



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

Langkah Praktikum

Hashing

1. Pastikan **md5sum** telah terinstall di perangkat yang digunakan.

```
user@kali: ~  
File Actions Edit View Help  
(user@kali)-[~]  
$ which md5sum  
/usr/bin/md5sum  
(user@kali)-[~]  
$
```

~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3 |

Penjelasan :

Output dari perintah `which md5sum` adalah `/usr/bin/md5sum`, berarti `md5sum` sudah terinstal dan siap digunakan.

Fungsi utama dari `md5sum` adalah untuk menghasilkan hash nilai MD5 dari sebuah berkas atau teks. Hash nilai MD5 digunakan untuk memverifikasi integritas berkas, karena perubahan sedikit pun pada berkas akan menghasilkan hash nilai yang berbeda.



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

2. Download file pada tautan berikut (kamu bisa menggunakan command wget) dan ekstrak.
 - a. message1.bin & message2.bin

wget <https://github.com/isfahany/infosec-module-downloadable-file/raw/master/collision-example/file.zip>

```
(user@kali) - [~/Documents/praktikum5]
$ wget https://github.com/isfahany/infosec-module-downloadable-file/raw/master/collision-example/file.zip
--2024-05-10 19:49:45-- https://github.com/isfahany/infosec-module-downloadable-file/raw/master/collision-example/file.zip
Resolving github.com (github.com)... 20.205.243.166
Connecting to github.com (github.com)|20.205.243.166|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://raw.githubusercontent.com/isfahany/infosec-module-downloadable-file/master/collision-example/file.zip [following]
--2024-05-10 19:49:45-- https://raw.githubusercontent.com/isfahany/infosec-module-downloadable-file/master/collision-example/file.zip
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 454 [application/zip]
Saving to: 'file.zip'

file.zip                               100%[=====] 454 --KB/s in 0s

2024-05-10 19:49:46 (56.2 MB/s) - 'file.zip' saved [454/454]

(user@kali) - [~/Documents/praktikum5]
$ ls
file.zip
(user@kali) - [~/Documents/praktikum5]
$ unzip file.zip
Archive: file.zip
  extracting: message1.bin
  extracting: message2.bin
(user@kali) - [~/Documents/praktikum5]
$
```

The screenshot also shows a file explorer window titled "DocumentName - Mousetrap" containing the following text:

```
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

Penjelasan:

wget: perintah untuk mengunduh file dari web

Perintah unzip akan digunakan untuk mengekstrak. File didalamnya berupa message1.bin dan message2.bin

3. Jalankan command berikut pada file terkait.
 - a. sha1sum



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
sha1sum message1.bin message2.bin
```

```
(user@kali) ~/Documents/praktikum5
$ cat message1.bin
M+h\  r#w{r+o+VJ=+x>{+ (K+n+KU+ _Bu+Igm+U]+`+_+

(user@kali) ~/Documents/praktikum5
$ cat message2.bin
M+h\  r#w{r+o+VJ=+x>{+ (K+n+KU+ _Bu+Igm+U]+`+_+

(user@kali) ~/Documents/praktikum5
$ sha1sum message1.bin message2.bin
c6b384c4968b28812b676b49d40c09f8af4ed4cc  message1.bin
c728d8d93091e9c7b87b43d9e33829379231d7ca  message2.bin

(user@kali) ~/Documents/praktikum5
$
```

```
~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3 |
```

Penjelasan :
Output `sha1sum` berupa SHA-1 dari setiap berkas. SHA-1 adalah nilai hash yang dihasilkan dari algoritma fungsi hash SHA-1 (Secure Hash Algorithm 1).

Setiap berkas memiliki hash nilai SHA-1 yang berbeda, yang menunjukkan data di dalam berkas tersebut berbeda. Dapat dibandingkan apakah kedua berkas memiliki konten (isi) yang sama atau berbeda.

- Hash nilai SHA-1 dari `message1.bin` adalah :
c6b384c4968b28812b676b49d40c09f8af4ed4cc.
- Hash nilai SHA-1 dari `message2.bin` adalah :
c728d8d93091e9c7b87b43d9e33829379231d7ca.

Kedua hash nilai nya berbeda, artinya kedua berkas memiliki konten yang berbeda.

b. `sha256sum`

```
sha256sum message1.bin message2.bin
```

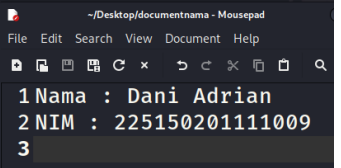


LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ sha256sum message1.bin message2.bin
54bcb9a4fda31e4f254303e3959acd5e420ad18a80949d56a3000c3716fbd1a0 message1.bin
90774a6455a2bdb7d106e533923ecbefe81392ca55bed0ce81cfab2c1a7f0afe message2.bin

(user@kali)-[~/Documents/praktikum5]
$
```



Penjelasan :

Output dari `sha256sum` adalah hash nilai SHA-256 dari setiap berkas. Hash nilai SHA-256 adalah nilai hash yang dihasilkan dari algoritma fungsi hash SHA-256 (Secure Hash Algorithm 256-bit).

- Hash nilai SHA-256 dari `message1.bin` adalah :
54bcb9a4fda31e4f254303e3959acd5e420ad18a80949d56a3000c3716fbd1a0.
- Hash nilai SHA-256 dari `message2.bin` adalah :
90774a6455a2bdb7d106e533923ecbefe81392ca55bed0ce81cfab2c1a7f0afe.

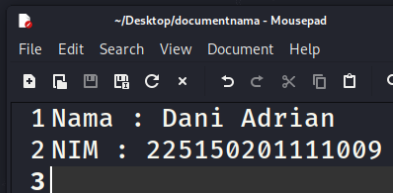
Kedua hash nilai berbeda, artinya kedua berkas tersebut memiliki konten yang berbeda.

c. `md5sum`

```
md5sum message1.bin message2.bin
```

```
(user@kali)-[~/Documents/praktikum5]
$ md5sum message1.bin message2.bin
008ee33a9d58b51cfef425b0959121c9 message1.bin
008ee33a9d58b51cfef425b0959121c9 message2.bin

(user@kali)-[~/Documents/praktikum5]
$
```



Penjelasan :



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

Output `md5sum` adalah hash nilai MD5 dari setiap berkas yang diberikan. Hash nilai MD5 adalah nilai hash yang dihasilkan dari algoritma fungsi hash MD5 (Message-Digest Algorithm 5).

- Hash nilai MD5 dari `message1.bin` adalah `008ee33a9d58b51cf425b0959121c9`.
- Hash nilai MD5 dari `message2.bin` juga adalah `008ee33a9d58b51cf425b0959121c9`.

Faktor kekurangan `md5sum` adalah memiliki kekurangan dalam menangani hash collision, di mana dua input berbeda dapat menghasilkan nilai hash yang sama. Hal ini menyebabkan mengapa nilai hash `md5sum` untuk `message1.bin` dan `message2.bin` sama.

4. Analisislah output dari nomor tiga. Menurutmu, mengapa hal tersebut bisa terjadi? Hash function mana yang lebih aman digunakan?

Pada output `md5sum`, nilai hash untuk `message1.bin` dan `message2.bin` sama, yaitu `008ee33a9d58b51cf425b0959121c9`. Ini dikarenakan algoritma `md5sum` memiliki kelemahan yang disebut **hash collision**.

Hash collision terjadi ketika dua input yang berbeda menghasilkan nilai hash yang sama. Dalam kasus ini, `message1.bin` dan `message2.bin` memiliki konten yang berbeda, namun menghasilkan nilai `md5sum` yang sama. Ini terjadi karena algoritma `md5sum` tidak memiliki distribusi nilai hash yang merata.

Kelemahan hash collision pada `md5sum` membuatnya rentan terhadap serangan. Contohnya, seorang penyerang dapat membuat dua file yang berbeda dengan nilai `md5sum` yang sama. Salah satu file dapat dibuat sebagai file yang berbahaya, dan file lainnya dibuat sebagai file yang tampaknya sah. Penyerang kemudian dapat mendistribusikan file yang berbahaya dengan nama file yang sah, menipu pengguna untuk mengunduhnya dan menjalankannya.



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

Alasan md5sum Kurang Aman:

Hash collision: Kelemahan utama md5sum adalah kerentanannya terhadap hash collision.

Preimage attack: Preimage attack memungkinkan penyerang untuk menemukan input yang menghasilkan nilai hash md5sum tertentu. Hal ini dapat digunakan untuk memalsukan tanda tangan digital atau memverifikasi file yang berbahaya.

Second preimage attack: Second preimage attack memungkinkan penyerang untuk menemukan input lain yang menghasilkan nilai hash md5sum yang sama dengan input yang dipilih. Hal ini dapat digunakan untuk menyembunyikan file berbahaya dalam file lain.

Hash function yang lebih aman:

Karena kelemahan md5sum, disarankan untuk menggunakan hash function yang lebih aman untuk integritas data dan keamanan yang kuat, yaitu :

- SHA-256: Algoritma hash yang populer dan aman dengan output 256 bit. SHA-256 lebih tahan terhadap hash collision dan serangan lainnya dibandingkan md5sum.
- SHA-3: Algoritma hash terbaru dari NIST (*National Institute of Standards and Technology*) dengan output 256 bit. SHA-3 menawarkan tingkat keamanan yang lebih tinggi dibandingkan SHA-256 dan direkomendasikan untuk aplikasi yang membutuhkan keamanan terbaik.

Encoding

1. Pastikan python telah terinstall di perangkat yang digunakan

```
python3 --version
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ python3 --version
Python 3.11.8

(user@kali)-[~/Documents/praktikum5]
$
```

```
~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

2. Buka file dengan nama encode.py (file akan langsung terbuat), lalu copy-paste kode berikut:

```
import binascii
import base64

text = "Road to Security Engineer"
text_utf8 = text.encode("utf-8")
text_b64 = base64.b64encode(text_utf8)
text_hex = binascii.hexlify(text_utf8)

print("text = " + text)
print("utf8 encode = " + str(text_utf8))
print("base64 encode = " + str(text_b64))
print("hexadec encode = " + str(text_hex))
```

Apa yang dilakukan kode tersebut ?



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ ls
file.zip message1.bin message2.bin

(user@kali)-[~/Documents/praktikum5]
$ nano encode.py
```

```
GNU nano 7.2 encode.py *
import binascii
import base64

text = "Road to Security Engineer"
text_utf8 = text.encode("utf-8")
text_b64 = base64.b64encode(text_utf8)
text_hex = binascii.hexlify(text_utf8)

print("text = " + text)
print("utf8 encode = " + str(text_utf8))
print("base64 encode = " + str(text_b64))
print("hexadec encode = " + str(text_hex))

File Name to Write: encode.py
^G Help      M-D DOS Format  M-A Append
^C Cancel    M-M Mac Format  M-P Prepend
```




LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ ls
encode.py  file.zip  message1.bin  message2.bin

(user@kali)-[~/Documents/praktikum5]
$ python encode.py
text = Road to Security Engineer
utf8 encode = b'Road to Security Engineer'
base64 encode = b'Um9hZCB0byBTZWNIcmloeSBFbmdpbmVlcg=='
hexadec encode = b'526f616420746620536563757269747920456e67696e656572'
```

~Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3

Penjelasan :

Tujuan kode tersebut adalah mengilustrasikan proses encode teks menggunakan beberapa metode yang umum digunakan dalam pemrosesan data, yaitu UTF-8, base64, dan hexadecimal encoding.

Dengan menggunakan kode tersebut, kita dapat melihat bagaimana teks "Road to Security Engineer" diubah menjadi representasi yang berbeda-beda.

- Konversi ke UTF-8: Teks string diubah menjadi urutan byte UTF-8 menggunakan metode encode("utf-8").
- Base64 Encoding: Teks string yang diubah menjadi urutan byte UTF-8 dienkripsi menggunakan base64 encoding menggunakan fungsi base64.b64encode().
- Hexadecimal Encoding: Teks string yang diubah menjadi urutan byte UTF-8 dienkripsi ke dalam format hexadecimal menggunakan fungsi binascii.hexlify().

3. Tambahkan fungsionalitas untuk mengembalikan base64 dan hexadecimal encode pada source code python tersebut menjadi semula.



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
import binascii
import base64

text = "Road to Security Engineer"
text_utf8 = text.encode("utf-8")
text_b64 = base64.b64encode(text_utf8)
text_hex = binascii.hexlify(text_utf8)

# Decode base64
decoded_base64 = base64.b64decode(text_b64).decode("utf-8")

# Decode hexadecimal
decoded_hex = binascii.unhexlify(text_hex).decode("utf-8")

print("text = " + text)
print("utf8 encode = " + str(text_utf8))
print("base64 encode = " + str(text_b64))
print("base64 decode = " + decoded_base64) # Menambahkan
base64 decode
print("hexadec encode = " + str(text_hex))
print("hexadec decode = " + decoded_hex) # Menambahkan
hexadecimal decode
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ nano encode.py

~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3

user@kali: ~/Documents/praktikum5
GNU nano 7.2 encode.py *
import binascii
import base64

text = "Road to Security Engineer"
text_utf8 = text.encode("utf-8")
text_b64 = base64.b64encode(text_utf8)
text_hex = binascii.hexlify(text_utf8)

# Decode base64
decoded_base64 = base64.b64decode(text_b64).decode("utf-8")

# Decode hexadecimal
decoded_hex = binascii.unhexlify(text_hex).decode("utf-8")

print("text = " + text)
print("utf8 encode = " + str(text_utf8))
print("base64 encode = " + str(text_b64))
print("base64 decode = " + decoded_base64) # Menambahkan base64 decode
print("hexadec encode = " + str(text_hex))
print("hexadec decode = " + decoded_hex) # Menambahkan hexadecimal decode

File Name to Write: encode.py
^G Help M-D DOS Format M-A Append
^C Cancel M-M Mac Format M-P Prepend
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ python encode.py
text = Road to Security Engineer
utf8 encode = b'Road to Security Engineer'
base64 encode = b'Um9hZCB0byBTZW1cm10eSBFbmdpbmVlcg=='
base64 decode = Road to Security Engineer
hexadec encode = b'526f616420746f20536563757269747920456e67696e656572'
hexadec decode = Road to Security Engineer

(user@kali)-[~/Documents/praktikum5]
$
```

Encoding

1. Pastikan python telah terinstall di perangkat yang digunakan

```
python3 --version
```

```
(user@kali)-[~/Documents/praktikum5]
$ python3 --version
Python 3.11.8

(user@kali)-[~/Documents/praktikum5]
$
```

2. Buka file dengan nama caesar.py (file akan langsung terbuat), lalu copy-paste kode berikut:

```
#!/usr/bin/python3

plaintext = "Infosec"
ciphertext = ""
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
for i in range(len(plaintext)):  
    if(plaintext[i].isupper()):  
        ciphertext += chr((ord(plaintext[i]) + 7  
- 65) % 26 + 65)  
    else:  
        ciphertext += chr((ord(plaintext[i]) + 7  
- 97) % 26 + 97)  
print("plaintext = " + plaintext)  
print("ciphertext = " + ciphertext)
```

```
(user@kali)-[~/Documents/praktikum5]  
$ ls  
encode.py  file.zip  message1.bin  message2.bin  
  
(user@kali)-[~/Documents/praktikum5]  
$ nano caesar.py
```

~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3 |



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

The screenshot displays a Kali Linux terminal window with the nano text editor open, editing a file named `caesar.py`. The code implements a Caesar cipher encryption for the plaintext "Infosec". It iterates through each character, shifting uppercase letters by 7 positions forward and lowercase letters by 7 positions backward in the alphabet. The output shows the original plaintext and the resulting ciphertext "Jvphvjd".

```
GNU nano 7.2 caesar.py
#!/usr/bin/python3

plaintext = "Infosec"
ciphertext = ""
for i in range(len(plaintext)):
    if(plaintext[i].isupper()):
        ciphertext += chr((ord(plaintext[i]) + 7 - 65) % 26 + 65)
    else:
        ciphertext += chr((ord(plaintext[i]) + 7 - 97) % 26 + 97)
print("plaintext = " + plaintext)
print("ciphertext = " + ciphertext)
```

Below the terminal, a terminal window shows the execution of the program. The user runs `ls` in the directory `~/Documents/praktikum5`, listing files including `caesar.py`, `encode.py`, `file.zip`, `message1.bin`, and `message2.bin`. A second terminal window shows the execution of the program, which outputs the name and NIM of the student.

```
(user@kali)-[~/Documents/praktikum5]
$ ls
caesar.py  encode.py  file.zip  message1.bin  message2.bin

(user@kali)-[~/Documents/praktikum5]
$
```

The Mousepad window shows the output of the program:

```
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

The screenshot shows a terminal window with the following commands and output:

```
(user@kali)-[~/Documents/praktikum5]
$ python caesar.py
plaintext = Infosec
ciphertext = Pumvzlj
```

Below the terminal is a text editor window titled "~Desktop/documentnama - Mousepad". It contains the following text:

```
1 Nama : Dani Adrian
2 NIM : 225150201111009
3 |
```

3. Apa yang terjadi pada line 7 dan 9? Mengapa ada angka 65, 97, dan 26 disana?

Line 7 dan 9 pada kode melakukan operasi enkripsi Caesar pada huruf-huruf dalam plaintext.

Line 7

Jika karakter dalam plaintext adalah huruf kapital (uppercase), dilakukan operasi :

```
ciphertext += chr((ord(plaintext[i]) + 7 - 65) % 26 + 65)
```

65 untuk mengubah posisi Unicode kembali ke posisi huruf A dalam ASCII (65 adalah nilai Unicode untuk huruf A dalam huruf kapital).

% 26 untuk memastikan bahwa nilai yang dihasilkan tetap dalam rentang 0 hingga 25 (0-25 mengelilingi A-Z dalam alfabet).



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

Line 9

Jika karakter dalam plaintext adalah huruf kecil (lowercase), dilakukan operasi :

```
ciphertext += chr((ord(plaintext[i]) + 7 - 97) % 26 + 97)
```

97 digunakan untuk mengubah posisi Unicode kembali ke posisi huruf a dalam ASCII (65 adalah nilai Unicode untuk huruf a).

% 26 untuk memastikan bahwa nilai yang dihasilkan tetap dalam rentang 0 hingga 25 (0-25 mengelilingi a-z dalam alfabet).

4. Kriptografi apakah kode tersebut? Bagaimana caranya bekerja?

Kriptografi tersebut merupakan contoh sederhana dari **metode enkripsi Caesar**. Dinamakan demikian karena Julius Caesar, yang menurut Suetonius, menggunakannya dengan pergeseran tiga huruf (A menjadi D saat dienkripsi, dan D menjadi A saat didekripsi) untuk melindungi pesan-pesan penting militer. Meskipun Caesar adalah pengguna pertama yang tercatat menggunakan skema ini, sandi substitusi lain diketahui telah digunakan sebelumnya.

“Jika ia memiliki suatu rahasia yang akan disampaikan, ia menuliskannya dalam sandi, dengan mengganti urutan abjad, sehingga tak satu kata pun dapat dimengerti. Jika ada yang ingin membaca pesan-pesan ini, ia harus mengganti huruf keempat dalam alfabet, yaitu D, untuk A, dan seterusnya untuk huruf-huruf lain.” - Suetonius, Life of Julius Caesar

Metode Caesar bekerja dengan menggeser setiap huruf dalam teks plaintext sejumlah langkah tertentu dalam alfabet. Dalam contoh kode tersebut, setiap huruf digeser



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

sebanyak 7 langkah ke depan. Proses ini menghasilkan teks ciphertext yang nantinya dapat dikirimkan secara aman.

- Jika karakter adalah **huruf kapital**, kode menambahkan karakter hasil enkripsi ke dalam ciphertext menggunakan rumus enkripsi Caesar untuk huruf kapital.
- Jika karakter adalah **huruf kecil**, kode menambahkan karakter hasil enkripsi ke dalam ciphertext menggunakan rumus enkripsi Caesar untuk huruf kecil.
- Jika karakter **bukan huruf**, kode tidak melakukan enkripsi dan langsung menambakkannya ke dalam ciphertext.

5. Optimalkan cara kerja algoritma kriptografi tersebut! Ambil input dari user dan juga shift lompatan angka dari user (interactive) !

```
plaintext = input("Masukkan teks plaintext: ")
shift = int(input("Masukkan jumlah langkah geser: "))

ciphertext = ""
for char in plaintext:
    if char.isupper():
        ciphertext += chr((ord(char) + shift - 65) % 26
+ 65)
    elif char.islower():
        ciphertext += chr((ord(char) + shift - 97) % 26
+ 97)
    else:
        ciphertext += char

print("plaintext = " + plaintext)
print("ciphertext = " + ciphertext)
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

The image shows a terminal window and a text editor window. The terminal window displays the command prompt for a user on a Kali machine, showing the current directory as ~/Documents/praktikum5. The user has opened a file named caesar.py using the nano text editor. The text editor window shows the content of caesar.py, which is a Python script for a Caesar cipher. The script prompts the user for a plaintext and a shift value, then calculates the ciphertext. The terminal window shows the output of the program, displaying the plaintext and the resulting ciphertext.

```
(user@kali)-[~/Documents/praktikum5]
$ nano caesar.py
```

```
~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

```
user@kali: ~/Documents/praktikum5
GNU nano 7.2 caesar.py *
plaintext = input("Masukkan teks plaintext: ")
shift = int(input("Masukkan jumlah langkah geser: "))

ciphertext = ""
for char in plaintext:
    if char.isupper():
        ciphertext += chr((ord(char) + shift - 65) % 26 + 65)
    elif char.islower():
        ciphertext += chr((ord(char) + shift - 97) % 26 + 97)
    else:
        ciphertext += char

print("plaintext = " + plaintext)
print("ciphertext = " + ciphertext)
```

```
~/Desktop/documentnama - Mousepad
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

The screenshot shows a terminal window with the following commands and output:

```
(user@kali)-[~/Documents/praktikum5]
$ python caesar.py
Masukkan teks plaintext: Augustus
Masukkan jumlah langkah geser: 7
plaintext = Augustus
ciphertext = Hbnzbabz

(user@kali)-[~/Documents/praktikum5]
$ python caesar.py
Masukkan teks plaintext: Veni,Vidi,Vici
Masukkan jumlah langkah geser: 3
plaintext = Veni,Vidi,Vici
ciphertext = Yhql,Ylgf,Ylfl

(user@kali)-[~/Documents/praktikum5]
$
```

Overlaid on the terminal is a text editor window titled "~/Desktop/documentnama - Mousepad" containing the following text:

```
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

Penjelasan :

Kode ini, menggunakan fungsi `input()` untuk mengambil input dari user. User diminta untuk memasukkan teks plaintext dan jumlah langkah geser. Variabel `shift` akan berisi jumlah langkah geser yang dimasukkan oleh pengguna.

Kemudian, kode melakukan loop melalui setiap karakter dalam plaintext dan melakukan enkripsi sesuai dengan jumlah langkah geser yang ditentukan oleh user. Hasil enkripsi disimpan dalam variabel `ciphertext`.

Setelah loop selesai, teks `plaintext` dan `ciphertext` dicetak untuk ditampilkan kepada pengguna. Sehingga, algoritma kriptografi Caesar menjadi lebih interaktif dan lebih mudah digunakan user.

Tugas

1. Jelaskan perbedaan hash, enkripsi, dan encoding!



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

Hashing:

- Hashing adalah proses mengubah data (teks, file, dll.) menjadi nilai hash yang unik.
- Tujuan utama hashing adalah untuk menghasilkan nilai hash yang konsisten untuk setiap input yang diberikan.
- Hash function digunakan untuk menghasilkan nilai hash, yang biasanya memiliki panjang yang tetap, tidak peduli seberapa besar atau kecil data inputnya.
- Contoh algoritma hashing termasuk MD5, SHA-1, SHA-256, dan sebagainya.
- Hasil hashing tidak dapat diubah kembali menjadi data asli (tidak reversibel).
- Hashing sering digunakan untuk mengamankan kata sandi, verifikasi integritas data, dan pembuatan identifikasi unik untuk data.

Enkripsi:

- Enkripsi adalah proses mengubah data asli menjadi bentuk yang tidak terbaca atau tidak dapat dimengerti tanpa kunci enkripsi yang sesuai.
- Tujuan utama enkripsi adalah untuk menyembunyikan atau melindungi data dari akses yang tidak sah.
- Enkripsi menggunakan algoritma enkripsi dan kunci enkripsi untuk mengubah data menjadi bentuk terenkripsi yang aman.
- Hasil enkripsi dapat didekripsi kembali menjadi data asli menggunakan kunci yang tepat.
- Enkripsi sering digunakan untuk melindungi informasi sensitif dalam penyimpanan atau transmisi, seperti data pengguna, pesan rahasia, dan sebagainya.



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

Encoding:

- Encoding adalah proses mengubah data dari satu format ke format lain, biasanya dalam konteks representasi teks.
- Tujuan utama encoding adalah untuk memastikan bahwa data dapat disampaikan dan disimpan dengan benar tanpa kehilangan informasi.
- Encoding tidak menyembunyikan data atau melindunginya dari akses yang tidak sah; hanya mengubah cara data direpresentasikan.
- Beberapa jenis encoding umum termasuk Base64, ASCII, UTF-8, dan sebagainya.
- Hasil encoding dapat didekode kembali ke bentuk aslinya tanpa kehilangan informasi.
- Encoding sering digunakan dalam pertukaran data, komunikasi, dan penyimpanan data untuk memastikan konsistensi format dan interoperabilitas.

2. Diberikan source code berikut, buatlah dekripsi dalam bahasa pemrograman python:

```
encrypt.py
#!/usr/bin/python3
def encrypt(plaintext):
    plaintext = plaintext[::-1]
    ciphertext = ""
    for i in plaintext:
        copy = "X" * ((ord(i) ^ 0x50) + 9)
        copy += "-"
        ciphertext += copy
    return ciphertext

print ("Infosec Module Encryptor")
plaintext = input("Masukkan string yang ingin di-encrypt: ")
print ("Result : ")
```



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
print (encrypt(plaintext))
```

```
decrypt.py
```

```
def decrypt(ciphertext):  
    ciphertexts = ciphertext.split('-')  
    decrypted = ''  
    for text in ciphertexts:  
        countx = 0  
        for x in text:  
            countx += 1  
        if (countx!=0):  
            decrypted += chr((countx-9^0x50))  
    decrypted = decrypted[::-1]  
    return decrypted  
  
print ("Infosec Module Decryptor")  
ciphertext = input("Masukkan string yang ingin di-  
decrypt: ")  
print ("Result :", decrypt(ciphertext))
```

encrypt.py

```
(user@kali)-[~/Documents/praktikum5]  
$ ls  
caesar.py  encode.py  file.zip  message1.bin  message2.bin  
  
(user@kali)-[~/Documents/praktikum5]  
$ nano encrypt.py
```

~/Desktop/documentnama - Mousepad

File Edit Search View Document Help

1 Nama : Dani Adrian
2 NIM : 225150201111009
3



BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

[illegible]



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

```
(user@kali)-[~/Documents/praktikum5]
$ ls
caesar.py  encode.py  encrypt.py  file.zip  message1.bin  message2.bin

(user@kali)-[~/Documents/praktikum5]
$ nano decrypt.py
```

~/Desktop/documentnama - Mousepad

```
File Edit Search View Document Help
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

user@kali: ~/Documents/praktikum5

```
File Actions Edit View Help
GNU nano 7.2 decrypt.py *
def decrypt(ciphertext):
    ciphertexts = ciphertext.split('-')
    decrypted = ''
    for text in ciphertexts:
        countx = 0
        for x in text:
            countx += 1
            if (countx % 50):
                decrypted += chr((countx-9^0x50))
        decrypted = decrypted[:-1]
    return decrypted

print ("Infosec Module Decryptor")
ciphertext = input("Masukkan string yang ingin di-decrypt: ")
print ("Result :", decrypt(ciphertext))
```

File Name to Write: decrypt.py

^G Help	M-D DOS Format	M-A Append	M-B Backup File
^C Cancel	M-M Mac Format	M-P Prepend	^T Browse



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

The screenshot shows a terminal window on a Kali Linux system. The user is in the directory `~/Documents/praktikum5` and has run the command `python decrypt.py`. The script, titled "Infosec Module Decryptor", prompts the user to enter a string to decrypt. The user has entered a long string of 'X' characters. The script outputs "Result : got it!". In the background, a text editor window titled "documentnama - Mousepad" is open, showing the following text:

```
1 Nama : Dani Adrian
2 NIM : 225150201111009
3
```

3. Jelaskan source code yang telah anda buat pada nomor 2

Source `decrypt.py` di atas adalah implementasi dari fungsi decrypt dalam bahasa Python. Tujuan dari source code diatas adalah untuk menyediakan fungsi yang dapat mendekripsi teks yang telah dienkripsi oleh fungsi. Dengan fungsi ini, kita dapat mengembalikan teks plaintext dari teks ciphertext yang dihasilkan oleh `encrypt.py`.

4. Jelaskan perbedaan antara

- Asymmetric encryption dan symmetric encryption

1. Symmetric Encryption:

- Dalam symmetric encryption, kunci yang sama digunakan untuk enkripsi dan dekripsi data.



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

- Lebih cepat daripada enkripsi asimetris karena kunci yang digunakan lebih pendek.
- Kekurangannya adalah kunci harus dibagikan secara aman antara pengirim dan penerima.
- Contoh algoritma simetris termasuk AES (Advanced Encryption Standard) dan DES (Data Encryption Standard).

2. Asymmetric Encryption:

- Dalam asymmetric encryption, ada sepasang kunci: kunci publik dan kunci pribadi. Kunci publik digunakan untuk enkripsi dan dapat dibagikan secara publik, sementara kunci pribadi digunakan untuk dekripsi dan harus disimpan secara rahasia.
- Enkripsi asimetris lebih aman karena tidak memerlukan pertukaran kunci rahasia. Namun, prosesnya lebih lambat daripada enkripsi simetris.
- Contoh algoritma asimetris termasuk RSA (Rivest-Shamir-Adleman) dan ECC (Elliptic Curve Cryptography).

a. Stream cipher dan block cipher

1. Stream Cipher:

- Stream cipher mengenkripsi data bit demi bit atau byte demi byte.
- Stream cipher menggunakan aliran kunci acak yang dihasilkan secara bersamaan dengan proses enkripsi untuk mengacak data.
- Cocok untuk mengenkripsi data secara real-time seperti komunikasi VoIP atau video streaming.
- Contoh stream cipher termasuk RC4 (Rivest Cipher 4) dan ChaCha20.

2. Block Cipher:



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

- Block cipher mengenkripsi blok data yang lebih besar, biasanya dalam ukuran tetap, seperti 64 bit atau 128 bit.
- Data dipecah menjadi blok-blok yang sama ukurannya sebelum dienkripsi.
- Cocok untuk mengenkripsi data yang disimpan, seperti file atau pesan yang akan dikirim.
- Contoh block cipher termasuk AES (Advanced Encryption Standard) dan DES (Data Encryption Standard).

Kesimpulan

Hashing:

- Hashing diawali dengan memastikan md5sum terinstal dan melakukan hashing terhadap dua file dengan algoritma SHA-1, SHA-256, dan MD5.
- Pada output md5sum, nilai hash untuk message1.bin dan message2.bin sama, yaitu 008ee33a9d58b51cfef425b0959121c9. Hal ini dikarenakan algoritma md5sum memiliki kelemahan yang disebut hash collision.

Encoding:

- Kode Python encode.py mengonversi teks ke dalam format UTF-8, base64, dan hexadecimal.
- Fungsionalitas tambahan ditambahkan untuk mengembalikan teks dari base64 dan hexadecimal encode.

Enkripsi:

- Source code caesar.py melakukan enkripsi plaintext dengan menggunakan algoritma Caesar Cipher.

Dekripsi:

- decrypt.py ditambahkan untuk mendekripsi teks yang telah dienkripsi.

Evaluasi



**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA**

BAB : KRIPTOGRAFI
NAMA : DANI ADRIAN
NIM : 225150201111009
TANGGAL : 09/05/2024
ASISTEN : Bernas Cakra Sakti Harisna
Mohammad Seto Aji Pamungkas

- Rangkaian tugas praktikum kriptografi ini memberikan pemahaman yang baik tentang konsep dasar kriptografi, seperti hashing, enkripsi, dan encoding.
- Implementasi praktis dalam bahasa pemrograman Python membantu untuk memahami cara kerja algoritma dan teknik kriptografi.
- Evaluasi keseluruhan menunjukkan pemahaman yang kuat tentang konsep-konsep kriptografi yang diuji dalam praktikum BAB 5 Kriptografi.

