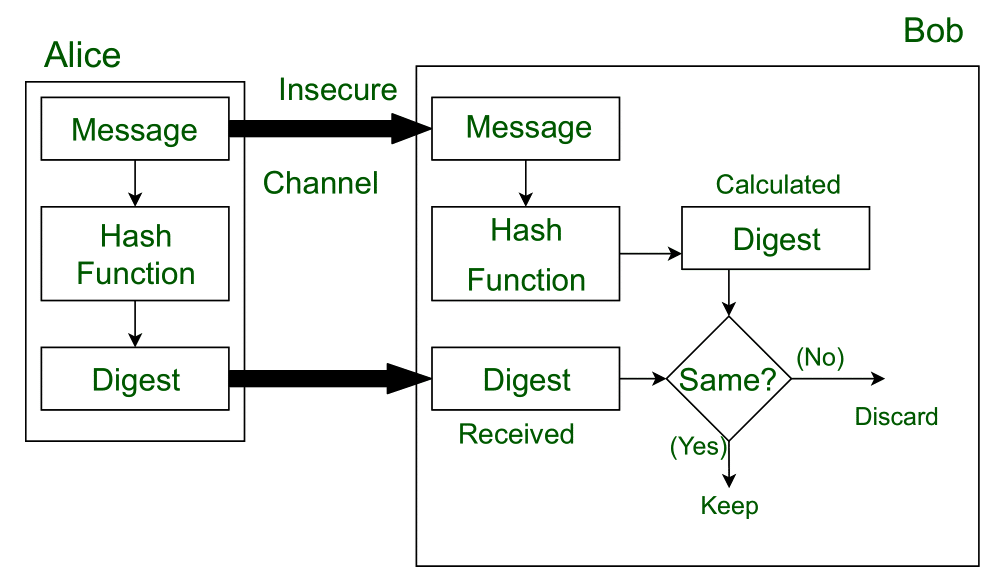
***MD-5 Algorithm***

Message Digest 5 or MD5 in short, is a cryptographic hashing algorithm and a one-way function that takes a message of arbitrary length and squishes them to produce a 128-bit digest. These MD5 hashes are represented using 32 hexadecimal characters – for example, MD5 digest of “Hello” is 09f7e02f1290be211da707a266f153b3. Cryptographic hashes are used to verify the integrity of files, store secret information like passwords in databases or convert data to fixed-length output.



This message and digest pair is equivalent to a physical document and fingerprint of a person on that document. Unlike the physical document and the fingerprint, the message and the digest can be sent separately.

* Most importantly, the digest should be unchanged during the transmission.
* The cryptographic hash function is a one-way function, that is, a function which is practically infeasible to invert. This cryptographic hash function takes a message of variable length as input and creates a **digest / hash / fingerprint** of fixed length, which is used to verify the integrity of the message.
* Message digest ensures the integrity of the document. To provide authenticity of the message, digest is encrypted with sender’s private key. Now this digest is called digital signature, which can be only decrypted by the receiver who has sender’s public key. Now the receiver can authenticate the sender and also verify the integrity of the sent message.

## **Java Standard Library implementation**

Java Standard Library has Message Digest class which provides applications the functionality of a message digest algorithm, such as MD5, SHA-1 or SHA-256. Message digests are secure one-way hash functions that take arbitrary-sized data and output a fixed-length hash value.

**Using Java Standard Library**

import javax.xml.bind.DatatypeConverter;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;class Utils {public String md5(String input) throws NoSuchAlgorithmException { MessageDigest md = MessageDigest.getInstance("MD5"); md.update(input.getBytes()); byte[] digest = md.digest();1\*

return DatatypeConverter.printHexBinary(digest).toUpperCase();2\* }

}

1\* - Computes Message Digest Code in byte array format

2\* - Converts byte array to base 16 Hex encoded String

### **How do the MD5 Algorithm works?**

As we all know that MD5 produces an output of 128-bit hash value. This encryption of input of any size into hash values undergoes 5 steps, and each step has its predefined task.

#### **Step1: Append Padding Bits**

* Padding means adding extra bits to the original message. So, in MD5 original message is padded such that its length in bits is congruent to 448 modulo 512. Padding is done such that the total bits are 64 less, being a multiple of 512 bits length.
* Padding is done even if the length of the original message is already congruent to 448 modulo 512. In padding bits, the only first bit is 1, and the rest of the bits are 0.

#### **Step 2: Append Length**

After padding, 64 bits are inserted at the end, which is used to record the original input length. Modulo 2^64. At this point, the resulting message has a length multiple of 512 bits.

#### **Step 3: Initialize MD buffer.**

A four-word buffer (A, B, C, D) is used to compute the values for the message digest. Here A, B, C, D are 32- bit registers and are initialized in the following way

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Word A | 01 | 23 | 45 | 67 |
| Word B | 89 | Ab | Cd | Ef |
| Word C | Fe | Dc | Ba | 98 |
| Word D | 76 | 54 | 32 | 10 |

#### **Step 4: Processing message in 16-word block**

MD5 uses the auxiliary functions, which take the input as three 32-bit numbers and produce 32-bit output. These functions use logical operators like OR, XOR, NOR.

|  |  |
| --- | --- |
| F(X, Y, Z) | XY v not (X)Z |
| G(X, Y, Z) | XZ v Y not (Z) |
| H(X, Y, Z) | X xor Y xor Z |
| I(X, Y, Z) | Y xor (X v not (Z)) |

The content of four buffers are mixed with the input using this auxiliary buffer, and 16 rounds are performed using 16 basic operations.

**Output-**

After all, rounds have performed, the buffer A, B, C, D contains the MD5 output starting with lower bit A and ending with higher bit D.

### **Advantages and Disadvantages of MD5 Algorithm:**

Below are the advantages and disadvantages explained:

* MD5 Algorithms are useful because it is easier to compare and store these smaller hashes than store a large variable length text. It is a widely used algorithm for one-way hashes used to verify without necessarily giving the original value. Unix systems use the MD5 Algorithm to store the passwords of the user in a 128-bit encrypted format. MD5 algorithms are widely used to check the integrity of the files.
* Moreover, it is very easy to generate a message digest of the original message using this algorithm. It can perform the message digest of a message having any number of bits; it is not limited to a message in the multiples of 8, unlike MD5sum, which is limited to octets.
* But for many years, MD5 has prone to hash collision weakness, i.e., it is possible to create the same hash function for two different inputs. MD5 provides no security over these collision attacks. Instead of MD5, SHA (Secure Hash Algorithm, which produces 160-bit message digest and designed by NSA to be a part of digital signature algorithm) is now acceptable in the cryptographic field for generating the hash function as it is not easy to produce SHA-I collision and till now no collision has been produced yet.
* Moreover, it is quite slow then the optimized [SHA algorithm](https://www.educba.com/sha-algorithm/). SHA is much secure than the MD5 algorithm, and moreover, it can be implemented in existing technology with exceeding rates, unlike MD5. Nowadays, new hashing algorithms are coming up in the market, keeping in mind higher security of data like SHA256 (which generates 256 bits of signature of a text).