

Zero-Trust Web Security: OAuth 2.1, OIDC, JWT & Nonce CSP Hardening

Authorization Code + PKCE flow, JWT signature verification, Strict-Dynamic CSP nonces, and CSRF/XSS mitigation.

EXECUTIVE SUMMARY

Enterprise application security reference covering OAuth 2.1 PKCE flows, Firebase/OIDC session cookies, cryptographic per-request Content Security Policy (CSP) nonces, and SSRF defenses.

KEY EQUATIONS & INVARIANTS

* OAuth 2.1 PKCE S256 Code Challenge:

LaTeX:
$$\text{code_challenge} = \text{BASE64URL}(\text{SHA256}(\text{code_verifier}))$$

Note: Binds the authorization code to the initiating client to prevent authorization code interception.

1. Token Storage & Session Architecture Comparison

Prefer `HttpOnly; Secure; SameSite=Lax` session cookies paired with short-lived in-memory bearer tokens to defend against both XSS token exfiltration and cross-site request forgery.

Storage Mechanism | XSS Token Theft Risk | CSRF Risk | Recommended Mitigation

localStorage / sessionStorage | High (readable by any JS) | None | Never store long-lived refresh tokens here

HttpOnly + Secure Cookie | Low (invisible to document.cookie) | Medium | Pair with SameSite=Lax/Strict + Origin check

In-Memory JS Closure + HttpOnly Refresh | Minimal | Low | Gold-standard SPA + SSR hybrid pattern

COMMON PRODUCTION & INTERVIEW PITFALLS

- Including `unsafe-inline` alongside a cryptographic `nonce-...` in `script-src` without understanding browser fallback rules, or reusing static nonces across requests (nonces MUST be generated per HTTP request).
- Accepting JWTs without explicitly pinning allowed asymmetric algorithms (`RS256` / `ES256`), enabling `alg: none` or HMAC key-confusion attacks.

FREQUENTLY ASKED QUESTIONS

Q: What is the difference between OAuth 2.0 and OpenID Connect (OIDC)?

A: OAuth 2.0 is a delegated authorization framework (granting an Access Token for APIs), whereas OIDC is an identity layer built on top of OAuth 2.0 that issues a signed JWT ID Token proving who the user is.