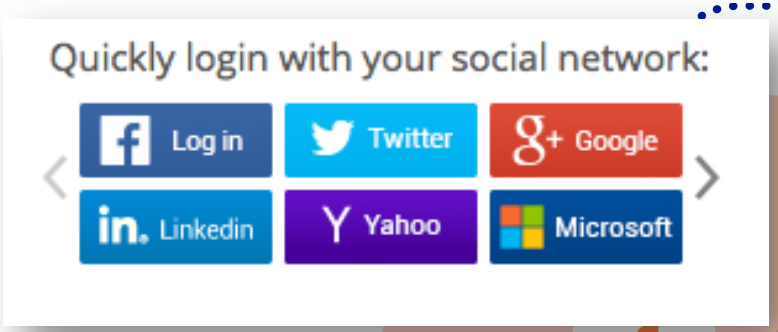
User-Managed Access (UMA) in the ACE Context

Eve Maler, chair
@UMAWG | @xmlgrrl
13 January 2015
tinyurl.com/umawg

Agenda

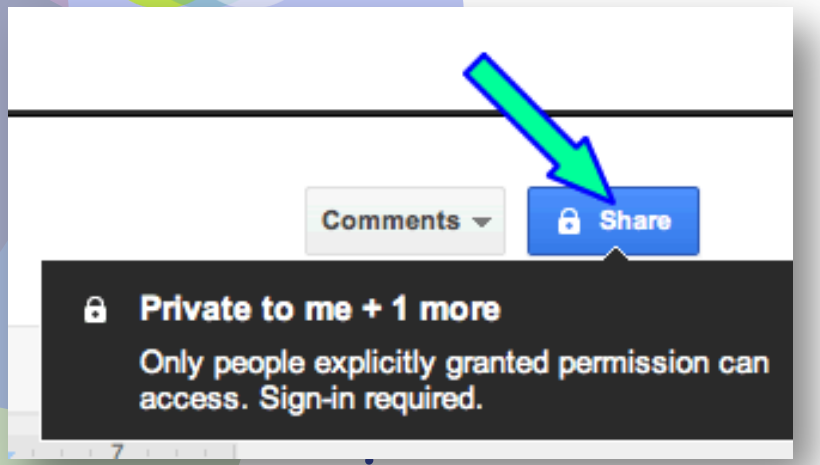
- UMA's design center, progress, and status
- A quick "UMA 101" primer
- Measuring UMA against ACE use cases
- Discussion and next steps

The “new Venn” of web access control and consent



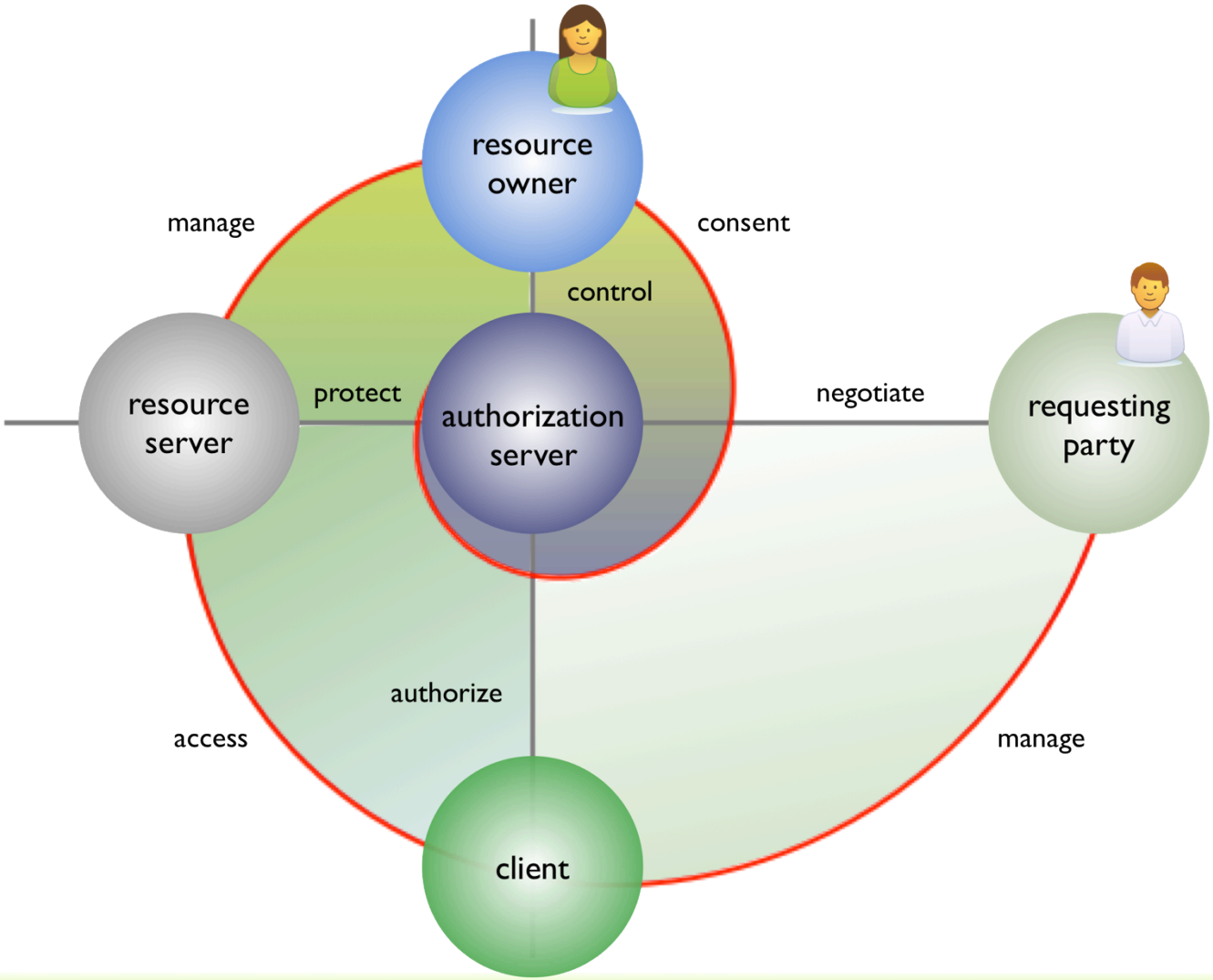
OpenID Connect

UMA

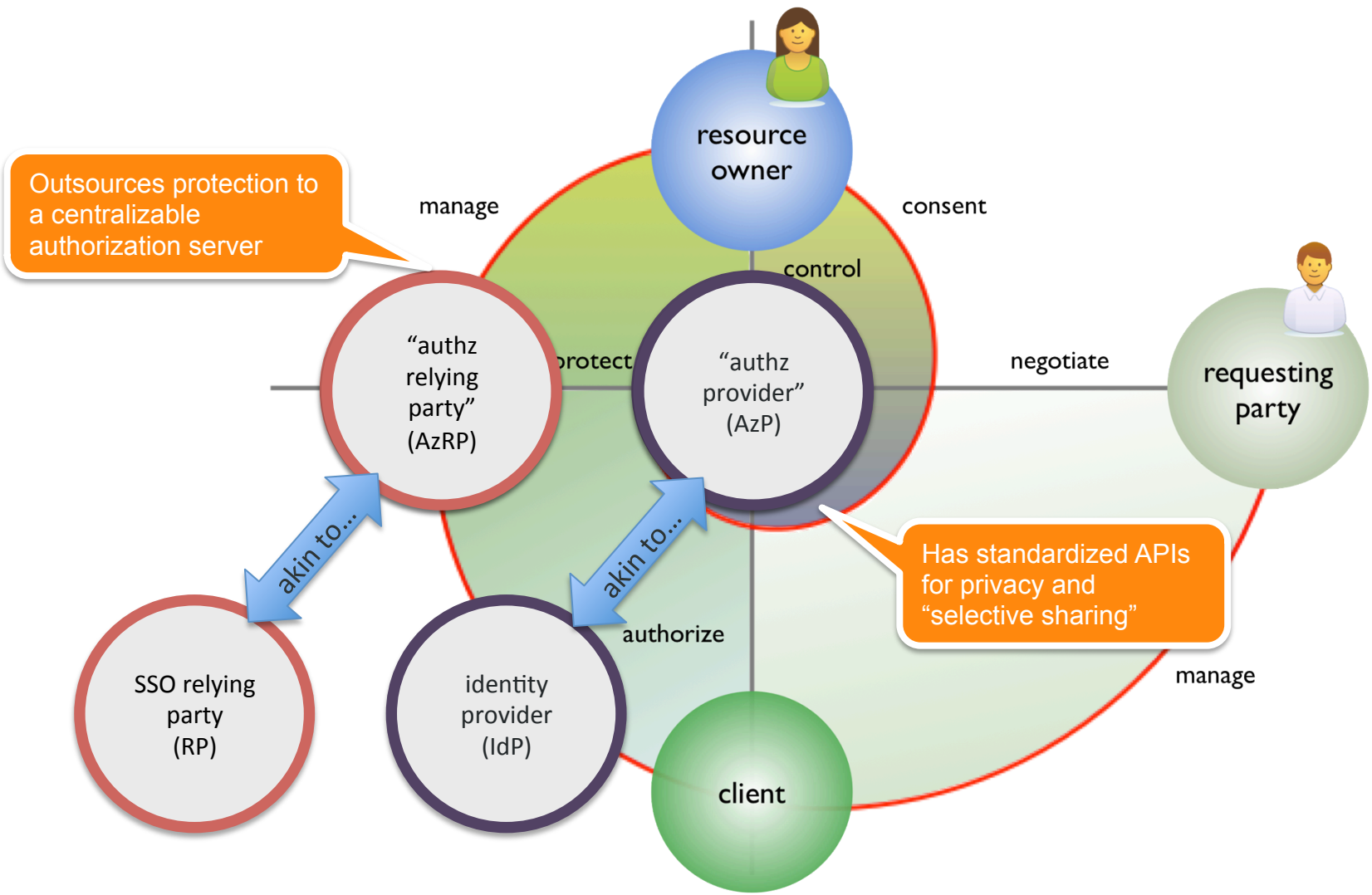


OAuth 2.0

The marvelous spiral of controlled personal data/access sharing



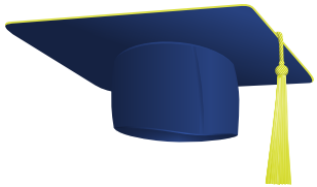
Interoperable, RESTful authorization-as-a-service



Use-case domains

Health

Financial



Education

Personal

Government

Media

Behavioral

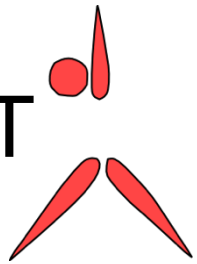
Enterprise

Web

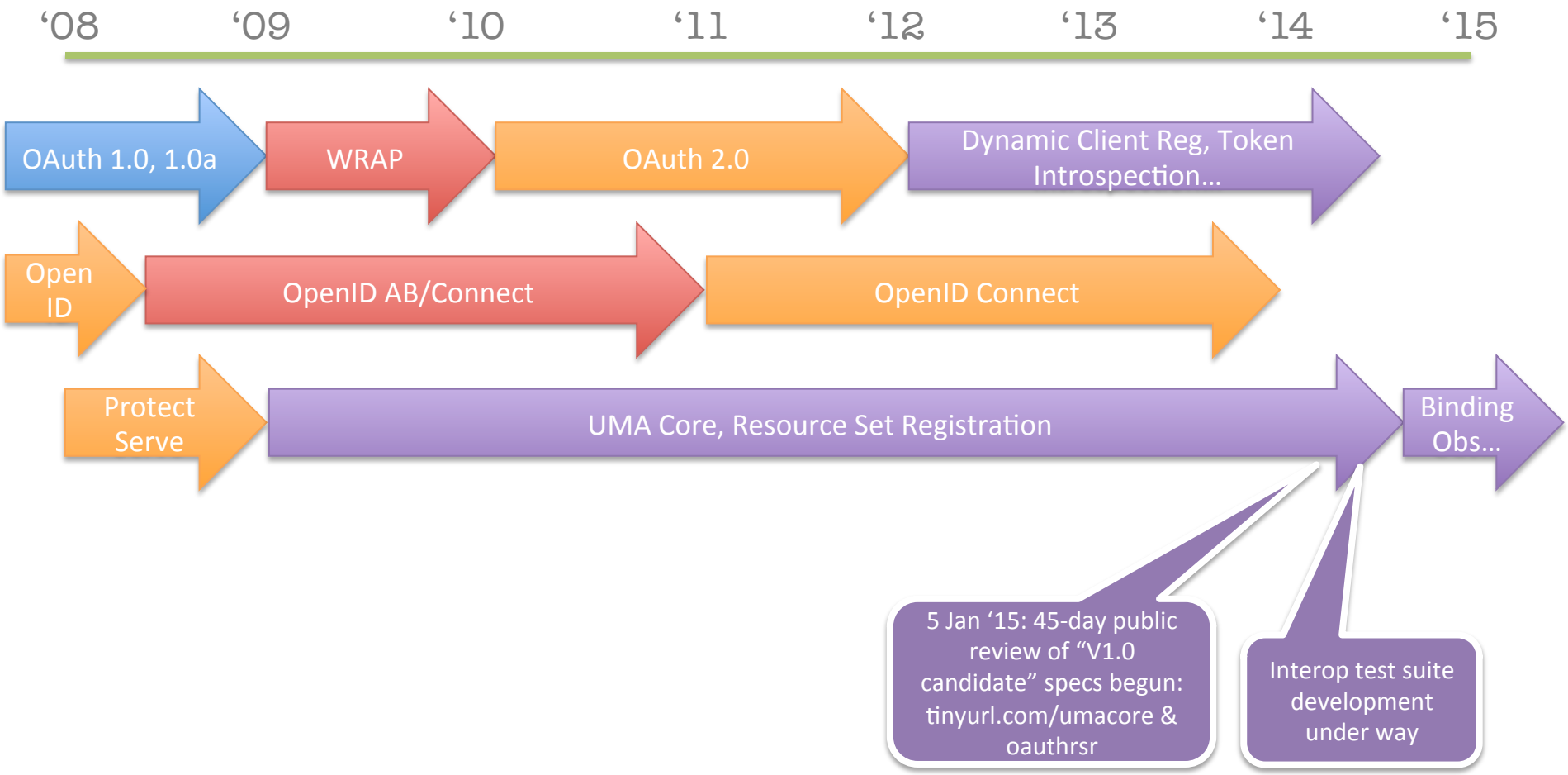
Mobile

API

IoT



Web/API identity and security specification progress in context



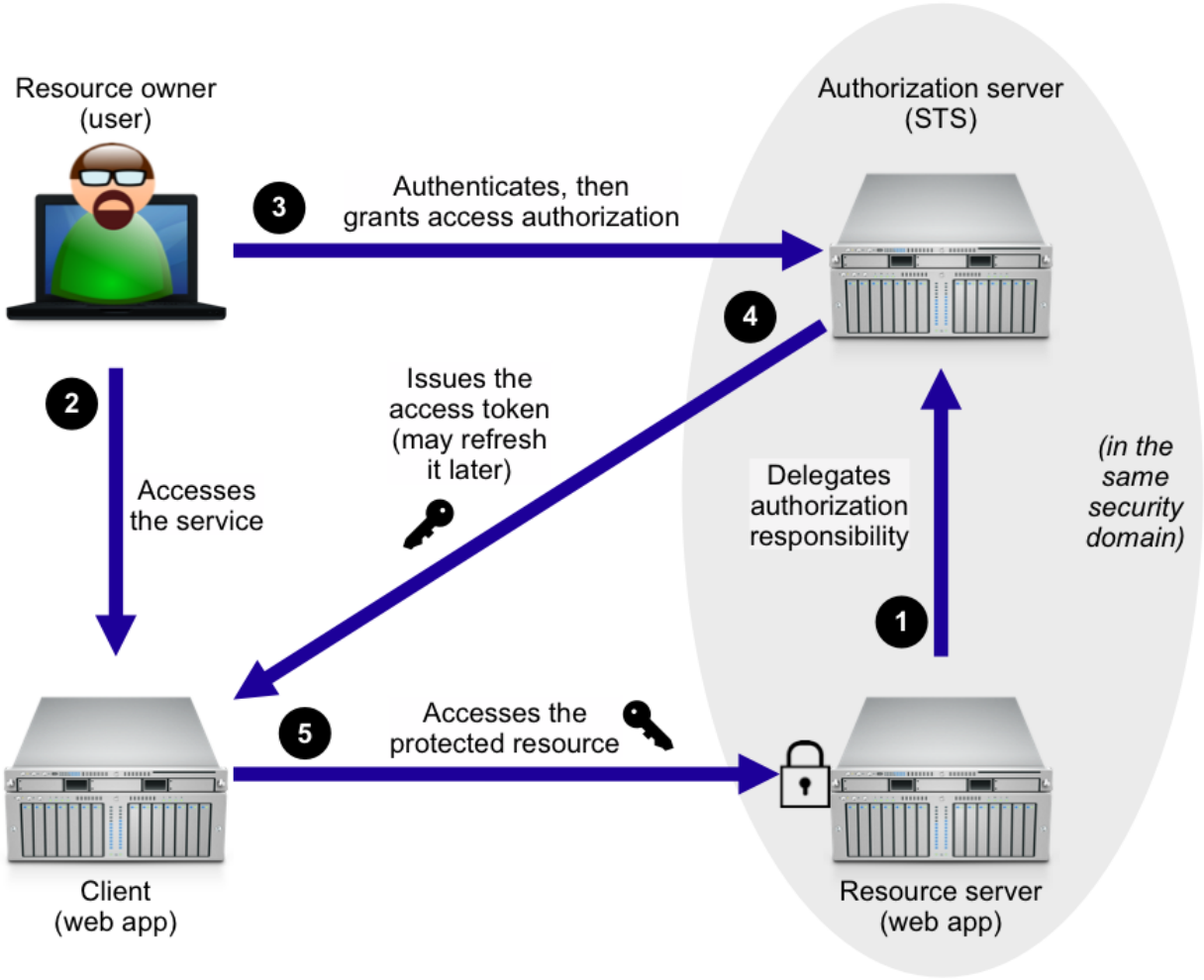
Other major news items

- EIC award in Munich
- HEART WG at OpenID Foundation
- New open-source community: OpenUMA at ForgeRock.org

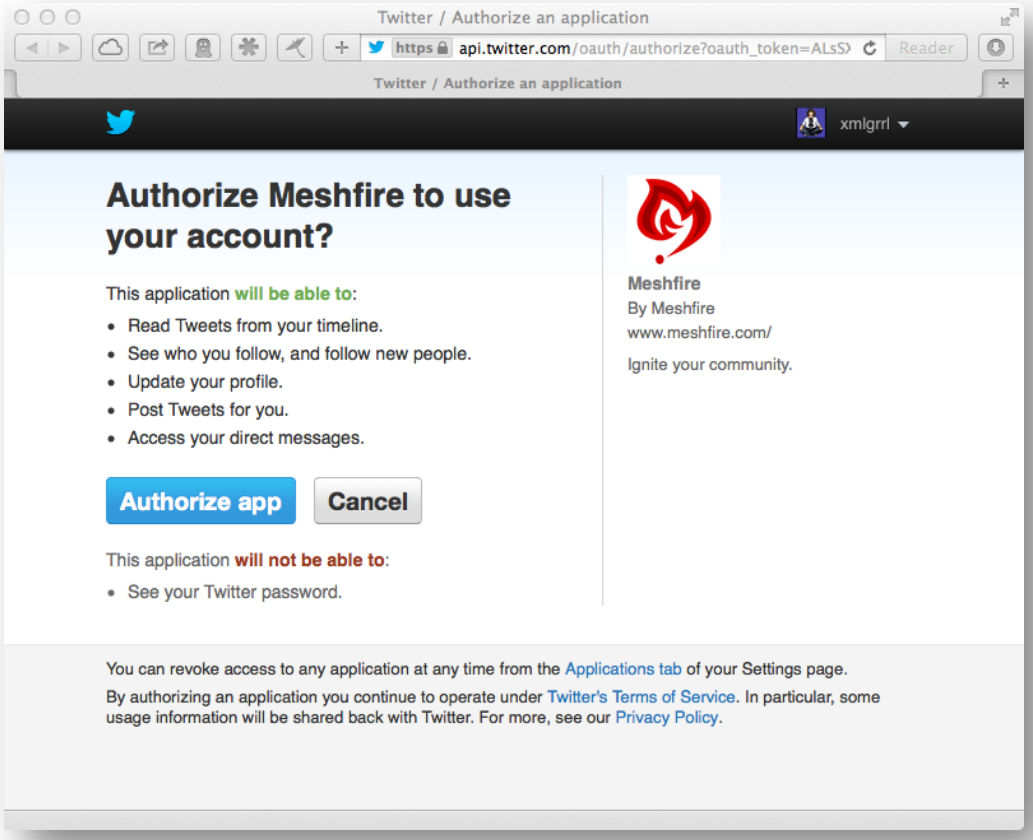
Agenda

- UMA's design center, progress, and status
- A quick "UMA 101" primer
- Measuring UMA against ACE use cases
- Discussion and next steps

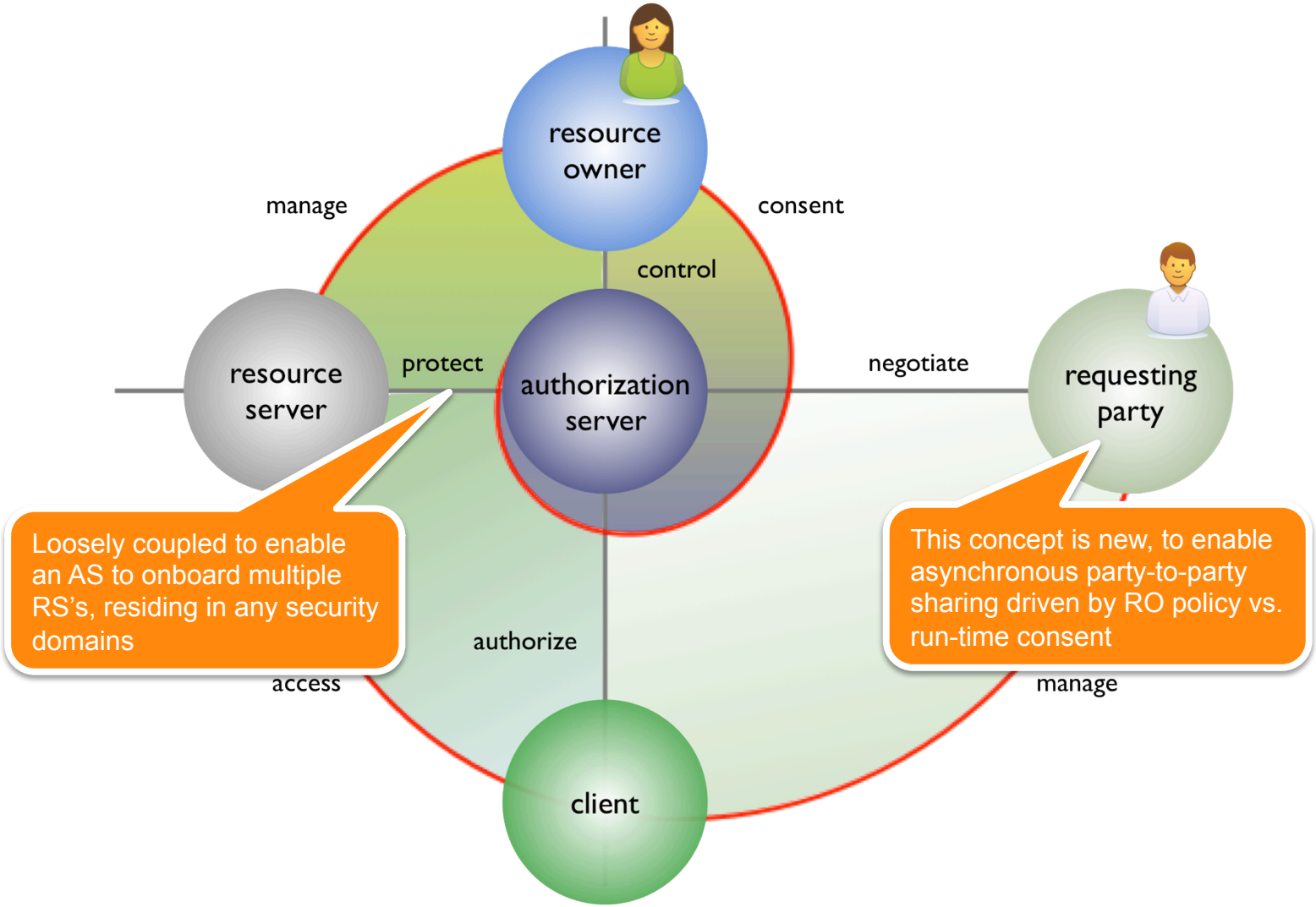
OAuth architecture



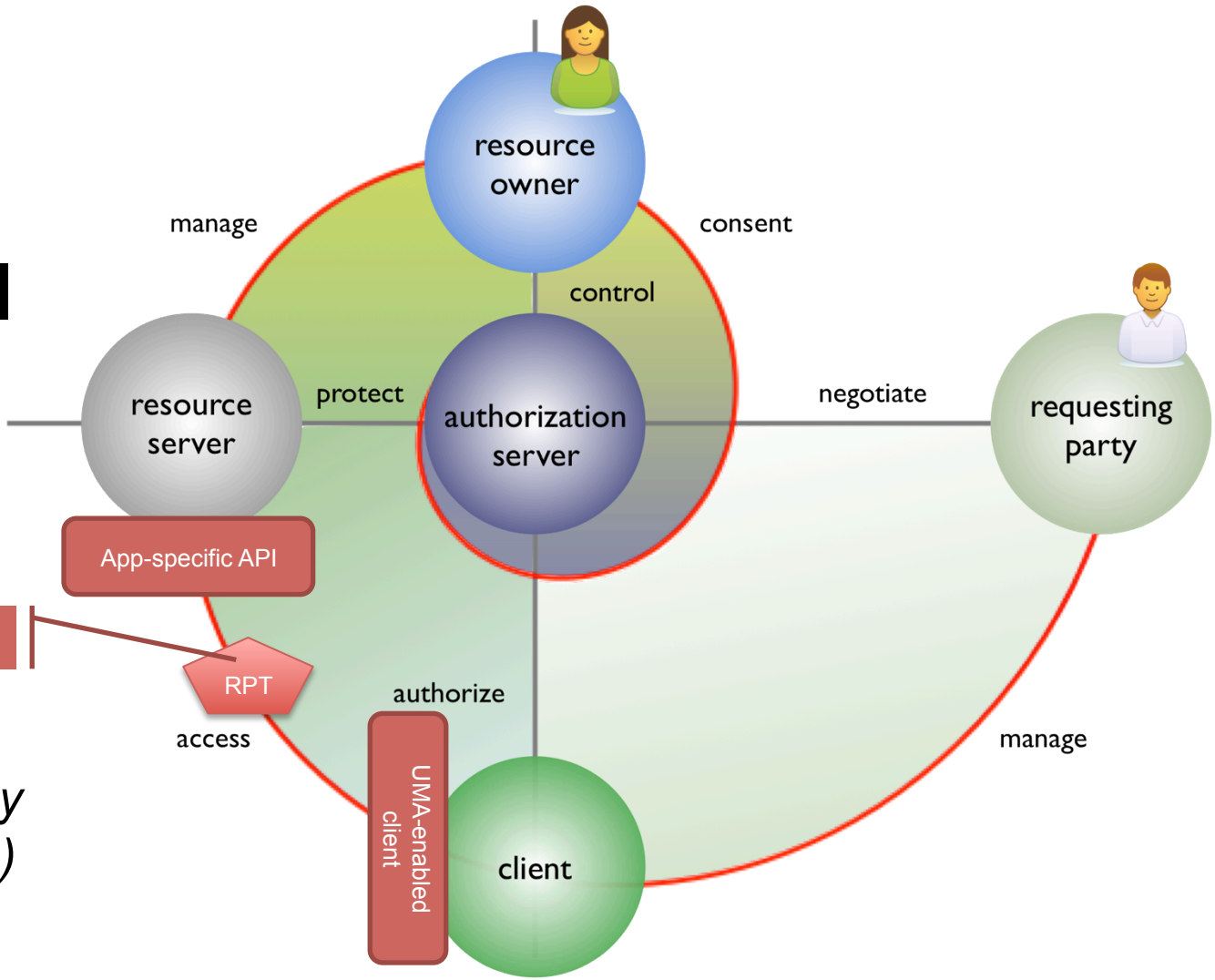
OAuth experience



Under the hood, UMA is “OAuth++”

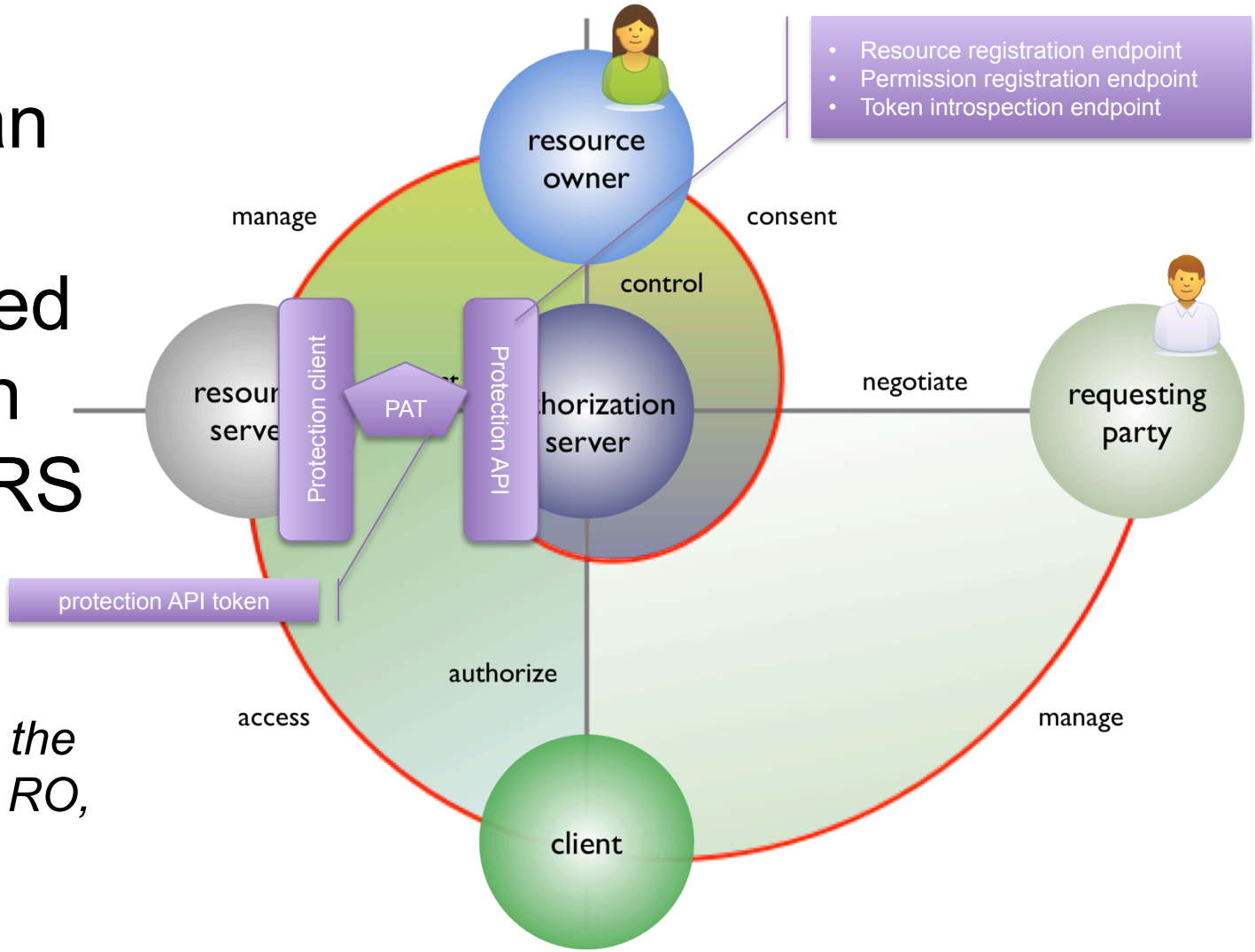


The RS exposes whatever value-add API it wants, protected by an AS



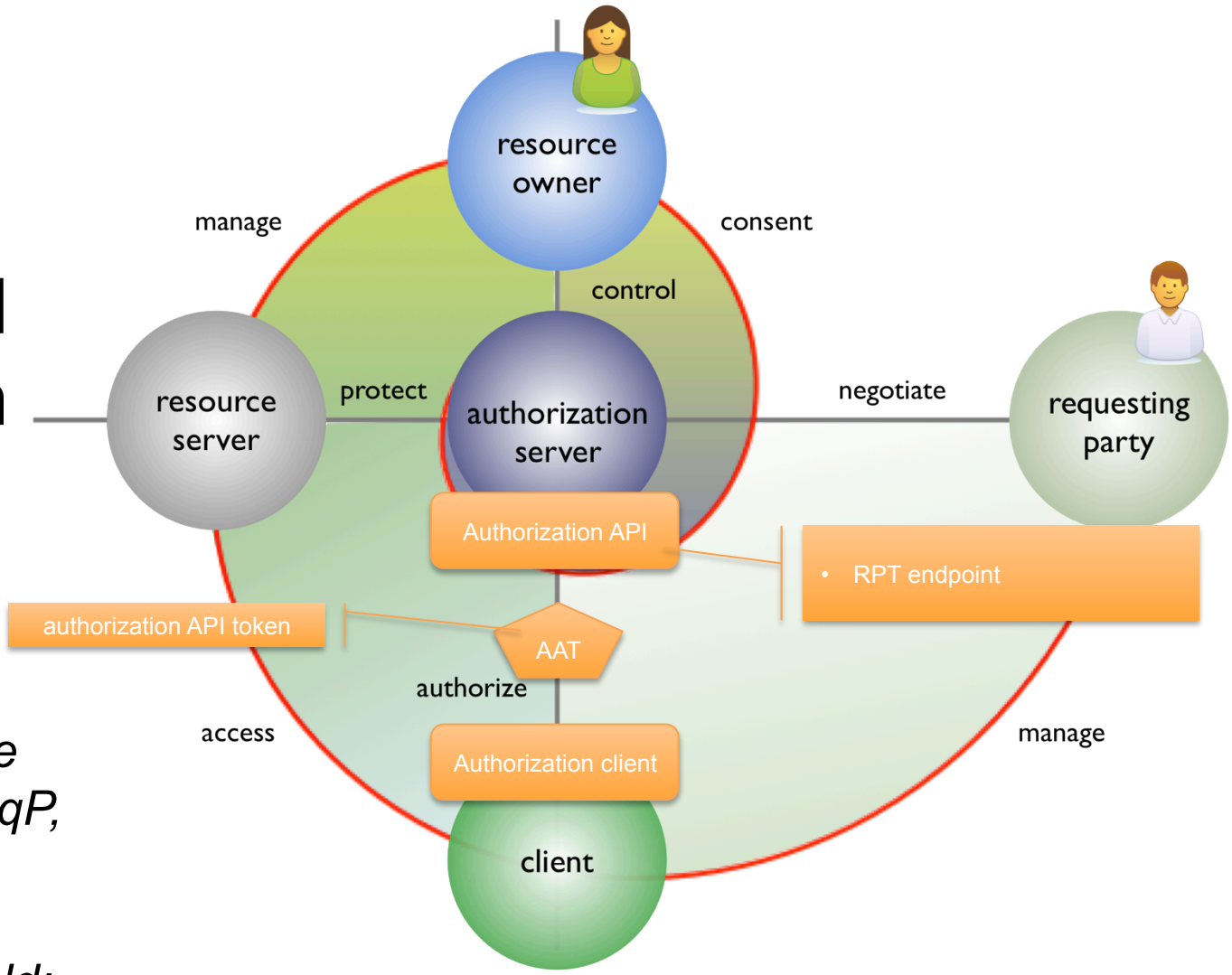
The RPT is the main "access token" and (by default – it's profilable) is associated with time-limited, scoped permissions

The AS exposes an UMA-standardized protection API to the RS



The PAT protects the API and binds the RO, RS, and AS

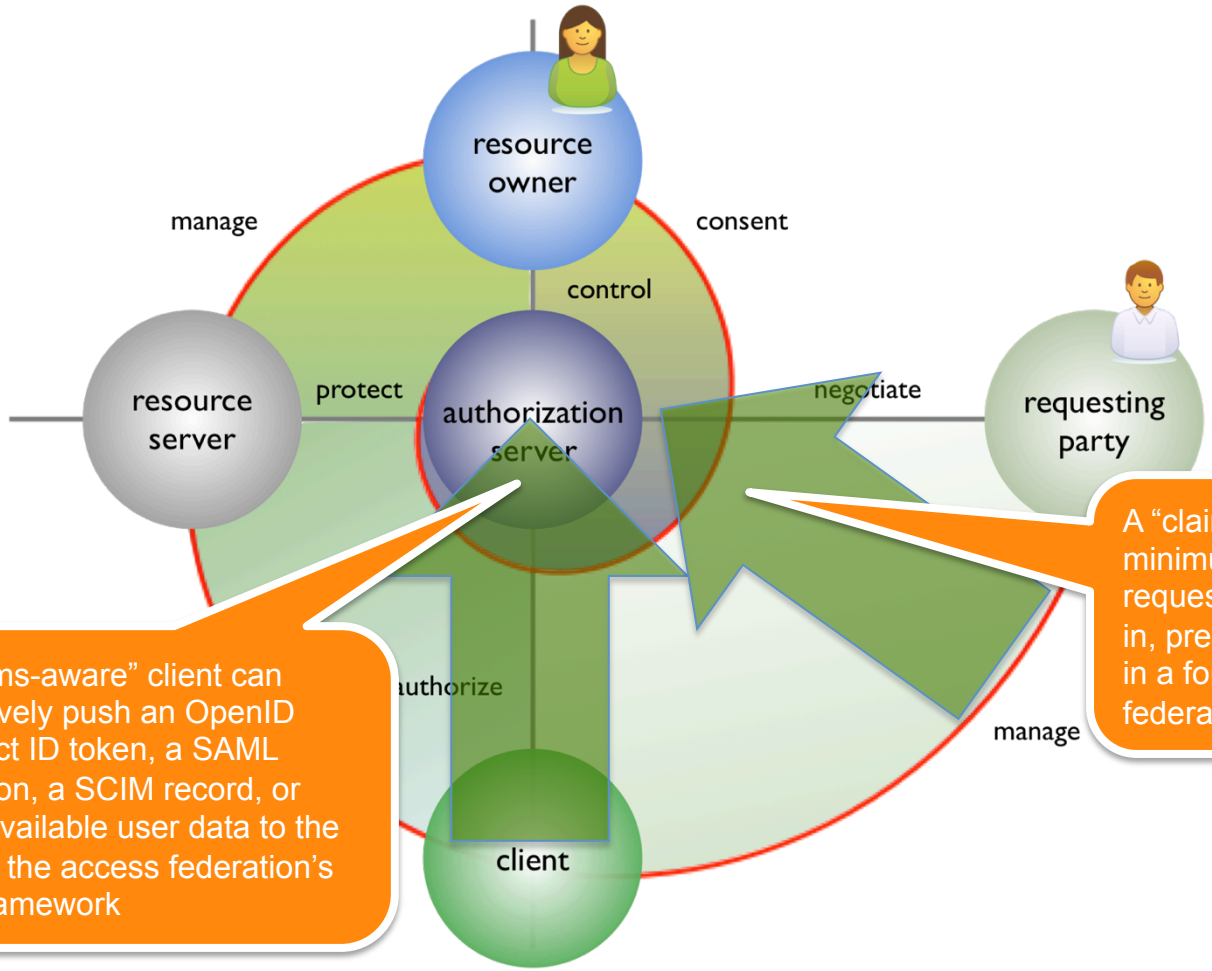
The AS exposes an UMA-standardized authorization API to the client



The AAT protects the API and binds the RqP, client, and AS

The client may be told: "need_info"

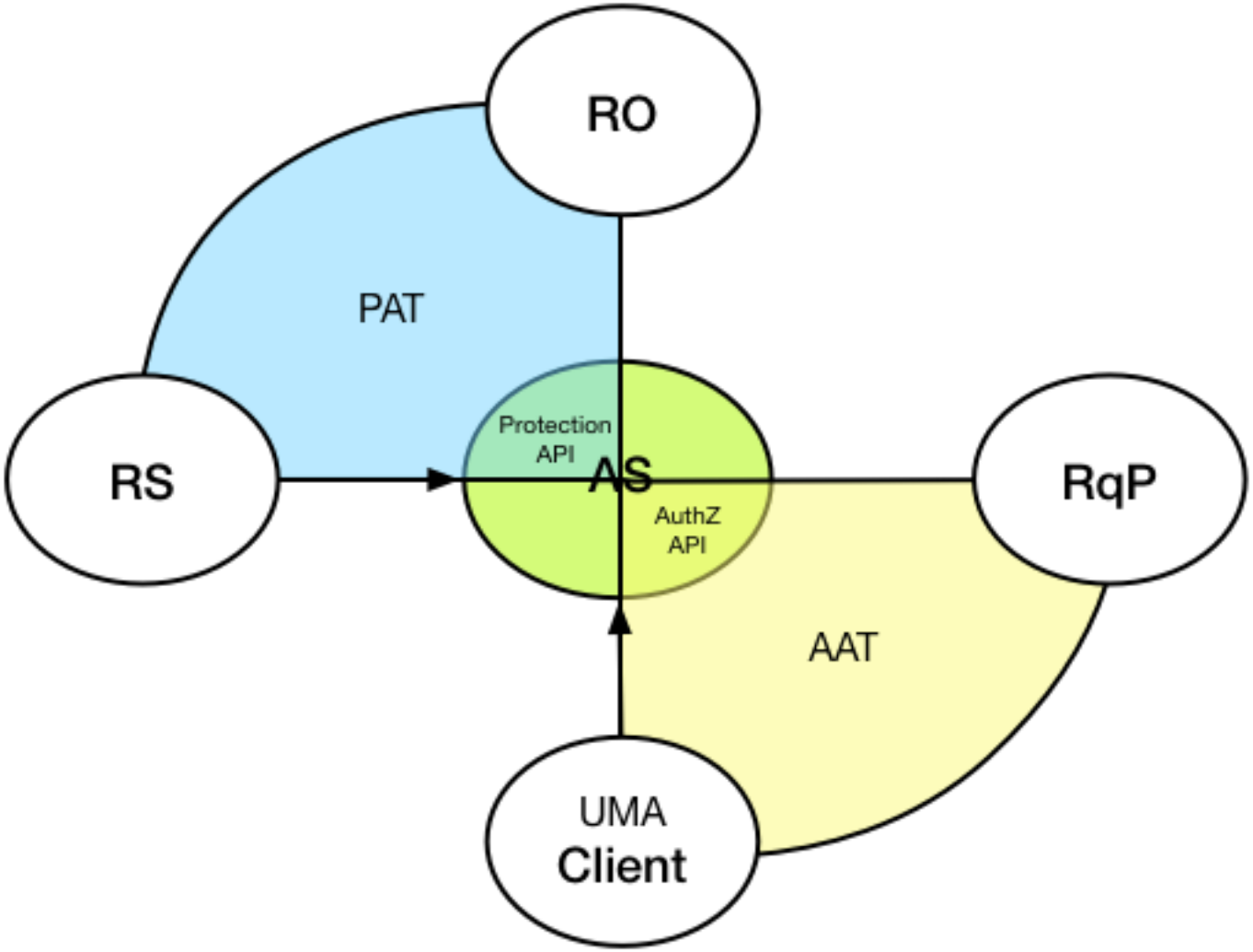
The AS can collect requesting party claims or otherwise elevate trust to assess policy



A "claims-aware" client can proactively push an OpenID Connect ID token, a SAML assertion, a SCIM record, or other available user data to the AS per the access federation's trust framework

A "claims-unaware" client can, at minimum, redirect the requesting party to the AS to log in, press an "I Agree" button, fill in a form, follow a NASCAR for federated login, etc.

The RO and RqP have opposite consent/ privacy relationships with the AS



How an individual user might experience setting sharing preferences

demo

Default burdens on apps

Resource server

- Gets client creds from AS
- Gets RO-specific access token (PAT) from AS
- Registers protected resources at AS as required (PUT)
- Registers permissions at AS for unauthorized client access attempts (POST)
- Introspects clients' RPTs at AS (GET)

Client

- Learns AS location and endpoints
- Gets client creds from AS
- Gets RqP-specific access token (AAT) from AS
- Requests authz data from AS (POST)
- Pushes user claims (optional) or redirects user to AS

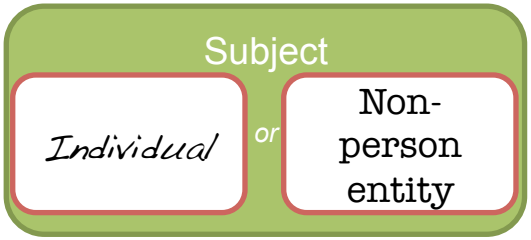
- All REST
- All JSON on both request and response sides
- Endpoints all TLS- and OAuth-protected

Profiling and extensibility enable efficiencies and non-HTTP bindings

- “Protection API extensibility profile” for AS-RS interactions
- “Authorization API extensibility profile” for AS-client interactions
- “Resource interface extensibility profile” for resource server-client interactions
  - E.g., to replace HTTP/TLS with CoAP/DTLS or co-locate entities
- RPT profiling
  - E.g., to enable disconnected token introspection or AS “hunt list”
- JSON extensibility all over the place
  - E.g., to enable general experimentation and escape hatches
- Claim token format profiling
  - E.g., to enable a variety of deployment-specific trust frameworks

UMA Binding Obligations

- Distributed authorization across domains? Scary!
- This “legal” spec enables parties operating and using software entities (and devices) to distribute rights and obligations fairly in *access federation* trust frameworks



- Authorizing Party
- Requesting Party
- Requesting Party Agent
- Resource Server Operator
- Authorization Server Operator
- Client Operator

Important state changes when new pairwise obligations tend to appear:

- Token issuance
- Token status checks
- Permission registration
- Claims gathering
- Access requests
- Successful access

Agenda

- UMA's design center, progress, and status
- A quick "UMA 101" primer
- Measuring UMA against ACE use cases
- Discussion and next steps

Strong architectural matches

- ☑ Owner grants different resource access rights to different parties
  - U1.1, U2.3, U.3.2, (U3.3)
- ☑ Owner grants different access rights for different resources on a device (including read, write, admin)
  - U1.3, U4.4, U5.2
- ☑ Owner not always present at time of access
  - U1.6, U5.5
- ☑ Owner grants temporary access permissions to a party
  - U1.7
- ☑ Owner applies verifiable context-based conditions to authorizations
  - U2.4, U4.5, U6.3
- ☑ Owner grants temporary access permissions to a party
  - U1.7
- ☑ Owner preconfigures access rights to specific data
  - U3.1, U6.3
- ☑ Owner adds a new device under protection
  - U4.1
- ☑ Owner puts a previously owned device under protection
  - U4.2
- ☑ Owner removes a device from protection
  - U4.3
- ☑ Owner preconfigures access rights to specific data
  - U3.1
- ☑ Owner revokes permissions
  - U4.6
- ☑ Owner grants access only to authentic, authorized clients
  - U7.1, U7.2

Potential profiling/extension opportunities

- Constrained device might not always be able to reach the Internet
  - U1.9, U5.4, U6.5, U7.3
  - Or proxy/gateway approach
- Impossible or inefficient to contact all affected devices directly when policies are updated
  - U5.6

Potential user experience and system configuration opportunities

- Spontaneous device provisioning
  - U2.1
- Spontaneous/dynamic policy changes
  - U2.2, U6.1
- Secure-by-default policies
  - U2.6, U3.6
- Easy-to-edit policies
  - U2.7, U2.9, U2.10, U3.6, U6.2

Apparent OOS challenges

- Sensor data integrity
  - U1.2
- Sensor data confidentiality
  - U1.2
- Client-RS messages forwarded over multiple hops?
  - U1.8, U5.7
- Smart home devices communicate with different control devices
  - U2.5
- Owner prevents eavesdroppers on home network
  - U2.8
- Prevent (all) DoS
  - U3.7
- High security to prevent owner fatalities
  - U3.8
- Multicast protocol?
  - U4.8
- Physical device security
  - U5.1
- Wired and wireless
  - U7.4
- Mitigate risk of financial damage
  - U7.5
  - UMA Binding Obligations spec helps do this

Agenda

- UMA's design center, progress, and status
- A quick "UMA 101" primer
- Measuring UMA against ACE use cases
- Discussion and next steps

Questions? Thank you!

Eve Maler, chair
@UMAWG | @xmlgrrl
13 January 2015
tinyurl.com/umawg