

Project : Jeder Kann Kochen

Gruppe 18

Security – OWASP Top 10

Feature 1 : Login

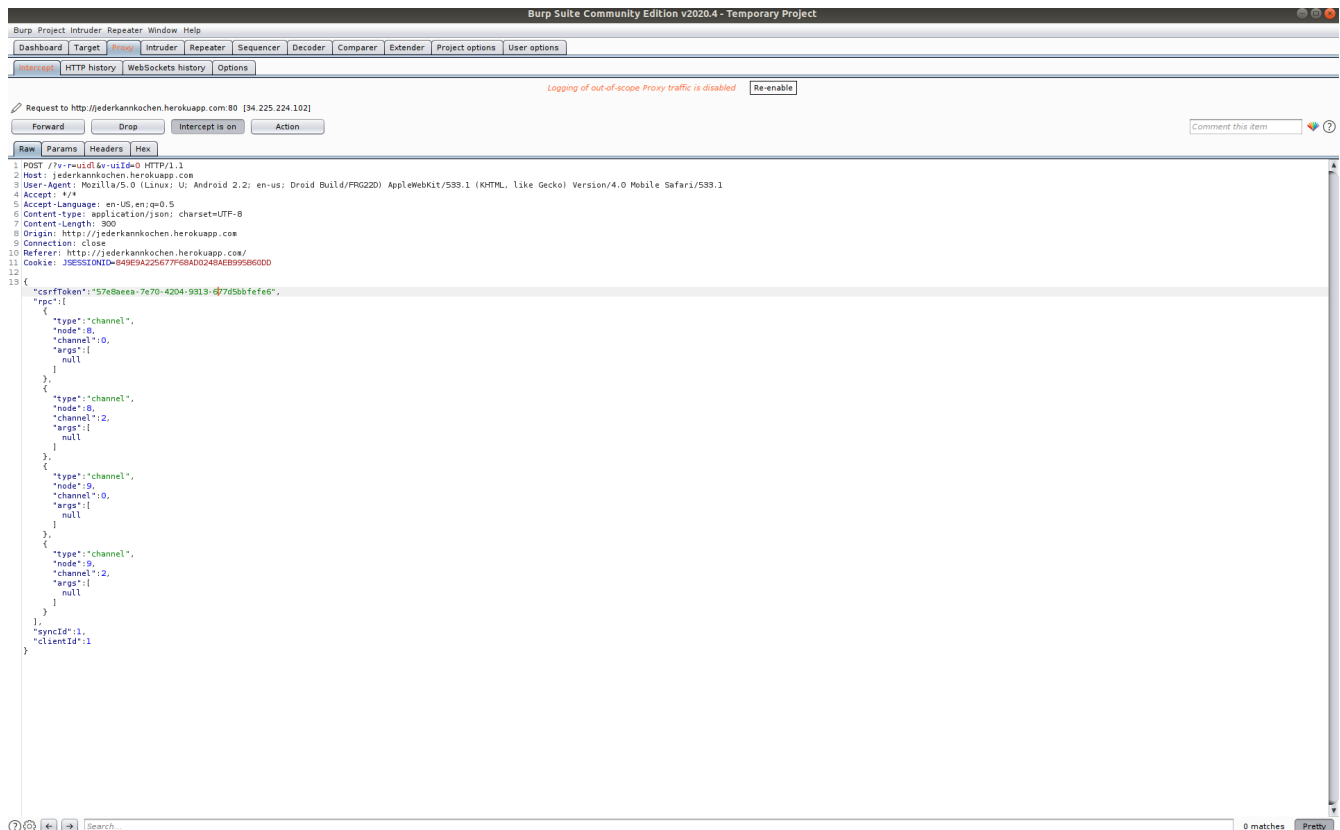
- **Cross-site Request Forgery in Login Form**

If there is a page that's different for every user (such as "edit my profile") and vulnerable to XSS (Cross-site Scripting) then normally it cannot be exploited. However if the login form is vulnerable, an attacker can prepare a special profile, force victim to login as that user which will trigger the XSS exploit. Again attacker is still quite limited with this XSS as there is no active session. However the attacker can leverage this XSS in many ways such as showing the same login form again but this time capturing and sending the entered username/password to the attacker.

Remediation:

You need to implement a token system in your code to prevent Login CSRF.

The important thing is to make sure the token is something the user has (but not the attacker), so that you can make sure it really is the user submitting a login request.



Using SQL Injection to Bypass Authentication :

To check for potential SQL injection vulnerabilities we have entered a single quote in to the "Username" field and submitted the request using the "Login" button.

Enter some appropriate syntax to modify the SQL query into the "Name" input.

In this example we used ' or 1=1 -- .

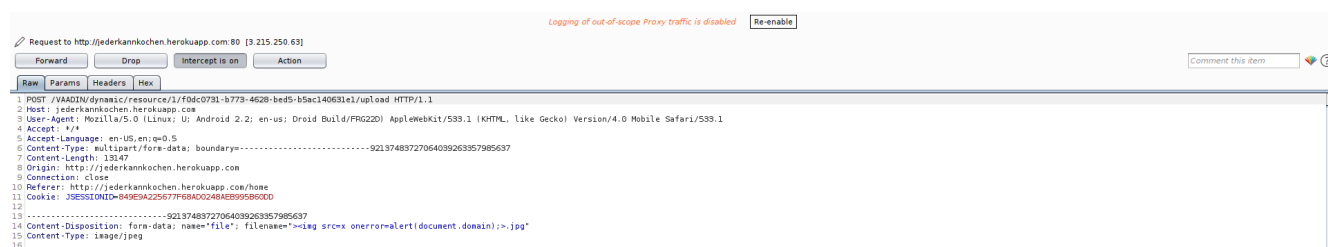
This causes the application to perform the query:

```
SELECT * FROM users WHERE username = " OR 1=1-- ' AND password = 'foo'
```

Because the comment sequence (--) causes the remainder of the query to be ignored, this is equivalent to:

```
SELECT * FROM users WHERE username = ' ' OR 1=1
```

Unrestricted File Upload :



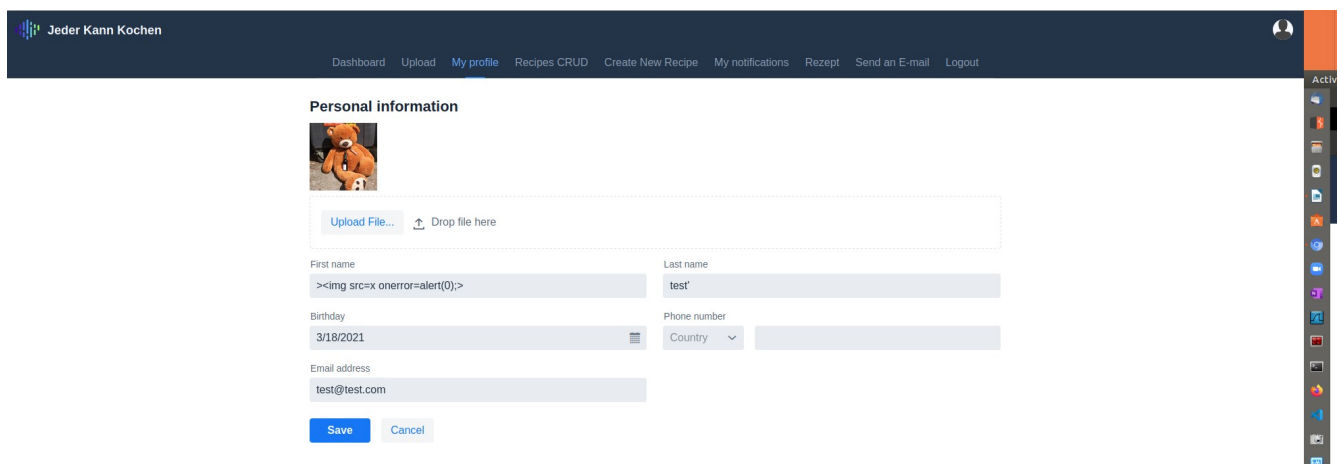
Uploaded files represent a significant risk to applications. The first step in many attacks is to get some code to the system to be attacked. Then the attack only needs to find a way to get the code executed. Using a file upload helps the attacker accomplish the first step.

So in our Project we check the type of the file, which is the images type "jpg" are only allowed.

The file types allowed to be uploaded should be restricted to only those that are necessary for business functionality.

Profile - Input Validation : Cross-site scripting (XSS)

Input validation is performed to ensure only properly formed data is entering the workflow in an information system, preventing malformed data from persisting in the database and triggering malfunction of various downstream components. Input validation should happen as early as possible in the data flow, preferably as soon as the data is received from the external party.



The screenshot shows a web application interface for a user named 'Jeder Kann Kochen'. The navigation bar includes links for Dashboard, Upload, My profile, Recipes CRUD, Create New Recipe, My notifications, Rezept, Send an E-mail, and Logout. The 'Personal information' section displays a profile picture of a teddy bear and a file upload area. Below this, there are input fields for First name, Last name, Birthday, Phone number, and Email address. The First name field contains the malicious payload: `>`. The Last name field contains 'test'. The Birthday field contains '3/18/2021'. The Phone number field has a 'Country' dropdown menu. The Email address field contains 'test@test.com'. At the bottom of the form are 'Save' and 'Cancel' buttons. The browser's taskbar is visible on the right side of the screen.

Input validation strategies

Input validation should be applied on both syntactical and Semantic level.

Syntactic: validation should enforce correct syntax of structured fields (e.g. SSN, date, currency symbol).

Semantic: validation should enforce correctness of their values in the specific business context (e.g. start date is before end date, price is within expected range).

It is always recommended to prevent attacks as early as possible in the processing of the user's (attacker's) request. Input validation can be used to detect unauthorized input before it is processed by the application.

Recipes CRUD, Create new recipe, Rezept:

We checked also the input validation for these features, so the attacker will not be allowed to pretend Cross site scripting, XSS vulnerabilities most often happen when user input is incorporated into a web server's response (i.e., an HTML page) without proper escaping or validation.

Dashboard Upload My profile Recipes CRUD Create New Recipe My notifications Rezept Send an E-mail Logout	
Suchen durch Zutaten	
Rezept Name	Inhalte
Kaesebroetchen	Pizza
koshary	Zwiebel, Knolauch
pasta	Zwiebel, Knolauch
Cremiger Nudelauflauf	Zwiebel Knoblauch Sahne Mozzarella Olivenöl Salz und Pfeffer, schwarzer
Spaghetti mit Bärlauch	Spaghetti Knoblauch Parmesan
Vegane Pilz-Reispfanne	Austernpilze Pflanzenöl Gemüsebrühe
Mexikanisches Spicy-Rindfleisch mit Erdnüssen	1Paprikaschote(n) grün300 gHüftsteak hilischote BundKoriander 2 ELPflanzenöl 125g Kidneybohnen aus der Dose 140g Mais a
test	<p>test</p>

Notifications and Send E-mail : rate limiting and lack of resources

rate limiting is used to control the **rate** of traffic sent or received by a network interface controller and is used to prevent DoS attacks. **No rate limit** means there is **no** mechanism to protect against the requests you made in a short frame of time.

How to prevent

- Define proper rate limiting.
- Limit payload sizes.
- Tailor the rate limiting to be match what API methods, clients, or addresses need or should be allowed to get.
- Add checks on compression ratios.
- Define limits for container resources.

Tools :

- OWASP
- Burp Suite
- Dirbuster
- Nmap