OAuth2 SCIM Client Registration & Software Statement Exchange

Phil Hunt

July 31, 2013

IETF87 OAuth and SCIM WG

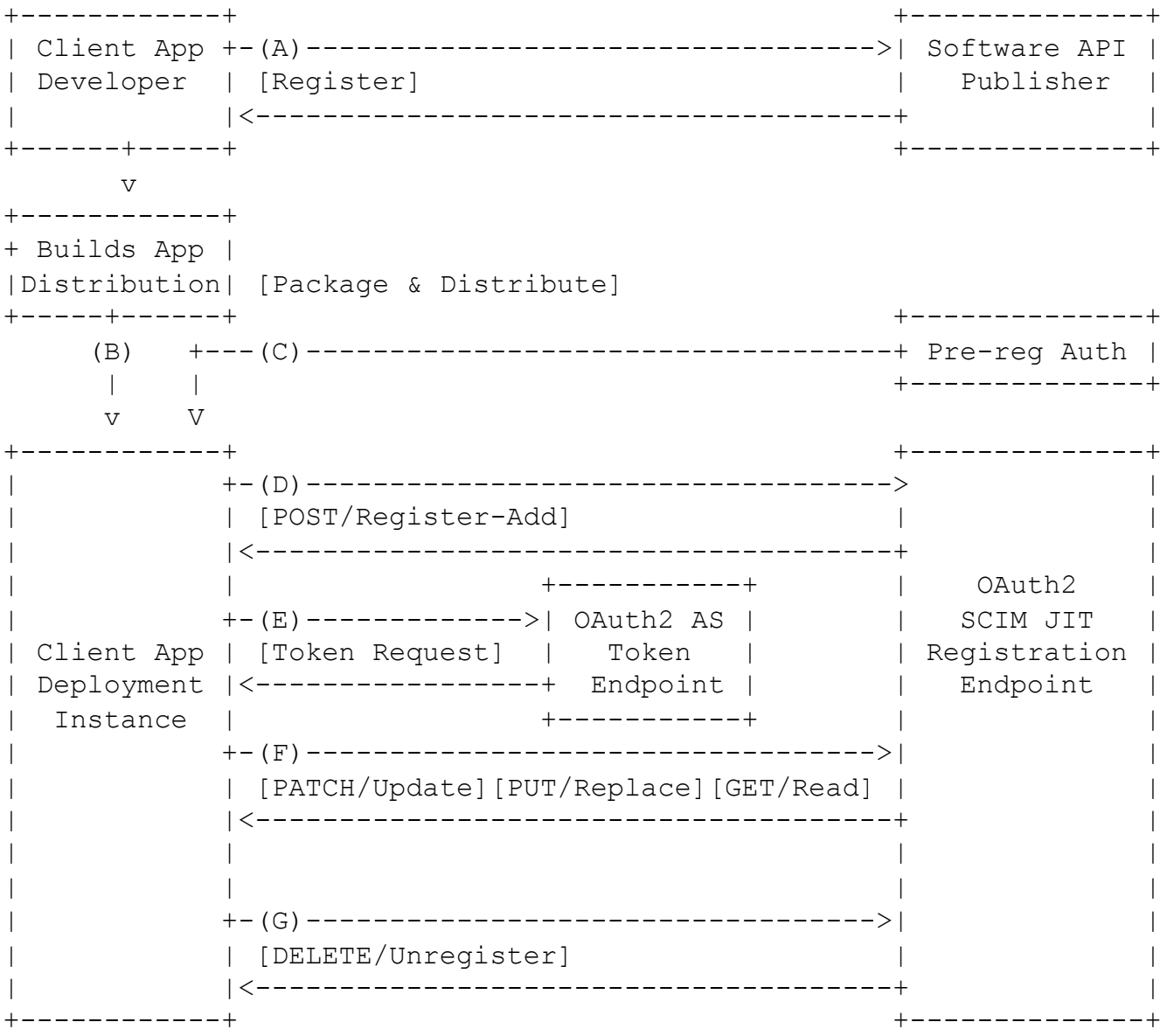
OAuth-SCIM-Client-Reg Intro

- Draft enables OAuth 2 clients to register with a SCIM endpoint to obtain client id and optional credentials.
- Based on draft-ietf-oauth-dyn-reg-12
  - Essentially the same attribute model
- Uses SCIM API and SCIM Schema Extensions
- Additional items
  - Uses software_id, software_version
  - Software Assertion
  - Scope, and target API

Agenda

- Profile Flow and Representation
- Software Assertion (Statement)
- Alternate Software Statement Exchange Flow
- Discussion

Basic Flow


Client App Developer
(A)
[Register]
Software API Publisher
Builds App Distribution
[Package & Distribute]
(B)
(C)
Pre-reg Auth
(D)
[POST/Register-Add]
(E)
[Token Request]
OAuth2 AS Token Endpoint
OAuth2 SCIM JIT Registration Endpoint
Client App Deployment Instance
(F)
[PATCH/Update][PUT/Replace][GET/Read]
(G)
[DELETE/Unregister]

Example Client Representation


```
{
  "schemas":["urn:scim:schemas:core:1.0",
             "urn:scim:schemas:oauth:2.0:Client"],
  "id":"2060107e82-fbe3-42bd-b199-15df7081a8ae",
  "software_id":"5ed2dd14-3ef7-4655-a41d-b5bd4c5266cc",
  "software_version":"5.1.2.3.4",
  "client_name":"Example Social Client",
  "logo_uri":"https://client.example.org/logo.png",
  "jwks_uri":"https://client.example.org/my_public_keys.jwks",
  "token_endpoint_auth_method":"client_secret_post",
  "scope":"read write dolphin",
  "client_id":"2060107e82-fbe3-42bd-b199-15df7081a8ae",
  "client_secret":"Z7tk2XqLKo1CfE14374teR4V554e8JUS",
  "redirect_urls":[""https://client.example.org/callback",
    "https://client.example.org/callback2"],
  "targetEndpoint":"https://social.example.com/base"
}
```

Software Statement

- A signed JWT bearer assertion issued by resource API software publisher to developer that may be used during registration
- Enables extended developer registration when publisher is not the deployer of resource APIs
- Enables OAuth Registration endpoint to recognize publisher registered software
- Enables OAuth registration endpoint to approve clients, or publishers for automatic registration
  - registration endpoints do not need continuous updates
- Statement is not an authentication or proof that the client is in fact the software asserted
  - Purpose is to serve as a "letter of introduction"
  - Client attributes are signed by publisher

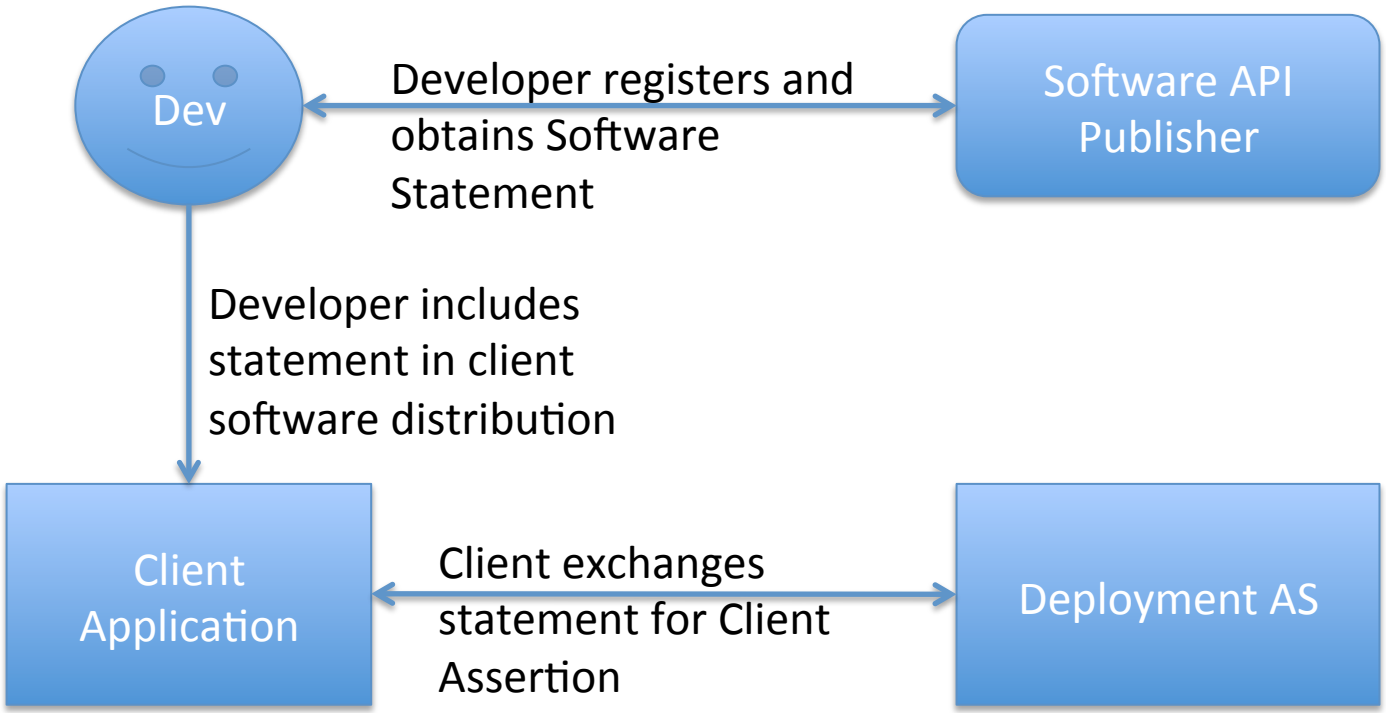
Security Consideration

- Concern that some sites may mistake this as "proof" that a client is what it claims
  - Are we confusing registration with authentication?
  - The "statement" is intended to indicate to the registration server as to what the client claims to be
  - A client that says it is one thing and behaves differently stands out (caught in a lie)
    - Making no statement means clients are totally anonymous except by looking at other registration data.
  - Use techniques like secure app store for distribution
  - Administrators can test, and create local distributions with locally issued "initial access tokens".

Software Statement Attrs

- iss – a unique identifier (URI) for the entity that issued the JWT. The value corresponds to the Software API Publisher
- sub – a unique value corresponding to "software_id". Typically assigned by the Software API Publisher
- aud – contains a value that identifies one or more Software API deployments where the client MAY be registered OR "urn:oauth:scim:reg:generic" indicating the assertion is intended for any OAuth registration endpoint.
- exp – an expiry date for the assertion.
- May contain any other client attribute from schema.

Software Statement Flow



Software Statement Exchange Flow

- Eliminate registration API
  - Registration occurs primarily between developer and Software API Publisher (not standardized)
- Uses signed Software Statements
- Leverages JWT/SAML Bearer flow
  - Client exchanges Software Statement for a Client Bearer Assertion
- Concern: How could this work for other client credential types?

Discussion

- What are the primary objectives?
  - Assign a client_id
  - Issue a client authentication credential
  - Provide service provider with information about client
  - other?

- 3 possible flows
  - Dynamic Reg
  - SCIM Client Reg
  - Software Statement Exchange
    - Which 1 or 2 is the way to go?

- Should the software statement (assertion) be generalized in its own draft