**Cybersecurity 401**

**Module 8 - Web Application Security**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

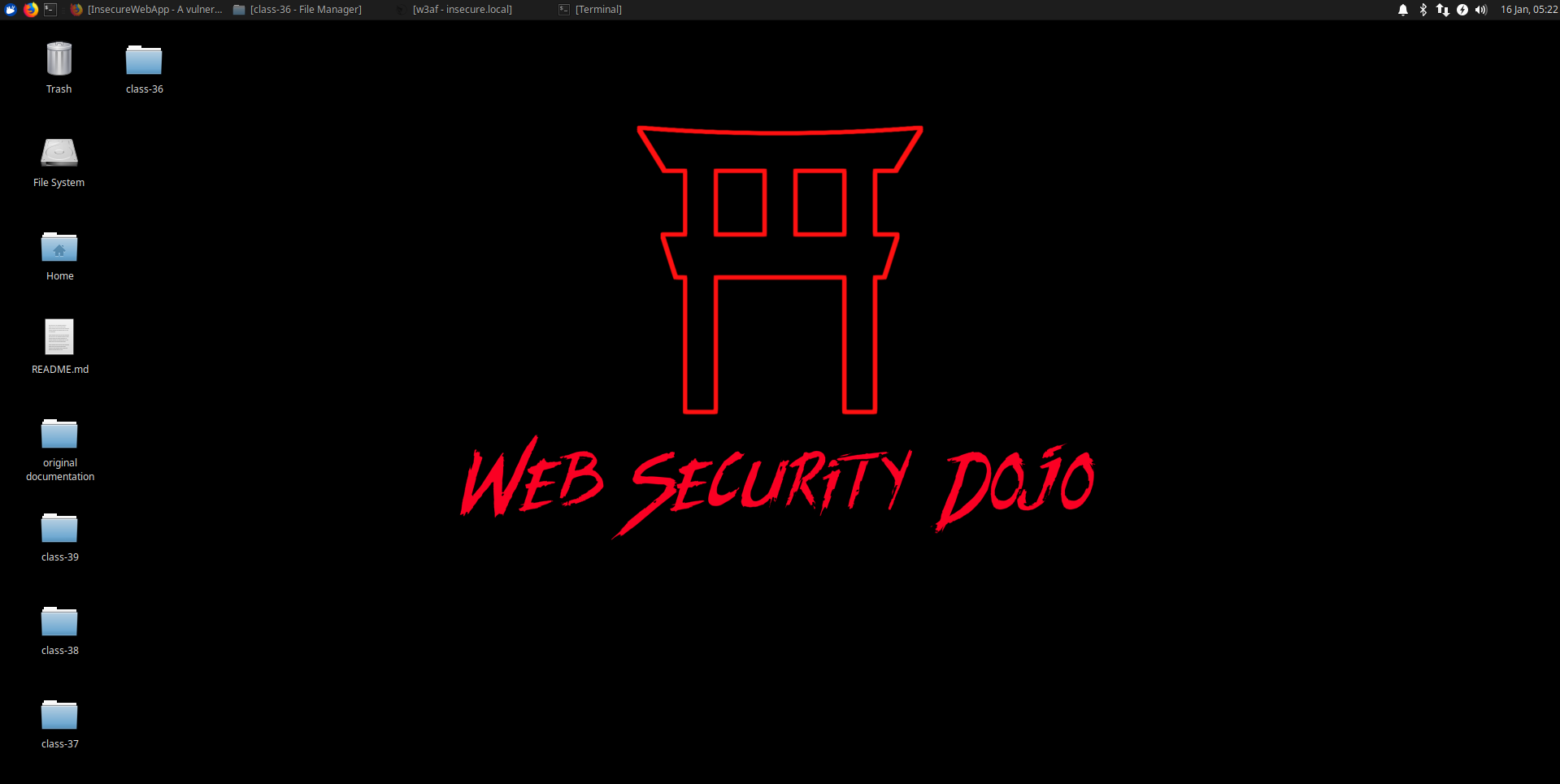
## **Lab 37 - Automated AppSec with OWASP ZAP**

## 

**| Rodrigo Brasil 01/2024 |**

### **Part 1: Staging**

**This lab requires Web Security Dojo VM.**

****

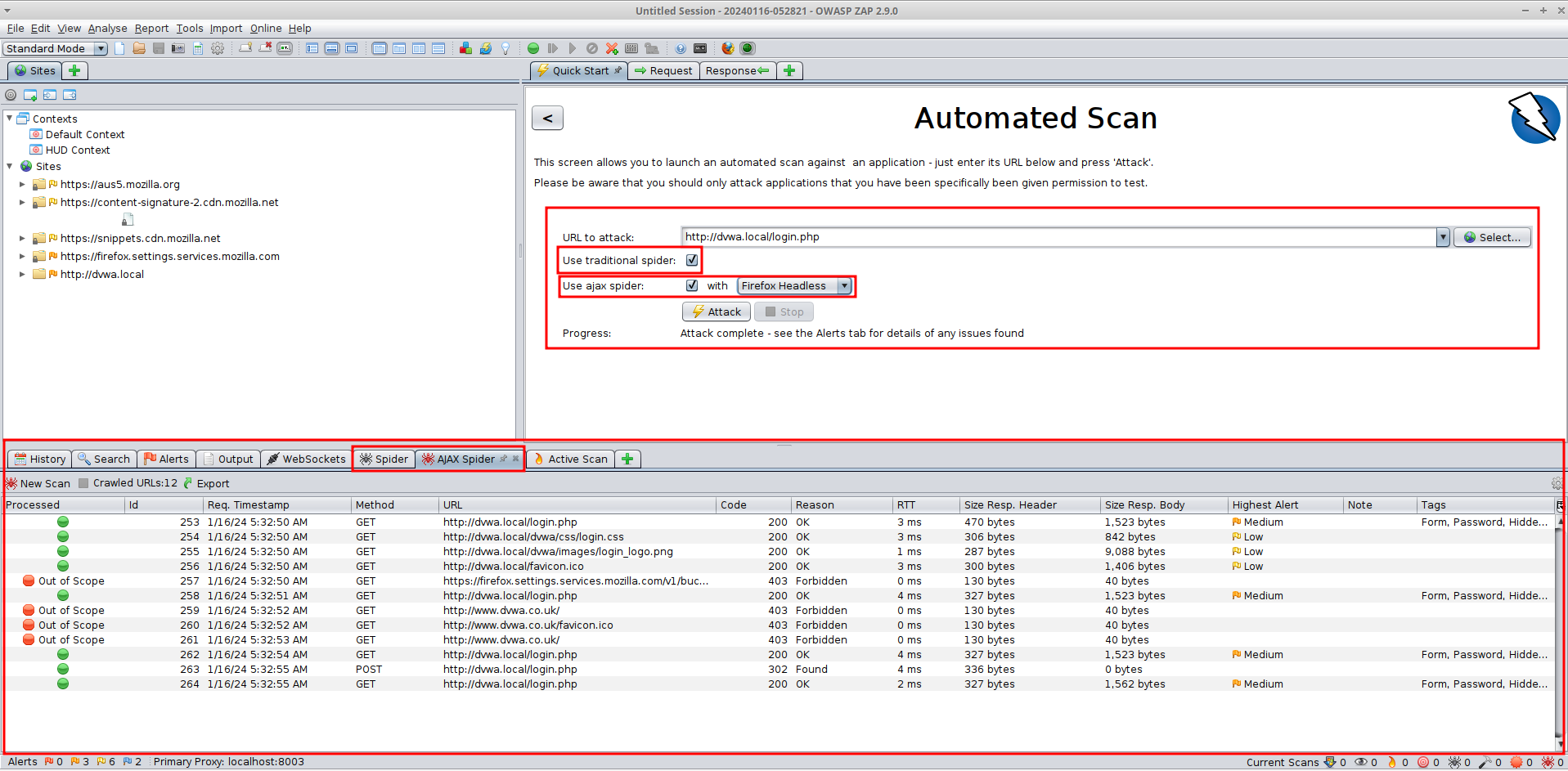
### 

### **Part 2: Test DVWA using ZAP**

**For this part of the lab, access** [**TryHackMe Introduction to OWASP ZAP**](https://tryhackme.com/room/learnowaspzap)**.**

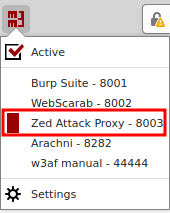
***Note that AttackBox, TryHackMe’s lab environment, is not required for this lab since we’re using Web Security Dojo for our entire environment. Furthermore, the AJAX Spider is already installed on our edition of ZAP, so that does not need to be installed in Task 4.***

* **Complete TryHackMe’s tasks 4-10 using the Web Security Dojo. Document the things you learn in your submission for the day.**
  + **Tasks 1-3: These are not necessary since we’re using Web Security Dojo. Skip these steps.**
  + **Task 4: AJAX Spider is already installed in ZAP. Run AJAX Spider in conjunction with the original spider to perform a crawl of DVWA. Paste your AJAX Spider scan output in your submission.**

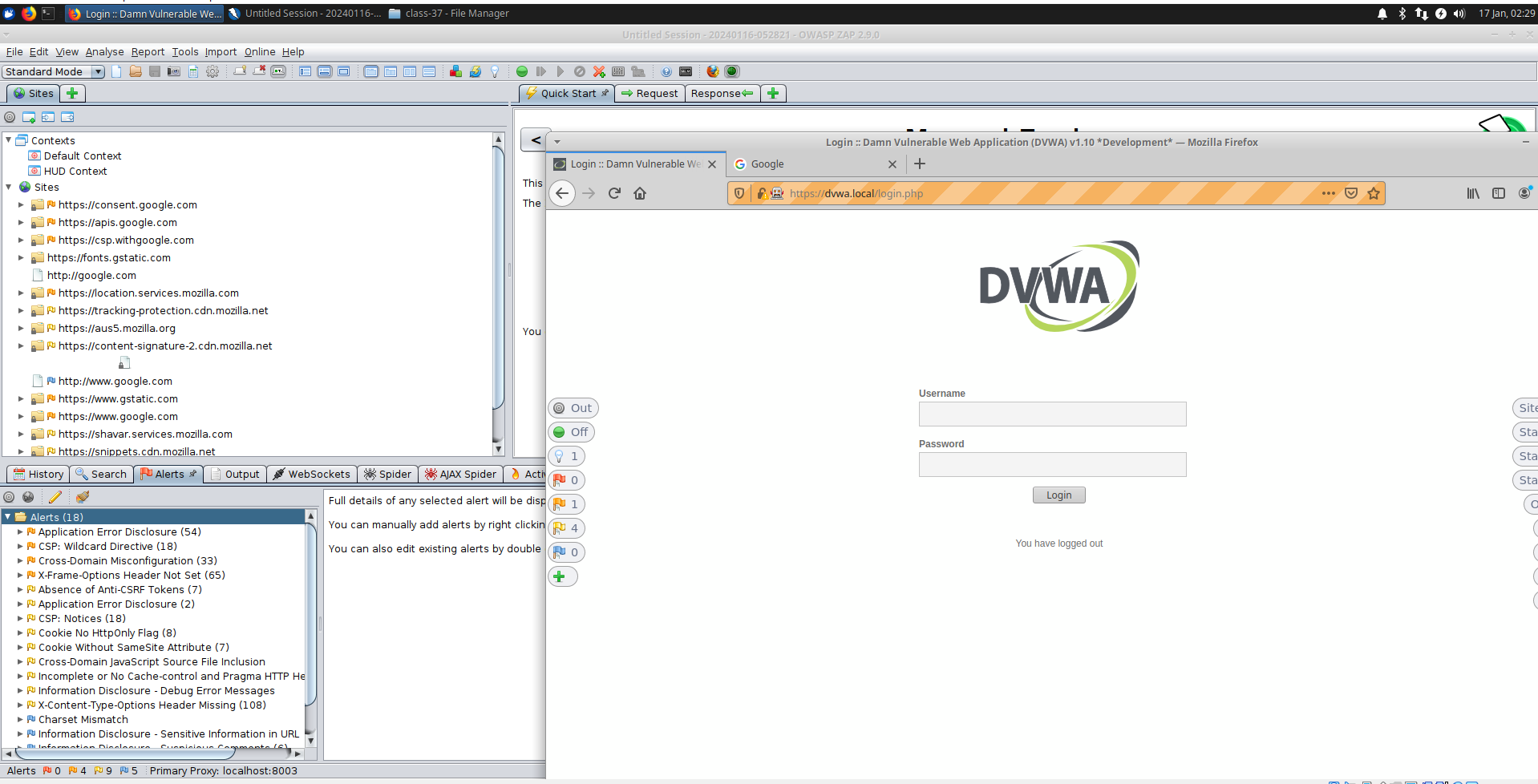
****

* + **Task 5: Configure ZAP as a proxy server in your browser (typically Firefox).**
    - **There are two paths you can take here: either utilize FoxyProxy + MM3-ProxySwitch to quickly toggle the proxy to ZAP …OR… disable these Firefox addons and manually configure a proxy as depicted in the TryHackMe instructions.**
    - **Describe the steps taken to establish proxy connectivity.**

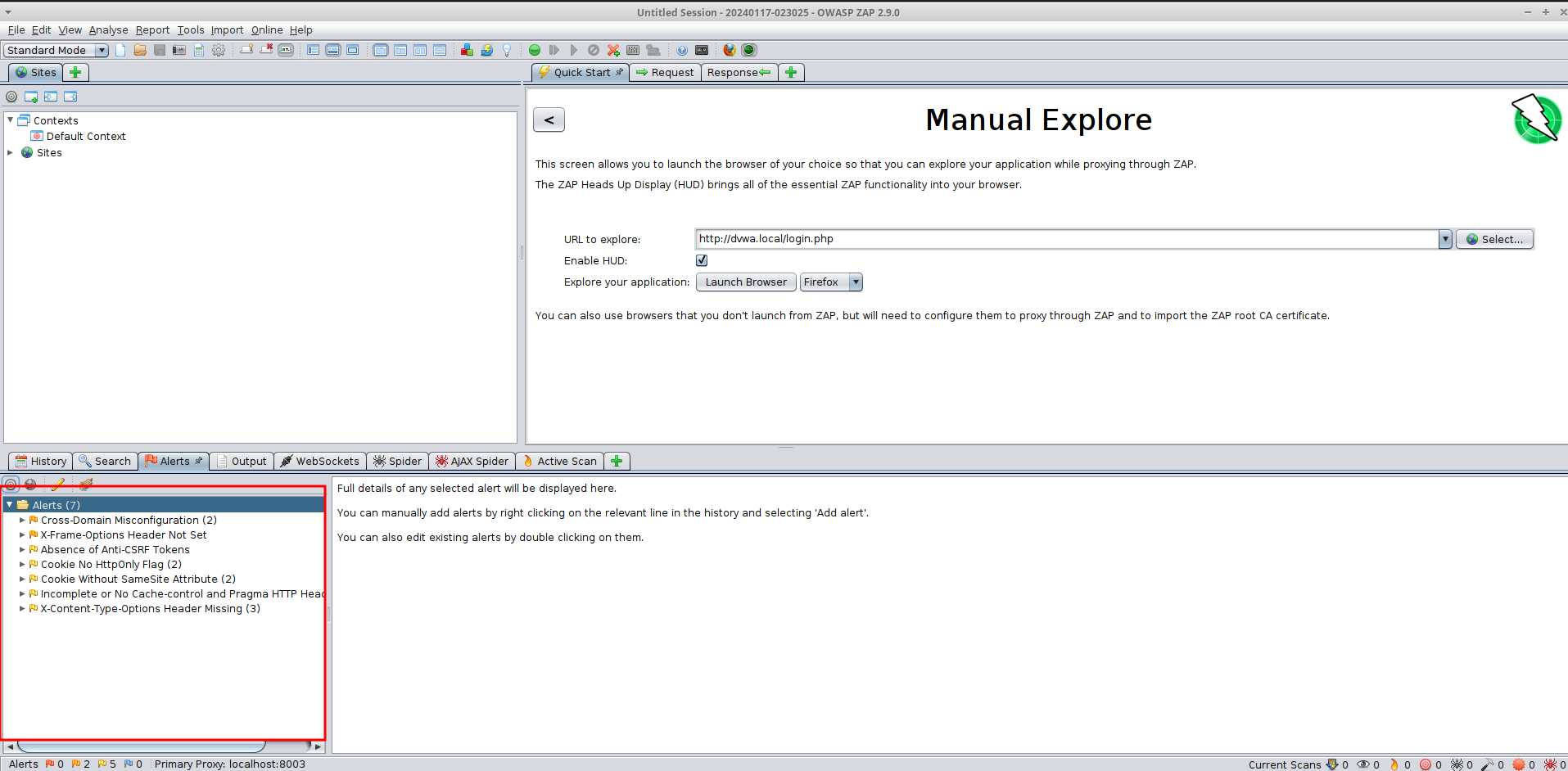
****

****

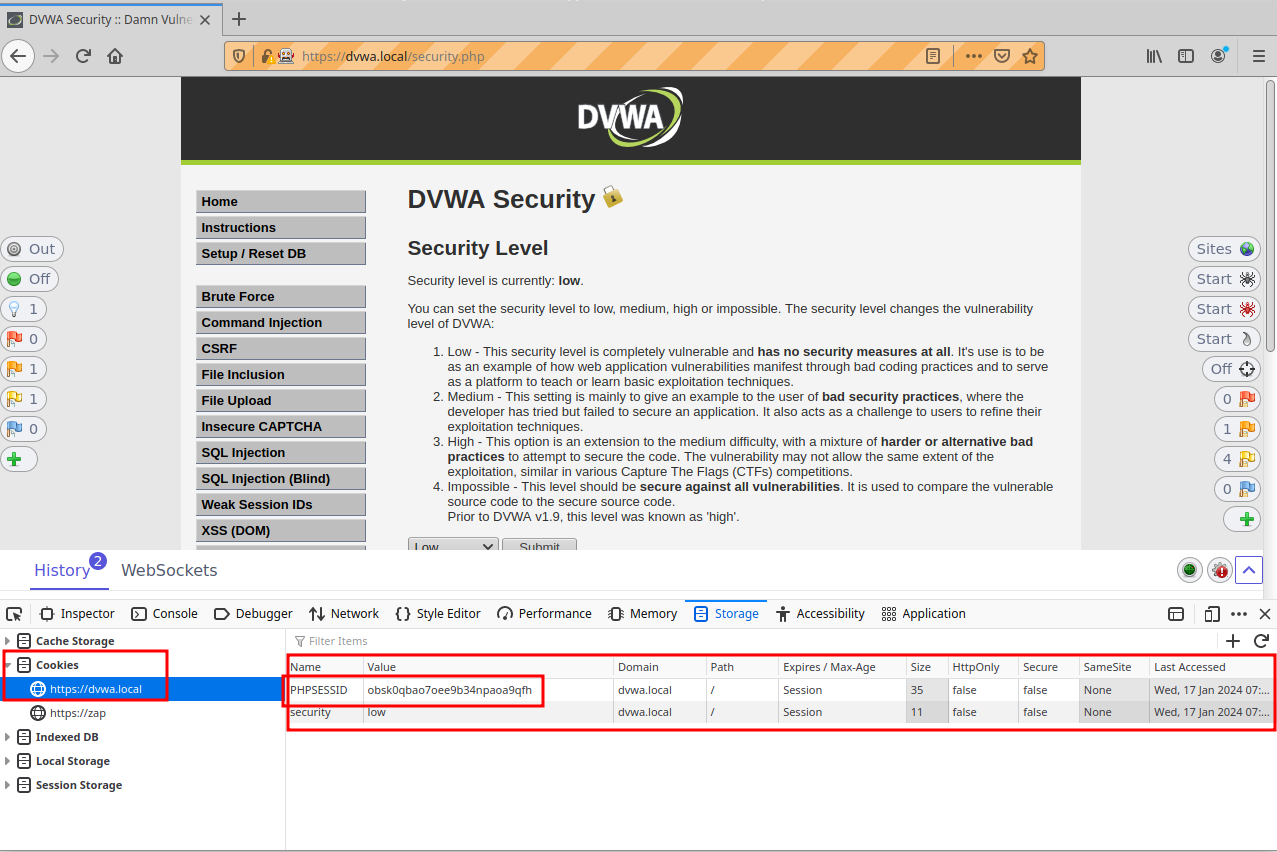
* + - **Include a screenshot of the traffic updating in ZAP as you are navigating via a proxied Firefox window.**

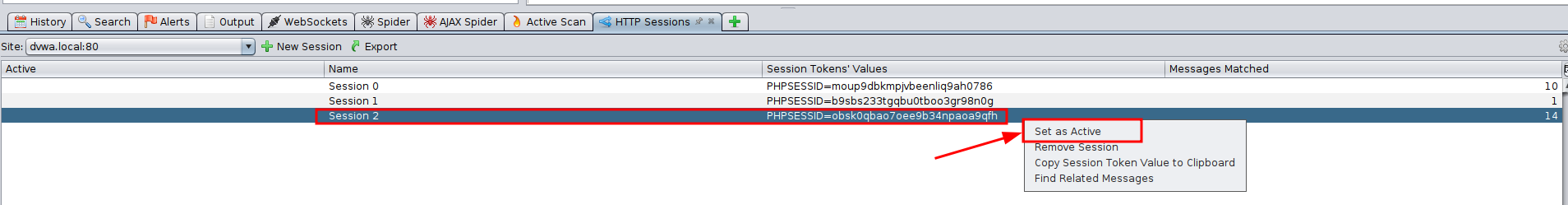
****

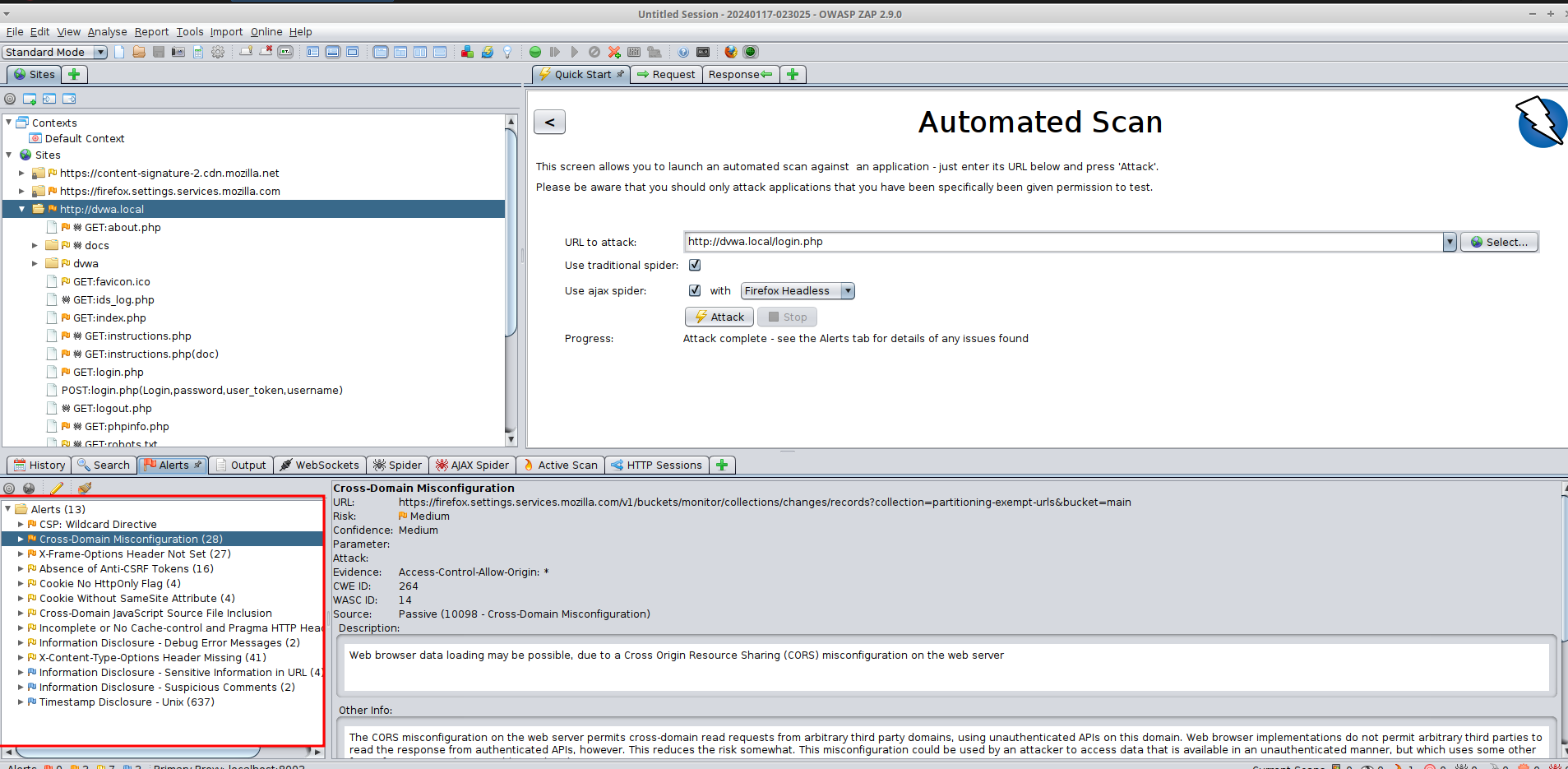
* + - **Scan DVWA login page and include a screenshot of the results.**

****

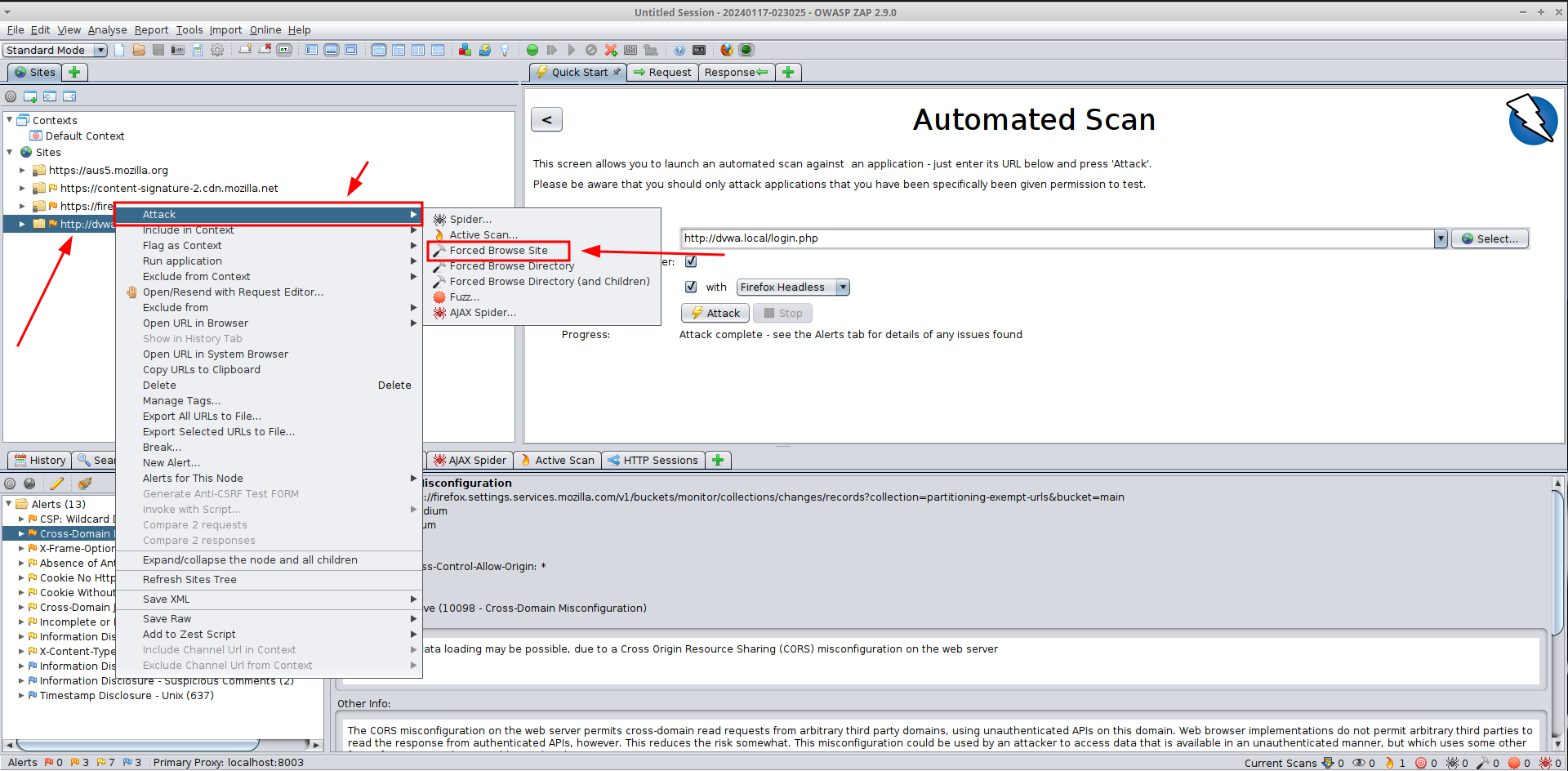
* + **Task 6: This time scan DVWA as an authenticated user and include a screenshot of the results.**
    - **How did you establish an authenticated session as ZAP?**

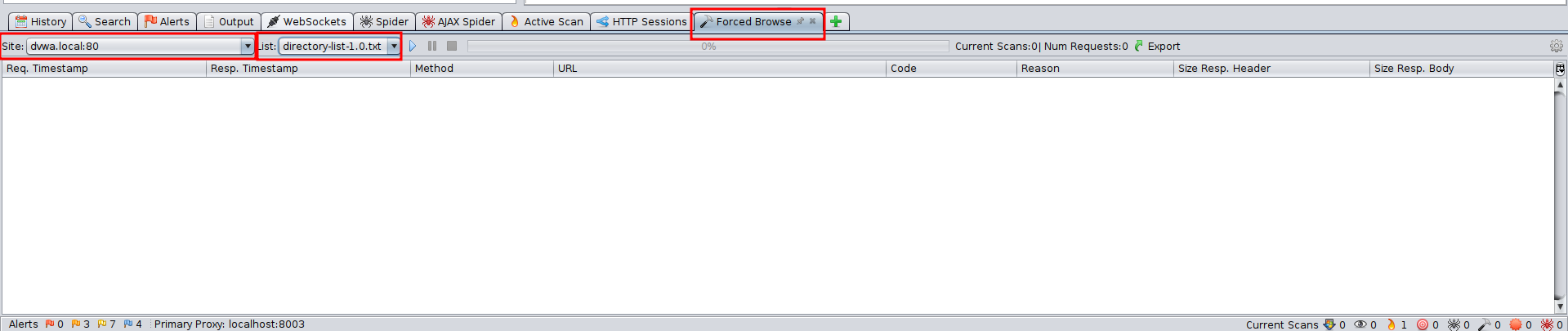
****

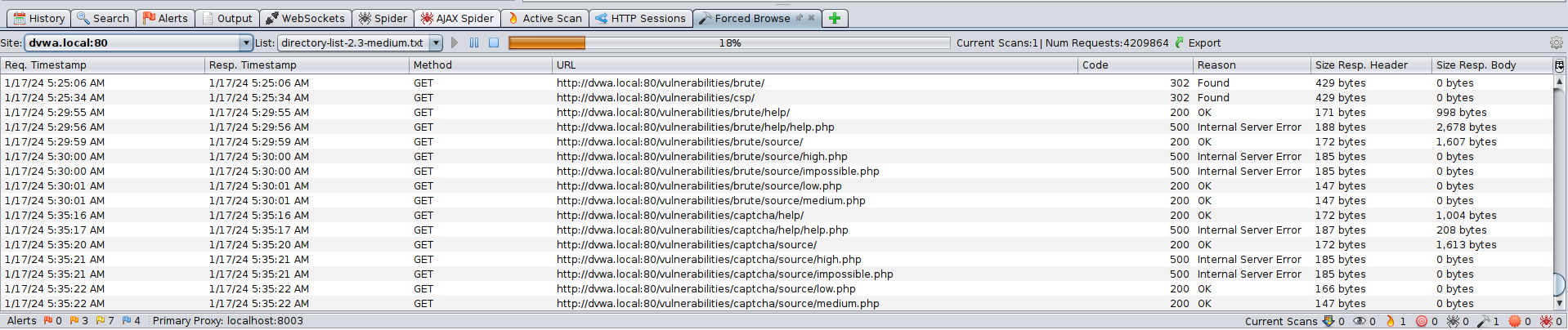
****

****

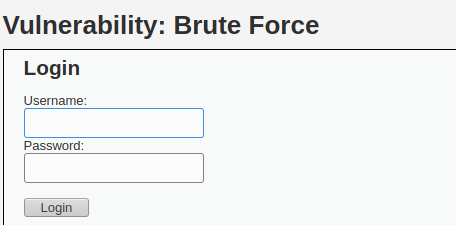
* + **Task 7: Brute force DVWA with a word list. Include the results.**

****

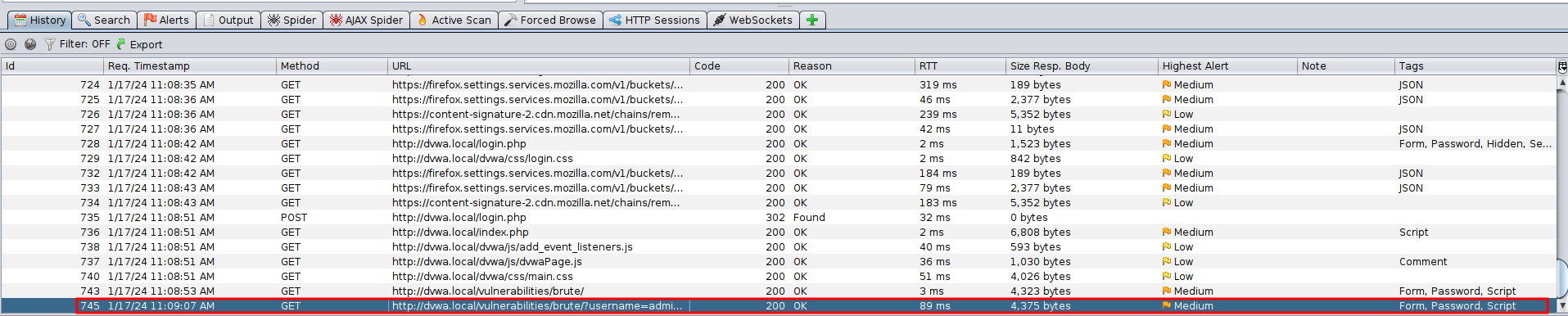
****

****

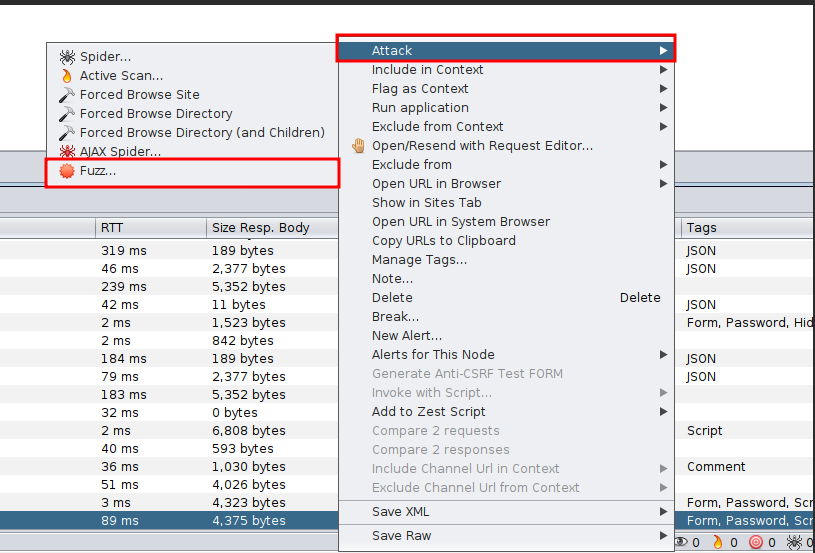
* + **Task 8: Use ZAP to bruteforce the DVWA ‘brute-force’ page. What’s the password?**

****

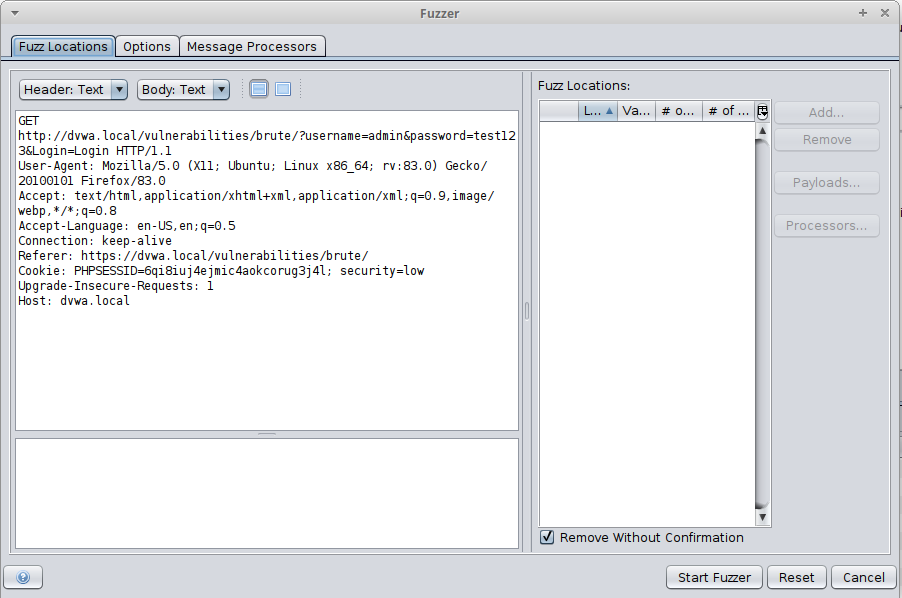
**Enter some credentials to get a GET from the web app**

****

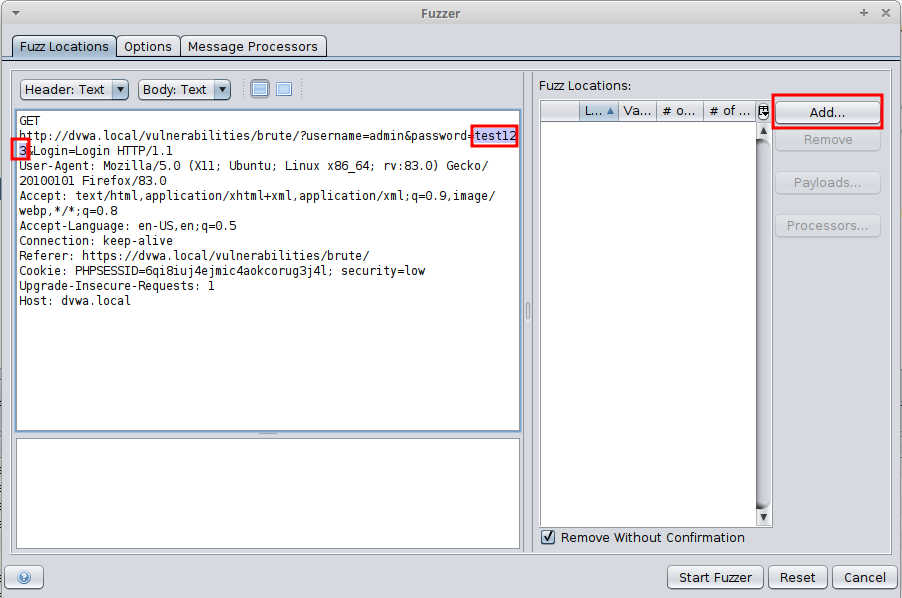
**The GET**

****

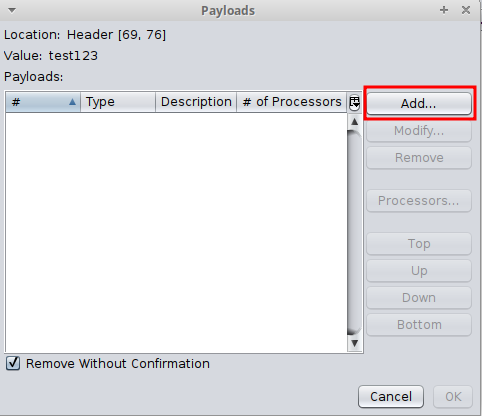
**Right click it, select Attack and Fuzz…**

****

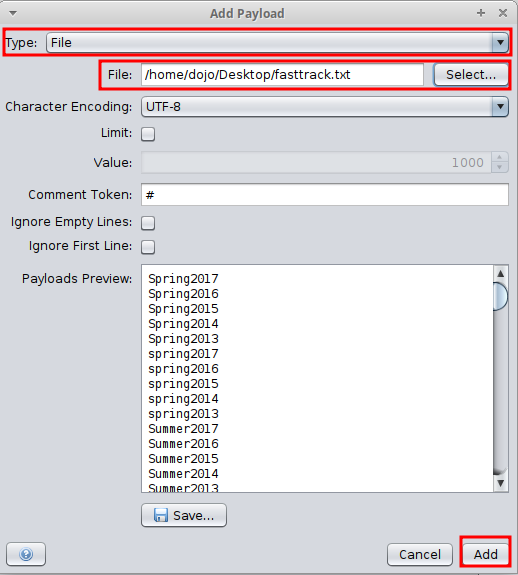
**It will open this window**

****

**in here highlight the password we used on the site and click add**

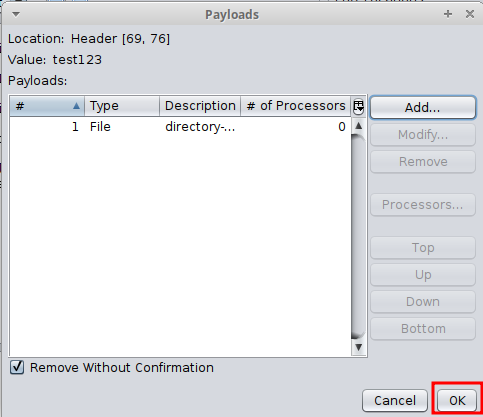
****

**click add again**

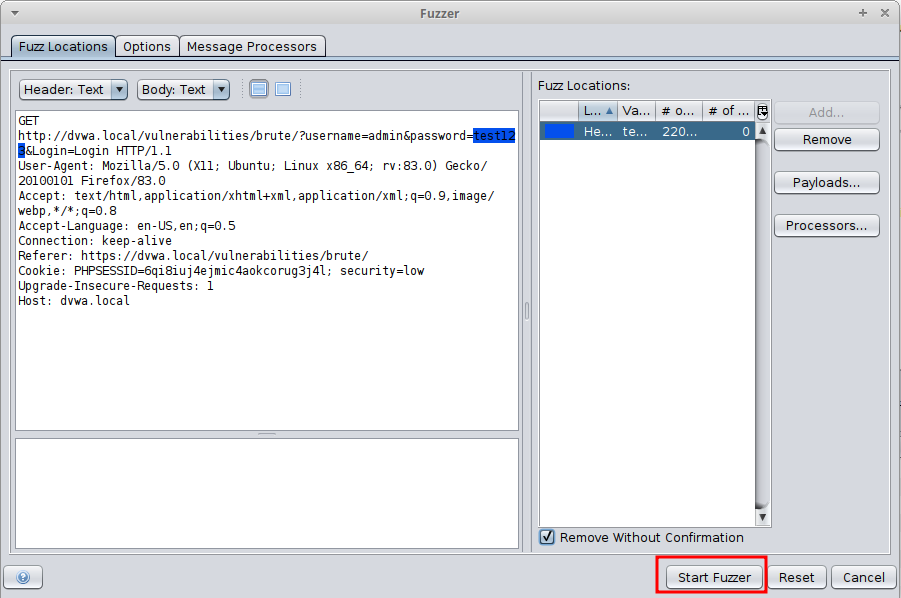
****

**In here we will select our dictionary to brute-force the password**

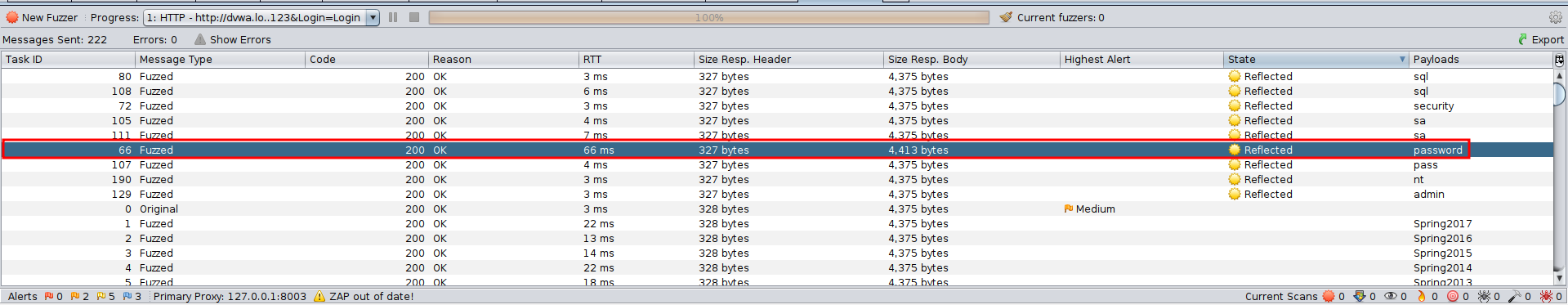
**On “Type” select file and on “File” locate, select the dictionary and click add**

****

**in here simply click “OK”**

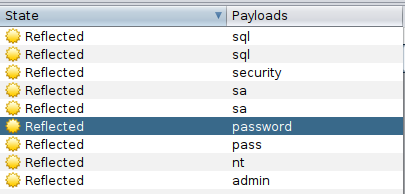
****

**And “Start Fuzzer”**

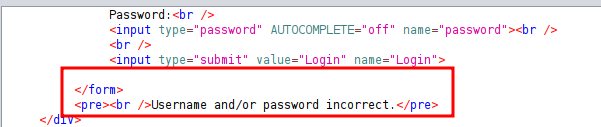
****

**It will start scanning, after finishing we will click the state to show the reflected on top.**

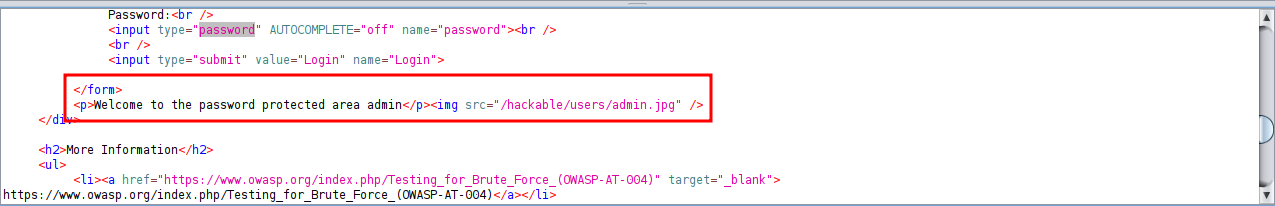
**In here we only want passwords that are at least 8 characters long**

****

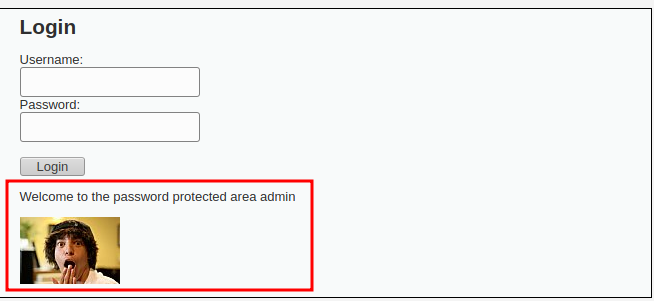
**only “security” and “password” match that requirement**

****

**we can see that security didn't work**

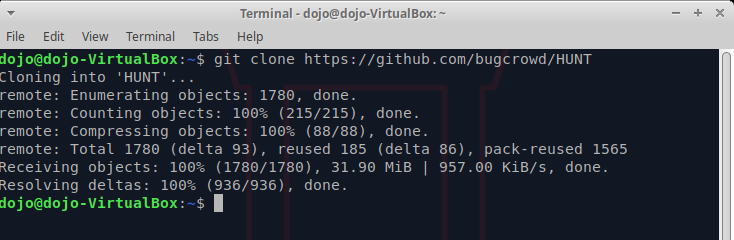
****

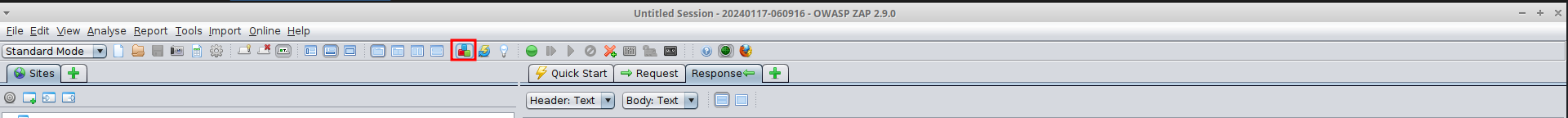
**but password did**

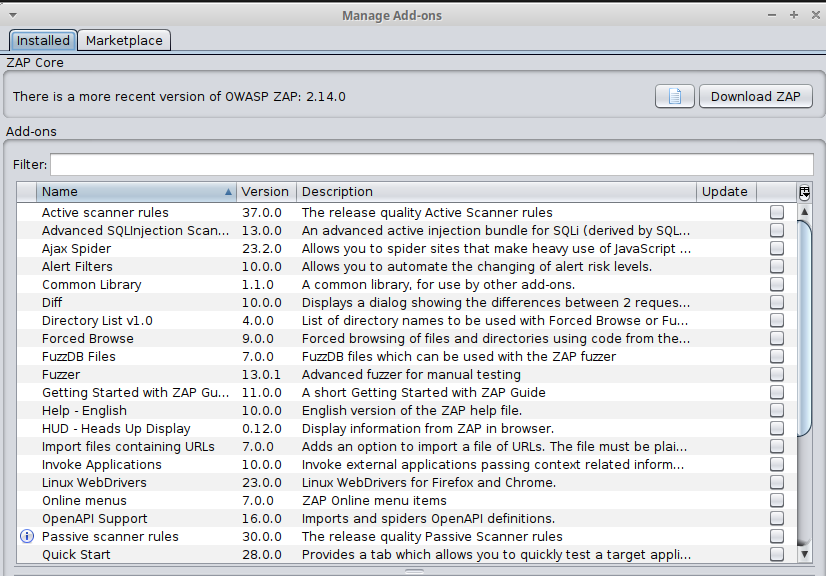
****

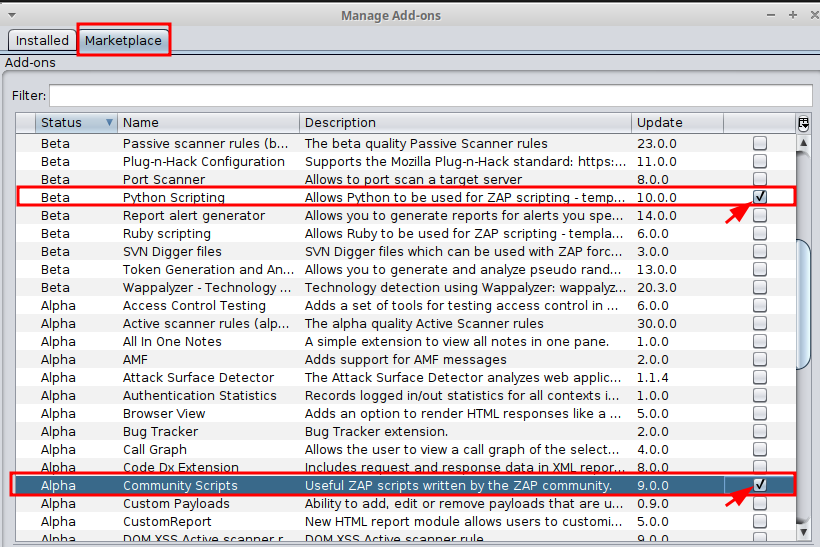
**now changing the password from test123 to password we can successfully login**

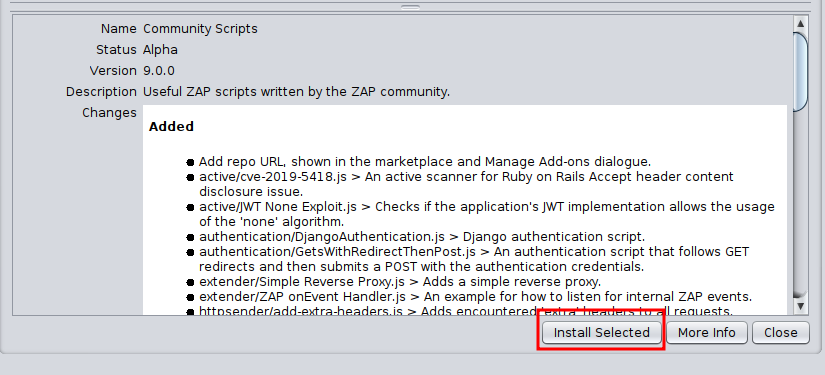
* + **Task 9: Install the** [**bugcrowd HUNT extensions**](https://github.com/bugcrowd/HUNT) **for OWASP ZAP. Include a screenshot of the installed HUNT scripts.**

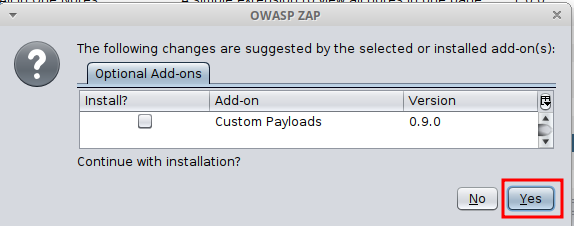
****

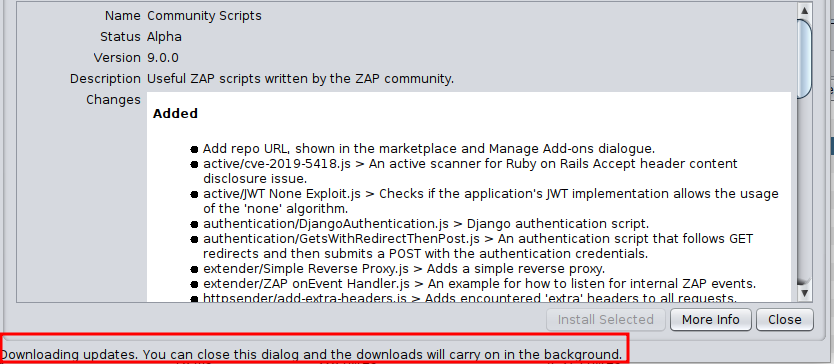
****

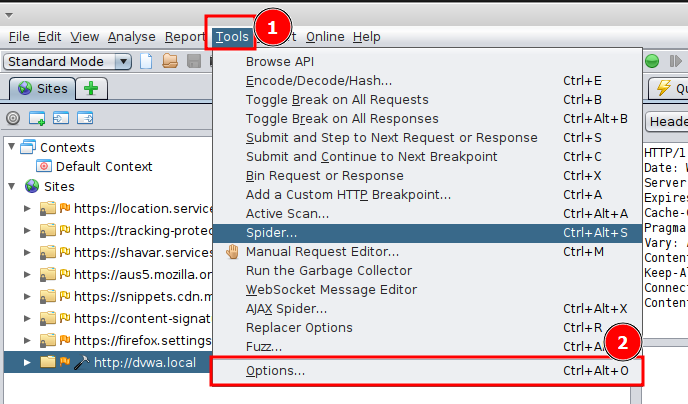
****

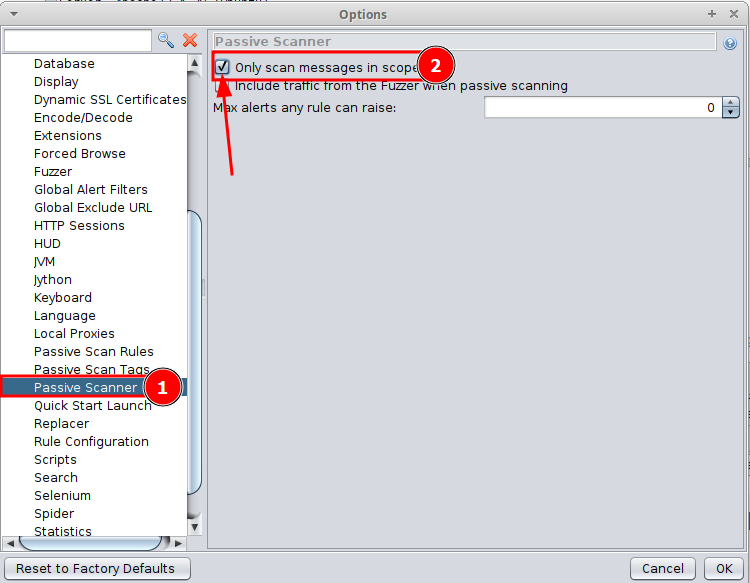
****

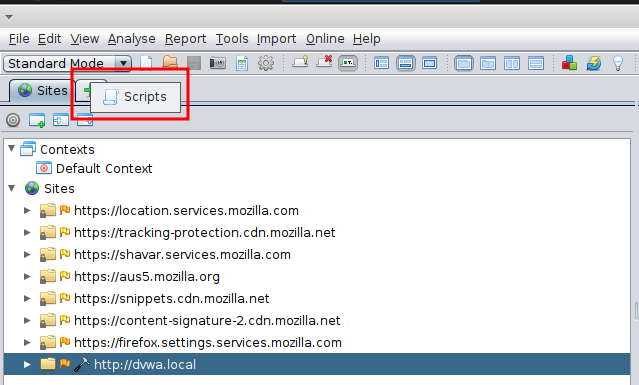
****

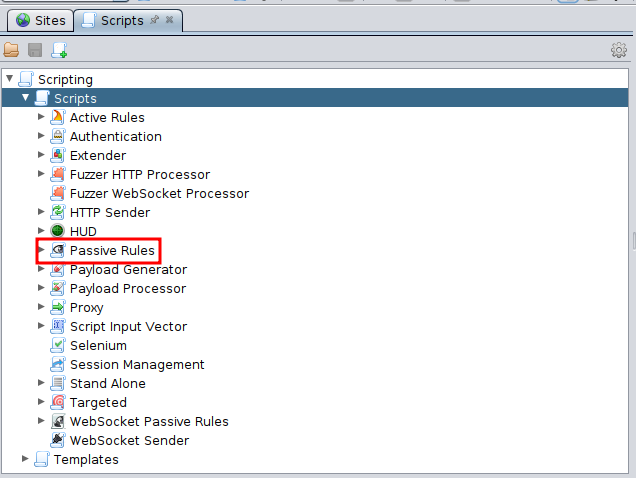
****

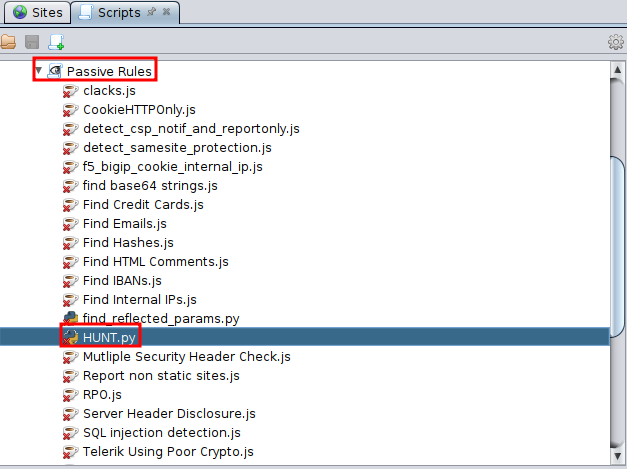
****

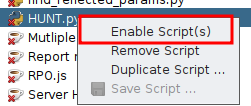
****

****

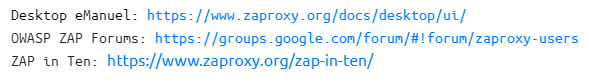
****

****

****

****

* + **Task 10: Bookmark the provided links as resources to continue your ZAP training in the future.**

****

### 

### **Part 3: Report Your Findings**

**Take stock of your findings and perform the following reporting tasks:**

* **Examine your findings. What are the most severe vulnerabilities present on the DVWA and why?**
  + **Brute-force, clickjacking and SQL injection.**
* **Select an interesting vulnerability you’d like to research from the list of findings.**
  + **From the list of findings, the clickjacking attack looks interesting.**
* **Follow the** [**OWASP Vulnerability Template**](https://owasp.org/www-community/vulnerabilities/Vulnerability_template)**, complete a comprehensive description of this vulnerability in your submission doc. You may wish to also include this in your public-facing GitHub repo.**

**Description**

The vulnerability involves a brute force attack leveraging the penetration testing tool Zap (OWASP Zed Attack Proxy) to attempt all possible combinations of passwords until a successful login is achieved.

The vulnerability arises from a potential lack of proper security measures, allowing attackers to exploit weaknesses in authentication mechanisms.

What are the attacks that target this vulnerability?

Attackers targeting this vulnerability may employ Brute Force Attacks, Dictionary Attacks, Credential Stuffing, Rainbow Table Attacks, Password Spraying, Session Hijacking

What are the technical impacts of this vulnerability?

The technical impacts of this vulnerability involve the risk of unauthorized access, potential data breaches, compromise of user accounts leading to misuse of privileges, possible application and system compromise, denial of service scenarios, reputational damage, and legal and compliance issues arising from the compromise of sensitive data.

**Risk Factors**

The likelihood of this vulnerability being exploited depends on several factors. Firstly, the effectiveness of the brute force attack is influenced by the strength of the authentication mechanisms in place. If the application has weak or easily guessable passwords, the vulnerability becomes more likely to be exploited.

The primary consequence is gaining unauthorized access to the web app. Attackers could potentially infiltrate sensitive areas, compromising confidentiality and privacy.

Depending on the role and permissions associated with the compromised account, a successful exploit could lead to unauthorized access to sensitive data, resulting in a data breach.

User accounts with weak passwords are especially vulnerable. A successful brute force attack may lead to the compromise of individual user accounts, allowing attackers to impersonate legitimate users.

If the compromised account has limited privileges, a successful exploit might involve escalating those privileges, gaining higher-level access within the system and increasing the potential impact of the breach.

Once inside the system, attackers may use the compromised account as a foothold for further reconnaissance and lateral movement, exploring and compromising other parts of the network.

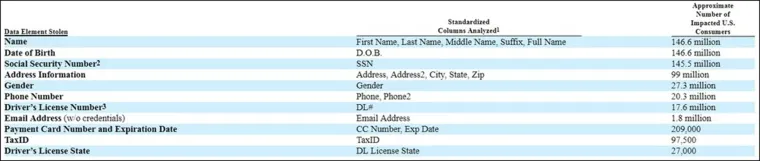
A successful attack may lead to a data breach, exposing sensitive information. The costs associated with investigating, mitigating, and recovering from a data breach can be substantial, including legal expenses, regulatory fines, and potential compensation to affected parties.

The compromise of sensitive data and unauthorized access can severely damage the reputation of the business. Loss of customer trust and confidence may result in a decline in customer loyalty, impacting long-term relationships and brand value.

**Examples**

**2017 Equifax data breach**

**In September 2017, Equifax had a data breach that exposed the personal information of approximately 147 million people. The breach started on May 12, 2017, when Equifax had yet to update its credit dispute website with the new version of Struts. The hackers used the exploit to gain access to internal servers on Equifax's corporate network.**

****

[**https://www.csoonline.com/article/567833/equifax-data-breach-faq-what-happened-who-was-affected-what-was-the-impact.html**](https://www.csoonline.com/article/567833/equifax-data-breach-faq-what-happened-who-was-affected-what-was-the-impact.html)

**Short example name**

**- A short example description, small picture, or sample code with links**

**Related Attacks**

**Attack 1**

**Attack 2**

**Related Vulnerabilities**

**Vulnerability 1**

**Vulnerabiltiy 2**

**Note: the contents of “Related Problems” sections should be placed here**

**Related Controls**

**Control 1**

**Control 2**

**Note: contents of “Avoidance and Mitigation” and “Countermeasure” related Sections should be placed here**

**Related Technical Impacts**

**Technical Impact 1**

**Technical Impact 2**

**References**

**Note: A reference to related CWE or CAPEC article should be added when exists. Eg:**

**CWE 79.**

**http://www.link1.com**

**Title for the link2**

**In addition, one should classify vulnerability based on the following subcategories: Ex:[[Category:Error\_Handling\_Vulnerability|Category:Error Handling Vulnerability]]**

**Availability Vulnerability**

**Authorization Vulnerability**

**Authentication Vulnerability**

**Concurrency Vulnerability**

**Configuration Vulnerability**

**Cryptographic Vulnerability**

**Encoding Vulnerability**

**Error Handling Vulnerability**

**Input Validation Vulnerability**

**Logging and Auditing Vulnerability**

**Session Management Vulnerability**