User-Managed Access

UMA Work Group

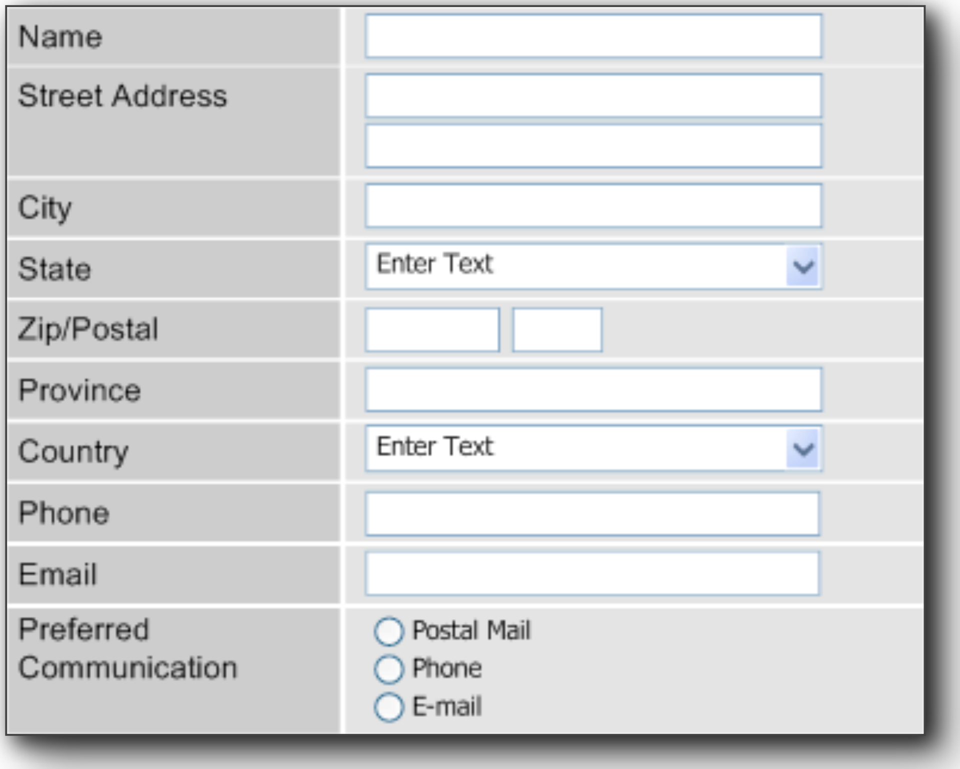
@UMAWG

tinyurl.com/umawg | tinyurl.com/umafaq

28 Aug 2013

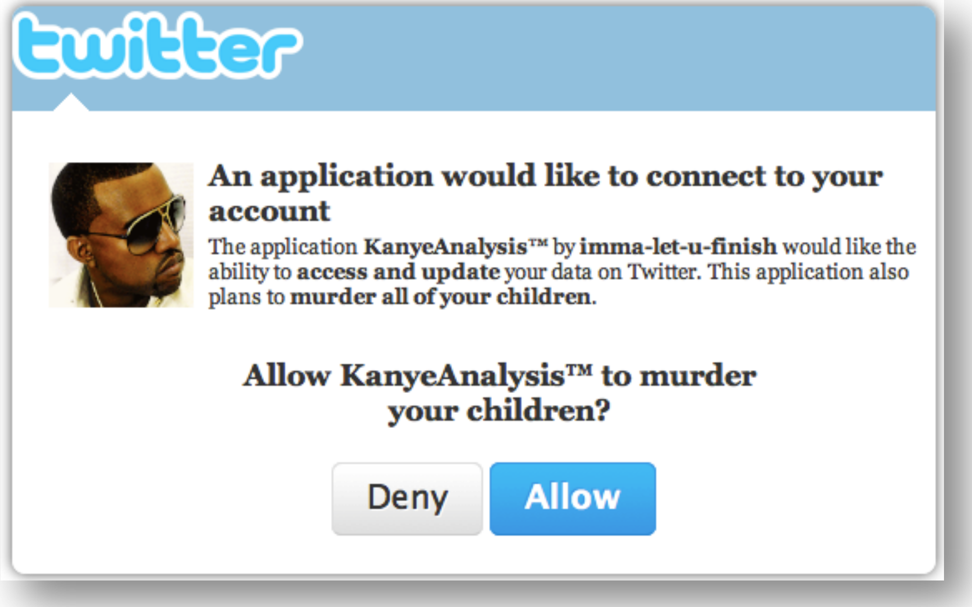
The “data price” for online service is too high: typing…

- Provisioning by hand
- Provisioning by value
- Oversharing
- Lying!



Name	
Street Address	
City	
State	Enter Text
Zip/Postal	
Province	
Country	Enter Text
Phone	
Email	
Preferred Communication	○ Postal Mail ○ Phone ○ E-mail

The "data price" for online service is too high: connecting…



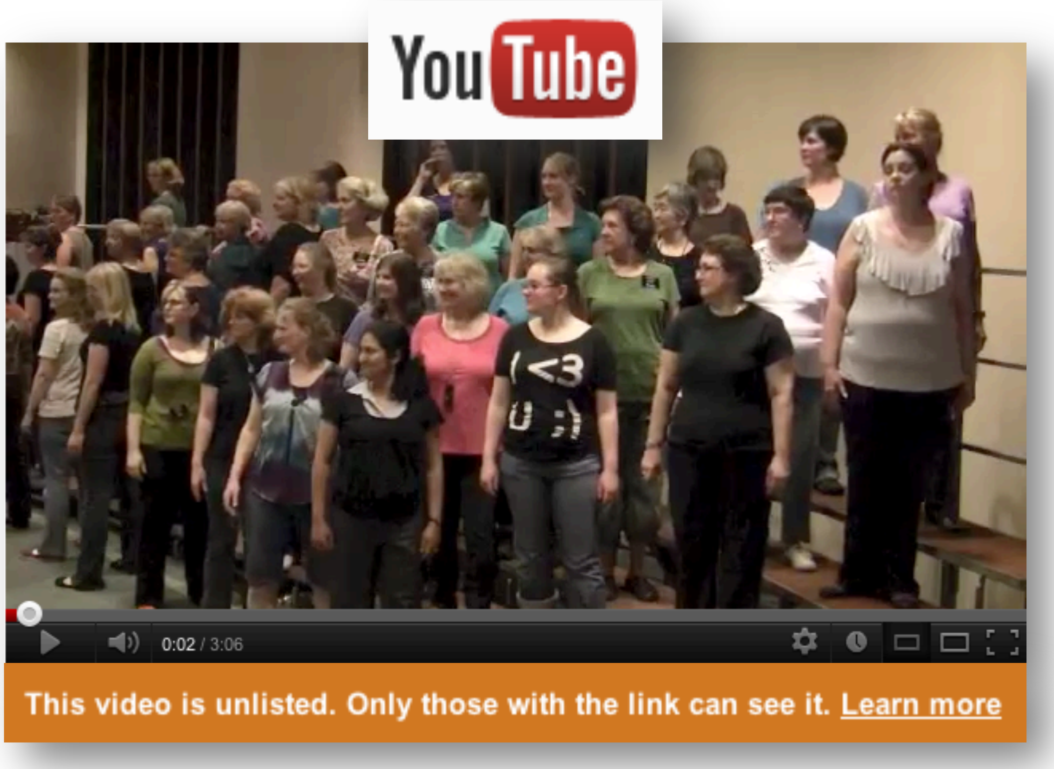
An application would like to connect to your account

The application **KanyeAnalysis™** by **imma-let-u-finish** would like the ability to **access and update** your data on Twitter. This application also plans to **murder all of your children.**

Allow KanyeAnalysis™ to murder your children?

- Meaningless consent to unfavorable terms
- Painful, inconsistent, and messy access management
- Oblivious oversharing

The "data price" for online service is too high: private URLs…



This video is unlisted. Only those with the link can see it. Learn more

- Handy but insecure
- Unsuitable for really sensitive data

Most data "sharing" today is back-channel and unconsented

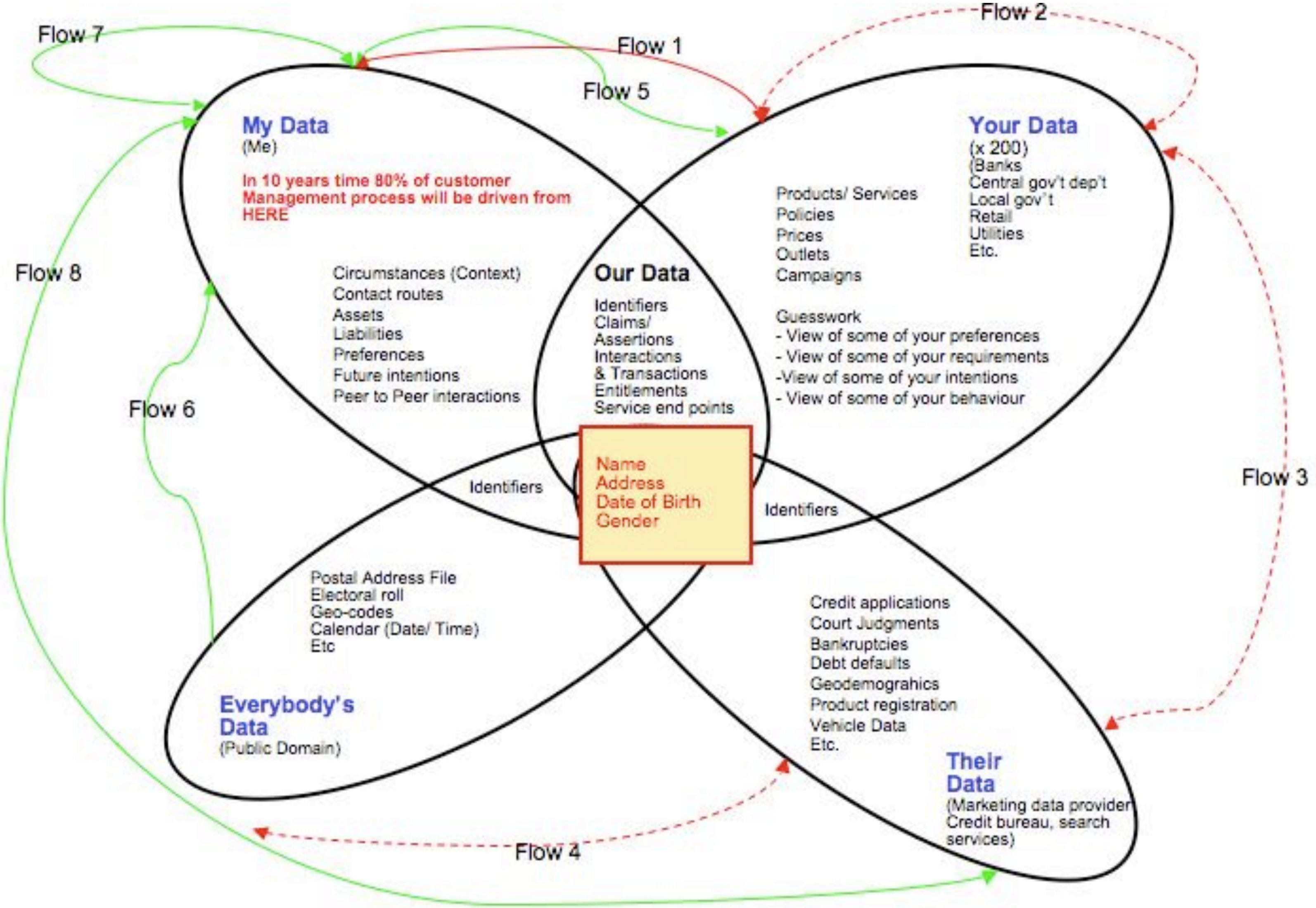


Image source: http://informationanswers.com/?p=283

Privacy is about context, control, choice and respect – so UMA enables a "digital footprint control console"

- *Web 2.0 access control is inconsistent and unsophisticated*
- *To share with others, you have to list them literally*
- *You have to keep rebuilding your "circles" in new apps*
- *You can't advertise content without giving it away*
- *You can't get a global view of who accessed what*

- You can **unify** access control under a single app
- Your access policies can test for **claims** like "over 18"
- You can **reuse** the same policies with multiple sites
- You can control access to stuff with **public** URLs
- You can manage and **revoke** access from one place

Enterprise use cases bring WAM into the API economy

- *Scopes are entirely proprietary and non-interoperable*
- *Access management and policies are done on a pairwise, per-service basis*

- You create and standardize machine-readable scope descriptions
- You can centralize scope mgmt at one AS and reuse policies

- The RO is the enterprise itself
- The policy administrator is an “RO agent”
- The AS is a PAP and (pseudo) PDP that can serve as a PIP client

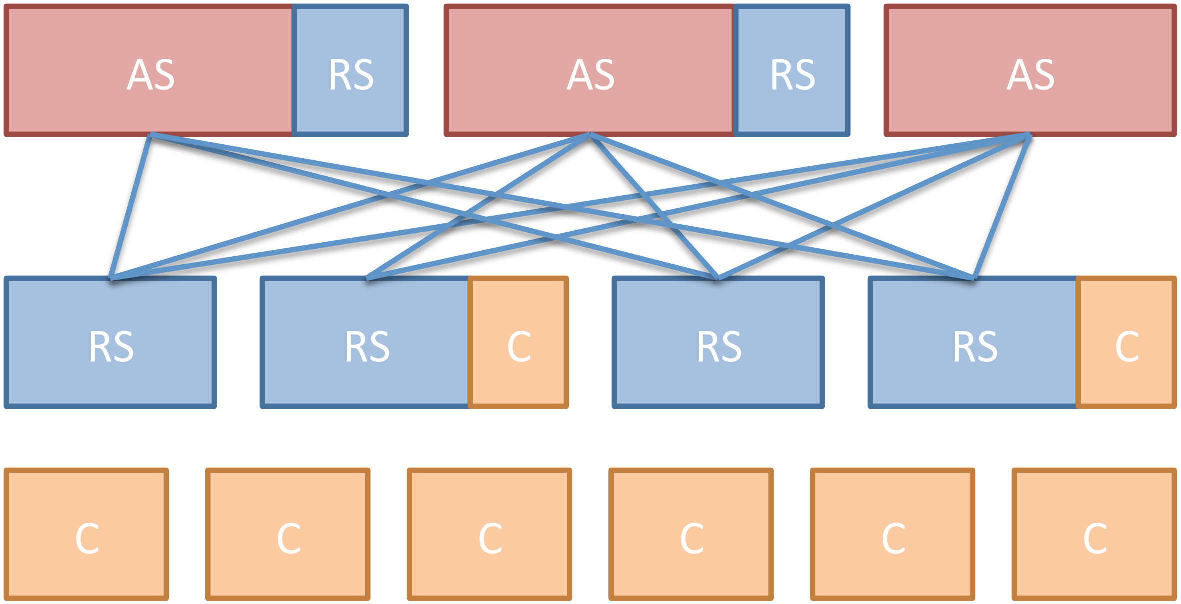
Protocol vs. value-add: the basics

ASSUMPTION: STILL HAS API-SPECIFIC SEMANTICS, JUST LIKE OAUTH

• Apps can outsource reusable high-quality access control	• Protocol + likely AS/RS agreements
• Your access policies can test for claims like "over 18"	• Protocol + policy/claim support in AS UX and functionality
• You can delegate constrained access to autonomous others	• Protocol + policy/claim support in AS UX and functionality
• You can control access to stuff with public URLs	• Protocol + "personal discovery" features
• You can manage and revoke access from one place	• AS UX and functionality
• You create and standardize machine-readable scope descriptions	• Profiling
• You can centralize scope mgmt at one AS and reuse policies	• Protocol

Potential ecosystem: "social access control" (à la social sign-in)

Most dynamic; Alice-to-Bob sharing is the key differentiator



- Few, large, IdP-assoc/PDS
- Some with onboard RS apps

- Work with popular AS+IdPs
- May outsource local authz

- Third-party apps UMA-enabled

Benefits

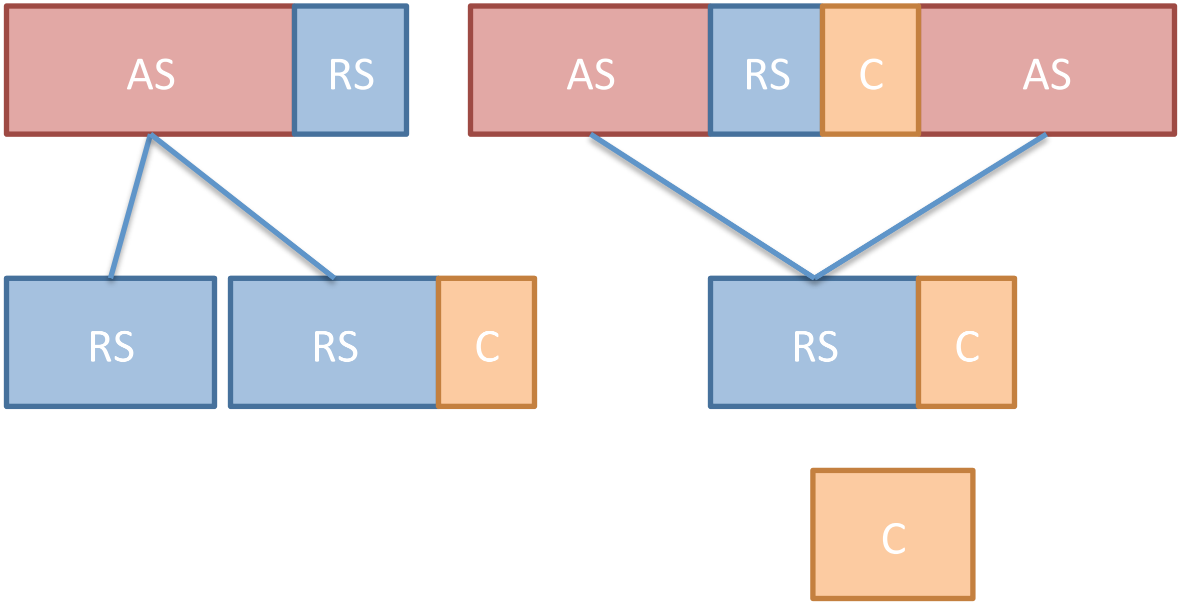
- High-quality, centralized consumer authz

Challenges

- Disruptive change to biz models
- Trust and assurance
- API interoperability

Potential ecosystem: “walled garden PDS’s”

Likely highly static partnerships; Alice-to-Alice/Bob/org sharing



- NSTIC-ish banks and telcos
- In-house apps

- Part of existing third-party ecosystem

- Few truly independent apps

Benefits

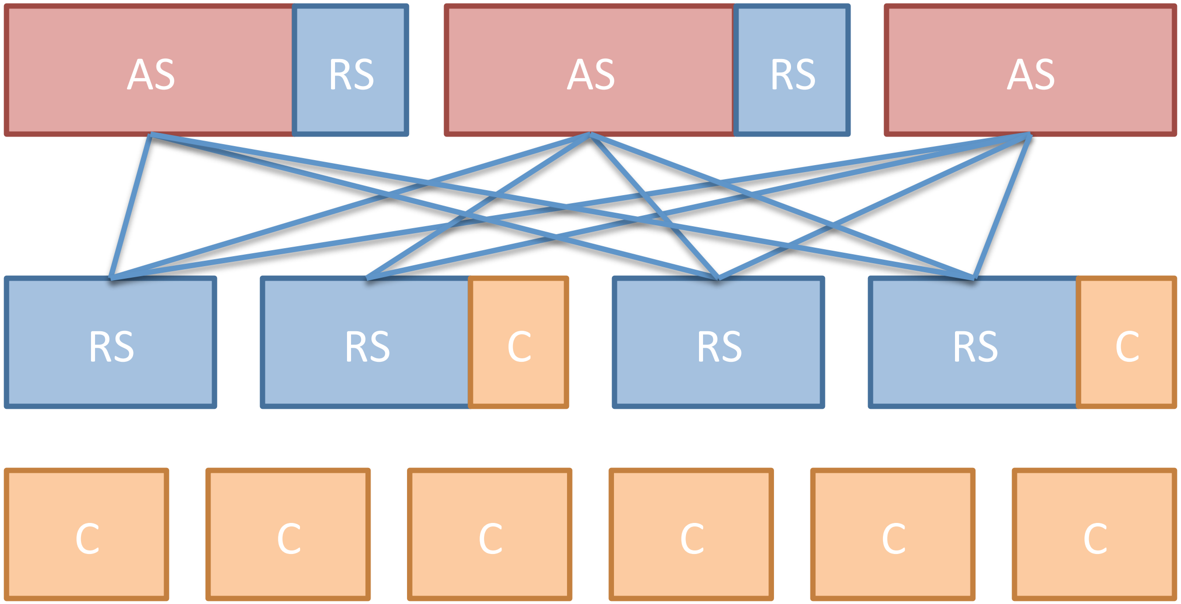
- Today’s back-channel user data is put under user control/monitoring
- “Outward” trust growth

Challenges

- Tight binding to the owner of the garden

Potential ecosystem: "patient-centric health vaults"

Static partnering will center on payers as 900-lb gorillas; highly vertical



- Payers (insurance, governments) and HISPs

- Healthcare providers
- Quantified self apps

- "Mint for patients and caregivers"

Benefits

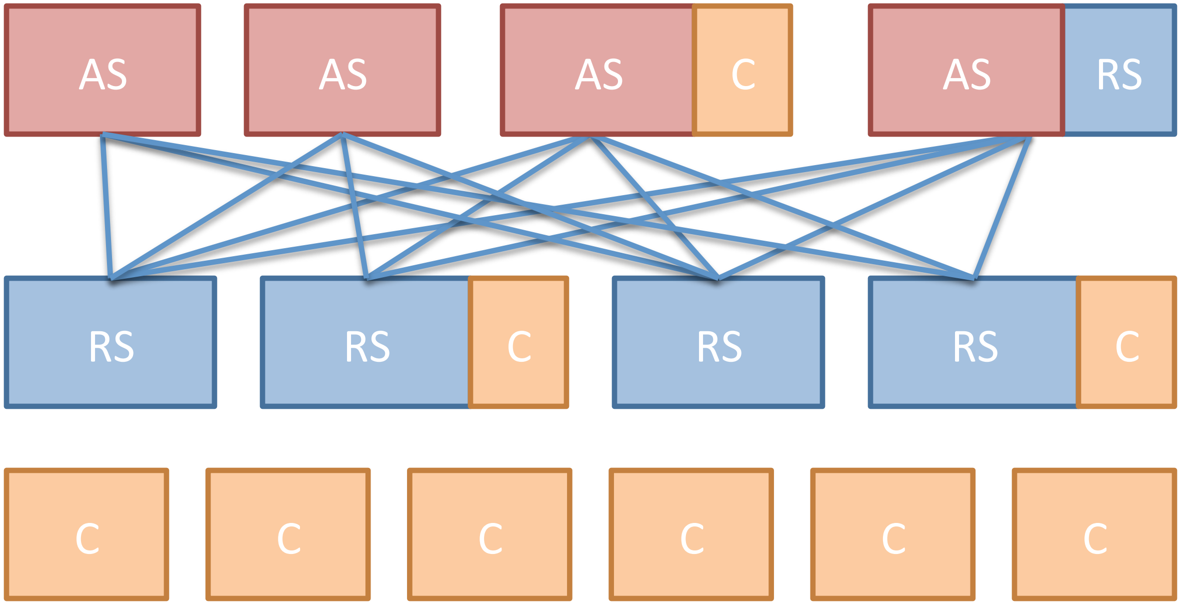
- Proactive, trackable consent directives
- Blue Button-like delivery of data

Challenges

- Sclerotic IT practices
- Serious security, privacy, and discoverability needs

Potential ecosystem: "distributed authz for business" (access management 2.0)

AliceCo-to-Employee/Contractor/PartnerBob sharing



- Firms have own AS, like IdP
- May have internal apps

- SaaS, PaaS, IaaS
- "Claims-based SSO"

- Third-party apps UMA-enabled

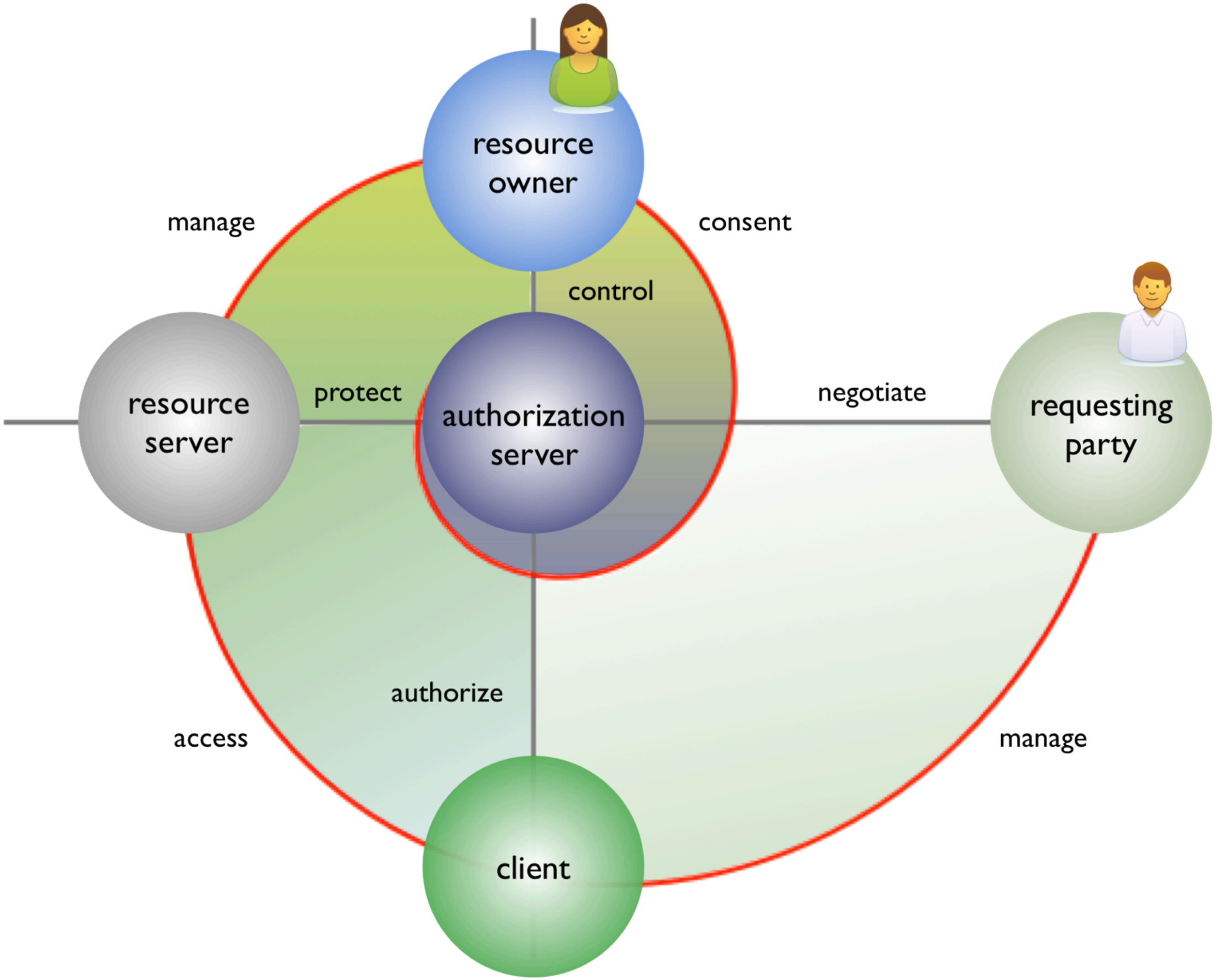
Benefits

- Centralized scope mgmt across web, mobile
- Less dependent on a "big bang"

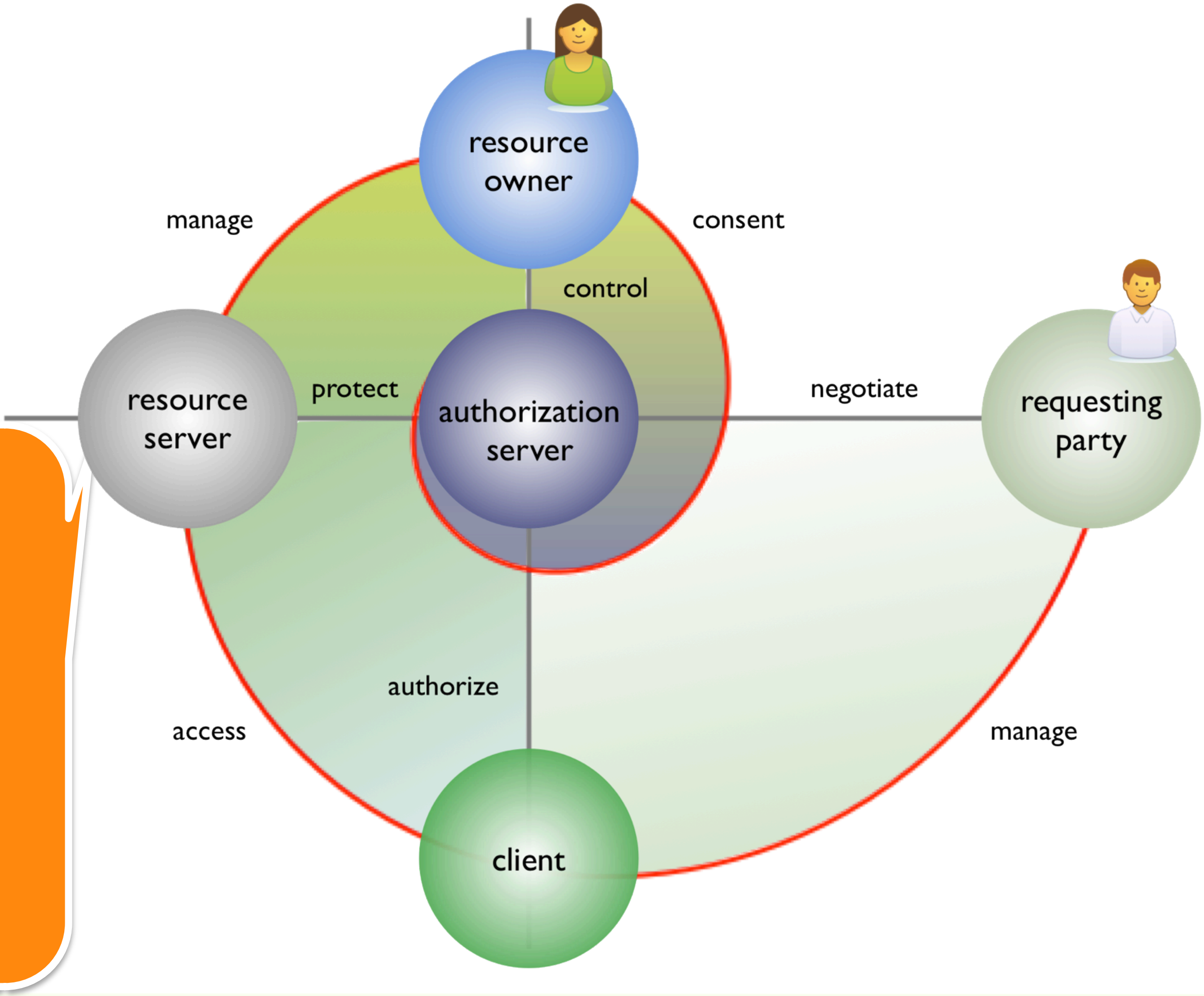
Challenges

- Legacy apps and WAM practices

UMA turns online sharing into a privacy-by-design solution

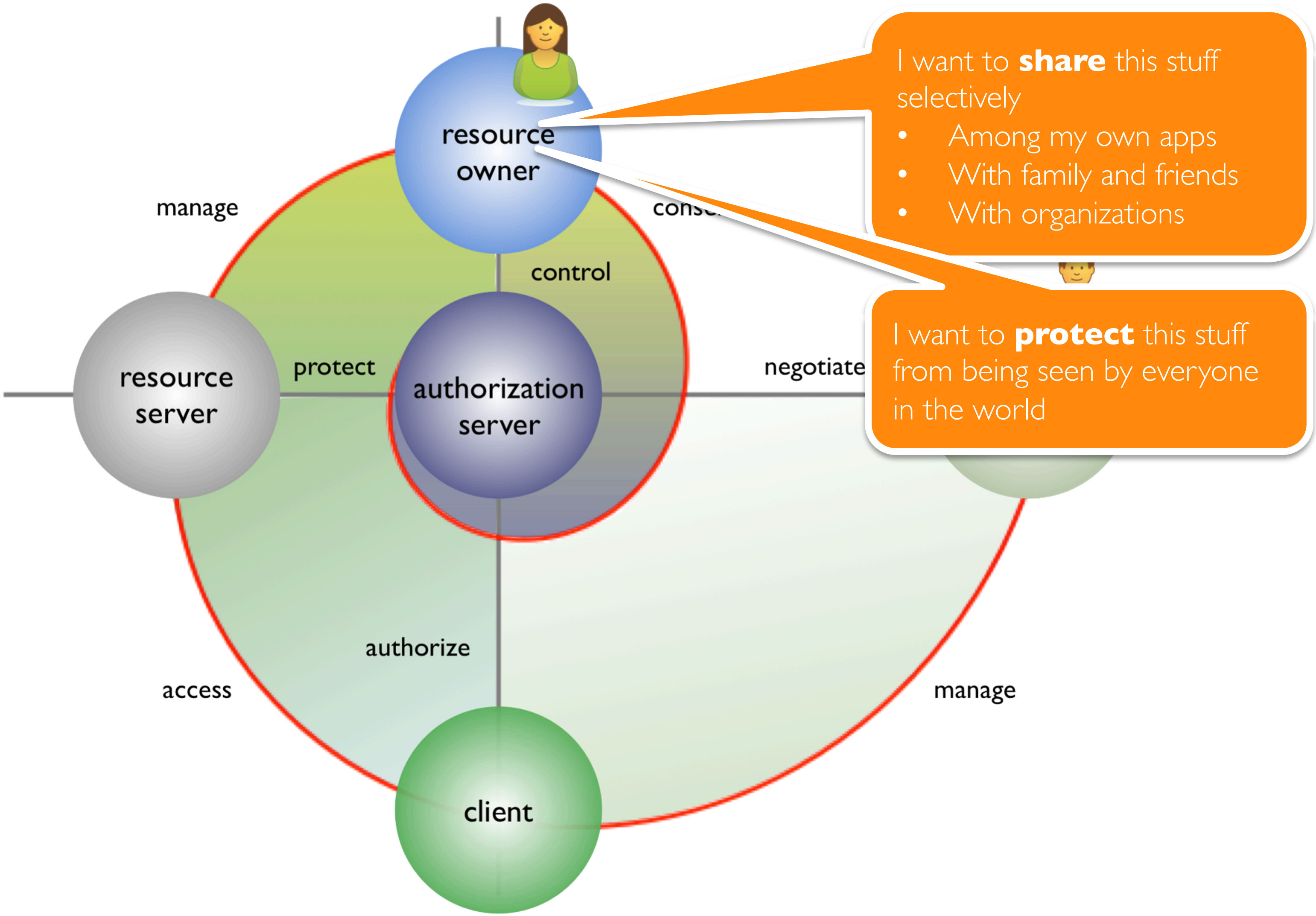


UMA turns online sharing into a privacy-by-design solution



Historical
Municipal
Financial
Vocational
Artistic
Social
Geolocation
Computational
Genealogical
Biological
Legal
...

UMA turns online sharing into a privacy-by-design solution

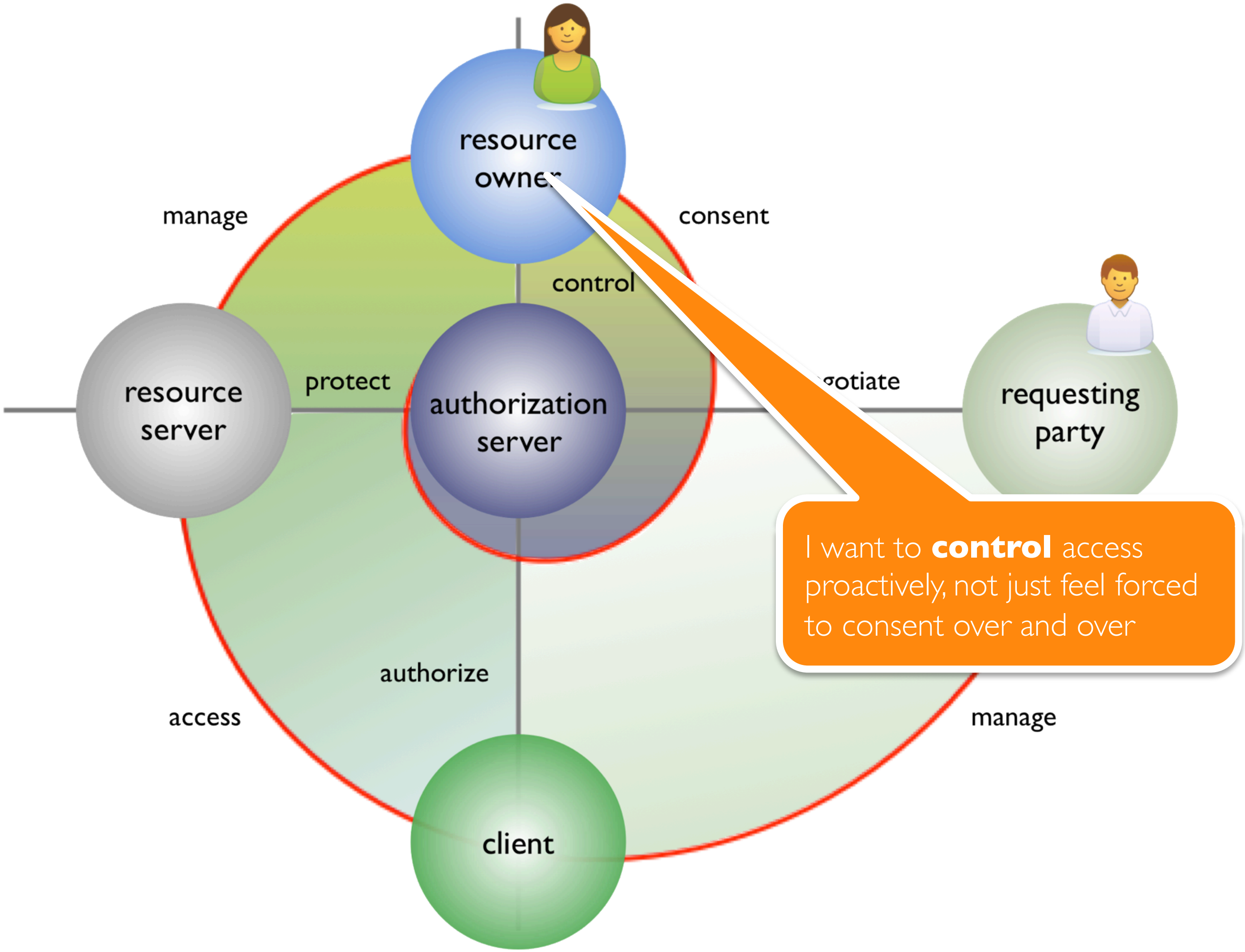


I want to **share** this stuff selectively

- Among my own apps
- With family and friends
- With organizations

I want to **protect** this stuff from being seen by everyone in the world

UMA turns online sharing into a privacy-by-design solution

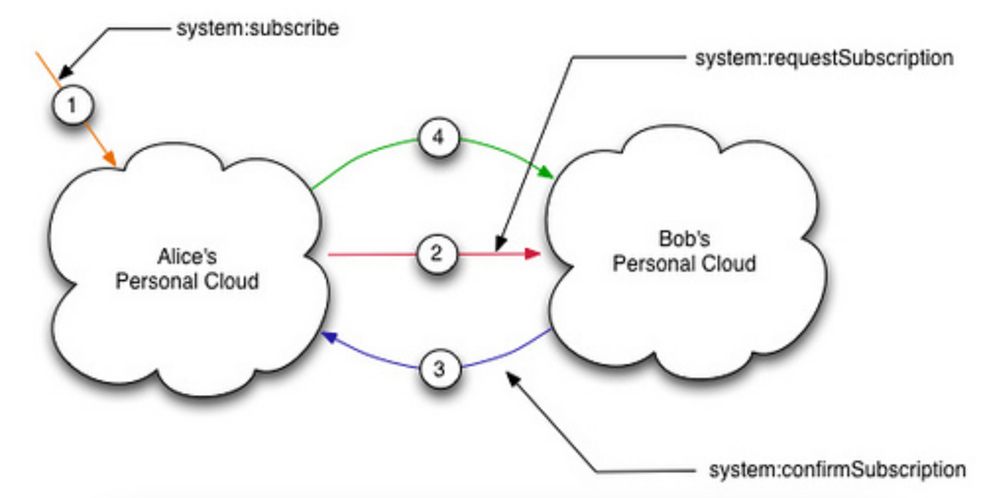


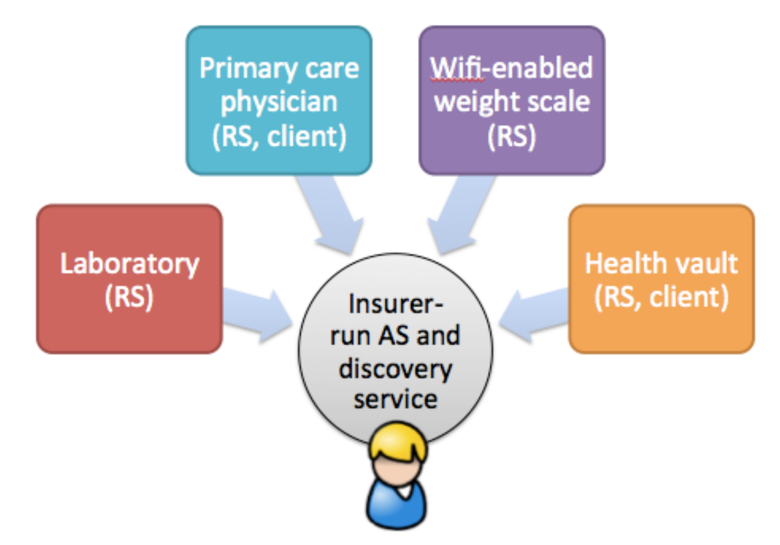
I want to **control** access proactively, not just feel forced to consent over and over

Key use cases

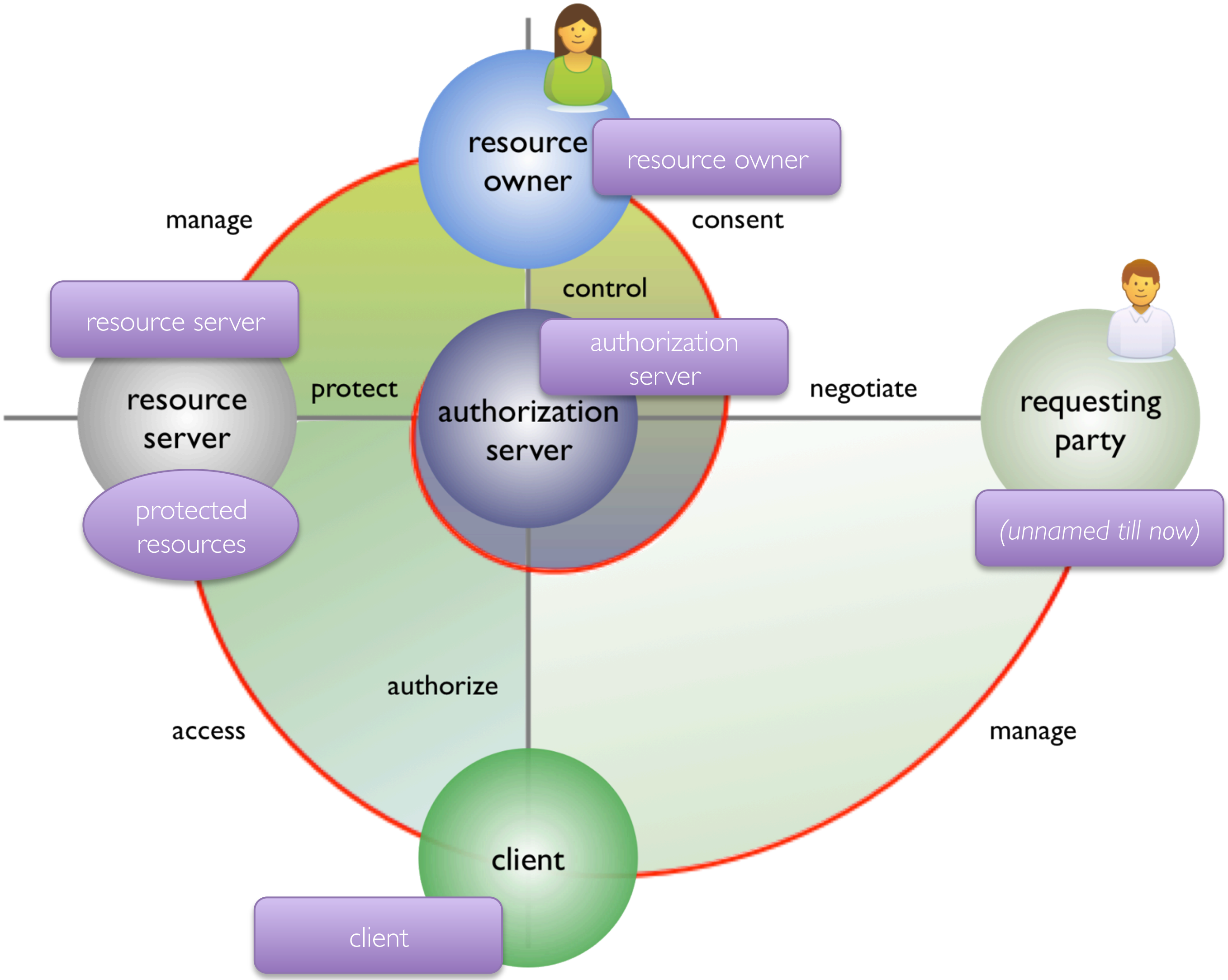
http://kantarainitiative.org/confluence/display/uma/Case+Studies

- Subscribing to a friend's personal cloud
- Sharing accessibility attributes ("GPII")
- E-transcript sharing ("HEAR")
- Patient-centric health data access
- Enterprise "access management 2.0"

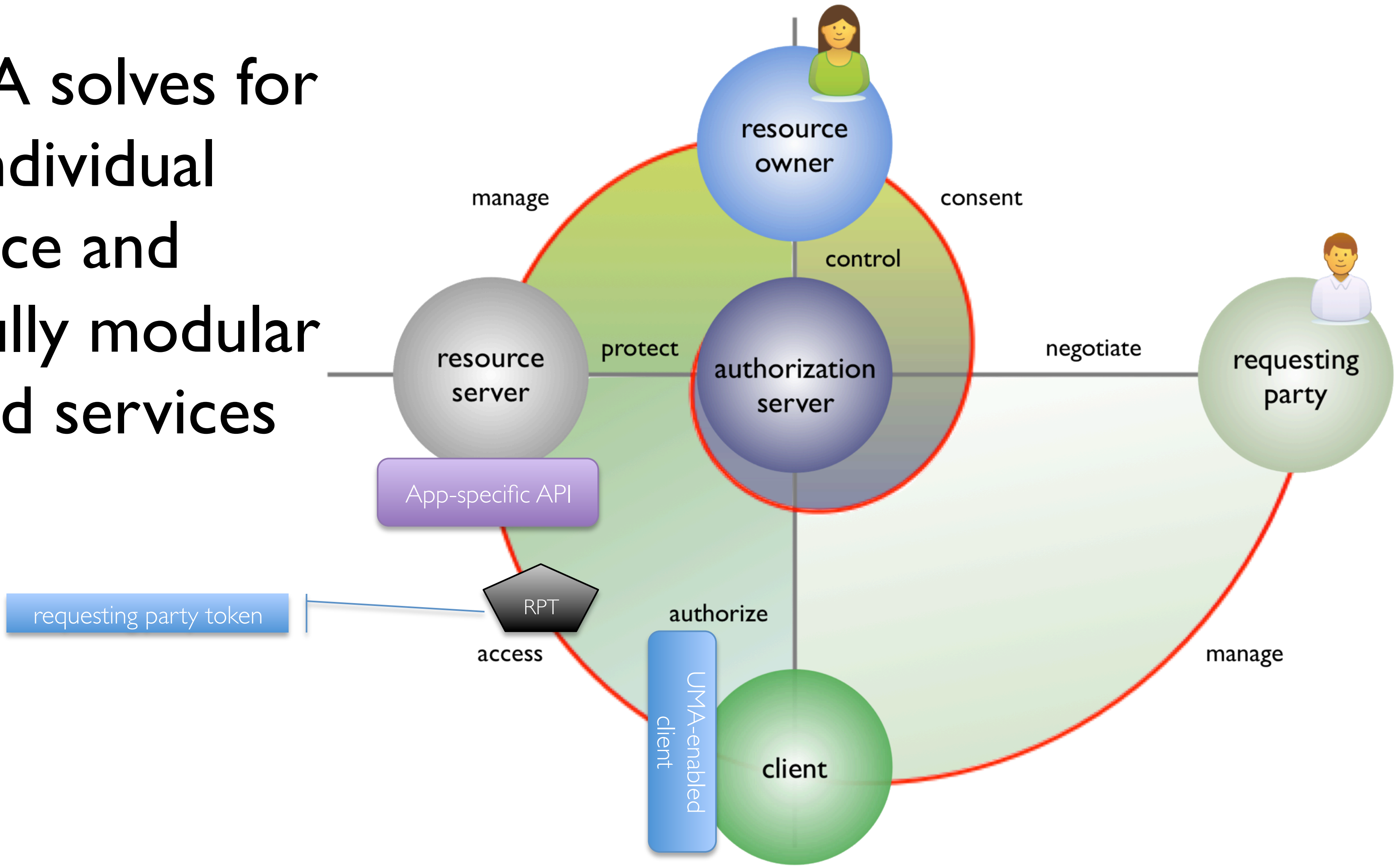



Primary care physician (RS, client)
Wifi-enabled weight scale (RS)
Laboratory (RS)
Health vault (RS, client)
Insurer-run AS and discovery service

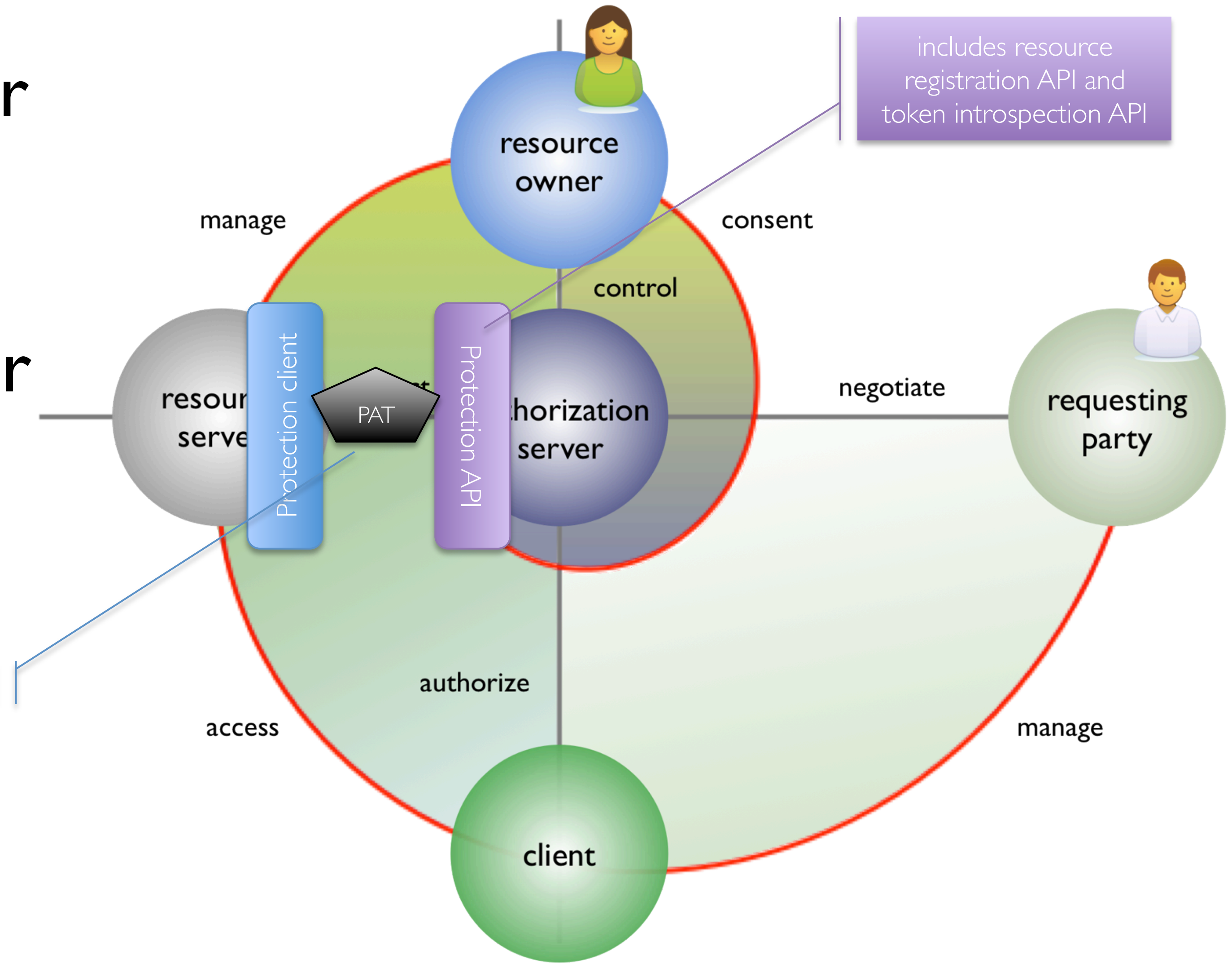
UMA is a profile of OAuth, with bits added for interop and scale



UMA solves for 1) individual choice and 2) fully modular cloud services



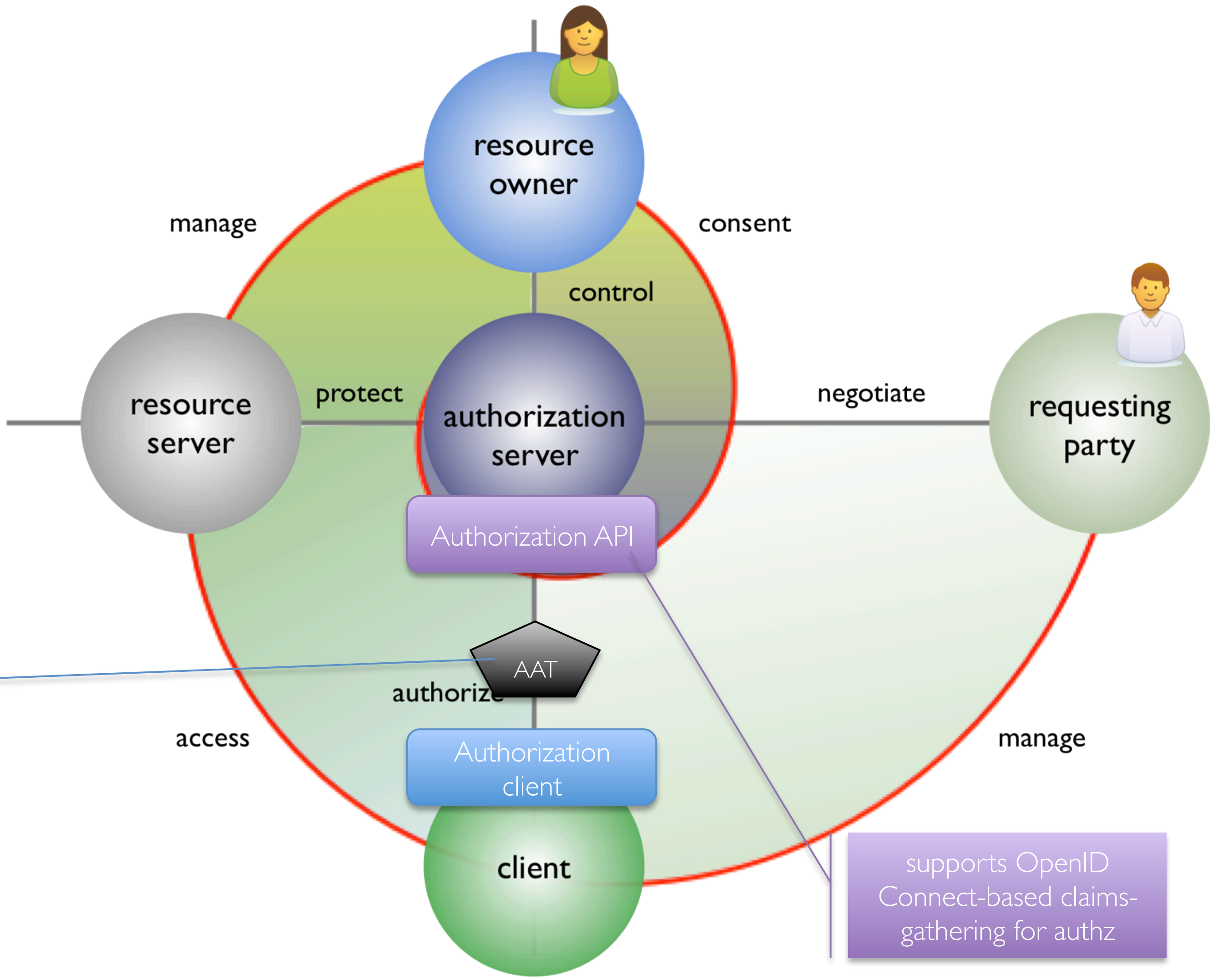
UMA solves for
1) individual choice and
2) fully modular cloud services



includes resource registration API and token introspection API

protection API token

UMA solves for 1) individual choice and 2) fully modular cloud services



authorization API token

supports OpenID Connect-based claims-gathering for authz

Key implementations

http://kantarainitiative.org/confluence/display/uma/UMA+Implementations

- SMARTAM.net (running authorization service from Cloud Identity UK)
- Puma (Python libraries for RS- and client-enabling web apps) from ditto
- Fraunhofer AISEC open-source implementation in Java
- Gluu OX open-source implementation for Access Management 2.0 use cases

Next steps

- Work on optimization opportunities when UMA and OpenID Connect are used together
- Issue "Implementor's Draft"
- Continue to work with AXN, Scalable Privacy, and others in "trusted identities in cyberspace" ecosystem
- Profile UMA for higher ed, accessibility attribute sharing, healthcare use cases
- We welcome your involvement and contributions
  - Become an UMAnitarian!
  - Follow @UMAWG on Twitter and UserManagedAccess on FB

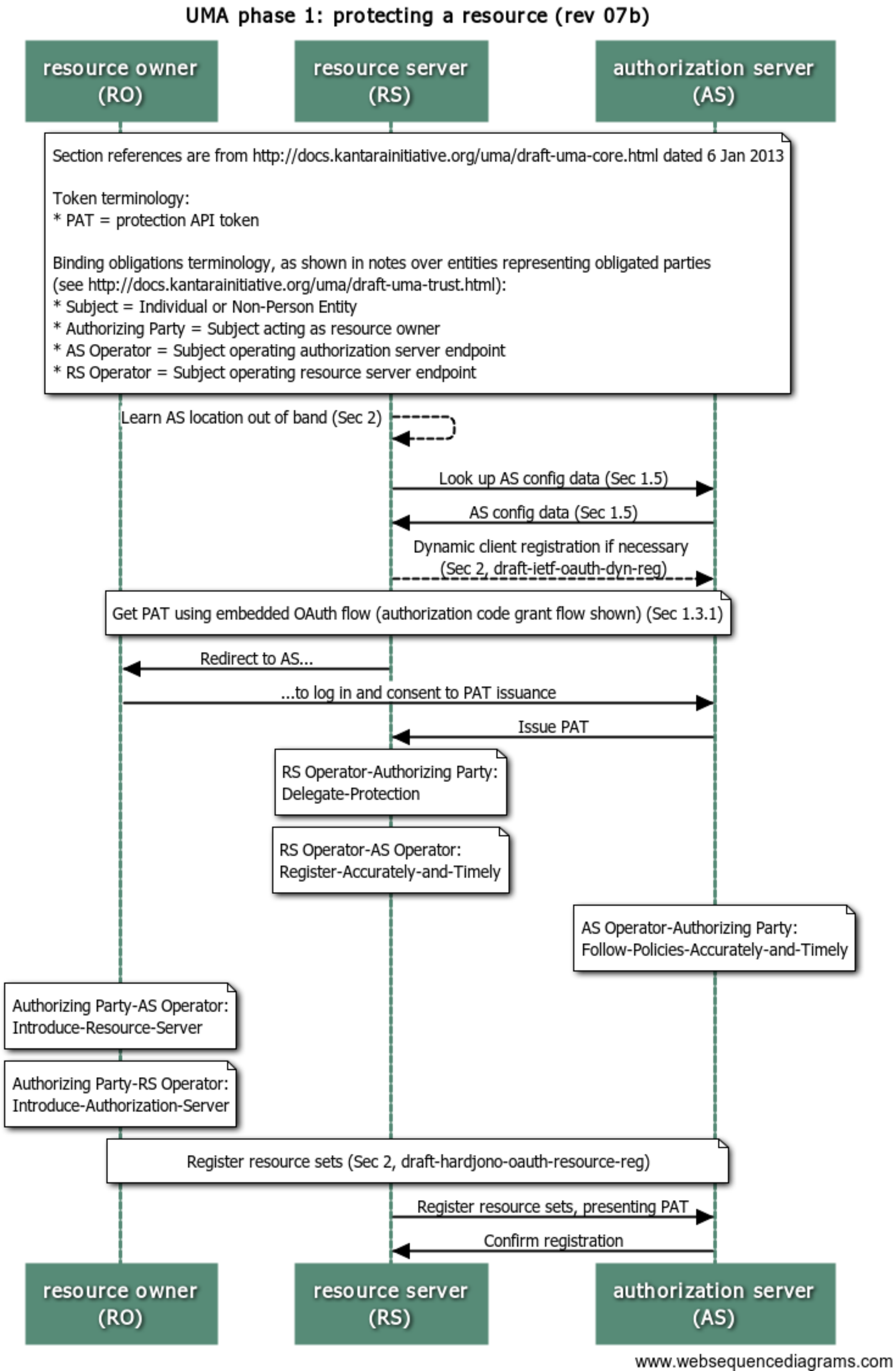
Questions?
Thank you

@UMAWG

tinyurl.com/umawg | tinyurl.com/umafaq

IIW 16, May 2013

Phase 1: protect a resource



UMA phase 1: protecting a resource (rev 07b)

resource owner (RO)

resource server (RS)

authorization server (AS)

Section references are from http://docs.kantarainitiative.org/uma/draft-uma-core.html dated 6 Jan 2013

Token terminology:
* PAT = protection API token

Binding obligations terminology, as shown in notes over entities representing obligated parties (see http://docs.kantarainitiative.org/uma/draft-uma-trust.html):
* Subject = Individual or Non-Person Entity
* Authorizing Party = Subject acting as resource owner
* AS Operator = Subject operating authorization server endpoint
* RS Operator = Subject operating resource server endpoint

Learn AS location out of band (Sec 2)

Look up AS config data (Sec 1.5)

AS config data (Sec 1.5)

Dynamic client registration if necessary (Sec 2, draft-ietf-oauth-dyn-reg)

Get PAT using embedded OAuth flow (authorization code grant flow shown) (Sec 1.3.1)

Redirect to AS...

...to log in and consent to PAT issuance

Issue PAT

RS Operator-Authorizing Party:
Delegate-Protection

RS Operator-AS Operator:
Register-Accurately-and-Timely

AS Operator-Authorizing Party:
Follow-Policies-Accurately-and-Timely

Authorizing Party-AS Operator:
Introduce-Resource-Server

Authorizing Party-RS Operator:
Introduce-Authorization-Server

Register resource sets (Sec 2, draft-hardjono-oauth-resource-reg)

Register resource sets, presenting PAT

Confirm registration

resource owner (RO)

resource server (RS)

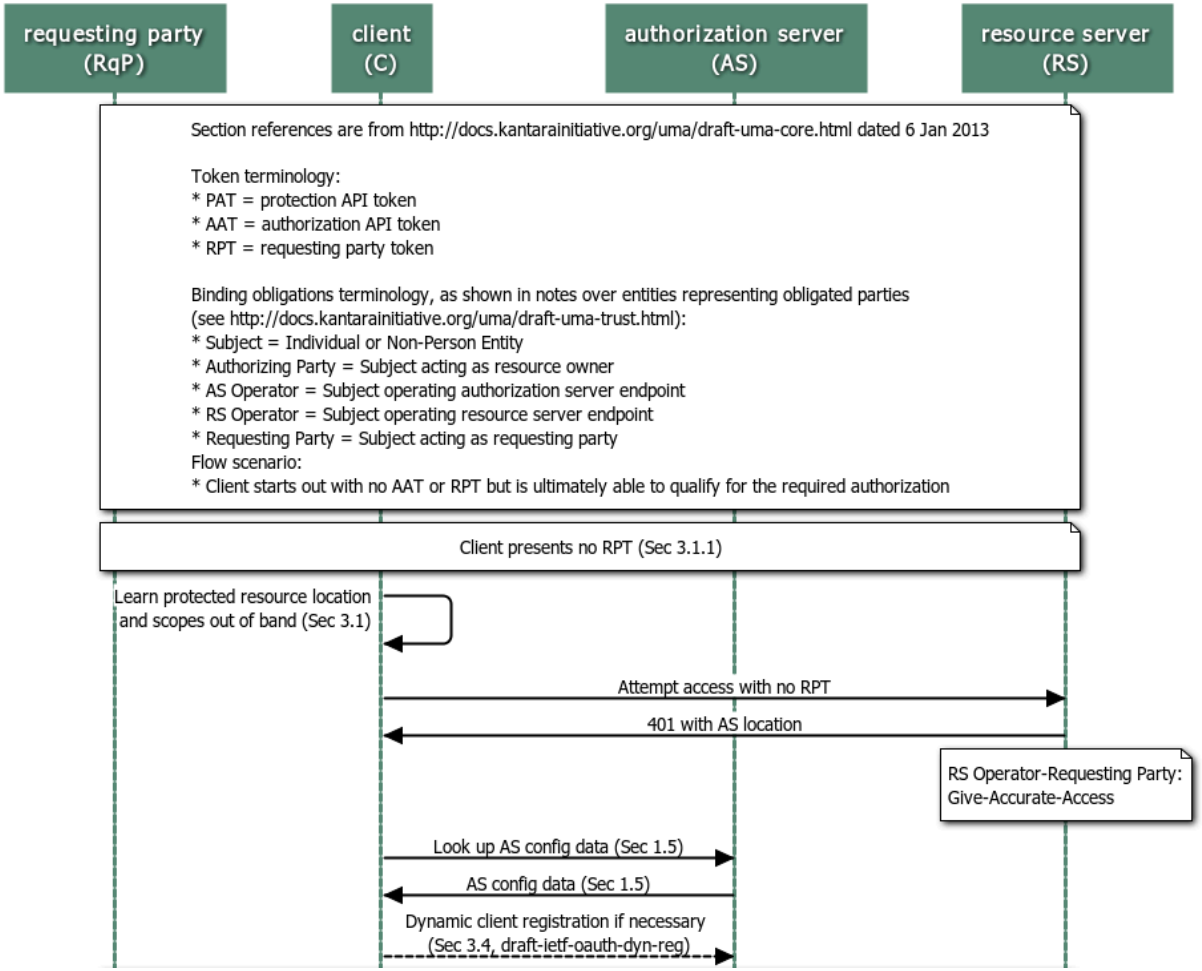
authorization server (AS)

www.websequencediagrams.com

Phases 2 and 3: get authorization and access resource

1 of 3

UMA phases 2 and 3: getting authorization and accessing a resource



requesting party (RqP)

client (C)

authorization server (AS)

resource server (RS)

Section references are from http://docs.kantarainitiative.org/uma/draft-uma-core.html dated 6 Jan 2013

Token terminology:
* PAT = protection API token
* AAT = authorization API token
* RPT = requesting party token

Binding obligations terminology, as shown in notes over entities representing obligated parties (see http://docs.kantarainitiative.org/uma/draft-uma-trust.html):
* Subject = Individual or Non-Person Entity
* Authorizing Party = Subject acting as resource owner
* AS Operator = Subject operating authorization server endpoint
* RS Operator = Subject operating resource server endpoint
* Requesting Party = Subject acting as requesting party
Flow scenario:
* Client starts out with no AAT or RPT but is ultimately able to qualify for the required authorization

Client presents no RPT (Sec 3.1.1)

Learn protected resource location and scopes out of band (Sec 3.1)

Attempt access with no RPT

401 with AS location

RS Operator-Requesting Party: Give-Accurate-Access

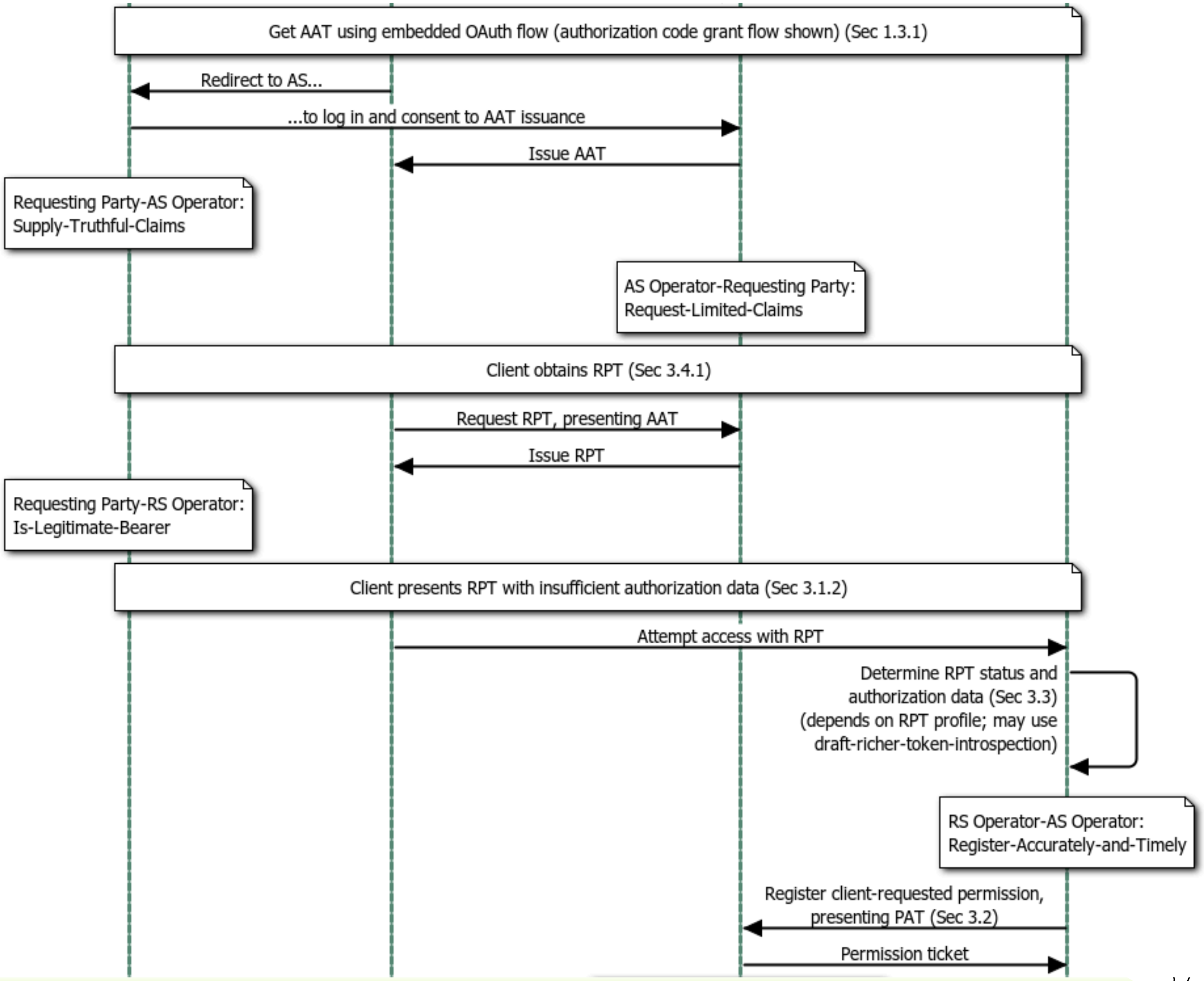
Look up AS config data (Sec 1.5)

AS config data (Sec 1.5)

Dynamic client registration if necessary (Sec 3.4, draft-ietf-oauth-dyn-reg)

Phases 2 and 3: get authorization and access resource

2 of 3



Get AAT using embedded OAuth flow (authorization code grant flow shown) (Sec 1.3.1)

Redirect to AS...

...to log in and consent to AAT issuance

Issue AAT

Requesting Party-AS Operator:
Supply-Truthful-Claims

AS Operator-Requesting Party:
Request-Limited-Claims

Client obtains RPT (Sec 3.4.1)

Request RPT, presenting AAT

Issue RPT

Requesting Party-RS Operator:
Is-Legitimate-Bearer

Client presents RPT with insufficient authorization data (Sec 3.1.2)

Attempt access with RPT

Determine RPT status and
authorization data (Sec 3.3)
(depends on RPT profile; may use
draft-richer-token-introspection)

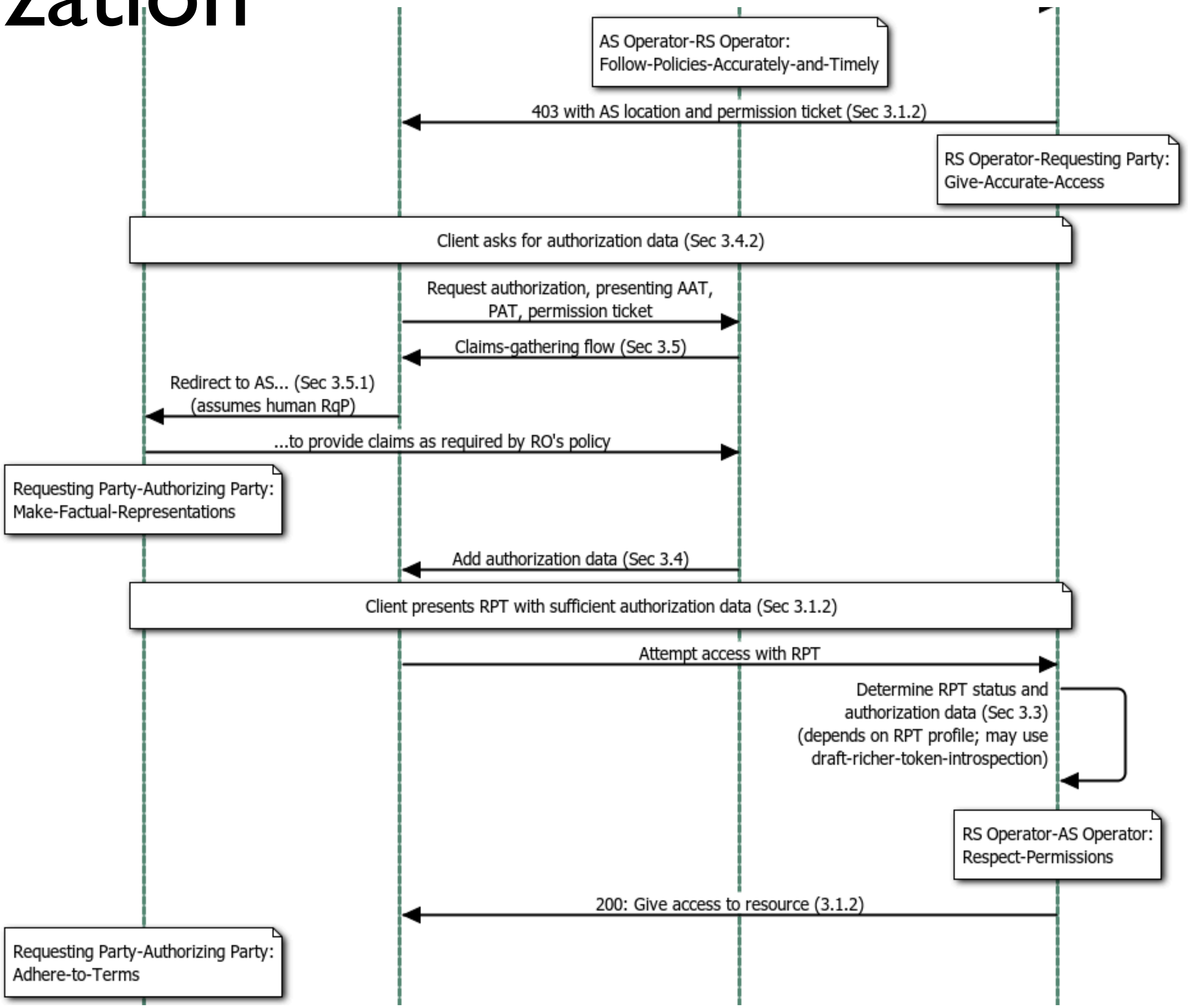
RS Operator-AS Operator:
Register-Accurately-and-Timely

Register client-requested permission,
presenting PAT (Sec 3.2)

Permission ticket

Phases 2 and 3: get authorization and access resource

1 of 3



AS Operator-RS Operator: Follow-Policies-Accurately-and-Timely

403 with AS location and permission ticket (Sec 3.1.2)

RS Operator-Requesting Party: Give-Accurate-Access

Client asks for authorization data (Sec 3.4.2)

Request authorization, presenting AAT, PAT, permission ticket

Claims-gathering flow (Sec 3.5)

Redirect to AS... (Sec 3.5.1) (assumes human RqP)

...to provide claims as required by RO's policy

Requesting Party-Authorizing Party: Make-Factual-Representations

Add authorization data (Sec 3.4)

Client presents RPT with sufficient authorization data (Sec 3.1.2)

Attempt access with RPT

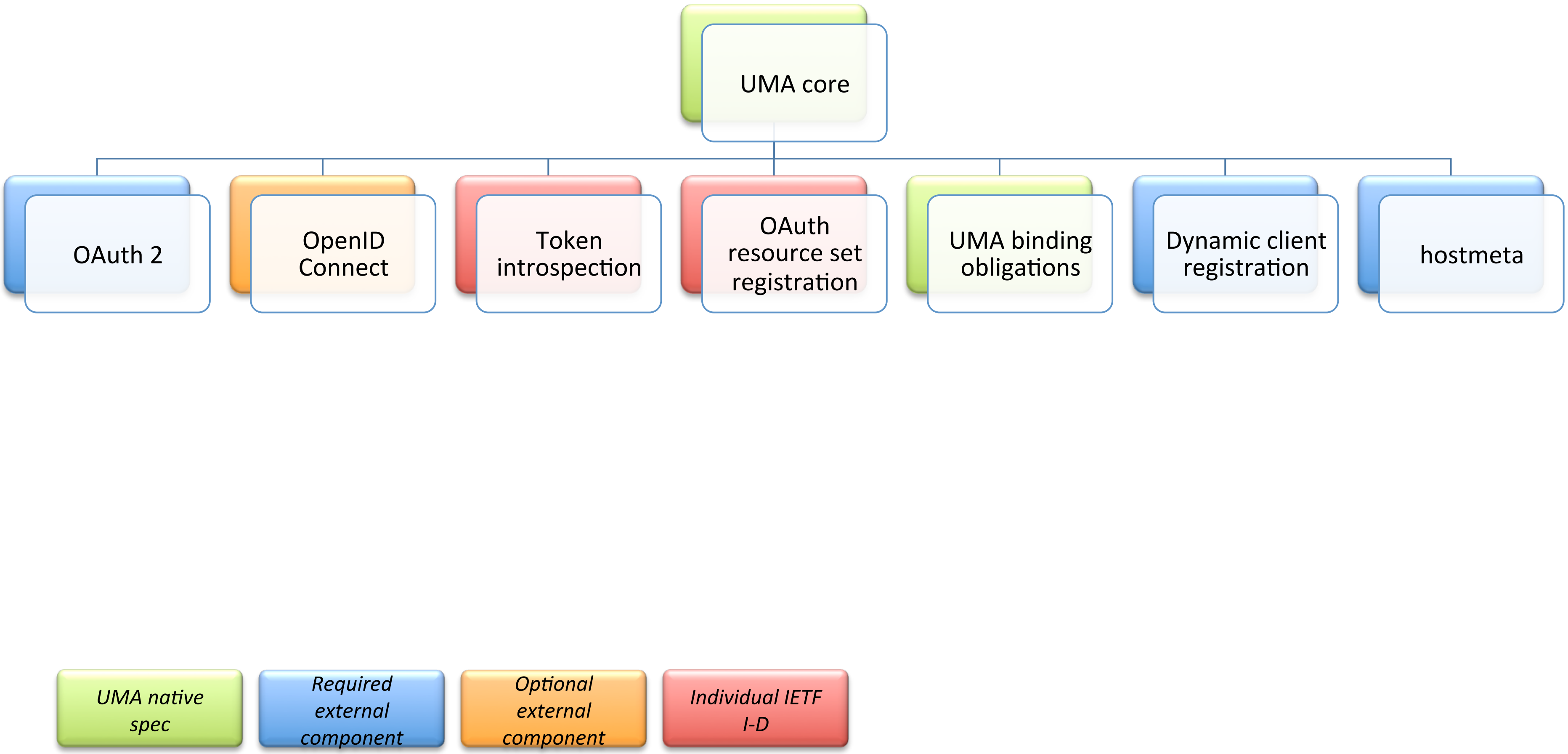
Determine RPT status and authorization data (Sec 3.3) (depends on RPT profile; may use draft-richer-token-introspection)

RS Operator-AS Operator: Respect-Permissions

200: Give access to resource (3.1.2)

Requesting Party-Authorizing Party: Adhere-to-Terms

Spec call tree for the UMA profile of OAuth



UMA core

OAuth 2

OpenID Connect

Token introspection

OAuth resource set registration

UMA binding obligations

Dynamic client registration

hostmeta

UMA native spec

Required external component

Optional external component

Individual IETF I-D