OAuth 2.0 Token Binding

Brian Campbell
Michael B. Jones
John Bradley

IETF 98
Chicago
March 2017

Why Again?

- Specify a proof-of-possession mechanism based on Token Binding for OAuth 2.0 (& OpenID Connect) to defeat replay of lost or stolen tokens
Quick Token Binding
Review/Overview

- Uses a public-private key pair generated by the client to sign TLS exported keying material and create long-lived TLS binding
  - Application tokens then can be bound to those keys
- 3 documents making their way though WGLC
  - draft-ietf-tokbind-negotiation-07 (TBNEGO)
    - TLS extension for token binding protocol negotiation
  - draft-ietf-tokbind-protocol-13 (TBPROTO)
    - Token Binding protocol message format
      - provided & referred types
  - draft-ietf-tokbind-https-08 (HTTPSTB)
    - Embedding token binding messages in HTTPS
      - Sec-Token-Binding request header
      - Include-Referred-Token-Binding-ID response header
Significant Changes in -02/-03

- Many examples added (& fixed)
- Binding for Authorization Codes
  - Basically what was proposed in Seoul
PKCE based Authorization

Code Token Binding

- Bind to the Token Binding ID the native client uses to resolve the code at the token endpoint
  - code_challenge=BASE64URL(SHA256(Provided Token Binding ID between client and AS token endpoint))
  - code_challenge_method=TB-S256
  - code_verifier=provided_tb (and use the value of the provided Token Binding ID)

- Bind to the Token Binding ID the browser uses to deliver the code to a web server client
  - code_challenge=referred_tb (use the value of the referred Token Binding ID)
  - code_challenge_method=referred_tb
  - code_verifier=BASE64URL(Provided Token Binding ID between browser and Client’s redirect URI)
POST /as/token.oauth2 HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
Sec-Token-Binding: AIkAAgBBQGto7hHRR0Y5nkOWqc9KNfwW95dEFmSI_tCZ_Cbl7LWlt6Xjp3DbjiDjavGFiKP2HV_2JSE42VzmKOVVV8m7eqAAQOKiDK1Oi0z6v4X5BP7uc0pFestVZ42TTODJmoHpji06Qq3jsCiCRSJx9ck2fWJYx8tLVXRZPATB3x6c24aY0ZEAAAA

grant_type=authorization_code&code=4bwcZesc7Xacc330ltc66Wxk8EAfP9j2&code_verifier=2x6_y1S390-8V7jaT9wj.8qP9nKmYCf.V-rD9O4r_1&client_id=example-native-client-id

HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store

{  "access_token":"EdRs7qMrLb167Z9fV2dcwoLTC",
"refresh_token":"ACClZEIQTjW9arT9GOJGGd7QNWqOMmUYfsJTiv8his4",
"token_type":"Bearer",
"expires_in":3600
}
POST /as/token.oauth2 HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
Sec-Token-Binding: AIkAAgBBQGto7hHRR0Y5nkOWqc9KNfwW95dEFmSI_tCZ_Cbl
7LWlt6Xjp3DbjiDJavGFiKP2HV_2JSE42VzmKOVVV8m7eqAAQCpGbaG_YRf27q0ra
L0UT4fsKKjL6PukuOT00qzamoAXxOq7m_id7O3mLpnb_sM7kwSxLi7iNHzzDgCAkP
pt31HwAAA

refresh_token=ACClZEIQTjW9arT9GOJGGd7QNwgOMmUYfsJTiv8his4
&grant_type=refresh_token&client_id=example-native-client-id

HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store

{
  "access_token":"bwcESCwC4yOCQ8iPsgcn117k7",
  "token_type":"Bearer",
  "expires_in":3600
}
Example: Access Token Issued from the Authorization Endpoint

GET /as/authorization.oauth2?response_type=token  
&client_id=example-client-id&state=rM8pZxG1c3gKy6rEbsD8s  
&redirect_uri=https%3A%2F%2Fclient%2Example%2Em%2F.org%2Fcb HTTP/1.1
Host: server.example.com
Sec-Token-Binding: ARIAAgBBQIEE8mSMtDy2dj9EEBdXaQT9W3Rq1NS-jW8ebPoF  
6FyL0jIfATVE5S2lirccgOTZmEqglxeIrC3DsGegwjs4hw14AQPDK1AXFFMyQkZegeC  
wlbTlqX3F9HTt-1JxFU_pi16ezka7qVRcpSF0BQLfSqlsxMbYfSSCJX1BDtrIL7PX  
j__fuAAAECAYCAElBNUnP3te5WrwlEwiejEz0OesmC5PE1Wc7kZ5n1LSqQTj1ciIp  
5vQ30LLUCyM_a2BYTUPKtd5EdS-Pa1T4t6ABADgeizRa5NkTMuX4zOdC-R4cLNVV  
O81Lu2Psco-UJLR_XAH4Q0H7-m0_nQR1zBN78nYMKpvHsz8L3zWKRVvLEYgAA

HTTP/1.1 302 Found
Location: https://client.example.org/cb#state=rM8pZxG1c3gKy6rEbsD8s  
&expires_in=3600&token_type=Bearer  
&access_token=eyJhbGciOiJFUVzI[...omitted for brevity...]8xy5W5sQ

{
  ...other claims omitted for brevity...
  "cnf": {
    "tbh": "vowQESa_MgbGJwxIAm_BTN2QDPwh8PhuBm-EtUAgxc"
  }
}
Example: Access Token Issued from the Token Endpoint

POST /as/token.oauth2 HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
Sec-Token-Binding: ARIAAGBBQFXJir2w4gbJ7grBx9uTYWIr5s9V50-PW4ZijegQ
   0LUM-_bGnGT6DizxUK-m5n3dQUIkeH7ybn6wb1C5dGyV_IAAQDDFToFrHt41Zppq7
   u_SEMF_E-KimAB-HewW12MvZzAQ9QkoWiJCLFiCkjjtr1RrA2-jaJvoB8051DTGXQ
   ydWykAAAECAEFAuC1GlyU83rqTGHEau0qvNwy0fDsdXzIyT_4x1FcdsMxjFkJac
   IBJFGuYcccvnCak_duF13QKFENuwxql-H9ABAMcU7IjJOUA4IyE6YoEcfz9BMPQqw
   M5M6hw4RZNQd58fsTCCslQE_NmNCl9JXy4NkdEZXqvZGP0y8QZ_bmAwAA

refresh_token=gZr_Z18EAhLgWR-qWxBimbgbZRzi_8EAhLgWRgWxBimbfb
   &grant_type=refresh_token&client_id=example-client-id

HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-cache, no-store

{
   "access_token":"eyJhbGciOiJFUzI1NiIsImtp[...omitted...]lcs29j5c3",
   "token_type":"Bearer",
   "expires_in":3600
}
Example: Protected Resource Request (using Access Token Issued from the Authorization Endpoint)

```json
{
  "cnf": {
    "tbh": "7NRBu9iDdJ1YCTQyeYuLxV0b1EA-yTpmGIrAwKAws"
  }
}
```

GET `/api/stuff` HTTP/1.1
Host: resource.example.org
Authorization: Bearer eyJhbGciOiJFUzI1NiIsI[...omitted...]
Sec-Token-Binding: AIkAAgBBQLgtRpWFPN66kxhxGrtakrzcmHtw7HV8yMk__MdR_XJXbDMYxZCwnCASRRrmHHHL5wmpP3bhYt0ChRDb5Mapfh_QAQN1He3Ftj4Wa_S_fz_ZVns4saLfj6aBoMSQW6rLs19I1vHze7LrGjKyCfPTKXjajebxp-TLPFZCc0JTqTY5 _0MBAAAA
Example Bound Code: Native App Client

GET /as/authorization.oauth2?response_type=code
   &client_id=example-native-client-id&state=oUC2jyYtzRCrMyWrVnGj
   &code_challenge=rBlgOyMY4teiuJMDg0wkrpsAjPyI07D2WsEM-dnq6eE
   &code_challenge_method=TB-S256 HTTP/1.1
Host: server.example.com

POST /as/token.oauth2 HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
Sec-Token-Binding: AIkAAgBBQEOO9GRFP-LM0hoWw6-2i318BsuuUum5AL8bt1sz1r1EFfp5DMXMNW3O8WjcIXr2DKJnI4xnuGse6GywQd9RbD0AQJDb3xyo9PBxj8M6YjLT-60axgDkyoBoTkyrnNBLc8tJQ0JtXomKzBbj5qPtHDduXc6xz_lzvNpxSPxi428m7wkAAA

grant_type=authorization_code&code=mJAReTWKK7zI3oHUNd4o3PeNqNqxKGP6&code_verifier=provided_tb&client_id=example-native-client-id
Example Bound Code: Web Server Client Authorization Request

HTTP/1.1 302 Found
Location: https://server.example.com?response_type=code
  &client_id=example-web-client-id&state=P4FUFqYzs1ij3ffsYCP34d3
  &redirect_uri=https%3A%2F%2Fclient%2Eexample%2Eorg%2Fcb
  &code_challenge=referred_tb&code_challenge_method=referred_tb
Include-Referred-Token-Binding-ID: true

GET /as/authorization.oauth2?response_type=code
  &client_id=example-web-client-id&state=dryo8YFpWacbUPjhBf4Nvt51
  &redirect_uri=https%3A%2F%2Fclient%2Eexample%2Eorg%2Fcb
  &code_challenge=referred_tb
  &code_challenge_method=referred_tb HTTP/1.1
Host: server.example.com
Sec-Token-Binding: ARIAaBBQB-XOPf5ePlf7ikATiAFEGOS5031PmRfkyyymzdWw
  HCxl0njjxC3D0E_OVfBNqrIQxzIfkF7tWbyZfyaE6XpwTsAQBYqhFX78vMOgDX_F
d_b2d1HyHLMkIz8iMVBY_reM98OUaJFz5IB7PG9nZ11j58LoG5QhmQoI9NXYktKZ
RXxrYAAAECFAEFAdUFTnfQADkn1uDbQnvJEk6oQs38L92gv-KO-q1YadLoDIKe2h53
hSiKwIP98iRj_unedkNkAMyg9e2mY4Gp7WwBAeDU0waSXNz1e6gKohwN4SAZ5eNyx
45Mh8VI4woL1BipLoqrJRoK6dxFkWgHRMuBROcLGUj5PiOoxybQH_Tom3gAA
Example Bound Code: Authorization Response to Web Server Client and Token Request

GET /cb?state=dryo8YFpWacbUPjhBf4Nvt51&code=jwD3o0a5cQvvLc81bwc4CMw
Host: client.example.org
Sec-Token-Binding: AIkAAgBBQHVBU530AA5J9bg20J7yRJOqELN_C_dOL_ijvqpW
Gns6AyCnToed4UoisCD_fIkY_7p3nZDZADMoPXtpmOBqelsAQEwgC9Zpg7QFCDBib
6GlZki3MhH32KNfLefLJc1vR1xE817OMfPLZHP2Woxh6rEtmgBcAABubEbTz7muNl
Ln8uoAAA

POST /as/token.oauth2 HTTP/1.1
Host: server.example.com
Content-Type: application/x-www-form-urlencoded
Authorization: Basic b3JnLmV4YW1wbGUuY2xpZW50Om1ldGY5OGNoaWNhZ28=
grant_type=authorization_code&code=jwD3o0a5cQvvLc81bwc4CMw
&redirect_uri=https%3A%2F%2Fclient%2Eexample%2Eorg%2Fcb
&client_id=example-web-client-id
&code_verifier=AgBBQHVBU530AA5J9bg20J7yRJOqELN_C_dOL_ijvqpW
Gns6AyCnToed4UoisCD_fIkY_7p3nZDZADMoPXtpmOBqels
Open Issues

- Should the scope of this document include standardizing or recommending how to convey token binding information of an access token via RFC 7662 OAuth 2.0 Token Introspection?
- Should the scope of this document include standardization or guidance on token binding of JWT Client Authentication and/or Authorization Grants from RFC 7523?
Open Issues Part Deux

- The metadata and what can and cannot be reliably inferred from 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.
Open Issues Part Tres

- 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?
  - Fair question…
  - Raises another question: clustered web server clients likely really won’t want to have refresh tokens bound
    - private key access in distributed systems
    - Individual RT to TB key associations
    - APIs in support thereof
  - What can/should be done?
    - Rely on client_refresh_token_token_binding_supported?
    - Allow for a parameter to express the Token Binding ID to the token endpoint? (maybe useful for other reasons)
    - Something else?
    - Let ‘em deal with it?
Looking Ahead

- Token Binding documents progress to RFC
- Work through open issues
- Implementation experience and feedback
- Get the band back together again for IETF 99 in Prague