

## PalArch's Journal of Archaeology of Egypt / Egyptology

### Implementation of Data Integrity using MD5 and MD2 Algorithms in IoT Devices

<sup>1</sup>Sandeep K V, <sup>2</sup>Dr.Sayed Abdulhayan

<sup>1</sup>Research Scholar, Department of Electronics & Communication Engineering, Dayananda Sagar College of Engineering, Visvesvaraya Technological University, Bangalore, Karnataka, India

<sup>2</sup>Associate Professor, Department of Electronics & Telecommunication Engineering, Dayananda Sagar College of Engineering, Visvesvaraya Technological University, Bangalore, Karnataka, India.

Email: <sup>1</sup>sandeep.kv38@gmail.com, <sup>2</sup>sabdulhayan@gmail.com

**Sandeep K V, Dr.Sayed Abdulhayan: Implementation of Data Integrity using MD5 and MD2 Algorithms in IoT Devices -- Palarch's Journal Of Archaeology Of Egypt/Egyptology 17(6). ISSN 1567-214x**

**Keywords: Secure Hash Algorithm version (SHA), Message Digest version (MD), Block chain, IoT, Integrity**

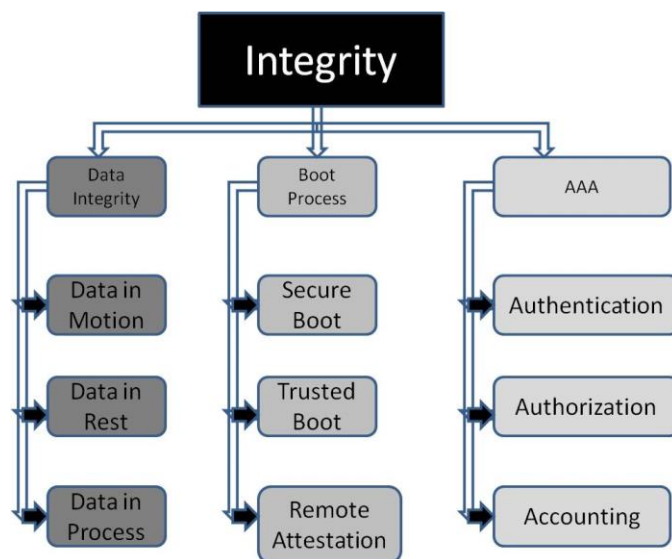
#### ABSTRACT

It is important that the data received by the IoT device is legit so that it can process the data and provide correct output that is necessary for the other stages of a system dependent on the output, to work smoothly. This is where data integrity is put into effect. In order to check the legitimacy, precision and authenticity of the data, a technique called hashing is used. This hashing can be done by using the Secure Hash Algorithm version (SHA) and Message Digest version (MD). MD5 algorithm is implemented by building block chains using python in which various functions like hash H(x) and verify( ) are used for verification of data. Block chains are nothing but a data structure that holds transactional records or in simple terms, it is a chain of blocks that holds information. A block chain is built because it is resistant to any kind of modifications to the data making it helpful to check integrity of data. Also MD2 Integrity Algorithm is implemented to check the Integrity status of Message data after being transferred from IoT transmitter to IOT receiver using Wi-Fi Communication Technology.

#### 1. Introduction

The growth in the IOT field is increasing and hence there is a surplus number of devices and large amounts of information produced by it. This information is nothing but data accessed from the environment by the device, data transferred from one device to another, data as output from the device. All this is done through an open internet making the data exposed to attacks like data tampering. It is very important for IOT devices to process valid data to give

accurate output and hence the data's integrity is checked. In the direction to lead the procedure of integrity, it will be useful to consider three special states that the information can exist, that is in the motion at rest and in process. Further, fig. 1 below describes the de-composition of Integrity principles into sub-principles and implementation of the Internet of Things devices that can be incorporated in defending the integrity of its information.



**Fig. 1: De-composition of the Integrity principle into sub-principles**

"Implementation of data Integrity in IoT devices may violate any data integrity which mean that an IoT devices may not perform the task correctly. This will potentially expose the device to being exploited" [8] and becomes a compromised platform from where the additional attacks can launch. "This typical means of verifying the data integrity is by using a mathematical algorithm known as hash" [9].

Hashing is a process in which the data is converted into a text string called a hash. It is basically assigning a key to a block of data and this key will be difficult to clone but easy to verify. The key will satisfy these below property:

- The Similar input  $x$ , forever produces the similar output  $H(x)$ .
- The different (or similar) input  $x$  must generate completely different output  $H(x)$ .
- It is simple to get  $H(x)$  from  $x$ . But, it's difficult to reverse the procedure and to get  $x$  from a known hash value  $H(x)$ .

**Hashing same data:** If you copy a file and therefore have two files containing the same data, and if you hash them with the same hashing algorithm, it will always produce the same hash value.

**Verifying Integrity:** During forensic analysis, the scientist takes a copy of the data prior to investigation. To ensure that he/she has not tampered with it during investigation, he/she will hash the data before starting and then compare the hash to the data when he/she has finished. If the hash matches, then we know that the integrity of the data is intact.

**Minimal Block:** The Minimal Block ( ) which is an object class can be made to initialize by specifying an index, an time stamp, some data that we want to store and earlier hash value. This is one of the key block of the earlier one and acts as a pointer and tells us how the blocks are connected to each other. This means, Block [x] has index x, a time stamp, some data, and the hash of the previous block that is [x-1]  $H(\text{Block}[x-1])$ . Here this block is complete, and it can be hashed to generate  $H(\text{Block}[x])$  indicates as a pointer in the next block.

**Minimal Chain:** Chain of blocks is basically known as Block chain. Block chain connection is done by saving hash value of the earlier blocks. Hence, implementation of chain can be done by using the python list. Also,  $\{i\}^{\text{th}}$  block is represented by `blocks[i]`. When a chain is initialized, 0<sup>th</sup> block is assigned to the chain using the function known as `get_genesis_block()`. This blocks shows the start of chain and a block can be added by using `add_block()`.

**Data Verification:** For data base and block chains to provide an easy method to validate all the data, data integrity is essential.

## 2. Literature Review

A brief review of the relevant works published in the past decade is described in this section.

[1] "Author gives an inspection to dissimilar security issue at each layer together with the cross-layer heterogeneous mix security issues and propose some encouraging courses of action" [1].

[2] "In order to avoid and minimize these attacks, abundant security procedures and cryptographic algorithms been developed. The study on several existing protocols had been carried along with the study on authentication algorithms. Also discussed briefly about the key necessities for the data security and authentication for the future aspects along with necessity of memory and power utilization" [2]. The location of source and sink nodes are most vulnerable for data hacking, therefore our prime concern should be preserving these nodes first. IoT generate huge sum of data which makes it complex for the researchers for handling data.

[3] "A K-means cluster based algorithm has been used for preservation of source and sink nodes. The real source and sink nodes are protected by using fake nodes" [3].

[4] Authors had used the concepts of "Artificial Immune system are utilized for generating a non-redundant device signature which is used to differentiate between authentic and malicious nodes". There algorithm can be used for providing security in IOT devices with limited battery life and processing power such as IOT enabled and remotely deployed Wireless Sensor Networks for forest fire detection, power plant monitoring, remote military applications etc" [4].

[5] authors have done "comparison of the existing algorithms. Also, Subsequently, Augmented Security and Optimized memory space is achieved for the data channelized via IoT by using the combination of the Light weight masked AES (Advanced Encryption Standard) and MD5 hash algorithms. In

the proposed algorithm, area, power and timing factors are reduced using optimization techniques, which drastically reduces the power consumed, and chip area. Chip area is calculated in terms of gate equivalents and power consumption is reduced through clock gating and operand isolation techniques" [5].

### 3. Existing Algorithms

In this section few important and most widely used existing algorithms has been discussed.

#### A. *SHA-256*

'Secure Hash Algorithm is abbreviated as SHA. Cryptographic hash functions are mathematical operations which run on digital data by comparing the computed "hash" to a known and expected hash value, and can determine the data integrity. A one way hash can be generated from any piece of data, but the data cannot be generated from the hash. It is a one way function that converts a text of any length into a string of 256 bits. It is like a signature for a text or a file. "This procedure works with information broken down into pieces of 512 bits. It produces it's cryptographic 'mixing' and then issues a 256 bit hash code. This algorithm include a relatively simple round, that is repeated by 64 times" [1]. A round comprises block chains to create a function that repeats itself multiple times. Used to create bit coin addresses for privacy and security.

#### B. *MD5*

The MD5 is widely used as hash function that producing a 128 bit hash. A MD5 hash esteem is created by utilizing a line of any length and encoding it into a 128-bit unique mark. Encoding a similar string utilizing the MD5 calculation will consistently bring about a similar 128-cycle hash yield. MD5 hashes are ordinarily utilized with more modest strings while putting away passwords, charge card numbers or other touchy information in data sets, for example, the famous MySQL. This instrument gives a brisk and simple approach to encode a MD5 hash from a basic line of up to 256 characters long. MD5 hashes are likewise used to guarantee the information honesty of records. Since the MD5 hash calculation consistently delivers a similar yield for a similar given information, clients can look at a hash of the source document with a recently made hash of the objective record to watch that it is unblemished and unmodified. A MD5 hash isn't encryption. It is basically a unique mark of the given info. Notwithstanding, it is a single direction exchange and all things considered, it is practically difficult to figure out a MD5 hash to recover the first string.

The over two techniques depicted use hash. A hash isn't encryption. It can't be decoded back to the first content. This makes it reasonable when it is suitable to look at hashed adaptations of writings, instead of decoding the content to get the first form.

### C. MD2

The 128-bit hash estimation of any message is framed by cushioning it to a numerous of the square length (128 pieces or 16 bytes) and adding a 16-byte checksum to it. For the real figuring, a 48-byte helper block and a 256-byte S-table created by implication from the digits of pi are utilized. The calculation goes through a circle where it permutes every byte in the assistant square multiple times for each 16 info bytes prepared. When the entirety of the squares of the (protracted) message have been prepared, the primary incomplete square of the assistant square turns into the hash estimation of the message.

These methods just compare hashed values of a file to make sure the received file is the same as the sent file and do not encrypt or decrypt files.

## 4. Methodology

### A. MD5 Algorithm

As discussed above we have different methods to implement data integrity and check the legitimacy of the data in a certain file. Out of these methods we choose the MD5 algorithm because this method allows us to compare files easily compared to the other methods.

We implemented this algorithm by writing a code in python language. First, we created a file in the file explorer and put in some data. After this the python code is written on the software which represents and implements the MD5 algorithm. The file is then uploaded so that it can be read and processed by the software. The unique hash value of the file is retrieved by running the code. Then, we make a small change in the data of the same file i.e., tampering of the file. After this, we again run the code on the tampered file to get the hash value of this. We finally compare the hash value of the original file and the tampered file to see that it does not match. This way we have made sure there is data integrity.

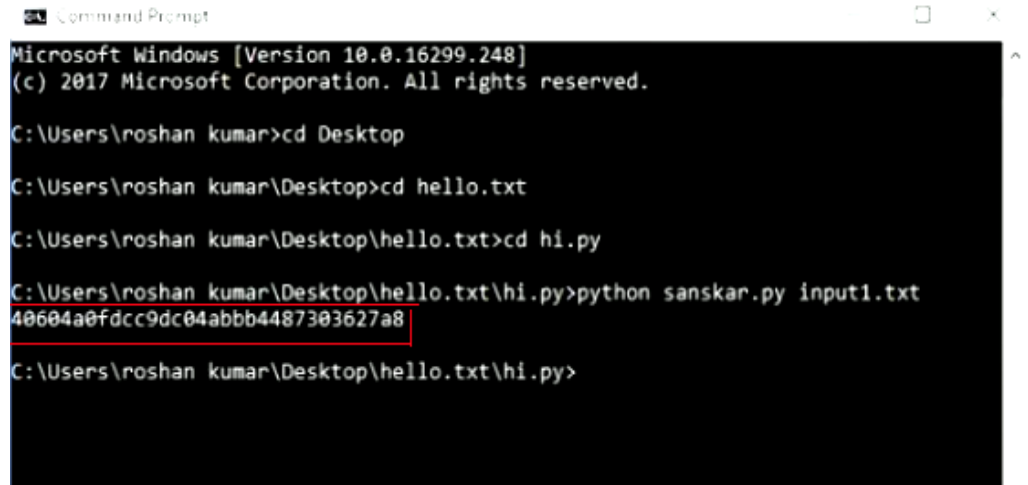
### B. MD2 Algorithm

Here we have used MD2 Integrity Algorithm to check the Integrity status of Message data after being transferred from IoT Transmitter to IOT receiver using Wi-Fi Communication Technology. For this we generate a HASH Data at IOT Transmitter using the MD2 Algorithm and Plain Message Data to be send. Then this HASH data is attached with the actual Message Data and send to IOT receiver from IOT Transmitter using Wi-Fi Technology. The Demarcation between HASH Data and Actual Message Data is known prior to both the IOT transmitter and Receiver.

At IOT Receiver the same MD2 Algorithm is processed on Actual Message Data and another HASH Data is created. This HASH data generated at Receiver is compared with the HASH data attached and send from transmitter. If both the generated and transmitted HASH are same, then we conclude that the data is not tampered or corrupted. If the generated HASH and Transmitted HASH are different we conclude that Message data send was tampered and Corrupted by intruder during transmission.

## 5. Results And Discussions

A file with some data has been taken and run the code to get hash value for this file as shown below.



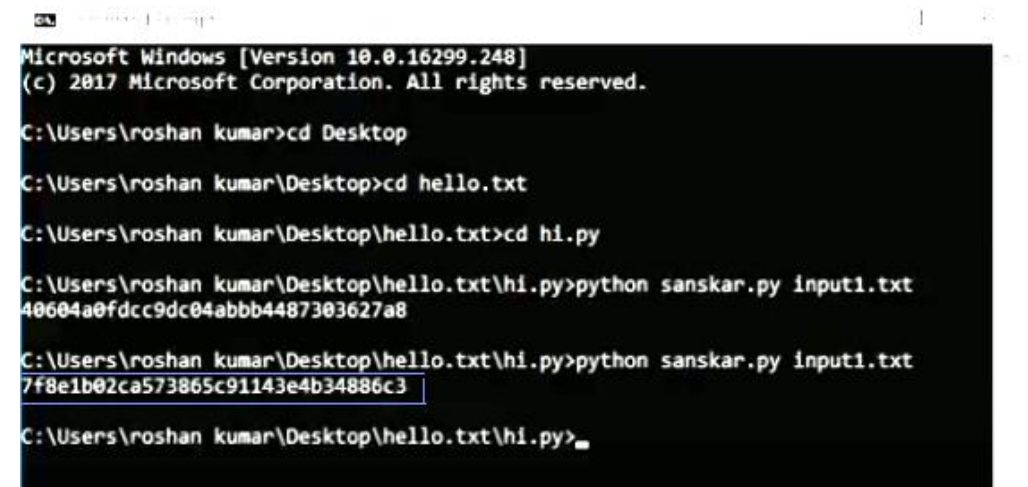
```

Microsoft Windows [Version 10.0.16299.248]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\roshan kumar>cd Desktop
C:\Users\roshan kumar\Desktop>cd hello.txt
C:\Users\roshan kumar\Desktop\hello.txt>cd hi.py
C:\Users\roshan kumar\Desktop\hello.txt\hi.py>python sanskar.py input1.txt
40604a0fdcc9dc04abbb4487303627a8
C:\Users\roshan kumar\Desktop\hello.txt\hi.py>
  
```

**Fig 2: Hash value of input file**

The input file data is tampered and produced the hash value of that file as shown in fig.3



```

Microsoft Windows [Version 10.0.16299.248]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\roshan kumar>cd Desktop
C:\Users\roshan kumar\Desktop>cd hello.txt
C:\Users\roshan kumar\Desktop\hello.txt>cd hi.py
C:\Users\roshan kumar\Desktop\hello.txt\hi.py>python sanskar.py input1.txt
40604a0fdcc9dc04abbb4487303627a8
C:\Users\roshan kumar\Desktop\hello.txt\hi.py>python sanskar.py input1.txt
7f8e1b02ca573865c91143e4b34886c3
C:\Users\roshan kumar\Desktop\hello.txt\hi.py>
  
```

**Fig 3: Hash value of tampered input file**

On comparison of both the values one can observe although they are the same files, the hash value is different for them because a change in data was made. So even if there is a smallest of a change, hash value differs and let us know that file has been tampered. This denotes that data integration hasn't been implied to the file as the input hash value does not match our comparison to tampered file hash value.

The code executes just when the hash estimation of the altered record is equivalent to the hash estimation of the info document which proposes there has been no altering never really input document and information is coordinated effectively.

In the method 2, we are using GCC compiler for Compiling Server program and Client Program as IOT transmitter and Receiver are working on Client Server Socket programming using Wi-Fi technology.

Fig. 4 below shows compilation and execution of Server Program for MD2 integrity algorithm at one of IOT Module using GCC compiler of Linux OS.

```
CC=gcc
CFLAGS=-I.

server: server.o MD2.o
    gcc -o server server.o MD2.o
clean:
    rm server *.o
~
~
~
```

**Fig 4: Server Make File**

Fig. 5 below shows compilation and execution of Client Program for MD2 integrity algorithm at one of IOT Module using gcc compiler of Linux OS.

```
CC=gcc
CFLAGS=-I.

server: client.o MD2.o
    gcc -o client client.o MD2.o
clean:
    rm client *.o
~
~
~
```

**Fig 5: Client Make File**

A Server and Client Program is written with MD2 integrity algorithm at IOT Modules using GCC compiler of Linux OS.

Fig. 6 below shows, MD2 integrity algorithm succeeded once Transmitted HASH value and generated HASH value at receiver are compared and found out to be same.

File Edit View Search Terminal Help	File Edit View Search Terminal Help
fatyaz@fatyaz-desktop:~/project/server\$ ./server	fatyaz@fatyaz-desktop:~/project/client\$ ./client
Listening	MD2 tests: SUCCEEDED
fatyaz@fatyaz-desktop:~/project/server\$	fatyaz@fatyaz-desktop:~/project/client\$

**Fig 6: (a) Server Output Screen (b) Client Output Screen**

## 6. Conclusions

IOT devices are used in health care systems, homes, city planning etc. If the data in such systems gets tampered with then it can cause huge problems. Even

if a small number is changed in the data fed to the device used health care system, the entire diagnosis will change. Hence, since everything depends on data and its integrity, it is important to make sure that we come up with methods to verify, encrypt and decrypt data. Some of these methods include implementation of algorithms like SHA-256, MD2, MD5, AES, DES. In the proposed methods, MD2 and MD5 is used because it is easier to compare hash values of files. MD5 algorithm gives the hash value of a file. We compare the hash values of the original file with the current file, if they match then data integrity is implemented. In MD2 algorithm, if both the generated and transmitted HASH values are same, then we concluded that the data is not tampered or corrupted. If the generated HASH values and Transmitted HASH values are different we conclude that Message or data sent was tampered and corrupted by intruder during transmission. Therefore, in order to benefit from the potential of all connected IOT devices, the need for a strict and reliable approach to security is essential.

## References

- Kakani Divya, Vinod Kumar.J, Raja Sujana.j, T.Pavan Kumar, "Enhanced IOT Accessing Security", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958, vol-8, issue-4, 2019.
- Anshul Jain, Tanya Singh, "Securing Communication in IoT Ecosystem Using Cryptographic Algorithms", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249-8958, vol-9, issue-1, 2019.
- Anjum Sheikh, Asha Ambhaikar, Sunil Kumar, "Quality of Services Improvement for Secure IoT Networks", International Journal of Engineering and Advanced Technology (IJEAT) ISSN: 2249-8958, vol-9, issue-2, 2019.
- Bhagya Shree, Suman Bhakar, "Securing the IOT Devices with Artificial Immune System", International Journal of Engineering and Advanced Technology (IJEAT) ISSN:2249-8958, vol-9, issue-2, 2019.
- B.Nagajayanthi, "Energy Efficient Light weight Security Algorithm for Low Power IOT Devices", International Journal of Engineering and Advanced Technology (IJEAT) ISSN:2249-8958, vol-9, issue-1, 2019.
- Eden Au, "Building a Minimal Block chain in Python, understanding block chain by coding", published in edenau.github.io., 2019.
- Dr. Jason Polak, Aleph Zero Categorical, "A python 3 implementation of the MD5 hash", 2018.
- Chalwe Musonda, Monica M.K.–Kabemba, Mayumbo Nyirenda, Jackson Phiri, "Security, Privacy and Integrity in Internet of Things – A Review", proceedings of the ICTSZ international conference in ICTs (ICICT2018) - Lusaka, Zambia, 2018.
- <http://files.iccmmedia.com/pdf/windriver160823.pdf>