ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN MÔN HỌC
(Đồ án phát triển ứng dụng)
Lớp: IT003.O21.CTTN

SINH VIÊN THỰC HIỆN

Mã sinh viên: 23521045 **Họ và tên: Nguyễn Hữu Đặng Nguyên**

TÊN ĐỀ TÀI: ỨNG DỤNG MÔ PHỎNG CÁC MÔ HÌNH PASSWORD CRACKING.

Mục lục

I. Giới thiệu đồ án:

1. Mô tả chung về ứng dụng:

Đây là báo cáo trình bày về những nghiên cứu về Password Cracking (bẻ khoá mật khẩu) và những tác động của nó. Đồng thời bài báo cáo cũng cung cấp một tool tổng hợp nhiều phương thức giúp bẻ khoá mật khẩu của một file Zip và đánh giá khách quan các phương thức đó. Về phần Visualization, em có cung cấp một repo github có chứa các file làm việc.

2. Các CTDL và giải thuật đã được sử dụng:

Thuật toán sinh ra các tổ hợp từ một bộ ký tự cho trước.

Giải thuật đệ quy để sinh ra các chuỗi phù hợp với mong muốn

Cấu trúc dữ liệu dictionary của Python

Sử dụng các kiến thức về hướng đối tượng trong Python

Một vài kiến thức tổ hợp để áp dụng các công thức toán, …

II. Quá trình thực hiện

1. **Tuần 1:**

   Hoàn thành phần ý tưởng và một số phần của bài báo cáo bao gồm:

   - Lý thuyết Password Cracking (Khái niệm, phương pháp, tác động và ví dụ)
   - Một vài mô hình Password Cracking phổ biến
   - Giải thích chi tiết kèm phần Code của bruteforce attack

2. **Tuần 2:**

   - Hoàn thiện phần giải thuật và code cho các mô hình còn lại.
   - Thử nghiệm sản phẩm.
   - Viết bài báo cáo.

III. Kết quả đạt được

1. Lý thuyết về Password Cracking.

a. Khái niệm

- Password Cracking hay bẻ khóa mật khẩu là quá trình sử dụng nhiều cách thức có thể thủ công hoặc thông qua chương trình máy tính để xác định được mật khẩu của máy tính hay tài nguyên trên mạng.

b. Phương pháp

Có nhiều cách để bẻ khóa mật khẩu, một vài cách tiếp cận phổ biến như:

- Đánh cắp trực tiếp mật khẩu từ người dùng.
- Liên tục đoán mật khẩu hay sử dụng một danh sách các mật khẩu phổ biến và kiểm tra với giá trị Hash có sẵn của tài khoản.
- Sử dụng một liên kết hay một tệp có gán các chương trình chi tiết nhằm giả mạo máy tính người dùng hay theo dõi hoạt đồng người dùng từ màn hình, bàn phím từ đó thu thập thông tin.

Tấn công trực tiếp vào cơ sở dữ liệu của máy chủ và lấy thông tin.

c. Hậu quả của việc bị crack password

Đa phần người sử dụng Password Cracking vì mục đích xấu. Khi mật khẩu bị các Hacker phát hiện ra, nó có thể gây ra nhiều hậu quả khó lường như:

- Bị lộ thông tin cá nhân, thông tin riêng tư.
- Bị đánh cắp tài khoản ngân hàng.

- Tài khoản bị sử dụng cho các mục đích xấu.

d. Một vài vụ bẻ khóa mật khẩu nổi tiếng:

- Gần đây, Streamer Độ Mixi đã bị bẻ khóa mật khẩu nhiều tài khoản mạng xã hội. Đó là do anh đã nhấn vào một file có chứa mã độc từ đó bị đánh cắp thông tin.
- Năm 2012, một hacker mật danh "Peace" đã thực hiện một vụ tấn công mạng vào hơn 100 triệu tài khoản LinkedID. Hắn sử dụng một thuật toán nhằm bẻ khóa mã Hash SHA1 của đa số các tài khoản trên nền tảng này.

2. Một vài mô hình Password Cracking phổ biến

a. Brute force Attack

Trong mô hình này, hacker sẽ cố gắng đoán mật khẩu thông qua việc thử một cách có hệ thống mọi loại kết hợp ký tự cho đến khi tìm được mật khẩu hợp lệ.

- Ưu điểm:

- Trên lý thuyết, phương pháp này có thể bẻ khóa tất cả mật khẩu và chống lại tất cả loại mã hoá.
- Rất hiệu quả khi chống lại các mật khẩu ngắn và đơn giản

-Nhược điểm

- Tốn rất nhiều thời gian và tài nguyên. Dễ bị phát hiện
- Không hiệu quả với mật khẩu đủ dài và phức tạp.

b.Dictionary Attack

Trong mô hình này, hacker sử dụng một danh sách các password phổ biến là từ điển (dictionary) để thử mật khẩu.

-Ưu điểm:

- Có thể bẻ khóa nhanh được các mật khẩu đơn giản
- Sử dụng danh sách có sẵn, không mất công sinh ra mật khẩu.

-Nhược điểm

- Chỉ hiệu quả với các mật khẩu phổ biến.
- Nếu mật khẩu được đặt ngẫu nhiên thì phương pháp này coi như vô dụng

c.Hybrid Attack

Phương pháp này kết hợp hai phương pháp Brute force attack và Dictionary Attack. Trong mô hình này, Hacker sẽ vẫn sử dụng các mật khẩu có thể từ Dictionary có sẵn. Sau đó hacker sẽ dùng chương trình để thay thế, thêm bớt một vài ký tự bằng các ký tự có liên quan để sinh ra mật khẩu khả thi mới.

d. Rainbows Attack

Hiện nay, nhiều máy chủ, hệ thống sử dụng thuật toán Hash(md5, sha) để mã hóa thông tin. Khi người dùng đăng ký tài khoản, server sẽ sử dụng các thuật toán này để mã hóa mật khẩu lại thành một mã Hash. Sau đó khi đăng nhập thì server sẽ tiến hành so sánh với mã Hash thay vì so với mật khẩu thuần. Mô hình Rainbows Attack sử dụng một bảng thống kê gọi là Rainbows Table lưu trữ các giá trị Hash đã được tính toán. Phương pháp này sẽ giống việc Hacker sẽ gian lận bằng việc không nhất thiết đoán chính xác mật khẩu mà là đoán mã Hash của mật khẩu đó. Phương pháp này có thể kết hợp Dictionary Attack.

Ưu điểm:

- Có thể nhanh chóng bẻ khóa được mật khẩu được mã hóa yếu
- Tiền xử lý giúp việc tìm được mã Hash diễn ra nhanh hơn.

Nhược điểm:

- Yêu cầu quy trình xử lý phức tạp và lượng tài nguyên lớn
- Không hiệu quả với mật khẩu được mã hóa mạnh.

e.Phising and Vising

Đây được đánh giá là phương pháp dễ dàng, phổ biến và hiệu quả nhất để lấy thông tin về tài khoản. Trong phương pháp này, hacker gửi một trang đăng nhập giả mạo tới nạn nhân. Hacker sẽ gửi trang đăng nhập giả thông qua email, chat ... Khi nạn nhân đăng nhập với username/password, họ sẽ bị hacker phát hiện.

f. Rat and Keylogging

Trong phương pháp này, hacker sẽ gửi một keylogger server hoặc RAT server tới nạn nhân. Keylogger sẽ ghi lại mọi thao tác trên bàn phím của nạn nhân và bản ghi này sẽ được gửi tới hacker.

3. Chi tiết và giải thích một số mô hình Password Cracking

a. Các thư viện và ngôn ngữ sử dụng

Ngôn ngữ sử dụng: Python

Các thư viện sử dụng:

- Argparse: Hỗ trợ quá trình tạo tool và làm việc với CLI
- Itertools: Dùng hỗ trợ cho việc tạo các tổ hợp nhanh chóng
- Zipfile: Để tương tác và làm việc với file zip
- Math

b. Cài đặt môi trường và class

```
dictionary_file = 'dictionary.txt' # Bộ password cho dictionary attack
hybrid_file = 'hybrid.txt' # Bộ dữ liệu thô cho hybrid attack
tested_file = open('tested_password.txt', 'w') # File chứa các password đã thử nghiệm và kết quả đưa ra
```

Các file làm việc gồm có:

- Password_cracking_tool.py: Chứa phần code cho sản phẩm
- Dictionary.txt, hybrid.txt: Chứa phần dữ liệu phục vụ cho dictionary attack và hybrid atack
- Tested_password.txt: Chứa phần kết quả đầu ra

```
❯ python3 password_cracking_tool.py -h
usage: password_cracking_tool.py [-h] [--algo [{bfa,da,hb}]] [-min MIN_LENGTH]
                                 [-max MAX_LENGTH] [-cset CHARACTER_SET]
                                 [--zipfile ZIPFILE]

Password cracking tool of Dang Nguyen - TWINHTER

options:
  -h, --help            show this help message and exit
  --algo [{bfa,da,hb}]  brute force attack, dictionary attack or hybrid attack
  -min MIN_LENGTH, --min-length MIN_LENGTH
                        Minimum password length for brute force attack
  -max MAX_LENGTH, --max-length MAX_LENGTH
                        Maximum password length for brute force attack
  -cset CHARACTER_SET, --character-set CHARACTER_SET
                        Character set for brute force attack
  --zipfile ZIPFILE     Zip file to crack
```

Các module của tool cracking password.

Tổng quan về class Password Cracker

```python
# Cracking với bộ password đã được sinh.
def cracking(self, myPassword):
    # Lấy từng xâu trong list để test
    for password in myPassword:
        self.total_passwords += 1
        tested_file.write(password + '\n')
        # Tiến hành test password
        try:
            with ZipFile(self.zip_file) as zipObj:
                zipObj.extractall(pwd=password.encode())
                self.matched_password = password
                break
        except:
            pass
```

Hàm cracking dùng để crack zip file với một list password đã được sinh ra dựa trên các thuật toán. Hàm sử dụng những function có sẵn được định nghĩa trong thư viện Zipfile để hỗ trợ so khớp mật khẩu.

```
# Đưa ra kết luận cuối cùng.
def print_statistics(self):
    tested_file.write(f"Total Number of Passwords Tried: {self.total_passwords}\n")
    if self.matched_password:
        tested_file.write(f"Password Cracked! Password: {self.matched_password}\n")
    else:
        tested_file.write("Password Failed.\n")
```

Hàm dùng để in ra kết quả

c. Brute force Attack

```
# Sinh password cho bruteforce attack.
def generate_passwords(self, min_length, max_length, character_set):
    # List để lưu những password khả thi
    passwords = []
    # Hàm để thực hiện sinh các tổ hợp thoả mãn min_length < password_length < max_length với bộ ký tự thuộc character_set
    for length in range(min_length, max_length + 1):
        for combination in itertools.product(character_set, repeat=length):
            password = ''.join(combination)
            passwords.append(password)
    return passwords
```

```
# Thuật toán Bruteforce Attack.
def crack_passwords_with_brute_force(self, min_length, max_length, character_set):
    tested_file.write('Cracking with brute force attack \n')

    # Tính toán trước số lượng password sẽ phải sinh
    maximumGeneratePassword = 0
    for i in range(min_length, max_length+1):
        maximumGeneratePassword += int(math.pow(len(character_set), i))
    # Dừng thuật toán khi thấy số lượng tìm kiếm quá lớn
    if maximumGeneratePassword > self.limit and self.limit != -1:
        tested_file.write('Cannot crack because take too much time\n')
        tested_file.write(f'Expected password have to test: {maximumGeneratePassword}\n')
        exit()

    # Sinh ra một list bao gôm các password thoả mãn yêu cầu được truyền vào
    passwords = self.generate_passwords(min_length, max_length, character_set)
    self.cracking(passwords)
```

-Với các tham số về min length, max length, character set được đưa vào, chương trình sẽ tính trước số tổ hợp ký tự có thể được tạo ra.

maximumGeneratePassword = $\sum_{i=min}^{max} len(character\ set)^i$

-Nếu con số này lớn hơn giới hạn cho trước thì không kiểm tra mà báo ra là không crack vì số lượng password thử là quá lớn. Nếu không thì thực thi hàm generate_passwords và tiến hành cracking với list password được trả về.

- Nó sẽ tạo ra các password với độ dài lần lượt từ min length đến max length.

- Mỗi độ dài sẽ sử dụng bộ dữ liệu character set cùng hàm product của itertools (repeat = độ dài đang xét) để tạo ra các tổ hợp password thoả mãn.
- Lưu các password vào một list sau đó return lại.

-Thời gian dự kiến O(maximumGeneratePassword)

d. Dictionary Attack

```
# Thuật toán cho dictionary attack.
def crack_passwords_with_dictionary(self):
    tested_file.write('Cracking with dictionary attack \n')
    # Mở file dictionary để làm việc
    with open(dictionary_file, 'r', encoding="latin-1") as dictionary:
        # Đưa các password từ file dictionary.txt vào list passwords và xử lý
        passwords = dictionary.read().splitlines()
        self.cracking(passwords)
```

- Function trong class: crack_passwords_with_dictionary.
- Sử dụng file chứa các password phổ biến (dictionary.txt). (> 300000 mật khẩu)
- Đọc các password trong file và lưu vào một list passwords để tiến hành cracking.
- **Thời gian dự kiến: O(len(dictionary.txt))**

e. Hybrid Attack

```
# Thuật toán cho hybrid attack.
def crack_passwords_with_hybrid_attack(self):
    tested_file.write('Cracking with hybrid attack \n')

    # List để chứa các tổ hợp của nums .
    nums = []
    for num_length in range(2):
        Lim = int(math.pow(10, num_length+1))
        for i in range(Lim):
            s = str(i).zfill(num_length)
            nums.append(s)

    nums3 = ['', '000', '123', '321', '111', '222', '333', '444', '555', '666', '777', '888', '999', '987', '789', '1234', '123456', 'abc', 'xyz
    nums.extend(nums3)

    # List chứa các password khả thi.
    myValidPassword = []
    with open(hybrid_file, 'r', encoding="latin-1") as dictionary:
        passwords = dictionary.read().splitlines()
        for password in passwords:
            myList = list(map(str, password))
            # List chứa các dạng lai của pass đang xét.
            extendPassword = self.generate_hybrid_passwords(myList)

            # Tiến hành tạo các password dạng num + pass hay pass + num.
            for newPass in extendPassword:
                for num in nums:
                    myValidPassword.append(newPass + num)
                    myValidPassword.append(num + newPass)

    self.cracking(myValidPassword)
```

```python
# Sinh password cho hybrid attack.
def generate_hybrid_passwords(self, myList, cur = 0):
    # List trả về
    res = []
    # Điều kiện dừng của đệ quy
    if cur == len(myList):
        myPassword = ''.join(myList)
        # print(myList)
        res.append(myPassword)
        return res

    # Không thay đổi ký tự hiện tại
    res.extend(self.generate_hybrid_passwords(myList, cur+1))

    # Nếu ký tự hiện tại thuộc self.dict thì tiến hành thay đổi
    if myList[cur] in self.dict:
        tmp = myList[cur]
        myList[cur] = self.dict[tmp]
        res.extend(self.generate_hybrid_passwords(myList, cur+1))
        myList[cur] = tmp

        # Trường hợp đặc biệt với ký tự 'a'
        if myList[cur] == 'a':
            myList[cur] = '@'
            res.extend(self.generate_hybrid_passwords(myList, cur+1))
            myList[cur] = 'a'

    return res
```

- Sử dụng file chứa các password phổ biến làm không gian chuẩn (hybrid.txt) (> 100 mật khẩu).
- Với mỗi password từ trong file trên, ta tiến hành:

- Tạo ra một list mới từ password đang xét chứa các password tương tự với nó. Ví dụ admin ⇔ @dmin ⇔ 4dmin.
- Dùng một list num chứa các phần có thể mở rộng của password. Nó là những phần ở phần đầu hoặc cuối hay được thêm vào(VD : ‘123’, ‘111’, ‘abc’, …)
- Tạo ra các tổ hợp password + num và num + password để thử mật khẩu

  Thời gian dự kiến O(len(hybrid.txt) * len(num) * A) với A là hằng số với mỗi password cụ thể.

IV. Tài liệu tham khảo

StackOverFlow

WhiteHat.vn

Wikipedia

https://github.com/topics/password-wordlist

https://stytch.com/blog/top-10-password-cracking-techniques/

https://github.com/HalilDeniz/PassBreaker/tree/main

V. Phụ lục 1: Giới thiệu (demo) kết quả

1. Link github sản phẩm: https://github.com/TwinHter/23521045_DoAnCK

2. Demo sản phẩm

Sử dụng câu lệnh trong terminal để thực hiện sản phẩm

Câu lệnh: python3 password_cracking_tool.py <content>

Với content bao gồm:

--algo	'bfa', 'hb', 'da'	Brute force attack, hybrid attack, dicitonary attack (default)
--zipfile	Text	Đường dẫn đến file zip cần crack
--min-length, -min	Int	Default = 1, độ dài tối thiểu của mật khẩu (dùng cho brute force attack)
--max-length, -max	Int	Default = 6, độ dài tối đa của mật khẩu (dùng cho brute force attack)
--character-set, -cset	Text	Default = 'abcdefghijklmnopqrstuvwxyz0123456789', chứa bộ ký tự của mật khẩu

Demo sản phẩm: https://youtu.be/QzgecMvyD44 (Kênh Youtube TwinHter - Demo Password Cracking Ver2 – 23521045 UIT)

3. Nhận xét

- Mô hình chạy tốt với mật khẩu ngắn và bộ ký tự đơn giản.
- Dictionary và Hybrid attack có thể bao quát được hơn 600000 mật khẩu phổ biến.
- Với mật khẩu mang tính ngẫu nhiên và dài dòng, mô hình gần như sẽ không đưa ra được kết quả.