Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE
TECHNOLOGY, RESEARCH, SOCIAL INNOVATION & PARTNERSHIPS

T.Y.B.Tech (CSE)

Information Security

Lab Assignment No – B2

Name: Aniruddha Shende

Roll number: PE04

Batch: E1

Panel: E

PE04 Aniruddha Shende

Name:- Aniruddha Arun Shende.
Roll no:- PE04
Batch :- E1
Panel :- E

Subject :- Information Security.

IS LAB ASSIGNMENT - B2

Aim:- To program basic cryptography hash algorithm SHA1 / MD5 use Java or Python or C++API. Additionally demonstrate client server authentication using socket programming

Objective:-

1. To program code for SHA using API.
2. To learn about cryptographic hash algorithm.

Theory:-

★ Threats to Data Integrity:

Passive Threats: These data errors are likely to occur due to noise in communication channel. Error-correcting codes & simple checksums like CRC's are used to detect loss of data integrity.

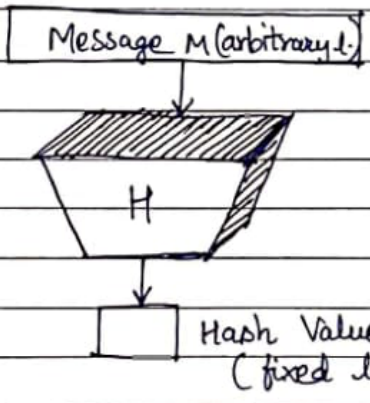
Active Threats: At simplest level, if data is without digest, it can be modified without detection

PE04 Aniruddha Shende

The system can use techniques of appending CRC to data for detecting any active modification.

- Security mechanism such as Hash functions are used to tackle the active modification threats.

A hash function is a mathematical function that converts a numerical input value ~~for~~ into another compressed numerical value. The input to the hash function is of arbitrary length but output is fixed length.



Features of Hash Function :-

- Fixed length Output (Hash Value)
- Efficiency of Operation

Property of Hash functions:

- Pre-Image Resistance
- Second Pre-Image Resistance
- Collision Resistance.

Secure Hash Function (SHA1):

- The original version is SHA-0, a 160-bit hash function, published by the NIST in 1993

PE04 Aniruddha Shende

- SHA-1 is the most widely used of the existing hash function.
- SHA family has 4 other variants: SHA-0, SHA-1, SHA-2, & SHA-3.
- SHA-2 family has four further SHA variants, SHA-224, SHA-256, SHA-384, SHA-512.

SHA algorithms

Conclusion:- Hence, we have successfully implemented basic cryptography hash algorithm.

★ FAQ's:-

1) What is role of digest algorithm?

Ans 1) A message digest algorithm / hash function is a procedure that maps input data to a arbitrary length to fixed length.

2) How digest algorithm are used for digital signature?

Ans 2) The generated message digest algorithm with DSA algorithm is what gives the digital signature. Signature is sent along with message at receving & same hash is used to authenticate source & data via digest algorithm

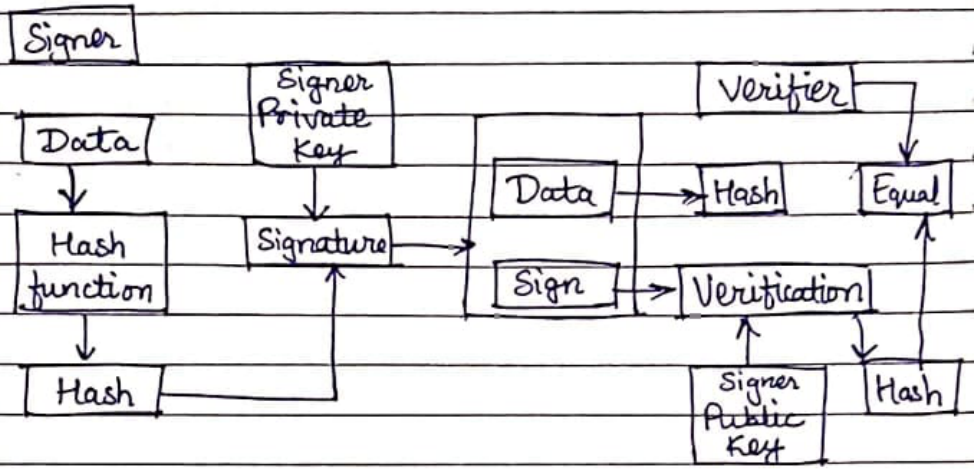
3) What are the properties of digest algorithm?

Ans 3) Message digest is used to ensure the integrity of a message transmitted over an insecure channel. The message is panel by Hash function.

PE04 Aniruddha Shende

4) Draw the diagram of tamper proof verification.

Ans 4)



5) How digest algorithms are used for software distribution?

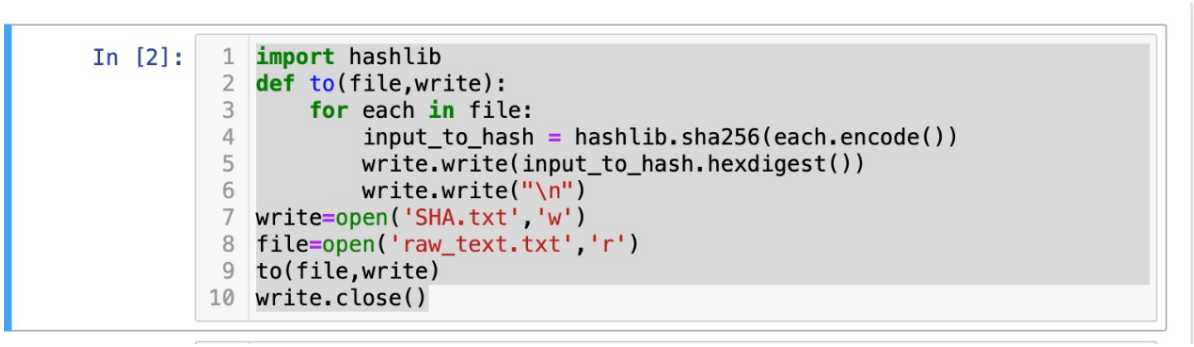
Ans 5) Digest algorithms help to safeguard against piracy & crack software. This ensure legitimacy of software during its distribution.

6) How digest algorithm are used for anti-virus solutions?

Ans 6) The anti-virus checks for big patterns in certain area of each file. If they are found, then there is a virus present. This like signature checks can be done by digest algorithm.

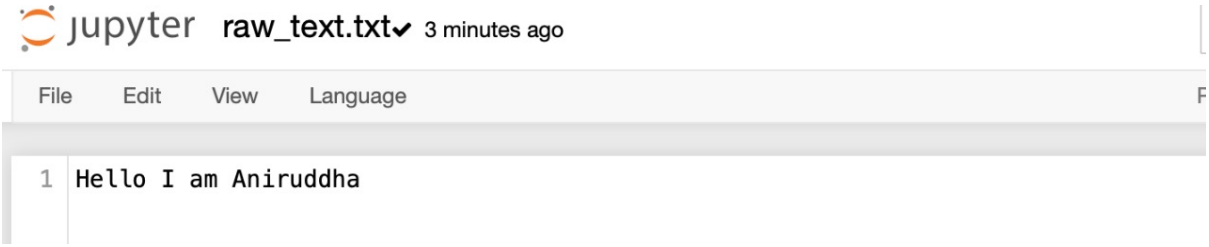
SHA Code :

```
import hashlib
def to(file,write):
    for each in file:
        input_to_hash = hashlib.sha256(each.encode())
        write.write(input_to_hash.hexdigest())
        write.write("\n")
write=open('SHA.txt','w')
file=open('raw_text.txt','r')
to(file,write)
write.close()
```



```
In [2]:
import hashlib
def to(file,write):
    for each in file:
        input_to_hash = hashlib.sha256(each.encode())
        write.write(input_to_hash.hexdigest())
        write.write("\n")
write=open('SHA.txt','w')
file=open('raw_text.txt','r')
to(file,write)
write.close()
```

File which contains the raw text



Resultant text file

Client server authentication using socket programming :

Server code :

```
//Name : Aniruddha Shende
//Roll no : PE04
//Batch : E1
//Panel : E
#include <unistd.h>
#include <stdio.h>
#include <sys/socket.h>
#include <stdlib.h>
#include <netinet/in.h>
#include <string.h>

int main()
{
    int server_fd, new_socket;
    struct sockaddr_in address;
    int addrlen = sizeof(address);
    char server_buffer[1024] = {0};
    char message[1024] = {0};

    if ((server_fd = socket(AF_INET, SOCK_STREAM, 0)) == 0)
    {
        perror("\n\nsocket failed to create");
        exit(0);
    }
    address.sin_family = AF_INET;
    address.sin_addr.s_addr = INADDR_ANY;
    address.sin_port = htons(6542);
```

```
    bind(server_fd, (struct sockaddr *)&address, sizeof(address));
    listen(server_fd, 3);
    new_socket = accept(server_fd, (struct sockaddr *)&address,
(socklen_t *)&addrlen);
    printf("\n\nClient Connected!\n");
    int z = 0;
    while (z!=1)
    {
        memset(&server_buffer, '\0', 1024);
        memset(&message, '\0', 1024);

        if (read(new_socket, server_buffer, 1024) < 0)
        {
            printf("\n\nCan't Listen...");
        }
        printf("\n\nChallenge number received from the client is :
%s", server_buffer);
        if (strcmp(server_buffer, "exit") == 0)
            break;
        printf("\n\nCipher reply sent to client is : ");
        gets(message);

        if (send(new_socket, message, strlen(message), 0) < 0)
        {
            printf("\n\nMessage not sent");
        }
        if (strcmp(message, "exit") == 0)
            break;
        z++;
    }

    close(server_fd);
    close(new_socket);
```

```
    printf("\n");
    return 0;
}
```

Client code :

```
//Name : Aniruddha Shende
//Roll no : PE04
//Batch : E1
//Panel : E
#include <stdio.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <unistd.h>
#include <string.h>

int main()
{
    int sock = 0;
    struct sockaddr_in serv_addr;
    char message[1024] = {0};
    char client_buffer[1024] = {0};
    if ((sock = socket(AF_INET, SOCK_STREAM, 0)) < 0)
    {
        printf("\n\nSocket creation error \n");
        exit(0);
    }

    serv_addr.sin_family = AF_INET;
    serv_addr.sin_port = htons(6542);
    inet_pton(AF_INET, "127.0.0.1", &serv_addr.sin_addr);
```

```
    if (connect(sock, (struct sockaddr *)&serv_addr,
sizeof(serv_addr)) < 0)
    {
        printf("\nConnection Failed \n");
        exit(0);
    }
    int z = 0;
    while (z!=1)
    {
        memset(&client_buffer, '\0', 1024);
        memset(&message, '\0', 1024);

        printf("\n\nEnter the number to be sent to the server :
");
        gets(message);
        printf("\n\nChallenge number sent to the server is :
%s",message);
        if (send(sock, message, strlen(message), 0) < 0)
        {
            printf("\n\nMessage not sent");
        }

        if (strcmp(message, "exit") == 0)
            break;

        if (read(sock, client_buffer, 1024) < 0)
        {
            printf("\n\nCan't Listen...");
        }
        printf("\n\nCipher reply received from the server is :
%s", client_buffer);
        if (strcmp(client_buffer, "exit") == 0)
            break;
        z++;
```

```
    }
    printf("\n\nREcovered number is : %s",message);
    printf("\n\nClient has successfully verified the server\n");
    close(sock);
    printf("\n");
    return 0;
}
```

Output Screenshot :

On server side

```
cd "/Users/ani/Downloads/Client server code/" && gcc server_test.
c -o server_test && "/Users/ani/Downloads/Client server code/"ser
ver_test
[3] 39763
ani@Aniruddhas-MacBook-Pro Client server code % cd "/Users/ani/Do
wnloads/Client server code/" && gcc server_test.c -o server_test
&& "/Users/ani/Downloads/Client server code/"server_test


Client Connected!


Challenge number received from the client is : 41

warning: this program uses gets(), which is unsafe.
Cipher reply sent to client is : 4

ani@Aniruddhas-MacBook-Pro Client server code %
```

On client Side

```
[3] 39786
ani@Aniruddhas-MacBook-Pro Client server code % cd "/Users/ani/Do
wnloads/Client server code/" && gcc client_test.c -o client_test
&& "/Users/ani/Downloads/Client server code/"client_test


warning: this program uses gets(), which is unsafe.
Enter the number to be sent to the server : 41


Challenge number sent to the server is : 41

Cipher reply received from the server is : 4

REcovered number is : 41

Client has successfully verified the server

ani@Aniruddhas-MacBook-Pro Client server code %
```