# Oauth 2.0 Token Binding

draft-ietf-oauth-token-binding

Brian Campbell
Michael B. Jones
John Bradley
William Denniss

IETF 103
Bangkok
Nov. 2018

# Token Binding Overview

- Enables a long-lived binding of cookies or **other security tokens** to a client generated public-private key pair
- Use is negotiated in TLS handshake via TLS extension
- Possession of key is proven by signing the TLS exported keying material (EKM) and sending as an HTTP header in every request
- Cookies and tokens can be bound to the key
- Key is scoped to the effective top-level domain + 1
- Federated/cross-domain use-cases supported via referred token binding (vs. provided)
  - "Include-Referred-Token-Binding-ID" response header on HTTP 3XX redirection tells the browser that it should reveal the Token Binding ID (the key) used between itself and the token consumer (referred) in addition to the normal one used between itself and the token provider (provided)
  - Implementation Considerations: "Token Binding implementations should provide APIs ... to generate Token Binding messages containing Token Binding IDs of various application-specified Token Binding types, to be conveyed by the Sec-Token-Binding header field"

# Overview: OAuth 2.0 Token Binding

- Provide an OAuth 2.0 proof-of-possession mechanism based on Token Binding to defeat (re)play of lost or stolen tokens
  - Bind access tokens with referred Token Binding ID
    - For access tokens issued from the token endpoint
    - For access tokens issued from the authorization endpoint via the so called implicit flow
    - Representation in JWT access tokens and introspection responses
      - “cnf” confirmation claim with a “tbh” member that’s the hash of the token binding ID
  - Bind refresh tokens with provided Token Binding ID
  - Bind authorization codes via PKCE
    - Native app clients
    - Web server clients
  - Binding for JWT Authorization Grants and JWT Client Authentication

# Happenings since(ish) Montreal

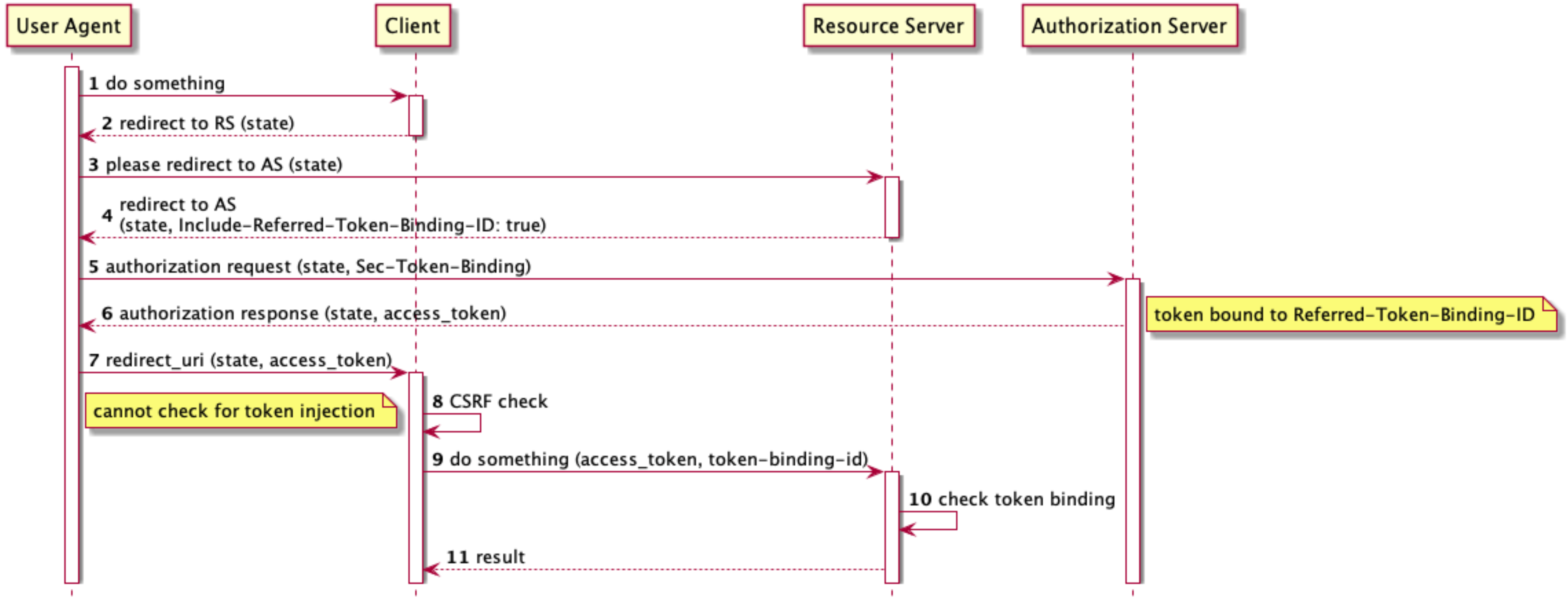
- Question on the list: “it is not very clear how an Access Token issued from the Authorization Endpoint is Token Bound” *
- OAuth 2.0 Authorization Server Metadata published as RFC 8414 and parameters registry established
- A widely used browser decides to drop its support of Token Binding
- Token Binding Protocol, Negotiation, and HTTPS published as RFCs 8471, 8472, and 8473 respectively
- Published draft -08 with updated references to the RFC versions of the core token binding specs and AS Metadata
- Question off the list: “don’t understand how the token binding ID of the RS is determined in this [the implicit] flow”

* https://mailarchive.ietf.org/arch/msg/oauth/BFtR_Sd0EpLUVEYU27vHGgbCRHo

## How does token binding of access tokens with implicit work anyway?

- It’s a little awkward…
- For an OAuth client that is running as script in the user-agent
- AS binds the access token to the referred token binding sent to the authorization endpoint, which is the token binding between the user-agent (where the client is) and the protected resource or API
- “Include-Referred-Token-Binding-ID” response header on HTTP 3XX is the only way to have the browser send the referred token binding
- Implies that there must be an HTTPS request from the browser to the protected resource which then gets full page redirected to the authorization endpoint with a proper authorization request (state, client_id, response_type, redirect_uri, etc.)
- Also token+id_token response types don’t really work
  - With form_post response mode the access token will have the wrong binding
  - With fragment response mode the ID Token binding doesn’t make sense unless it’s passed back to the client’s backend and the bound access token is used to call protected resources in the same ETLD+1 as the client backend

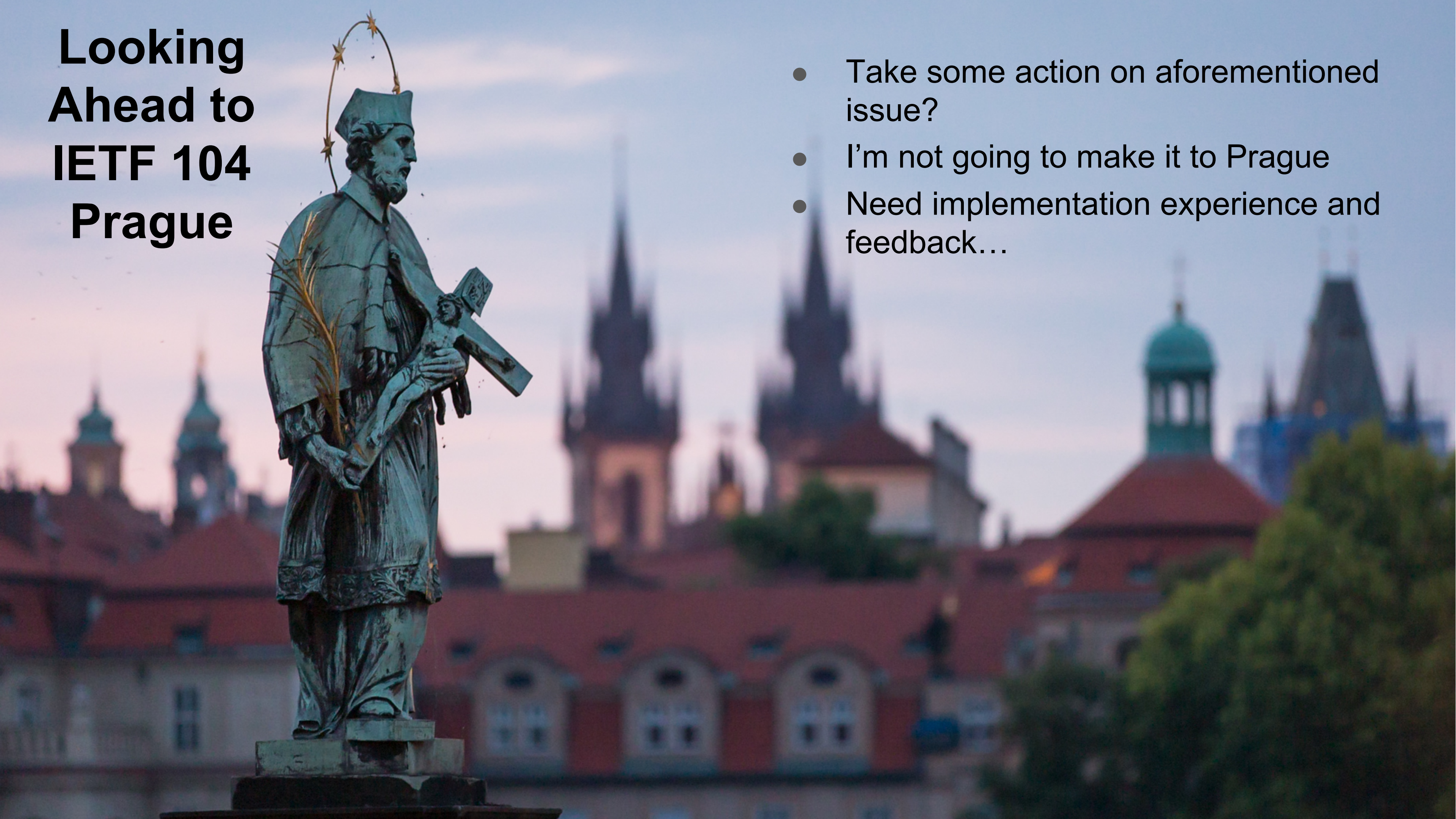
# "… tried to capture the way it is supposed to work in this sequence chart …"



User Agent
Client
Resource Server
Authorization Server
1 do something
2 redirect to RS (state)
3 please redirect to AS (state)
4 redirect to AS (state, Include-Referred-Token-Binding-ID: true)
5 authorization request (state, Sec-Token-Binding)
6 authorization response (state, access_token)
token bound to Referred-Token-Binding-ID
7 redirect_uri (state, access_token)
cannot check for token injection
8 CSRF check
9 do something (access_token, token-binding-id)
10 check token binding
11 result

## So, what to do about token binding & implicit?

1) Leave it be and let folks figure it out (or not)
2) Attempt to add some explanatory/cautionary text and let folks figure it out (or not)
3) Attempt to standardize a flow that goes from an XHR/fetch protected resource request to a full page browser HTTPS request to somewhere at the protected resource which then is redirected to the authorization endpoint with a proper authorization request including state, client_id, response_type, redirect_uri, etc.
4) Remove support/description of binding of access tokens issued from the authorization endpoint
    1) Noting, however, that JavaScript clients can't really send the referred token binding to the token endpoint
5) Other ideas here



# Looking Ahead to IETF 104 Prague

- Take some action on aforementioned issue?
- I’m not going to make it to Prague
- Need implementation experience and feedback…