

Website Security Exposure Audit

What anyone on the internet can see about your site

HolicAudit (holicaudit.pages.dev)

No high-severity exposure was found on our own site, but it is missing two browser protections we would ask any client to add. This is a real audit of our own website, published as the sample. Nothing here is staged.

SEVERITY	MEANING	ITEMS
HIGH	Fix now. Direct exposure of data or a listed vulnerability in software we could identify.	0
MEDIUM	Fix this month. A missing control that makes a common attack easier.	1
LOW	Hardening. Cheap to fix, limited impact on its own.	3
INFO	Context or optional. No action needed unless it fits your setup.	7
PASS	Checks that came back clean.	12

Passive, non-intrusive checks only, run 2026-10-31 04:16:16 UTC with the owner's authorisation. 14 HTTP requests and 5 TLS handshakes in total, at most one per second. This report was prepared with AI assistance and checked against the raw data by the analyst.

Why our own site? We only audit sites whose owners have authorised us. For a public sample, the only site we can honestly use is our own, holicaudit.pages.dev, a static site on Cloudflare Pages.

Summary and fix order

The basics are solid: TLS 1.2/1.3 only, a valid auto-renewing certificate, no cookies, no third-party scripts, no version banners and none of the common sensitive files exposed. The gaps are all response headers: there is no Content-Security-Policy and no protection against framing, which matter more now that the site shows a payment address. Everything listed is fixable in one `_headers` file in about 30 minutes. The DNS and email items are informational because the domain sits under Cloudflare's `pages.dev`, where we can't add records.

Fix in this order

- | | | |
|----|--------|---|
| 01 | MEDIUM | No Content-Security-Policy (CSP)
Add a policy that only allows your own origin, then loosen it only for sources you actually use. Deploy it as Content-Security-Policy-Report-Only for a few days first if the site is complex, then switch to the enforcing header. Your inline script must either move to a .js file (simplest) or be allowed by its SHA-256 hash. Effort: 30–60 min. |
| 02 | LOW | No Strict-Transport-Security (HSTS) header
Send HSTS on every HTTPS response. Start with a shorter max-age if you are unsure every subdomain supports HTTPS. Effort: 10 min. |
| 03 | LOW | No security.txt (no published way to report a vulnerability)
Publish a short text file at <code>/.well-known/security.txt</code> . Effort: 5 min. |
| 04 | LOW | Page can be embedded in other sites' frames (no clickjacking protection)
Forbid framing unless you need it. <code>frame-ancestors</code> in CSP is the modern control; <code>X-Frame-Options</code> covers older browsers. Effort: 5 min. |

Optional / context

- DNSSEC not enabled
- Homepage allows cross-origin reads (`Access-Control-Allow-Origin: *`)
- No CAA record
- No DMARC record
- No Permissions-Policy
- No SPF record
- No cross-origin isolation headers (`COOP`, `CORP`)

Findings in detail

Highest severity first. Each item quotes the raw evidence, explains the risk in plain English and gives the exact fix for your host (Cloudflare Pages).

No Content-Security-Policy (CSP)

MEDIUM

EVIDENCE	Response header Content-Security-Policy is absent. Homepage has 1 inline <script> block(s) and 0 external script(s).
WHY IT MATTERS	CSP is the browser's second line of defence: if an attacker ever injects HTML or script (through a compromised dependency, a form, or a CMS plugin), the policy decides whether that script may run and where it may send data. Without it, any injected script runs with full access to the page.
YOUR SETUP	Static site on Cloudflare Pages with one inline script (the copy-address button) and some inline style attributes. Fastest route: move the inline script to /assets/site.js, then ship the policy in _headers.
FIX	<p>Add a policy that only allows your own origin, then loosen it only for sources you actually use. Deploy it as Content-Security-Policy-Report-Only for a few days first if the site is complex, then switch to the enforcing header. Your inline script must either move to a .js file (simplest) or be allowed by its SHA-256 hash. Effort: 30–60 min.</p> <p>CLOUDFLARE PAGES / NETLIFY (_HEADERS FILE IN THE PUBLISH FOLDER)</p> <pre>/* Content-Security-Policy: default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline'; img-src 'self' data:; font-src 'self'; connect-src 'self'; object-src 'none'; base-uri 'self'; form-action 'self'; frame-ancestors 'none'; upgrade-insecure-requests</pre> <p>Other servers and hosts: see "All header fixes in one place".</p>

No Strict-Transport-Security (HSTS) header

LOW

EVIDENCE	Response header Strict-Transport-Security is absent on <code>https://holicaudit.pages.dev/</code> . <code>hstspreload.org: status 'preloaded' via 'dev'.</code>
WHY IT MATTERS	Because the whole .dev domain is on the browser HSTS preload list, modern browsers already force HTTPS here, so the practical risk is low. Sending the header anyway covers clients that do not ship the preload list and keeps protection if you later move to a domain that is not preloaded.
YOUR SETUP	On Cloudflare Pages, add the header in _headers. Do not use the HSTS preload directive on a *.pages.dev subdomain; it only applies to registrable domains.
FIX	Send HSTS on every HTTPS response. Start with a shorter max-age if you are unsure every subdomain supports HTTPS. Effort: 10 min.

```
/*
Strict-Transport-Security: max-age=31536000; includeSubDomains
```

Other servers and hosts: see "All header fixes in one place".

No security.txt (no published way to report a vulnerability)

LOW

EVIDENCE

```
GET /.well-known/security.txt -> 404
```

WHY IT MATTERS

If a researcher finds a problem, security.txt (RFC 9116) tells them who to email. Without it, reports often never reach you.

YOUR SETUP

Cheap and worth doing: one file in /.well-known/.

FIX

Publish a short text file at /.well-known/security.txt. Effort: 5 min.

/.WELL-KNOWN/SECURITY.TXT

```
Contact: mailto:security@mail.grokbots.com
Expires: 2027-10-31T00:00:00.000Z
Preferred-Languages: en
Canonical: https://holicaudit.pages.dev/.well-known/security.txt
```

Page can be embedded in other sites' frames (no clickjacking protection)

LOW

EVIDENCE

```
Neither X-Frame-Options nor CSP frame-ancestors is set.
```

WHY IT MATTERS

Another site can load your page invisibly inside an iframe and trick visitors into clicking buttons on it (clickjacking). The risk grows if the page has buttons that change something, like an order or copy-to-clipboard action.

YOUR SETUP

The homepage has a copy-to-clipboard button for the USDT address. Framing protection stops another site from overlaying it.

FIX

Forbid framing unless you need it. frame-ancestors in CSP is the modern control; X-Frame-Options covers older browsers. Effort: 5 min.

CLOUDFLARE PAGES / NETLIFY (_HEADERS FILE IN THE PUBLISH FOLDER)

```
/*
X-Frame-Options: DENY
Content-Security-Policy: frame-ancestors 'none'
```

Other servers and hosts: see "All header fixes in one place".

DNSSEC not enabled

INFO

EVIDENCE

```
No DS record for the zone; resolver did not set the AD (authenticated) flag.
```

WHY IT MATTERS

DNSSEC signs DNS answers so they can't be forged in transit. It's good hygiene, with a small operational cost (keys must be managed at the registrar).

YOUR SETUP

Controlled by Cloudflare for pages.dev.

FIX	Enable DNSSEC at your DNS host and add the DS record at your registrar. Effort: 15 min.
-----	---

Homepage allows cross-origin reads (Access-Control-Allow-Origin: *) INFO	
EVIDENCE	Access-Control-Allow-Origin: * on the homepage; no Access-Control-Allow-Credentials header.
WHY IT MATTERS	Any website's JavaScript can read this page's HTML. For public pages with no logged-in content this exposes nothing new, which is why some hosts set it by default. It matters only if private or per-user content is ever served under the same rule.
YOUR SETUP	Cloudflare Pages adds Access-Control-Allow-Origin: * to static responses by default. The site has no logins or private data, so this exposes nothing. It can be removed with ! Access-Control-Allow-Origin in _headers.
FIX	<p>Remove the header if you do not need it, and never combine it with credentials on API routes. Effort: 5 min.</p> <p>CLOUDFLARE PAGES / NETLIFY (_HEADERS)</p> <pre>/* ! Access-Control-Allow-Origin</pre>

No CAA record INFO	
EVIDENCE	CAA lookup on holicaudit.pages.dev and parent domains returned nothing.
WHY IT MATTERS	CAA restricts which certificate authorities may issue certificates for your domain, which limits mis-issuance.
YOUR SETUP	No DNS control on pages.dev. Info here.
FIX	<p>Add CAA records for the CA(s) you actually use (your host may use more than one; check before adding). Effort: 10 min.</p> <p>DNS CAA RECORD</p> <pre>holicaudit.pages.dev CAA 0 issue "letsencrypt.org" holicaudit.pages.dev CAA 0 issue "pki.goog" holicaudit.pages.dev CAA 0 iodef "mailto:security@mail.grokbots.com"</pre>

No DMARC record INFO	
EVIDENCE	TXT _dmarc.holicaudit.pages.dev: none found.
WHY IT MATTERS	DMARC tells receiving mail servers what to do with mail that fails SPF/DKIM and sends you reports. It is the control that actually stops exact-domain spoofing.
YOUR SETUP	Same as SPF: no DNS control on pages.dev. Info for this site; Low on a custom domain that never sends mail.
FIX	For a domain that never sends mail, publish p=reject. Effort: 10 min.

DNS TXT RECORD

```
_dmarc.holicaudit.pages.dev  TXT  "v=DMARC1; p=reject; rua=mailto:dmarc@mail.grokbots.com"
```

No Permissions-Policy

INFO

EVIDENCE

Header absent.

WHY IT MATTERS

This header switches off browser features (camera, microphone, location, payment) the site does not use, so injected or embedded third-party code cannot ask for them. It is hardening, not a fix for a known problem.

FIX

Disable features you do not use. Effort: 5 min.

CLOUDFLARE PAGES / NETLIFY (_HEADERS FILE IN THE PUBLISH FOLDER)

```
/*
  Permissions-Policy: camera=(), microphone=(), geolocation=(), payment=(), usb=
  (), interest-cohort=()
```

Other servers and hosts: see "All header fixes in one place".

No SPF record

INFO

EVIDENCE

TXT holicaudit.pages.dev: no v=spf1 record. MX records: none.

WHY IT MATTERS

Even a domain that never sends email can be spoofed. A 'null' SPF record tells receivers to reject any mail claiming to be from it.

YOUR SETUP

holicaudit.pages.dev is a subdomain of Cloudflare's pages.dev, so we can't add DNS records. Downgraded to Info for that reason. On a custom domain this would be Low.

FIX

Publish an SPF record. Effort: 10 min.

DNS TXT RECORD

```
holicaudit.pages.dev  TXT  "v=spf1 -all"
```

No cross-origin isolation headers (COOP, CORP)

INFO

EVIDENCE

Absent: cross-origin-opener-policy, cross-origin-resource-policy.

WHY IT MATTERS

COOP stops pages you open (or that open you) from keeping a handle on your window; CORP stops other sites from embedding your files. Both are defence-in-depth against cross-site leaks.

FIX

Add both with same-origin values unless you deliberately serve files to other sites. Effort: 5 min.

CLOUDFLARE PAGES / NETLIFY (_HEADERS FILE IN THE PUBLISH FOLDER)

```
/*  
  Cross-Origin-Opener-Policy: same-origin  
  Cross-Origin-Resource-Policy: same-origin
```

Other servers and hosts: see "All header fixes in one place".

All header fixes in one place

Every header recommended in this report, combined. Pick the block for your server or host, deploy it, then reload the site and click through every page: a CSP that is too strict shows up as missing styles or broken buttons (the browser console names the blocked item). If the site is large, ship the CSP as Content-Security-Policy-Report-only for a few days first.

NGINX (SERVER BLOCK)

```
add_header Content-Security-Policy "default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline'; img-src 'self' data:; font-src 'self'; connect-src 'self'; object-src 'none'; base-uri 'self'; form-action 'self'; frame-ancestors 'none'; upgrade-insecure-requests" always;
add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;
add_header X-Frame-Options "DENY" always;
add_header Permissions-Policy "camera=(), microphone=(), geolocation=(), payment=(), usb=(), interest-cohort=()" always;
add_header Cross-Origin-Opener-Policy "same-origin" always;
add_header Cross-Origin-Resource-Policy "same-origin" always;
```

APACHE (.HTACCESS, MOD_HEADERS)

```
Header always set Content-Security-Policy "default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline'; img-src 'self' data:; font-src 'self'; connect-src 'self'; object-src 'none'; base-uri 'self'; form-action 'self'; frame-ancestors 'none'; upgrade-insecure-requests"
Header always set Strict-Transport-Security "max-age=31536000; includeSubDomains"
Header always set X-Frame-Options "DENY"
Header always set Permissions-Policy "camera=(), microphone=(), geolocation=(), payment=(), usb=(), interest-cohort=()"
Header always set Cross-Origin-Opener-Policy "same-origin"
Header always set Cross-Origin-Resource-Policy "same-origin"
```

CLOUDFLARE PAGES / NETLIFY (_HEADERS FILE IN THE PUBLISH FOLDER)

```
/*
  Content-Security-Policy: default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline';
img-src 'self' data:; font-src 'self'; connect-src 'self'; object-src 'none'; base-uri 'self'; form-
action 'self'; frame-ancestors 'none'; upgrade-insecure-requests
  Strict-Transport-Security: max-age=31536000; includeSubDomains
  X-Frame-Options: DENY
  Permissions-Policy: camera=(), microphone=(), geolocation=(), payment=(), usb=(), interest-cohort=
()
  Cross-Origin-Opener-Policy: same-origin
  Cross-Origin-Resource-Policy: same-origin
  ! Access-Control-Allow-Origin
```

VERCEL (VERCEL.JSON)

```
{
  "headers": [
    {
      "source": "/(.*)",
      "headers": [
        {
          "key": "Content-Security-Policy",
          "value": "default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline'; img-src 'self' data:; font-src 'self'; connect-src 'self'; object-src 'none'; base-uri 'self'; form-action 'self'; frame-ancestors 'none'; upgrade-insecure-requests"
        },
        {
```

```

    "key": "Strict-Transport-Security",
    "value": "max-age=31536000; includeSubDomains"
  },
  {
    "key": "X-Frame-Options",
    "value": "DENY"
  },
  {
    "key": "Permissions-Policy",
    "value": "camera=(), microphone=(), geolocation=(), payment=(), usb=(), interest-cohort=()"
  },
  {
    "key": "Cross-Origin-Opener-Policy",
    "value": "same-origin"
  },
  {
    "key": "Cross-Origin-Resource-Policy",
    "value": "same-origin"
  }
]
}
]
}

```

CLOUDFLARE ZONE (ANY ORIGIN)

```

Rules > Transform Rules > Modify Response Header > Set static:
  Content-Security-Policy = default-src 'self'; script-src 'self'; style-src 'self' 'unsafe-inline';
  img-src 'self' data:; font-src 'self'; connect-src 'self'; object-src 'none'; base-uri 'self'; form-
  action 'self'; frame-ancestors 'none'; upgrade-insecure-requests
  Strict-Transport-Security = max-age=31536000; includeSubDomains
  X-Frame-Options = DENY
  Permissions-Policy = camera=(), microphone=(), geolocation=(), payment=(), usb=(), interest-cohort=
  ()
  Cross-Origin-Opener-Policy = same-origin
  Cross-Origin-Resource-Policy = same-origin

```

What came back clean

These checks found nothing to fix. They are listed so you can see exactly what was tested.

	AREA	CHECK AND EVIDENCE
PASS	TLS	Only modern protocols TLS 1.0 and 1.1 refused; TLS 1.2, TLS 1.3 accepted
PASS	TLS	Certificate chain valid C = US, O = Let's Encrypt, CN = YE1; verify result 0 (ok)
PASS	TLS	Certificate not near expiry expires Jan 29 02:58:37 2027 GMT (89 days left); auto-renewed by the platform
PASS	Headers	MIME sniffing disabled X-Content-Type-Options: nosniff
PASS	Headers	Referrer-Policy set strict-origin-when-cross-origin
PASS	Cookies	Cookie flags No cookies set on 3 page(s) checked
PASS	Content	No mixed content No http:// scripts, styles, images, frames or form targets on 3 page(s) checked
PASS	Content	Third-party scripts No third-party scripts on 3 page(s) checked
PASS	Content	No directory listing on / Homepage is not an auto-generated index
PASS	Software	No version numbers in headers server: cloudflare
PASS	Software	No versioned libraries or server software detected Nothing to match against OSV/NVD; see limitations
PASS	Exposure	7 common sensitive paths not exposed /.git/HEAD 404, /.env 404, /wp-config.php.bak 403, /server-status 404, /phpinfo.php 404, /.DS_Store 404, /backup.zip 404

Raw data: software, TLS and headers

Software and versions we could identify

COMPONENT	VERSION	WHERE WE SAW IT
No versioned JavaScript libraries, CMS or server software were identifiable from headers, HTML generator tags or script URLs.		
CVE matching: nothing to look up (no identifiable versions). Sources used when versions are found: OSV.dev and NVD API 2.0.		

TLS

PROTOCOL	RESULT	CIPHER / NOTE
TLS 1.0	refused	server refused handshake
TLS 1.1	refused	server refused handshake
TLS 1.2	accepted	ECDHE-ECDSA-CHACHA20-POLY1305
TLS 1.3	accepted	TLS_AES_256_GCM_SHA384

Certificate: CN = holicaudit.pages.dev · issuer C = US, O = Let's Encrypt, CN = YE1 · valid Oct 31 02:58:38 2026 GMT → Jan 29 02:58:37 2027 GMT · SAN DNS:*.holicaudit.pages.dev, DNS:holicaudit.pages.dev · chain served: 4 certificates · verify: 0 (ok)

Security response headers on https://holicaudit.pages.dev/

strict-transport-security	absent
content-security-policy	absent
content-security-policy-report-only	absent
x-frame-options	absent
x-content-type-options	nosniff
referrer-policy	strict-origin-when-cross-origin
permissions-policy	absent
cross-origin-opener-policy	absent
cross-origin-resource-policy	absent
cross-origin-embedder-policy	absent
access-control-allow-origin	*
access-control-allow-credentials	absent

Raw data: pages and hostnames

Pages checked

PATH	STATUS	CSP	XFO	HSTS	NOSNIFF	REFERRER	COOKIES	3RD-PARTY JS
/	200	—	—	—	yes	yes	0	0
/terms	200	—	—	—	yes	yes	0	0
/privacy	200	—	—	—	yes	yes	0	0

Hostnames in public certificate logs

1 hostnames under holicaudit.pages.dev appear in Certificate Transparency logs (source: crt.sh). We only list them; none were scanned unless you named them in your order. Forgotten ones are worth reviewing or retiring.

holicaudit.pages.dev

Raw data: DNS, email and exposure

DNS and email authentication

A / AAAA	172.66.46.224, 172.66.45.32, 2606:4700:310c::ac42:2ee0, 2606:4700:310c::ac42:2d20
NS	zara.ns.cloudflare.com., nico.ns.cloudflare.com.
MX	none
SPF	none
DMARC	none
DKIM (common selectors)	none of 13 common selectors
CAA	none (checked domain and parents)
DNSSEC (DS present)	holicaudit.pages.dev: no, pages.dev: no

Resolver: dns.google (DNS-over-HTTPS). DKIM keys use custom selector names, so 'none found' only covers the common names we tried.

Common sensitive paths (existence only)

PATH	STATUS	RESULT
/.git/HEAD	404	not exposed
/.env	404	not exposed
/wp-config.php.bak	403	not exposed
/server-status	404	not exposed
/phpinfo.php	404	not exposed
/.DS_Store	404	not exposed
/backup.zip	404	not exposed

Baseline: a random non-existent path returned 404. A path counts as exposed only if it returns 200/206 and its first bytes match the expected file signature. We read at most 256 bytes and stored only the yes/no result, never the content. /wp-config.php.bak returned 403 from Cloudflare's edge rather than our 404 page; that is Cloudflare blocking a well-known attack path, not a sign the file exists (this is a static site with no PHP).

Scope, method and limits

Authorisation

Target: <https://holicaudit.pages.dev>, owned and operated by HolicAudit (us). No third-party site was scanned to produce this sample.

What we did

- Read your public homepage and 2 other page(s) you listed, robots.txt and security.txt, the HTTP response headers, and the TLS handshake (protocol versions, certificate chain).
- Checked 7 well-known sensitive paths for existence only, with a short partial read to rule out false positives.
- Looked up public DNS records (SPF, DMARC, common DKIM selectors, MX, CAA, DNSSEC) through DNS-over-HTTPS.
- Listed hostnames that appear in public Certificate Transparency logs (no requests to those hosts).
- Matched any software versions we could identify against OSV.dev and the NVD.

What we did not do

- No exploitation, no attack payloads, no password guessing, no fuzzing or directory brute-forcing, no logins, no load testing. We use a fixed, short request list at no more than one request per second.
- No authenticated areas, APIs, server internals, source code or third-party services were reviewed.

Limits you should know

- A passive audit sees what the internet sees. It can't prove a site is secure, and a clean result does not mean there are no vulnerabilities.
- Version detection only works when software announces its version. Hidden versions are not checked against CVE lists.
- 'Listed as affected by CVE-...' comes from public databases. Confirm against the vendor's advisory before acting. We did not test whether any issue can be exploited.
- Plain-HTTP behaviour (<http://> → <https://>) could not be tested from our network: our outbound proxy answers plain-HTTP requests itself. For this domain it doesn't matter in browsers, because the whole .dev top-level domain is HSTS-preloaded, so browsers never make a plain-HTTP request to it.
- The site changes often; this reflects the deployment live at the time of the scan.

AI assistance

This report was prepared with AI assistance: software collected the data, and an AI model drafted the explanations from fixed rules. Every finding quotes the raw evidence above, and the analyst checked each one against the data.

Every request we sent

UTC	TYPE	TARGET	RESULT
04:16:16	GET	https://holicaudit.pages.dev/robots.txt	200

UTC	TYPE	TARGET	RESULT
04:16:17	TLS handshake	holicaudit.pages.dev:443 TLS 1.0	refused
04:16:18	TLS handshake	holicaudit.pages.dev:443 TLS 1.1	refused
04:16:20	TLS handshake	holicaudit.pages.dev:443 TLS 1.2	accepted
04:16:21	TLS handshake	holicaudit.pages.dev:443 TLS 1.3	accepted
04:16:22	TLS handshake	holicaudit.pages.dev:443 certificate chain	0 (ok)
04:16:22	GET	https://holicaudit.pages.dev/	200
04:16:23	GET	https://holicaudit.pages.dev/terms	200
04:16:24	GET	https://holicaudit.pages.dev/privacy	200
04:17:25	GET	http://holicaudit.pages.dev/	500
04:17:26	GET	https://holicaudit.pages.dev/holicaudit-404-check-e733c85e4b	404
04:17:27	GET	https://holicaudit.pages.dev/.git/HEAD	404
04:17:29	GET	https://holicaudit.pages.dev/.env	404
04:17:30	GET	https://holicaudit.pages.dev/wp-config.php.bak	403
04:17:31	GET	https://holicaudit.pages.dev/server-status	404
04:17:32	GET	https://holicaudit.pages.dev/phpinfo.php	404
04:17:33	GET	https://holicaudit.pages.dev/.DS_Store	404
04:17:35	GET	https://holicaudit.pages.dev/backup.zip	404
04:17:36	GET	https://holicaudit.pages.dev/.well-known/security.txt	404