    provided_hash_md5 = "d40f9e4c66d0531b55d4db1d90b9b145"  # Example provided MD5 hash
    provided_hash_sha1 = "dc724af18fbdd4e59189f5fe768a5f8311527050"  # Example provided SHA-1 hash

    print("Message:", message)
    print("Provided MD5 Hash:", provided_hash_md5)
    print("Provided SHA-1 Hash:", provided_hash_sha1)

    # Verify integrity using MD5
    start_time_md5 = time.time()
    if verify_integrity_md5(message, provided_hash_md5):
        print("MD5 Integrity verified: The provided hash matches the calculated hash.")
    else:
        print("MD5 Integrity verification failed: The provided hash does not match the calculated hash.")
    end_time_md5 = time.time()

    # Verify integrity using SHA-1
    start_time_sha1 = time.time()
    if verify_integrity_sha1(message, provided_hash_sha1):
        print("SHA-1 Integrity verified: The provided hash matches the calculated hash.")
    else:
        print("SHA-1 Integrity verification failed: The provided hash does not match the calculated hash.")
    end_time_sha1 = time.time()
```



```
Message: This is a sample message.
Provided MD5 Hash: d40f9e4c66d0531b55d4db1d90b9b145
Provided SHA-1 Hash: dc724af18fbdd4e59189f5fe768a5f8311527050
MD5 Integrity verification failed: The provided hash does not match the calc
SHA-1 Integrity verification failed: The provided hash does not match the ca

MD5 Calculation Time: 0.0 seconds
```

Conclusion: We have successfully implemented both the algorithms and compared their performance.