A blue and orange logo

Description automatically generated

**IE2062 – Web Security**

**Year 2, Semester 2**

**Scanning report –**

**IT21831904 – K.M. Weerasinghe**

Table of Contents

[Scope of the target 3](#_Toc149509778)

[In scope and rewards 3](#_Toc149509779)

[Out of scope 3](#_Toc149509780)

[Reconnaissance 4](#_Toc149509781)

[Subdomain enumeration 4](#_Toc149509782)

[Gathering information about the target 5](#_Toc149509783)

[Subdomain takeover 7](#_Toc149509784)

[Scanning 8](#_Toc149509785)

[Using Nmap for scanning 8](#_Toc149509786)

[Using final recon for scanning 9](#_Toc149509787)

[Vulnerability checking 11](#_Toc149509788)

[Vulnerabilities found using NIKTO 11](#_Toc149509789)

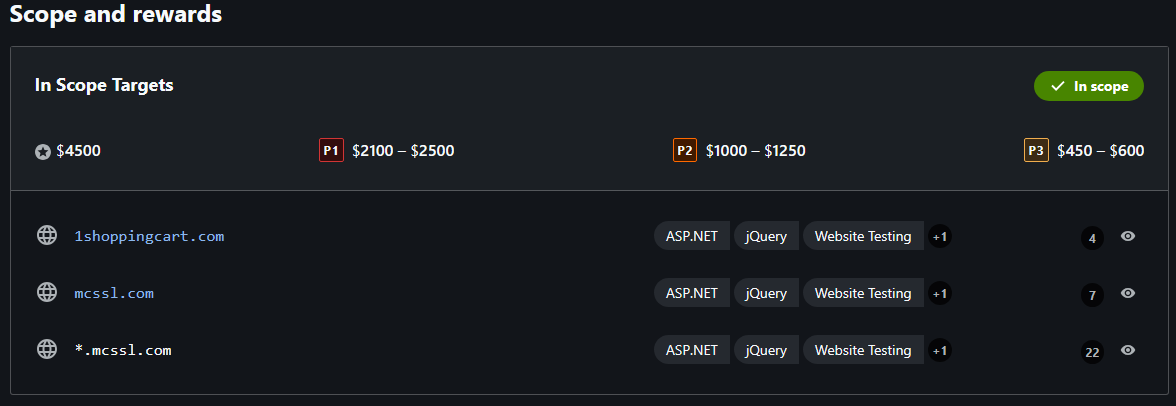
[Vulnerabilities found using rapidscan 12](#_Toc149509790)

# Scope of the target

A screenshot of a computer

Description automatically generated

## In scope and rewards



## Out of scope

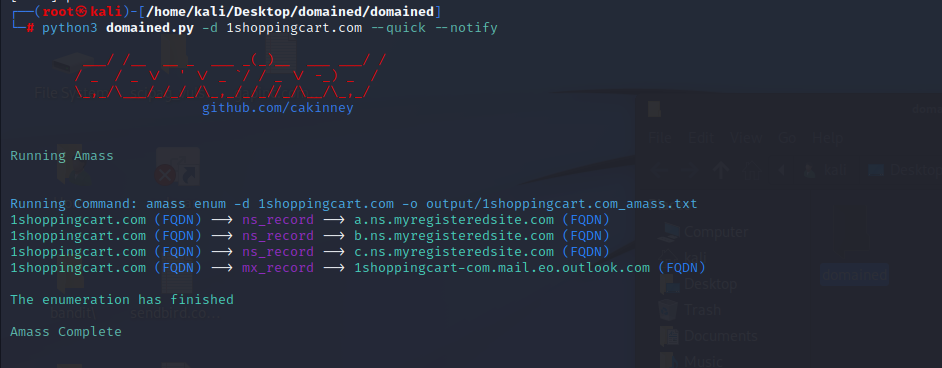
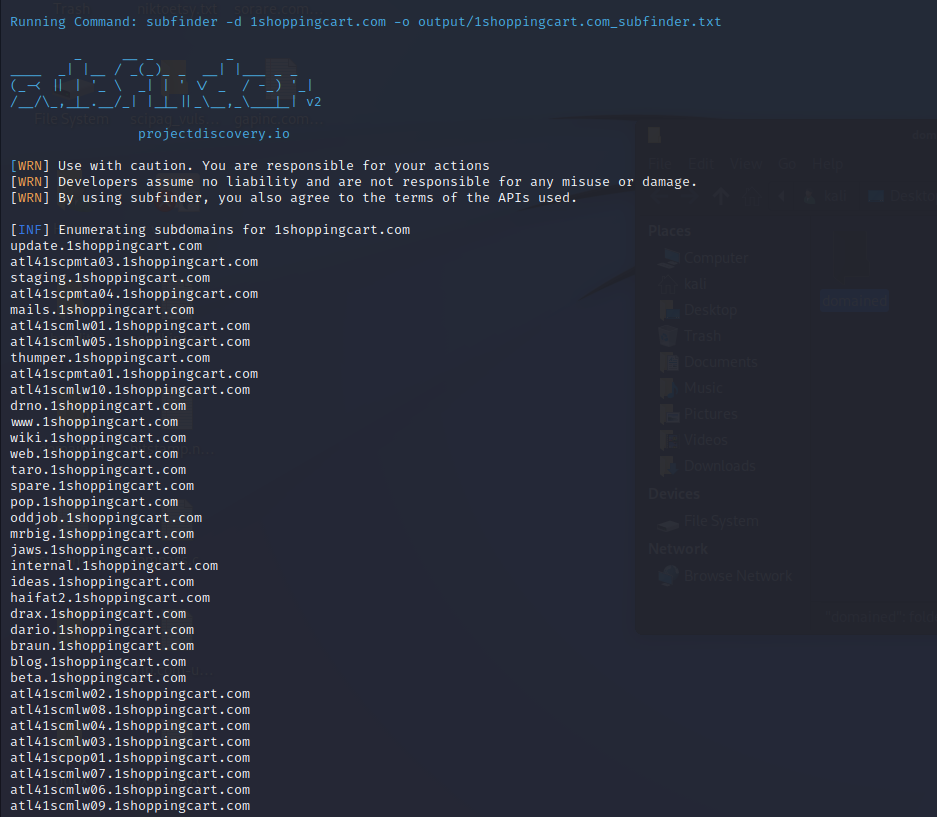
A black rectangular object with white text

Description automatically generated

# Reconnaissance

## Subdomain enumeration

* └─# python3 domained.py -d 1shoppingcart.com --quick --notify



## Gathering information about the target

* Security header checks

A screenshot of a computer

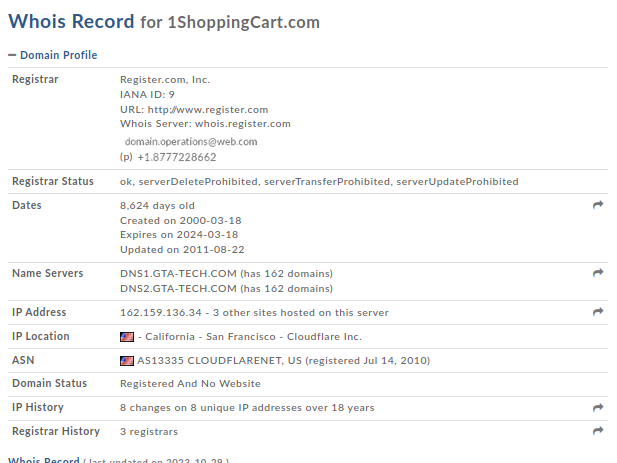
Description automatically generated

Missing headers

* + Content - Security-Policy Content Security Policy is an effective measure to protect your site from XSS attacks. By whitelisting sources of approved content, you can prevent the browser from loading malicious assets.
  + Referrer - Policy Referrer Policy is a new header that allows a site to control how much information the browser includes with navigations away from a document and should be set by all sites.
  + Permissions – Policy Permissions Policy is a new header that allows a site to control which features and APIs can be used in the browser.

Domain profile and whois records

* domain profile



* A screenshot of a computer

  Description automatically generatedWhois records

## Subdomain takeover

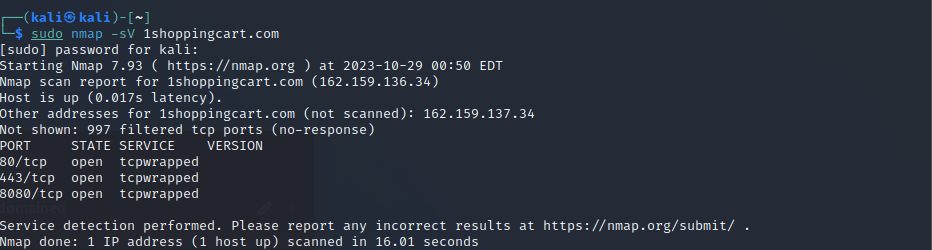
* A screen shot of a computer

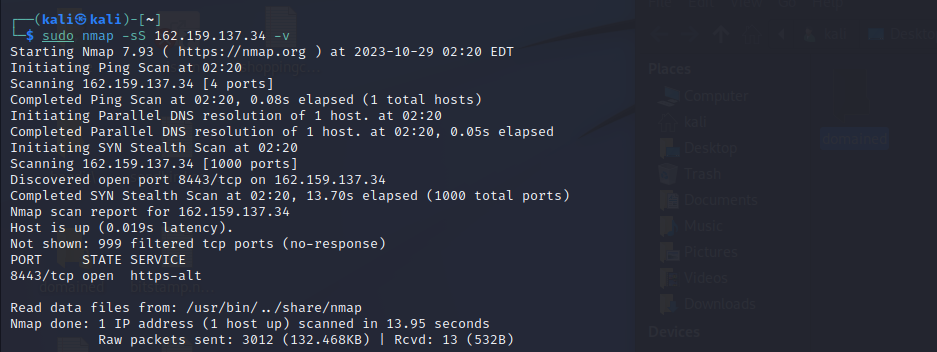
  Description automatically generated└─$ subjack -w Desktop/1shoppingcart.com-unique.txt -v
* Identifying if any of the found subdomains from the previous scans are vulnerable to takeover
* No vulnerable subdomains were found
* All subdomains appear to be secure
* No issues were identified in the subdomain analysis

# Scanning

## Using Nmap for scanning

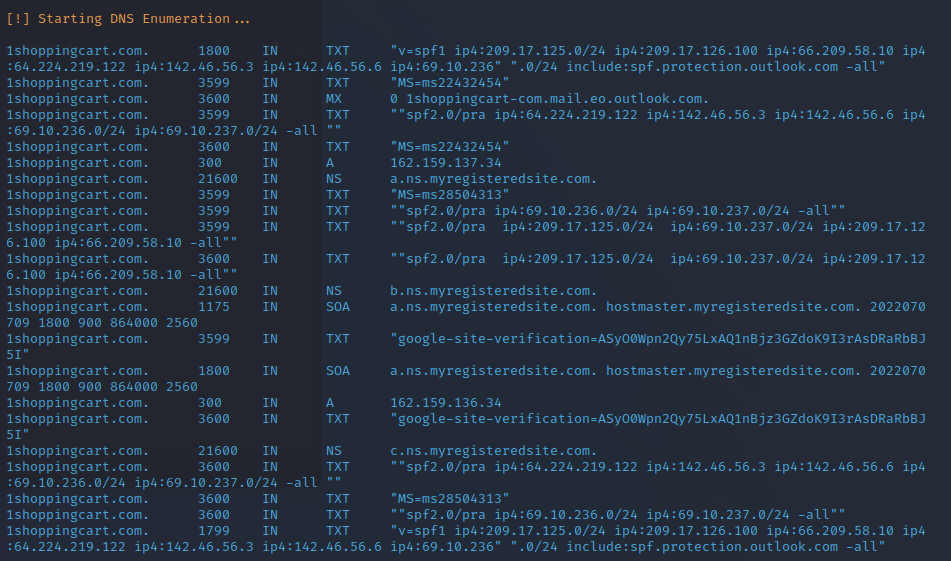
* └─$ sudo nmap -sV 1shoppingcart.com



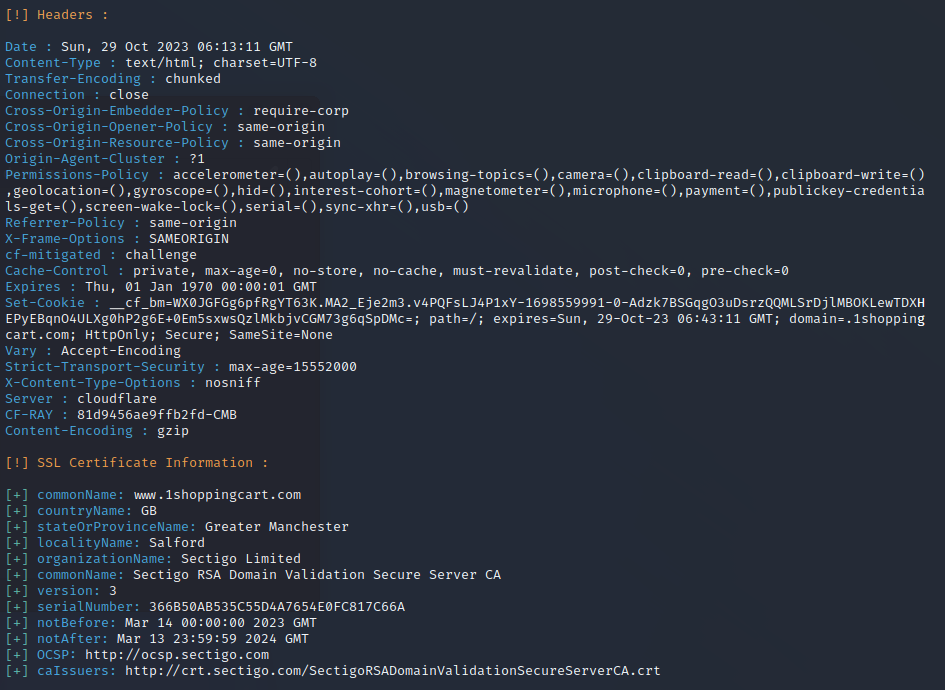
* └─$ sudo nmap -sS 162.159.137.34 -v

## Using final recon for scanning

DNS records gathered



Headers and SSL certificates information



# Vulnerability checking

## Vulnerabilities found using NIKTO

* └─$ nikto -h 1shoppingcart.com

- Nikto v2.1.6

---------------------------------------------------------------------------

+ Target IP: 162.159.136.34

+ Target Hostname: 1shoppingcart.com

+ Target Port: 80

+ Message: Multiple IP addresses found: 162.159.136.34, 162.159.137.34

+ Start Time: 2023-10-29 00:50:58 (GMT-4)

---------------------------------------------------------------------------

+ Server: cloudflare

+ The anti-clickjacking X-Frame-Options header is not present.

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

+ Root page / redirects to: https://1shoppingcart.com/

+ No CGI Directories found (use '-C all' to force check all possible dirs)

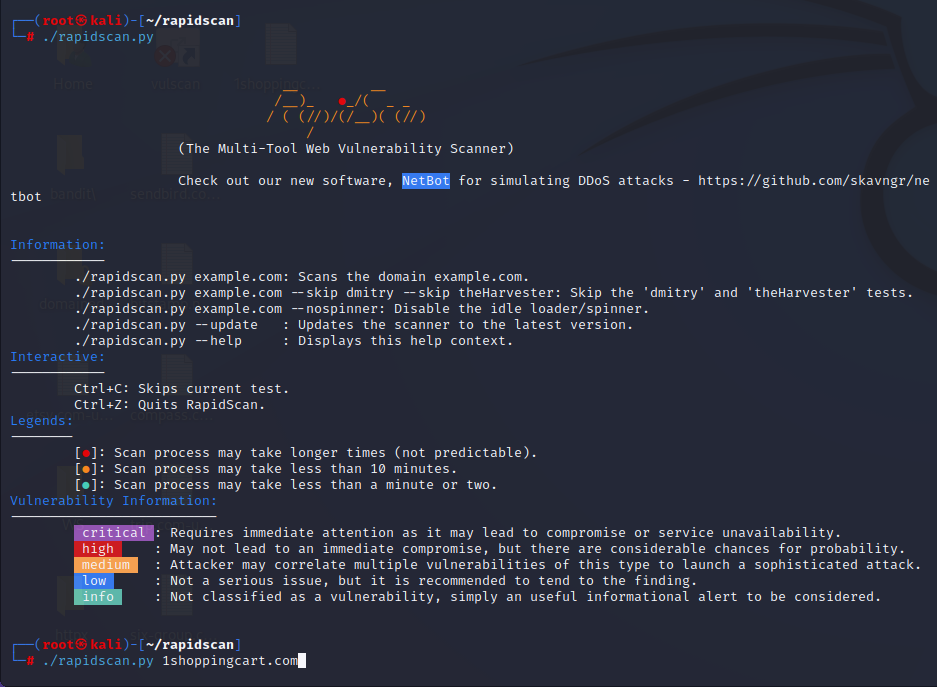
+ 7785 requests: 0 error(s) and 2 item(s) reported on remote host

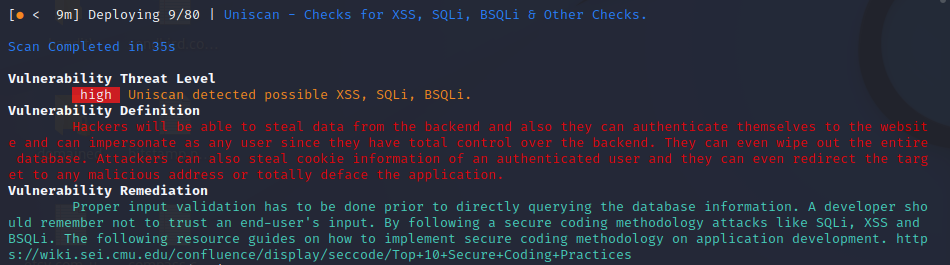
+ End Time: 2023-10-29 01:02:28 (GMT-4) (690 seconds)

---------------------------------------------------------------------------

+ 1 host(s) tested

## Vulnerabilities found using rapidscan

* └─# ./rapidscan.py
* └─# ./rapidscan.py 1shoppingcart.com

1. Vulnerability title - possible XSS, SQLi, BSQLi

* vulnerability threat level
  + high
* vulnerability description
  + Hackers will be able to steal data from the backend and, they can authenticate themselves to the website and can impersonate as any user since they have total control over the backend. They can even wipe out the entire database. Attackers can also steal cookie information of an authenticated user and they can even redirect the target to any malicious address or totally deface the application.
* Affected components
  + This vulnerability impacts the application's backend and data storage components
* Impact assessment
  + The impact of this vulnerability is high, as it allows for data theft, unauthorized access, potential data loss, and various forms of user manipulation.
* Steps to reproduce
  + Identify input fields or areas in the application that lack proper validation.
  + Inject malicious scripts or SQL queries into these input fields.
  + Observe how the application responds to these inputs, and whether it executes the injected code.
* Proposed mitigation or fix
  + Proper input validation needs to be implemented before querying the database information
  + A developer should remember not to trust an end users’ input
  + Follow secure coding methodology

1. A screen shot of a computer

   Description automatically generatedVulnerability title – SNMP service detected

* Vulnerability threat level
  + Medium
* Vulnerability description
  + Attackers will be able to read the community strings through the service and enumerate quite a bit of information from the target
  + Also there are multiple remote code execution and denial of service vulnerabilities related to SNMP services.
* Affected components
  + SNMP service
* Proposed mitigation or fix
  + Use a firewall to block the ports from the outside world

Vulnerability title - Secure client-initiated renegotiation vulnerability A screen shot of a computer error

Description automatically generated

* Vulnerability threat level
  + Medium
* Vulnerability description
  + This vulnerability, otherwise known as a Plain-Text Injection attack, allows Man-in-the-Middle (MiTM) attackers to insert data into HTTPS sessions, and potentially other types of sessions protected by TLS or SSL, by sending an unauthenticated request that is processed retroactively by a server in a post-renegotiation context.
* Affected components.
  + Secure client renegotiation that supports TLS or SSL protocols
* Proposed mitigation or fix
  + Disable Secure Client Initiated Renegotiation
  + Use strict certificate validation.

