SECURITY ASSESSMENT REPORT
SKYHIGH24.NL

EXTERNAL PENETRATION TEST

(Black Box: OSINT, Active & Passive reconnaissance)

Prepared by:

Violeta Baranovska

Report Version 1.0

Date: 27-11-2025

Location: Nijmegen, Netherlands

I. EXECUTIVE SUMMARY

This security assessment of SkyHigh24.nl was performed as an external black-box penetration test, focusing on OSINT, passive reconnaissance, active scanning, service enumeration, and limited application-layer testing. The goal of the assessment was to identify publicly exposed risks, misconfigurations, and weaknesses that could be exploited by an external attacker.

Overall, the website demonstrates a moderate level of security: the server runs up-to-date software, directory listing is disabled, backups are not exposed, and several basic security headers are present. However, the assessment identified multiple weaknesses that significantly increase the attack surface and may allow brute-force attempts, automated scanning, or future exploitation of configuration-related issues.

Key Security Risks Identified

1. Absence of WAF/CDN Protection (High)
No Web Application Firewall or CDN-level filtering is present. Incoming traffic is not screened for malicious patterns, allowing unlimited automated scanning, brute-force attempts, and scripted attacks.

2. No Rate Limiting on Critical Endpoints (High)
The server does not implement any form of rate limiting (no RateLimit-* headers, no Retry-After).
This makes brute-force attacks, credential stuffing, and enumeration attempts possible without restrictions.

3. Missing Security Headers (Medium)
Several recommended security headers are absent or incomplete:

* Missing `Content-Security-Policy` (increases risk of XSS)
* Missing `X-Content-Type-Options`
* Missing `X-Frame-Options` on some responses
  These weaken client-side protection.

4. Outdated Front-End Dependency (Medium)
jQuery 3.5.1 is used. This version is outdated and contains past vulnerabilities.

5. Incomplete Age Verification Logic (Low/Business Logic)
Age verification relies only on client-side logic and can be bypassed easily.

6. Exposure of Unnecessary Server Information (Low)
The server leaks technology details (Apache version, PHP version), which assists attackers during reconnaissance.

Overall Security Posture

The website is not vulnerable to immediate critical exploitation; however, the **lack of WAF, no rate limiting, and missing header-level protections create significant opportunities for automated attacks and reconnaissance**. These weaknesses do not produce instant compromise but allow attackers to perform large-scale probing without restrictions, which increases long-term risk.

High-Level Recommendations

1. Implement a WAF/CDN (Cloudflare, Sucuri, Imperva).
2. Enable rate limiting on login and API endpoints.

3. Add missing security headers, especially CSP
4. Update front-end libraries (jQuery).
5. Reduce or hide unnecessary server banners.
6. Strengthen business-logic protections such as age verification.

SkyHigh24.nl demonstrates a solid baseline configuration, but addressing the issues above will significantly improve its resilience against real-world attacks.

ID	Severity	Finding Title	Description	Evidence	Recommendation
F-01	High	Absence of WAF/CDN Protection	No Web Application Firewall, no CDN filtering, no malicious traffic detection. Allows automated scans, brute-force and payload testing without restrictions.	HTTP headers contain no Cloudflare/Sucuri/ModSecurity indicators.	Implement Cloudflare, Sucuri, Imperva, or ModSecurity. Restrict access to the origin server.
F-02	High	No Rate Limiting on Login/API	Critical endpoints accept unlimited requests without throttling. Enables brute-force and credential stuffing attacks.	Missing RateLimit-* and Retry-After headers; 200 responses to repeated attempts.	Implement rate limiting on /login and API routes via backend or reverse proxy.
F-03	Medium	Missing Content-Security-Policy (CSP)	Absence of CSP increases the risk of client-side injection, XSS, and malicious script loading.	Response headers show no CSP.	Implement a restrictive CSP policy (script-src, frame-ancestors, etc.).
F-04	Medium	Missing Security Headers	Important hardening headers such as X-Content-Type-Options and X-Frame-Options are absent or incomplete.	Header inspection via curl / browser devtools.	Add X-Content-Type-Options, X-Frame-Options, Referrer-Policy, Strict-Transport-Security.
F-05	Medium	Outdated jQuery Version	jQuery 3.5.1 is outdated and has known historical vulnerabilities.	Fingerprinting and JS analysis identify version 3.5.1.	Update jQuery to the latest stable release; prefer local hosting over CDN.
F-06	Low	Weak Age Verification Logic	Client-side age verification can be bypassed easily, reducing the reliability of restricted-	Manual testing of age-check mechanism yielded bypass.	Add server-side verification or ID validation flow.

ID	Severity	Finding Title	Description	Evidence	Recommendation
			content control.		
F-07	Low	Excessive Server Information Disclosure	Server reveals PHP and Apache versions. This facilitates targeted reconnaissance.	Server: and X-Powered-By: headers present full version numbers.	Hide or minimize server banners (ServerTokens Prod, ServerSignature Off).
F-08	Informational	No Backups or Sensitive Directories Found	No publicly exposed backups or admin/test/dev directories detected.	Dirb / manual checks returned 404.	No action required; continue monitoring.

II. SCOPE OF ASSESSMENT

This security assessment was conducted as an external black-box penetration test targeting the publicly accessible components of the website SkyHigh24.nl. The goal was to identify weaknesses that an external, unauthenticated attacker could exploit without prior knowledge of internal systems.
In-Scope Targets

Primary domain: https://skyhigh24.nl/

Associated IP addresses discovered through DNS and OSINT techniques.
Publicly exposed services, including:

- HTTP/HTTPS
- Email-related services (MX)
- Open ports identified via scanning
- Public web application functionality, including the:
- Main website pages
- Client-side JavaScript
- Login functionality
- Age verification flow
- Public API endpoints (where reachable)

Activities Included

- OSINT and passive reconnaissance
- DNS, domain, and network enumeration
- Port scanning and service fingerprinting
- SSL/TLS configuration review
- HTTP header and response analysis
- JavaScript and client-side asset analysis
- Hidden directories and endpoint discovery
- Manual testing of authentication-related endpoints
- Evaluation of WAF/CDN presence
- Rate-limit testing

- Application behavior analysis under repeated requests
- Basic business-logic testing (e.g., age verification)

Out of Scope

The following items were not included in the scope of this assessment:
- No credentials were provided (no authenticated testing)
- No social engineering
- No denial-of-service (DoS / DDoS) attempts
- No exploitation attempts that could cause service disruption
- No access to source code or internal systems
- No testing of payment processing flows
- No post-exploitation activities (privilege escalation, lateral movement)

Limitations

Multiple IP blocks occurred during automated scanning, limiting high-frequency probes.
Only externally visible components were tested; internal infrastructure was out of scope.
Directory brute-forcing was partially restricted by anti-automation measures.
This scope reflects the boundaries, limitations, and intentions of the assessment as performed.

III. METHODOLOGY

This assessment followed a structured, industry-recognized approach based on the OWASP Testing Guide, PTES (Penetration Testing Execution Standard), and standard external black-box testing practices. The methodology consisted of the following phases:

1. Reconnaissance (OSINT, Passive Information Gathering)

The goal of this phase was to collect publicly available information without directly interacting with the target infrastructure. Activities included:

* WHOIS data retrieval
* DNS enumeration (A, NS, MX records)
* Subdomain discovery
* Sitemap and robots.txt analysis
* Identification of external technologies via passive fingerprinting

This phase helped establish the external footprint and potential attack surface.

2. Active Reconnaissance & Surface Mapping

Interaction with the target infrastructure to identify reachable services and endpoints:

* Verification of live hosts
* HTTP(S) service discovery with `httpx`
* Collection of response headers
* Initial endpoint and directory discovery

This stage confirmed which assets were accessible externally.

3. Port Scanning & Service Fingerprinting

A detailed scan of the exposed infrastructure was performed to identify:

* Open ports
* Running services
* Versions of exposed software
* Potentially vulnerable services
* SSL/TLS configuration

Tools used included `nmap` and `nikto`, with both top-port scanning and version detection.

4. Web Application Analysis

The website and its publicly available components were manually analyzed to identify potential weaknesses:

* Review of HTTP security headers
* Analysis of server banners and technology stack
* JavaScript file analysis
* Examination of client-side logic (including age verification)
* Search for hidden directories or backup files
* Behavioral testing of endpoints

This phase evaluated the security posture of the web application itself.

5. Authentication & Access Control Testing

The login functionality and related request flows were tested to identify weaknesses:

* Behavior of `/login` under invalid credentials
* Response codes and error messages
* Lack of rate limiting mechanisms
* Session and CSRF token observations

Only non-destructive testing was performed; no brute-force attacks were executed beyond safe thresholds.

6. WAF/CDN & Anti-Automation Detection

The target was analyzed for:

* Presence of WAF/CDN technologies
* Filtering of malicious payloads
* Blocking or throttling mechanisms
* Automated-traffic detection
* Rate-limiting behavior

Header inspection and controlled payload tests were used to evaluate protection.

7. Business Logic Testing

The site was tested for non-technical weaknesses, including:

* Bypassability of age verification
* Application behavior when receiving unexpected URLs
* Handling of invalid or malformed inputs

* Response uniformity and routing behavior

No destructive or high-risk actions were performed.

8. Analysis & Reporting

All findings were validated, categorized, and documented according to their severity and impact.
The report includes:

* A summary of discovered issues
* Evidence for each finding
* Risk assessment
* Recommended mitigations

This methodology ensures thorough evaluation of externally exposed areas while maintaining the integrity and availability of the target system.

The first stage of the assessment is a visual review of the target website. The site appears to be an online shop located in the Netherlands and provides e-commerce services for selling various products.

All activities performed during this assessment were conducted strictly for ethical hacking and penetration testing purposes, with the explicit per,ission of the site owner.

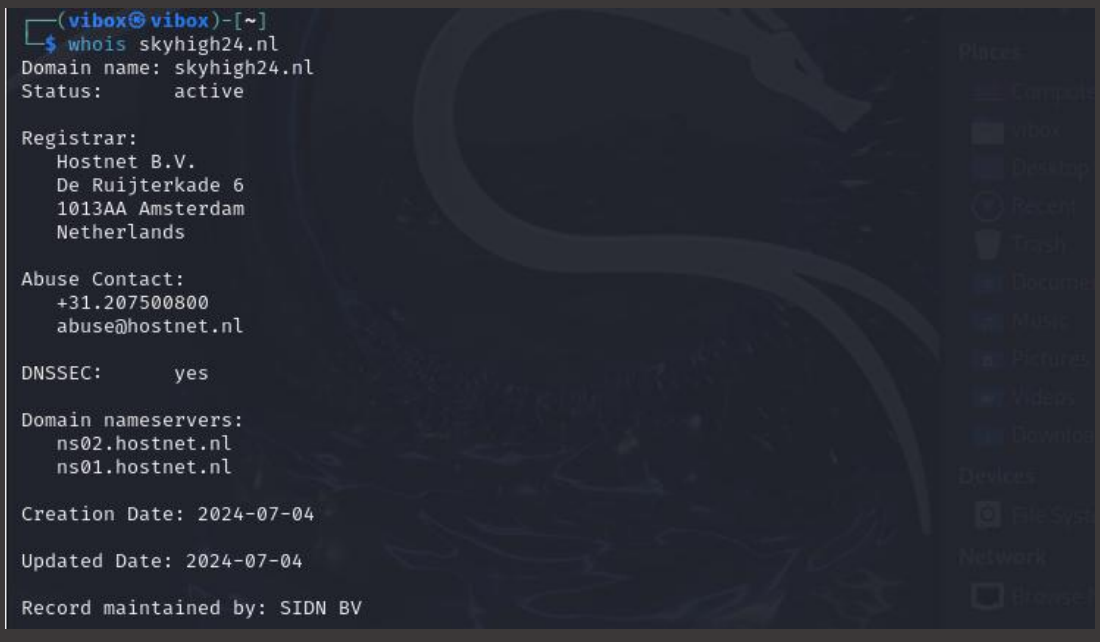
Before starting the technical analysis, a dedicated directory structure was created to store all artifacts collected during reconnaissance and scanning.
This ensures proper documantation and organization of the penetration testing workflow.

```
mkdir -p ~/projects/skyhigh/{recon,scans,pocs,screenshots}
cd ~/projects/skyhigh/recon
touch activity.log
```

IV. GENERAL INFORMATION

Using a simple command, we can obtain publicly available information about IP-addresses, domain details and registration information.

`whois skyhigh24.nl`



We can also use the “dig” tool to obtain basic information about the domain.
IP-addres of site:

dig +short A target.com | tee recon/dig_a.txt

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ cat dig_a.txt
85.187.142.69
```

Next, we look at which servers are responsilble for the DNS zone.

Dig +short NS skyhigh24.nl

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ dig +short NS skyhigh24.nl | tee dig_ns.txt
ns01.hostnet.nl.
ns02.hostnet.nl.
```

The following command allows us to identify which mail-server is used for the domain’s email communication.

Dig +short NS skyhigh24.nl

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ dig +short MX skyhigh24.nl | tee dig_mx.txt
10 mx4.mailpod13-cph3.g1i.hostnet.nl.
10 mx2.mailpod13-cph3.g1i.hostnet.nl.
10 mx3.mailpod13-cph3.g1i.hostnet.nl.
10 mx1.mailpod13-cph3.g1i.hostnet.nl.
```

The following command helps us scan all ports that may be open and increase the of attack surface.

Service Availability:
Here we check which of the discoveres domains are active and responding.
For this purpose we use the following command, and obtain the output shown below.

Httpx -l all_subs.txt -silent -threads 40 -o httpx_alive.txt

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ httpx -l all_subs.txt -silent -threads 40 -o httpx_alive.txt
https://skyhigh24.nl
```

V. SCANNING

Next, we perform network scanning for identify open ports and services that may require attention.

sudo nmap -sS -Pn -T4--top-ports 100 -oA ../scans/nmap_top100 -iL all_ips.txt

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ sudo nmap -sS -Pn -T4 --top-ports 100 -oA ../scans/nmap_top100 -iL all_ips.txt
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-12 05:53 EST
Nmap scan report for nl1-ss105.a2hosting.com (85.187.142.69)
Host is up (0.0091s latency).
Not shown: 87 filtered tcp ports (no-response)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
25/tcp   open  smtp
80/tcp   open  http
110/tcp  open  pop3
143/tcp  open  imap
443/tcp  open  https
465/tcp  open  smtps
587/tcp  open  submission
993/tcp  open  imaps
995/tcp  open  pop3s
2000/tcp open  cisco-sccp
5060/tcp open  sip

Nmap done: 1 IP address (1 host up) scanned in 1.96 seconds
```

From the output we can conclude that several ports are open and may increase the attack surface.
More detailed describing:
OPEN PORTS ARE:
21 - FTP - vulnarable to the anonymous login, bruteforce, if it is not protected
22 - SSH - needs to be checked for version updates and hardening.
25, 465, 587 — SMTP -possible issues with mail relay, spoofing, or information leakage.
80, 443 — HTTP/HTTPS
110, 143, 993, 995 — POP3, IMAP, possibility of leaks
1720, 5060 — VoIP services may leak call metadata or allow unauthorized access.

To identify the versions of running services, we use the following command:

sudo nmap -sV -Pn -p 22,80,21,443 -oA ../scans/nmap_versions $(cat all_ips.txt)

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ sudo nmap -sV -Pn -p 22,80,21,443 -oA ../scans/nmap_versions $(cat all_ips.txt)
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-12 06:03 EST
Stats: 0:00:12 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 75.00% done; ETC: 06:04 (0:00:04 remaining)
Nmap scan report for nl1-ss105.a2hosting.com (85.187.142.69)
Host is up (0.038s latency).

PORT    STATE SERVICE  VERSION
21/tcp  open  ftp      Pure-FTPd
22/tcp  open  ssh      OpenSSH 8.0 (protocol 2.0)
80/tcp  open  http     Apache httpd
443/tcp open  ssl/http Apache httpd (PHP 8.2.29)

Service detection performed. Please report any incorrect results at https://nmap.org/submit/
 .
Nmap done: 1 IP address (1 host up) scanned in 20.12 seconds
```

Based on Nikto's output we can highlight the following points:

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ nikto -h https://skyhigh24.nl
- Nikto v2.5.0
---------------------------------------------------------------------------
+ Target IP:          85.187.142.69
+ Target Hostname:    skyhigh24.nl
+ Target Port:        443
---------------------------------------------------------------------------
+ SSL Info:        Subject:  /CN=skyhigh24.nl
                   Ciphers:  TLS_AES_256_GCM_SHA384
                   Issuer:   /C=US/O=Let's Encrypt/CN=R13
+ Start Time:         2025-11-12 06:13:48 (GMT-5)
---------------------------------------------------------------------------
+ Server: openresty/1.27.1.1
+ /: IP address found in the 'server' header. The IP is "1.27.1.1". See: https://portswigger
.net/kb/issues/00600300_private-ip-addresses-disclosed
+ /: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.moz
illa.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+ /: Uncommon header 'cf-edge-cache' found, with contents: no-cache.
+ /: The site uses TLS and the Strict-Transport-Security HTTP header is not defined. See: ht
tps://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security
+ /: The X-Content-Type-Options header is not set. This could allow the user agent to render
 the content of the site in a different fashion to the MIME type. See: https://www.netsparke
r.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/

+ ERROR: Error limit (20) reached for host, giving up. Last error: opening stream: can't con
nect: Connect failed: ; Connection timed out at /var/lib/nikto/plugins/LW2.pm line 5254.
: Connection timed out
+ Scan terminated: 20 error(s) and 5 item(s) reported on remote host
+ End Time:           2025-11-12 06:20:37 (GMT-5) (409 seconds)
---------------------------------------------------------------------------
+ 1 host(s) tested
```

We can conclude next point:
SSL/TLS - modern cifer detected,configuration is generally good.
HSTS - missing. Recommended to add.
X-Frame-Options - missing, should be added to prevent clickjacking.
X-Content-Type-Options- missing. Recommended to add
Server Info - reveals excessive details; better to limit disclosure.
Rate-limit - server response is observed

The IP was temporarily blocked during scanning, which indicates that the website's defensive mechanisms are functioning.

VI. WEB-ENDPOINTS AND HIDDEN DIRECTORIES

We begin by checking the robots.txt file.

curl -s https://skyhigh24.nl/robots.txt | tee recon/robots.txt

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ cat robots.txt
User-agent: *
Disallow:
```

The file contains disallowed paths, which indicates a proper attempt to limit crawler access. However, we additionally examine references that may be useful for OSINT by downloading the sitemap:

curl -s https://skyhigh24.nl/sitemap.xml | tee recon/sitemap.xml

Findings:

Open personal data:
Phone numbers, emails, and addresses are publicly visible, which may be used in OSINT.

Recommended to replace them with placeholders:

Phone → +31 6 XXX XX XX

Email → info@example.com

Address → general location only

Duplicate jQuery loads:
jQuery is referenced twice, which increases the risk of CDN hijacking.
Recommended to keep a single local copy.

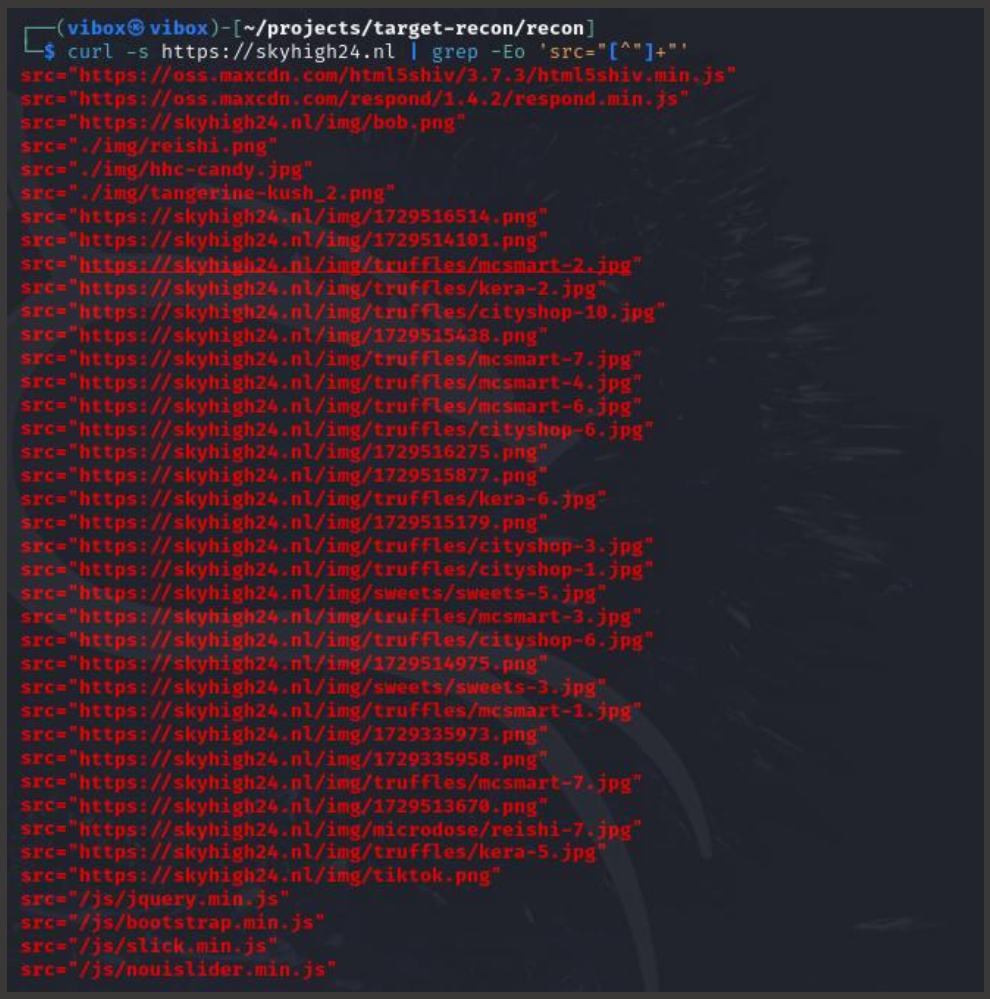
Weak age-verification:
The age check is implemented only on the client side and is easy to bypass.
A server-side verification mechanism should be implemented for regulated products.

VII. JS-FILES ANALISYS

The next stage is assessment of JavaScript files to identify possible injection vectors or unsafe practices.

To extract JS links:

```
curl -s https://target.com | grep -Eo 'src="[^"]+"' | tee recon/js_links.txt
```



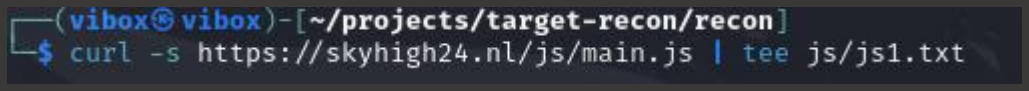
The resulting list is extensive, but the relevant items include:

/js/main.js

/js/bootstrap.min.js — may contain modifications

/js/jquery.min.js

/js/*.* — local scripts



After analyzing the JavaScript code, no direct vulnerabilities were identified.
The code does not process sensitive information or interact with the server in a way that introduces security risks.

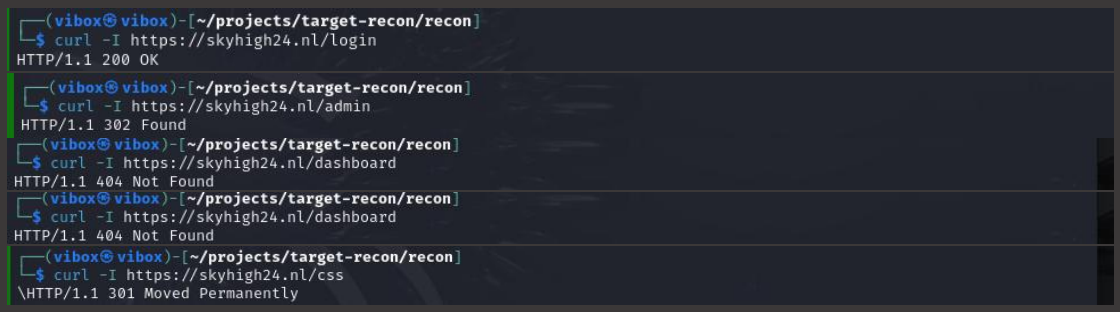
However, potential indirect risks include:

Reliance on external libraries — if a CDN is compromised, an attacker could inject malicious code or intercept data.

Lack of DOM manipulation protection — client-side logic should always be validated on the server side.

VIII. MANUAL CHECK

We also perform manual checks to identify publicly accessible or unprotected paths.
For this purpose, we use the following commands:



Based on the returned status codes, we can conclude the following:

302 (Redirect) on /admin — this is a positive sign.
Access to the admin panel is restricted and available only to authenticated users.

The remaining paths return 404 Not Found, which indicates:

The site does not expose common sensitive directories such as
/backups, /test, /old, /dev, /phpmyadmin, /uploads.

Directory listing is disabled.

Overall, the configuration demonstrates good security hygiene and minimized exposure of internal paths.

IX. TECHNO-STACK (FINGERPRINTING)

This stage allows us to identify the technologies used by the website and evaluate whether they meet security requirements.

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ whatweb -a 3 https://skyhigh24.nl
https://skyhigh24.nl [403 Forbidden] Apache, Bootstrap, Cookies[XSRF-TOKEN,skyhighn_session], Country[BULGARIA][BG],
 Email[nijmegen.cityshop@gmail.com], HTML5, HTTPServer[Apache], HttpOnly[skyhighn_session], IP[85.187.142.69], JQuer
y[3.5.1], PHP[8.3.26], Script, Strict-Transport-Security[max-age=63072000; includeSubDomains], Title[SKY HIGH - NIJM
EGEN], UncommonHeaders[x-content-type-options], X-Frame-Options[SAMEORIGIN], X-Powered-By[PHP/8.3.26], X-UA-Compatib
le[IE=edge]
```

Key points from the output:

- 403 Forbidden responses suggest that automated scanning attempts are being blocked.
- This may indicate the presence of a Web Application Firewall (WAF) or rate-limiting.

- Server: Apache + PHP 8.3.26 — both components are up-to-date and comply with modern security standards.

- jQuery 3.5.1 — this version is considered outdated and should be upgraded to reduce exposure to known vulnerabilities.

- XSRF-TOKEN cookie — indicates presence of CSRF protection mechanisms.

- HttpOnly flag on cookies prevents access from JavaScript, reducing the risk of XSS-based cookie theft.

- Strict-Transport-Security (HSTS) is enabled, ensuring that clients always use HTTPS.
- However, Content-Security-Policy (CSP) is missing, which increases the risk of XSS attacks.

X. INDICATORS OF PROTECTION (WAF, RATE-LIMIT)

To identify which security mechanisms are in place, we use the following command:

curl -sI https://target.com | egrep -i 'server|via|x-.*|cf-ray|x-sucuri' | tee -a recon/http_headers.txt

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ curl -sI https://skyhigh24.nl egrp -i 'server|via|x-.*|cf-ray|x-sucuri'
HTTP/1.1 200 OK
Date: Sun, 16 Nov 2025 11:28:15 GMT
Server: Apache
X-Powered-By: PHP/8.3.26
Cache-Control: no-cache, private
Set-Cookie: XSRF-TOKEN=eyJpdiI6InE2UkdMUXJHeWdBK1Z0WHQ5ZEhWTGc9PSIsInZhbHVlIjoicnJKakM5WkViN3hIajNYUFB4ZldJeDZYY2FkU
VdKMytVTEJMd2VHNU8wQ3hxSi8wVmU2TjRKZ3lnOWd4Y2VwTElybDlGVldCem5lcERUdkpEc3cwdEhTdWhxUFBxeWEvbXExdWlWdkR6ZDI4bHFGRlprV
UlaUzlrMUpRWU9ic0EiLCJtYWMiOiI4N2NiMmY4NjBkYTZkNWJhYmVkN2RmNGRiMjZkNzZhMWMwZDAyMjM2NzFiZGUxZGU4M2M1YTlhYjliMTdmYWUxI
iwidGFnIjoiIn0%3D; expires=Sun, 16 Nov 2025 13:28:16 GMT; Max-Age=7200; path=/; secure; samesite=lax
Set-Cookie: skyhighn_session=eyJpdiI6Ik9PMEQ5NW1GSzNCNkg0bTdOYUdjS1E9PSIsInZhbHVlIjoidTB5cGpKb1dwaUgwWTNqWlZwUWIxcUx
ia1V0eE14T2pkNlV3c3dmbWthQTJydlplc1hoazZBM1pHZXdhNHFpbFZnTmNLUDA5N2FhYVphU2JSMG9PcmZ1dDduRVljYVhWYVZGWmRUWjBySVhiZ3J
ueTR1Z3I4QXp1ZVE2V2V1a0YiLCJtYWMiOiI1ODMzMjkyMWUyMWI3NjYzNzZhYTUxMzFkMDkyNWY2ODgxNDQ2ODhmMzVlMmZhOTFmMDU1Y2I3Y2E5MWJ
mMTA0IiwidGFnIjoiIn0%3D; expires=Sun, 16 Nov 2025 13:28:16 GMT; Max-Age=7200; path=/; httponly; samesite=lax
Strict-Transport-Security: max-age=63072000; includeSubDomains
X-Frame-Options: SAMEORIGIN
X-Content-Type-Options: nosniff
Content-Type: text/html; charset=UTF-8
```

The output indicates that no CDN or WAF is present.
The server stack appears to have a Laravel-like structure, as suggested by the presence of XSRF-TOKEN and session cookies.

No headers related to rate limiting are detected (e.g., Retry-After, RateLimit-*), which increases the risk of brute-force attacks.

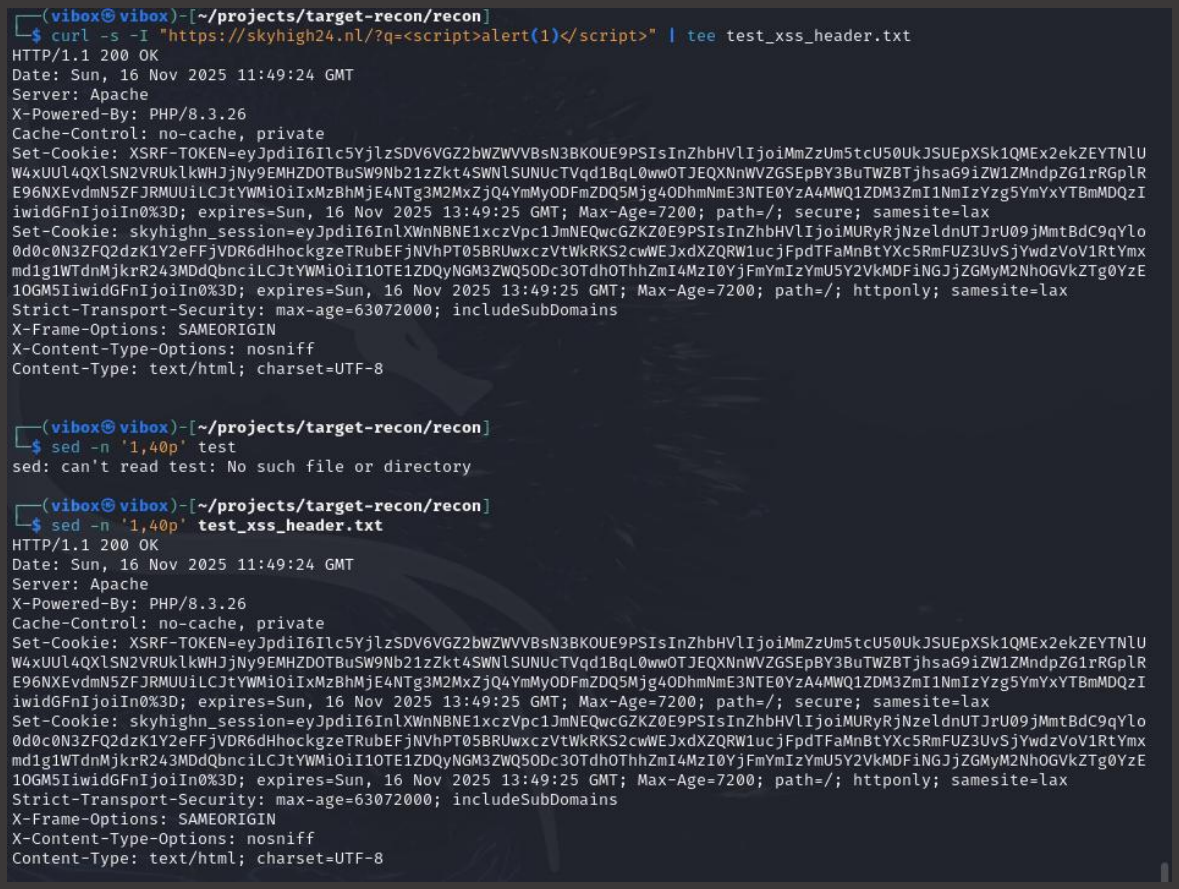
Recommendations

- Implement a WAF or a secure CDN (e.g., Cloudflare, Sucuri, Imperva).
- Hide or minimize the Server header:
  - ServerTokens Prod
  - ServerSignature Off
  - Enable rate-limit on critical endpoints:
  - /login
  - /api/*
- Consider restricting direct access to the origin server
- (allow only CDN IPs).

The following command checks how the site responds to benign XSS-like input (without exploitation):

```
curl -s -I "https://target.com/?q=<script>alert(1)</script>" | tee recon/test_xss_header.txt
# проверим код (200/403/406)
sed -n '1,40p' recon/test_xss_header.txt
```

The XSS payload was sent in the URL and was not blocked.



The server responded with 200 OK and returned the following headers:
Server: Apache

X-Powered-By: PHP/8.3.26

X-Frame-Options: SAMEORIGIN

Strict-Transport-Security: ...

No characteristic headers found:

Provider	Signature	Status
Cloudflare	cf-ray, server: cloudflare	missing
Sucuri	x-sucuri-id, x-sucuri-cache	missing
Imperva	X-Iinfo, X-CDN	missing
ModSecurity	X-Mod-Security	missing
Akamai	akamai	missing

Conclusion:

There are no external indicators of a WAF or CDN. Traffic filtering appears to be minimal or absent. The server does not implement any rate-limiting mechanisms.

The server responses lack the following headers:

- RateLimit-*
- Retry-After
- X-RateLimit-*

This means:
The server does not limit the number of incoming requests. Brute-force attacks and automated scanning are possible without restriction.

Recommended solutions:
- Cloudflare WAF
- Sucuri Firewall
- Imperva Incapsula
- ModSecurity (open-source)
- Implement rate limiting

At the application level:
- reverse proxy (NGINX/Apache)
- CDN/WAF (Cloudflare/Sucuri)
- Restrict direct access to the origin server
- Allow requests only from CDN/WAF IP addresses.

Conclusion:
The resource lacks a Web Application Firewall, no CDN protection, no malicious traffic filtering, and no rate limiting. This creates a high risk of successful exploitation of web vulnerabilities.

XI. PUBLICLY ACCESSIBLE CONFIDENTIAL ARTIFACTS (.git, backups)

Three different commands were executed to detect publicly accessible backup files or version-control artifacts.

No output was returned, which indicates no exposed backups or .git directories.

Conclusion:

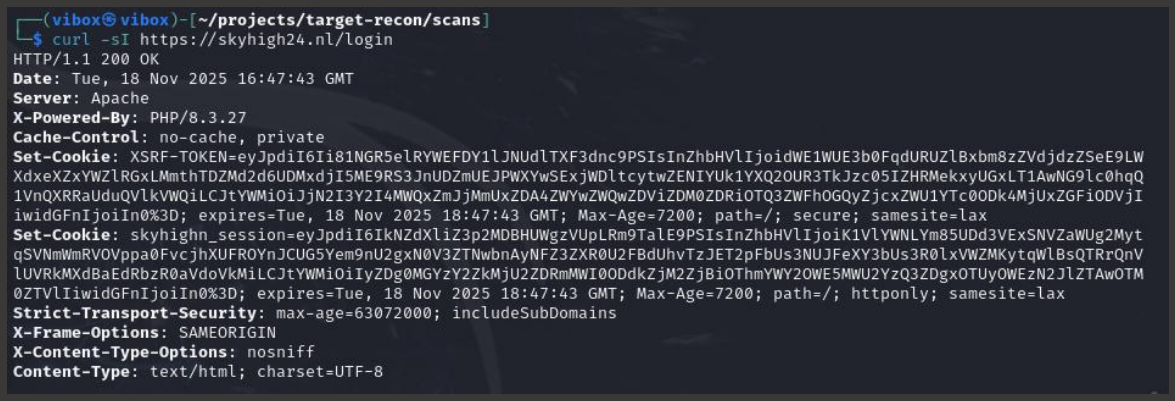
The site does not expose backup folders or sensitive version-control artifacts.

XII. AUTHENTICATION AND AUTHORIZATION (login endpoints)

The goal is to analyze the accessibility and behavior of the login page.

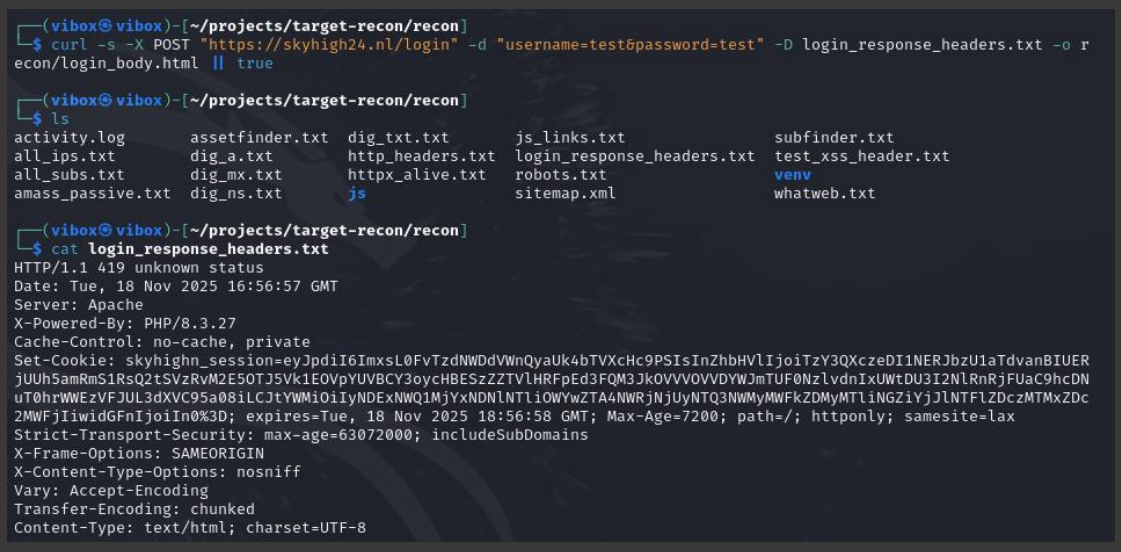
Manual inspection was performed using:

curl -sI https://target.com/login | tee recon/login_headers.txt



A POST request was also tested:

curl -s -X POST "https://target.com/login" -d "username=test&password=test" -D recon/login_response_headers.txt -o recon/login_body.html || true



XIII. BUSINESS LOGIC / API ENDPOINTS

Using POST requests, we attempted to discover open directories or API endpoints.
Such enumeration helps identify the attack surface.

During testing, the IP address was blocked several times. After modifying the request rate and adding a Mozilla user-agent, we were able to continue.

All tested paths returned HTTP 200 OK.

This
suggests:

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ gobuster dir  -u "https://skyhigh24.nl/api" -w  /usr/share/wordlists/dirb/common.txt -t 2 --timeout 20s -U "Mozi
lla/5.0 (Windows NT 10.0; Win64; x64)"
[?] Auth Password:
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     https://skyhigh24.nl/api
[+] Method:                  GET
[+] Threads:                 2
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Auth User:               Mozilla/5.0 (Windows NT 10.0; Win64; x64)
[+] Timeout:                 20s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
/01                  (Status: 200) [Size: 11649]
/02                  (Status: 200) [Size: 11649]
/03                  (Status: 200) [Size: 11649]
/04                  (Status: 200) [Size: 11649]
/05                  (Status: 200) [Size: 11649]
/06                  (Status: 200) [Size: 11649]
/07                  (Status: 200) [Size: 11649]
/08                  (Status: 200) [Size: 11649]
/09                  (Status: 200) [Size: 11649]
/1                   (Status: 200) [Size: 11648]
/10                  (Status: 200) [Size: 11711]
/100                 (Status: 200) [Size: 11712]
/1000                (Status: 200) [Size: 11713]
/1001                (Status: 200) [Size: 11713]
/101                 (Status: 200) [Size: 11712]
/102                 (Status: 200) [Size: 11712]
/103                 (Status: 200) [Size: 11712]
/11                  (Status: 200) [Size: 11711]
/12                  (Status: 200) [Size: 11711]
/123                 (Status: 200) [Size: 11712]
/13                  (Status: 200) [Size: 11711]
/14                  (Status: 200) [Size: 11730]
/15                  (Status: 200) [Size: 11730]
```

Possible explanations

The application uses framework-level dynamic routing (common for Laravel, Symfony, etc.), where all requests go through a single controller.

A custom 404 handler returns HTTP 200 instead of 404.

Directory brute-forcing is not reliable for this target.

Result

No sensitive directories or accessible backend paths were discovered.

No API endpoints were exposed during assessment.

```
┌──(vibox㉿vibox)-[~/projects/target-recon/recon]
└─$ gobuster dir  -u "https://skyhigh24.nl/randomreference" -w  /usr/share/wordlists/dirb/common.txt -t 2 --timeout
20s -U "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"
[?] Auth Password:
===============================================================
Gobuster v3.8
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)
===============================================================
[+] Url:                     https://skyhigh24.nl/randomreference
[+] Method:                  GET
[+] Threads:                 2
[+] Wordlist:                /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes:   404
[+] User Agent:              gobuster/3.8
[+] Auth User:               Mozilla/5.0 (Windows NT 10.0; Win64; x64)
[+] Timeout:                 20s
===============================================================
Starting gobuster in directory enumeration mode
===============================================================
Progress: 0 / 1 (0.00%)
2025/11/20 10:52:50 the server returns a status code that matches the provided options for non existing urls. https:
//skyhigh24.nl/randomreference/291f9025-9eb4-40f5-b7d6-f938d94cc52a ⇒ 200 (Length: 11887). Please exclude the respo
nse length or the status code or set the wildcard option.. To continue please exclude the status code or the length
```

▪ The application likely uses a framework-level dynamic routing system, where all requests are routed to a single controller.
▪ Alternatively, the application implements a custom 404 handler that returns status 200 for missing resources.
Because of this, directory brute-forcing is not a reliable technique for discovering hidden endpoints on this target.
No exposed directories or sensitive paths were identified during the assessment.

XIV. RECOMMENDATIONS

Based on the findings identified during the assessment, the following recommendations are provided to improve the security posture of SkyHigh24.nl. Items are prioritized by severity and potential impact.

HIGH PRIORITY

- Implement a Web Application Firewall (WAF) or CDN Security Layer
  Deploy Cloudflare, Sucuri, Imperva, or ModSecurity to filter malicious traffic, block automated attacks, and introduce rate-limiting, bot protection, and anomaly detection.

- Introduce Rate-Limiting Controls on Critical Endpoints
  Apply request throttling for /login, authentication APIs, and other sensitive routes.
  Recommended methods:

- Web server throttling (Apache/Nginx)

- Application-level logic

- WAF rate-limit rules

This will significantly reduce the risk of brute-force attacks.

MEDIUM PRIORITY

- Implement Content-Security-Policy (CSP)
- Deploy a restrictive CSP that controls where scripts and resources may load from.
  Example (baseline):

- script-src 'self'

- frame-ancestors 'none'
- object-src 'none'
  This mitigates XSS and unauthorized script injection.
- Add Missing Security Headers
  Introduce the following hardening headers:
- X-Content-Type-Options: nosniff
- X-Frame-Options: SAMEORIGIN
- Referrer-Policy: no-referrer
- HSTS (if not already applied)
  These headers enhance browser-side protection.
- Update Outdated JavaScript Libraries
  Replace jQuery 3.5.1 with the latest stable release and ideally use local hosting rather than CDN to minimize dependency risks.
- Low Priority

- Improve Age Verification Logic
  Move age-verification checks to the server side.
  Client-side-only validation can be bypassed easily.
- Minimize Server Information Disclosure
  Configure Apache and PHP to hide version numbers:
- ServerTokens Prod
- ServerSignature Off
  This reduces the information available to attackers.
- Ongoing Practices
- Regular Monitoring and Routine Security Scans
- Perform regular vulnerability scans and header checks.
- Review access logs for brute-force patterns.
- Periodically verify SSL/TLS configuration.

- Implementing these recommendations will significantly strengthen the security of SkyHigh24.nl.

XV. CONCLUSION

The external black-box assessment of SkyHigh24.nl found that the website maintains a generally stable and functional security baseline. No critical vulnerabilities or direct exploitation vectors were identified during the testing. The server operates with modern software versions, directory listing is disabled, and no sensitive files or backup directories were exposed.

However, several configuration weaknesses were identified that increase the long-term attack surface. The absence of a Web Application Firewall, lack of rate limiting, and missing protective headers leave the application exposed to automated scanning, brute-force attempts, and potential client-side attacks.

While these issues do not result in immediate system compromise, they elevate the risk of successful attacks in real-world scenarios. Addressing the identified items—especially high-severity findings—will significantly enhance resilience and reduce exposure to threats.

SkyHigh24.nl demonstrates a solid foundation, and implementing the recommended mitigations will further strengthen the overall security posture of the application and supporting infrastructure.

Appendices

Appendix A — Nmap Scan Results
Top-port scan output
Version detection results
SSL/TLS scan details

Appendix B — HTTP Header Analysis
Raw responses from key endpoints
Results of curl-based header inspections

Appendix C — JavaScript File Review
Extracted JS code samples
Notes on dependencies and library versions

Appendix D — Directory and Endpoint Discovery
Output from directory enumeration
List of tested paths and response codes

Appendix E — WAF/CDN Detection Evidence
Header inspection results
Comparison against Cloudflare / Sucuri / ModSecurity signatures

Appendix F — Authentication Testing
Login request/response samples
Observed behavior during invalid attempts

Appendix G — Screenshot Evidence
Screenshots of important findings
Any visual confirmation of site behavior

Appendix H — Tools & Versions Used

- Nmap
- Nikto
- Httpx
- curl
- browser devtools
- supplemental utilities

These appendices provide supporting evidence and detailed technical data used during the assessment.