PENGGUNAAN BRUTE FORCE UNTUK MERETAS PASSWORD FILE RAR

Fajar Zaki Al Faris – NIM : 13505084

Program studi Teknik Informatika, STEI, Institut Teknologi Bandung
Jln. Ganesha 10, Bandung
e-mail: if15084@students.if.itb.ac.id

ABSTRAK

Makalah ini membahas tentang aplikasi algoritma *brute force* dalam lingkungan teknologi informasi, yaitu untuk meretas file rar yang ber-*password*. Algoritma brute force yang dipakai untuk meretas sebuah *password* biasa dikenal dengan *brute force attack*.

***Brute force attack* menggunakan sebuah himpunan karakter yang akan digunakan sebagai kombinasi karakter-karakter dari *password* yang akan diretas.**

Adanya himpunan karakter dalam *brute force attack* merupakan salah satu tolak ukur kemangkusan algoritma, semakin banyak anggota dalam himpunan karakter, semakin besar persentase sebuah *password* bisa diretas. Akan tetapi, semakin banyak anggota dalam himpunan karakter dibayar dengan semakin lamanya algoritma *brute force attack* berjalan.

Adanya peretasan *password* file rar menggunakan algoritma *brute force* ini membantu seseorang dalam membuka file rar yang *password*nya hilang atau lupa.

Kata kunci: *brute force attack*, *brute force*, *password*, himpunan karakter, *password cracking*.

1. PENDAHULUAN

1.1 Definisi *Brute Force*

Brute Force adalah pendekatan yang lempang (*straightforward*) untuk memecahkan suatu masalah. Biasanya didasarkan pada pernyataan masalah (*problem statement*) dan definisi konsep yang dilibatkan.

Algoritma *brute force* memecahkan masalah dengan sangat sederhana, langsung, dan jelas (*obvious way*).

1.2 Definisi *Password Cracking*

Password cracking adalah proses menemukan kata kunci rahasia dari data yang telah disimpan dan atau dikirim oleh sistem komputer. Pendekatan umumnya dengan secara terus-menerus menebak *password* yang ingin di-*crack*. Tujuan *password cracking* adalah untuk membantu *user* memperoleh kembali *password* yang hilang/lupa, untuk mendapatkan hak-hak akses ke sebuah sistem, atau sebagai ukuran pencegahan oleh administrator sistem untuk mengecek *password-password* yang dapat di-*crack* dengan mudah.

Istilah *password cracking* terbatas untuk menemukan kembali satu atau lebih *plaintext password* dari *password* yang di-*hash*. *Password cracking* membutuhkan *attacker* yang dapat mengakses *hashed password*, ataupun dengan membaca database verifikasi *password* maupun dengan mencegah *hashed password* dikirim ke jaringan luar. Dapat juga dengan sebuah cara mencoba-coba memasukkan *password* sampai benar seperti yang dikenal dengan metode *brute force attack.*

1.3 Brute Force Attack

Brute Force Attack adalah metode meretas password (*password cracking*) yang sangat terkenal. Metode ini hanya mencoba semua kemungkinan kombinasi karakter. Untuk mendapatkan satu karakter *password* dibutuhkan percobaan sebanyak 26 kombinasi (‘a’ sampai ‘z’), dua karakter *password* membutuhkan akan membutuhkan 26x26 = 676 kombinasi.

Metode ini dijamin akan berhasil menemukan *password* yang ingin diretas. Namun, jumlah kemungkinan kombinasi meningkat secara drastis mengikuti panjang *password* yang ingin diretas dan metode *brute force attack* menjadi tidak berguna.

Disamping panjang karakter *password*, himpunan karakter yang ingin dimasukkan dalam kombinasi juga perlu dipertimbangkan. Semakin panjang himpunan karakter, semakin lama pula waktu yang diperlukan untuk meretas. Inilah biasanya yang menjadi masalah dalam menggunakan metode *brute force attack*, di satu sisi, himpunan karakter harus ditentukan, di sisi lain, semakin

panjang himpunan karakter akan membuat metode ini berjalan semakin lambat.

Cara yang terbaik jika menggunakan metode *brute force attack* ini adalah memulai dengan mencoba *password* pendek menggunakan himpunan karakter penuh, kemudian naikkan panjang *password* secara simultan sambil mengurangi himpunan karakter untuk menjaga waktu tetap stabil.

Hal-hal yang perlu diperhatikan dalam menggunakan metode *brute force attack* :

a. Asumsikan bahwa *password* diketik dalam huruf kecil (*lower case*).
   Pada kasus ini, waktu yang dibutuhkan akan cenderung sama tetapi jika *password* mengandung huruf kapital (*upper case*) cara ini tidak akan berhasil.
b. Coba semua kemungkinan.
   Tujuh karakter *lower case* membutuhkan sekitar 4 jam untuk berhasil mendapatkan *password* tetapi jika dicoba semua kemungkinan kombinasi antara karakter *upper case* dan *lower case* akan membutuhkan waktu sekitar 23 hari.
c. Metode ketiga adalah *trade-off*.
   Hanya kombinasi-kombinasi yang mungkin yang dimasukkan dalam pencarian, sebagai contoh "password", "PASSWORD" dan "Password". Kombinasi rumit seperti "pAssWOrD" tidak dimasukkan dalam proses. Dalam kasus ini, lambatnya proses dapat tertangani tetapi ada kemungkinan *password* tidak ditemukan.

2. METODE *BRUTE FORCE ATTACK* UNTUK MERETAS FILE RAR BER-*PASSWORD*

2.1 Metode Umum Pemecahan Masalah

Penggunaan brute force attack dalam meretas password file rar secara umum dijelaskan sebagai berikut :

a. buat sebuah himpunan karakter dari karakter–karakter yang biasanya dipakai dalam menuliskan sebuah password
b. agar dapat menangani *lower case* dan *upper case* maka dalam himpunan karakter yang dibuat dimasukkan semua anggota himpunan abjad huruf kecil dan huruf besar
c. mulai dari password dengan panjang satu, coba semua karakter dalam himpunan karakter
d. jika gagal naikkan panjang password sebanyak satu karakter kemudian coba semua kemungkinan kombinasi dalam himpunan karakter untuk dicocokkan dengan password sebenarnya.
e. Ulangi langkah d sampai ditemukan password sebenarnya.

2.2 Pseudocode

```
deklarasi global :
  const ABCLEN : integer = 62

  ABC : array of {'0','1','2','3','4','5','6','7','8','9',
  'a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p',
  'q','r','s','t','u','v','w','x','y','z','A','B','C','D','E','
  F','G','H','I','J','K','L','M','N','O','P','Q','R','S','
  T','U','V','W','X','Y','Z'}
  curr_len      : integer = 0
  password      : array of char [0..20]
  filename      : array of char [0..255]
  statname      : array of char [0..255]
```

```
procedure loadstatus ()
{ mendapatkan status password yaitu kombinasi karakter
  terakhir yang dicobakan }

Deklarasi
  status        : array of char [0..255]

Algoritma
  if (status sudah terisi) then
      password ← status
      curr_len ← panjang(password)
```

```
procedure loadstatus ()
{ menyimpan status password yaitu kombinasi karakter
  terakhir yang dicobakan }

Deklarasi
  status        : array of char [0..255]

Algoritma
  if (status sudah terisi) then
      status ← password
```

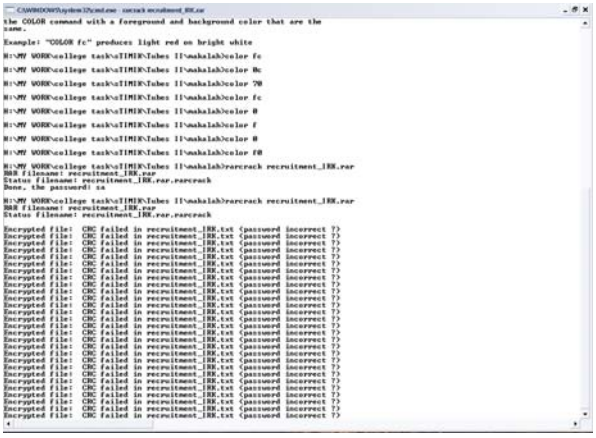
```
function abcnumb (input a : character) → integer
{ menghasilkan indeks dari himpunan karakter ABC
  yang berelemen 'a' }

Deklarasi
  i     : integer

Algoritma
  for i ← 0 to (ABCLEN-1) do
      if ( ABC[i] = 'a') then
          return i
      else
          return 0
      endif
  endfor
```

```
procedure nextpass (input/output  p : array of character
                    [0..255], input a : integer)
{ mencoba karakter ke-n dari password sementara untuk
  dicoba masukkan sebagai input }

Deklarasi
  i      : integer

Algoritma
  if  (p[n] = ABC[ABCLEN-1]) then
      p[n] ← ABC[0]
      if  (n > 0) then
         nextpass(p, n-1)
      else
        for i ← curr_len downto 0 do
            p[i + 1] ← p[i]
        endfor
        p[0] ← ABC[0]
      endif
  else
      p[n] ← ABC[abcnumb(p[n]) + 1]
  endif
```

```
procedure brute_force_attack (input arg : array of
                              character)
{ menggunakan metode brute force attack untuk meretas
  file berekstensi rar}

Deklarasi
  unrar        : file
  cmd          : array of character [0..400]
  ret          : array of character [0..255]
  found        : boolean
  savenow      : integer = 0

Algoritma
  unrar ← arg
  if (curr_len = 0) then
     password[0] = ABC[0];   curr_len = 1;
  endif
  while (found)  do
      savenow = savenow + 1
      bukafile(unrar)
      if (unrar tidak kosong) then
        while (unrar belum akhir file) do
            ret ← unrar
            if (ret <> "All OK") then
              savestatus()
              found = true
              EXIT_SUCCES
            endif
        endwhile
        tutupfile(unrar)
      else
        EXIT_FAILURE
      endif
```

```
      if (savenow >= 500) then
         savenow ← 0
         savestatus()
      endif

      nextpass(password, curr_len - 1)
  endwhile
```

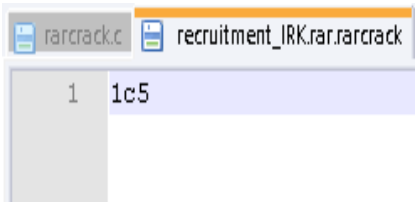
2.3 Contoh aplikasi



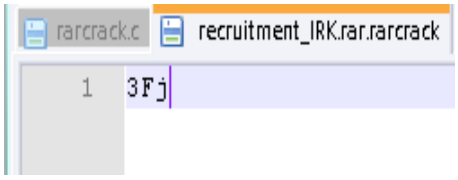
Gambar 1. Program sedang berusaha meretas *password* file 'recruitment_IRK.rar'

Aplikasi di atas sedang berusaha meretas *password* dari file 'recruitment_IRK.rar' dengan panjang *password* asli tiga karakter. Dengan menggunakan *brute force attack*, program di atas berusaha memasukkan kombinasi karakter yang ada di dalam himpunan karakter mulai dari kombinasi 1 digit karakter kemudian berlanjut ke 2 digit kombinasi karakter jika dalam satu kombinasi digit tidak ditemukan, demikian seterusnya sampai ditemukan password yang sebenarnya.

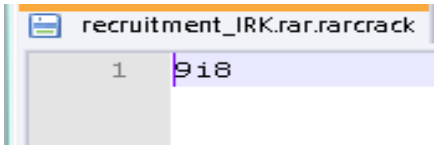
Untuk program di atas, panjang *password* satu karakter membutuhkan 62 (26 huruf kecil, 26 huruf besar, 10 angka) kombinasi, jika panjang *password* dua karakter maka kombinasi yang dibutuhkan adalah 62x62 = 3844 kombinasi. Berarti untuk panjang *password* tiga karakter kombinasi yang dibutuhkan 62 x 62 x 62 = 238328 kombinasi.



Gambar 2. Kombinasi karakter yang diperoleh pada menit ke-20

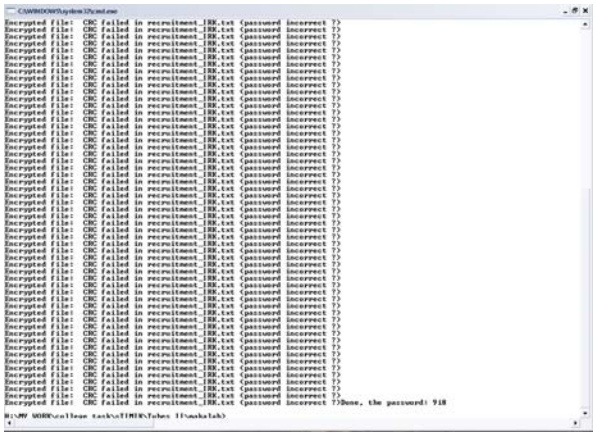


Gambar 3. Kombinasi karakter yang diperoleh pada menit ke-32



Gambar 4. Kombinasi karakter yang diperoleh pada menit ke-58

Pada menit ke-58 diperoleh kombinasi karakter '9i8' yang merupakan *password* asli dari file 'recruitment_IRK.rar'.



Gambar 5. Program berhasil meretas *password* dari file 'recritment_IRK.rar'

IV. KESIMPULAN

a. Penggunaan *brute force* untuk meretas sebuah *password* merupakan suatu hal yang tidak efektif. Namun, jika diinginkan hasil yang pasti, *brute force*lah salah satu jawabannya.
b. Kemangkusan *brute force attack* dalam meretas *password* sangat bergantung pada anggota himpunan karakter yang didefinisikan.
c. *Brute force attack* sangat mangkus untuk meretas *password* sederhana dengan panjang karakter kecil.
d. *Brute force attack* juga akan lebih mangkus jika sebelumnya diketahui panjang *password* yang ingin diretas.
e. Semakin banyak anggota himpunan karakter semakin lama pula *brute force attack* bekerja karena semakin banyak pula kemungkinan kombinasi yang harus dicoba.
f. Salah satu cara untuk mempercepat kinerja *brute force attack* adalah dengan cara menggunakan superkomputer (yang terpenting adalah kecepatan CPU), menggunakan grid komputing, dan sebagainya.
g. Salah satu cara menghitung waktu yang dibutuhkan oleh *brute force attack* untuk memecahkan sebuah *password* adalah dengan menggunakan *password calculator*.

Tabel 1. Kebutuhan waktu *brute force attack* meretas *password*. Asumsi kecepatan *brute force* 500.00/detik

Panjang password	Himpunan karakter			
	Huruf *lower case*	Huruf dan angka lower case	Huruf lower case dan upper case	Semua karakter ASCII
<= 4	instan			2 menit
5	Instan	2 menit	12 menit	4 jam
6	10 menit	72 menit	10 jam	18 hari
7	4 jam	43 jam	23 hari	4 tahun
8	4 hari	65 hari	3 tahun	463 tahun
9	4 bulan	6 tahun	178 tahun	44530 tahun

REFERENSI

[1] *http://en.wikipedia.org/w/index.php?title=Brute_force_attack.*
tanggal akses : 21 Mei 2007 pukul 17.00

[2] *http://lastbit.com/password-recovery-methods.asp*
tanggal akses : 21 Mei 2007 pukul 17.00

[3] *http://en.wikipedia.org/wiki/Plaintext*
tanggal akses : 21 Mei 2007 pukul 17.00

[4] Kedves, David Zoltan, (2006). rarcrack-0.1. *http://sourceforge.net/rarcrack-0.1.*
tanggal akses : 21 Mei 2007 pukul 17.00

[5] Munir, Rinaldi. (2007). Diktat Kuliah IF2251 Strategi Algoritmik. Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.