XYZ vs XAuth

IETF 107 : TxAuth Bof
Interaction

- XYZ
  - Client expresses all possible interaction capabilities such as redirect, user_code, didcomm as separate fields
  - AS responds to any interaction capabilities it supports and requires per policy

- XAuth
  - Client states if it can do a redirect interaction (GS can redirect back to client), or must do an indirect interaction (GS won't be able to redirect back to client)
  - GS responds with parameters to use, or an error if not supported
Data Representation

• XYZ
  • Protocol is centered around a transaction (akin to OAuth “grant”)
  • uses a single URL for interactions around transaction
  • handles represent the transaction for continuity between requests

• XAuth
  • protocol is RESTful (GET, PATCH, POST, PUT, DELETE, OPTIONS)
  • GS URI is identifier for GS, and is URI to create Grants
  • URIs represent Grants and Authorizations with associated access tokens (and any other objects such as Sessions created later)
Client Authentication

- **XYZ**
  - client proves use of bound keys via general-purpose mechanisms, including detached JWS, DPoP, OAuth PoP, HTTP Sig, and MTLS
  - RS access via bearer token or proof-of-possession through any allowable key binding mechanism

- **XAuth**
  - client proves use of bound keys through an auth mechanism at GS
  - specifies default mechanism using JOSE for GS and RS proof-of-possession calls
  - RS access via bearer token just like OAuth 2.0
  - extensions can define other mechanisms such as HTTP Sig or MTLS to replace JOSE for either GS and/or RS calls
OAuth 2.0 / OIDC compatibility

• XYZ
  • use key handles to identify Client, or uses public key presented by value (no explicit difference between dynamic and static clients in the protocol)
  • support for subject, email, phone, ID Token claims
  • rich resource request, supports OAuth/OIDC style scopes in the same structure through resource handles
  • access token refresh is done with transaction handle to transaction endpoint (transaction / grant oriented, similar to refresh token)
  • support for OIDC UserInfo Endpoint through access token for additional claims

• XAuth
  • uses Client ID to identify registered Clients, just as it was used in OAuth 2.0 / OIDC
  • Dynamic Clients are identified by public key value (same as XYZ)
  • directly reuses OAuth scopes
  • allows rich resource requests from RAR
  • support for all OIDC Claims in an ID Token, or separately
  • uses a per-access-token refresh token and URL (token / authorization oriented)
Discovery

- XYZ
  - Client always starts at the tx endpoint, all other information is dispatched from responses from the endpoint
  - Clients sends capabilities list in transaction request, AS selects and returns which capabilities are supported
- XAuth
  - Client sends an OPTIONS call to the GS URI, Grant URI, or AZ URI