Sistem Recovery Password dengan Menggunakan Secret Sharing Scheme

Mohammad Rafi Adyatma - 13518121
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13518121@std.stei.itb.ac.id

***Abstract*—Penggunaan sandi dalam kebutuhan otentikasi sudah sangat meluas dan mudah ditemukan sehari-hari. Sandi bersifat rahasia dan hanya pemilik sandi yang diperbolehkan untuk mengetahui. Manusia, sebagai salah satu makhluk hidup yang dapat lupa terhadap sesuatu menyebabkan sandi harus memiliki sistem pemulihan. Sandi harus dapat dipulihkan tanpa harus mengetahui sandi lama sehingga diperlukan sistem pemulihan sandi yang baik. Sudah ada beberapa sistem pemulihan sandi yang telah digunakan, namun dicoba pendekatan lain sebagai variasi sistem yang telah ada. Makalah ini mencoba membangun sistem pemulihan sandi menggunakan skema pembagian rahasia Shamir.**

***Keywords*—pemulihan sandi, pembagian rahasia Shamir, desentralisasi**

I. PENDAHULUAN

Sandi atau *password* adalah hal penting dalam dunia siber. Penggunaan sandi selalu dibutuhkan untuk kebutuhan otentikasi pengguna. Kebutuhan tersebut dibutuhkan untuk menentukan apakah seorang pengguna dapat melakukan sesuatu atau tidak, apa saja yang dapat dilakukan pengguna, dan siapa pengguna saat ini. Oleh karena itu, sandi harus bersifat rahasia dan memiliki keamanan yang tinggi, yaitu dari segi bagaimana sandi disimpan pada sistem dan bagaimana sandi dibentuk oleh pengguna. Dari sisi sistem, sandi harus disimpan sedemikian sehingga sandi-sandi pengguna aman dari kebocoran. Sedangkan dari sisi pengguna, sandi harus dibuat dengan kompleksitas yang tidak rendah dan tidak mudah untuk ditebak. Namun, karena keterbatasan yang dimiliki oleh manusia, sandi dapat terlupakan dan menyebabkan seorang pengguna tidak dapat memverifikasi dirinya sendiri sebagai pengguna yang sah. Hal tersebut harus ditangani, salah satu solusinya adalah membangun sistem pemulihan sandi. Pada tahun 2019, terdapat riset terhadap 500 responden bahwa terdapata 78 persen yang pernah lupa dengan sandi mereka dan menggunakan sistem pemulihan sandi untuk mendapatkan akses akunnya kembali dalam 90 hari terakhir. Sebenarnya, sudah ada beberapa metode yang digunakan dalam sistem pemulihan sandi seperti mengirim tautan *reset* sandi melalui email, menggunakan kode rahasia yang sudah dibentuk sejak akun dibuat, verifikasi ulang menggunakan SMS, dan metode-metode lainnya. Pada makalah ini, akan dicoba pendekatan baru yaitu menggunakan skema pembagian rahasia Shamir dalam sistem pemulihan sandi. Pendekatan ini terlihat lebih rumit, namun dapat menutupi kekurangan metode lain seperti tautan *reset* sandi yang dikirim lewat email. Metode ini sangat berguna ketika pengguna juga tidak memiliki akses ke email yang terhubung dengan akun tersebut atau merasa email tersebut tidak aman. Kasus seperti itu beberapa kali terjadi, sehingga pendekatan yang diusulkan pada makalah ini pantas untuk dicoba.

II. LANDASAN TEORI

A. Kriptografi

Kriptografi adalah ilmu yang mempelajari teknik-teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasiaan, integritas data, serta otentikasi (Menez, 1996). Kriptografi merupakan kakas yang sangat penting di dalam keamanan informasi. Kriptografi sendiri berasal dari bahasa Yunani, yaitu *cryptós* (rahasia) dan *gráphein* (tulisan) yang diartikan sebagai tulisan rahasia. Layanan yang disediakan oleh kriptografi diantaranya adalah :

1. Kerahasiaan pesan (*Confidentially* / *Privacy*)
2. Keaslian pesan (*Data Integrity*)
3. Keaslian pengirim dan penerima pesan (*Authentication*)
4. Anti penyangkalan (*Non-repudiation*)

Pada konteks kriptografi, terdapat empat terminologi yang digunakan :

1. Pesan
   Informasi yang dapat dibaca dan dimengerti maknanya (baik dipersepsi secara visual maupun audial). Pesan dapat berupa plainteks, plain-image, plain-video, dan bentuk lain.

2. Enkripsi
   Proses menyandikan plainteks atau pesan menjadi cipherteks. Pada proses ini dibutuhkan fungsi enkripsi dan kunci enkripsi. Nama lain : *enciphering*

3. Dekripsi
   Proses mengembalikan cipherteks menjadi plainteks atau pesan semula. Pada proses ini dibutuhkan fungsi dekripsi dan kunci dekripsi. Fungsi dan kunci dekripsi bisa saja berbeda dengan fungsi dan kunci

pada enkripsi. Nama lain : *deciphering*

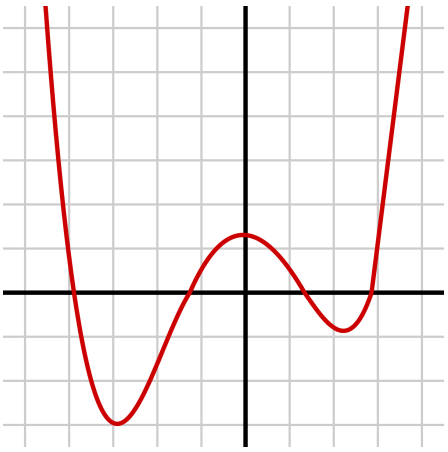
4. Cipherteks
   Cipherteks adalah pesan atau plainteks yang telah dilakukan enkripsi sehingga tidak bermakna lagi. Tujuan dibentuk cipherteks adalah agar pesan tidak dapat dibaca oleh pihak yang tidak berhak. Pesan semula dapat dipulihkan dengan melakukan dekripsi pada cipherteks dengan menggunakan kunci yang sesuai.

B. *Polynomial*

Polinomial adalah ekspresi matematika yang terdiri dari variabel, koefisien, dan operasi penambahan, pengurangan, perkalian, dan eksponen bilangan bulat non-negatif. Fungsi polinomial adalah sebuah fungsi yang dapat didefinisikan dengan mengevaluasi sebuah polinomial. Persisnya, sebuah fungsi f terdiri atas satu argumen dari domain yang diberikan adalah sebuah fungsi polinomial jika terdapat sebuah polinomial

$$a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_nx^n$$

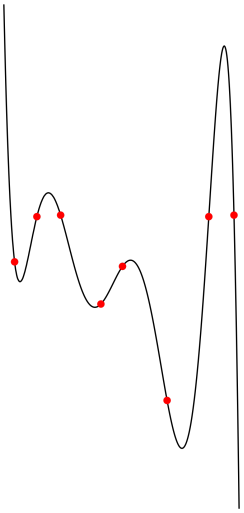
yang dapat dievaluasi menjadi f(x) untuk semua nilai x dengan domain *f*. Pada polinomial, a_1, a_2, ..., a_n disebut juga dengan koefisien (dengan kondisi tidak boleh semuanya bernilai 0), sedangkan a_0 disebut dengan konstanta. Polinomial disebut memiliki derajat m jika m merupakan derajat tertinggi yang dimiliki oleh polinomial dengan koefisien a_m tidak bernilai 0. Berikut adalah contoh grafik polinomial berderajat 4.



Gambar 1. Grafik dari fungsi polinomial berderajat 4
Sumber : https://en.wikipedia.org/wiki/Quartic_function

C. *Interpolasi*

Dalam analisis numerik, interpolasi polinomial adalah interpolasi yang dilakukan dari kumpulan data yang diberikan oleh polinomial dengan derajat serendah mungkin yang melewati titik-titik kumpulan data (Tiemann, Jerome J., 1981). Koefisien dan konstanta dari suatu polinomial dihitung menggunakan data yang sudah tersedia. Dengan memasukkan data ke dalam polinomial, dihasilkan sistem persamaan linear. Sistem persamaan linear dapat diselesaikan sehingga koefisien dan konstanta polinomial dapat diperkirakan (namun tidak sepenuhnya tepat). Polinomial hasil interpolasi hanya berupa gambaran kasar dari polinomial asli. Untuk mendapatkan polinomial asli dari interpolasi, jumlah data yang diperlukan berbeda-beda sesuai dengan derajat polinomial. Jika suatu polinomial memiliki derajat N, maka dibutuhkan N+1 data berbeda agar mendapatkan polinomial asli.



Gambar 2. Delapan Titik dan Hasil Interpolasi Polinomial
Sumber :
https://upload.wikimedia.org/wikipedia/commons/thumb/4/4c/Polynomial-interpolation.svg/1200px-Polynomial-interpolation.svg.png

D. *Interpolasi Lagrange*

Interpolasi Lagrange adalah salah satu metode yang dapat digunakan untuk melakukan interpolasi polinomial. Sama dengan metode interpolasi lainnya, interpolasi lagrange mengambil nilai-nilai tertentu pada titik-titik arbitrer. Teorema interpolasi Lagrange sebagai berikut :

Untuk *n* bilangan riil berbeda x_1, x_2, x_3, ..., x_n dan *n* bilangan riil y_1, y_2, y_3, ..., y_n (tidak harus berbeda), terdapat sebuah polinomial unik P berderajat kurang dari *n* yang memiliki koefisien-koefisien riil yang memenuhi $P(x_i) = y_i$.

Polinomial interpolasi Lagrange merupakan kombinasi linear

$$L(x) := \sum_{j=0}^{k} y_j \ell_j(x)$$

dari basis polinomial Lagrange

$$\ell_j(x) := \prod_{\substack{0 \le m \le k \\ m \ne j}} \frac{x - x_m}{x_j - x_m} = \frac{(x - x_0)}{(x_j - x_0)} \cdots \frac{(x - x_{j-1})}{(x_j - x_{j-1})} \frac{(x - x_{j+1})}{(x_j - x_{j+1})} \cdots \frac{(x - x_k)}{(x_j - x_k)},$$

dengan kondisi 0 <= j <= k.

E. Skema Pembagian Rahasia Shamir

Skema pembagian rahasia shamir adalah algoritma dalam kriptografi yang diciptakan oleh Adi Shamir pada tahun 1979. Algoritma ini termasuk ke dalam kategori pembagian rahasia dan bahkan merupakan salah satu algoritma pembagian rahasia yang pertama kali muncul. Secara kasar, algoritma pembagian rahasia ini bekerja menggunakan skema ambang (*threshold schemes*). Misalkan *t, w* adalah bilangan bulat positif dengan t <= w. Skema ambang (t, w) adalah metode pembagian pesan M kepada *w* partisipan sedemikian sehingga sembarang himpunan bagian yang terdiri dari *t* partisipan dapat merekonstruksi M, tetapi jika kurang dari *t* maka M tidak dapat direkonstruksi. Ide dari pembagian rahasia Shamir terletak pada persoalan interpolasi polinomial. Untuk membentuk persamaan linear :

- $y = a_0 + a_1x$, diperlukan 2 buah titik (x_1, y_1) dan (x_2, y_2)
- $y = a_0 + a_1x + a_2x^2$, diperlukan 3 buah titik (x_1, y_1), (x_2, y_2), dan (x_3, y_3)

Sehingga untuk membentuk polinomial $y = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ dibutuhkan n + 1 titik. Dengan fakta ini, kita dapat menyimpan rahasia (M) sebagai a_0 dengan n pembagian rahasia (x_1, y_1), (x_2, y_2), … , (x_n, y_n). Berikut adalah algoritma pembagian rahasia Shamir :

1. Pilih bilangan prima *p*, yang harus lebih besar dari semua kemungkinan nilai pesan *M* dan juga lebih besardari jumlah *w* partisipan. Semua komputasi dihasilkan dalam modulus *p*.

2. Pilih *t - 1* buah bilang bulat acak dalam modulus *p*, misalkan $s_1, s_2, \ldots, s_{t-1}$ dan nyatakan polinomial $s(x) \equiv M + s_1x + s_2x^2 + ... + s_{t-1}x^{t-1} \pmod p$ sedemikian sehingga $s(0) \equiv M \pmod p$

3. Untuk *w* partisipan, kita pilih bilangan bulat berbeda $x_1, x_2, \ldots, x_w \pmod p$ dan setiap orang memperoleh share (x_i, y_i) yang dalam hal ini $y_i \equiv s(x_i) \pmod p$. Untuk mempermudah, dipilih $x_1 = 1, x_2 = 2, \ldots, x_w = w$.

Untuk melakukan rekonstruksi rahasia dengan *share-share* yang ada, digunakan interpolasi Lagrange untuk mendapatkan polinomial.

III. SISTEM PEMULIHAN SANDI MENGGUNAKAN SECRET SHARING SCHEME

Pada makalah ini, sistem pemulihan sandi yang menggunakan pembagian rahasia Shamir diimplementasikan menggunakan bahasa Python. Sistem yang dibuat masih bersifat sangat general supaya dapat diimplementasikan di berbagai sistem lainnya. Beberapa asumsi dan batasan yang digunakan pada implementasi sistem pemulihan sandi pada makalah ini diantaranya :

1. Setiap pengguna memiliki akun yang dikenali berdasarkan *username*.

2. Setiap pengguna yang ingin melakukan pemulihan *password* menggunakan metode ini, sudah menentukan siapa saja yang menerima *share* (pada makalah ini disebut dengan *fragment*) untuk memulihkan *password*.

3. Setiap pengguna yang telah meminta untuk melakukan pemulihan *password* dengan metode ini, disimpan datanya pada basis data (dalam bentuk file txt) *secrets.txt* dengan format "[Username], [*p*], [*M*], $[s_1]$, $[s_2]$, … $[s_n]$". Sehingga seorang user hanya bisa meminta pemulihan *password* dalam satu waktu.

```
rafi.adyatma, 10000453, 738420, 291388, 454393
raissa.azzahra, 10000303, 494872, 82161
```

Gambar 3. Basis Data dalam bentuk file txt (*secrets.txt*)

4. Metode untuk membagikan pembagian rahasia (*fragments*) belum ditentukan, hal ini bertujuan untuk kepentingan generalisasi. Untuk saat ini, pembagian rahasia disimpan menggunakan file txt dengan nama sesuai dengan *username* (Contoh : rafi.adyatma.txt). *Fragments* disimpan dengan format "$[x_i]$, $[y_i]$" per baris.

```
1, 1484201
2, 3138768
3, 5702121
4, 9174260
5, 3554732
```

Gambar 4. Contoh Shares dalam file txt

5. Melakukan pembentukan polinomial awal dan pembentukan polinomial berdasarkan *fragments* pada file yang berbeda dan dihitung sebagai dua proses yang berbeda. Permintaan pemulihan dilakukan pada *request_recovery.py*, sedangkan rekonstruksi *password* dilakukan pada *retrieve_password.py*.

Berikut kode *main* untuk *request_recovery.py* :

```
if __name__ == "__main__" :
    while (True):


print("--------------------------------
Request Password
Recovery--------------------------------
-")
```

```
        username = str(input("Username
: "))
        w = int(input("Number of Shares
(w) : "))
        t = int(input("Minimum Shares
to Recover password (t / Threshold) :
"))

        if(t > w):
            print("Number of Shares
must be larger than minimum
threshold")
            continue

        # Initialize Secret
        status =
initialize_secret(username, t, w)

        if (status):
            print(f"Password-Recovery
for {username} has been Requested!")
            break

        else :
            print(f"Failed to Request
Password-Recovery for {username}")
            break

        print()
```

Berikut kode *main* untuk *retrieve_password.py* :

```
if __name__ == "__main__" :

    while(True):

print("-------------------------------
Retrieve
Password-------------------------------
-")
        username = str(input("Username
: "))
        total_fragments =
int(input("Total shares you have : "))

        fragments = []
        secret_found = False

        for i in
range(total_fragments):

            x = int(input(f"X{i+1} :
"))
            y = int(input(f"Y{i+1} :
"))
            print()

            fragment = [x, y]
            fragments.append(fragment)

        status = find_secret(username,
fragments)

        break
```

request_recovery.py menggunakan fungsi initialize_secret, berikut adalah kode dari fungsi tersebut :

```
def initialize_secret(username, t, w):
    # Read DB
    f = open("secrets.txt", "r")
    usernames, p_array, all_coefs =
read_secrets()
```

```
    f.close()

    idx = find_username(username,
usernames)

    # Username is already on DB
    if (idx != -1):
        print("User already has key")
        return False

    coefs, p = generate_polynomial(t)
    fragments =
generate_fragments(coefs, w, p)

    # Write fragments to file
    f = open(username + ".txt", "a")

    for pair in fragments:
        curr_line = f"{pair[0]},
{pair[1]}\n"
        f.write(curr_line)
    f.close()

    # Write to secrets.txt as well
    write_secrets(username, p, coefs)

    return True
```

retrieve_password.py menggunakan fungsi find_secret, berikut adalah kode dari fungsi tersebut :

```
def find_secret(username, fragments):

    # Read DB
    usernames, p_array, all_coefs =
read_secrets()
```

```
    idx = find_username(username,
usernames)

    # If username not found, idx = -1
    if (idx == -1):
        print("User has no secret key")
        return False

    # Get essential attributes
    p = int(p_array[idx])
    coefs = all_coefs[idx]

    t = len(coefs)
    total_fragments = len(fragments)

    if (t > total_fragments):
        print(f"Require {t -
total_fragments} more fragments to get
the password !")
        return False

    if (total_fragments > t):
        fragments = fragments[0:t]
        total_fragments =
len(fragments)

    M = int(coefs[0])

    # Test keys using lagrange
interpolation
    M_forged =
lagrange_interpolation(fragments, p)

    if (M != M_forged):
        print("Invalid Keys !")
        return False

    else :
        print("Key forged ! \nYour
```

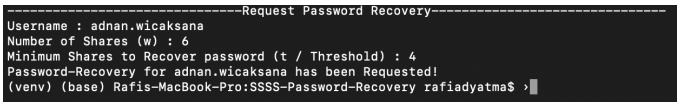
```
Recovery-Password : ", M_forged)


   return True
```

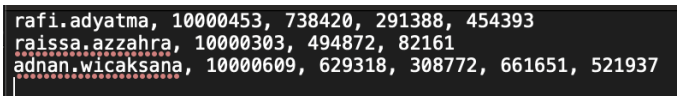
6. Sandi pengguna berbeda dengan rahasia (polinomial) yang di-*generate* secara acak oleh komputer. Sehingga, rahasia *M* digunakan sebagai *token* untuk melakukan *reset password* untuk pengguna yang bersangkutan.

IV. SCREENSHOT PROGRAM

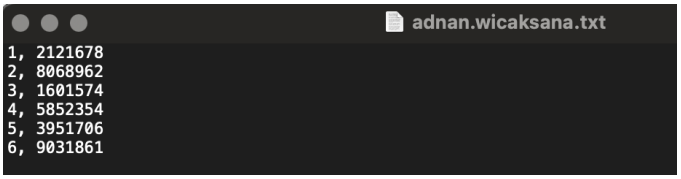
Berikut adalah *screenshot* hasil permintaan pemulihan *password* untuk user "adnan.wicaksana" dengan t = 4 dan w = 6 :



Gambar 5. Contoh penggunaan *request_recovery.py*

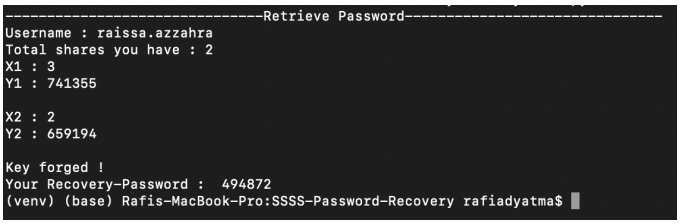


Gambar 6. Basis Data terbaru setelah user adnan.wicaksana meminta pemulihan *password*

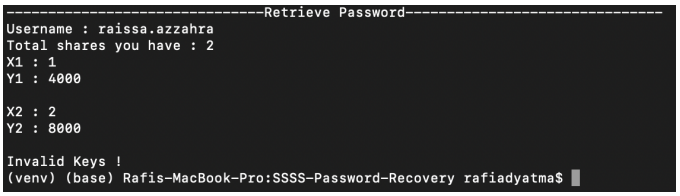


Gambar 7. Shares untuk user adnan.wicaksana

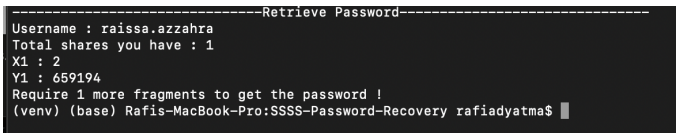
Berikut adalah *screenshot* hasil rekonstruksi *password* untuk user "raissa.azzahra" :



Gambar 8. Rekonstruksi *password* dengan luaran berhasil



Gambar 9. Rekonstruksi *password* gagal akibat terdapat kunci *share* yang salah



Gambar 10. Rekonstruksi *password* gagal akibat *shares* yang dimiliki kurang dari *threshold*

V. KESIMPULAN

Metode pembagian rahasia Shamir dapat menjadi salah satu alternatif yang bisa digunakan pada sistem pemulihan sandi. Pendekatan ini menjadi variasi dari metode sebelumnya yang telah ada seperti tautan *reset password* yang dikirim melalui email maupun SMS. Pendekatan ini menutupi kekurangan sistem yang telah ada, misalnya pada kasus pengguna yang sudah tidak memiliki akses ke email yang terhubung dengan akun tersebut. Pendekatan ini juga sangat layak untuk diimplementasikan pada sistem yang lebih kompleks karena cukup cepat untuk rahasia (*M*) yang tidak terlalu besar.

VI. REPOSITORY GITHUB

https://github.com/Rainburn/SSSS-Password-Recovery

VII. UCAPAN TERIMAKASIH

Penulis mengucapkan syukur kepada Allah yang Maha Esa karena berkat RahmatNya-lah penulis dapat menyelesaikan makalah ini dengan baik. Penulis juga ingin mengucapkan terima kasih kepada semua pihak yang telah membantu penulis dalam membuat makalah berjudul "Sistem Recovery Password dengan Menggunakan Secret Sharing Scheme". Penulis juga berterima kasih kepada Bapak Dr. Ir. Rinaldi, M.T. selaku pembimbing dan dosen mata kuliah IF4020 / Kriptografi.

VIII. REFERENSI

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2017-2018/Skema-Pembagian-Data-Rahasia-(2018).pdf. [Diakses 14 Desember 2021]

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Kriptografi/2021-2022/Pengantar-Kriptografi-(2021).pdf. [Diakses 14 Desember 2021]

[3] https://www.studyfinds.org/forgetting-passwords-locked-out-onliine-accounts/#:~:text=More%20than%20half%20of%20Americans,password%20immediately%20upon%20resetting%20it. [Diakses 14 Desember 2021]

[4] https://www.digitalinformationworld.com/2019/12/new-password-study-finds-78-of-people-had-to-reset-a-password-they-forgot-in-past-90-days.html. [Diakses 14 Desember 2021]

[5] https://brilliant.org/wiki/lagrange-interpolation/. [Diakses 14 Desember 2021]

[6] https://brilliant.org/wiki/polynomials/. [Diakses 14 Desember 2021]

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Jakarta, 16 Desember 2021

Mohammad Rafi Adyatma 13518121