APLIKASI PENDETEKSI CELAH KEAMANAN APLIKASI *WEB* DENGAN *PENETRATION TESTING* MENGGUNAKAN METODE *INPUT VALIDATION TESTING*

Tamam Achmad Hanafi[1], Catur Iswahyudi[2], Rr. Yuliana Rachmawati[3]
Program Studi Informatika, Fakultas Teknologi Industri
Institut Sains & Teknologi AKPRIND Yogyakarta
E-mail: [1]tamamhanafi@gmail.com, [2]catur@akprind.ac.id, [3]yuliana@akprind.ac.id

ABSTRACT

Security in a website is an absolute thing, because it is development a website can make buying and selling transactions, entering personal data, and so on. These activities contain important data such as identity numbers, account numbers, to personal data. Therefore security in an application is an absolute thing.

Apencesting is an application made with the Java programming language using the Input Validation Testing method using Remote File Inclusion, Local File Inclusion, and SQL Injection techniques found in the Open Web Application Security Project (OWASP). An open community dedicated to the security of an application.

The results of this study are an application that can detect the vulnerability of a web-based application using the Input Validation Testing method with Remote File Inclusion, Local File Inclusion, SQL Injection techniques.

***Keywords**:* Apencesting, OWASP, *Input Validation Testing*, *Java*.

INTISARI

Keamanan dalam sebuah *website* adalah hal yang mutlak, karena dalam perkembangannya sebuah *website* dapat melakukan transaksi jual beli, memasukkan data pribadi, dan sebagainya. Aktivitas-aktivitas tersebut terdapat data penting seperti nomor identitas, nomor rekening, hingga data pribadi. Oleh karena itu keamanan dalam sebuah aplikasi adalah sebuah hal yang mutlak.

Apencesting adalah sebuah aplikasi yang dibuat dengan bahasa pemrograman *Java* dengan metode *Input Validation Testing* menggunakan teknik *Remote File Inclusion, Local File Inclusion,* dan *SQL Injection* yang terdapat dalam *Open Web Application Security Project* (OWASP). Sebuah komunitas terbuka yang didedikasikan pada keamanan sebuah aplikasi.

Hasil dari penelitian ini adalah sebuah aplikasi yang dapat mendeteksi kerentanan sebuah aplikasi berbasis *web* dengan metode *Input Validation* Testing dengan teknik *Remote File Inclusion, Local File Inclusion, SQL Injection*.

Kata Kunci: Apencesting, OWASP, *Input Validation Testing*, *Java.*

PENDAHULUAN

Website (situs web) merupakan salah satu aplikasi yang ada saat ini, selain aplikasi lain yang berbasis *mobile* maupun *desktop*. Situs *web* merupakan aplikasi yang dibuat dari berbagai bahasa pemrograman seperti *Hypertext Preprocessor* (PHP), *Javascript, Perl, C, C++*, *Java*, *Ruby, Swift* dan bahasa pemrograman *Pyhton.*

Situs-situs *website* seperti facebook.com, detik.com, twitter.com, akprind.ac.id, youtube.com, pemda-diy.go.id merupakan salah satu aplikasi yang berbasis *web,* walaupun dalam perkembangannya selain dalam berbasis *website*, situs-situs tersebut ada yang berbasis *mobile*, baik itu *native application* maupun *web mobile application*. Pada perkembangannya *website* mengalami inovasi, dimulai dari pemrograman terstruktur, pemrograman berbasis *Object Oriented*, berbasis *Framework*, hingga *web mobile application* sebuah aplikasi situs web yang diperuntukkan untuk perangkat telepon genggam.

Aplikasi berbasis *web* mempunyai kelebihan seperti yang dikutip dari (Sosmed, 2014), disebutkan bahwa kelebihan dari aplikasi berbasis *web* adalah sebagai berikut:

1. *File* dan *Database* dari *Software* akan terpusat dan hanya perlu melakukan installasi di *Server,* dan memudahkan untuk proses *update* atau perawatan *software*.

2. Dapat dengan mudah diakses dari jarak jauh melalui *browser* tanpa harus melakukan installasi *software*.

Dari keunggulan diatas, aplikasi berbasis *web* tersebut bisa diandalkan karena dapat diakses dimana saja tanpa perlu melakukan *installasi* pada masing-masing perangkat, yang terpenting adanya koneksi *internet* dan *browser*.

Dibalik keunggulan dari aplikasi berbasis *web*, mempunyai kelemahan-kelemahan, dikutip dari sumber yang sama (Sosmed, 2014) , kekurangan dari aplikasi berbasis *web* adalah sebagai berikut:

1. Harus menggunakan koneksi *internet* untuk mengakses aplikasi berbasis *web*.
2. Tingkat keamanan *data* dan *file* rentan untuk disabotase.

Beberapa kelemahan tersebut dapat ditemukan dalam aplikasi berbasis *web*, karena pada dasarnya setiap teknologi selain memiliki kelebihan, juga terdapat kekurangan.

Dari kekurangan-kekurangan dalam sebuah aplikasi, terdapat sebuah celah yang dapat dimanfaatkan oleh oknum yang disebut *cracker*, yaitu orang maupun sekelompok orang yang tidak mempunyai hak untuk mengakses suatu aplikasi maupun sistem dengan tujuan-tujuan yang dapat merugikan orang maupun organisasi yang terkait dengan sistem tersebut.

Permasalahan yang diangkat adalah bagaimana menerapkan metode *Input Validation Testing* untuk mendeteksi celah kerentenan sebuah situs web serta membuat aplikasi pendeteksi celah kerentanan situs web dengan metode *Input Validation Testing.*

Pada penelitian ini membatasi ruang lingkup objek penelitian yaitu aplikasi yang dibuat hanya untuk tipe pemrograman yang berbasis *Hypertext Preprocessor* (PHP) dan *database* yang digunakan adalah *MySQL* dengan metode yang digunakan yaitu *Input Validation Testing* dengan serangan yang dideteksi yaitu *SQL Injection, Local File Inclusion* serta *Remote File Inclusion.*

Penelitian ini bertujuan untuk menerapkan metode *Input Validation Testing* untuk mendeteksi celah kerentanan atau keamanan suatu situs web dan membuat sebuah aplikasi pendeteksi celah keamanan situs web dengan metode *Input Validation Testing.*

Manfaat yang diharapkan dari aplikasi adalah dengan dilakukan sebuah penetrasi, maka akan ditemukan sebuah celah keamanan dalam sebuah situs web, sehingga pengembang dapat memanfaatkannya untuk mengevaluasi celah keamanan tersebut, sehingga keamanan dalam sebuah situs web menjadi lebih baik.

TINJAUAN PUSTAKA

Penelitian ini menggunakan beberapa referensi yang berhubungan dengan objek pembahasan. Adapun referensi diambil dari tugas akhir atau jurnal yang berhubungan dengan penelitian dan hasil penelitian yang telah dilakukan.

Penelitian yang dilakukan (Kirit I., 2015) membuat sebuah konsep IDS yang tidak bergantung pada satu bahasa, namun dirancang untuk aplikasi web apa pun yang dikembangkan dengan dukungan PHP, JAVA, Dotnet, dan lain sebagainya. *Intrusion detection system* (IDS) adalah sebuah sistem yang melakukan pengawasan terhadap lalu lintas (*traffic*) jaringan dan pengawasan terhadap kegiatan-kegiatan yang mencurigakan di dalam sebuah sistem jaringan. Konsep IDS tersebut membantu untuk mendeteksi kelemahan validasi *input* seperti *SQL Injection, Cross site scripting, Command injection,* dan *Directory traversal* yang tidak terdeteksi oleh IDS yang ada.

Referensi utama adalah sebuah tugas akhir dari (Nababan, 2014) yang membuat sebuah aplikasi pendeteksi celah keamanan aplikasi berbasis *website*. Dalam penelitian ini menggunakan metode *Data Validation Testing* untuk mencari celah-celah yang terdapat dalam sebuah *website* yang dilakukan pengujian. Hasil dari penelitian ini adalah sebuah aplikasi yang dapat mendeteksi celah keamanan yang rentan terhadap *SQL Injection, Cross-Site Scripting,* dan *File Inclusion* pada sebuah *website*.

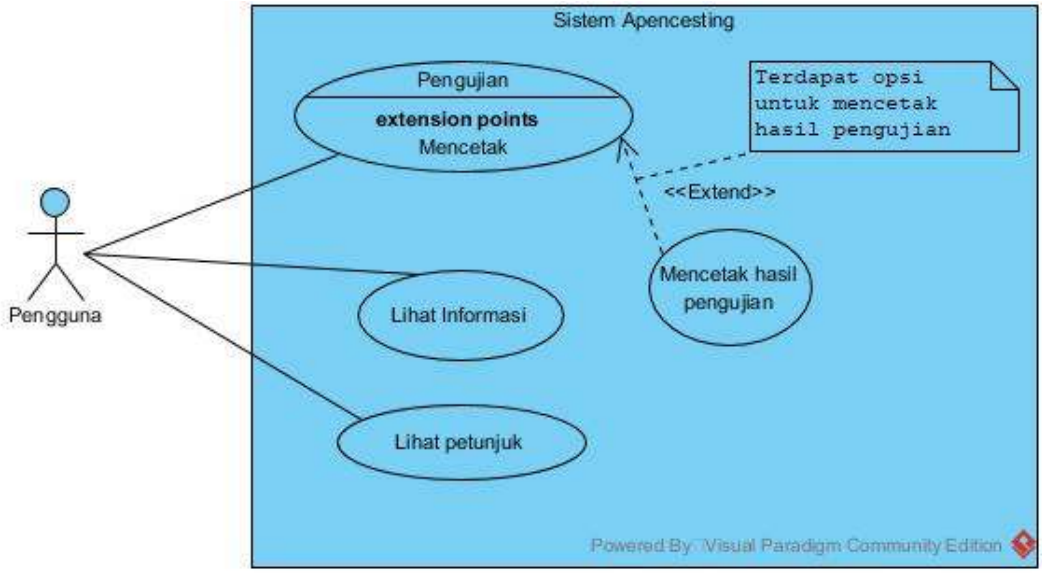
Dalam penelitian ini terdapat juga referensi dari (Fajaryanto, dkk, 2015) yaitu menguji sebuah *web server* suatu instansi pendidikan IKIP PGRI Madiun, metode yang digunakan ialah metode ISSAF (*Information Systems Security Assessment Framework)* dan OWASP versi 4. Hasil dari penelitian ini menunjukkan bahwa *web server* IKIP PGRI Madiun masih dapat ditembus dan mengambil hak akses administrator.

Referensi tentang metode penetrasi testing didapat dari (Riadi, dkk, 2016) yaitu menganalisis kemanan *web server* menggunakan penetrasi testing, metode pengujian yang digunakan mencakup persiapan ujian, tes dan analisis tes. Sedangkan tahap uji coba melibatkan pengumpulan informasi, analisis kerentanan dan kerentanan mengeksploitasi. Hasil atau tujuan dari penelitian ini adalah untuk menguji aplikasi *web server*.

Dari keempat referensi yang telah dijabarkan yang mempunyai tema tentang keamanan sebuah *website*, dapat dijadikan acuan dalam penelitian tentang pembuatan aplikasi yang mendeteksi kerentanan sebuah aplikasi berbasis *web* dengan metode *Input Validation Testing*. Perbedaan dengan pustaka yang terdahulu adalah pada penelitian ini yaitu metode *Input Validation Testing* dengan menggunakan 3 teknik meliputi *SQL Injection, Local File Inclusion* serta *Remote File Inclusion* yang belum pernah digunakan secara bersamaan. Diharapkan dengan adanya aplikasi ini dapat membantu pengembang berbasis *web* dengan *Hypertext Preprocessor.*

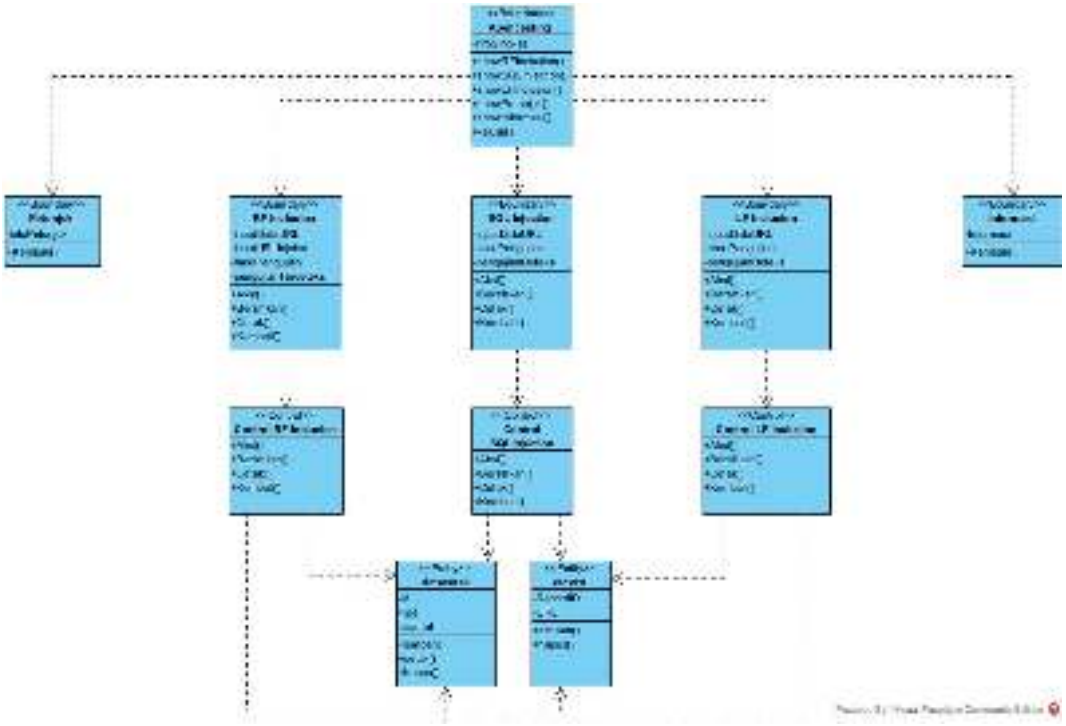
PEMBAHASAN

Use case Diagram menggambarkan siapa yang akan menggunakan sistem dan dalam cara seperti apa pengguna (*User*) akan berinteraksi dengan sistem tersebut. Pada penelitian ini hanya terdapat satu aktor yaitu pengguna atau tester yang menjalankan aplikasi tersebut untuk menguji tingkat kerentanan suatu situs web yang diuji, yaitu dengan memilih salah satu tipe jenis serangan maupun semua jenis serangan yang ada di dalam aplikasi dan menerima hasil pengujian. Selain itu, pengguna dapat mencetak hasil pengujian serta mengakses informasi maupun petunjuk penggunaan. *Use case diagram* aplikasi Apencesting ditunjukkan pada Gambar 1.



Gambar 1. *Use Case Diagram*

Class diagram digunakan untuk menggambarkan struktur statis dari sistem dimana diagram ini menunjukkan kelas objek yang menyusun sistem. *Class Diagram* dari sebuah sistem yang terdiri dari *boundary, control,* dan *entity* yang di dalamnya terdapat beberapa *atribut* dan *operation*. Pada sistem terdapat 6 *boundary*, 3 *control*, dan 2 buah *entity.* Class *Diagram* sistem ini dapat dilihat pada Gambar 2.

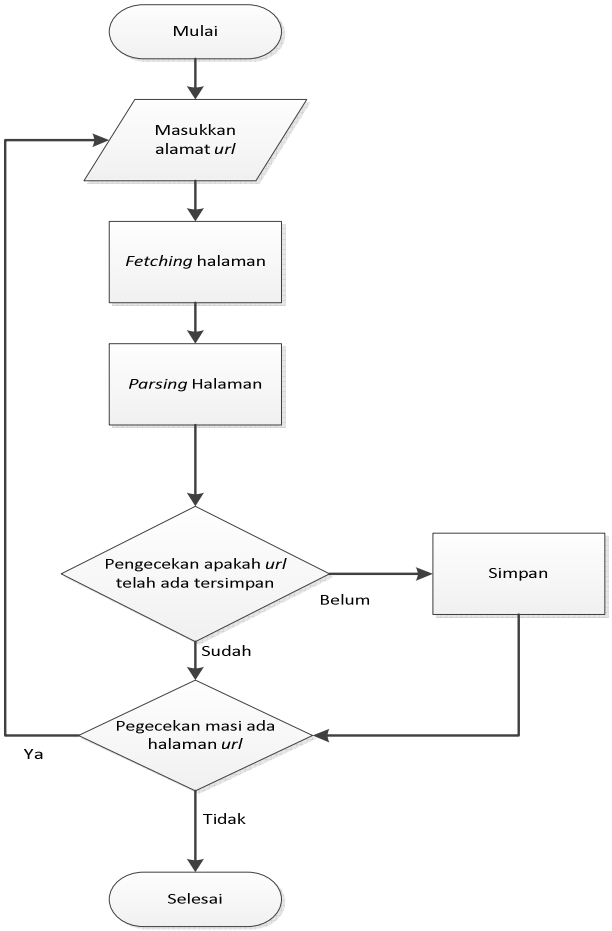


Gambar 2. *Class Diagram*

Pada *Class Diagram* Apencesting *boundary* Apencesting terhubung dengan kelima *boundary* yaitu Petunjuk, RF *Inclusion, SQL Injection,* LF *Injection*, dan Informasi. *Boundary* ini menggambarkan setiap *interface* dari setiap tampilan antarmuka halaman utama yaitu pada *boundary* Apencesting, tampilan pengujian dengan RFI yaitu *boundary* RF *Inclusion* yang terhubung dengan c*ontrol* RF *Inclusion* yaitu yang memproses setiap instruksi dan *entity* yang berupa tabel datacetak dan *record*, tampilan pengujian dengan LFI yaitu *boundary* LF *Inclusion* yang terhubung dengan c*ontrol* LF *Inclusion* yaitu yang memproses setiap instruksi dan *entity* yang berupa tabel datacetak dan *record*, tampilan pengujian dengan *SQL Injection* yaitu *boundary SQL Injection* yang terhubung dengan c*ontrol SQL Injection* yaitu yang memproses setiap instruksi dan *entity* yang berupa tabel datacetak dan *record*, tampilan halaman petunjuk yaitu *boundary* Petunjuk dan tampilan halaman informasi yaitu *boundary* Informasi.

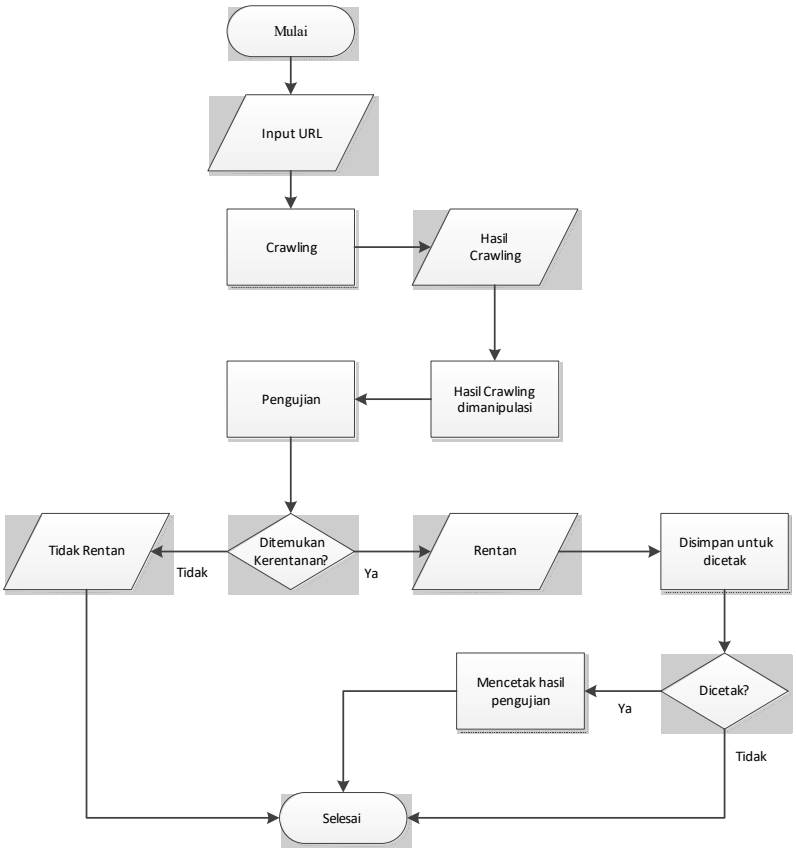
Algoritma metode *Crawling* atau *Crawler* ada proses untuk menemukan halaman-halaman sebuah *website*. Maka algoritma dari metode *crawling* tersebut ditunjukkan pada Gambar 3 dan penjelasannya sebagai berikut :

1. Pertama pengguna memasukkan halaman *url*.
2. Selanjutnya melakukan proses *fetching* yaitu sebuah proses mengambil *url* dan halaman web dari *internet*.
3. Pada halaman yang telah diambil, melakukan proses *parsing* yaitu mengambil seluruh informasi dan memandu langkah *crawling* selanjutnya atau mengambil *link-link* ke halaman lain.
4. Setelah melewati melewati proses *parsing* maka *link* atau halaman web tersebut dicek apakah telah tersimpan belum, jika belum makaakan disimpan.
5. Apabila masih terdapat halaman *url* maka akan kembali ke proses nomor dua, jika tidak maka proses *crawling* selesai.



Gambar 3. Algoritma Metode *Crawling*

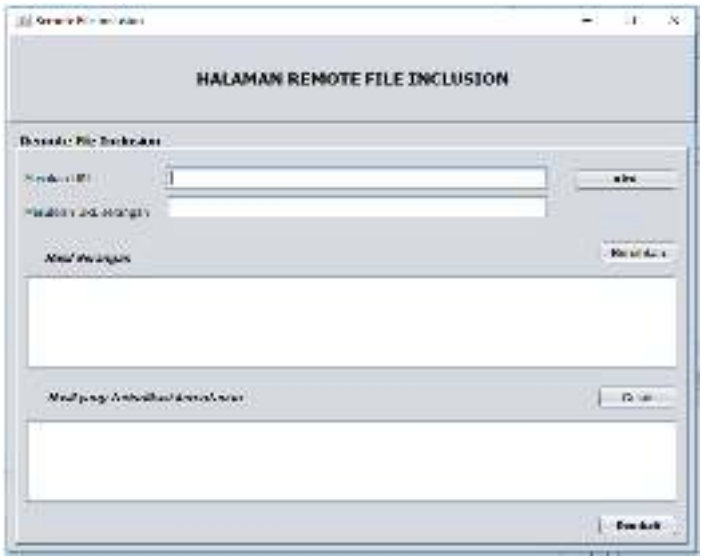
Proses sebuah pengujian *website* dimulai dari memasukkan sebuah halaman url, dari halaman *url* tersebut akan diproses *crawling* yaitu menemukan semua halaman-halaman yang ada di dalam *website*. Setelah proses *crawling* selesai maka akan dilakukan manipulasi terhadap *url* tersebut dengan ditambahkan sesuai jenis pengujian. Hasil dari pengujian akan dibaca oleh aplikasi dan ditentukan apakah halaman tersebut terdapat kerentanan atau tidak. Apabila ditemukan kerentanan maka dapat dicetak sebagai bahan evaluasi. Flowchart pengujian pada aplikasi Apencesting ditunjukkan pada Gambar 4.



Gambar 4. *Flowchart* Apencesting

Hasil yang diperoleh dari penelitian ini adalah aplikasi yang berbasis *desktop* dengan menggunakan bahasa pemrograman *java* yang mempunyai 6 tampilan antarmuka, dan yang ditulis dalam naskah publikasi antara lain sebagai berikut :

1. Tampilan Halaman Pengujian *Remote File Inclusion*



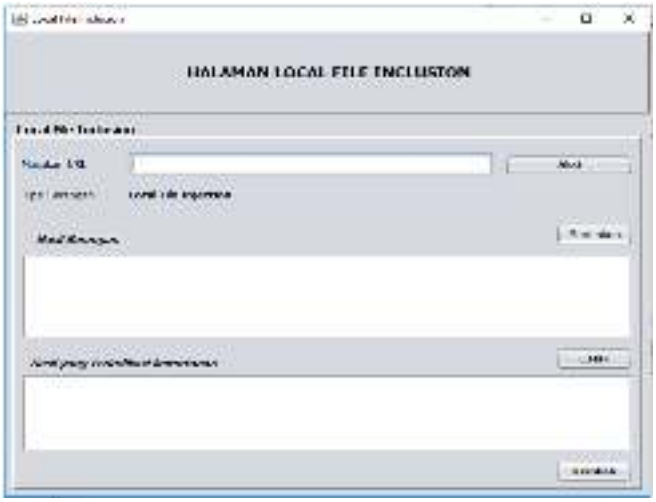
Gambar 5. Halaman Pengujian *Remote File Inclusion*

Tampilan halaman pengujian *Remote File Inclusion* adalah sebuah halaman untuk melakukan pengujian suatu *website* dengan teknik *Remote File Inclusion*. Pada halaman ini terdapat 2 buah *inputan* yang digunakan untuk memasukkan alamat *url* yang diuji dan *url*

pengujian. Lalu 2 buah *TextArea* untuk mengeluarkan hasil pengujian, dan 4 tombol masing-masing adalah tombol Aksi untuk melakukan serangan atau pengujian, tombol Bersihkan untuk membersihkan *TextField* dan *TextArea*, tombol Cetak untuk mencetak hasil pengujian dan tombol Kembali untuk kembali ke halaman utama. Tampilan halaman pengujian *Remote File Inclusion* ditunjukkan pada Gambar 5. Algoritma pengujian *Remote File Inclusion* ditunjukkan pada Gambar 6.

Gambar 6. Algoritma Pengujian *Remote File Inclusion*

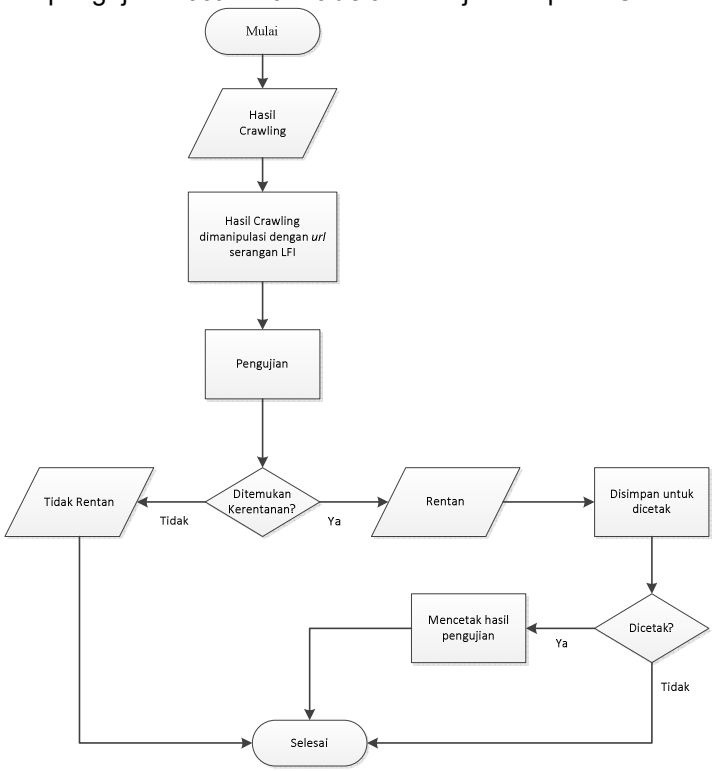
2. Tampilan Halaman Pengujian *Local File Inclusion*



Gambar 7. Halaman Pengujian *Local File Inclusion*

Tampilan halaman pengujian *Local File Inclusion* adalah sebuah halaman untuk melakukan pengujian suatu *website* dengan teknik *Local File Inclusion*. Pada halaman ini terdapat 1 buah *inputan* yang digunakan untuk memasukkan alamat *url*, 2 buah *TextArea* untuk mengeluarkan hasil pengujian, dan 4 tombol masing-masing adalah tombol Aksi untuk

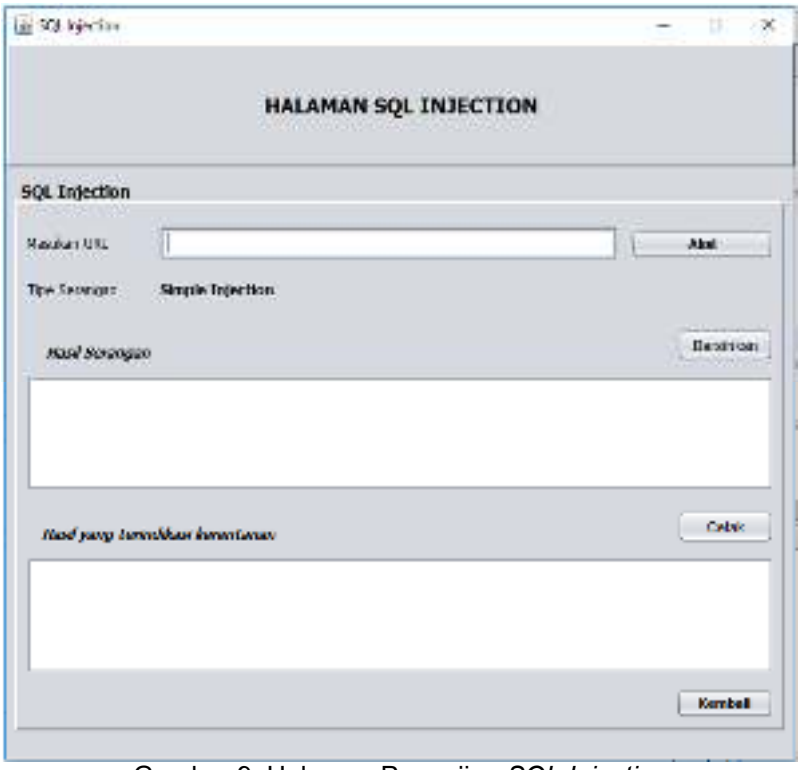
melakukan serangan atau pengujian, tombol Bersihkan untuk membersihkan *TextField* dan *TextArea*, tombol Cetak untuk mencetak hasil pengujian dan tombol Kembali untuk kembali ke halaman utama. Tampilan halaman pengujian *Local File Inclusion* ditunjukkan pada Gambar 7. Algoritma pengujian *Local File Inclusion* ditunjukkan pada Gambar 8.



Gambar 8. Algoritma Pengujian *Local File Inclusion*

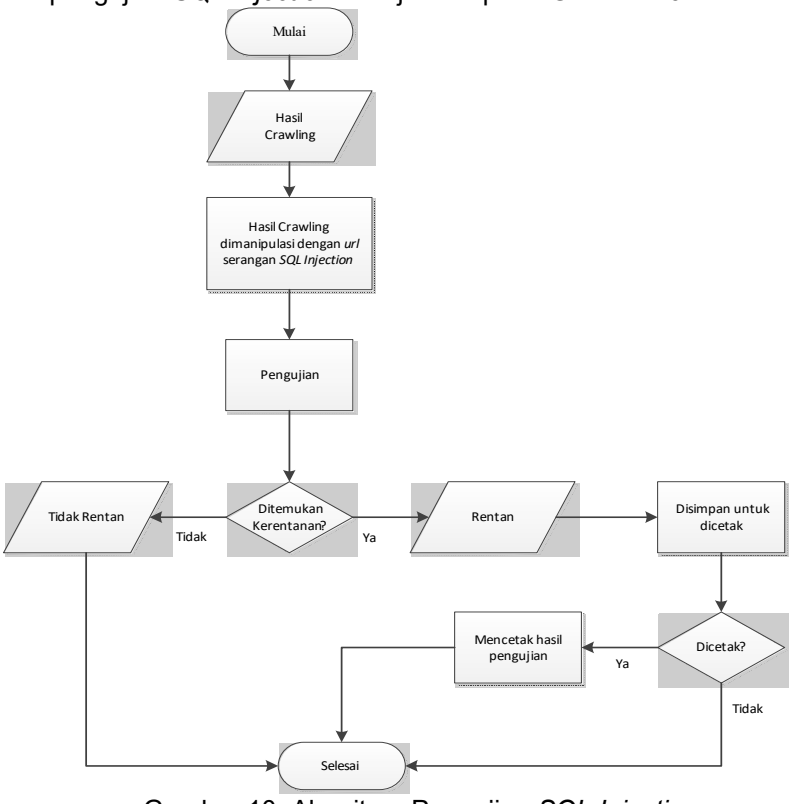
3. Tampilan Halaman Pengujian *SQL Injection*

Tampilan halaman pengujian *SQL Injection* adalah sebuah halaman untuk melakukan pengujian suatu *website* dengan teknik *SQL Injection*. Pada halaman ini terdapat 1 buah *inputan* yang digunakan untuk memasukkan alamat *url*, 2 buah *TextArea* untuk mengeluarkan hasil pengujian, dan 4 tombol masing-masing adalah tombol Aksi untuk melakukan serangan atau pengujian, tombol Bersihkan untuk membersihkan *TextField* dan *TextArea*, tombol Cetak untuk mencetak hasil pengujian dan tombol Kembali untuk kembali ke halaman utama. Tampilan halaman pengujian *SQL Injection* ditunjukkan pada Gambar 9.



Gambar 9. Halaman Pengujian *SQL Injection*

Algoritma pengujian *SQL Injection* ditunjukkan pada Gambar 10.



Gambar 10. Algoritma Pengujian *SQL Injection*

Setelah dilakukan pengujian pada aplikasi di masing-masing teknik serangan maka hasil pengujian dari ketiga teknik serangan ditunjukkan pada Tabel 1.

Tabel 1. Hasil Pengujian Aplikasi

No	Jenis Serangan	Kondisi yang diharapkan	Hasil	Kondisi yang diharapkan	Hasil
		Rentan		Lebih Aman	
1	*Remote File Inclusion*	√	Sesuai	√	Sesuai
2	*Local File Inclusion*	√	Sesuai	√	Sesuai
3	*SQL Injection*	√	Sesuai	√	Sesuai

Penjelasan mengenai Tabel 1 ialah pada kolom rentan dan lebih aman tersebut adalah mengacu pada 2 *website* yang telah disiapkan yaitu *website* yang dikondisikan rentan dan *website* yang dikondisikan aman. Pada Tabel 1 semua jenis pengujian dapat berjalan dengan baik dikarenakan ketika menguji *website* yang rentan dapat mendeteksi dan pada kondisi yang lebih aman tidak mendeteksi. Perbedaan dengan referensi terdahulu ialah aplikasi menggunakan metode *Input Validation Testing* dengan teknik *Remote File Inclusion, Local File Inclusion,* dan *SQL Injection*.

KESIMPULAN

Berdasarkan hasil pengujian aplikasi dapat dijadikan kesimpulan pada penelitian ini antara lain:

1. Dapat menerapkan dan mengembangkan aplikasi dengan metode *Input Validation Testing* untuk mendeteksi celah keamanan *website*.
2. Berdasarkan hasil pengujian aplikasi pada Tabel 1 maka dapat disimpulkan bahwa aplikasi dapat berjalan dengan baik dan sesuai dengan tujuan penelitian.
3. Dapat membantu pengembang sebuah *website* dalam menemukan celah keamanan sebuah aplikasi berbasis PHP.

DAFTAR PUSTAKA

Fajaryanto, A., Prayudi, Y., & Dirgahayu, R. T. (2015). Penerapan Metode ISSAF dan OWASP versi 4 Untuk Uji Kerentanan Web Server. *Jurnal Ilmiah NERO*.

Kirit I., C. (2015). Comparative Study of Detect IVAs over the Web Application. *Comparative Study of Detect IVAs over the Web Application*.

Nababan, I. M. (2014). Pendeteksi Celah Keamanan Pada Aplikasi Web Dengan Penetration Testing Menggunakan Data Validation Testing. 14.

Sosmed, E. (2014, Februari 6). *Kelebihan dan Kekurangan Program Desktop dan Web Based*. Retrieved from essiitech.com: http://essiitech.com/portfolio/kelebihan-dan-kekurangan-program-desktop-dan-web-based

Yunanri, Riadi, I., & Yudhana, A. (2016). Analisis Keamanan Webserver Menggunakan Metode Penetrasi Testing.