CyberTestify

Automated Security Scan Report

rest.vulnweb.com · External Surface & Configuration Bundle

CyberTestify Security Scan — Completed

Report No: **CT-ÖRNEK-7CE5**

Verification Code: **73DD-A368**

This report is not a formal penetration test / certification.

Table of Contents

TARGET	PACKAGE	REPORT NO
rest.vulnweb.com	External Surface & Configuration Bundle	CT-ÖRNEK-7CE5

1. Executive Summary

OVERALL ASSESSMENT — High Risk

This target does not respond over HTTPS; communication is carried in plaintext — the priority is to move to HTTPS with a valid TLS certificate. The other areas were examined over http://. The highest risk was detected in the SSL/TLS Configuration Audit area (an issue requiring urgent attention was detected at the certificate and/or protocol level.); priority remediation is recommended. Each area is reported separately below.

⚡ Management Decision

3 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** — 5 areas were examined; the highest risk is in the **SSL/TLS Configuration Audit** area (an issue requiring urgent attention was detected at the certificate and/or protocol level.).
- ⚠ **HTTPS is not supported:** The target did not respond over HTTPS (443); communication is carried in plaintext. The scan was run over http://. This is a serious finding in itself (below).
- **Recommended first step:** Start with the highest-risk area; for each finding, ready-made step-by-step commands are provided in the "AI Fix Suggestions" section.

Improvement Priorities

⚡ **Most Urgent:** Outdated component (CVE).

🛠 **Quick Wins (~1-2 days):** Missing HSTS; Exposed sensitive file — /db.sql; Missing Content-Security-Policy; Missing X-Frame-Options (clickjacking); Missing X-Content-Type-Options; Insufficient DMARC policy (missing or weak).

2. Findings

2.1 Vulnerability Distribution

A total of **7** 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	Exposed sensitive file — /db.sql	Open	High
CT-3	Outdated component (CVE)	Open	High
CT-4	Missing Content-Security-Policy	Open	Medium
CT-5	Missing X-Frame-Options (clickjacking)	Open	Medium
CT-6	Missing X-Content-Type-Options	Open	Medium
CT-7	Insufficient DMARC policy (missing or weak)	Open	Medium

8 control areas were run — 4 came back clean, 4 produced a finding

✓ CHECKED — NO ISSUE FOUND

- **CORS & Cookie Security** No issue found
- **Directory listing (autoindex / "Index of /") · CWE-548** 6 directories tried, no listing
- **Verbose error / server-path disclosure · CWE-209** No indicator found
- **Password-field autocomplete policy · CWE-522** Appropriate / no password field observed

⚠ FINDING PRESENT — DETAILED IN THE SECTIONS BELOW

- **SSL/TLS Configuration Audit** Finding present (High — detailed above)
- **Security Headers & Information Leakage** Finding present (High — detailed above)
- **DNS & Email Security** Finding present (Medium — detailed above)
- **CSP (Content Security Policy) Analysis** Finding present (Medium — detailed above)

2.3 Detailed Findings

[HIGH] CT-1 · Missing HSTS

State: Open

Description: No HSTS; browser not forced to HTTPS.

How it was detected: The site does not respond over HTTPS; all traffic is carried in plaintext — it can be intercepted/modified and sessions/passwords stolen. Fix: 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 · Exposed sensitive file — /db.sql

State: Open

Description: Sensitive file (.git/.env/backup) exposed.

How it was detected: Content verified; risk of configuration/source-code leakage. Access must be blocked immediately.

Business Impact: Sensitive files (source, backups, secrets) are exposed; risk of credential/secret leakage.

RECOMMENDED FIX

Block these paths; move source/backup/secret files out of web root.

Reference: CWE-538 · OWASP A05:2021 Security Misconfiguration

[HIGH] CT-3 · Outdated component (CVE)

State: Open

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

How it was detected: The PHP 7.x series is officially end-of-life (security updates for 7.x ended in late 2022). Numerous known vulnerabilities remain unpatched. CWE-1104 · OWASP A06:2021 (Vulnerable & Outdated Components). Remediation: upgrade to a current, supported PHP version (8.2+) and hide the version signature (expose_php=Off).

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

[MEDIUM] CT-4 · Missing Content-Security-Policy

State: Open

Description: No Content-Security-Policy; no browser-side XSS mitigation.

How it was detected: The browser cannot restrict which resources are loaded; there is no basic defence against XSS and content injection. Missing on ALL 2 scanned unique pages.

Business Impact: No browser-side mitigation against content injection/XSS; injected scripts can execute.

RECOMMENDED FIX

Define a strict CSP (`default-src 'self'`) and allowlist third-party sources.

Reference: CWE-693 · OWASP A05:2021 Security Misconfiguration

[MEDIUM] CT-5 · Missing X-Frame-Options (clickjacking)

State: Open

Description: The page can be framed by other sites (no X-Frame-Options / CSP frame-ancestors).

How it was detected: The page can be embedded in another site’s iframe; users can be deceived via clickjacking. Missing on ALL 2 scanned unique pages.

Business Impact: The page can be framed invisibly to trick users into unintended actions (clickjacking), weakening account and transaction safety.

RECOMMENDED FIX

Add `X-Frame-Options: SAMEORIGIN` or CSP `frame-ancestors 'self'`.

Reference: CWE-1021 · OWASP A05:2021 Security Misconfiguration

[MEDIUM] CT-6 · Missing X-Content-Type-Options

State: Open

Description: No X-Content-Type-Options; browser may MIME-sniff.

How it was detected: Defence in depth is weak. (missing on 2/2 pages)

Business Impact: The browser may sniff content types and execute malicious content, widening the XSS/injection surface.

RECOMMENDED FIX

Add `X-Content-Type-Options: nosniff`.

Reference: CWE-693 · OWASP A05:2021 Security Misconfiguration

[MEDIUM] CT-7 · Insufficient DMARC policy (missing or weak)

State: Open

Description: No DMARC; SPF/DKIM not enforced.

How it was detected: SPF/DKIM results are not enforced.

Business Impact: Without a DMARC policy, spoofed emails are not blocked; phishing and brand-reputation risk.

RECOMMENDED FIX

Publish `_dmarc` `v=DMARC1; p=quarantine` (start monitoring, then reject).

Reference: CWE-290 · OWASP A07:2021 Identification & Authentication Failures

3. Controls & Methodology

IDENTIFIED RISKS

Finding	Severity	Description
HTTPS is not supported (unencrypted communication)	High	The site does not respond over HTTPS; all traffic is carried in plaintext — it can be intercepted/modified and sessions/passwords stolen. Fix: valid TLS certificate + HTTP→HTTPS redirect + HSTS.

SSL/TLS Configuration Audit

TLS CERTIFICATE STATUS

⚠ This target **did not respond over HTTPS (443)**; no valid TLS certificate was found. The site is served only over **unencrypted HTTP**.

TLS PROTOCOL & CIPHER

- **Active protocol:** not detectable
- **Cipher:** not detectable
- **Legacy/weak version support:** Not observed (only TLS 1.2+ seen)

HSTS (HTTP Strict Transport Security)

- **Status:** Absent — The browser is not instructed to enforce HTTPS; there is a risk of SSL-stripping/downgrade attacks on initial requests.

IDENTIFIED RISKS

Finding	Severity	Description
HTTPS not supported (unencrypted communication)	High	The site does not respond over HTTPS; all traffic is carried in the clear (plaintext). An attacker on the same network can eavesdrop, steal sessions/passwords or alter content. Fix: valid TLS certificate + HTTP→HTTPS redirect + HSTS.
HSTS missing	Medium	HTTPS enforcement is not signalled to the browser; open to downgrade attacks.

Security Headers & Information Leakage

HTTP SECURITY HEADERS

Header	Status	Description
Strict-Transport-Security	Absent	HTTPS enforcement is not signalled; SSL-stripping risk.
Content-Security-Policy	Absent	No browser defence against XSS/injection.
X-Frame-Options	Absent	Open to clickjacking; can be embedded in an iframe.
X-Content-Type-Options	Absent	MIME-sniffing is possible.
Referrer-Policy	Absent	Referrer information may leak to external sources.
Permissions-Policy	Absent	Sensitive browser APIs are not restricted.
X-XSS-Protection	Absent	The legacy browser XSS filter is not set (not critical in modern browsers).

INFORMATION LEAKAGE / EXPOSED FILES

Common sensitive paths were checked with a single GET (content verified — HTTP 200 alone is not treated as evidence):

Path	Status	Note
`/.git/config`	Closed	HTTP 400 / empty body — not reachable
`/.env`	Closed	HTTP 404 / empty body — not reachable
`/.git/HEAD`	Closed	HTTP 400 / empty body — not reachable
`/backup.zip`	Closed	HTTP 404 / empty body — not reachable
`/.DS_Store`	Closed	HTTP 404 / empty body — not reachable
`/wp-config.php.bak`	Closed	HTTP 404 / empty body — not reachable
`/ftp`	Closed	HTTP 404 / empty body — not reachable
`/backup`	Closed	HTTP 404 / empty body — not reachable
`/backups`	Closed	HTTP 404 / empty body — not reachable
`/uploads`	Closed	HTTP 404 / empty body — not reachable
`/files`	Closed	HTTP 400 / empty body — not reachable
`/admin`	Closed	HTTP 404 / empty body — not reachable

Path	Status	Note
`/.svn/entries`	Closed	HTTP 400 / empty body — not reachable
`/.htaccess`	Closed	HTTP 403 / empty body — not reachable
`/config.php.bak`	Closed	HTTP 404 / empty body — not reachable
`/db.sql`	⚠ **EXPOSED**	expected file format confirmed (not catch-all)
`/dump.sql`	Closed	HTTP 404 / empty body — not reachable
`/backup.tar.gz`	Closed	HTTP 404 / empty body — not reachable
`/backup.tar`	Closed	HTTP 404 / empty body — not reachable
`/www.zip`	Closed	HTTP 404 / empty body — not reachable
`/site.zip`	Closed	HTTP 404 / empty body — not reachable
`/backup.old`	Closed	HTTP 404 / empty body — not reachable
`/backup.backup`	Closed	HTTP 404 / empty body — not reachable
`/index.php.bak`	Closed	HTTP 404 / empty body — not reachable
`/index.php~`	Closed	HTTP 404 / empty body — not reachable
`/.env.bak`	Closed	HTTP 404 / empty body — not reachable
`/.env.old`	Closed	HTTP 404 / empty body — not reachable
`/database.sql`	Closed	HTTP 404 / empty body — not reachable

ADDITIONAL INFORMATION-LEAKAGE OBSERVATIONS

Control	Result
Directory listing (autoindex / "Index of /") · CWE-548	✓ **6 directories tried, no listing**
Verbose error / server-path disclosure · CWE-209	✓ **No indicator found**
Password-field autocomplete policy · CWE-522	✓ **Appropriate / no password field observed**

> All GET-only/passive observation — content is NOT fetched/shown; a leaked path is REDACTED. "No indicator found" does NOT PROVE it is secure; it only shows that no indicator emerged from the passive methods attempted.

IDENTIFIED RISKS

Finding	Severity	Description
Sensitive file accessible (`/db.sql`)	High	Content verified; risk of configuration/source-code leakage. Access must be blocked immediately.
Outdated / unsupported software version disclosed (PHP/7.1.26 — EOL)	High	The PHP 7.x series is officially end-of-life (security updates for 7.x ended in late 2022). Numerous known vulnerabilities remain unpatched. CWE-1104 · OWASP A06:2021 (Vulnerable & Outdated Components). Remediation: upgrade to a

Finding	Severity	Description
		current, supported PHP version (8.2+) and hide the version signature (expose_php=Off).
Out-of-date software version disclosed (Apache/2.4.25 — very old patch level)	Medium	The Apache 2.4 series is supported, but Apache/2.4.25 is a very old patch level; intervening security patches appear not to have been applied. CWE-1104 · OWASP A06:2021 (Vulnerable & Outdated Components). Remediation: upgrade to the current patch of the 2.4 series and hide the version signature (ServerTokens Prod).
Critical security header missing: Content-Security-Policy	Medium	The browser cannot restrict which resources are loaded; there is no basic defence against XSS and content injection. Missing on ALL 2 scanned unique pages.
Critical security header missing: X-Frame-Options	Medium	The page can be embedded in another site's iframe; users can be deceived via clickjacking. Missing on ALL 2 scanned unique pages.
Additional headers missing (Strict-Transport-Security, X-Content-Type-Options, Referrer-Policy, Permissions-Policy, X-XSS-Protection)	Medium	Defence in depth is weak. (missing on 2/2 pages)

DNS & Email Security

SPF (Sender Policy) — checked domain: `vulnweb.com`

- **Status:** Present — `v=spf1 ~all`
- **Strictness:** `~all` — soft (softfail; acceptable, not ideal).

DMARC (Authentication Policy) — checked domain: `vulnweb.com`

- **Status:** Absent — No DMARC record was found. No enforcement is applied based on SPF/DKIM results; protection against spoofing is weak.

DKIM (Signature)

- **Status:** Not detected — No DKIM record was found at common selectors (default/google/selector1...). You may be using a different selector; this does not definitely mean "no record".

DNSSEC

- **Status:** Passive/absent — DNS responses are not signed; more exposed to DNS spoofing/cache-poisoning risk.

IDENTIFIED RISKS

Finding	Severity	Description
DMARC missing	Medium	SPF/DKIM results are not enforced.
DKIM not detected	Informational	Not found at common selectors (may be a different selector).
DNSSEC passive	Informational	DNS responses are not signed.

CORS & Cookie Security

CORS CONFIGURATION (tested on 2/2 pages)

- **Test Origin:** `https://cybertestify-cors-probe.example` — a harmless Origin header was sent to each page and the response evaluated.
- **Most permissive observed policy** (`/`): Access-Control-Allow-Origin: not sent (closed — secure default); Allow-Credentials: not sent.

COOKIE FLAGS (all cookies observed across 2 pages)

- No Set-Cookie was observed on any of the 2 scanned pages.

IDENTIFIED RISKS

- No notable risk stood out in the CORS and cookie configuration.

CSP (Content Security Policy) Analysis

CSP STATUS

- **Status:** Absent — No Content-Security-Policy header is sent at all.

CSP DIRECTIVE ANALYSIS

- As there is no enforced CSP, no directive analysis could be performed.

IDENTIFIED RISKS

Finding	Severity	Description
CSP entirely missing	Medium	No browser-level defence against XSS and content injection. The CSP header is missing on ALL 2 of the scanned pages.

POSITIVE ASSURANCE — AREAS CHECKED

Including the areas with no finding, the external-surface checks were genuinely run across **2 unique pages** including the home page. The table below also shows the "no issue found" results transparently:

Control area	Result
SSL/TLS Configuration Audit	⚠ **Finding present (High — detailed above)**
Security Headers & Information Leakage	⚠ **Finding present (High — detailed above)**
DNS & Email Security	⚠ **Finding present (Medium — detailed above)**
CORS & Cookie Security	✓ **No issue found**
CSP (Content Security Policy) Analysis	⚠ **Finding present (Medium — detailed above)**
Directory listing (autoindex / "Index of /") · CWE-548	✓ **6 directories tried, no listing**
Verbose error / server-path disclosure · CWE-209	✓ **No indicator found**
Password-field autocomplete policy · CWE-522	✓ **Appropriate / no password field observed**

> **Three-state distinction (honesty):** ✓ *No issue found* = the check ran and came back clean · ⚠ *Finding present* = detailed above · ⚠ *Not assessable* = no data could be collected (does NOT mean secure).

What this package DOES and DOES NOT check

> **DOES (passive — only page retrieval via GET + harmless Origin/DNS query):** TLS/certificate, HTTP security headers, CORS policy, cookie flags (Secure/HttpOnly/SameSite), Content-Security-Policy, DNS/email records (SPF/DKIM/DMARC/DNSSEC), exposed sensitive files (incl. common backup patterns), directory listing (autoindex), verbose-error/server-path disclosure (redacted), password-field autocomplete policy, outdated/unsupported software versions — across the 2 discovered pages.

> **DOES NOT:** Active vulnerability verification (payload/probe attempts such as SQLi/XSS/IDOR), authenticated flow testing, business-logic abuse. These are within the scope of the **Active Verification** and **Full-Scope Pentest** packages. This report relies on passive observation; "no finding" in an area **does NOT PROVE** it is secure, because no active exploit was attempted — it only shows that the externally observed configuration is clean.

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 suggestions for all the gaps detected in your external-surface scan. Copy the examples that suit your server (Nginx/Firebase/Apache/DNS).

SSL/TLS Configuration Audit

The following recommendations strengthen the TLS/encryption configuration for rest.vulnweb.com.

1. Add the HSTS header

Tells the browser to connect to the site only over HTTPS (apply only if your site runs entirely over HTTPS):

Nginx:

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

Firebase Hosting — `firebase.json`:

```
{
  "hosting": {
    "headers": [
      {
        "source": "**",
        "headers": [
          { "key": "Strict-Transport-Security", "value": "max-age=31536000; includeSubDomains" }
        ]
      }
    ]
  }
}
```

Apache — `.htaccess`:

```
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
```

Security Headers & Information Leakage

The following recommendations address the missing security headers detected on the home page of rest.vulnweb.com. Each header has a short explanation followed by an Nginx example; at the end you will find ready-made blocks for non-Nginx platforms (Firebase, Vercel, Next.js, Apache). Copy the one that suits your server.

1. Add a Content-Security-Policy (CSP)

This is the most effective browser-level defence against XSS and content injection. Start with a baseline policy suited to your site and tighten it over time:

```
add_header Content-Security-Policy "default-src 'self'; img-src 'self' data:; script-src 'self'; style-src '
```

If you use third-party scripts (GTM, Analytics, Pixel), add the relevant domains to `script-src` / `connect-src`.

2. Add X-Frame-Options

Prevents your page from being embedded in another site's iframe and exposed to clickjacking:

```
add_header X-Frame-Options "SAMEORIGIN" always;
```

As a modern alternative, CSP `frame-ancestors 'self'` provides the same protection.

3. Add X-Content-Type-Options

Prevents the browser from MIME-type sniffing and misinterpreting content (and the XSS risk that follows):

```
add_header X-Content-Type-Options "nosniff" always;
```

4. Add Strict-Transport-Security (HSTS)

Enforces HTTPS and makes SSL-stripping/MITM attacks harder. (Apply only if your site runs entirely over HTTPS.)

```
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
```

5. Add Referrer-Policy

Reduces information leakage via the `Referer` header sent to external links:

```
add_header Referrer-Policy "strict-origin-when-cross-origin" always;
```

6. Add Permissions-Policy

Restricts browser APIs (camera, microphone, location, etc.) and prevents unwanted access by third-party iframes:

```
add_header Permissions-Policy "geolocation=(), microphone=(), camera=()" always;
```

7. X-XSS-Protection (defence-in-depth)

A legacy header for older browsers; the real protection lies in CSP. You may add it optionally:

```
add_header X-XSS-Protection "1; mode=block" always;
```

Combined configuration — Nginx

Add these to your server block (server { ... }), verify with `nginx -t`, then reload with `systemctl reload nginx`:

```
add_header Content-Security-Policy "default-src 'self'; frame-ancestors 'self'" always;
add_header X-Frame-Options "SAMEORIGIN" always;
add_header X-Content-Type-Options "nosniff" always;
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
add_header Referrer-Policy "strict-origin-when-cross-origin" always;
add_header Permissions-Policy "geolocation=(), microphone=(), camera=()" always;
```

Ready-made configuration for other platforms

If your server is not Nginx, you can add the same headers using one of the ready-made blocks below.

Firebase Hosting — `firebase.json`:

```
{
  "hosting": {
    "headers": [
      {
        "source": "**",
        "headers": [
          { "key": "Content-Security-Policy", "value": "default-src 'self'; frame-ancestors 'self'" },
          { "key": "X-Frame-Options", "value": "SAMEORIGIN" },
          { "key": "X-Content-Type-Options", "value": "nosniff" },
          { "key": "Strict-Transport-Security", "value": "max-age=31536000; includeSubDomains" },
          { "key": "Referrer-Policy", "value": "strict-origin-when-cross-origin" },
          { "key": "Permissions-Policy", "value": "geolocation=(), microphone=(), camera=()" }
        ]
      }
    ]
  }
}
```

Vercel — `vercel.json`:

```
{
  "headers": [
    {
      "source": "/(.*)",
      "headers": [
```

```
          { "key": "Content-Security-Policy", "value": "default-src 'self'; frame-ancestors 'self'" },
          { "key": "X-Frame-Options", "value": "SAMEORIGIN" },
          { "key": "X-Content-Type-Options", "value": "nosniff" },
          { "key": "Strict-Transport-Security", "value": "max-age=31536000; includeSubDomains" },
          { "key": "Referrer-Policy", "value": "strict-origin-when-cross-origin" },
          { "key": "Permissions-Policy", "value": "geolocation=(), microphone=(), camera=()" }
      ]
    }
  ]
}
```

Next.js — `next.config.js`:

```
module.exports = {
  async headers() {
    return [
      {
        source: '/(.*)',
        headers: [
          { key: 'Content-Security-Policy', value: 'default-src 'self'; frame-ancestors 'self'' },
          { key: 'X-Frame-Options', value: 'SAMEORIGIN' },
          { key: 'X-Content-Type-Options', value: 'nosniff' },
          { key: 'Strict-Transport-Security', value: 'max-age=31536000; includeSubDomains' },
          { key: 'Referrer-Policy', value: 'strict-origin-when-cross-origin' },
          { key: 'Permissions-Policy', value: 'geolocation=(), microphone=(), camera=()' }
        ],
      },
    ];
  },
};
```

Apache — `.htaccess` (mod_headers):

```
Header always set Content-Security-Policy "default-src 'self'; frame-ancestors 'self'"
Header always set X-Frame-Options "SAMEORIGIN"
Header always set X-Content-Type-Options "nosniff"
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
Header always set Referrer-Policy "strict-origin-when-cross-origin"
Header always set Permissions-Policy "geolocation=(), microphone=(), camera=()"
```

IIS / ASP.NET — `web.config`:

```
<configuration>
  <system.webServer>
    <httpProtocol>
      <customHeaders>
      <add name="Content-Security-Policy" value="default-src 'self'; frame-ancestors 'self'" />
      <add name="X-Frame-Options" value="SAMEORIGIN" />
      <add name="X-Content-Type-Options" value="nosniff" />
      <add name="Strict-Transport-Security" value="max-age=31536000; includeSubDomains" />
      <add name="Referrer-Policy" value="strict-origin-when-cross-origin" />
      <add name="Permissions-Policy" value="geolocation=(), microphone=(), camera=()" />
      </customHeaders>
    </httpProtocol>
```

```
  </system.webServer>
</configuration>
```

After making the changes, verify with your browser developer tools (Network tab) or `curl -I https://rest.vulnweb.com` that the headers are present in the response.

Block access to exposed files

Detected paths: `/db.sql` . Block access at the server level:

Nginx:

```
location ~ /\.(git|env|ht) { deny all; return 404; }
location ~* \.(bak|old|zip|sql)$ { deny all; return 404; }
```

Apache — `.htaccess` :

```
RedirectMatch 404 /\.git
RedirectMatch 404 /\.env
<FilesMatch "\.(bak|old|zip|sql)$">
  Require all denied
</FilesMatch>
```

It is also best not to place these files in the web root (public) at all.

DNS & Email Security

The following recommendations strengthen email authentication for the domain vulnweb.com. Add the records via your DNS provider's (Cloudflare/GoDaddy/...) TXT interface.

Add/harden a DMARC record

First monitor with `p=none` ; once you have reviewed the reports and ruled out false positives, move to `quarantine` → `reject` :

```
_dmarc.vulnweb.com.  TXT  "v=DMARC1; p=quarantine; rua=mailto:dmarc@vulnweb.com; fo=1"
```

Enable DKIM signing

Generate DKIM from your email provider's panel (Google Workspace/Microsoft 365/sending service) and add the TXT record it provides under `selector._domainkey` :

```
google._domainkey.vulnweb.com.  TXT  "v=DKIM1; k=rsa; p=<key-provided-by-your-provider>"
```

Enable DNSSEC

Turn on **DNSSEC** in your DNS provider's panel; the provider generates the DS record, which you enter at your domain registrar. It signs DNS responses and makes cache poisoning harder.

CORS & Cookie Security

The following recommendations strengthen CORS and cookie security for rest.vulnweb.com.

Your CORS and cookie configuration is close to secure defaults. When adding new endpoints, keep the origin-allowlist and cookie-flag (Secure/HttpOnly/SameSite) principle.

CSP (Content Security Policy) Analysis

The following recommendations strengthen the Content-Security-Policy for rest.vulnweb.com. Test the CSP first with the `Content-Security-Policy-Report-Only` header, and only switch to the enforced header once you are sure the site is not

broken.

Baseline (starter) CSP

Extend it by adding your own third-party domains (analytics, CDN, fonts) to `script-src` / `connect-src` / `img-src`:

Nginx:

```
add_header Content-Security-Policy "default-src 'self'; img-src 'self' data:; script-src 'self'; style-src '
```

Firebase Hosting — `firebase.json`:

```
{
  "hosting": {
    "headers": [
      {
        "source": "**",
        "headers": [
          { "key": "Content-Security-Policy", "value": "default-src 'self'; img-src 'self' data:; script-src
        ]
      }
    ]
  }
}
```

Apache — `.htaccess`:

```
Header always set Content-Security-Policy "default-src 'self'; img-src 'self' data:; script-src 'self'; styl
```

Test with Report-Only

```
Content-Security-Policy-Report-Only: default-src 'self'; report-uri /csp-report
```

After monitoring the reports and clearing false positives, publish the header as `Content-Security-Policy`.

5. Appendix — Glossary

SQL Injection	An injection flaw where user input reaches a database query.
XSS	Cross-Site Scripting — injection of malicious scripts into pages.
IDOR	Insecure Direct Object Reference — accessing others' records via ID manipulation.
CWE	Common Weakness Enumeration.
OWASP	Open Web Application Security Project.
TLS	Transport Layer Security.
HSTS	HTTP Strict Transport Security header.
CSP	Content Security Policy.
CORS	Cross-Origin Resource Sharing.

Clickjacking	Tricking clicks via invisible framing.