WHERE WE ARE IN THE WIMSE ARCHITECTURE
WHERE WE ARE NOT THE WIMSE ARCHITECTURE

• Conveying the call chain or stack or full context of invocation via a directed hopefully acyclic graph
• Cross domain
  • JWT Profile for OAuth [...] Authorization Grants (RFC7523)
  • OAuth Identity and Authorization Chaining Across Domains (draft-ietf-oauth-identity-chaining)
  • Whatever some huge providers insist on calling OpenID Connect
  • Whatever WIMSE might be doing with Token Exchange
• Conveying anything about the external authentication/authorization context of the call (but allow for checking the integrity thereof)
  • Transaction Tokens (draft-ietf-oauth-transaction-tokens)
  • OAuth bearer token (original or exchanged) or BYO token thing
• Distributing or provisioning anything to these workloads
OPTIONS

Transport-Level

MTLS

Application-Level

WIMSE Identity Token (WIT)

DPoP-Inspired

HTTP Message Signatures

Choose One!
Universal Uniform Resource Identifiers
⇒ Architecture draft, architecture slides
{  
  "alg": "ES256",
  "typ": "wimse-id+jwt",
  "kid": "July 4"
}
.
{
  "iss": "wimse://example.com/trusted-central-authority",
  "sub": "wimse://example.com/specific-workload",
  "exp": 1717612470,
  "jti": "x-_1CTL2cca3CSE4cwb__",
  "cnf": {
    "jwk": {
      "kty": "OKP",
      "crv": "Ed25519",
      "x": "_amRC3YrYbHhH1RtYrL8cs..."
    }
  }
}
.
[Signature]

- iss, sub, exp, jti and cnf claims are **required**.
- HTTP-Header “Workload-Identity-Token”. Not a Bearer token!
- Key binding via RFC 7800. (Self-contained via JWK in the confirmation claim)
- No audience
- Differs from JWT-SVID: added iss and cnf claims, removed aud claim
DPOP-INSPIRED OPTION: WIMSE PROOF TOKEN (WPT)

{  
  "alg": "EdDSA",
  "typ": "wimse-proof+jwt"
}

{  
  "iss": "wimse://example.com/specific-workload",
  "aud": "https://service.example.com/path",
  "exp": 1717612820,
  "jti": "__bwc4ESC3acc2LTC1-_x",
  "ath": "CL4wjfpRmNf-bdYI...GwKYE10wUwzC0jI",
  "tth": "18_Ffsx-34tV3hRI...lpBhF42UQfWVAW"
}

[Signature]

- Signature corresponds to key in ‘cnf’ of Workload Identity Token.
- optional ‘ath’, ‘tth’ and ‘oth’ hashes bind access-, transaction- and “other” token.
- HTTP-Header “Workload-Proof-Token”
GET /gimme-ice-cream?flavor=vanilla HTTP/1.1
Host: example.com
Signature: wimse=:...dkqa2WfCQ==:
Signature-Input: wimse="@method" "@request-target" "workload-identity-token";created=1718291357;
expires=1718291657;nonce="abcd1111";
tag="wimse-service-to-service"
Workload-Identity-Token: aGVhZGVyCg.VGh...Lgo.c2l...JlCg

- The draft defines which headers must be signed
- Easy to extend for different context headers
- Optionally: sign the response
THE TLS IS MUTUAL

• Similar to the SPIFFE X.509 approach
• Open issues
  • Handling hostname validation
  • Trust domain mapping
Discussion
NEXT STEPS

• Ready for WG adoption?
Which mechanism does the working group prefer? Should we focus on one or on both?
Authorization header and error responses · Issue #15 · yaronf/wimse-s2s (github.com)

- 401 HTTP response code requires WWW-Authenticate header. 
  *WWW-Authenticate: <type> realm=<realm>*
- Is 401 the correct choice?
- New type for WWW-Authenticate required?
• Which aspects should the proof of possession cover? Opinionated list of headers? Body?
• Should it be a choice of the deployment?
Currently proof of possession is one-shot, no request-response pattern (e.g. to exchange a nonce).

Now, “Freshness” is only determined by claims.
Should a hash of the Workload Identity Token be present in the proof?

Now:
- public key WIT <=> Signature of proof token
- WIT “sub” == Proof “iss”