SENAFTI SEMINAR NASIONAL MAHASISWA FAKULTAS TEKNOLOGI INFORMASI

5th Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)
7 September 2024 – Jakarta, Indonesia
Volume 3, Nomor 2, September 2024 - ISSN 2962-8628 (*online*)

DETEKSI CELAH KEAMANAN XSS PADA WEBSITE DENGAN METODE *BRUTE FORCE*

Ailza Zandra[1], Mardi Hardjianto[2*]

[1,2] Teknik Informatika, Fakultas Teknologi Informasi, Universitas Budi Luhur, Jakarta, Indonesia

Email: [1]2011501067@student.budiluhur.ac.id, [2*]mardi.hardjianto@budiluhur.ac.id
(* : corresponding author)

Abstrak-Kerentanan keamanan aplikasi web, khususnya *Cross-Site Scripting*(XSS), menjadi ancaman bagi integritas dan kerahasiaan data pengguna. Penelitian ini bertujuan untuk mengembangkan metode deteksi dini serangan XSS menggunakan teknik metode penetration testing dengan pendekatan *Brute Force*. Untuk menerapkan metode ini, dibuat sebuah alat deteksi kerentanan XSS bernama Ecaxss dengan memanfaatkan 99 *payload* dari PortSwigger yang diinjeksikan ke sebuah situs untuk mendeteksi kerentanan XSS. Ecaxss dirancang untuk melakukan *scanning* pada situs web secara menyeluruh dan efektif. Hasil penelitian pada situs web http://testphp.vulnweb.com/ menunjukkan bahwa alat ini memiliki tingkat keberhasilan yang tinggi dalam mendeteksi serangan XSS. Dengan durasi *scanning* rata-rata 1 menit 14 detik, Ecaxss berhasil mendeteksi 77 dari 99 *payload* yang di-inject ke situs yang diuji. Pengujian lebih lanjut pada berbagai situs web lainnya juga menunjukkan konsistensi hasil yang serupa, membuktikan bahwa metode *Brute Force* yang digunakan oleh Ecaxss efektif untuk mendeteksi dan melaporkan kerentanan XSS. Keberhasilan alat ini dalam mendeteksi serangan XSS menegaskan pentingnya pengembangan alat keamanan yang baik untuk perlindungan data pengguna. Dengan demikian, Ecaxss berpotensi menjadi alat keamanan yang penting dalam dunia keamanan siber, khususnya dalam mendeteksi berbagai skenario serangan XSS. Penelitian ini berkontribusi pada peningkatan keamanan aplikasi web dengan menyediakan solusi efektif untuk mendeteksi dan mengatasi ancaman XSS yang semakin berkembang.

Kata Kunci: XSS, Penetration Testing, *Brute Force*, Keamanan Web, *Payload*, Cross-Site Scripting

DETECTION OF XSS VULNERABILITIES ON WEBSITES USING BRUTE FORCE METHOD

Abstract-*Web application vulnerabilities, particularly Cross-Site Scripting(XSS), pose significant threats to the integrity and confidentiality of user data. This study aims to develop an early detection method for XSS attacks using penetration testing techniques with a Brute Force approach. To implement this method, a tool named Ecaxss was created to detect XSS vulnerabilities, utilizing 99 payloads from PortSwigger, which are injected into various websites for testing. Ecaxss is designed to perform comprehensive and effective scanning on web applications. The research conducted on the website http://testphp.vulnweb.com/ demonstrated that Ecaxss has a high success rate in detecting XSS attacks. With an average scanning duration of 1 minute and 14 seconds, Ecaxss successfully detected 77 payloads injected into the tested website. Further testing on various other websites also showed consistent results, proving that the Brute Force method used by Ecaxss is effective in detecting and reporting XSS vulnerabilities. The success of this tool in detecting XSS attacks underscores the importance of developing reliable security tools to protect user data. Ecaxss not only detects vulnerabilities quickly and accurately but also provides detailed reports that can be used to fix system weaknesses. Consequently, Ecaxss has the potential to become an essential security tool in the field of cybersecurity, particularly in detecting various XSS attack scenarios. This research contributes to enhancing web application security by providing an effective solution to detect and address the evolving threat of XSS.*

Keywords*: XSS, Penetration Testing, Brute Force, Web Security, Payload, Cross-Site Scripting*

1. PENDAHULUAN

Dalam era global saat ini, perkembangan Teknologi Informasi telah mengalami kemajuan yang sangat pesat, terutama berkat hadirnya internet yang mempermudah komunikasi antarindividu. Namun, kemudahan akses informasi ini juga membawa risiko penyalahgunaan data oleh pihak yang tidak bertanggung jawab. Oleh karena itu, keamanan jaringan menjadi sangat krusial[1]. Laporan tahunan pemantauan keamanan siber dari BSSN tahun 2021 mencatat 332 insiden keamanan siber yang menargetkan sektor pemerintah, ekonomi digital, dan sektor lainnya, dengan tiga jenis insiden terbanyak yaitu SQL *Injection*, XSS, dan informasi yang diungkapkan [2].

Kesalahan dalam penulisan kode aplikasi web sering dimanfaatkan untuk serangan seperti SQL *Injection*, Authentication, dan XSS. Statistik dari webappsec.org (diperbarui Januari 2010) menunjukkan bahwa XSS (43%) dan SQL *Injection* (6%) adalah serangan yang paling sering digunakan [3]. Kerentanan aplikasi web, terutama *Cross-Site Scripting* (XSS), memungkinkan penyerang menyisipkan kode berbahaya di halaman web, yang dapat

mengakibatkan manipulasi data, pencurian informasi, dan pengambilalihan akses. Kelemahan ini telah ada sejak tahun 1990 dan sering mempengaruhi banyak situs web [4].

Keberadaan kerentanan XSS ini memudahkan *hacker* untuk melancarkan serangan, memungkinkan mereka untuk memanipulasi data, mengubah tampilan web, mencuri data, dan bahkan mengambil alih hak akses pengelolaan aplikasi web. Serangan XSS terjadi ketika penyerang menggunakan aplikasi web untuk mengirimkan kode jahat dalam bentuk skrip yang dijalankan di sisi browser. Skrip ini bisa mendapatkan akses ke cookie, token sesi, atau data sensitif lainnya yang disimpan oleh browser, serta dapat mengubah konten halaman HTML [5]. Kehilangan data dan pencurian informasi penting dari perusahaan atau organisasi untuk kepentingan pribadi dapat menimbulkan kerugian bagi pihak lain. Misalnya, mengakses data pribadi suatu perusahaan atau lembaga untuk melakukan transaksi berlebihan yang dapat menghabiskan uang korban, menggunakan data dan akses untuk melakukan penipuan, dan lain sebagainya [6].

Penelitian sebelumnya telah mengusulkan metode seperti penggunaan metacharacter untuk mengatasi serangan XSS, namun masih memiliki keterbatasan dalam hal skalabilitas dan efektivitas di lingkungan yang dinamis. Pendekatan proaktif seperti pemindaian rutin dan deteksi dini juga perlu lebih dieksplorasi [7]. Penetration testing, sebagai salah satu metode pengujian keamanan, bertujuan untuk mengidentifikasi nilai risiko terkait potensi pelanggaran keamanan dengan mensimulasikan serangan nyata [8]. Teknik ini memungkinkan pengujian aplikasi secara aktif guna menemukan kerentanan keamanan tanpa mengetahui cara kerja aplikasi tersebut [9].

Penelitian ini mengusulkan strategi untuk mengatasi serangan XSS melalui pemindaian rutin dan deteksi dini menggunakan teknik *Brute Force*, dengan fokus pada penggunaan $_POST. Metode ini bertujuan untuk mendeteksi celah keamanan secara lebih mendalam dan mencegah terjadinya serangan XSS sebelum bisa dimanfaatkan oleh penyerang. Algoritma *Brute Force* digunakan untuk memeriksa setiap posisi string dalam teks mulai dari karakter awal hingga karakter akhir, memungkinkan deteksi lebih mendalam terhadap potensi celah keamanan [10].

Penelitian ini juga membandingkan efektivitas teknik *Brute Force* pada $_POST dengan metode lain yang sudah ada. Selain itu, penelitian ini menggunakan studi kasus nyata untuk menunjukkan bagaimana metode ini dapat secara efektif mendeteksi dan mencegah serangan XSS. Dengan demikian, penelitian ini memberikan kontribusi penting dalam meningkatkan keamanan aplikasi web, terutama dalam mencegah serangan XSS melalui metode $_POST.

2. METODE PENELITIAN

2.1 Pengumpulan Data

Penelitian ini membutuhkan beberapa data yang digunakan untuk memeriksa suatu website apakah terdapat kerentanan XSS atau tidak. Data yang dibutuhkan dalam penelitian ini adalah *payload* XSS dan alamat URL. Data alamat URL adalah target dari situs yang akan diperiksa celah keamanannya. Data *Payload* XSS ini nantinya akan digunakan untuk meng-inject suatu URL agar dapat mengetahui apakah terdapat celah keamanan XSS atau tidak. *Payload* XSS diperoleh dari situs Portswigger dan berjumlah 99 buah, dipilih karena mencakup berbagai skenario serangan XSS yang umum dan kompleks sehingga dapat memberikan gambaran menyeluruh tentang kerentanan yang ada. Target dari penelitian ini adalah situs http://testphp.vulnweb.com/, yang merupakan situs yang sering digunakan untuk pengujian keamanan web. Data *payload* XSS yang diperoleh ini akan digunakan oleh peneliti sebagai daftar *payload* untuk alat Ecaxss. Ecaxss adalah alat yang dikembangkan khusus untuk mendeteksi kerentanan XSS pada aplikasi web dengan menggunakan metode *Brute Force*. Dalam konteks ini, data *payload* yang telah dikumpulkan akan diujikan satu per satu pada formulir input di halaman web target. Setiap *payload* akan dikirimkan melalui metode POST, dan respons dari Server akan dianalisis untuk menentukan apakah *payload* tersebut berhasil dieksekusi, yang akan mengindikasikan adanya kerentanan XSS. Contoh data *payload* dapat dilihat pada Table 1 berikut ini.

Tabel 1. Data *Payload*

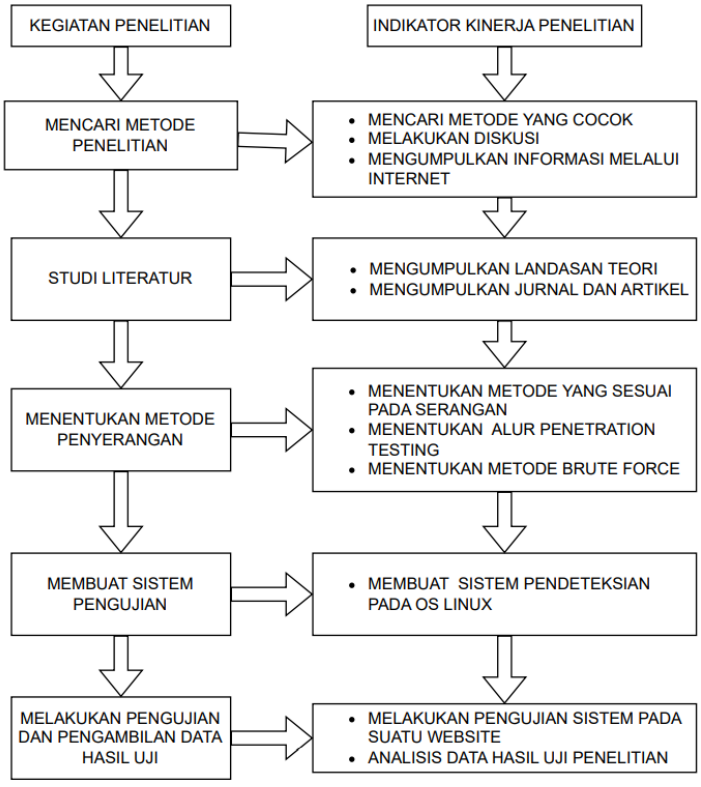
No	*Script*	Ket
1	<xss onafterscriptexecute=alert(1)><script>1</script>	Menampilkan alert dengan pesan "1" setelah script di dalam <script> selesai dieksekusi.
2	<xss onmouseover="alert(1)"style=display:block>test</xss>	Menjalankan JavaScript saat pengguna mengarahkan kursor ke elemen tersebut
3	<marquee onstart=alert(1)>XSS</marquee>	Menjalankan JavaScript ketika elemen marquee mulai bergerak. Bisa memicu tindakan yang tidak diinginkan saat halaman diakses.

4	test	Menjalankan JavaScript sebelum konten disalin oleh pengguna, berpotensi menyuntikkan kode atau mencuri data clipboard
5	<body onload=alert(1)>	Menjalankan kode JavaScript segera setelah halaman dimuat. Dapat digunakan untuk menampilkan pesan pop-up atau menjalankan script berbahaya.

2.2 Penerapan Metode Penelitian

Pada bagian ini, akan dibahas tahapan uji coba yang melibatkan alur penelitian menggunakan penetration testing pada sistem operasi Linux. Sistem ini dikembangkan dengan menggunakan bahasa pemrograman Python. Python adalah bahasa pemrograman yang serbaguna dan dapat digunakan untuk berbagai keperluan, seperti pengembangan aplikasi web, aplikasi desktop, Internet of Things (IoT), dan aplikasi lainnya. Python juga dapat terintegrasi dengan sistem basis data serta mampu membaca dan memodifikasi file, sehingga sering digunakan untuk prototyping atau pengembangan perangkat lunak secara cepat dan efektif. Selain itu, Python banyak dipilih oleh para peneliti karena kemampuannya dalam menangani data besar dan melakukan perhitungan matematika yang kompleks[11].

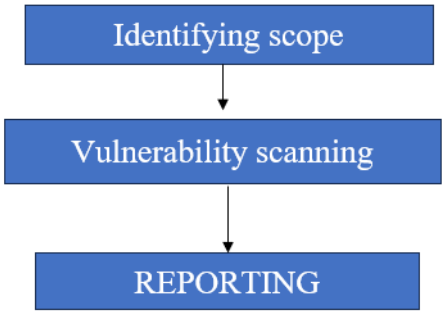
Langkah pertama dalam penelitian ini adalah menyusun daftar dan mencari data dari berbagai sumber untuk menentukan metode pencegahan yang tepat terhadap serangan *Brute Force*. Selanjutnya, pada tahap Studi Literatur, peneliti mengumpulkan landasan teori yang mendukung penelitian. Literatur yang digunakan mencakup jurnal, artikel, dan buku terkait metode *Brute Force*. Sebagai contoh, Mukaromah. pada tahun 2021 menjelaskan bahwa Algoritma *Brute Force* adalah algoritma yang berfungsi untuk mencocokkan string atau pola dengan seluruh teks [12]. Pada penelitian ini algoritma *Brute Force* berfungsi untuk mencoba semua data *payload* yang terdapat pada *payload*1.txt dan secara bertahap akan mencoba menembus pertahanan komputer. Tahap berikutnya adalah menentukan metode pendeteksian, di mana peneliti memilih metode *Brute Force* untuk mencegah serangan XSS. Setelah itu, peneliti akan membuat sistem pengujian menggunakan sistem operasi Linux, dan menjalankan pemindaian melalui terminal Linux. Terakhir, pada tahap pengujian dan pengambilan data, peneliti melakukan pemindaian terhadap sebuah situs web dan mengevaluasi hasilnya untuk memastikan apakah pemindaian berhasil. Alur penelitian ini dapat dilihat pada Gambar 1 berikut ini.



Gambar 1. Alur Penelitian

2.3 Implementasi Metode

Implementasi metode ini menggunakan alur pengujian penetrasi sebagai panduan untuk melakukan pengembangan sistem berikut dapat dilihat pada Gambar 2.



Gambar 2. Implementasi Metode

a. *Identifying scope*, Tahap pertama dalam proses pengujian ini berfungsi untuk menentukan ruang lingkup pengujian yang akan diuji. Penentuan ruang lingkup ini adalah langkah krusial yang memastikan bahwa semua aspek penting dari situs web yang akan diuji telah diidentifikasi dan diprioritaskan. Pada pengujian ini, peneliti memutuskan untuk menggunakan alamat URL (Uniform Resource Locator) http://testphp.vulnweb.com/ sebagai objek dari pengujian yang akan dilakukan. Pemilihan URL ini didasarkan pada karakteristik situs yang sering digunakan untuk tujuan pengujian keamanan, memungkinkan peneliti untuk menguji metode yang digunakan dalam kondisi yang mendekati kenyataan.
b. Vuln*erability Scanning*, Tahap ini berfungsi untuk mencari dan mengidentifikasi kerentanan pada URL situs web http://testphp.vulnweb.com/. Proses ini menggunakan alat khusus yang dirancang untuk mendeteksi berbagai jenis kerentanan keamanan. Alat yang digunakan dalam tahap ini yaitu Ecaxss. Ecaxss akan mengirimkan berbagai *payload* XSS yang telah disiapkan ke input-input pada situs web target. Setiap respons dari situs akan dipantau untuk mencari tanda-tanda kerentanan XSS, seperti eksekusi skrip yang tidak diinginkan atau perubahan pada halaman web yang menunjukkan adanya celah keamanan. Pada tahap ini, metode *Brute Force* diterapkan dengan mencoba mencocokkan setiap *payload* XSS dengan berbagai input pada situs web untuk mendeteksi kemungkinan kerentanan.
c. *Reporting*, Pada tahap ini, hasil dari *scanning* akan muncul dan akan dicatat durasi *scanning* dan akurasi *scanning*. Tahap pelaporan ini merupakan langkah kritis dalam siklus pengujian keamanan karena menyajikan temuan secara sistematis dan terstruktur, memungkinkan peneliti dan pihak terkait untuk memahami secara jelas status keamanan situs web target. Laporan ini akan mencakup jenis-jenis kerentanan yang ditemukan, lokasi kerentanan pada situs web, dan rekomendasi untuk perbaikan. Informasi ini sangat penting untuk pengembang dan administrator keamanan dalam mengambil tindakan korektif yang diperlukan untuk meningkatkan keamanan situs web. Berikut ini merupakan algoritma untuk mengimplementasikan metode dan dapat di lihat pada Table 2 berikut ini.

Tabel 2. Algoritma Implementasi Metode

1. Mulai
   Inisialisasi proses pengujian kerentanan.
2. Identifikasi Ruang Lingkup
   1. Tentukan Objek Pengujian: Pilih URL yang akan diuji, yaitu http://testphp.vulnweb.com/.
   2. Verifikasi Ruang Lingkup: Pastikan bahwa URL yang dipilih merupakan target yang tepat untuk pengujian keamanan.
3. Pemindaian Kerentanan
   1. Siapkan Alat Pemindaian: Siapkan alat Ecaxss untuk pemindaian kerentanan XSS.
   2. Eksekusi Pemindaian:
      1. Brute Force *Payload*s: Gunakan metode Brute Force untuk mengirimkan berbagai *payload* XSS ke semua input pada URL target.
      2. Monitor Respon: Pantau setiap respons dari situs untuk mendeteksi tanda-tanda kerentanan, seperti eksekusi skrip yang tidak diinginkan.
   3. Catat Hasil: Rekam semua kerentanan yang ditemukan beserta detail respon yang diterima.
4. Pelaporan
   1. Buat Laporan:
      1. Durasi dan Akurasi: Catat durasi dan akurasi dari pemindaian.

2. Detail Kerentanan: Deskripsikan jenis kerentanan yang ditemukan, lokasi pada situs, dan potensi dampak.
3. Rekomendasi: Sertakan rekomendasi untuk memperbaiki kerentanan yang ditemukan.

2. Distribusi Laporan: Bagikan laporan kepada pengembang dan tim keamanan untuk tindakan korektif.

5. Selesai
Proses pengujian kerentanan selesai.

2.4 Rancangan Pengujian

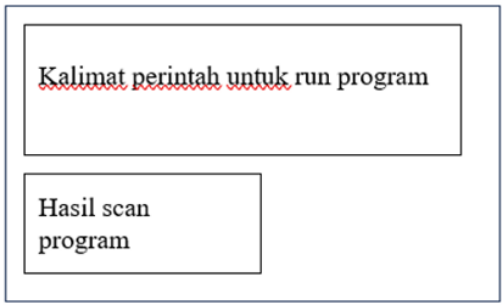
Rancangan pengujian ini dirancang untuk menguji kerentanan situs web menggunakan *tools* pemindai yang memanfaatkan metode *Brute Force*, dilakukan untuk memastikan bahwa situs web tersebut aman dari berbagai jenis serangan siber yang dapat dieksploitasi oleh peretas. Rancangan pengujian ini menggunakan metode black box testing dan dirancang untuk memberikan gambaran menyeluruh tentang keamanan situs web target. Pada rancangan pengujian Tabel 3 ini menggunakan salah 5 dari 99 *payload* yang dimiliki oleh peneliti.

Tabel 3. *Payload* yang digunakan

No	Script	Ket
1	<xss onafterscriptexecute=alert(1)><script>1</script>	Menampilkan alert dengan pesan "1" setelah script di dalam <script> selesai dieksekusi.
2	<xss onmouseover="alert(1)"style=display:block>test</xss>	Menjalankan JavaScript saat pengguna mengarahkan kursor ke elemen tersebut
3	<marquee onstart=alert(1)>XSS</marquee>	Menjalankan JavaScript ketika elemen marquee mulai bergerak. Bisa memicu tindakan yang tidak diinginkan saat halaman diakses.
4	test	Menjalankan JavaScript sebelum konten disalin oleh pengguna, berpotensi menyuntikkan kode atau mencuri data clipboard
5	<body onload=alert(1)>	Menjalankan kode JavaScript segera setelah halaman dimuat. Dapat digunakan untuk menampilkan pesan pop-up atau menjalankan script berbahaya.

2.5 Rancangan Layar

Desain ini selalu di perlukan saat merancang sebuah aplikasi. Desain layar adalah langkah pertama dalam menciptakan tampilan yang mudah di jalankan. Rancangan layar tools ini dapat dilihat pada Gambar 3 berikut ini.



Gambar 3. Rancangan Layar

3. HASIL DAN PEMBAHASAN

3.1 Algoritma Program

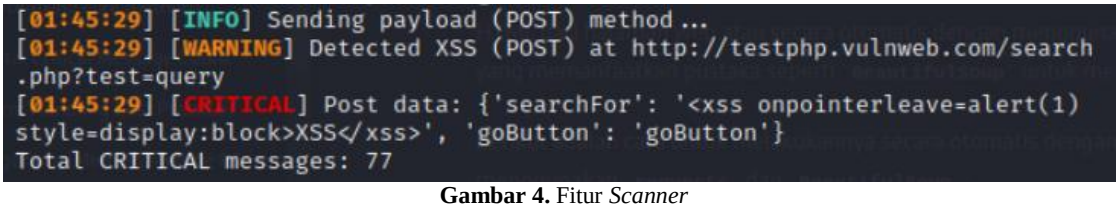
Berikut ini adalah algoritma untuk keseluruhan program *vulnerability scanning* yaitu ecaxss, seperti yang ditunjukkan pada Table Algoritma 4 berikut ini:

Table 4 Algoritma Program Ecaxss

1.	Mulai
2.	tampilkan "Masukkan URL target:"
3.	baca URL dari pengguna
4.	jika URL tidak diisi maka
5.	tampilkan "URL tidak boleh kosong. Program berhenti."
6.	siapkan catatan kegiatan (logging)
7.	tampilkan "Memuat data uji dari file..."
8.	data_uji = baca_data_dari_file('*payload*1.txt')
9.	tampilkan "Memeriksa koneksi ke URL..."
10.	jika koneksi ke URL gagal maka
11.	tampilkan "Koneksi gagal. Program berhenti."
12.	tampilkan "Koneksi berhasil. Mulai menguji XSS..."
13.	mulai_uji_XSS(URL, data_uji, metode POST)
14.	tampilkan "Membuat laporan akhir..."
15.	buat_laporan_akhir()
16.	tampilkan "Program selesai"
17.	Selesai

3.2 Fitur *Scanner*

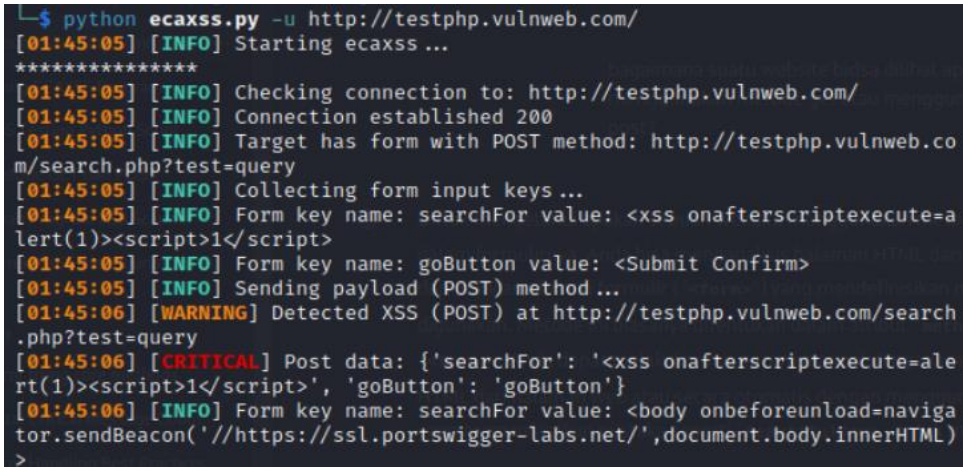
Setiap kali *payload* dikirim, hasilnya dicatat sebagai *info*, *warning*, atau *critical*. Ini dilakukan untuk memastikan semua langkah dan hasil pengujian tercatat dengan baik. Informasi ini sangat berguna untuk analisis lebih lanjut dan pembuatan laporan akhir. Setelah semua pengujian selesai, dibuat laporan yang menunjukkan jumlah pesan kritis yang dicatat selama proses. Laporan ini memberikan gambaran tentang hasil pengujian dan berapa banyak serangan XSS yang berhasil dilakukan. Laporan ini penting untuk memahami seberapa rentan halaman web yang diuji. Jika program mendeteksi adanya kerentanan XSS pada URL target setelah mengirim *payload*, program akan menampilkan pesan peringatan (WARNING) yang menunjukkan bahwa XSS terdeteksi pada URL tertentu. Selain itu, pesan kritis (CRITICAL) akan menunjukkan data yang dikirimkan dalam form POST dan (INFO) memiliki arti bahwa *payload* dikirimkan menggunakan metode POST ke URL yang ditentukan oleh atribut action dari formulir. Fitur *Scanner* ini dapat dilihat pada Gambar 4 berikut ini.



Gambar 4. Fitur *Scanner*

3.3 Tampilan Layar

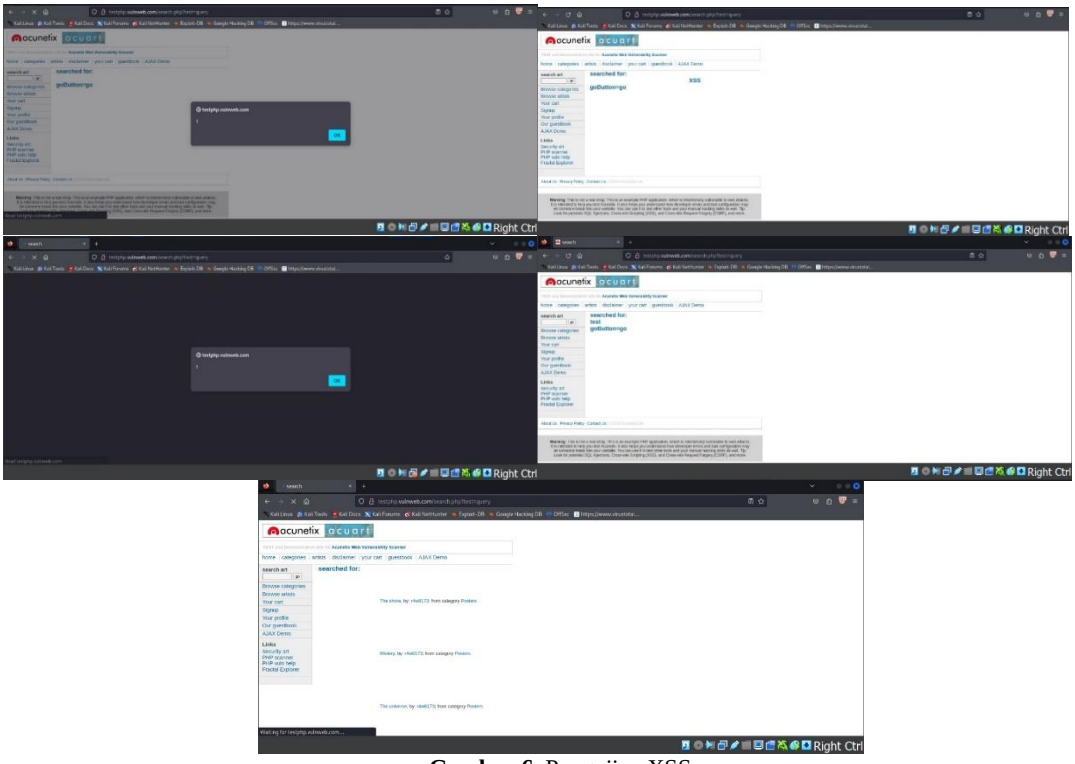
Berikut ini adalah tampilan layar ketika program dijalankan. Dapat dilihat pada Gambar 5 berikut ini.



Gambar 5. Tampilan Layar

3.4 Hasil Pengujian XSS

Berdasarkan hasil pemindaian, situs web http://testphp.vulnweb.com/ memiliki keamanan yang rendah. Pengujian Gambar 6, keberhasilan *payload* menunjukkan bahwa situs ini tidak memiliki perlindungan yang cukup untuk mencegah serangan XSS.



Gambar 6. Pengujian XSS

Berikut ini merupakan hasil laporan yang didapati ketika proses di jalankan. Berdasarkan hasil yang di dapati ini, *tools* ecaxss dinilai efektif dikarenakan keamanan situs bisa ditembus, menimbulkan risiko serius bagi pengguna. Hasil pemindaian lengkap dapat dilihat pada Tabel 4 berikut ini.

Tabel 4. Hasil Deteksi

Tools	Durasi *Scanning*	Hasil *payload*
Ecaxss	1 menit 11,42 detik	77/99

4. KESIMPULAN

Berdasarkan hasil *scanning* yang telah di jalankan, ditemukan bahwa situs web http://testphp.vulnweb.com memiliki tingkat keamanan yang rendah. Hal ini dibuktikan dengan keberhasilan alat pengujian ecaxss dalam menembus lapisan keamanan yang dimiliki oleh situs web tersebut. Dari pengujian ini, dapat disimpulkan bahwa situs tersebut rentan terhadap serangan *Cross-Site Scripting*(XSS). XSS merupakan salah satu jenis serangan injeksi di mana penyerang dapat memasukkan kode berbahaya ke dalam halaman web yang kemudian dijalankan oleh browser pengguna lain. Serangan ini dapat digunakan untuk mencuri informasi sensitif pengguna, seperti cookie sesi, atau bahkan mengendalikan akun pengguna.

DAFTAR PUSTAKA

[1] D. T. Yuwono, "Analysis Performance Intrusion Detection System in Detecting Cyber-Attack on Apache Web Server," *IT Journal Research and Development*, pp. 169–178, Feb. 2022, doi: 10.25299/itjrd.2022.7853.
[2] BSSN, "Laporan Tahunan Monitoring Keamanan Siber 2021," 2022.
[3] I. M. Edy Listartha, I. M. A. Premana Mitha, M. W. Aditya Arta, and I. Km. W. Yuda Arimika, "Analisis Kerentanan Website SMA Negeri 2 Amlapura Menggunakan Metode OWASP (Open Web Application Security Project)," *SIMKOM*, vol. 7, no. 1, pp. 23–27, Jan. 2022, doi: 10.51717/simkom.v7i1.63.

[4] S. Suroto and A. Asman, "Ancaman Terhadap Keamanan Informasi Oleh Serangan Cross-Site Scripting (XSS) Dan Metode Pencegahannya," *Zona Komputer: Program Studi Sistem Informasi Universitas Batam*, *11*(1), pp.11-19, 2021. [Online]. Available: http://www.*hacker*s.com?yid=

[5] S. Khan, "LL-XSS: End-to-End Generative Model-based XSS *Payload* Creation," in 21st International Learning and Technology Conference: Reality and Science Fiction in Education, L and T 202*4*, Institute of Electrical and Electronics Engineers Inc., 2024, pp. 121–126. doi: 10.1109/LT60077.2024.10469151.

[6] I. Riadi, R. Umar, and T. Lestari, "Analisis Kerentanan Serangan Cross Site Scripting (XSS) pada Aplikasi Smart Payment Menggunakan Framework OWASP," *JISKA (Jurnal Informatika Sunan Kalijaga)*, vol. 5, no. 3, pp. 146–152, Nov. 2020, doi: 10.14421/jiska.2020.53-02.

[7] A. Wira Utama and A. Senja Fitrani, "Techniques For Testing Website Security Using The Escaping Metacharacter Method Teknik Menguji Keamanan Website Dengan Menggunakan Metode Escaping Metacharacter," *Procedia of Engineering and Life Science Vol*, *2*(2), 2022.

[8] G. Suprianto, "Penetration Testing Pada Sistem Informasi Jabatan Universitas Hayam Wuruk Perbanas," *InComTech : Jurnal Telekomunikasi dan Komputer*, vol. 12, no. 2, p. 129, Aug. 2022, doi: 10.22441/incomtech.v12i2.15093.

[9] S. Hidayatulloh and D. Saptadiaji, "Penetration Testing pada Website Universitas ARS Menggunakan Open Web Application Security Project (OWASP)," *Jurnal Algoritma*, *18*(1), pp.77-86, 2021. [Online]. Available: http://jurnal.itg.ac.id/

[10] A. Sinaga, "Implementasi Algoritma *Brute Force* Dalam Pencarian Menu Pada Aplikasi Pemesanan Coffee (Studi Kasus : Tanamera Coffee)," *Jurnal Ilmu Komputer dan Sistem Informasi (JIKOMSI)*, vol. 4, no. 1, pp. 6–15, 2021.

[11] S. Rahman *et al.*, Python: Dasar dan Pemrograman Berorientasi Objek Tahta Media Group. Penerbit Tahta Media, 2023.

[12] I. A. Mukaromah *et al.*, "Analisis Pencocokan String Menggunakan Algoritma *Brute Force*," *Jurnal Teknik Informatika dan Sistem Informasi*, *1*(1), pp.18-22, 2021.