RFC 10070 client-authentication audience rule: a five-implementation conformance snapshot

Nobuo Otoi

ba-cybersecurity.com, Tokyo, Japan

independent cybersecurity researcher

Draft note, 2026-11-09

Abstract

RFC 10070, an IETF standard published 2026-11, tightened the audience (`aud`) rule for the JWT used in OAuth client authentication: the `aud` MUST be the authorization server's issuer identifier as the sole value, and the token-endpoint URL MUST NOT be used. A read-only source review of five major open OAuth authorization-server implementations finds that, against their default-branch HEADs, *all five still accept the token-endpoint URL as the client-authentication* `aud` — the value RFC 10070 now forbids — and four also accept a multi-valued `aud`. The per-implementation severity is Low (conformance, not a new flaw): the underlying issue is a previously disclosed, spec-level vulnerability (OpenID Foundation, 2025-01; ePrint 2025/629), already fixed by RFC 10070, for which server-side acceptance of the endpoint URL is only a necessary condition. This is a timed measurement of ecosystem adoption of a published fix, not a scorecard.

1 Summary

RFC 10070 ("Updates to OAuth 2.0 JWT Client Authentication and Assertion-Based Authorization Grants"; updates RFC 7521/7522/7523/9126) tightened the audience rule for the JWT used in **client authentication** (`private_key_jwt` / `client_secret_jwt`, and the PAR endpoint): the `aud` MUST be the authorization server's **issuer identifier as the sole value**, and the token-endpoint URL **MUST NOT** be used. The **authorization-grant** path (JWT bearer, RFC 7523 §3 item 3 case A) stays lenient and is out of scope.

Against their default-branch HEADs as of 2026-11-09, **all five implementations still accept the token-endpoint URL as the client-authentication `aud`**. Four also accept a multi-valued `aud`. Two are partially aligned: Keycloak rejects a multi-valued `aud` by default, and node-oidc-provider is strict under its FAPI 2.0 profile.

2 Why this is Low, not a vulnerability

The per-implementation severity is **Low**, and this is not a report of a novel flaw in any one project. But the underlying issue is real. The audience restriction fixes a **previously disclosed, spec-level vulnerability**: a malicious authorization server can advertise a legitimate server's token-endpoint URL in its own metadata — a client can authenticate a server's *issuer* identifier but not its endpoints (RFC 8414), as RFC 10070 §1 notes — so a client that reuses one key pair across servers and sets `aud` to the endpoint URL hands the malicious server a client assertion it can replay to the legitimate one. This was disclosed across standards bodies (OpenID Foundation, 2025-01-24) and analyzed in the literature (ePrint 2025/629); RFC 10070 §1 calls it a "security vulnerability," and RFC 10070 is the fix.

A server accepting the token-endpoint URL as `aud` is a **necessary condition** for that attack; the residual real-world risk depends on whether such key-reusing, endpoint-URL-audience clients exist against a given server. So this note measures **non-adoption of a published fix**, not a latent bug in any single project — hence Low, conformance-framed, and suitable (if raised at all) as a conformance heads-up, not a vulnerability report.

3 The rule is not new; adoption is phased

The restriction has been public for roughly two years: disclosed 2025-01 (OIDF), analyzed in 2025 (ePrint 2025/629), and carried through draft-ietf-oauth-rfc7523bis, FAPI 2.0, and OIDC Core errata before RFC 10070 finalized it in 2026-11. Implementers have had notice and move on their own schedules: Keycloak's 26.2 upgrade notes already recommend the issuer identifier as `aud`, and Spring Security #18226 records the maintainer deferring the change until the draft was approved. RFC 10070 itself advises phased adoption — §6 recommends updating clients before or alongside server enforcement, and §4 cautions that rejecting JWTs without an explicit `typ` is NOT RECOMMENDED. A server still accepting the old `aud` may be mid-transition by design. The value of this snapshot is as a timed measurement of that transition.

Table 1: Client-authentication `aud` handling at default-branch HEAD, 2026-11-09. "Divergent" = accepts the token-endpoint URL that RFC 10070 now forbids; "mixed" = partially aligned.

Implementation	Accepted `aud` (client auth)	Multi-valued `aud`	Conformant by config alone?	Class
ory/fosite (in Hydra)	configured token URL(s) only; issuer identifier *not* accepted	accepted	partial (no clean path)	divergent
node-oidc-provider	issuer / token / called-endpoint URL (default); issuer-only under FAPI 2.0	accepted (default); rejected under FAPI 2.0	yes (hook; strict under FAPI 2.0)	mixed
Keycloak	issuer / token / introspect / PAR / CIBA URLs	rejected by default	no (accept-set hardcoded)	mixed
Spring Authz Server	issuer / token / introspect / revoke / PAR URLs	accepted	yes (`setJwtValidatorFactory`)	divergent
Authlib	developer-configured; sole-value not enforced	accepted	partial (needs `verify_claims` override)	divergent

4 The rule (verified against RFC 10070)

- Client authentication — RFC 7523 §3 item 3 case B, as replaced by RFC 10070 §4: the `aud` "MUST indicate only the issuer identifier of the intended authorization server"; the server "MUST reject any JWT that does not contain its issuer identifier as its sole audience value"; the token-endpoint URL "MUST NOT be used as an audience value."
- PAR — RFC 9126 §2, as replaced by RFC 10070 §5: the issuer identifier "MUST be used as the sole value of the audience"; reject otherwise.
- Authorization grant (out of scope) — RFC 7523 §3 item 3 case A: an authorization server "MAY be identified by either its issuer identifier or its token endpoint URL."
- RFC 10070 does not forbid a single-element array; "sole audience value" is the operative phrase. No backward-compatibility text permits servers to keep accepting the old values.

5 Method

Read-only source review; no build, no network, nothing sent. For each project the **client-authentication** `aud` validator was located and its accept-set read at the pinned commit, deliberately distinguished from the authorization-grant validator (fosite `handler/rfc7523/handler.go`, Keycloak `JWTAuthorizationGrantType.java`, Authlib `jwt_bearer.py`; node-oidc-provider and Spring have no JWT-bearer grant). The multi-valued-`aud` behavior was confirmed in code for each. The RFC 10070 normative text, the OIDF disclosure, and the ePrint abstract were read against primary sources. Deciding code at the pinned commits: fosite `client_authentication.go` + `fositex/config.go:230--236` (Hydra in-tree); node-oidc-provider `jwt_client_auth.js:48--58`, `oidc_context.js:133--135`, `defaults.js:632--638`; Keycloak `AbstractBaseJWTValidator.java:126--137`, `JWTClientValidator.java:29--39`; Spring `JwtClientAssertionDecoderFactory.java:174,194--212`; Authlib `rfc7523/client.py:64--94`. Pins: fosite `a5f0b09` / Hydra `4174065` / nop `3ed87be` / Keycloak `d4d5554` / Spring `4283973` / Authlib `0f9826c`.

AI-assistance disclosure

Most of the source review and drafting was performed with AI assistance. Before finalizing, the author reopened each cited file:line at its pinned commit and confirmed the classification, and verified the RFC 10070 rules, the client-auth-vs-grant split, and the disclosure history against the primary documents.