TM Forum Specification

User Roles and Permissions Management API User Guide

TMF672

Team Approved Date: 28-May-2020

Release Status: Pre-production	Approval Status: Team Approved
Version 4.0.0	IPR Mode: RAND

NOTICE

This document and the information contained herein is provided on an "AS IS" basis and TM FORUM DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Direct inquiries to the TM Forum office:

4 Century Drive, Suite 100
Parsippany, NJ 07054, USA
Tel No. +1 973 944 5100
Fax No. +1 973 998 7196
TM Forum Web Page: www.tmforum.org

Table of Contents

List of Tables

N/A

Introduction

The following document is the specification of the REST API for user Role and Permission management. It includes the model definition as well as all available operations for creating user permissions to access manageable assets.

For the purpose of this API, the following definitions apply:

- A manageable asset is the realization of something that can be used and managed by users (e.g.: any of the resources created as part of a purchased product, a service provided to individuals, a block of personal data of an individual, a shopping cart entity…..).
- An user is the individual who can make use and manage the functions exposed by a given manageable asset. It can be the existing registered customer or any other individual who has been granted access to use and/or manage the asset
- A basic permission provides information regarding the access privileges of a given user over manageable assets (or different functions within each asset).
- A privilege defines an independent allowed access level over any of the operations that can be performed over a given asset (e.g.: CRUD on the different menu/function/UI elements)
- A user role is defined as the entity that defines a set of privileges covering various functions and/or manageable assets. When a user is assigned a given role then it is actually allocated all the privileges defined for that roletype and the corresponding permissions are created for that user

This API allows the following operations

Create new permission granted by an individual to another individual to access his owned manageable assets

Read existing permissions. It can be filtered for specific criteria (e.g.: date recorded, granter, …)

Read specific existing permission

Modify specific existing permission (total or partial update)

Read permissions recorded for a specific user as granter

Read permissions recorded for a specific user as grantee

Read permissions recorded for a specific asset

Create new user role

Read existing user roles

Read specific existing user role

Assigning specific user role to an individual (Party) over a given manageable asset.

Consuming this API must be done following a secured mechanism (e.g.: OAuth2.0) so that permissions to access manageable assets is only granted by consumers holding a valid authorization to operate on those manageable assets and grant permissions.

This API assumes that once a product is purchased by a customer (under a given account), as part of the product instantiation during the provisioning process, if a manageable asset is created under that product instance (e.g.: an eCare system registration, an account into a digital service platform, …) then this manageable assets will be assigned as owner to the individual that has admin rights over the customer and account entity under which the product was purchased. This association will be the first permission registered in the system (root permission) over the specific manageable asset, granting that individual (or another one if the purchasing process allows defining another admin for the manageable assets created) owner access to that asset and then the owner can use this API to grant access, with different access levels, to other individuals (users).

SAMPLE USE CASES

This section includes a set of main use cases that can be performed with this API. Additional use cases can be generated using the operations and resources defined in this specification.

Use Case 1: New permission created

Description

The main purpose of this use case is the creation of a new permission by an individual so that another individual is authorized to get access to some of his assets. For instance a user that owns a TV service can grant access to another user in order to make use of some of the functions within the service, for instance view only children movies, configure TV service or view documentaries.

Main Actors

- The owner of the assets (granter)
- The receiver of the permission (user)

Prerequisite: This API assumes that once a product is purchased by a customer (under a given account), as part of the product instantiation during the provisioning process, if a manageable asset is created under that product instance (e.g.: an eCare system registration, an account into a digital service platform, …) then this manageable assets will be assigned as owner to the individual that has admin rights over the customer and account entity under which the product was purchased.

Use Case Steps

1. The owner of the assets, sends a request to allocate a permission to another user on his assets

2. The Operator receives the permission creation request including the following minimum information :
   a. The period during which the permission is valid
   b. Impacted user that is being granted access
   c. Information on the manageable assets the user is granted access
   d. The level of access granted over each of the manageable assets (indicating the functions that can be accessed on the asset and the actions that can be performed on those functions)

3. The Operator confirms that the requestor is authorized to assign permissions on the referenced manageable assets (i.e.: either is the owner or has access to the assets with appropriate level). This could be based on just the requestor identifier or via a more sophisticated token-based authorization mechanisms

4. The Operator allocates the requested access level for the referred manageable assets to the individual that has been granted by the owner.

Example of API Usage in the Context of the Use Case

The following API interactions support the use case:

- The owner of the manageable assets consumes the service offered by the Operator to create a new permission record.

Success Outcome

After completion of these API interactions, the individual that has been granted access to the referred manageable assets can make use according to the access level granted.

Use Case 2: New User Role assigned to an individual

Description

The main purpose of this use case is the creation of a new user role indicating the access level authorized on a given set of manageable assets to whoever is allocated this role.

Main Actors

- The admin operator that generates user roles
- The owner of the assets (granter)
- The individual allocated a given role (user)

Prerequisite: This API assumes that once a product is purchased by a customer (under a given account), as part of the product instantiation during the provisioning process, if a manageable asset is created under that product instance (e.g.: an eCare system registration, an account into a digital service platform, …) then this manageable assets will be assigned as owner to the individual that has admin rights over the customer and account entity under which the product was purchased.

Use Case Steps

1. The admin operator, sends a request to create a new user role including the following minimum information :
    a. The level of access granted over a set of functions that can be accessed on an asset and the actions that can be performed on those functions

2. Once the user role is defined, the owner of the assets (or the admin), sends a request to allocate a permission to another user based on the user role definition, including the following minimum information:
    a. The period during which the permission is valid
    b. Impacted user that is being granted access
    c. Information on the manageable assets the user is granted access
    d. The user role allocated to the user on teh referenced asset

3. The Operator confirms that the requestor is authorized to assign permissions on the referenced manageable assets (i.e.: either is the owner or has access to the assets with appropriate level). This could be based on just the requestor identifier or via a more sophisticated token-based authorization mechanisms

4. The Operator allocates the requested access level for the referred manageable assets to the individual that has been granted by the owner.

Example of API Usage in the Context of the Use Case

The following API interactions support the use case:

- The admin operator consumes a service to create a new user role
- The owner of the manageable assets consumes the service offered by the Operator to create a new permission record based on assigning a role to a user over a given asset.

Success Outcome

After completion of these API interactions, the individual that has been granted access to the referred manageable assets can make use according to the access level defined for the allocated user role.

Support of polymorphism and extension patterns

Support of polymorphic collections and types and schema based extension is provided by means of a list of generic meta-attributes that we describe below. Polymorphism in collections occurs when entities inherit from base entities, for instance a BillingAccount and SettlementAccount inheriting properties from the abstract Account entity.

Generic support of polymorphism and pattern extensions is described in the TMF API Guidelines v3.0 Part 2 document.

The @type attribute provides a way to represent the actual class type of an entity. For example, within a list of Account instances some may be instances of BillingAccount where other could be instances of SettlementAccount. The @type gives this information. All resources and sub-resources of this API have a @type attributes that can be provided when this is useful.

The @referredType can be used within reference entities (like for instance an AccountRef object) to explicitly denote the actual entity type of the referred class. Notice that in reference entities the @type, when used, denotes the class type of the reference itself, such as BillingAccountRef or SettlementAccountRef, and not the class type of the referred object. However since reference classes are rarely sub-classed, @type is generally not useful in reference objects.

The @schemaLocation property can be used in resources to allow specifying user-defined properties of an Entity or to specify the expected *characteristics* of an entity.

The @baseType attribute gives a way to provide explicitly the base of class of a given resource that has been extended.

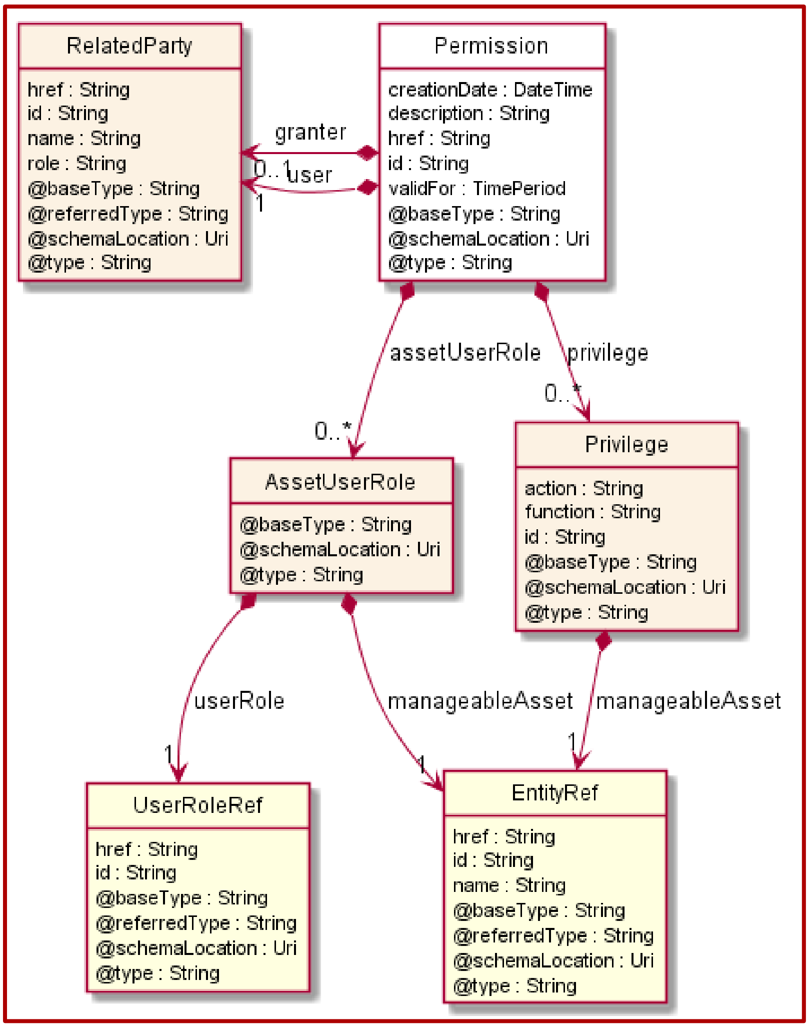
RESOURCE MODEL

Managed Entity and Task Resource Models

Permission resource

The Permission resource represents the entitlement given by an individual (granter) to another individual (user) to get access to a set of his owned manageable assets. One single permission resource can hold information referring to privileges granted for multiple manageable assets.

Resource model



Field descriptions

Permission fields

assetUserRole A list of asset user roles (AssetUserRole [*]). The AssetUserRole is the detailed information concerning an individual user role.

creationDate A date time (DateTime). Date when the payment was performed.

description A string. Text describing the contents of the payment.

granter A related party (RelatedParty). Related Entity reference. A related party defines party or party role linked to a specific entity.

href A string. Hypertext Reference of the permission.

id A string. Unique identifier of the permission.

privilege A list of privileges (Privilege [*]). A Privilege is a detailed information concerning an individual access entitlement.

user A related party (RelatedParty). Related Entity reference. A related party defines party or party role linked to a specific entity.

validFor A time period. The period for which the permission is valid.

AssetUserRole sub-resource

The AssetUserRole is the detailed information concerning an individual user role.

manageableAsset An entity reference (EntityRef). Entity reference schema to be use for all entityRef class.

userRole A user role reference (UserRoleRef). A UserRoleRef is a detailed information concerning an individual access entitlement.

Privilege sub-resource

A Privilege is a detailed information concerning an individual access entitlement.

action A string. Level of access granted as part of the permission.

function A string. Specific function that can be managed over a given asset.

id A string. Identifier of the privilege.

manageableAsset An entity reference (EntityRef). Entity reference schema to be use for all entityRef class.

RelatedParty sub-resource

Related Entity reference. A related party defines party or party role linked to a specific entity.

@referredType A string. The actual type of the target instance when needed for disambiguation.

href A string. Reference of the related entity.

id A string. Unique identifier of a related entity.

name A string. Name of the related entity.

role A string. Role played by the related party.

EntityRef relationship

Entity reference schema to be use for all entityRef class.

@referredType A string. The actual type of the target instance when needed for disambiguation.

href A string. Reference of the related entity.

id A string. Unique identifier of a related entity.

name A string. Name of the related entity.

UserRoleRef relationship

A UserRoleRef is a detailed information concerning an individual access entitlement.

@referredType A string. The actual type of the target instance when needed for disambiguation.

href A string. Hypertext Reference of the user role.

id A string. Unique identifier of the user role.

Json representation sample

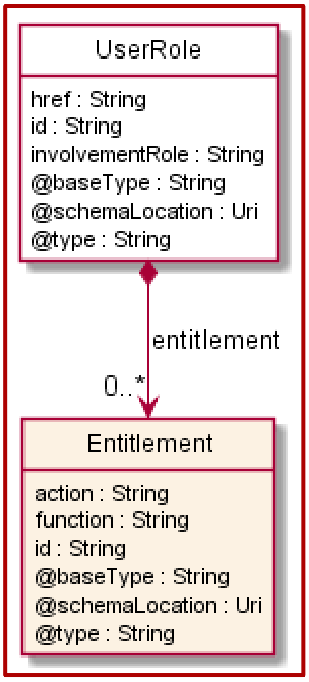
We provide below the json representation of an example of a 'Permission' resource object

```
{
   "id": "Prms-jnzgh456",
   "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-jnzgh456",
   "creationDate": "2017-11-01T09:37:29.961Z",
   "description": "superProfile Granted for user",
   "validFor": {
      "startDateTime": "2019-10-01T00:00:00.000Z",
      "endDateTime": "2019-10-31T23:59:59.000Z"
   },
   "@type": "Permission",
   "granter": {
      "id": "cclt-456745",
      "href": "https://host:port/partyManagement/v4/individual/cclt-456745",
      "name": "John Baker",
      "role": "customer advisor",
      "@type": "RelatedParty",
      "@referredType": "Individual"
   },
   "user": {
      "id": "cust-745712-A",
      "href": "https://host:port/partyManagement/v4/individual/cust-745712-A",
      "name": "Jack Marshall",
      "role": "client",
      "@type": "RelatedParty",
      "@referredType": "Individual"
   },
   "assetUserRole": [
      {
          "userRole": {
              "id": "413920",
              "href": "https://host:port/partyManagement/v4/userRole/413920",
              "@type": "UserRoleRef"
          },
          "manageableAsset": {
              "id": "12k-47",
              "@type": "ManageableAsset",
              "@baseType": "EntityRef",
              "@referredType": "Product",
              "product": {
                  "id": "UJF6-B61654"
              }
          }
      }
   ]
}
```

User Role resource

A UserRole defines access levels to operate over a given function that can be included in an asset.

Resource model



Field descriptions

UserRole fields

entitlement A list of entitlements (Entitlement [*]). An Entitlement defines access levels to operate over a given function that can be included in an asset.

href A string. Unique URI used to access to the userRole resource.

id A string. Unique identifier of the userRole.

involvementRole A string. Indication of the part that a user plays in its involvement with a manageable asset (product, service or resource).

Entitlement sub-resource

An Entitlement defines access levels to operate over a given function that can be included in an asset.

action A string. Level of access granted as part of the permission.

function A string. Specific function that can be managed over a given asset.

id A string. Identifier of the entitlement.

Json representation sample

We provide below the json representation of an example of a 'UserRole' resource object

```
{
    "id": "413920",
    "href": "http://serverlocation:port/userRolePermission/v4/userRole/413920",
    "involvementRole": "MyTV familyAdmin Profile",
    "@type": "UserRole",
    "entitlement": [
        {
            "id": "gdf-1324",
            "action": "R,W",
            "function": "SubAccounts creation Granted",
            "@type": "Entitlement"
        },
        {
            "id": "eq-120",
            "action": "add",
            "function": "Downloading is activated",
            "@type": "Entitlement"
        }
    ]
}
```

Notification Resource Models

8 notifications are defined for this API

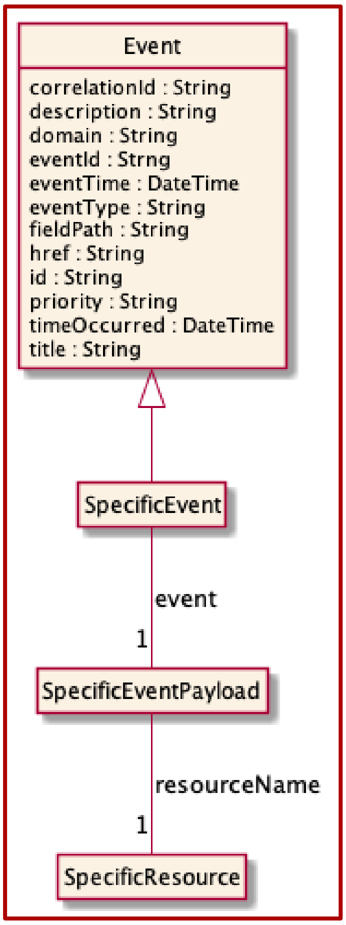
Notifications related to Permission:

- PermissionCreateEvent
- PermissionAttributeValueChangeEvent
- PermissionStateChangeEvent
- PermissionDeleteEvent

Notifications related to UserRole:

- UserRoleCreateEvent
- UserRoleAttributeValueChangeEvent
- UserRoleStateChangeEvent
- UserRoleDeleteEvent

The notification structure for all notifications in this API follow the pattern depicted by the figure below.
A notification event resource (depicted by "SpecificEvent" placeholder) is a sub class of a generic Event structure containing at least an id of the event occurrence (eventId), an event timestamp (eventTime), and the name of the resource (eventType).
This notification structure owns an event payload structure ("SpecificEventPayload" placeholder) linked to the resource concerned by the notification using the resource name as access field ("resourceName" placeholder).



Permission Create Event

Notification PermissionCreateEvent case for resource Permission

Json representation sample

We provide below the json representation of an example of a 'PermissionCreateEvent' notification event object

```
{
   "eventId":"00001",
   "eventTime":"2015-11-16T16:42:25-04:00",
   "eventType":"PermissionCreateEvent",
    "event": {
      "permission" :
         {-- SEE Permission RESOURCE SAMPLE --}
   }
}
```

Permission Attribute Value Change Event

Notification PermissionAttributeValueChangeEvent case for resource Permission

Json representation sample

We provide below the json representation of an example of a 'PermissionAttributeValueChangeEvent' notification event object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"PermissionAttributeValueChangeEvent",
     "event": {
        "permission" :
            {-- SEE Permission RESOURCE SAMPLE --}
    }
}
```

Permission State Change Event

Notification PermissionStateChangeEvent case for resource Permission

Json representation sample

We provide below the json representation of an example of a 'PermissionStateChangeEvent' notification event object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"PermissionStateChangeEvent",
     "event": {
        "permission" :
            {-- SEE Permission RESOURCE SAMPLE --}
    }
}
```

Permission Delete Event

Notification PermissionDeleteEvent case for resource Permission

Json representation sample

We provide below the json representation of an example of a 'PermissionDeleteEvent' notification event object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"PermissionDeleteEvent",
     "event": {
        "permission" :
            {-- SEE Permission RESOURCE SAMPLE --}
    }
}
```

User Role Create Event

Notification UserRoleCreateEvent case for resource UserRole

Json representation sample

We provide below the json representation of an example of a 'UserRoleCreateEvent' notification event object

```
{
   "eventId":"00001",
   "eventTime":"2015-11-16T16:42:25-04:00",
   "eventType":"UserRoleCreateEvent",
    "event": {
      "userRole" :
         {-- SEE UserRole RESOURCE SAMPLE --}
   }
}
```

User Role Attribute Value Change Event

Notification UserRoleAttributeValueChangeEvent case for resource UserRole

Json representation sample

We provide below the json representation of an example of a 'UserRoleAttributeValueChangeEvent' notification event object

```
{
   "eventId":"00001",
   "eventTime":"2015-11-16T16:42:25-04:00",
   "eventType":"UserRoleAttributeValueChangeEvent",
    "event": {
      "userRole" :
         {-- SEE UserRole RESOURCE SAMPLE --}
   }
}
```

User Role State Change Event

Notification UserRoleStateChangeEvent case for resource UserRole

Json representation sample

We provide below the json representation of an example of a 'UserRoleStateChangeEvent' notification event object

```
{
   "eventId":"00001",
   "eventTime":"2015-11-16T16:42:25-04:00",
   "eventType":"UserRoleStateChangeEvent",
    "event": {
       "userRole" :
           {-- SEE UserRole RESOURCE SAMPLE --}
   }
}
```

User Role Delete Event

Notification UserRoleDeleteEvent case for resource UserRole

Json representation sample

We provide below the json representation of an example of a 'UserRoleDeleteEvent' notification event object

```
{
    "eventId":"00001",
    "eventTime":"2015-11-16T16:42:25-04:00",
    "eventType":"UserRoleDeleteEvent",
     "event": {
        "userRole" :
            {-- SEE UserRole RESOURCE SAMPLE --}
    }
}
```

API OPERATIONS

Remember the following Uniform Contract:

Operation on Entities	Uniform API Operation	Description
Query Entities	GET Resource	GET must be used to retrieve a representation of a resource.
Create Entity	POST Resource	POST must be used to create a new resource
Partial Update of an Entity	PATCH Resource	PATCH must be used to partially update a resource
Complete Update of an Entity	PUT Resource	PUT must be used to completely update a resource identified by its resource URI
Remove an Entity	DELETE Resource	DELETE must be used to remove a resource
Execute an Action on an Entity	POST on TASK Resource	POST must be used to execute Task Resources
Other Request Methods	POST on TASK Resource	GET and POST must not be used to tunnel other request methods.

Filtering and attribute selection rules are described in the TMF REST Design Guidelines.

Notifications are also described in a subsequent section.

Operations on Permission

List permissions

GET /permission?fields=...&{filtering}

Description

This operation list permission entities.
Attribute selection is enabled for all first level attributes.
Filtering may be available depending on the compliance level supported by an implementation.

Usage Samples

Here is an example of a request for retrieving a list of Permission(s). In this example, the returned permissions are based either on privilege and assetUserRole.

Request

```
GET serverRoot/payment/v4/permission?user.id=34
Accept: application/json
```

Response

```
200

[
  {
    "id": "Prms-jnzgh456",
    "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-jnzgh456",
    "creationDate": "2017-11-01T09:37:29.961Z",
    "description": "superProfile Granted for user",
    "validFor": {
      "startDateTime": "2019-10-01T00:00:00.000Z",
      "endDateTime": "2019-10-31T23:59:59.000Z"
    },
    "@type": "Permission",
    "granter": {
      "id": "cclt-456745",
      "href": "https://host:port/partyManagement/v4/individual/cclt-456745",
      "name": "John Baker",
      "role": "customer advisor",
      "@type": "RelatedParty",
      "@referredType": "Individual"
    },
    "user": {
      "id": "34",
      "href": "https://host:port/partyManagement/v4/individual/34",
      "name": "Jack Marshall",
      "role": "client",
      "@type": "RelatedParty",
      "@referredType": "Individual"
    },
    "assetUserRole": [
      {
         "userRole": {
            "id": "413920",
            "href": "https://host:port/partyManagement/v4/userRole/413920",
            "@type": "UserRoleRef"
         },
         "manageableAsset": {
            "id": "12k-47",
            "@type": "ManageableAsset",
            "@baseType": "EntityRef",
            "@referredType": "Product",
            "product": {
               "id": "UJF6-B61654"
            }
         }
      }
    ]
  },
  {
    "id": "Prms-obrfge-654",
    "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-obrfge-654",
    "creationDate": "2019-11-13T09:37:29.000Z",
    "validFor": {
      "startDateTime": "2019-11-14T00:00:00.000Z",
      "endDateTime": "2020-11-31T23:59:59.000Z"
    },
    "@type": "Permission",
    "user": {
      "id": "34",
      "href": "https://host:port/partyManagement/v4/individual/34",
      "name": "Jack Marshall",
      "role": "client",
      "@type": "RelatedParty",
      "@referredType": "Individual"
    },
    "privilege": [
      {
         "id": "621458",
         "function": "IPTV access",
         "action": "enabled",
         "userRole": {
            "id": "413920",
            "href": "https://host:port/partyManagement/v4/userRole/413920",
            "@type": "UserRoleRef"
         },
         "manageableAsset": {
```

```
          "id": "a123",
          "name": "mobile line",
          "@type": "ManageableAsset",
          "@baseType": "EntityRef",
          "@referredType": "Product",
          "product": {
            "id": "FYF3543"
          }
        }
      }
    ]
  }
]
```

Retrieve permission

GET /permission/{id}?fields=...&{filtering}

Description

This operation retrieves a permission entity.
Attribute selection is enabled for all first level attributes.
Filtering on sub-resources may be available depending on the compliance level supported by an implementation.

Usage Samples

Here is an example of a request for a permission.

Request

```
GET serverRoot/payment/v4/permission/Prms-jnzgh456
Accept: application/json
```

Response

```
200

{
  "id": "Prms-jnzgh456",
  "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-jnzgh456",
  "creationDate": "2017-11-01T09:37:29.961Z",
  "description": "superProfile Granted for user",
  "validFor": {
    "startDateTime": "2019-10-01T00:00:00.000Z",
    "endDateTime": "2019-10-31T23:59:59.000Z"
  },
  "@type": "Permission",
  "granter": {
    "id": "cclt-456745",
    "href": "https://host:port/partyManagement/v4/individual/cclt-456745",
    "name": "John Baker",
    "role": "customer advisor",
    "@type": "RelatedParty",
    "@referredType": "Individual"
  },
  "user": {
    "id": "cust-745712-A",
    "href": "https://host:port/partyManagement/v4/individual/cust-745712-A",
    "name": "Jack Marshall",
    "role": "client",
    "@type": "RelatedParty",
    "@referredType": "Individual"
  },
  "assetUserRole": [
    {
      "userRole": {
        "id": "413920",
        "href": "https://host:port/partyManagement/v4/userRole/413920",
        "@type": "UserRoleRef"
      },
      "manageableAsset": {
        "id": "12k-47",
        "@type": "ManageableAsset",
        "@baseType": "EntityRef",
        "@referredType": "Product",
        "product": {
          "id": "UJF6-B61654"
        }
      }
    }
  ]
}
```

Create permission

```
POST /permission
```

Description

This operation creates a permission entity.

Mandatory and Non Mandatory Attributes

The following tables provide the list of mandatory and non mandatory attributes when creating a Permission, including any possible rule conditions and applicable default values. Notice that it is up to an implementer to add additional mandatory attributes.

Mandatory Attributes	Rule
validFor	
user	

Non Mandatory Attributes	Rule
assetUserRole	
creationDate	
description	
granter	
privilege	

Additional Rules

The following table provides additional rules indicating mandatory fields in sub-resources or relationships when creating a Permission resource.

Context	Mandatory Sub-Attributes
privilege	manageableAsset, action, function
assetUserRole	manageableAsset, userRole

Usage Samples

Here is an example of a request for a permission creation, based on the use of a formerly created userRole

Request

```
POST serverRoot/payment/v4/permission
Content-Type: application/json

{
   "creationDate": "2017-11-01T09:37:29.961Z",
   "description": "superProfile Granted for user",
   "validFor": {
      "startDateTime": "2019-10-01T00:00:00.000Z",
      "endDateTime": "2019-10-31T23:59:59.000Z"
   },
   "@type": "Permission",
   "granter": {
      "id": "cclt-456745",
      "href": "https://host:port/partyManagement/v4/individual/cclt-456745",
      "name": "John Baker",
      "role": "customer advisor",
      "@type": "RelatedParty",
      "@referredType": "Individual"
   },
   "user": {
      "id": "cust-745712-A",
      "href": "https://host:port/partyManagement/v4/individual/cust-745712-A",
      "name": "Jack Marshall",
      "role": "client",
      "@type": "RelatedParty",
      "@referredType": "Individual"
   },
   "assetUserRole": [
      {
          "userRole": {
              "id": "413920",
              "href": "https://host:port/partyManagement/v4/userRole/413920",
              "@type": "UserRoleRef"
          },
          "manageableAsset": {
              "id": "12k-47",
              "@type": "ManageableAsset",
              "@baseType": "EntityRef",
              "@referredType": "Product",
              "product": {
                  "id": "UJF6-B61654"
              }
          }
      }
   ]
}
```

Response

```
201

{
    "id": "Prms-jnzgh456",
    "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-jnzgh456",
    "creationDate": "2017-11-01T09:37:29.961Z",
    "description": "superProfile Granted for user",
    "validFor": {
        "startDateTime": "2019-10-01T00:00:00.000Z",
        "endDateTime": "2019-10-31T23:59:59.000Z"
    },
    "@type": "Permission",
    "granter": {
        "id": "cclt-456745",
        "href": "https://host:port/partyManagement/v4/individual/cclt-456745",
        "name": "John Baker",
        "role": "customer advisor",
        "@type": "RelatedParty",
        "@referredType": "Individual"
    },
    "user": {
        "id": "cust-745712-A",
        "href": "https://host:port/partyManagement/v4/individual/cust-745712-A",
        "name": "Jack Marshall",
        "role": "client",
        "@type": "RelatedParty",
        "@referredType": "Individual"
    },
    "assetUserRole": [
        {
             "userRole": {
                  "id": "413920",
                  "href": "https://host:port/partyManagement/v4/userRole/413920",
                  "@type": "UserRoleRef"
             },
             "manageableAsset": {
                  "id": "12k-47",
                  "@type": "ManageableAsset",
                  "@baseType": "EntityRef",
                  "@referredType": "Product",
                  "product": {
                       "id": "UJF6-B61654"
                  }
             }
        }
    ]
}
```

Here is an example of a request for a permission creation, made 'on the fly' with a privilege declaration.

Request

```
POST serverRoot/payment/v4/permission
Content-Type: application/json

{
    "id": "Prms-obrfge-654",
    "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-obrfge-654",
    "creationDate": "2019-11-13T09:37:29.000Z",
    "validFor": {
         "startDateTime": "2019-11-14T00:00:00.000Z",
         "endDateTime": "2020-11-31T23:59:59.000Z"
    },
    "@type": "Permission",
    "user": {
         "id": "34",
         "href": "https://host:port/partyManagement v4/individual/34",
         "name": "Jack Marshall",
         "role": "client",
         "@type": "RelatedParty",
      "@referredType": "Individual"
   },
   "privilege": [
      {
         "id": "621458",
         "function": "IPTV access",
         "action": "enabled",
         "manageableAsset": {
            "id": "a123",
            "name": "mobile line",
            "@type": "ManageableAsset",
            "@baseType": "EntityRef",
            "@referredType": "Product",
            "product": {
               "id": "FYF3543"
            }
         }
      }
   ]
}
```

Response

```
201

{
   "id": "Prms-obrfge-654",
   "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-obrfge-654",
   "creationDate": "2019-11-13T09:37:29.000Z",
   "validFor": {
      "startDateTime": "2019-11-14T00:00:00.000Z",
      "endDateTime": "2020-11-31T23:59:59.000Z"
   },
   "@type": "Permission",
   "user": {
      "id": "34",
      "href": "https://host:port/partyManagement/v4/individual/34",
      "name": "Jack Marshall",
      "role": "client",
      "@type": "RelatedParty",
      "@referredType": "Individual"
   },
   "privilege": [
      {
         "id": "621458",
         "function": "IPTV access",
         "action": "enabled",
         "manageableAsset": {
            "id": "a123",
            "name": "mobile line",
            "@type": "ManageableAsset",
            "@baseType": "EntityRef",
        "@referredType": "Product",
        "product": {
          "id": "FYF3543"
        }
      }
    }
  ]
}
```

Patch permission

PATCH /permission/{id}

Description

This operation allows partial updates of a permission entity. Support of json/merge (https://tools.ietf.org/html/rfc7386) is mandatory, support of json/patch (http://tools.ietf.org/html/rfc5789) is optional.

Note: If the update operation yields to the creation of sub-resources or relationships, the same rules concerning mandatory sub-resource attributes and default value settings in the POST operation applies to the PATCH operation. Hence these tables are not repeated here.

Patchable and Non Patchable Attributes

The tables below provide the list of patchable and non patchable attributes, including constraint rules on their usage.

Patchable Attributes	Rule
assetUserRole	
description	
granter	
privilege	
user	
validFor	

Non Patchable Attributes	Rule
id	
href	
creationDate	

Usage Samples

Here's an example of a request for updating a permission. In this example the initial permission contains 2 provileges : 621458 (IPTV access) and 784513 (Premium mobile Gaming access). With this patch operation, the second one is removed from the permission.

Request

```
PATCH serverRoot/payment/v4/permission/Prms-obrfge-654
Content-Type: application/json-patch+json

[
  {
    "op": "remove",
    "path": "/privilege?privilege.id=784513"
  }
]
```

Response

```
200

{
  "id": "Prms-obrfge-654",
  "href": "http://serverlocation:port/userRolePermission/v4/permission/Prms-obrfge-654",
  "creationDate": "2019-11-13T09:37:29.000Z",
  "validFor": {
    "startDateTime": "2019-11-14T00:00:00.000Z",
    "endDateTime": "2020-11-31T23:59:59.000Z"
  },
  "@type": "Permission",
  "user": {
    "id": "34",
    "href": "https://host:port/partyManagement/v4/individual/34",
    "name": "Jack Marshall",
    "role": "client",
    "@type": "RelatedParty",
    "@referredType": "Individual"
  },
  "privilege": [
    {
      "id": "621458",
      "function": "IPTV access",
      "action": "enabled",
      "userRole": {
        "id": "413920",
        "href": "https://host:port/partyManagement/v4/userRole/413920",
        "@type": "UserRoleRef"
      },
      "manageableAsset": {
        "id": "a123",
        "name": "mobile line",
        "@type": "ManageableAsset",
        "@baseType": "EntityRef",
        "@referredType": "Product",
        "product": {
          "id": "FYF3543"
        }
      }
    }
  ]
}
```

Operations on User Role

List user roles

GET /userRole?fields=...&{filtering}

Description

This operation list user role entities.
Attribute selection is enabled for all first level attributes.
Filtering may be available depending on the compliance level supported by an implementation.

Usage Samples

Here is an example of a request for retrieving a list of userRole(s).

Request

```
GET serverRoot/payment/v4/userRole?fields=id,href,name,description,state&relatedParty.id=34
Accept: application/json
```

Response

```
200

[
  {
    "id": "UR001",
    "href": "http://serverlocation:port/userRolePermission/v4/userRole/UR001",
    "involvementRole": "owner",
    "entitlement": [
      {
        "id": "4319559",
        "function": "all",
        "action": "R&W",
        "@type": "Entitlement"
      }
    ]
  },
  {
    "id": "UR073",
    "href": "http://serverlocation:port/userRolePermission/v4/userRole/UR073",
    "involvementRole": "member",
    "entitlement": [
      {
        "id": "7851245",
        "function": "all",
        "action": "R/O",
        "@type": "Entitlement"
      }
    ]
  },
  {
    "id": "UR210",
    "href": "http://serverlocation:port/userRolePermission/v4/userRole/UR210",
    "involvementRole": "configure IPTV and watch news",
    "entitlement": [
      {
        "id": "3621587",
        "function": "Netflix configuration",
        "action": "R&W",
        "@type": "Entitlement"
      },
      {
        "id": "301248",
        "function": "Sport basic package",
        "action": "watch",
        "@type": "Entitlement"
      }
    ]
  }
]
```

Retrieve user role

GET /userRole/{id}?fields=...&{filtering}

Description

This operation retrieves a user role entity.
Attribute selection is enabled for all first level attributes.
Filtering on sub-resources may be available depending on the compliance level supported by an implementation.

Usage Samples

Here is an example of a request for a userRole.

Request

```
GET serverRoot/payment/v4/userRole/413920
Accept: application/json
```

Response

```
200

{
   "id": "413920",
   "href": "http://serverlocation:port/userRolePermission/v4/userRole/413920",
   "involvementRole": "MyTV familyAdmin Profile",
   "@type": "UserRole",
   "entitlement": [
      {
         "id": "gdf-1324",
         "action": "R,W",
         "function": "SubAccounts creation Granted",
         "@type": "Entitlement"
      },
      {
         "id": "eq-120",
         "action": "add",
         "function": "Downloading is activated",
         "@type": "Entitlement"
      }
   ]
}
```

Create user role

POST /userRole

Description

This operation creates a user role entity.

Mandatory and Non Mandatory Attributes

The following tables provide the list of mandatory and non mandatory attributes when creating a UserRole, including any possible rule conditions and applicable default values. Notice that it is up to an implementer to add additional mandatory attributes.

Mandatory Attributes	Rule
involvementRole	
entitlement	

Non Mandatory Attributes	Rule

Usage Samples

Here is an example of a request for creating a userRole.

Request

```
POST serverRoot/payment/v4/userRole
Content-Type: application/json

{
   "involvementRole": "MyTV familyAdmin Profile",
   "@type": "UserRole",
   "entitlement": [
      {
         "id": "gdf-1324",
         "action": "R,W",
         "function": "SubAccounts creation Granted",
         "@type": "Entitlement"
      },
      {
         "id": "eq-120",
         "action": "add",
         "function": "Downloading is activated",
         "@type": "Entitlement"
      }
   ]
}
```

Response

```
201

{
   "id": "413920",
   "href": "http://serverlocation:port/userRolePermission/v4/userRole/413920",
   "involvementRole": "MyTV familyAdmin Profile",
   "@type": "UserRole",
   "entitlement": [
      {
         "id": "gdf-1324",
         "action": "R,W",
         "function": "SubAccounts creation Granted",
         "@type": "Entitlement"
      },
      {
         "id": "eq-120",
         "action": "add",
         "function": "Downloading is activated",
         "@type": "Entitlement"
      }
   ]
}
```

Patch user role

PATCH /userRole/{id}

Description

This operation allows partial updates of a user role entity. Support of json/merge (https://tools.ietf.org/html/rfc7386) is mandatory, support of json/patch (http://tools.ietf.org/html/rfc5789) is optional.

Note: If the update operation yields to the creation of sub-resources or relationships, the same rules concerning mandatory sub-resource attributes and default value settings in the POST operation applies to the PATCH operation. Hence these tables are not repeated here.

Patchable and Non Patchable Attributes

The tables below provide the list of patchable and non patchable attributes, including constraint rules on their usage.

Patchable Attributes	Rule
entitlement	
involvementRole	

Non Patchable Attributes	Rule
id	
href	

Usage Samples

Here's an example of a request for updating a userRole. In this example, the entitlement 632514 is updated with an action set to readOnly. The 2 other entitlements (12014 and 960124) are not modified.

Request

```
PATCH serverRoot/payment/v4/userRole/413920
Content-Type: application/json-patch+json

[
   {
      "op": "replace",
      "path": "/entitlement/action?entitlement.id=632514",
      "value": "readOnly"
   }
]
```

Response

```
200

{
   "id": "413920",
   "href": "http://serverlocation:port/userRolePermission/v4/userRole/413920",
   "involvementRole": "MyTV SPORT+ familyAdmin Profile",
   "@type": "UserRole",
   "entitlement": [
      {
         "id": "632514",
         "action": "readOnly",
         "function": "SubAccounts creation Granted",
         "@type": "Entitlement"
      },
      {
         "id": "12014",
         "action": "add",
         "function": "Downloading is activated",
         "@type": "Entitlement"
      },
      {
         "id": "960124",
         "action": "add",
         "function": "Access to sport channels",
         "@type": "Entitlement"
      }
   ]
}
```

API NOTIFICATIONS

For every single of operation on the entities use the following templates and provide sample REST notification POST calls.

It is assumed that the Pub/Sub uses the Register and UnRegister mechanisms described in the REST Guidelines reproduced below.

Register listener

POST /hub

Description

Sets the communication endpoint address the service instance must use to deliver information about its health state, execution state, failures and metrics. Subsequent POST calls will be rejected by the service if it does not support multiple listeners. In this case DELETE /api/hub/{id} must be called before an endpoint can be created again.

Behavior

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 409 if request is not successful.

Usage Samples

Here's an example of a request for registering a listener.

Request
POST /api/hub Accept: application/json {"callback": "http://in.listener.com"}
Response
201 Content-Type: application/json Location: /api/hub/42 {"id":"42","callback":"http://in.listener.com","query":null}

Unregister listener

DELETE /hub/{id}

Description

Clears the communication endpoint address that was set by creating the Hub.

Behavior

Returns HTTP/1.1 status code 204 if the request was successful.

Returns HTTP/1.1 status code 404 if the resource is not found.

Usage Samples

Here's an example of a request for un-registering a listener.

Request
DELETE /api/hub/42 Accept: application/json
Response
204

Publish Event to listener

POST /client/listener

Description

Clears the communication endpoint address that was set by creating the Hub.

Provides to a registered listener the description of the event that was raised. The /client/listener url is the callback url passed when registering the listener.

Behavior

Returns HTTP/1.1 status code 201 if the service is able to set the configuration.

Usage Samples

Here's an example of a notification received by the listener. In this example “EVENT TYPE” should be replaced by one of the notification types supported by this API (see Notification resources Models section) and EVENT BODY refers to the data structure of the given notification type.

Request

```
POST /client/listener
Accept: application/json

{
   "event": {
            EVENT BODY
         },
   "eventType": "EVENT_TYPE"
}
```

Response

```
201
```

For detailed examples on the general TM Forum notification mechanism, see the TMF REST Design Guidelines.

Acknowledgements

Version History

Version Number	Date	Release led by:	Description
1.0	04/15/2017	Pierre Gauthier TM Forum pgauthier@tmforum.org Mariano Belaunde Orange Labs	First Release of the Document.
2.0	11/06/2018	Mariano Belaunde Orange Labs	Alignment with Guidelines 3.0
4.0.0	28-May-2020	Pierre Gauthier -TM Forum pgauthier@tmforum.org Grégoire Laurent-Orange gregoire.laurent@orange.com Ludovic Robert-Orange ludovic.robert@orange.com	Version 4.0 of the API REST

Release History

Release Number	Date	Release led by:	Description
Pre-production	28-May-2020	Pierre Gauthier -TM Forum Grégoire Laurent-Orange Ludovic Robert-Orange	Version 4.0 of the API REST