

SECURITY REPORT



SCAN TARGET

https://www.chess.com

Open ports detected: 0 · Public IP: 104.18.137.67

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.chess.com found 16 medium and low severity issues. No critical vulnerabilities were detected, but 3 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 HTTP Not Redirecting to HTTPS: Add a permanent redirect from HTTP to HTTPS.
- 2 Insecure Cookie Flags (psid): 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).
- 3 CSP script-src Too Permissive: Tighten script-src to an explicit allowlist of trusted domains and drop 'unsafe-inline'/'unsafe-eval'/wildcards.

DETAILED FINDINGS

HTTP Not Redirecting to HTTPS

Port HTTP · HTTPS Redirect

MEDIUM

WHAT IT IS	Visiting <code>http://</code> doesn't redirect to <code>https://</code> — some visitors may use an unencrypted connection without knowing
BUSINESS RISK	Anyone who types or clicks an <code>http://</code> link is sending their activity on your site — potentially including form data or passwords — unencrypted, where it can be read by anyone on the same public wifi or compromised network.
REAL-WORLD EXAMPLE	Example: A customer types the domain into their browser without <code>'https://'</code> , lands on the unencrypted version of the site, and submits a form before ever reaching the secure page — sending that data in plain text the whole time.
HOW TO FIX	Add a permanent redirect from HTTP to HTTPS. Nginx: add <code>'return 301 https://\$host\$request_uri;'</code> in your port 80 server block. Apache: add <code>'RewriteRule ^https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]'</code> in your <code>.htaccess</code> . Cloudflare users: go to <code>SSL/TLS > Edge Certificates > enable 'Always Use HTTPS'</code> .
URGENCY	Fix within 1 week
REFERENCE	CWE-319 — Cleartext Transmission of Sensitive Information Sensitive data is sent over the network unencrypted, so anyone monitoring the connection can read it.

Insecure Cookie Flags (psid)

Port HTTPS · Cookie Security

MEDIUM

WHAT IT IS	Cookie 'psid' is missing the HttpOnly flag(s). Observed attributes: expires=Sat, 03 Oct 2026 09:44:50 GMT, Max-Age=1800, path=/, domain=chess.com, secure, samesite=lax.
BUSINESS RISK	This cookie's name suggests it may hold a session or authentication token, though its actual contents weren't inspected. If it does, a cookie missing these flags is easier to steal through cross-site scripting or to intercept over an unencrypted connection — and a stolen session cookie can let an attacker impersonate that logged-in user without ever needing their password.
REAL-WORLD EXAMPLE	Example: A visitor on public wifi has their session cookie intercepted because it wasn't marked Secure, or a malicious ad script reads it directly because it wasn't marked HttpOnly — either way, the attacker is now logged in as that user without ever seeing their password.
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 week
REFERENCE	CWE-1004 — Sensitive Cookie Without 'HttpOnly' Flag The cookie isn't marked HttpOnly, so a malicious script on the page (such as one injected via XSS) can read and steal it.

CSP script-src Too Permissive

Port HTTPS · Content Security Policy

MEDIUM

WHAT IT IS	CSP script-src allows 'unsafe-eval' — an injected or hijacked script can still run freely
BUSINESS RISK	This setting defeats most of the protection CSP is meant to provide. If a third-party script you rely on is ever compromised — the exact mechanism behind most 'watering hole' attacks — it will execute without restriction and with no warning.
REAL-WORLD EXAMPLE	Example: A vendor's analytics or chat-widget script gets compromised at the source; because the policy still allows it (or allows inline/eval scripts generally), the malicious code runs exactly as if it belonged on the site, with no warning to anyone.
HOW TO FIX	Tighten script-src to an explicit allowlist of trusted domains and drop 'unsafe-inline'/'unsafe-eval'/wildcards. Move inline scripts to external files or use a nonce/hash. Test changes with https://csp-evaluator.withgoogle.com before deploying.
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.

Public IP Flagged for Abuse — 104.18.137.67

Port HTTPS · IP Reputation

LOW

WHAT IT IS	Public IP 104.18.137.67 (hosted by Cloudflare, Inc.) has an abuse confidence score of 0/100 from 2 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/104.18.137.67 .
URGENCY	Monitor

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 (ATtribution_V1)

Port HTTPS · Cookie Security

LOW

WHAT IT IS	Cookie 'ATtribution_V1' is missing the Secure, HttpOnly flag(s). Observed attributes: expires=Tue, 03 Nov 2026 10:14:50 GMT, Max-Age=2682000, path=/, domain=chess.com, samesite=lax.
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 'ATtribution_V1' 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-614, CWE-1004 — Sensitive Cookie in HTTPS Session Without 'Secure' Attribute; Sensitive Cookie Without 'HttpOnly' Flag The cookie isn't marked Secure, so a browser could send it over an unencrypted connection where it can be intercepted. The cookie isn't marked HttpOnly, so a malicious script on the page (such as one injected via XSS) can read and steal it.

Insecure Cookie Flags (me)

Port HTTPS · Cookie Security

LOW

WHAT IT IS	Cookie 'me' is missing the Secure, HttpOnly flag(s). Observed attributes: expires=Sun, 03 Oct 2027 09:14:50 GMT, Max-Age=31536000, path=/, domain=chess.com, samesite=strict.
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 'me' 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-614, CWE-1004 — Sensitive Cookie in HTTPS Session Without 'Secure' Attribute; Sensitive Cookie Without 'HttpOnly' Flag The cookie isn't marked Secure, so a browser could send it over an unencrypted connection where it can be intercepted. The cookie isn't marked HttpOnly, so a malicious script on the page (such as one injected via XSS) can read and steal it.

Insecure Cookie Flags (BUCKETING_ID_V2)

Port HTTPS · Cookie Security

LOW

WHAT IT IS	Cookie 'BUCKETING_ID_V2' is missing the HttpOnly flag(s). Observed attributes: expires=Sun, 03 Oct 2027 09:14:50 GMT, Max-Age=31536000, path=/, domain=chess.com, secure, samesite=lax.
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 'BUCKETING_ID_V2' 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 — Sensitive Cookie Without 'HttpOnly' Flag The cookie isn't marked HttpOnly, so a malicious script on the page (such as one injected via XSS) can read and steal it.

Insecure Cookie Flags (visitorid)

Port HTTPS · Cookie Security

LOW

WHAT IT IS	Cookie 'visitorid' is missing the Secure flag(s). Observed attributes: expires=Sun, 03 Oct 2027 09:14:50 GMT, Max-Age=31536000, path=/, httponly, samesite=lax.
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 'visitorid' 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-614 — Sensitive Cookie in HTTPS Session Without 'Secure' Attribute The cookie isn't marked Secure, so a browser could send it over an unencrypted connection where it can be intercepted.

Insecure Cookie Flags (BUCKETING_ID)

Port HTTPS · Cookie Security

LOW

WHAT IT IS	Cookie 'BUCKETING_ID' is missing the HttpOnly flag(s). Observed attributes: path=/, domain=.chess.com, secure, samesite=lax.
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 'BUCKETING_ID' 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 — Sensitive Cookie Without 'HttpOnly' Flag The cookie isn't marked HttpOnly, so a malicious script on the page (such as one injected via XSS) can read and steal it.

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.

Hosting Network Identified

Port HTTPS · IP WHOIS /
Network Ownership

INFO

WHAT IT IS	Public IP 104.18.137.67 is registered to Cloudflare, Inc. — network: CLOUDFLARENET — block: 104.16.0.0 – 104.31.255.255.
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

DKIM Not Detected Under Common Selectors

Port DNS · DKIM Record

INFO

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.

Content Discovery Scan Throttled

Port PATH · Content Discovery

INFO

WHAT IT IS

The target began responding with repeated rate-limit/unavailable statuses, so the remaining 89 path(s) in this scan were skipped rather than risk overloading the server.

HOW TO FIX

Re-run content discovery later, or use a lower-concurrency profile.

URGENCY

Monitor

PORT REFERENCE TABLE

PORT	SERVICE	RISK	DESCRIPTION
------	---------	------	-------------

HTTP	HTTPS Redirect	MEDIUM	Visiting http:// doesn't redirect to https:// — some visitors may use an unencrypted connection with
HTTPS	Cookie Security	MEDIUM	Cookie 'psid' is missing the HttpOnly flag(s). Observed attributes: expires=Sat, 03 Oct 2026 09:44:5
HTTPS	Content Security Policy	MEDIUM	CSP script-src allows 'unsafe-eval' — an injected or hijacked script can still run freely
HTTPS	IP Reputation	LOW	Public IP 104.18.137.67 (hosted by Cloudflare, Inc.) has an abuse confidence score of 0/100 from 2 r
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 'ATTRIBUTION_V1' is missing the Secure, HttpOnly flag(s). Observed attributes: expires=Tue, 0
HTTPS	Cookie Security	LOW	Cookie 'me' is missing the Secure, HttpOnly flag(s). Observed attributes: expires=Sun, 03 Oct 2027 0
HTTPS	Cookie Security	LOW	Cookie 'BUCKETING_ID_V2' is missing the HttpOnly flag(s). Observed attributes: expires=Sun, 03 Oct 2
HTTPS	Cookie Security	LOW	Cookie 'visitorid' is missing the Secure flag(s). Observed attributes: expires=Sun, 03 Oct 2027 09:1
HTTPS	Cookie Security	LOW	Cookie 'BUCKETING_ID' is missing the HttpOnly flag(s). Observed attributes: path=/, domain=.chess.co
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
HTTPS	IP WHOIS / Network Ownership	INFO	Public IP 104.18.137.67 is registered to Cloudflare, Inc. — network: CLOUDFLARENET — block: 104.16.0
DNS	DKIM Record	INFO	No DKIM record found under common selector names — this does not confirm DKIM is unconfigured, only
PATH	Content Discovery	INFO	The target began responding with repeated rate-limit/unavailable statuses, so the remaining 89 path(

NEXT STEPS

→ Address the 3 MEDIUM risk finding(s) this week. Start with 'HTTP Not Redirecting to HTTPS': Add a permanent redirect from HTTP to HTTPS.

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.