OAuth 2.0 Security Best Current Practice

Torsten Lodderstedt, John Bradley, Andrey Labunets, Daniel Fett
Refines and enhances security guidance for OAuth 2.0 implementers

Updates, but does not replace:
- OAuth 2.0 Threat Model and Security Considerations (RFC 6819)
- OAuth 2.0 Security Considerations (RFC 6749 & 6750)

Updated, more comprehensive Threat Model
Description of Attacks and Mitigations
Simple and actionable recommendations
Working Group Last Call

- On Version -13
- Several comments, from editorial to content
Changes After WGLC (-13)
Editorial Changes

- Attacker Model moved to Section 2, Recommendations is Section 1
- Various small improvements, e.g.,
  - clarifications,
  - definitions (CSRF and open redirector),
  - expanded some attack details and examples,
  - restructured discussions on mitigations and removed some less-relevant discussions.
- And, yes, we fixed the reference to RFC 84184
Normative Changes

- Open redirectors:
  - Clients “SHOULD NOT avoid forwarding…”, … otherwise “are advised to implement countermeasures against open redirection”
  - “Clients MUST NOT expose [open redirectors]”
Normative Changes

- PKCE:
  - AS “SHOULD provide a way to detect their support for PKCE” (implementation-specific or metadata)
  - → “MUST”
Implicit Grant

Before:

Clients SHOULD NOT use Implicit unless ATs are sender-constrained and AT injection is prevented.

Clients SHOULD use code, which also allows for sender-constraining.

Clients SHOULD use sender-constraining.

After:

Clients SHOULD NOT use Implicit unless AT injection is prevented and token leakage vectors are mitigated.

Clients SHOULD use code.

Clients SHOULD use sender-constraining.
In order to avoid these issues, clients SHOULD NOT use the implicit grant (response type "token") or any other response type issuing access tokens in the authorization response, such as "token id token" and "code token id token", unless the issued access tokens are sender-constrained and access token injection in the authorization response is prevented.

A sender-constrained access token scopes the applicability of an access token to a certain sender. This sender is obliged to demonstrate knowledge of a certain secret as prerequisite for the acceptance of that token at the recipient (e.g., a resource server).

Clients SHOULD instead use the response type "code" (aka authorization code grant type) as specified in Section 2.1.1 or any other response type that causes the authorization server to issue access tokens in the token response, such as the "code id token" response type. This allows the authorization server to detect replay attempts by attackers and generally reduces the attack surface since access tokens are not exposed in URLs. It also allows the authorization server to sender-constrain the issued tokens (see next section).

### 2.2. Token Replay Prevention

A sender-constrained access token scopes the applicability of an access token to a certain sender. This sender is obliged to demonstrate knowledge of a certain secret as prerequisite for the acceptance of that token at the recipient (e.g., a resource server).

Authorization and resource servers SHOULD use mechanisms for sender-constrained access tokens to prevent token replay as described in Section 4.8.1.1.2. The use of Mutual TLS for OAuth 2.0 [RFC8785] is RECOMMENDED. Refresh tokens MUST be sender-constrained or use refresh token rotation as described in Section 4.12.

Authorization servers SHOULD use TLS-based methods for sender-constrained access tokens as described in Section 4.8.1.2, such as token binding [I-D.ietf-oauth-token-binding] or Mutual TLS for OAuth 2.0 [I-D.ietf-oauth-mtls] in order to prevent token replay. Refresh tokens MUST be sender-constrained or use refresh token rotation as described in Section 4.12.
Ready for Publication?
Q & A