JUPITER
Teknologi Informatika & Komputer
Vol. 5 No. 2 Agustus 2024
Institut Bisnis Muhammadiyah Bekasi
P-ISSN: 2798-3129 , E-ISSN: 2798-6187

ANALISIS EFEKTIVITAS TEKNIK PARAMETERIZED QUERIES DALAM MENCEGAH SERANGAN *SQL INJECTION* MENGGUNAKAN DVWA

Muhammad Fadillah[1], Yustian Servanda[1]

[1]Teknologi Informasi, Universitas Mulia, muhammadfadillah@students.universitasmulia.ac.id, yustians@universitasmulia.ac.id

ABSTRAK

Salah satu jenis serangan *SQL Injection* yang paling umum dan mengancam aplikasi web menggunakan celah keamanan dalam input data untuk mengeksekusi perintah SQL yang tidak diinginkan. Teknik Parameterized Queries, juga dikenal sebagai pernyataan yang dibuat, dianggap sebagai salah satu cara terbaik untuk mengatasi masalah ini. Tujuan penelitian ini adalah untuk mengevaluasi seberapa efektif metode pertanyaan parameter dalam mencegah serangan *SQL Injection* menggunakan aplikasi web Damn Vulnerable Web Application (DVWA). Dengan berbagai tingkat kerentanan yang dapat dikonfigurasi, DVWA digunakan sebagai platform pengujian karena memungkinkan evaluasi teknik keamanan yang menyeluruh. Eksperimen dilakukan sebelum dan sesudah penerapan Parameterized Queries. Hasil menunjukkan bahwa menggunakan Parameterized Queries secara signifikan mampu mencegah berbagai jenis serangan *SQL Injection* yang diuji, termasuk serangan dasar dan serangan yang lebih kompleks. Penggunaan metode Parameterized Queries saat membuat aplikasi web untuk meningkatkan keamanan dan menghindari eksploitasi oleh penyerang. Dengan menggunakan teknik ini untuk mengamankan input data, risiko kebocoran data dan kompromi sistem dapat diminimalkan. Studi ini menemukan bahwa metode terbaik yang harus digunakan pengembang untuk melindungi aplikasi web dari ancaman *SQL Injection* adalah Parameterized Queries.

Kata Kunci: DVWA, Parameter Kueri, Pengujian Penetrasi, SQLI, Web Aplikasi

ABSTRACT

One of the most common types of SQL Injection attacks that threaten web applications uses security holes in input data to execute unwanted SQL commands. Parameterized Queries technique, also known as crafted statements, is considered as one of the best ways to address this issue. The purpose of this research is to evaluate how effective the parameterized question method is in preventing SQL Injection attacks using the Damn Vulnerable Web Application (DVWA) web application. With various configurable vulnerability levels, DVWA was used as a testing platform as it allows a thorough evaluation of security techniques. Experiments were conducted before and after the implementation of Parameterized Queries. Results showed that using Parameterized Queries was significantly able to prevent the different types of SQL Injection attacks tested, including basic attacks and more complex attacks. The use of Parameterized Queries method when creating web applications to increase security and avoid exploitation by attackers. By using this technique to secure data input, the risk of data leakage and system compromise can be minimized. This study found that the best method developers should use to protect web applications from SQL Injection threats is Parameterized Queries.

Keywords: *DVWA, Parameterized Queries, Penetrasi Testing, SQLI, Web Application*

PENDAHULUAN

Haikal Muhammad et al, 2023 (dalam Bangkit Wiguna, et al, 2020) mengatakan pada saat ini banyak jenis serangan web saat ini terjadi. Salah satu yang sering digunakan oleh penyerang adalah *SQL Injection*. Serangan *SQL Injection* adalah jenis serangan injeksi kode di mana penyerang mengeksekusi perintah SQL berbahaya dengan menggunakan input yang tidak divalidasi oleh aplikasi. Ini memberi penyerang kemampuan untuk mengakses, mengubah, atau menghapus data yang tersimpan di basis data.

Menurut penelitian yang dilakukan (Risky & Yuhandri, 2021:216),Teknik *SQL Injection* (SQLI) memeriksa keamanan situs web dengan memanfaatkan elemen bahasa markup HTML atau karakter khusus yang dimaksudkan untuk memanipulasi kueri SQL pada *database*. Teknik ini dapat menyebabkan sejumlah masalah besar, seperti pencurian data sensitif dan kerusakan system. Serangan SQLI in-band, inferential (blind), dan out-of-band memiliki metode dan tingkat risiko yang berbeda. Serangan SQLI dapat dicegah dengan menggunakan pernyataan yang dibuat, validasi input yang ketat, dan penggunaan prosedur penyimpanan.

Menurut (Mutedi & Tjahjono, 2022:151), Di sisi *Server*, ada banyak prosedur keamanan yang ada, namun kurang efektif karena sulit untuk diterapkan. Di sisi klien, pemasangan aplikasi keamanan hanya memperburuk kondisi klien. Namun, konfigurasi dan pemeliharaan aplikasi keamanan ini mungkin membingungkan bagi pengguna yang tidak terbiasa dengan teknologi, menambah beban dan meningkatkan risiko kesalahan penggunaan. Untuk meningkatkan efektivitas tanpa mengorbankan kinerja dan kenyamanan pengguna, penelitian ini menekankan kebutuhan akan solusi keamanan yang lebih sederhana dan efektif untuk *Server* dan aplikasi klien yang lebih ringan dan mudah digunakan.

Ada juga penelitian yang dilakukan (Sampurna, 2022:103), Uji Penetrasi adalah metode pengujian keamanan aplikasi secara manual yang optimal untuk aplikasi penting, terutama yang sering mengalami perubahan besar. Evaluasi tersebut mencakup pemeriksaan logika bisnis dan pengujian yang berfokus pada skenario serangan untuk mengidentifikasi potensi serangan yang lebih canggih. Serangan *SQL Injection* dapat mengakibatkan kerugian finansial selain mengancam reputasi organisasi dan privasi data pengguna.Aplikasi web tersimpan pada web *Server* dan dieksekusi oleh web *Server*, dan merupakan salah satu media yang paling umum digunakan untuk pengiriman data melalui internet saat ini.

Dalam mengatasi serangan *SQL Injection*, Teknik yang telah dikembangkan, salah satunya Parameterized Queries atau prepared statements, adalah salah satu metode mitigasi yang telah dikembangkan untuk mengatasi ancaman tersebut. Teknik ini menghindari eksekusi perintah SQL berbahaya dengan memisahkan logika SQL dari data yang dimasukkan oleh pengguna. Dengan menggunakan pertanyaan parameterisasi, setiap input pengguna dianggap sebagai parameter dan bukan sebagai bagian dari perintah SQL, yang menghilangkan kemungkinan injeksi kode.

Menurut penelitian yang dilakukan (Abdullah, 2020), Peneliti melakukan proses pemeriksaan secara otomatis dengan menggunakan pemindai kerentanan aplikasi web. Proses ini terdiri dari tiga komponen: pertama, pengumpulan data dari situs web; kedua, pengirim input tidak valid secara acak oleh penyerang ke aplikasi web; ketiga, analisis data untuk deteksi kerentanan dan penyusunan laporan. Dengan menggunakan proses ini, pengembang dapat meningkatkan keamanan aplikasi web dan melindungi data pengguna

dengan menemukan dan memperbaiki masalah keamanan sebelum mereka dapat digunakan oleh penyerang nyata.

Penelitian ini bertujuan untuk mengevaluasi seberapa efektif metode pertanyaan parameterisasi dalam mencegah serangan *SQL Injection*. Untuk platform pengujian, studi ini menggunakan Damn Vulnerable Web Application (DVWA). DVWA dipilih karena kemampuan untuk mensimulasikan berbagai tingkat kerentanan keamanan aplikasi web, yang memungkinkan penilaian menyeluruh terhadap metode mitigasi yang digunakan.

Penelitian ini akan melihat seberapa baik teknik Parameterized Queries melindungi aplikasi web dari berbagai jenis serangan *SQL Injection*. Penelitian ini juga akan mendorong pengembang dan organisasi untuk menerapkan praktik keamanan yang lebih baik. Selain itu, penelitian ini juga diharapkan dapat meningkatkan pemahaman tentang pentingnya keamanan aplikasi web dalam era digital yang semakin kompleks serta memberikan saran praktis untuk memperkuat infrastruktur keamanan siber. Selain itu, temuan penelitian ini diharapkan dapat membantu mengembangkan standar keamanan industri yang lebih ketat dan efisien.

TINJAUAN PUSTAKA

Website

Menurut (Asmara, 2019). Website adalah kumpulan halaman web di dalam suatu domain yang menyediakan informasi.Sebuah website terdiri dari sejumlah halaman web yang saling terkait dan diakses melalui suatu domain tertentu. Setiap halaman web pada website tersebut berfungsi untuk menyampaikan berbagai jenis informasi kepada pengunjung, mulai dari teks, gambar, video, hingga elemen interaktif lainnya. Definisi ini mencakup esensi utama dari sebuah website, yaitu sebagai media digital yang menyajikan dan mengorganisasi informasi secara terstruktur dan dapat diakses secara online.

Internet

Pada penelitian yang disampaikan oleh (Huda & Priyatna, 2019). Internet adalah bukti nyata dari kemajuan teknologi yang saat ini memiliki pengaruh besar pada kehidupan manusia. Internet juga memainkan peran penting dalam penyebaran informasi. Internet telah menjadi bagian integral dari kehidupan sehari-hari, mempengaruhi berbagai aspek, mulai dari komunikasi hingga bisnis, pendidikan, dan hiburan. Peran internet dalam penyaluran informasi sangat krusial karena memungkinkan akses cepat dan luas ke berbagai sumber informasi, sehingga mendukung penyebaran pengetahuan dan perkembangan masyarakat secara global. Dengan demikian, internet bukan hanya sekadar teknologi, tetapi juga merupakan fondasi penting bagi pertukaran informasi dan konektivitas global.

Web Server

Rahmadani et al 2024 (dalam Josi, 2017). Mengatakan *Server internet* adalah perangkat lunak yang menerima permintaan dari pengguna melalui HTTPS atau HTTP, yang sering disebut sebagai *browser*, kemudian mengembalikan outputnya sebagai halaman web, biasanya dalam bentuk dokumen HTML. *Server* ini menjalankan peran krusial dalam komunikasi web dengan memproses permintaan, mengambil data yang diperlukan, dan merangkainya menjadi halaman yang dapat ditampilkan oleh browser. Dengan mendukung berbagai elemen seperti HTML, CSS, dan JavaScript, *Server* memastikan tampilan dan interaktivitas situs web. Selain itu, *Server* internet harus mampu

menangani banyak permintaan secara simultan, menjaga keamanan, dan memastikan keandalan serta ketersediaan layanan, menjadikannya komponen kunci dalam infrastruktur digital.

Web Browser

Risaldy dan Hardinata, 2023 (dalam Sophian, 2020). Menyatakan bahwa web browser adalah perangkat utama yang digunakan untuk menampilkan halaman web yang pada dasarnya terbuat dari HTML dan CSS. Saya yakin setiap sistem operasi telah dilengkapi dengan web browser bawaan seperti Internet Explorer (Windows), Safari (Mac), dan Firefox (Linux Ubuntu). Keberadaan browser bawaan ini memudahkan pengguna dalam menjelajahi web tanpa perlu mengunduh atau menginstal browser tambahan, memastikan pengalaman pengguna yang lancar dan efisien dalam menjelajahi dunia maya.

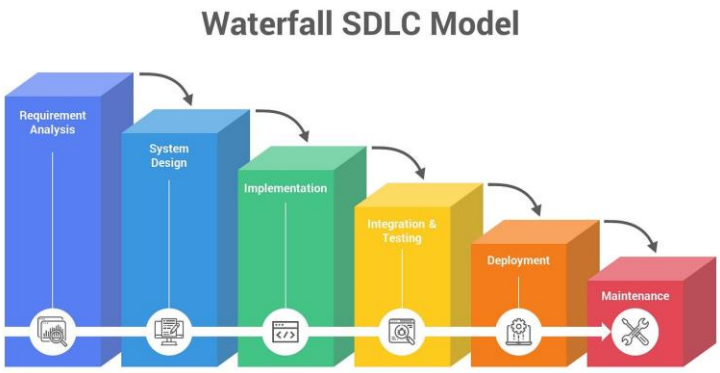
SQL Injection

Nursapdahi et al, 2022 (dalam Justin Clarke, 1967). Mengatakan bahwa *SQL Injection* adalah metode eksploitasi aplikasi web dengan menggunakan data yang disediakan atau disisipkan dalam query SQL. *SQL Injection* bekerja dengan menyisipkan query SQL atau perintah sebagai input melalui halaman web atau command prompt. Penting bagi pengembang web untuk memahami ancaman ini dan menerapkan langkah-langkah pencegahan yang efektif, seperti validasi input yang ketat dan penggunaan pernyataan yang dipersiapkan (prepared statements). Kesadaran dan pendidikan yang lebih baik tentang teknik-teknik pengamanan ini dapat membantu mencegah serangan *SQL Injection* dan melindungi integritas serta kerahasiaan data.

Menurut (Pratama et al., 2022), Teknik *SQL Injection* merupakan salah satu ancaman paling serius dalam keamanan aplikasi web. Melalui eksploitasi kelemahan dalam validasi input, penyerang dapat mengakses basis data situs web dan melakukan berbagai tindakan berbahaya, termasuk pencurian data sensitif dan manipulasi data. Konsekuensi dari serangan ini bisa sangat merusak, baik bagi pemilik situs web maupun penggunanya, karena data yang dicuri dapat mencakup informasi pribadi, kredensial login, dan data finansial. Oleh karena itu, sangat penting bagi pengembang web untuk menerapkan praktik pengkodean yang aman, seperti menggunakan pernyataan yang dipersiapkan dan parameter yang terikat, serta melakukan pengujian keamanan yang menyeluruh sebelum meluncurkan aplikasi. Pencegahan yang proaktif dapat mengurangi risiko serangan *SQL Injection* dan melindungi integritas serta kerahasiaan data.

METODE PENELITIAN

Metode yang digunakan pada penelitian ini menggunakan metode SDLC (*System Development Life Cycle*) (Rahmadani et al., 2024)(Mallisza et al., 2022), Teknik pengumpulan data yang digunakan yaitu dengan analisis, desain, pengujian dan pemeliharaan.



Gambar 1. Tahapan Model Waterfall

Metode ini mempunyai beberapa tahapan sebagai berikut:

1. *Requirement Analysis* (Analisis Kebutuhan), bertujuan untuk menemukan dan mencatat kebutuhan sistem dan keamanan dengan mengumpulkan informasi tentang *SQL Injection* dan parameterized queries, serta menentukan standar keberhasilan pengujian.
2. System Design (Desain Sistem), merancang mekanisme serangan *SQL Injection*, implementasi parameterized queries dalam aplikasi DVWA, serta diagram alir dan diagram *entity-relationship* jika diperlukan.
3. Implementation (Implementasi), fokus pada pembuatan dan implementasi sesuai dengan desain yang telah dibuat, seperti membuat konfigurasi lingkungan pengujian dengan DVWA, menerapkan parameterized queries, dan membuat skrip serangan *SQL Injection.*
4. *Integration and Testing* (Integrasi dan Pengujian), komponen sistem akan dipasang dan diuji untuk memastikan bahwa semua fungsi berjalan dengan baik. Untuk mengevaluasi efektivitas parameterized queries, pengujian fungsional dan serangan *SQL Injection* akan dilakukan.
5. *Deployment* (Penempatan), mencatat hasil penelitian atau menyebarkan sistem ke lingkungan yang lebih luas jika diperlukan.
6. *Maintenance* (Pemeliharaan), memelihara dan memperbarui sistem berdasarkan umpan balik dan hasil pengujian, memperbaiki kesalahan, dan membuat rekomendasi untuk pengembangan lebih lanjut.

HASIL DAN PEMBAHASAN

Konfigurasi DVWA

Pada tahap ini, sama seperti yang dilakukan sebelumnya, konfigurasi DVWA dilakukan pada perangkat yang akan menjadi target serangan. Untuk menjalankan aplikasi web DVWA, langkah-langkah berikut harus diikuti:

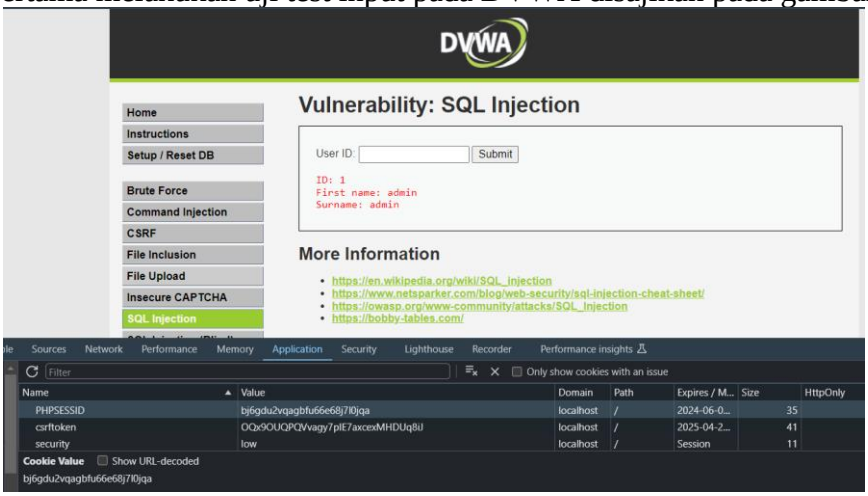
1. *Download file* DVWA, mendownload *source code* DVWA dari pembuat aplikasi web DVWA, yang bisa dicari melalui Google, DuckDuckGo, atau *search engine* lainnya.
2. Konfigurasi DVWA, Setelah mendownload file DVWA, pindahkan file ke lokasi *database* perangkat, dilanjutkan dengan pengaturan file konfigurasi DVWA untuk mengidentifikasi pengguna, nama, dan password *database* yang digunakan.
3. Konfigurasi MySQL, Kita perlu membuat pengguna baru pada MySQL dan memberi

hak akses kepada pengguna tersebut untuk mengakses *database* DVWA yang telah dikonfigurasi.

4. Konfigurasi Apache2, Pada tahap ini, perlu melakukan konfigurasi pada Apache2 untuk menyalakan *Server* agar *database* DVWA dapat diakses.
5. Mulai DVWA, Pada tahap ini, kita memulai layanan yang menjalankan DVWA sehingga DVWA dapat digunakan.

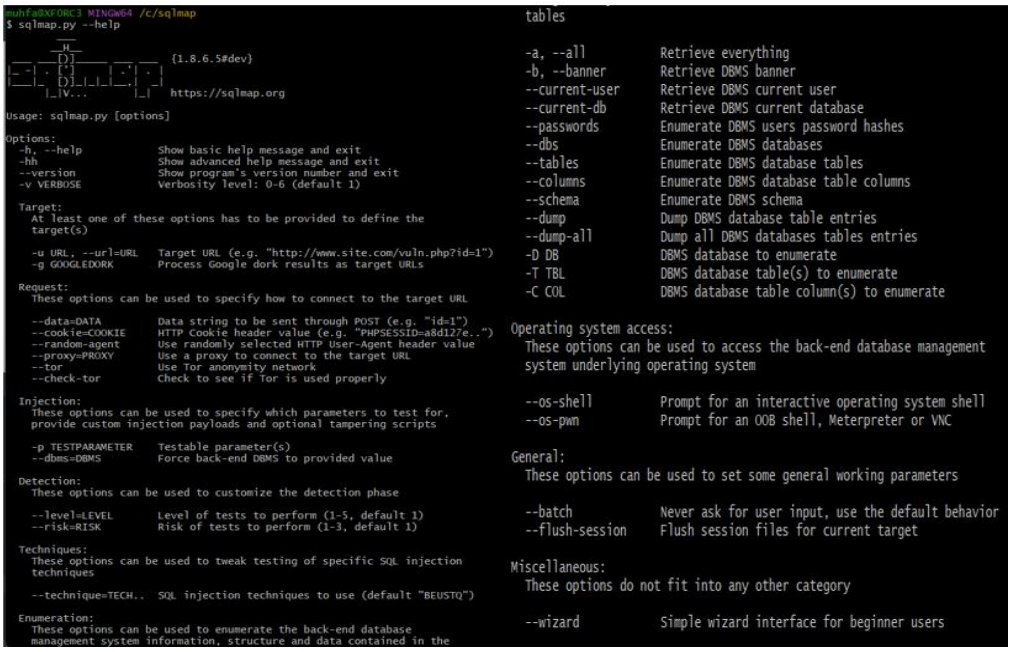
Mencari Kerentanan Menggunakan SQLMap

Pada tahap ini menggunakan tools SQLMap untuk mencari kerentanan pada DVWA, kita akan mencari informasi dari *database* yang sering digunakan oleh *attacker*. Tahapan pertama melakukan uji test input pada DVWA disajikan pada gambar 2.



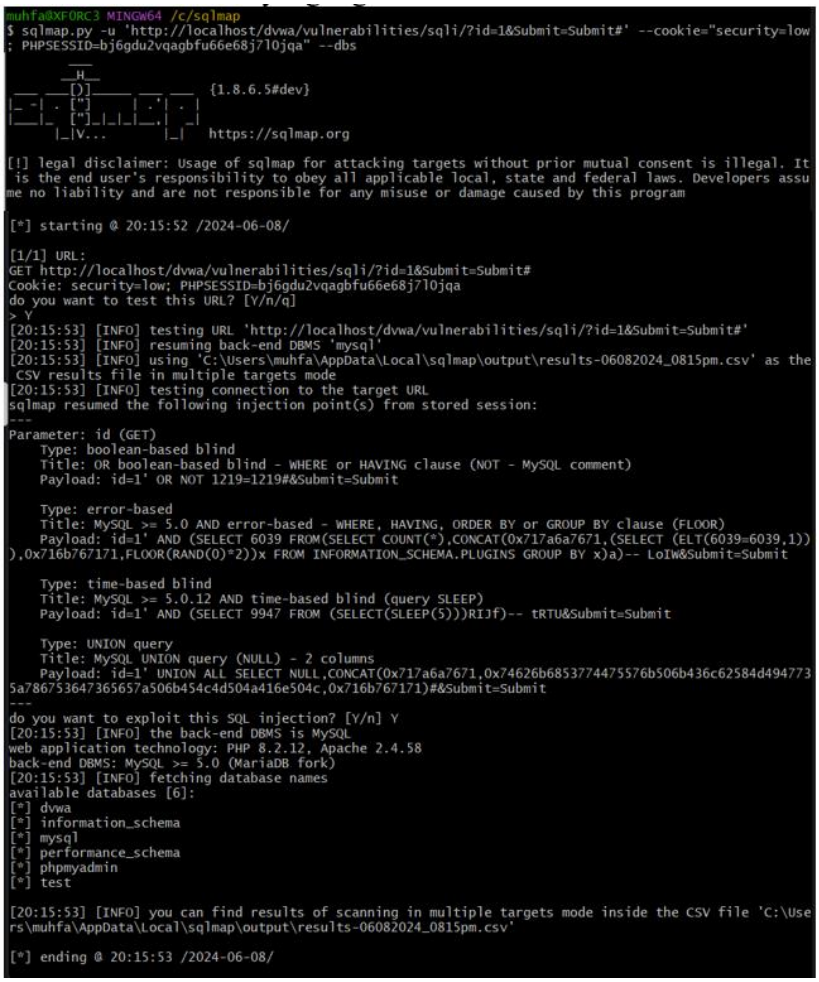
Gambar 2. Menampilkan hasil input

Pada gambar 2 diatas melakukan uji test input menggunakan ID 1 dan mendapatkan *cookies* pada DVWA serta menggunakan security "*low*". Tahapan kedua melakukan perintah "*sqlmap –help*" disajikan pada gambar 3.



Gambar 3. Menampilkan Perintah *Tools Sqlmap*

Pada gambar 3 menampilkan perintah untuk menggunakan *tools sqlmap* dengan menggunakan perintah "*sqlmap –help*". Mencari nama *database* yang digunakan disajikan pada gambar 4.



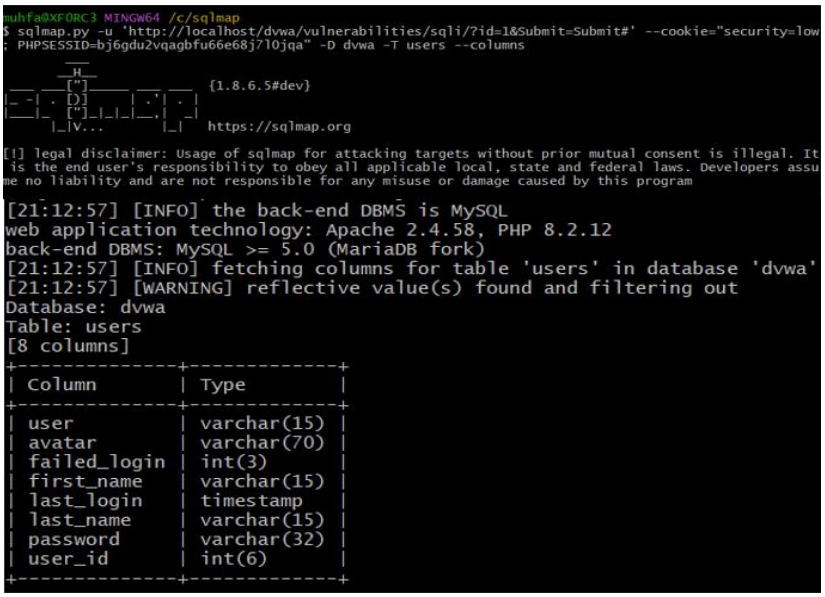
Gambar 4. Menampilkan Info *Database* dan Kerentanan *SQL Injection*

Pada gambar 4 terdapat informasi dari *database* yang digunakan dan terdapat kerentanan terhadap *SQL Injection* pada parameter id dengan metode (*GET*). Mencari isi *database* yang digunakan disajikan pada gambar 5.

Gambar 5. Menampilkan Informasi Dari Isi *Database*

Pada gambar 5 mendapatkan informasi *database* dari dvwa, terdapat tabel dari

guestbook dan *users*. Mencari isi dari tabel *users* yang terdapat pada *database* disajikan pada gambar 6.



Gambar 6. Menampilkan Informasi dari Isi Tabel *Users*

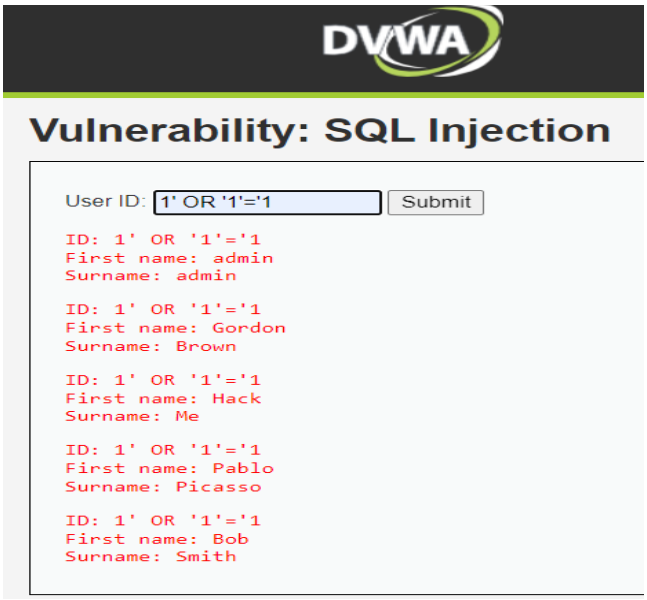
Pada gambar 6 terdapat informasi dari nama pengguna, avatar, *user* dan *password* pengguna untuk melakukan login. Mencari *user* dan *password* disajikan pada gambar 7.

Gambar 7. Menampilkan *User* Dan *Password* yang Digunakan

Pada gambar 7 terdapat informasi dari *user* dan *password* yang digunakan untuk melakukan login, akan tetapi untuk *password terencrypt*.

Melakukan Test Kerentanan *SQL Injection* Pada DVWA

Pada tahap ini melakukan test kerentanan *SQL Injection* dengan menggunakan *input parameter query* pada menu input yang ada pada DVWA.

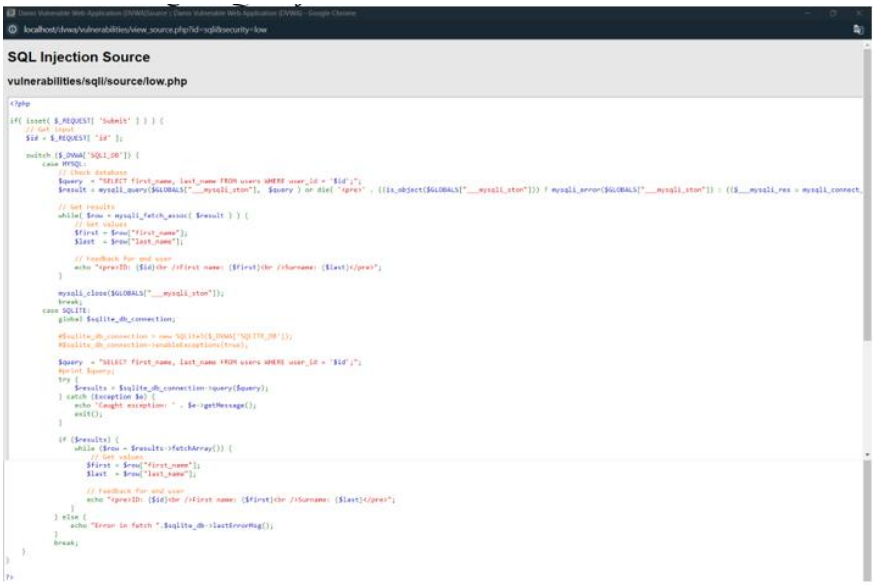


Gambar 8. Test Kerentanan *Input Parameter Query*

Pada gambar 8 diketahui bahwa menu input parameter pada dvwa terdapat kerentanan yang dapat menyebabkan informasi yang lain dapat ditampilkan.

Analisis dan Mengatasi Serangan *SQL Injection*

Pada tahap ini melakukan analisis dan mengatasi serangan *SQL Injection* dengan mengubah *input parameter query* pada menu input yang ada pada DVWA. Analisis serangan *SQL Injection* disajikan pada gambar 9 berikut.



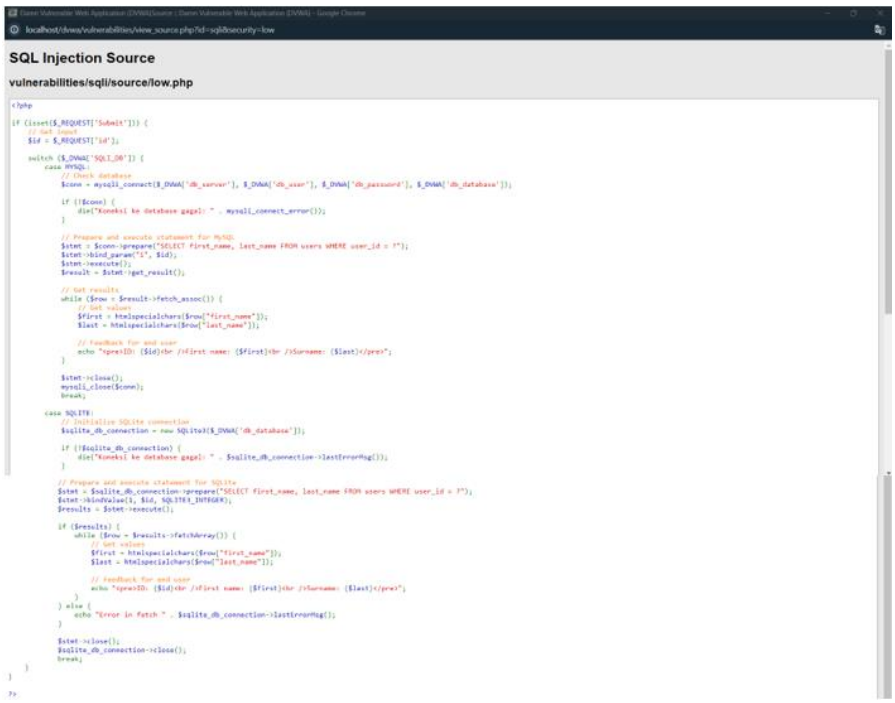
Gambar 9. Source code *SQL Injection* pada DVWA

Pada gambar 9 terdapat beberapa kesalahan untuk mengatasi serangan *SQL Injection*, antara lain sebagai berikut:

1. Penggunaan Variabel Tidak Divalidasi: Variabel $id yang digunakan dalam query

tidak dibersihkan atau divalidasi dengan benar sebelum digunakan. Ini memungkinkan adanya kerentanan terhadap serangan *SQL Injection*.

2. Query yang Tidak Aman: Pada bagian MySQL, query dibentuk secara langsung menggunakan nilai $id. Ini memungkinkan penggunaan $id secara langsung tanpa dilindungi oleh parameterized queries atau fungsi pembatasan lainnya. Penyerang dapat memanfaatkannya untuk memasukkan perintah SQL tambahan.
3. Koneksi SQLite Tidak Aman: Pada bagian SQLite, penggunaan fungsi permintaan membuatnya tidak langsung rentan terhadap serangan *SQL Injection*, tetapi penggunaan $id secara langsung dalam query tetap dapat menimbulkan kerentanan. Selain itu, koneksi ke *database* harus diatur dengan benar agar informasi koneksi tidak terlihat dalam kode jika menggunakan SQLite.
4. Pesan Kesalahan yang Tidak Terkontrol: Pesan kesalahan yang ditampilkan kepada pengguna dalam kasus kesalahan dapat mengungkapkan informasi sensitif tentang struktur *database* atau detail implementasi lainnya. Ini dapat membantu penyerang merencanakan dan melakukan serangan.
5. Penanganan Kesalahan yang Tidak Tepat: Pesan kesalahan yang ditampilkan kepada pengguna tidak jelas dan tidak memberikan informasi yang berguna karena penanganan kesalahan pada bagian SQLite tidak memadai.
6. Kelemahan Pengaturan *Security*: Penggunaan fungsi include atau require yang tidak aman dalam kode juga dapat menjadi masalah keamanan, tergantung pada bagaimana $_DVWA['SQLI_DB'] diatur.



Gambar 10. Source code untuk mengatasi *SQL Injection* pada DVWA

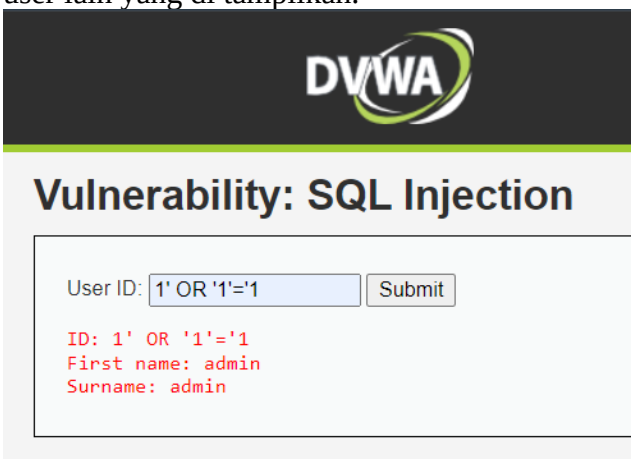
Mengatasi serangan SQL Injection pada gambar 10 terdapat perbaikan dari source code *SQL Injection* pada DVWA dengan menambahkan beberapa Langkah untuk mengatasi *SQL Injection*, sebagai berikut:

1. Gunakan Permintaan Parameter: Gunakan permintaan yang dibuat atau permintaan

parameter untuk menggantikan metode langsung penggabungan input sensitif ke dalam permintaan.

2. Validasi Input: Validasi input pengguna untuk memastikan bahwa aplikasi hanya menerima nilai yang diharapkan.
3. Pesan Kesalahan yang Dikelola dengan Benar: Berikan pengguna pesan kesalahan yang terkontrol dan informatif tanpa memberikan informasi berlebihan yang dapat dimanfaatkan oleh penyerang.
4. Koneksi *Database* yang Aman: Gunakan konfigurasi yang aman, seperti menyimpan informasi koneksi dalam file konfigurasi terpisah yang tidak dapat diakses langsung dari web, untuk memastikan bahwa koneksi *database* dilakukan secara aman dan informasi koneksi tidak dapat dilihat dalam kode.
5. Validasi Konfigurasi DVWA: Pastikan bahwa pengaturan lingkungan, seperti nilai $_DVWA['SQLI_DB'], diatur dengan benar dan mencegah penyerang menerima akses yang tidak perlu.

Melakukan Test Kerentanan Ulang *SQL Injection* Pada DVWA, Pada gambar 11 terdapat perbaikan untuk input parameter query yang hanya menampilkan user id yang di input dan tidak ada user lain yang di tampilkan.



Gambar 11. Test Ulang Kerentanan *SQL Injection*

Pembahasan

Dalam sistem keamanan website menggunakan DVWA. Internet adalah bukti dari kemajuan teknologi yang saat ini berpengaruh besar terhadap manusia. Internet juga memainkan peran yang sangat vital dalam proses penyaluran informasi (Huda & Priyatna, 2019). Aplikasi website harus menggunakan keamanan dari serangan yang akan terjadi. Pada penelitian ini system keamanan web aplikasi DVWA menggunakan *SQL Injection*, SQL Ijection merupakan sebuah teknik pengeksploitasian sebuah aplikasi web memakai data yang diberikan atau yang disisipkan dalam query SQL. *SQL Injection* bekerja dengan cara memasukkan query SQL atau perintah sebagai input yang dimungkinkan melalui halaman web atau command prompt (Nursapdahi et al., 2022).

Hasil pengujian web aplikasi DVWA dengan menggunakan *SQL Injection* situs untuk melakukan uji aplikasi dalam system keamanan dapat disimpulkan bahwa terdapat beberapa kerentanan yang ada pada aplikasi DVWA, hal ini dikarenakan metode *SQL Injection* digunakan untuk memasukkan perintah SQL sebagai input pada sebuah website untuk mengakses *database*. Dengan memperoleh akses ke basis data suatu situs web,

penyerang dapat dengan mudah mengambil data sensitif yang beragam serta memanipulasi atau bahkan menghancurkan data di dalam situs web tersebut (Pratama et al., 2022).

Dalam analisis serangan dan keamanan *SQL Injection*, ditemukan bahwa integritas URL yang terenkripsi lebih terjaga keamanannya karena metode injeksi SQL tidak dapat digunakan terhadap URL yang terenkripsi. (Rahmadani et al., 2024).

PENUTUP

Simpulan

Berdasarkan eksperimen dan analisis pada platform Damn Vulnerable Web Application (DVWA), penelitian ini menyimpulkan bahwa teknik query berparameter sangat efektif dalam mencegah serangan injeksi SQL. Teknik ini memisahkan logika SQL dari data yang dimasukkan oleh pengguna, sehingga setiap masukan dianggap sebagai parameter aman dan struktur atau tujuan kueri SQL tidak dapat diubah. Hasilnya menunjukkan bahwa penggunaan kueri berparameter secara signifikan mengurangi risiko injeksi kode berbahaya dan meningkatkan keamanan aplikasi web secara keseluruhan. Meskipun terdapat tantangan implementasi, seperti mengadaptasi kode yang ada, teknik ini relatif mudah diterapkan dan tidak memerlukan perubahan besar pada arsitektur aplikasi. Oleh karena itu, pengembang dan organisasi didorong untuk mengadopsi kueri berparameter sebagai praktik standar untuk meningkatkan keamanan dan mengurangi beban kerja yang terkait dengan manajemen keamanan manual. Penelitian ini bertujuan untuk memberikan bukti empiris yang menunjukkan pentingnya teknologi ini dalam pengembangan aplikasi web yang aman dan untuk mendorong penerapan praktik keamanan yang lebih baik serta pengembangan standar keamanan industri yang lebih ketat

Saran

Untuk peneliti selanjutnya diharapkan lebih memfokuskan pada pengembangan teknik query berparameter dalam mencegah serangan injeksi SQL pada berbagai platform aplikasi web. Penelitian selanjutnya juga dapat mengeksplorasi cara-cara untuk mengatasi tantangan implementasi yang mungkin terjadi, serta mengevaluasi dampak penggunaan teknik ini terhadap performa dan keamanan aplikasi secara keseluruhan.

REFERENSI

Abdullah, H. S. (2020). Evaluation of Open Source Web Application Vulnerability Scanners. *Academic Journal of Nawroz University*, *9*(1), 47. https://doi.org/10.25007/ajnu.v9n1a532

Asmara, J. (2019). Rancang Bangun Sistem Informasi Desa Berbasis Website (Studi Kasus Desa Netpala). *Jurnal Pendidikan Teknologi Informasi*, *2*(1), 1–7. https://ojs.cbn.ac.id

Haikal Muhammad, H., Id Hadiana, A., & Ashaury, H. (2023). Pengamanan Aplikasi Web Dari Serangan *SQL Injection* Dan Cross Site Scripting Menggunakan Web Application Firewall. *Jurnal Mahasiswa Teknik Informatika*, *7*(5), 3265–3273.

Huda, B., & Priyatna, B. (2019). Penggunaan Aplikasi Content Manajement System (CMS) Untuk. *SYSTEMATICS*, *1*(2), 81–88. https://journal.unsika.ac.id/

Mallisza, D., Hadi, H. S., & Aulia, A. T. (2022). Implementasi Model Waterfall Dalam Perancangan Sistem Surat Perintah Perjalanan Dinas Berbasis Website Dengan Metode SDLC. *Jurnal Teknik, Komputer, Agroteknologi Dan Sains*, *1*(1), 24–35. https://doi.org/10.56248/marostek.v1i1.9

Mutedi, A., & Tjahjono, B. (2022). Systematic Literature Review: Preventing *SQL Injection* Attacks Using Tools OWASP CSR Web Application Firewall. *Jurnal Informatika Universitas Pamulang*, *7*(1), 151–156. https://doi.org/10.32493/informatika.v7i1.17590

Nursapdahi, Senja Fitrani, A. S., Rosid, M. A., & Aji, S. (2022, July 23). STUDI ANALISA SERANGAN *SQL INJECTION*. *Seminar Nasional Inovasi Teknologi*, 185–190. https://proceeding.unpkediri.ac.id/

Pratama, T. I. M., Songida, M. D. F., & Gunawan, I. (2022). Analisis Serangan dan Keamanan pada *SQL Injection*: Sebuah Review Sistematik. *Jurnal Ilmiah Informatika & Komputer*, *1*(2), 27–32. https://www.sttrcepu.ac.id/

Rahmadani, N. L., Lamada, M. S., & Dewi, S. S. (2024). Desain Dan Analisis Sistem Keamanan Dari Serangan *SQL Injection* Dalam Sistem Pendataan Tagihan Sampah. *PINISI JOURNAL OF SCIENCE AND TECHNOLOGY*, *1*(1), 36–49. https://ojs.unm.ac.id/

Risaldy, H. A., & Hardinata, R. S. (2023). Rancang Bangun Sistem Informasi Menu makanan Berbasis Web (Studi Kasus: Rumah Makan Sipirok). *Jurnal Teknologi Sistem Informasi Dan Sistem Komputer TGD*, *6*(2), 539–548. https://ojs.trigunadharma.ac.id/index.php/jsk/index

Risky, M. A. Z., & Yuhandri. (2021). Optimalisasi dalam Penetrasi Testing Keamanan Website Menggunakan Teknik *SQL Injection* dan XSS. *Jurnal Sistim Informasi Dan Teknologi*, *3*(4), 215–220. https://doi.org/10.37034/jsisfotek.v3i4.68

Sampurna, M. R. (2022). NetPLG Journal of Network and Computer Applications Implementasi Hydra, FFUF, dan WFUZZ dalam Brute Force DVWA. *Journal of Network and Computer*, *1*(2), 102–112. https://jurnal.netplg.com/