Leveraging UMA’s Power for Compliance and User Control

Eve Maler | VP Innovation & Emerging Technology | @xmlgrrl
#EIC18
www.brighttalk.com/channel/16337/forgerock
Applying Innovative Tools for GDPR Success
Some use cases/ecosystems involving UMA

• Financial
  • Discovering and aggregating UK pension accounts and sharing access to financial advisors
  • Examining suitability for permissioning call center worker access

• IoT
  • “ACE actors” architecture identifies requirements for RqP authorization

• Healthcare
  • Profiled in Health Relationship Trust (HEART) at OpenID Foundation
  • Part of the new OpenMedReady framework, along with HEART
OAuth, OIDC, and UMA2: breaking it down
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”

Resource owner

Client

Authorization server

Resource server

Authorizes (consents) at run time after authenticating, at the AS

App gets consent based on the API scopes it requested; it has its own identity distinct from the RO’s
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”

- **Authorization server**
  - Standard OAuth endpoints for authorization and access token issuance

- **Resource server**
  - App gets consent based on the API scopes it requested; it has its own identity distinct from the RO’s

- **Client**
  - Authorizes (consents) at run time after authenticating, at the AS

- **Resource owner**
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”

- Authorization server
  - Standard OAuth endpoints for authorization and access token issuance

- Resource server
  - Some number of API endpoints that deliver the data or other value-add

- Client
  - Authorizes (consents) at run time after authenticating, at the AS
  - App gets consent based on the API scopes it requested; it has its own identity distinct from the RO’s

- Resource owner
  - It has helped to kill the “password anti-pattern”
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”

- Authorization Request
- Authorization Grant
- Authorization Grant
- Access Token
- Access Token
- Protected Resource

Resource owner
Client
Authorization server
Resource server

Authorizes (consents) at runtime after authenticating, at the AS
App gets consent based on the API scopes it requested; it has its own identity distinct from the RO’s
Standard OAuth endpoints for authorization and access token issuance
Some number of API endpoints that deliver the data or other value-add
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”

This can come with a refresh token for renewal without the RO’s intervention.

Authorizes (consents) at run time after authenticating, at the AS.

App gets consent based on the API scopes it requested; it has its own identity distinct from the RO’s.

Standard OAuth endpoints for authorization and access token issuance.

Some number of API endpoints that deliver the data or other value-add.
OAuth is for constrained delegation to apps
It has helped to kill the “password anti-pattern”

The RO can revoke the token to withdraw authorization (consent)

Authorizes (consents) at run time after authenticating, at the AS

App gets consent based on the API scopes it requested; it has its own identity distinct from the RO’s

Standard OAuth endpoints for authorization and access token issuance

Some number of API endpoints that deliver the data or other value-add

This can come with a refresh token for renewal without the RO’s intervention
OpenID Connect does modern-day federation
It is an OAuth-protected identity API, plus a bit more
OpenID Connect does modern-day federation
It is an OAuth-protected identity API, plus a bit more

- Resource owner = Federation user
- Client
- Authorization server
- Resource server
OpenID Connect does modern-day federation
It is an OAuth-protected identity API, plus a bit more

- Resource owner = Federation user
- Client = Relying party
- Authorization server
- Resource server
OpenID Connect does modern-day federation
It is an OAuth-protected identity API, plus a bit more

- Resource owner
  - = Federation user
- Client
  - = Relying party
- Authorization server
  - = Identity provider ("OpenID provider")
- Resource server
OpenID Connect does modern-day federation
It is an OAuth-protected identity API, plus a bit more
OpenID Connect does modern-day federation
It is an OAuth-protected identity API, plus a bit more

Resource owner = Federation user
Client = Relying party
Authorization server = Identity provider ("OpenID provider")
Resource server

Token endpoint typically delivers an “ID token” similar to a SAML assertion
Standard UserInfo endpoint can be called with an access token to look up identity claims
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth

Resource owner

Requesting party

Client

Authorization server

Resource server
User-Managed Access is for cross-party sharing

UMA brings next-gen delegation and consent to OAuth

Resource owner

Authorization server

Client

Requesting party
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth

Resource owner

Authorization server

Client

Requesting party
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth

UX

Resource owner

At run time

Authorization server

Requesting party

Client

Resource server

Resource server

Resource server

© 2018 ForgeRock. All rights reserved.
User-Managed Access is for cross-party sharing

UMA brings next-gen delegation and consent to OAuth

UX
Share
Opt in

Ahead of time
At run time

Authorization server

Requesting party

Client

Resource server

Resource server

Resource server

© 2018 ForgeRock. All rights reserved.
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth

UX
Share
Opt in
Approve

Ahead of time
At run time
After the fact

Authorization server

Resource server

Resource server

Resource server

Requesting party

Client
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth

- **UX**
  - Share
  - Monitor

- **Resource owner**
  - Ahead of time
  - Anytime
  - At run time
  - After the fact

- **Authorization server**

- **Requesting party**

- **Client**

- **Resource server**

© 2018 ForgeRock. All rights reserved.
User-Managed Access is for cross-party sharing
UMA brings next-gen delegation and consent to OAuth

Resource owner

Ahead of time  Anytime  Anytime  At run time  After the fact

UX Share  Monitor  Withdraw  Opt in  Approve

Authorization server

Requesting party

Client

Resource server

Resource server

Resource server

© 2018 ForgeRock. All rights reserved.
Key benefits of UMA to service providers

- True security of delegated access
- Scalability of resource permissioning
- API-first protection strategy
- Fosters control for compliance and trust
Key benefits of UMA to consumers

- Constrained party-to-party delegation
- Granting consent without external influence
- Centralized monitoring and management
- Control of consents at a fine grain
The BLT sandwich:
Business scenarios needing solutions

- A “too-young” individual has a legal guardian managing their digital assets
- That individual starts using digital assets but is too young to consent to their use
- Multiple individuals manage digital assets jointly
- An individual who may become mentally incapacitated or die needs to designate someone else to manage digital assets for them
  - RUFADAA in the US and other laws

Custodians of digital assets – operators of resource servers – are concerned about the liability and risk of these situations too!
The new UMA business model defines how the UMA protocol enables a license-based model for controlling access rights to personal digital assets.
The new UMA business model defines how the UMA protocol enables a license-based model for controlling access rights to personal digital assets.
UMA + identity relationship management

• Model the relationships in, say, a graph database
• Implement each life (relationship) stage in UMA
• We have mapped legal devices to technical artifacts: OAuth/UMA token, policies, etc.
  • These artifacts are auditable
  • UMA assists in unique properties for compliance and user trust
• When a relationship changes, the artifacts can be torn down and new ones can be built up
  • These changes themselves can be made auditable
  • Much like “right to erasure” workflows, they can be hardened
Thank you!

Questions?

Eve Maler | VP Innovation & Emerging Technology | @xmlgrrl

#EIC18