One Identity Manager

CyberArk Solution Accelerator

April, 2021

V 1.2

Contents

[Introduction 3](#_Toc69196694)

[Synchronized SCIM Objects 3](#_Toc69196695)

[User 3](#_Toc69196696)

[Group 4](#_Toc69196697)

[Container 4](#_Toc69196698)

[PrivilegedData 4](#_Toc69196699)

[ContainerPermissions 4](#_Toc69196700)

[Data Model 5](#_Toc69196701)

[System 5](#_Toc69196702)

[User 5](#_Toc69196703)

[Group 5](#_Toc69196704)

[Container 6](#_Toc69196705)

[PrivilegedData 6](#_Toc69196706)

[ContainerPermissions 7](#_Toc69196707)

[Configuration 9](#_Toc69196708)

[Processes and Scripts 11](#_Toc69196709)

[Processes 11](#_Toc69196710)

[Scripts 12](#_Toc69196711)

[Record Samples 13](#_Toc69196712)

[User 13](#_Toc69196713)

[Group 14](#_Toc69196714)

[Container 15](#_Toc69196715)

[PrivilegedData 16](#_Toc69196717)

[ContainerPermission 17](#_Toc69196718)

Introduction

Cyber Ark solution accelerator provides synchronization project, schema extension and configuration parameters to speed up development of the Cyber Ark integration.

This solution accelerator uses Cyber Ark SCIM server to communicate with Cyber Ark.

Requirements

SCIM server needs to be installed

Synchronized SCIM Objects

User

Users records with field “**urn:ietf:params:scim:schemas:cyberark:1.0:User~ldapFullDN**” populated refer to an Active Directory account. Records with this field empty refer to Cyber Ark local accounts.

Group

Groups records with field “**urn:ietf:params:scim:schemas:cyberark:1.0:Group~ldapFullDN**”   
populated refer to an Active Directory group. This record is called “DirectoryGroup”

Records with this field empty refer to Cyber Ark local groups and called “LocalGroup”

Container

Container corresponds to Cyber Ark object “Safe”.   
Safe in Cyber Ark contains PrivilegedData – privileged accounts, documents etc…  
Safe also contains ContainerPermissions.

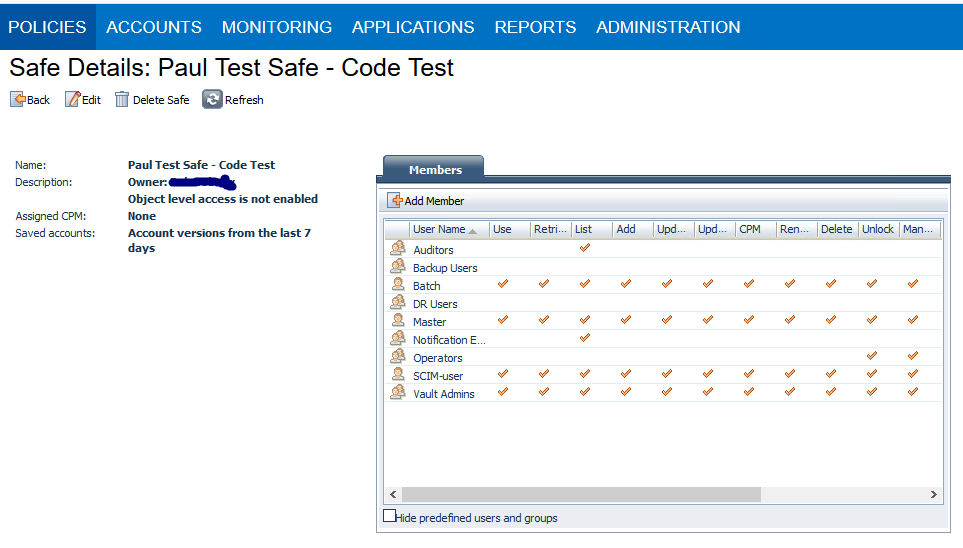
PrivilegedData

PrivilegedData records point to the privileged accounts contained in the Safe.  
In Safeguard, they are called “assets”.

ContainerPermissions

These permissions grant access on a container and are effective for all “Privileged Data” within the container.

* Cyber Ark provides a list of pre-defined permissions: “List”, “Read”, “View audit”, “Retrieve” etc…
* ContainerPermissions are combinations between Safe + User/Group + List of Permissions
* Image below demonstrates Safe and Users/Groups with permissions



Data Model

System

* Cyber Ark system is stored in UNSRootB

User

* SCIM object “User” is stored in UNSAccountB
* It always belongs to the corresponding Cyber Ark system (UNSRootB)

|  |  |  |  |
| --- | --- | --- | --- |
| UNSAccountB | | SCIM – User | Comments |
| CCC\_LinkedAttribute | urn:ietf:params:scim:schemas:cyberark:1.0:User~ldapFullDN | | For Directory User. |
| CCC\_ObjectKeyNamespace |  | | If CCC\_LinkedAttribute is populated – finds a corresponding AD account and puts XObjectKey from table ADSAccount into this field. |
| CCC\_ResourceType | vrtUserTypeDynamic | | If CCC\_LinkedAttribute is populated: DirectoryUser, Else: LocalUser |

Minimum data to create a new record in UNSAccountB:

1. UID\_UNSRootB – Select existing CyberArk target system
2. CN – name of the User

Group

* SCIM object “Group” is stored in UNSGroupB
* It always belongs to the corresponding Cyber Ark system (UNSRootB)

|  |  |  |  |
| --- | --- | --- | --- |
| UNSGroupB | | SCIM – Group | Comments |
| CCC\_LinkedAttribute | urn:ietf:params:scim:schemas:cyberark:1.0:Group~ldapFullDN | | For Directory Group. |
| CCC\_ObjectKeyNamespace |  | | If CCC\_LinkedAttribute is populated – finds a corresponding AD group and puts XObjectKey from table ADSGroup into this field. |
| CCC\_ResourceType | meta~resourceType | | Value = “Group”. It is generated by SCIM server. |
| GroupType | vrtGroupTypeDynamic | | If CCC\_LinkedAttribute is populated: DirectoryGroup, Else: LocalGroup |

Minimum data to create a new record in UNSAccountB:

1. UID\_UNSRootB – Select existing CyberArk target system
2. CN – name of the Group
3. CCC\_ResourceType – must be set to “**Group**”

Container

* SCIM object “Container” is stored in UNSContainerB
* It always belongs to the corresponding Cyber Ark system (UNSRootB)

Minimum data to create a new record

1. UID\_UNSRootB – Select existing CyberArk target system
2. CN – name of the new Safe (Container)

PrivilegedData

* SCIM object “PrivilegedData” is stored in “UNSGroupB”
* It always belongs to the corresponding Cyber Ark system (UNSRootB)
* It always belongs to the corresponding container (Safe)
* SCIM attribute “properties” contains a list of Key/Value pairs  
  In SCIM connector these properties are presented as two MultiValue fields   
  “urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~properties~key”  
  “urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~properties~value”
  + Keys
    - PolicyID
    - UserName
    - DeviceType
    - Address
    - ExtraPass3Name
    - ExtraPass3Safe
    - ExtraPass3Folder

When PrivilegedData is created – it gets an account, assigned to it for reconciling password. Last three values “ExtraPass3Name”, “ExtraPass3Safe” and “ExtraPass3Folder” are there to assign that reconciliation account.

|  |  |  |  |
| --- | --- | --- | --- |
| UNSGroupB | | SCIM – PrivilegedData | Comments |
| CCC\_ResourceType | meta~resourceType | | Value = “PrivilegedData”. It is generated by SCIM server. |
| CCC\_LinkedAttribute | urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~folder | | We use hardcoded value “Root”. It is stored in Config parameter “Custom\CyberArk\PrivilegedDataFolder” for use in scripts. |
| CCC\_PrivDataKeys | urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~properties~key | |  |
| CCC\_PrivDataValues | urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~properties~value | |  |
| GroupType | meta~resourceType | | Value = “PrivilegedData”. It is generated by SCIM server. |
| UID\_UNSContainerB | urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~safe | |  |
| vrtDefaultUserPassword | urn:ietf:params:scim:schemas:cyberark:1.0:PrivilegedData~password | |  |

Minimum data to create a new record in UNSAccountB:

1. UID\_UNSRootB – Select existing CyberArk target system
2. UID\_UNSContainerB – Select existing Safe (Container)
3. CN – name of the Privileged Data
4. CCC\_ResourceType – must be set to “**PrivilegedData**”
5. CCC\_LinkedAttribute – (Folder inside Safe) – “**Root**”
6. CCC\_PrivDataKeys – Keys of the Privileged Data properties. MultiValue field.

Examples:

* 1. PolicyID
  2. UserName - Last part of the CN.
  3. DeviceType -
  4. Address – Server Name or IP.
  5. OwnerEmail
  6. ExtraPass3Name – Reconcile account name. Usually one per domain.
  7. ExtraPass3Safe – Safe where reconcile accounts reside. Usually “VaultInsternal”.
  8. ExtraPass3Folder – Reconcile account folder. Usually “Root”.

1. CCC\_PrivDataKeys – Corresponding values.

**CN** - Operating System-WinDomain-MC-tech.hub-NewTestPrivAccount

**CCC\_PrivDataKeys**: Username,DeviceType,Address,PolicyID,OwnerEmail

**CCC\_PrivDataValues**: NewTestPrivAccount, Operating System, 10.215.146.136, CyberArk, jroberts@yahoo.com

ContainerPermissions

* SCIM object “ContainerPermissions” is stored in “UNSGroupB”
* It always belongs to the corresponding Cyber Ark system (UNSRootB)
* It always belongs to the corresponding container (Safe)
* It has either User or Group data populated. Based on this:
  + There are 2 schema classes in Target System Connection in Sync project:

ContainerPermission\_User and ContainePermission\_Group

* + These classes are defined by the group~value and user~value fields empty value

|  |  |  |  |
| --- | --- | --- | --- |
| UNSGroupB | | SCIM – ContainerPermissions | Comments |
| CCC\_ResourceType | meta~resourceType | | Value = “PrivilegedData”. It is generated by SCIM server. |
| CCC\_LinkedAttribute | user~value | | ID of the User or Group in CyberArk. |
| CCC\_Permissions | Rights | | MultiValue field – a list of rights a user or group has for this safe. |
| UID\_UNSContainerB | container~value | | Safe (Container) |

Minimum data to create a new record in UNSAccountB:

1. UID\_UNSRootB – Select existing CyberArk target system
2. UID\_UNSContainerB – Select existing Safe (Container)
3. CCC\_LinkedAttribute – ObjectGUID of the UNSGroupB or UNSAccountB
4. GroupType – If permissions are for User then “**ContainerPermission\_User**”

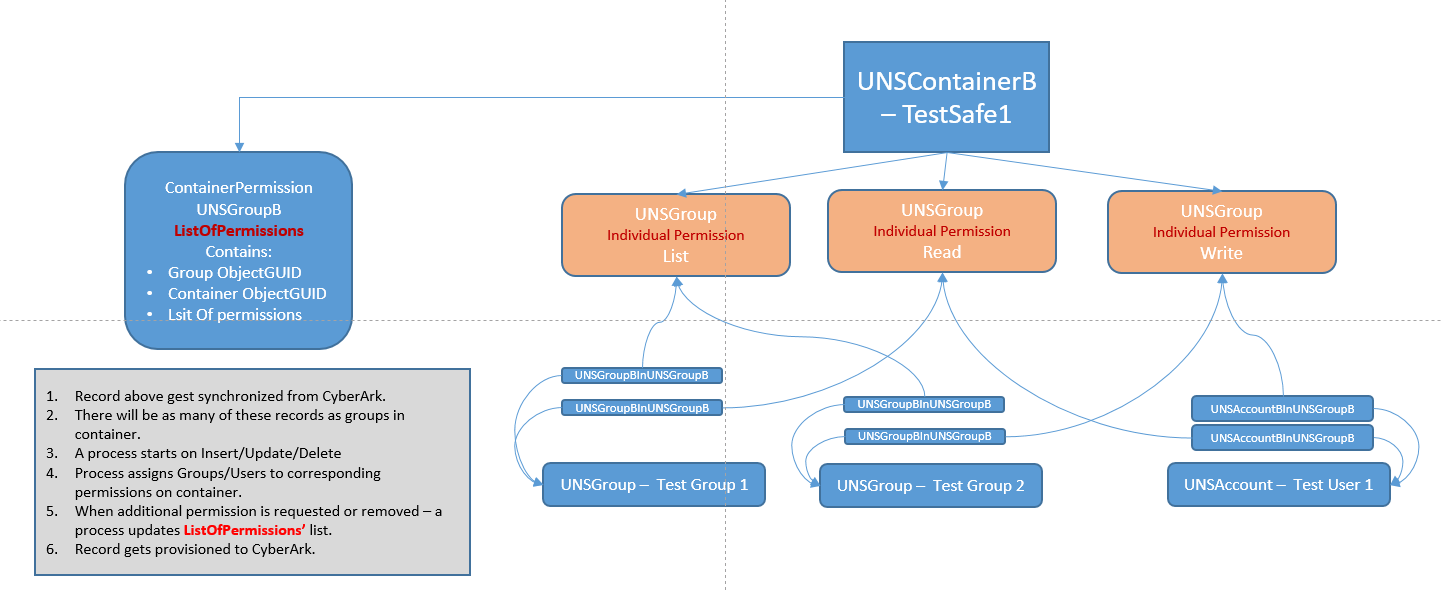
If for Group then “**ContainerPermission\_Group**”

1. CN – “Any Value”. It will get overwritten by CyberArk.
2. CCC\_ResourceType – “ContainerPermission”
3. CCC\_Permissions – MultiValue field of Rights. They are pre-defined in CybeArk.

Delete;Create folder;View audit;Retrieve;Initiate CPM change;View permissions;Backup;Store;Administer;Supervise;Move into;Create object;List;Move from;Initiate CPM change with manual password;Events list;Update object properties;Rename object;Delete folder;Access no confirmation;Manage owners;Validate safe content;Use password;Add events;Unlock object

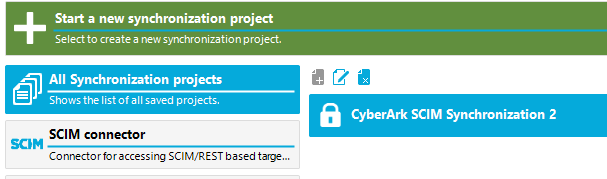
After this record is created either by synchronization or by request, the process “**CCC\_CyberArk\_Populate\_Container\_Permissions**” is executed. It runs script “**CCC\_CyberArk\_Populate\_ContainerPermissions\_ByID**”.

This script takes a list of Rights from field CCC\_Permissions and generates either UNSGroupBInUNSGroupB or UNSAccountBInUNSGroupB records for each permission and user or group from ContainerPermission. See diagram below:

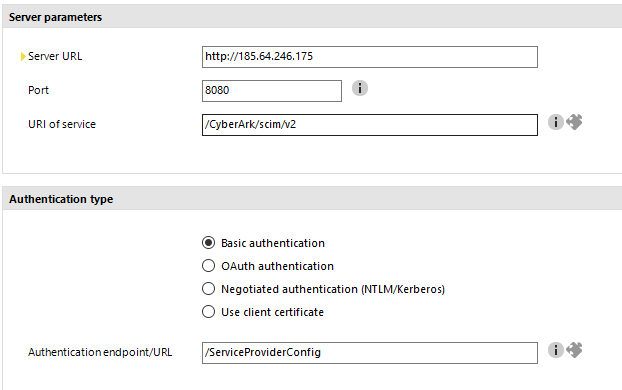


Configuration

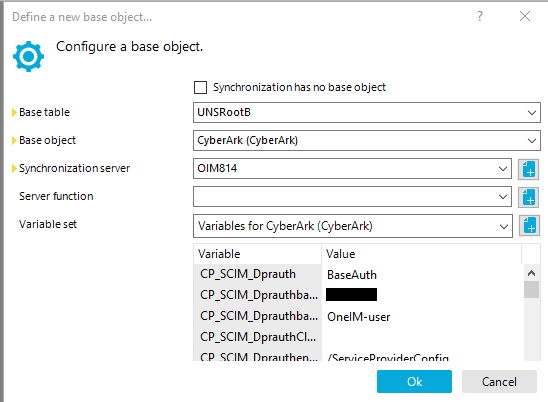
1. Import transport files in the correct order:
   1. Schema
   2. Target System
   3. Sync Project
   4. Integration
2. Open Sync Editor, open project “CyberArk SCIM Synchronization 2”



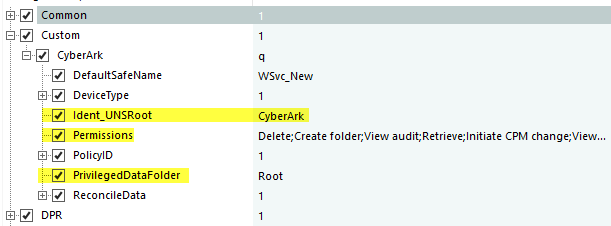
1. Modify One Identity Manager Connection
2. Modify Target System Connection. Values necessary: (examples)
   1. Server URL: http:// <Server Name or IP>
   2. Port: 8080
   3. URI of Service: /CyberArk/scim/v2
   4. Authentication endpoint/URL: /ServiceProviderConfig
   5. User name
   6. Password



1. Add Base Object:



1. Verify Configuration Parameters:
   1. Ident\_UNSRoot – Should be the same as in UNSRootB for CyberArk
   2. PrivilegedDataFolder – For future use cases – creating new PrivilegedData entries
   3. DefaultSafeName – For future use cases - creating new PrivilegedData entries
   4. The rest of the configuration parameters are for specific use cases. They are used in scripts for creating new PrivilegedData entries but might be not necessary depending on actual use cases.



Processes and Scripts

Processes

1. **UNSContainerB - CCC\_CyberArk\_Create\_PreDefined\_SafePermissions**

Calls script **CCC\_CyberArk\_Create\_PreDefined\_SafePermissions** to create a set of pre-defined permission groups per container when a new container is created. [See pic](#ContainerPermission).

1. **UNSGroupB - CCC\_CyberArk\_Populate\_Container\_Permissions**

Calls script **CCC\_CyberArk\_Populate\_ContainerPermissions\_ByID**.

From a List of Rights in ContainerPermission record generates either UNSGroupBInUNSGroupB or UNSAccountBInUNSGroupB records for each permission and user or group from ContainerPermission.

1. **PersonWantsOrg - CCC\_CyberArk\_ContainerPermission\_Request**

Runs on approval of the request to create a new ContainerPermission.

Calls script **CCC\_CyberArk\_Create\_ContainerPermission\_Record** to create a new ContainerPermission record, which is provisioned to CyberArk.

1. **PersonWantsOrg - CCC\_CyberArk\_Privileged\_Account\_Request**

Runs on approval of the request to create a new Privileged Account.

Calls script **CCC\_CyberArk\_CreatePrivelegedData\_FromRequest** to create a new Privileged Account record, which is provisioned to CyberArk.

1. **PersonWantsOrg - CCC\_CyberArk\_Safe\_Request**

Runs on approval of the request to create a new Safe (Cotainer).

Calls script **CCC\_CyberArk\_CreateSafe** to create a new Safe (Container) record, which is provisioned to CyberArk.

Scripts

1. **CCC\_CyberArk\_Create\_PreDefined\_SafePermissions**

Parameters:

* + **uidUNSContainerB –** ID of the new Safe record (UNSContainerB)

1. **CCC\_CyberArk\_Populate\_ContainerPermissions\_ByID**

When UNSGroupB record with CCC\_ResourceType = 'ContainerPermission' is created/updated - it contains a string of rights (permissions) and also either User name or Group name

The string of rights is parsed and records 'UNSAccountBInUNSGroupB' or 'UNSGroupBInUNSGroupB' are created connecting UNSGroupB (representing a permission in container (safe)) and UNSAccountB or UNSGroupB depending on what is present in the original ContainerPermission record.

Parameters:

* + **uidContainerPermission** – ID of the UNSGroupB record representing Container Permission

1. **CCC\_CyberArk\_Create\_ContainerPermission\_Record**

Create an UNSGrouB record with CCC\_ResourceType = 'ContainerPermission' based on IT Shop request.

Parameters:

* + **safeName** – Name of the selected safe (“cn” in UNSContainerB)
  + **userGroupName** – Name of the User or Group (“cn” of the UNSAccountB or UNSGroupB)
  + **strIsGroup** – Specifies whether value in **userGroupName**  is Group or User.

Values: 1/0, Y/N, True/False

* + **strPermissions** – List of selected permissions.

1. **CCC\_CyberArk\_CreatePrivelegedData\_FromRequest**

This script should be modified based on the customer requirements. Various additional parameters can be added to the PrivilegedData record including emails, names of the owners etc…

Parameters:

* + uidPersonWantsOrg – ID of the request.

Values, contained in the request:

* + 1. privAccName
    2. safeName
    3. address
    4. domainType
    5. Other Additional parameters.

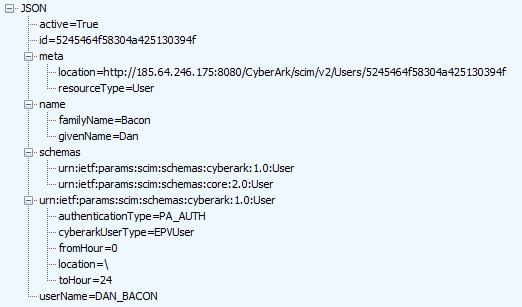
1. **CCC\_CyberArk\_CreateSafe**

Parameters:

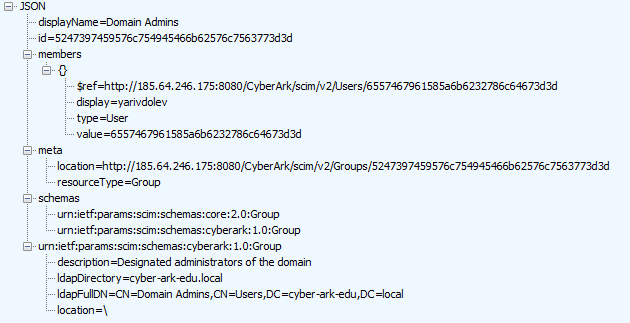
* + safeName
  + description

Record Samples

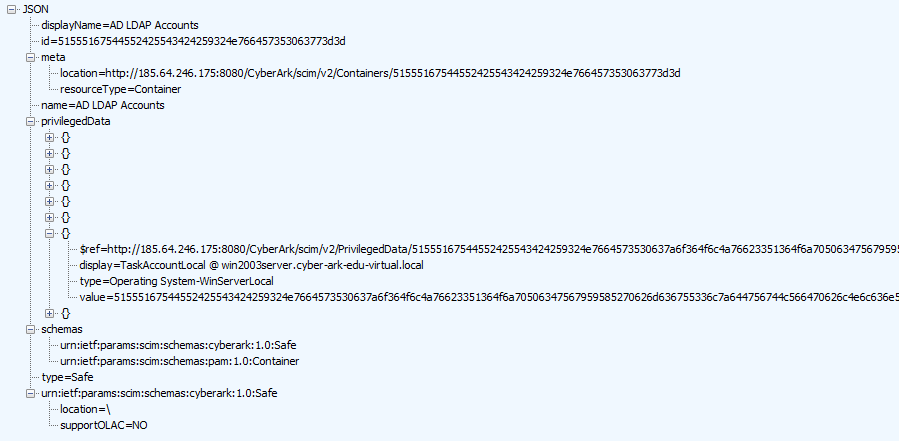
User



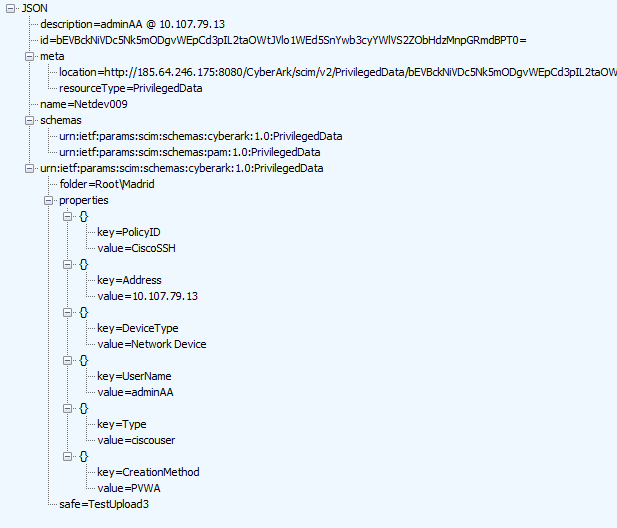
Group



Container



PrivilegedData



ContainerPermission

