NeoShield Quantum X

Website security reporting for freelance developers and web agencies

CLIENT REPORT

Website Security Remediation Report

www.client-shop.example

Passive assessment of publicly accessible security configuration



Observed configuration score

Coverage 100% · confidence Medium

Prepared for	**Example Retail Co. (demonstration)**
Prepared by	**Your agency name · NeoShield Quantum X**
Website	**https://www.client-shop.example/**
Report ID	**DEMO-2026-0001**
Assessment date	**2026-10-01 · Passive assessment**
Verification scan	**2026-10-08**
Status	**4 confirmed findings (0 high or critical), 0 contextual observations**

Sample report using demonstration data.

CONFIDENTIAL

This report describes the security configuration of the website named above. Share it only with people responsible for the website. It is not a penetration test and does not certify that the website is secure.

Contents

Executive summary 3
How the score was calculated 4
Findings overview 5
Findings, evidence and remediation 6
Checks passed 13
Remediation roadmap 14
Appendix A: Standards mapping 15
Appendix B: Definitions 15
Appendix C: Scope, authorization and method 15
Appendix D: Limitations 16
Appendix E: Report integrity 16

Section titles are clickable, and the same sections are available as PDF bookmarks.

Executive summary

Observed configuration score	Coverage	Contextual review	Immediate action
55/100 · D	100%	0	None confirmed

17 of 17 applicable checks completed · 4 confirmed findings · 2 verified fixed · Confidence: Medium · Passive assessment

This score covers only the publicly observable configuration checks completed in this assessment. It does not measure application vulnerabilities or overall business risk.

Confirmed findings were observed directly and reduce the score. Contextual observations depend on how the website, DNS and email services are used; they do not reduce the score.

Severity of all findings in this report



Items to act on or review, by area



What this means for the business

The website is reachable over HTTPS with a valid certificate, but several browser security controls are missing. The most important open gaps are the missing Content-Security-Policy, a defense-in-depth control, and session cookies without the Secure and HttpOnly flags. Two issues were fixed and verified by a follow-up scan on 2026-10-08. Four issues remain open and are listed in priority order with configuration examples. This assessment covers only publicly observable configuration at the stated time. It does not assess authenticated functionality, source code, internal infrastructure, business logic, or all application vulnerabilities.

Technical summary

Passive assessment of publicly observable HTTP response headers, TLS configuration, cookies, redirects, email-authentication DNS records and common exposed paths. 6 findings (1 high, 3 medium, 2 low); 2 verified fixed; 11 checks passed; DKIM not assessed (no selector supplied).

Items requiring review

1	Medium	Content-Security-Policy header not set	Remediate within 30 days
2	Medium	Session cookie missing Secure and HttpOnly flags	Remediate within 30 days
3	Medium	No clickjacking protection	Remediate within 30 days
4	Low	Server software version disclosed	Review within 90 days

After verified fixes: 73/100, grade C

How the score was calculated

The score starts at 100. Each confirmed open finding deducts a fixed number of points by severity (critical 25, high 15, medium 8, low 3); a root cause is charged once. Items to review, informational items and checks that could not run do not deduct points.

Finding	Severity	Points
Strict-Transport-Security header missing	High	–15
Content-Security-Policy header not set	Medium	–8
Session cookie missing Secure and HttpOnly flags	Medium	–8
No clickjacking protection	Medium	–8
X-Content-Type-Options header missing	Low	–3
Server software version disclosed	Low	–3
Total deductions		–45

100 – 45 = 55/100, grade D.

Grades: A 90–100, B 80–89, C 70–79, D 55–69, F below 55.

Coverage and confidence

Coverage	100% · 17 of 17 applicable checks completed
Confidence	Medium · some confirmed findings need interpretation
Not completed	DKIM (no selector supplied)
Not applicable	1 (did not apply to this website)

Coverage shows how much of the planned assessment produced a result. Checks that could not run are listed so they can be completed or reviewed by hand; they are never counted as passed.

Findings overview

Ref	Finding	Severity	Priority	Classification	Workflow status
F-01	Content-Security-Policy header not set	Medium	P2	Confirmed finding	Still observed
F-03	Session cookie missing Secure and HttpOnly flags	Medium	P2	Confirmed finding	Still observed
F-04	No clickjacking protection	Medium	P2	Confirmed finding	Still observed
F-06	Server software version disclosed	Low	P3	Confirmed finding	Still observed
F-02	Strict-Transport-Security header missing	High	P1	Confirmed finding	Resolved
F-05	X-Content-Type-Options header missing	Low	P3	Confirmed finding	Resolved

Priority: P0 immediate action, P1 remediate within 7 days, P2 within 30 days, P3 review within 90 days, P4 optional hardening consideration. Only confirmed findings reduce the score. See Appendix B.

Findings, evidence and remediation

F-01 Content-Security-Policy header not set

Medium

Finding ID	DEMO-2026-0001-F-01 · rule csp
Classification	Confirmed finding: Observed directly in the website's responses or DNS. Reduces the score.
Workflow status	Still observed
Fix priority	P2 · Remediate within 30 days
Score impact	–8 points
Confidence	High
Affected URL	https://www.client-shop.example/
Detection method	Passive HTTP response header inspection (GET /, redirects followed).
Observed at	2026-10-01T09:14:05Z
Evidence	GET / responded HTTP/2 200 with no Content-Security-Policy or Content-Security-Policy-Report-Only header.
Why it matters	Without a Content Security Policy, the browser has fewer restrictions available to reduce the impact of a successful content-injection or cross-site scripting vulnerability.
For the client	The website does not tell browsers which code it is allowed to run. If an attacker ever manages to slip code onto a page, nothing stops it from running in visitors' browsers. Adding the policy takes testing, because a strict policy can break parts of the site.
Potential business impact	If a script injection flaw is ever introduced, for example through a plugin, there is no browser-side control limiting what injected code can load or send.
Technical impact	No script-src, object-src, base-uri or frame-ancestors restrictions are applied by the browser.
Limitation	The absence of a Content Security Policy does not independently prove that the website is vulnerable to cross-site scripting. CSP is a defense-in-depth control.
Remediation	Start with a report-only policy, review the reports, then enforce it.
Verification criteria	A rescan observes an enforced Content-Security-Policy header that restricts script sources. A report-only policy is recorded as partially resolved.
Verification status	Still present on the latest scan
Human validation	Required before remediation is planned.
Standards	OWASP A05:2021 · CWE-693 · NIST SP 800-53 SC-18

Configuration examples by platform

Nginx

```
# server { } block (or http { })
add_header Content-Security-Policy-Report-Only "default-src 'self'; object-src 'none'; base-uri 'self';
frame-ancestors 'self'" always;
```

Start in Report-Only mode, add the origins your pages really use, then rename the header to Content-Security-Policy to enforce it. Reload with: nginx -t && systemctl reload nginx. add_header in a location block replaces headers set at server level.

Apache HTTP Server

```
# Requires mod_headers (a2enmod headers)
Header always set Content-Security-Policy-Report-Only "default-src 'self'; object-src 'none'; base-uri 'self';
 frame-ancestors 'self'"
```

Start in Report-Only mode, add the origins your pages really use, then rename the header to Content-Security-Policy to enforce it. Place in the VirtualHost or .htaccess, then run apachectl configtest and reload.

Microsoft IIS

```
<?xml version="1.0" encoding="utf-8"?>
<!-- web.config: merge into the existing <configuration> element -->
<configuration>
  <system.webServer>
    <httpProtocol>
      <customHeaders>
        <add name="Content-Security-Policy-Report-Only" value="default-src 'self'; object-src 'none'; base-uri
 'self'; frame-ancestors 'self'" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
</configuration>
```

Start in Report-Only mode, add the origins your pages really use, then rename the header to Content-Security-Policy to enforce it. Applies to the site or application that owns this web.config.

Cloudflare

```
Rules > Transform Rules > Modify Response Header > Create rule
1. Apply to: All incoming requests (or this hostname)
2. Action: Set static
3. Header name: Content-Security-Policy-Report-Only
4. Value: default-src 'self'; object-src 'none'; base-uri 'self'; frame-ancestors 'self'
5. Deploy
```

Start in Report-Only mode, add the origins your pages really use, then rename the header to Content-Security-Policy to enforce it. Cloudflare sets the header at the edge for proxied (orange-cloud) traffic only.

WordPress

```
<?php
// In a small site-specific plugin (or the child theme's functions.php)
add_action( 'send_headers', function () {
    header(
        "Content-Security-Policy-Report-Only: "
        . "default-src 'self'; "
        . "object-src 'none'; "
        . "base-uri 'self'; "
        . "frame-ancestors 'self'"
    );
} );
```

Start in Report-Only mode, add the origins your pages really use, then rename the header to Content-Security-Policy to enforce it. Setting the header in the web server or CDN is preferred; use this when you only control WordPress.

AWS CloudFront

```
CloudFront > Policies > Response headers > Create response headers policy
1. Security headers or Custom headers: add Content-Security-Policy-Report-Only = default-src 'self';
object-src 'none'; base-uri 'self'; frame-ancestors 'self'
2. Origin override: enabled
3. Attach the policy to the distribution's cache behavior(s)
```

Start in Report-Only mode, add the origins your pages really use, then rename the header to Content-Security-Policy to enforce it. Changes propagate to all edge locations within a few minutes.

> Test configuration changes in a staging environment before production deployment. An incorrect policy can break scripts, sign-in, checkout, embedded services, analytics, or third-party integrations.

References	MDN: Content-Security-Policy OWASP Content Security Policy Cheat Sheet developer.mozilla.org owasp.org

F-03 Session cookie missing Secure and HttpOnly flags

Medium

Finding ID	`DEMO-2026-0001-F-03 · rule cookies`
Classification	Confirmed finding: Observed directly in the website's responses or DNS. Reduces the score.
Workflow status	Still observed
Fix priority	P2 · Remediate within 30 days
Score impact	–8 points
Confidence	High
Affected URL	`https://www.client-shop.example/`
Detection method	Passive inspection of Set-Cookie headers on public pages.
Observed at	`2026-10-01T09:14:05Z`
Evidence	Set-Cookie: PHPSESSID=[redacted]; path=/
Why it matters	Without Secure, the cookie can be sent over unencrypted HTTP; without HttpOnly, page scripts can read it.
For the client	The website's sign-in cookies are missing standard protections, which makes it easier to steal a logged-in session or misuse it from another website. The change is a configuration setting in the site or its platform.
Potential business impact	Increased risk of session theft and account takeover for signed-in customers.
Technical impact	The session identifier is readable by page scripts and may be sent over plain HTTP.
Limitation	Only cookies set on publicly accessible pages are visible to a passive scan.
Remediation	Set Secure, HttpOnly and SameSite=Lax on session cookies.
Verification criteria	A rescan observes the session cookie with the Secure and HttpOnly attributes.
Verification status	Still present on the latest scan
Human validation	Required before remediation is planned.
Standards	`CWE-614 · CWE-1004 · CWE-1275`

Configuration examples by platform

Nginx

```
# When Nginx proxies an application (nginx 1.19.3+)
proxy_cookie_flags ~ secure httponly samesite=lax;
```

Applies to cookies set by the proxied application. Fixing the flags in the application itself is preferred.

Apache HTTP Server

```
# Requires mod_headers. Prefer fixing the application; use this as a stopgap.
Header edit Set-Cookie "(?i)^((?:(?!;\s?HttpOnly).)+)$" "$1; HttpOnly"
Header edit Set-Cookie "(?i)^((?:(?!;\s?Secure).)+)$" "$1; Secure"
```

Each rule adds the flag only when it is missing. Use "Header edit" (not "always") so it applies to cookies set by the application. Test that sign-in still works.

Microsoft IIS

```
<?xml version="1.0" encoding="utf-8"?>
<!-- web.config (ASP.NET): merge into the existing <configuration> element -->
<configuration>
  <system.web>
    <httpCookies requireSSL="true" httpOnlyCookies="true" sameSite="Lax" />
  </system.web>
</configuration>
```

The sameSite attribute requires .NET Framework 4.7.2 or later.

WordPress

```
; php.ini (or .user.ini) for the PHP session cookie
session.cookie_secure = 1
session.cookie_httponly = 1
session.cookie_samesite = "Lax"
```

WordPress's own login cookies are already HttpOnly and use Secure on HTTPS sites; plugins that set their own cookies may need updating.

Test configuration changes in a staging environment before production deployment. An incorrect policy can break scripts, sign-in, checkout, embedded services, analytics, or third-party integrations.

References

MDN: Set-Cookie
OWASP Session Management Cheat Sheet
developer.mozilla.org

F-04 No clickjacking protection

Medium

Finding ID	`DEMO-2026-0001-F-04 · rule xfo`
Classification	Confirmed finding: Observed directly in the website’s responses or DNS. Reduces the score.
Workflow status	Still observed
Fix priority	P2 · Remediate within 30 days
Score impact	–8 points
Confidence	High
Affected URL	`https://www.client-shop.example/`
Detection method	Passive HTTP response header inspection.
Observed at	`2026-10-01T09:14:05Z`
Evidence	Neither X-Frame-Options nor a CSP frame-ancestors directive was returned.
Why it matters	Any other website can load these pages inside a hidden frame and trick visitors into clicking.
For the client	Another website can show this website hidden inside its own page and trick people into clicking buttons they cannot see, such as “confirm” or “delete”. One server setting stops it.
Potential business impact	Visitors could be manipulated into unintended actions such as changing account settings.
Technical impact	The pages can be embedded in a cross-origin frame.
Remediation	Send X-Frame-Options: SAMEORIGIN, or CSP frame-ancestors 'self'.
Verification criteria	A rescan observes X-Frame-Options (DENY or SAMEORIGIN) or a CSP frame-ancestors directive.
Verification status	Still present on the latest scan
Human validation	Required before remediation is planned.

Standards	CWE-1021

Configuration examples by platform

Nginx

```
# server { } block (or http { })
add_header X-Frame-Options "SAMEORIGIN" always;
```

If you already send a Content-Security-Policy, adding frame-ancestors 'self' gives the same protection in modern browsers. Reload with: nginx -t && systemctl reload nginx. add_header in a location block replaces headers set at server level.

Apache HTTP Server

```
# Requires mod_headers (a2enmod headers)
Header always set X-Frame-Options "SAMEORIGIN"
```

If you already send a Content-Security-Policy, adding frame-ancestors 'self' gives the same protection in modern browsers. Place in the VirtualHost or .htaccess, then run apachectl configtest and reload.

Microsoft IIS

```
<?xml version="1.0" encoding="utf-8"?>
<!-- web.config: merge into the existing <configuration> element -->
<configuration>
  <system.webServer>
    <httpProtocol>
      <customHeaders>
        <add name="X-Frame-Options" value="SAMEORIGIN" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
</configuration>
```

If you already send a Content-Security-Policy, adding frame-ancestors 'self' gives the same protection in modern browsers. Applies to the site or application that owns this web.config.

Cloudflare

```
Rules > Transform Rules > Modify Response Header > Create rule
1. Apply to: All incoming requests (or this hostname)
2. Action: Set static
3. Header name: X-Frame-Options
4. Value: SAMEORIGIN
5. Deploy
```

If you already send a Content-Security-Policy, adding frame-ancestors 'self' gives the same protection in modern browsers. Cloudflare sets the header at the edge for proxied (orange-cloud) traffic only.

WordPress

```
<?php
// In a small site-specific plugin (or the child theme's functions.php)
add_action( 'send_headers', function () {
    header(
        "X-Frame-Options: "
        . "SAMEORIGIN"
    );
} );
```

If you already send a Content-Security-Policy, adding frame-ancestors 'self' gives the same protection in modern browsers. Setting the header in the web server or CDN is preferred; use this when you only control WordPress.

AWS CloudFront

```
CloudFront > Policies > Response headers > Create response headers policy
1. Security headers > X-Frame-Options: SAMEORIGIN
2. Origin override: enabled
3. Attach the policy to the distribution's cache behavior(s)
```

If you already send a Content-Security-Policy, adding frame-ancestors 'self' gives the same protection in modern browsers. Changes propagate to all edge locations within a few minutes.

> Test configuration changes in a staging environment before production deployment. An incorrect policy can break scripts, sign-in, checkout, embedded services, analytics, or third-party integrations.

References	MDN: X-Frame-Options OWASP Clickjacking Defense Cheat Sheet developer.mozilla.org

F-06 Server software version disclosed

Low

Finding ID	`DEMO-2026-0001-F-06 · rule server_banner`
Classification	Confirmed finding: Observed directly in the website's responses or DNS. Reduces the score.
Workflow status	Still observed
Fix priority	P3 · Review within 90 days
Score impact	–3 points
Confidence	Medium
Affected URL	`https://www.client-shop.example/`
Detection method	Passive inspection of the Server and X-Powered-By response headers.
Observed at	`2026-10-01T09:14:05Z`
Evidence	Server: Apache/2.4.41 (Ubuntu) X-Powered-By: PHP/7.4.3
Why it matters	Exact version numbers make it easier to match the site against known vulnerabilities.
Potential business impact	Helps attackers prioritise the site when a new vulnerability is published.
Technical impact	Software names and versions are disclosed to every visitor.
Limitation	A version banner can be inaccurate: distributions often backport security fixes without changing the version string.
Remediation	Hide version details in the web server and PHP configuration, and keep both updated.
Verification criteria	A rescan observes no version numbers in the Server or X-Powered-By headers.
Verification status	Still present on the latest scan
Human validation	Required before remediation is planned.
Standards	`CWE-200`

Configuration examples by platform

Nginx

```
# http { } block
server_tokens off;
```

Hides the version number; the Server header still says "nginx".

Apache HTTP Server

```
# Main server configuration (not .htaccess)
ServerTokens Prod
ServerSignature Off
```

Reduces the Server header to "Apache". For X-Powered-By, set expose_php = Off in php.ini.

Microsoft IIS

```
<?xml version="1.0" encoding="utf-8"?>
<!-- web.config, IIS 10 (version 1709+): merge into the existing <configuration> element -->
<configuration>
  <system.webServer>
    <security>
      <requestFiltering removeServerHeader="true" />
    </security>
    <httpProtocol>
      <customHeaders>
        <remove name="X-Powered-By" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
</configuration>
```

removeServerHeader is available from IIS 10 version 1709.

WordPress

```
; php.ini (or .user.ini): removes the X-Powered-By: PHP/x.y header
expose_php = Off
```

The Server header itself is controlled by the web server (Nginx or Apache), not by WordPress.

Test configuration changes in a staging environment before production deployment. An incorrect policy can break scripts, sign-in, checkout, embedded services, analytics, or third-party integrations.

References

OWASP: Fingerprint Web Server
owasp.org

F-02 Strict-Transport-Security header missing

High

Finding ID	`DEMO-2026-0001-F-02 · rule hsts`
Classification	Confirmed finding: Observed directly in the website’s responses or DNS. Reduces the score.
Workflow status	Resolved
Fix priority	P1 · Remediate within 7 days
Score impact	–15 points
Confidence	High
Affected URL	`https://www.client-shop.example/`
Detection method	Passive HTTP response header inspection over HTTPS.
Observed at	`2026-10-01T09:14:05Z`
Evidence	HTTPS is available, but the response carries no Strict-Transport-Security header.
Why it matters	HSTS tells browsers to always use HTTPS for this site, closing the window in which a first request can be downgraded to HTTP.
For the client	Browsers are not told to always use the secure version of the website. On public Wi-Fi, a visitor could be sent to an insecure copy of a page without noticing. The fix is a one-line server setting.
Potential business impact	Visitors on untrusted networks could be redirected to an insecure version of the site.
Technical impact	The first plain-HTTP request to the site is not upgraded by the browser.
Remediation	Send Strict-Transport-Security with a max-age of at least one year once HTTPS is stable on every subdomain.

Verification criteria	A rescan observes Strict-Transport-Security with max-age of at least 31536000 on the HTTPS response.
Verification status	Verified fixed
Human validation	Required before remediation is planned.
Standards	`CWE-319 · NIST SP 800-53 SC-8`
References	MDN: Strict-Transport-Security OWASP HTTP Security Response Headers developer.mozilla.org

F-05 X-Content-Type-Options header missing

Low

Finding ID	`DEMO-2026-0001-F-05 · rule xcto`
Classification	Confirmed finding: Observed directly in the website's responses or DNS. Reduces the score.
Workflow status	Resolved
Fix priority	P3 · Review within 90 days
Score impact	–3 points
Confidence	High
Affected URL	`https://www.client-shop.example/`
Detection method	Passive HTTP response header inspection.
Observed at	`2026-10-01T09:14:05Z`
Evidence	No X-Content-Type-Options header in the response.
Why it matters	Browsers may guess the type of a file and treat uploaded text as script.
Potential business impact	Low on its own; it removes one layer of defence against content-type confusion.
Technical impact	MIME type sniffing is not disabled.
Remediation	Send X-Content-Type-Options: nosniff.
Verification criteria	A rescan observes X-Content-Type-Options: nosniff.
Verification status	Verified fixed
Human validation	Required before remediation is planned.
Standards	`CWE-16`
References	MDN: X-Content-Type-Options developer.mozilla.org

Checks passed

- Valid TLS certificate with a trusted chain
- Certificate not close to expiry (74 days remaining)
- HTTP redirects to HTTPS
- No mixed content on the scanned page
- Referrer-Policy present
- Permissions-Policy present
- No duplicate or conflicting security headers
- No commonly exposed files found (.git, .env, backups)
- security.txt published
- SPF record present
- DMARC record present

Remediation roadmap

Remediate within 30 days (3)

- F-01 Content-Security-Policy header not set
- F-03 Session cookie missing Secure and HttpOnly flags
- F-04 No clickjacking protection

Review within 90 days (1)

- F-06 Server software version disclosed

Verify

After each change, run Verify Fix. NeoShield rescans the same website and records each finding as resolved, partially resolved, still present or unable to verify. Keep the updated report as the record of the work.

These timeframes are a common starting point. Adjust them to the website's risk, your agreement with the client and any compliance obligations.

Appendix A: Standards mapping

Each finding maps to the public weakness, protocol or control references below. The references explain where a check comes from; they do not mean the website or NeoShield is certified against them.

Ref	Finding	References
F-01	Content-Security-Policy header not set	OWASP A05:2021, CWE-693, NIST SP 800-53 SC-18
F-03	Session cookie missing Secure and HttpOnly flags	CWE-614, CWE-1004, CWE-1275
F-04	No clickjacking protection	CWE-1021
F-06	Server software version disclosed	CWE-200
F-02	Strict-Transport-Security header missing	CWE-319, NIST SP 800-53 SC-8
F-05	X-Content-Type-Options header missing	CWE-16

Appendix B: Definitions

P0	Immediate action. Directly exposes the site or its data.
P1	Remediate within 7 days. Removes an important protection for every visitor, for example HTTPS enforcement.
P2	Remediate within 30 days. Weakens a defence that matters when another flaw exists, such as CSP or cookie flags.
P3	Review within 90 days. Hardening or information disclosure with limited direct impact.
P4	Optional hardening consideration. Not required; consider it when convenient.
Confirmed finding	Observed directly in the website’s responses or DNS. Reduces the score.
Contextual observation	Depends on how the website, DNS or email service is used. Review it; it does not reduce the score.
Informational observation	Recorded for awareness. No action is required.
Inconclusive check	The check could not complete. It is never counted as passed.
Resolved	A later scan of the same website no longer observed the finding.
Still observed	A later scan still observed the finding.
Coverage	Share of applicable checks that produced a result. Grades are capped when coverage is low.
Confidence	How directly the evidence supports the results: high for direct HTTP or DNS evidence, medium when interpretation is needed, low when responses were incomplete or modules could not run. It is not derived from coverage alone.

Appendix C: Scope, authorization and method

- Target: https://www.client-shop.example/ (one public website, apex and www)
- Method: non-intrusive requests to publicly accessible pages and DNS lookups
- Authorization: confirmed by the account holder before the assessment
- Checks: HTTP security headers, TLS and certificate, cookies, HTTP to HTTPS redirect, SPF, DMARC, security.txt, common exposed files

Appendix D: Limitations

- This is a passive configuration assessment, not a penetration test.
- Authenticated areas, application logic, injection flaws and business-logic issues were not tested.
- Results reflect the responses observed at the time of the assessment; a CDN or load balancer may serve different headers to other visitors.
- Configuration examples must be tested in a staging environment before production deployment.

Appendix E: Report integrity

Report ID	`DEMO-2026-0001`
Scanned at	`2026-10-01`
Generated	`2026-11-03 17:16 UTC`
Scanner version	`demo`
Scoring model	`2026.07-b`
SHA-256	`f30fc7b4279939943242f1d65e84b6cc5334ccc48bf537f686bbcff8f005213d`

The SHA-256 value is a fingerprint of the stored scan data this PDF was generated from. NeoShield can use it to confirm that a copy matches the stored report.

Prepared with NeoShield Quantum X. Findings are generated automatically from publicly observable responses at the time of the assessment and are not reviewed by NeoShield staff; the person sending this report reviews them before changes are made. This report is not a penetration test, a security certification or a guarantee.