Brian Campbell
Joe Salowey
Arndt Schwenkschuster
Yaron Sheffer

IETF 122, Bangkok

draft-ietf-wimse-s2s-protocol-03

https://datatracker.ietf.org/doc/draft-ietf-wimse-s2s-protocol/

WIMSE SERVICE TO SERVICE AUTHENTICATION

Brian Campbell
Joe Salowey
Arndt Schwenkschuster
Yaron Sheffer

IETF 122, Bangkok

draft-ietf-wimse-s2s-protocol-03

https://datatracker.ietf.org/doc/draft-ietf-wimse-s2s-protocol/

WORKLOAD TO WORKLOAD

WIMSE ~~SERVICE TO SERVICE~~ AUTHENTICATION

Changes Since Last IETF: versions -02 and -03

- **Coexistence with bearer tokens**
- Improve the architecture diagram
- Some more ABNF
- Clarified identifiers and URIs
- Moved an author to acknowledgments

- Consistently use "workload"
- **Implement comments from the SPIFFE community**
- Make *iss* claim in WIT optional and add wording about its relation to key distribution
- Remove *iss* claim from WPT
- Make *jti* claim in WIT optional
- Error handling for the application level methods

Discussion

Use of WIT without proof of possession and interop with legacy systems - Issue #53

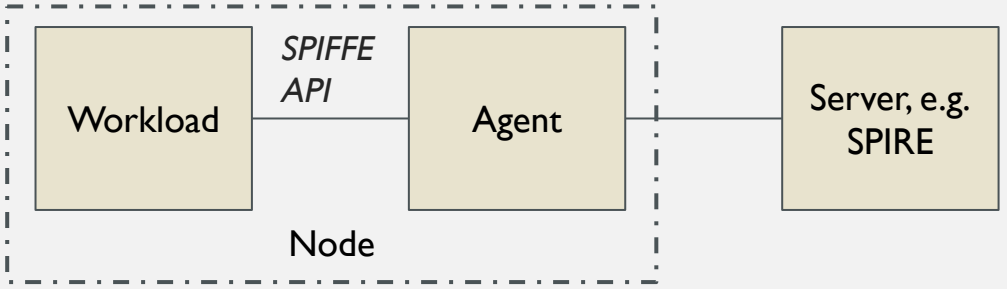
- Can WITs be used as bearer tokens to directly authenticate messages, in a migration scenario?
- But WITs are placed in their own (new) HTTP header, and have their own format which likely differs from legacy tokens
- New Sec. 4.6 discusses possible policies for migration, and the related interaction with Transaction Tokens

- The issue is still open, we propose to close it

SPIFFE and WIMSE WITs - SPIFFE Issue #315

- The SPIFFE community supports the WIMSE work
  - Reviewed the draft, several minor issues discussed
  - Clear delineation between S2S and SPIFFE
- Proposed API for a separate WIT-SVID
  - The API will remain a draft until S2S is stable, but plan is to implement it
  - The format definition will link to S2S

- Trust bundles are part of the API, may need to be discussed here
- The SPIFFE proposal is for WIT private keys to be provisioned from the Agent



Verifying the "aud" claim of a WPT - Issue #26

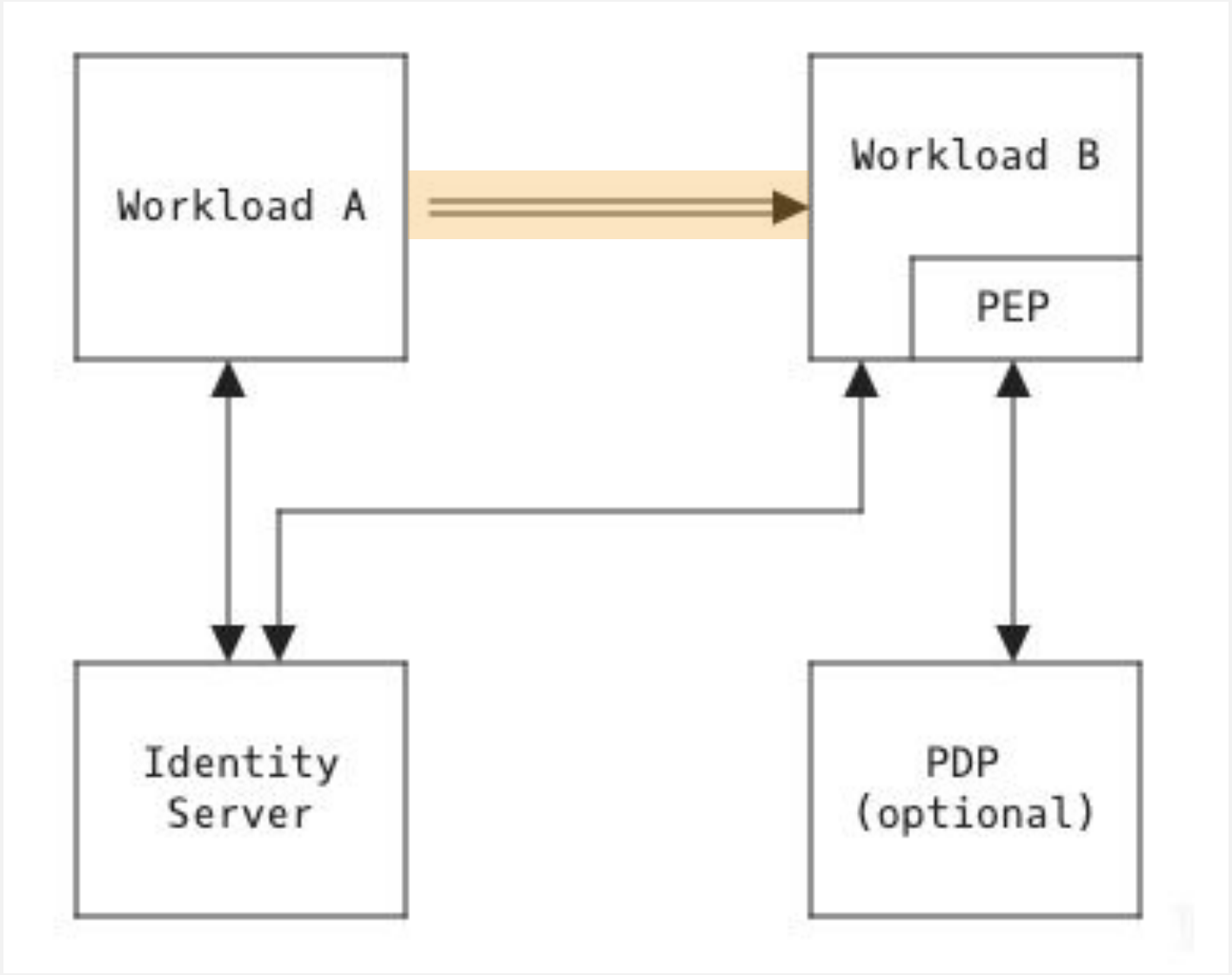
- How can the callee verify the *aud* claim of a WPT?
  - The caller may not know the callee's WIMSE ID
  - The callee may not know the Target URI used to call it
- The solution should allow for general purpose SDKs on the caller and the callee side, as well as local tweaks

- Still an ongoing discussion


No shortage of work ahead…
See you in Madrid this coming July!

Backup

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 large providers insist on calling OpenID Connect
  - Whatever WIMSE might be trying to do with Token Exchange
- Conveying **anything** about the external authentication/authorization context of the call (but kinda 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 or projecting anything to these workloads

WIMSE IDENTITY TOKEN (WIT)

```
{
    "alg": "ES256",
    "typ": "wimse-id+jwt",
    "kid": "July 4"
}
.
{
    "iss": "https://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": "l8_Ffsx-34tV3hRI...lpBhF42UQUfWVAW",
  "wth": "Ji8TlMNFk3qmzmWe...uFAFFSV1RgFOr3b"
}
.
[Signature]
```

- Signature corresponds to key in ‘cnf’ of Workload Identity Token.
- optional ‘ath’, ‘tth’ and ‘WTH’ hashes bind access-, transaction- and WIT token. ‘oth’ for other token
- HTTP-Header “Workload-Proof-Token”

MESSAGE SIGNATURES

```
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

Comparison

- WPT (DPOP-inspired)
  - +less HTTP specific, adaptable to other scenarios (messaging)
  - +less complex
  - +easier to leverage/layer existing implementations
  - +easier to adapt to other protocols
- HTTPSig
  - +existing spec and implementation
  - +response signing
  - +more flexible (additional fields may be signed)
  - +as a result: resistant to rogue middleboxes

What do you think?

THE MUTUAL TLS UPDATES

- Recommend Including DNS SAN for server certs for servers with non-WIMSE clients
- MUST validate trust domain portion of the identity
- SHOULD include client and/or server auth EKUs