WIMSE
Key Distribution & Discovery

Arndt Schwenkschuster - no draft

WIMSE Architecture specification

3.1.1 Trust Domain

“A trust domain maps to one or more trust anchors for validating X.509 certificates and a mechanism to securely obtain a JWK Set [RFC7517] for validating WIMSE WIT tokens. This mapping MUST be obtained through a secure mechanism that ensures the authenticity and integrity of the mapping is fresh and not compromised. ***This secure mechanism is out of scope for this document.****”*

WIMSE S2S Protocol specifications

3.1. The Workload Identity Token

*“The public key(s) of the Identity Server **need to be known to all workloads** in order to verify the signature of the WIT.”*

3.1.3. A note on iss claim and key distribution

“Implementations MAY include the iss claim in the form of a https URL to facilitate key distribution via mechanisms like the jwks_uri from [RFC8414] but alternative key distribution methods may make use of the trust domain included in the workload identifier which is carried in the mandatory sub claim.”

Existing mechanism: OIDC Discovery

- Defakto standard for key discovery across human & workload identity frameworks.
- Broad support with libraries and issuers.
- Pull-model: Revocation relies on verifiers to periodically check if key is still considered trusted.
- Effectively relies on WebPKI.
- OAuth2 Authorization Server Metadata exists too, with a potential different well-known endpoint.

Existing mechanism: SPIFFE

- SPIFFE bundle (JWKS, X.509).
- Distributed via the Workload API to the workload.
- Push-model: Workload is notified of updated trust.
- Not discoverable because trust domain is arbitrary and not resolvable.
  - spiffe://example.com is not necessarily a trust domain of the DNS owner of example.com.
- Almost every SPIFFE deployment add `iss` claim to JWT to allow for OIDC Discovery.
- Effectively 2 ways to discover keys:
  - SPIFFE via trust domain in the SPIFFE-ID (“sub”-claim)
  - OIDC Discovery + jwks_uri via iss claim
  - Not great for interop, security & operations.


```
{
  “sub”: “spiffe://my-trust-domain.com/my-workload”,
  “iss”: “https://ietf.org”
}
```

WIMSE?

- Key distribution & discovery is undefined.
- Similar to SPIFFE, trust domain is arbitrary and not resolvable too.
- Okay-ish for inter-trust-domain use cases
    - Deployment can be opinionated to only allow one.
- Problem for intra-/cross trust domain use cases
    - Workloads need to support both.
    - Unclear security promises (time to revocate unclear when some use pull instead of push)

Let’s work on this!