JURNAL ILMU KOMPUTER KHARISMA .TECH

**Jurnal KHARISMA Tech** | ISSN:1907-2317 | e-ISSN: 2810-0344
https://jurnal.kharisma.ac.id/kharismatech/
published by: Pusat Penelitian STMIK KHARISMA Makassar

Volume: 17 no. 01 – Maret 2022- hlm. 1-15

# ANALISIS KERENTANAN WEBSITE RENOVACTION MENGGUNAKAN RANGKAIAN SECURITY TOOLS PROJECT BERDASARKAN FRAMEWORK OWASP

**Oleh :**

**Erlan Wismunandar Darwis[1*], Junaedy[2], Izmy Alwiah Musdar[3]**
[1,3]Informatika, STMIK KHARISMA Makassar
[2]Teknik Informatika, Universitas Islam Makassar
e-mail: [1] erlanwismunandar_18@kharisma.ac.id, [2] junaedy@uim-makassar.ac.id,
[3] izmyalwiah@kharisma.ac.id

***Abstrak:*** *Tujuan dari penelitian ini adalah untuk menganalisis kerentanan website agar terhindar dari serangan siber khususnya pada jenis serangan Cross Site Scripting & Sql Injection dengan menerapkan kaidah OWASP Top 10 2017 untuk menemukan celah keamanan dengan cara melakukan automated scan menggunakan ajax spider setelah itu dilakukan active scan dan manual scan menggunakan fuzzer untuk melakukan pengujian yang lebih spesifik pada jenis kerentanan Cross-Site Scripting(XSS) dan SQL Injection. Setelah melakukan pengujian terhadap web RenovAction kerentanan yang ditemukan diantaranya Cross-Domain Misconfiguration, Secure Pages Include Mixed Content, X-Frame-Options Header Not Set, Absence of Anti-CSRF Tokens, Cookie No HttpOnly Flag, Cross-Domain JavaScript Source File Inclusion, Incomplete or No Cache-control Header Set, X-Content-Type-Options Header Missing, Charset Mismatch, Information Disclosure - Suspicious Comments, dan Timestamp Disclosure – Unix. Selain mendapatkan kerentanan pada web RenovAction, penulis juga memberikan solusi untuk mengatasi kerentanan pada web RenovAction berdasarkan tool Zed Attack Proxy(ZAP)*
.
***Kata kunci:*** *Kerentanan, ZAP, RenovAction, SQL Injection, Cross Site Scripting(XSS)*

***Abstract:*** *The purpose of this research is to analyze website vulnerabilities to avoid cyber attacks, especially on cross site scripting &sql injection types by applying OWASP Top 10 2017 rules to find security gaps by performing automated scans using ajax spiders after which active scans and manual scans use fuzzer to perform more specific exposures to cross-site scripting (XSS) and SQL injection types. After testing the web RenovAction vulnerabilities found Cross-Domain Misconfiguration, Secure Pages Include Mixed Content, X-Frame-Options Header Not Set, Absence of Anti-CSRF Tokens, Cookie No HttpOnly Flag, Cross-Domain JavaScript Source File Inclusion, Incomplete or No Cache-control Header Set, X-Content-Type-Options Header Missing, Charset Mismatch, dan Information Disclosure - Suspicious Comments, Timestamp Disclosure – Unix., in addition to getting vulnerabilities in the RenovAction web, the author also provided a solution to overcome vulnerabilities in the RenovAction web based on the Zed Attack Proxy (ZAP) tool.*

***Keywords:*** *Vulnerability, ZAP, RenovAction, SQL Injection, Cross Site Scripting(XSS)*

## 1. PENDAHULUAN

Dengan perkembangan zaman dan pertumbuhan teknologi yang semakin inovatif dalam penciptaan media baru, aplikasi berbasis website sangat diminati di berbagai bidang, terutama

[*] Corresponding author : Erlan Wismunandar Darwis (erlanwismunandar_18@kharisma.ac.id)

***Diterima :*** *Januari, 2022* ***Disetujui :*** *Maret, 2002* ***Dipublikasikan :*** *Mei, 2022*

bagi para pelaku bisnis untuk mengembangkan usahanya. internet sangat menguntungkan, karena produk bisnis tidak hanya dapat ditemukan di kota dan negara, tetapi juga di dalam dan luar negeri. Disamping itu, bahaya akan serangan siber menjadi sangat beresiko baik untuk pelaku usaha maupun konsumen. Hal yang sama berlaku untuk RenovAction.

RenovAction merupakan aplikasi berbasis web yang bertujuan untuk mempermudah pembangunan tempat tinggal serta melakukan renovasi, dengan fitur pemanggilan jasa tukang maka orang-orang tidak harus keluar rumah lagi untuk mencari jasa tukang.

Keamanan pada website RenovAction belum teruji sebelumnya, ini memungkinkan terjadinya serangan pada website yang akan merugikan data pengguna dimana informasi sensitif seperti data pribadi pengguna dapat disebarluaskan membuat pengguna website menjadi tidak merasa aman saat akan melakukan transaksi akibatnya kurangnya kredebilitas terhadap website RenovAction akan menjadi masalah yang serius bagi penulis.

Demi memberikan rasa aman terhadap user maka keamanan pada RenovAction harus diuji untuk mencegah serangan dan intrusi dari peretas. Menurut Sunardi pengetesan keamanan ini dapat dicapai melalui pengujian yang dapat dilakukan dengan berbagai cara. Salah satunya adalah *self test* menggunakan *Open Web Application Security Project* (*OWASP) Top 10*[1]*. OWASP)* adalah organisasi nirlaba yang bertujuan untuk meningkatkan keamanan perangkat lunak. OWASP bekerja pada model "komunitas terbuka", di mana siapa pun dapat berpartisipasi dan berkontribusi pada proyek, acara, obrolan online, dan lainnya. Prinsip dasar *OWASP* adalah semua materi dan informasi di situs web ini gratis dan mudah diakses oleh siapa saja[2].

*OWASP Top 10* OWASP Top 10 adalah dokumen online di situs web *OWASP* yang memberikan penilaian dan pedoman untuk menyelesaikan 10 risiko keamanan teratas untuk aplikasi web. Laporan ini didasarkan pada konsensus para pakar keamanan di seluruh dunia[3]. OWASP Top 10 sendiri memiliki banyak versi seiring dengan perkembangan waktu dan diantara semua versi *OWASP Top 10* penulis memilih versi 2017 dimana pada saat penelitian ini dilakukan versi *OWASP TOP 10* 2021 masih belum dirilis dan pada versi OWASP Top 10 2017 masih relevan digunakan pada penelitian ini dengan menggunakan metode *self test* menggunakan tool *OWASP Zap*.

Metode *self test* adalah sebuah pengujian dimana penulis memposisikan diri sebagai pengetes pada web RenovAction[4], serta penggunaan *security tool* yakni: *Zed Attack Proxy(ZAP)* dapat membantu penulis menerapkan metode *self test.* Adetya Dewantoro memaparkan bahwa *OWASP Zap* adalah alat pemindai kerentanan yang dikembangkan oleh organisasi *OWASP*. Alat ini terus berkembang dan karenanya merupakan proyek *OWASP* yang paling aktif. Alat ini bersifat *open source*, sehingga siapa pun dapat mengembangkannya[5].

Selain *Tools OWASP Zed Attack Proxy (ZAP)* adapun *Acunetix Vulnerability Scanner*. *Acunetix Web Vulnerability Scanner (WVS)* adalah sebuah pemindai kerentanan yang secara otomatis memindai aplikasi web dengan memeriksa kerentanan seperti *SQL injection, cross-site scripting (XSS),* dan peretasan lain yang dieksploitasi oleh kerentanan. Acunetix WVS

dapat diakses melalui browser web dan dapat memindai situs web atau aplikasi web apa pun yang menggunakan protokol HTTP maupun HTTPS.

Diantara berbagai macam serangan penulis memilih dua macam serangan yang akan diuji antara lain: *Cross-Site Scripting & Sql Injection*. Yulianingsih menyatakan bahwa *Cross-Site Scripting* jenis kejahatan keamanan aplikasi yang memanfaatkan kelalaian pengguna melalui Meminta atau mendorong penggunaan email atau Uniform Resource Locator (URL) yang digunakan oleh penyerangu[6]. Alex Sandro Irawan, Eko Sakti Pramukantoro, dan Ari Kusyanti mengatakan *Sql Injection* adalah teknik serangan yang dapat mengeksploitasi sintaks SQL dengan memanfaatkan kerentanan basis data untuk melakukan bypass login, merusak data, dan menyuntikkan kode berbahaya[7].

## 2. METODE PENELITIAN

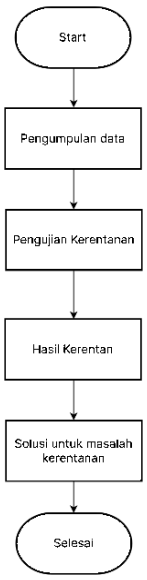
### 2.1. Jenis Data dan Sumber Data

Penelitian ini menggunakan jenis data kualitatif, penulis memperoleh sumber data tersebut dari data priemer dan sekuder.

1. Pada data primer penulis memperolehnya dari studi literatur melalui jurnal, e-book, dan dokumentasi pada situs penyedia *tools* keamanan.
2. Pada data sekunder penulis melakukan *self test* sebagai langkah pengujian pada web RenovAction untuk menegatahui tingkat kerentanannya.

### 2.2. Tahapan Penelitian

Dalam melakukan penelitian diperlukan beberapa tahap-tahap yang akan dilakukan agar penelitian ini lebih terstruktur dimulai dengan melakukan pemindaian secara otomatis yang bertujuan mendeteksi ada tidaknya celah keamanan berupa *SQL Injection* maupun *Cross-Site Scripting(XSS)*. Setelah proses pemindaian otomatis dilakukan dan menujukkan adanya celah keamanan yang dimaksud maka pengujian pada web RenovAction telah selesai, namun apabila tidak ditemukan maka penulis melanjutkan pengujian dengan menggunakan metode manual test berdasarkan *OWASP Top 10*.



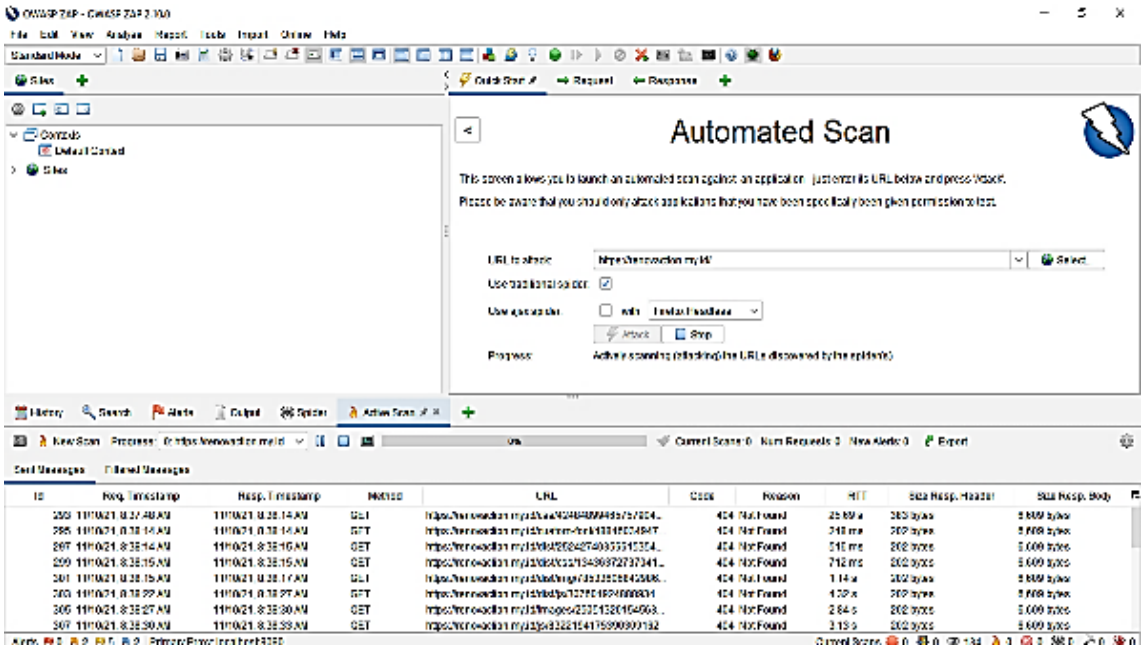

Gambar 1. Tahapan penelitian pada web RenovAction

Gambar 1 adalah diagram alur yang menunjukkan langkah-langkah yang dilakukan oleh penulis dalam penelitian ini. Pengumpulan data adalah langkah awal untuk mempersiapkan instrumen serta metode yang digunakan dalam penelitian. Pengujian kerentanan adalah proses yang dilakukan untuk menemukan kerentanan dalam web RenovAction. Setelah mengetahui kerentanan web RenovAction, langkah selanjutnya adalah menunjukkan kerentanan yang ada dengan menampilkan hasil uji kerentanan. Langkah terakhir dalam penelitian ini adalah memberikan solusi untuk mengatasi kerentanan yang terjadi pada web RenovAction.

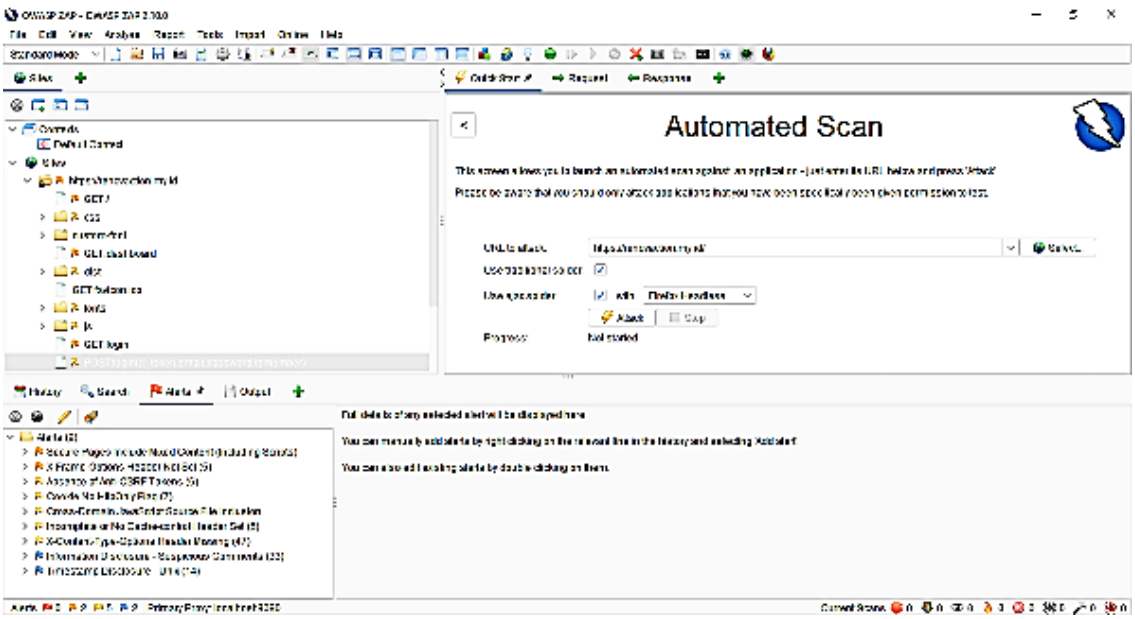
## 3. HASIL DAN PEMBAHASAN

### 3.1. Analisis Kondisi Awal

Untuk mengetahui kondisi awal web RenovAction penulis menggunakan metode *discovery phase* dengan cara melakukan *automated scan* menggunakan *OWASP ZAP* pada web RenovAction. Pada Gambar 2 proses *automated scan* menggunakan *traditional spider* dan dikombinasikan dengan *ajax spider* untuk mendeteksi kerentanan pada *single page apllication*. Dengan cara memasukkan url: https://renovaction.my.id/ kemudian mencentang *traditional spider* dan *ajax spider*.




Gambar 2. Proses *automated scan* menggunakan *OWASP ZAP*




Gambar 3. Hasil automated scan menggunakan *OWASP ZAP*

Pada Gambar 3 merupakan hasil dari *automated scan* yang memerlukan waktu sekitar 15 menit untuk proses pemindaiannya, dan terdapat 9 macam kerentanan. Daftar kerentan dapat dilihat pada Tabel 1.

Tabel 1. Laporan hasil kerentanan pada web RenovAction

| Jenis kerentanan | Url | Risk | Confidence |
|---|---|---|---|
| Secure Pages Include Mixed Content | https://renovaction.my.id/ | Medium | Medium |
| X-Frame-Options Header Not Set | https://renovaction.my.id/ | Medium | Medium |
| Absence of Anti-CSRF Tokens | https://renovaction.my.id/login | Low | Medium |
| Cookie No HttpOnly Flag | https://renovaction.my.id/ | Low | Medium |
| Cross-Domain JavaScript Source File Inclusion | https://renovaction.my.id/ | Low | Medium |
| Incomplete or No Cache-control Header Set | https://renovaction.my.id/ | Low | Medium |
| X-Content-Type-Options Header Missing | https://renovaction.my.id/ | Low | Medium |
| Information Disclosure - Suspicious Comments | https://renovaction.my.id/dashboard | Informational | Low |
| Timestamp Disclosure - Unix | https://renovaction.my.id/css/bootstrap.min.css | Informational | Low |

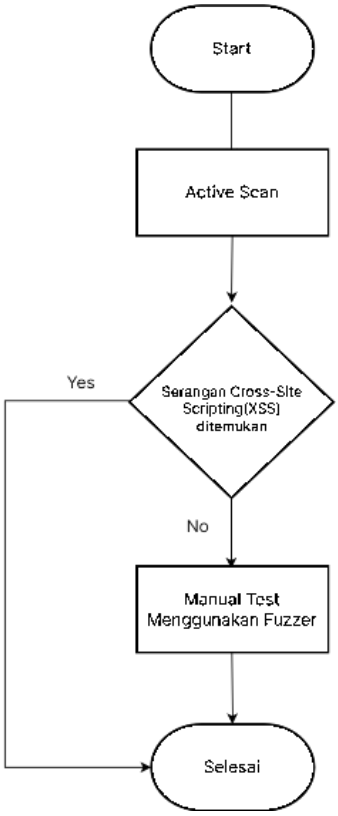
Padal Tabel 1 merupakan daftar kerentanan yang telah ditemukan pada web RenovAction, diantaranya: 2 kerentanan memiliki *medium risk*, 5 kerentanan memiliki *low risk,*dan 2 kerentanan tersisa memiliki *informational status.*

.

### 3.2. Pengujian kerentanan *Cross-Site Scripting(XSS)*

#### A. Metode *Active Scan Rules*

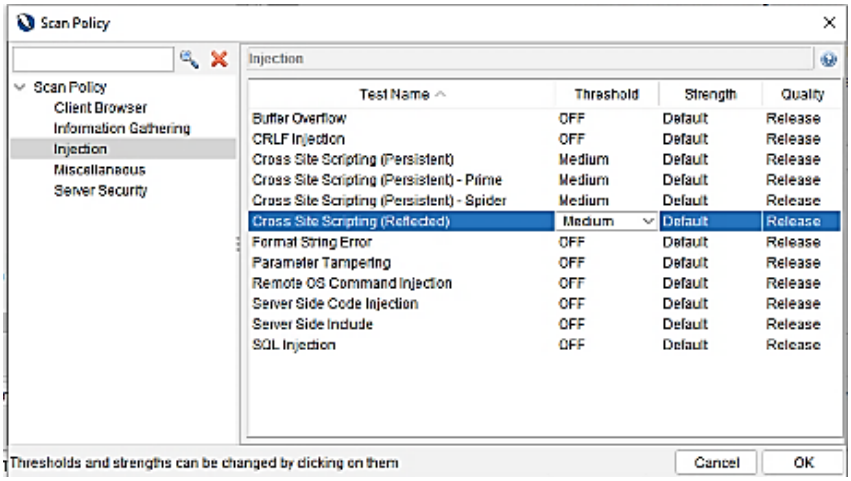
Setelah melakukan analisis kondisi awal dengan menerapkan metode *discovery phase*, selanjutnya penulis melakukan pengujian kerentanan menggunakan metode *active scan rules* sebagai acuan untuk melakukan pengetesan lebih lanjut terhadap serangan *Cross-Site Scripting(XSS)*[8].alur pengetesan dapat dilihat pada Gambar 4.




Gambar 4. Alur pengetesan serangan *Cross-Site Scripting(XSS)*

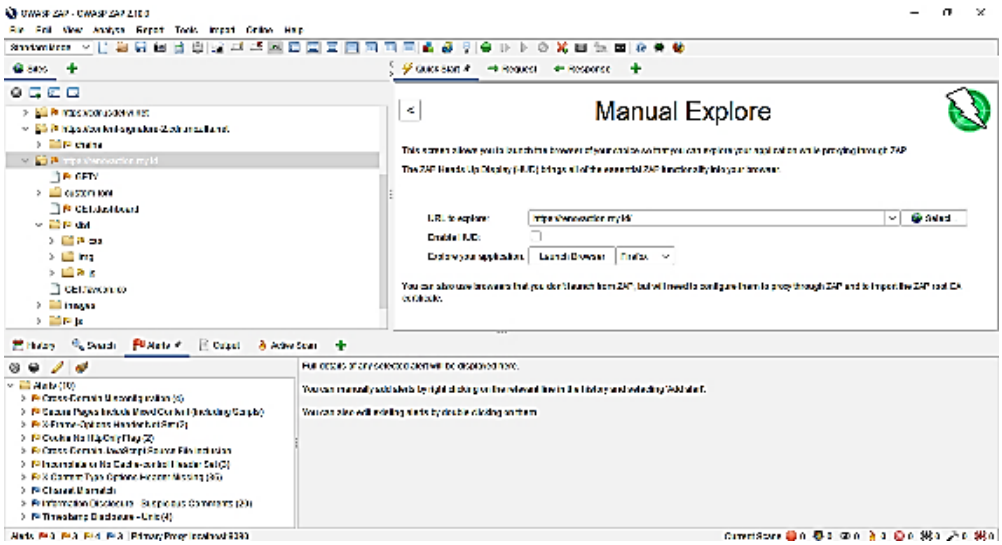
Gambar 4 merupakan *flowchart* pengetesan untuk serangan *Cross-Site Scripting(XSS)* dimana apabila pada proses active scan serangan *Cross-Site Scripting(XSS)* tidak ditemukan maka proses pengetesan akan dilanjutkan dengan metode *manual test* dengan menggunakan *Fuzz* dan jika kerentanan yang diuji tidak ditemukan, maka web RenovAction bebas dari serangan Cross-Site Scripting(XSS).

Proses pertama penulis memilih opsi *manual discovery* dan memasukkan url:https://renovaction.my.id/ lalu mengklik opsi *lunch browser(firefox)* pada aplikasi *Zed Attack Proxy.* Setelah web RenovAction selesai dimuat, dilakukan pengujian active scan. Hasil Pengujian ditunjukkan pada Gambar 5.




Gambar 5. Penambahan *Scan Policy*

Sebelum melakukan *active scan* pada web RenovAction penulis menambahkan *scan policy* untuk pengetesan serangan *Cross-Site Scripting(XSS)*, kemudian pada url login RenovAction penulis melakukan active scan dengan menggunakan *scan policy* yang sudah diatur untuk menguji serangan *Cross-Site Scripting(XSS)*[9].



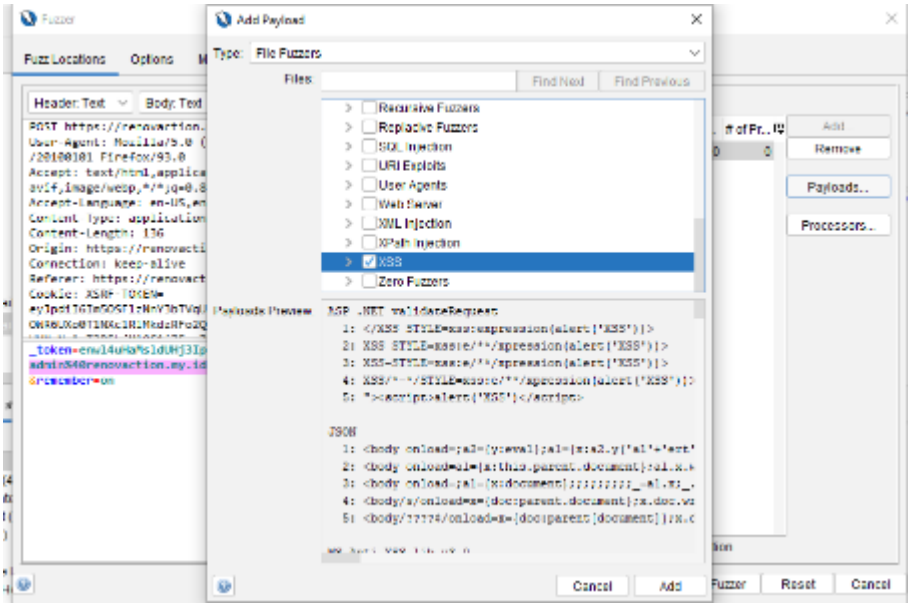

Gambar 6. Hasil *Active Scan*

Setelah menerapkan metode *active scan* dan menambahkan *scan policy* untuk menguji kerentanan *Cross-Site Scripting(XSS)* hasil dari pengetesan pada Gambar 6 menunjukkan bahwa tidak ditemukannya kerentanan pada *Cross-Site Scripting(XSS)*, namun mendeteksi 1 kerentanan dengan *risk medium*.

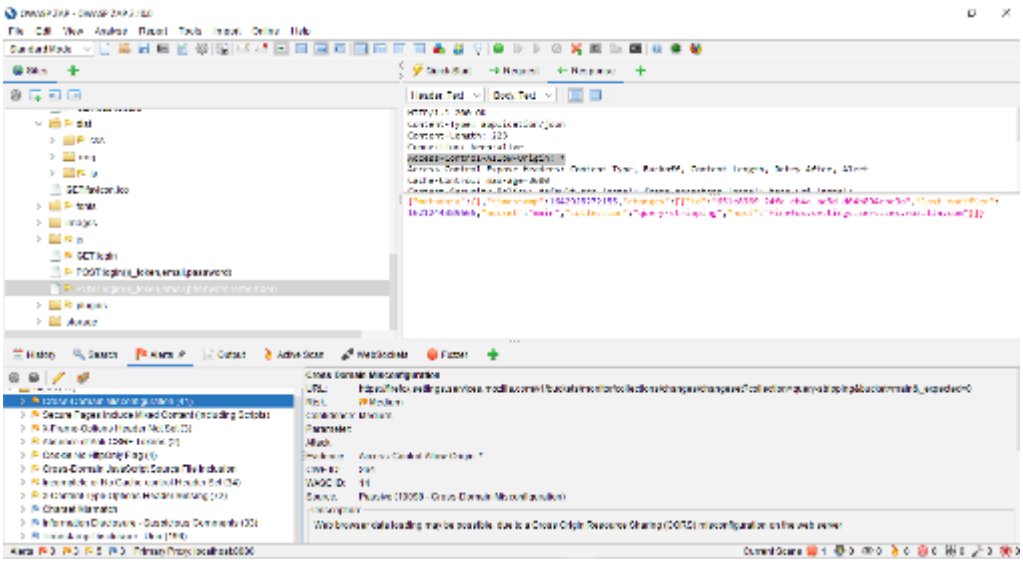
## B. Metode Fuzzing

Tahapan ini menjadi tahapan akhir untuk pengujian akan kerentanan *Cross-Site Scripting(XSS)* dimana pada tahap ini dilakukan *manual explore* dengan mengirimkan banyak data yang tidak valid ke url RenovAction. Dengan memilih post login kemudian menyeleksi username admin dan menambahkan payload bertipe file fuzzers dengan mencentang XSS[10]. Proses ditunjukkan pada Gambar 7.

Setelah melakukan pengujian secara manual menggunakan fuzzer ditemukan satu kerentanan yang bertipe *informational* dan untuk kerentanan Cross-Site Scripting(XSS) tidak ditemukan. Hasil test dengan fuzzer ditunjukkan pada Gambar 8.




Gambar 7. File fuzzers untuk pengujian Cross-Site Scripting(XSS)

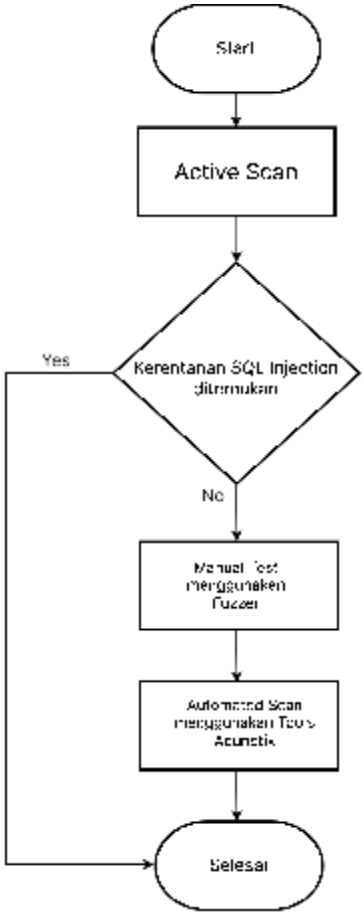


Gambar 8. Hasil manual test dengan fuzzer

### 3.3. Pengujian kerentanan *SQL Injection*

### A. Metode *Active Scan Rules*

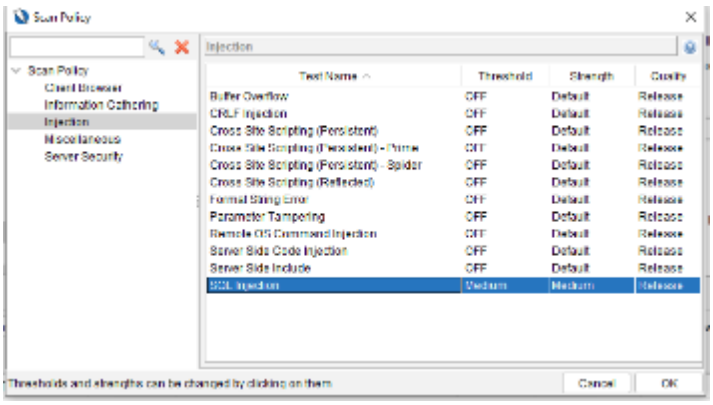
Pada tahap ini penulis menguji kerentanan akan serangan *SQL Injection* menggunakan metode *active scan rules*, kemudian dilanjutkan dengan metode *advanced SQL* Injection menggunakan tool *SQLmap* dan untuk memastikan bahwa jika tidak ditemukannya resiko akan SQL Injection maka penulis akan melakukan automated scan menggunakan tool Acunetix. Penulis menggunakan tools Acunetix karena pada penelitian yang telah dilakukan oleh JASH MATHEW menjabarkan jika Acunetix dapat memindai seluruh kerentanan pada SQL Injection[11].



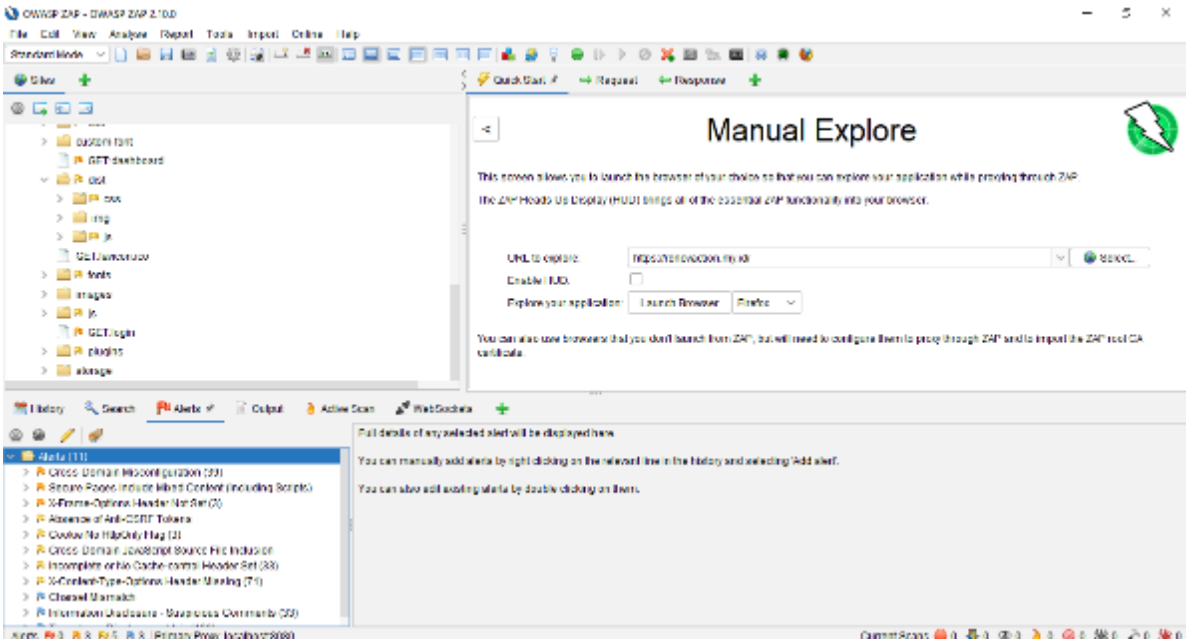

Gambar 9. *Flowchart* pengujian *SQL Injection*

Alur pengetesan untuk serangan *SQL Injection* hampir mirip dengan alur pengetesan pada serangan *XSS.* Perbedanya ada pada penambahan pemindaian yang dilakukan dimana apabila pada proses active scan serangan *SQL Injection* tidak ditemukan maka proses pengetesan akan dilanjutkan dengan metode manual test dengan menggunakan *Fuzzer* dan terakhir menggunakan tools acunetix untuk memastikan bahwa web RenovAction terbebas dari serangan *SQL Injection*. Tahapan pengujian SQL Injection ditunjukkan pada Gambar 9.

Seperti pengujian sebelumnya ditambahkan *scan policy* untuk menguji kerentanan SQL Injection kemudian dilakukan active scan dengan menambahkan scan policy yang telah diatur pada url RenovAction. Prosesnya ditunjukkan pada Gambar 10.




Gambar 10. *Scan Policy* untuk mengetes serangan *SQL Injection*

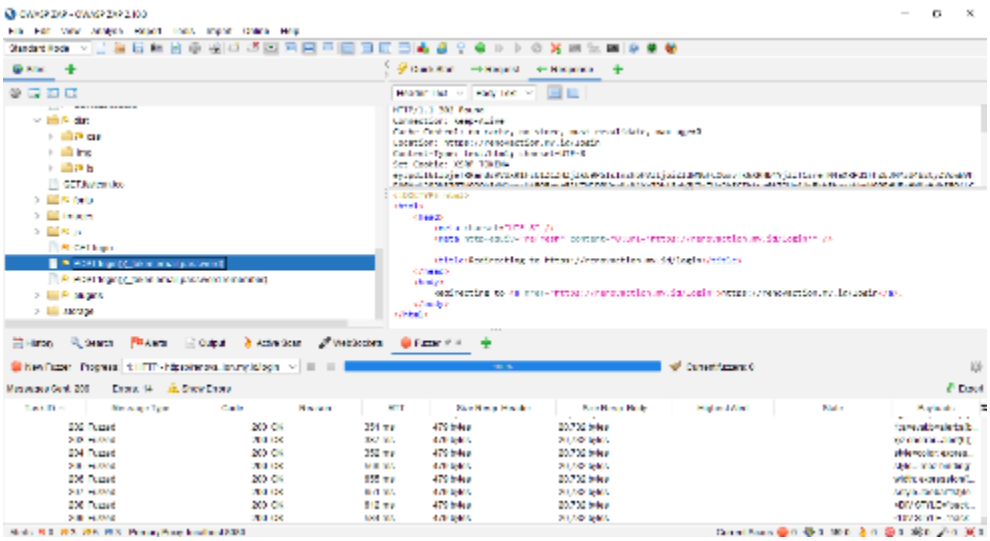



Gambar 11. Hasi active scan SQL Injection

Hasil yang diperoleh dari active scan menunjukkan kerentanan SQL Injection tidak ditemukan, maka pengujian dilanjutkan ke tahap manual test menggunakan fuzzer. Hasil dari pengetesan untuk SQL Injection ditunjukkan pada Gambar 11.

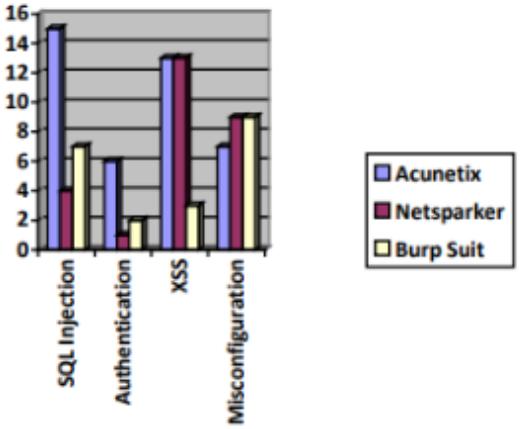
## B. Metode Fuzzing

Pada tahapan ini penulis menggunakan fuzzer dengan menambahkan payload bertipe file fuzzer dengan mencentang berbagai serangan SQL Injection. Untuk hasil pengujian Dapat dilihat pada Gambar 12.



Gambar 12. Hasil pengetesan SQL Injection menggunakan metode fuzzing

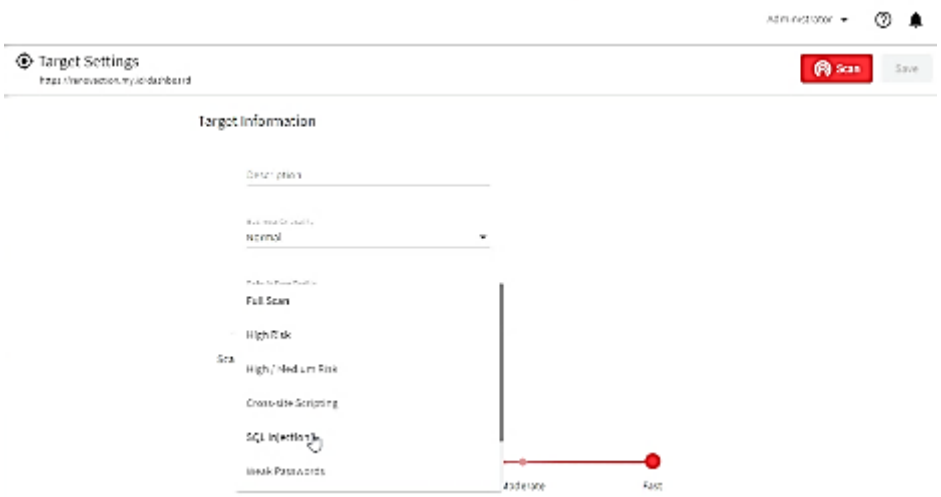
Setelah menerapkan pengujian manual berupa fuzzer, tidak ditemukan adanya kerentanan SQL Injection. Langkah selanjutnya adalah melakukan scan menggunakan tool acunetix.




Gambar 13. Grafik *Tools* Keamanan
Sumber: JASH MATHEW, 2016

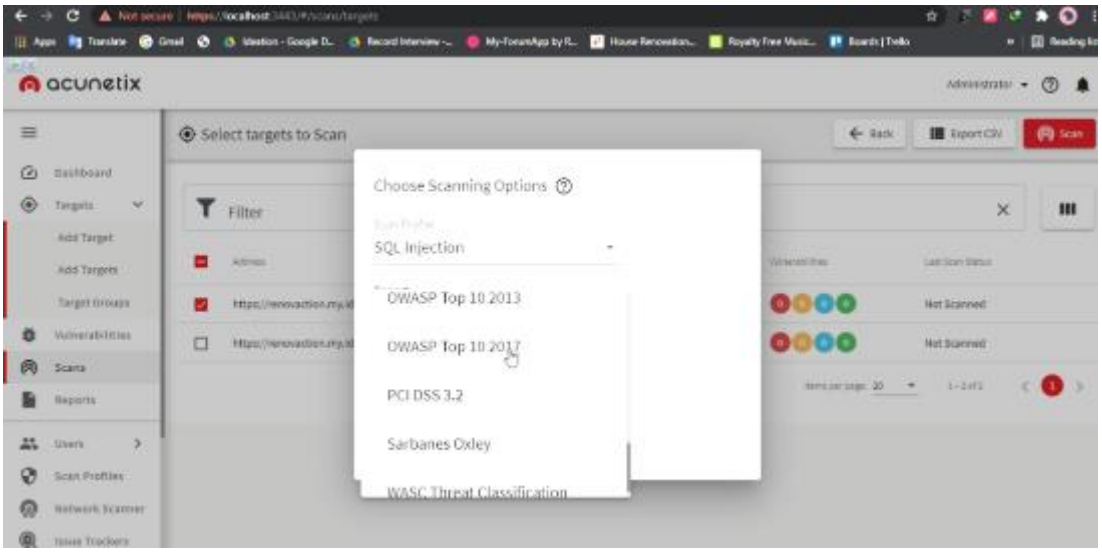
Hasil pemindaian serangan *SQL Injection* yang menggunakan aplikasi *Acunetix* pada Gambar 13 menunjukkan jika *Acunetix* dapat menemukan tiap celah kerentanan pada *SQL Injection*.

Langkah awal pemindaian menggunakan tool Acunetix adalah dengan menambahkan new scan kemudian pada pengaturan target scan ditunjukkan pada Gambar 14, diatur beberapa hal sebelum melakuan pemindaian dianataranya mengubah metode pemindaian yang awalnya *full scan* menjadi *SQL Injection.* Setelah itu memasukkan url RenovAction beserta *username* dan *password* user dilanjutkan dengan mengatur kecepatan *scan* beserta *browser yang akan digunakan* dan yang terakhir menentukan report hasil scan dengan memilih OWASP Top 10 2017.



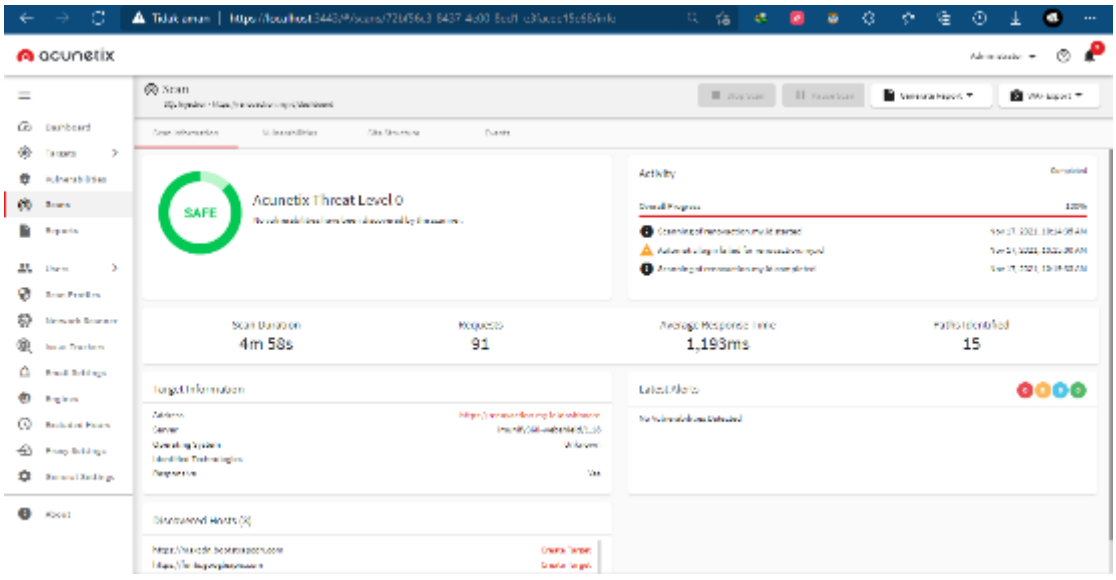

Gambar 14. *Target setting* dengan *SQL Injection*

Tujuan diubahnya Full Scan menjadi SQL Injection pada menu *target infromation* agar pemindaian hanya berfokus pada kerentanan SQL Injetion.




Gambar 15**.** Laporan Pemindaian Diatur Menjadi OWASP Top 10 2017

Dengan diaturnya laporan hasil pemindaian berdasarkan OWASP Top 10 2017, ditunjukkan pada Gambar 15, dari pengujian kerentanan SQL Injection dapat memudahkan dalam mengumpulkan data pengujian.




Gambar 16. Hasil Pemindaian Menggunakan Acunetix

Pada Gambar 16. Merupakan hasil pemindaian menggunakan acunetix dengan pengaturan pemindaian yang secara spesifik memindai kerentanan *SQL Injection,* tidak menemukan kerentanan SQL Injection pada url RenovAction.

### 3.4 Laporan Hasil Pengujian

Setelah menguji dua jenis kerentanan pada tools OWASP ZAP dan Acunetix penulis tidak menemukan adanya celah keamanan dari dua jenis yang telah diuji akan tetapi penulis menemukan 2 kerentanan pasif dengan resiko medium dan informational. Tabel 2 menunjukkan daftar celah keamanan pada web RenovAction.

Tabel 2. Laporan kerentanan pada web RenovAction

| NO. | Kerentanan | Solusi |
|---|---|---|
| 1. | *Cross-Domain Misconfiguration* | Pastikan bahwa data sensitif tidak tersedia dalam cara yang tidak terotentikasi (menggunakan alamat IP whitelisting, misalnya). Atur tajuk HTTP Access-Control-Allow-Origin' HTTP ke set yang lebih ketat dari domain, atau Hapus semua tajuk CORS seluruhnya, untuk memungkinkan peramban web untuk menegakkan kebijakan asal yang sama (SOP) dalam cara yang lebih terbatas. |
| 2. | *Secure Pages Include Mixed Content (Including Scripts)* | Sebuah halaman yang tersedia di atas SSL/TLS harus terdiri sepenuhnya dari isi yang dikirim ke SSL / TLS. Halaman tidak boleh memuat isi apapun yang diteruskan melalui HTTP yang tidak terenkripsi. termasuk isi dari situs pihak ketiga. |
| 3. | *X-Frame-Options Header Not Set* | Dengan menggunakan perintah SAMEORIGIN yang memungkinkan untuk sebuah halaman meloading bingkai dalam halaman, sesuai dengan halaman asli atau bisa juga menggunakan perintah DENY yang secara garis besar menonaktifkan sepenuhnya loading halaman dalam bingkai, telepas dari situs apapun itu. namun dapat menghentikan banyak fungsi pada website |
| 4. | *Absence of Anti-CSRF Tokens* | Tahap: arsitektur dan desain<br>menggunakan library yang telah diperiksa atau kerangka kerja yang tidak memungkinkan kelemahan ini terjadi atau menyediakan konstruksi yang membuat kelemahan ini lebih mudah untuk dihindari. Sebagai contoh, gunakan paket anti-CSRF seperti OWASP CSRFGuard.<br>Tahap: implementasi<br>memastikan bahwa aplikasi bebas dari masalah cross site scripting, karena kebanyakan pertahanan CSRF dapat dilewati dengan menggunakan skrip pengendali penyerang. |
| 5. | *Cookie No HttpOnly Flag* | Jika sebuah peramban tidak mendukung HttpOnly dan usaha situs web untuk mengatur sebuah cookie HttpOnly, HttpOnly flag akan diabaikan oleh peramban, sehingga membuat sebuah cookie tradisional, skrip dapat diakses. Akibatnya, cookie menjadi rentan terhadap pencurian modifikasi oleh skrip berbahaya. |
| 6. | *Cross-Domain JavaScript Source File Inclusion* | Pastikan berkas sumber JavaScript dimuat dari sumber yang terpercaya, dan sumber tidak dapat dikontrol dari pengguna apliksi langsung |
| 7. | *Incomplete or No Cache-control Header Set* | Bila memungkinkan pastikan cache-control HTTP header diset dengan tanpa-cache, no-store, harus-revalidate. |
| 8. | *X-Content-Type-Options Header Missing* | Pastikan bahwa server aplikasi/web mengatur header Content-Type secara tepat, dan itu menentukan header X-Content-Type-Options ke 'nosniff' untuk semua halaman web. Jika memungkinkan, pastikan bahwa pengguna langsung menggunakan standart- |

| | | |
|---|---|---|
| | | compliant dan peramban web modern yang tidak melakukan MIME-sniffing sama sekali, atau yang dapat diarahkan oleh web aplikasi/server untuk tidak melakukan MIME-sniffing. |
| 9. | *Charset Mismatch* | Mengatur UTF-8 untuk semua isi teks dalam header HTTP dan meta tag dalam HTML atau deklarasi pengkodean dalam XML. |
| 10. | *Information Disclosure - Suspicious Comments* | Menghapus semua komentar yang mengembalikan informasi yang dapat memberikan peretas akses dan memperbaiki setiap masalah berdasarkan rujukan mereka. |
| 11. | *Timestamp Disclosure - Unix* | Verifikasi secara manual bahwa data stempel waktu tidak sensitif dan bahwa data tidak dapat digabungkan untuk mengungkapkan pola yang disalahgunakan. |

## 4. KESIMPULAN

Setelah melakukan pengujian untuk kedua jenis serangan yaitu Cross-Site Scripting(XSS) dan SQL Injection dapat disimpulkan jika web RenovAction tidak memliki celah keamanan pada kedua jenis serangan tersebut akan tetapi hanya memliki kerentanan dasar saja yang berjumlah 11 resiko serta solusi penanganannya sebagai berikut:

I. *Cross-Domain Misconfiguration*
   Memastikan bahwa data sensitif tidak tersedia dengan cara yang tidak diautentikasi
II. *Secure Pages Include Mixed Content (Including Scripts)*
   Halaman yang tersedia melalui SSL / TLS harus terdiri sepenuhnya dari konten yang dikirimkan melalui SSL / TLS
III. *X-Frame-Options Header Not Set*
   Mengatur agar X-Frame-Options header hanya dapat diakses pada alamat IP tertentu
IV. *Absence of Anti-CSRF Tokens*
   menambahkan token yang valid sementara ke pengiriman formulir apa pun pada aplikasi
V. *Cookie No HttpOnly Flag*
   Pastikan bahwa *HttpOnly flag* diatur untuk semua cookie.
VI. *Cross-Domain JavaScript Source File Inclusion*
   Pastikan file sumber *JavaScript* dimuat hanya dari sumber tepercaya, dan sumber tidak dapat dikontrol oleh *end users* pada aplikasi
VII. *Incomplete or No Cache-control Header Set*
   Bila memungkinkan pastikan *header HTTP cache-control* diatur dengan *no-cache, no-store,* tervalidasi ulang
VIII. *X-Content-Type-Options Header Missing*
   Pastikan bahwa server Aplikasi / web menetapkan header Tipe konten dengan tepat, dan itu menetapkan header *X-Content-Type-Options* ke ' *no sniff*' untuk semua halaman web.
IX. *Charset Mismatch*
   Mengatur *UTF-8* untuk semua isi teks dalam *header HTTP* dan meta tag dalam *HTML* atau deklarasi pengkodean dalam *XML.*
X. *Information Disclosure - Suspicious Comments*
   *Menghapus semua komentar yang mengembalikan informasi yang dapat memberikan peretas akses dan memperbaiki setiap masalah berdasarkan rujukan peretas.*

XI. *Timestamp Disclosure – Unix*
Verifikasi secara manual bahwa *Timestamp Disclosure* tidak sensitif dan membuktikan jika data tidak dapat digabungkan bertujuan untuk mengungkapkan pola yang disalahgunakan

**DAFTAR PUSTAKA**

[1] A. P. Dewanto, “Penetration Testing padaDomain uii.ac.id Menggunakan OWASP 10,” Https://Dspace.Uii.Ac.Id/, 2018, [Online]. Available: https://dspace.uii.ac.id/bitstream/handle/123456789/11281/13523025-Adetya Putra D-laporan skripsi.pdf?sequence=1&isAllowed=y.

[2] A. S. Irawan, E. S. Pramukantoro, and A. Kusyanti, “Pengembangan Intrusion Detection System Terhadap SQL Injection Menggunakan Metode Learning Vector Quantization,” J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya, vol. 2, no. 6, pp. 2295–2301, 2018.

[3] B. Ghozali, M. Teknik, I. Universitas, and A. Yogyakarta, “Mendeteksi Kerentanan Keamanan Aplikasi Website Menggunakan Metode Owasp ( Open Web Application Security Project ) untuk Penilaian Risk Rating,” pp. 264–275.

[4] J. Mathew, C. Joshi, and U. K. Singh, “Performance Evaluation of Web Application Security Scanners for More Effective Defense Performance Evaluation of Web Application Security Scanners for More Effective Defense,” 2016.

[5] M. Yunus, “Analisis Kerentanan Aplikasi Berbasis Web Menggunakan Kombinasi Security Tools Project Berdasarkan Framework Owasp Versi 4,” *J. Ilm. Inform. Komput.*, vol. 24, no. 1, pp. 37–48, 2019, doi: 10.35760/ik.2019.v24i1.1988.

[6] Sunardi, I. Riadi, and P. A. Raharja, “Vulnerability analysis of E-voting application using open web application security project (OWASP) framework,” Int. J. Adv. Comput. Sci. Appl., vol. 10, no. 11, pp. 135–143, 2019, doi: 10.14569/IJACSA.2019.0101118.

[7] The OWASP Foundation, “About the OWASP Foundation,” OWASP foundation, 2022. https://owasp.org/about/.

[8] The OWASP Foundation, “Active Scan Rules,” *zaproxy*, 2017. https://www.zaproxy.org/docs/desktop/addons/active-scan-rules/.

[9] The OWASP Foundation, “Payloads dialog,” *zaproxy*, 2017. https://www.zaproxy.org/docs/desktop/addons/fuzzer/payloads/.

[10] The OWASP Foundation, “Scan Policy,” *zaproxy*, 2017. https://www.zaproxy.org/docs/desktop/start/features/scanpolicy/.

[11] Y. Yulianingsih, “Melindungi Aplikasi dari Serangan Cross Site Scripting dengan Metode Metacharacter,” J. Nas. Teknol. dan Sist. Inf., vol. 3, no. 1, pp. 83–88, 2017, doi: 10.25077/teknosi.v3i1.2017.83-88.