Documentation - privman-api - 10.8.0

Table of Contents

Privilege Manager API

With release 10.8 of Privilege Manager Thycotic is providing a customer facing API for access to public product endpoints.

Thycotic is following the OpenAPI standard and our customers are offered the standard Swagger UI interface to interact with and learn how to use Privilege Manager's public API endpoints.

Installing the API

The Privilege Manager Application Programming Interface packages are installed through the main Privilege Manager console. Navigate to **Admin | Setup** and follow the steps as documented under Upgrades

> **Note**: Cloud instances have the API installed by default, just like other features, such as foreign system connectors etc.

Creating an API Client User

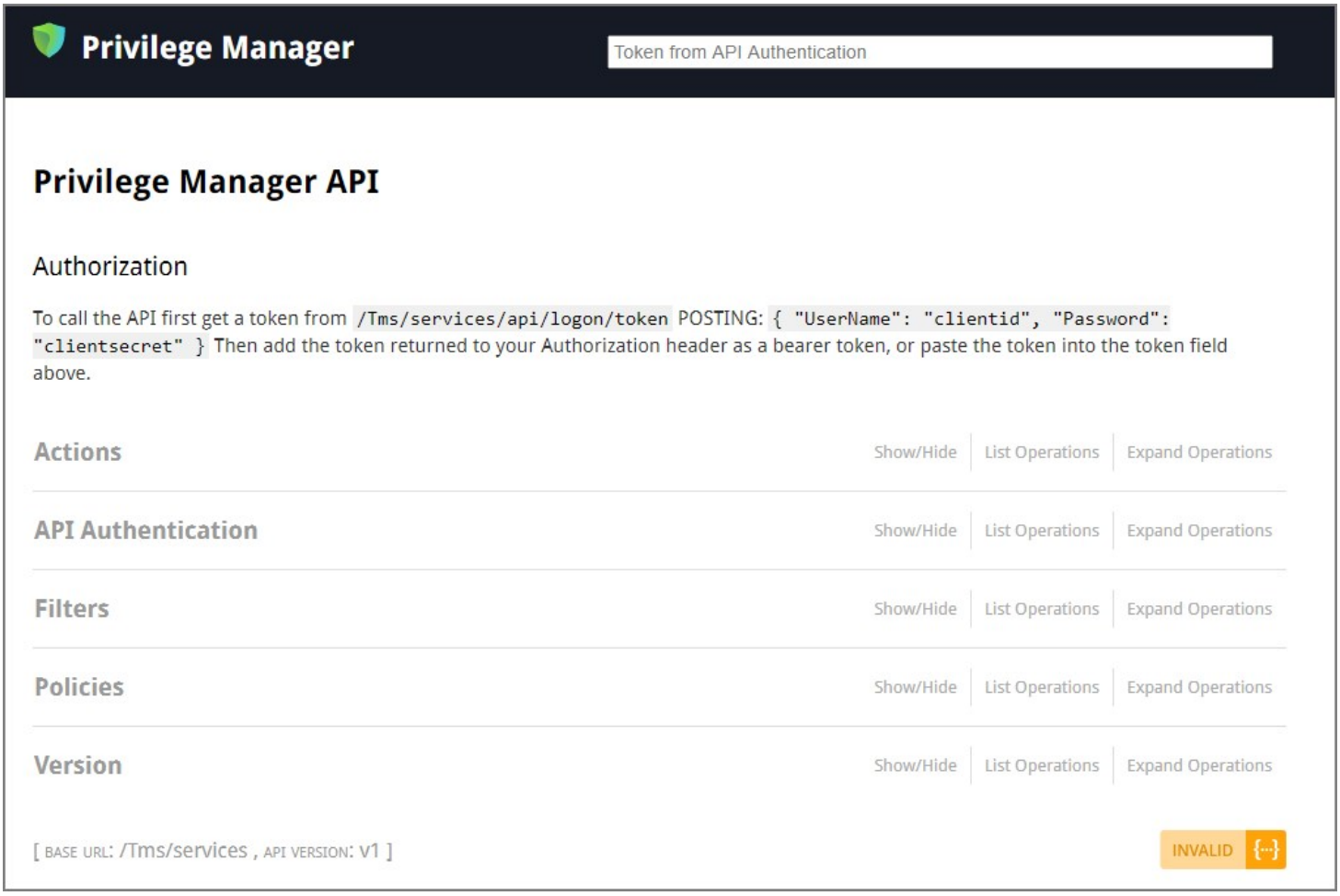
Before you can access the API and start using the API endpoints, you need to setup an API Client User in the Privilege Manager Console. For details on the API Client User setup refer to the Users topic in the main Privilege Manager documentation, specifically access How to Manually Add API Client Users and the Add Roles to a User information.

Refer to the Roles and Application Roles topics to learn more about the type of roles required to execute tasks in Privilege Manager. For example, to make changes to policies, that API Client User needs to be added to an administrator role (macOS, Windows, or full Privilege Manager Admin). To simply read a policy, filter, or action, the Privilege Manager Users role is sufficient.

Accessing the API

To access the Privilege Manager API,

1. in the Privilege Manager Console in the upper right-hand corner of the page, navigate to the **Help** icon.
2. Select **API Reference**.



The standard URL to your API is for

- on-prem: `https://myserver.example.com/Tms/services/swagger/ui/index`
- cloud: `https://mycompany.privilegemanagercloud.com/Tms/services/swagger/ui/index`

API Authentication

To call the API first user need to get a token from `.../Tms/services/api/logon/token`.

You will need to post a request message with the following details:

```
POST /api/logon/token
{
  "Password": "string",
  "UserName": "string"
}
```

Parameter	Value
Password	clientsecret
UserName	clientid

The auth token will be returned to you. Copy and paste that token into the **Token from API Authentication** field in the header section of the API Reference page.

> **Note**: **DO NOT** hit Enter, just paste the token text and access the available API methods.

You may also add the returned token to your Authorization header as a bearer token.

Refer to "How to Manually Add API Client Users" and "Add Roles to a User" to setup your API Client User and to add that user to the Privilege Manager Administrators role.

Actions

Actions can be retrieved from the instance through either a GetAll call or by requesting a specific itemId.

actions/GetAll

GET method to request all actions currently available on your Privilege Manager instance.

```
GET /api/v1/actions/GetAll
{ }
```

actions/

GET method to request a specific action.

```
GET /api/v1/actions/{itemId}
```

Parameter	Value
itemId	A 32-bit character string representing a specific action, for example: `54bfa458-bdfc-4e1b-8033-9c7888179f6c`. This represents the **Add Administrative Rights** action in Privilege Manager.

Filters

Filters can be retrieved from the instance through either a GetAll call or by requesting a specific itemId.

filters/GetAll

GET method to request all filters currently available on your Privilege Manager instance.

```
GET /api/v1/filters/GetAll
{ }
```

filters/

GET method to request a specific filter.

```
GET /api/v1/filters/{itemId}
```

Parameter	Value
itemId	A 32-bit character string representing a specific filter, for example: `df207907-9f9a-4777-afea-ff786d5399f2`. This represents the **Administrative Rights Required Application Compatibility Filter** read-only filter template in Privilege Manager.

Policies

Policies can be retrieved from the instance through either a GetAll call or by requesting a specific itemId.

Policies can also be created via various available post methods

GET

policies/GetAll

GET method to request all policies currently available in your Privilege Manager instance.

```
GET /api/v1/policies/GetAll
{ }
```

policies/

GET method to request a specific policy from your Privilege Manager instance.

```
GET /api/v1/policies/{itemId}
```

Parameter	Value
itemId	A 32-bit character string representing a specific policy, for example: `44290cc3-9c86-4995-aa33-79b8ee74daf7`. This represents the **Elevate Privilege Manager Remove Programs Utility Policy** read-only policy template in Privilege Manager.

POST

policies/new

POST method to create a new policy.

```
POST /api/v1/policies/new
{
  "Name": "string",
  "Description": "string",
  "OS": "string"
}
```

Parameter	Value
Name	Sensible name to easily find and place the newly created policy.
Description	Sensible description to correctly identify the newly created policy.
OS	This can be either `win` for Windows or `mac` for MacOS.

policies/

POST method to add filters to an existing policy.

```
POST /api/v1/policies/{policyId}/add-filters
{
  "Filters": [
    {
      "Id": "string",
      "FilterRole": 1
    }
```

```
  ]
}
```

Parameter	Value
Filters	An `Array[FilterModel]`, can be used to add multiple filters at the same time.
Id	Filter Ids, 32-bit character string representing a specific filter, to be added to the policy.
FilterRole	Public enum, where 1=ApplicationTarget, 2=Inclusion, and 3=Exclusion.

policies/

POST method to remove filters from an existing policy.

```
POST /api/v1/policies/{policyId}/remove-filters
{
  "Filters": [
    {
      "Id": "string",
      "FilterRole": 1
    }
  ]
}
```

Parameter	Value
Filters	An `Array[FilterModel]`, can be used to remove multiple filters at the same time.
Id	Filter Ids, 32-bit character string representing a specific filter, to be removed from the policy.
FilterRole	Public enum, where 1=ApplicationTarget, 2=Inclusion, and 3=Exclusion.

policies/

POST method to add actions to a specific policy.

```
POST /api/v1/policies/{policyId}/add-actions
{
  "Actions": [
    {
      "Id": "string",
      "IsChildAction": true
    }
  ]
}
```

Parameter	Value
Actions	An `(Array[ActionModel]`, can be used to add multiple actions at the same time.
Id	Action Ids, 32-bit character string representing a specific action, to be added from the policy.
IsChildAction	Boolean that can be set to `true` or `false` to indicate if the action is considered a child or not.

policies/

POST method to remove actions from a specific policy.

```
POST /api/v1/policies/{policyId}/remove-actions
{
  "Actions": [
    {
      "Id": "string",
      "IsChildAction": true
    }
  ]
}
```

Parameter	Value
Actions	An `(Array[ActionModel]`, can be used to remove multiple actions at the same time.
Id	Action Ids, 32-bit character string representing a specific action, to be removed from the policy.
IsChildAction	Boolean that can be set to `true` or `false` to indicate if the action is considered a child or not.

policies/

POST method to change the status of a specific policy from inactive to active or back.

```
GET /api/v1/policies/{policyId}/enable/{enable}
```

Parameter	Value
policyId	A 32-bit character string representing a specific policy, for example: `44290cc3-9c86-4995-aa33-79b8ee74daf7`. This represents the **Elevate Privilege Manager Remove Programs Utility Policy** read-only policy template in Privilege Manager.
enable	Boolean that can be set to `true` or `false` to set the policy to active or inactive. By default newly created policies are inactive and need to be enabled.

Version

GET method to request the current version information of the API.

```
GET /api/v1/version/Current
```

Changelog

Journal of chronological changes to the Privilege Manager API documentation:

August 2020

Privilege Manager 10.8, API v1: Initial public API release with supporting API Reference and documentation.