OAuth 2.0 Token Binding

https://tools.ietf.org/html/draft-ietf-oauth-token-binding-04

Brian Campbell
Michael B. Jones
John Bradley

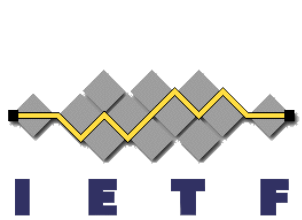
IETF 99
Prague
July 2017

The Setting of the Context

Provide an OAuth 2.0 proof-of-possession mechanism based on Token Binding to defeat (re)play of lost or stolen tokens (access, refresh, and authorization codes)

from IETF 99

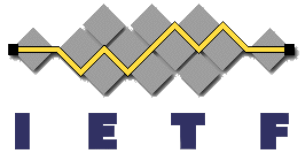
Current Status



- Token Binding WG documents; -tokbind-negotiation, -tokbind-protocol, and -tokbind-https are all very close to being Submitted to the IESG for Publication
  - Waiting for WG Chair Go-Ahead and/or Shepherd Writeup
- Published -04 of draft-ietf-oauth-token-binding on July 3rd
  - Minor editorial fixes
  - Defined how to convey token binding information of an access token via RFC 7662 OAuth 2.0 Token Introspection
    - Introspection Response Registration request for cnf is now in draft-ietf-oauth-mtls, which will likely be published and registered before draft-ietf-oauth-token-binding
  - Added an open issue about needing to allow for web server clients to opt-out of having refresh tokens bound while still allowing for binding of access tokens

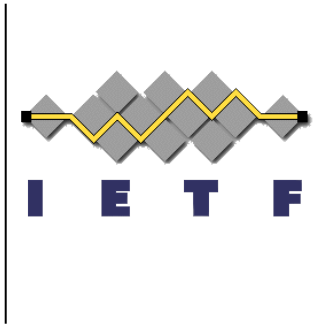
Open Issues I

(defying the conventional wisdom about lots of text on a slide)



- What should we do in the case that a refresh request for a token bound access token is received when the refresh token used in the request is not token bound?
- Currently the only way to request a token bound access token is via the referred token binding. By definition the referred token binding also comes with the provided token binding and the provided token binding is what is used to bind the refresh token. However, web server clients will typically be distributed/ clustered and very likely will not want to, or be capable of, dealing with token bound refresh tokens. Such clients will have credentials established with the AS for authenticating to the token endpoint and refresh tokens are already bound to the client. So token binding the refresh tokens doesn't add much, if anything, in this case. But accessing private token binding keys in a distributed system will be cumbersome or even impossible. Tracking and properly utilizing the association of a token binding key with each individual refresh token would also be exceptionally cumbersome (whereas client credentials are for the client and decoupled from individual refresh tokens) but without some such mechanism the token binding key cannot be changed without implicitly invalidating all the bound refresh tokens the web server client has stored for that AS. It seems necessary to provide some mechanism for a client to opt-out of having refresh tokens token bound while still allowing for token binding of access tokens.
  - Potential solutions:
    - Toggle behavior based on client metadata
    - Allow for a parameter to express the Token Binding ID to the token endpoint? (maybe useful for other reasons)

Open Issues II



- Should the scope of this document include standardization or guidance on token binding of JWT Client Authentication and/or Authorization Grants from RFC 7523?
- The Metadata and what can and cannot be reliably inferred from it need additional evaluation and work. OAuth 2.0 Protected Resource Metadata is no longer a going concern, but is currently referenced herein. Boolean values do not adequately convey Token Binding support, as different components may support different key parameters types. And successful negotiation likely doesn't provide the application layer info about all the supported key parameters types but rather just the one that was negotiated.

Looking Ahead

- Token Binding documents progress to RFC
- Work through open issues
- Implementation experience and feedback
- Get the band back together again for IETF 100 in Singapore

from CIS Chicago

from IETF 93