|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| D:\sep2k3\COLLEG~1\LOGO.JPG | SHRI SANT GAJANAN MAHARAJ COLLEGE OF ENGG. SHEGAON | | | **LABORATORY MANUAL** | |
| **PRACTICAL EXPERIMENT INSTRUCTION SHEET** | | | | |
| LABORATORY MANUAL NO.:SSGMCE/WI/IT/BCF/7IT08/15 | | ISSUE NO.: 00 | | ISSUE DATE: 18/07/2022 | |
| REV.DATE: | | DEPTT:INFORMATION TECHNOLOGY | | | |
| LABORATORY: Block Chain Fundamental | | | SEMESTER: 4N/VII | | PAGE: 1 OF 3 |

**AIM: Study of Cryptographic Primitives Part –II ,MAC,SHA256, Digital Signature, Merkel Tree**

1. **THEORY**

1)A Cipher Block Chaining Message Authentication Code (CBC-MAC), is a technique for constructing a message authentication code from a block cipher.

**About the experiment:**

In the experiment we provide a "dummy" block cipher (that is just a function and has none of the properties of a real block cipher) and ask you to use it to compute the CBC\_MAC tag of an arbitrary message. A message authentication code (MAC) is a [cryptographic checksum](https://www.techtarget.com/searchsecurity/definition/cryptographic-checksum) on data that uses a [session key](https://www.techtarget.com/searchsecurity/definition/session-key) to detect both accidental and intentional modifications of the data.

A MAC requires two inputs: a message and a [secret key](https://www.techtarget.com/searchsecurity/definition/private-key) known only to the originator of the message and its intended recipient(s). This allows the recipient of the message to verify the integrity of the message and authenticate that the messege's sender has the shared secret key. If a sender doesn’t know the secret key, the hash value would then be different, which would tell the recipient that the message was not from the original sender.

There are four types of MACs:  unconditionally secure, hash function-based, stream cipher-based  and block cipher-based  In the past, the most common approach to creating a MAC was to use [block ciphers](https://www.techtarget.com/searchsecurity/definition/block-cipher) like [Data Encryption Standard](https://www.techtarget.com/searchsecurity/definition/Data-Encryption-Standard) (DES), but [hash-based MACs](https://www.techtarget.com/searchsecurity/definition/Hash-based-Message-Authentication-Code-HMAC) (HMACs) which use a secret key in conjunction with a cryptographic hash function to produce a hash, have become more widely used.

## 2) SHA-256 Cryptographic Hash Algorithm

A cryptographic hash (sometimes called ‘digest’) is a kind of ‘signature’ for a text or a data file. SHA-256 generates an almost-unique 256-bit (32-byte) signature for a text.

A hash is not ‘encryption’ – it cannot be decrypted back to the original text (it is a ‘one-way’ cryptographic function, and is a fixed size for any size of source text). This makes it suitable when it is appropriate to compare ‘hashed’ versions of texts, as opposed to decrypting the text to obtain the original version.

Such applications include hash tables, integrity verification, challenge handshake authentication, digital signatures, etc.

* ‘*challenge handshake authentication*’ (or ‘challenge hash authentication’) avoids transmissing passwords in ‘clear’ – a client can send the hash of a password over the internet for validation by a server without risk of the original password being intercepted
* *anti-tamper* – link a hash of a message to the original, and the recipient can re-hash the message and compare it to the supplied hash: if they match, the message is unchanged; this can also be used to confirm no data-loss in transmission
* *digital signatures* are rather more involved, but in essence, you can sign the hash of a document by encrypting it with your private key, producing a digital signature for the document. Anyone else can then check that you authenticated the text by decrypting the signature with your public key to obtain the original hash again, and comparing it with their hash of the text.

Note that hash functions are not appropriate for storing encrypted passwords, as they are designed to be fast to compute, and hence would be candidates for brute-force attacks. Key derivation functions such as [bcrypt](https://en.wikipedia.org/wiki/Bcrypt) or [scrypt](https://en.wikipedia.org/wiki/Scrypt) are designed to be slow to compute, and are more appropriate for password storage (npm has [bcrypt](https://www.npmjs.com/package/bcrypt) and [scrypt](https://www.npmjs.com/package/scrypt) libraries, and PHP has a bcrypt implementation with [password\_hash](https://php.net/manual/en/function.password-hash.php)).

**SHA-256** is one of the successor hash functions to SHA-1 (collectively referred to as SHA-2), and is one of the strongest hash functions available. SHA-256 is not much more complex to code than SHA-1, and has not yet been compromised in any way. The 256-bit key makes it a good partner-function for AES. It is defined in the NIST (National Institute of Standards and Technology) standard ‘[FIPS 180-4](http://csrc.nist.gov/groups/ST/toolkit/secure_hashing.html)’.

3) Digital signatures are the public-key primitives of message authentication. In the physical world, it is common to use handwritten signatures on handwritten or typed messages. They are used to bind signatory to the message.Similarly, a digital signature is a technique that binds a person/entity to the digital data. This binding can be independently verified by receiver as well as any third party.Digital signature is a cryptographic value that is calculated from the data and a secret key known only by the signer.In real world, the receiver of message needs assurance that the message belongs to the sender and he should not be able to repudiate the origination of that message. This requirement is very crucial in business applications, since likelihood of a dispute over exchanged data is very high.

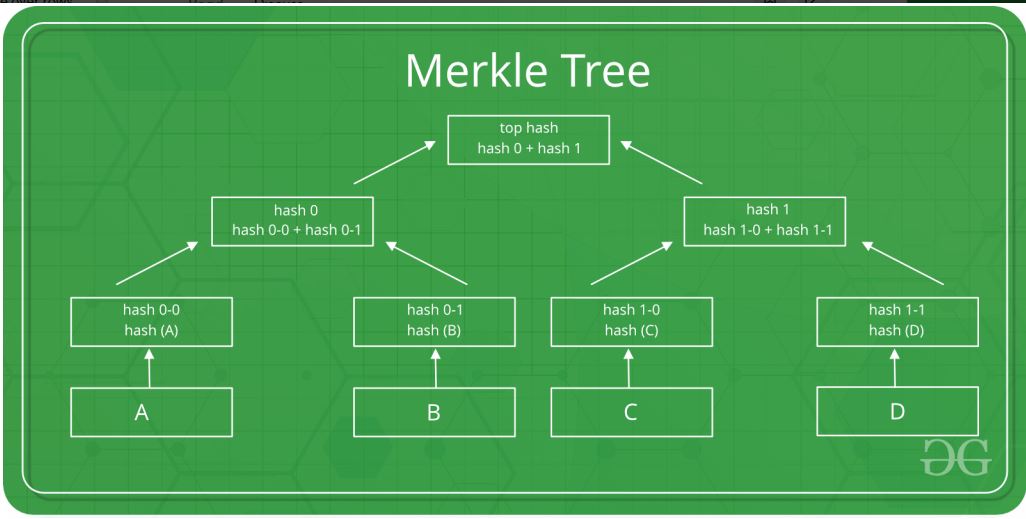
The following points explain the entire process in detail −

* Each person adopting this scheme has a public-private key pair.
* Generally, the key pairs used for encryption/decryption and signing/verifying are different. The private key used for signing is referred to as the signature key and the public key as the verification key.
* Signer feeds data to the hash function and generates hash of data.
* Hash value and signature key are then fed to the signature algorithm which produces the digital signature on given hash. Signature is appended to the data and then both are sent to the verifier.
* Verifier feeds the digital signature and the verification key into the verification algorithm. The verification algorithm gives some value as output.
* Verifier also runs same hash function on received data to generate hash value.
* For verification, this hash value and output of verification algorithm are compared. Based on the comparison result, verifier decides whether the digital signature is valid.
* Since digital signature is created by ‘private’ key of signer and no one else can have this key; the signer cannot repudiate signing the data in future.

4) Merkle tree also known as hash tree is a data structure used for data verification and synchronization. It is a tree data structure where each non-leaf node is a hash of it’s child nodes. All the leaf nodes are at the same depth and are as far left as possible.   
It maintains data integrity and uses hash functions for this purpose.

**Hash Functions:**

So before understanding how Merkle trees work, we need to understand how hash functions work.A hash function maps an input to a fixed output and this output is called hash. The output is unique for every input and this enables fingerprinting of data. So, huge amounts of data can be easily identified through their hash.



This is a **binary merkel tree**, the top hash is a hash of the entire tree.

* This structure of the tree allows efficient mapping of huge data and small changes made to the data can be easily identified.
* If we want to know where data change has occurred then we can check if data is consistent with root hash and we will not have to traverse the whole structure but only a small part of the structure.
* The root hash is used as the fingerprint for the entire data.

**03 CONCLUSION:** In this way, we studied, MAC, SHA256 , Digital Signature, Merkel Tree.