Analisis dan Perbandingan Pencegahan SQL Injection pada Framework CodeIgniter dengan Escaping Query dan Framework Laravel dengan Eloquent ORM

Aedhelio Pratama, Parman Sukarno, S.T., M.Sc., Ph.D, Aulia Arif Wardana, S.Kom., M.T.

Fakultas Informatika, Univeritas Telkom, Bandung

aedhelio@students.telkomuniversity.ac.id, psukarno@telkomuniversity.ac.id, aulwardana@telkomuniversity.ac.id

Abstrak

***SQL Injection* adalah suatu serangan yang dilakukan dengan cara menyisipkan suatu perintah *query SQL* ke dalam bagian *input* agar dapat mengakses *database* yang terdapat pada aplikasi *web*. *SQL Injection* sering terjadi karena tidak adanya *filtering* yang dilakukan pada saat data *input* masuk ke dalam *database*. Oleh karena itu, diperlukan sebuah pencegahan agar *input* yang masuk dapat diseleksi terlebih dahulu sebelum masuk ke dalam *database*. Pencegahan tersebut dapat diperoleh dari fungsi yang terdapat pada bahasa pemrograman serta *framework* dari bahasa pemrograman tersebut.**

Pada penelitian ini, dilakukan analisis dan perbandingan akurasi pencegahan serangan *SQL injection* pada *framework* CodeIgniter dan *framework* Laravel. Pada *framework* CodeIgniter digunakan fungsi *escaping query* untuk mencegah serangan *SQL injection*. Sedangkan, pada *framework* Laravel digunakan fungsi *eloquent ORM* untuk mencegah serangan *SQL injection*.

Berdasarkan hasil pengujian dan analisa diperoleh bahwa serangan *SQL injection* pada *framework* CodeIgniter dan *framework* Laravel dapat dicegah dengan sama baik menggunakan fungsi *escaping query* pada *framework* CodeIgniter dan fungsi *eloquent ORM* pada *framework* Laravel. Dari hasil pengujian diperoleh akurasi 100% dari 293 serangan *SQL injection.*

Kata kunci : *SQL*, *SQL Injection*, CodeIgniter, Laravel, Akurasi

Abstract

SQL Injection is an attack carried out by inserting a SQL query command into the input section in order to access the database contained in the web application. SQL Injection often occurs because there is no filtering done when input data enters the database. Therefore, we need a precaution so that incoming input can be selected before entering the database. Prevention can be obtained from the functions contained in the programming language and the framework of the programming language.

In this study, an analysis and comparison of the accuracy of preventing SQL injection attacks on the CodeIgniter and Laravel frameworks was carried out. In the CodeIgniter framework the escaping query function is used to prevent SQL injection attacks. Meanwhile, the Laravel framework uses the ORM eloquent function to prevent SQL injection attacks.

Based on the results of testing and analysis, it was found that SQL injection attacks on the CodeIgniter framework and the Laravel framework can be prevented equally by using the escaping query function on the CodeIgniter framework and the ORM eloquent function on the Laravel framework. From the test results obtained 100% accuracy of 293 SQL injection attacks.

Keywords: SQL, SQL Injection, CodeIgniter, Laravel, Accuracy

1. Pendahuluan

Latar Belakang

Structured Query Language (*SQL*) *injection* adalah sebuah teknik injeksi menggunakan kode yang digunakan untuk menyerang *database* pada aplikasi *web*. Pada *SQL injection* penyerang menggunakan perintah

SQL melalui parameter *input user* yang tidak dilakukan *filtering* ke dalam *query SQL* dan meneruskannya ke *database*[1]. Berdasarkan informasi dari OWASP (*Open Web Application Security Project*) sebuah *web* yang meneliti khusus tentang masalah keamanan sistem, *SQL injection* masih menjadi peringkat pertama dalam masalah keamanan[2].

SQL injection terjadi pada suatu aplikasi *web* yang menggunakan *database*. *SQL injection* akan menyerang *database* suatu aplikasi *web* yang terdapat pada *back-end website*. Bahasa pemrograman yang paling banyak digunakan dalam membuat *back-end website* adalah *PHP*. Namun, saat ini sudah jarang aplikasi *web* yang menggunakan bahasa *PHP* asli atau *Native PH*P. Kebanyakan aplikasi *web* saat ini dibuat dengan menggunakan *framework PHP*.

Codeigniter dan Laravel adalah contoh *framework* yang banyak digunakan saat ini. Berdasarkan informasi tentang *framework* yang paling banyak digunakan pada tahun 2019 *framework* Laravel menempati peringkat pertama dan *framework* CodeIgniter menempati peringkat kedua[3]. Dengan menggunakan *framework programer* akan menjadi lebih dipermudah dengan adanya kelas serta fungsi yang dapat membantu *programer*. Kelebihan yang dapat kita peroleh dengan menggunakan *framework* Codeigniter dan *framework* Laravel yaitu : mudah digunakan, kemudahan instalasi, terdapat *library* validasi, dokumentasi lengkap, mendukung *PHP* 4 dan *PHP* 5, serta menyediakan banyak fungsi seperti fungsi enkripsi, *session, cookies, XSS filtering,* dan lain - lain[4,5]. Selain fungsi – fungsi tersebut, *framework* CodeIgniter dan *framework* Laravel memiliki fungsi pendukung terhadap pencegahan serangan *SQL Injection*. Berdasarkan popularitas *framework* CodeIgniter dan *framework* Laravel dengan didukung beberapa kelebihan menjadi alasan dilakukan penelitian ini.

Pada penelitian sebelumnya telah dilakukan analisis penanganan serangan *SQL injection* pada *framework* Codeigniter dan tanpa *framework* (*PHP*). Dari penelitian tersebut diperoleh hasil penanganan serangan *SQL injection* pada *framework* Codeigniter dan tanpa *framework* (*PHP*)[6]. Pada tugas akhir ini dilakukan analisis dan perbandingan akurasi pencegahan serangan *SQL injection* pada *database* MySQL dengan *framework* CodeIgniter dan *framework* Laravel sehingga dapat dilihat kemampuan *framework* CodeIgniter dan *framework* Laravel dalam mencegah serangan *SQL injection.*

Topik dan Batasannya

Permasalahan yang akan dibahas dalam tugas akhir ini adalah belum adanya solusi yang mengukur akurasi pencegahan serangan *SQL injection* dari sistem pencegahan terhadap serangan *SQL injection* pada *framework* CodeIgniter dan *framework* Laravel.

Tujuan

Tujuan tugas akhir ini dibuat adalah untuk mengukur akurasi pencegahan serangan *SQL injection* dari sistem pencegahan terhadap serangan SQL injection pada *framework* CodeIgniter dan *framework* Laravel.

Organisasi Tulisan

Pada bab dua menjelaskan studi terkait yang berisi teori yang mendukung penelitian. Pada bab tiga menjelaskan rancangan sistem yang dibangun. Pada bab empat menjelaskan pengujian dan analisis dari penelitian yang dilakukan. Pada bab lima menjelaskan kesimpulan dari penelitian dan saran untuk penelitian selanjutnya.

2. Studi Terkait

2.1. Penelitian Terkait SQL Injection

Pada tugas akhir sebelumnya, dilakukan penelitian mengenai analisis penanganan *SQL injection* pada *framework* CodeIgniter dan tanpa *framework* (*PHP*)[6]. Penanganan serangan *SQL injection* pada *framework* CodeIgniter menggunakan *active class record*. Sementara, penanganan serangan *SQL injection* pada tanpa *framework* (*PHP*) menggunakan *mysql_real_escape_query*. Dari hasil penelitian tersebut menunjukkan bahwa *framework* CodeIgniter dan tanpa *framework* (*PHP*) memiliki teknik penanganan terhadap serangan *SQL injection* yang berbeda. *Framework* CodeIgniter dan tanpa *framework* (*PHP*) dapat menangani serangan *SQL injection* dengan baik.

Pada paper "*Comparison of Procedural PHP with Codeigniter and Laravel Framework*", dilakukan penelitian mengenai analisis perbandingan *framework* CodeIgniter dan *framework* Laravel[7]. Pada penelitian tersebut dilakukan pengujian performansi dari *framework* CodeIgniter dan *framework* Laravel. Pengujian dilakukan dengan menghitung waktu proses dalam hal *create, read, update* dan *delete*. Dari hasil pengujian diperoleh bahwa *framework* Laravel memiliki waktu proses yang jauh lebih cepat dari *framework* CodeIgniter. Hal ini menunjukkan bahwa *framework* Laravel memiliki performansi yang lebih baik dari *framework* CodeIgniter dalam hal *create, read, update* dan *delete*.

2.2. *Framework* CodeIgniter[8]

CodeIgniter adalah sebuah *framework PHP* yang bersifat *open source* yang digunakan dalam membuat aplikasi *web php* dinamis. CodeIgniter menggunakan model *MVC* (Model, *View, Controller*) untuk membangun website dinamis dengan menggunakan *PHP* sehingga dapat mempermudah programmer dalam membuat sebuah aplikasi *web*.

2.3. Escaping Query[9]

Framework CodeIgniter memiliki beberapa teknik dalam mencegah serangan *SQL injection* yaitu : *escaping query*, *query binding*, dan *active record class*. Pada penelitian ini akan digunakan teknik *escaping query*. Teknik *escaping query* dipilih karena pada *user guide* CodeIgniter teknik *escaping query* menjadi teknik pertama yang disarankan oleh CodeIgniter untuk mencegah serangan *SQL injection*. Selain itu, sintaks *escaping query* tidak perlu mengubah semua *query* cukup dengan menambahkan beberapa kode.

Escaping query bekerja dengan cara membebaskan karakter *input* dari *query* sebelum *query* dieksekusi oleh *database*. *Escaping query* menentukan tipe data sehingga hanya data dengan tipe data *String* saja yang akan dieksekusi. *Escaping query* secara otomatis menambahkan tanda kutip tunggal pada awal dan akhir data.

Berikut adalah *source code framework* CodeIgniter dengan *Escaping Query* :

```
return $this->db->query('SELECT * FROM user WHERE nama = '.$this->db->escape($keyword).");
```

Fungsi *Escaping Query* dapat digunakan dengan cara menanbahkan kode $this->db->escape seperti pada contoh di atas. Dengan menggunakan *Escaping Query*, karakter input akan dieksekusi secara terpisah dari *query*. Sehingga, tidak akan menyebabkan terjadinya serangan SQL injection.

2.4. Framework Laravel[10]

Laravel adalah sebuah *framework PHP* yang dibuat dengan konsep *MVC* (*model*, *view*, *controller*). Laravel merupakan pengembangan *website* berbasis *MVC* yang menggunakan bahasa *PHP* sehingga dapat meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan.

2.5. Eloquent ORM[11]

Teknik pencegahan serangan *SQL injection* pada *framework* Laravel adalah *eloquent ORM*. *Eloquent ORM* merupakan suatu *query builder* yang terdapat pada *framework* Laravel. *Eloquent ORM* menggunakan *PDO parameter binding* untuk menghindari serangan *SQL injection*. *PDO parameter binding* memastikan bahwa pengguna tidak dapat mengirim suatu *input* yang dapat mengubah maskud dari *query*. *Eloquent ORM* bekerja dengan cara melakukan *filtering* dengan mengubah variabel *input user* menjadi variabel *String*. Dengan menggunakan *eloquent ORM* data *input* akan dieksekusi menggunakan parameter dan tidak digabung ke dalam *query*.

Berikut adalah *source code framework* Laravel dengan *Eloquent ORM* :

```
$pegawai = Pegawai::where('pegawai_nama', $cari)->get();
```

Fungsi *Eloquent ORM* dapat digunakan dengan menggunakan sintaks seperti pada contoh di atas. Selain itu, model perlu didefinisikan agar *Eloquent ORM* dapat mengetahui variabel yang telah dibuat. Dengan menggunakan *Eloquent ORM*, karakter input akan dieksekusi menggunakan parameter. Sehingga, tidak akan menyebabkan terjadinya serangan *SQL injection*.

2.6. SQL Injection[12]

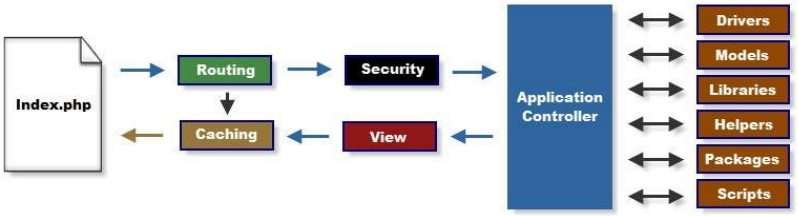
SQL injection adalah sebuah serangan dimana sebuah kode berbahaya telah tertanam pada suatu *string* yang kemudian diteruskan menuju *database* untuk diolah dan dieksekusi. *String* berbahaya tersebut menghasilkan hasil *query* yang memiliki informasi sensitif seperti kredensial akun atau data bisnis internal.

Penyebab *SQL injection* relatif sederhana dan mudah untuk dimengerti. Serangan *SQL injection* merupakan kode injeksi yang mengambil keuntungan dari kurangnya validasi *input user*. Serangan *SQL injection* memungkinkan penyerang untuk menghancurkan fungsionalitas atau kerahasiaan *database* aplikasi *web*.

3. Sistem yang Dibangun

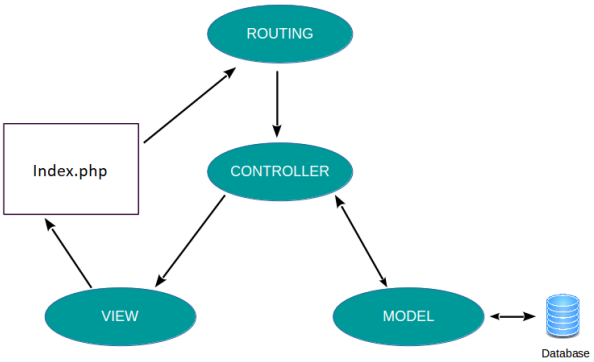
Pada tugas akhir ini, sistem yang dibuat merupakan bagian dari sebuah aplikasi berbasis *website*. Sistem ini merupakan tahap *filtering* atau *preprocessing* sebelum sebuah variabel *input* digabung ke dalam *query SQL*, sehingga diperoleh hasil *output* yang aman dari serangan *SQL injection*.

Berikut adalah gambaran umum sistem pada *framework* CodeIgniter dan *framework* Laravel.



Gambar 3.1 Gambaran Umum Sistem pada *Framework* CodeIgniter

Pada gambar 3.1, sistem pada CodeIgniter merupakan sebuah fungsi yang berada di dalam blok model yang digunakan untuk melakukan interaksi dengan *database*. Proses *filtering* variabel *input* terjadi pada bagian model dengan menggunakan *escaping query*.



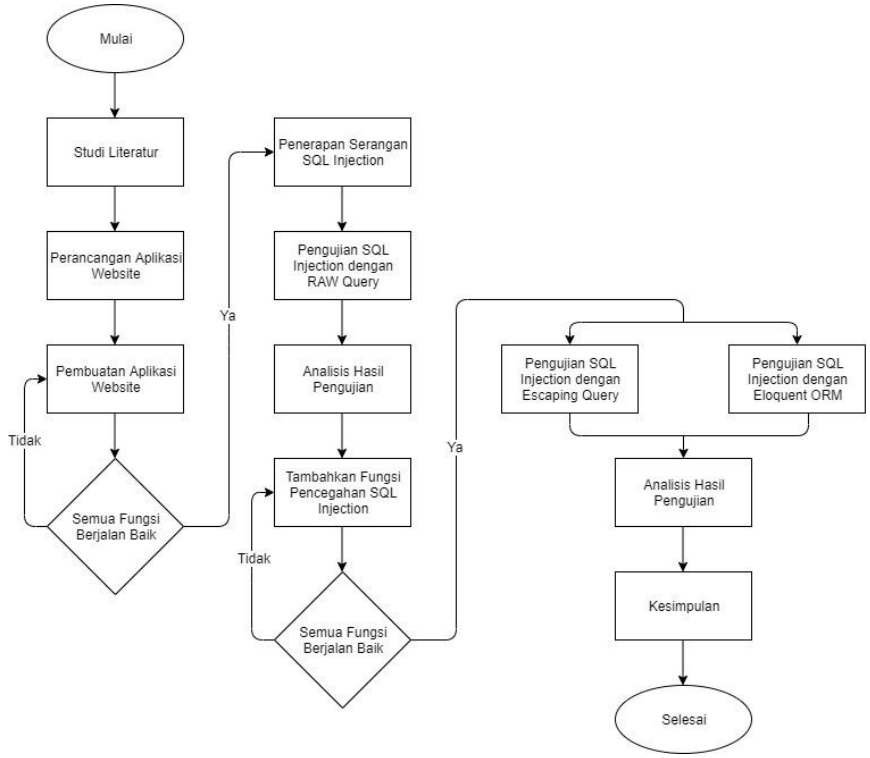
Gambar 3.2 Gambaran Umum Sistem pada *Framework* Laravel

Pada gambar 3.2, sistem pada Laravel merupakan sebuah fungsi yang berada di dalam blok model yang digunakan untuk melakukan interaksi dengan *database*. Proses *filtering* variabel *input* terjadi pada bagian model dengan menggunakan *eloquent ORM*.

Sistem akan diujicoba pada setiap aplikasi menggunakan *database* MySQL. *Database* MySQL digunakan karena MySQL merupakan salah satu *Relational Database Management System* (*RDBMS*) yang populer digunakan di dunia serta bersifat *open source* sehingga lebih mudah dalam penggunaan serta pengembangan. Selain itu, MySQL juga kompatibel dengan beberapa bahasa pemrograman, tidak memerlukan perangkat lunak maupun perangkat keras khusus dalam penggunaannya, serta memiliki tingkat *reliability* dan performansi yang tinggi. Dalam hal keamanan MySQL menyediakan fitur administrasi *user* yang dapat membatasi *privilege* pada *user*.

3.1. Alur Pengerjaan

Berikut adalah *flowchart* berupa langkah-langkah dalam pengerjaan tugas akhir.

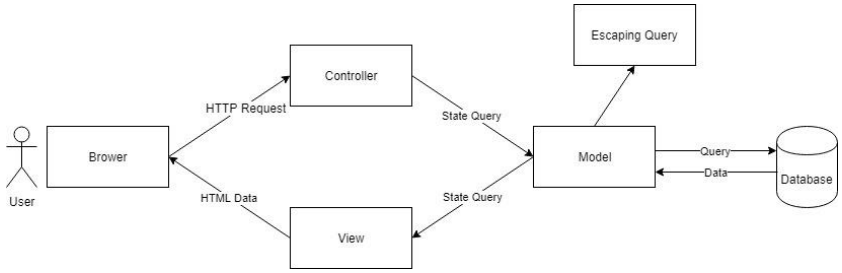


Gambar 3.3 *Flowchart* Pengerjaan

3.2. Pemodelan Pencegahan SQL Injection

Berikut adalah gambaran mengenai sistem pencegahan *SQL Injection* pada *framework* CodeIgniter dan *framework* Laravel.

3.2.1. *Framework* CodeIgniter



Gambar 3.4 Model Pencegahan *SQL Injection* pada *Framework* CodeIgniter

Berikut adalah proses pencegahan *SQL Injection* pada *framework* CodeIgniter dengan menggunakan fungsi *Escaping Query* pada blok model seperti pada Gambar 3.3.

1. *Controller*

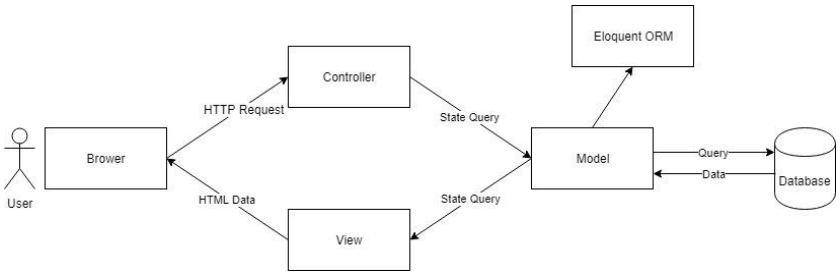
   Pada *framework* CodeIgniter *controller* berfungsi sebagai pengontrol semua alur yang bekerja. Ketika *user* melakukan *HTTP request controller* akan menangani hal tersebut. Jika dalam proses *request* terdapat variabel *input* yang berhubungan dengan *database*, maka *controller* akan memanggil blok model.

2. Model

   Pada *framework* CodeIgniter model berfungsi sebagai bagian yang berinteraksi dengan *database*. Pada bagian model akan digunakan fungsi *escaping query* yang berfungsi untuk melakukan pembebasan karakter yang memiliki *sintaks query* dari *query* yang akan menyebabkan terjadinya serangan *SQL*

Injection. Sehingga, ketika *user* memasukkan *sintaks query SQL* pada variabel *input* tidak akan dieksekusi karena telah dilakukan *filtering* oleh fungsi *Escaping Query*.

3.1.2 *Framework* Laravel



Gambar 3.5 Model Pencegahan *SQL Injection* pada *Framework* Laravel

Berikut adalah proses pencegahan *SQL Injection* pada *framework* Laravel dengan menggunakan *Eloquent ORM* pada blok model seperti pada Gambar 3.4.

1. *Controller*

   Pada *framework* Laravel *controller* berfungsi sebagai pengontrol semua alur yang bekerja. Ketika *user* melakukan *HTTP request controller* akan menangani hal tersebut. Jika dalam proses *request* terdapat variabel *input* yang berhubungan dengan *database*, maka *controller* akan memanggil blok model.

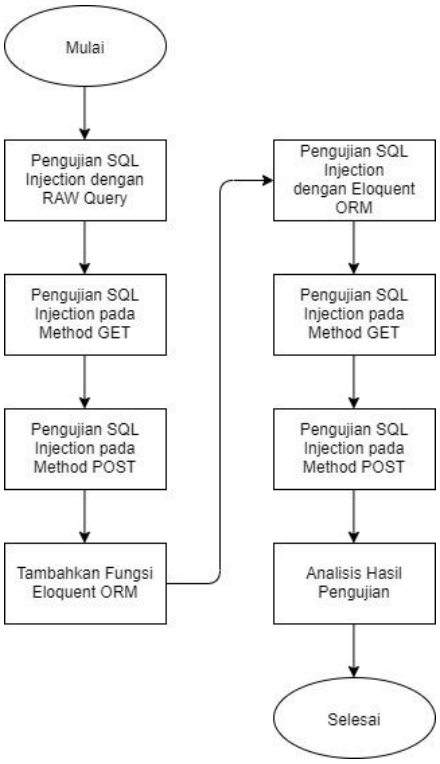
2. Model

   Pada *framework* Laravel model berfungsi sebagai bagian yang berinteraksi dengan *database*. Pada bagian model akan digunakan fungsi *Eloquent ORM* yang berfungsi untuk melakukan *filtering* dengan cara mengubah variabel *input user* menjadi variabel *String*. Sehingga, ketika *user* memasukkan *sintaks query SQL* pada variabel *input* tidak akan diekseskusi karena sistem membaca variabel *input* tersebut berupa *String*.

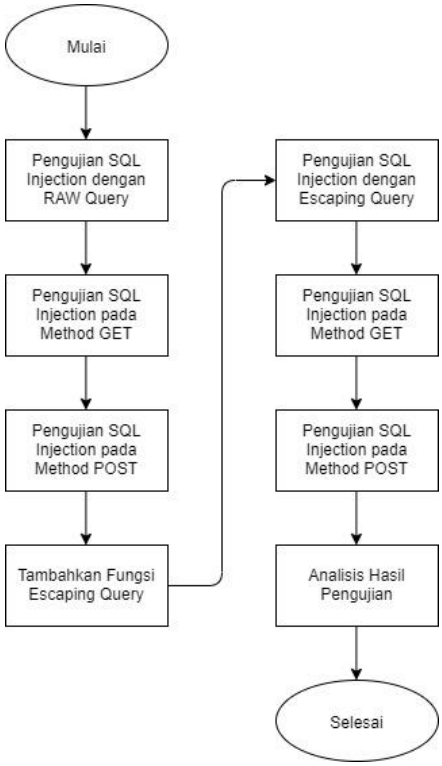
3.3. Tools SQL Injection[13]

Pada penelitian ini akan digunakan *tools* Burp Suite untuk melakukan serangan *SQL injection*. Burp Suite merupakan salah satu *tools* untuk melakukan serangan *SQL injection* yang populer dan banyak digunakan. Burp Suite adalah pencari kerentanan atau *vulnerability scanner* dengan *tool penetration testing*. Burp Suite dapat mencari target, menyusup sebuah *request* dan dapat mengulang *request* tersebut dengan nilai parameter yang dimodifikasi. Burp Suite juga bekerja sebagai *tool proxy* antara *server* dan *browser*. Beberapa fitur yang terdapat pada Burp Suite yaitu : *HTTP Proxy*, *Scanner*, *Intruder*, *Spider*, *Repeater*, *Decoder*, *Comparer*, *Extender*, dan *Sequencer*. Burp Suite dapat diunduh pada semua sistem operasi melalui *website* Burp Suite. Dalam penggunaannya Burp Suite memerlukan *wordlist* untuk melakukan *SQL Injection*. *Wordlist* yang digunakan pada penelitian ini adalah SQL.txt, MySQL.txt, MySQL_MSSQL.txt, mysql-injection-login-bypass.txt, mysql-read-local-files.txt, payloads-sql-blind-MySQL-INSERT.txt, payloads-sql-blind-MySQL-ORDER_BY.txt, dan payloads-sql-blind-MySQL-WHERE.txt. Semua *wordlist* tersebut akan digabung menjadi SQLi.txt. *Wordlist* tersbut diperoleh dari wfuzz dan fuzzdb pada github. *Wordlist* tersebut dipilih karena sering digunakan pada pengujian *SQL Injection* dan memiliki tingkat keberhasilan tinggi.

3.4. Skenario Pengujian



Gambar 3.6 Flowchart Skenario Pengujian Framework CodeIgniter



Gambar 3.7 Flowchart Skenario Pengujian Framework Laravel

Pengujian dilakukan menjadi 2 bagian yaitu pengujian pada aplikasi dengan *framework* CodeIgniter sebelum dan setelah terdapat pencegahan terhadap serangan *SQL injection* seperti pada gambar 3.4 dan pengujian pada aplikasi dengan *framework* Laravel sebelum dan setelah terdapat pencegahan terhadap serangan *SQL injection* seperti pada gambar 3.5. Pengujian akan dilakukan pada *method GET* dan *method POST* pada *framework* CodeIgniter dan *framework* Laravel.

Tipe serangan *SQL injection* yang digunakan pada penelitian ini adalah *in-band SQL injection* (*Classic SQL injection*). *In-band SQL injection* dipilih karena *in-band SQL injection* merupakan teknik serangan *SQL injection* yang paling umum digunakan dan memiliki kemungkinan berhasil lebih tinggi dibandingkan dengan teknik lainnya.

Pengujian dilakukan sebanyak 293 serangan *SQL injection* yang terdapat pada wordlist SQLi.txt. Akurasi akan didapat dari banyak serangan *SQL injection* yang dapat dicegah oleh *framework* CodeIgniter dan *framework* Laravel dibagi dengan banyak serangan yang dilakukan. Rumus akurasi adalah sebagai berikut :

$$\text{Akurasi} = \frac{Banyak\ serangan\ yang\ dapat\ dicegah}{Banyak\ serangan\ yang\ dilakukan} \text{ x } 100\% \quad (1)$$

Hal – hal yang akan dianalisis sebagai berikut :

1. Akurasi pencegahan serangan *SQL Injection* pada *framework* CodeIgniter dan *framework* Laravel sebelum terdapat pencegahan terhadap *SQL injection*
2. Akurasi pencegahan serangan *SQL Injection* pada *framework* CodeIgniter dan *framework* Laravel setelah terdapat pencegahan terhadap *SQL injection*
3. Perbandingan akurasi pencegahan serangan *SQL Injection* pada *framework* CodeIgniter dan *framework* Laravel

Vektor pengujian dibagi menjadi dua bagian, yaitu :

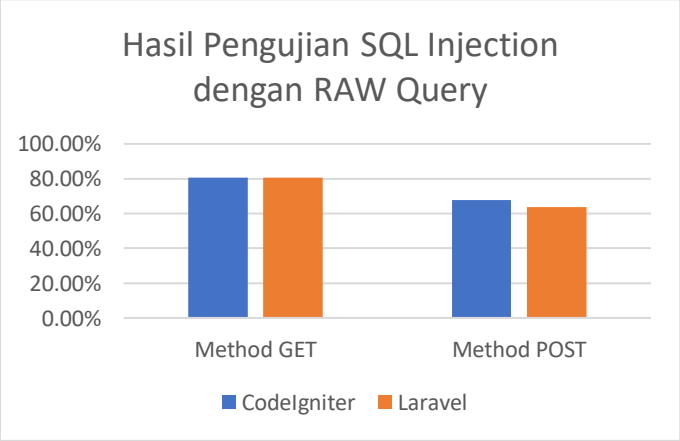
1. Pengujian melalui *form* pencarian data dengan menggunakan *method GET*
2. Pengujian melalui *form input* data dengan menggunakan *method POST*

4. Evaluasi

4.1. Hasil Pengujian

Pengujian dilakukan pada *framework* CodeIgniter dan *framework* Laravel menggunakan *database* MySQL. Skenario yang digunakan adalah sebelum dan setelah adanya fungsi pencegahan terhadap serangan *SQL injection*. Kemudian, hasil dari pengujian dianalisis dan dibandingkan dari sisi akurasi pencegahan serangan *SQL injection* antara *framework* CodeIgniter dan *framework* Laravel.

4.1.1. Hasil Pengujian SQL Injection dengan RAW Query



Gambar 4.1 Chart Hasil Pengujian *SQL Injection* dengan *RAW Query*

Gambar 4.1 merupakan akurasi hasil pengujian pencegahan serangan *SQL injection* dengan *RAW Query* pada *framework* CodeIgniter dan *framework* Laravel. Berikut detail penjelasan hasil pengujian untuk masing – masing vektor pengujian :

1. Pengujian pada *method GET*

   Hasil pengujian pencegahan serangan *SQL injection* pada *framework* CodeIgniter dengan *method GET* diperoleh akurasi sebesar 80,55% dari 293 serangan *SQL injection*. Sebanyak 57 serangan dapat

menimbulkan *error* pada saat melakukan *SQL injection*. Hal ini dapat dilihat pada status yaitu 500 atau *Internal Server Error*. Dari data hasil pengujian dapat dilihat bahwa karakter *double quote* (“) menyebabkan terjadinya *error* pada saat *query* dijalankan. Hal ini terjadi karena tidak adanya fungsi yang membebaskan karakter *double quote* sehingga ketika *query* dijalankan akan menyebabkan kesalahan *sintaks*. Selain itu, beberapa *wordlist* yang mengandung karakter *double quote* (“) juga menyebabkan *error* yang sama.

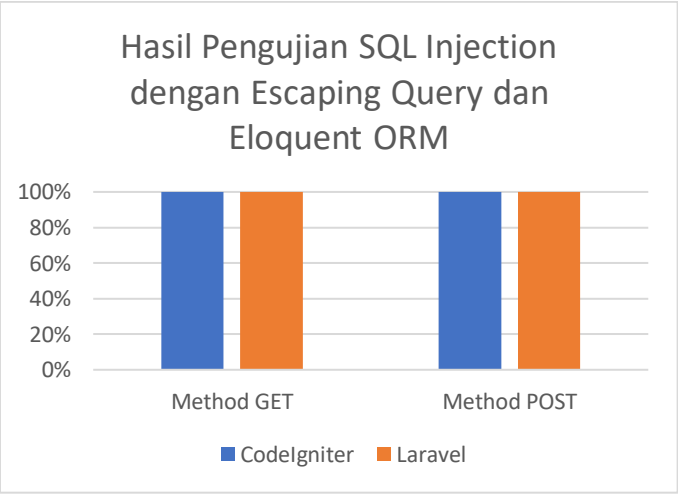
Sedangkan, hasil pengujian pencegahan serangan *SQL injection* pada *framework* Laravel dengan *method GET* diperoleh akurasi sebesar 80,55% dari 293 serangan *SQL injection*. Sebanyak 57 serangan dapat menimbulkan *error* pada saat melakukan *SQL injection*. Hal ini dapat dilihat pada status yaitu 500 atau *Internal Server Error*. Sama seperti pada *framework* CodeIgniter, dari data hasil pengujian dapat dilihat bahwa karakter *double quote* (“) menyebabkan terjadinya *error* pada saat *query* dijalankan. Hal ini terjadi karena tidak adanya fungsi yang membebaskan karakter *double quote* sehingga ketika *query* dijalankan akan menyebabkan kesalahan *sintaks*. Selain itu, beberapa *wordlist* yang mengandung karakter *double quote* (“) juga menyebabkan *error* yang sama.

2. Pengujian pada *method POST*

Hasil pengujian pencegahan serangan *SQL injection* pada *framework* CodeIgniter dengan *method POST* diperoleh akurasi sebesar 67,58% dari 293 serangan *SQL injection*. Sebanyak 95 serangan dapat menimbulkan *error* pada saat melakukan *SQL injection*. Hal ini dapat dilihat pada status yaitu 500 atau *Internal Server Error*. Dari data hasil pengujian dapat dilihat bahwa karakter *single quote* (‘) menyebabkan terjadinya *error* pada saat *query* dijalankan. Hal ini terjadi karena tidak adanya fungsi yang membebaskan karakter *single quote* sehingga ketika *query* dijalankan akan menyebabkan kesalahan *sintaks*. Selain itu, beberapa *wordlist* yang mengandung karakter *single quote* (‘) juga menyebabkan *error* yang sama.

Sedangkan, hasil pengujian pencegahan serangan *SQL injection* pada *framework* Laravel dengan *method POST* diperoleh akurasi sebesar 64,51% dari 293 serangan *SQL injection*. Sebanyak 104 serangan dapat menimbulkan *error* pada saat melakukan *SQL injection*. Sama seperti pada *framework* CodeIgniter, dari data hasil pengujian dapat dilihat bahwa karakter *single quote* (‘) menyebabkan terjadinya *error* pada saat *query* dijalankan. Hal ini terjadi karena tidak adanya fungsi yang membebaskan karakter *single quote* sehingga ketika *query* dijalankan akan menyebabkan kesalahan *sintaks*. Selain itu, beberapa *wordlist* yang mengandung karakter *single quote* (‘) juga menyebabkan *error* yang sama.

4.1.2. Hasil Pengujian SQL Injection dengan Escaping Query dan Eloquent ORM



Gambar 4.2 Chart Hasil Pengujian *SQL Injection* dengan *Escaping Query* dan *Eloquent ORM*

Gambar 4.2 merupakan akurasi hasil pengujian pencegahan serangan *SQL injection* dengan *Escaping Query* pada *framework* CodeIgniter dan *Eloquent ORM* pada *framework* Laravel. Berikut detail penjelasan hasil pengujian untuk masing – masing vektor pengujian :

1. Pengujian pada *method GET*

Hasil pengujian pencegahan serangan *SQL injection* pada *framework* CodeIgniter dengan *method GET* diperoleh akurasi sebesar 100% dari 293 serangan *SQL injection*. Tidak ada serangan yang dapat

menimbulkan *error* pada saat melakukan *SQL injection*. Fungsi *Escaping Query* pada *framework* CodeIgniter dapat bekerja dengan baik. Dengan menggunakan fungsi *Escaping Query* karakter *double quote* (“) akan secara otomatis dibebaskan dari *query*, sehingga *query* tersebut aman untuk dieksekusi dan terhindar dari serangan *SQL injection*.

Sedangkan, hasil pengujian pencegahan serangan *SQL injection* pada *framework* Laravel dengan *method GET* diperoleh akurasi sebesar 100% dari 293 serangan *SQL injection*. Sama seperti pada *framework* CodeIgniter, tidak ada serangan yang dapat menimbulkan *error* pada saat melakukan *SQL injection*. Fungsi *Eloquent ORM* pada *framework* Laravel dapat bekerja dengan baik. Dengan menggunakan fungsi *Eloquent ORM* karakter *double quote* (“) akan dieksekusi menggunakan parameter, sehingga *query* tersebut aman untuk dieksekusi dan terhindar dari serangan *SQL injection*.

2. Pengujian pada *method POST*

Hasil pengujian pencegahan serangan *SQL injection* pada *framework* CodeIgniter dengan *method POST* diperoleh akurasi sebesar 100% dari 293 serangan *SQL injection*. Tidak ada serangan yang dapat menimbulkan *error* pada saat melakukan *SQL injection*. Fungsi *Escaping Query* pada *framework* CodeIgniter dapat bekerja dengan baik. Dengan menggunakan fungsi *Escaping Query* karakter *single quote* (‘) akan secara otomatis dibebaskan dari *query*, sehingga *query* tersebut aman untuk dieksekusi dan terhindar dari serangan *SQL injection*.

Sedangkan, hasil pengujian pencegahan serangan *SQL injection* pada *framework* Laravel dengan *method POST* diperoleh akurasi sebesar 100% dari 293 serangan *SQL injection*. Sama seperti pada *framework* CodeIgniter, tidak ada serangan yang dapat menimbulkan *error* pada saat melakukan *SQL injection*. Fungsi *Eloquent ORM* pada *framework* Laravel dapat bekerja dengan baik. Dengan menggunakan fungsi *Eloquent ORM* karakter *double quote* (“) akan dieksekusi menggunakan parameter, sehingga *query* tersebut aman untuk dieksekusi dan terhindar dari serangan *SQL injection*.

4.2. Analisis Hasil Pengujian

Berdasarkan akurasi hasil pengujian serangan *SQL injection* pada *framework* CodeIgniter dan *framework* Laravel dengan menggunakan *database* MySQL dapat diperoleh kemampuan *framework* CodeIgniter dan *framework* Laravel dalam mencegah serangan *SQL injection*. Dengan menggunakan *RAW Query* pada *framework* CodeIgniter dan *framework* Laravel, serangan *SQL injection* masih dapat terjadi pada *method GET* dan *method POST*. Dari akurasi hasil pengujian dapat terlihat bahwa peluang terjadinya serangan *SQL injection* pada *method POST* lebih besar daripada *method GET*. Jika tidak terdapat suatu fungsi yang dapat membebaskan karakter *input* dari *query*, serangan *SQL injection* masih mungkin terjadi.

Dengan menggunakan fungsi *Escaping Query* pada *framework* CodeIgniter dan fungsi *Eloquent ORM* pada *framework* Laravel, serangan *SQL injection* tidak dapat terjadi baik pada *method GET* maupun *method POST*. Fungsi *Escaping Query* pada *framework* CodeIgniter dapat mencegah terjadinya serangan *SQL injection* dengan membebaskan karakter *input* sebelum dieksekusi. Fungsi *Eloquent ORM* pada *framework* Laravel juga dapat mencegah terjadinya serangan *SQL injection* dengan mengeksekusi karakter *input* menggunakan parameter. Akurasi yang didapat dengan menggunakan fungsi *Escaping Query* pada *framework* CodeIgniter dan fungsi *Eloquent ORM* pada *framework* Laravel sama baik yaitu 100%. Dengan demikian dapat disimpulkan bahwa *framework* CodeIgniter dengan menggunakan fungsi *Escaping Query* dan *framework* Laravel dengan menggunakan fungsi *Eloquent ORM* dapat mencegah serangan *SQL injection* dengan sama baik.

5. Kesimpulan dan Saran

5.1. Kesimpulan

Berdasarkan hasil pengujian serta analisis pencegahan serangan *SQL Injection* pada *framework* CodeIgniter dan *framework* Laravel sebelum dan setelah dilakukan pencegahan yaitu :

1. Pada *framework* CodeIgniter, penggunaan fungsi *Escaping Query* dapat mencegah serangan *SQL injection* dengan baik yang dapat dilihat pada akurasi hasil pengujian setelah terdapat pencegahan serangan *SQL injetion* yaitu 100% dari 293 serangan *SQL injection*.
2. Penggunaan fungsi *Escaping Query* dapat mencegah serangan *SQL injection* dengan cara membebaskan karakter *input* dengan *query* sehingga akan menghasilkan *query* yang aman pada saat dieksekusi.

3. Pada *framework* Laravel, penggunaan fungsi *Eloquent ORM* dapat mencegah serangan *SQL injection* dengan baik yang dapat dilihat pada akurasi hasil pengujian setelah terdapat pencegahan serangan *SQL injetion* yaitu 100% dari 293 serangan *SQL injection.*
4. Penggunaan fungsi *Eloquent ORM* dapat mencegah serangan *SQL injection* dengan cara mengeksekusi karakter *input* menggunakan parameter sehingga akan menghasilkan *query* yang aman pada saat dieksekusi.
5. *Framework* CodeIgniter dengan fungsi *Escaping Query* dan *framework* Laravel dengan fungsi *Eloquent ORM* dapat mencegah serangan *SQL injection* sama baik dengan akurasi 100%.

5.2. Saran

Analisis pencegahan *vulnerability* pada *framework* CodeIgniter dan *framework* Laravel dapat dilihat dari sisi keamanan lain seperti *XSS* (*Cross Site Scripting*). Dapat juga dilakukan pengujian pencegahan serangan *SQL Injection* pada *framework* lain seperti Symfony.

Daftar Pustaka

[1] Sadeghian, Amirmohammad, Mazdak Zamani, and Azizah Abd Manaf. "A taxonomy of SQL injection detection and prevention techniques." *2013 International Conference on Informatics and Creative Multimedia.* IEEE, 2013.

[2] https://www.owasp.org/index.php/Top_10-2017_Top_10 diakses pada tanggal 30 Desember 2019.

[3] https://coderseye.com/best-php-frameworks-for-web-developers/ diakses pada tanggal 30 Desember 2019.

[4] Yicheng, Li. "Development of a blog system using Codeigniter framework." *Finland: Oulu University of Applied Sciences.* 2011.

[5] He, Ren Yu. "Design and implementation of web based on Laravel framework." *2014 International Conference on Computer Science and Electronic Technology (ICCSET 2014)*. Atlantis Press, 2015.

[6] Efendi, Muhammad Rizal. "Analisis Penanganan SQL Injection pada Basis Data MySQL dengan Framework Code Igniter dan PHP." *Prosiding SENIATI* Book-2. 2016.

[7] Das, Ripunjit, and Lakshmi Prasad Saikia. "Comparison of Procedural PHP with Codeigniter and Laravel Framework." *International Journal of Current Trends in Engineering & Research* 2.6. 2016.

[8] Wardana, S. Hut, and M. Si. *Menjadi Master PHP dengan Framework Codeigniter*. Elex Media Komputindo, 2010.

[9] https://www.codeigniter.com/userguide3/database/queries.html diakses pada tanggal 2 April 2019.

[10] Enterprise, Jubilee. *Mengenal PHP Menggunakan Framework Laravel*. Elex Media Komputindo, 2016.

[11] https://laravel.com/docs/5.8/eloquent diakses pada tanggal 7 April 2019.

[12] Clarke-Salt, Justin. *SQL injection attacks and defense*. Elsevier, 2009.

[13] https://portswigger.net/burp diakses pada tanggal 30 Desember 2019.

[14] Sharma, Chandershekhar, and S. C. Jain. "Analysis and classification of SQL injection vulnerabilities and attacks on web application." *2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014)*. IEEE, 2014.

[15] Halfond William G., Jeremy Viegas, and Alessandro Orso. "A classification of SQL-injection attacks and countermeasures." *Proceedings of the IEEE International Symposium on Secure Software Engineering*. Vol. 1. IEEE, 2006.

[16] Gilmore, W. Jason. *Beginning PHP and MySQL: from novice to professional*. Apress, 2010.

Lampiran

Hasil pengujian serangan *SQL injection* pada *framework* CodeIgniter dengan menggunakan RAW Query :

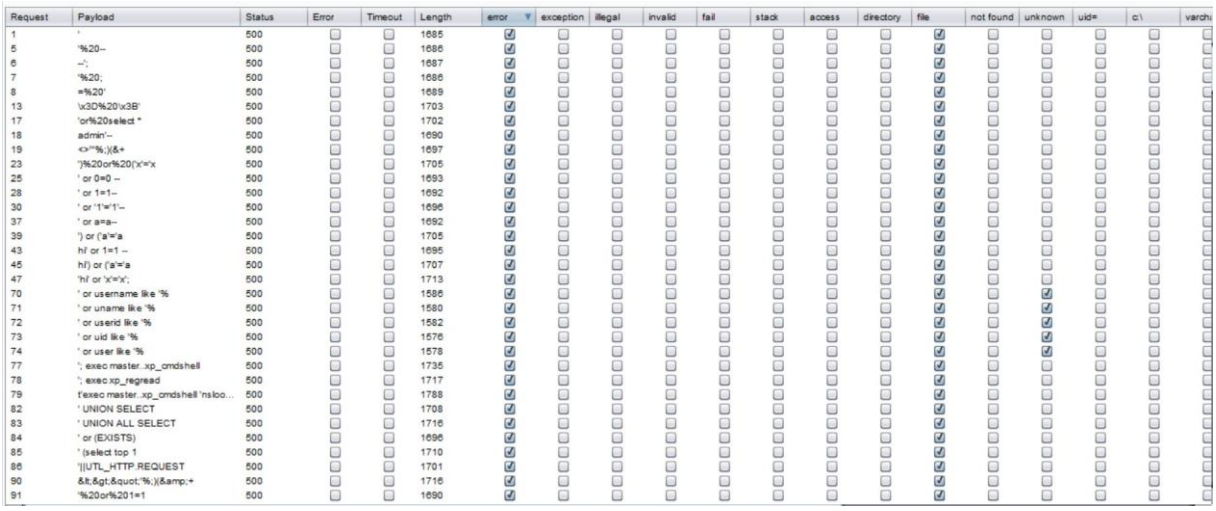
- Method GET

Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varchi
2	"	500			1649	☑								☑					
19	<>""%;)(&+	500			1663	☑								☑					
26	" or 0=0 --	500			1657	☑								☑					
29	" or 1=1--	500			1656	☑								☑					
40	") or ("a"="a	500			1671	☑								☑					
42	hi' or 1=1 --	500			1659	☑								☑					
46	hi") or ("a"="a	500			1673	☑								☑					
128	" and 0=benchmark(3000000,MD...	500			1684	☑								☑					
129	" and 0=benchmark(3000000,MD...	500			1682	☑								☑					
146	") and 0=benchmark(3000000,M...	500			1719	☑								☑					
147	") and 0=benchmark(3000000,M...	500			1719	☑								☑					
148	") and 0=benchmark(3000000,M...	500			1717	☑								☑					
152	")) and 0=benchmark(3000000,M...	500			1721	☑								☑					
153	")) and 0=benchmark(3000000,M...	500			1721	☑								☑					
154	")) and 0=benchmark(3000000,M...	500			1719	☑								☑					
158	"))) and 0=benchmark(3000000,M...	500			1723	☑								☑					
159	"))) and 0=benchmark(3000000,M...	500			1723	☑								☑					
160	"))) and 0=benchmark(3000000,M...	500			1721	☑								☑					
164	")))) and 0=benchmark(3000000,...	500			1725	☑								☑					
165	")))) and 0=benchmark(3000000,...	500			1725	☑								☑					
166	")))) and 0=benchmark(3000000,...	500			1723	☑								☑					
173	",(select%20if(count(*)!=-1,bench...	500			1806	☑								☑					
174	",(select%20if(count(*)!=-1,bench...	500			1806	☑								☑					
175	",(select%20if(count(*)!=-1,bench...	500			1804	☑								☑					
182	"),(select%20if(count(*)!=-1,bench...	500			1807	☑								☑					
183	"),(select%20if(count(*)!=-1,bench...	500			1807	☑								☑					
184	"),(select%20if(count(*)!=-1,bench...	500			1806	☑								☑					
191	'"+if(benchmark(3000000,MD5(1)),...	500			1695	☑								☑					
192	'"+if(benchmark(3000000,MD5(1)),...	500			1695	☑								☑					
193	'"+if(benchmark(3000000,MD5(1)),...	500			1693	☑								☑					
200	'"+if(benchmark(3000000,MD5(1)),...	500			1706	☑								☑					
201	'"+if(benchmark(3000000,MD5(1)),...	500			1704	☑								☑					
202	'"+if(benchmark(3000000,MD5(1)),...	500			1702	☑								☑					

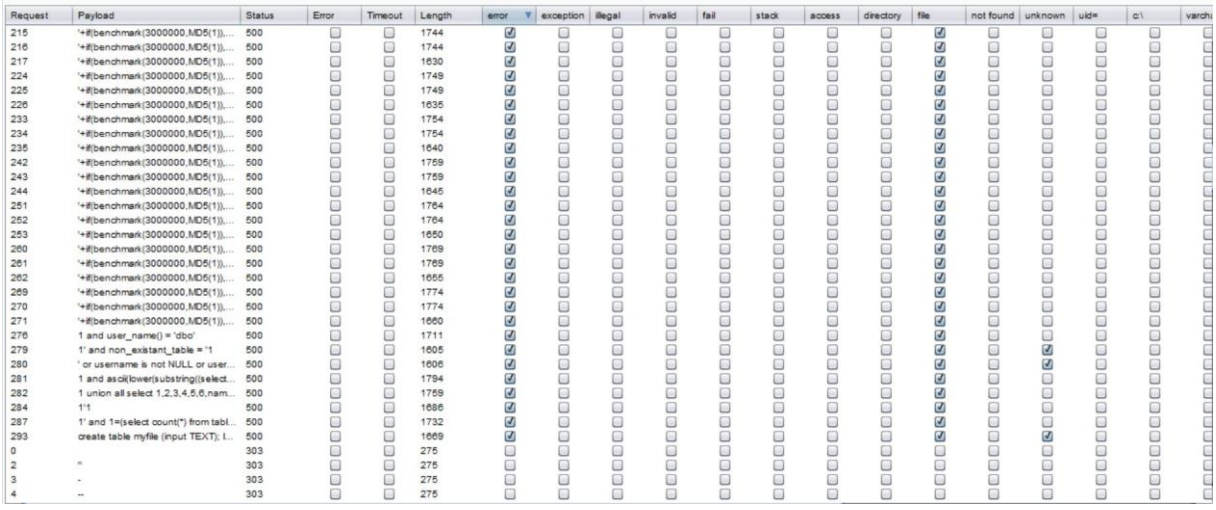
Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varchi
209	'"+if(benchmark(3000000,MD5(1)),...	500			1714	☑								☑					
210	'"+if(benchmark(3000000,MD5(1)),...	500			1714	☑								☑					
211	'"+if(benchmark(3000000,MD5(1)),...	500			1712	☑								☑					
218	'"+if(benchmark(3000000,MD5(1)),...	500			1724	☑								☑					
219	'"+if(benchmark(3000000,MD5(1)),...	500			1724	☑								☑					
220	'"+if(benchmark(3000000,MD5(1)),...	500			1722	☑								☑					
227	'"+if(benchmark(3000000,MD5(1)),...	500			1734	☑								☑					
228	'"+if(benchmark(3000000,MD5(1)),...	500			1734	☑								☑					
229	'"+if(benchmark(3000000,MD5(1)),...	500			1732	☑								☑					
236	'"+if(benchmark(3000000,MD5(1)),...	500			1744	☑								☑					
237	'"+if(benchmark(3000000,MD5(1)),...	500			1744	☑								☑					
238	'"+if(benchmark(3000000,MD5(1)),...	500			1742	☑								☑					
245	'"+if(benchmark(3000000,MD5(1)),...	500			1754	☑								☑					
246	'"+if(benchmark(3000000,MD5(1)),...	500			1754	☑								☑					
247	'"+if(benchmark(3000000,MD5(1)),...	500			1752	☑								☑					
254	'"+if(benchmark(3000000,MD5(1)),...	500			1764	☑								☑					
255	'"+if(benchmark(3000000,MD5(1)),...	500			1764	☑								☑					
256	'"+if(benchmark(3000000,MD5(1)),...	500			1762	☑								☑					
263	'"+if(benchmark(3000000,MD5(1)),...	500			1774	☑								☑					
264	'"+if(benchmark(3000000,MD5(1)),...	500			1774	☑								☑					
265	'"+if(benchmark(3000000,MD5(1)),...	500			1772	☑								☑					
272	'"+if(benchmark(3000000,MD5(1)),...	500			1784	☑								☑					
273	'"+if(benchmark(3000000,MD5(1)),...	500			1784	☑								☑					
274	'"+if(benchmark(3000000,MD5(1)),...	500			1782	☑								☑					
0		200			680														
1	'	200			680														
3	-	200			680														
4	--	200			680														
5	'%20--	200			680														
6	--';	200			680														
7	'%20;	200			680														
8	=%20'	200			680														
9	=%20;	200			680														

- Method POST

Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varchi
92	%27%20or%201=1	500			1690	☑								☑					
94	%20'sleep%2050'	500			1702	☑								☑					
97	'sqlattempt1	500			1705	☑								☑					
125	' and 0=benchmark(3000000,MD...	500			1720	☑								☑					
126	' and 0=benchmark(3000000,MD...	500			1718	☑								☑					
127	' and 0=benchmark(3000000,MD...	500			1707	☑								☑					
143	') and 0=benchmark(3000000,MD...	500			1753	☑								☑					
144	') and 0=benchmark(3000000,MD...	500			1753	☑								☑					
145	') and 0=benchmark(3000000,MD...	500			1751	☑								☑					
149	')) and 0=benchmark(3000000,M...	500			1756	☑								☑					
150	')) and 0=benchmark(3000000,M...	500			1756	☑								☑					
151	')) and 0=benchmark(3000000,M...	500			1754	☑								☑					
155	')) and 0=benchmark(3000000,M...	500			1758	☑								☑					
156	')) and 0=benchmark(3000000,M...	500			1758	☑								☑					
157	')) and 0=benchmark(3000000,M...	500			1756	☑								☑					
161	'))) and 0=benchmark(3000000,...	500			1760	☑								☑					
162	'))) and 0=benchmark(3000000,...	500			1760	☑								☑					
163	'))) and 0=benchmark(3000000,...	500			1758	☑								☑					
170	',(select%20if(count(*)!=-1,bench...	500			1765	☑								☑					
171	',(select%20if(count(*)!=-1,bench...	500			1763	☑								☑					
172	',(select%20if(count(*)!=-1,bench...	500			1752	☑								☑					
179	'),(select%20if(count(*)!=-1,bench...	500			1834	☑								☑					
180	'),(select%20if(count(*)!=-1,bench...	500			1834	☑								☑					
181	'),(select%20if(count(*)!=-1,bench...	500			1833	☑								☑					
188	'"+if(benchmark(3000000,MD5(1)),...	500			1729	☑								☑					
189	'"+if(benchmark(3000000,MD5(1)),...	500			1729	☑								☑					
190	'"+if(benchmark(3000000,MD5(1)),...	500			1615	☑								☑					
197	'"+if(benchmark(3000000,MD5(1)),...	500			1736	☑								☑					
198	'"+if(benchmark(3000000,MD5(1)),...	500			1734	☑								☑					
199	'"+if(benchmark(3000000,MD5(1)),...	500			1620	☑								☑					
206	'"+if(benchmark(3000000,MD5(1)),...	500			1739	☑								☑					
207	'"+if(benchmark(3000000,MD5(1)),...	500			1739	☑								☑					
208	'"+if(benchmark(3000000,MD5(1)),...	500			1606	☑								☑					



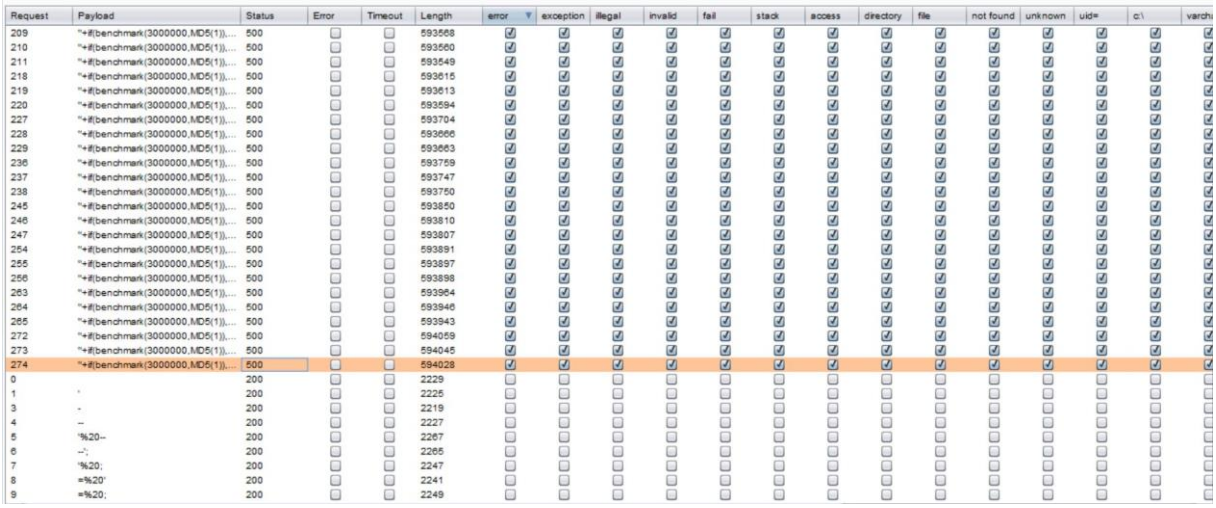
Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
1	'	500			1685	✓								✓					
5	'%20--	500			1686	✓								✓					
6	--';	500			1687	✓								✓					
7	'%20;	500			1686	✓								✓					
8	=%20'	500			1689	✓								✓					
13	\x3D%20\x3B'	500			1703	✓								✓					
17	'or%20select *	500			1702	✓								✓					
18	admin'--	500			1690	✓								✓					
19	<>""%;)(&+	500			1697	✓								✓					
23	')%20or%20('x'='x	500			1705	✓								✓					
25	' or 0=0 --	500			1693	✓								✓					
28	' or 1=1--	500			1692	✓								✓					
30	' or '1'='1'--	500			1696	✓								✓					
37	' or a=a--	500			1692	✓								✓					
39	') or ('a'='a	500			1705	✓								✓					
43	hi' or 1=1 --	500			1695	✓								✓					
45	hi') or ('a'='a	500			1707	✓								✓					
47	'hi' or 'x'='x';	500			1713	✓								✓					
70	' or username like '%	500			1586	✓								✓		✓			
71	' or uname like '%	500			1580	✓								✓		✓			
72	' or userid like '%	500			1582	✓								✓		✓			
73	' or uid like '%	500			1576	✓								✓		✓			
74	' or user like '%	500			1578	✓								✓		✓			
77	'; exec master..xp_cmdshell	500			1735	✓								✓					
78	'; exec xp_regread	500			1717	✓								✓					
79	t'exec master..xp_cmdshell 'nsloo...	500			1788	✓								✓					
82	' UNION SELECT	500			1708	✓								✓					
83	' UNION ALL SELECT	500			1716	✓								✓					
84	' or (EXISTS)	500			1696	✓								✓					
85	' (select top 1	500			1710	✓								✓					
86	'||UTL_HTTP.REQUEST	500			1701	✓								✓					
90	<>"%;)(&+	500			1716	✓								✓					
91	'%20or%201=1	500			1690	✓								✓					



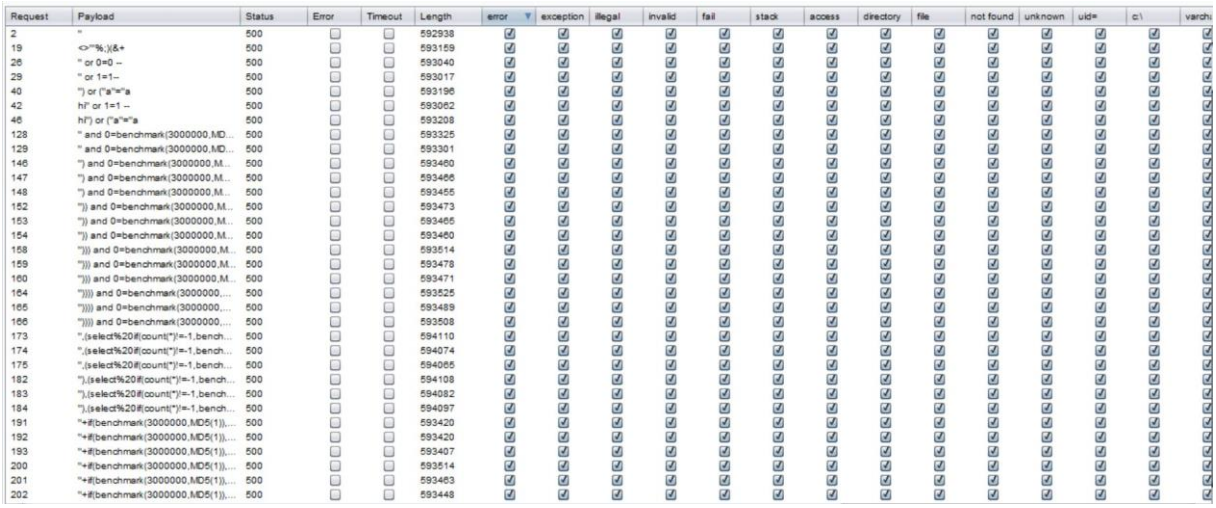
Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
215	'+if(benchmark(3000000,MD5(1)),...	500			1744	✓								✓					
216	'+if(benchmark(3000000,MD5(1)),...	500			1744	✓								✓					
217	'+if(benchmark(3000000,MD5(1)),...	500			1630	✓								✓					
224	'+if(benchmark(3000000,MD5(1)),...	500			1749	✓								✓					
225	'+if(benchmark(3000000,MD5(1)),...	500			1749	✓								✓					
226	'+if(benchmark(3000000,MD5(1)),...	500			1635	✓								✓					
233	'+if(benchmark(3000000,MD5(1)),...	500			1754	✓								✓					
234	'+if(benchmark(3000000,MD5(1)),...	500			1754	✓								✓					
235	'+if(benchmark(3000000,MD5(1)),...	500			1640	✓								✓					
242	'+if(benchmark(3000000,MD5(1)),...	500			1759	✓								✓					
243	'+if(benchmark(3000000,MD5(1)),...	500			1759	✓								✓					
244	'+if(benchmark(3000000,MD5(1)),...	500			1645	✓								✓					
251	'+if(benchmark(3000000,MD5(1)),...	500			1764	✓								✓					
252	'+if(benchmark(3000000,MD5(1)),...	500			1764	✓								✓					
253	'+if(benchmark(3000000,MD5(1)),...	500			1650	✓								✓					
260	'+if(benchmark(3000000,MD5(1)),...	500			1769	✓								✓					
261	'+if(benchmark(3000000,MD5(1)),...	500			1769	✓								✓					
262	'+if(benchmark(3000000,MD5(1)),...	500			1655	✓								✓					
269	'+if(benchmark(3000000,MD5(1)),...	500			1774	✓								✓					
270	'+if(benchmark(3000000,MD5(1)),...	500			1774	✓								✓					
271	'+if(benchmark(3000000,MD5(1)),...	500			1660	✓								✓					
276	1 and user_name() = 'dbo'	500			1711	✓								✓					
279	1' and non_existant_table = '1	500			1605	✓								✓		✓			
280	' or username is not NULL or user...	500			1606	✓								✓		✓			
281	1 and ascii(lower(substring((select...	500			1794	✓								✓					
282	1 union all select 1,2,3,4,5,6,nam...	500			1759	✓								✓					
284	1'1	500			1686	✓								✓					
287	1' and 1=(select count(*) from tabl...	500			1732	✓								✓					
293	create table myfile (input TEXT); l...	500			1669	✓								✓		✓			
0		303			275														
2	"	303			275														
3	-	303			275														
4	--	303			275														

Hasil pengujian serangan *SQL injection* pada *framework* Laravel dengan menggunakan RAW Query :

- Method GET

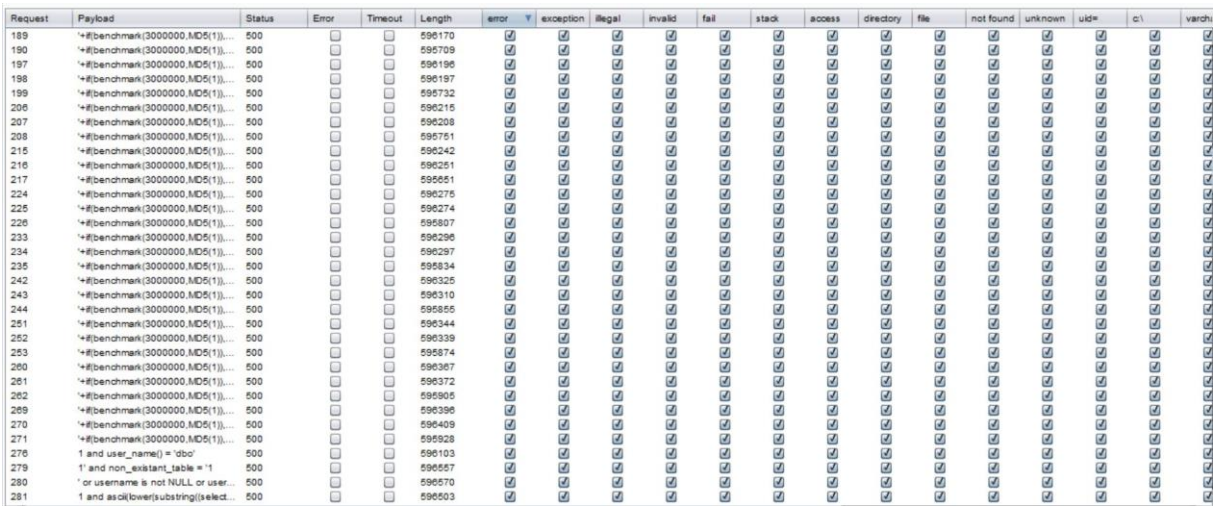


Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
209	'+if(benchmark(3000000,MD5(1)),...	500			593568	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
210	'+if(benchmark(3000000,MD5(1)),...	500			593560	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
211	'+if(benchmark(3000000,MD5(1)),...	500			593549	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
218	'+if(benchmark(3000000,MD5(1)),...	500			593615	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
219	'+if(benchmark(3000000,MD5(1)),...	500			593613	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
220	'+if(benchmark(3000000,MD5(1)),...	500			593594	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
227	'+if(benchmark(3000000,MD5(1)),...	500			593704	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
228	'+if(benchmark(3000000,MD5(1)),...	500			593666	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
229	'+if(benchmark(3000000,MD5(1)),...	500			593663	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
236	'+if(benchmark(3000000,MD5(1)),...	500			593759	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
237	'+if(benchmark(3000000,MD5(1)),...	500			593747	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
238	'+if(benchmark(3000000,MD5(1)),...	500			593750	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
245	'+if(benchmark(3000000,MD5(1)),...	500			593850	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
246	'+if(benchmark(3000000,MD5(1)),...	500			593810	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
247	'+if(benchmark(3000000,MD5(1)),...	500			593807	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
254	'+if(benchmark(3000000,MD5(1)),...	500			593891	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
255	'+if(benchmark(3000000,MD5(1)),...	500			593897	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
256	'+if(benchmark(3000000,MD5(1)),...	500			593898	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
263	'+if(benchmark(3000000,MD5(1)),...	500			593964	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
264	'+if(benchmark(3000000,MD5(1)),...	500			593946	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
265	'+if(benchmark(3000000,MD5(1)),...	500			593943	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
272	'+if(benchmark(3000000,MD5(1)),...	500			594059	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
273	'+if(benchmark(3000000,MD5(1)),...	500			594045	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
274	'+if(benchmark(3000000,MD5(1)),...	500			594028	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
0		200			2229														
1	'	200			2225														
3	-	200			2219														
4	--	200			2227														
5	'%20--	200			2267														
6	--';	200			2265														
7	'%20;	200			2247														
8	=%20'	200			2241														
9	=%20;	200			2249														

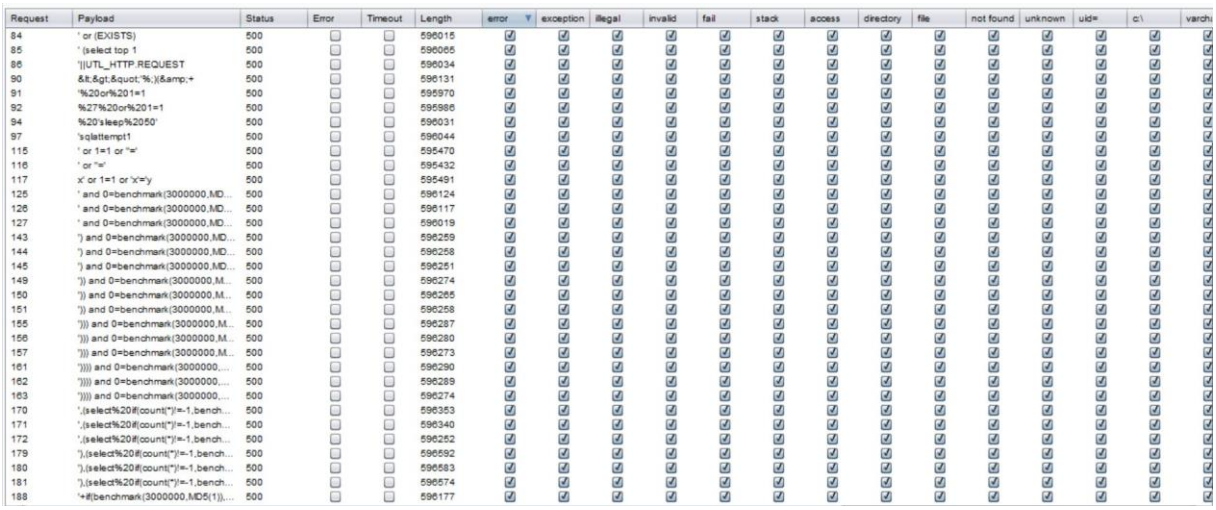


Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varchi
2	"	500			592938														
19	<>""%;)(&+	500			593159														
26	" or 0=0 --	500			593040														
29	" or 1=1--	500			593017														
40	") or ("a"="a	500			593196														
42	hi' or 1=1 --	500			593062														
46	hi") or ("a"="a	500			593208														
128	" and 0=benchmark(3000000,MD...	500			593325														
129	" and 0=benchmark(3000000,MD...	500			593301														
146	") and 0=benchmark(3000000,M...	500			593460														
147	") and 0=benchmark(3000000,M...	500			593466														
148	") and 0=benchmark(3000000,M...	500			593455														
152	")) and 0=benchmark(3000000,M...	500			593473														
153	")) and 0=benchmark(3000000,M...	500			593465														
154	")) and 0=benchmark(3000000,M...	500			593460														
158	"))) and 0=benchmark(3000000,M...	500			593514														
159	"))) and 0=benchmark(3000000,M...	500			593478														
160	"))) and 0=benchmark(3000000,M...	500			593471														
164	")))) and 0=benchmark(3000000,...	500			593525														
165	")))) and 0=benchmark(3000000,...	500			593489														
166	")))) and 0=benchmark(3000000,...	500			593508														
173	",(select%20if(count(*)!=-1,bench...	500			594110														
174	",(select%20if(count(*)!=-1,bench...	500			594074														
175	",(select%20if(count(*)!=-1,bench...	500			594065														
182	"),(select%20if(count(*)!=-1,bench...	500			594108														
183	"),(select%20if(count(*)!=-1,bench...	500			594082														
184	"),(select%20if(count(*)!=-1,bench...	500			594097														
191	"+if(benchmark(3000000,MD5(1)),...	500			593420														
192	"+if(benchmark(3000000,MD5(1)),...	500			593420														
193	"+if(benchmark(3000000,MD5(1)),...	500			593407														
200	"+if(benchmark(3000000,MD5(1)),...	500			593514														
201	"+if(benchmark(3000000,MD5(1)),...	500			593463														
202	"+if(benchmark(3000000,MD5(1)),...	500			593448														

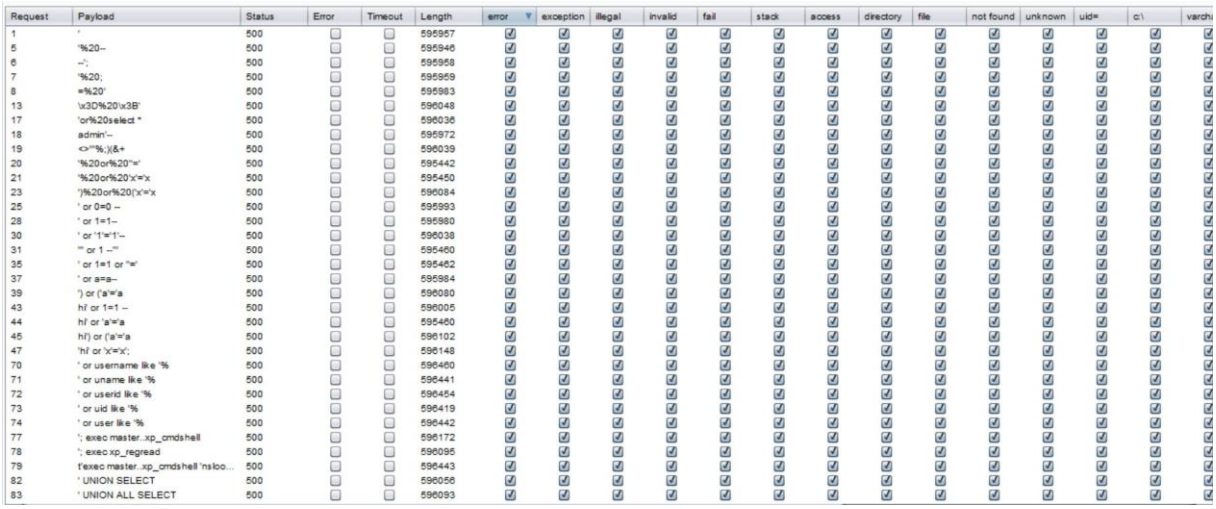
- Method POST



Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varchi
189	'+if(benchmark(3000000,MD5(1)),...	500			596170														
190	'+if(benchmark(3000000,MD5(1)),...	500			595709														
197	'+if(benchmark(3000000,MD5(1)),...	500			596196														
198	'+if(benchmark(3000000,MD5(1)),...	500			596197														
199	'+if(benchmark(3000000,MD5(1)),...	500			595732														
206	'+if(benchmark(3000000,MD5(1)),...	500			596215														
207	'+if(benchmark(3000000,MD5(1)),...	500			596208														
208	'+if(benchmark(3000000,MD5(1)),...	500			595751														
215	'+if(benchmark(3000000,MD5(1)),...	500			596242														
216	'+if(benchmark(3000000,MD5(1)),...	500			596251														
217	'+if(benchmark(3000000,MD5(1)),...	500			595651														
224	'+if(benchmark(3000000,MD5(1)),...	500			596275														
225	'+if(benchmark(3000000,MD5(1)),...	500			596274														
226	'+if(benchmark(3000000,MD5(1)),...	500			595807														
233	'+if(benchmark(3000000,MD5(1)),...	500			596296														
234	'+if(benchmark(3000000,MD5(1)),...	500			596297														
235	'+if(benchmark(3000000,MD5(1)),...	500			595834														
242	'+if(benchmark(3000000,MD5(1)),...	500			596325														
243	'+if(benchmark(3000000,MD5(1)),...	500			596310														
244	'+if(benchmark(3000000,MD5(1)),...	500			595855														
251	'+if(benchmark(3000000,MD5(1)),...	500			596344														
252	'+if(benchmark(3000000,MD5(1)),...	500			596339														
253	'+if(benchmark(3000000,MD5(1)),...	500			595874														
260	'+if(benchmark(3000000,MD5(1)),...	500			596367														
261	'+if(benchmark(3000000,MD5(1)),...	500			596372														
262	'+if(benchmark(3000000,MD5(1)),...	500			595905														
269	'+if(benchmark(3000000,MD5(1)),...	500			596396														
270	'+if(benchmark(3000000,MD5(1)),...	500			596409														
271	'+if(benchmark(3000000,MD5(1)),...	500			595928														
276	1 and user_name() = 'dbo'	500			596103														
279	1' and non_existant_table = '1	500			596557														
280	' or username is not NULL or user...	500			596570														
281	1 and ascii(lower(substring((select...	500			596503														



Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varchi
84	' or (EXISTS)	500			596015														
85	' (select top 1	500			596065														
86	'||UTL_HTTP.REQUEST	500			596034														
90	<>"%;)(&+	500			596131														
91	'%20or%201=1	500			595970														
92	%27%20or%201=1	500			595986														
94	%20'sleep%2050'	500			596031														
97	'sqlattempt1	500			596044														
115	' or 1=1 or ''='	500			595470														
116	' or ''='	500			595432														
117	x' or 1=1 or 'x'='y	500			595491														
125	' and 0=benchmark(3000000,MD...	500			596124														
126	' and 0=benchmark(3000000,MD...	500			596117														
127	' and 0=benchmark(3000000,MD...	500			596019														
143	') and 0=benchmark(3000000,MD...	500			596259														
144	') and 0=benchmark(3000000,MD...	500			596258														
145	') and 0=benchmark(3000000,MD...	500			596251														
149	')) and 0=benchmark(3000000,M...	500			596274														
150	')) and 0=benchmark(3000000,M...	500			596265														
151	')) and 0=benchmark(3000000,M...	500			596258														
155	'))) and 0=benchmark(3000000,M...	500			596287														
156	'))) and 0=benchmark(3000000,M...	500			596280														
157	'))) and 0=benchmark(3000000,M...	500			596273														
161	''''') and 0=benchmark(3000000,...	500			596290														
162	''''') and 0=benchmark(3000000,...	500			596289														
163	''''') and 0=benchmark(3000000,...	500			596274														
170	',(select%20if(count(*)!=-1,bench...	500			596353														
171	',(select%20if(count(*)!=-1,bench...	500			596340														
172	',(select%20if(count(*)!=-1,bench...	500			596252														
179	'),(select%20if(count(*)!=-1,bench...	500			596592														
180	'),(select%20if(count(*)!=-1,bench...	500			596583														
181	'),(select%20if(count(*)!=-1,bench...	500			596574														
188	'+if(benchmark(3000000,MD5(1)),...	500			596177														

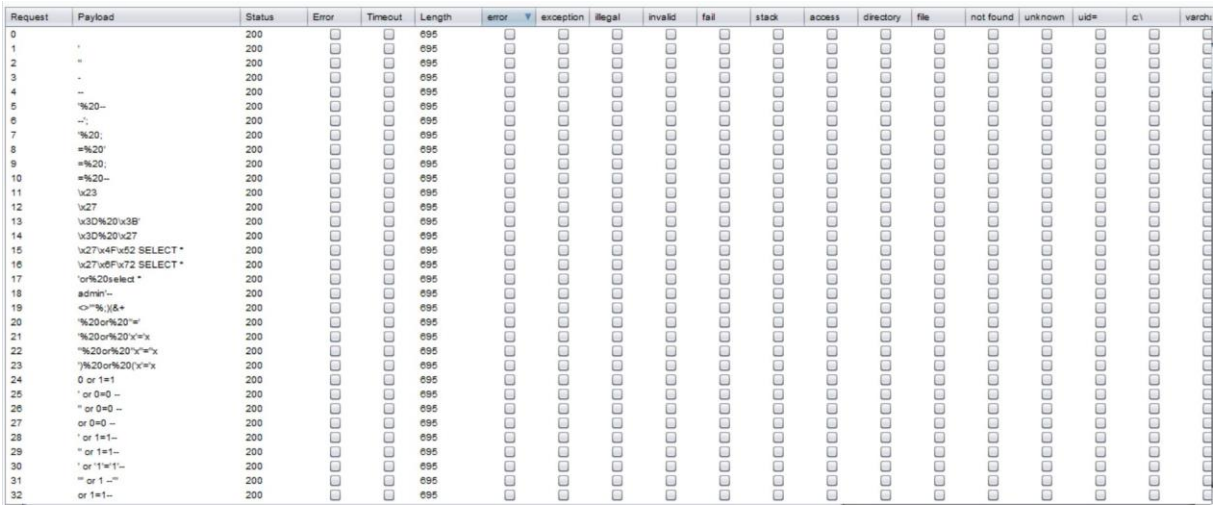


Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
1	'	500			595957	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
5	'%20--	500			595946	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
6	--';	500			595958	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
7	'%20;	500			595959	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
8	=%20'	500			595983	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
13	\x3D%20\x3B'	500			596048	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
17	'or%20select *	500			596036	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
18	admin'--	500			595972	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
19	<>'";);&+	500			596039	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
20	'%20or%20'='	500			595442	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
21	'%20or%20'x'='x	500			595450	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
23	')%20or%20('x'='x	500			596084	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
25	' or 0=0 --	500			595993	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
28	' or 1=1--	500			595980	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
30	' or '1'='1'--	500			596038	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
31	'" or 1 --'"	500			595460	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
35	' or 1=1 or ''='	500			595462	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
37	' or a=a--	500			595984	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
39	') or ('a'='a	500			596080	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
43	hi' or 1=1 --	500			596005	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
44	hi' or 'a'='a	500			595460	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
45	hi') or ('a'='a	500			596102	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
47	'hi' or 'x'='x';	500			596148	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
70	' or username like '%	500			596460	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
71	' or uname like '%	500			596441	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
72	' or userid like '%	500			596454	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
73	' or uid like '%	500			596419	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
74	' or user like '%	500			596442	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
77	'; exec master..xp_cmdshell	500			596172	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
78	'; exec xp_regread	500			596095	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
79	t'exec master..xp_cmdshell 'nsloo...	500			596443	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
82	' UNION SELECT	500			596056	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
83	' UNION ALL SELECT	500			596093	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
282	1 union all select 1,2,3,4,5,6,nam...	500			596332	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
284	1'1	500			595957	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
287	1' and 1=(select count(*) from tabl...	500			596181	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
292	fake@ema'or'il.nl'=il.nl	500			595551	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
293	create table myfile (input TEXT); l...	500			596897	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
0		302			2078														
2	"	302			2070														
3	-	302			2064														
4	--	302			2070														
9	=%20;	302			2068														
10	=%20--	302			2072														
11	\x23	302			2074														
12	\x27	302			2074														
14	\x3D%20\x27	302			2082														
15	\x27\x4F\x52 SELECT *	302			2068														
16	\x27\x6F\x72 SELECT *	302			2076														
22	"%20or%20"x"="x	302			2086														
24	0 or 1=1	302			2074														
26	" or 0=0 --	302			2076														
27	or 0=0 --	302			2068														
29	" or 1=1--	302			2068														
32	or 1=1--	302			2070														
33	or%201=1	302			2074														
34	or%201=1 --	302			2072														
36	" or 1=1 or ""="	302			2070														
38	" or "a"="a	302			2072														
40	") or ("a"="a	302			2080														
41	hi" or "a"="a	302			2068														
42	hi" or 1=1 --	302			2074														
46	hi") or ("a"="a	302			2072														
48	@variable	302			2070														
49	,@variable	302			2074														
50	PRINT	302			2076														

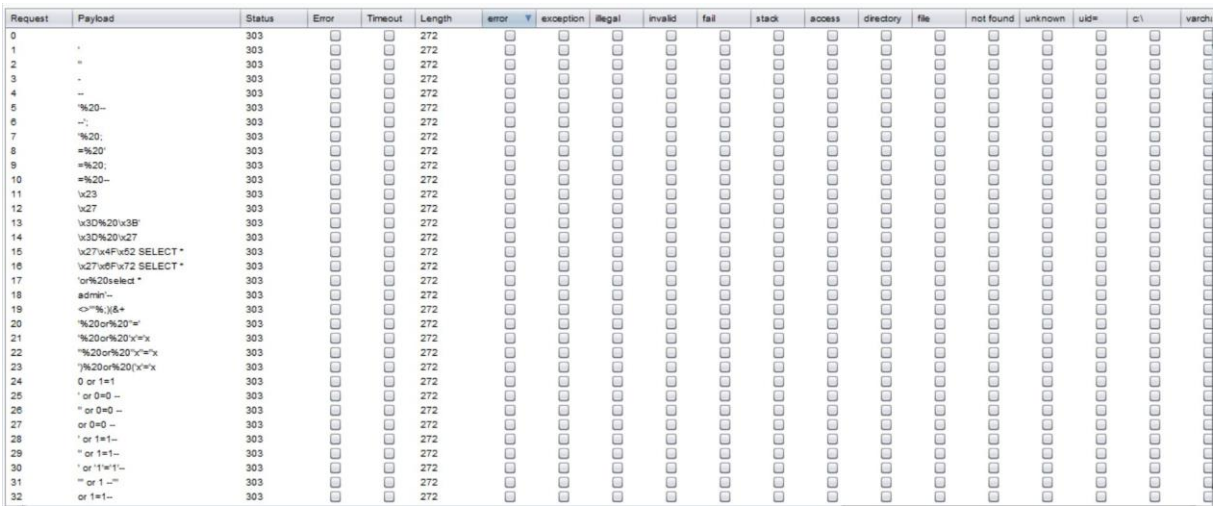
Hasil pengujian serangan *SQL injection* pada *framework* CodeIgniter dengan menggunakan Escaping Query :

- Method GET



Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
0		200			695														
1	'	200			695														
2	"	200			695														
3	-	200			695														
4	--	200			695														
5	'%20--	200			695														
6	--';	200			695														
7	'%20;	200			695														
8	=%20'	200			695														
9	=%20;	200			695														
10	=%20--	200			695														
11	\x23	200			695														
12	\x27	200			695														
13	\x3D%20\x3B'	200			695														
14	\x3D%20\x27	200			695														
15	\x27\x4F\x52 SELECT *	200			695														
16	\x27\x6F\x72 SELECT *	200			695														
17	'or%20select *	200			695														
18	admin'--	200			695														
19	<>'";);&+	200			695														
20	'%20or%20'='	200			695														
21	'%20or%20'x'='x	200			695														
22	"%20or%20"x"="x	200			695														
23	')%20or%20('x'='x	200			695														
24	0 or 1=1	200			695														
25	' or 0=0 --	200			695														
26	" or 0=0 --	200			695														
27	or 0=0 --	200			695														
28	' or 1=1--	200			695														
29	" or 1=1--	200			695														
30	' or '1'='1'--	200			695														
31	'" or 1 --'"	200			695														
32	or 1=1--	200			695														

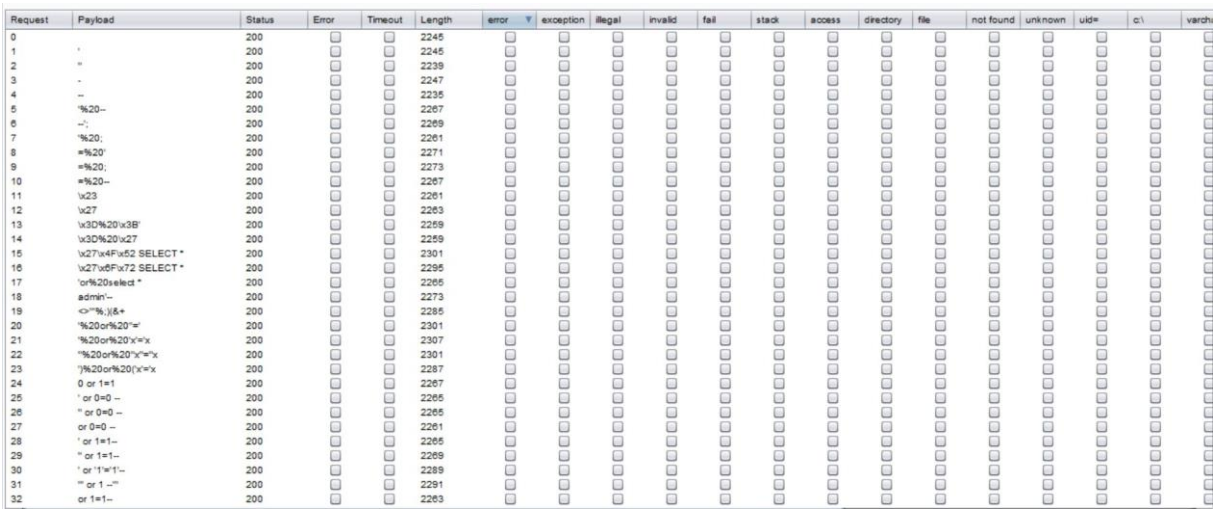
- Method POST



Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
0		303			272														
1	'	303			272														
2	"	303			272														
3	-	303			272														
4	--	303			272														
5	'%20--	303			272														
6	--';	303			272														
7	'%20;	303			272														
8	=%20'	303			272														
9	=%20;	303			272														
10	=%20--	303			272														
11	\x23	303			272														
12	\x27	303			272														
13	\x3D%20\x3B'	303			272														
14	\x3D%20\x27	303			272														
15	\x27\x4F\x52 SELECT *	303			272														
16	\x27\x6F\x72 SELECT *	303			272														
17	'or%20select *	303			272														
18	admin'--	303			272														
19	<>""%;)(&+	303			272														
20	'%20or%20"=	303			272														
21	'%20or%20'x'='x	303			272														
22	"%20or%20"x"="x	303			272														
23	')%20or%20('x'='x	303			272														
24	0 or 1=1	303			272														
25	' or 0=0 --	303			272														
26	" or 0=0 --	303			272														
27	or 0=0 --	303			272														
28	' or 1=1--	303			272														
29	" or 1=1--	303			272														
30	' or '1'='1'--	303			272														
31	'" or 1 --'"	303			272														
32	or 1=1--	303			272														

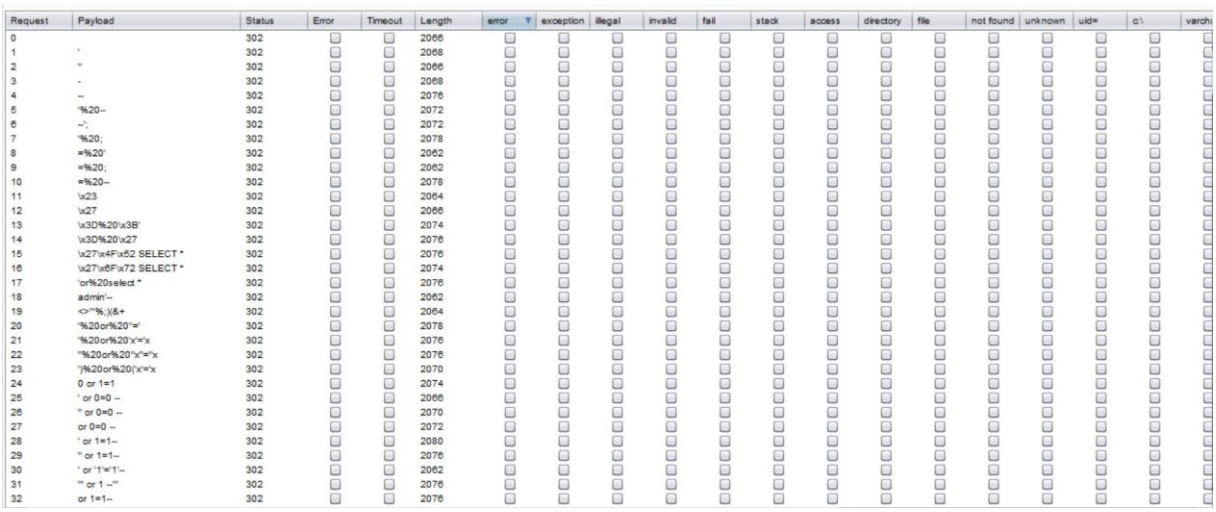
Hasil pengujian serangan *SQL injection* pada *framework* Laravel dengan menggunakan Eloquent ORM :

- Method GET



Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
0		200			2245														
1	'	200			2245														
2	"	200			2239														
3	-	200			2247														
4	--	200			2235														
5	'%20--	200			2267														
6	--';	200			2269														
7	'%20;	200			2261														
8	=%20'	200			2271														
9	=%20;	200			2273														
10	=%20--	200			2267														
11	\x23	200			2261														
12	\x27	200			2263														
13	\x3D%20\x3B'	200			2259														
14	\x3D%20\x27	200			2259														
15	\x27\x4F\x52 SELECT *	200			2301														
16	\x27\x6F\x72 SELECT *	200			2295														
17	'or%20select *	200			2265														
18	admin'--	200			2273														
19	<>""%;)(&+	200			2285														
20	'%20or%20"=	200			2301														
21	'%20or%20'x'='x	200			2307														
22	"%20or%20"x"="x	200			2301														
23	')%20or%20('x'='x	200			2287														
24	0 or 1=1	200			2267														
25	' or 0=0 --	200			2265														
26	" or 0=0 --	200			2265														
27	or 0=0 --	200			2261														
28	' or 1=1--	200			2265														
29	" or 1=1--	200			2269														
30	' or '1'='1'--	200			2289														
31	'" or 1 --'"	200			2291														
32	or 1=1--	200			2263														

- Method POST



Request	Payload	Status	Error	Timeout	Length	error	exception	illegal	invalid	fail	stack	access	directory	file	not found	unknown	uid=	c:\	varch
0		302			2066														
1	'	302			2068														
2	"	302			2066														
3	-	302			2068														
4	--	302			2076														
5	'%20--	302			2072														
6	--';	302			2072														
7	'%20;	302			2078														
8	=%20'	302			2062														
9	=%20;	302			2062														
10	=%20--	302			2078														
11	\x23	302			2064														
12	\x27	302			2066														
13	\x3D%20\x3B'	302			2074														
14	\x3D%20\x27	302			2076														
15	\x27\x4F\x52 SELECT *	302			2076														
16	\x27\x6F\x72 SELECT *	302			2074														
17	'or%20select *	302			2076														
18	admin'--	302			2062														
19	<>""%;)(&+	302			2064														
20	'%20or%20"=	302			2078														
21	'%20or%20'x'='x	302			2076														
22	"%20or%20"x"="x	302			2076														
23	')%20or%20('x'='x	302			2070														
24	0 or 1=1	302			2074														
25	' or 0=0 --	302			2066														
26	" or 0=0 --	302			2070														
27	or 0=0 --	302			2072														
28	' or 1=1--	302			2080														
29	" or 1=1--	302			2076														
30	' or '1'='1'--	302			2062														
31	'" or 1 --'"	302			2076														
32	or 1=1--	302			2076														