OAuth SPIFFE Client Authentication

https://datatracker.ietf.org/doc/draft-ietf-oauth-spiffe-client-auth/

Stian Thorgersen

Scott Rose

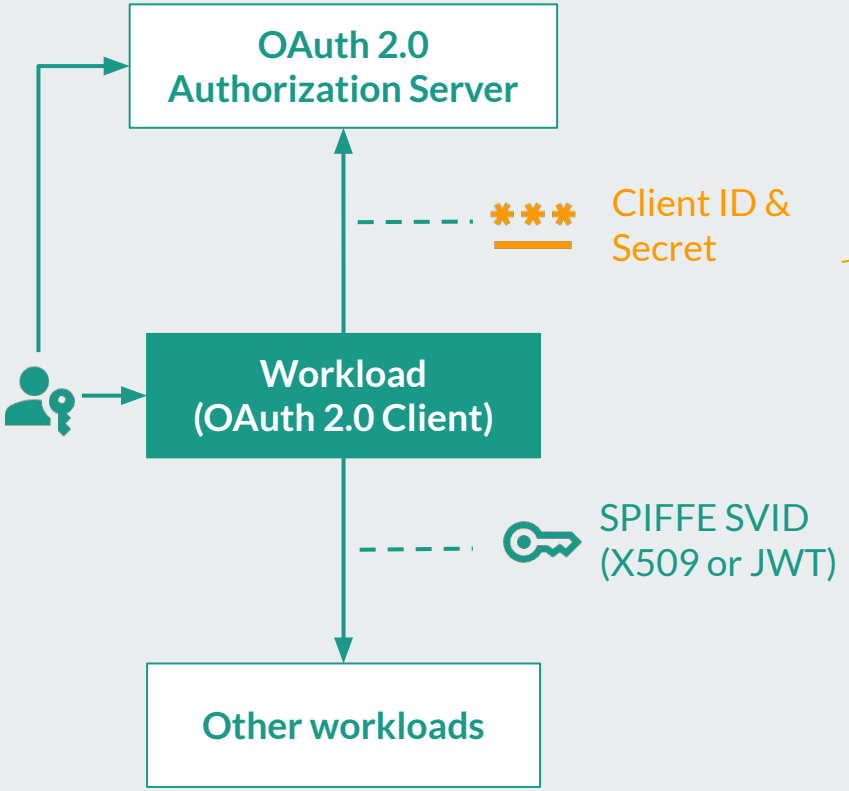
Pieter Kasselman

Arndt Schwenkschuster

SPIFFE (Secure Production Identity Framework For Everyone)

- CNCF project
- Identity framework for workloads
- SVID = Single Verifiable Identity Document
  - X509-SVID
  - JWT-SVID
  - **WIT-SVID (Workload Identity Token, WIMSE)**
- Used in the industry, mainly in enterprise environments:
  - https://learn.microsoft.com/en-ie/entra/workload-id/workload-identity-federation
  - https://cloud.google.com/iam/docs/managed-workload-identity
  - Open Source Implementation (SPIRE) & multiple commercial providers
  - Used at Uber, Netflix, Bytedance, Indeed, HPE, Confluent, Google and more.

Workloads acting as OAuth clients - Current situation



- Manual provisioning & rotation
- Long-lived
- Bearer credential

- Automatic provisioning & rotation
- Short-lived
- Proof of key possession possible (mTLS, WIT)

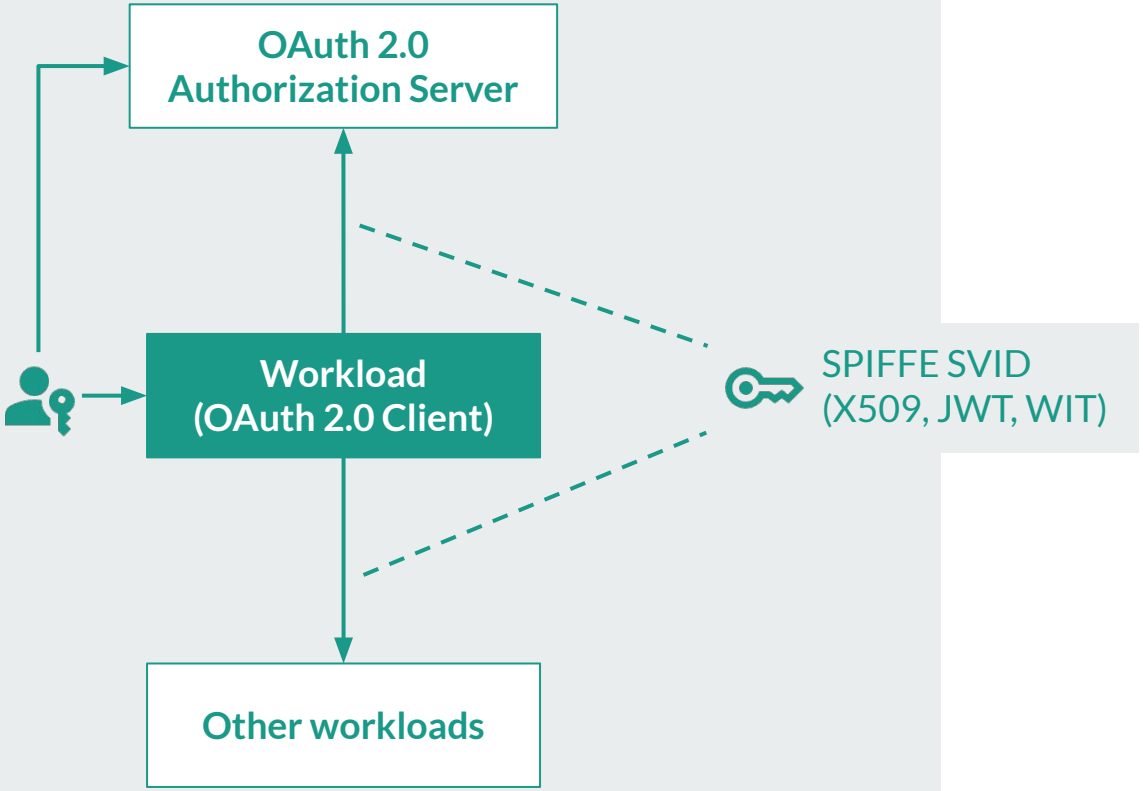
Workloads acting as OAuth clients - Current situation

```
spring:
 security:
  oauth2:
   client:
    registration:
     my-service:
      client-id: my-client-id
      client-secret: my-super-secret
      authorization-grant-type: client_creds
      scope: api.read,api.write
    provider:
     my-service:
      token-uri:
https://auth-server.example.com/oauth2/token
```

```
{
  "OAuth": {
    "TokenUrl": "https://auth.example.com/token",
    "ClientId": "my-client-id",
    "ClientSecret": "my-super-secret",
    "Scope": "api.read api.write"
  },
  "ExternalApi": {
    "BaseUrl": "https://api.example.com"
  }
}
```

* or a key used to produce a client assertion

Workloads acting as OAuth clients - The goal



- Leverage existing workload identity infrastructure
- Remove need for secret lifecycle management
- Reduce secret sprawl

Proposal

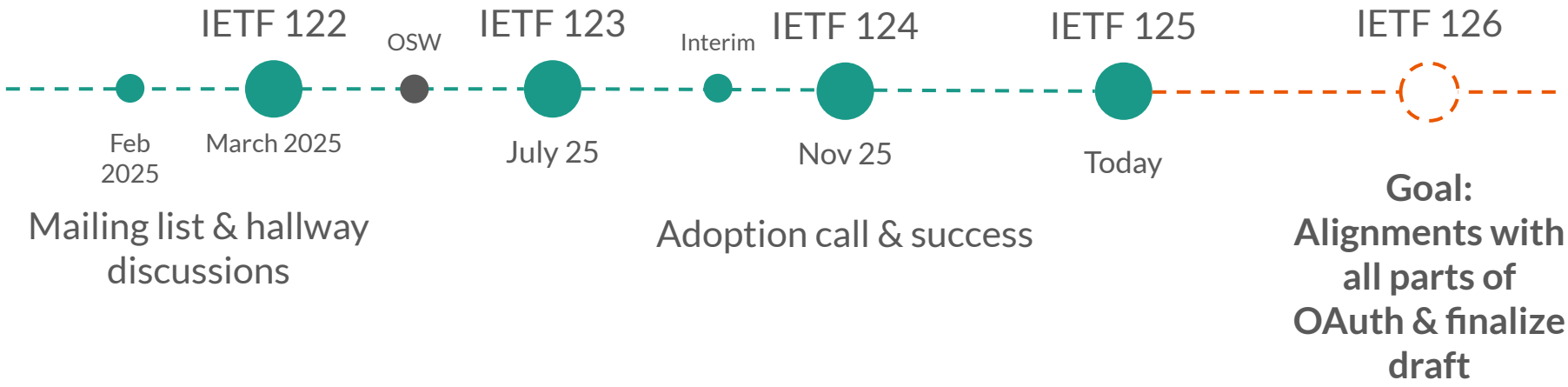
JWT-SVID	X.509-SVID	WIT-SVID
Profiles **RFC7521**	Profiles **RFC8705**	Profiles **Attestation-Based Client Auth** draft
Bearer JOSE token, carrying subject, audience & TTL	X.509 certificate issued by SPIFFE	WIMSE Workload Identity Token issued by SPIFFE

New!

Since IETF 124 - Alignment with other OAuth work

- Attestation-based Client Authentication support for key-bound Workload Identity Token (WIT-SVID)
- Client Instance Metadata Document support - thank you Aaron and Taka!
- New “token_endpoint_auth_methods_supported” (RFC8414) values:
  - spiffe_jwt
  - spiffe_x509
  - spiffe_wit

Timeline & Goals

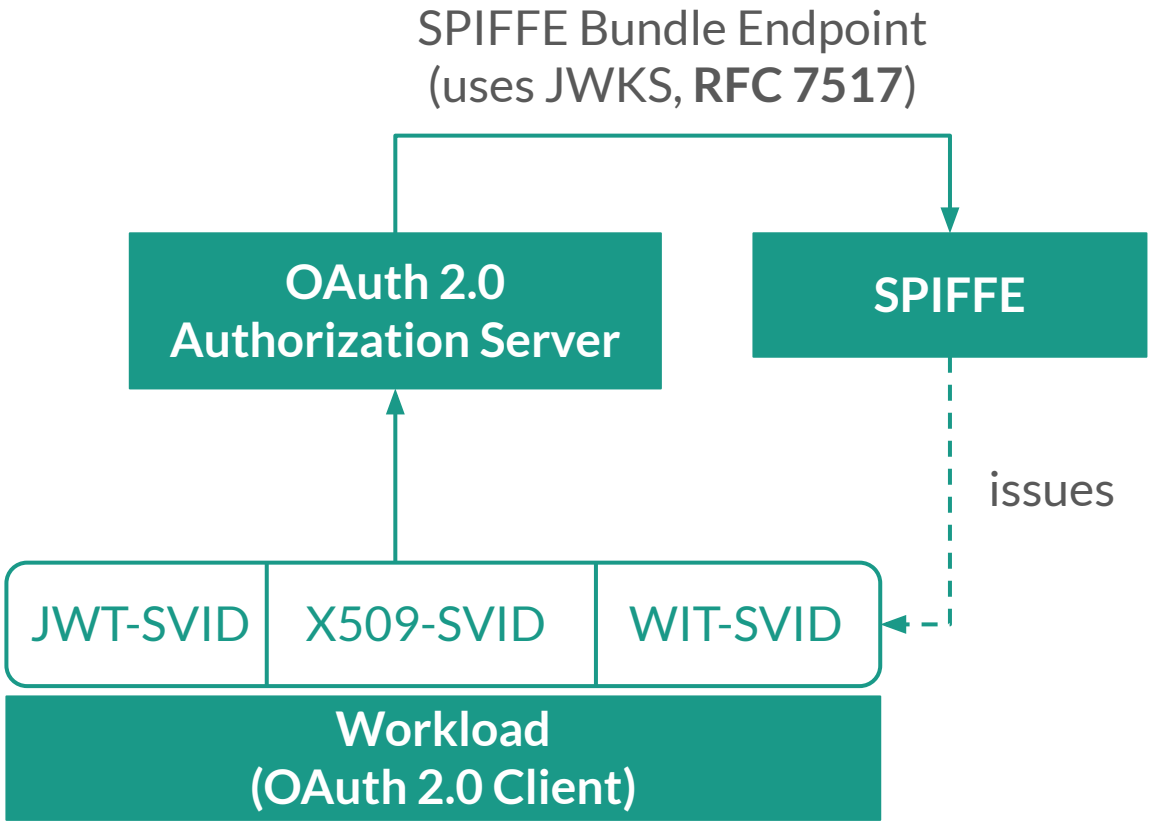


Backup

Proposal

Client Authentication using existing methods:

- RFC7521 - Assertion Framework
- RFC8705 - Mutual-TLS Client Authentication
- Attestation-based Client Authentication draft.



Focus & Scope

- Purely client authentication.
- Trust establishment and client registration required.
- Some of the automatic client registration individual drafts make use of this mean of client auth.
- There's value regardless:
  - Many SPIFFE-deployments use OAuth.
  - Enterprise environment allows/requires client registration anyway.
  - Works for flows involving humans (3-legged, such as auth_code flow with PKCE)
  - Or without humans (2-legged, client credentials flow)

Current draft outline

- OAuth Client Authentication Using SPIFFE
  - Client Authentication with JWT-SVIDs
  - Client Authentication using X509-SVID
- SPIFFE Trust Establishment and Client Registration
- SPIFFE Key Distribution and Validation
  - SPIFFE Bundle Endpoint
- Security Considerations

Since IETF 123 - Widespread interest & individual implementations

Various individual implementations:

- https://github.com/CarrettiPro/keycloak-spiffe
- https://github.com/christian-posta/spiffe-svid-client-authenticator
    - Corresponding blog post & demo: https://blog.christianposta.com/authenticating-mcp-oauth-clients-with-spiffe/

Other parties contacted the authors and expressed strong interest.

Since IETF 123 - Preview implementation in Keycloak 26.4

- Keycloak implemented this draft in this PR and released it under “preview” in their latest release 26.4

Keycloak 26.4.0

This release features new capabilities focused on security enhancements, deeper integration, and improved server administration. The highlights of this release are:

- Passkeys for seamless, passwordless authentication of users.
- Federated Client Authentication to use SPIFFE or Kubernetes service account tokens for client authentication.
- Simplified deployments across multiple availability zones to boost availability.
- FAPI 2 Final: Keycloak now supports the final specifications of FAPI 2.0 Security Profile and FAPI 2.0 Message Signing.
- DPoP: The OAuth 2.0 Demonstrating Proof-of-Possession at the Application Layer (DPoP) is now fully supported. Improvements include the ability to bind only refresh tokens for public clients, and securing all Keycloak endpoints with DPoP tokens.

Read on to learn more about each new feature. If you are upgrading from a previous release, review also the changes listed in the upgrading guide.