

snyk
API & Web

ISO 27001 COMPLIANCE REPORT

Trueforce - https://airsoft.trueforce.ca

Report generated on Oct. 25, 2026 at 15:36 UTC

Summary

This section contains the scan summary

TARGET https://airsoft.trueforce.ca

Report generated on Oct. 25, 2026 at 15:36 UTC

STARTED	ENDED	DURATION	SCAN PROFILE
Oct. 25, 2026, 13:53 UTC	Oct. 25, 2026, 14:27 UTC	34 minutes	Full

NUMBER OF FINDINGS

	CURRENT SCAN	FROM LAST SCAN	PENDING FIX
CRITICAL	0	= 0	0
HIGH	0	▼ 1	0
MEDIUM	0	= 0	0
LOW	1	▼ 1	0

TOP 5

Insecure Content Security Policy	1

ISO 27001 REQUIREMENTS CHECKLIST

	TESTED	PASSED
A.5.14 Information transfer	✓	✓
A.5.33 Protection of records	✓	✓
A.5.34 Privacy and protection of personally identifiable information (PII)	✓	✓
A.8.2 Privileged access rights	✓*	—
A.8.3 Information access restriction	✓*	—
A.8.4 Access to source code	✓	✓
A.8.5 Secure authentication	✓	✓
A.8.8 Management of technical vulnerabilities	✓*	—
A.8.9 Configuration management	✓	✓
A.8.12 Data leakage prevention	✓	✓
A.8.15 Logging	✓	✓

A.8.24	Use of cryptography	✓	✓
A.8.25	Secure development life cycle	✓	✓
A.8.26	Application security requirements	✓	✓
A.8.27	Secure system architecture and engineering principles	✓	✓
A.8.28	Secure coding	✓	✓
A.8.29	Security testing in development and acceptance	✓*	—

* Partially tested

Settings

This section contains the summary of settings that were used during this scan

---

✓ **SCAN PROFILE**

Full

Tests for all supported vulnerabilities, like the normal scanning profile, but including a more extensive set of payloads. Scan time might be longer than when scanning with other profiles.

Technical Summary

The following table summarizes the findings, ordered by their severity

#	SEVERITY	VULNERABILITY	STATE
1	LOW	**Insecure Content Security Policy** https://airsoft.trueforce.ca/	**ACCEPTED**

Exhaustive Test List

The following pages contain the list of vulnerabilities we tested in this scan, taking into consideration the chosen profile

---

- Reflected cross-site scripting
- Cookie without HttpOnly flag
- Open redirection
- SQL Injection
- Missing cross-site request forgery protection
- Missing clickjacking protection
- Stored cross-site scripting
- Insecure crossdomain.xml policy
- SSL cookie without Secure flag
- HTTP TRACE method enabled
- Directory Listing
- ASP.NET tracing enabled
- Path traversal
- Remote File Inclusion
- ASP.NET ViewState without MAC
- Session Token in URL
- Application error message
- Private IP addresses disclosed
- OS command injection
- XML external entity injection
- ASP.NET debugging enabled
- Insecure Silverlight clientaccesspolicy.xml policy
- PHP code injection
- Server-side JavaScript injection
- Ruby code injection
- Python code injection
- Server-side template injection
- Unencrypted communications
- HSTS header not enforced
- Mixed content
- Cross Origin Resource Sharing: Arbitrary Origin Trusted
- Certificate with insufficient key size or usage, or insecure signature algorithm
- Expired TLS certificate
- Insecure SSL protocol version 3 supported
- Deprecated TLS protocol version 1.0 supported
- Deprecated TLS protocol version 1.1 supported
- Secure TLS protocol version 1.2 not supported
- Weak cipher suites enabled
- Server Cipher Order not configured
- Untrusted TLS certificate
- Heartbleed
- Secure Renegotiation is not supported
- TLS Downgrade attack prevention not supported
- Drupal version with known vulnerabilities
- WordPress version with known vulnerabilities
- Joomla! version with known vulnerabilities

- Certificate without revocation information
- Full path disclosure
- Log file disclosure
- Backup file disclosure
- HSTS header set in HTTP
- HSTS header with low duration and no subdomain protection
- HSTS header with low duration
- HSTS header does not protect subdomains
- Inclusion of cryptocurrency mining script
- Insecure SSL protocol version 2 supported
- Browser content sniffing allowed
- Referrer policy not defined
- Insecure referrer policy
- Potential DoS on TLS Client Renegotiation
- JQuery library with known vulnerabilities
- AngularJS library with known vulnerabilities
- Bootstrap library with known vulnerabilities
- JQuery Mobile library with known vulnerabilities
- JQuery Migrate library with known vulnerabilities
- TLS certificate about to expire
- Moment.js library with known vulnerabilities
- Prototype library with known vulnerabilities
- React library with known vulnerabilities
- SWFObject library with known vulnerabilities
- TinyMCE library with known vulnerabilities
- Backbone library with known vulnerabilities
- Mustache library with known vulnerabilities
- Handlebars library with known vulnerabilities
- Dojo library with known vulnerabilities
- jPlayer library with known vulnerabilities
- CKEditor library with known vulnerabilities
- DWR library with known vulnerabilities
- Flowplayer library with known vulnerabilities
- DOMPurify library with known vulnerabilities
- Plupload library with known vulnerabilities
- easyXDM library with known vulnerabilities
- Ember library with known vulnerabilities
- YUI library with known vulnerabilities
- Sessvars library with known vulnerabilities
- prettyPhoto library with known vulnerabilities
- jQuery UI library with known vulnerabilities
- WordPress plugin with known vulnerabilities
- Invalid referrer policy
- Insecure PHP Object deserialization
- Missing Content Security Policy header
- Insecure Content Security Policy
- GraphQL Introspection enabled
- Log4Shell
- Vue.js library with known vulnerabilities
- Spring Cloud SPEL Code Injection (CVE-2022-22963)
- Spring4Shell
- Knockout library with known vulnerabilities
- Weak JWT HMAC secret
- JWT accepting none algorithm

- JWT signature is not being verified
- Using jwk parameter to verify JWTs
- JWT algorithm confusion
- MongoDB Injection
- Next.js library with known vulnerabilities
- Underscore.js library with known vulnerabilities
- Chart.js library with known vulnerabilities
- JSZip library with known vulnerabilities
- GraphQL Misconfiguration
- Insecure browser XSS protection enabled
- Hidden file found
- Svelte library with known vulnerabilities
- Axios library with known vulnerabilities
- Cookie with SameSite attribute set to None
- Server-side request forgery
- CRLF injection
- Froala library with known vulnerabilities
- Highcharts library with known vulnerabilities
- Supply Chain Compromise
- PDF.js library with known vulnerabilities
- Lodash library with known vulnerabilities
- Select2 library with known vulnerabilities
- UAParser.js library with known vulnerabilities
- MathJax library with known vulnerabilities
- JQuery Validation library with known vulnerabilities
- Ext JS library with known vulnerabilities
- jQuery File Upload library with known vulnerabilities
- React2Shell
- markdown-it library with known vulnerabilities
- highlight.js library with known vulnerabilities
- Swiper library with known vulnerabilities
- Quill library with known vulnerabilities
- MooTools library with known vulnerabilities
- Marked library with known vulnerabilities
- protobuf.js library with known vulnerabilities
- Video.js library with known vulnerabilities
- GSAP library with known vulnerabilities
- PHP2Shell

Detailed Finding Descriptions

This section contains the findings in more detail, ordered by severity

---

1 Insecure Content Security Policy

LOW

CVSS SCORE
3.7

CVSS:3.0/AV:N/AC:H/PR:N/UI:N/S:U/C:L/I:N/A:N

METHOD

GET

PATH

https://airsoft.trueforce.ca/

DESCRIPTION

The Content Security Policy (CSP) is an HTTP header through which site owners define a set of security rules that the browser mus t follow when rendering their site. The most common usage is to define a list of approved sources of content that the browser can load. This can be used to effectively mitigate Cross-Site Scripting (XSS) and Clickjacking attacks.

CSP is flexible enough for you to define from where the browser can load JavaScript, Stylesheets, images, or fonts, among other op tions. It can also be used in report mode only, a recommended approach before deploying strict rules in a live environment. Howe ver, please note that report mode does not protect you, it just logs policy violations.

The following issues need to be addressed:

- unsafe 'unsafe-inline' in the style-src directive

EVIDENCE

Response headers, with an insecure Content-Security-Policy header:

```
content-security-policy: default-src 'self'; script-src 'self'  *.cofys.de *.ucdn.de; style-src 'self' 'unsafe-
inline' *.cofys.de; object-src 'none'; base-uri 'self'; connect-src 'self' *.mapbox.com *.cofys.de *.ucdn.de; f
ont-src 'self' data: *.cofys.de *.ucdn.de; frame-src 'self' https://www.youtube-nocookie.com https://www.youtub
e.com; img-src * blob: data:; manifest-src 'self'; media-src 'self'; worker-src 'self' blob:; form-action 'sel
f'  https://checkout.stripe.com https://www.paypal.com https://www.sandbox.paypal.com; frame-ancestors 'self'; u
pgrade-insecure-requests;
```

REQUEST

```
GET / HTTP/2
accept: */*
accept-encoding: gzip, deflate, br
connection: keep-alive
user-agent: Mozilla/5.0 (compatible; +https://probely.com/sos)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/144.0.0.0 Safari/537.36
ProbelySPDR/0.2.0
host: airsoft.trueforce.ca
```

RESPONSE

```
HTTP/2 200 OK
alt-svc: h3=":443"; ma=2592000
cache-control: max-age=0, must-revalidate, private
content-encoding: gzip
content-security-policy: default-src 'self'; script-src 'self'  *.cofys.de
*.ucdn.de; style-src 'self' 'unsafe-inline' *.cofys.de; object-src 'none'; base-
```

```
uri 'self'; connect-src 'self' *.mapbox.com *.cofys.de *.ucdn.de; font-src
'self' data: *.cofys.de *.ucdn.de; frame-src 'self' https://www.youtube-
nocookie.com https://www.youtube.com; img-src * blob: data:; manifest-src
'self'; media-src 'self'; worker-src 'self' blob:; form-action 'self'
https://checkout.stripe.com https://www.paypal.com htt...
content-type: text/html; charset=UTF-8
date: Sun, 25 Oct 2026 13:55:04 GMT
expires: Sun, 25 Oct 2026 13:55:04 GMT
permissions-policy: autoplay=(self), camera=(self), encrypted-media=(self),
fullscreen=(self), geolocation=(self), microphone=(self), midi=(self),
payment=(self)
referrer-policy: strict-origin-when-cross-origin
server: FrankenPHP Caddy
set-cookie: SESSION_v18=d9537212ca94f66b66b33acd85fa777b; expires=Sun, 25 Oct
2026 14:55:04 GMT; Max-Age=3600; path=/; secure; httponly; samesite=lax
strict-transport-security: max-age=63072000; includeSubDomains; preload
vary: Accept-Encoding
x-content-type-options: nosniff
x-frame-options: SAMEORIGIN
x-powered-by: PHP/8.5.12
x-xss-protection: 1; mode=block
<!DOCTYPE html>
<html lang="de" dir="ltr" data-nav-layout="horizontal" data-theme-mode="light"
data-header-styles="light"
       data-menu-styles="light" data-toggled="close" loader="enable" data-nav-
style="menu-hover"
       data-header-position="scrollable" data-menu-position="fixed">
<head>
     <!-- Meta data -->
     <meta charset="UTF-8">
     <meta name='viewport' content='width=device-width, initial-scale=1.0, user-
scalable=0'>
     <meta name="theme-color" content="white"/>
     <meta name="mobile-web-app-capable" content="yes">
     <meta name="apple-mobile-web-app-status-bar-style" content="black">
```

Glossary

Term	Definition
Vulnerability	A type of security weakness that might occur in applications (e.g. Broken Authentication, Information Disclosure). Some vulnerabilities take their name not from the weakness itself, but from the attack that exploits it (e.g. SQL Injection, XSS, etc.).
Findings	An instance of a Vulnerability that was found in an application.

Severity Legend

To each finding is attributed a severity which sums up its overall risk

The severity is a compound metric that encompasses the likelihood of the finding being found and exploited by an attacker, the skill required to exploit it, and the impact of such exploitation. A finding that is easy to find, easy to exploit and the exploitation has high impact, will have a greater severity.

Different findings of the same type could have a different severity: we consider multiple factors to increase or decrease it, such as if the application has an authenticated area or not.

The following table describes the different severities:

Severity	Description	Examples
CRITICAL	These findings require immediate attention and remediation due to their potentially devastating impact.	OS command injection SQL Injection
HIGH	These findings may have a direct impact on the application security, either clients or service owners, for instance, by granting the attacker access to sensitive information.	Reflected cross-site scripting Path Traversal
MEDIUM	Medium findings don't usually have an immediate impact alone, but combined with other findings, may lead to a successful compromise of the application.	Cross-site Request Forgery Unencrypted Communications
LOW	Findings where either the exploit is not trivial, the impact is low, or the finding cannot be exploited by itself.	Directory Listing Clickjacking

Category Descriptions

The following pages contain descriptions of each vulnerability. For each vulnerability you will find a section explaining its impact, causes and prevention methods. These descriptions are very generic, and whenever they are not enough to understand or fix a given finding, more information is provided for that finding in the Detailed Finding Descriptions section.

Insecure Content Security Policy

Description

The Content Security Policy (CSP) is an HTTP header through which site owners define a set of security rules that the browser must follow when rendering their site. The most common usage is to define a list of approved sources of content that the browser can load. This can be used to effectively mitigate Cross-Site Scripting (XSS) and Clickjacking attacks.

CSP is flexible enough for you to define from where the browser can load JavaScript, Stylesheets, images, or fonts, among other options. It can also be used in report mode only, a recommended approach before deploying strict rules in a live environment. However, please note that report mode does not protect you, it just logs policy violations.

Fix

You can define a Content Security Policy by setting a header in your application. The header can look like this:

```
Content-Security-Policy: frame-ancestors 'none'; default-src 'self', script-src '*://*.example.com:*'
```

In this example, the **frame-ancestors** directive set to **'none'** indicates that the page cannot be placed inside a frame, not even by itself. The **default-src** defines the loading policy for all resources, in this case, they can be loaded from the current origin (protocol + domain + port). The example sets a more specific policy for scripts, through the **script-src**, restricting script loading to any subdomain of example.com.

The policy can be with different directives, and there are other less strict options for the directives above.