

SECURITY REPORT



SCAN TARGET

<https://www.elsazon.com/>

Open ports detected: 0 · Public IP: 198.185.159.144

A risk score of 2/10 (LOW) indicates your business has security issues that need attention. The higher the score, the greater the chance of a breach, ransomware, or data theft.

EXECUTIVE SUMMARY

The security scan of www.elsazon.com found 24 medium and low severity issues. No critical vulnerabilities were detected, but 7 issues should be addressed this week to harden your security posture. Detailed fix instructions are included for each finding.

✓ No high-risk open ports were detected from the internet — your firewall appears to be blocking dangerous services.

TOP RECOMMENDATIONS

- 1 Public IP Flagged for Abuse — 198.185.159.144: If this is a dedicated server, check for signs of compromise (unexpected processes, outbound traffic, unfamiliar scheduled tasks) and consider a malware scan.
- 2 Missing Content-Security-Policy (CSP): Add a Content-Security-Policy header, but roll it out carefully — a misconfigured CSP can break legitimate scripts/styles on the site.
- 3 Environment Secrets Exposed (Unconfirmed): 1.

DETAILED FINDINGS

Public IP Flagged for Abuse — 198.185.159.144

Port HTTPS · IP Reputation

MEDIUM

WHAT IT IS	Public IP 198.185.159.144 (hosted by Squarespace, Inc.) has an abuse confidence score of 29/100 from 25 report(s) in the last 90 days on AbuseIPDB.
BUSINESS RISK	A flagged IP usually means one of two things: this server has already been compromised and is being used to send spam or attack other systems, or this is a shared-hosting IP where a different customer on the same server is doing something malicious — either way, some mail providers and firewalls will treat traffic from this IP with extra suspicion, which can affect email deliverability and how visitors' security software treats your site.
HOW TO FIX	If this is a dedicated server, check for signs of compromise (unexpected processes, outbound traffic, unfamiliar scheduled tasks) and consider a malware scan. If this is shared hosting, contact your hosting provider — ask whether the IP is shared and whether they can move you to a clean one. You can see the specific reports at https://www.abuseipdb.com/check/198.185.159.144 .
URGENCY	Fix within 1 week

Missing Content-Security-Policy (CSP)

Port HTTPS · Content Security
Policy

MEDIUM

WHAT IT IS	Missing Content-Security-Policy (CSP) — this site has no CSP defense-in-depth layer, so if any cross-site scripting weakness exists elsewhere it's easier to exploit; this finding on its own doesn't mean an XSS vulnerability exists on this site
BUSINESS RISK	CSP is a defense-in-depth browser control, not evidence of an active vulnerability by itself — but if an attacker ever manages to slip malicious script onto your site through some other weakness (e.g. a vulnerable plugin or a comment field), a good CSP is often what stops that script from running and stealing customer data such as login sessions or payment details. Without it, that second layer of protection isn't there.
REAL-WORLD EXAMPLE	Example: A vulnerable comment form or compromised ad widget lets an attacker inject a script tag; with a properly scoped CSP in place, the browser would refuse to run it — without CSP, that same injected script runs freely and can forward a visitor's session cookie to the attacker.
HOW TO FIX	Add a Content-Security-Policy header, but roll it out carefully — a misconfigured CSP can break legitimate scripts/styles on the site. Start by deploying it in Report-Only mode first, which reports violations without blocking anything: <code>'add_header Content-Security-Policy-Report-Only "default-src \'self\'; script-src \'self\'; object-src \'none\'; report-uri /csp-report";'</code> Review the reports for a while to catch anything the policy would break, adjust the policy, then switch the header name to 'Content-Security-Policy' (without '-Report-Only') to start enforcing it. For WordPress or complex sites, use https://csp-evaluator.withgoogle.com to help build the policy.
URGENCY	Fix within 1 week
REFERENCE	CWE-693 — Protection Mechanism Failure A security safeguard that should be protecting the system is missing, disabled, or not strong enough.

Environment Secrets Exposed (Unconfirmed)

Port HTTPS · Environment
Secrets Exposed

MEDIUM

WHAT IT IS	The path <code>/.env</code> returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms <i>something</i> is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes credentials or sensitive configuration that should never be public
BUSINESS RISK	Exposed credentials in <code>/.env</code> (API keys, database passwords, AWS access keys) give an attacker direct access to your critical systems — databases, cloud infrastructure, third-party APIs, and internal tools. A single exposed credential can lead to total infrastructure compromise.
REAL-WORLD EXAMPLE	Example: An AWS credentials file exposed at <code>/.env</code> contained a key with full EC2/RDS access. An attacker used it to launch crypto miners on the account, incurring \$50k in charges within 72 hours before discovery. The attacker also captured all database dumps and sold them on underground forums.
HOW TO FIX	1. IMMEDIATELY rotate all exposed credentials (API keys, database passwords, SSH keys). Treat them as fully compromised. 2. Block public access to <code>/.env</code> via web server rules (Nginx: <code>location ~ /\.env { deny all; }</code>). 3. Never commit secrets to version control — use <code>.gitignore</code> and environment variable loaders (python-dotenv, Node's dotenv). 4. Use a secrets manager (AWS Secrets Manager, HashiCorp Vault, Doppler) for production credentials.
URGENCY	Fix within 1 week
REFERENCE	CWE-798 — Use of Hard-coded Credentials A username, password, or key is built directly into the code or configuration, where it can be found and reused by an attacker.

Exposed .git Directory (Unconfirmed)

Port HTTPS · Exposed .git
Directory

MEDIUM

WHAT IT IS	The path <code>/.git/HEAD</code> returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms *something* is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes the site's Git repository metadata, not an admin login
BUSINESS RISK	This is not a login page to brute-force — a publicly accessible <code>/.git/HEAD</code> can let an attacker reconstruct this site's Git repository: full source code, configuration files, commit history, internal file paths, and any secrets that were ever committed, even ones later removed from the latest version but never rotated.
REAL-WORLD EXAMPLE	Example: A free, widely available tool rebuilds this site's entire source code and commit history from the exposed <code>.git</code> folder within minutes — sometimes turning up a hardcoded API key or password that was deleted from the current code but never actually rotated.
HOW TO FIX	Block public access to <code>/.git/HEAD</code> entirely — it should never be servable. Nginx: <code>'location ~ /\.git { deny all; }'</code> . Apache: <code>'<DirectoryMatch "\.git"> Require all denied </DirectoryMatch>'</code> . Better yet, don't deploy the <code>.git</code> folder to the live server at all. If any credentials were ever committed to this repo, rotate them — treat them as compromised.
URGENCY	Fix within 1 week
REFERENCE	CWE-527

WordPress Admin Panel Exposed (Unconfirmed)

Port HTTPS · WordPress Admin Panel Exposed

MEDIUM

WHAT IT IS	The path <code>/wp-admin/</code> returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms *something* is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes administration functionality to anyone on the internet
BUSINESS RISK	Publicly reachable admin panels are a primary target for automated attacks. An attacker who can reach <code>/wp-admin/</code> can attempt brute-force login, exploit known vulnerabilities in the admin software, or leverage it as a stepping stone to full server compromise — without needing any insider knowledge of the site.
REAL-WORLD EXAMPLE	Example: Automated bots constantly scan the internet for paths like <code>/wp-admin/</code> . A business that left a default admin URL accessible was compromised when a bot found it, brute-forced a weak password in minutes, and installed backdoor malware — all without any human attacker being involved.
HOW TO FIX	Restrict access to <code>/wp-admin/</code> by IP allowlist, move it to a non-standard path, or disable it if not needed. Nginx: <code>location ^~ /wp-admin/ { allow YOUR_IP; deny all; }</code> . Also ensure strong, unique credentials are set for any admin accounts, and enable multi-factor authentication where supported.
URGENCY	Fix within 1 week
REFERENCE	CWE-284

phpMyAdmin Exposed (Unconfirmed)

Port HTTPS · phpMyAdmin
Exposed

MEDIUM

WHAT IT IS	The path /phpmyadmin/ returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms *something* is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes administration functionality to anyone on the internet
BUSINESS RISK	Publicly reachable admin panels are a primary target for automated attacks. An attacker who can reach /phpmyadmin/ can attempt brute-force login, exploit known vulnerabilities in the admin software, or leverage it as a stepping stone to full server compromise — without needing any insider knowledge of the site.
REAL-WORLD EXAMPLE	Example: Automated bots constantly scan the internet for paths like /phpmyadmin/. A business that left a default admin URL accessible was compromised when a bot found it, brute-forced a weak password in minutes, and installed backdoor malware — all without any human attacker being involved.
HOW TO FIX	Restrict access to /phpmyadmin/ by IP allowlist, move it to a non-standard path, or disable it if not needed. Nginx: 'location ^~ /phpmyadmin/ { allow YOUR_IP; deny all; }'. Also ensure strong, unique credentials are set for any admin accounts, and enable multi-factor authentication where supported.
URGENCY	Fix within 1 week
REFERENCE	CWE-284

Admin Panel Exposed (Unconfirmed)

Port HTTPS · Admin Panel
Exposed

MEDIUM

WHAT IT IS

The path `/admin.php` returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms **something** is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes administration functionality to anyone on the internet

BUSINESS RISK

Publicly reachable admin panels are a primary target for automated attacks. An attacker who can reach `/admin.php` can attempt brute-force login, exploit known vulnerabilities in the admin software, or leverage it as a stepping stone to full server compromise — without needing any insider knowledge of the site.

REAL-WORLD EXAMPLE

Example: Automated bots constantly scan the internet for paths like `/admin.php`. A business that left a default admin URL accessible was compromised when a bot found it, brute-forced a weak password in minutes, and installed backdoor malware — all without any human attacker being involved.

HOW TO FIX

Restrict access to `/admin.php` by IP allowlist, move it to a non-standard path, or disable it if not needed. Nginx: `location ^~ /admin.php { allow YOUR_IP; deny all; }`. Also ensure strong, unique credentials are set for any admin accounts, and enable multi-factor authentication where supported.

URGENCY

Fix within 1 week

REFERENCE

CWE-284

Missing Referrer-Policy Header

Port HTTPS · Referrer Policy

LOW

WHAT IT IS	Missing Referrer-Policy — page URLs (which may include sensitive data) are shared with third-party sites your pages link to
BUSINESS RISK	If any of your page addresses contain sensitive details (like a password-reset token or account ID), that information could leak to outside sites your pages link to — a small but easily avoidable privacy gap.
REAL-WORLD EXAMPLE	Example: A customer clicks an outbound link from a page whose URL happens to include an account ID or a password-reset token, and that full address — token included — is handed to the destination site in the Referer header.
HOW TO FIX	Add: 'add_header Referrer-Policy "strict-origin-when-cross-origin" always;' (Nginx) or 'Header always set Referrer-Policy strict-origin-when-cross-origin' (Apache).
URGENCY	Fix within 1 month
REFERENCE	CWE-16 — Configuration The weakness comes from how the software was configured, not from a flaw in its code.

Missing Permissions-Policy Header

Port HTTPS · Permissions Policy

LOW

WHAT IT IS	Missing Permissions-Policy — browser features like camera, microphone, and location aren't restricted for embedded third-party scripts
BUSINESS RISK	If you ever embed third-party ads, widgets, or analytics scripts, they could request a visitor's camera, microphone, or location without you intending to allow it — an avoidable privacy risk for your customers.
REAL-WORLD EXAMPLE	Example: An embedded ad network's script requests the visitor's location or microphone access through a permission prompt the site owner never intended to allow, simply because nothing in the page's headers restricted it.
HOW TO FIX	Add: 'add_header Permissions-Policy "camera=(), microphone=(), geolocation=()" always;' Adjust based on what your site actually uses. This limits what ad/analytics scripts can access.
URGENCY	Monitor
REFERENCE	CWE-693 — Protection Mechanism Failure A security safeguard that should be protecting the system is missing, disabled, or not strong enough.

Insecure Cookie Flags (crumb)

Port HTTPS · Cookie Security

LOW

WHAT IT IS	Cookie 'crumb' is missing the HttpOnly, SameSite flag(s). Observed attributes: Secure, Path=/.
BUSINESS RISK	This cookie's name doesn't clearly indicate it holds session or authentication data, so the practical impact depends on what value it actually stores — anywhere from low-stakes (a UI preference) to more sensitive (tracking or personalization data). Missing these flags means whatever the cookie does hold is more exposed than it needs to be to interception or script access.
REAL-WORLD EXAMPLE	Example: whatever value 'crumb' holds could be read by an injected script (no HttpOnly) or intercepted on an unencrypted connection (no Secure) — the actual severity depends on how sensitive that value turns out to be.
HOW TO FIX	Add the missing flag(s) when setting this cookie: Secure (only send over HTTPS), HttpOnly (block JavaScript access), SameSite=Lax or Strict (limit cross-site sending). Most frameworks expose this as a one-line config option — e.g. Flask: <code>app.config['SESSION_COOKIE_SECURE']=True, SESSION_COOKIE_HTTPONLY=True, SESSION_COOKIE_SAMESITE='Lax'.</code>
URGENCY	Fix within 1 month
REFERENCE	CWE-1004, CWE-1275 — Sensitive Cookie Without 'HttpOnly' Flag; Sensitive Cookie with Improper SameSite Attribute The cookie isn't marked HttpOnly, so a malicious script on the page (such as one injected via XSS) can read and steal it. The cookie's SameSite attribute isn't set to a safe value, making it easier for other sites to trick a visitor's browser into sending it in cross-site requests.

AWS Credentials Exposed (Unconfirmed)

Port HTTPS · AWS Credentials
Exposed

LOW

WHAT IT IS	The path <code>/.aws/credentials</code> returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms *something* is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes credentials or sensitive configuration that should never be public
BUSINESS RISK	Exposed credentials in <code>/.aws/credentials</code> (API keys, database passwords, AWS access keys) give an attacker direct access to your critical systems — databases, cloud infrastructure, third-party APIs, and internal tools. A single exposed credential can lead to total infrastructure compromise.
REAL-WORLD EXAMPLE	Example: An AWS credentials file exposed at <code>/.aws/credentials</code> contained a key with full EC2/RDS access. An attacker used it to launch crypto miners on the account, incurring \$50k in charges within 72 hours before discovery. The attacker also captured all database dumps and sold them on underground forums.
HOW TO FIX	1. IMMEDIATELY rotate all exposed credentials (API keys, database passwords, SSH keys). Treat them as fully compromised. 2. Block public access to <code>/.aws/credentials</code> via web server rules (Nginx: <code>location ~ /\.env { deny all; }</code>). 3. Never commit secrets to version control — use <code>.gitignore</code> and environment variable loaders (python-dotenv, Node's dotenv). 4. Use a secrets manager (AWS Secrets Manager, HashiCorp Vault, Doppler) for production credentials.
URGENCY	Fix within 1 month
REFERENCE	CWE-798 — Use of Hard-coded Credentials A username, password, or key is built directly into the code or configuration, where it can be found and reused by an attacker.

SSH Private Key Exposed (Unconfirmed)

Port HTTPS · SSH Private Key
Exposed

LOW

WHAT IT IS	The path <code>/.id_rsa</code> returns a restricted-access response (401/403) that is genuinely distinct from how this site handles random nonexistent paths — this confirms <i>something</i> is being specially handled at this path, but the response contains no content confirming what it actually is — this exposes credentials or sensitive configuration that should never be public
BUSINESS RISK	Exposed credentials in <code>/.id_rsa</code> (API keys, database passwords, AWS access keys) give an attacker direct access to your critical systems — databases, cloud infrastructure, third-party APIs, and internal tools. A single exposed credential can lead to total infrastructure compromise.
REAL-WORLD EXAMPLE	Example: An AWS credentials file exposed at <code>/.id_rsa</code> contained a key with full EC2/RDS access. An attacker used it to launch crypto miners on the account, incurring \$50k in charges within 72 hours before discovery. The attacker also captured all database dumps and sold them on underground forums.
HOW TO FIX	1. IMMEDIATELY rotate all exposed credentials (API keys, database passwords, SSH keys). Treat them as fully compromised. 2. Block public access to <code>/.id_rsa</code> via web server rules (Nginx: <code>location ~ /\.env { deny all; }</code>). 3. Never commit secrets to version control — use <code>.gitignore</code> and environment variable loaders (python-dotenv, Node's dotenv). 4. Use a secrets manager (AWS Secrets Manager, HashiCorp Vault, Doppler) for production credentials.
URGENCY	Fix within 1 month
REFERENCE	CWE-798 — Use of Hard-coded Credentials A username, password, or key is built directly into the code or configuration, where it can be found and reused by an attacker.

Missing SPF Record — Email Spoofing Possible

Port DNS · SPF Record

LOW

WHAT IT IS	No SPF record — there's no DNS-level list of which mail servers are authorized to send using this domain's envelope sender, so any server can pass an SPF check for your domain (note: this domain publishes no MX records, so it may not send email at all — adding SPF/DMARC here is preventative, to stop spoofing of a non-sending domain, and is low priority)
BUSINESS RISK	Without SPF, nothing tells receiving mail servers which senders are legitimate for this domain's SMTP envelope, making it easier for forged mail to get through — and easier for your own legitimate mail to be misjudged as spam. Note that SPF on its own, even correctly configured, doesn't stop the visible 'From:' address a recipient sees from being spoofed — that requires DMARC (see below) to enforce alignment between SPF/DKIM and the From header.
REAL-WORLD EXAMPLE	Example: A customer receives an email that looks exactly like it's from the business — same display name, same domain — asking them to 'confirm' a payment or click a link, with nothing in DNS to stop the forgery or warn the recipient.
HOW TO FIX	Add an SPF TXT record to your DNS listing every service that actually sends mail for this domain — leaving one out will cause its mail to fail SPF and possibly get rejected. Log into your domain registrar, go to DNS, add a TXT record for '@' with a value matching your provider: Google Workspace: 'v=spf1 include:_spf.google.com -all' Microsoft 365: 'v=spf1 include:spf.protection.outlook.com -all'. If you use additional senders (helpdesk, marketing platform, invoicing tool, etc.), add each as its own 'include:' entry in the same record — a domain can only have one SPF TXT record, so don't create a second one. Use https://mxtoolbox.com/spf.aspx to verify, then add a DMARC record too so the visible From address is covered as well.
URGENCY	Fix within 1 month
REFERENCE	CWE-290 — Authentication Bypass by Spoofing A system trusts information that can be forged, so an attacker can impersonate a trusted party — for email, missing SPF/DMARC/DKIM lets a sender fake your domain.

Missing DMARC Record — Email Impersonation Risk

Port DNS · DMARC Record

LOW

WHAT IT IS	No DMARC record — your domain has no DMARC email-authentication policy, which makes it easier to impersonate your business in email (note: this domain publishes no MX records, so it may not send email at all — this is preventative hardening and low priority)
BUSINESS RISK	This makes it significantly easier for scammers to send convincing fake emails 'from' your business — a common tactic in invoice fraud and phishing — which can directly cost your customers money and damage trust in your brand.
REAL-WORLD EXAMPLE	Example: A scammer sends an invoice-fraud email that appears to come straight from the business's own domain; with no DMARC record in place, nothing tells the recipient's mail provider the message is forged, so it lands in the inbox looking completely legitimate.
HOW TO FIX	Add a DMARC TXT record to your DNS. Go to your domain registrar's DNS settings, add a TXT record for '_dmarc' (not '@') with value: 'v=DMARC1; p=none; rua=mailto:youremail@yourdomain.com' Start at p=none — this only turns on reporting, nothing is blocked yet. Review the aggregate reports for a few weeks to confirm every legitimate mail source for this domain (email provider, helpdesk, marketing tools, etc.) is passing DMARC alignment, then move to p=quarantine, and finally p=reject once you're confident nothing legitimate will be caught. Verify at https://mxtoolbox.com/dmarc.aspx
URGENCY	Fix within 1 month
REFERENCE	CWE-290 — Authentication Bypass by Spoofing A system trusts information that can be forged, so an attacker can impersonate a trusted party — for email, missing SPF/DMARC/DKIM lets a sender fake your domain.

Protected Path — /.env

Port PATH · Content Discovery

LOW

WHAT IT IS	/.env returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /config.php

Port PATH · Content Discovery

LOW

WHAT IT IS	/config.php returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /.git/HEAD

Port PATH · Content Discovery

LOW

WHAT IT IS	/.git/HEAD returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /wp-config.php.bak

Port PATH · Content Discovery

LOW

WHAT IT IS	/wp-config.php.bak returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /.git/config

Port PATH · Content Discovery

LOW

WHAT IT IS	/.git/config returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /.htpasswd

Port PATH · Content Discovery

LOW

WHAT IT IS	/.htpasswd returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /.svn/entries

Port PATH · Content Discovery

LOW

WHAT IT IS	/.svn/entries returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Protected Path — /phpinfo.php

Port PATH · Content Discovery

LOW

WHAT IT IS	/phpinfo.php returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 like this is often a generic WAF / default-deny rule rather than proof a real file exists at this path — verify manually before treating it as a real resource.
HOW TO FIX	No action needed if this is intentionally restricted or is just a WAF default-deny. If it maps to a real protected resource, confirm the protection is deliberate and any credentials aren't left at a default.
URGENCY	Fix within 1 month
REFERENCE	CWE-284

Hosting Network Identified

Port HTTPS · IP WHOIS /
Network Ownership

INFO

WHAT IT IS	Public IP 198.185.159.144 is registered to Squarespace, Inc. — network: SQUARESPACE — block: 198.185.159.0 – 198.185.159.255.
BUSINESS RISK	This is informational — it tells you (and anyone else who looks it up) which hosting provider or network actually operates this IP. Useful for confirming you're hosted where you expect, or for context if the IP reputation check above flags anything.
HOW TO FIX	No action required — this is identifying information, not a vulnerability.
URGENCY	Informational

WHAT IT IS	No DKIM record found under common selector names — this does not confirm DKIM is unconfigured, only that it wasn't found under any of the selector names checked; many providers use a custom or provider-specific selector this check can't guess
BUSINESS RISK	If DKIM genuinely isn't configured, email providers increasingly use it as a trust signal, and its absence can make legitimate emails more likely to be flagged as suspicious or land in spam. But this check only queries a fixed list of common selector names (default, google, mail, etc.) — many providers assign a random or account-specific selector, so this finding should be treated as 'couldn't confirm DKIM,' not 'DKIM is definitely missing.'
REAL-WORLD EXAMPLE	Example: A legitimate invoice email from the business gets flagged as suspicious or dropped into spam by the recipient's mail provider, simply because there's no DKIM signature to prove the message wasn't altered or forged in transit — this only actually happens if DKIM is truly unconfigured, which this check alone can't confirm.
HOW TO FIX	First confirm whether DKIM is actually configured: check your email provider's admin console (Google Workspace: Admin console > Apps > Gmail > Authenticate email; Microsoft 365: Defender > Email authentication > DKIM) for the exact selector name in use, since it's often not one of the common defaults this scan checks. If it turns out DKIM genuinely isn't enabled, turn it on there and add the resulting TXT record to DNS. If you're not sure how to check, share the selector name your provider gives you and this can be verified directly.
URGENCY	Fix within 1 month
REFERENCE	CWE-290 — Authentication Bypass by Spoofing A system trusts information that can be forged, so an attacker can impersonate a trusted party — for email, missing SPF/DMARC/DKIM lets a sender fake your domain.

PORT REFERENCE TABLE

PORT	SERVICE	RISK	DESCRIPTION
HTTPS	IP Reputation	MEDIUM	Public IP 198.185.159.144 (hosted by Squarespace, Inc.) has an abuse confidence score of 29/100 from
HTTPS	Content Security Policy	MEDIUM	Missing Content-Security-Policy (CSP) — this site has no CSP defense-in-depth layer, so if any cross
HTTPS	Environment Secrets Exposed	MEDIUM	The path /.env returns a restricted-access response (401/403) that is genuinely distinct from how th
HTTPS	Exposed .git Directory	MEDIUM	The path /.git/HEAD returns a restricted-access response (401/403) that is genuinely distinct from h

HTTPS	WordPress Admin Panel Exposed	MEDIUM	The path /wp-admin/ returns a restricted-access response (401/403) that is genuinely distinct from h
HTTPS	phpMyAdmin Exposed	MEDIUM	The path /phpmyadmin/ returns a restricted-access response (401/403) that is genuinely distinct from
HTTPS	Admin Panel Exposed	MEDIUM	The path /admin.php returns a restricted-access response (401/403) that is genuinely distinct from h
HTTPS	Referrer Policy	LOW	Missing Referrer-Policy — page URLs (which may include sensitive data) are shared with third-party s
HTTPS	Permissions Policy	LOW	Missing Permissions-Policy — browser features like camera, microphone, and location aren't restricte
HTTPS	Cookie Security	LOW	Cookie 'crumb' is missing the HttpOnly, SameSite flag(s). Observed attributes: Secure, Path=/.
HTTPS	AWS Credentials Exposed	LOW	The path /.aws/credentials returns a restricted-access response (401/403) that is genuinely distinct
HTTPS	SSH Private Key Exposed	LOW	The path /.id_rsa returns a restricted-access response (401/403) that is genuinely distinct from how
DNS	SPF Record	LOW	No SPF record — there's no DNS-level list of which mail servers are authorized to send using this do
DNS	DMARC Record	LOW	No DMARC record — your domain has no DMARC email-authentication policy, which makes it easier to imp
PATH	Content Discovery	LOW	/.env returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a 403 I
PATH	Content Discovery	LOW	/config.php returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a
PATH	Content Discovery	LOW	/.git/HEAD returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a
PATH	Content Discovery	LOW	/wp-config.php.bak returned HTTP 403 — access control blocked the request and nothing was accessed.
PATH	Content Discovery	LOW	/.git/config returned HTTP 403 — access control blocked the request and nothing was accessed. Note:
PATH	Content Discovery	LOW	/.htpasswd returned HTTP 403 — access control blocked the request and nothing was accessed. Note: a
PATH	Content Discovery	LOW	/.svn/entries returned HTTP 403 — access control blocked the request and nothing was accessed. Note:
PATH	Content Discovery	LOW	/phpinfo.php returned HTTP 403 — access control blocked the request and nothing was accessed. Note:

HTTPS	IP WHOIS / Network Ownership	INFO	Public IP 198.185.159.144 is registered to Squarespace, Inc. — network: SQUARESPACE — block: 198.185
DNS	DKIM Record	INFO	No DKIM record found under common selector names — this does not confirm DKIM is unconfigured, only

NEXT STEPS

→ Address the 7 MEDIUM risk finding(s) this week. Start with 'Public IP Flagged for Abuse — 198.185.159.144': If this is a dedicated server, check for signs of compromise (unexpected processes, outbound traffic, unfamiliar scheduled tasks) and consider a malware scan.

This report is for informational purposes only and represents a point-in-time automated scan. It is not a substitute for a professional penetration test.