Implementasi Kombinatorial untuk Menghitung Lama Waktu dalam Memecahkan Password dengan Serangan *Brute-Force*

Ariya Adinatha - 13519048
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13519048@std.stei.itb.ac.id

***Abstract*—Password atau sandi merupakan suatu gabungan huruf, angka, maupun simbol yang digunakan untuk mengkonfirmasi identitas pengguna. Keunikan suatu sandi diperlukan untuk menghindari mudahnya sandi untuk dipecahkan. Dengan menggunakan aplikasi *Burp Suite* dan metode kombinatorial, dapat diketahui waktu maksimal yang dibutuhkan untuk memecahkan sandi dengan kriteria tertentu.**

***Keywords*— Brute Force, Combinatory, Password, Password Cracking.**

I. PENDAHULUAN

Sandi merupakan suatu kumpulan karakter ataupun angka yang biasa digunakan untuk mengkonfirmasi identitas seseorang. Dalam aplikasi internet diperlukan autentikasi untuk membuktikan identitas orang tersebut sehingga orang tersebut dapat mengakses akunnya.

Pesatnya perkembangan internet tidak luput dari aksi kejahatan siber, pencurian akun merupakan salah satunya. Hal ini dapat terjadi dikarenakan kelalaian pembuat situs maupun kelalaian pengguna seperti kata sandi yang lemah dan mudah ditebak, seseorang bisa saja melakukan serangan menggunakan aplikasi bantuan seperti *Burp Suite* untuk meretas sandi tersebut.

Dengan menggunakan teori kombinatorial dan *Burp Suite*, dapat dihitung berapa lama waktu maksimal yang diperlukan untuk meretas kata sandi seseorang.

II. KOMBINATORIAL

Kombinatorial adalah cabang matematika untuk menghitung jumlah penyusunan objek tanpa harus mengenumerasi semua kemungkinan susunannya. Terdapat 2 kaidah dasar dalam menghitung kombinatorial yaitu kaidah perkalian dan kaidah penjumlahan [2].

A. Kaidah Perkalian

Kaidah perkalian adalah prinsip menghitung banyak kemungkinan **a** dan banyak kemungkinan **b.** Kaidah perkalian dimana **a.b** akan menghasilkan banyaknya kemungkinan **a** dan **b** terjadi. Dapat dirumuskan dengan:

Total Kemungkinan = **a** x **b**

Dengan **a** sebagai kemungkinan pertama dan **b** sebagai kemungkinan kedua. Maka hasil perkaliannya akan menunjukkan banyaknya 2 kemungkinan tersebut terjadi.

B. Kaidah Penjumlahan

Kaidah penjumlahan adalah penjumlahan kemungkinan **a** dan kemungkinan **b** dimana hanya terjadi salah satu dari kemungkinan tersebut. Dapat dirumuskan sebagai :

Total Kemungkinan = **a** + **b**

Dengan **a** sebagai kemungkinan pertama dan **b** sebagai kemungkinan kedua, dan total kemungkinan merupakan penjumlahan keduanya, yang menunjukkan banyaknya kemungkinan dimanan hanya 1 kemungkan tersebut yang terjadi.

III. BRUTE FORCE

Brute-force adalah serangan siber dengan cara mencoba semua karakter yang ada hingga ditemukan sandi yang cocok. Dengan menggunakan kombinasi dari karakter, angka, dan simbol, sandi yang lemah dapat dengan mudah diretas.

Metode ini memiliki tingkat keberhasilan yang lebih tinggi dibandingkan yang lainnya, tanpa mempedulikan teknik enkripsi sandi yang digunakan. Masalah dari teknik brute force ini akan memakan waktu yang lama. Semakin banyak kombinasi karakter dan semakin panjang password yang dicari, memerlukan waktu yang lebih lama[1].

IV. CHACRACTER SET

Karakter set merujuk pada bilangan komposit dari karakter berbeda yang digunakan dan didukung oleh *software* dan *hardware* komputer. Berisikan kode, *bit pattern* yang mendefinisikan karakter tertentu. Character set juga dapat disebut sebagai *character map*, *charset*, atau *character code[9]*.

Set dari karakter ini dibedakan menjadi 4 bagian, yaitu :

A. Huruf Besar

Karakter set huruf besar, yang berjumlah 26 karakter, mulai dari 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

B. Huruf Kecil

Karakter set huruf kecil yang berjumlah 26 karakter, mulai dari 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

C. Angka

Karakter set angka yang berjumlah 10 karakter yang hanya berisi numerik, dimulai dari angka 0,1,2,3,4,5,6,7,8,9

D. Simbol

Karakter set simbol yang berjumlah 32 karakter, dimulai dari symbol !, @, #, $, %, ^, &, *, (,), `, ~, -, _, =, +, [, {,], }, :, ;, ”, ’, <, ,, >, ., /, ?, |, \.

V. *BURP SUITE*

Burp atau *Burp Suite* adalah aplikasi untuk menguji keamanan suatu situs. Aplikasi ini ditulis menggunakan bahasa *Java* dan dikembangkan oleh PortSwigger Security[3].

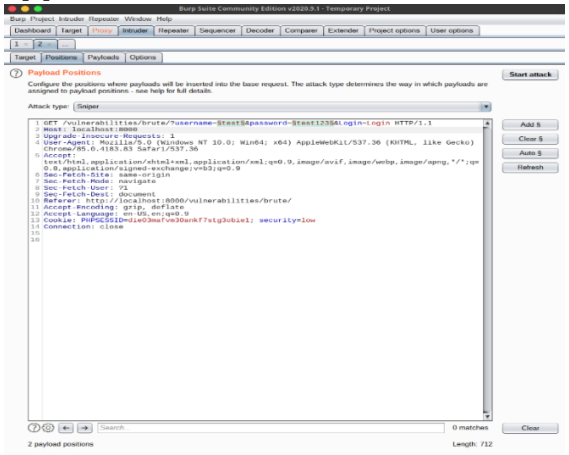
Tool ini memiliki dua versi: versi gratis yang dapat diunduh secara gratis (Free Edition) dan versi lengkap yang dapat dibeli setelah masa uji coba (Professional Edition). Versi gratis telah dikurangi fungsionalitas-nya secara signifikan. Ini dikembangkan untuk memberikan solusi komprehensif untuk pemeriksaan keamanan aplikasi web. Selain fungsionalitas dasar, seperti *server proxy*, pemindai, dan penyusup, alat ini juga berisi opsi yang lebih canggih seperti *spider*, *repeater*, *decoder*, *comparer*, *extender* dan *sequencer*[3].

A. Spider

Fitur *Spider* pada *Burp Suite* berguna untuk melakukan proses *crawling* (mengindeks isi suatu situs) pada situs yang ditargetkan. Setelah proses selesai, *Burp Suite* akan menampilkan hasil halaman yang terindeks pada situs tersebut[9].

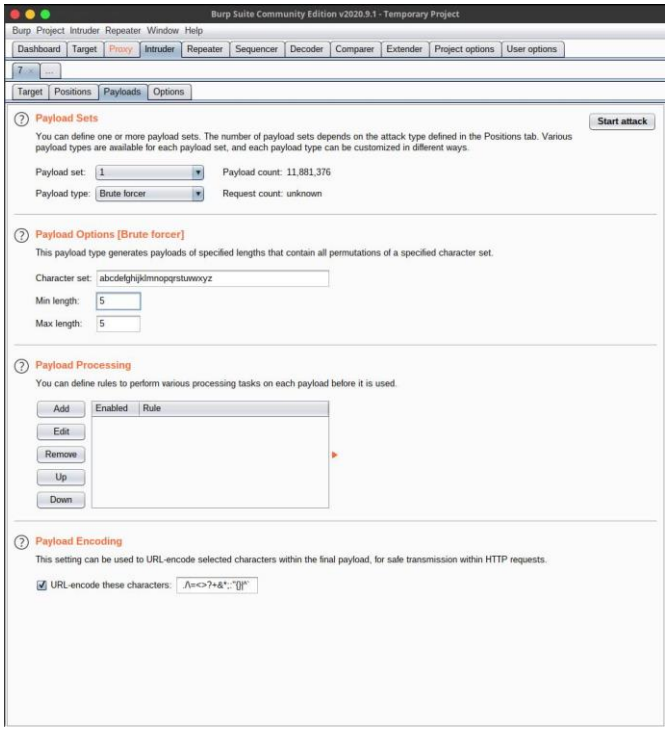
B. Intruder

Fitur *Intruder* digunakan untuk melakukan serangan pada aplikasi web secara otomatis. Fitur ini sangatlah berguna dan mudah untuk digunakan, beberapa contoh serangan yang dapat dilakukan adalah serangan *brute-force*, *blind SQL Injection*, dan lain-lain[7].



Gambar 1. *Intruder*

Intruder bekerja dengan cara menerima *HTTP request* dan memodifikasi dan menganalisis *request* tersebut, Untuk melancarkan serangan, pengguna harus menentukan *payload* yang akan digunakan dan posisinya pada *request* yang telah didapatkan. *Burp Suite* sendiri menyediakan berbagai macam *payload* seperti *Custom Iterator*, dimana *payload* ini akan membuat permutasi dari karakter atau dari set karakter yang telah diberikan[7].



Gambar 2. *Payload*

C. Repeater

Fitur *Repeater* pada *Burp Suite* berguna untuk memanipulasi *HTTP* yang berguna untuk menganalisis respons dari situs. *Repeater* dapat digunakan untuk mengubah nilai dari parameter *input* untuk melakukan *testing* terhadap celah *input* secara otomatis. Fitur ini biasa digunakan bersama dengan *intruder*[8].

D. Decoder

Fitur *Decoder* biasa digunakan untuk mengubah data yang telah di-*encode* ke dalam bentuk lain, seperti hash dan *raw data*. Dengan menggunakan teknik *heuristic*, *Decoder* dapat mengenali beberapa format *encoding[5]*.

E. Comparer

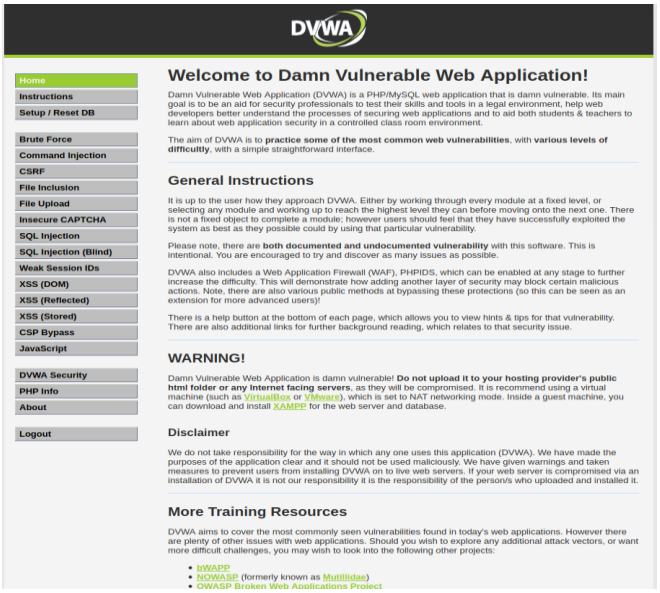
Fitur *Comparer* digunakan untuk melakukan komparasi visual diantara 2 data. Fitur ini sering digunakan untuk komparasi *login*, dimana respons situs terhadap *login* yang gagal akan berbeda tergantung dengan kegagalannya. Selain itu fitur ini juga sering digunakan bersama dengan *intruder*[4].

VI. DVWA

DVWA atau dikenal sebagai *Damn Vulnerable Web App*

adalah sebuah aplikasi web yang berbasis PHP/MySQL, dimana aplikasi web ini sangatlah rentan dan memiliki banyak *bugs*. Aplikasi web ini bertujuan untuk mengasah kemampuan *security progessionals* dalam melakukan pengujian kemampuan maupun *tools* pada *environment* yang legal. Selain itu, aplikasi web ini juga dapat membantu *web developer* dalam memahami proses – prose dalam mengamankan sebuah situs, juga untuk guru maupun murid sebagai bahan pembelajaran terhadap pengamanan sebuah website.

DVWA menyediakan pilihan tingkat kesulitan, mulai dari *low, medium, high,* dan *impossible.* Tingkat kesulitan ini akan meningkatkan keamanan dari DVWA. Semakin tinggi tingkatannya, maka semakin sedikit *bugs* yang terdapat pada DVWA.

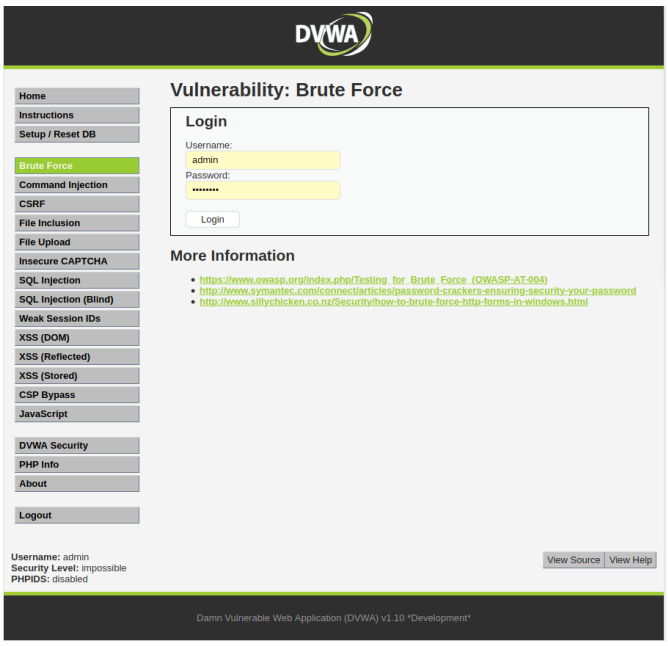


Gambar 3. *DVWA*

Terdapat beberapa *bugs* yang telah disediakan di dalam DVWA, yaitu :

A. Brute Force

Serangan yang dilakukan dengan menebak dan juga mencoba suatu *username* maupun *password* hingga ditemukan sandi yang tepat. Penyerang dapat melakukan kombinasi antara huruf, angka, dan symbol untuk menghasilkan sandi yang tepat. Umumnya serangan dilakukan dengan bantuan *wordlist*, yaitu sebuah kumpulan sandi yang sering digunakan oleh orang – orang dengan harapan sandi yang digunakan terdapat pada *wordlist* tersebut. Beberapa contoh dari sandi yang sering digunakan adalah “12345”, “password”, “admin123”, dan sebagainya. Salah satu *wordlist* yang sering digunakan untuk melakukan seragan *brute-force* adalah *rockyou.txt* dimana *rockyou.txt* ini merupakan kompulan sandi dari sosial media yang telah bocor di internet sebelumnya.



Gambar 4. *Brute Force*

B. Command Injection

Command Injection merupakan kerentanan yang disebabkan oleh *input string* yang tidak difilter dan diteruskan ke dalam *shell system*. Penyerang dapat melalukan eksploitasi terhadap kerentanan ini dengan menggunakan format yang sesuai untuk memanggil perintah pada *shell*. Pengujian *Command Injection* dapat menggunakan symbol &, |, ;, &&, $().

C. CSRF

Cross-Site Request Forgery atau dikenal juga dengan *one click attack atau session riding* disingkat dengan CSRF atau XSRF, merupakan bentuk eksploitasi website yang dieksekusi atas wewenang korban, tanpa dikehendakinya. CSRF menipu website melalui *request* dari *user* yang dipercaya. Serangan bekerja melalui *link* atau *script* pada halaman yang diakses *user*. *Link* tersebut dapat berupa gambar/image yang terhubung ke website tertentu. Jika website menyimpan informasi otentikasi dalam sebuah *cookie* yang belum *expire*, maka dengan melakukan klik ke *link* tersebut akan menyebabkan website diakses menggunakan *cookie user* yang melakukan klik. Dengan kata lain, penyerang menipu browser user untuk mengirimkan HTTP *requests* ke website target[11].

D. SQL Injection

Serangan *SQL injection* atau Injeksi *SQL* merupakan teknik serangan injeksi kode yang memanfaatkan celah keamanan yang terjadi pada *layer* basis data dari sebuah aplikasi. Hal ini terjadi sebagai akibat dari data yang diinputkan oleh pengguna tidak dilakukan validasi dan dimuat di dalam baris perintah *query SQL*. Dengan demikian menjadikan sebagian data yang diinputkan pengguna tersebut diperlakukan sebagai bagian dari kode SQL[12].

Kerentanan ini yang menyerang database dengan menjalankan perintah berupa *SQL Query* sehingga penyerang dapat melihat maupun memanipulasi data yang seharusnya tidak dapat diakses. Untuk memahami serangan *SQL injection*

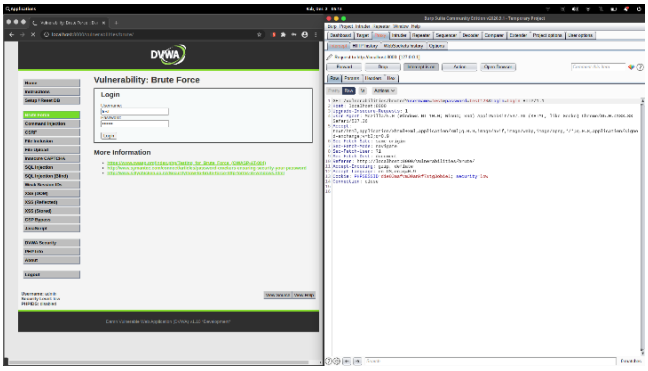
diperlukan pemahaman mengenai *SQL*, yaitu sebuah bahasa yang digunakan untuk mengakses, menyimpan, maupun mengubah data yang memiliki basis relasional.

E. XSS

Cross Site Scripting atau biasa disebut dengan XSS merupakan sebuah kerentanan pada aplikasi web yang memungkinkan seseorang untuk menginjeksi *script* pada *client-side* di halaman website sehingga menyebabkan serangan ini seolah-olah datang dari website itu sendiri. Serangan ini dapat digunakan untuk melewati *access control* dan dapat berdampak pada pencurian infomasi sensitif seperti *cookie* maupun penyisipan aplikasi berbahaya. Alasan mengapa kerentanan ini disingkat menjadi XSS bukannya CSS yaitu untuk menghindari keambiguan dengan *Cascading Style Sheet*, sebuah markup language yang digunakan untuk mengatur tampilan sebuah website. Terdapat 4 jenis XSS mulai dari *Reflected XSS, Stored XSS, DOM Based XSS, Self XSS.* Namun diantara ke 4 jenis XSS ini yang paling sering ditemukan pada website adalah *Reflected XSS.*

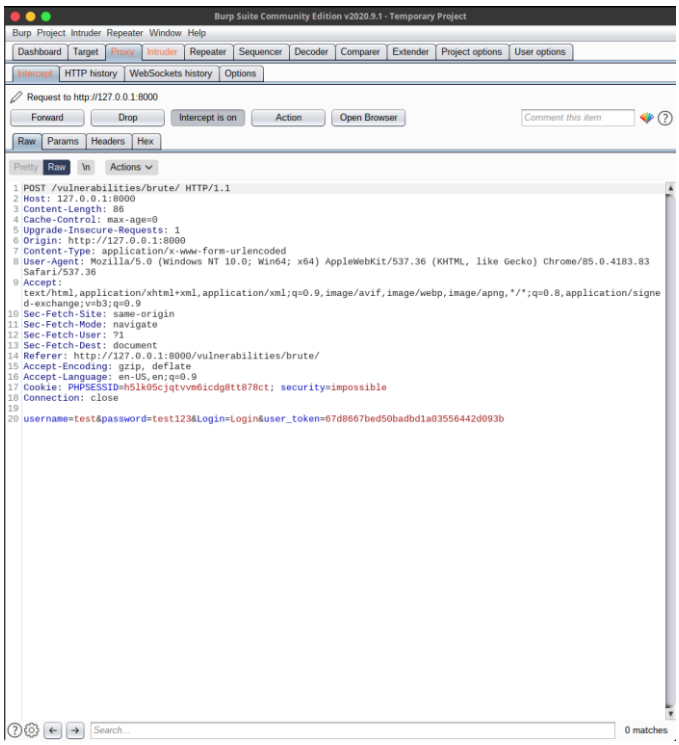
VI. MENGHITUNG WAKTU YANG DIBUTUHKAN UNTUK MEMECAHKAN SANDI MELALUI BRUTE FORCE

Salah satu teknik yang masih tetap digunakan seiring berjalannya waktu adalah teknik *brute-force*. Tidak dapat dipungkiri efektivitas teknik ini, hanya saja lamanya waktu yang diperlukan untuk mendapatkan hasil yang sesuai menjadi kendala. Untuk mempermudah serangan *brute-force* ini digunakan sebuah *tools* yang bernama *Burp Suite*. Target serangan kali ini adalah sebuah situs lemah bernama DVWA yang dijalankan menggunakan *local machine*. DVWA sendiri merupakan sebuah aplikasi web yang dirancang untuk melatih *penetration tester*.



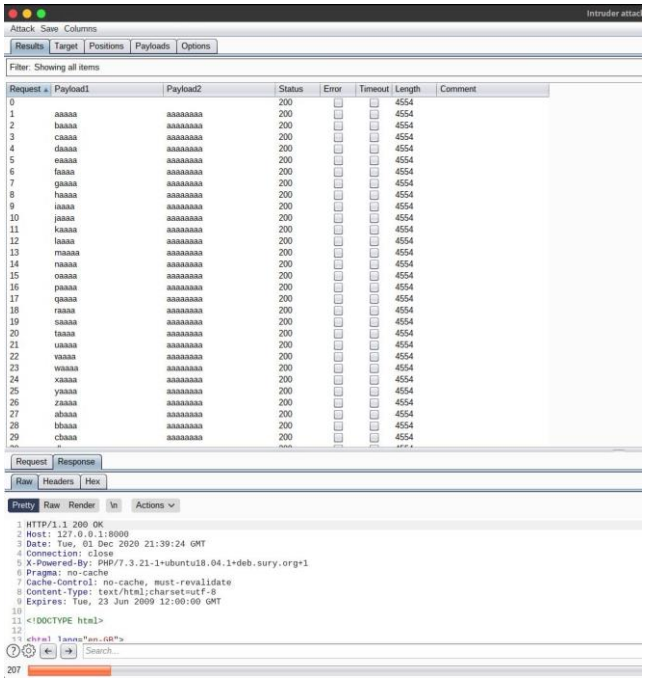
Gambar 5. DVWA dan *Burp*

Ketika masuk ke dalam DVWA, terdapat pilihan *brute-force* pada bagian kiri. Darisana ditampilkan sebuah halaman sederhana yang memiliki 2 parameter *input* yaitu *username* dan *password.* Sebelum memulai serangan *Burp Suite* dikonfigurasi terhadap *browser* agar mampu menangkap *request.* Setelah selesai dikonfigurasi *Burp Suite* akan menahan *request* tersebut sebelum dikirimkan kepada *server*. Hal ini memungkinkan pengguna untuk melakukan pengubahan terhadap parameter – parameter yang akan dikirim, dalam hal ini yaitu *username* dan *password*.



Gambar 6. *Request*

Terlihat pada bagian bawah dari gambar 6 terdapat parameter *username* dengan isi “test” dan parameter *password* yang berisi dengan “test123”, kedua parameter inilah yang akan diubah dengan menggunakan *intruder* dan dikirim secara berulang – ulang oleh *Burp Suite* hingga ditemukan *username* dan *password* yang tepat.

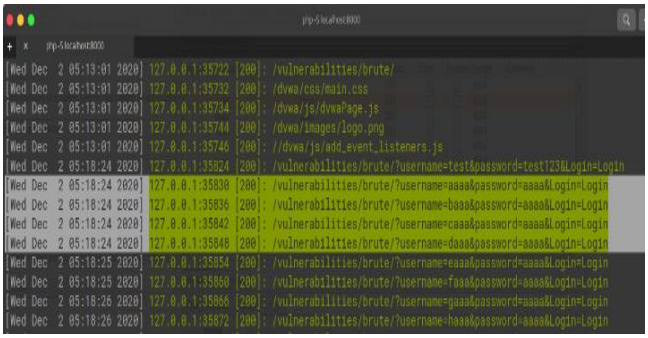


Gambar 7. *Brute-Force*

Terlihat dari gambar ke 7 *Burp Suite* memulai penebakan parameter *password* dan menghasilkan *Status* 200, yang artinya tebakan tersebut tidak benar, *Burp* akan terus mengirim *status*

200 jika tebakan masih salah.

Untuk itu dicari kecepatan *request* yang dapat dilakukan oleh *Burp Suite Free Edition* dalam melakukan serangan *brute-force* ini. Dilakukan pengecekan terhadap *log server*.



Gambar 8. *Log Server*

Terlihat pada gambar 8, Setelah *Burp Suite* dijalankan dan melakukan serangan *brute-force*, diketahui dari *log server* bahwa *Burp Suite* mampu mengirimkan 5 *request* / detik. Dapat dilihat pada waktu 05:18:24 *Burp Suite* mengirimkan 5 *request* pada waktu tersebut.

Hal ini dikarenakan *Burp Suite Free Edition* membatasi kecepatan *request* yaitu 5 *request* per detiknya, untuk menambahkan banyak *request* per detiknya dapat menggunakan *Burp Suite Professional Edition* atau menggunakan *tools* lain yang mampu melakukan *brute-force* seperti *Hydra* yang mampu mengirimkan 16 *request* per detiknya.

Setelah diketahui kecepatan serangan, selanjutnya akan dicari banyak kemungkinan dari sandi dengan menggunakan kombinatorial. Dimana banyak kemungkinan sandi dirumuskan dengan:

$$\text{Total Kemungkinan} = \mathbf{a} \times \mathbf{b}$$

Total kemungkinan didapatkan dengan menggunakan teori kombinatorial dan waktu yang dihasilkan dalam satuan sekon. Untuk kecepatan *request* didapatkan dari percobaan dengan menggunakan *Burp Suite* dan melihat catatan *request* pada *log server*. Semakin panjang suatu sandi dan semakin banyak karakter set yang digunakan, maka akan semakin lama waktu yang dibutuhkan untuk memecahkan sandi tersebut.

Untuk mencari total kemungkinan tersebut dapat dicari dengan memangkatkan banyak karakter set dengan panjang sandi banyak kemungkinannya dapat dirumuskan:

$$\text{Total Kemungkinan} = \mathbf{karakter^n}$$

Karakter sebagai banyak karakter pada karakter set dan n merupakan panjang karakternya.

Maka, untuk mencari kecepatan dalam meretas sandi dengan asumsi kecepatan *request* stabil dan sandi tidak berhasil diretas hingga kemungkinan terakhir, dapat dirumuskan dengan:

$$\text{Waktu} = \frac{\textbf{Total Kemungkinan}}{\textbf{Kecepatan } \boldsymbol{Request}}$$

$$\text{Waktu} = \frac{\textbf{Karakter}^{\boldsymbol{n}}}{\textbf{Kecepatan } \boldsymbol{Request}}$$

Dengan **n** yang berarti panjang karakter. Sebagai contoh, pada pin yang memiliki 4 digit angka memiliki karakter sebanyak 10, mulai dari 0 sampai dengan 9, kemungkinan sandi yang ada sebanyak

$$10^4 = 10 \times 10 \times 10 \times 10 = 10{,}000 \text{ kemungkinan}$$

kemudian untuk menghitung waktu yang dibutuhkan untuk meretas sandi tersebut, dilakukan pembagian terhadap total kemungkinan dengan kecepatan *request* per detiknya, yaitu

$$10{,}000/5 = 2{,}000 \text{ detik}$$

Jika diubah dalam bentuk menit didapatkan 33,3 menit. Sedangkan perhitungan untuk sandi yang memiliki huruf besar dengan 26 karakter, huruf kecil dengan 26 karakter, angka dengan 10 karakter, symbol dengan 32 karakter dan panjang sandi 8 karakter

$$\text{Karakter} = 26+26+10+32 = 94$$

Setelah karakternya didapatkan kemudian dipangkatkan sesuai dengan panjang sandi, yaitu 8

$$\text{Total Kemungkinan} = 94^8$$
$$98^8 = 6{,}095{,}689{,}385{,}410{,}816 \text{ kemungkinan}$$

Didapatkan total kemungkinannya dari sandi tersebut sebanyak 6,095,689,385,410,816 kemudian total kemungkinan ini dibagi dengan kemampuan *Burp Suite* untuk melakukan *request* kepada situs.

$$\text{Waktu} = 6{,}095{,}689{,}385{,}410{,}816 / 5$$
$$\text{Waktu} = 1{,}219{,}137{,}877{,}082{,}163 \text{ sekon}$$

Jika waktu tersebut diubah ke dalam satuan tahun,

$$\text{Waktu} = 1{,}219{,}137{,}877{,}082{,}163 / 31{,}536{,}000$$
$$\text{Waktu} = 38{,}658{,}608.4818$$

Jika waktu dalam bentuk sekon dibagi dengan banyaknya detik dalam satu tahun (31,536,000 detik), maka didapatkan waktu yang dibutuhkan untuk meretas *password* tersebut dalam bentuk tahun.

Maka diperlukan sekitar 38,658,608 tahun untuk meretas sandi yang menggunakan kombinasi dari huruf besar, huruf kecil, angka, dan simbol menggunakan *Burp Suite Free Edition*.

Dibuat tabel yang berisi perhitungan waktu dengan menggunakan kecepatan *request* sebesar 5 *request* per detik

Kecepatan *Request / detik* = 5 $\text{Waktu} = \frac{\textbf{Karakter}^{\boldsymbol{n}}}{\textbf{Kecepatan } \boldsymbol{Request}}$

Panjang Karakter	Karakter Set			
	Angka (10 karakter)	Huruf kecil dan angka (36 karakter)	Huruf besar, huruf kecil, dan angka (62 karakter)	Huruf besar, huruf kecil, angka, dan symbol (94 karakter)
3	3,3 menit	2,5 jam	13 jam	46 jam
4	33,3 menit	93 jam	34 hari	180 hari
5	5,5 jam	139 hari	5,8 tahun	46 tahun
6	55,5 jam	13 tahun	3,6 abad	43 abad
7	23 hari	496 tahun	223 abad	4112 abad
8	231 hari	178 abad	1384 milenium	38658 milenium
9	6 tahun	6440 abad	85851 milenium	363390 milenium
10	6,3 tahun	231872 abad	5322801 milenium	341587464 milenium
11	634 tahun	834739 milenium	330013703 milenium	3.21092 E+10 milenium
12	6341 tahun	30050617 milenium	2.0460E+12 milenium	3.01827 E+12 milenium

V. Conclusion

Teori kombinasi dapat digunakan diketahui banyak kemunginan sandi yang ada, dengan menggunakan kaidah perkalian yang telah dibahas dan menggunakan banyak karakter sebagai *variable* yang akan dikali sebanyak panjang karakternya.

Untuk mencari banyak karakter dari suatu set karakter dan memangkatkannya dengan panjang dari sandi, maka akan didapatkan banyaknya kemungkinan sandi yang ada.

Ketika banyak kemunginan sandi tersebut dibagi dengan banyaknya *request* yang dapat dilakukan per detiknya, maka ditemukanlah waktu yang dibutuhkan untuk memecahkan sandi tersebut. Kecepatan *request* ini bervariasi dan bergantung dari *tools* yang digunakan untuk menyerang. Pada *Burp Suite Free Edition* kecepatan pengguna dibatasi hanya pada 5 *request* / detiknya, sedangkan pada *tools* sejenis mampu mengirimkan 16 *request* / detiknya.

Metode ini digunakan dengan asumsi *request* yang dilakukan dengan kecepatan yang stabil dan sandi tidak ditemukan hingga kemungkinan yang paling terakhir.

Untuk menghindari jenis serangan *brute-force* ini, sandi yang berukuran panjang dan memiliki karakter set yang beragam sangat direkomendasikan.

VI. Appendix

Makalah ini merujuk pada "Penerapan Kombinatorial untuk Menghitung Kekuatan Sandi dari Serangan *Brute-Force*" yang dibuat oleh Fahziar Riesad Wutono dengan NIM 13512012. Perbedaan makalah ini terletak pada bagian perhitungan. Makalah ini menghitung waktu untuk memecahkan sandi tersebut sedangkan makalah rujukan menghitung kekuatan dari sandi.

VII. Ucapan Terima Kasih

Terima kasih kepada Tuhan YME, berkat rahmayNya makalah ini dapat diselesaikan dengan baik, terima kasih juga kepada dosen penulis, Dr. Nur Ulfa Maulidevi, ST., M.Sc. yang telah membimbing penulis dalam mata kuliah IF2120/Matematika Diskirt. Juga kepada warga HMIF yang telah memberikan banyak dukungan dan bantuan.

Refrensi

[1] Efvy Zam, "Buku Sakti Hacker", Jakarta:mediakita, 2011, pp. 127.
[2] Rinaldi Munir, "Diktat Kuliah IF2120 Matematika Diskrit", Bandung:ITB, 2006, pp. VI 1 - VI 6.
[3] www.onnocenter.or.id/wiki/index.php/Burp_Suite diakses tanggal 1 Desember 2020.
[4] www.portswigger.net/burp/documentation/desktop/tools/comparer diakses tanggal 1 Desember 2020.
[5] www.portswigger.net/burp/documentation/desktop/tools/decoder diakses tanggal 1 Desember 2020.
[6] www.portswigger.net/burp/documentation/desktop/tools/extender diakses tanggal 1 Desember 2020.
[7] www.portswigger.net/burp/documentation/desktop/tools/intruder diakses tanggal 1 Desember 2020.
[8] www.portswigger.net/burp/documentation/desktop/tools/repeater diakses tanggal 1 Desember 2020.
[9] www.portswigger.net/burp/documentation/desktop/tools/spider diakses tanggal 1 Desember 2020.
[10] www.techopedia.com/definition/941/character-set diakses tanggal 1 Desember 2020.
[11] www.mti.binus.ac.id/2018/07/11/cross-site-request-forgery diakses tanggal 9 Desember 2020.
[12] https://bssn.go.id/mengenal-sql-injection-dan-cara-mencegahnya/ diakses tanggal 9 Desember 2020.

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.

Bandung, 9 Desember 2020

Ariya Adinatha - 13519048