HISTORY

This draft profile was created by Origo Services Ltd (Origo). Contact Kenneth May (kenneth.may@origo.com)

Version 1.0, dated November 2019 was authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo from March – Sept 2019.

This profile and the corresponding design document are outputs from Origo's extensive work on the UK Pensions Dashboard initiative since 2014. Origo widely demonstrated a solution based on UMA version 1 in 2016 and 2017 before undertaking further work to develop an UMA2 based profile.

In December 2020 this draft profile and the corresponding design document were contributed to the Kantara Initiative's UMA Working Group by Origo.

INTRODUCTION

This document profiles UMAGrant and UMAFedAuthz and presents additional material and standards. It defines the technical protocol standards for the UK's Pension Dashboard Eco-system.

DN *Other introductory blurb and references*

A design document is available to provide readers and implementors with context and more detail which would not be appropriate to include in a formal profile.

DN *Ref to design doc*

The two UMA standards are profiled, for each, relevant additional standards are referenced and where necessary profiled. Headings in these sections refer to the section numbers of the UMA standards which are being profiled.

A further section presents considerations not mentioned in context of the previous two profiles.

SECTION 1 Pension Dashboard Profile of UMA Grant [PDP UMAGrant]

1 [UMAGrant] 1. Introduction

This is a profile and extension of [UMA Grant]. This profile is the standard for delegated authorisation and access for the UK Pension Dashboard Eco-system. Its URI is:
URI_this_PD_UMA_profile

The Pension Dashboard standards REQUIRE the use of the profile of Federated Authorisation for UMA [PDP UMAFedAuthz].

A **Pension Dashboard (PD)** is an UMA client, used by a Pension Owner or a Delegate, acting in the capacity of UMA requesting party to access UMA protected pension resources. There is *no* assumption that the PD and the AS share the same source or assurance level of identity assertions of the requesting parties. This profile assumes that the requesting party at the PD is delegated authorisation to access the Pension Owner's pensions. So, UMA delegation is applied for PD users, whether they be Pension Owners or Delegates.

The PD also initiates a Find operation (not in scope of this UMA profile) optionally following which the protection of found resources occurs (see [PDP UMAFedAuthz]).

2 [UMAGrant] 1.2 Roles

A **Pension Owner** is the natural person who owns the financial value of the pension investment. This person is able to authorise UMA protection of her resources (see [PDP UMAFedAuthz]), and then to access them using the UMA grant using a client called a Pension Dashboard. She is also able to control access policy for **Delegates** who are other natural persons to also access her resources.

When the Pension Owner authorises protection of her resources, she is authorising the Resource Server to access the UMA Authorisation Server to manage access to her resources (see [PDP UMAFedAuthz]).

The **Resource Owner** is the Pension Owner.

The **Requesting Party** is either the Pension Owner using a Pension Dashboard (PD), or a Delegate of the Pension Owner using a Pension Dashboard.

3 [UMA Grant] 1.3.1 Authorisation Process

Non-normative *section.*

Authorisation decisions will be made by the Pension Dashboard UMA AS using policy configured by the Pension Owner acting as Resource Owner to perform authorisation of Requesting Party, and claims made by the PD or other sources of trusted claims in the Pension Dashboard Ecosystem.

The Policy is at least one instance of this policy template:

- *<Pension Owner> grants <scope="value"> to <Requesting Party> of <role="owner" xor "delegate"> at <specific identified Pension Dashboard> to <list of Pension resources> until <time limit>.*
- *Any such policy statement may be revoked by the Pension Owner before the time limit.*

The claims used by the AS to make authorisation decisions against this policy for specific pension resources are:

- *Identity of requesting party (at specific identified pension dashboard)*
- *Identified pension dashboard (making access request)*
- *Identity of requesting party at AS (using federated identity service)*
- *Trusted professional status of the requesting party at AS*
- *Trusted status of requesting party (owner or delegate)*
- *PCT representing the association of Identity of requesting party at PD and at AS, specific identified dashboard, status of requesting party.*

4 [UMAGrant] 2 Authorisation Server Metadata

REQUIRED claims_interaction_endpoint <as_static_claims_interation_URI>

This static endpoint is used to support client-initiated redirection. This will occur when a PD is aware that authorisation will require the Requesting Party to authenticate (typically when a PD knows that

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

it needs to refresh claims or PCT, or when a user has initialised a (new) PD with her resources so the PD knows that the user will need to be stepped up at the AS).

(Note: Initially during Find operations there will be no issued Permission Ticket so [PDP UMA Grant] does not apply. Note in that context the AS will also be behaving as a trust anchor for the customer authentication and subsequent authorisation for RS actions. See [PDP UMAFedAuthz]).

REQUIRED uma_profiles_supported ***DN:*** *URI_this_PD_UMA_profile*

The AS may support dynamic client registration, and if so it is REQUIRED to support claims_redirect_uris.

[BCPOAuthSec] Modified by this profile to apply to UMA: Requesting Party `Clients MUST use a unique` claims `redirect URI`(s) `for the` UMA `authorization server`(s) `used by the application. The client MUST store the redirect URI along with the session data (e.g. along with "state") and MUST verify that the URI on which the authorization response was received exactly matches.`

5 [UMAGrant] 3.1 Client Requests Resource Without Providing an Access Token

Non-normative *section.*

The access attempt is assumed to contain at least the following information to the Resource Server. See also section 29 below for related information.

- *An identifier for the 'Pension Owner at the RS'*
- *An identifier for the 'pension resource' owned by the Pension Owner at the RS which is being accessed, OR*
  a token meaning 'all pension resources' (but see 11 for an issue with this) owned by the Pension Owner at the RS
- *The type of requesting party (owner or delegate)*

6 [UMAGrant] 3.2.1 Resource Server Response to Client on Permission Request Success

WWW-Authenticate header must state the unique realm string for the pension dashboard eco-system: "PensionDashboard".

DN *the type header field 'UMA' in the standard appears not to be registered with IANA[1]. So, the assumption is that all browsers will ignore the authentication header, which is desired behaviour for this profile.*

7 [UMAGrant] 3.3.1 Client Request to Authorization Server for RPT

[1] Check intention with UMA WG? (Not critical at all, just strange it is not registered)

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

REQUIRED claim_token. The client must provide a claim of type pension_dashboard_rqp. The contents of this token must represent the current requesting party user at the dashboard client.

REQUIRED claim_token_format. The claim (above) is of a specific type 'pension_dashboad_rqp'. UMA requires this parameter to match the claim(s). (See 15.2 Claim *pension_dashboard_rqp*.) ***DN** this type name needs agreed URI format.*

OPTIONAL pct. The client MUST provide its existing PCT for the requesting party (ie for the combination of the client and the its user), for all resources the client is seeking to access for that requesting party, if it has one even if it knows that the PCT has expired.

OPTIONAL rpt. The client SHOULD provide its existing RPT for the resources it requested in the previous call to the same RS for the same requesting party[2], if it has one, even if it knows that this RPT is expired. *(Although the profile will not upgrade RPTs, this is included to enable flexibility for possible future extensions.)*

REQUIRED scope. The client request MUST contain the requested pension dashboard scopes.

8 [UMAGrant] 3.3.2 Client Redirect of Requesting Party to Authorization Server for Interactive Claims-Gathering

REQUIRED claims_redirect_uri. Must be provided by the client. Must match (one of) the URIs that must have been provided at registration time for that client.

REQUIRED state. Must be provided.

9 [UMAGrant] 3.3.4. Authorization Assessment and Results Determination

Non-normative: The AS will perform authorisation assessment, ensuring that the Pension Owner's policy is applied controlling access to her resources for either her delegate(s) or for herself at specific dashboard instances. It is envisaged that the AS will provide an interface enabling her to prevent a dashboard from accessing her resources even if she no longer has a relationship with the dashboard.

The results of an authorisation assessment will be an UMA compliant RPT granting scoped access to one or more of her resources with appropriate scopes, or it will result in a failure requesting client redirection, the client to wait for an out of band message from the owner to try again, or a permanent authorisation failure.

[2] Note connection with decision on the arity of access request and RPT (see also footnotes 3,4). Also note that, if the there is more than one RPT for the same RqP at the same RS, the UMA syntax doesn't appear to support this! (3.3.1) Question for UMA WG? Can we repeat the rpt= parameter in the POST?

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

10 [UMAGrant] 3.3.5. Authorization Server Response to Client on Authorization Success

REQUIRED pct. PCT must be of type 'pension_dashboard_pct'.

OPTIONAL upgraded. This profile does not upgrade RPTs. AS must return upgraded: false. *(Although the profile does not upgrade RPTs, this remains OPTIONAL to enable flexibility for possible future extensions.)*

REQUIRED token_type. The RPT is profiled and the token type must be 'pension_dashboard_rpt'.

11 [UMAGrant] 3.3.5.1. Authorization Server Upgrades RPT

The AS does not implement RPT upgrading. *(Although the profile will not upgrade RPTs, this remains optional to enable flexibility for possible future extensions if access scopes were to be extended.)*

Each issued RPT must be new.

Whether a new RPT is issued or not, the AS must revoke existing RPTs for the resources owned by the Pension Owner at the Resource Server to which the permission ticket relates[3] which are also related to the relevant instance(s) of pension_dashboard_rqp and pension_dashboard_pct.

Note Re multiple customers of an IFA – the same pension_dashboard_rqp and pension_dashboard_pct will describe the IFA - revocation applies only to RPTs relating to the pension resources for the relevant Pension Owner at the RS, not to all pension owners related to that IFA.

Whether a new RPT is issued or not, the client[4] must discard any existing RPT(s) for the resources owned by the Pension Owner at the Resource Server to which the permission ticket relates.

12 [UMAGrant] 3.3.6. Authorization Server Response to Client on Authorization Failure

12.1 Authorisation can proceed

In cases where the authorisation process can proceed, the AS requests the client using the need_info structure as profiled as below.

[3] Issue with possible overlap of resources of existing RPT and new permission ticket/new RPT: what if the resources in the new RPT are not the same as the old RPT? This could remove resource authorisation from the client. Eg RPT1 {r1, r2}, RPT2 {r1, r3} revoking RPT1 leaves client unauthorised to access r2. Question: is this ever in practice going to happen? Is it simpler (albeit less efficient) to have an RPT per pension resource? Related to the topic of how to identify a pension resource across the industry and the semantics of the statement "Alice's pensions at the specific Resource Server".

[4] This can only be done by a client if it examines the contents of the permission ticket (!) to determine which resources the RS has included in the ticket **or** it discards RPTs related to the Pension Owner at the RSs. Note connection with discussion of arity of the requests: the only way a PD client actually knows which resources are being requested in the permission ticket is if it asks for them explicitly in the original tokenless attempt – either one or explicit references to each one in the same call. If we have the syntax 'all alice's pensions' then the client doesn't know what they are!

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

need_info redirect_user. The AS must provide a redirection uri as a parameter, to which the AS is expecting the client to be redirected so that the AS can perform interactive gathering with the user.

12.2 Authorisation must wait for Pension Owner action

In cases where the authorisation fails because the Pension Owner has not established policy for a Delegate the eco-system may decide to implement notification of the Pension Owner.

OPTIONAL request_submitted. In the context of optional implementation under this profile of the request_submitted variant, and contrary to the UMAGrant standard, the AS must *not* include a permission ticket in its response, and consequently, must *not* include an interval.

The client must *not* poll the token endpoint.

Non-normative: Rationale: The AS may implement a process to contact the Pension Owner in the event that a Delegate attempts access prior to the Pension Owner configuring her policy. However, this process would be expected to take a significant time and it is unwise to initiate polling policies for what in the Pension Dashboard eco-system should be out of band communications between the Pension Owner and her delegates.

13 [UMAGrant] 3.4. Client Requests Resource and Provides an RPT

The Client must use only the token type pension_dashboad_rpt for RPT.

A Resource Server[5] must accept only the token type pension_dashboad_rpt for RPT.

14 [UMAGrant] 3.6. Authorization Server Refreshes RPT

The AS must not issue refresh tokens.

The lifetime of RPT (given token binding) could be long enough for initial uses by the user. The PCT retains the assured user state so can serve as an 'enhanced' refresh token over longer timeframes.

15 [UMAGrant] 4. Profiles and Extensions

This section provides extensions to [UMAGrant] and defines the token types used by the profile. Elsewhere this profile defines restrictions of [UMAGrant].

15.1 Scopes

Scopes used by the Pension Dashboard eco-system are stated in this section. All other scopes must be ignored by the PD AS and should not be used by Pension Dashboard clients.

Scopes are plain text string literals.

Pension Dashboard valid scopes are "value"[6], "owner", "delegate".

[5] Resource Server here is a Pension Dashboard Resource Server – note that the same pension provider may offer different external interfaces and participate in different eco-systems. This is a constraint for the Pension Dashboard standards and profiles only.

[6] The value scope means obtain the financial value of the pension right represented at the resource. Although this is the only current action permitted in the pension dashboard client others are expected over time. The authorisation is to be explicitly represented by this scope.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

Pension Dashboard valid **requested** scopes (in a permission ticket) must be a list of two containing "value" and "owner" or "delegate".

Pension Dashboard valid **granted** scopes (in a Requesting Party Token, RPT) must be a list of two containing "value" and "owner" or "delegate".

15.2 Claim *pension_dashboard_rqp*

***DN** this type name needs agreed URI to comply with [UMAGrant].*

An extension defining the claim of token type 'pension_dashboard_rqp' which a Pension Dashboard client must issue, containing user@dbi, dbi, owner|delegate.

The client issues a pension_dashboard_rqp to identify (at application level[7]) the user, the dashboard instance, and the role the user is playing in using the dashboard. It is presented with every authorisation request and asserts that the **user** acting in **a role** is **controlling** the **dashboard instance**.

The AS uses the PCT in association with the asserted pension_dashboard_rqp claim to determine whether to require the user to (re)prove their identity and professional credentials at the claims interaction endpoint.

Logical description of the RqP:
{
User: useri@DBi
// UUID of the user at the dashboard instance –
// Allocated by the dashboard, unique for each user of the DB
DB: Dbi
// UUID[8] of the dashboard instance
// Perhaps need logical DBi AND ClientID

Role: "owner" | "delegate"
}

Formal description of the token. The RqP token is a JWT as defined in [JWT] and profiled here.

REQUIRED iss. Registered claim name. Defined [JWT]. Profiled: unique identifier within dashboard eco-system of the dashboard instance issuing the JWT. *(This is Dbi.)*

REQUIRED sub. Registered claim name. Defined [JWT]. Profiled: unique identifier within scope of iss, of the requesting party which is authenticated to iss at the time the JWT is issued. *(This is user@dbi.)*

REQUIRED aud. Registered claim name. Defined [JWT]. Profiled: unique identifier within the scope of the dashboard eco-system of the authorisation server.

REQUIRED iat. Registered claim name. Defined [JWT]. Profiled: time of issue.

[7] See elsewhere for token binding considerations providing sender and audience constrained tokens.
[8] Dbi == ClientID of DB. Need to be tied but maybe not the same value. CleintId policy needs to be defined. ClientIDs for public clients - ClientID per instance is defined later in this specification, so a single parameter here is fine, i.e. Dbi=OAuth ClientID. Note connection with lifetime of client see section 40 (bul 2).

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

REQUIRED exp. Registered claim name. Defined [JWT]. Profiled: time of expiry.

REQUIRED jti. Registered claim name. Defined [JWT]. Profiled: using jti as the unique token identifier.

Public claim name. *none*

REQUIRED role. Private claim name. The iss states the role in which the requesting party is acting. String value. One of “owner” or “delegate”.

The token MUST be signed by the issuer. The token may be encrypted for the AS.

The token may be persisted[9] by the AS to enable correlation across presentations of such tokens.

The token will always be presented to the token or claims interaction endpoints at the AS by the DB client along with an AS issued token. It does not need to be bound.

Privacy: The token carries only identifiers, not explicit names/details of persons, so enables correlation of identifiers across the eco-system. (This is its purpose.) If the subject (the requesting party) uses the same assured identity at the AS across dashboard accounts, the tokens enable the AS to track requesting party across dashboard uses. (This is its purpose if the user wishes to use a single set of consents at the AS.)

15.3 Permission Ticket *pension_dashboard_pmt*

An extension defining the Permission Ticket which the AS must issue.

The AS issues a permission ticket as defined by [UMAGrant]. The permission ticket is a structured token (as permitted in [UMAGrant] 5.5).

This extension is to minimise state management[10] at the AS.

Logical description of the PMT:
{
// step 1
// initial permissions as stated by the RS to the AS in asking for the PMT
RequestedPermissions: [{
resourceID: <string> _id,
scopes: [“value”, “owner” | “delegate”]
},]
// present from initial token-less access attempt

// step 1
ResourceOwner: owner // from the PAT – but derivable by the RS from the _ids
RequestingRS: rs // from the PAT – and from the origin RS of the permission ticket request
// present from initial token-less access attempt

// step 2

[9] Depending on the decisions regarding AS state management and permission token design.
[10] This is a design decision so strictly not a profile decision. However, if this token is defined here then we make explicit the state the AS needs to make dashboard eco-system UMA authorisation decisions. Implementors may decide to take some of this state ‘back into’ the AS.
IMO the AS needs to be as light on data load as possible – we are building a big eco-system with UMA and the AS is in the centre of everything!

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

pension_dashboard_rqp: authorisingParty | null
// null/absent before the AS issues a need_info request
// present as the AS reissues a new permission ticket because it is asking the client to redirect for claims gathering
// and thereafter in subsequent re-issues of the same authorisation journey

// step 3
User: {USERi@IDPi, IDPi} | null
Role: null | owner | delegate
// nulls/absent before AS has determined the assured identity and professional status at the AS
// present if the AS reissues a new permission ticket after obtaining identity & professional status claims
// and thereafter in subsequent re-issues of the same authorisation journey (although none currently expected)
}

Formal description of the token. The PMT token is a JWT as defined in [JWT] and profiled here.

REQUIRED iss. Registered claim name. Defined [JWT]. Profiled: unique identifier within dashboard eco-system of the AS issuing the JWT.

REQUIRED sub. Registered claim name. Defined [JWT]. Profiled: unique identifier within scope of iss of the Resource Owner identifier at the AS *(derived from the PAT used in the initial permission ticket request).*

REQUIRED aud. Registered claim name. Defined [JWT]. Profiled: unique identifier within the scope of the dashboard eco-system of the authorisation server.

REQUIRED iat. Registered claim name. Defined [JWT]. Profiled: time of issue.

REQUIRED exp. Registered claim name. Defined [JWT]. Profiled: time of expiry.

REQUIRED jti. Registered claim name. Defined [JWT]. Profiled: using jti as the unique token identifier.

Public claim name. *none*

REQUIRED rs. Private claim name. The identifier of the Resource Server. *(Derived from the PAT used in the initial permission ticket request.)*

OPTIONAL owner. Private claim name. The identifier of the RO at the Resource Server. *(Derived from the PAT used in the initial permission ticket request.) Maybe of use to the RS to minimise lookup time of resource _id to derive the owner of the resource.*

REQUIRED permissions. Private claim name. UMA permissions as defined in [UMAGrant] and [UMAFedAuthz] using scopes defined in this profile Section 15.1.

OPTIONAL rqp. Private claim name. Structured representation (JSON object) of the *pension_dashboard_rqp* which was presented with the permission ticket (if any) in a previous call to the AS. Claim must be present when the permission ticket is presented for the second or subsequent time by a dashboard client. The AS must populate this claim with the contents of the

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

pension_dashboard_rqp which was presented in the call for which it is reissuing a permission ticket, having checked that the content is in accord with the same requesting party presenter.

OPTIONAL assuredID. Private claim name. Structured representation (JSON object) of the identity of the requesting party (as uniquely represented at the AS) and the asserting identity provider reference. Claim is present when the AS reissues it after assured identification and if necessary, confirmation of the assured professional status of a 'delegate' requesting party.

The token must be signed by the issuer. It must be encrypted for the AS.

The token may be persisted by the AS to enable correlation across presentations of such tokens.

As per [UMAGrant] 5.5 permission tickets are single use: the AS must issue a new token with a new jti for every iteration of the permission process. The AS must ensure that authorisation process and any dependent tokens are revoked if a permission ticket is replayed.

The token will always be presented to the token or claims interaction endpoints at the AS by the DB client so it does not need to be bound further than application level checking by the AS which must ensure that the pension_dashboard_rqp details match across related calls.

Privacy: The token carries only identifiers, not explicit names/details of persons, so enables correlation of identifiers across the eco-system. (This is its purpose.) If the requesting party uses the same assured identity at the AS across dashboard accounts, the AS can track requesting party across dashboard uses. (This is its purpose if the owner-user wishes to use a single set of consents at the AS, or if the delegate-user wishes to manage several owners' resources from the same dashboard). As the token is encrypted it does not leak information from the RS or AS to the dashboard.

15.4 RPT *pension_dashboard_rpt*

An extension defining the RPT (requesting party token) which the AS must issue, to contain the key fields of the pension_dashboard_rqp and the permissions granted to that requesting party.

The AS issues a RPT as defined by [UMAGrant].

Logical description of the RPT:
{
RequiredRqP: {User: useri@DBi, DB: Dbi, Role: self|delegate },
// sender constrained –how does the RS prove sender *person*[11]
// (sender *client* is to be managed with OAuth MTLS)

GrantedPermissions: [{
resourceID: <string> _id,
scopes: ...
},]

ResourceOwner: ro // the identifier of the RO at the RS
}

[11] See MP comment in the Design notes doc 0v5, c. pg 8, and comments in 16.2 below.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

Formal description of the token. The RPT token is a JWT as defined in [JWT] and profiled here.

REQUIRED iss. Registered claim name. Defined [JWT]. Profiled: unique identifier within dashboard eco-system of the AS issuing the JWT.

REQUIRED sub. Registered claim name. Defined [JWT]. Profiled: unique identifier within scope of iss of the requesting party which was assured by the AS when the RPT was issued. *(This is the assured user as known at the AS, based on the PCT or on the contents of a PMT during step-up authentication.)*

REQUIRED aud. Registered claim name. Defined [JWT]. Profiled: unique identifier within the scope of the dashboard eco-system of the resource server. *(Derived from the PAT used in the initial permission ticket request, via the PMTs.)*

REQUIRED iat. Registered claim name. Defined [JWT]. Profiled: time of issue.

REQUIRED exp. Registered claim name. Defined [JWT]. Profiled: time of expiry.

REQUIRED jti. Registered claim name. Defined [JWT]. Profiled: using jti as the unique token identifier.

Public claim name. *none*

REQUIRED rqp. Private claim name. Structured representation (JSON object) of the *pension_dashboard_rqp* which was presented with the permission ticket in the successful authorisation call to the AS.

REQUIRED permissions. Private claim name. UMA permissions as granted by the AS following permission requests represented in the PMT defined above.

OPTIONAL owner. Private claim name. The identifier of the RO at the resource server. *This is from the PMT, initially from PAT. Maybe of use to the RS to minimise lookup time of resource _id to derive the owner of the resource.*

The token must be signed by the issuer. It should be encrypted for the RS.

The token may be persisted by the dashboard in accordance with policy. (See time to live considerations, and caching policy.) The token may be deleted by the dashboard if/when it is known to have exceeded its lifetime, is revoked, or when the dashboard makes an unsuccessful attempt to access a resource using it.

The RPT is bound [OMTLS] to the dashboard client by the AS when it is issued.

The token may be persisted by the resource service in accordance with policy. (See time to live considerations, and caching policy.)

The AS may revoke a token for its own reasons at any time (including RO revocation of policy for owner or delegate access).

The token will always be presented to the RS by the dashboard. The token must be introspected by the RS at the AS introspection endpoint. The results of introspection may be cached according with policy.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

The RS must make access decisions in accordance with [UMAGrant] and may make its own access decisions for instance due to considerations in Section 16.2 in this profile.

Privacy: The token carries only identifiers, not explicit names/details of persons, so enables correlation of identifiers across the eco-system. (This is its purpose.) The AS can correlate access requests made by requesting parties to pension resources across the eco-system.

If the token is encrypted it does not leak information from the RS or AS to the dashboard. Information potentially leaked to the dashboard is: resource owner identifier at the RS, resource _ids of the resources at the AS/RS, the 'sub' claim contains the identifier for the requesting party at the AS.

15.5 PCT *pension_dashboard_pct*

An extension defining the PCT token which the AS must issue to the DB instance requesting authorisation, containing user@DBi, DBi, owner|delegate, USER@IDP, IDP.

The AS issues a PCT as defined by [UMAGrant] and specifically for the purpose defined in this profile.

Its purpose is to prevent the user having to step up their authentication level for every authorisation request by creating a persistent association between the user and role at the dashboard instance with their identity and professional status at the AS assured to (a higher) standard.

Logical description of the PCT:
{
RequiredRqP: {User: useri@DBi, DB: Dbi, Role: self|delegate },
Assured: {User: USERi, IDP:IDPi }
}

Note that the 'User' in the Assured structure is the same 'User' to which the RequiredRqP refers, i.e. the whole token refers to the requesting party, whether this is a delegate or an owner. Thus, this token is independent of the resources being accessed. The Role field of the RequiredRqP will determine in what role the presenter is expected to be acting. The PCT will have been issued only if a User acting in role 'delegate' had been proved to have suitable professional qualifications at the time of the proving the delegate's assured identity. (The AS will include in its authorisation decision whether the pension_dashboard_rqp claim presented at the time of the authorisation request matches that in the associated PCT.)

Formal description of the token. The PCT token is a JWT as defined in [JWT] and profiled here.

REQUIRED iss. Registered claim name. Defined [JWT]. Profiled: unique identifier within dashboard eco-system of the AS issuing the JWT.

REQUIRED sub. Registered claim name. Defined [JWT]. Profiled: unique identifier within scope of iss of the requesting party which was authenticated to the AS as the authorisation is granted. *(This is user@IDP).*

REQUIRED aud. Registered claim name. Defined [JWT]. Profiled: unique identifier within the scope of the dashboard eco-system of the authorisation server.

REQUIRED iat. Registered claim name. Defined [JWT]. Profiled: time of issue.

REQUIRED exp. Registered claim name. Defined [JWT]. Profiled: time of expiry.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

REQUIRED jti. Registered claim name. Defined [JWT]. Profiled: using jti as the unique token identifier.

Public claim name. *none*

REQUIRED rqp. Private claim name. Structured representation (JSON object) of the *pension_dashboard_rqp* which was presented with the permission ticket in the successful authorisation call to the AS.

REQUIRED assuredID. Private claim name. Structured representation (JSON object) of the identity of the requesting party (as uniquely represented at the AS) and the asserting identity provider reference. Claim is present when the AS reissues it after assured identification and if necessary, confirmation of the assured professional status of a 'delegate' requesting party.

The token must be signed by the issuer. It should be encrypted for the AS.

The token may be persisted by the dashboard to support future authorisation requests for the same requesting party in the same role. The DB must delete the PCT when it is presented with a new PCT by the AS for the same user in the same role, or if it knows the PCT is revoked or has expired.

The token will be presented to the token endpoint at the AS by the DB client.

The PCT is bound [OMTLS] to the dashboard client by the AS when it is issued.

Privacy: if the token is encrypted it does not leak information from the RS or AS to the dashboard.

16 [UMAGrant] 5.2. RPT and PCT Exposure

Security consideration: who can play the PCT and RPT?

16.1 PCT Considerations

The PCT is issued from, and presented at, the AS token end point as part of an authorisation request and/or response. In this profile the PCT contains the association of identity at the DB and (a more assured) identity at the AS. The AS has the opportunity to check that the PCT's representation of the dashboard identity matches that presented by the dashboard (an instance of pension_dashboard_rqp) at the time of authorisation. This checks that the dashboard's view of its user is the same as that at the time the PCT was issued and thus enables the AS to determine the assured identity and professional status of the dashboard user. The purpose of the PCT is to avoid the user to having to step up their identity for every authorisation attempt, both across multiple resource servers and over time, whilst preserving the context of the dashboard user and the user's role. The PCT has the characteristic of a refresh token for a relying party user's identity, role and professional status at a specific dashboard client.

The PCT does not protect against a malicious/erroneous dashboard implementation which could

- fabricate its assertion of the current user to match that contained in the PCT (if it knows this by another route and can sign its assertion)
- quote a PCT for a legitimate user in the hope of obtaining access for a different malicious user.

The PCT itself does not protect against replay of the PCT if the PCT were to be stolen. However, the combination of PCT and client assertion of the dashboard identity (an instance of pension_dashboard_rqp) does provide an application level mechanism for the AS to check that the

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

user identities match and that the presenter and the audience for which the PCT was issued correspond.

In addition, this profile proposes PCT token binding by MTLS to constrain the token to the dashboard client to which it was issued by the AS. Notwithstanding [OMTLS] 7.1, concerning implicit binding of refresh tokens in cases where confidential clients use tls based client authentication, this profile proposes OAuth mutual TLS token binding for all client types and all client authentication types.

TBD: [BCPOauth] recommends refresh token cycling for public clients. PD eco-system decisions on public clients is TBD. The AS may implement PCT cycling. That is a new PCT is issued every time a PCT presented and if a PCT is ever presented more than once, all related instances (including an unplayed one) will be revoked by the AS, forcing the legitimate RqP to reauthenticate at the AS.

16.2 RPT Considerations

The RPT represents authorisation for a specific (human) requesting party at a specific dashboard client to access specific resources at a RS with stated scopes. It is an audience restricted token by virtue of the resource _ids, which were issued by the AS to the RS, thus, it represents permissions for specific named resources uniquely identified at a specific RS.

In addition, this profile proposes RPT token binding by OAuth mutual TLS to constrain the token to the dashboard client to which it was issued by the AS.

However, such binding provides sender constrained RPTs but only constrained to the client. In the PD eco-system, the RPT is (logically) issued to the UMA relying party, a human user, represented at authorisation time by the pension_dashboard_rqp claim (an instance of which must match that provided when the PCT was issued).

The RPT does not protect against a malicious/erroneous dashboard implementation which could

- Present an RPT in such a manner as to gain access to a resource without the user being present or for a purpose other than that requested by the user.

The RPT, defined as above, contains the details of the RqP as presented to the AS at authorisation time and at PCT issue time. This is so that the RS can apply policies as it may require. (For instance, audit of stated user/dashboard/role access, or for any other purpose.) See [UMAFedAuthz] 1.4. Separation of Responsibility and Authority, para "`The separation of...`".

TBD: Additional application tier mechanisms for binding the current user of the client to the RPT at the time of access to the resource are possible. These may be more invasive of privacy and or constraining on dashboard design.

The least impactful is to require that the RS API includes the pension_dashboard_rqp of the current user at the time the RPT is played. Thus, the RS (or AS by using extension to token introspection RFC7662 2.1) could check that the current user is that to which the RPT was issued. However, as noted above, this mechanism does not protect against malicious dashboard implementation, but it would detect errors and it would enable checks if the identity of the user changed.

Return to this topic in discussion of lifetimes of tokens and of public clients.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

17 [UMAGrant] 5.3. Strengthening RPT Protection Using Proof of Possession

The AS, client and RS must use OAuth Mutual TLS [OMTLS] for token binding.

The RS must accept an RPT only if OMTLS proves its sender is as required by the AS.

The RPT is defined as a structured token defined as an extension in this profile.

The RS must introspect the token at the AS if ... **DN** *considerations of time to live, cache time and check for revocation time, minimising load on the AS, if the token is opaque, forced to introspect, so load on AS is forced every access. See also 21* [UMAGrant] 6.1. Policy Condition Setting, Time-to-Live Management, and Removal of Authorization Grants.

The RS must examine the permissions contained therein to determine appropriate access is authorised prior to returning data to the client.

RS may apply other controls as it requires based on the RPT, the contained assertion of the requesting party, an associated assertion of the current user which it may require as a parameter to its API, or by any other means as it determines.

18 [UMAGrant] 5.5. Permission Ticket Management

As discussed above (15.3 Permission Ticket *pension_dashboard_pmt*) this document extends the profile with a definition of the UMA Permission Ticket.

Permission Tickets have very short lifetimes since in this profile they represent transitory state during requesting party user interaction.

Note also that request_submitted responses must not contain a permission ticket. (see 12.2 Authorisation must wait for Pension Owner action.)

19 [UMAGrant] 5.7. Requirements for Pre-Established Trust Regarding Claim Tokens

This profile requires the use of tokens as defined in Section 15. Audience and Issuer are discussed there.

In addition, the AS enforces authentication of the requesting party using a federated identity service, resulting in an ‘assured identity’ for the requesting party. The AS in that context is the relying party of the chosen Identity Provider. The IDP is the issuer and the AS is the audience.

The profile requires the assured identity is interactively gathered (12.1, 15.3).

20 [UMAGrant] 5.8. Profiles and Trust Establishment

Pension Dashboard is expected to be operated in a controlled or regulated governance framework. Pension Dashboard operators will be required to operate within controls. For this reason, it is envisaged that the requirements for ‘well behaved’ clients implied by section 16, [UMAGrant] 5.2. RPT and PCT Exposure, will be adequately met.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

In addition, technical controls outside the scope of this profile are also expected to be required, for example a private PKI for all participants and an associated registry of participants and their operational behaviour.

21 [UMAGrant] 6.1. Policy Condition Setting, Time-to-Live Management, and Removal of Authorization Grants

Time to live of an RPT is *TBD (maybe 5 days)* a period of time[12] which is sufficient for initial user experience to be acceptable AND the load on the AS to be acceptable. *Although an RPT life time measured in days is unusual, for dashboard clients which can protect the token, the explicit representation of the permissions in the token, and the requirement for token binding, it is not considered to be an undue attack surface.*

To support this requirement (to avoid reauthorisation and avoid excessive load on the AS) the RS will not have to introspect the RPT for an initial period *(maybe 2 days)*. However, this introduces a risk that the Resource Owner (Pension Owner) revokes consent to access pensions within this period; *DN consultation with users and industry needed on time windows of commitment after authorisation of a delegation.*

The PCT will have a time to live measured in months. This will support the user experience of being able to silently re-authorise across all their resource servers based on their assured identity. Of course, this will also support as short a RPT lifetime as is desired but at the expense of load on the AS.

22 [UMAGrant] 6.2. Requesting Party Information at the Authorization Server

The AS should have minimal PII. In the PD eco-system it can be expected to manage in excess of 15 million Pension Owner's authorisation policies, however, the AS itself does not need to persist PII on these parties.

As above (15.3 15.5) we propose that the PMT and PCT manage state externally to the AS (relying on cryptographic means to ensure integrity and confidentiality as required). Both of these tokens are issued and consumed by the AS, accordingly they can be opaque to other parties.

The AS (and RS and Clients) will also operate privacy preserving mechanisms for handling identifiers. *DN these are defined in sections (...) concerning encryption of tokens in this specification, and in the associated Pension Dashboard design documentation.*

The AS will not store any biographic details (name, address, contact details, NINO) of the Pension Owner user.

[12] RPT lifetime may have to be dependent upon client type: public web app clients may not be able to support protection of the token. To be discussed at a later date, re client types.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

23 [UMAGrant] 6.3. Resource Owner Information at the Resource Server

Although as UMAGrant points out the initial access request could enable a client to obtain the AS address, this is not of privacy significance in the Pension Dashboard service as (initially at least) all users are served by the same AS.

The return from an initial access attempt containing a valid permission ticket implies that the Pension Owner associated with the resources is confirmed to have a pension at that resource server. The pension dashboard clients must not use this process to determine the resource server which handles resources.

24 [UMAGrant] 6.4. Profiles and Trust Establishment

Dashboard clients and RS and AS will also have to comply with other governance and security controls. Also referenced in section 20 above.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

SECTION 2 Pension Dashboard Profile of UMA Federated Authorisation [PDP UMAFedAuthz]

25 [UMAFedAuthz] 1. Introduction

This is a profile and extension of [UMAFedAuthz]. This profile is the standard for Federated Authorisation of Pension Providers' Resource Servers for the UK Pension Dashboard Eco-system.

The Pension Dashboard standards REQUIRE the use of the profile of Federated Authorisation for UMA [PDP UMAFedAuthz].

In the Pension Dashboard eco-system, the Pension Owner typically does not know the resource server(s) which can serve her pension information; indeed, typically she does not have an online account with the provider, and many providers do not currently offer online services. This means that the mechanism of finding the pension resources, and placing them under UMA protection needs to be provided by the eco-system, so that the Pension Owner can then see her pensions in the Pension Dashboard client.

This profile standardises how a pension provider UMA Resource Server will register the Pension Owner's pension resources with the UMA Authorisation Server. For future technical and commercial extensibility, the profile allows such resources to be registered with an AS other than the Pension Dashboard AS. Resource Servers can remove resources from authorisation should they need to do this, using the standard UMA protection API. The profile specifies that the RS must use the permission and token introspection APIs at the Pension Dashboard AS.

26 [UMAFedAuthz] 1.3. HTTP Usage, API Security, and Identity Context

The Pension Owner is the UMA Resource Owner. She may consent to her resources being registered with the UMA AS, as part of the service provided by the Pension Finder Service (the latter is outside this specification). At the RO's request, a PAT, required by [UMAFedAuthz] and this profile, is issued to the RS and kept with her record there. The RS uses the PAT in accord with this profile.

Non-normative: Since the resource owner has no online account with the RS, she will use a user interface at the Pension Dashboard AS to consent to resource registration. Once a resource is identified belonging to the RO, the relevant RS will obtain a PAT on behalf of the RO (if it does not already have one arising from prior activity). Since the RS and the AS are in a secure eco-system and the RO is authenticated to the AS at the time of the resource discovery, the RS can use temporary credentials specific to the RO at the AS to obtain the PAT, perhaps using a Resource Owner Password Credentials Grant [OAuth] Section 1.3.3, or perhaps using a JWT authorisation grant type `"urn:ietf:params:oauth:granttype:jwt-bearer"` *as in [JWTOAuth] Section 2.1*[13]*.*

[13] Implementation consideration. See RFC7521 4.1 re lifetime of token issued should not exceed that of the assertion (i.e. of the JWT). In the AS will issue the JWT representing a grant, i.e. 'temporary credentials', and this is exchanged for the PAT. The lifetime of the PAT needs to be very long since it is required for permission tickets etc. In this application the stipulation of RFC7521 4.1 should be ignored as other mitigations will be in place to protect and to refresh the PAT.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

Non-normative: Similarly, when necessary, the PAT can be refreshed when the Resource Owner re-authenticates to the AS for reasons of access to her resources. Alternatively, the RO can at that time instruct the AS/RS to deregister her resources and the AS to revoke the PAT.

Non-normative: To enable a RO to utilise an AS outside the Pension Dashboard eco-system (enabling future commercial and technical extensibility), this profile requires that the PAT is issued explicitly on behalf of the RO. This enables the RO to direct the RS to deregister her resources and to revoke the PAT. A simpler approach, in which the Pension Provider is the (logical UMA) resource owner, and thus the RS has only one PAT, was rejected for this reason of extensibility.

27 [UMAFedAuthz] 1.5 Protection API Summary

The AS must provide the token introspection endpoint to enable the RS to introspect the RPT and to enable revocation of authorisation by the RO.

28 [UMAFedAuthz] 2. Authorization Server Metadata

REQUIRED introspection_endpoint.

REQUIRED uma_profiles_supported (as defined in [PDP UMGrant] 4 above).

29 [UMAFedAuthz] 3. Resource Registration Endpoint

In the Pension Dashboard eco-system, UMA Resource Servers protect resources on behalf of Pension Owners who are natural persons who have ownership of one or more pension rights represented as resources at a Resource Server. A PO is also an UMA Resource Owner and may own resources represented at several Resource Servers. Typically, an UMA Resource Server will be operated on behalf of a commercial entity, a Pension Provider, so that it can participate in the Pension Dashboard eco-system.

This profile requires that Resource Servers register the Resource Owner's resources with the UMA Authorisation Server.

The RS must persist the following, for each Resource Owner, discoverable using its externally addressable resource URI[14](s) as key(s):

- Resource _id (as defined in [UMAFedAuthz] 3.2)
- Resource owner's PAT
- AS URI which issued the PAT (at which the resource _id is registered)

30 [UMAFedAuthz] 3.1. Resource Description

REQUIRED resource_scopes. ["value", "owner", "delegate"]. All three scopes must be used for all registrations.

Similar considerations apply to the possible use of ROPCG as the RS client will not actually have long lived RO credentials at the AS.

[14] The design of the URIs which are used by the Dashboard clients to access the resources at the RS. Outside the scope of this profile, but included in the Pension Dashboard design which accompanies the standards.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

REQUIRED name. This is the URI or URN of the resource, i.e. of the pension right at the RS. It will be used as the unique name[15] against which the AS will apply authorisation protection and is the representation of the pension as is available to a Pension Dashboard client.

REQUIRED type. This is a URI of the type of all 'name' parameters used in the Pension Dashboard eco-system. It is required for future extension, e.g. to support resources which have wider scope options than defined here, or for specialised RS-AS relationships in the future.

OPTIONAL description. *TBD Data standards for the PD eco-system will define 'type' and 'name' mandated here. It may be that human readable derivations of 'name' will be aided by this 'description'.*

31 [UMAFedAuthz] 3.2.1. Create Resource Description

The RS must register each pension right as a separate resource (to enable delegation and access at the most granular level).

The RS must register all such resources with the description mandated in Section 30, [UMAFedAuthz] 3.1. Resource Description.

32 [UMAFedAuthz] 3.2.4. Delete Resource Description

If a Resource Server is required by a Resource Owner to register one of her resources with another AS[16] (other than that defined in the Pension Dashboard standards), then the RS must deregister that resource with the PD AS.

33 [UMAFedAuthz] 4. Permission Endpoint

The RS will request permissions for resources which match the call from the client. See [PDP UMAGrant] Section 5, and this profile, Section 29, for how it derives the relevant information.

***DN** On decision concerning single or multiple permissions (footnotes 2 3 4), add explicit statement here too.*

The RS must request scopes "value" and either "delegate" (if the inbound call explicitly requested this), or, "owner" (by default or as explicitly requested), but not both.

The Permission Ticket is defined in [PDP UMAGrant] Section 15.3. The ticket is opaque to the RS but is returned to the Client by the RS as per [UMAFedAuthz] Section 4.2.

34 [UMAFedAuthz] 5. Token Introspection Endpoint

The RS must introspect the RPT associated with an access attempt, in accord with policy [PDP UMAGrant] Section 21, noting that caching the results of introspection is permitted, and that the RPT may also be validated locally. *DN: User vs performance issue as discussed in referenced profile.*

[15] Note there is no requirement that the URI/URN is displayed to the user in its entirety. Textual representation can be separate and aided by the optional description parameter.

[16] This enables an alternative Authorisation Server to be appointed by a user, perhaps in a future, more mature financial services eco-system. Such a user would, at that time, understand that, in so doing, she is preventing her pension resource(s) from participating in Pension Dashboards as supported by this profile.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

As is noted in [UMAFedAuthz] 5, and [RFC7662] 4, there are security concerns. In this profile, as in [UMAFedAuthz] some of these are mitigated by OAuth protection (PAT) of the introspection endpoint.

35 [UMAFedAuthz] 7. Security Considerations

The lifetime of the PAT, PAT protection at RS, and PAT issuance and refresh are considerations for the PD standards as whole, but are not strictly part of [UMAFedAuthz]; non-normative remarks are made in section 26 above. These matters are discussed in SECTION 3 below.

36 [UMAFedAuthz] 8. Privacy Considerations

The PD AS will have resource '_ids' and 'name' descriptors related to the RO's resources at the RS. If the optional 'description' property is used, due consideration should be given to privacy concerns. This profile has limited the RO's information held at the AS to that defined in Section 30, [UMAFedAuthz] 3.1. Resource Description above.

Note in particular that there is no information at the AS related to RO accounts. Should the RO user request that the resources are deregistered and the PAT be revoked, there will be no information at the AS related to the RO in respect of her connection with the RS and resources there.

[PDP UMAGrant] Section [UMAGrant] 6.2. Requesting Party Information at the Authorization Server also limits personal information at the AS

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

SECTION 3 Pension Dashboard Standard – Matters related to UMA profiles

This section covers matters related to the two UMA profiles but which are not explicitly covered in them.

37 Token Design of PDP UMAFedAuthz PAT

Whilst the design of the PAT is outside the scope of the UMAFedAuthz profile, it is defined here as part of the Pension Dashboard standards.

A PAT is an OAuth token of scope uma_protection ([UMAFedAuthz] 1.2).

PATs are persisted by the RS to which they were issued, and associated with the RO's record(s) there. See also Section 29 for other information persisted at the RS.

The PAT 'enables the AS to map the RS's request to the appropriate RO' ([UMAFedAuthz] Sect 4 para 3).

Formal description of the token. This specification defines *pension_dashboard_pat* a structured token type profile of [JWT] with the following claims.

REQUIRED iss. Registered claim name. Defined [JWT]. Profiled: unique identifier within dashboard eco-system of the AS issuing the JWT.

REQUIRED sub. Registered claim name. Defined [JWT]. Profiled: unique identifier within scope of iss of the Resource Owner which was assured by the AS when the PAT was issued. (This is the assured user as known at the AS, based step-up authentication.) *Owner@AS This enables the AS to identify the RO resources and policies on receipt of a request to its protection API without storing any information specific to the RS, and without cross referencing each resource _id.*

REQUIRED aud. Registered claim name. Defined [JWT]. Profiled: unique identifier within the scope of the dashboard eco-system of the resource server.
Note: Used in the initial PMT. This also enables the AS to interact with the RS where necessary. (E.g. identifying its key material for encryption of RPT, e.g. arranging for PAT reissuance on RO permission.) This may be the same as the ClientID of the RS, implementation decision (see also audience).

REQUIRED iat. Registered claim name. Defined [JWT]. Profiled: time of issue.

REQUIRED exp. Registered claim name. Defined [JWT]. Profiled: time of expiry.

REQUIRED jti. Registered claim name. Defined [JWT]. Profiled: using jti as the unique token identifier.

Public claim name. *none*

OPTIONAL owner. Private claim. String representing the unique identifier of the pension owner at the Resource Server. *Owner@RS This enables the AS to propagate this identifier via RPT to simplify the access control of RO resources at the RS.*

The token must be signed by the issuer.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

The token must be persisted by the RS. (See time to live considerations.) The token may be deleted by the RS if/when it is known to have exceeded its lifetime, is revoked, or when the RS makes an unsuccessful attempt to access a AS api using it.

The PAT (or its refresh token) may be bound to the RS client [OMTLS] by the AS when it is issued.

The AS may revoke a PAT for its own reasons at any time (including RO revocation of policy for federated authorisation).

The token will always be presented to the AS by the RS when invoking the AS protection API.

Privacy: The token carries only identifiers, not explicit names/details of persons, so enables correlation of identifiers across the eco-system. (This is its purpose.)

Privacy: The AS could correlate the RO user identifiers across all RS in the pension dashboard eco-system (but this standard does not require such correlation).

Privacy: The RS can know the RO's assured identity identifier at the AS from the claim in the PAT. If this is considered unacceptable, the token could be encrypted to the AS. (However, the RS logically 'knows' the actual RO identity, since PAT issuance requires this.)

38 Assumptions about PKI

The Pension Dashboard eco-system will include a strong cryptographic mechanism to identify all participants. This expected to be a domain specific PKI providing technical controls to support business and process governance over participants (aka the Pension Dashboard Governance Register).

The PKI will enable the signing and encrypting of messages and related content as may be required by these standards and other Pension Dashboard designs.

The PKI will also support the requirement for token binding [OMTLS] (but not for public clients).

39 Considerations of OAuth Client types and Client related Controls

[UMAGrant] 3.3 bul 3, states that the grant is useable by public and private clients. It also (Section 5) gives some security considerations for implementers, as does this specification. More generally the best practice advice of [BCPOAuthSec] should be applied. This section raises specific issues and makes recommendations.

The UMA flow is initiated by a PMT obtained from a back channel from the RS to the AS. The permission ticket is protected from replay by UMA (single use ticket). PMT lifetime is very short.

Consideration may be given to the potential benefits of binding the PMT to the dashboard client. This can be done as in the DB eco-system the clients are identified and the RS api could be designed to take a parameter of type pension_dashboard_rqp which identifies the requesting party (to which the PMT is returned and then presented at the AS). The whole chain from initial attempt through to RPT/PCT issue would then be bound to the same client/requesting party. However, it is not clear whether there are significant advantages from this given that the PMT is encrypted and has a very short lifetime, and the authorisation process performs this binding from the first presentation of the PMT to the AS by the DB.

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

UMA provides resource-specific authorisation (i.e. the RPT may only be played at a specific endpoint).

Dashboard clients used by owner requesting parties MAY be public. *Tokens (RPT, PCT) can be protected using per user, per platform controls. The impact of loss is mitigated by binding and by the loss impacting a single user at once.*

Dashboard clients used by delegate requesting parties SHOULD be confidential. *Rationale: clients for delegates are likely to handle significant volumes of customer tokens, managed within the scope of each user (or the whole of an organisation if the delegation is to the organisation rather than to a person).*

Longer lived tokens (PCT, RPT) MUST be token bound to provide sender constrained protection [OMTLS], as stated elsewhere in this specification. Considerations of implementation of OMTLS will be covered in later specifications, noting the connection for the RPT with token introspection [OMTLS] 3.2, and the need to extend or wrap the PCT and RPT definitions in this specification with the certificate thumbprint [OMTLS] 3.1.

Clients must implement best security practice based on [BCPOAuthSec] and related references. Public clients must implement controls appropriate to their platform to prevent hijack or impersonation, redirect interception or MITM attacks (e.g. native app specific redirect url handling [RFC8252]). Hybrid public client, confidential host designs should be considered to mitigate some of the risks of public clients (e.g. RFC7591 Section 5 [ODynClient]).

40 Client Registration

Registration of OAuth clients (i.e Dashboards and Resource Servers) is required for the operation of the UMA profiles. The 'governance register' will have to provide services for client registration.

Static registration is more secure but dynamic registration [ODynClient] may be more expected by industry participants (c.f. open banking), may involve fewer manual processes, and is necessary to support public client types.

Client registration may use [ODynClient]. Registration must comply with profiled requirements on callback claims redirect uris [PDP UMAGrant] Section 4.

If the Authorisation Server supports public clients these must be explicitly managed, including the following characteristics:

- Software statements must be issued by the appropriate authority (governance register) [ODynClient] A.2.2
- Client id issued per instance of the public client [ODynClient] A.4.1 (noting appropriate AS and client design decisions made concerning the lifetime of the ClientID and the permitted nature of configuration storage at the endpoint, or at the client's confidential host server)
- Client secret must not be issued to public client
- Clients must register specified URIs in accordance with good practice [BCPOAuthSec]
- AS metadata states support for token binding [OMTLS] 3.3
- [OMTLS] characterises MTLS for *client authentication* of confidential clients only.
- [OMTLS] 4.0 for token binding implies certificate management for public clients by self-signed certs (for confidential clients by eco-system PKI).

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

41 Recommended Token Lifetimes

This section presents recommendations for token lifetimes for the application of these profiles and standards to the Pension Dashboard eco-system.

41.1 Pension Owner Policy

Pension Owners set policy at the AS for delegation to themselves at dashboards, and optionally for their delegates at dashboards, to access 'value' scoped, protected resources.

It is expected that the timeframe for delegation to oneself (role owner) at a dashboard will be 3 months and for a third-party (role delegate) will be 1 month. The Pension Owner can revoke rights early via the AS.

41.2 PAT

Pension Owners, in the role of Resource Owners, grant permission for the AS to determine their access policy to their protected resources at one or more resource servers exposing pension rights managed by pension providers. The PAT lifetime will be refreshed at every visit by the assured Resource Owner to the AS, unless the RO (also the PO) explicitly revokes such a permission (perhaps because she is moving her authorisation to a different AS outside the scope of the pension dashboard ecosystem). At each issuance of a refreshed PAT (to a secure RS client) the TTL (of the refresh mechanism) shall be 18 months.

The PAT is sender constrained: bound by the AS to the Resource Server when the PAT is issued.

This specification has left open the OAuth2 grant used to obtain the PAT and the details of whether a refresh token is used, or an access token. Note however the structure of the PAT has been defined (Section 37) so that the data and state design is explicit.

41.3 PMT

The permission token lifetime will be very short; just sufficient for the round trip concerned (from AS, to RS, to Dashboard and re-presentation at the AS, or from AS to Dashboard and re-presentation back at the AS). TTL less than 60 seconds.

41.4 RqPT pension_dashboard_rqp

The dashboard issues a RqP token for the AS when it needs to assert its user's identity (ie the relying party at the client and the client identity). These tokens have very short lifetimes. TTL less than 60 seconds.

41.5 PCT

The AS issues a Persistent Claims Token so that a dashboard can present it in lieu of the AS forcing the requesting party user to step up their identity (and professional status) claims.

PCT for a person in 'owner' role has TTL 3 months.

PCT for a person in 'delegate' role has a maximum TTL of 1 month.

PCT is sender constrained: bound by the AS to the dashboard client requesting the authorisation which resulted in the PCT issuance.

41.6 RPT

The AS issues an RPT for a specific scope, requesting party, resource and resource server. Its lifetime needs to be sufficient for the dashboard to access the resource and to hold it in memory for the

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd

user. The access needs to be repeated for each session the user initiates at the client (depending on the persistence cache policy of the client).

RPT is sender constrained: bound by the AS to the dashboard client requesting the authorisation.

Consideration needs to be given by implementors whether the AS should be made to re-issue an RPT (using PMT, PCT etc) for *each* access, or whether a period of access is to be permitted using an issued RPT. Trade-offs need to be made between: ease of use of the token, AS authorisation load, AS introspection load, step-up cost (for user), and revocation periods (for RO revoking delegations within RPT lifetime).

A further factor of consideration is the dashboard eco-system's policy on the length of time it may take for an RS to respond to a find request, and thus for a dashboard to accumulate all of the requesting party's resources and RPTs, noting that the user may have more than one dashboard session if this period is prolonged.

It is suggested that the RPT maximum TTL be 5 days, and that introspections can be cached at the RS to reduce AS load for a maximum of 2 days.

SECTION 4 References

[UMAGrant] https://tools.ietf.org/html/draft-maler-oauth-umagrant-00

[UMAFedAuthz] https://tools.ietf.org/html/draft-maler-oauth-umafedauthz-00

[OMTLS] https://tools.ietf.org/html/draft-ietf-oauth-mtls-14

[BCPOAuthSec] https://tools.ietf.org/html/draft-ietf-oauth-security-topics-12

[JWT] https://tools.ietf.org/pdf/rfc7519.pdf

[OAuth] https://tools.ietf.org/html/rfc6749

[JWTOAuth] https://tools.ietf.org/pdf/rfc7523.pdf JSON Web Token (JWT) Profile for OAuth 2.0 Client Authentication and Authorization Grants. (A profile of RFC7521.)

[RFC7662] https://tools.ietf.org/html/rfc7662 OAuth 2.0 Token Introspection

[ODynClient] https://tools.ietf.org/html/rfc7591 OAuth 2.0 Dynamic Client Registration Protocol

Authored by Mike Pegman (info@vinesolutions.co.uk), Vine Solutions Ltd for Origo Services Ltd Mar-Sept 2019
Contributed to Kantara Initiative by Kenneth May (kenneth.may@origo.com), Origo Services Ltd