LAPORAN TUGAS BESAR
SISTEM PARALEL dan TERDISTRIBUSI
DISTRIBUTED PASSWORD BRUTE FORCER

Disusun untuk Memenuhi Tugas Besar Mata Kuliah Sistem Paralel dan Terdistribusi

Disusun Oleh:

Athalla Arli Abhinaya	1301213217
Thoriq Dwi Laksono	1301210556
Zaidan Rizq	1301213203

Kelompok 2

Kelas IF-45-03

PROGRAM STUDI S1 INFORMATIKA

FAKULTAS INFORMATIKA

UNIVERSITAS TELKOM

2023

1. PENDAHULUAN

1.1. Latar Belakang

Dalam dunia keamanan informasi dan siber, terdapat orang dengan peran untuk melakukan penetration test. Peran tersebut akan melakukan serangan simulasi resmi yang dilakukan pada sistem komputer untuk mengevaluasi keamanannya. Salah satu elemen kunci dalam menjaga keamanan data adalah melalui penggunaan kata sandi (password) yang kuat dan sulit ditebak. Password yang disimpan di database akan dienkripsi dengan suatu metode hashing yang dibuat agar tidak bisa diubah kembali. Untuk mendapatkan value asli dari password tersebut dapat dilakukan dengan berbagi cara salah satunya dengan metode Brute Force. Masalahnya metode ini bisa memakan waktu yang lama, maka dari itu untuk mempercepat proses Brute Force kita dapat membuat sistemnya menjadi sistem terdistribusi dan itu program yang akan kami buat.

1.2. Rumusan Masalah

Salah satu masalah dari penggunaan metode Brute Force untuk memecahkan suatu password yang kompleks itu dapat memakan waktu yang sangat lama. Lama waktu proses cracking password ini memiliki beberapa faktor diantaranya kompleksitas password pengguna, metode hashing yang digunakan, dan lain-lain. Maka dari itu terdapat beberapa solusi salah satunya dengan membuat Brute Forcer ini secara sistem terdistribusi.

1.3. Tujuan

Tujuan dari program yang kami buat ini untuk memberikan solusi dari salah satu masalah penggunaan metode Brute Force untuk cracking password yang dihash, yaitu memakan waktu yang lama. Dengan program ini proses Brute Force dapat dilakukan dengan menggunakan lebih dari 1 device komputer malalui sistem terdistribusi.

2. PEMBAHASAN

2.1. Pengertian

2.1.1. Brute Force

"Brute force" dalam konteks umum mengacu pada metode pemecahan masalah yang mengandalkan kekuatan dan usaha semata, daripada metode yang lebih halus, strategis, atau efisien. Metode ini melibatkan mencoba semua solusi yang mungkin sampai solusi yang tepat ditemukan, tanpa menggunakan jalan pintas atau teknik yang cerdik.

Dalam konteks penetration testing, "brute force" mengacu pada metode coba-coba yang digunakan untuk mendapatkan akses tidak sah ke sistem, jaringan, atau data. Pendekatan ini melibatkan secara sistematis mencoba semua kemungkinan kombinasi kredensial (seperti nama pengguna dan kata sandi) untuk membobol sistem.

2.1.2. Distributed System

Sistem terdistribusi adalah jaringan komputer independen yang tampak oleh penggunanya sebagai satu sistem yang koheren. Dalam sistem ini, komputer berkomunikasi dan mengoordinasikan tindakan mereka dengan menyampaikan pesan satu sama lain.Sistem ini didistribusikan di lokasi jaringan yang berbeda, tetapi mereka bekerja sama untuk mencapai tujuan yang sama.

2.1.3. Distributed System Password Brute Forcer

"Distributed Password Brute Forcer" mengacu pada perangkat lunak atau sistem khusus yang dirancang untuk memecahkan kata sandi terenkripsi atau hash dengan menggunakan metode brute force dalam lingkungan komputasi terdistribusi

2.2. Penjelasan Kode

Server

```
C: > Users > Athalla arli > Downloads > Code > server.py >
1   import json
2   import os
3   import sys
4
5   import pika
6   from art import text2art
```

- Kode diatas Mengimpor library yang diperlukan seperti json, os, dan sys untuk operasi dasar.
- Mengimpor library pika untuk komunikasi dengan server RabbitMQ menggunakan protokol AMQP.
- Mengimpor fungsi text2art dari library art untuk membuat teks seni ASCII.

```
10  serverIP = "192.168.1.8"
11  jobQueueName = "Job Queue"
12  resultQueueName = "Result Queue"
13  jobName = "Job "
14  password_dictionary =
    '10-million-password-list-top-100000.txt'
15  batch_size = 5
```

- Kode diatas Berfungsi untuk mendefinisikan parameter-parameter konfigurasi seperti alamat IP server, nama antrian pekerjaan, nama antrian hasil, dan sebagainya

```
20  jobOrder = 1
21  crackedPassword = []
```

- Mendefinisikan variabel konstan untuk melacak urutan pekerjaan dan menyimpan password yang berhasil dipecah.

```
25  def showMainMenu():
26
27      allowedInput = [0,1,2]
28
29      print("=========================================================================")
30      print(text2art("CryptBreaker"))
31      print(text2art("(Server)"))
32      print("=========================================================================")
33      print("MENU UTAMA")
34      print("1. Tambah hashed password yang ingin di crack.")
35      print("2. Tampilkan hashed password yang berhasil di crack.")
36      print("0. Keluar dari program.")
37
38      while True:
39          try:
40              menuPilihan = int(input("PILIH MENU: "))
41              if menuPilihan in allowedInput:
42                  break
43              else:
44                  print("Menu yang tersedia hanya 0, 1, 2. Pilih kembali!")
45          except:
46              print("Menu yang tersedia hanya 0, 1, dan 2. Pilih kembali!")
47
48      return menuPilihan
```

- Kode diatas berfungsi Menampilkan menu utama pada server dan mengembalikan pilihan menu yang dipilih oleh pengguna.

```
def tambahJob():

    global jobOrder

    allowedInput = [0,1]
    batchMessage = []

    print("=============================================================================")
    print(text2art("CryptBreaker"))
    print(text2art("(Server)"))
    print("=============================================================================")
    print("MENU TAMBAH HASHED PASSWORD")
    print("0. Kembali ke menu utama.")

    while True:
        try:
            manyJob = int(input("Masukkan berapa banyak hashed password yang ingin di crack: "))
            break
        except:
            print("Masukkan harus berupa angka!")

    if manyJob == 0:
        return None
    else:
        if jobOrder == 1:
            channel.queue_declare(queue = jobQueueName)

        with open(password_dictionary, 'r') as file:
            passwords = file.read().splitlines()

        for i in range(manyJob):
            namaJob = jobName + str(jobOrder)
            print(f"\n{jobName} {jobOrder}")
            encryptedPassword = input(f"Masukkan hashed password ke {jobOrder}: ")

            for j, password in enumerate(passwords):
                job_message = {
                        'nama_job' : f'{jobName}{jobOrder}',
                        'password_terenkripsi' : encryptedPassword,
                        'input_sekuen' : password
                    }
                batchMessage.append(job_message)

                if (j+1) % batch_size == 0 or (j+1) == len(passwords):
                    sendBatchMessage(channel, jobQueueName, batchMessage)
                    batchMessage = []

            jobOrder += 1

        print("\nPenambahan berhasil.")
        print("0. Kembali ke menu utama.")
        print("1. Keluar dari program.")

        while True:
            try:
                menuPilihan = int(input("PILIH MENU: "))
                if menuPilihan not in allowedInput:
                    print("Menu yang tersedia hanya 0 dan 1. Pilih kembali!")
                else:
                    return menuPilihan
            except:
                print("Menu yang tersedia hanya 0 dan 1. Pilih kembali!")
```

- Kode memungkinkan pengguna untuk menambahkan pekerjaan cracking password ke dalam antrian.
- Melibatkan pengguna untuk memasukkan hashed password dan menentukan jumlah pekerjaan yang ingin ditambahkan.

```
113  def sendBatchMessage(channel, queue_name, message):
114
115      batch_message = json.dumps(message)
116      channel.basic_publish(exchange = '', routing_key = queue_name, body = batch_message)
```

- Kode ini mengirim batch pesan berisi informasi pekerjaan dari server ke antrian pekerjaan pada client.

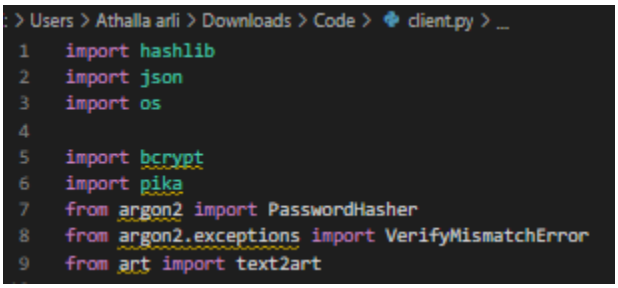
```
def showCrackedPassword():

    global crackedPassword

    allowedInput = [0,1]

    maxShowMessage = 100

    print("================================================================================")
    print(text2art("CryptBreaker"))
    print(text2art("(Server)"))
    print("================================================================================")
    print("MENU TAMPILKAN CRACKED PASSWORD\n")

    try:
        channel.queue_declare(queue=resultQueueName, passive=True)

        for i in range(maxShowMessage):
            method_frame, header_frame, body = channel.basic_get(queue=resultQueueName,
auto_ack=False)if method_frame:
                crackedPassword.append(body.decode())
                channel.basic_ack(method_frame.delivery_tag)
            else:
                break

        for j in crackedPassword:
            print(j)

    except pika.exceptions.ChannelClosedByBroker:
        print("BELUM ADA PASSWORD YANG BERHASIL DI CRACK!!!")
        print("JALANKAN ULANG PROGRAM!!!")
        sys.exit(1)

    print("\n0. Kembali ke menu utama.")
    print("1. Keluar dari program.")

    while True:
        try:
            menuPilihan = int(input("PILIH MENU: "))
            if menuPilihan not in allowedInput:
                print("Menu yang tersedia hanya 0 dan 1. Pilih kembali!")
            else:
                return menuPilihan
        except:
            print("Menu yang tersedia hanya 0 dan 1. Pilih kembali!")
```

- Kode ini menampilkan hasil password yang berhasil dipecah kepada pengguna.

```python
if __name__ == "__main__":

    connectionParameters = pika.ConnectionParameters(serverIP, heartbeat=0)
    connection = pika.BlockingConnection(connectionParameters)
    channel = connection.channel()

    try:
        while True:
            menuPilihan = showMainMenu()
            match menuPilihan:
                case 0:
                    break
                case 1:
                    os.system('cls')
                    menuPilihan = tambahJob()
                    if menuPilihan == 1:
                        break
                    else:
                        os.system('cls')
                case 2:
                    os.system('cls')
                    menuPilihan = showCrackedPassword()
                    if menuPilihan == 1:
                        break
                    else:
                        os.system('cls')
    finally:
        if channel.is_open or connection.is_open:
            try:
                channel.close()
                connection.close()
                print("Closing Channel.")
                print("Closing Connection.")
            except:
                print("Closing Channel.")
                print("Closing Connection.")

```

- Kode ini membuka koneksi ke server RabbitMQ dan menjalankan loop utama untuk menangani interaksi dengan pengguna. Dalam loop tersebut, program menampilkan menu utama, dan berdasarkan pilihan pengguna, dapat menambahkan pekerjaan cracking password atau menampilkan hasil password yang berhasil dipecah. Program terus berjalan hingga pengguna memilih untuk keluar. Setelah loop selesai, program menutup koneksi dan antrian pada server.

Client



- Import ashlib : Menyediakan antarmuka umum untuk berbagai algoritma hash dan message digest yang aman.
- Import json: Digunakan untuk penyandian dan penguraian kode JSON.
- Import os : Menyediakan cara untuk berinteraksi dengan sistem operasi, digunakan untuk membersihkan layar konsol (os.system('cls')).
- Import bcrypt: Pustaka untuk mengenkripsi kata sandi.
- Import pika : Pustaka Python murni untuk berinteraksi dengan RabbitMQ, sebuah perantara pesan.
- Import PasswordHasher dan VerifyMismatchError dari argon2 : Digunakan untuk menangani hashing kata sandi Argon2.
- Import text2art : Sebuah pustaka untuk menghasilkan seni ASCII dari teks.

```
13  serverIP = "192.168.1.8"
14  resultQueueName = "Result Queue"
15  jobQueueName = "Job Queue"
16  processedJobQueueName = "Processed Job Notificatoin Queue"
17  #---------------------------------
18
19  #Const
20  #---------------------------------
21  processed_jobs = set()
22  #---------------------------------A
```

- Kode diatas sebagai parameter konfigurasi untuk alamat IP server RabbitMQ dan nama antrean.
- melacak pekerjaan yang sedang diproses.

```
24  def showMainMenu():
25
26      allowedInput = [0,1]
27
28      print("================================================================================")
29      print(text2art("CryptBreaker"))
30      print(text2art("(Client)"))
31      print("================================================================================")
32      print("MENU UTAMA")
33      print("1. Mulai cracking password.")
34      print("0. Keluar dari program.")
35
36      while True:
37          try:
38              menuPilihan = int(input("PILIH MENU: "))
39              if menuPilihan in allowedInput:
40                  break
41              else:
42                  print("Menu yang tersedia hanya 0, 1, 2, dan 3. Pilih kembali!")
43          except:
44              print("Menu yang tersedia hanya 0, 1, 2, dan 3. Pilih kembali!")
45
46      return menuPilihan
```

- Kode ini berfungsi menampilkan menu utama program CryptBreaker dan meminta pengguna untuk memilih opsi. Ini menggunakan ASCII art untuk presentasi visual dan memastikan bahwa pengguna memilih opsi yang valid (0 atau 1) sebelum melanjutkan. Loop akan terus berlanjut hingga pengguna memilih opsi yang benar, dan fungsi mengembalikan pilihan pengguna.

```python
def startCracking():

    allowedInput = [0,1]

    print("================================================================================")
    print(text2art("CryptBreaker"))
    print(text2art("(Client)"))
    print("================================================================================")
    print('Menunggu pesan. Untuk berhenti dari proses cracking, tekan CTRL+C')
    print("PROGRAM RUNNING...\n")
    try:
        channel.queue_declare(queue = resultQueueName)
        channel.queue_declare(queue = processedJobQueueName)
        channel.basic_qos(prefetch_count=1)
        channel.basic_consume(queue = jobQueueName, auto_ack = False, on_message_callback = onMessageReceived)
        channel.basic_consume(queue = processedJobQueueName, auto_ack = True, on_message_callback = onNotificationReceived)
        consuming = True
        channel.start_consuming()

    except KeyboardInterrupt:
        if consuming:
            channel.stop_consuming()

        print("\nProses berhenti.")
        print("0. Kembali ke menu utama.")
        print("1. Keluar dari program.")

        while True:
            try:
                menuPilihan = int(input("PILIH MENU: "))
                if menuPilihan not in allowedInput:
                    print("Menu yang tersedia hanya 0 dan 1. Pilih kembali!")
                else:
                    return menuPilihan
            except:
                print("Menu yang tersedia hanya 0 dan 1. Pilih kembali!")
```

- Kode ini akan memulai proses cracking password dalam program CryptBreaker. Ini mengatur koneksi ke RabbitMQ untuk menerima dan memproses pesan terkait tugas cracking. Selama proses berjalan, pengguna dapat menghentikannya dengan menekan CTRL+C. Jika proses dihentikan, pengguna diberikan opsi untuk kembali ke menu utama atau keluar dari program, dengan validasi input untuk memastikan pilihan yang benar.

```python
def sendMessage(channel, queue_name, message):
    channel.basic_publish(exchange = '', routing_key = queue_name,
                          body = json.dumps(message))
    print(f"Sent {message} to {queue_name}")
```

- Kode ini digunakan untuk mengirim pesan dalam format JSON ke sebuah antrian RabbitMQ. Ini melibatkan penggunaan objek channel untuk berkomunikasi dengan RabbitMQ, dan pesan dikirim ke antrian yang ditentukan. Informasi pengiriman juga dicetak untuk pemantauan.

```python
def onMessageReceived(ch, method, properties, body):

    global processed_jobs

    batchMessage = json.loads(body)
    hashType = guessHashType(batchMessage[0]['password_terenkripsi'])

    for message in batchMessage:
        if json.dumps(message['nama_job']) in processed_jobs:
            break

        encryptedPassword = message['password_terenkripsi']
        inputSequence = message['input_sekuen']
        result = checkInputSequence(hashType,inputSequence, encryptedPassword)

        if result:
            print(f"{message['nama_job']}: Password cracked")
            resultMessage = {
                'nama_job' : message['nama_job'],
                'hash_type' : hashType,
                'input_sekuen' : inputSequence
            }
            sendMessage(ch, resultQueueName, resultMessage)
            sendMessage(ch, processedJobQueueName, message['nama_job'])
            print("")
            break

    channel.basic_ack(delivery_tag=method.delivery_tag)
```

- Kode ini merupakan callback yang dijalankan saat pesan diterima dari antrian pekerjaan RabbitMQ. Fungsi ini memproses batch pesan, mencoba memecahkan password, dan mengirim hasil serta notifikasi ke antrian yang sesuai. Variabel global processed_jobs digunakan untuk melacak pekerjaan yang sudah diproses. Acknowledge digunakan untuk memberitahu RabbitMQ bahwa pesan sudah diproses.

```python
def onNotificationReceived(ch, method, properties, body):
    global processed_jobs
    processed_job = body.decode('utf-8')
    processed_jobs.add(processed_job)
```

- Kode ini adalah callback yang menangani notifikasi yang diterima dari antrian notifikasi RabbitMQ. Fungsi ini mengonversi payload notifikasi menjadi string dan menambahkan nama pekerjaan yang sudah diproses ke dalam set processed_jobs.

```python
def checkInputSequence(hash_type, input_sequence, encrypted_password):
    if hash_type == "MD5":
        return hashlib.md5(input_sequence.encode()).hexdigest() == encrypted_password
    elif hash_type == "SHA1":
        return hashlib.sha1(input_sequence.encode()).hexdigest() == encrypted_password
    elif hash_type == "SHA256":
        return hashlib.sha256(input_sequence.encode()).hexdigest() == encrypted_password
    elif hash_type == "bcrypt":
        encoded_input_sequence = input_sequence.encode()
        encoded_encrypted_password = encrypted_password.encode()
        return bcrypt.checkpw(encoded_input_sequence, encoded_encrypted_password)
    elif hash_type == "Argon2":
        ph = PasswordHasher()
        try:
            ph.verify(encrypted_password, input_sequence)
            return True
        except VerifyMismatchError:
            return False
    else:
        return None
```

- Fungsi checkInputSequence memeriksa apakah urutan input menghasilkan hash password yang cocok dengan hash yang terenkripsi, berdasarkan tipe hash yang diberikan. Ini mencakup penggunaan metode hash seperti MD5, SHA1, SHA256, bcrypt, dan Argon2. Fungsi mengembalikan True jika cocok, False jika tidak cocok, dan None jika tipe hash tidak dikenali.

```python
def guessHashType(hash_string):
    length = len(hash_string)
    hash_string_lower = hash_string.lower()

    if length == 32:  # MD5 hash length
        return 'MD5'
    elif length == 40:  # SHA1 hash length
        return 'SHA1'
    elif length == 64:  # SHA256 hash length
        return 'SHA256'
    elif hash_string.startswith(("$2a$", "$2b$", "$2y$")):
        return 'bcrypt'
    elif hash_string.startswith(("$argon2i$", "$argon2d$", "$argon2id$")):
        return 'Argon2'
    else:
        return 'Unknown'
```

- Kode ini digunakan untuk menebak tipe hash berdasarkan panjang hash atau format khusus yang ditemukan dalam string hash. Fungsi ini mengembalikan 'MD5', 'SHA1', 'SHA256', 'bcrypt', atau 'Argon2' berdasarkan karakteristik hash yang dianalisis. Jika tidak ada kecocokan, fungsi mengembalikan 'Unknown'.

```python
if __name__ == "__main__":

    connectionParameters = pika.ConnectionParameters(serverIP, heartbeat=0)
    connection = pika.BlockingConnection(connectionParameters)
    channel = connection.channel()

    try:
        while True:
            menuPilihan = showMainMenu()
            match menuPilihan:
                case 0:
                    break
                case 1:
                    os.system('cls')
                    menuPilihan = startCracking()
                    if menuPilihan == 1:
                        break
                    else:
                        os.system('cls')
    finally:
        if channel.is_open or connection.is_open:
            try:
                channel.close()
                connection.close()
                print("Closing Channel.")
                print("Closing Connection.")
            except:
                print("Closing Channel.")
                print("Closing Connection.")
```

- Kode ini merupakan bagian utama dari program CryptBreaker. Itu menetapkan koneksi ke RabbitMQ, memulai loop utama untuk menangani menu, dan membersihkan koneksi RabbitMQ saat program berakhir. Loop utama memungkinkan pengguna untuk memilih opsi dari menu, seperti memulai proses cracking password atau keluar dari program. Saat program berakhir, koneksi dan saluran RabbitMQ ditutup.

2.3. Cara Menjalankan

2.3.1. Install RabbitMQ sebagai message broker.
https://www.rabbitmq.com/download.html

2.3.2. Install Python untuk menjalankan program
https://www.python.org/downloads/

2.3.3. Install package library pika, art, bcrypt, argon2

2.3.4. Jalankan RabbitMQ service

2.3.5. Buka console RabbitMQ melalui browser
http://SERVER_IP_ADDRESS:15672

2.3.6. Login sebagai guest dengan username dan password guest

2.3.7. Jalankan program server.py

2.3.8. Ikuti petunjuk menu program

2.3.9. Jalankan program client.py

2.3.10. Ikuti petunjuk menu program

3. KESIMPULAN

3.1. Kesimpulan

Kode program Distributed Password Brute Forcer yang kami buat terdiri dari dua bagian yaitu client dan server. Kode ini dirancang untuk melakukan cracking password dengan metode Distributed Password Brute Forcer. Arsitektur Sistem memanfaatkan message broker RabbitMQ. Proses cracking password dilaksanakan melalui pengiriman batch pesan yang mengandung informasi pekerjaan (job) dari server ke client. Dengan pendekatan sistem terdistribusi, password dipecah menggunakan sumber daya dari beberapa mesin atau node, meningkatkan secara signifikan kecepatan dan efisiensi dalam proses tersebut.

Link Video: https://youtu.be/sFL4-91txm0