CyberTestify

Automated Security Scan Report

rest.vulnweb.com · Discovery Bundle

CyberTestify Security Scan — Completed

Report No: **CT-ÖRNEK-F82E**

Verification Code: **5A53-5E8E**

This report is not a formal penetration test / certification.

Table of Contents

TARGET
rest.vulnweb.com

PACKAGE
Discovery Bundle

REPORT NO
CT-ÖRNEK-F82E

1. Executive Summary

OVERALL ASSESSMENT High Risk

This target does not respond over HTTPS; communication is carried unencrypted (HTTPS should be adopted as a priority). The highest risk was detected in the CMS & known CVE Scan area (No known CMS fingerprint detected · 21 known CVEs in the server/software version); priority remediation is recommended. Each area is reported separately below.

⚡ Management Decision

2 high/critical indicator(s) requiring priority attention were found; an urgent remediation plan is recommended. Medium/low items can be addressed via planned improvement.

- **Overall risk level: High** — 3 areas examined; the highest risk is in the **CMS & known CVE Scan** area (No known CMS fingerprint detected · 21 known CVEs in the server/software version).
- ⚠ **HTTPS not supported:** The target did not respond over HTTPS (443); reconnaissance was carried out over http://. Unencrypted communication is a serious finding in itself (see below).
- **Scope (real numbers):** 0 subdomains inventoried · 12 API/Swagger paths checked (2 pages scanned) · 6+ passive CMS/technology signals examined.
- **Recommended first step:** Start with the highest-risk area; step-by-step ready-made solutions for each finding are provided in the “AI Solution Suggestions” section.

Improvement Priorities

⚡ **Most Urgent:** Outdated component (CVE).

🛠 **Quick Wins (~1-2 days):** Missing HSTS.

2. Findings

2.1 Vulnerability Distribution

A total of **2** finding indicators were detected; the severity distribution is below.

2.2 Master Findings Table

ID	Title	State	Severity
CT-1	Missing HSTS	Open	High
CT-2	Outdated component (CVE)	Open	High

What was tested, and what came out?

✓ CHECKED — NO INDICATOR FOUND

- **Subdomain Takeover Scan** 0 subdomain records checked (sub-addresses attached to your site — itemised list: "Subdomain Takeover Scan" section); no takeover indicator found (no dangling record a third party could claim)
- **API & Swagger Reconnaissance** 12 paths checked (12 fixed + 0 sitemap candidates, from 2 pages — FULL path list: "API & Swagger Reconnaissance" section); no publicly accessible API schema found (no externally readable API document)
- **CMS / Framework CVE Match** 6 passive signal categories examined (headers, meta generator, HTML traces, version files, known CMS paths, library hints — full list: "CMS & known CVE" section); no known CMS/framework fingerprint detected
- **Sitemap + robots.txt path reconnaissance** 0 paths derived from the sitemap + robots.txt Disallow tried (existence check only — candidate list: "API & Swagger Reconnaissance" section); no sensitive/administrative endpoint found

⚠ INDICATOR FOUND — DETAILED IN THE SECTIONS BELOW

- **Server/Software Banner → Known CVE** 21 known CVEs in banner version (PHP 7.1.26)

2.3 Detailed Findings

[HIGH] CT-1 · Missing HSTS

State: Open

Description: No HSTS; browser not forced to HTTPS.

How it was detected: The target does not respond over HTTPS; all traffic is carried unencrypted (plaintext) — interceptable/modifiable. Solution: valid TLS certificate + HTTP→HTTPS redirect + HSTS.

Business Impact: Without forced HTTPS, traffic can be intercepted/redirected via man-in-the-middle.

RECOMMENDED FIX

Add `Strict-Transport-Security: max-age=31536000; includeSubDomains` (if fully HTTPS).

Reference: CWE-319 · OWASP A05:2021 Security Misconfiguration

[HIGH] CT-2 · Outdated component (CVE)

State: Open

Description: Outdated component/CMS fingerprint and possible known CVE.

How it was detected: See the "CMS & known CVE Scan" section below for details.

Business Impact: An outdated component is exposed to known CVEs; targeted-attack risk.

RECOMMENDED FIX

Keep components/plugins/themes updated; add auto-update + dependency scanning.

Reference: CWE-1104 · OWASP A06:2021 Vulnerable & Outdated Components

3. Controls & Methodology

IDENTIFIED RISKS

Finding	Severity	Description
HTTPS not supported (unencrypted communication)	High	The target does not respond over HTTPS; all traffic is carried unencrypted (plaintext) — interceptable/modifiable. Solution: valid TLS certificate + HTTP→HTTPS redirect + HSTS.
CMS & known CVE Scan — No known CMS fingerprint detected · 21 known CVEs in the server/software version	High	See the “CMS & known CVE Scan” section below for details.

SCOPE & METHODOLOGY

This report was generated automatically from externally observable data using **passive** (non-exploitative) techniques across three reconnaissance areas:

- **Subdomain Takeover:** Subdomains are collected from Certificate Transparency logs (crt.sh, with certSpotter as a fallback); each is put through DNS/CNAME resolution via Cloudflare DoH and compared against a signature database of known “dangling” (abandoned cloud services).
- **API & Swagger Reconnaissance:** A fixed list of common API documentation paths is checked via GET; any OpenAPI/Swagger schemas found are parsed and sensitive/unauthenticated endpoints are flagged (the endpoints are not called).
- **CMS & known CVE:** CMS and version fingerprinting is derived from HTTP headers, `<meta generator>` and HTML patterns; if the generator is hidden, it is verified via the EXISTENCE of known CMS paths (GET/existence only — no login attempt). The detected version is matched against known CVEs that explicitly cover the version by querying the NVD (NIST National Vulnerability Database); if the version cannot be read, no CVE mapping is performed (no fabricated CVEs).

> All data was collected externally, without causing harm to the target. Authentication-protected areas, the internal network and active exploitation are outside the scope of this package.

Subdomain Takeover Scan

Overall risk level: Low — No subdomain seen in the Certificate Transparency records

From the Certificate Transparency records (crt.sh / certSpotter), **0** unique subdomains were inventoried; of these, the CNAME record of **0** was resolved and examined for takeover (subdomain takeover).

No **takeover-able (dangling)** subdomain pointing to an abandoned cloud resource was detected among the resolved CNAME records. This indicates that your externally visible subdomain surface currently appears **narrow and controlled**.

> Scope: Passive sources only (Certificate Transparency logs + observable DNS). No subdomain brute-force / active scan was performed.

API & Swagger Reconnaissance

Overall risk level: Low — No publicly accessible API/Swagger documentation found

The following **12** common API documentation/discovery paths were checked via GET. No endpoint was called/exploited (passive reconnaissance).

Paths checked (full list)

Path	HTTP	Status
/openapi.json	404	Not found
/swagger.json	404	Not found

Path	HTTP	Status
/v2/api-docs	400	Not found
/v3/api-docs	400	Not found
/api-docs	404	Not found
/api/docs	400	Not found
/api/v1/docs	400	Not found
/swagger-ui.html	404	Not found
/swagger/index.html	400	Not found
/redoc	404	Not found
/.well-known/openapi.json	400	Not found
/graphql	404	Not found

None of the paths checked returned a publicly accessible API schema/interface. The absence of public API documentation means attackers cannot easily **map** your API surface from outside — this is a positive sign for the external attack surface.

> Scope: Only publicly accessible documentation paths were checked via GET; no endpoint was called/exploited (passive reconnaissance).

CMS & known CVE Scan

Overall risk level: High — No known CMS fingerprint detected · 21 known CVEs in the server/software version

Fingerprint sources examined

For CMS/framework and version detection, the following **6 passive signal categories** in the home page response were examined (the phrase “6+ passive CMS/technology signals” in the Executive Summary refers to exactly this list; the “+” denotes the multiple sub-signals within each category):

- HTTP response headers (`Server` , `X-Powered-By` , `X-Generator` , `X-Drupal-Cache` , `X-Magento-Cache-Debug`)
- `<meta name="generator">` tag
- HTML path/pattern traces (`/wp-content/` , `/wp-includes/` , `Drupal.settings` , `/sites/all/` , `option=com_` , `/media/jui/` , `typo3conf` , `Magento_`)
- Common version files (WordPress `/readme.html` , Drupal `/CHANGELOG.txt`)
- EXISTENCE of known CMS paths (`/wp-login.php` , `/wp-json/` , `/administrator/` , `/user/login` , `/typo3/` — presence/absence check only; NO login/password attempt)
- Library/plugin hints (WooCommerce, jQuery version)

None of these signals matched a known CMS/framework. This suggests a custom-built application or an installation that deliberately hides CMS traces; either case makes automatic CMS/CVE mapping from outside harder.

Server/Software Banner Version — Known CVE (NVD)

The version(s) derived from the banner were matched against the NVD (NO exploitation/verification — only the "are there known CVEs for this version" indicator):

Software/Version	NVD result	Example CVE
Apache httpd 2.4.25	not queryable (NOT clean)	—
PHP 7.1.26	⚠ 21 known CVEs	CVE-2017-8923

Note: The CVE list below contains known vulnerabilities that were **automatically mapped via the NVD (NIST National Vulnerability Database)** from the detected version; whether they are exploitable for your version is **not verified**, and some may originate from plugins/themes. For a confirmed assessment, an update + targeted verification are recommended.

Scope: Passive fingerprint + known-CVE mapping via the NVD. No CVE was **exploited/verified**.

POSITIVE ASSURANCE — RECONNAISSANCE METHODS CHECKED

Reconnaissance comes out clean on most healthy targets; this section also makes the "nothing found" result TRANSPARENT — it shows what was ACTUALLY checked. The **2 pages** below are the home page plus the pages derived from the sitemap and robots.txt (no page outside your own site was fetched):

Reconnaissance Area	Result
Subdomain Takeover Scan	✓ 0 subdomain records checked (sub-addresses attached to your site — itemised list: "Subdomain Takeover Scan" section); no takeover indicator found (no dangling record a third party could claim)
API & Swagger Reconnaissance	✓ 12 paths checked (12 fixed + 0 sitemap candidates, from 2 pages — FULL path list: "API & Swagger Reconnaissance" section); no publicly accessible API schema found (no externally readable API document)
CMS / Framework CVE Match	✓ 6 passive signal categories examined (headers, meta generator, HTML traces, version files, known CMS paths, library hints — full list: "CMS & known CVE" section); no known CMS/framework fingerprint detected
Server/Software Banner → Known CVE	⚠ 21 known CVEs in banner version (PHP 7.1.26)
Sitemap + robots.txt path reconnaissance	✓ 0 paths derived from the sitemap + robots.txt Disallow tried (existence check only — candidate list: "API & Swagger Reconnaissance" section); no sensitive/administrative endpoint found

Three-state distinction (honesty): ✓ *No indicator found* = method ran, clean · ⚠ *Indicator present* = detailed above · ⚠ *Not assessable* = no data collectable (does NOT mean secure).

What this package assesses — and what it does NOT

ASSESSES (passive reconnaissance — GET only, external sources): subdomain inventory + takeover (dangling CNAME), publicly accessible API/Swagger/OpenAPI document, CMS/framework and server/software banner (Apache/nginx/PHP) fingerprint + known CVE match (NVD), EXISTENCE determination of API/administrative-looking paths derived from the sitemap + robots.txt Disallow entries — across the 2 pages defined above (home page + sitemap/robots.txt derivations).

DOES NOT ASSESS: active injection/IDOR/XSS verification and authorisation testing of discovered endpoints (**Active Verification / Full Pentest** scope), HTTP security header/CORS/cookie/CSP details (**Basic Scan / External Attack Surface**

> scope), GDPR/PCI/ISO framework mapping (**Compliance** scope). The statement "no indicator found" in an area **DOES NOT PROVE** you are secure — it only shows that no indicator emerged with the passive methods checked.

BEST PRACTICES / RECOMMENDED NEXT STEPS

Regardless of the outcome of this scan, the following are recommended lasting practices to keep your attack surface narrow:

- **Clean up unused CNAME records regularly** — records pointing to abandoned cloud resources carry a subdomain takeover risk; remove the DNS record before deleting the cloud resource.
- **If you have API documentation (Swagger/OpenAPI),** keep it open only to authenticated access; do not publish it publicly in production.
- **Keep your CMS, plugin and theme versions** up to date via auto-update or regular tracking; stay patched against known CVEs.
- **Detect new/unexpected subdomain certificates early** with Certificate Transparency (CT) log monitoring tools (crt.sh, certSpotter, etc.).
- **Reduce version/technology disclosure** — do not leak unnecessary version information through headers/tags such as `Server`, `X-Powered-By`, `<meta generator>`.
- **Document your subdomain inventory** — knowing which subdomain belongs to which service/team helps you quickly spot idle records.

Methodology

Stage	Description
Discovery	The external surface, pages and (for SPAs) real endpoints/params from the JS bundle are extracted.
Automated detection	Deterministic, safe (read-only) indicators are sought on the discovered surface.
Manual-deterministic verification	Findings are verified with code-based rules; "prove, don't exploit".
Authz & session	In the authenticated package, cookie/session/authorization and post-login surface are examined.
Configuration	Header, TLS, email/DNS and exposure configurations are observed.

Risk Rating Criteria

Severity	Definition
Critical	Directly/easily exploitable indicator with serious data/access impact.
High	Prominent, priority-to-fix strong indicator.
Medium	Requires attention; context-dependent verification recommended.
Low	Hardening/maturity opportunity; low priority.

Severity is a band label only (no numeric CVSS score); all findings are framed as "indicator, verification required".

---

4. AI Fix Suggestions

This section contains area-by-area remediation recommendations for all shortcomings detected in your reconnaissance scan.

Subdomain Takeover — proactive monitoring guide

No takeover-able subdomain is currently detected. To keep this state **permanently**, set up a Certificate Transparency (CT) monitoring system that catches new/unexpected subdomain certificates early:

1) Free email/webhook alert with certSpotter

Add your domain (e.g. `nomorelink.com`, including subdomains) to monitoring on `sslmate.com/certspotter`; when a new certificate is issued you receive an email/webhook alert (a new subdomain certificate may be a record you did not create).

2) A simple cron that queries crt.sh periodically (on your own server)

An example script that fetches the subdomain list daily, compares it with the previous one, and alerts on a newly added subdomain:

```
#!/usr/bin/env bash
# /etc/cron.daily/ct-watch  (chmod +x)
DOMAIN="ornek.com"
STATE="/var/lib/ct-watch/$DOMAIN.txt"
mkdir -p "$(dirname "$STATE")"; touch "$STATE"
curl -s "https://crt.sh/?q=%25.$DOMAIN&output=json" \
  | jq -r ".[].name_value" | tr "[:upper:]" "[:lower:]" | sed "s/^\*\.//" \
  | sort -u > /tmp/ct-now.txt
NEW=$(comm -13 "$STATE" /tmp/ct-now.txt)
if [ -n "$NEW" ]; then
  echo "$NEW" | mail -s "[CT] Yeni alt domain: $DOMAIN" siz@ornek.com
  cp /tmp/ct-now.txt "$STATE"
fi
```

3) Keep a subdomain inventory — document which subdomain belongs to which service/team; remove idle (unused) CNAME records from DNS before deleting the cloud resource (the order of removal matters).

API & Swagger Reconnaissance — proactive hardening guide

No publicly accessible API documentation was found. To make this permanent, place schema endpoints such as Swagger/OpenAPI/ReDoc behind authentication or an IP restriction in production. Apply the example matching your platform:

Nginx — IP allowlist + Basic Auth for `/swagger*`, `/api-docs*`, `/openapi.json`

```
location ~* ^/(swagger|api-docs|v2/api-docs|v3/api-docs|openapi\.json|redoc) {
    allow 203.0.113.0/24;   # your office/VPN IP block
    deny all;               # closed to everyone else
    auth_basic "Restricted";
    auth_basic_user_file /etc/nginx/.htpasswd;  # create with htpasswd
    # ... your existing proxy_pass/try_files directives ...
}
```

Apache (.htaccess)

```
<LocationMatch "^/(swagger|api-docs|openapi\.json|redoc)">
    AuthType Basic
    AuthName "Restricted"
    AuthUserFile /etc/apache2/.htpasswd
    Require valid-user
    Require ip 203.0.113.0/24
</LocationMatch>
```

Caddy

```
@apidocs path /swagger* /api-docs* /openapi.json /redoc*
basic_auth @apidocs {
    admin $2a$14$...   # generate with caddy hash-password
}
```

Additional recommendations: Add a global `security` definition to your OpenAPI schema; if you use GraphQL, disable introspection in production (`introspection: false`).

CMS & known CVE — proactive currency guide

No known CMS was detected (may be a custom/obfuscated application). Automate staying up to date and tracking known vulnerabilities:

1) If you use WordPress — automatic minor + security updates

In `wp-config.php` :

```
define( 'WP_AUTO_UPDATE_CORE', 'minor' );  // security/minor versions automatically
```

For automatic plugin/theme updates (WP-CLI):

```
wp plugin auto-updates enable --all
wp theme auto-updates enable --all
```

2) Add a free dependency scan (SCA) to your CI/CD

- **Dependabot** (GitHub, free): add `.github/dependabot.yml` to the repository:

```
version: 2
updates:
  - package-ecosystem: "composer"   # for WordPress/PHP; npm/pip/… are also supported
    directory: "/"
    schedule: { interval: "weekly" }
```

- **npm audit** (Node projects): add `npm audit --audit-level=high` to your CI step; the build should fail on high/critical vulnerabilities.

3) Reduce version disclosure — remove/disable version-leaking points such as `<meta generator>` , `X-Powered-By` , `/readme.html` , `/CHANGELOG.txt` ; this makes automatic CVE mapping harder for attackers.

5. Appendix — Glossary

XSS	Cross-Site Scripting — injection of malicious scripts into pages.
IDOR	Insecure Direct Object Reference — accessing others' records via ID manipulation.
TLS	Transport Layer Security.
HSTS	HTTP Strict Transport Security header.
CSP	Content Security Policy.
CORS	Cross-Origin Resource Sharing.