Insecure Direct Object Reference (IDOR) in Grubhub Order APIs Enables Cross-User Order Disclosure

Reported by: Jalen Francis and Jayson Clark

September 26, 2025

Executive Summary

We identified an Insecure Direct Object Reference (IDOR) in Grubhub's order APIs that allows any normal user account to retrieve *other users'* order details by enumerating order identifiers. This enables access to PII (names, phone numbers, addresses) and complete order information, including items and pickup details. Attackers can impersonate customers at pickup locations or harvest PII at scale. This is a high-severity vulnerability requiring immediate attention.

Impact

- **Real-world attacks**: Any Grubhub user can access other users' orders, steal food at pickup locations, or harvest PII
- **Data exposure**: Names, phone numbers, addresses, order contents, payment details
- **Regulatory risk**: PII exposure may trigger breach notifications and fines (CCPA/CPRA)

Data Elements Exposed

Based on endpoint analysis, the following PII and sensitive data are accessible through the IDOR:

- **Diner PII**: Full name, email address, phone number
- **Order details**: Complete itemization with prices, quantities, special instructions
- **Pickup information**: Store name, phone, email, pickup instructions, order instructions
- **Payment data**: Payment methods, amounts, transaction details
- **Restaurant information**: Store IDs, names, contact details, media assets
- **Order metadata**: Order timing, fulfillment type, order status, tracking information

Severity

- **Category**: IDOR / Broken Object-Level Authorization (BOLA)
- **OWASP API Top 10**: API1:2023 Broken Object Level Authorization
- **CVSS v3.1 (est.)**: AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:N/A:N → **8.2 High**

Affected Components (observed)

Based on observed behavior:

- **Order history/detail endpoint**: Returns order data given an order identifier associated to any diner, while only requiring a valid but unrelated diner bearer token.
- **Shop-line (order status/line items) endpoint**: Returns itemization and status given an order identifier without enforcing ownership.

Attack Vector

Any normal Grubhub user can:

- Sign up for a free account and get valid tokens
- Use their tokens to access any other user's orders by guessing order IDs
- Maintain long-lived sessions for sustained attacks

Vulnerability Details

Summary. With a normal authenticated diner bearer token, requests to order-related endpoints accept a numeric order identifier (observed as a string with a leading hyphen) and return full order details even when the order belongs to a different diner. The API does not strictly verify object ownership (authorization) for the provided order identifier. Order identifiers are sufficiently predictable to permit enumeration.

Technical Indicators. In testing, the following patterns were observed:

- Order identifiers appear numeric and monotonically increasing over time (predictable).
- The API returns 200 OK and full order bodies for identifiers that belong to other diners.
- Rate limiting is present but permissive enough to allow practical enumeration.
- Diner identifiers are simple and readily discoverable (e.g., returned in login responses), which facilitates cross-user targeting.

Proof of Concept

1. Create a normal Grubhub account and login to get tokens
2. Use your tokens to request any order ID (e.g., `-557539591`)
3. API returns full order details including PII, even though it's not your order
4. Order IDs are predictable, enabling systematic enumeration

Responsible Testing Guidance

- Use only vendor-owned test accounts and identifiers in a non-production or sanctioned test environment.
- Do not access real diners' orders or PII; avoid using live order identifiers in demonstrations.
- Obtain explicit written authorization for any automated testing and stay within program scope and rate limits.
- Capture timestamps and request IDs to aid vendor validation and triage.

Example Endpoints (illustrative placeholders). Use vendor-internal documentation to find the exact paths corresponding to:

- GET `/tapingo/diners/{DINER_UUID}/order-history/{ORDER_ID}` (order detail view)
- GET `/api/order/{ORDER_ID}/shop-line` (line-items/status)

Both appear to authorize the request based on possession of *any* diner token rather than ownership of $\{ORDER\_ID\}$.

Root Cause Hypothesis

- **Missing ownership check**: The service returns objects by identifier without verifying that the authenticated principal owns the object.
- **Predictable identifiers**: Sequential or guessable identifiers enable practical enumeration.
- **Insufficient rate limiting/anomaly detection**: Allows sustained probing across large identifier ranges.

Recommended Fixes

- **Verify ownership**: Check that the authenticated user owns the order before returning data
- **Use random order IDs**: Replace predictable numeric IDs with unguessable identifiers (UUIDs)

Vendor Validation Checklist

- Verify that requests for orders not owned by the authenticated diner return `404` or `403`.
- Verify that identifiers cannot be guessed to enumerate other diners' orders.
- Verify rate limits and anomaly detection trigger on sequential identifier access attempts.
- Confirm that shop-line and any *derived* endpoints perform the same ownership enforcement.
- Ensure login/auth responses minimize client-exposed claims; avoid returning identifiers not strictly necessary.
- Scope tokens to the diner and device; prefer short-lived access tokens, refresh rotation, and proof-of-possession.

- Confirm that refresh tokens cannot be used to access or enumerate cross-owner objects without ownership checks.

Timeline

- **Discovery**: 2025-09-21
- **Initial Report**: September 26, 2025
- **Vendor Acknowledgment**: [to be filled by vendor]
- **Mitigation/Remediation**: [to be filled by vendor]

Credits

Discovered and responsibly reported by **Jalen Francis** and **Jayson Clark**, Computer Science students at The Ohio State University.

Contact

Please contact us at: jalenfran@proton.me. We are college students interested in security and would be happy to assist with validation.

Responsible Disclosure

We did not access or retain any third-party user data beyond our own test accounts, and we are providing this report under responsible disclosure. We will not publish details prior to remediation and vendor approval.

Appendix: Additional Technical Observations

During testing, we observed that:

- A normal diner bearer token successfully retrieved orders for identifiers not owned by the token holder.
- Order identifiers appear predictable across time, facilitating enumeration.
- Rate limiting allowed sustained requests without immediate blocking, indicating practical exploitability.
- The login endpoint returns comprehensive session data including long-lived refresh tokens.

No exploit code is included. Vendor can validate internally using their test environments.