CANDIDATE STUDY PLAN

Penetration Testing Resources

Abstract

This document will compile the results of the resources that I have been tasked to complete. It will contain screenshots from following along with the Beginner Web Application Hacking course, a breakdown of chapter 21 from 'A Web Application Hacker's Methodology' and some details of the things that I found useful within the other chapters. I will also include research of OWASP and MITRE common weaknesses, Portswigger Burpsuite, NMAP network scanning, and mobile application security. Lastly, I will include progress that I make in Portswigger Academy and TryHackMe hands on labs.

Rory James Murphy

roryjamesmurphy@hotmail.com

Table of Contents

Web Application

Beginner Web Application Hacking Course

https://www.youtube.com/watch?v=24fHLWXGS-M

Week 1: Reconnaissance

Target Validation

Important to validate that your target is indeed the target you are required to test.

Tools: WHOIS, nslookup, dnsrecon

Finding Subdomains

Useful information can be obtained by publicly known subdomains of your acquired target.

Tools: Google Fu, dig, Nmap, Sublist3r, Bluto, crt.sh

Fingerprinting

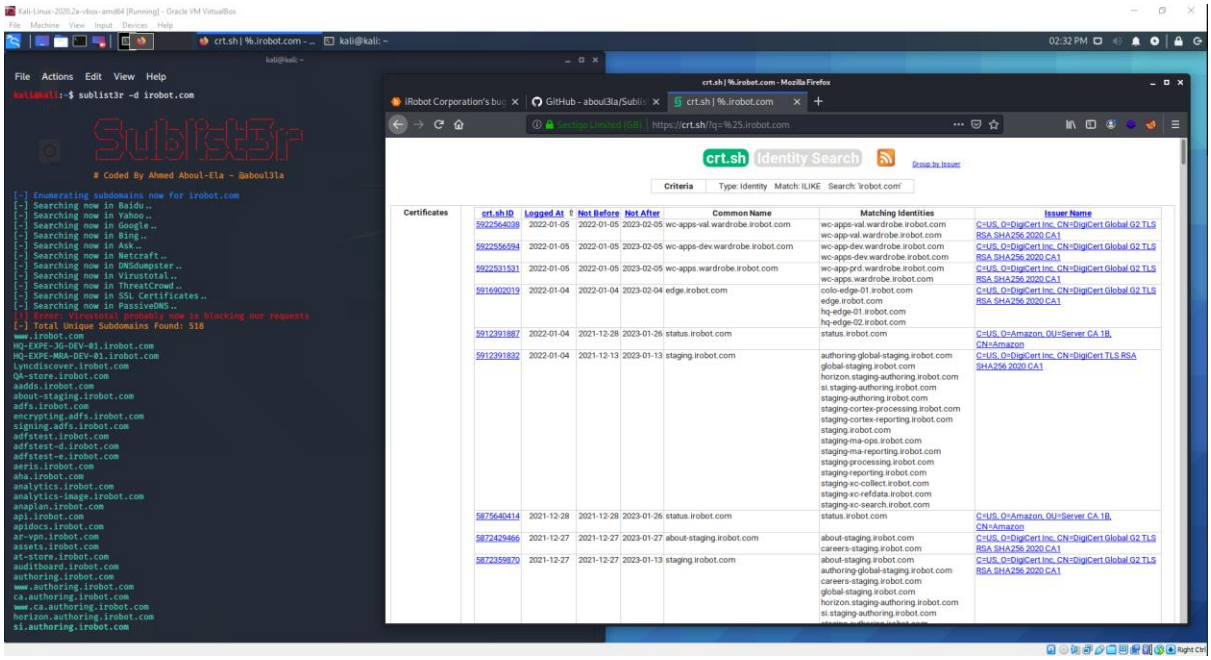
Find out what systems are running and what OS models are in use.

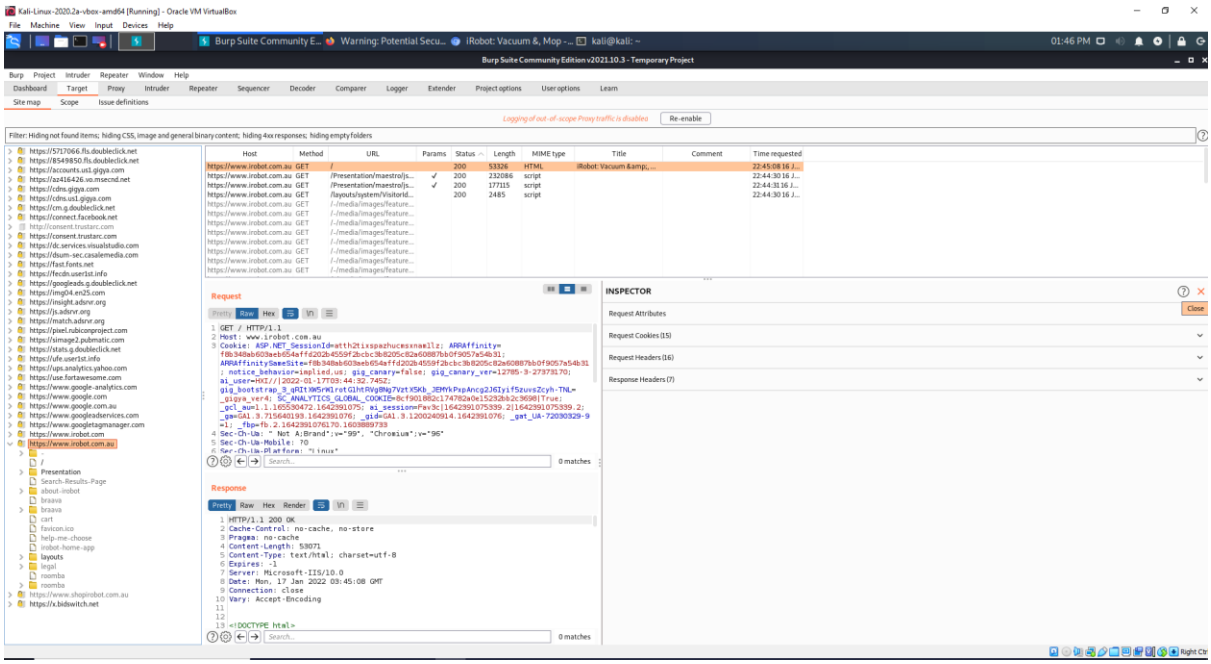
Tools: Nmap, Wappalyzer, WhatWeb, BuiltWith, Netcat

Data Breaches

Viable information through info dumps such as passwords or account names.

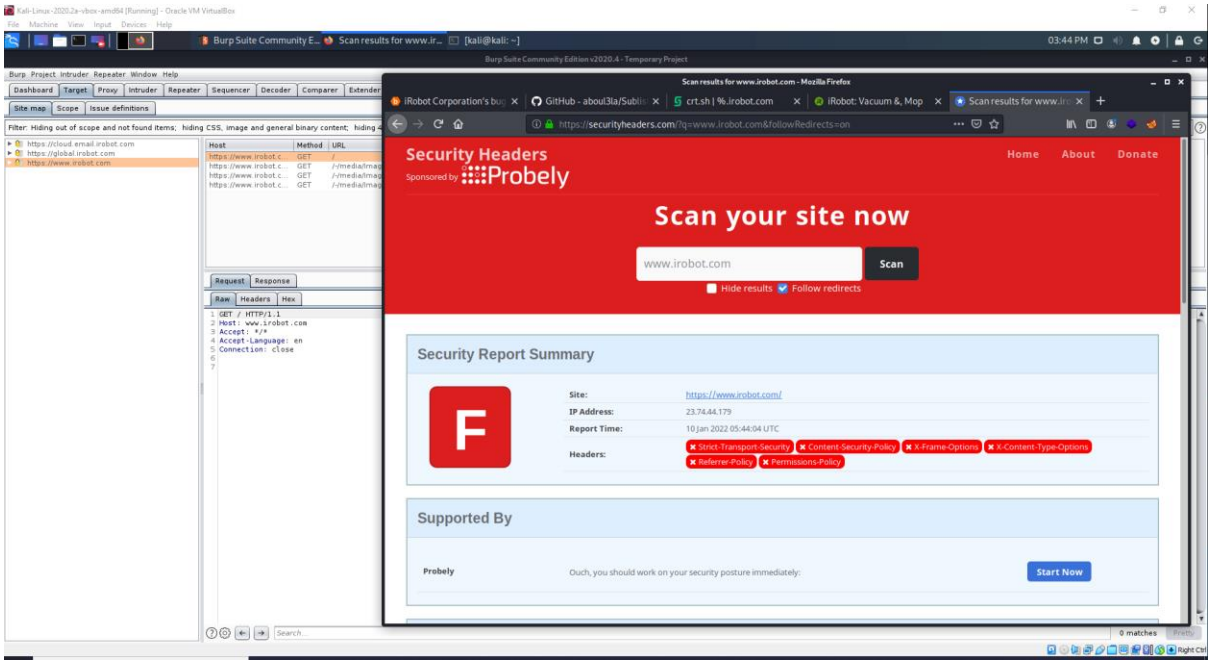
Tools: HaveIBeenPwned



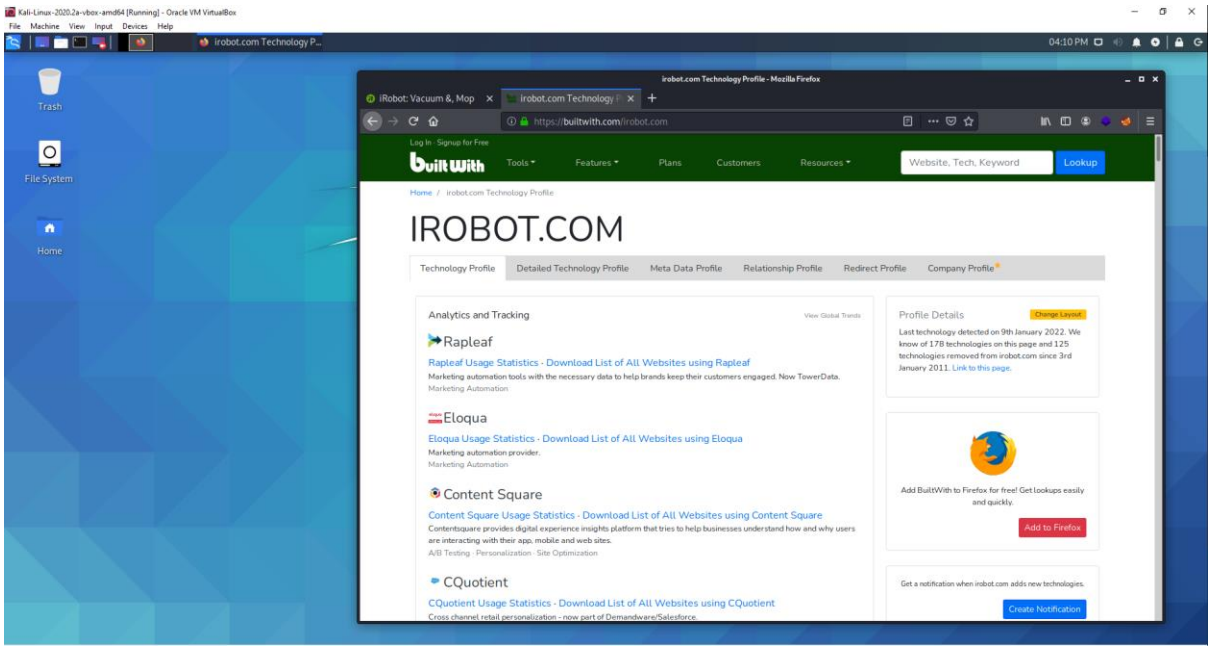


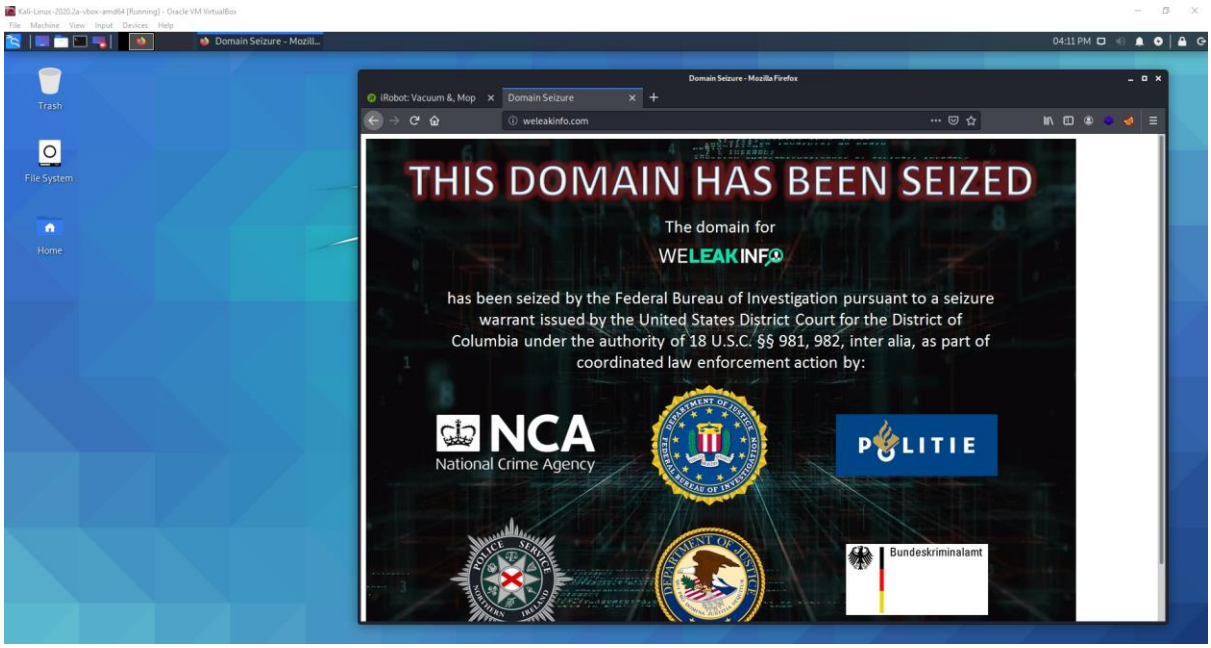
Add scope: .*\.url\.com$

Filter: show only in scope and remove any out of scope websites.

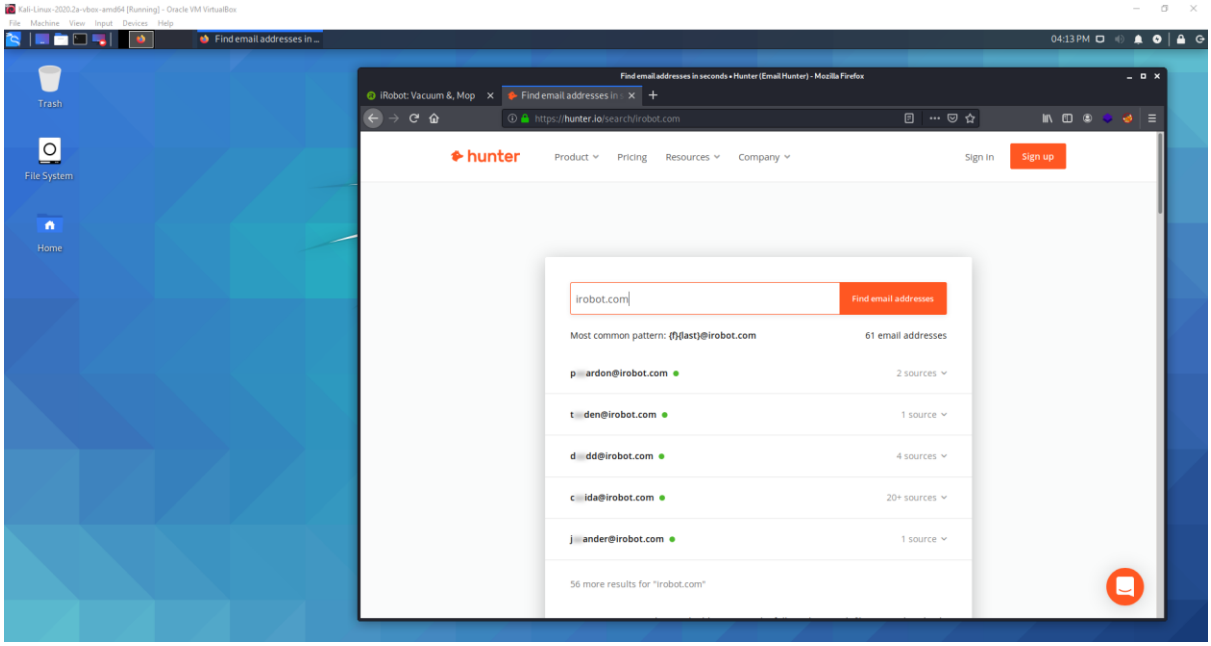


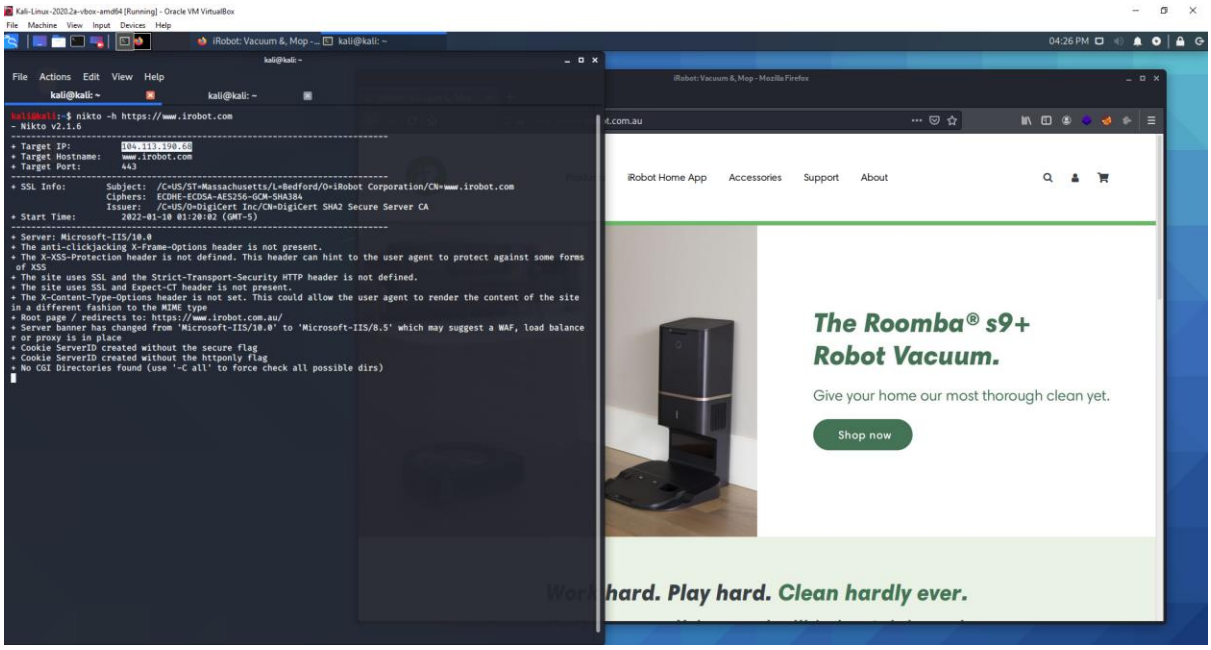
Header issues are low priority but still noteworthy to add into a pen testing report.

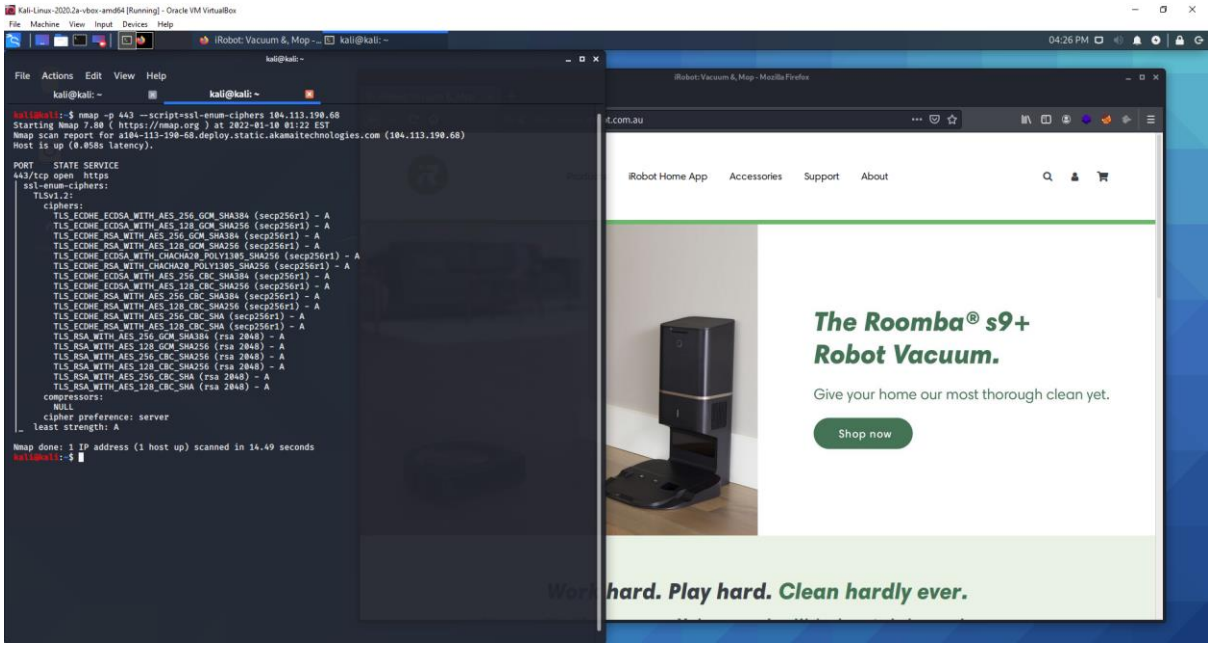




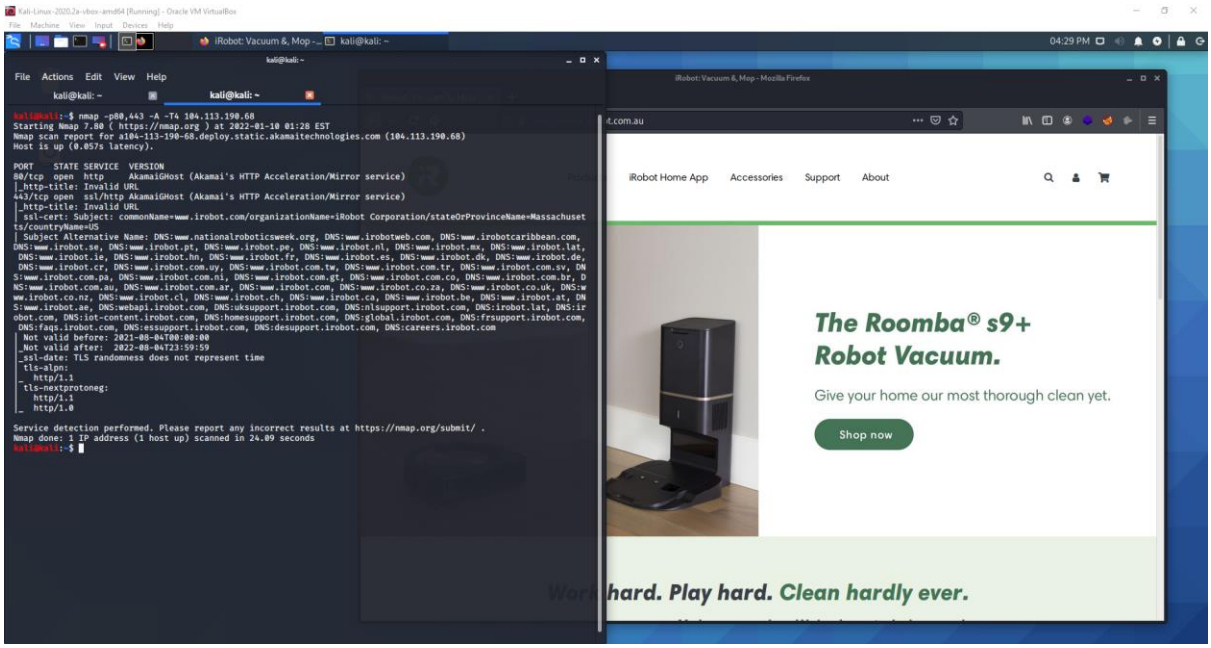
This tutorial must be a bit dated by now, but I thought it was pretty funny that one of the tool domains got seized.







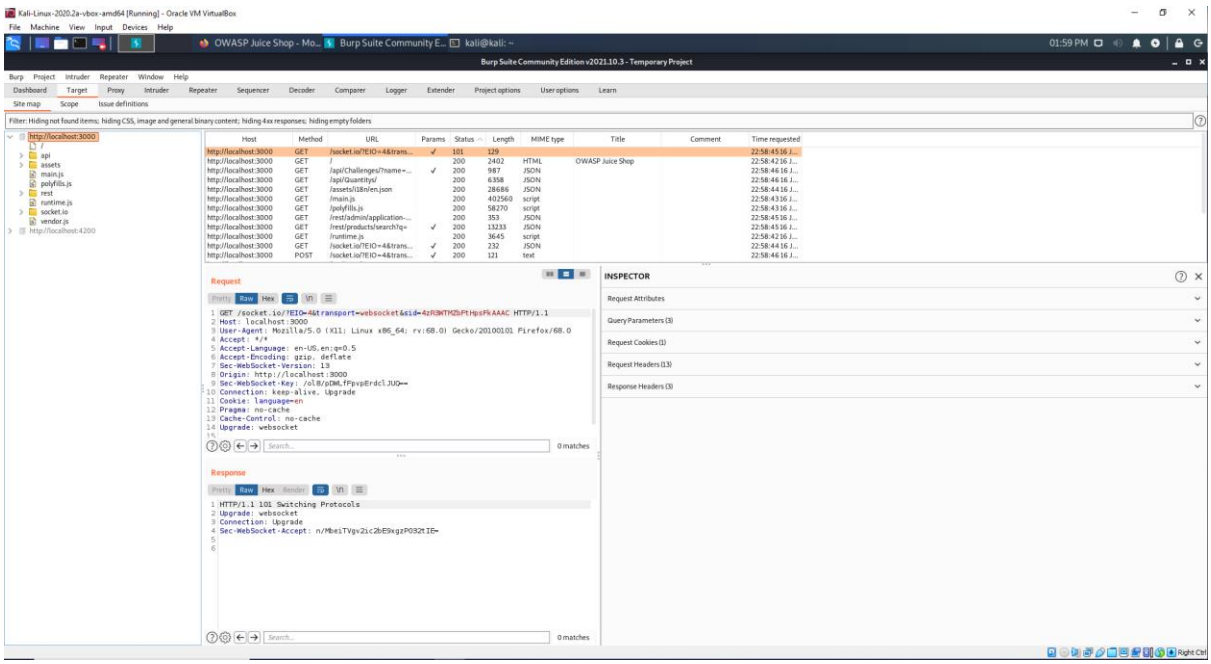
Ciphers are hard to break into but noteworthy in adding into a report if they report back with a low grade.

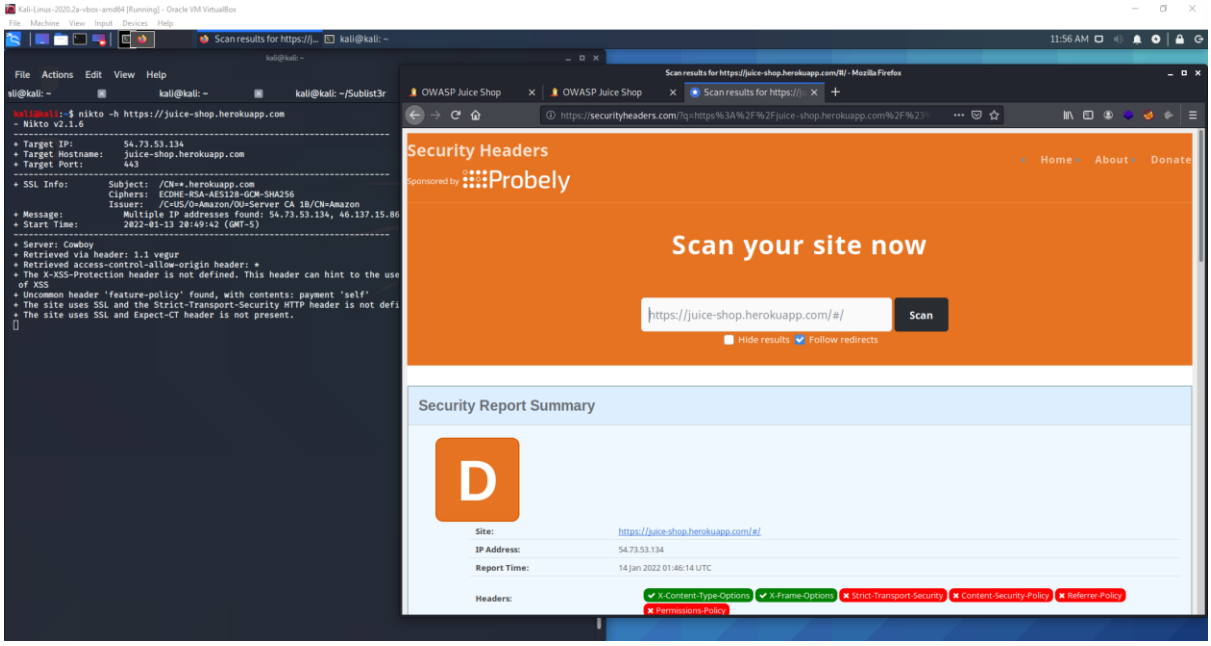


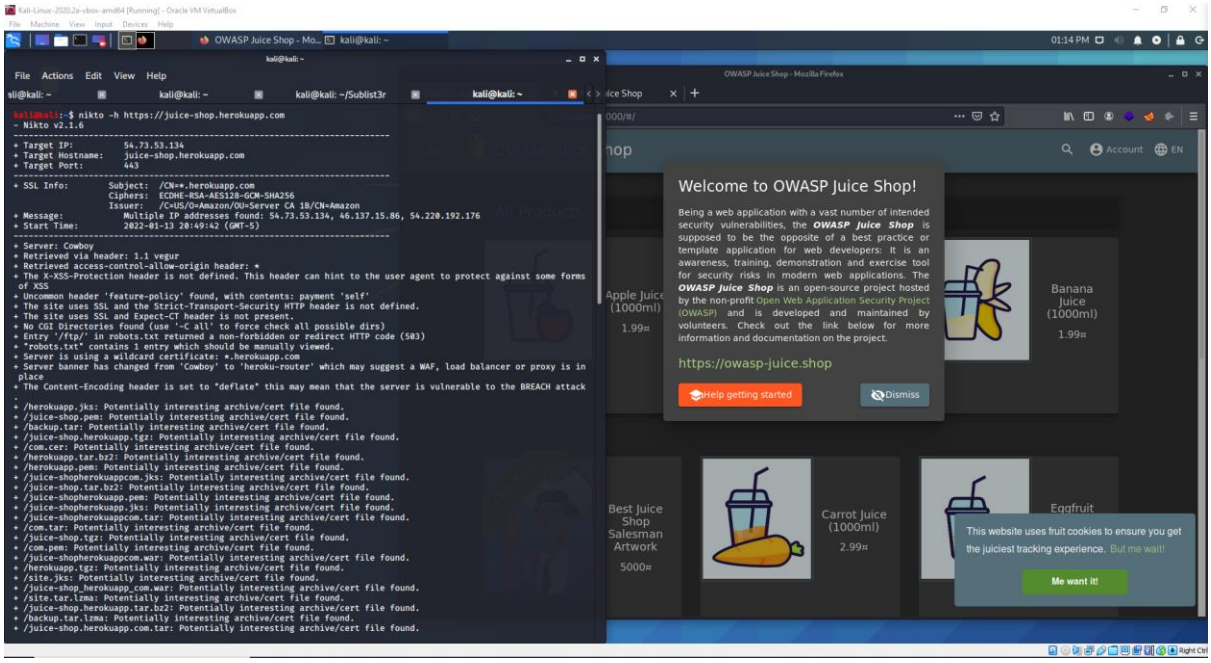
Loud scanning is an ideal way to pen test because if it is not picked up by security systems, it can be included within the report to help fine tune anti-intrusion software.

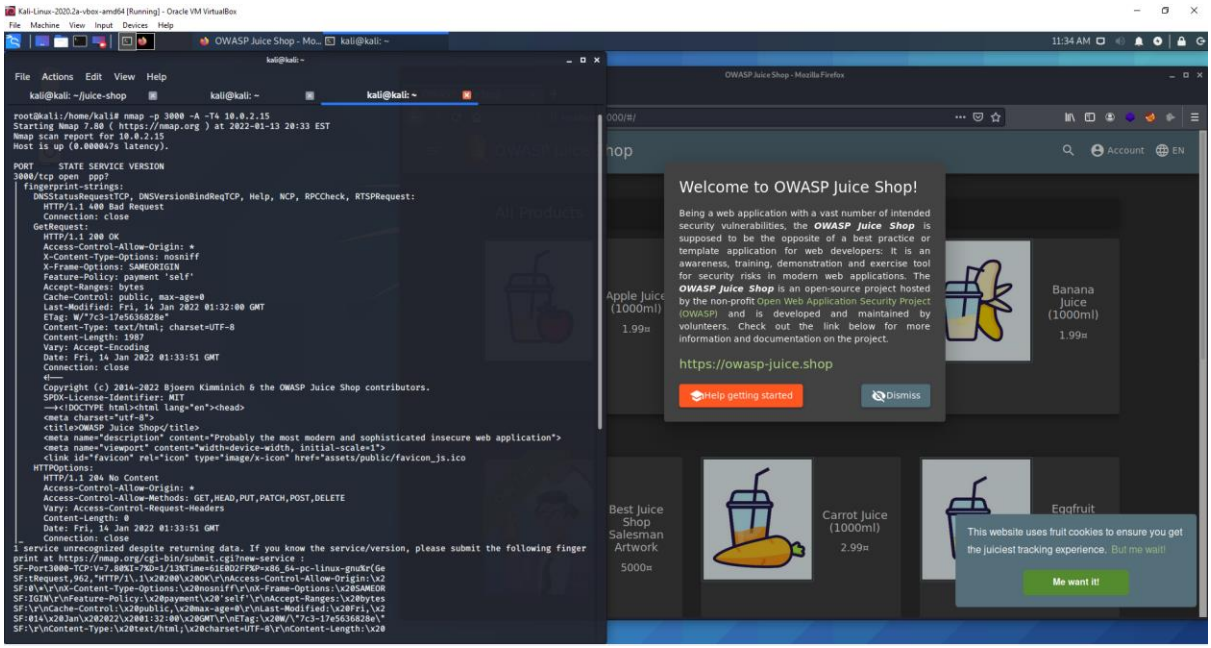
Enumerate Juice Shop

No results found on crt.sh, hunter.io, or builtwith.com

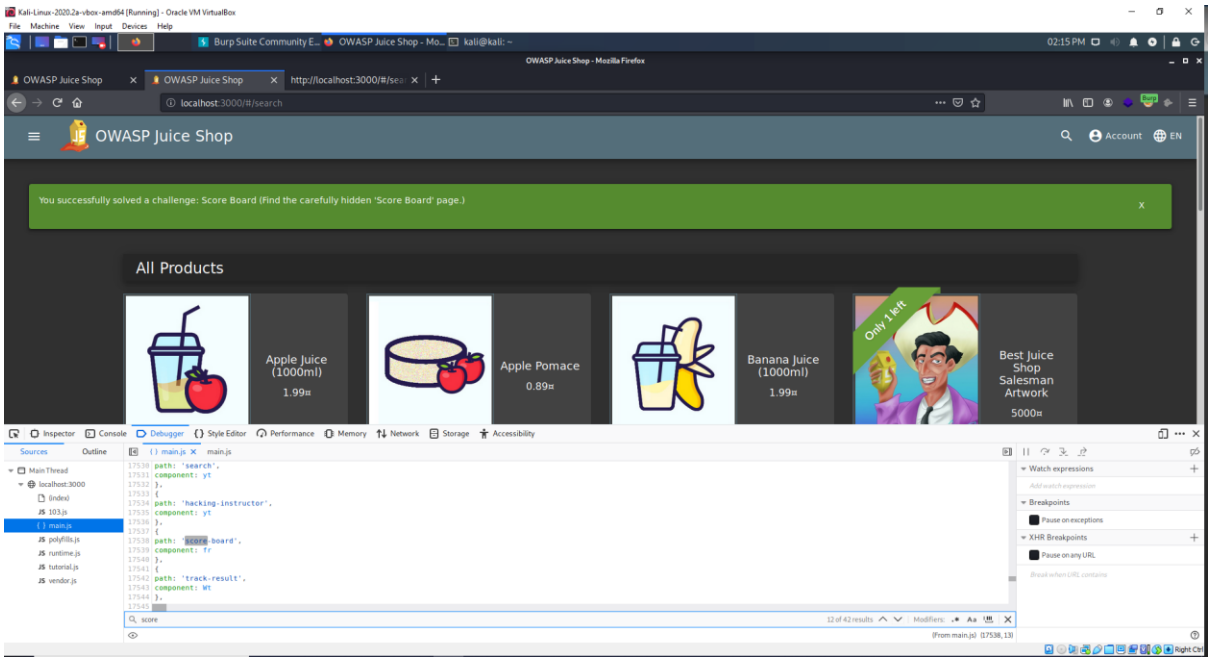


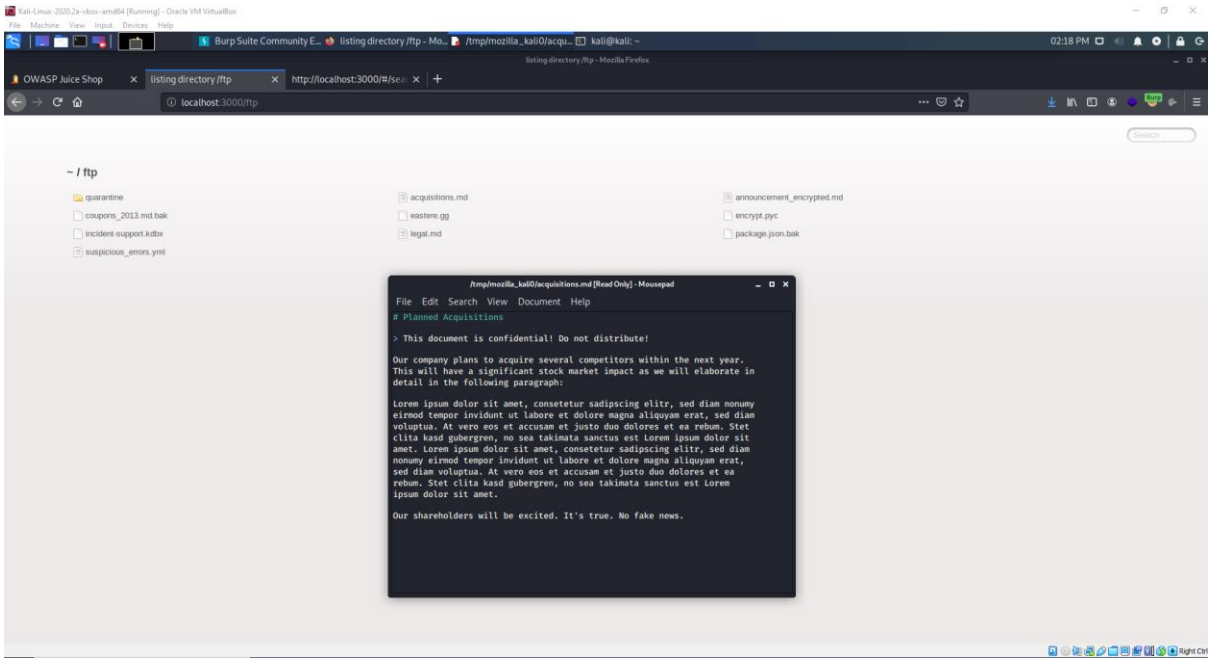

Security Headers
Probely
Scan your site now
https://juice-shop.herokuapp.com/#/
Scan
Security Report Summary
D
Site: https://juice-shop.herokuapp.com/#/
IP Address: 54.73.53.134
Report Time: 14 Jan 2022 01:46:14 UTC
Headers: X-Content-Type-Options X-Frame-Options Strict-Transport-Security Content-Security-Policy Referrer-Policy Permissions-Policy
nikto -h https://juice-shop.herokuapp.com
+ Target IP: 54.73.53.134
+ Target Hostname: juice-shop.herokuapp.com
+ Target Port: 443
+ SSL Info: Subject: /CN=*.herokuapp.com
Ciphers: ECDHE-RSA-AES128-GCM-SHA256
Issuer: /C=US/O=Amazon/OU=Server CA 1B/CN=Amazon
+ Message: Multiple IP addresses found: 54.73.53.134, 46.137.15.86
+ Start Time: 2022-01-13 20:49:42 (GMT-5)
+ Server: Cowboy
+ Retrieved via header: 1.1 vegur
+ Retrieved access-control-allow-origin header: *
+ The X-XSS-Protection header is not defined. This header can hint to the use
of XSS
+ Uncommon header 'feature-policy' found, with contents: payment 'self'
+ The site uses SSL and the Strict-Transport-Security HTTP header is not defi
+ The site uses SSL and Expect-CT header is not present.

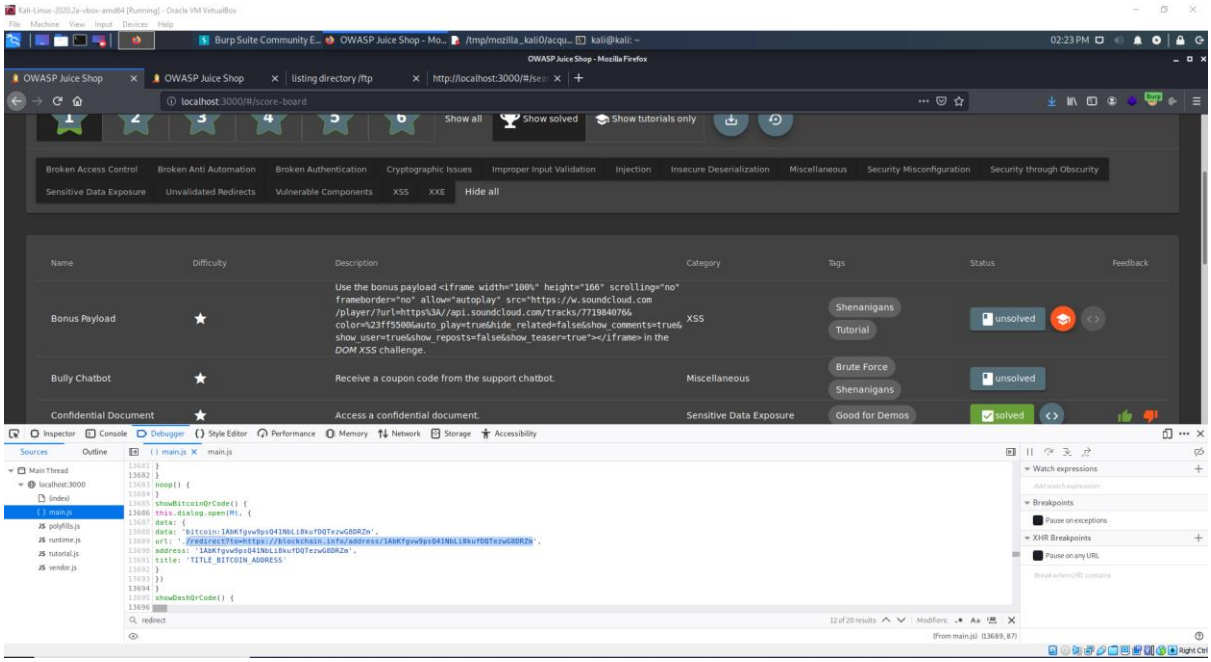

Welcome to OWASP Juice Shop!
Being a web application with a vast number of intended security vulnerabilities, the OWASP Juice Shop is supposed to be the opposite of a best practice or template application for web developers: It is an awareness, training, demonstration and exercise tool for security risks in modern web applications. The OWASP Juice Shop is an open-source project hosted by the non-profit Open Web Application Security Project (OWASP) and is developed and maintained by volunteers. Check out the link below for more information and documentation on the project.
https://owasp-juice.shop
Help getting started
Dismiss
Banana Juice (1000ml)
1.99¤
Carrot Juice (1000ml)
2.99¤
Best Juice Shop Salesman Artwork
5000¤
This website uses fruit cookies to ensure you get the juiciest tracking experience. But me wait!
Me want it!
nikto -h https://juice-shop.herokuapp.com
+ Target IP: 54.73.53.134
+ Target Hostname: juice-shop.herokuapp.com
+ Target Port: 443
+ SSL Info: Subject: /CN=*.herokuapp.com
Ciphers: ECDHE-RSA-AES128-GCM-SHA256
Issuer: /C=US/O=Amazon/OU=Server CA 1B/CN=Amazon
+ Message: Multiple IP addresses found: 54.73.53.134, 46.137.15.86, 54.220.192.176
+ Start Time: 2022-01-13 20:49:42 (GMT-5)
+ Server: Cowboy
+ Retrieved via header: 1.1 vegur
+ Retrieved access-control-allow-origin header: *
+ The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS
+ Uncommon header 'feature-policy' found, with contents: payment 'self'
+ The site uses SSL and the Strict-Transport-Security HTTP header is not defined.
+ The site uses SSL and Expect-CT header is not present.
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Entry '/ftp/' in robots.txt returned a non-forbidden or redirect HTTP code (503)
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Server is using a wildcard certificate: *.herokuapp.com
+ Server banner has changed from 'Cowboy' to 'heroku-router' which may suggest a WAF, load balancer or proxy is in place
+ The Content-Encoding header is set to "deflate" this may mean that the server is vulnerable to the BREACH attack
+ /herokuapp.jks: Potentially interesting archive/cert file found.
+ /juice-shop.pem: Potentially interesting archive/cert file found.
+ /backup.tar: Potentially interesting archive/cert file found.
+ /juice-shop.herokuapp.tgz: Potentially interesting archive/cert file found.
+ /com.cer: Potentially interesting archive/cert file found.
+ /herokuapp.tar.bz2: Potentially interesting archive/cert file found.
+ /herokuapp.pem: Potentially interesting archive/cert file found.
+ /juice-shopherokuappcom.jks: Potentially interesting archive/cert file found.
+ /juice-shop.tar.bz2: Potentially interesting archive/cert file found.
+ /juice-shopherokuapp.pem: Potentially interesting archive/cert file found.
+ /juice-shopherokuapp.jks: Potentially interesting archive/cert file found.
+ /juice-shopherokuappcom.tar: Potentially interesting archive/cert file found.
+ /com.tar: Potentially interesting archive/cert file found.
+ /juice-shop.tgz: Potentially interesting archive/cert file found.
+ /com.pem: Potentially interesting archive/cert file found.
+ /juice-shopherokuappcom.war: Potentially interesting archive/cert file found.
+ /herokuapp.tgz: Potentially interesting archive/cert file found.
+ /site.jks: Potentially interesting archive/cert file found.
+ /juice-shop_herokuapp_com.war: Potentially interesting archive/cert file found.
+ /site.tar.lzma: Potentially interesting archive/cert file found.
+ /juice-shop.herokuapp.tar.bz2: Potentially interesting archive/cert file found.
+ /backup.tar.lzma: Potentially interesting archive/cert file found.
+ /juice-shop.herokuapp.com.tar: Potentially interesting archive/cert file found.



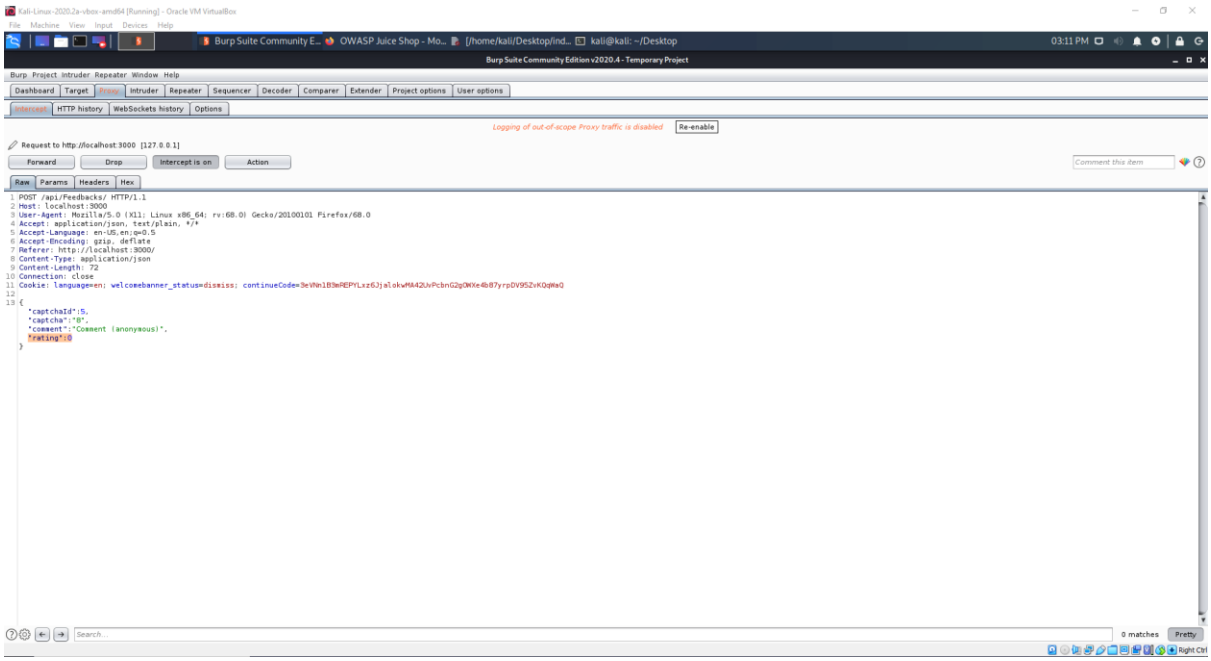
Week 2: Enumeration and XSS



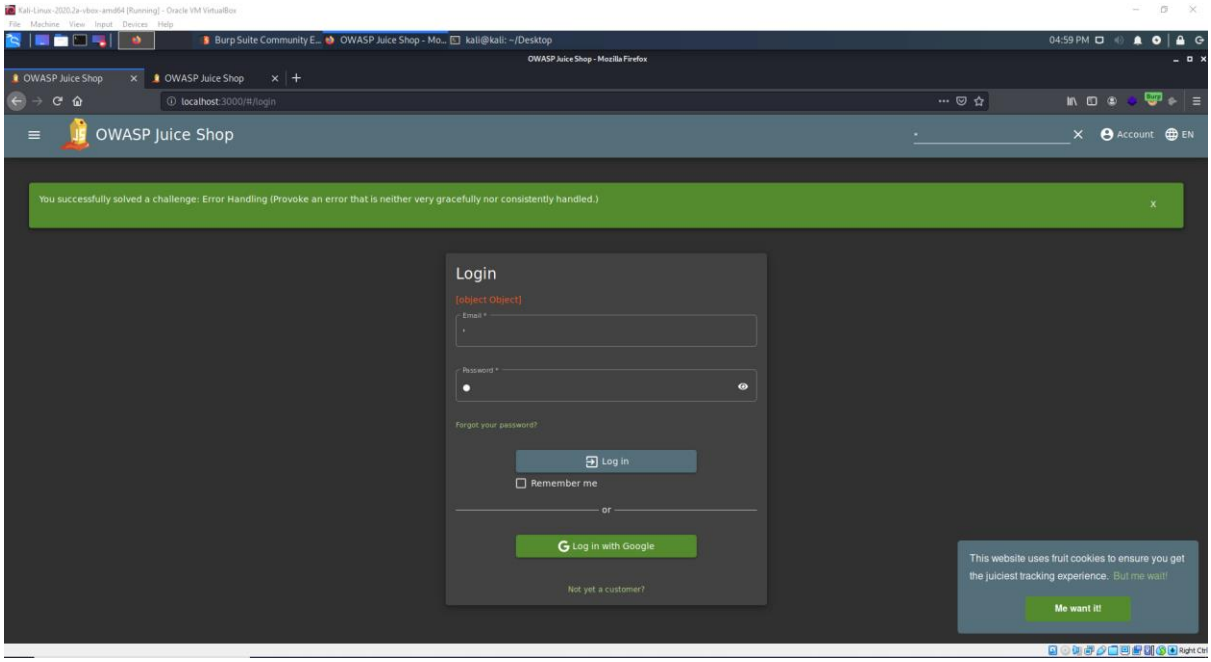




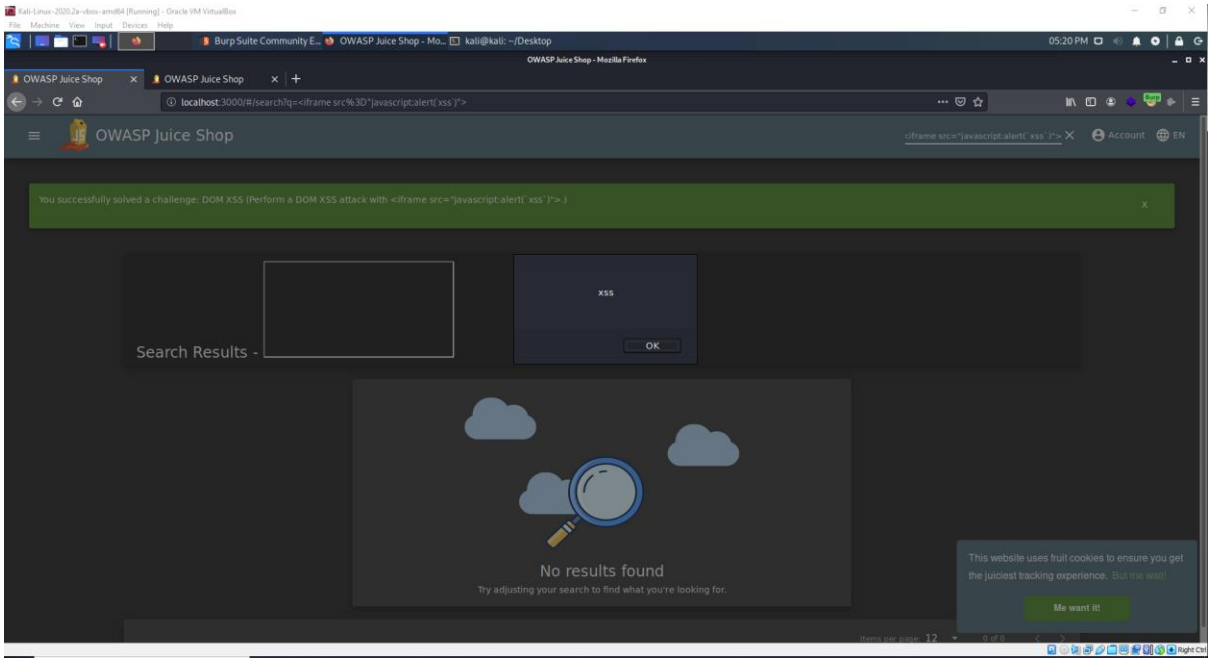
By filling in the user creation form, but returning to the first instance to change the password, Juice Shop does not register that you have two different passwords written in the form and completes the challenge; Repetitive Registration: Follow the DRY principle while registering a user.

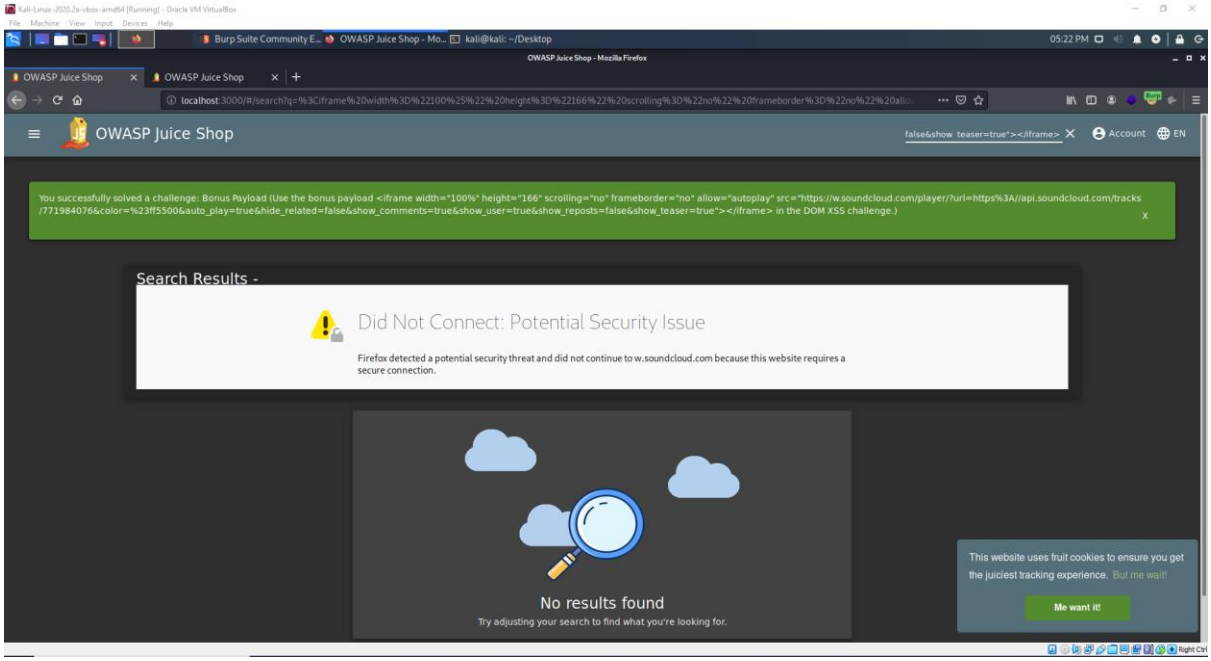


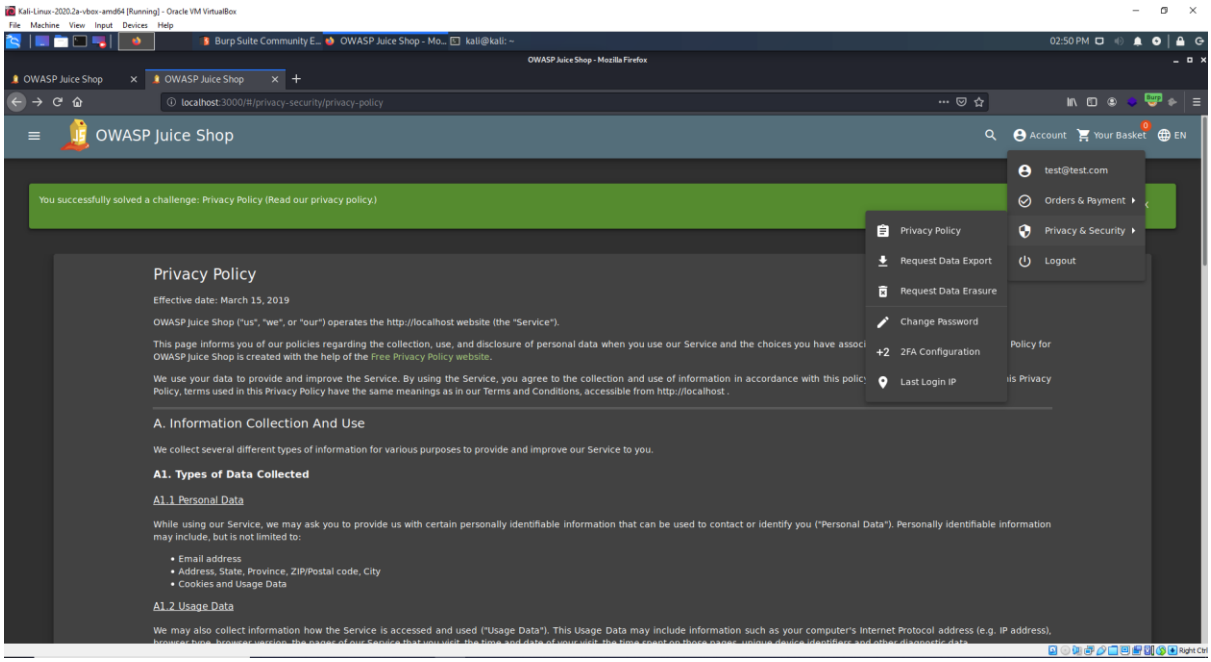
I have an updated version of Juice-Shop installed on my machine so instead of manipulating the source code through inspect element, I went and edited the post method through burpsuite since they changed the 0 rating challenge from a form input to a slider.

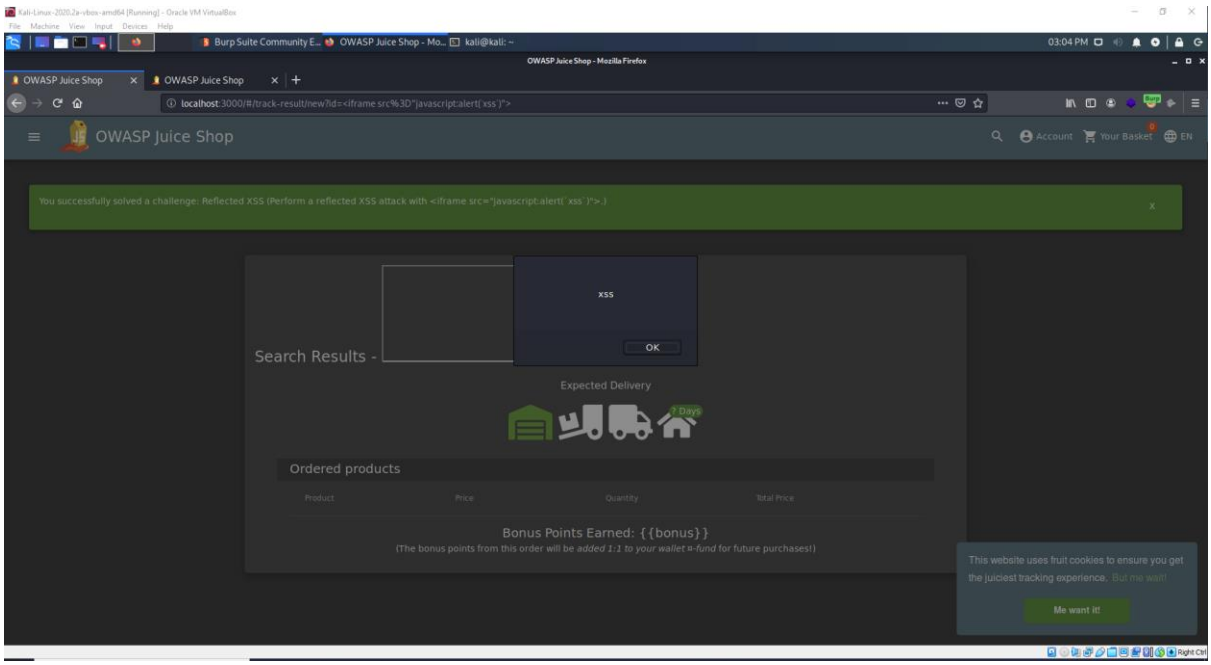


Using an apostrophe can fuzz to determine whether SQL injection is possible.


OWASP Juice Shop
You successfully solved a challenge: DOM XSS (Perform a DOM XSS attack with <iframe src="javascript:alert(`xss`)">.)
Search Results -
xss
OK
No results found
Try adjusting your search to find what you're looking for.
This website uses fruit cookies to ensure you get the juiciest tracking experience. But me wait!
Me want it!


OWASP Juice Shop
You successfully solved a challenge: Bonus Payload (Use the bonus payload <iframe width="100%" height="166" scrolling="no" frameborder="no" allow="autoplay" src="https://w.soundcloud.com/player/?url=https%3A//api.soundcloud.com/tracks/771984076&color=%23ff5500&auto_play=true&hide_related=false&show_comments=true&show_user=true&show_reposts=false&show_teaser=true"></iframe> in the DOM XSS challenge.)
Search Results -
Did Not Connect: Potential Security Issue
Firefox detected a potential security threat and did not continue to w.soundcloud.com because this website requires a secure connection.
No results found
Try adjusting your search to find what you're looking for.
This website uses fruit cookies to ensure you get the juiciest tracking experience. But me wait!
Me want it!





XSS Prevention

Encoding

<script>

< == <

<script>

Filtering

Instead of replacing the script, it will remove them.

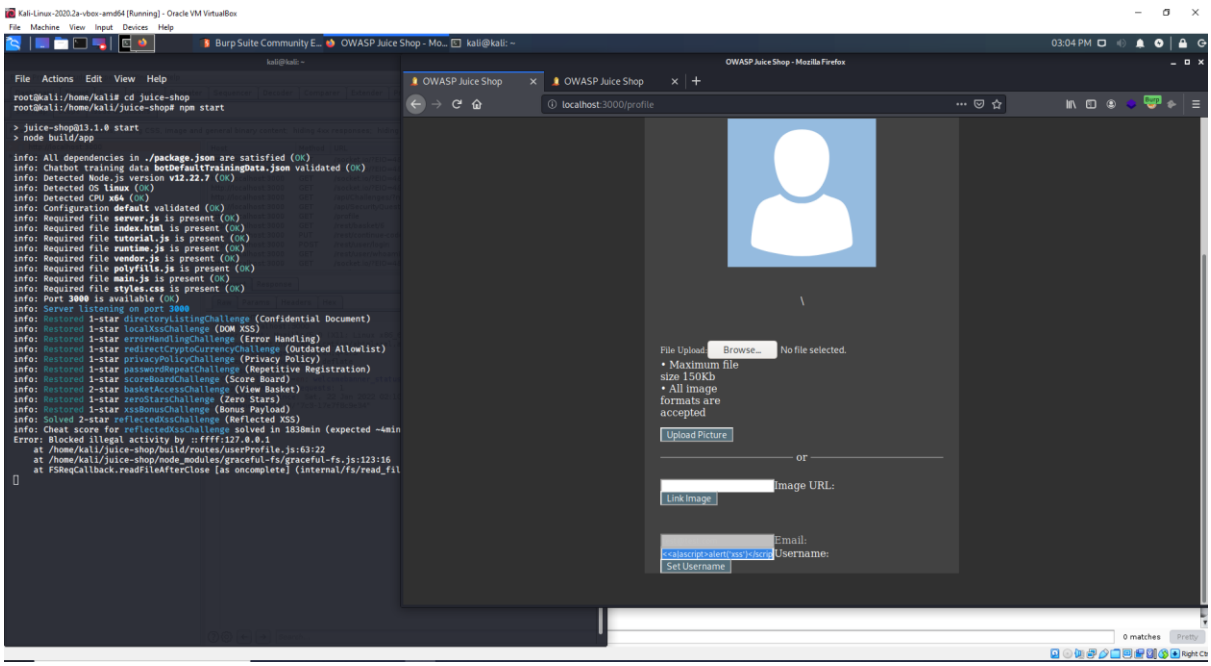
Validation

Comparison against a list or payloads. Can completely remove script tags.

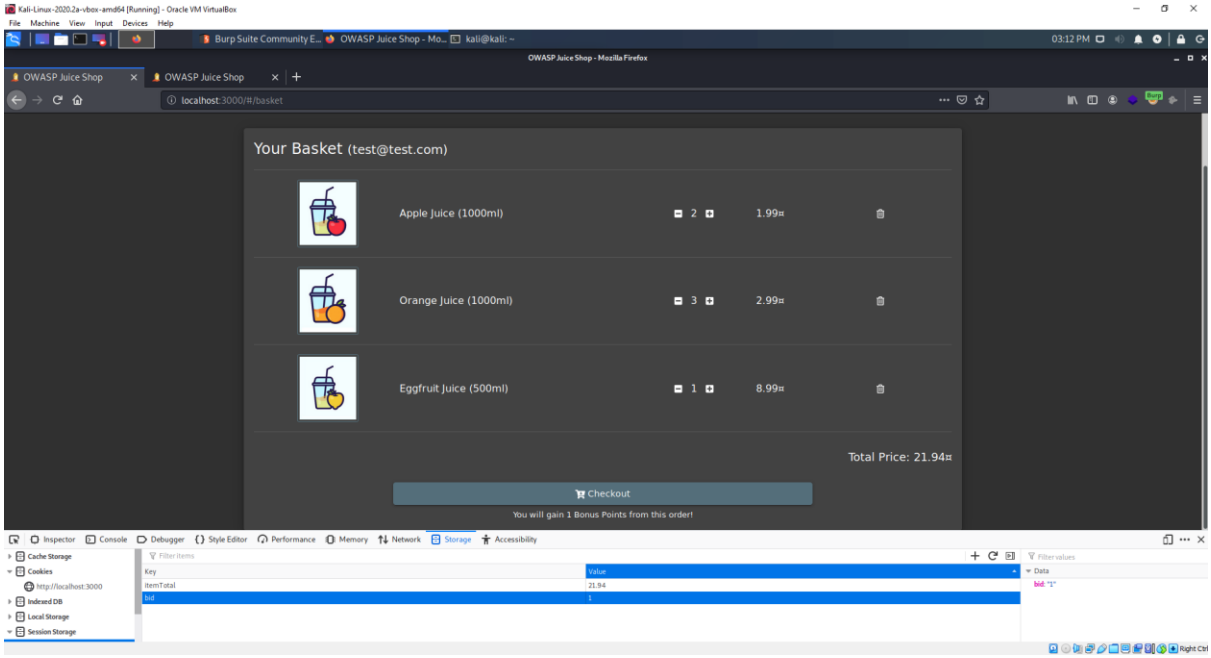
Sanitisation

A mixture of all the previous methods of control.

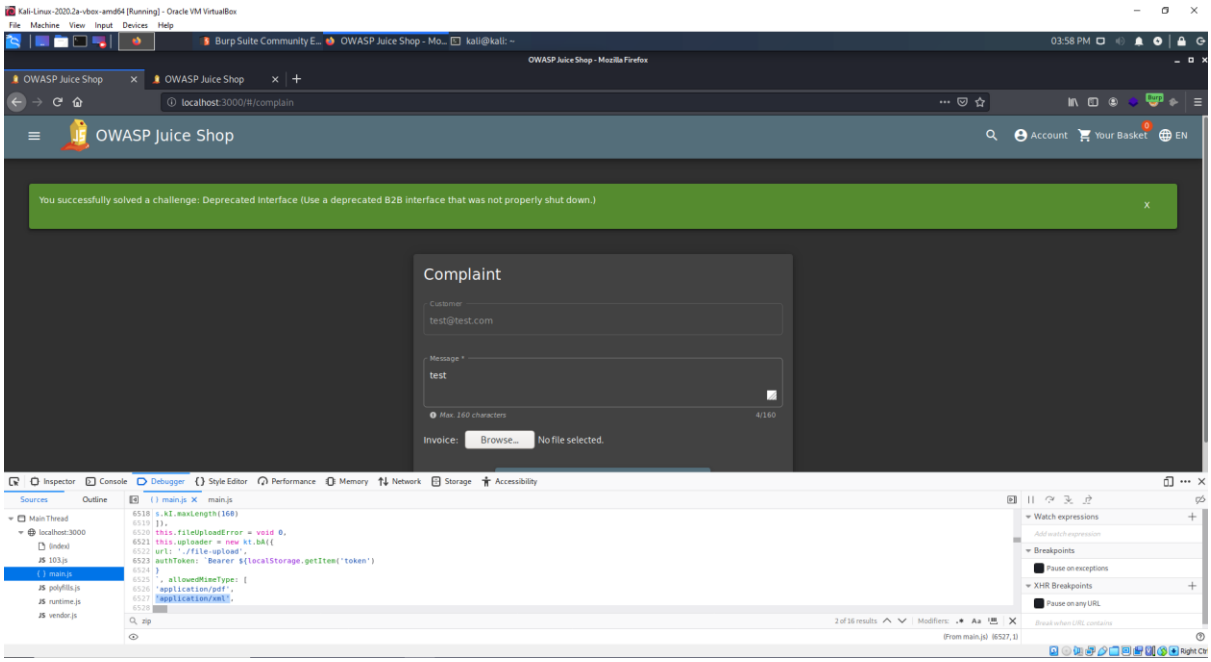
Week 3: XSS, SQL Injection and Broken Access Control



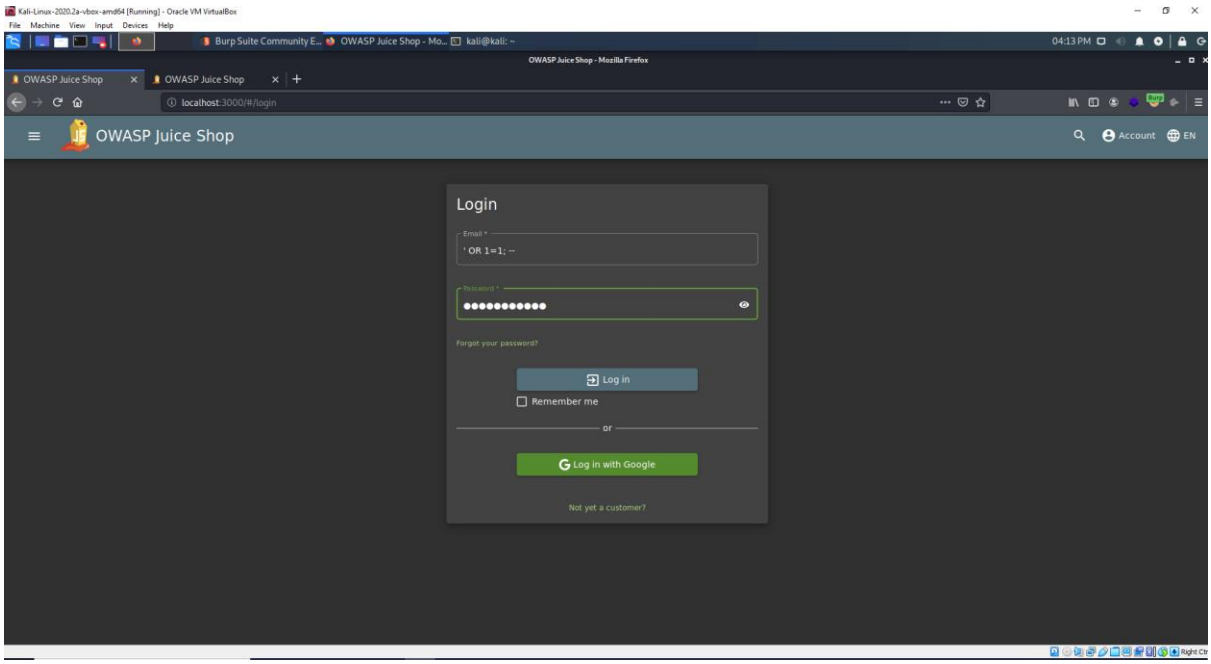
Can use a bitwise operation to bypass some sanitisation ie. a|a, a^a, a&a



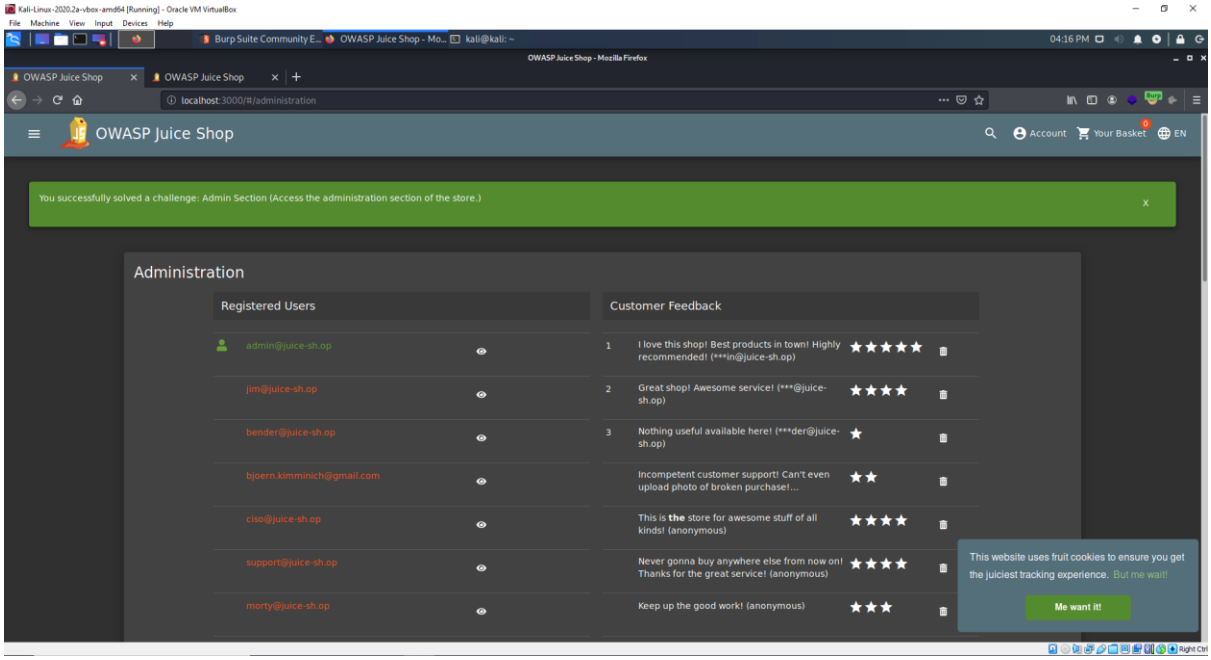
Changing the buyer id (bid) in the session storage can allow you to view another customer's cart



Application allows for the upload of xml files, which can be used for xxe attacks

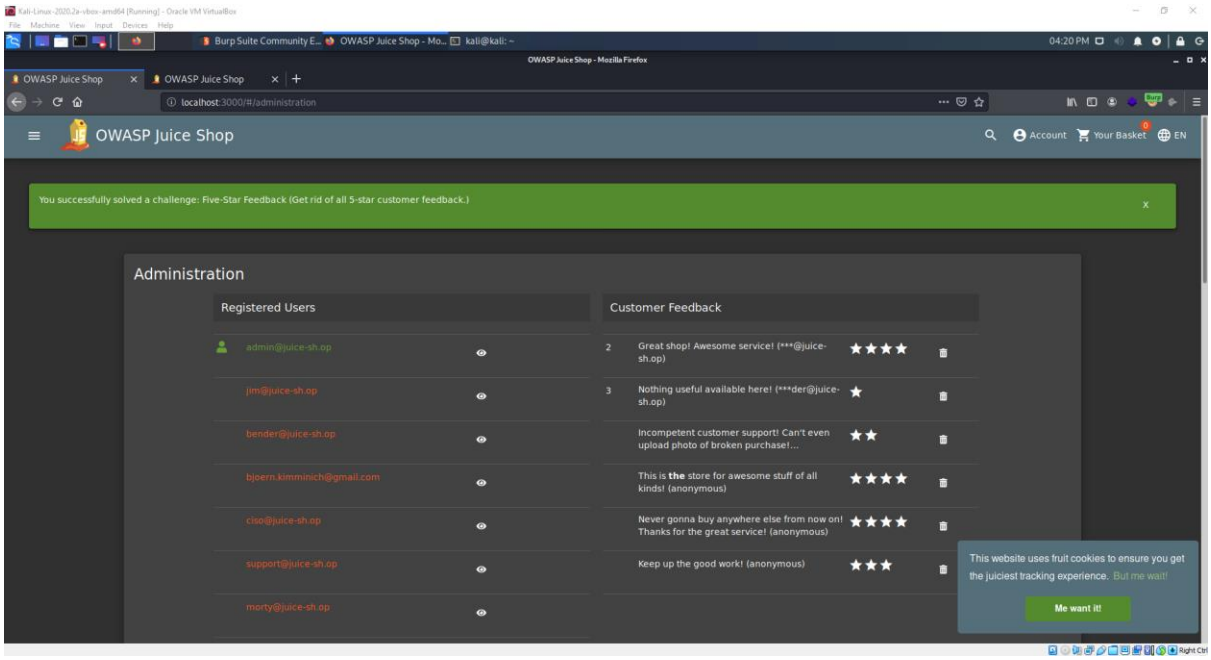


SQL injection to log in as the first user in the database which happens to be the admin account.

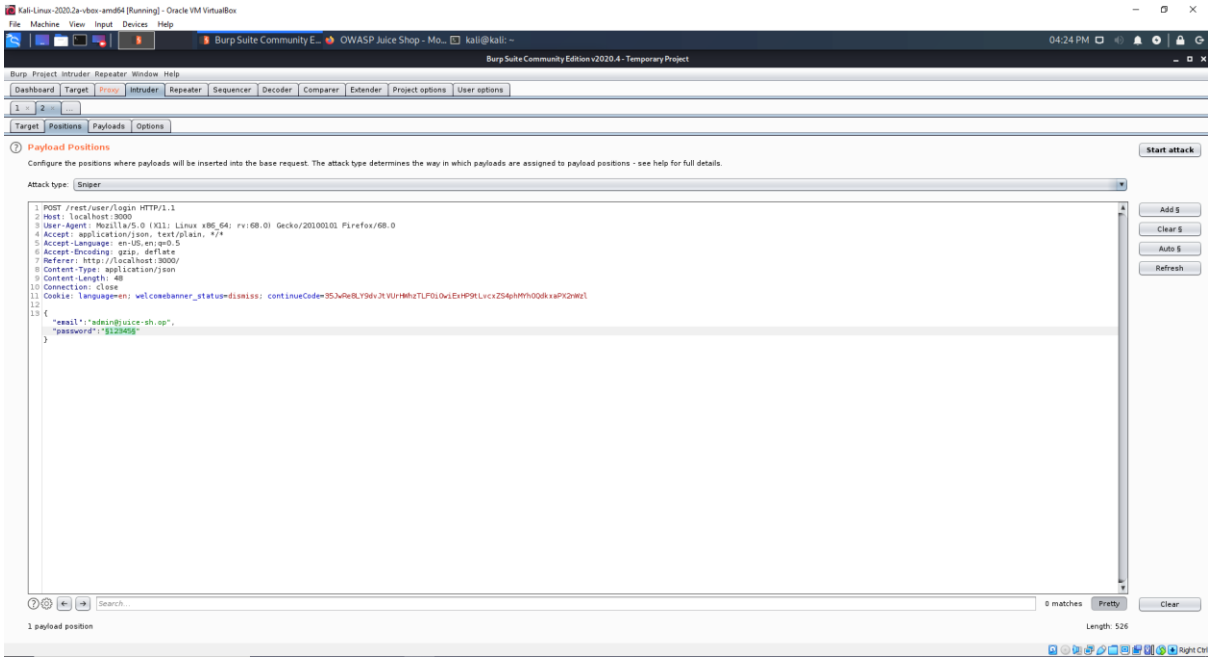


Admin page found through url /administration

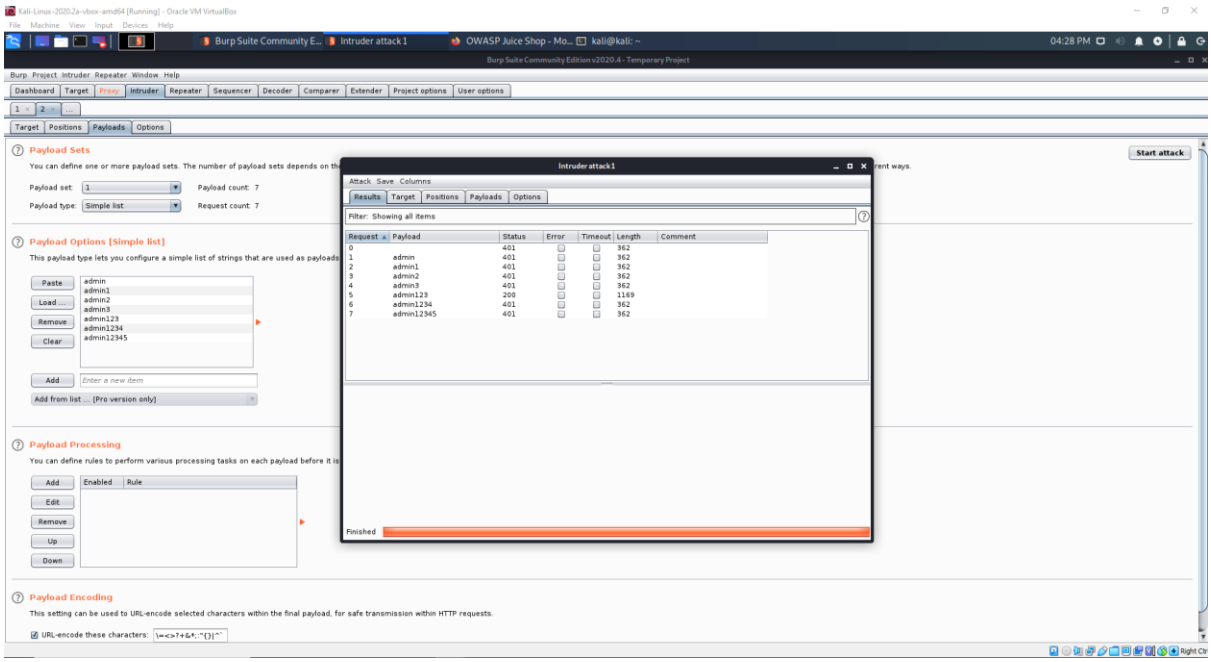
Can also be found through debugger searching for admin pages.



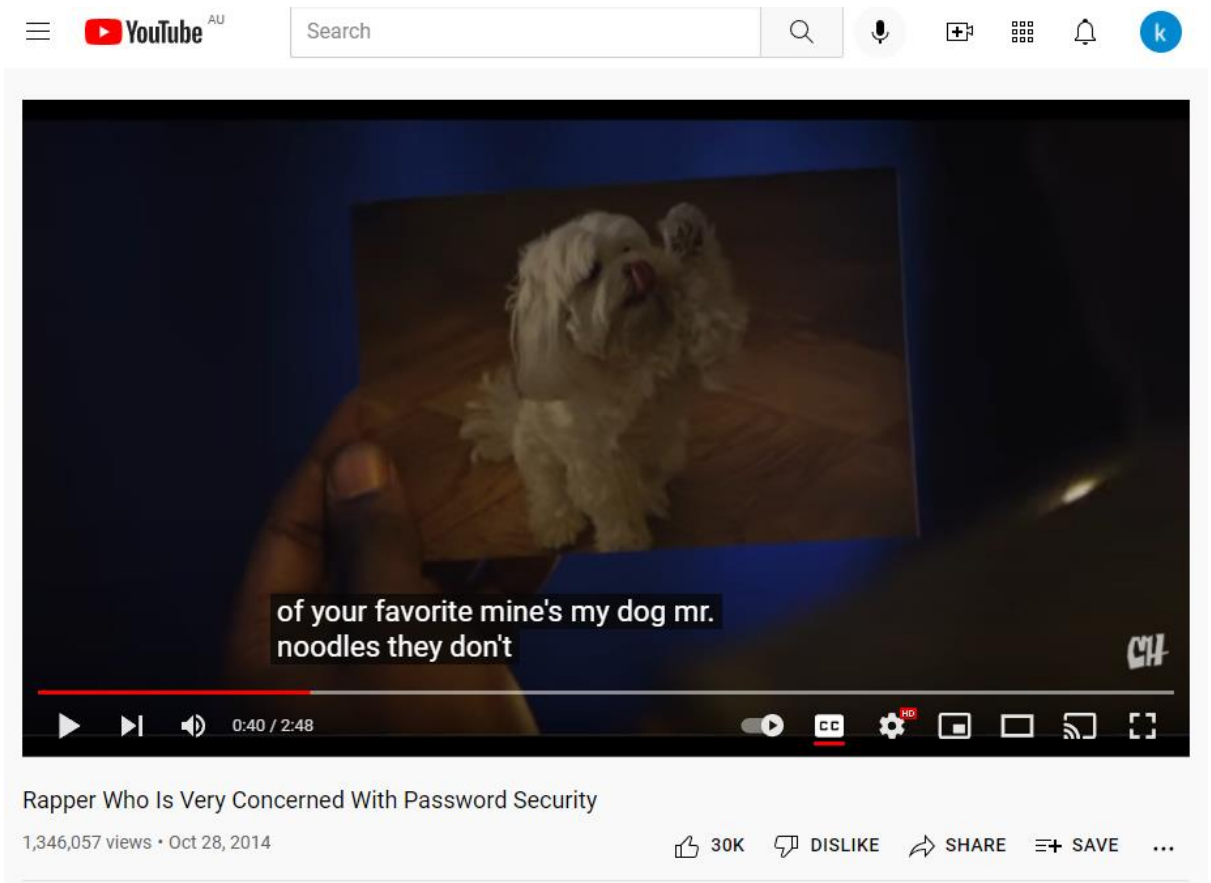
Can use admin privileges to remove 5 star reviews.



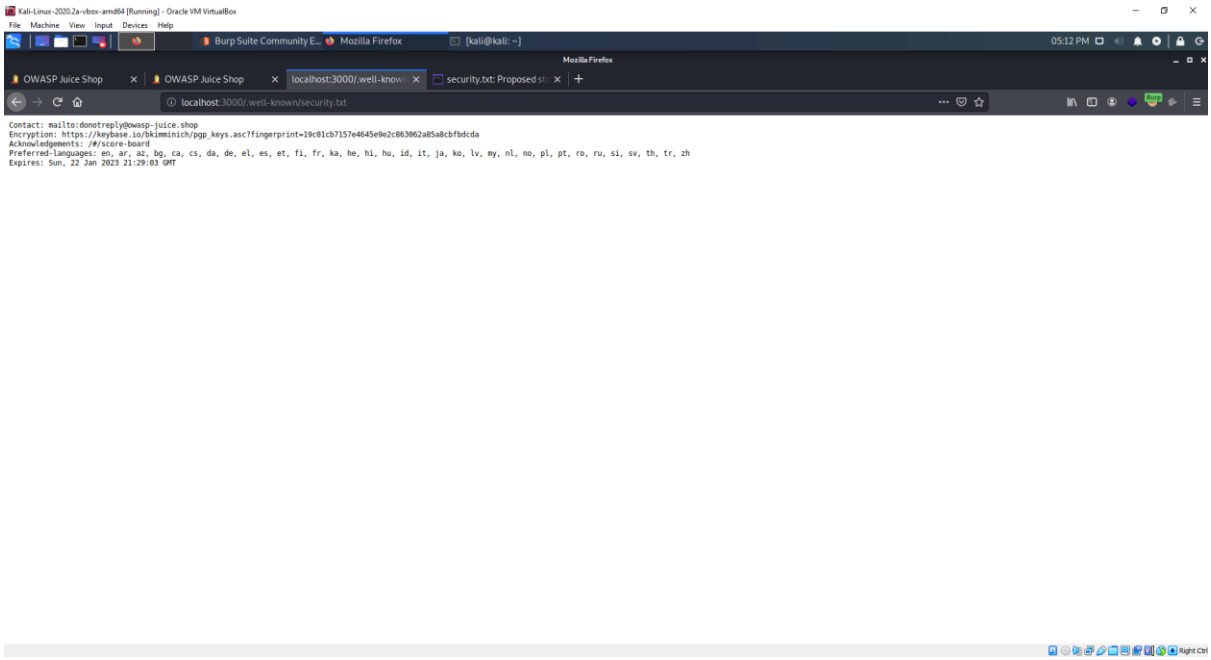
Using intruder to bruteforce the admin password



Check status codes and length for outliers. Also check responses for “Invalid email OR password” for enumeration in professional tests. This is good security practice. Can also grep for invalid responses.

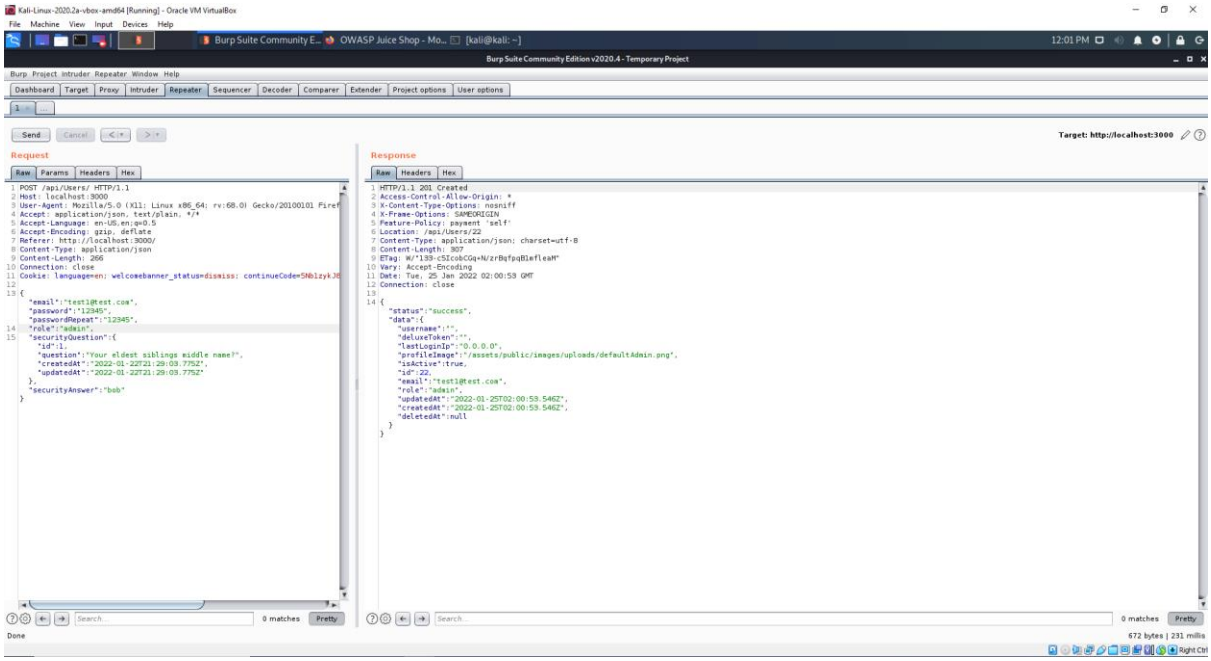


Username: mc.safesearch@juice-sh.op Password: Mr. N00dles

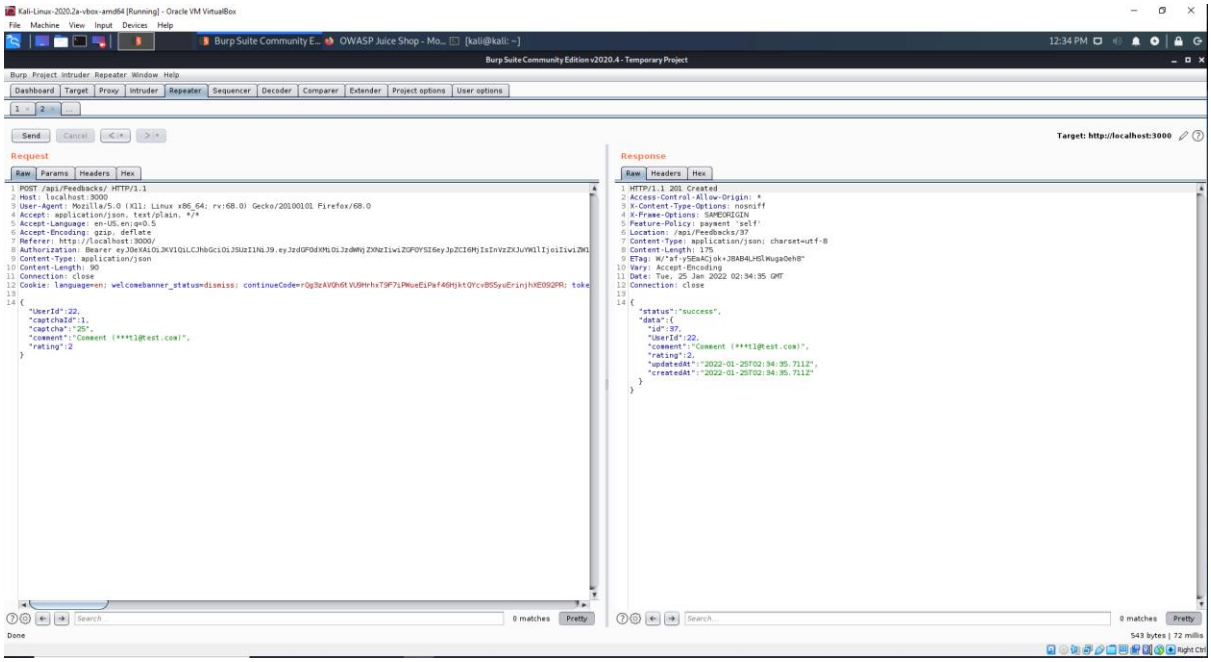


Following the instructions at https://securitytxt.org/

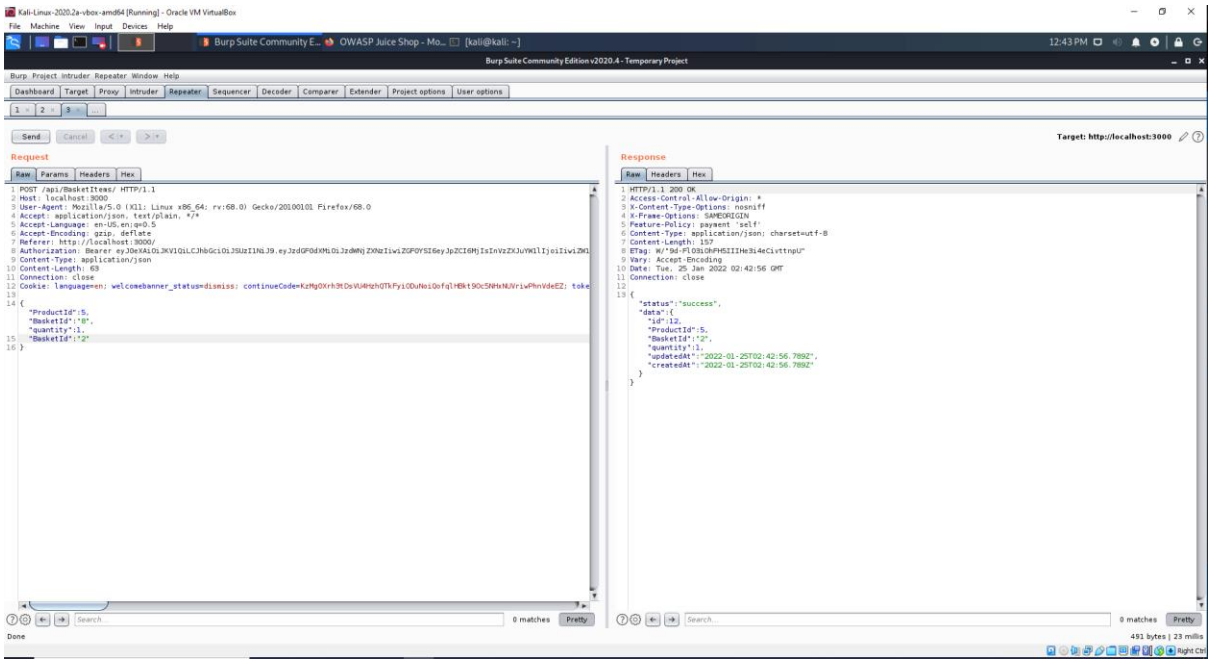
Week 4: XXE, Input Validation, Broken Access Control and More XSS



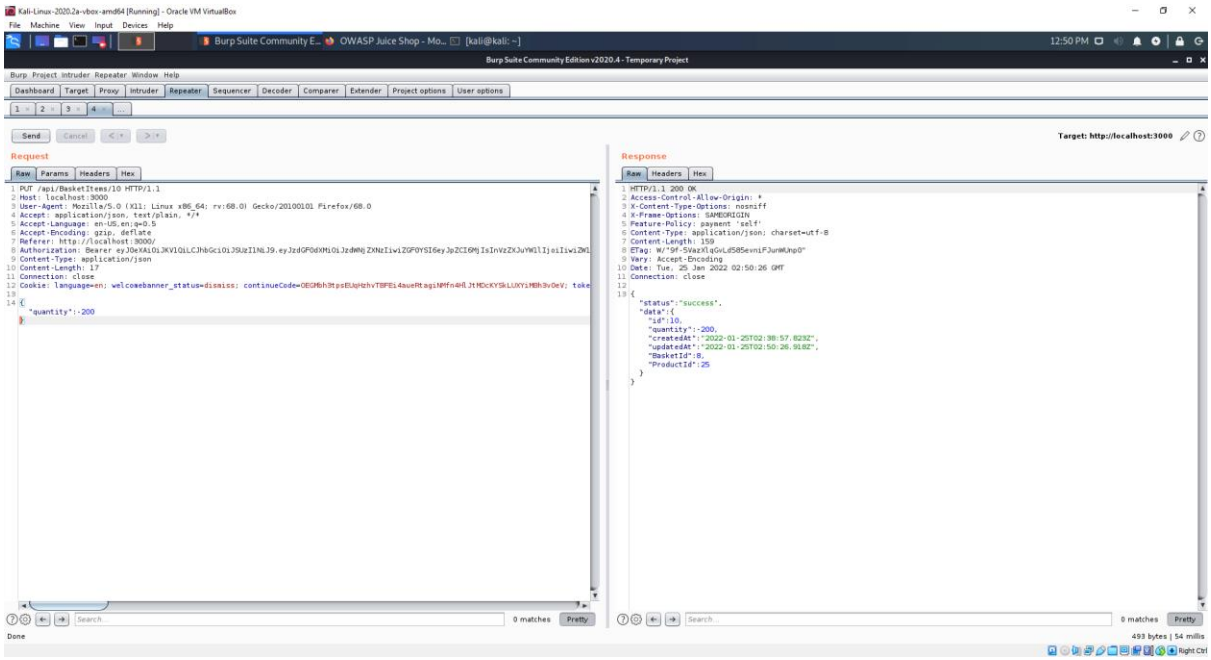
Adding “role” : “admin” in the repeater will give the new user admin privileges.



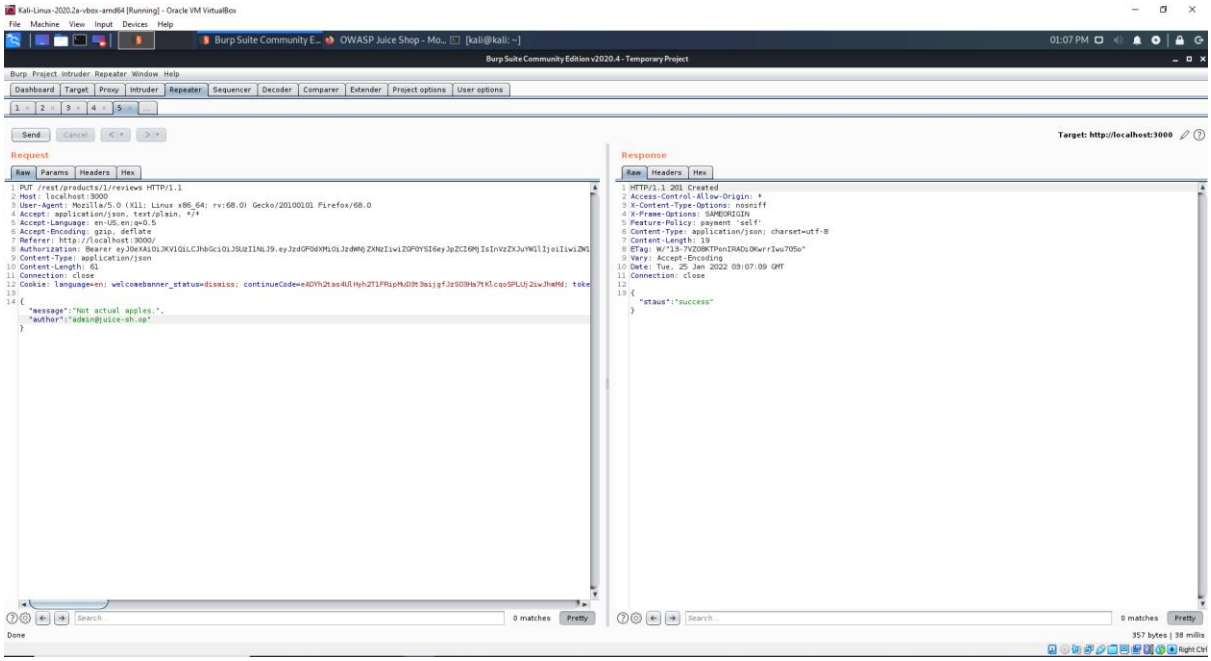
Can bypass the captcha using the correct parameters in repeater as each captcha has a captcha ID tied to it.



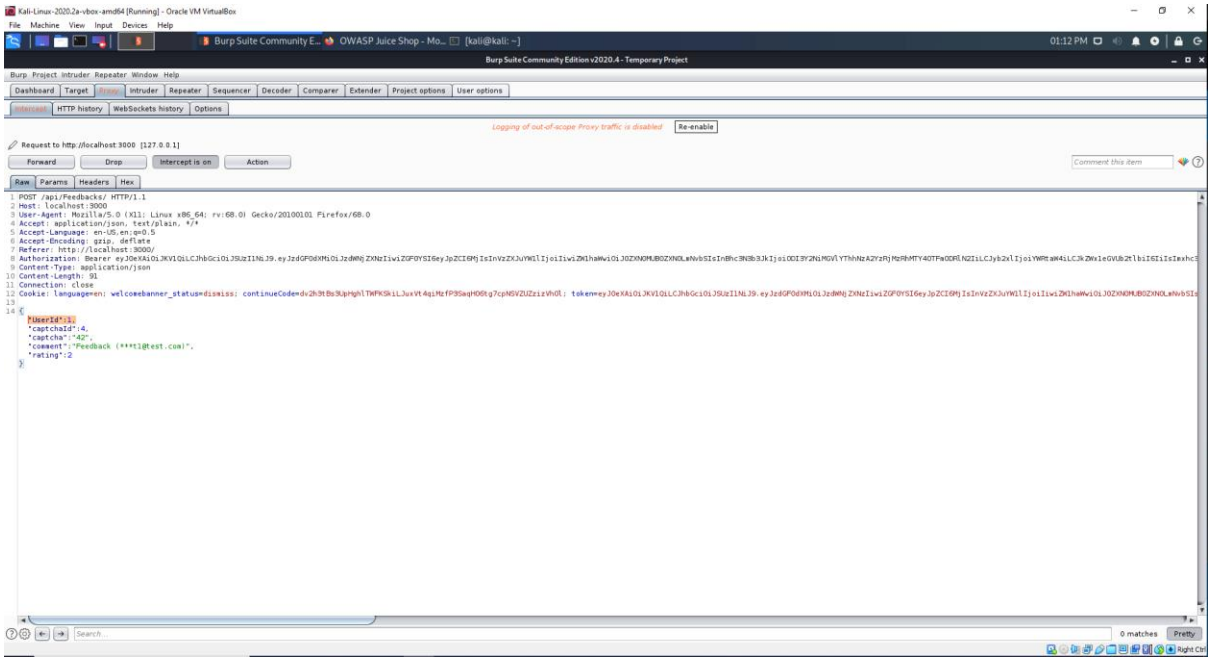
Adding an additional BasketId in repeater can allow you to add another item to another user's basket.



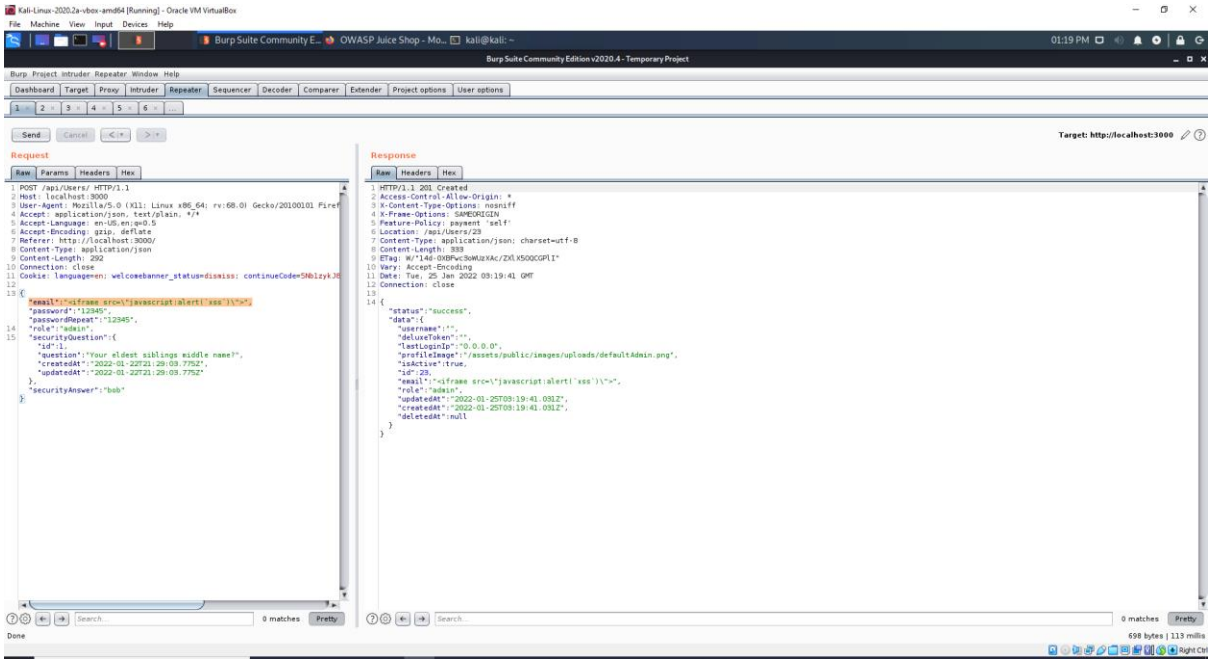
Can manipulate the quantity through repeater to give a negative value that will instead pay the user.



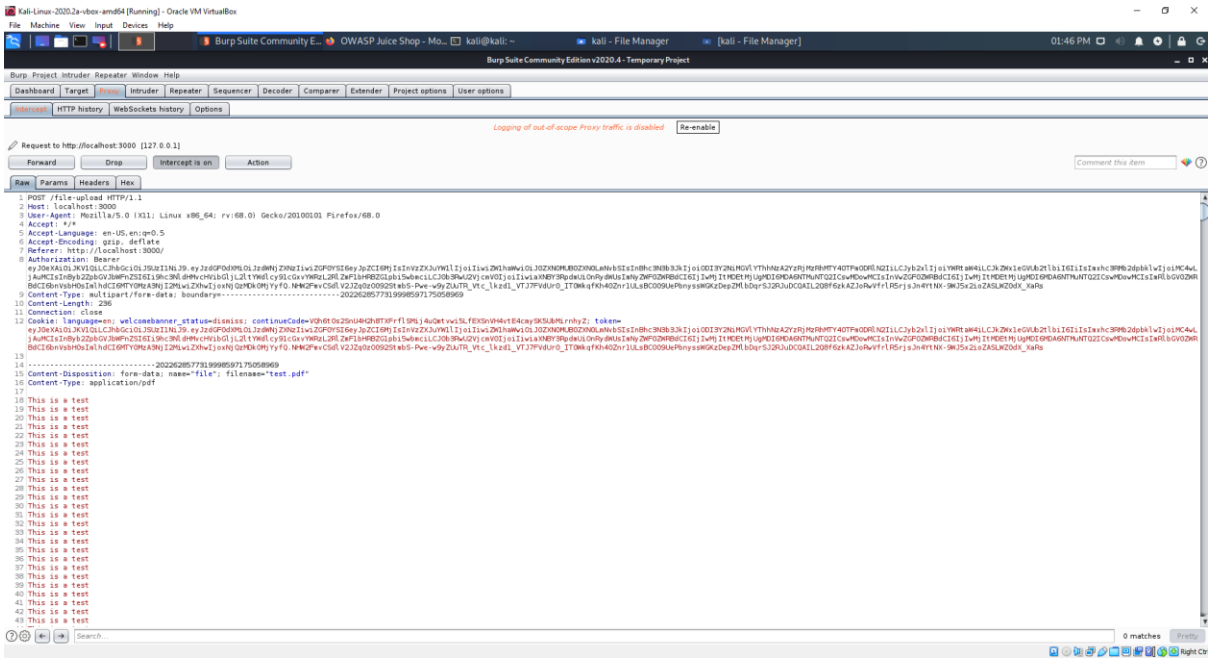
Adding a comment has an author parameter so you can change it to post as whoever you like.



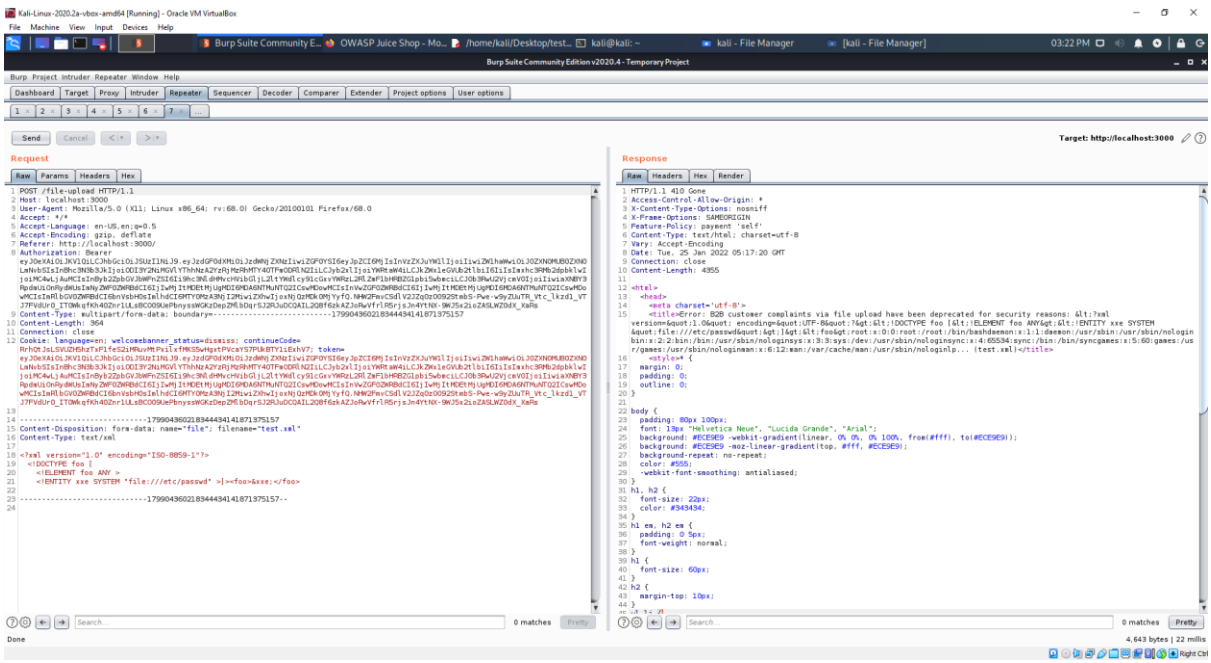
You can alter the UserId parameter to post as someone else.



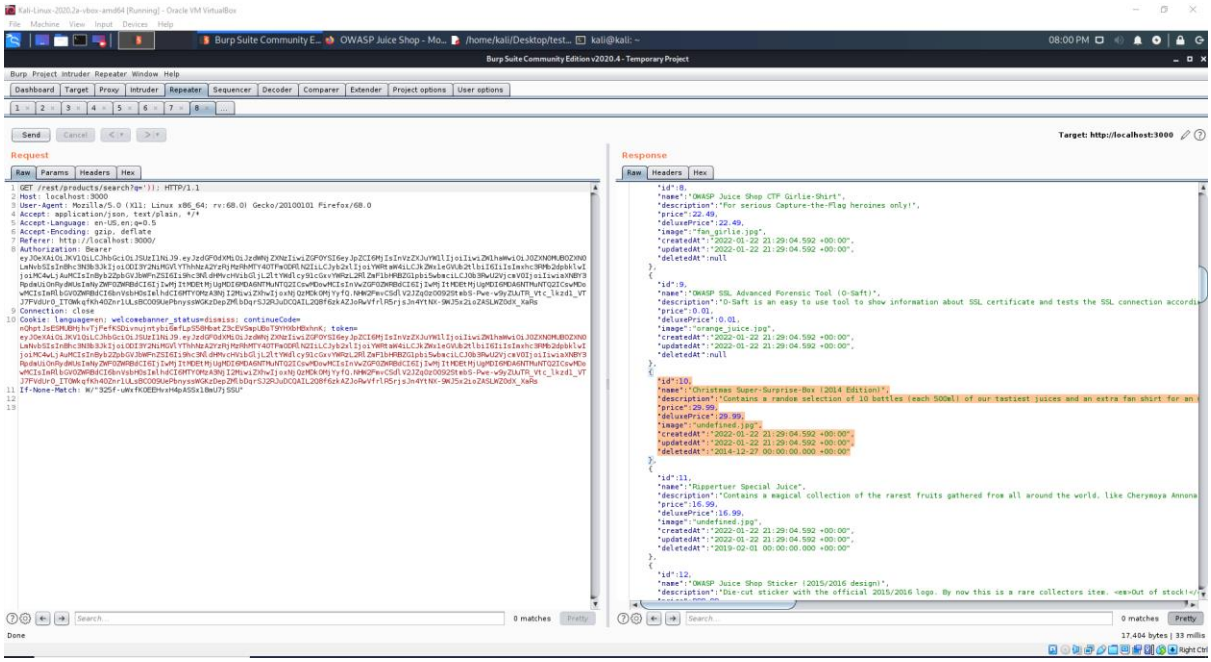
Can inject XSS through the API with repeater.



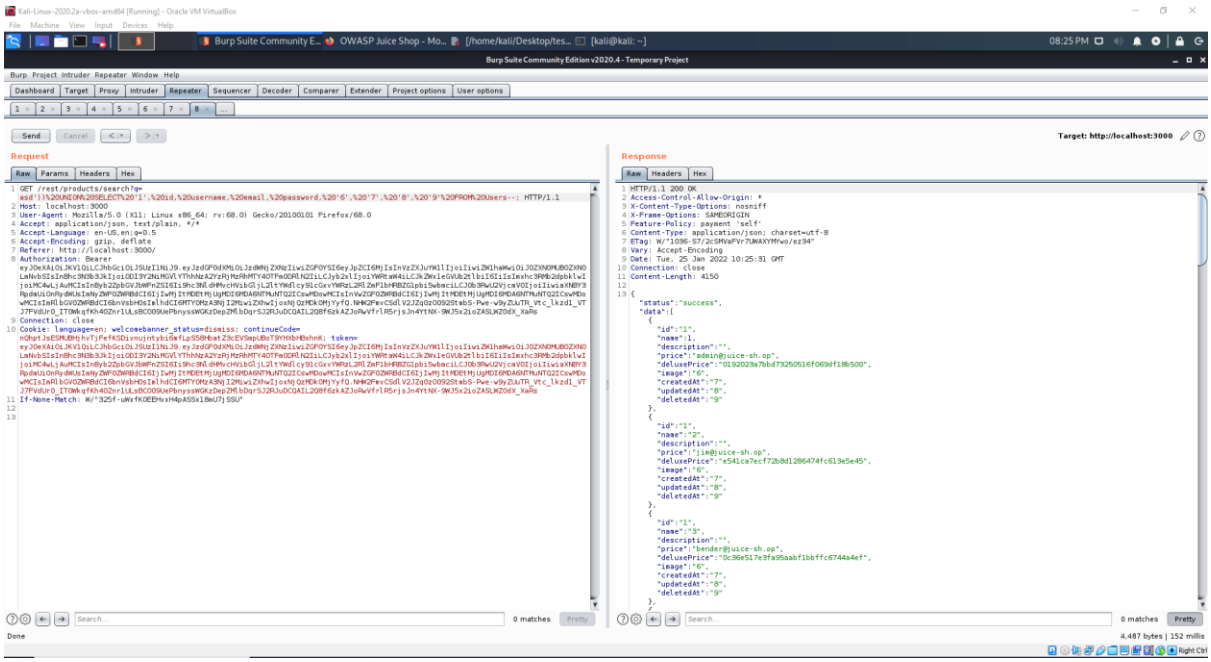
You can add onto the file you upload through intercept to exceed the file size limit.



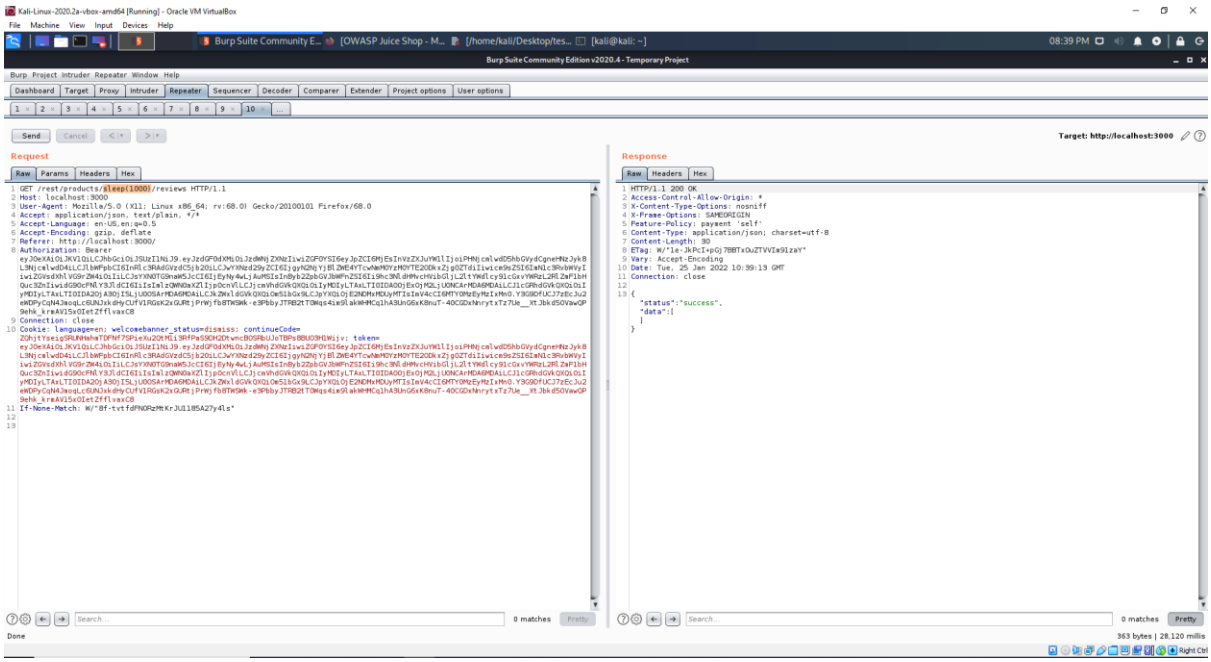
Using XXE to show the passwd file. Prevented by disabling DTD’s.



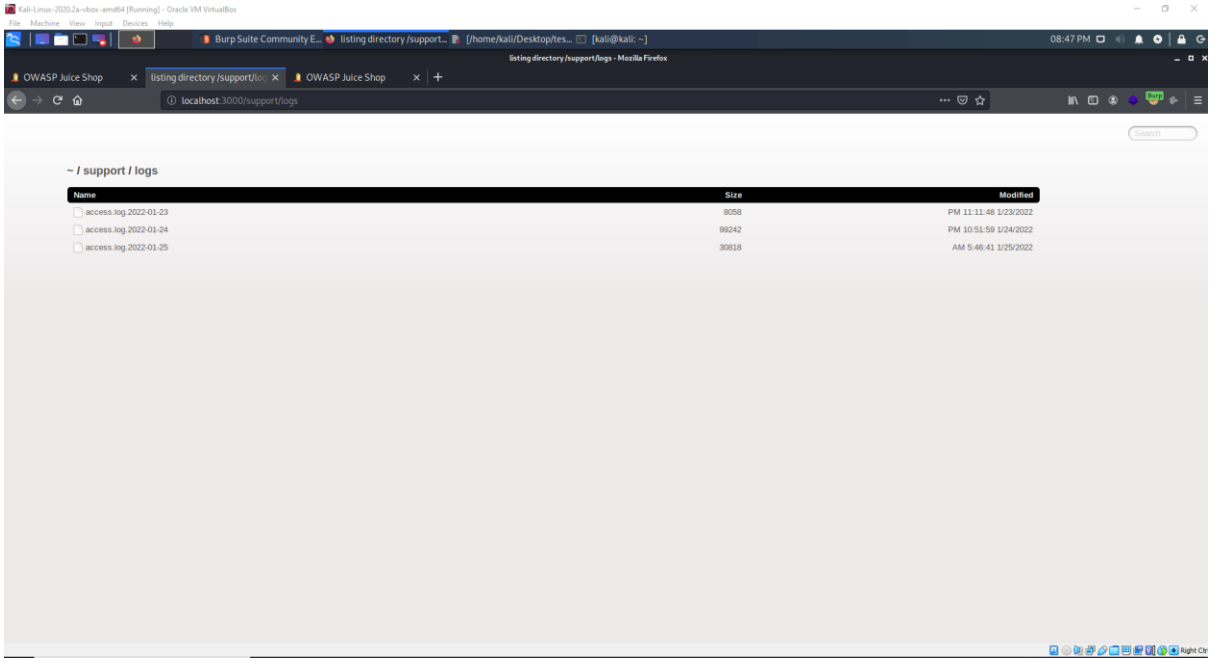
Using SQL injection with ‘)); you can view the items in the database and find which items have been deleted and their time of deletion to use the item ID and add it to the basket through intercept.



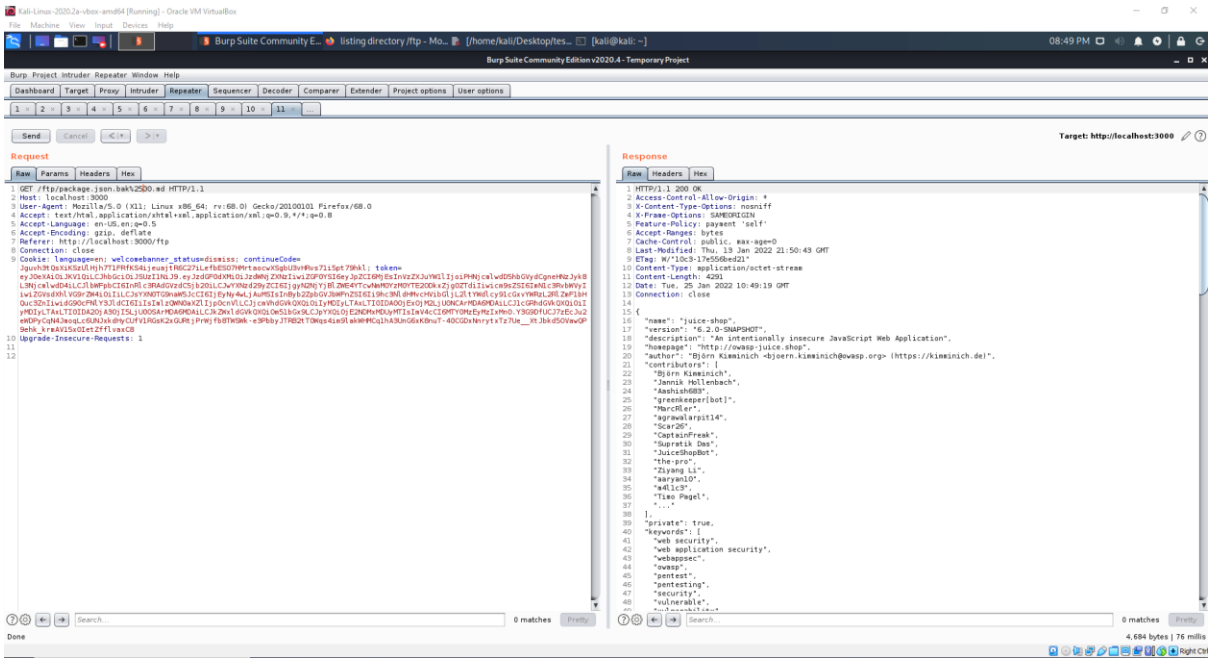
Using SQL Injection to select information from the users table.



Using sleep can be a good indication of SQL injection when performing blind injections.



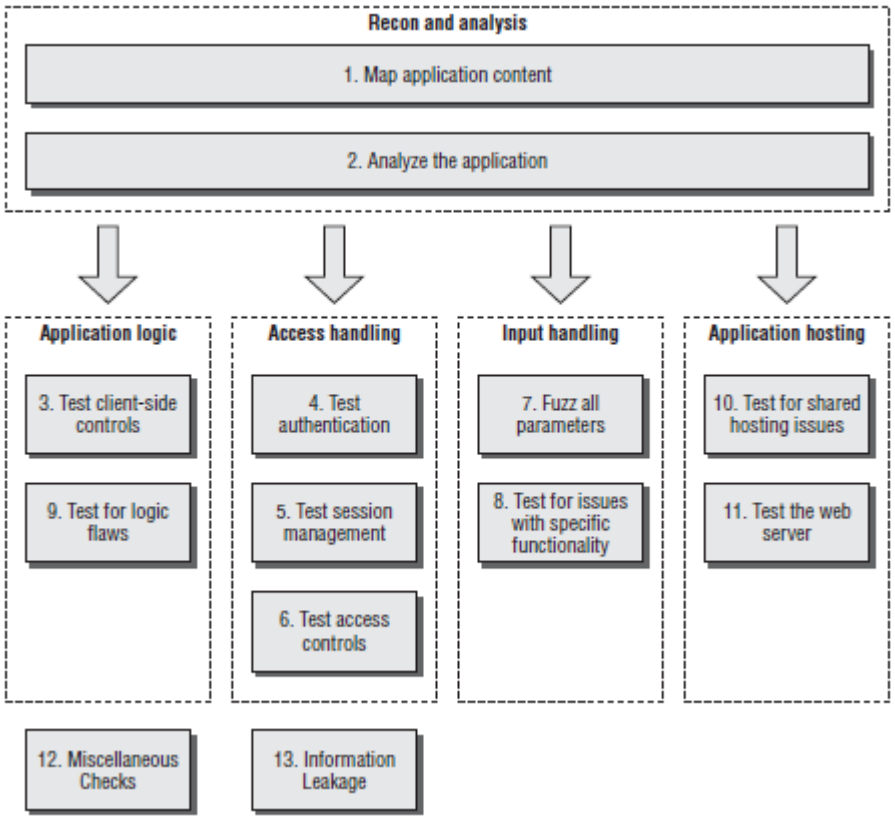
Using dirbuster can show hidden directories, one of which being the support logs.



Using the poison nullbyte %00 or %2500 in coding can help allow access to the developers log json.bak file.

A Web Application Hacker's Handbook

A Web Application Hacker's Methodology



1. Explore visible content. Use a passive sitemap like Burp Suite.
2. Consult public resources. (wayback machine, google site:)
3. Discover hidden content. Check requests for non-existent items.
4. Discover default content. Run nikto, dirbuster.
5. Enumerate identifier-specified functions. Request parameters.
6. Test for debug parameters.
7. Identify application functionality
8. Identify data entry points
9. Identify technologies used. Wappalyzer, Nmap
10. Map the attack surface.
11. Test data transmission via the client
12. Test client-side controls over user input
13. Test browser extension components
14. Understand the mechanism. (forms, certificates, multifactor)
15. Test password quality
16. Test username enumeration
17. Test resilience to password guessing
18. Test account recovery functions
19. Test “Remember me” functions
20. Test username uniqueness
21. Test predictability of autogenerated credentials
22. Check for unsafe transmission of credentials
23. Check for unsafe distribution of credentials
24. Test for insecure storage
25. Test for logic flaws
26. Test tokens
27. Check session data
28. Check for CSRF
29. Check cookie scope
30. Test access controls
31. Test with multiple accounts
32. Test with limited access
33. Test for insecure access control methods
34. Test for input vulnerabilities. (fuzz all request parameters, SQL injection, XSS, OS commands, path traversal, script injection, file inclusion)
35. Test for function specific input vulnerabilities. (SMTP injection, SOAP injection, LDAP injection, XPath injection, XXE injection)
36. Test for logic flaws
37. Test for shared hosting vulnerabilities
38. Test for application server vulnerabilities
39. Misc checks (DOM-based, local privacy, weak SSL ciphers, same-origin policy)
40. Follow up on any information leakage

OWASP Top 10

A01:2021 – Broken Access Control

Broken access control commonly leads to unauthorised information disclosure, modification or destruction of all data or performing a business function outside of the user's limits.

A02:2021 – Cryptographic Failures

Cryptographic failures compromise the protection needs of data in transit and at rest. Examples include passwords, credit card numbers, health records, personal information, and business secrets.

A03:2021 – Injection

Injection includes attacks such as cross site scripting, SQL injection and external control of file names or paths. These attacks can occur when an application hasn't been validated, filtered, or sanitized correctly.

A04:2021 – Insecure Design

This is a broad category representing different weaknesses expressed as missing or ineffective control design. These include generating error messages containing sensitive information, unprotected storage of credentials, trust boundary violation and insufficiently protected credentials.

A05:2021 – Security Misconfiguration

Misconfiguration includes missing appropriate security hardening across the application stack, the inclusion of unnecessary features, default accounts and passwords, and poorly configured security headers. Notable CWEs include improper restriction of XML eternal entity reference and configurations.

A06:2021 – Vulnerable and Outdated Components

You can be vulnerable if you do not know what versions you are running, your nested dependencies, out of date software, unsupported software, etc.

A07:2021 – Identification and Authentication Failures

Authentication weaknesses can occur when an application permits automated attacks, permits brute force attacks, uses weak or ineffective credential recovery processes, uses default or well-known passwords, uses plain text or weakly hashed passwords, has missing multi-factor authentication, exposes identifiers in the URL or does not correctly invalidate session IDs.

A08:2021 – Software and Data Integrity Failures

Software and data integrity failures relate to code and infrastructure that does not protect against integrity violations. An example of this is where an application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks. Prevention methods include using digital signatures to verify the integrity of the software or data, ensure libraries and dependencies are using trusted repositories and ensure that unsigned or unencrypted data is not sent to untrusted clients without a form of integrity check.

A09:2021 – Security Logging and Monitoring Failures

Insufficient logging, detection, monitoring and active response occurs when logs of applications are not monitored for suspicious activity, logs are only stored locally, warnings generate inadequate or unclear logs and if penetration tests and scans do not trigger alerts.

A10:2021 – Server-Side Request Forgery

SSRF flaws occur whenever a web application is fetching a remote resource without validating the user supplied URL allowing an attacker to coerce the application to send a crafted request to an unexpected destination.

MITRE Common Weaknesses

Common Weakness Enumeration (CWE) is a community developed list of common software and hardware security weaknesses. Weaknesses are flaws, faults, bugs or other errors in software or hardware implementation, code, design, or architecture that if left unaddressed could result in systems, networks or hardware being vulnerable to attack.

CWEs help developers and security practitioners to describe and discuss software and hardware weaknesses in a common language, check for weaknesses in existing software and hardware products, evaluate coverage of tools targeting these weaknesses, leverage a common baseline standard for weakness identification, mitigation, and prevention and prevent software and hardware vulnerabilities prior to deployment.

Portswigger Burpsuite

Portswigger's Burpsuite is a web application testing tool that helps users find a wide range of vulnerabilities in applications automatically and manually. It can crawl applications and create a site map, find issues within pages, includes essential tools like the interception proxy, repeater, intruder, decoder, sequencer, and comparer.

The interception proxy acts as proxy between you and the application, capturing packets sent to the application and displaying them in raw text. This can be used to view, manipulate, alter, or remove data from the packet before it reaches the application's server. It is set up on the host machine on a given port (usually 8080) and can be switched on and off. This can be useful in testing vulnerabilities

and creating exploits within the application API as noted in the beginner's web application hacking tutorial above.

Network Security

NMAP Network Scanning

TCP and UDP

TCP is a connection-oriented protocol, whereas UDP is a connectionless protocol and while more popular services on the internet run over the TCP protocol, UDP services are still widely deployed.

TCP requires an established connection to transmit data and the connection should be closed once transmission is complete. It is also able to sequence data, guarantees the delivery of data to the destination, can retransmit lost packets, has an extensive error checking and acknowledgement of data and is read as a byte stream. Common uses for TCP are HTTPS, HTTP, SMTP, POP, and FTP.

UDP is a connectionless protocol with no requirements for opening, maintaining, or terminating a connection. It is unable to sequence data, cannot guarantee the delivery of data and has no ability to retransmit lost packets. It uses a basic error checking mechanism using checksums and its packets are sent with defined boundaries, sent individually, and checked for integrity on arrival. Common uses for UDP are video conferencing, streaming, DNS, and VoiP.

Port Scanning

The port scanner is the core function of Nmap and unless instructed otherwise, scans the most used 1000 TCP ports on a host. Port scanning will classify each port into the state open, closed filtered, unfiltered, open|filtered or closed|filtered.

The open state means that an application is actively accepting TCP connections or UDP packets on this port. Finding these is often the primary goal of port scanning.

The closed state means that the port is accessible but there is no application listening on it.

The filtered state means Nmap cannot determine whether the port is open because packet filtering prevents its probes from reaching the port.

The unfiltered state means that the port is accessible, but Nmap is unable to determine whether it is opened or closed. This is only seen within an ACK scan.

The open|filtered state occurs when Nmap is unable to determine whether it is opened or filtered. This is seen in UDP, IP protocol, FIN, NULL and Xmas scans.

The closed|filtered state occurs when Nmap is unable to determine whether the port is closed or filtered. This is only seen in an ID idle scan.

Ports are simply software abstraction, used to distinguish between communication channels. Like how IP addresses identify machines on a network, ports identify specific applications used on a single machine.

Top 20 most common TCP ports

80 – HTTP

23 – Telnet

443 – HTTPS

21 – FTP

22 – SSH

25 – SMTP

3389 – ms-term-server

110 – POP3

445 – Microsoft-DS

139 – NetBIOS-SSN

143 – IMAP

53 – Domain

135 – MSRPC

3306 – MySQL

8080 – HTTP Proxy

1723 – PPTP

111 – RPCBind

995 – POP3S

993 – IMAPS

5900 – VNC

Top 20 most common UDP ports

631 – IPP

161 – SNMP

137 – NETBIOS-SN

123 – NTP

138 – NETBIOS-DGM

1434 – MS-SQL-DS

445 – Microsoft-DS

135 – MSRPC

67 – DHCPS

53 – Domain

139 – NETBIOS-SSN

500 – ISAKMP

68 – DHCPC

520 – Route

1900 – UPNP

4500 – nat-t-ike

524 – Syslog

49152 – IANA specified ports

162 – SNMPTrap

69 – TFTP

Service Identification

Using its nmap-services database of more than 2,200 well-known services and while this information is usually accurate, there are often times when hosts run applications from different ports. Nmap also has the ability to fingerprint which version of applications are running which can help dramatically in determining which exploits an application is vulnerable to however, keep in mind that security fixes are often back-ported to earlier versions of software so you cannot rely solely on the version number to prove a service is vulnerable.

Some service scans can sometimes reveal information about a target beyond the service type and version number. They can also include SSH protocol numbers, Apache modules and more. The

version detection can also discover operating systems and device types and a scan can be used in fingerprinting a system with the –o flag.

Useful Nmap flags

-p: port selection

-A: aggressive detection mode

-O: OS fingerprinting

-T4: Timing, increases the speed of scan. The faster the more noise it makes.

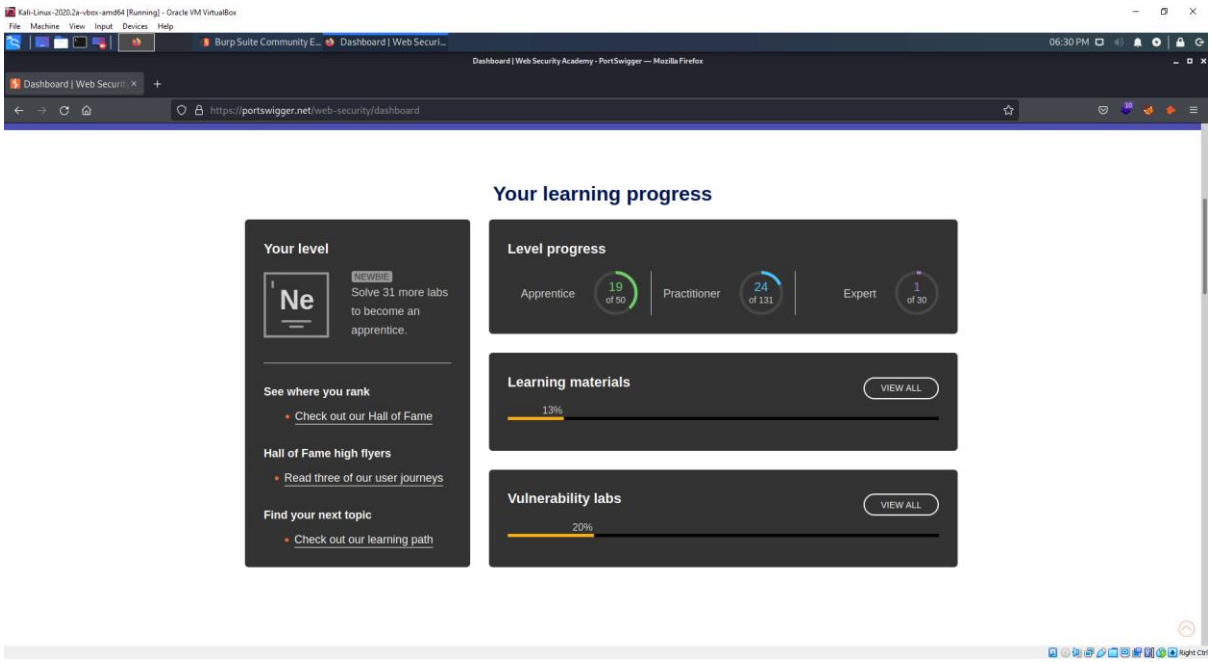
-sV: service scan

-sS: stealth TCP scan

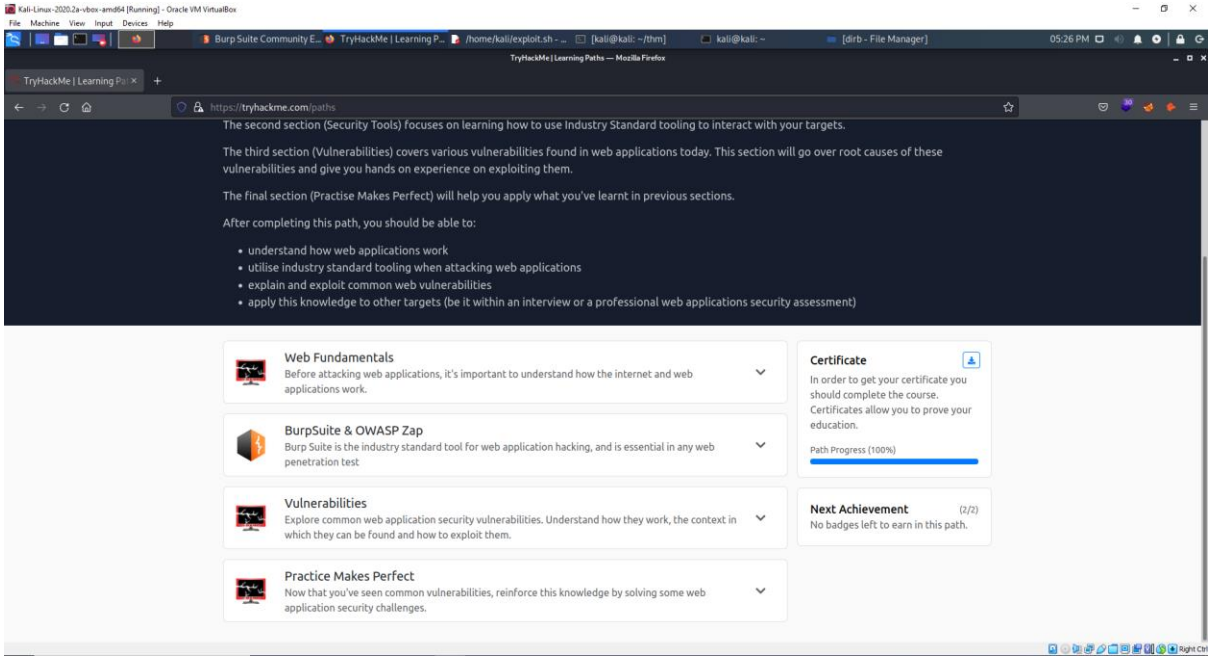
-sU: UDP scan

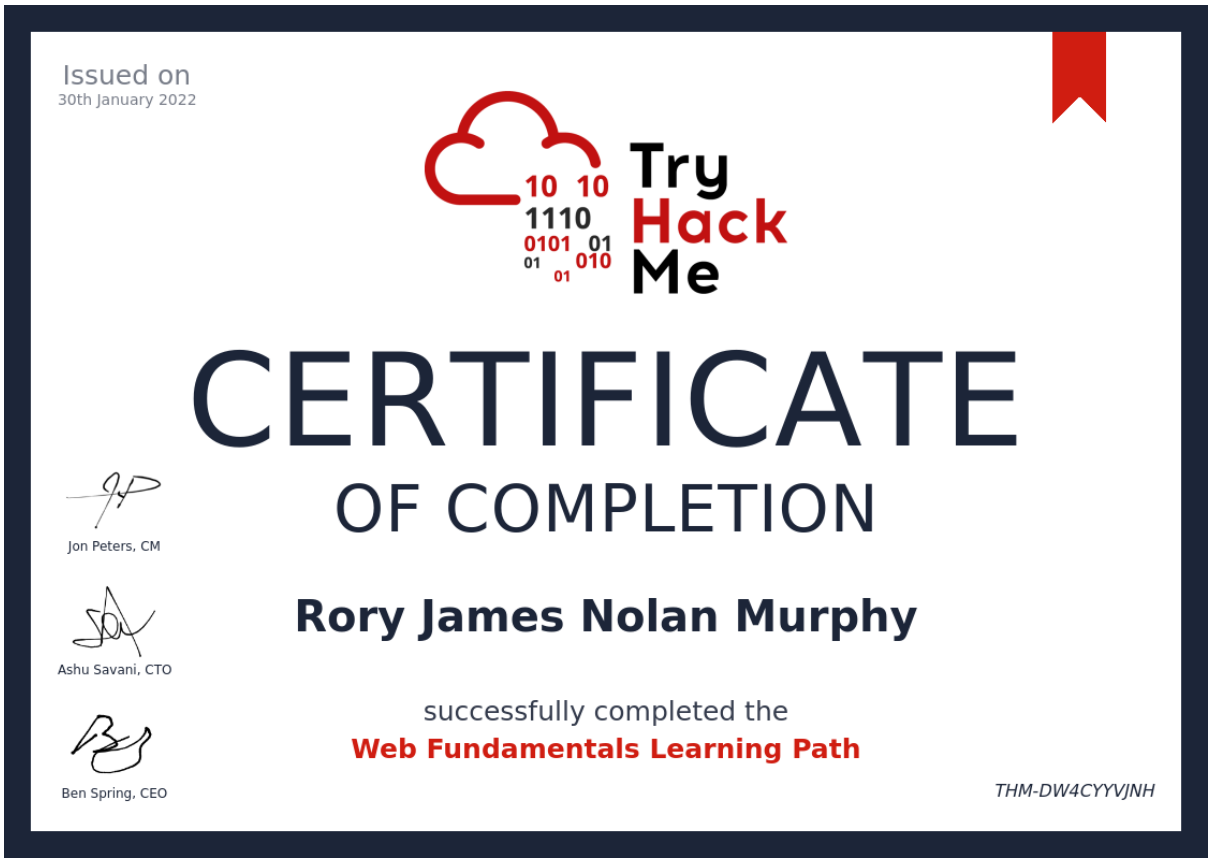
Hands on Labs

Portswigger Academy



TryHackMe


TryHackMe | Learning Paths — Mozilla Firefox
https://tryhackme.com/paths
The second section (Security Tools) focuses on learning how to use Industry Standard tooling to interact with your targets.
The third section (Vulnerabilities) covers various vulnerabilities found in web applications today. This section will go over root causes of these vulnerabilities and give you hands on experience on exploiting them.
The final section (Practise Makes Perfect) will help you apply what you've learnt in previous sections.
After completing this path, you should be able to:
• understand how web applications work
• utilise industry standard tooling when attacking web applications
• explain and exploit common web vulnerabilities
• apply this knowledge to other targets (be it within an interview or a professional web applications security assessment)
Web Fundamentals
Before attacking web applications, it's important to understand how the internet and web applications work.
BurpSuite & OWASP Zap
Burp Suite is the industry standard tool for web application hacking, and is essential in any web penetration test
Vulnerabilities
Explore common web application security vulnerabilities. Understand how they work, the context in which they can be found and how to exploit them.
Practice Makes Perfect
Now that you've seen common vulnerabilities, reinforce this knowledge by solving some web application security challenges.
Certificate
In order to get your certificate you should complete the course. Certificates allow you to prove your education.
Path Progress (100%)
Next Achievement (2/2)
No badges left to earn in this path.


Issued on
30th January 2022
Try Hack Me
CERTIFICATE
OF COMPLETION
Jon Peters, CM
Ashu Savani, CTO
Ben Spring, CEO
Rory James Nolan Murphy
successfully completed the
Web Fundamentals Learning Path
THM-DW4CYYVJNH

References

Lindsay, J., Chell, D., Erasmus, T., Colley, S., & Whitehouse, O. (2015). *The Mobile Application Hacker's Handbook* (1st ed.). Wiley Publishing.

Scambray, J., McClure, S., & Kurtz, G. (2012). *Hacking Exposed 7: Network Security Secrets and Solutions* (7th ed.). McGraw-Hill Education.

Stuttard, D., & Pinto, M. (2011). *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws* (2nd ed.). Wiley Publishing.