BSc. (Hons) Degree in IT
(Sp. Cyber Security) – 2nd Year 2nd Semester

Web Security – IE2062

Title:

Web Audit Assignment – Final Report

IT19127910

R.P.A. Ranasinghe

Content	Page

Introduction

This is my final report based on web security web audit assignment. To do this assignment I had to select a domain and scan that domain to check whether our selected domain has top 10 vulnerabilities listed by OWASP.

OWASP Top 10 Web Application Security Risks

Injection:

Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

Broken Authentication:

Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.

Sensitive Data Exposure:

Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

XML External Entities (XXE):

Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

Broken Access Control:

Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

Security Misconfiguration:

Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.

Cross-Site Scripting XSS:

XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. XSS allows attackers to execute scripts in the victim's browser which

can hijack user sessions, deface web sites, or redirect the user to malicious sites.

Insecure Deserialization:

Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

Using Components with Known Vulnerabilities:

Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

Insufficient Logging & Monitoring:

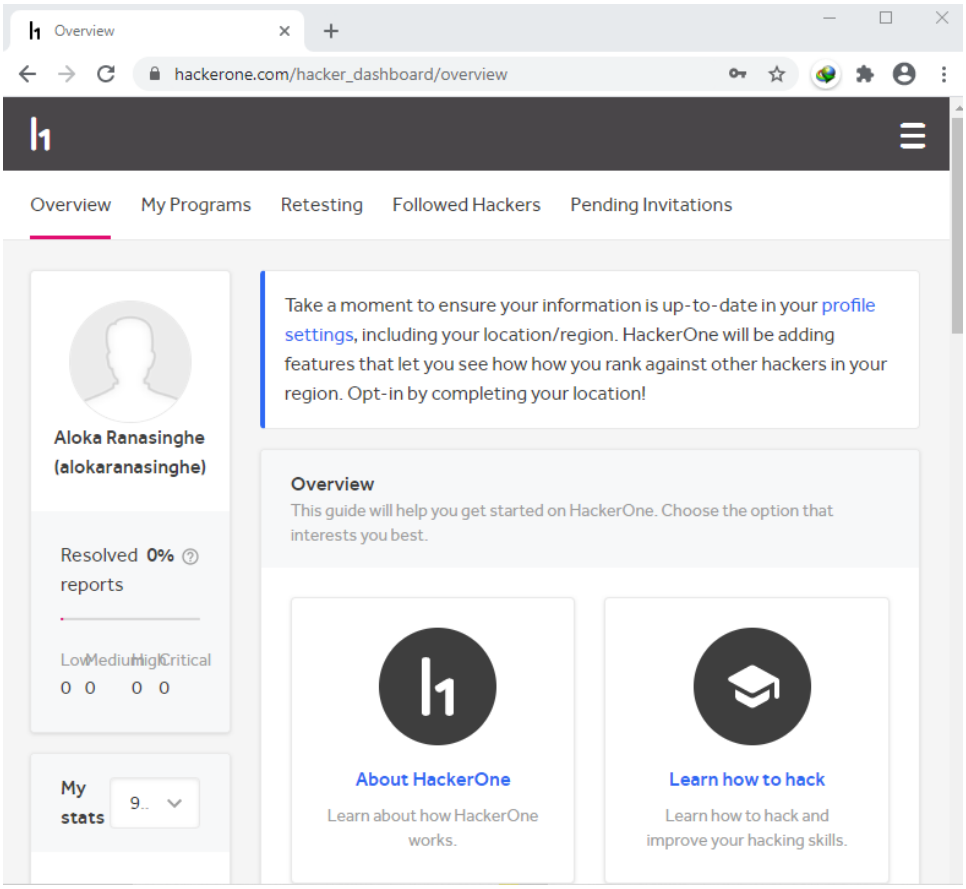
Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

Do that scanning processes I used different tools. They are

- Netsparker
- Nikto

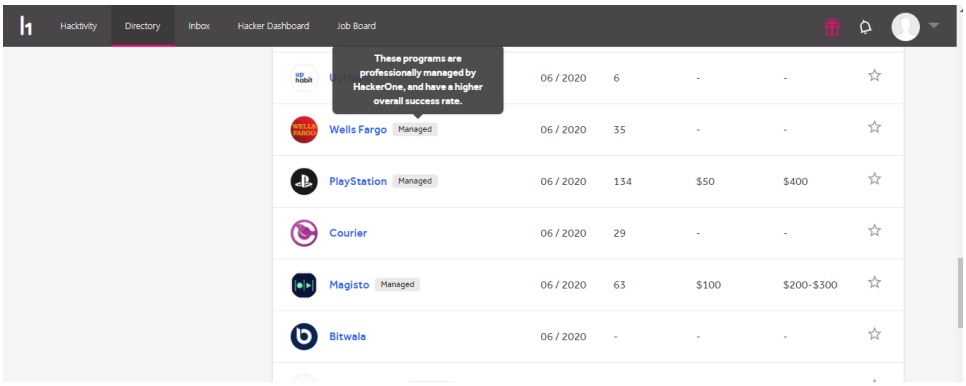
Steps of Web Audit

Before these scans I created an account in hackerone and searched domains.

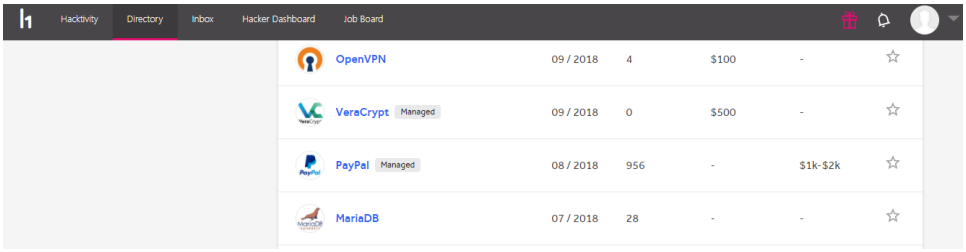


Then I selected 3 domains to scan sope of sub domains.They are

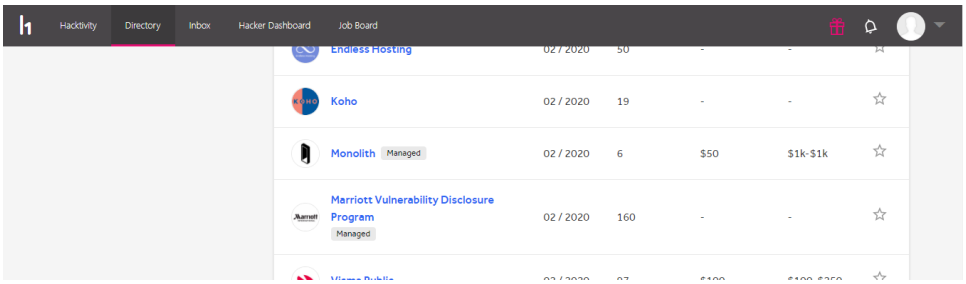
playstaion.com



paypal.com

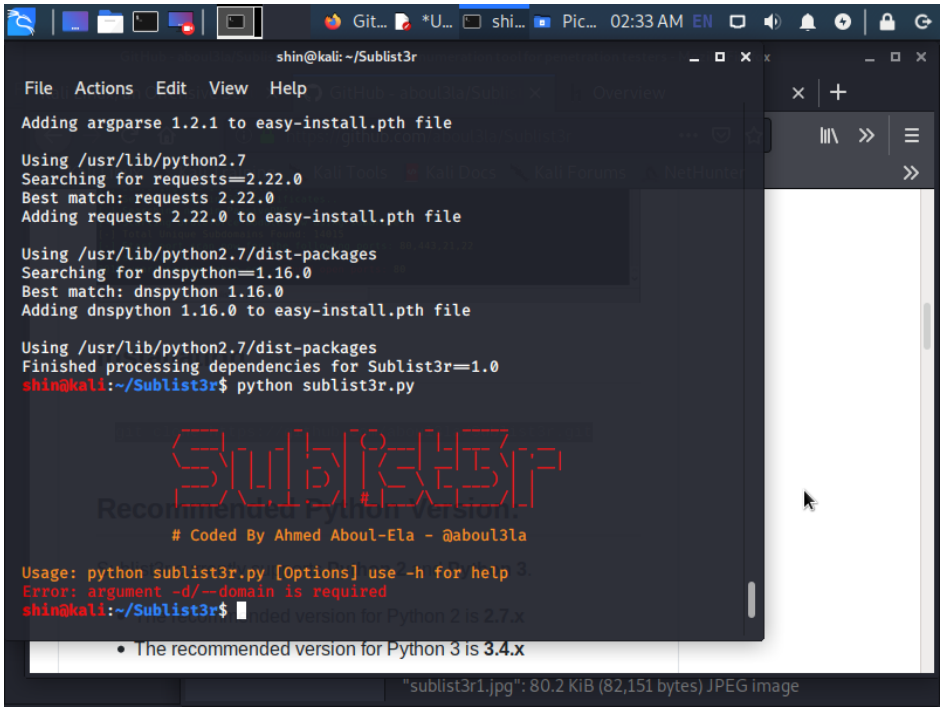


monolith.com



After that I had to install sublist3r in my kali virtual machine. To perform that firstly I searched sublist3r in google and then I got a GitHub link. By using that GitHub link, I got gitclone command. Then copy that command and paste it in my Kali machine terminal. After cloned I installed python3.pip and python.pip.

Then went to sublist3r directory and run this **sudo pip install –r requrements.txt command**. After that I had to install python setup. But when I tried to do that, I got an error So, I searched that error in google and I had to install python setup tools before install python setup. After complete above steps successfully I run the sublist3r.py.



Firstly, I scanned playstation.com. Then I scanned paypal.com and monolith.com.
After those scanning, I had to select a domain among them. So, I decided to select paypal.com.

All of the above-mentioned steps are in this video:

https://drive.google.com/file/d/1jHEWSci_XdoQqaQOizJM_8Lj6ZCIkDCS/view?usp=sharing

Then I started scanning process of my selected domain paypal.com. I selected these sub domains to scan:

- paypal.com
- xoom.com
- braintreegateway.com
- braintreepayments.com
- paypal.me
- paydiant.com
- paypalobjects.com
- prequal.swiftfinancial.com
- partner.swiftfinancial.com
- decision.swiftfinancial.com
- swiftcapital.com
- loanbuilder.com
- swiftfinancial.com
- api.swiftfinancial.com
- my.swiftfinancial.com
- api.loanbuilder.com
- my.loanbuilder.com

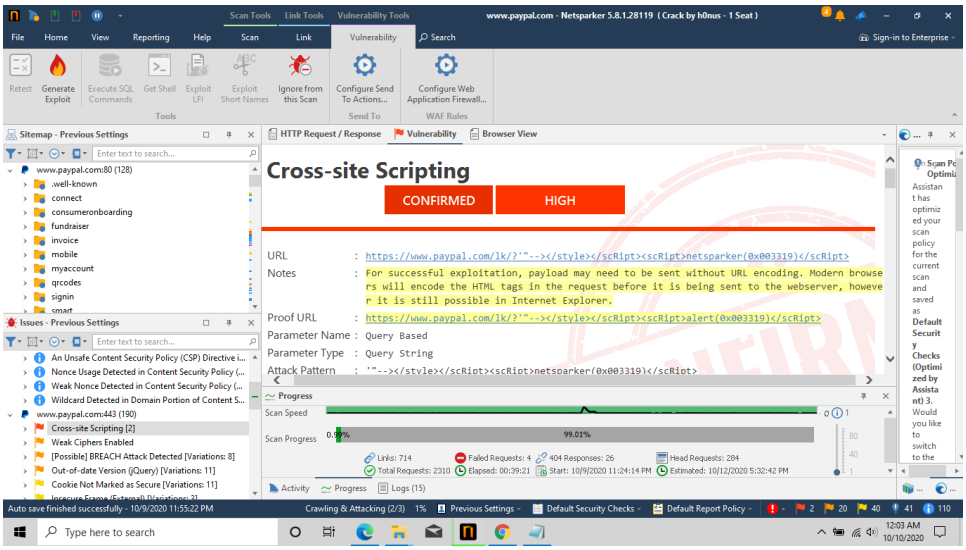
Netsparker

Firstly, I used Netsparker to scan some of my sub domains. They are,

- paypal.com
- xoom.com
- braintreegateway.com
- paypal.me
- braintreepayments.com

Paypal.com:

- 7 high (same cross site scripting vulnerability has detected seven times)
- 23 mediums
- 77 low vulnerabilities.



CLASSIFICATION

PCI DSS 3.2	6.5.7
OWASP 2013	A3
OWASP 2017	A7
CWE	79
CAPEC	19
WASC	8
HIPAA	164.308(A)
ISO27001	A.14.2.5

CVSS 3.1 SCORE

Base	7.4 (High)
Temporal	7.4 (High)
Environmental	7.4 (High)

CVSS Vector String

CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:N/A:N

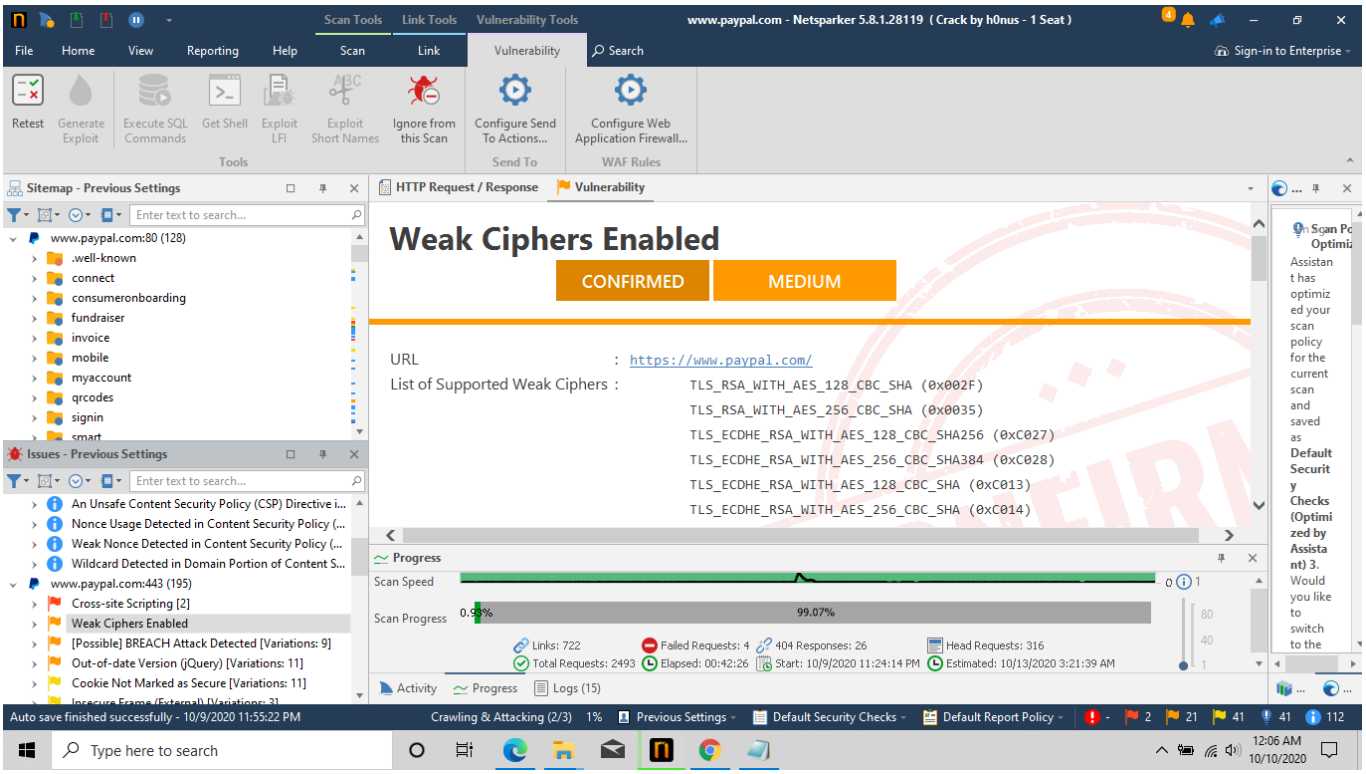
CVSS 3.0 SCORE

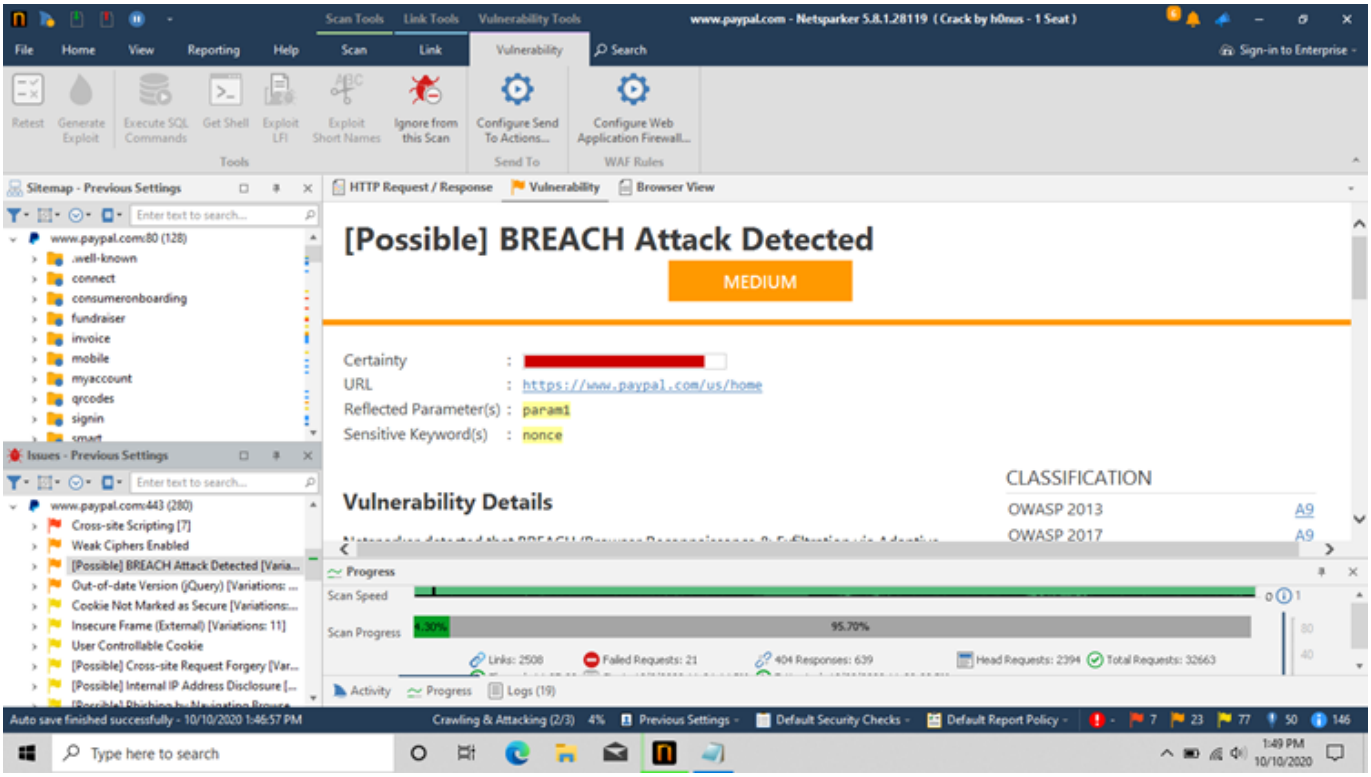
Base	7.4 (High)
Temporal	7.4 (High)
Environmental	7.4 (High)

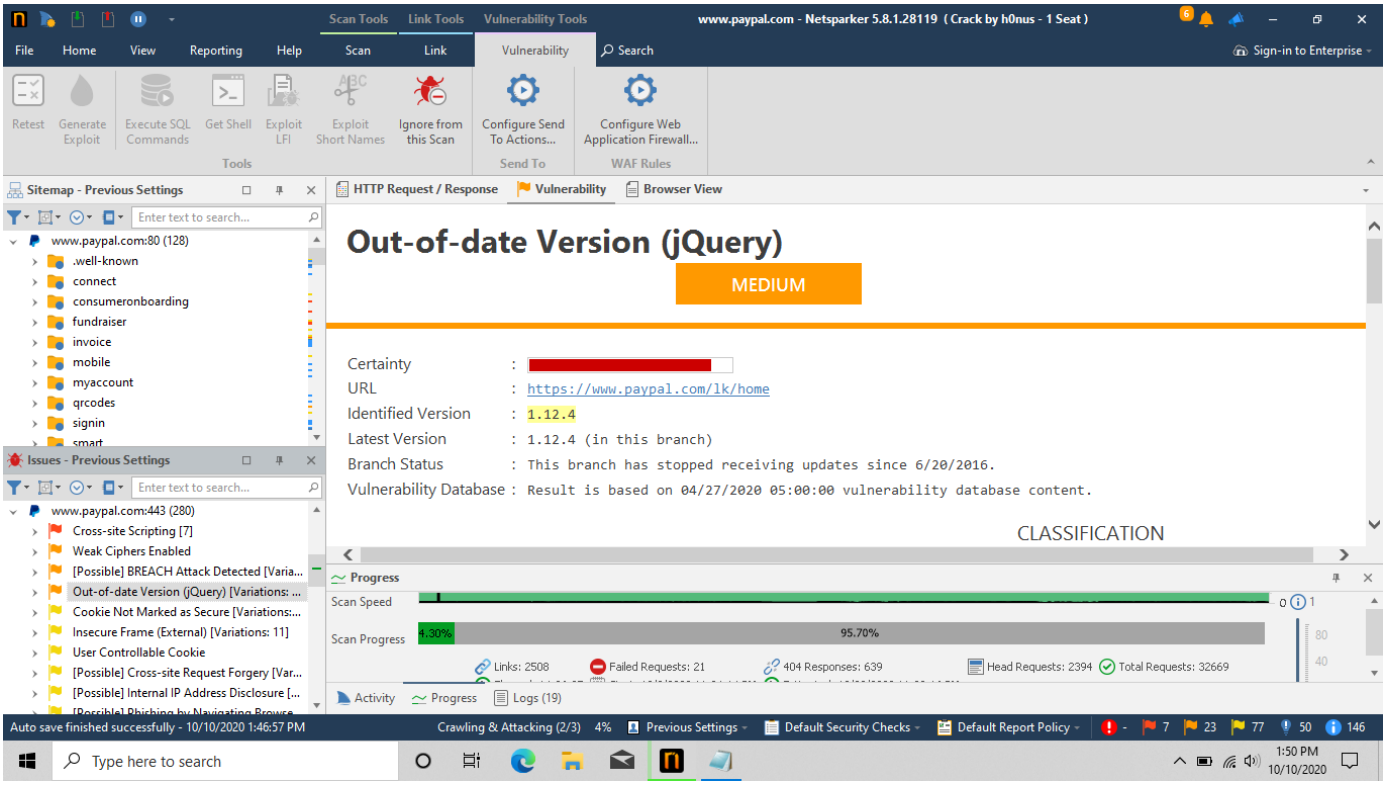
CVSS Vector String

CVSS:3.0/AV:N/AC:L/PR:N/UI:R/S:C/C:H/I:N/A:N

Medium Vulnerabilities




www.paypal.com - Netsparker 5.8.1.28119 (Crack by h0nus - 1 Seat)
[Possible] BREACH Attack Detected
MEDIUM
Certainty :
URL : https://www.paypal.com/us/home
Reflected Parameter(s) : param1
Sensitive Keyword(s) : nonce
Vulnerability Details
CLASSIFICATION
OWASP 2013 A9
OWASP 2017 A9
Progress
Scan Speed
Scan Progress
4.30%
95.70%
Links: 2508
Failed Requests: 21
404 Responses: 639
Head Requests: 2394
Total Requests: 32663
Activity
Progress
Logs (19)
Auto save finished successfully - 10/10/2020 1:46:57 PM
Crawling & Attacking (2/3) 4%
Previous Settings
Default Security Checks
Default Report Policy
Type here to search
1:49 PM
10/10/2020


www.paypal.com - Netsparker 5.8.1.28119 (Crack by h0nus - 1 Seat)
Out-of-date Version (jQuery)
MEDIUM
Certainty :
URL : https://www.paypal.com/lk/home
Identified Version : 1.12.4
Latest Version : 1.12.4 (in this branch)
Branch Status : This branch has stopped receiving updates since 6/20/2016.
Vulnerability Database : Result is based on 04/27/2020 05:00:00 vulnerability database content.
CLASSIFICATION
Progress
Scan Speed
Scan Progress
4.30%
95.70%
Links: 2508
Failed Requests: 21
404 Responses: 639
Head Requests: 2394
Total Requests: 32669
Activity
Progress
Logs (19)
Auto save finished successfully - 10/10/2020 1:46:57 PM
Crawling & Attacking (2/3) 4%
Previous Settings
Default Security Checks
Default Report Policy
Type here to search
1:50 PM
10/10/2020

Low Vulnerabilities

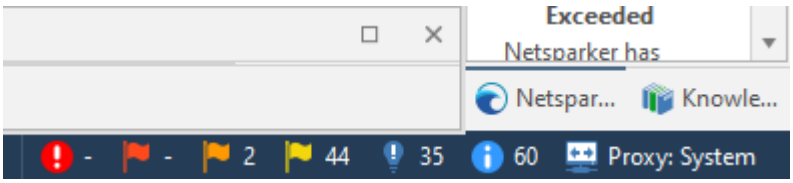
- Cookie not marked as secure
- Insecure frame (External)
- User controlled cookie
- [Possible] Cross-site request forgery
- [Possible] Internal IP address disclosure
- [Possible] Phishing by navigating browser tabs
- Cookie not marked as HTTPonly
- Missing content-type header

paypal.com scan video:

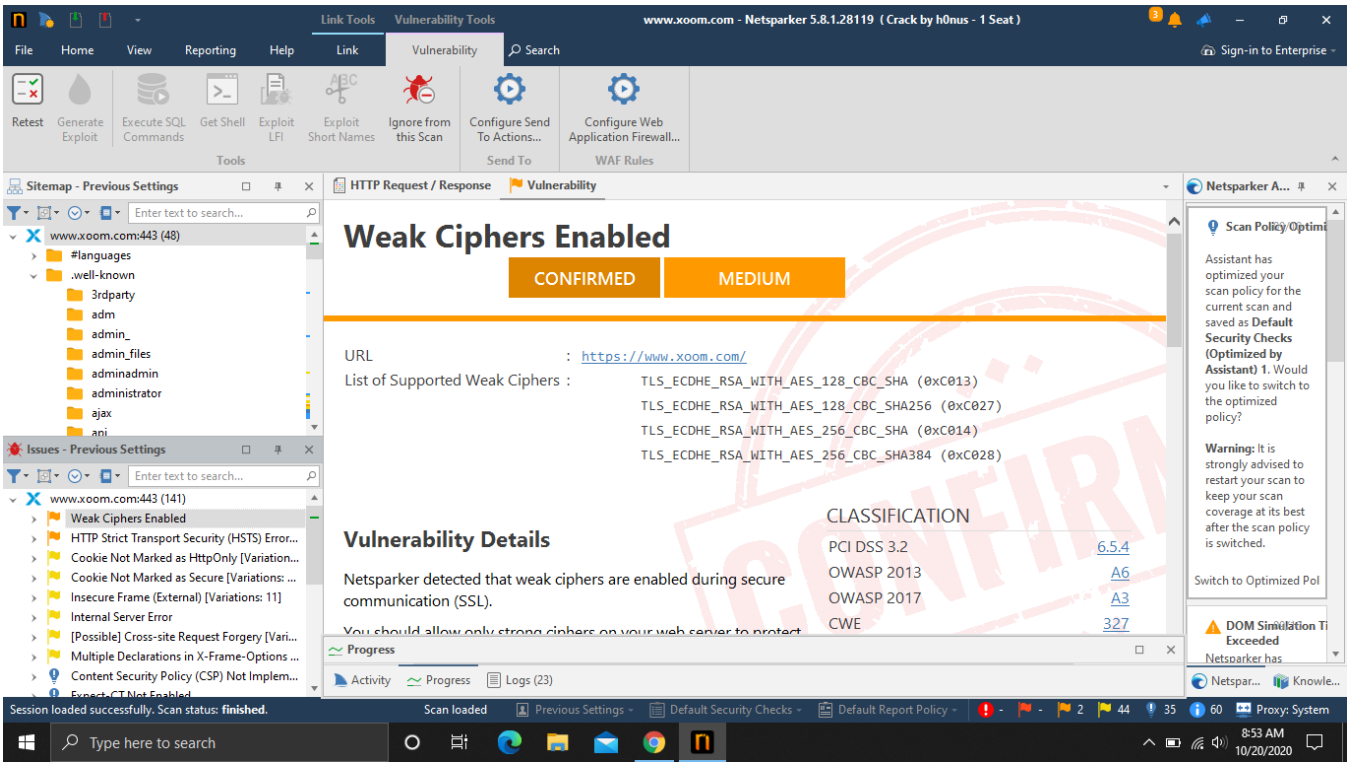
https://drive.google.com/file/d/1XmMaMcAGT0fHXYId6tHww_OjVii4A2MM/view?usp=sharing

xoom.com:

- 2 medium vulnerabilities
- 44 low vulnerabilities



Medium Vulnerabilities



CLASSIFICATION

PCI DSS 3.2	6.5.4
OWASP 2013	A6
OWASP 2017	A3
CWE	327
CAPEC	217
WASC	4
ISO27001	A.14.1.3

CVSS 3.0 SCORE

Base	6.8 (Medium)
Temporal	6.8 (Medium)
Environmental	6.8 (Medium)

CVSS Vector String

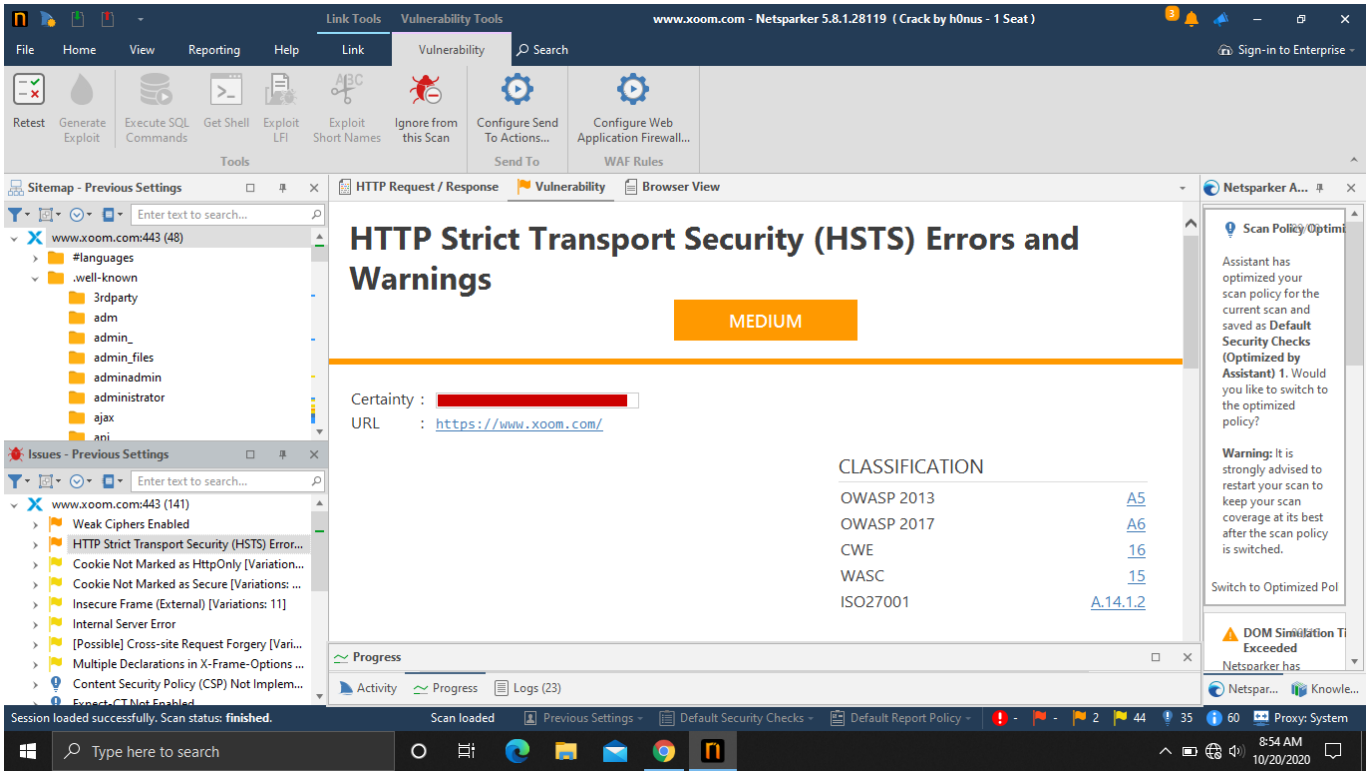
CVSS:3.0/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N

CVSS 3.1 SCORE

Base	6.8 (Medium)
Temporal	6.8 (Medium)
Environmental	6.8 (Medium)

CVSS Vector String

CVSS:3.1/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N



CLASSIFICATION

OWASP 2013	A5
OWASP 2017	A6
CWE	16
WASC	15
ISO27001	A.14.1.2

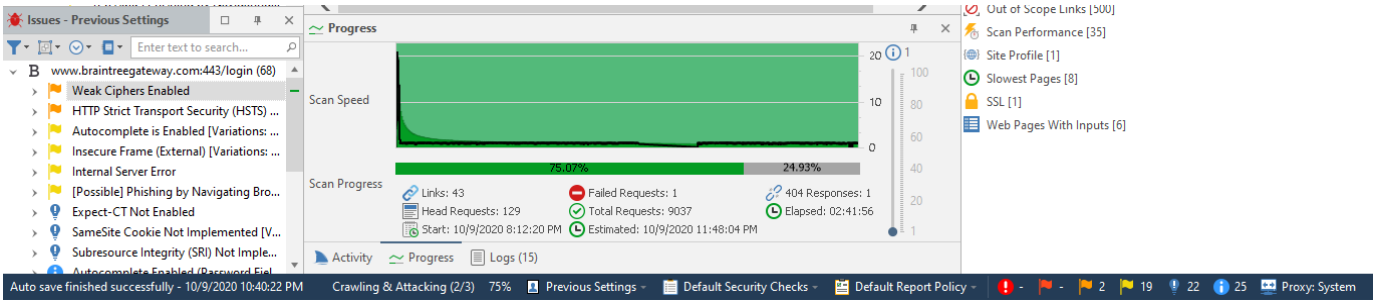
Low Vulnerabilities

- Cookie Not Marked as HttpOnly
- Cookie Not Marked as Secure
- Insecure Frame (External)
- Internal Server Error
- [Possible] Cross-site Request Forgery
- Multiple Declarations in X-Frame-Options Header

xoom.com scan video:
https://drive.google.com/file/d/1fE2sHzueBNVwBAq2t2LLjnwApqpiGpLN/view?usp=sharing

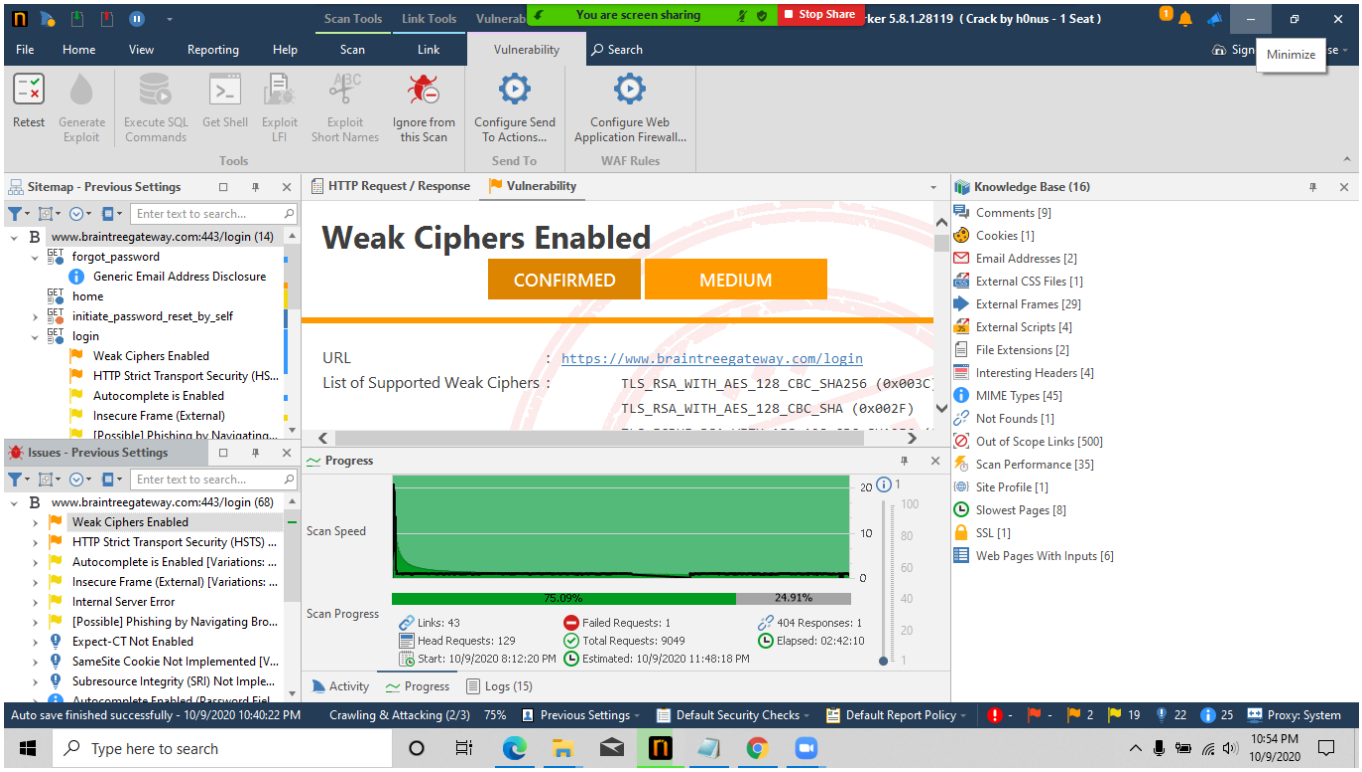
braintreegateway.com:

- 2 medium
- 19 low vulnerabilities.



Medium Vulnerabilities

1. Weak Cipher Enabled



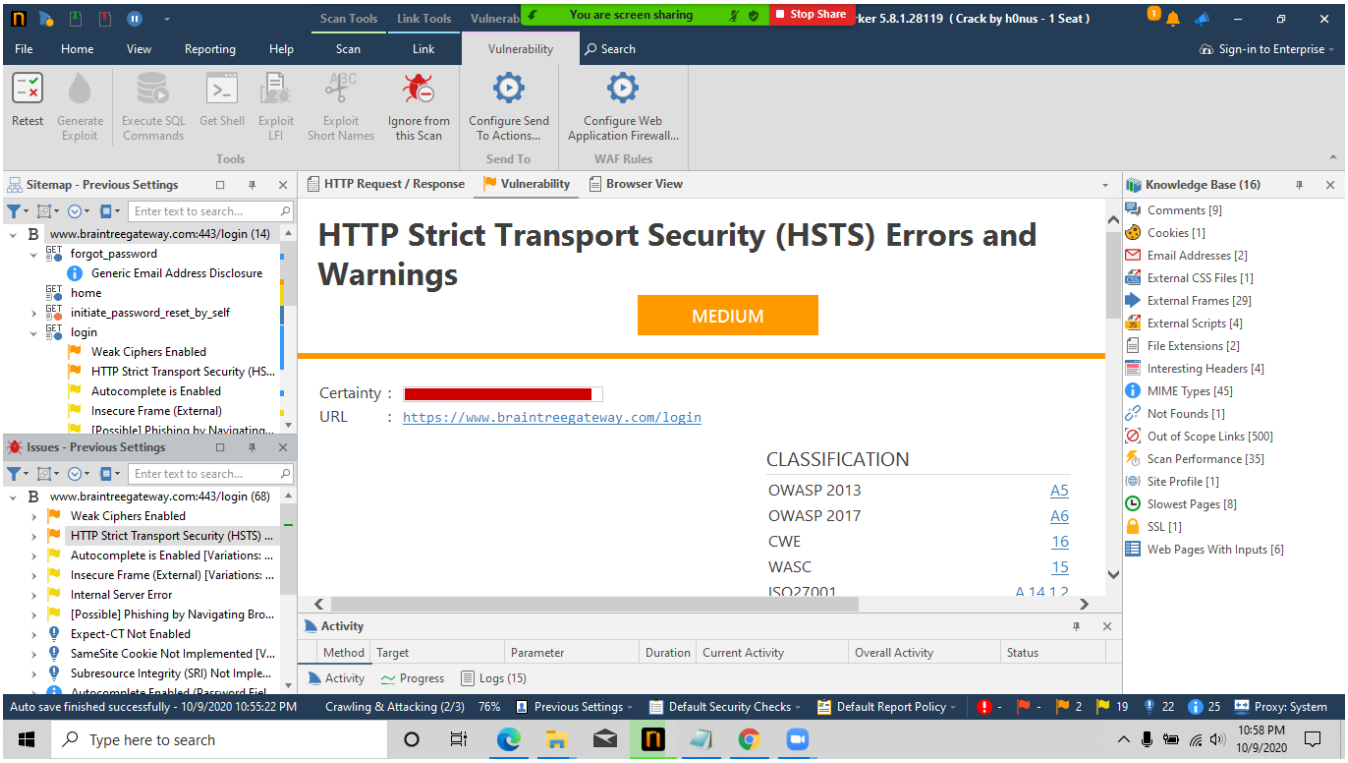
CLASSIFICATION

PCI DSS 3.2	6.5.4
OWASP 2013	A6
OWASP 2017	A3
CWE	327
CAPEC	217
WASC	4
ISO27001	A.14.1.3

CVSS 3.0 SCORE

Base	6.8 (Medium)
Temporal	6.8 (Medium)
Environmental	6.8 (Medium)

2. HTTP Strict Transport Security (HSTS) Errors and Warnings



CLASSIFICATION	
OWASP 2013	A5
OWASP 2017	A6
CWE	16
WASC	15
ISO27001	A.14.1.2

Low Vulnerabilities

Autocomplete is Enabled

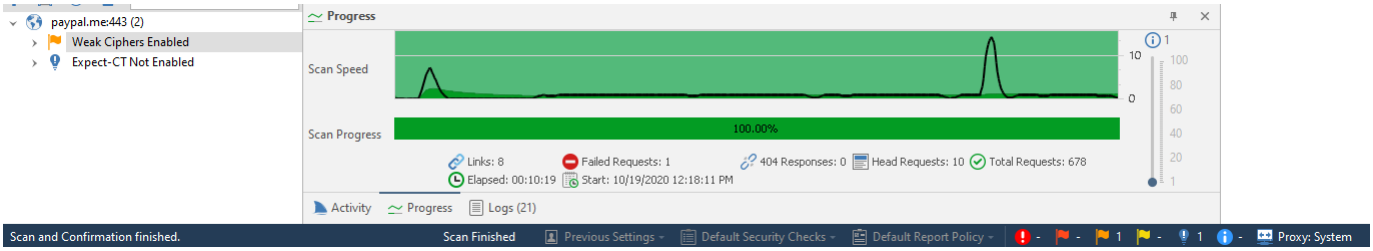
- Insecure Frame (External)
- Internal Server Error
- [Possible] Phishing by Navigating Browser Tabs

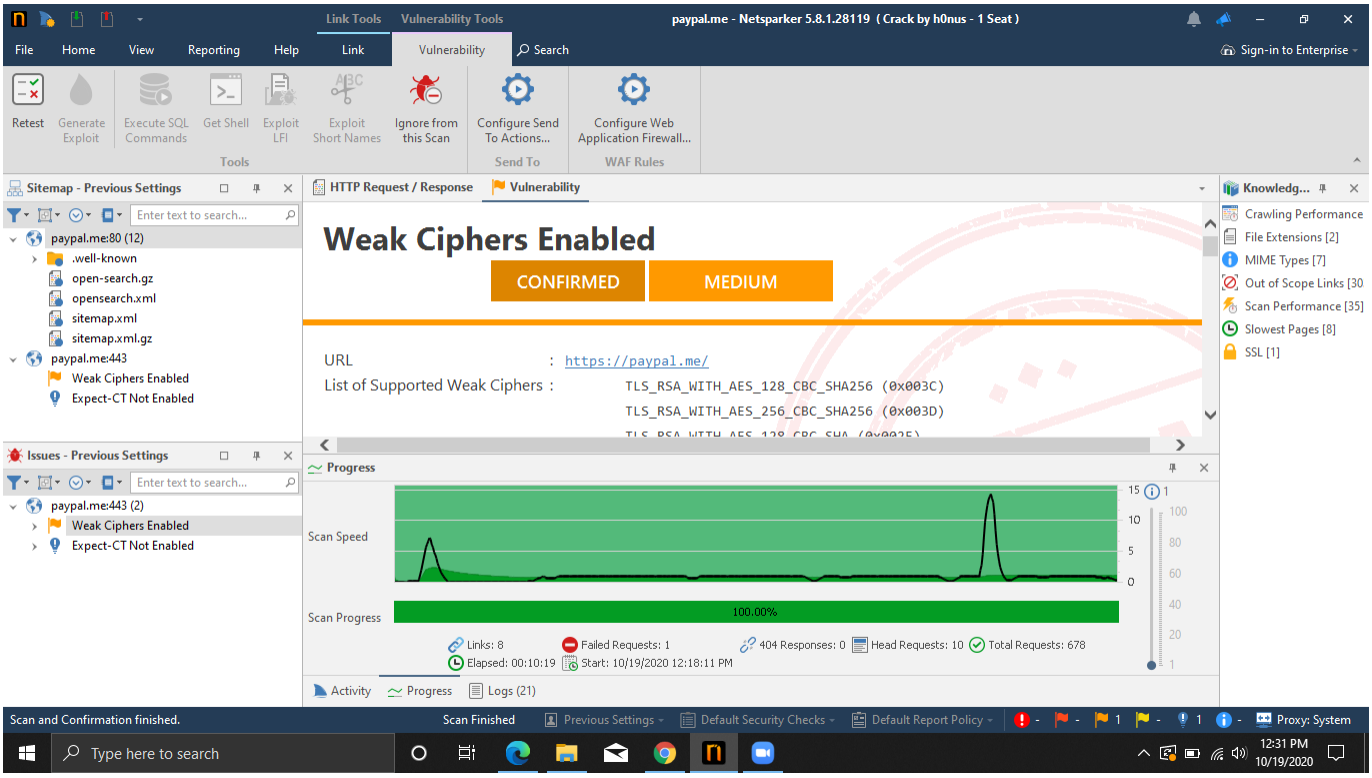
braintreegateway.com scan video:

https://drive.google.com/file/d/1m6vcBxnjRVe-BxuJlqF4SNtYAxIndZLC/view?usp=sharing

paypal.me:

only 1 medium vulnerability.





CLASSIFICATION

PCI DSS 3.2	6.5.4
OWASP 2013	A6
OWASP 2017	A3
CWE	327
CAPEC	217
WASC	4
ISO27001	A.14.1.3

CVSS 3.0 SCORE

Base	6.8 (Medium)
Temporal	6.8 (Medium)
Environmental	6.8 (Medium)

CVSS Vector String

CVSS:3.0/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N

CVSS 3.1 SCORE	
Base	6.8 (Medium)
Temporal	6.8 (Medium)
Environmental	6.8 (Medium)

CVSS Vector String

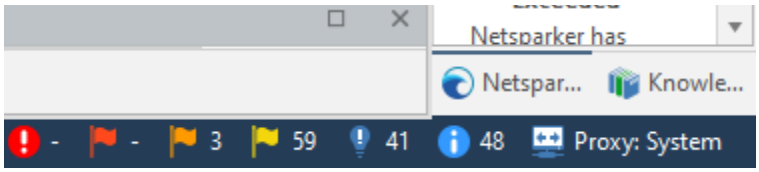
CVSS:3.1/AV:A/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:N

paypal.me scan video:

https://drive.google.com/file/d/1-8tPYfdOVR18xNXAzWvVzYroOtTAHMJe/view?usp=sharing

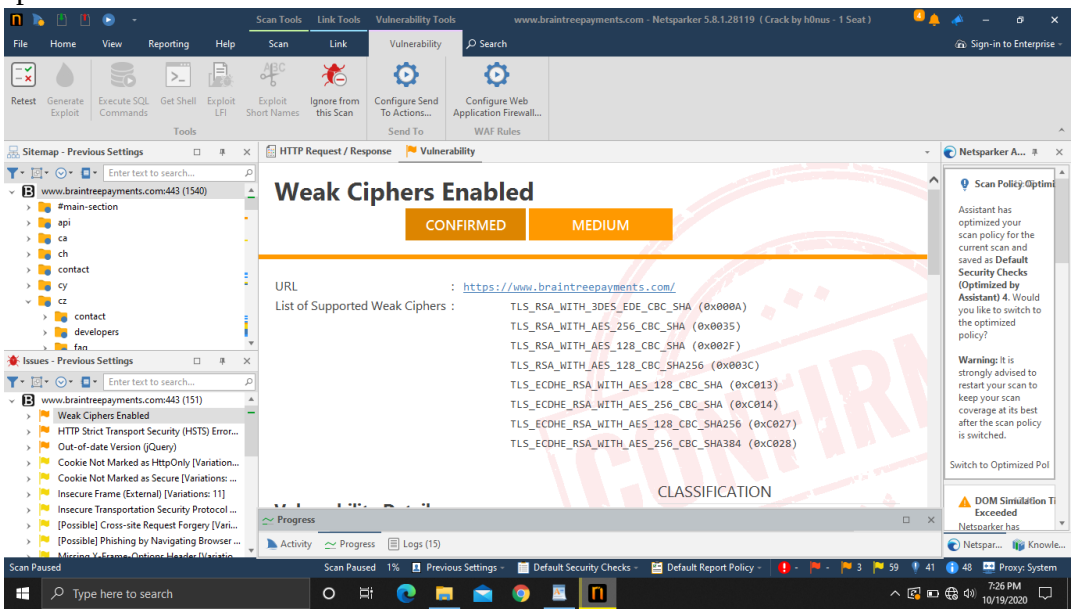
braintreepayments.com:

- 3 medium
- 59 low vulnerabilities

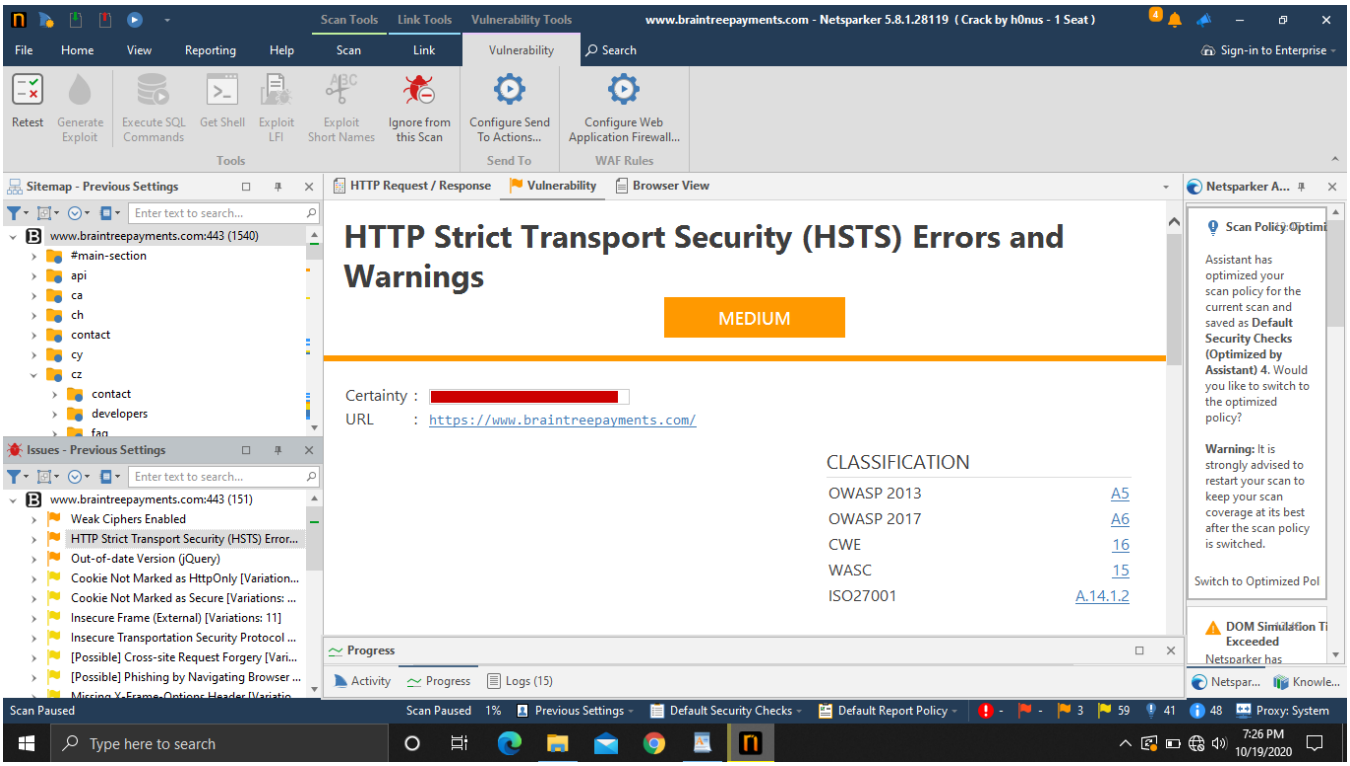


Medium Vulnerabilities

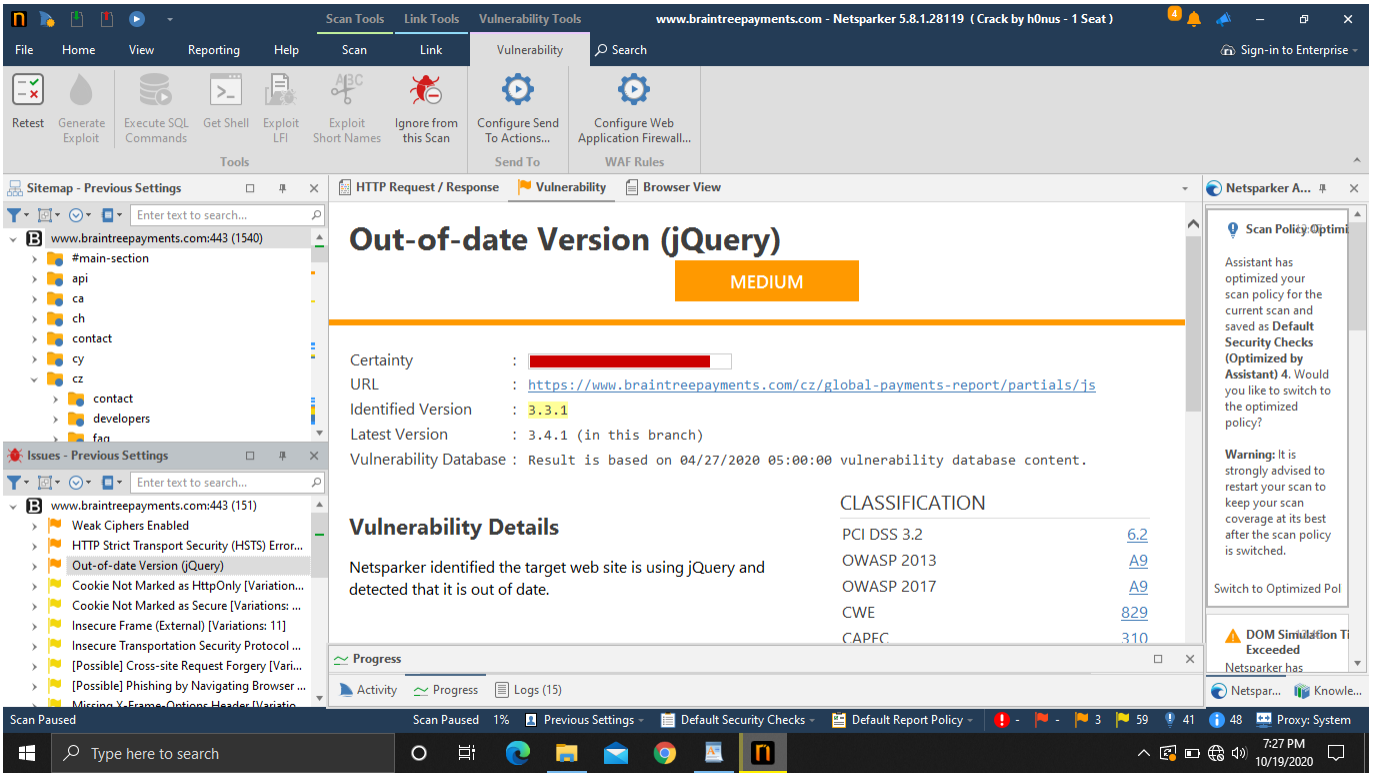
1. Weak Ciphers Enabled



2. HTTP Strict Transport Security (HSTS) Errors and Warnings



3. Out-of-date Version (jQuery)



Low Vulnerabilities

- Cookie Not Marked as HttpOnly
- Cookie Not Marked as Secure
- Insecure Frame (External)
- Insecure Transportation Security Protocol Supported (TLS 1.0)
- [Possible] Cross-site Request Forgery
- [Possible] Phishing by Navigating Browser Tabs
- Missing X-Frame-Options Header

braintreepayments.com scan video:

https://drive.google.com/file/d/1eMnvm-y6UOhlKUXBUQwZX-upQP0rFzk_/view?usp=sharing

When I consider about overall scans which I did using Netspaker, I got only one HIGH vulnerability. It is Cross-site scripting.

Cross-site scripting

It is a client-side code injection attack. In a victim's web browser, the attacker attempts to execute malicious scripts by inserting malicious code into a legitimate web page or web application. When the user enters the web page or web application that executes the malicious code, the actual attack takes place. Forums, message boards, and web pages allowing comments are widely used for cross-site scripting attacks. This occurs because the browser interprets the input entered by a user as HTML / JavaScript /VBScript.

There are three main types of XSS attacks. These are:

Reflected XSS, where the malicious script comes from the current HTTP request.
Stored XSS, where the malicious script comes from the website's database.
DOM-based XSS, where the vulnerability exists in client-side code rather than server-side code.

Impact
By using cross-site scripting, there are several different attacks that can be leveraged, including:

Hijacking user's active session.

Mounting phishing attacks.

Intercepting data and performing man-in-the-middle attacks.

Remedy

Output should be encoded according to the output position and meaning in order to prevent this. For example, if the output enters a JavaScript block within the HTML document, it is important to encode the output accordingly. Encoding can become very difficult, so using an encoding library like OWASP ESAPI and Microsoft Anti-cross-site scripting is highly recommended. Additionally, if an XSS vulnerability is accidentally implemented, you can enforce a strict Content security Policy (CSP) as a defense-in - depth measure. XSS vulnerabilities are still prevalent in web applications due to the difficulty of XSSPrevention and the lack of stable standard actions in programming languages and frameworks.

And I got 4 common Medium vulnerabilities. They are:

1. **Weak Ciphers Enabled**

List of Supported Weak Ciphers:

- TLS_RSA_WITH_AES_128_CBC_SHA (0x002F)
- TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 (0xC027)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xC028)
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xC013)
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xC014)

Netspaker detected that weak ciphers are enabled during secure communication (SSL).
You should allow only strong ciphers on your web server to protect secure communication with your visitors.

Impact

Attackers might decrypt SSL traffic between your server and your visitors.
Actions to Take

1. For Apache, you should modify the SSLCipherSuite directive in the httpd.conf.
2. SSLCipherSuite HIGH:MEDIUM:!MD5:!RC4
3. Lighttpd:
4. ssl.honor-cipher-order = "enable"
5. ssl.cipher-list = "EECDH+AESGCM:EDH+AESGCM"
6. For Microsoft IIS, you should make some changes to the system registry. Incorrectly editing the registry may severely damage your system. Before making changes to the registry, you should back up any valued data on your computer.

a. Click Start, click Run, type regedt32 or type regedit, and then click OK.

b. In Registry Editor, locate the following registry key:
HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders

c. Set "Enabled" DWORD to "0x0" for the following registry keys:

7. SCHANNEL\Ciphers\DES 56/56

SCHANNEL\Ciphers\RC4 64/128
SCHANNEL\Ciphers\RC4 40/128
SCHANNEL\Ciphers\RC2 56/128
SCHANNEL\Ciphers\RC2 40/128
SCHANNEL\Ciphers\NULL
SCHANNEL\Hashes\MD5

Remedy

Configure your web server to disallow using weak ciphers.

External References

- OWASP - Insecure Configuration Management
- OWASP Top 10-2017 A3-Sensitive Data Exposure
- Zombie Poodle - Golden Doodle (CBC)
- Mozilla SSL Configuration Generator
- Strong Ciphers for Apache, Nginx and Lighttpd

2. HTTP Strict Transport Security (HSTS) Errors and Warnings

Error Resolution

preload directive not present Submit domain for inclusion in browsers' HTTP Strict Transport Security (HSTS) preload list.

Vulnerability Details

Netsparker detected errors during parsing of Strict-Transport-Security header.

Impact

The HSTS Warning and Error may allow attackers to bypass HSTS, effectively allowing them to read and modify your communication with the website.

Remedy

Ideally, after fixing the errors and warnings, you should consider adding your domain to the the HSTS preload list. This will ensure that browsers automatically connect your website by using HTTPS, actively preventing users from visiting your site using HTTP. Since this list is hardcoded in users' browsers, it will enable HSTS even before they visit your page for the first time, eliminating the need for Trust On First Use (TOFU) with its associated risks and disadvantages. Unless you fix the errors and warnings your website won't meet the conditions required to enter the browser's preload list.

Browser vendors declared:

- Serve a valid certificate
- If you are listening on port 80, redirect all domains from HTTP to HTTPS on the same host. Serve all subdomains over HTTPS:
- In particular, you must support HTTPS for the www subdomain if a DNS record for that subdomain exists
- Serve an HSTS header on the base domain for HTTPS requests:
- The max-age must be at least 31536000 seconds (1 year)
- The includeSubDomains directive must be specified
- The preload directive must be specified
- If you are serving an additional redirect from your HTTPS site, that redirect must have the HSTS header (rather than the page it redirects to)

External References

HTTP Strict Transport Security (HSTS) HTTP Header

Wikipedia - HTTP Strict Transport Security Implementation
Check HSTS Preload status and eligibility

3. Out-of-date Version (jQuery)

Vulnerability Details

Netsparker identified the target web site is using jQuery and detected that it is out of date.

Impact

Since this is an old version of the software, it may be vulnerable to attacks.

Remedy

Please upgrade your installation of jQuery to the latest stable version.

Remedy References

Downloading jQuery
Known Vulnerabilities in this Version
.medium{fill:#FF9900;} jQuery Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') Vulnerability
In jQuery versions greater than or equal to 1.0.3 and before 3.5.0, passing HTML containing <option> elements from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.

Affected Versions
1.9.0 to 3.4.1

External References

CVE-2020-11023
.medium{fill:#FF9900;} jQuery Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') Vulnerability
In jQuery versions greater than or equal to 1.2 and before 3.5.0, passing HTML from untrusted sources - even after sanitizing it - to one of jQuery's DOM manipulation methods (i.e. .html(), .append(), and others) may execute untrusted code. This problem is patched in jQuery 3.5.0.

4. BREACH Attack Detected [Possible]

Vulnerability Details
Netsparker detected that BREACH (Browser Reconnaissance & Exfiltration via Adaptive Compression of Hypertext) attack is possible on this website.

Due to elements that make BREACH attack possible, SSL/TLS protected traffic remains vulnerable and can be attacked to uncover information from the website.

Regardless of which version of SSL/TLS you use, attacks are still possible. Attacks do not require TLS-layer compression and they can work against any cipher suite.

Impact
Even if you use an SSL/TLS protected connection, an attacker can still view the victim’s encrypted traffic

and cause the victim to send HTTP requests to the vulnerable web server (by using invisible frames). Following these steps, an attacker could steal information from the website and do the following:

- Inject partial plaintext they have uncovered into a victim's requests
- Measure the size of encrypted traffic

Remedy

Netsparker reported a Possible BREACH Attack issue because the target web page meets the following conditions that facilitate it:

- Served from a server that uses HTTP-level compression (ie. gzip)
- Reflects user-input in the HTTP response bodies
- Contains sensitive information (such as a CSRF token) in HTTP response bodies To mitigate the issue, we recommend the following solutions:

1. If possible, disable HTTP level compression
2. Separate sensitive information from user input
3. Protect vulnerable pages with CSRF token. The SameSite Cookie attribute will mitigate this issue, because to exploit this issue an attacker forces the victim to visit a target website using invisible frames. With the SameSite cookie attribute added, cookies that belong to the target won't be sent with a request that does not include top level navigation.
4. Hide the length of the traffic by adding a random number of bytes to the responses.
5. Add in a rate limit, so that the page maximum is reached five times per minute.

External References

Browser Reconnaissance & Exfiltration via Adaptive Compression of Hypertext
Using the Same-Site Cookie Attribute to Prevent CSRF Attacks

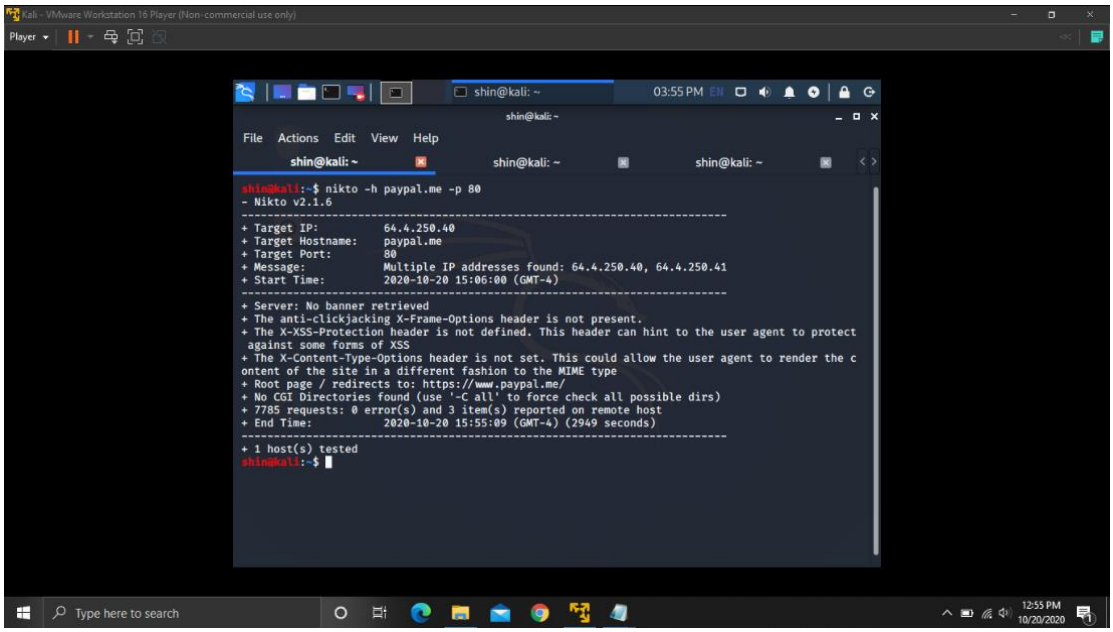
Nikto

After scanning sub domains using Netsparker, I used Nikto to scan another sub doamin set for port 80 and port 443.

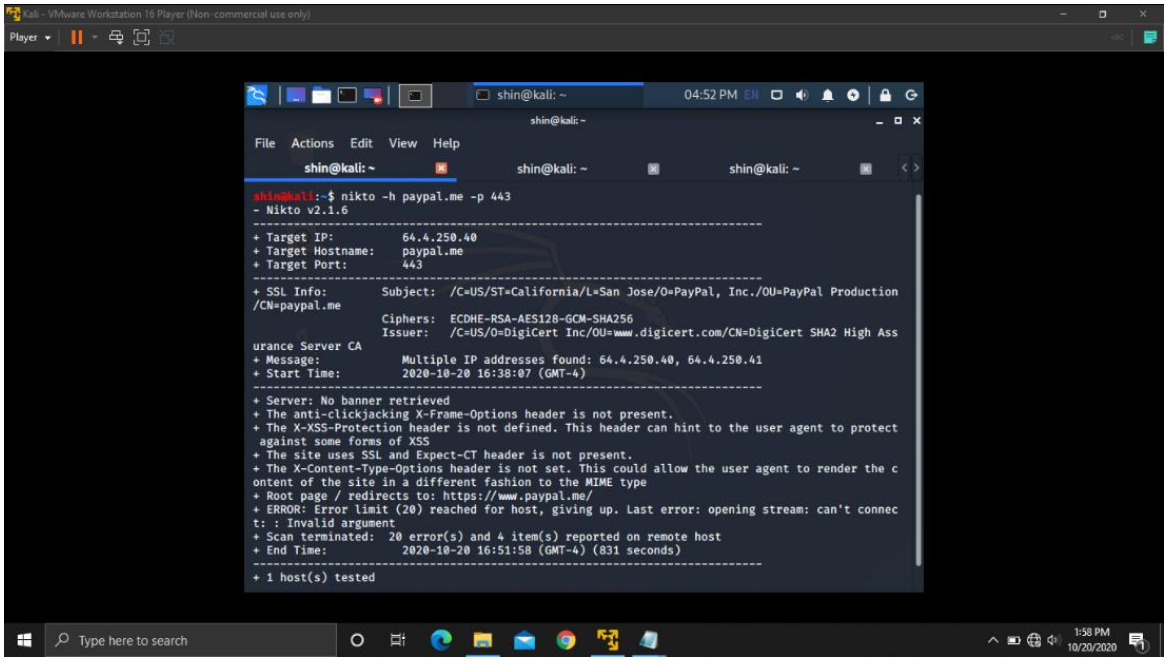
They are,

- paypal.me
- paydiant.com
- paypalobjects.com
- prequal.swiftfinancial.com
- partner.swiftfinancial.com
- decision.swiftfinancial.com
- swiftcapital.com
- loanbuilder.com
- swiftfinancial.com
- api.swiftfinancial.com
- my.swiftfinancial.com
- api.loanbuilder.com
- my.loanbuilder.com

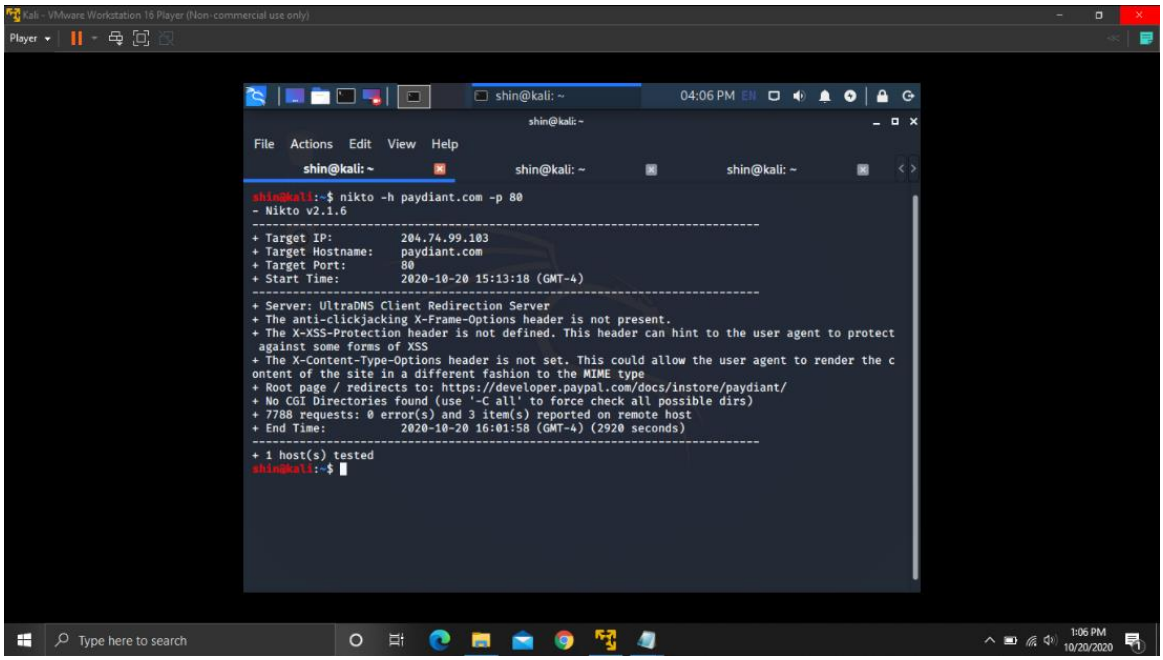
paypal.me (port 80)



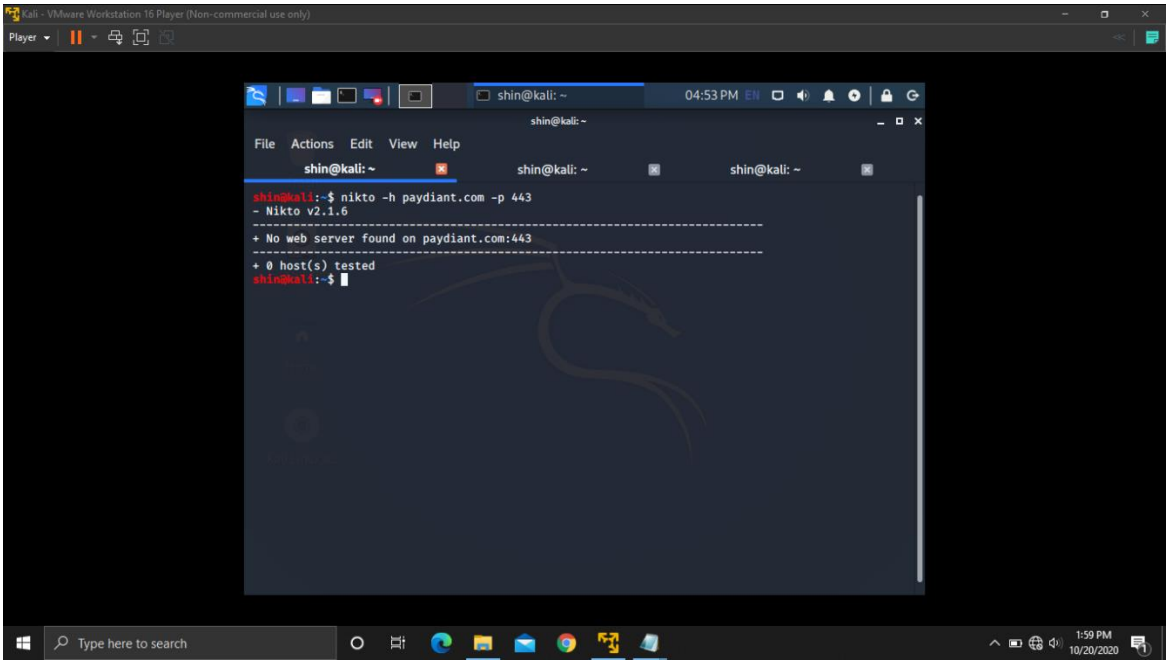
paydiant.com (port 443)



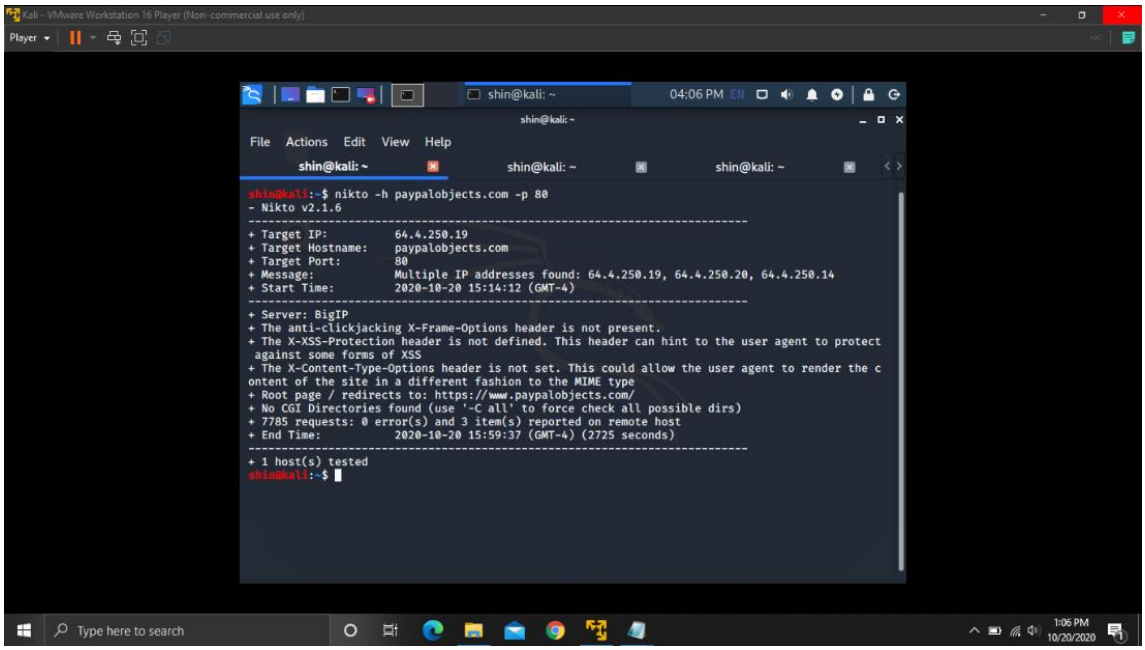
paydiant.com (port 80)



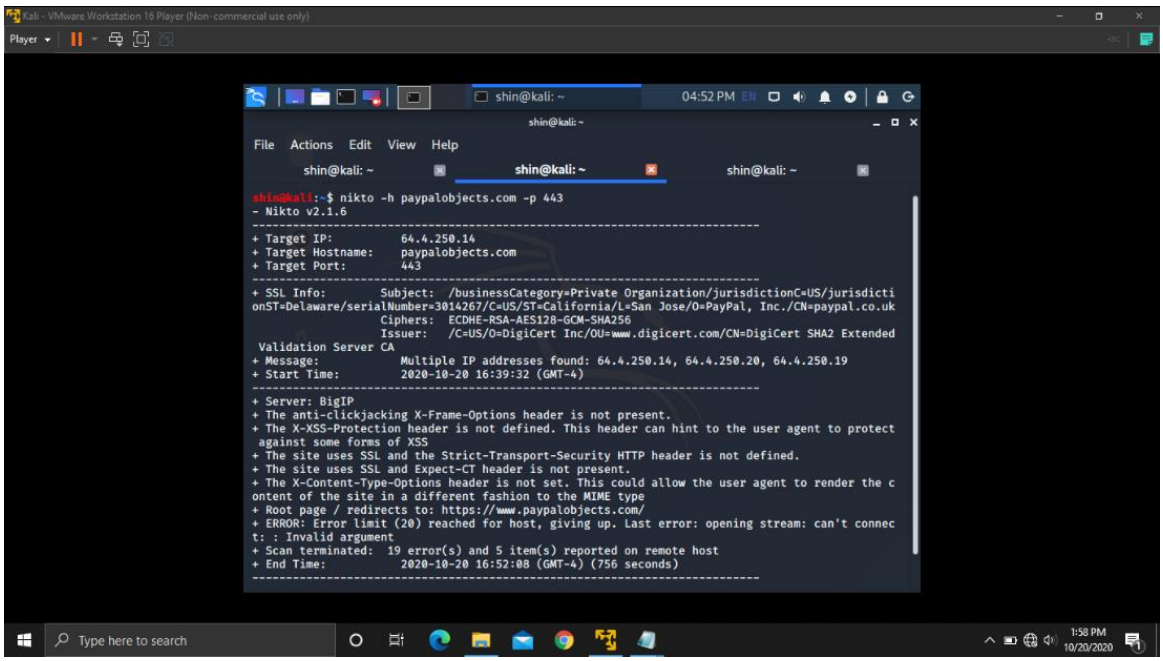
paydiant.com (port 443)



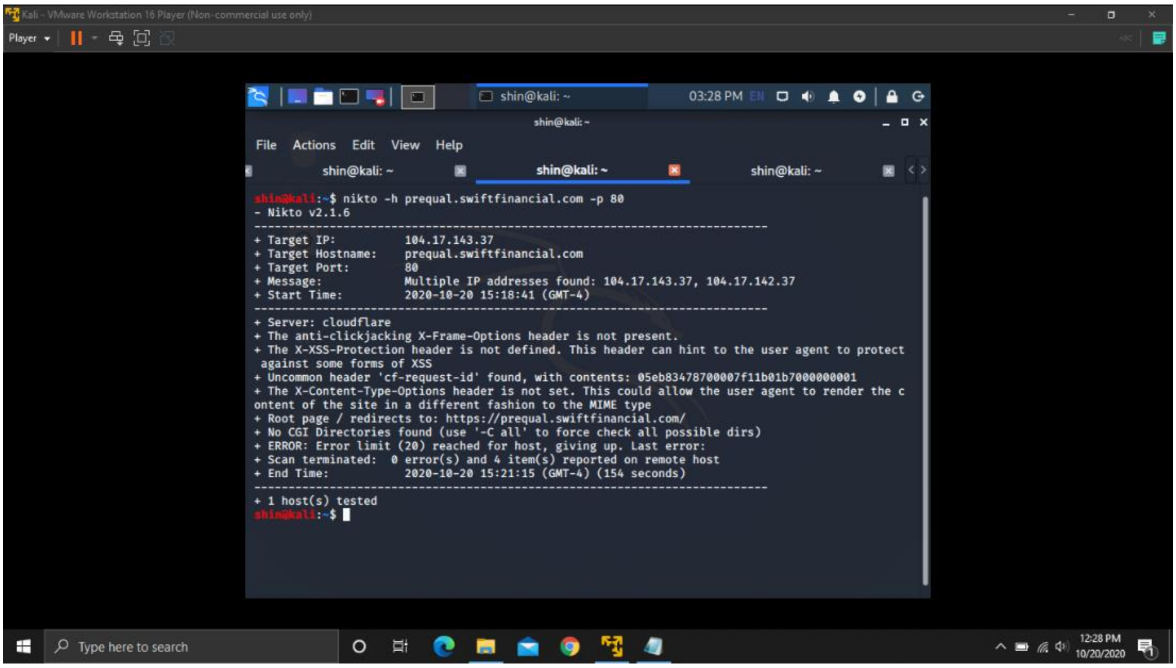
paypalobjects.com (port 80)



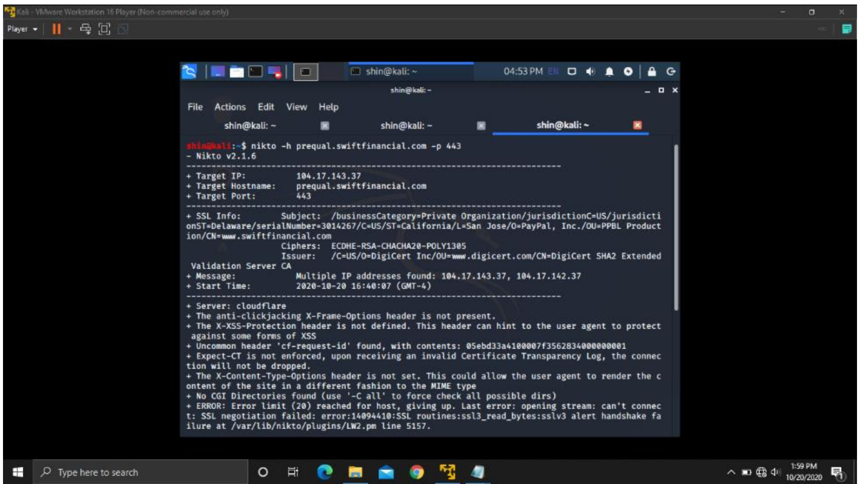
paypalobjects.com (port 443)

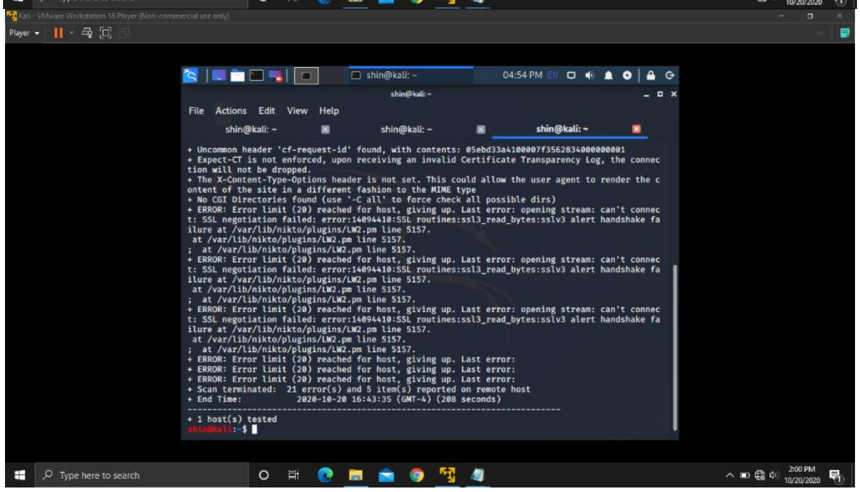


prequal.swiftfinancial.com (port 80)

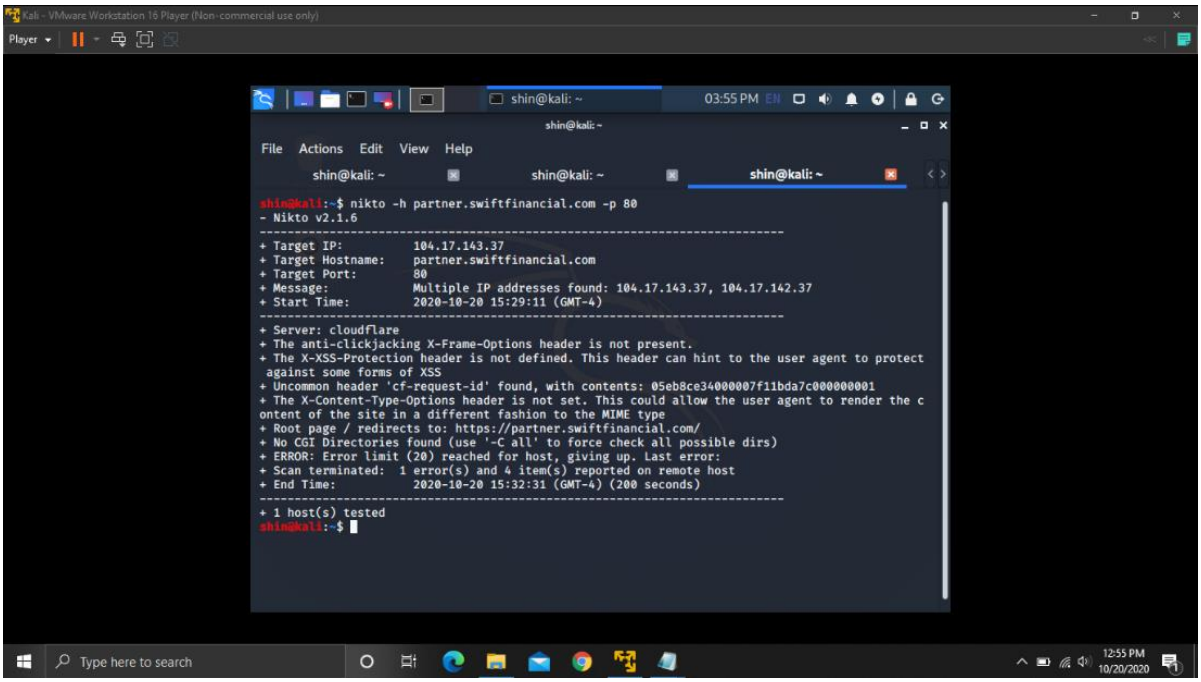


prequal.swiftfinancial.com (port 443)

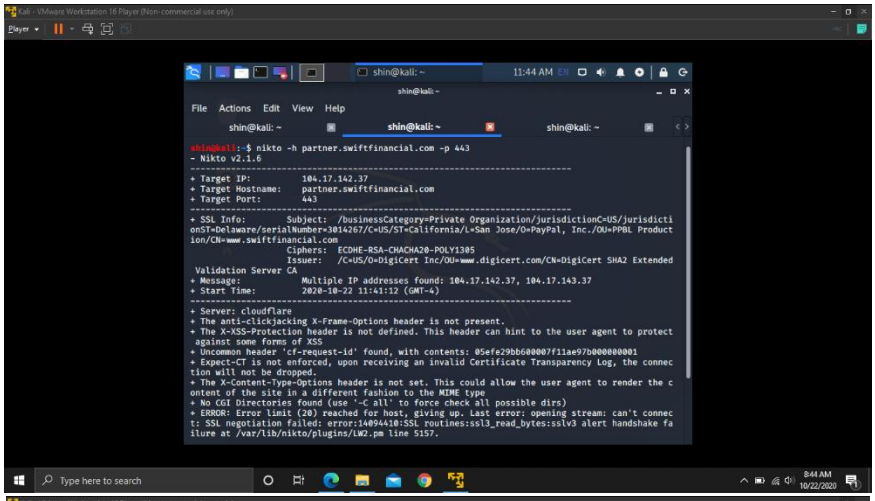




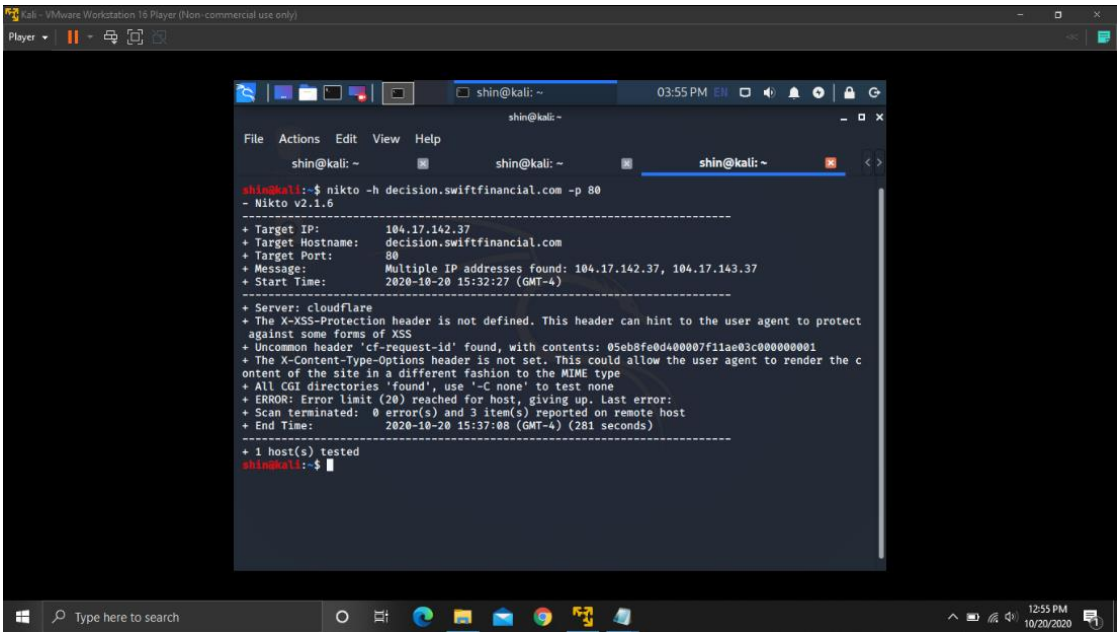
partner.swiftfinancial.com (port 80)



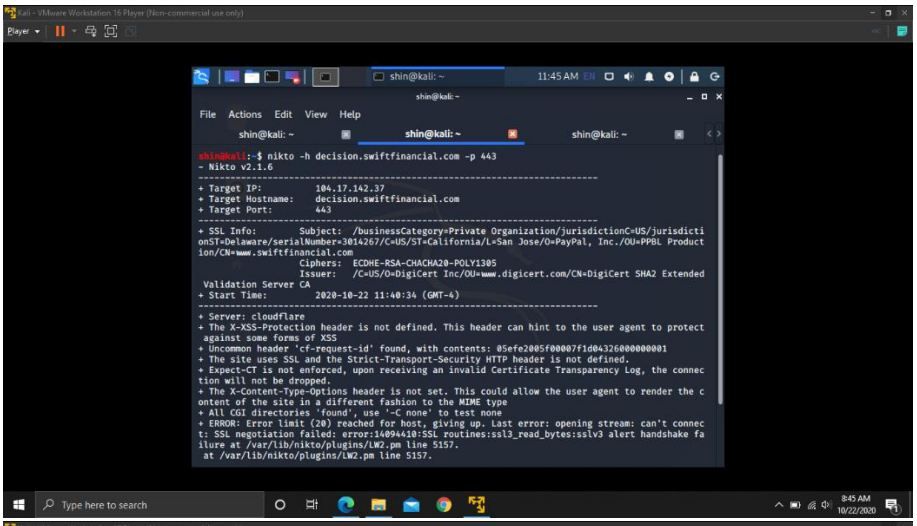
partner.swiftfinancial.com (port 443)

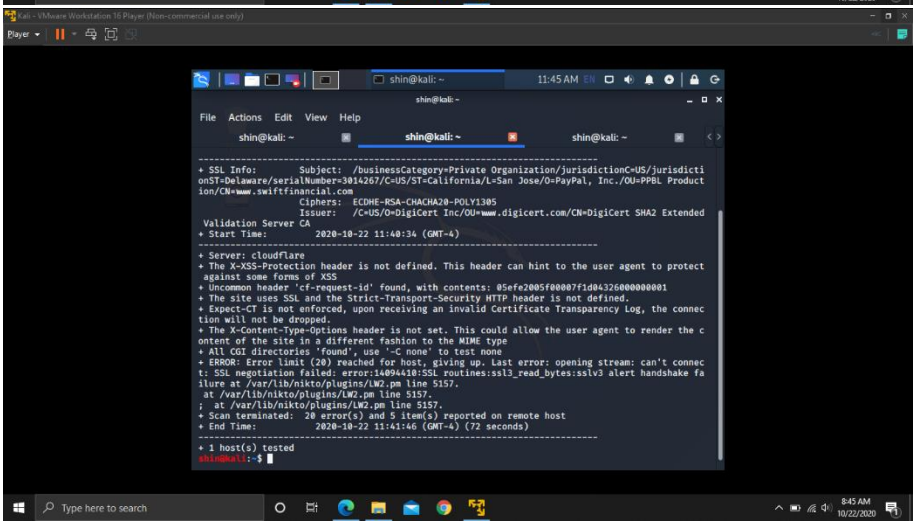


decision.swiftfinancial.com (port 80)

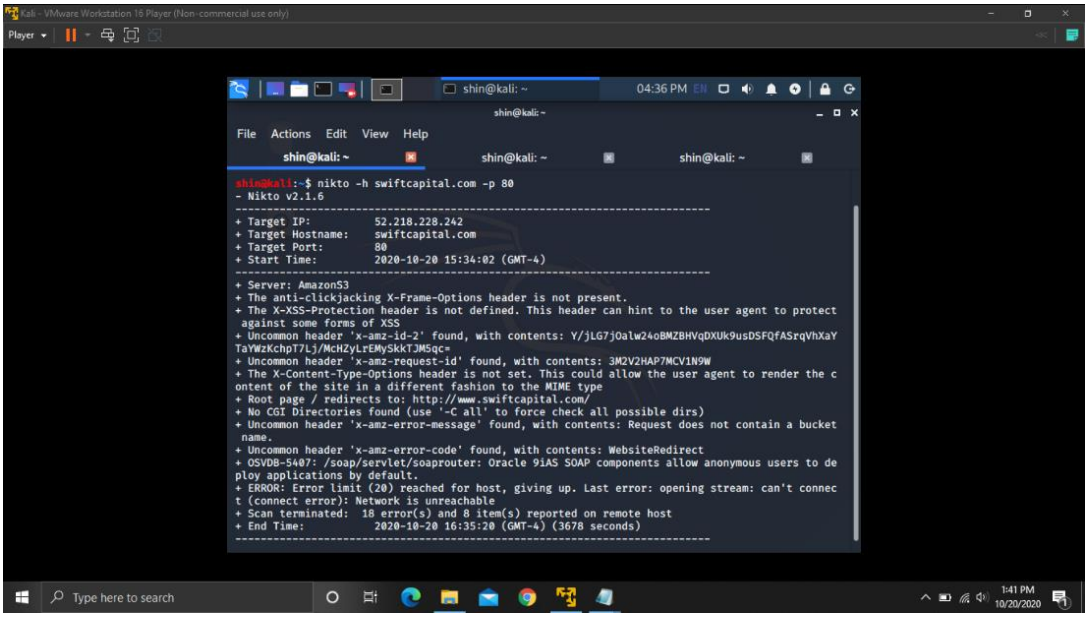


decision.swiftfinancial.com (port 443)

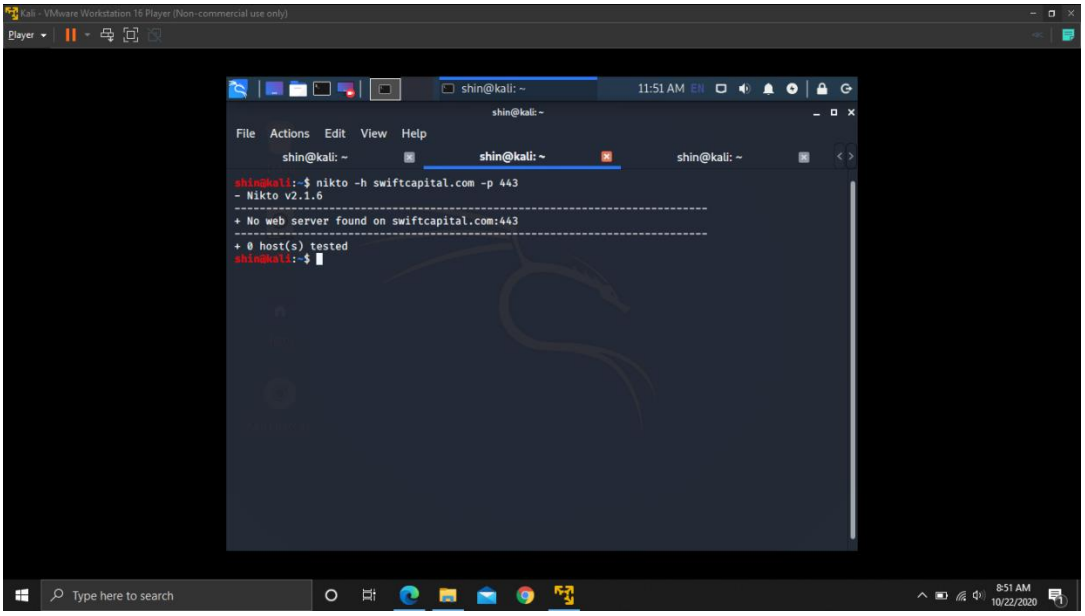




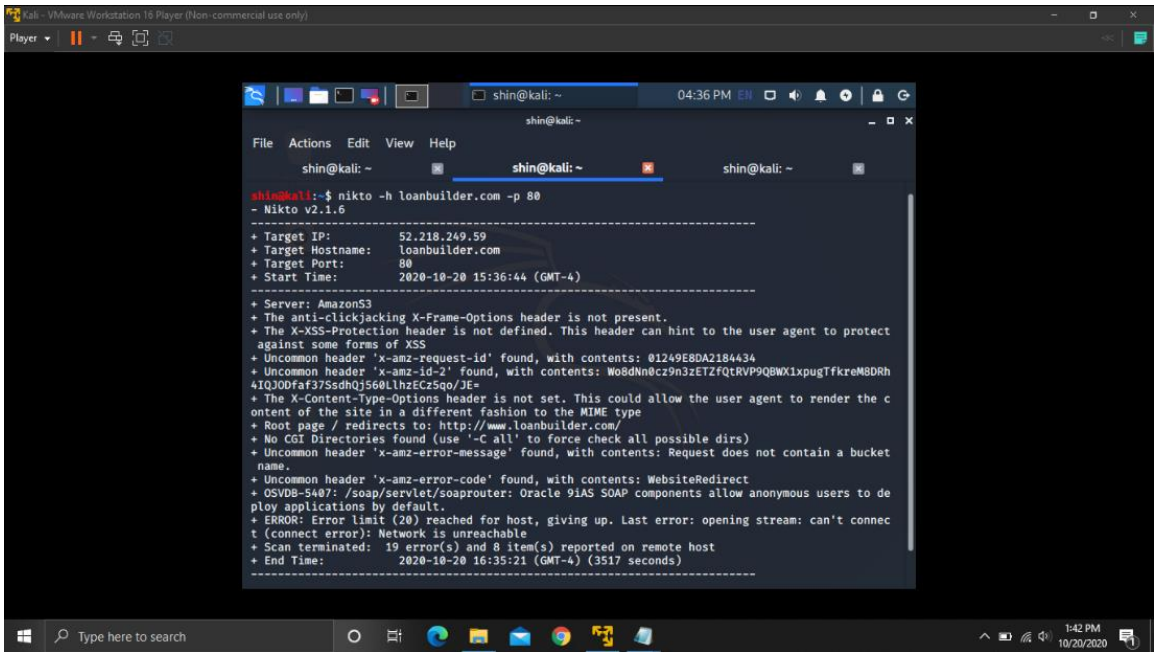
swiftcapital.com (port 80)



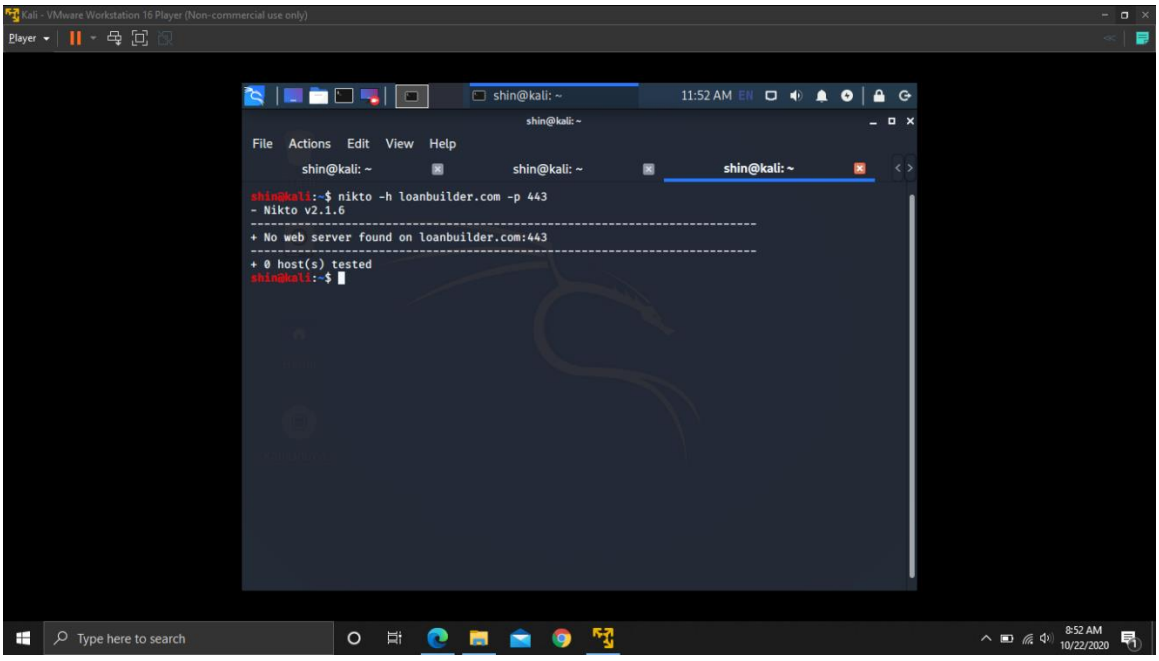
swiftcapital.com (port 443)



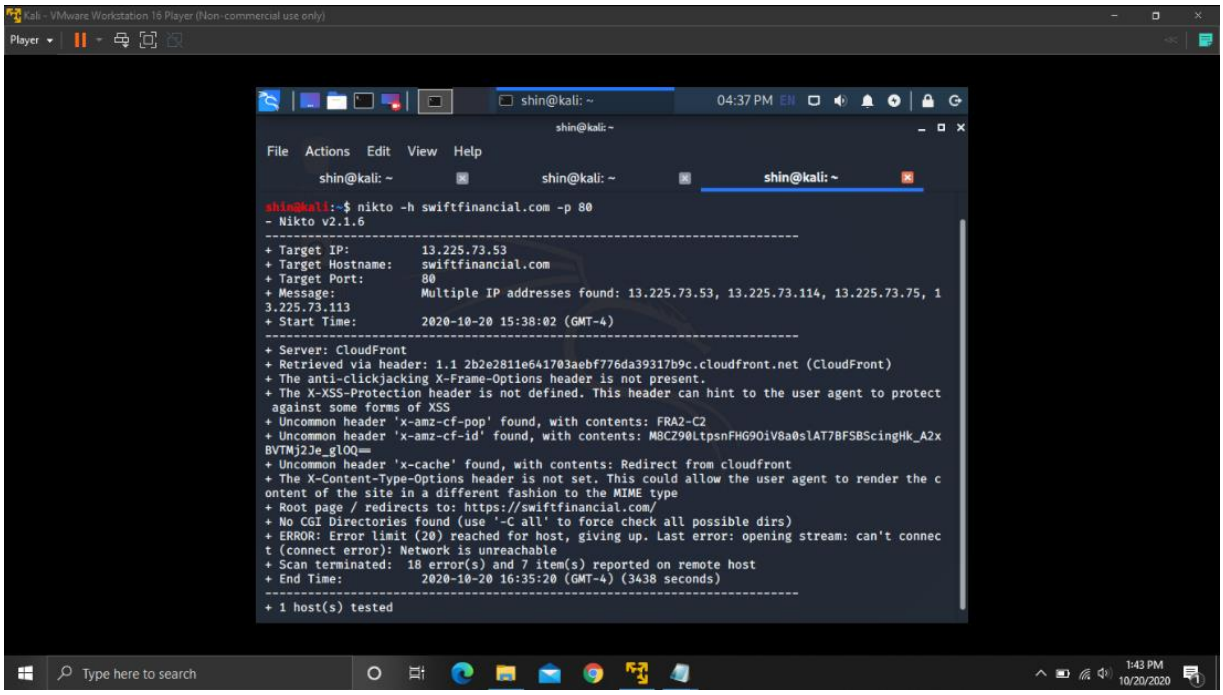
loanbuilder.com (port 80)



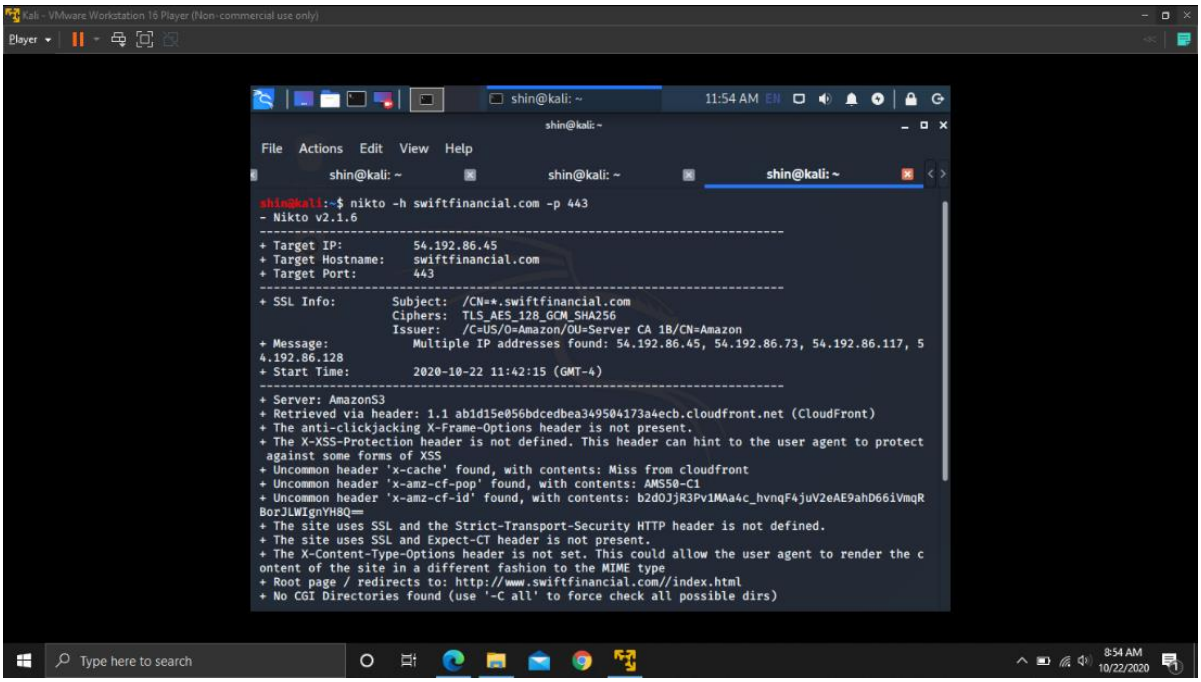
loanbuilder.com (port 443)

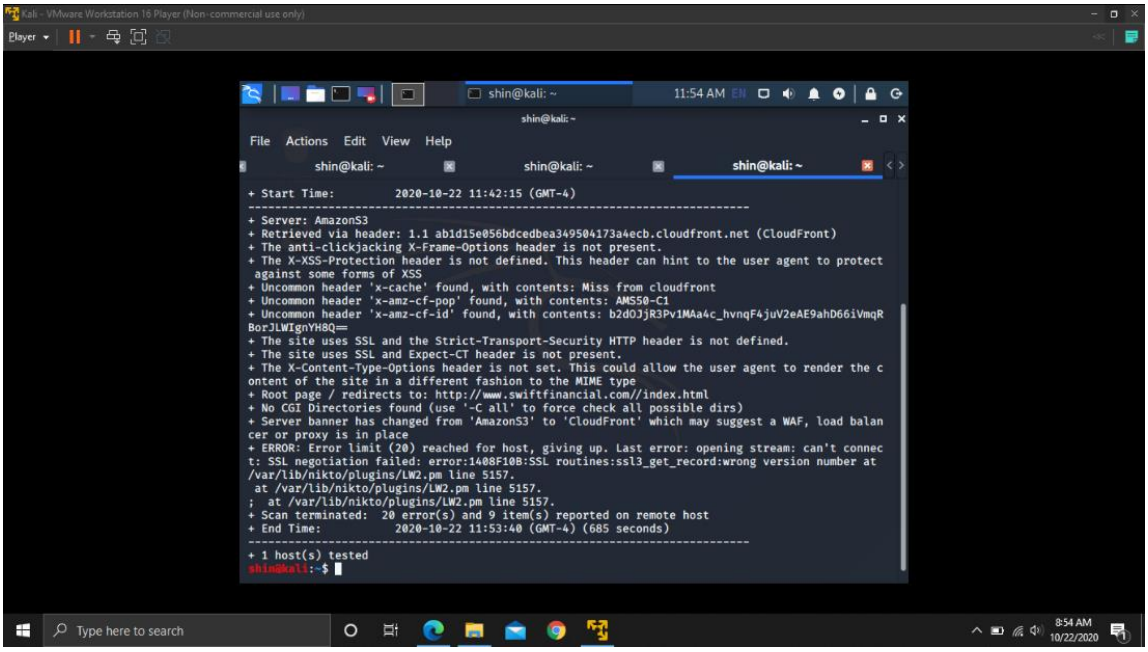


swiftfinancial.com (port 80)

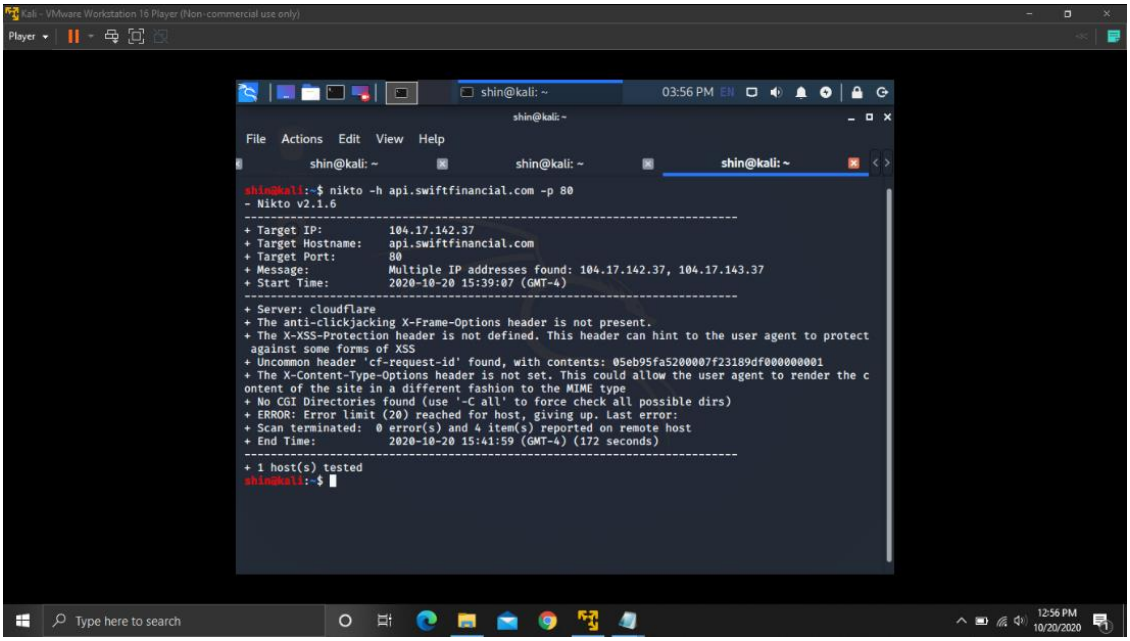


swiftfinancial.com (port 443)

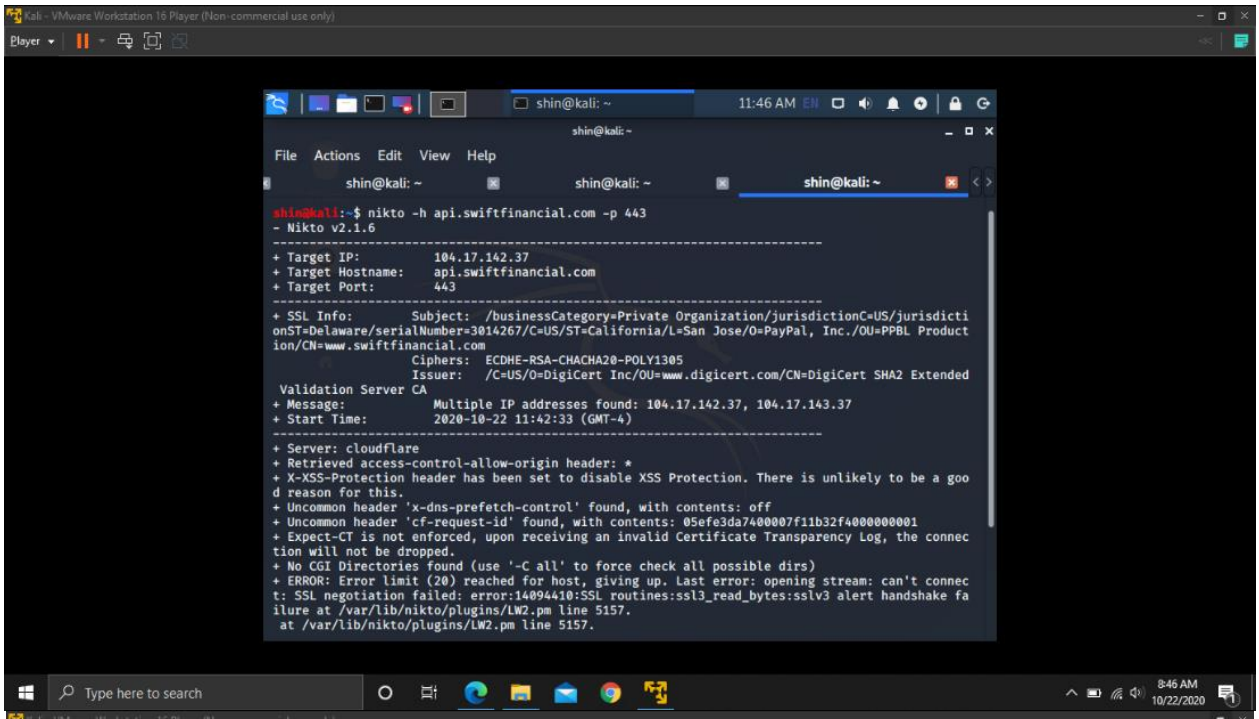


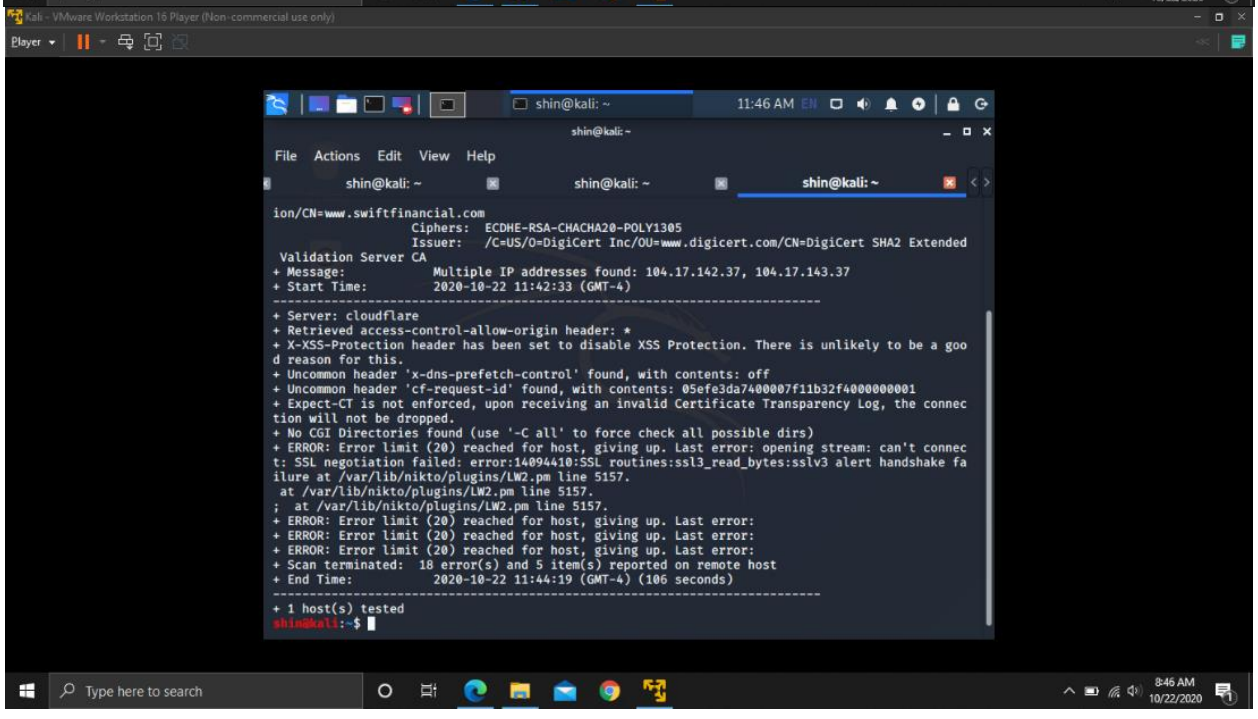


api.swiftfinancial.com (port 80)

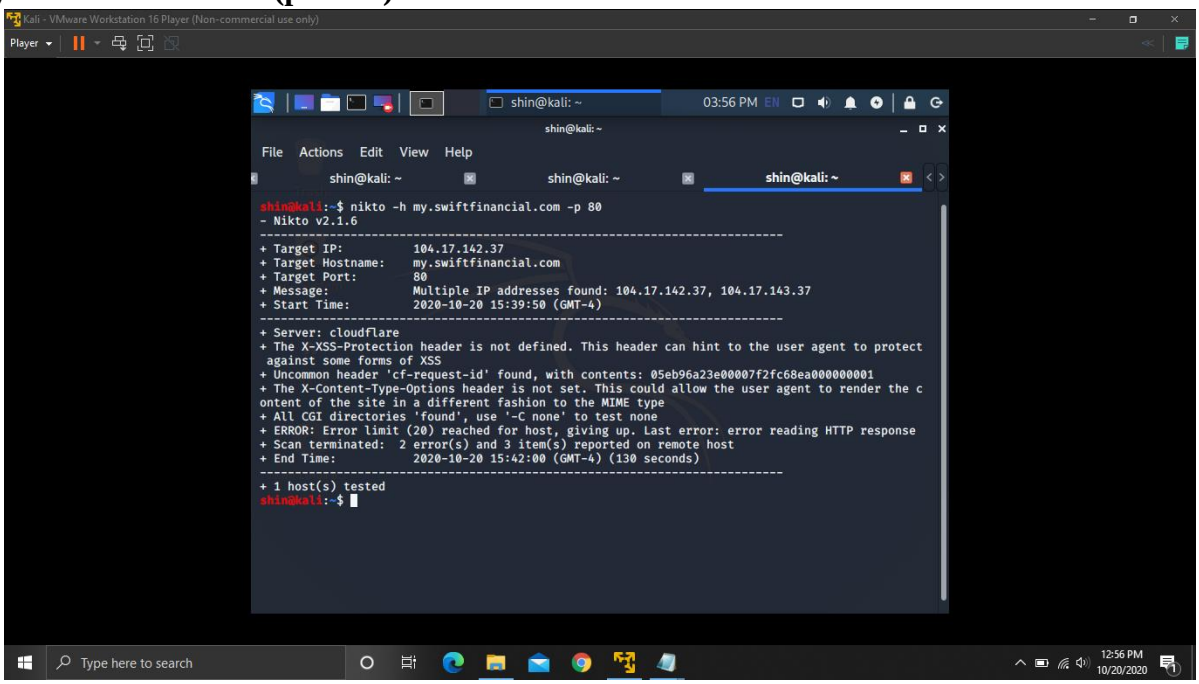


api.swiftfinancial.com (port 443)

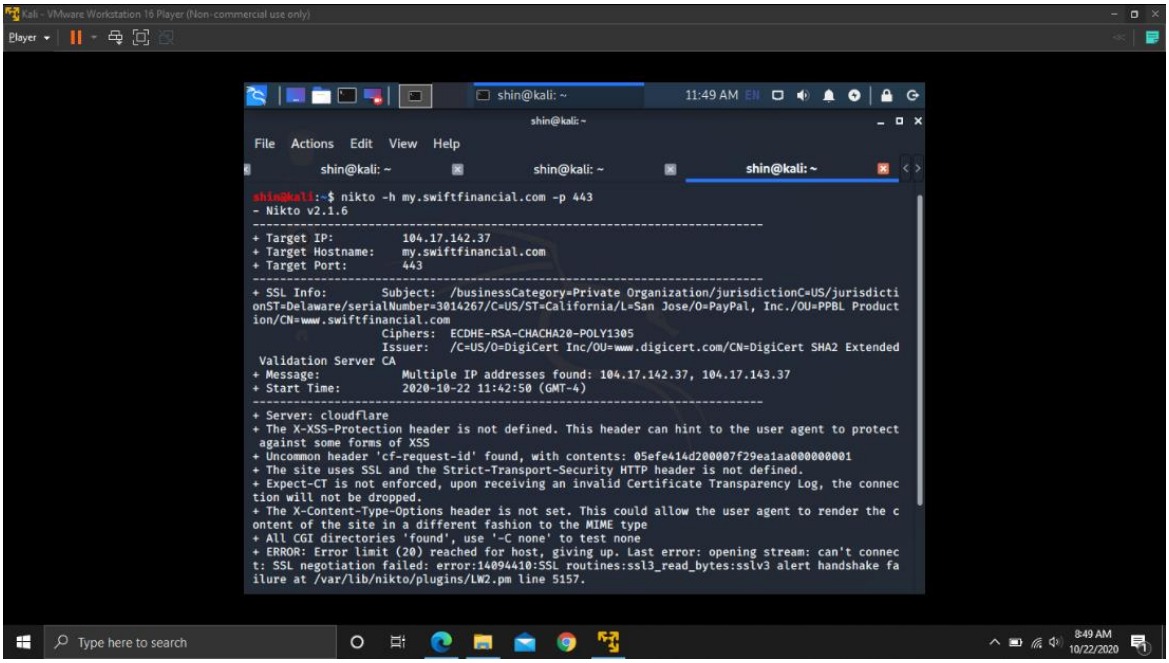


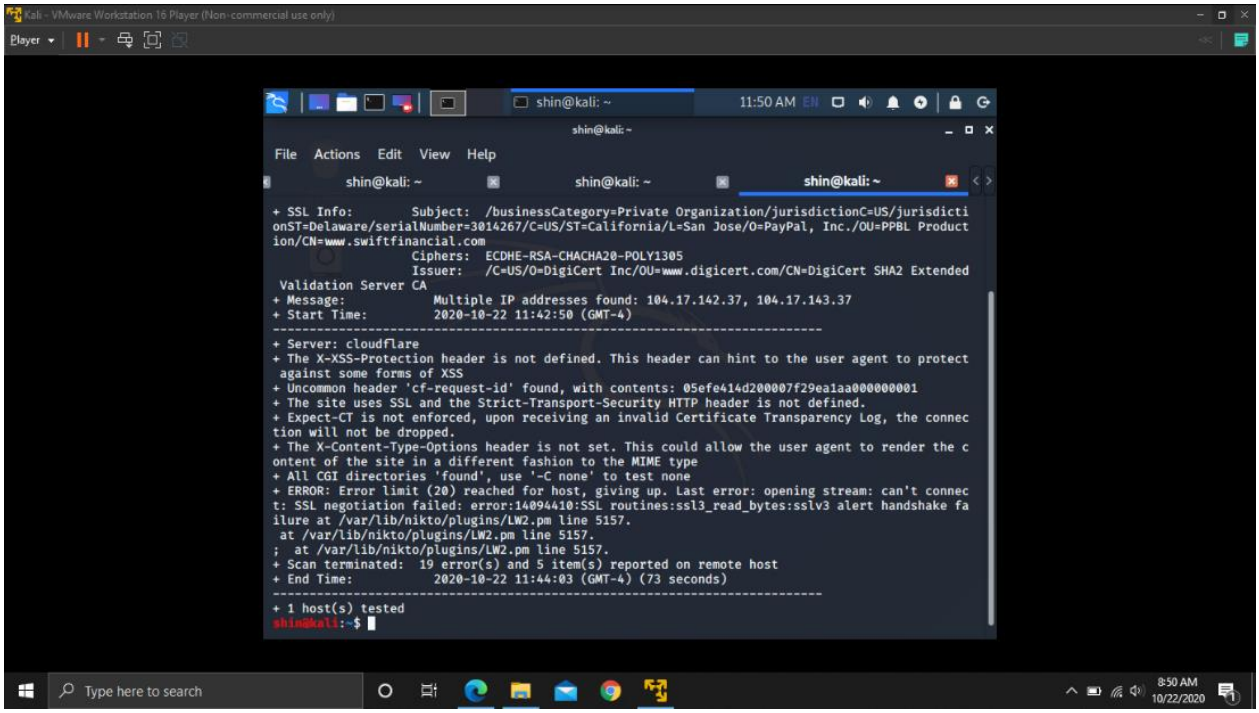


my.swiftfinancial.com (port 80)

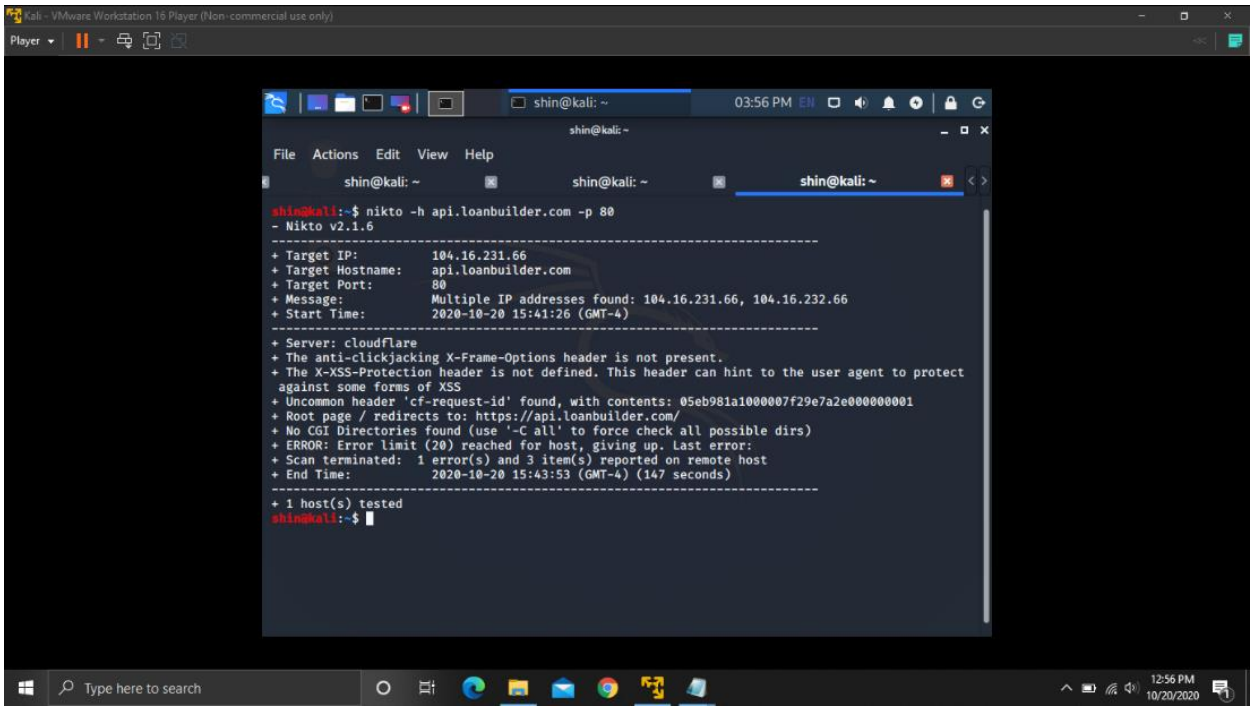


my.swiftfinancial.com (port 443)

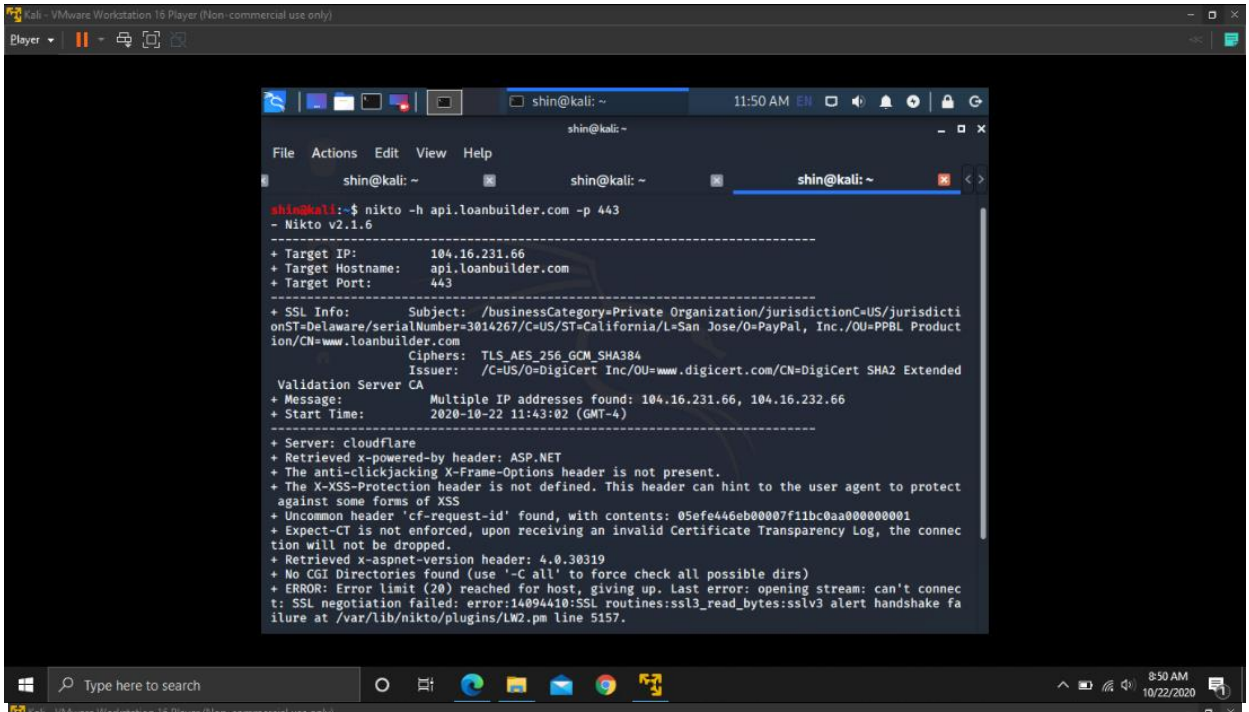


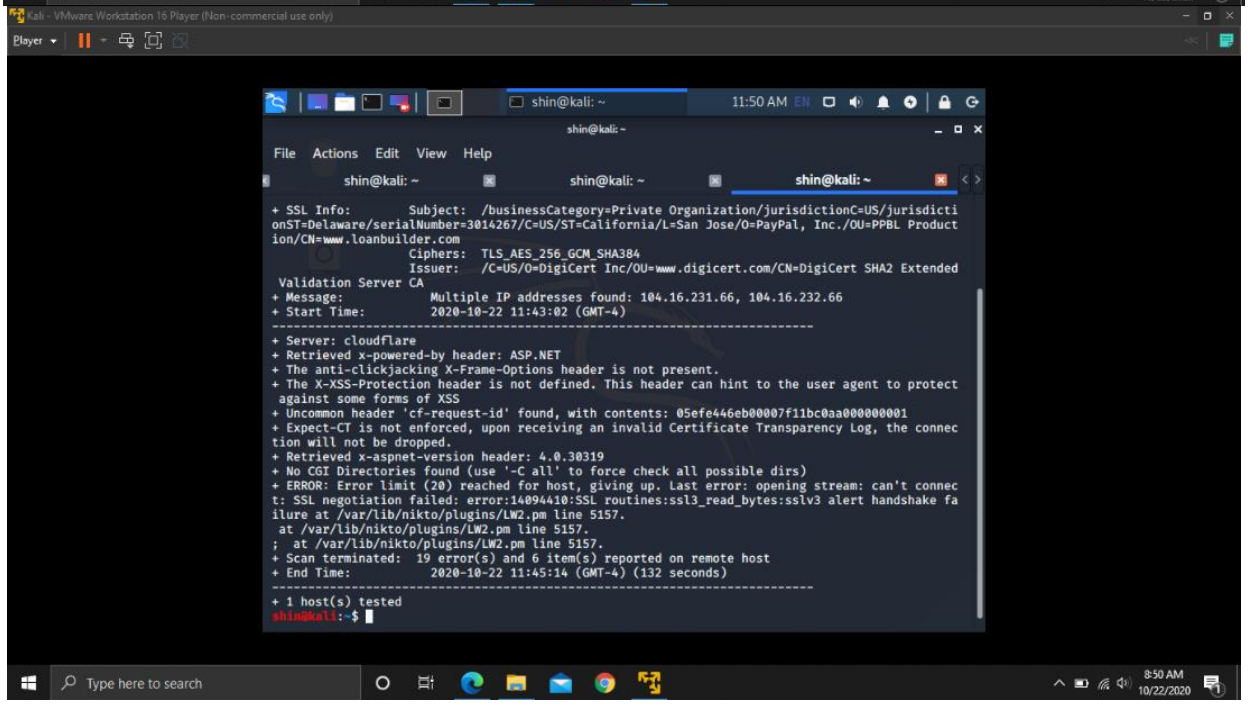


api.loanbuilder.com (port 80)

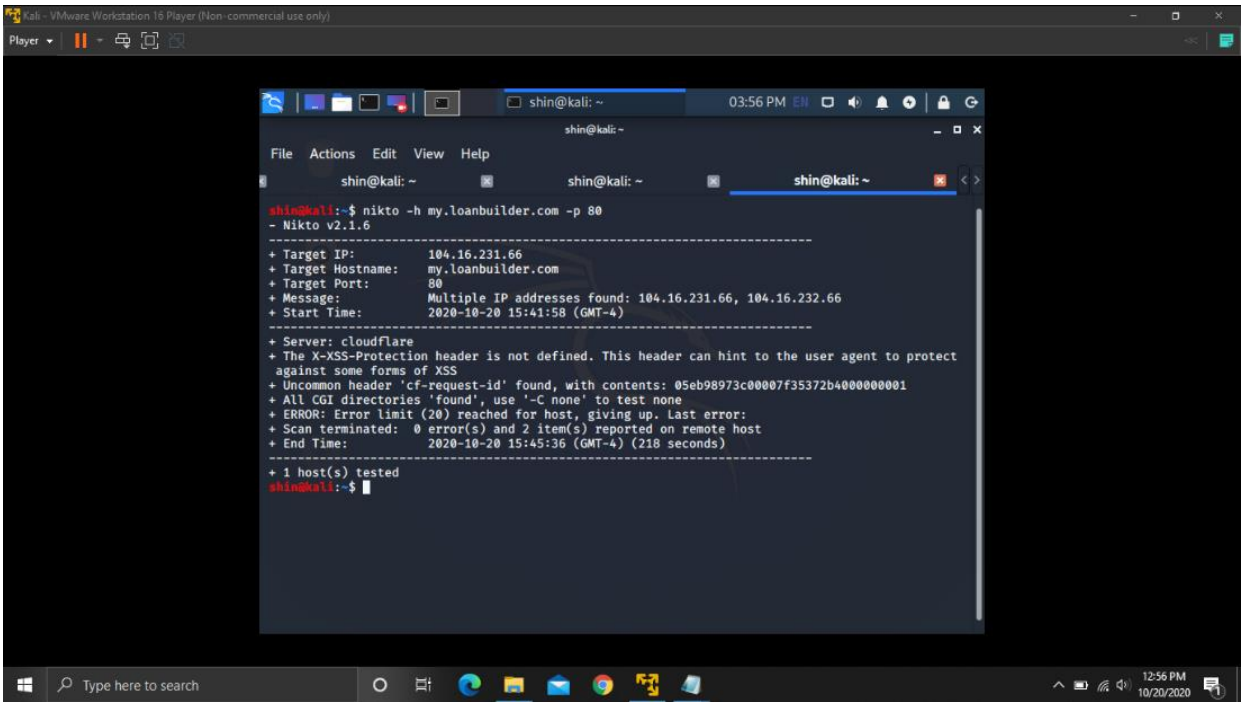


api.loanbuilder.com (port 443)

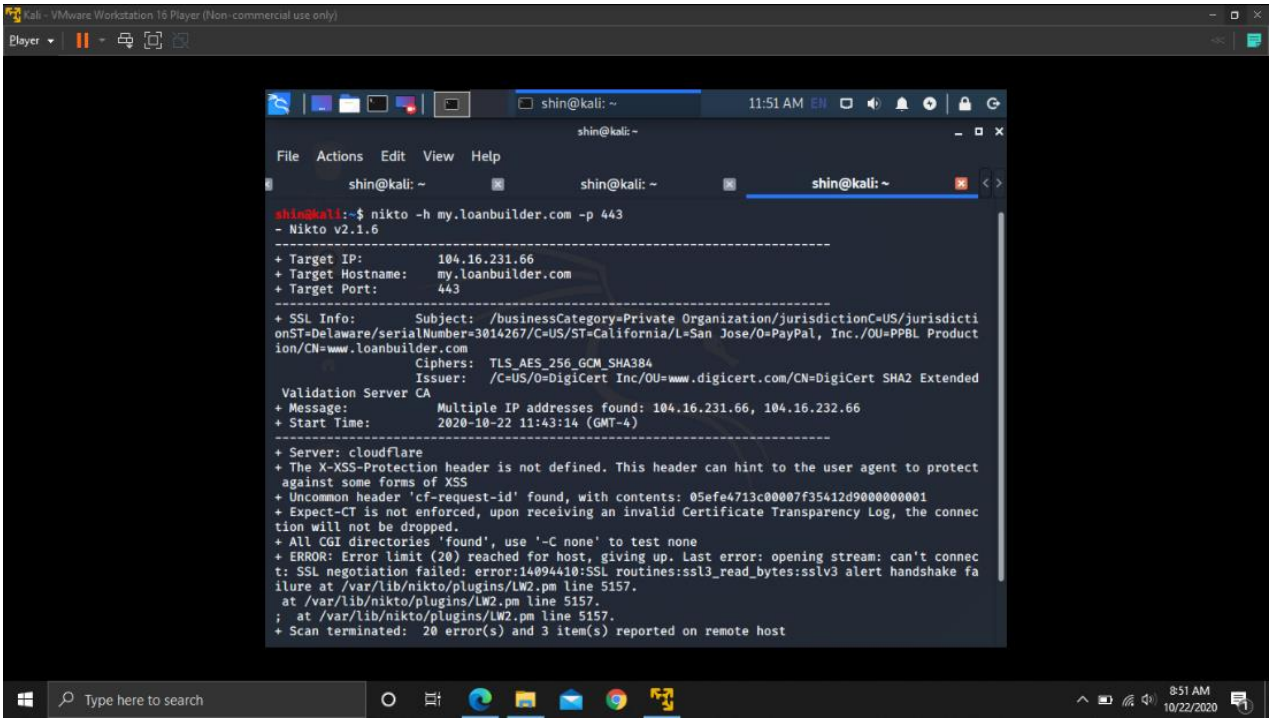


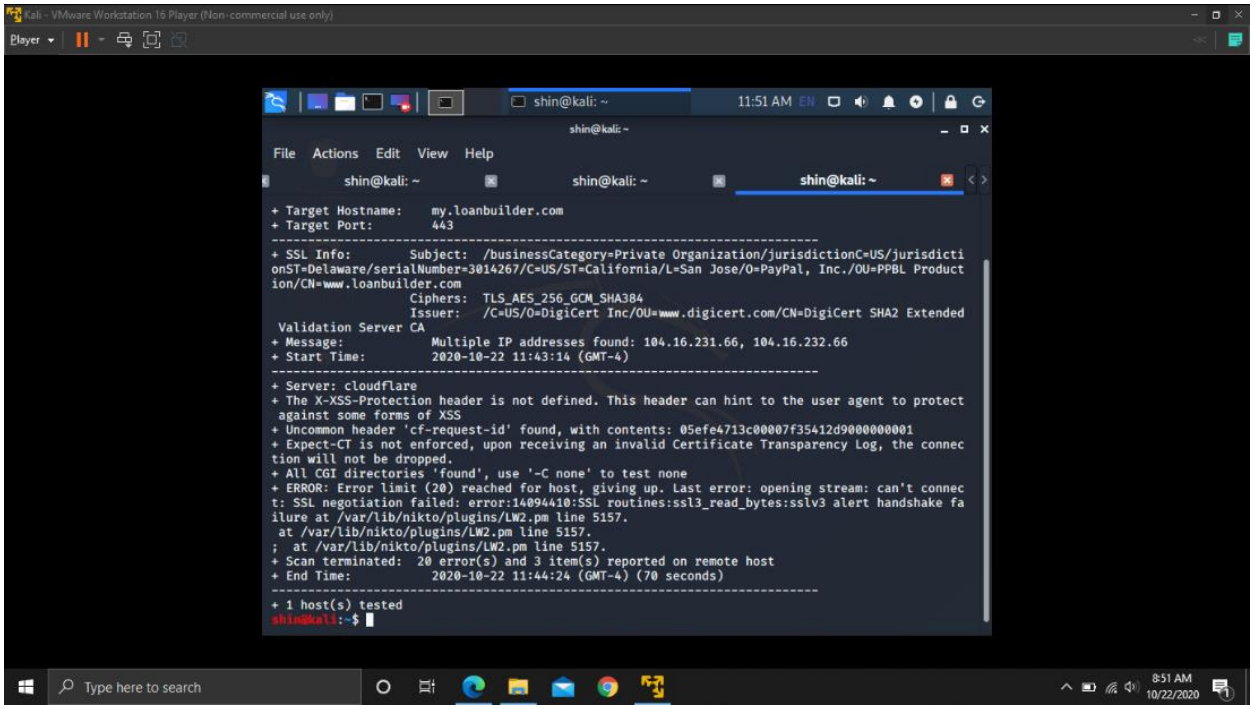


my.loanbuilder.com (port 80)



my.loanbuilder.com (port 443)





1. **The anti-clickjacking X-Frame-Options header is not present:**

Clickjacking, also known as a “UI redress attack”, is when an attacker uses multiple transparent or opaque layers to trick a user into clicking on a button or link on another page when they were intending to click on the top-level page. Thus, the attacker is “hijacking” clicks meant for their page and routing them to another page, most likely owned by another application, domain, or both

Severity: Low
CWE-693

2. **The X-XSS-Protection header is not defined. This header can hint to the user agent to protect against some forms of XSS:**

It means that this website could be at risk of a Cross-site Scripting (XSS) attacks.

Severity: Low

CWE: 693

3. **The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type:**

This means that this website could be at risk of a MIME-sniffing attacks

Severity: Low

CWE-16

4. **Missing 'Expect-CT' Header:**

Severity: Low

CWE: 693

5. **Missing 'X-XSS-Protection' Header:**

Severity: Low

CWE: 693

References:

Tenable®. 2020. *Tenable® - The Cyber Exposure Company*. [online] Available at: <https://www.tenable.com/> [Accessed 23 October 2020].

Owasp.org. 2020. *OWASP Foundation | Open Source Foundation For Application Security*. [online] Available at: <https://owasp.org/> [Accessed 23 October 2020].

Netsparker.com. 2020. *Web Vulnerability & Security Checks | Netsparker*. [online] Available at: <https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/> [Accessed 23 October 2020].

Academy, W. and scripting, C., 2020. *What Is Cross-Site Scripting (XSS) And How To Prevent It? | Web Security Academy*. [online] Portswigger.net. Available at: <https://portswigger.net/web-security/cross-site-scripting> [Accessed 23 October 2020].

2020. [online] Available at: <https://www.acunetix.com/websitesecurity/cross-site-scripting/> [Accessed 23 October 2020].

Netsparker.com. 2020. *Missing X-Frame-Options Header | Netsparker*. [online] Available at: <https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-x-frame-options-header/> [Accessed 23 October 2020].

Final Video Link:
https://drive.google.com/file/d/1L3KT7PY8zWwJ8KkrWQy4h1ZjuiR1ty1J/view?usp=sharing