Insecure Deserialization

Blocked RCE DoS

Tools Used

- **Web Browser**
- **GoBuster:**
- **Burp Suite**

Discovering the API Documentation

Using GoBuster, we discovered the Swagger API documentation for Juice Shop at `http://localhost:3000/api-docs`.

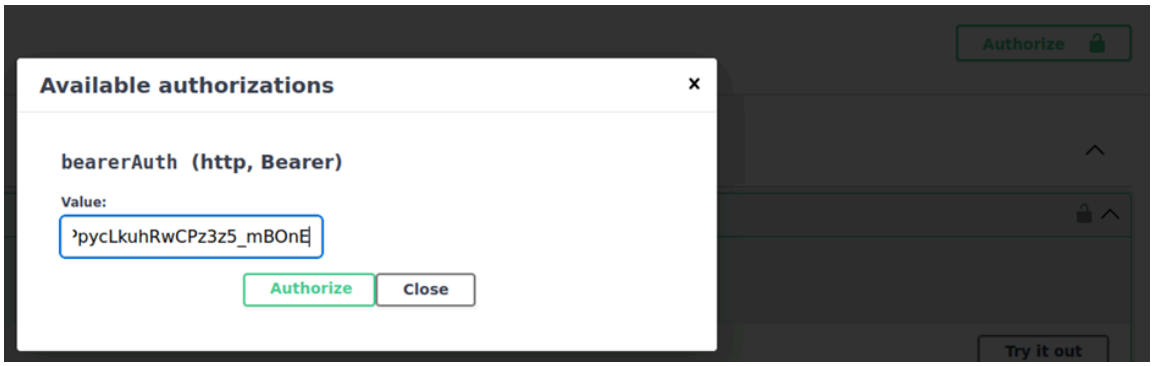
Identifying the Vulnerable Endpoint

Within the Swagger UI, we identified the `/orders` endpoint, which allows adding new orders. This endpoint accepts JSON data, including an attribute called `orderLinesData` that can contain arbitrary JSON strings.

Authorizing API Requests

To make example API requests (useful for next step)., we needed to include a bearer token for authorization:

1. **Retrieve Bearer Token:** Extract the token from the browser's developer tools.
2. **Authorize Request:** Input the token in the Swagger UI to authorize examples requests

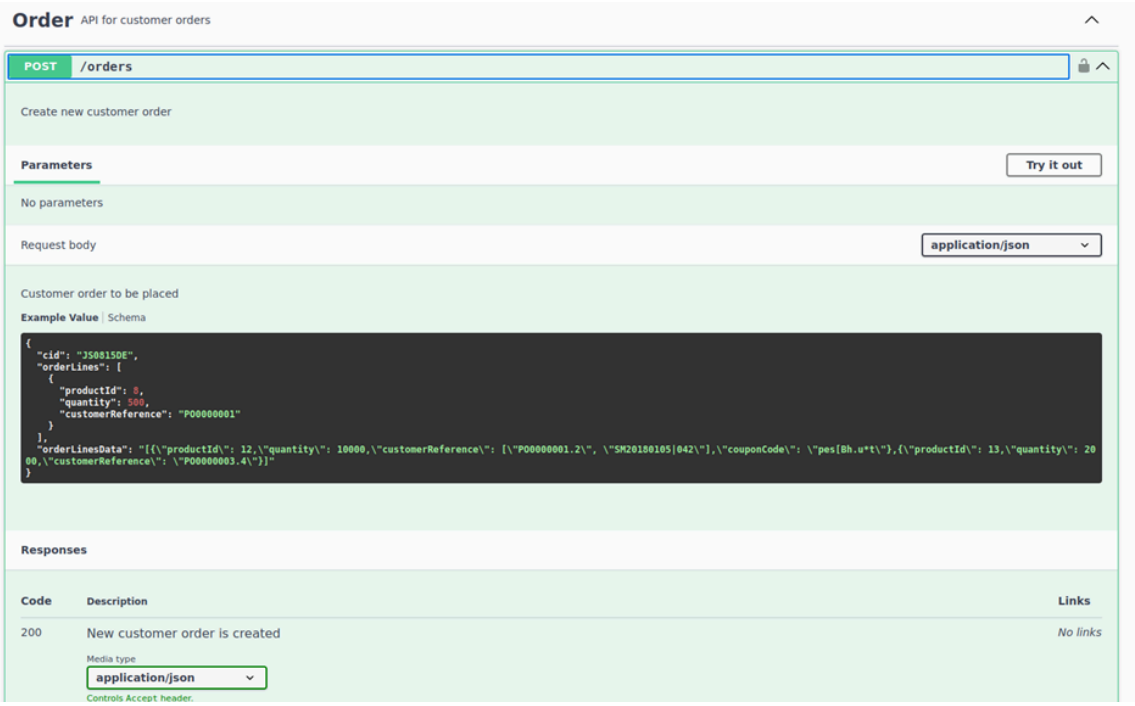


Testing the API Endpoint

1. **Try It Out:** Click "Try it out" in the Swagger UI for the `/orders` endpoint and observe the example payload structure:

```
{
  "cid": "JS0815DE",
  "orderLines": [
    {
      "productId": 8,
      "quantity": 500,
      "customerReference": "PO0000001"
    }
  ],
  "orderLinesData": "[{\"productId\": 12,\"quantity\":
10000,\"customerReference\": [\"PO0000001.2\",
\"SM20180105|042\"],\"couponCode\": \"pes[Bh.u*t\"},{\"productId\":
13,\"quantity\": 2000,\"customerReference\": \"PO0000003.4\"}]"
}
```

We can see the example page here :



Crafting the DoS Payload

To exploit the vulnerability, we crafted a payload designed to cause an infinite loop on the server:

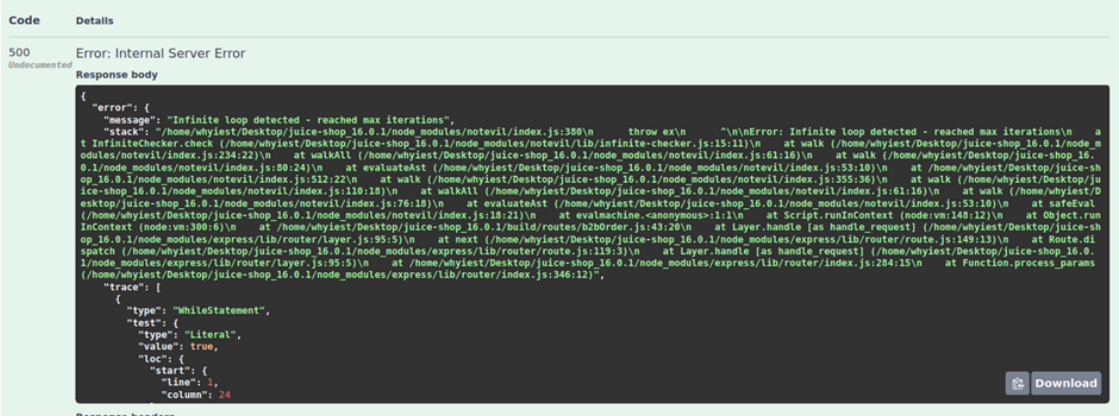
- **Payload:**
- ```
  {
    "orderLinesData": "(function dos() { while(true); })()"
  }
  ```

Injecting the Payload

1. **Replace Payload:** Modify the example payload in the Swagger UI to include our DoS payload.
2. **Send Request:** Execute the request by clicking "Execute".

Observing the Results

After injecting the payload, the server became unresponsive for about 2 seconds before recoverin and displaying an internal server error message. This indicates that the server detected and mitigated the DoS attack, but it's normal as Juice-Shop limite Dos attack to 2 seconds to avoid crashing our server.
  ```



Successful RCE DoS

Tools Used

- **Web Browser**
- **GoBuster**.

Discovering the Swagger API Documentation

Using GoBuster, we discovered the Swagger API documentation for Juice Shop at `http://localhost:3000/api-docs`.

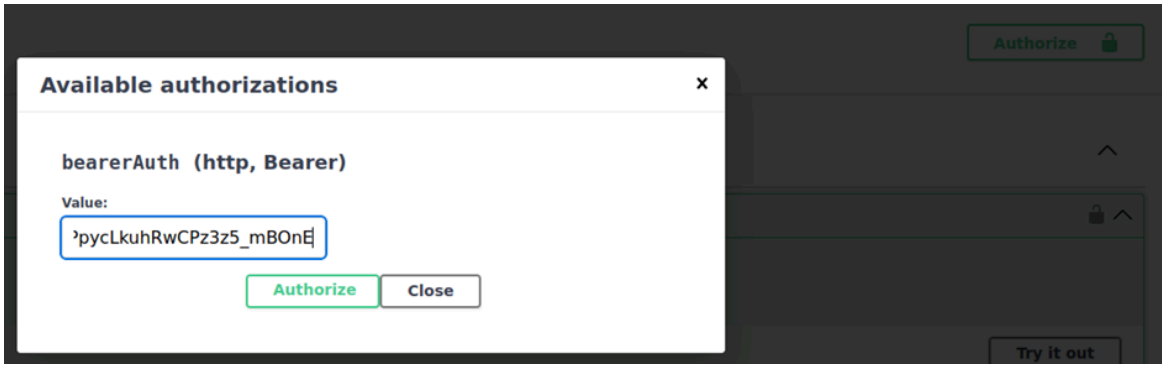
Identifying the Vulnerable Endpoint

Within the Swagger UI, we identified the `/orders` endpoint, which allows adding new orders. This endpoint accepts JSON data, including an attribute called `orderLinesData` that can contain arbitrary JSON.

Authorizing API Requests

To make API requests, we needed to include a bearer token for authorization:

1. **Retrieve Bearer Token:** Extract the token from the browser's developer tools.
2. **Authorize Request:** Input the token in the Swagger UI to authorize the request.



Testing the API Endpoint

1. **Try It Out:** Click "Try it out" in the Swagger UI for the `/orders` endpoint and observe the example payload structure:

```
{
  "cid": "JS0815DE",
  "orderLines": [
    {
      "productId": 8,
      "quantity": 500,
      "customerReference": "PO0000001"
    }
  ],
  "orderLinesData": "[{\"productId\": 12,\"quantity\":
10000,\"customerReference\": [\"PO0000001.2\",
\"SM20180105|042\"],\"couponCode\": \"pes[Bh.u*t\"},{\"productId\":
13,\"quantity\": 2000,\"customerReference\": \"PO0000003.4\"}]"
}
```

12. **Identify Vulnerability:** Notice that `orderLinesData` allows arbitrary JSON injection, making it a potential vector for injection attacks.

Crafting the DoS Payload

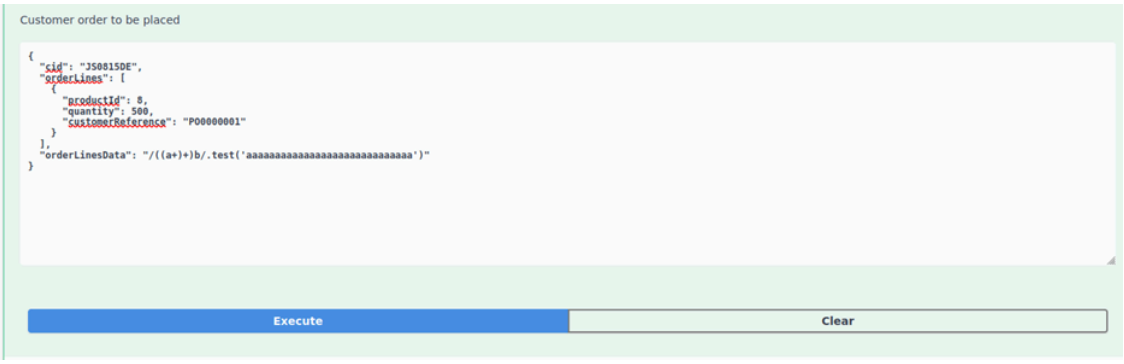
To exploit the vulnerability, we crafted a payload designed to cause a Regular Expression Denial of Service (ReDoS):

- **Payload:**
- {
- "orderLinesData": "/((a+)+)b/.test('aaaaaaaaaaaaaaaaaaaaaaaaaaaaa')"
}

This payload uses a complex regular expression that leads to excessive backtracking and processing time, causing the server to hang.

Executing the DoS Attack

1. **Input Payload:** Replace the example payload with our ReDoS payload in the Swagger UI.



2. **Send Request:** Execute the request and observe the server's response.

Observing the Result

As expected, the server responded with a 503 Service Unavailable error, indicating that the payload successfully caused a DoS condition.