Implementasi Algoritma *String Matching* untuk Mendeteksi Serangan *SQL Injection* Pada Aplikasi Web

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

***Abstract*—Algoritma pencocokan string dapat digunakan untuk mencari pola *string* yang cocok pada sebuah file. Terdapat 3 algoritman pencocokan *string* yang terkenal yaitu *Brute Force Algorithm, Knuth-Morris-Pratt Algorithm, Boyer-Moore Algorithm*. Algoritma – algoritma ini dapat juga digunakan untuk mendeteksi serangan *SQL Injection* pada suatu server.**

Keywords—component; SQL Injection; digital forensics; server log; string matching, patter recognition

I. PENDAHULUAN

Semenjak terjadinya pandemi virus corona manusia sudah tidak asing lagi dengan teknologi, pandemi ini memaksa setiap orang untuk tinggal di rumah dan juga menjaga jarak, tentu saja hal ini akan mengubah banyak kegiatan yang biasa dilakukan oleh masyarakat. Pandemi ini membuat banyak hal mengalami peleburan dengan teknologi digital, mulai dari fungsi budaya dan juga fungsi sosial masyarakat. Banyak kegiatan mulai berubah dengan adanya peleburan teknologi digital tersebut, beberapa kegiatan seperti pembelajaran, pekerjaan, pertemuan, dll semuanya dilakukan secara daring.

Walaupun masalah jarak dapat diselesaikan dengan internet, muncul pula masalah – masalah yang baru saat daring ini, yaitu kejahatan *online*. Kejahatan online ini sangatlah beragam mulai dari kejahatan yang memanfaatkan kelemahan sisi manusia maupun kelemahan pada sisi sistem yang ada. Jika tidak ditangani dengan baik, kejahatan ini dapat merugikan banyak pihak, terlebih lagi jika kejahatan tersebut memanfaatkan kelemahan dari sisi sistem, seluruh pengguna sistem tersebut dapat terkena dampak yang merugikan dari pelaku kejahatan *online*.

Untuk itu diperlukan sebuah "penjaga" yang dapat melindungi sistem dari serangan serangan siber tersebut. Penjaga ini harus bisa memantau dan menjaga sistem selama 24 jam, selain memantau dan menjaga, juga harus dapat memperbaiki jika terdapat kesalahan. Tugas ini terlalu berat jika dibebankan ke entitas saja, oleh karena itu penjaga ini memerlukan unsur manusia dan juga unsur program, disini program akan bertugas untuk memantau dan menjaga suatu program dari serangan – serangan yang datang dan memberikan informasi serangan tersebut terhadap manusia, sehingga hasil dari kerusakan yang dilakukan oleh manusia.

Dari banyak serangan yang dilakukan, *SQL Injection* merupakan salah satu jenis serangan yang paling sering dilakukan oleh para penyerang, serngan ini dapat dideteksi oleh program dengan menggunakan algoritma pencocokan *string.*

II. DASAR TEORI

A. SQL

Structured Query Language (SQL) adalah bahasa pemrograman yang digunakan dalam mengakses, mengubah, dan memanipulasi data yang berbasis relasional. Bahasa *query* yang dirancang untuk pengambilan informasi tertentu dari database. Awalnya, *SQL* muncul pada 1970 dengan nama *structured english query language (SEQUEL).* Pada 1986, IBM menggunakan *SEQUEL* dalam berbagai proyek database. Tak lama, nama *SEQUEL* pun diubah menjadi *SQL* agar lebih mudah dieja. Sejak saat itu, *SQL* menjadi semakin populer dalam pengolahan data dan database. Meskipun sudah ada sejak puluhan tahun silam, kini *SQL* masih menjadi salah satu bahasa pemrograman yang paling banyak digunakan. Alasannya, sebagian besar perusahaan menyimpan datanya dalam sebuah database. Meskipun saat ini sudah ada berbagai jenis database, seperti *MySQL*, *Microsoft SQL Server*, dan *PostgreSQL*, mayoritas database tersebut tetap menggunakan dasar *SQL*.

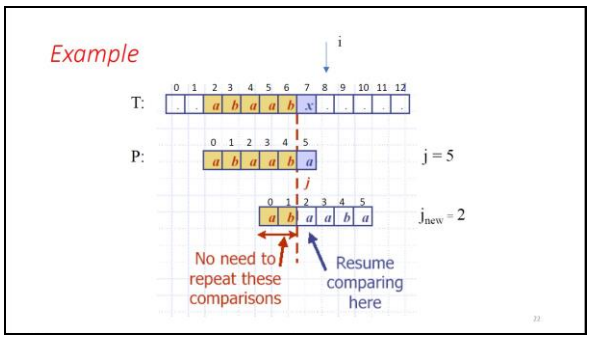
B. String Matching

String matching (pencocokan *string)* merupakan sebuah algoritma yang digunakan untuk mencari kemunculan suatu *string* dalam sebuah teks, *string* dianggap sebagai *pattern* dan dicari *pattern* yang sama pada kumpulan teks yang diinginkan. Terdapat beberapa algoritma *string matching* yang terkenal yaitu :

1. Algoritma *Knuth Morris Pratt*

   Algoritma *Knuth Morris Pratt* merupakan salah satu algoritma yang digunakan untuk mencari *string*, dikembangkan oleh Donald E. Knuth, James H. Morris, dan Vaughan R. Pratt pada tahun 1966. Algoritma ini merupakan jenis *Exact String Matching Algorithm* yang merupakan pencocokan *string* secara tepat berdasarkan susunan karakter maupun jumlah

karakter dari *string* yang sama. Pada algoritma *Knuth Morris Pratt* menyimpan informasi yang digunakan dalam melakukan pergeseran lebih jauh, tidak hanya satu karakter seperti algoritma *Brute Force*.

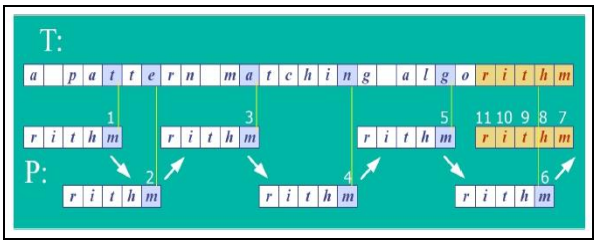


Gambar 1 Ilustrasi cara kerja algoritma *Knuth Morris Pratt* untuk *string matching*

(Sumber : Bahan Kuliah IF2211 Pencocokan String (String/*Pattern Matching*)

2. Algoritma *Boyer Moore*

Algoritma *Boyer Moore* merupakan salah satu algoritma pencarian *string* yang dipublikasikan oleh Robert S. Boyer dan J. Strother Moore pada tahun 1977. Algoritma ini dianggap sebagai algoritma yang paling efisien pada aplikasi umum, tidak seperti algoritma pencarian *string* yang ditemukan sebelumnya, algoritma *Boyer Moore* mulai mencocokkan karakter dari sebelah kanan dari sebuah *pattern.*



Gambar 2 Ilustrasi cara kerja algorima *Boyer Moore* untuk *string matching*

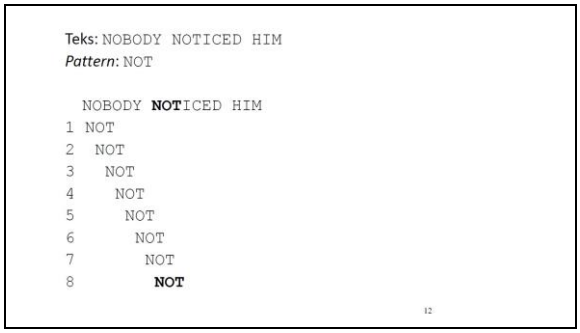
(Sumber : Bahan Kuliah IF2211 Pencocokan String (String/*Pattern Matching*)

3. Algoritma *Regular Expression*

Regular Expression atau disingkat menjadi regex merupakan notasi standar yang mendeskripsikan suatu pola berupa urutan *string* atau serangkaian karakter yang mendefinisikan sebuah pola pencarian. Pola tersebut biasanya digunakan oleh algoritma pencarian *string* untuk melakukan operasi pencarian pada *string.*

4. Algoritma *Brute Force*

Brute Force Algorithm dalam *string matching* dilakukan dengan menelusuri setiap *string* yang ada pada teks yang ingin dicari dan dicocokan secara satu persatu sesuai dengan *pattern* yang diinginkan. Algoritma ini bagus untuk diterapkan jika teks yang dicari memiliki varietas yang tinggi.



Gambar 3 Ilustrasi cara kerja algoritma *Brute Force* untuk *string matching*

(Sumber : Bahan Kuliah IF2211 Pencocokan String (String/*Pattern Matching*)

C. Aplikasi Web

Aplikasi berbasis web adalah aplikasi yang dikembangkan menggunakan HTML, PHP, CSS, JS yang membutuhkan web server dan browser untuk menjalankannya seperti Chrome, Firefox atau Opera. Aplikasi Web dapat berjalan pada jaringan internet maupun intranet (Jaringan LAN), Data terpusat dan kemudahan dalam akses adalah ciri utama yang membuat Aplikasi Web lebih banyak diminati dan lebih mudah diimplementasikan pada berbagai bidang kehidupan. Terdapat beberapa jenis dari aplikasi web, seperti :

1. Web media sosial

Website juga dapat dimanfaatkan untuk sarana komunikasi dalam bentuk percakapan online yang dapat dilakukan oleh setiap orang secara cepat dan real-time. Atau, biasa disebut dengan media sosial. Contohnya adalah Facebook, Twitter, Instagram, dll.

2. Web berbasis sistem informasi

Website juga digunakan untuk sarana membantu aktivitas usaha dan pekerjaan manusia. Sehingga proses pekerjaan yang dilakukan dapat tersistem, terpusat, dan termonitoring dengan baik menggunakan aplikasi. Saat ini dikenal dengan sistem informasi. Sistem informasi sendiri memiliki beberapa jenis, disesuaikan dengan kebutuhan dari bidang kerja masing – masing. Contohnya adalah sistem informasi koperasi, SIAKAD (Sistem Informasi Akademik), Fleet Management System, Hospital Management, dan masih banyak lagi SI yang lain.

3. Web jual beli dan bisnis

Website dapat digunakan untuk sarana transaksi jual beli secara online. Saat ini disebut dengan e-commerce. Dengan menggunakan e-commerce segala kebutuhan anda terkait produk barang atau jasa dapat diproses hanya dengan menggunakan aplikasi web. Contoh aplikasi yang banyak digunakan di Indonesia adalah Tokopedia, Shopee, Bukalapak, dan platform e-

commerce lainnya. Anda dapat memilih berbagai produk mulai dari yang baru, bekas, harga murah hingga termahal dapat anda dapatkan melalui aplikasi.

4. Web pencarian

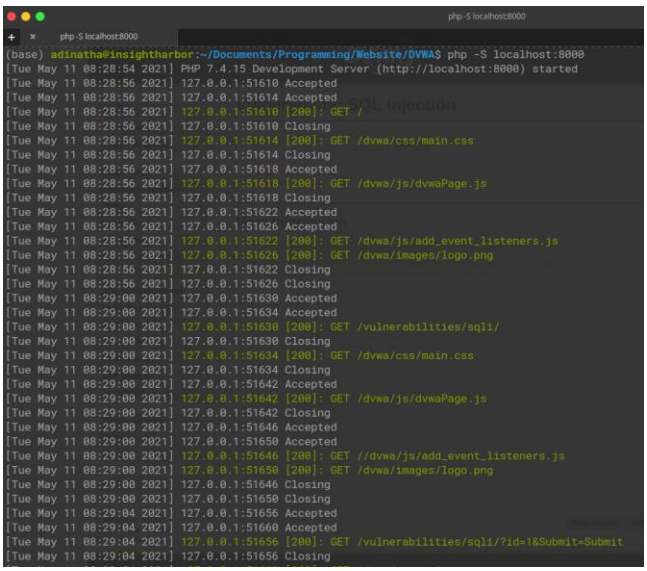
   Web pencarian biasa disebut dengan Search Engine. Tentunya, anda hampir setiap hari selalu mengakses yang namanya mesin pencari seperti Google, Yahoo, Youtube, dll. Mesin pencari dapat melakukan berbagai pencarian informasi secara cepat dan akurat.Web pencarian biasa disebut dengan Search Engine. Tentunya, anda hampir setiap hari selalu mengakses yang namanya mesin pencari seperti Google, Yahoo, Youtube, dll. Mesin pencari dapat melakukan berbagai pencarian informasi secara cepat dan akurat.

5. Web informasi dan berita

   Aplikasi berbasis website juga dapat menampilkan informasi dan berita teraktual dan terkini dari seluruh dunia. Contoh web berita di Indonesia adalah Detik.com, Kompas.com, Tribunnews, dll.

6. Aplikasi web server

   Definisi dari aplikasi web server adalah sebuah perangkat aplikasi, dimana anda dapat menerima request (permintaan) dan juga bisa mengirim respon atau tanggapan dalam protokol HTTP (Hypertext Transfer Protocol). Di dalam proses implementasinya, tentu saja sudah terprogram dengan bantuan bahasa pemrograman server-side atau lebih dikenal dengan istilah back end. Untuk jenis aplikasi web server dikembangkan oleh user yang ingin membangun sebuah client / server pada sebuah website, khususnya di kalangan IT development. Contoh dari jenis ini adalah Apache HTTP Web Server, Nginx, XAMPP, Apache Tomcat, Lighttpd, LAMP, WAMP, MAMP, dan masih banyak contoh yang lain lagi.



Gambar 4 Contoh dari *server log*

Pengguna dapat melakukan *request* kepada server dengan mengirimkan *method* berupa *GET* ataupun *POST*. Method *GET* digunakan untuk mendapatkan informasi, sedangkan method *POST* digunakan untuk mengirim informasi, selain itu juga terdapat beberapa method lain seperti *PUT* dan *DELETE*. Method *PUT* digunakan untuk melakukan *update* kepada data dan method *DELETE* digunakan untuk menghapus data. Semua *request* yang dilakukan oleh pengguna ke *server* akan tercatat pada *server log*.

D. SQL Injection

SQL Injection merupakan sebuah kerentanan pada aplikasi website 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 Injection*, yaitu sebuah bahasa yang digunakan untuk mengakses, menyimpan, maupun mengubah data yang memiliki basis relasional.

Penyerang dapat melakukan sernagan *SQL Injection* ini karena kurangnya penanganan terhadap karakter apostrof / petik satu (‘) dan juga tanda hubung (-). Sehingga hal ini menyebabkan program akan mengeksekusi *query* secara langsung pada *database*.

SQL Injection bekerja dengan cara mengeksekusi *query* dengan melakukan injeksi, misalkan terdapat sebuah situs login *admin* dimana pengguna diminta untuk mengisi *username* dan juga *password*, yang kemudian *SQL* akan mengecheck apakah username dan password tersebut terdapat pada tabel yang dinamakan ‘admin’

```
SELECT * FROM admin WHERE
username = input_username AND
password = input_password
```

pada halaman *login* ini dimasukkan *username* ‘user’ dan *password* ‘user123’. Maka SQL akan menerima perintah masukan berupa

```
SELECT * FROM admin WHERE
username = 'user' AND
password = 'user123'
```

Jika terdapat *username* ‘user’ dan juga *password* ‘user123’ pada database maka *SQL* akan memberikan *response* bernilai *True*, sehingga pengguna dapat masuk ke dalam situs *admin* tersebut. Namun jika *username* dimasukkan dengan ‘ OR 1=1 — , dimana 1=1 akan selalu menghasilkan *True*.

```
SELECT * FROM admin WHERE
username = '' OR 1=1--' AND
password = ''
```

Ini juga akan mengembalikan nilai *True*, karena pada bagian *password* terdapat argument *OR*. Dua tanda hubung (--)

merupakan cara untuk membuat membuat komentar pada *SQL* sehingga statement selanjutnya tidak akan dibaca. Maka perintah yang dijalankan adalah

```
SELECT * FROM admin WHERE
username = '' OR 1=1
```

Sehingga penyerang dapat masuk kedalam situs tanpa perlu mengetahui *username* maupun *password*.

Dampak dari *SQL Injection* dapat membuat penyerang untuk melihat, mengubah, menghapus, dan menambah data pada database. Selain itu penyerang juga mampu melakukan *bypass login*.

III. Pengujian dan Analisis

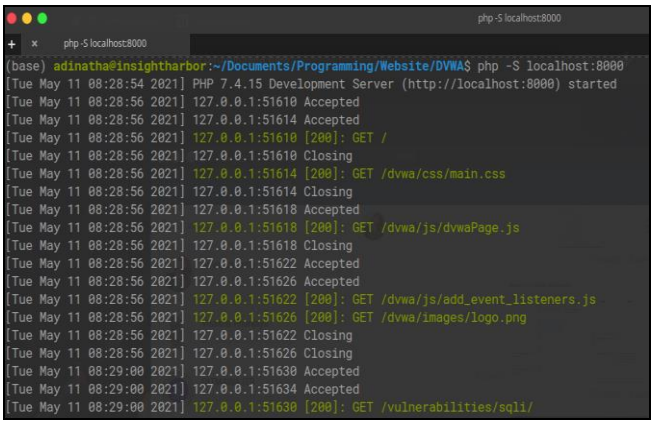
Pengujian yang dilakukan akan menggunakan program yang telah dibuat oleh penulis dan juga *log server* dari serangan yang telah disimulasikan dalam komputer lokal. Tujuan dari pengujian ini untuk mencari indikasi serangan *SQL Injection* dan juga membandingkan waktu eksekusi terhadap *Knuth Morris Pratt Algorithm*, *Boyer Moore Algorithm*, dan *Brute Force Algorithm*.

A. Program Uji

Program yang dibuat oleh penulis menerapkan algoritma sebagai berikut :

1. Menerima *file input* berupa *file* teks dari *server log dump*
2. Mencari kata *pattern* dari serangan *SQL Injection* dengan menggunakan *Knuth Morris Pratt Algorithm*, *Boyer Moore Algorithm*, dan *Brute Force Algorithm*.

   Pattern yang dicari adalah karakter apostrof (‘) atau encodingnya (‘%27’)
3. Menyimpan posisi *line* dari *log* jika terindikasi adanya serangan *SQL Injection*.
4. Menampilkan hasil pencarian berupa posisi *line*, *log server* pada *line* yang terindikasi, waktu program.



Gambar 5 *Server Log* yang digunakan

Berikut merupakan hasil dari program yang telah dibuat

1. **Algoritma *Brute Force***

```
Serangan terdeteksi pada line : [49, 52,
67, 85, 88, 91, 150]
[Tue May 11 08:29:15 2021] 127.0.0.1:5168
6 [200]: GET /vulnerabilities/sqli/?id=1%
27%20OR%201=1&Submit=Submit

[Tue May 11 08:29:31 2021] 127.0.0.1:5169
0 [200]: GET /vulnerabilities/sqli/?id=1%
27%20OR%20%271%27=%271&Submit=Submit

[Tue May 11 08:29:56 2021] 127.0.0.1:5171
8 [200]: GET /vulnerabilities/sqli/?id=1%
27%20SELECT%20@@version&Submit=Submit

[Tue May 11 08:30:39 2021] 127.0.0.1:5175
0 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users&Submit=Sub
mit

[Tue May 11 08:31:04 2021] 127.0.0.1:5175
4 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users--
&Submit=Submit

[Tue May 11 08:31:22 2021] 127.0.0.1:5176
2 [200]: GET /vulnerabilities/sqli/?id=3%
27%27%20SELECT%20*%20FROM%20users&Submit=
Submit

[Tue May 11 08:30:39 2021] 127.0.0.1:5175
0 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users&Submit=Sub
mit

Waktu yang dibutuhkan : 0.004621060999994
597
```

2. **Algoritma Knuth Morris Pratt**

```
Serangan terdeteksi pada line : [49, 52,
67, 85, 88, 91, 150]
[Tue May 11 08:29:15 2021] 127.0.0.1:5168
6 [200]: GET /vulnerabilities/sqli/?id=1%
27%20OR%201=1&Submit=Submit
```

```
[Tue May 11 08:29:31 2021] 127.0.0.1:5169
0 [200]: GET /vulnerabilities/sqli/?id=1%
27%20OR%20%271%27=%271&Submit=Submit

[Tue May 11 08:29:56 2021] 127.0.0.1:5171
8 [200]: GET /vulnerabilities/sqli/?id=1%
27%20SELECT%20@@version&Submit=Submit

[Tue May 11 08:30:39 2021] 127.0.0.1:5175
0 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users&Submit=Sub
mit

[Tue May 11 08:31:04 2021] 127.0.0.1:5175
4 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users--
&Submit=Submit

[Tue May 11 08:31:22 2021] 127.0.0.1:5176
2 [200]: GET /vulnerabilities/sqli/?id=3%
27%27%20SELECT%20*%20FROM%20users&Submit=
Submit

[Tue May 11 08:30:39 2021] 127.0.0.1:5175
0 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users&Submit=Sub
mit

Waktu yang dibutuhkan : 0.003203784999996
6284
```

3. **Algoritma Boyer Moore**

```
Serangan terdeteksi pada line : [49, 52,
67, 85, 88, 91, 150]
[Tue May 11 08:29:15 2021] 127.0.0.1:5168
6 [200]: GET /vulnerabilities/sqli/?id=1%
27%20OR%201=1&Submit=Submit

[Tue May 11 08:29:31 2021] 127.0.0.1:5169
0 [200]: GET /vulnerabilities/sqli/?id=1%
27%20OR%20%271%27=%271&Submit=Submit
```

```
[Tue May 11 08:29:56 2021] 127.0.0.1:5171
8 [200]: GET /vulnerabilities/sqli/?id=1%
27%20SELECT%20@@version&Submit=Submit

[Tue May 11 08:30:39 2021] 127.0.0.1:5175
0 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users&Submit=Sub
mit

[Tue May 11 08:31:04 2021] 127.0.0.1:5175
4 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users--
&Submit=Submit

[Tue May 11 08:31:22 2021] 127.0.0.1:5176
2 [200]: GET /vulnerabilities/sqli/?id=3%
27%27%20SELECT%20*%20FROM%20users&Submit=
Submit

[Tue May 11 08:30:39 2021] 127.0.0.1:5175
0 [200]: GET /vulnerabilities/sqli/?id=3%
27%20SELECT%20*%20FROM%20users&Submit=Sub
mit

Waktu yang dibutuhkan : 0.03637054799999362
```

B. Analisis

Analisis ini dilakukan dengan membandingkan waktu eksekusi antara 3 algoritma yaitu algoritma *Brute Force Algorithm*, *Knuth Morris Pratt Algorithm*, dan *Boyer Moore Algorithm* dalam mencari indikasi serangan *SQL Injection* pada *log server* yang telah disimpan dalam format txt. Pengujian ini dilakukan dengan menggunakan laptop penulis dengan spesifikasi sebagai berikut:

- Operating System : Linux Ubuntu 16.04.
- Processor : Intel® i5-8300H CPU @ 2.30Ghz
- Memory : 8192 MB RAM

Tabel 1 Hasil pengamatan waktu yang dibutuhkan oleh setiap algoritma dalam satuan detik

No	*Brute Force*	*Knuth Morris Pratt*	*Boyer Moore*
1	0.004621061	0.003203785	0.003637055
2	0.006773431	0.003052565	0.002494375
3	0.004447816	0.005682694	0.004295751

4	0.004435282	0.003029142	0.002643424
5	0.004183676	0.003907450	0.003301665
Rata -rata	0.004892253	0.003775127	0.003274454

Waktu rata rata yang dibutuhkan untuk menemukan seluruh indikasi serangan *SQL Injection* untuk setiap algoritmanya adalah 0.004892253 untuk algoritma *Brute Force*, 0.003775127 untuk algoritma *Knuth Morris Pratt*, dan 0.003274454 untuk algoritma *Boyer Moore*.

Dari tabel di atas diketahuin bahwa algoritma *Boyer Moore* dapat mendeteksi serangan *SQL Injection* lebih cepat dibadingkan algoritma *Brute Force* ataupun algoritma *Knuth Morris Pratt*. Hal ini dikarenakan struktur teks dari *server log* itu tersendiri, dimana pada bagian depan *server log* menyimpan informasi tanggal dan waktu kemudian dilanjutkan dengan *request method* yang diikuti dengan *link*. Serangan *SQL Injection* dapat dideteksi dari *link* yang telah di-*request* oleh pelaku. Dalam hal ini, *link* tersebut terletak dibagian belakang. Oleh karena itu algoritma *Boyer Moore* lebih tepat untuk digunakan karena melakukan pencarian dari belakang, pencarian yang dilakukan oleh *Boyer Moore Algorithm* menghasilkan waktu yang lebih cepat dibadingkan algoritma *Brute Force* dan algoritma *Knuth Morris Pratt*.

IV. Kesimpulan

Serangan *SQL Injection* dapat terjadi karena kurangnya penanganan terhadap karakter apostrof (‘) dan juga dikarenakan eksekusi langsung *input user* tanpa adanya filter, sehingga memungkinkan penyerang untuk mengeksekusi *query* yang seharusnya tidak dapat dieksekusi. Dari karakter apostrof tersebut dapat dikenali pola serangan yang terjadi sehingga pola serangan tersebut dapat dijadikan *pattern* dan dideteksi dengan algoritma *string matching*. Ketiga algoritma *string matching* yang digunakan yaitu algoritma *Brute Force*, algoritma *Knuth Morris Pratt*, algoritma *Boyer Moore* diketahui dari bahwa algoritma *Boyer Moore* dapat mendeteksi *pattern* serangan *SQL Injection* lebih cepat dibandingkan algoritma *Brute Force* ataupun algoritma *Knuth Morris Pratt*. Hal ini dapat terjadi karena algoritma *Boyer Moore* melakukan pemeriksaan *pattern* dari belakang, yang dimana indikasi serangan *SQL Injection* juga terletak pada bagian akhir teks.

Video Link Youtube

Berikut penulis melampirkan pranala untuk video yang menjelaskan topik mengenai serangan *SQL Injection* dan juga pendeteksian dengan membandingkan algoritma *Brute Force*, algoritma *Knuth Morris Pratt*, dan algoritma *Boyer Moore*. Video dapat diakses pada pranala https://youtu.be/rTvHrSemnDU

Ucapan Terima Kasih

Terima kasih kepada Tuhan YME, berkat rahmatNya makalah ini dapat diselesaikan dengan baik, terima kasih juga kepada dosen penulis, Dr. Ir. Rila Mandala, M.Eng. yang telah membimbing penulis dalam mata kuliah IF2211/Strategi Algoritma. Juga kepada teman baik saya Visakha yang telah memberikan banyak dukungan dan bantuan. Tidak lupa kepada warga HMIF yang telah banyak menemani perjalanan selama ini.

Referensi

[1] *https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf – diakses pada 10 Mei 2021.*

[2] *https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf – diakses pada 10 Mei 2021.*

[3] https://www.sekawanmedia.co.id/aplikasi-berbasis-web/ – *diakses pada 10 Mei 2021.*

[4] https://ariyaadinatha.medium.com/mengenal-sql-injection-53043e8dd8ff – *diakses pada 10 Mei 2021.*

[5] *https://glints.com/id/lowongan/sql-adalah/#.YJnCPagzaCg – diakses pada 10 Mei 2021.*

[6] https://www.dewaweb.com/blog/sql-pengertian-fungsi-beserta-perintah-dasarnya/ – *diakses pada 10 Mei 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.

Bandung, 11 Mei 2021

Ariya Adinatha - 13519048