Digital Transformation Technology (Digitech) | e-ISSN: 2807-9000
Volume 5, **Number** 2, September 2025
https://doi.org/10.47709/digitech.v5i2.6948

Analisis Kerentanan Insecure Direct Object Reference (IDOR) dan Broken Access Control pada Aplikasi Invoice Berbasis Web

Kusnana[1], Prayoga Gymnastiar[2], Beny Riswanto[3]
[1,2,3]STMIK Komputama Majenang
[1]nanakusnana@stmikkomputama.ac.id, [2]prayoga.gymnastiar15@email.com,
[3]benyriswanto@stmikkomputama.ac.id

open access
Histori Artikel:
Diajukan: 10 September 2025
Disetujui: 11 Oktober 2025
Dipublikasi: 12 Oktober 2025
Kata Kunci:
Keamanan Aplikasi Web, IDOR, Broken Access Control, Sensitive Data Exposure, Penetration Testing

Abstrak
Keamanan aplikasi web merupakan aspek krusial dalam melindungi data sensitif pengguna sekaligus menjaga kepercayaan terhadap layanan digital. Seiring meningkatnya pemanfaatan aplikasi berbasis web, risiko serangan siber juga semakin kompleks, terutama terkait kontrol akses dan kerahasiaan data. Penelitian ini bertujuan untuk mengidentifikasi serta menganalisis kerentanan keamanan pada aplikasi invoice berbasis web dengan alamat https://invoice.masdzikry.com. Metode penelitian menggunakan pendekatan eksperimen melalui penetration testing, dengan bantuan Burp Suite untuk melakukan manipulasi permintaan (request), analisis parameter, serta pengujian bypass otorisasi. Hasil pengujian mengungkapkan dua kerentanan utama yang bersifat kritis. Pertama, Insecure Direct Object Reference (IDOR), di mana pengguna biasa dapat mengakses faktur milik pengguna lain hanya dengan memodifikasi parameter id pada endpoint /invoices/{id}. Kedua, Broken Access Control yang disertai dengan Sensitive Data Exposure, ditemukan melalui penyisipan header X-Original-Url yang memungkinkan pengguna tanpa hak istimewa memperoleh akses tidak sah ke halaman administratif serta data sensitif administrator. Kedua temuan ini memiliki tingkat risiko tinggi menurut klasifikasi OWASP, karena berpotensi menyebabkan kebocoran data finansial, penyalahgunaan akses, hingga pengambilalihan akun. Berdasarkan temuan tersebut, penelitian ini merekomendasikan penerapan kontrol otorisasi berbasis peran (Role-Based Access Control), validasi parameter di sisi server, penonaktifan header manipulatif, serta penerapan logging, monitoring, dan audit keamanan secara berkala sesuai dengan standar OWASP Application Security Verification Standard (ASVS). Implementasi langkah-langkah mitigasi ini diharapkan dapat meningkatkan ketahanan aplikasi terhadap ancaman eksploitasi dan memperkuat kepercayaan pengguna terhadap sistem.

PENDAHULUAN

Perkembangan teknologi informasi pada era digital telah membawa dampak besar terhadap berbagai sektor, terutama sektor bisnis. Transformasi digital memungkinkan organisasi meningkatkan efisiensi operasional, memperluas jangkauan pasar, serta meningkatkan skalabilitas layanan yang ditawarkan. Menurut Laudon dan Laudon (2020), teknologi informasi telah menjadi salah satu faktor utama dalam penciptaan nilai bisnis modern, terutama melalui pemanfaatan aplikasi web. Aplikasi berbasis web kini menjadi tulang punggung layanan digital, mulai dari e-commerce, perbankan daring, layanan publik, hingga sistem manajemen dokumen yang digunakan dalam organisasi.

Namun, di balik manfaat tersebut, risiko keamanan aplikasi web juga meningkat secara signifikan. Laporan Verizon (2021) menunjukkan bahwa lebih dari 40% insiden kebocoran data berasal dari aplikasi web yang tidak aman. Hal ini menegaskan bahwa aplikasi web, meskipun berperan penting dalam mendukung aktivitas bisnis, juga menjadi target utama serangan siber. Serangan terhadap aplikasi web umumnya dilakukan dengan cara mengeksploitasi kelemahan pada mekanisme autentikasi dan otorisasi, penggunaan enkripsi yang lemah, serta celah dalam validasi input.

Salah satu permasalahan mendasar dalam keamanan aplikasi web adalah lemahnya otorisasi akses. Otorisasi yang seharusnya membedakan hak akses antara pengguna sah dengan pihak tidak berwenang sering kali diimplementasikan secara tidak konsisten. Anderson (2020) menekankan bahwa kelemahan dalam mekanisme otorisasi menjadi salah satu akar permasalahan dalam terjadinya kebocoran data dan pelanggaran integritas sistem. Banyak aplikasi gagal melakukan validasi pada sisi server, sehingga penyerang dapat memanfaatkan parameter sederhana dalam URL atau permintaan HTTP untuk memperoleh akses yang seharusnya dibatasi.

Kerentanan semacam ini menjadikan Insecure Direct Object Reference (IDOR) dan Broken Access Control sebagai salah satu isu keamanan paling menonjol. OWASP (2021) bahkan menempatkan kedua kerentanan tersebut ke dalam daftar sepuluh besar risiko keamanan aplikasi web (OWASP Top 10), menegaskan betapa seriusnya dampak yang dapat ditimbulkan. IDOR memungkinkan penyerang memperoleh akses langsung terhadap data atau sumber daya hanya dengan memodifikasi parameter yang ada, sementara Broken Access Control terjadi ketika sistem gagal membatasi tindakan pengguna sesuai dengan peran dan hak akses yang dimilikinya. Kedua celah tersebut tidak hanya mengancam kerahasiaan dan integritas data, tetapi juga berpotensi merusak kepercayaan pengguna terhadap sistem yang digunakan (Peltier, 2016).

Dalam konteks penelitian ini, aplikasi invoice.masdzikry.com dijadikan studi kasus nyata mengenai bagaimana kelemahan dalam implementasi kontrol akses dapat membuka peluang eksploitasi data sensitif. Aplikasi ini, yang berfungsi untuk mendukung pengelolaan transaksi dan pembuatan faktur, tentu menyimpan berbagai informasi penting, mulai dari data pelanggan hingga detail finansial. Apabila mekanisme keamanannya tidak dirancang dengan benar, risiko kebocoran data menjadi sangat tinggi. Hal ini tidak hanya membahayakan pengguna aplikasi, tetapi juga reputasi pengembang dan organisasi yang mengoperasikannya.

Untuk mengidentifikasi dan memahami potensi kelemahan tersebut, penelitian ini menggunakan pendekatan penetration testing. Menurut McGraw (2006), penetration testing merupakan bagian penting dari Secure Software Development Lifecycle (SSDLC) karena mampu memberikan gambaran nyata mengenai kondisi keamanan aplikasi. Melalui simulasi serangan yang terkontrol, pengujian penetrasi tidak hanya menemukan celah yang mungkin dieksploitasi oleh penyerang, tetapi juga memberikan wawasan mengenai seberapa efektif kontrol keamanan yang sudah diterapkan.

SANS Institute (2020) menekankan bahwa pengujian penetrasi perlu dilakukan secara rutin, mengingat pola serangan siber terus berkembang. Dengan demikian, organisasi tidak hanya dapat menutup kelemahan yang ditemukan, tetapi juga membangun mekanisme pertahanan yang lebih tangguh. Penelitian ini tidak hanya berfokus pada identifikasi kerentanan, tetapi juga menyajikan rekomendasi praktis yang dapat diadopsi untuk meningkatkan keamanan sistem serupa. Dengan pendekatan ini, hasil penelitian diharapkan dapat memberikan kontribusi nyata, baik dalam aspek akademis maupun praktis, khususnya dalam meningkatkan kesadaran akan pentingnya keamanan aplikasi web di era digital.

STUDI LITERATUR

Insecure Direct Object Reference (IDOR)

IDOR terjadi ketika aplikasi memberikan akses ke objek berdasarkan input pengguna, seperti parameter ID dalam URL, tanpa verifikasi otorisasi tambahan. Hal ini memungkinkan penyerang mengakses data milik pengguna lain hanya dengan mengubah nilai parameter (Fogie, 2010). Menurut Halfond et al. (2006), teknik manipulasi parameter seperti ini merupakan salah satu metode paling sering digunakan dalam eksploitasi aplikasi berbasis web. Viega dan McGraw (2001) menekankan bahwa kelemahan ini muncul akibat pengembang terlalu percaya pada client-side input.

Kelemahan IDOR termasuk dalam kategori kerentanan yang sangat berbahaya karena secara langsung berkaitan dengan kerahasiaan serta integritas data pengguna. Aplikasi web yang tidak menerapkan kontrol akses yang ketat seringkali menjadi sasaran eksploitasi melalui teknik ini. Dalam praktiknya, serangan IDOR umumnya memanfaatkan parameter yang dapat diubah secara manual, seperti user ID, document number, atau session token, yang seharusnya hanya bisa diakses oleh pemilik sahnya. Ketika mekanisme validasi otorisasi tidak diterapkan di sisi server, penyerang cukup melakukan parameter tampering untuk mendapatkan akses ilegal terhadap data sensitif milik pengguna lain.

Dari perspektif keamanan perangkat lunak, IDOR menunjukkan lemahnya penerapan prinsip least privilege dan defense in depth. Aplikasi yang hanya mengandalkan pemeriksaan otorisasi pada level antarmuka (client-side) sangat rentan terhadap eksploitasi karena input dari sisi klien tidak dapat dipercaya sepenuhnya (OWASP, 2021). Dengan demikian, setiap permintaan ke server seharusnya diverifikasi ulang apakah pengguna yang mengajukan permintaan benar-benar memiliki hak akses terhadap objek yang diminta.

Broken Access Control

Kontrol akses yang rusak (Broken Access Control) adalah kelemahan ketika pengguna dapat mengakses sumber daya atau fungsi yang seharusnya dibatasi. Rouse (2019) menjelaskan bahwa kelemahan ini sering kali muncul karena implementasi kontrol berbasis peran tidak dijalankan dengan benar. Whitman dan Mattord (2017) menegaskan bahwa kontrol akses adalah pilar utama dalam keamanan informasi, karena kegagalannya dapat memungkinkan penyerang bertindak seolah-olah mereka adalah administrator.

Kelemahan ini menjadi salah satu ancaman serius dalam keamanan aplikasi web modern. Menurut OWASP (2021), Broken Access Control menempati peringkat pertama dalam daftar OWASP Top 10 Web Application Security Risks, yang menunjukkan betapa dominannya kerentanan ini ditemukan pada berbagai sistem. Serangan

umumnya terjadi ketika aplikasi gagal membatasi apa saja yang bisa dilakukan oleh pengguna, misalnya pengguna biasa yang dapat mengakses fungsi administrasi, melihat atau mengubah data milik pengguna lain, hingga melakukan eskalasi hak akses.

Dari sisi teknis, kelemahan ini sering muncul akibat kurangnya penerapan validasi otorisasi pada setiap permintaan. Banyak pengembang hanya menambahkan kontrol akses pada antarmuka pengguna (UI-level security), tetapi lupa atau tidak konsisten melakukan pengecekan di sisi server. Kondisi ini membuat aplikasi rentan terhadap manipulasi URL, force browsing, maupun perubahan parameter untuk mendapatkan hak istimewa secara ilegal (Fernandes et al., 2016).

Dampak dari Broken Access Control sangatlah luas. Pada organisasi, serangan ini bisa menyebabkan kebocoran data, manipulasi informasi sensitif, kerugian finansial, bahkan pelanggaran hukum terkait perlindungan data. Oleh karena itu, pengembangan sistem harus menerapkan prinsip least privilege, kontrol berbasis peran yang konsisten, serta pengujian keamanan yang ketat sebelum sistem dioperasikan.

Sensitive Data Exposure

Kebocoran data sensitif adalah kondisi ketika informasi penting seperti data pribadi, kredensial, atau data keuangan tersimpan atau ditransmisikan tanpa perlindungan yang memadai (Stallings, 2019). OWASP (2017) mencatat bahwa Sensitive Data Exposure menjadi salah satu dari 10 risiko terbesar pada aplikasi web modern. Anderson (2020) menambahkan bahwa dampak kebocoran data dapat menghancurkan reputasi organisasi sekaligus memicu kerugian finansial besar.

Kebocoran data sensitif adalah kondisi ketika informasi penting—seperti data pribadi, kredensial, atau data keuangan tersimpan atau ditransmisikan tanpa perlindungan yang memadai (Stallings, 2019). OWASP (2017) mencatat bahwa Sensitive Data Exposure merupakan salah satu dari 10 risiko terbesar pada aplikasi web modern. Anderson (2020) menambahkan bahwa dampak kebocoran data dapat menghancurkan reputasi organisasi sekaligus memicu kerugian finansial yang signifikan.

Kerentanan ini umumnya terjadi karena penerapan mekanisme enkripsi yang lemah atau bahkan tidak adanya enkripsi sama sekali pada data yang bersifat rahasia. Misalnya, penggunaan protokol HTTP tanpa SSL/TLS, penyimpanan kata sandi dalam bentuk teks biasa (plaintext), serta penggunaan algoritma kriptografi yang sudah usang. Kondisi ini membuka peluang bagi penyerang untuk melakukan intersepsi, man-in-the-middle attack, atau pencurian data melalui akses tidak sah (Sharma & Joshi, 2020).

Selain faktor teknis, kebocoran data juga sering dipicu oleh kelemahan tata kelola keamanan informasi. Banyak organisasi tidak memiliki kebijakan pengelolaan data yang memadai, sehingga data sensitif tidak diklasifikasikan atau diperlakukan sesuai tingkat kerahasiaannya. Padahal, dalam era transformasi digital, data menjadi salah satu aset terpenting yang harus dijaga. Kebocoran data tidak hanya berdampak pada kerugian ekonomi, tetapi juga dapat mengganggu kepercayaan publik terhadap organisasi, serta berimplikasi hukum karena adanya regulasi perlindungan data seperti GDPR dan UU Perlindungan Data Pribadi (PDP) di Indonesia.

Oleh karena itu, pencegahan Sensitive Data Exposure harus dilakukan dengan pendekatan menyeluruh, termasuk penerapan enkripsi kuat (AES, RSA), penggunaan protokol komunikasi aman (HTTPS/TLS 1.3), manajemen kunci yang tepat, serta pengujian keamanan aplikasi secara rutin. Dengan demikian, organisasi dapat meminimalkan risiko kebocoran data sensitif sekaligus menjaga keberlangsungan operasional dan reputasi.

Penetration Testing

Penetration testing merupakan metode sistematis untuk menguji kelemahan aplikasi dengan simulasi serangan nyata (Allen, 2001). McGraw (2006) menekankan bahwa praktik ini perlu menjadi bagian dari Secure Software Development Lifecycle (SSDLC). SANS Institute (2020) bahkan menekankan pentingnya pengujian rutin terhadap kontrol keamanan. Deteksi dini melalui Intrusion Detection and Prevention Systems (IDPS) juga penting, karena dapat mencegah eksploitasi aktif (Scarfone & Mell, 2007; Kizza, 2017).

Dalam praktiknya, penetration testing tidak hanya berfokus pada aspek teknis serangan, tetapi juga pada evaluasi menyeluruh terhadap kebijakan keamanan, prosedur operasional, serta kesiapan organisasi dalam merespons insiden. Pengujian ini biasanya dilakukan dengan pendekatan black-box, white-box, maupun gray-box, yang masing-masing memberikan perspektif berbeda terkait potensi kerentanan yang ada pada sistem (Sharma & Kalra, 2018).

Selain itu, deteksi dini melalui implementasi Intrusion Detection and Prevention Systems (IDPS) menjadi pelengkap penting dalam strategi keamanan. Scarfone dan Mell (2007) menekankan bahwa IDPS mampu mendeteksi aktivitas berbahaya secara real-time, sementara Kizza (2017) menambahkan bahwa pencegahan eksploitasi aktif dapat dicapai bila organisasi mengintegrasikan hasil penetration testing dengan sistem monitoring yang berkelanjutan. Dengan kata lain, pengujian penetrasi yang terencana dan berkesinambungan bukan hanya berfungsi sebagai alat untuk menemukan celah, tetapi juga sebagai strategi proaktif dalam meningkatkan ketahanan sistem informasi.

Di era serangan siber yang semakin canggih, penetration testing bukan lagi opsi, melainkan kebutuhan esensial bagi organisasi. Dengan menerapkan pengujian berkala, organisasi dapat memperkuat postur keamanannya, mematuhi standar regulasi internasional, serta menjaga kepercayaan para pemangku kepentingan.

METODE

Penelitian ini menggunakan pendekatan eksperimen dengan metode penetration testing. Pendekatan ini dipilih karena mampu memberikan gambaran nyata mengenai kelemahan keamanan aplikasi melalui simulasi serangan yang terkontrol. Menurut OWASP (2021), penetration testing merupakan salah satu metode penting untuk mengidentifikasi kerentanan sebelum dapat dieksploitasi oleh penyerang sebenarnya. Shostack (2014) juga menekankan bahwa pengujian keamanan yang sistematis tidak hanya menemukan kelemahan, tetapi juga membantu organisasi memahami dampak risiko secara lebih komprehensif.

Gambar 1. Tahap Penelitian

a. Perencanaan dan Identifikasi Target

Aplikasi yang menjadi target pengujian adalah invoice.masdzikry.com. Pada tahap ini dilakukan identifikasi ruang lingkup pengujian, termasuk menentukan batasan uji agar proses tidak menimbulkan kerusakan pada sistem. Tahap perencanaan penting untuk memastikan bahwa pengujian berfokus pada aspek yang relevan dengan tujuan penelitian, serta sesuai dengan prinsip responsible disclosure (Allen, 2001).

b. Enumerasi dan Analisis

Proses enumerasi dilakukan untuk mengidentifikasi aset, endpoint, dan parameter penting dalam aplikasi. Dengan memanfaatkan traffic interception menggunakan Burp Suite, peneliti menganalisis alur request dan response HTTP/HTTPS. Analisis ini bertujuan mendeteksi potensi kerentanan, termasuk parameter manipulasi, endpoint sensitif, serta kemungkinan bypass otorisasi. Menurut Fernandes et al. (2016), tahap enumerasi merupakan kunci dalam menemukan celah yang sering tersembunyi dalam logika bisnis aplikasi.

c. Eksploitasi Kerentanan

Tahap eksploitasi dilakukan untuk membuktikan secara praktis bahwa kerentanan yang ditemukan dapat disalahgunakan. Dua skenario utama diuji, yaitu:

1. Insecure Direct Object Reference (IDOR): penyerang mencoba mengubah parameter id pada endpoint /invoices/{id} untuk mengakses data faktur milik pengguna lain.
2. Broken Access Control: dengan menyuntikkan header X-Original-Url, penyerang berusaha memanipulasi jalur request sehingga dapat mengakses endpoint sensitif /admin yang seharusnya terbatas pada peran administrator.

Eksploitasi ini mengikuti pedoman OWASP Testing Guide (2021), yang menekankan pentingnya menguji kelemahan berbasis otorisasi sebagai bagian dari access control testing.

d. Dokumentasi dan Analisis Dampak

Seluruh hasil pengujian didokumentasikan secara rinci, termasuk bukti eksploitasi, data yang terekspos, serta analisis risiko yang ditimbulkan. Dokumentasi ini penting untuk memberikan rekomendasi perbaikan yang tepat sasaran. Peltier (2016) menegaskan bahwa dokumentasi hasil pengujian harus bersifat objektif dan dapat digunakan sebagai dasar pengambilan keputusan manajemen keamanan informasi.

Dengan mengikuti tahapan tersebut, penelitian ini tidak hanya mengidentifikasi adanya kelemahan pada aplikasi, tetapi juga menganalisis dampaknya terhadap kerahasiaan, integritas, dan ketersediaan data. Pendekatan ini diharapkan mampu memberikan kontribusi praktis bagi pengembang aplikasi dalam meningkatkan keamanan sistem serupa.

HASIL

Kerentanan IDOR

Pengujian terhadap endpoint /invoices/{id} menunjukkan adanya kelemahan berupa Insecure Direct Object Reference (IDOR). Dalam kasus ini, pengguna dengan hak akses biasa dapat memperoleh data faktur milik pengguna lain hanya dengan melakukan modifikasi pada parameter id. Kondisi tersebut mengindikasikan bahwa sistem tidak melakukan validasi otorisasi tambahan sebelum menampilkan data yang diminta.

Eksploitasi kerentanan ini berdampak serius pada aspek kerahasiaan (confidentiality), karena data keuangan antar pengguna terekspos kepada pihak yang tidak berwenang. Misalnya, faktur milik User A dapat diakses sepenuhnya oleh User B tanpa adanya pembatasan yang semestinya. Temuan ini sejalan dengan konsep yang dikemukakan oleh Fogie (2010) dan Viega & McGraw (2001), bahwa kepercayaan berlebihan terhadap input pengguna sering kali menjadi akar penyebab IDOR pada aplikasi berbasis web.

Broken Access Control + Sensitive Data Exposure

Uji penetrasi selanjutnya mengidentifikasi kelemahan pada endpoint /admin, di mana pengguna biasa dapat menyuntikkan header X-Original-Url untuk memanipulasi jalur permintaan. Teknik ini memungkinkan akses ke halaman admin yang semestinya hanya dapat diakses oleh pengguna dengan hak istimewa.

Kerentanan ini tidak hanya menandakan adanya broken access control, tetapi juga berpotensi menyebabkan sensitive data exposure. Data sensitif berupa informasi administrator (misalnya alamat email dan peran pengguna) dapat diakses secara tidak sah. Lebih jauh lagi, kondisi ini membuka peluang bagi penyerang untuk melakukan eskalasi hak akses hingga menyerupai peran administratif.

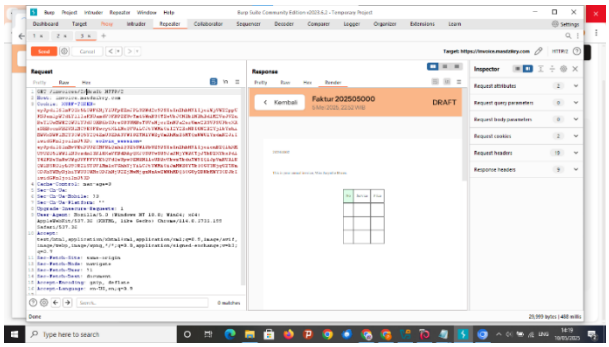
Temuan ini konsisten dengan pernyataan OWASP (2021) yang menempatkan broken access control sebagai salah satu kerentanan paling kritis dalam aplikasi web modern. Dampak kebocoran data sensitif juga sejalan dengan analisis Anderson (2020), yang menegaskan bahwa insiden semacam ini berpotensi menghancurkan reputasi organisasi sekaligus menimbulkan kerugian finansial signifikan

PEMBAHASAN

Temuan kerentanan menunjukkan bahwa aplikasi belum memiliki mekanisme otorisasi yang ketat pada lapisan server.

Analisis IDOR

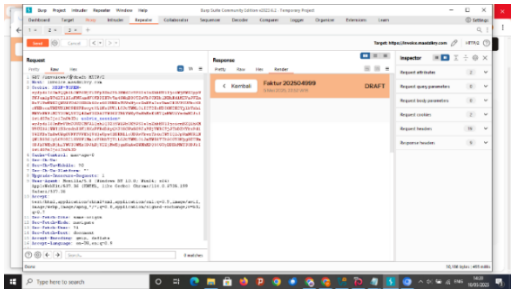
Hasil pengujian pada endpoint /invoices/{id} menunjukkan adanya kelemahan berupa Insecure Direct Object Reference (IDOR). Pada kerentanan ini, pengguna dengan hak akses biasa dapat mengakses faktur milik pengguna lain hanya dengan melakukan modifikasi terhadap parameter id pada URL. Eksploitasi ini berhasil dibuktikan melalui penggunaan Burp Suite, di mana manipulasi nilai parameter menghasilkan tampilan faktur yang seharusnya tidak dapat diakses



Gambar 2. Invoice

Pada Gambar 2 ditunjukkan hasil pengujian menggunakan Burp Suite, di mana aplikasi menampilkan invoice dengan parameter id=2. Meskipun pengujian dilakukan oleh pengguna biasa, sistem tetap memberikan respon berupa data faktur dengan nomor 2025050000. Hal ini mengindikasikan bahwa mekanisme otorisasi tidak diterapkan secara ketat pada sisi server, sehingga setiap pengguna dapat memodifikasi nilai parameter id untuk mengakses data milik pengguna lain.

Secara prinsip, data faktur hanya boleh diakses oleh pemilik akun yang sah. Namun, kelemahan ini memperlihatkan adanya kerentanan Insecure Direct Object Reference (IDOR), di mana kontrol akses pada objek tidak diverifikasi dengan benar. Kondisi ini sesuai dengan temuan OWASP (2021) yang menyebutkan bahwa IDOR merupakan salah satu bentuk Broken Object Level Authorization yang paling umum dijumpai pada aplikasi web modern. Dampaknya dapat berupa kebocoran informasi sensitif, khususnya data finansial antar pengguna, yang berimplikasi langsung terhadap aspek kerahasiaan (confidentiality) dan kepercayaan pengguna terhadap aplikasi.



Gambar 3. Invoice 3

Gambar 3 memperlihatkan hasil pengujian lanjutan menggunakan Burp Suite, di mana parameter id pada endpoint /invoices/{id} diubah dari id=2 menjadi id=3. Perubahan sederhana pada parameter ini tetap menghasilkan respon dari server berupa data faktur dengan nomor 202504999, meskipun pengguna yang melakukan permintaan hanya memiliki hak akses sebagai user biasa.

Temuan ini menegaskan bahwa aplikasi tidak memiliki mekanisme otorisasi tambahan di sisi server untuk memvalidasi apakah permintaan data sesuai dengan identitas pengguna yang sedang login. Kondisi ini merupakan karakteristik utama kerentanan Insecure Direct Object Reference (IDOR).

Dari sisi keamanan, kerentanan ini dikategorikan berisiko tinggi (High Severity) dengan skor CVSS 7.5, karena memungkinkan kebocoran data finansial antar pengguna tanpa hambatan. Sesuai dengan pedoman OWASP (2021), kelemahan otorisasi pada level objek (Broken Object Level Authorization) merupakan salah satu kerentanan paling kritis yang sering dieksploitasi oleh penyerang. Apabila tidak segera diperbaiki, kerentanan ini dapat menimbulkan konsekuensi serius, baik dari sisi kerahasiaan data (confidentiality) maupun kepercayaan pengguna terhadap sistem.

Tabel 1. Findings Report

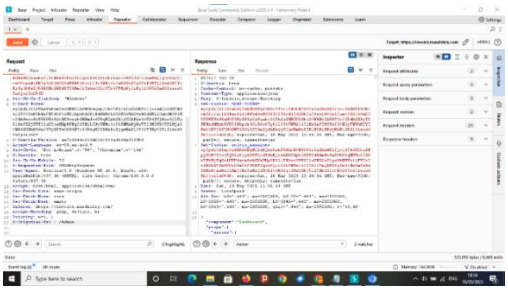
Finding	IDOR
Risk Rating	High (7.5)
CVSS	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N
Description	IDOR terjadi ketika aplikasi memberikan akses langsung ke objek (seperti file, entri database, dsb.) berdasarkan input pengguna (misalnya ID), tanpa melakukan otorisasi yang memadai. Contoh klasik adalah ketika kamu bisa mengakses faktur?id=1002 padahal seharusnya kamu hanya boleh mengakses faktur?id=1001.
Poc	Login User biasa lalu Membuat request idor di brupsuite berupa faktur dengan merubah input.

Melakukan request dengan merubah input invoices 2 ke 3, user biasa dapat mengakses yang buka hak akses. IDOR terjadi karena sistem hanya mengandalkan parameter id dari sisi klien tanpa adanya validasi apakah pengguna yang meminta memiliki hak akses terhadap objek tersebut. Hal ini berbahaya karena memungkinkan akses ilegal hanya dengan parameter tampering.

Temuan IDOR pada aplikasi menunjukkan lemahnya validasi otorisasi. Jika seorang pengguna dapat melihat faktur milik orang lain, maka privasi data tidak lagi terjamin. Masalah ini bukan sekadar isu teknis, melainkan pelanggaran langsung terhadap prinsip kerahasiaan informasi (Whitman & Mattord, 2017). Lebih jauh, Scarfone dan Mell (2007) menekankan bahwa sistem harus mampu mendeteksi permintaan anomali, sehingga aktivitas semacam ini segera dihentikan.

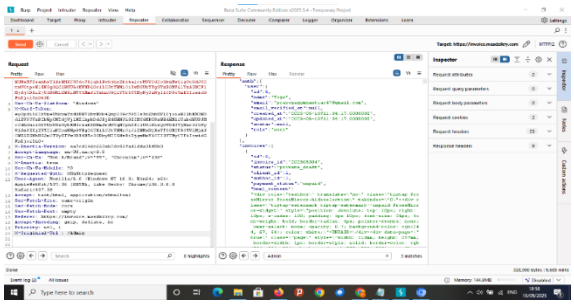
Analisis Broken Access Control

Selain IDOR, ditemukan juga kerentanan Broken Access Control yang dikombinasikan dengan Sensitive Data Exposure. Celah ini dieksploitasi dengan memanfaatkan header X-Original-Url, yang memungkinkan user biasa mengakses endpoint /admin. Berikut hasil uji yang berhasil didokumentasikan.



Gambar 2. Broken Access Control user

Gambar 4 Broken Access Control, Menunjukkan proses manipulasi header X-Original-Url menggunakan Burp Suite. User biasa menambahkan header palsu untuk mengelabui sistem sehingga request diarahkan ke endpoint /admin.

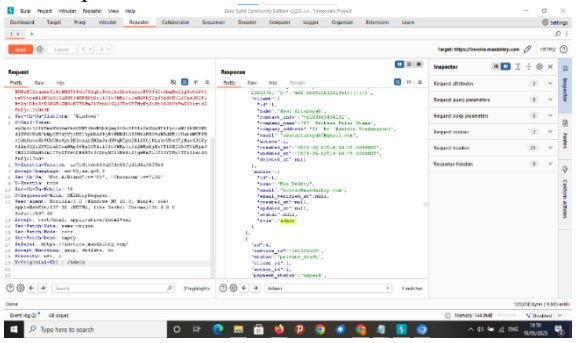


Gambar 3. Broken Access Control Data Respon User

Gambar 5 memperlihatkan hasil uji kerentanan Broken Access Control menggunakan Burp Suite. Pada pengujian ini, pengguna biasa menyisipkan header tambahan X-Original-Url: /admin ke dalam permintaan HTTP. Manipulasi tersebut berhasil mengelabui sistem sehingga permintaan diarahkan ke endpoint administratif /admin, yang seharusnya hanya dapat diakses oleh akun dengan hak istimewa.

Respon dari server menunjukkan keluaran berupa data sensitif, antara lain informasi akun administrator (alamat email, role, dan metadata lainnya), serta rincian faktur yang tidak seharusnya dapat diakses oleh pengguna dengan peran biasa. Hal ini membuktikan bahwa aplikasi belum menerapkan mekanisme validasi otorisasi yang memadai di sisi server untuk memastikan kesesuaian antara hak akses pengguna dengan sumber daya yang diminta.

Kerentanan ini berimplikasi ganda, yaitu mengindikasikan broken access control sekaligus menyebabkan sensitive data exposure. Berdasarkan kategori OWASP (2021), kelemahan seperti ini termasuk dalam salah satu risiko tertinggi keamanan aplikasi web modern. Selain mengancam kerahasiaan (confidentiality), kondisi ini juga membuka peluang terjadinya eskalasi hak akses (privilege escalation), yang dapat digunakan penyerang untuk mengambil alih kendali sistem secara penuh.



Gambar 4. Broken Access Control

Gambar 6 Broken Access Control, memperlihatkan detail informasi sensitif milik admin yang berhasil diakses oleh user biasa akibat kelemahan Broken Access Control. Temuan ini dikategorikan sebagai risiko kritis (Critical, CVSS 9.1) karena berpotensi membuka jalan bagi privilege escalation dan pengambilalihan akun admin. Setelah request dikirim, server merespons dengan menampilkan data sensitif milik admin, termasuk alamat email dan role pengguna. Informasi ini seharusnya hanya dapat diakses oleh akun dengan hak akses administrator.

Tabel 2. Findings Report

Finding	Broken Access Control + Sensitive Data Exposure
Risk Rating	Critical (9.1)
CVSS	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:N
Description	User biasa dapat memalsukan header X-Original-Url untuk mengakses endpoint admin dan mendapatkan informasi sensitif milik admin (email, role, dsb) tanpa otorisasi.
Poc	Login User biasa lalu Membuat request Broken Acces Control di brupsuite berupa faktur dengan merubah input.

Request dari burp suit untuk mengelabuhi header x-original berhasil, sehingga user biasa dapat mengetahui akses admin dan masuk sebagai admin. Manipulasi header X-Original-Url menunjukkan bahwa aplikasi masih memproses header dari pengguna tanpa validasi. Ini membuka peluang privilege escalation di mana user biasa dapat berperan sebagai admin.

Eksploitasi melalui X-Original-Url menunjukkan bagaimana kelemahan kontrol akses dapat mengarah pada privilege escalation. Rouse (2019) menjelaskan bahwa celah seperti ini sering kali dimanfaatkan untuk memperoleh kendali penuh atas sistem. Dampak dari akses ilegal tidak hanya terbatas pada kebocoran data, tetapi juga potensi manipulasi data transaksi, yang dapat mengacaukan integritas sistem (Anderson, 2020).

Fakta bahwa data sensitif terekspos menegaskan perlunya penerapan enkripsi dan kebijakan penyimpanan data yang lebih ketat (Stallings, 2019). Penerapan role-based access control yang tepat, dikombinasikan dengan audit log yang memadai, dapat menjadi langkah mitigasi efektif (Peltier, 2016; SANS Institute, 2020).

Selain itu, penelitian ini menegaskan bahwa keamanan aplikasi tidak dapat dianggap sebagai tambahan belaka, melainkan bagian integral dari siklus pengembangan perangkat lunak (McGraw, 2006). Viega dan McGraw (2001) menambahkan bahwa membangun perangkat lunak aman harus dimulai sejak tahap desain, bukan setelah produk dirilis.

Implikasi Keamanan

Kombinasi antara kerentanan Insecure Direct Object References (IDOR) dan Broken Access Control dengan paparan data sensitif (Sensitive Data Exposure) menimbulkan konsekuensi serius terhadap aspek kerahasiaan (confidentiality), integritas (integrity), dan ketersediaan (availability) sistem.

a. Celah IDOR memungkinkan penyerang memperoleh akses tidak sah terhadap data keuangan milik pengguna lain hanya dengan melakukan manipulasi parameter. Kondisi ini menimbulkan ancaman kebocoran data pribadi maupun finansial yang bersifat rahasia.
b. Kelemahan Broken Access Control yang dikombinasikan dengan paparan data sensitif memberikan peluang bagi penyerang untuk mengakses informasi administratif, seperti akun, alamat email, serta peran (role) pengguna dengan hak istimewa. Akses tersebut membuka kemungkinan terjadinya eskalasi hak akses (privilege escalation), sehingga akun administrator dapat diambil alih oleh pihak yang tidak berwenang.
c. Dengan menguasai data keuangan pengguna dan informasi administratif, penyerang berpotensi menyalahgunakan sistem untuk aktivitas berbahaya, seperti penipuan transaksi, rekayasa sosial (social engineering), atau pencurian identitas (identity theft). Risiko ini tidak hanya mengancam keamanan data, tetapi juga berdampak langsung pada kepercayaan pengguna terhadap sistem, serta menimbulkan potensi kerugian finansial maupun reputasi bagi penyedia layanan.

Dengan demikian, kedua kerentanan ini secara simultan meningkatkan tingkat risiko keamanan aplikasi ke kategori tinggi (High Risk), sesuai dengan klasifikasi OWASP Top 10 (2021).

Kombinasi kedua kerentanan ini sangat berisiko, karena penyerang dapat:

Rekomendasi Mitigasi

Untuk meminimalkan risiko keamanan yang ditimbulkan oleh temuan kerentanan, diperlukan penerapan mekanisme pengendalian keamanan yang sesuai dengan standar praktik terbaik (best practices) dalam pengembangan aplikasi web. Beberapa langkah mitigasi yang direkomendasikan antara lain:

a. Penerapan Role-Based Access Control (RBAC)

Aplikasi perlu mengimplementasikan sistem otorisasi berbasis peran yang ketat, sehingga setiap permintaan akses ke sumber daya akan divalidasi berdasarkan hak akses pengguna. Pendekatan ini mencegah pengguna dengan hak terbatas mengakses data atau fungsi yang seharusnya hanya tersedia bagi administrator.

b. Validasi Parameter di Sisi Server

Seluruh parameter, khususnya parameter identitas seperti id, harus divalidasi secara konsisten di sisi server. Validasi ini memastikan bahwa data yang diminta benar-benar milik pengguna yang melakukan permintaan, sehingga mencegah eksploitasi celah Insecure Direct Object References (IDOR).

c. Pembatasan Header Manipulatif

Header yang dapat dimanipulasi, seperti X-Original-Url, harus dinonaktifkan atau divalidasi secara ketat. Hal ini untuk mencegah penyalahgunaan header dalam mengakses endpoint sensitif seperti /admin.

d. Logging dan Monitoring Proaktif

Sistem harus dilengkapi dengan mekanisme pencatatan (logging) dan pemantauan (monitoring) yang mampu mendeteksi pola akses anomali, misalnya percobaan akses berulang ke sumber daya yang tidak sah. Dengan pendekatan ini, potensi serangan dapat diidentifikasi dan ditangani lebih cepat.

e. Audit Keamanan Berkala

Audit keamanan aplikasi secara rutin, berdasarkan standar OWASP Application Security Verification Standard (ASVS), penting dilakukan untuk memastikan kepatuhan terhadap prinsip keamanan aplikasi modern. Audit ini

juga berfungsi sebagai mekanisme pencegahan jangka panjang terhadap kerentanan baru yang mungkin muncul akibat pembaruan sistem.

KESIMPULAN

Berdasarkan hasil penelitian yang dilakukan pada aplikasi invoice.masdzikry.com, ditemukan dua kerentanan utama, yaitu Insecure Direct Object Reference (IDOR) pada parameter id yang memungkinkan pengguna biasa mengakses data faktur milik pengguna lain, serta Broken Access Control yang dieksploitasi melalui manipulasi header X-Original-Url sehingga memungkinkan akses data sensitif milik admin dan potensi eskalasi hak akses. Kombinasi kedua kerentanan ini dikategorikan sebagai risiko tinggi karena dapat menyebabkan kebocoran data keuangan, penyalahgunaan hak akses, hingga pencurian identitas dan pengambilalihan akun pengguna. Oleh karena itu, disarankan agar pengembang aplikasi segera menerapkan kontrol otorisasi berbasis peran, melakukan validasi parameter di sisi server, menonaktifkan header yang berbahaya, serta melaksanakan audit keamanan secara berkala sesuai standar OWASP, guna meningkatkan perlindungan aplikasi dari ancaman eksploitasi serupa di masa mendatang.

REFERENSI

Allen, J. H. (2001). The CERT guide to system and network security practices. Addison-Wesley Professional.

Anderson, R. (2020). Security engineering: A guide to building dependable distributed systems (3rd ed.). Wiley.

Fernandes, E., Jung, J., & Prakash, A. (2016). Security analysis of emerging smart home applications. Proceedings of the IEEE Symposium on Security and Privacy (SP), 636–654. https://doi.org/10.1109/SP.2016.44

Fogie, S., Grossman, J., Hansen, R., Rager, A., & Petkov, P. (2010). XSS attacks: Cross site scripting exploits and defense. Syngress.

Laudon, K. C., & Laudon, J. P. (2020). *Management information systems: Managing the digital firm* (16th ed.). Pearson Education.

Halfond, W. G. J., Viegas, J., & Orso, A. (2006). A classification of SQL-injection attacks and countermeasures. Proceedings of the IEEE International Symposium on Secure Software Engineering, 13–15.

McGraw, G. (2006). Software security: Building security in. Addison-Wesley Professional.

OWASP. (2021). *OWASP Testing Guide v4*. The Open Web Application Security Project. https://owasp.org/www-project-web-security-testing-guide/

Peltier, T. R. (2016). Information security policies, procedures, and standards: guidelines for effective information security management (2nd ed.). Auerbach Publications.

Rouse, M. (2019). Role-based access control (RBAC). TechTarget. https://www.techtarget.com/searchsecurity/definition/role-based-access-control-RBAC

SANS Institute. (2020). Penetration testing: An essential component of cybersecurity. SANS Institute. https://www.sans.org

Stallings, W. (2019). Computer security: Principles and practice (4th ed.). Pearson Education.

Verizon. (2021). 2021 Data Breach Investigations Report (DBIR). Verizon Enterprise Solutions. https://www.verizon.com/business/resources/reports/dbir/

Viega, J., & McGraw, G. (2001). Building secure software: How to avoid security problems the right way. Addison-Wesley Professional.

Whitman, M. E., & Mattord, H. J. (2017). Principles of information security (6th ed.). Cengage Learning.