

IMPLEMENTASI KEAMANAN INFORMASI MENGGUNAKAN METODE *WEB APPLICATION FIREWALL* TERHADAP *SQL INJECTION*

Wahdana¹, Kharis Hudaiby Hanif²

^{1,2}Teknik Komputer, Fakultas Teknik, Universitas Borneo Tarakan,
¹wahdarahman26@gmail.com, ²hudaiby21@borneo.ac.id

Abstrak

Era digital kontemporer telah membawa perubahan fundamental dalam lanskap keamanan aplikasi web, di mana sistem keamanan menghadapi eskalasi serangan yang mengancam konsistensi data. Kerentanan port terbuka pada platform web menjadi vektor utama eksploitasi oleh aktor jahat dalam dunia maya. Studi ini menginvestigasi implementasi *Web Application Firewall* (WAF) sebagai mekanisme defensif untuk mitigasi ancaman keamanan aplikasi web. Penelitian ini bertujuan untuk memberikan kontribusi empiris dalam bidang keamanan aplikasi web melalui analisis kuantitatif efektivitas ModSecurity WAF terhadap serangan SQL Injection, serta mengidentifikasi gap dan limitasi dalam implementasi WAF untuk pengembangan solusi keamanan yang lebih robust. ModSecurity digunakan sebagai solusi WAF yang mengoperasikan sistem pemblokiran *traffic malicious* melalui *rule-based filtering*. Serangan *SQL Injection* diteliti sebagai metode penetrasi sistem *database* melalui manipulasi *query Structured Query Language*. Metodologi penelitian melibatkan DVWA sebagai aplikasi target pada infrastruktur Apache2 dalam environment Kali Linux, dengan eksekusi 3 iterasi pengujian untuk menganalisis performa ModSecurity WAF. Temuan penelitian mengindikasikan tingkat efektivitas proteksi sebesar 99% yang sesuai dengan framework OWASP *Web Security Testing Guide*, dengan *margin error* 1% yang disebabkan oleh limitasi aksesibilitas *database* pada platform DVWA.

Kata kunci : *Web Application Firewall, SQL Injection, ModSecurity*

1. Pendahuluan

Perkembangan teknologi internet telah menjadikan aplikasi berbasis web sebagai platform utama untuk komunikasi, informasi, dan *e-commerce* (Kharis Hudaiby Hanif *et al*, 2022). Namun, popularitas ini juga meningkatkan risiko serangan siber, terutama melalui kerentanan yang ada pada aplikasi web yang menyimpan data pribadi pengguna (Dea Ummul Khabibah *et al*, 2024). Keamanan aplikasi web menjadi prioritas utama, dengan *Web Application Firewall* (WAF) sebagai salah satu solusi efektif yang dapat dipasang pada *web server*. WAF mampu mencegah berbagai jenis serangan seperti *SQL Injection*, *XSS*, *DDoS*, *MITM*, dan *Command Injection*. *SQL Injection* khususnya masih menjadi ancaman utama berdasarkan laporan OWASP Top 10. Penelitian sebelumnya telah mengeksplorasi penggunaan SQLMap dan berbagai implementasi WAF untuk mitigasi SQL Injection (Smith *et al*, 2023; Johnson & Brown, 2022). Namun, studi-studi tersebut memiliki beberapa keterbatasan signifikan: (1) terbatasnya variasi tools pengujian yang digunakan dalam evaluasi efektivitas, (2) kurangnya analisis mendalam terhadap dampak implementasi WAF terhadap performa website, dan (3) minimnya dokumentasi detail tentang implementasi payload pengujian dan metodologi evaluasi yang dapat direplikasi. Penelitian ini mengusulkan implementasi

WAF dengan ModSecurity yang terintegrasi pada Apache2 untuk menguji efektivitas perlindungan terhadap serangan *SQL Injection* pada aplikasi DVWA (*Damn Vulnerable Web Application*). Tujuannya adalah mengukur kinerja sistem keamanan WAF dan memberikan solusi pencegahan yang efektif untuk melindungi *database* dari pencurian atau manipulasi data oleh penyerang. Hasil penelitian diharapkan dapat meminimalkan risiko serangan *SQL Injection* pada aplikasi web serta memberikan panduan mitigasi yang komprehensif dengan presentase keberhasilan pengujian yang terukur.

2. Metode

Pada studi literatur penelitian ini akan dijelaskan beberapa diantaranya yaitu berupa teori, kerangka pemikiran, alat dan bahan serta Teknik pengumpulan data.

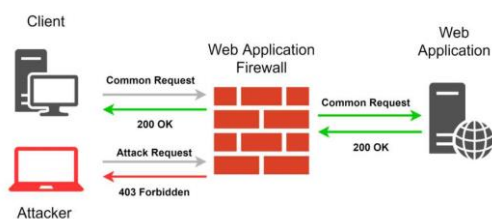
2.1 Aplikasi Web

Aplikasi web adalah suatu sistem informasi yang mendukung interaksi dengan pengguna melalui antarmuka berbasis web. Interaksi pengguna dengan web dibagi ke dalam tiga tahap, yaitu permintaan, pemrosesan, dan jawaban (Muhammad Al Khusnul Rizki & A Ferico OP, 2021). Secara teknis, aplikasi web adalah perangkat lunak komputer yang

dikembangkan menggunakan bahasa pemrograman yang kompatibel dengan peramban web, seperti *HTML*, *JavaScript*, *AJAX*, *Java*, dan teknologi web lainnya, serta bergantung pada peramban untuk renderisasi dan eksekusi aplikasi. Kualitas aplikasi web yang optimal ditentukan oleh pemenuhan berbagai kriteria teknis dan non-teknis, dengan aspek keamanan menjadi faktor kritis yang harus diimplementasikan oleh setiap pengguna dan pengembang. Karakteristik inherent dari aplikasi web menunjukkan adanya kerentanan spesifik yang dapat dieksploitasi, termasuk kesalahan dalam implementasi kode (*coding errors*), ketiadaan sistem *firewall* yang memadai, dan minimnya monitoring terhadap *traffic* HTTP yang melewati aplikasi.

2.3 Web Application Firewall (WAF)

Web Application Security Consortium (WASC) *Web application firewall* (WAF) diartikan sebagai sebuah perangkat perantara, yang berada antara web client dan web server, menganalisis pesan pada OSI Layer 7 ketika terjadi pelanggaran dalam kebijakan keamanan yang telah ditentukan. *Firewall* merupakan sebuah perangkat perantara, yang berada antara web client dan web server, menganalisis pesan pada OSI Layer-7 ketika terjadi pelanggaran dalam kebijakan keamanan yang telah ditentukan (Gregorius Hendita Artha Kusuma, 2021).



Gambar 1 Skema cara kerja WAF

Sumber Gambar: <https://it.telkomuniversity.ac.id/>

WAF berfungsi sebagai security gateway yang menganalisis dan memfilter seluruh incoming traffic sebelum mencapai web application yang dilindungi. Dalam skenario normal operation, ketika client legitimate mengirimkan common request ke aplikasi web, WAF akan melakukan proses inspeksi dan validasi terhadap request tersebut. Setelah memverifikasi bahwa request tidak mengandung elemen berbahaya dan sesuai dengan security rules yang telah dikonfigurasi, WAF akan meneruskan request ke web application. Aplikasi kemudian memproses request dan mengirimkan response "200 OK" kembali ke client melalui WAF, menandakan bahwa transaksi berhasil dilakukan dengan aman. Sebaliknya, ketika attacker mencoba mengirimkan attack request yang mengandung payload berbahaya seperti SQL injection, cross-site scripting, atau jenis serangan lainnya, WAF akan mendeteksi pola serangan tersebut berdasarkan signature-based detection dan behavioral analysis. Sistem kemudian secara otomatis memblokir request malicious tersebut

dan mengirimkan response "403 Forbidden" kepada attacker, mencegah serangan mencapai aplikasi web dan melindungi database serta sistem backend dari potensi eksploitasi. Mekanisme ini menunjukkan bagaimana WAF berperan sebagai first line of defense dalam arsitektur keamanan aplikasi web, memberikan proteksi proaktif terhadap berbagai ancaman siber. Keunggulan WAF terletak pada kemampuannya memberikan perlindungan komprehensif terhadap eksploitasi kerentanan, *malware*, dan ancaman keamanan lainnya yang tidak dapat ditangani oleh *firewall* konvensional, sehingga memberikan lapisan keamanan tambahan yang spesifik untuk aplikasi web (Husein Haikal Muhammad et al., 2023).

2.4 Apache Web Server

Apache Adalah web server yang dapat dijalankan di banyak sistem operasi (Windows, Linux, MAC) yang berguna untuk melayani dan memfungsikan situs web (Fatimah Kesuma Astuti & Dian Sri Agustina, 2022). Karakteristik Apache sebagai monolithic web server ditandai dengan penggunaan file konfigurasi terpusat yang mengelola seluruh parameter sistem secara *unified*. Implementasi WAF menggunakan *ModSecurity* pada Apache web server dipilih karena kompatibilitas dan dukungan yang optimal terhadap berbagai website yang menggunakan Apache sebagai platform *hosting*. Integrasi ini memungkinkan penerapan keamanan aplikasi web yang *seamless* tanpa mengganggu performa dan fungsionalitas *existing system*.

2.5 ModSecurity

ModSecurity merupakan *Web Application Firewall open-source* yang berfungsi sebagai modul tambahan (*add-on module*) untuk Apache HTTP Server. Platform ini menyediakan berbagai fitur keamanan canggih, termasuk kemampuan log inspection, akses kontrol terhadap seluruh komponen HTTP request (termasuk *request body*), dan *response mechanism* berdasarkan hasil analisis keamanan. Keunggulan teknis *ModSecurity* meliputi implementasi *rule-based system* yang menggunakan *regular expression* dengan fleksibilitas tinggi, mekanisme validasi file *upload*, *real-time validation*, dan proteksi terhadap *buffer overflow attacks*. Arsitektur modular *ModSecurity* memungkinkan *customization* dan *fine-tuning* sesuai dengan kebutuhan spesifik aplikasi web yang dilindungi (Riska et al., 2021).

2.9 OWASP Web Security Testing Guide

Web Security Testing Guide atau WSTG merupakan panduan untuk melakukan tes keamanan pada website, untuk mengevaluasi dan melibatkan analisis aktif dari aplikasi untuk setiap kelemahan, baik kelemahan teknis ataupun kerentanan (Albestty Islamyati Rafel et al, 2022). OWASP *Web Security Testing Guide* (WSTG) versi 4.2 mengklasifikasikan

metodologi pengujian ke dalam dua kategori utama: *passive testing* dan *active testing*. *Passive testing* melibatkan pemahaman mendalam terhadap *application logic* dan *behavior analysis* seolah-olah tester berperan sebagai legitimate user.

Penentuan risk level dari kerentanan yang ditemukan dilakukan menggunakan OWASP Risk Rating Methodology. Tingkat risiko atau *impact* yang dihasilkan oleh suatu *vulnerability* ditentukan berdasarkan lima faktor kritis: *exploitability* (*attack vector*), *weakness prevalence* (keberadaan kelemahan), *weakness detectability* (kemudahan deteksi), *technical impacts* (dampak teknis), dan *business impacts* (dampak bisnis) (Dimas Febriyan Priambodo et al., 2023).

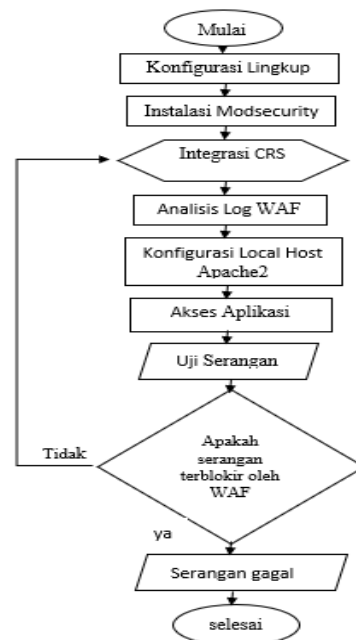
Tabel 1 OWASP Web Security Testing Guide versi 4.2

<i>exploitability</i>	<i>weakness prevalence</i>	<i>weakness detectability</i>	<i>technical impacts</i>	<i>business impacts</i>
Mudah ()	Tersebar luas ()	Mudah ()	Parah ()	?
Sedang ()	Biasa ()	Sedang ()	Cukup ()	
Sulit ()	Tidak tersebar ()	Sulit ()	Rendah ()	

Proses konfigurasi kemudian dilanjutkan dengan pengaturan file konfigurasi *ModSecurity* untuk mengaktifkan dan mengendalikan perilaku WAF sesuai dengan kebutuhan penelitian. Tahap berikutnya melibatkan integrasi *OWASP Core Rule Set* (CRS) yang berisi kumpulan aturan standar untuk memberikan perlindungan terhadap berbagai jenis serangan, khususnya *SQL Injection* yang menjadi fokus utama dalam penelitian ini. Untuk memastikan *ModSecurity* dapat beroperasi dengan optimal dan memberikan perlindungan yang efektif terhadap aplikasi DVWA, dilakukan modifikasi konfigurasi virtual host yang kemudian diikuti dengan *restart* web server Apache2 agar seluruh konfigurasi yang telah diatur dapat diterapkan dengan sempurna.

Fase pengujian dimulai dengan mengakses aplikasi web DVWA melalui browser untuk melakukan verifikasi bahwa sistem telah siap untuk diuji. Selanjutnya dilakukan serangkaian serangan *SQL Injection* untuk mengevaluasi efektivitas WAF dalam mendeteksi dan memblokir serangan tersebut. Proses evaluasi melibatkan analisis terhadap hasil yang diperoleh untuk menentukan apakah WAF berhasil melakukan deteksi dan pemblokiran terhadap serangan yang dilancarkan. Apabila WAF berhasil memblokir serangan, maka dapat disimpulkan bahwa sistem berfungsi dengan baik dan proses implementasi telah berhasil. Sebaliknya, jika serangan masih dapat menembus pertahanan WAF, maka diperlukan analisis lebih mendalam terhadap log WAF untuk mengidentifikasi penyebab kegagalan dan menganalisis informasi detail mengenai permintaan yang masuk ke sistem.

Setelah menyelesaikan seluruh rangkaian pengujian sesuai dengan metodologi yang telah ditetapkan, tahap akhir penelitian adalah menghitung dan mempresentasikan hasil akhir yang diperoleh dari proses pengujian. Perhitungan persentase keberhasilan ini dilakukan untuk mengukur tingkat efektivitas kinerja sistem keamanan WAF *ModSecurity* yang telah dikonfigurasi dan diuji menggunakan serangkaian serangan *SQL Injection* sebagai sampel serangan. Formula yang digunakan untuk menghitung nilai persentase keberhasilan dalam penelitian ini adalah $PK = (JS/TS) \times 100\%$, dimana PK merepresentasikan Persentase Keberhasilan, JS adalah Jumlah Serangan yang berhasil diblokir oleh sistem, dan TS merupakan Total Serangan yang dilakukan selama proses pengujian. Melalui formula ini, dapat diperoleh gambaran kuantitatif mengenai seberapa efektif WAF *ModSecurity* dalam melindungi aplikasi web dari ancaman serangan *SQL Injection*.



Gambar 2 Flowchart Penelitian

Gambar 2 menunjukkan alur metodologi penelitian yang digunakan dalam proses pengumpulan dan analisis data pada studi ini. Berdasarkan diagram tersebut, dapat diuraikan bahwa penelitian dimulai dengan tahap persiapan lingkungan pengujian yang melibatkan konfigurasi sistem operasi Kali Linux sebagai platform utama, instalasi web server Apache2, dan penyiapan aplikasi DVWA (*Damn Vulnerable Web Application*) sebagai target serangan yang akan digunakan dalam eksperimen. Setelah lingkungan dasar terkonfigurasi, langkah selanjutnya adalah implementasi *ModSecurity* pada server Apache2 yang berfungsi sebagai fondasi utama dari *Web Application Firewall* yang akan diuji.

2.12 Kerangka Berpikir

Web Application Firewall (WAF) ModSecurity berperan sebagai solusi keamanan yang memfilter *traffic* HTTP/HTTPS antara *user* dan aplikasi web. *ModSecurity* bekerja dengan cara memonitor setiap *request* yang masuk, lalu membandingkannya dengan *rule* yang telah ditetapkan untuk mengidentifikasi pola serangan. Ketika terdeteksi aktivitas mencurigakan, *ModSecurity* akan memblokir *request* tersebut sebelum mencapai aplikasi dan menampilkan pesan *error* kepada *user*. Untuk keperluan pembelajaran dan *testing* keamanan, digunakan platform *Damn Vulnerable Web Application (DVWA)* yang berjalan di Apache2. Platform ini dirancang khusus untuk pembelajaran *penetration testing*, memungkinkan praktisi mempelajari berbagai kerentanan sistem, terutama kelemahan terhadap serangan *SQL Injection* dalam lingkungan yang terkontrol.

3. Tahap Penelitian

Diagram flowchart ini menggambarkan metodologi penelitian yang sistematis untuk evaluasi efektivitas Web Application Firewall dalam mitigasi serangan SQL Injection. Proses penelitian dimulai dengan Studi Literatur sebagai fondasi teoretis, di mana peneliti mengumpulkan dan menganalisis referensi akademik, dokumentasi teknis, dan research paper terkait keamanan aplikasi web, teknologi WAF, dan teknik serangan SQL Injection untuk membangun landasan pengetahuan yang solid. Tahap selanjutnya adalah Mengidentifikasi Variabel penelitian, yang meliputi penentuan variabel independen (konfigurasi ModSecurity rules), variabel dependen (tingkat efektivitas proteksi), dan variabel kontrol (environment testing) yang akan digunakan dalam eksperimen. Setelah variabel ditetapkan, penelitian berlanjut ke fase Konfigurasi Lingkup Uji dengan mempersiapkan infrastructure testing yang mencakup instalasi dan konfigurasi Kali Linux, Apache2 web server, ModSecurity WAF, dan DVWA sebagai target aplikasi vulnerable.

Proses Skenario Uji kemudian dirancang untuk mencakup berbagai jenis attack vector SQL Injection dengan payload yang bervariasi, diikuti dengan Uji Awal untuk memastikan semua komponen sistem berfungsi dengan baik dan environment testing siap untuk eksekusi. Uji Sebenarnya dilakukan dengan melaksanakan serangkaian *penetration testing* menggunakan automated tools dan manual testing untuk mengevaluasi kemampuan ModSecurity dalam mendeteksi dan memblokir serangan SQL Injection. Akhirnya, Analisa Data Hasil dilakukan untuk mengolah data kuantitatif yang diperoleh, menghitung tingkat efektivitas proteksi, menganalisis false positive/negative rate, dan menyusun kesimpulan berdasarkan evidence empiris yang telah dikumpulkan selama proses pengujian.

3.1 Alat dan Bahan

Tabel 2 Alat Penelitian

Nama Alat	Keterangan
Laptop Lenovo-DESKTOP-B519B03 dengan spesifikasi:	Sebagai komputer untuk melakukan
a) Prosesor: AMD A4-9125 RADEON R3, 4 COMPUTE CORES 2C+2G (2.30 GHz).	pengimplementasian serta pengujian penetrasi
b) RAM: 4.00 GB	
c) Sytem Type: 64-bit	
ZTE Corporation Wifi Certified dengan spesifikasi:	
a) MAC: 28-FF-3E-4B-6A-C2	
b) GPON SN: ZTEGC1D941A0	
c) D-SN: ZTENQAJH5210288	Sebagai penghubung antar komputer dan jaringan internet
d) SSID1: ZTE-4b6ac2	
e) WPA/WPA2-PSK: 28ff3e4b	

Tabel 3 Bahan Penelitian

Nama Bahan	Keterangan
ModSecurity	<i>Web Application Firewall</i>
Kali linux	Sistem operasi yang digunakan untuk melakukan <i>penetration testing</i>
OWASP Web Security Testing Guide	Dokumen panduan dalam pengujian keamanan aplikasi web
Apache2	Web server yang digunakan untuk melakukan uji coba <i>ModSecurity</i>
CRS	Aturan keamanan pada <i>ModSecurity WAF</i> untuk meningkatkan keamanan aplikasi web
DVWA	Target uji coba serangan
Nikto	<i>Tools</i> yang digunakan untuk melakukan <i>scanning</i> pada web server yang akan dilakukan serangan
SQLMap	<i>Tools</i> yang digunakan untuk melakukan serangan

3.2 Teknik Pengumpulan Data

Penelitian ini bersifat eksperimen, sehingga teknik pengumpulan data yang digunakan akan lebih bersifat kuantitatif. Pada penelitian ini dilakukan implementasi *firewall* sebagai sistem keamanan web yang dilakukan sebanyak 3 kali pengujian dengan *script* payload SQLi (Riska, 2021). Hasil eksperimen selanjutnya didokumentasikan untuk melakukan analisa sehingga dihasilkan rekomendasi yang tepat untuk *firewall* sebagai sistem keamanan web. Berikut merupakan rumusan dalam mencari nilai presentase keberhasilan pada penelitian ini:

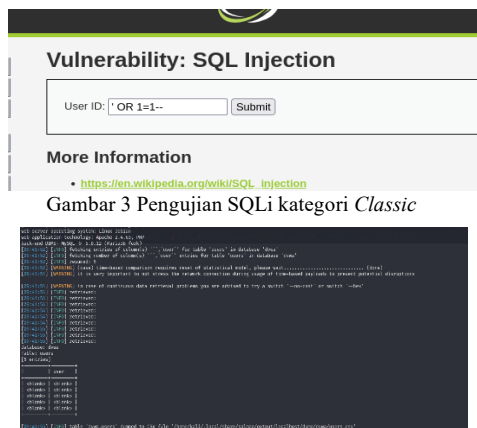
$$PK = \frac{JS}{TS} \times 100\% \quad (1)$$

PK = Presentase Keberhasilan
JS = Jumlah Serangan yang diblokir
TS = Total Serangan

4. Pembahasan

ModSecurity sebagai open-source *Web Application Firewall* memiliki kemampuan proteksi yang luas terhadap berbagai jenis serangan aplikasi web, tidak hanya terbatas pada *SQL Injection*. Evaluasi awal menunjukkan bahwa ModSecurity dengan *Core Rule Set* (CRS) yang dikonfigurasi dapat mendeteksi dan memblokir *multiple attack vectors* yang tercantum dalam OWASP Top 10.

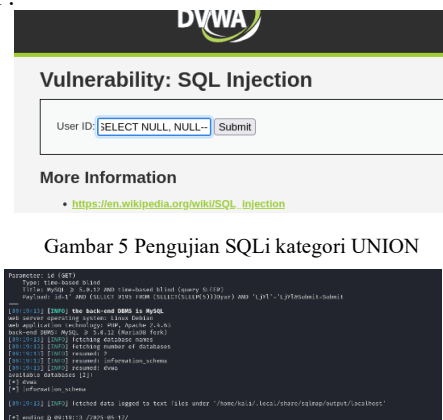
4.1 Pengujian tanpa sistem keamanan WAF



Gambar 3 Pengujian SQLi kategori Classic

Gambar 4 Output Command

Gambar 3 dan Gambar 4 menunjukkan hasil pengujian *SQL Injection* dengan kategori *Classic* pada aplikasi DVWA menggunakan payload ' OR 1=1-- secara manual dan tool SQLmap secara otomatis. Kedua metode berhasil mengeksploitasi kerentanan pada parameter *User ID*, mengakses *database* MySQL, dan mengidentifikasi data sensitif dalam tabel *users*. Pengujian membuktikan kerentanan aplikasi terhadap serangan *SQL Injection* yang memerlukan implementasi sistem keamanan WAF.

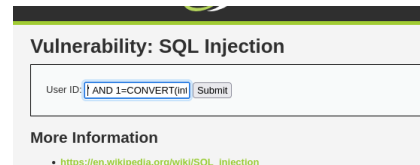


Gambar 5 Pengujian SQLi kategori UNION

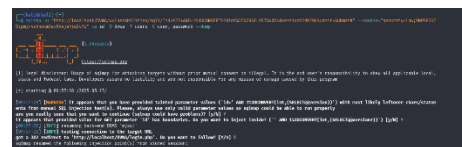
Gambar 6 Output Command

Gambar 5 dan Gambar 6 menunjukkan pengujian *UNION-Based SQL Injection*, dimana dari pengujian tersebut membuktikan kerentanan aplikasi DVWA

terhadap teknik *injection* yang lebih advanced, yang dapat mengekstrak data dengan menggabungkan *multiple query SQL*. Hal ini memperkuat urgensi implementasi WAF *ModSecurity* untuk melindungi aplikasi web dari berbagai varian serangan *SQL Injection*.



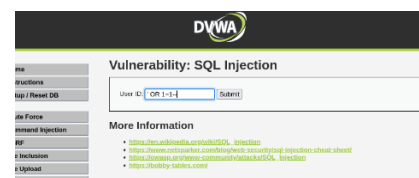
Gambar 7 Pengujian SQLi kategori Error



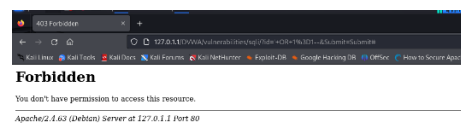
Gambar 8 Output Command

Gambar 7 dan Gambar 8 menunjukkan pengujian *Error-Based SQL Injection*, dimana gambar tersebut menunjukkan tingkat kerentanan yang paling tinggi, dimana tidak hanya berhasil mengeksploitasi *database* tetapi juga berhasil mengekstrak data sensitif secara lengkap. Hal ini menunjukkan urgensi kritical untuk implementasi sistem keamanan WAF *ModSecurity* guna mencegah kebocoran data pengguna.

4.2 Pengujian dengan sistem keamanan WAF

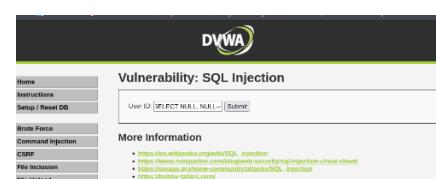


Gambar 9 Pengujian SQLi kategori Classic

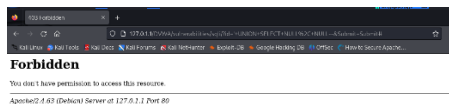


Gambar 10 Layanan ditolak web server

Gambar 9 dan Gambar 10 menunjukkan hasil pengujian yang telah dilakukan, dimana implementasi WAF *ModSecurity* terbukti efektif dalam melindungi aplikasi web dari serangan *Classic SQL Injection*. Sistem berhasil mendeteksi pola serangan dan memblokir *request* berbahaya sebelum mencapai *database*, sehingga mencegah potensi eksploitasi dan kebocoran data.



Gambar 11 Pengujian SQLi kategori UNION

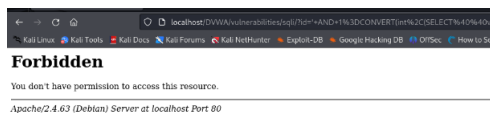


Gambar 12 Layanan ditolak web server

Pada Gambar 11 dan Gambar 12 WAF *ModSecurity* menunjukkan kemampuan proteksi yang konsisten tidak hanya terhadap *Classic SQL Injection*, tetapi juga terhadap teknik yang lebih *advanced* seperti *UNION-Based SQL Injection*. Sistem keamanan berhasil mengidentifikasi dan memblokir berbagai pola serangan, membuktikan efektivitasnya dalam melindungi aplikasi web dari *multiple attack vectors*.



Gambar 13 Pengujian SQLi kategori Error



Gambar 14 Layanan ditolak web server

Pada Gambar 13 dan Gambar 14 WAF *ModSecurity* mendemonstrasikan kemampuan proteksi yang komprehensif terhadap ketiga kategori *SQL Injection* (Classic, UNION-based, dan Error-based). Sistem keamanan terbukti efektif dalam mencegah berbagai teknik eksploitasi *database* dan melindungi integritas data pengguna dari upaya manipulasi dan pencurian informasi.

4.3 Hasil pengujian tanpa WAF

Tabel 4 Hasil perbandingan pengujian pada penelitian tanpa WAF

Exploitability	weakness prevalence	weakness detectability	technical impacts	business impacts
Classic SQL Injection	Tersebar luas (5)	Low (5)	Parah (5)	Database Terlihat
UNION – SQL Injection	Tersebar luas (5)	Low (5)	Parah (5)	Database Terlihat
Error – Based SQL Injection	Tersebar luas (5)	Low (5)	Parah (5)	Database Terlihat

Berdasarkan data pada Tabel 4, pengujian dilakukan menggunakan 3 *script SQL Injection* terhadap aplikasi web DVWA tanpa proteksi WAF, baik secara manual maupun otomatis menggunakan

SQLmap. Hasil menunjukkan bahwa **seluruh script berhasil mengeksploitasi** sistem target.

$$\text{Presentase Keberhasilan} = \frac{0}{3} \times 100\% = 0 \quad (2)$$

Perhitungan ini menunjukkan bahwa *server tidak memiliki kemampuan* untuk menggagalkan upaya serangan *SQL Injection* dan **gagal total** dalam melindungi *database* sensitif pengguna.

Tabel 5 Hasil perbandingan pengujian pada penelitian dengan WAF

Exploitability	weakness prevalence	weakness detectability	technical impacts	business impacts
Classic SQL Injection	Tidak tersebar (5)	Low (5)	Rendah (5)	Aplikasi web terblokir
UNION – SQL Injection	Tidak tersebar (5)	Low (5)	Rendah (5)	Aplikasi web terblokir
Error – Based SQL Injection	Tidak tersebar (5)	Low (5)	Rendah (5)	Aplikasi web terblokir

Berdasarkan pengujian yang dilakukan terhadap sistem keamanan menggunakan *Web Application Firewall* (WAF), dapat disimpulkan bahwa aplikasi web yang menggunakan WAF menunjukkan tingkat perlindungan yang lebih tinggi dibandingkan dengan yang tidak menggunakannya, serta dalam pengujian, total 3 skrip serangan *SQL Injection* berhasil diblokir sepenuhnya oleh *server WAF*, menghasilkan tingkat keberhasilan 100%.

$$\text{Presentase Keberhasilan} = \frac{3}{3} \times 100\% = 100 \quad (3)$$

Presentase keberhasilan yang telah dihitung menunjukkan bahwa WAF terbukti efektif dalam melindungi data sensitif pengguna dan mampu menggagalkan upaya serangan SQLi yang dilakukan oleh penyerang untuk memanipulasi *database* pengguna. Oleh karena itu, penerapan WAF sangat disarankan untuk meningkatkan keamanan aplikasi web.

5. Kesimpulan

Sistem keamanan WAF (*Web Application Firewall*) masih menjadi pilihan utama untuk melakukan pengujian standar terhadap berbagai serangan, termasuk *SQL Injection*. Pengujian yang dilakukan pada web server Apache2 dengan WAF jenis *ModSecurity* berhasil mengimplementasikan sistem keamanan yang efektif, mampu memblokir lalu lintas aplikasi web yang terdeteksi memiliki payload SQLi berbahaya di aplikasi DVWA. Hasil presentase keberhasilan pengujian sesuai dengan standar yang ditetapkan oleh OWASP *Web Security Testing Guide*. WAF berfungsi untuk melindungi kerentanan dari eksploitasi dengan menyediakan

lapisan keamanan untuk aplikasi web melalui pemantauan dan pemfilteran lalu lintas jaringan yang menggunakan protokol HTTP. Penelitian ini berhasil menguji lapisan keamanan WAF *ModSecurity* di web server dengan melakukan dua pengujian yaitu sebelum dan sesudah konfigurasi sistem keamanan WAF. Dari pengujian tersebut, 3 skrip dari 3 kategori serangan SQLi menunjukkan bahwa lapisan sistem keamanan WAF *ModSecurity* mampu mengurangi risiko pencurian data oleh penyerang hingga 99% dan efektif dalam melindungi informasi pengguna aplikasi web.

Daftar Pustaka:

- Annas, M., Adek, R. T., & Afrillia, Y. (2024). *Web Application Firewall (WAF) Design to Detect and Anticipate Hacking in Web-Based Applications. Journal of Advanced Computer Knowledge and Algorithms*, 1(3), 52.
- A. W. Adi Wijaya, Toibah Umi Kalsum, and Riska, "Penerapan OPNsense Sebagai Sistem Keamanan Web Server Menggunakan Metode Host Intrusion Prevention Sistem," *J. Amplif. J. Ilm. Bid. Tek. Elektro Dan Komput.*, vol. 13, no. 2, pp. 91–100, 2023.
- D. F. Priambodo, A. D. Rifansyah, and M. Hasbi, "Penetration Testing Web XYZ Berdasarkan OWASP Risk Rating," *Teknika*, vol. 12, no. 1, pp. 33–46, 2023.
- D. U. Khabibah, Y. Nurrohmah, K. Dewandaru, S. J. D. H. Balian, and A. Setiawan, "Strategi Mitigasi *SQL Injection* dengan Implementasi SQLMap dan Web Application Firewall," *J. Technol. Syst. Inf.*, vol. 1, no. 4, p. 12, 2024.
- F. Fachri, A. Fadlil, and I. Riadi, "Analisis Keamanan Webserver menggunakan Penetration Test," *J. Inform.*, vol. 8, no. 2, pp. 183–190, 2021.
- Fitria, E., Janitra, dkk, "Metode Penelitian Eksperimental". *Jurnal Kesehatan*, vol. 11, no. 2 2024.
- Firda, S. Putri, Y. B. Utomo, and H. Kurniadi, "Analisa Celah Keamanan Pada Website Pemerintah Kabupaten Kediri Menggunakan Metode Penetration Testing Melalui Kali Linux," *Pros. SEMNAS INOTEK (Seminar Nas. Inov. Teknol.*, vol. 7, no. 1, pp. 52–59, 2023, [Online]. Available
- H. Haikal Muhammad, A. Id Hadiana, and H. Ashaury, "Pengamanan Aplikasi Web Dari Serangan *SQL Injection* Dan Cross Site Scripting Menggunakan Web Application Firewall," *JATI (Jurnal Mhs. Tek. Inform.*, vol. 7, no. 5, pp. 3265–3273, 2024.
- Hanif, K. H., Yudhana, A., & Fadlil, A. (2022). Penentuan Guru Berprestasi Menggunakan Metode Analytical Hierarchy Process (AHP) dan ViseKriterijumska Optimizacija I Kompromisno Resenje (VIKOR). *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 9(6), 1119–1128.
- Kesuma Astuti, F., & Sri Agustina, D. (2022). Membangun Website MTS Negeri 01 OKU Timur Menggunakan Php dan Mysql. *Jik*, 13(1), 7–14.
- Kusuma, G. H. A. (2021). Perancangan Skema Sistem Keamanan Jaringan Web Server menggunakan Web Application Firewall dan Fortigate untuk Mencegah Kebocoran Data di Masa Pandemi Covid-19. *Journal of Informatics and Advanced ...*, 2(2), 1–4.
- N. H. Humaira, I. A. Hadiana, and H. Ashaury, "Analisis Ketahanan Web Application Firewall Terhadap Serangan *SQL Injection*," *J. Ilm. Wahana Pendidik.*, vol. 10, no. 5, pp. 403–412, 2024, [Online]. Available
- Rafeli, A. I., Seta, H. B., & Widi, I. W. (2022). Pengujian Celah Keamanan Menggunakan Metode OWASP Web Security Testing Guide (WSTG) pada Website XYZ. *Informatik: Jurnal Ilmu Komputer*, 18(2), 97.
- Rizki, M. A. K., & Ferico, A. (2021). Rancang Bangun Aplikasi E-Cuti Pegawai Berbasis Website (Studi Kasus: Pengadilan Tata Usaha Negara). *Jurnal Teknologi Dan Sistem Informasi (JTSI)*, 2(3), 1–13.
- R. Hermawan, "Teknik Uji Penetrasi Web Server Menggunakan *SQL Injection* dengan SQLmap di Kalilinux," *STRING (Satuan Tulisan Ris. dan Inov. Teknol.*, vol. 6, no. 2, p. 210, 2021.
- R. Riska and H. Alamsyah, "Penerapan Sistem Keamanan Web Menggunakan Metode Web Application Firewall," *J. Amplif. J. Ilm. Bid. Tek. Elektro Dan Komput.*, vol. 11, no. 1, pp. 37–42, 2021.

Halaman ini sengaja dikosongkan