Power Week 2025

18 - 19 - 20 novembre 2025

IBM Innovation Studio Paris

S71 – APIs RSE pour gérer les certificats TLS

20 novembre 11:15 - 12:15

Damien Trijasson
Gaia-Volubis
damien.trijasson@gaia.fr

Nathanaël Bonnet
Gaia-Volubis
nathanael.bonnet@gaia.fr

IBM

common FRANCE

Présentation

Nathanaël BONNET

IBM i depuis 1999

Expert IBM i

Damien Trijasson

IBM i depuis 1999

Expert IBM i

GAIA / VOLUBIS

Formation (débutant, perfectionnement)

Expertise IBM i

Centre de Services

GAIA
VOTRE EXPERT IBM i

VOLUBIS
DEVENEZ UN EXPERT

common FRANCE

MEMBER OF common EUROPE

Power Week

18 -19 - 20 novembre 2025

IBM

common FRANCE

IBM

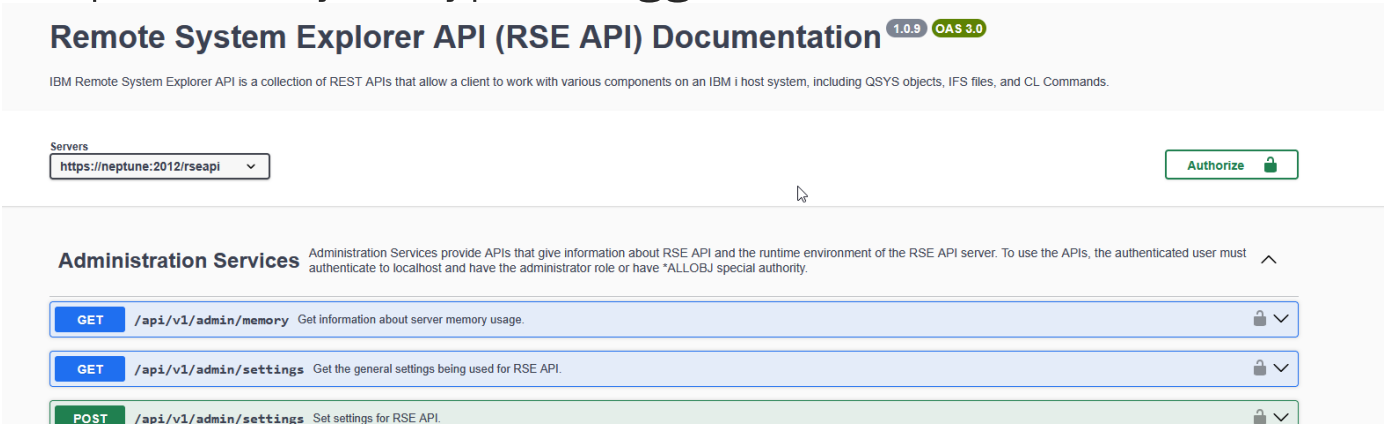
API RSE - Rappel

Où ?

- Depuis un navigateur :
  - En consultation sur le port 2011 : http://host:2011/rseapi/
  - En gestion sur le port 2012 :

https://host:2012/rseapi/ Ou plus direct https://neptune:2012/openapi/ui/

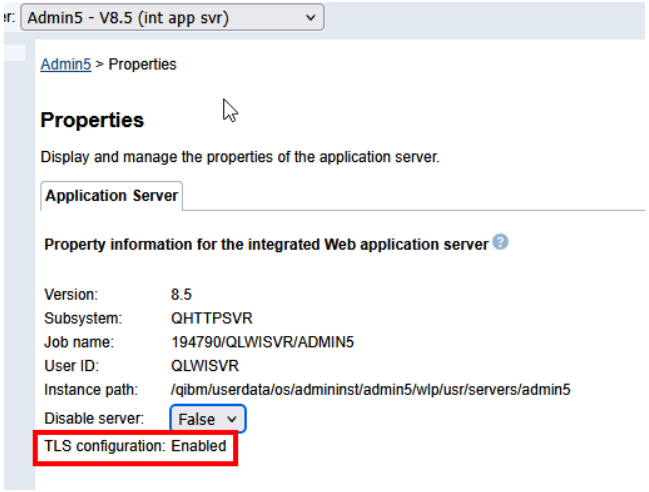
- Interface Open liberty de type swaggerUI

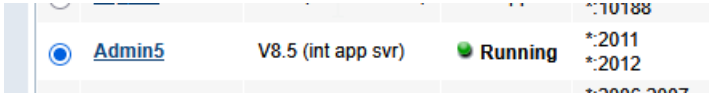


Comment ?

- Serveur admin5 démarré

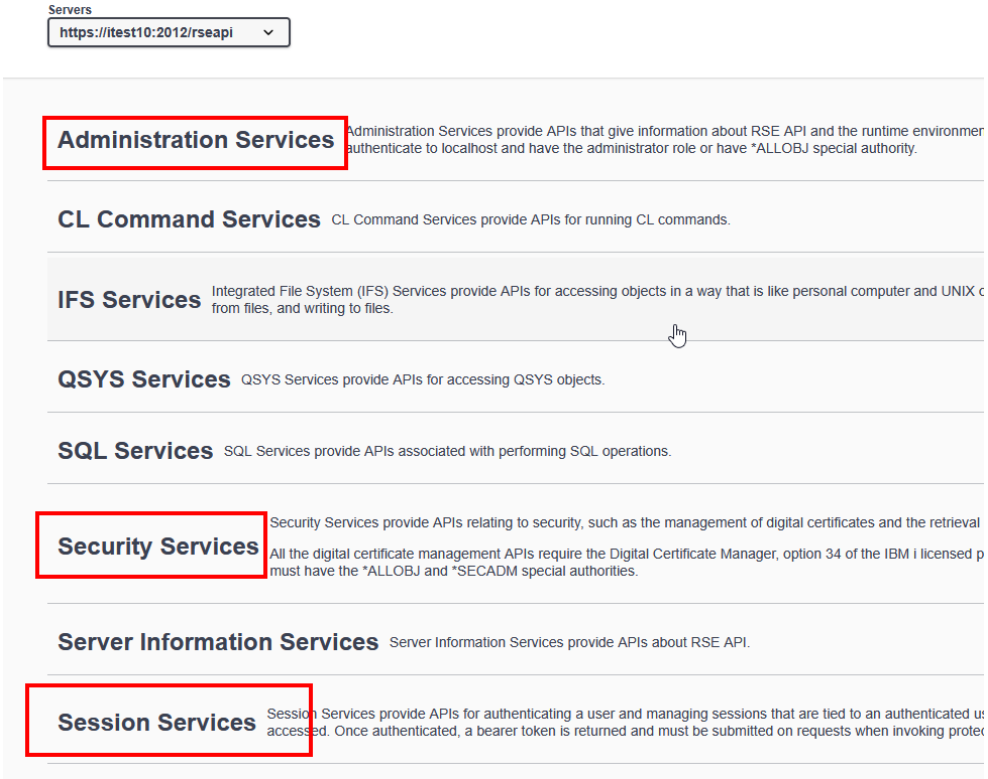
- Activer TLS sur le serveur admin5





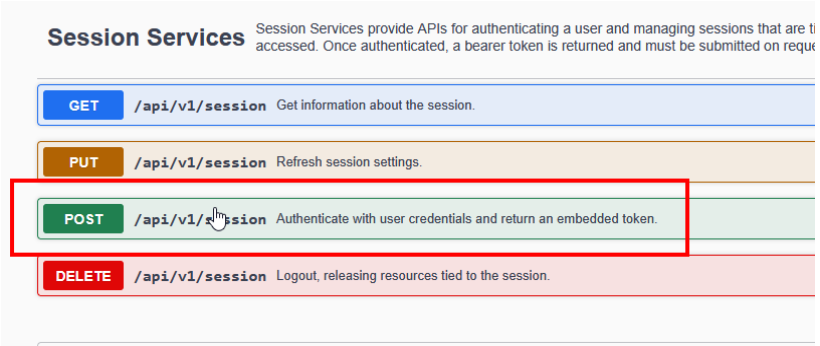
Quoi ?

- 8 services

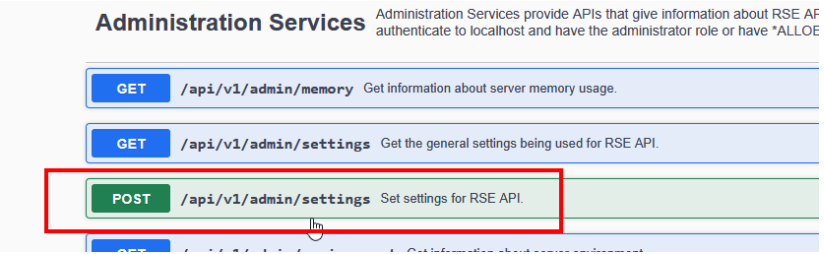


Quoi ?

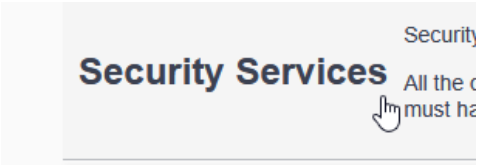
- Récupération d'un token



- Limitation des accès - paramétrage



- Gestion des certificats



Power Week

18 -19 - 20 novembre 2025

IBM

common FRANCE

IBM

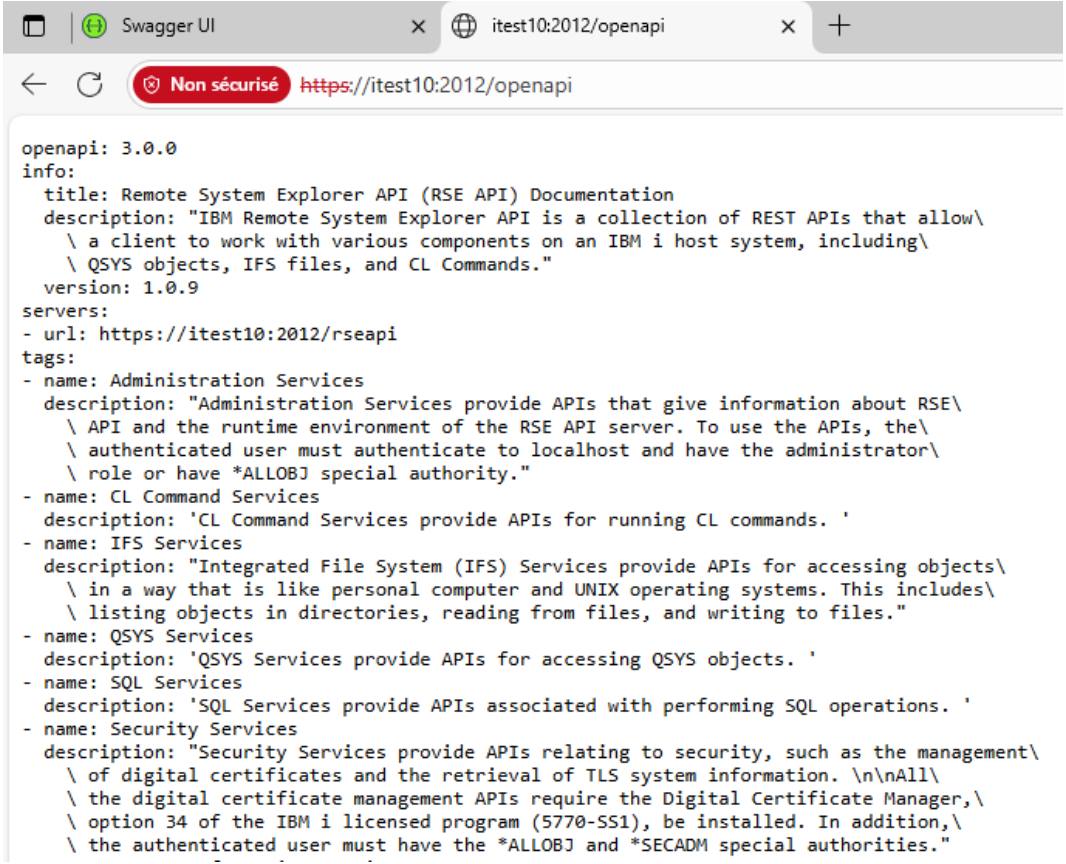
Gestion des certificats TLS

Description de l'API

- https://itest10.gaia.lan:2012/openapi/ui/

Importation dans Postman

- itest10:2012/openapi




```
openapi: 3.0.0
info:
  title: Remote System Explorer API (RSE API) Documentation
  description: "IBM Remote System Explorer API is a collection of REST APIs that allow\
    \ a client to work with various components on an IBM i host system, including\
    \ QSYS objects, IFS files, and CL Commands."
  version: 1.0.9
servers:
- url: https://itest10:2012/rseapi
tags:
- name: Administration Services
  description: "Administration Services provide APIs that give information about RSE\
    \ API and the runtime environment of the RSE API server. To use the APIs, the\
    \ authenticated user must authenticate to localhost and have the administrator\
    \ role or have *ALLOBJ special authority."
- name: CL Command Services
  description: 'CL Command Services provide APIs for running CL commands. '
- name: IFS Services
  description: "Integrated File System (IFS) Services provide APIs for accessing objects\
    \ in a way that is like personal computer and UNIX operating systems. This includes\
    \ listing objects in directories, reading from files, and writing to files."
- name: QSYS Services
  description: 'QSYS Services provide APIs for accessing QSYS objects. '
- name: SQL Services
  description: 'SQL Services provide APIs associated with performing SQL operations. '
- name: Security Services
  description: "Security Services provide APIs relating to security, such as the management\
    \ of digital certificates and the retrieval of TLS system information. \n\nAll\
    \ the digital certificate management APIs require the Digital Certificate Manager,\
    \ option 34 of the IBM i licensed program (5770-SS1), be installed. In addition,\
    \ the authenticated user must have the *ALLOBJ and *SECADM special authorities."
```

Prérequis

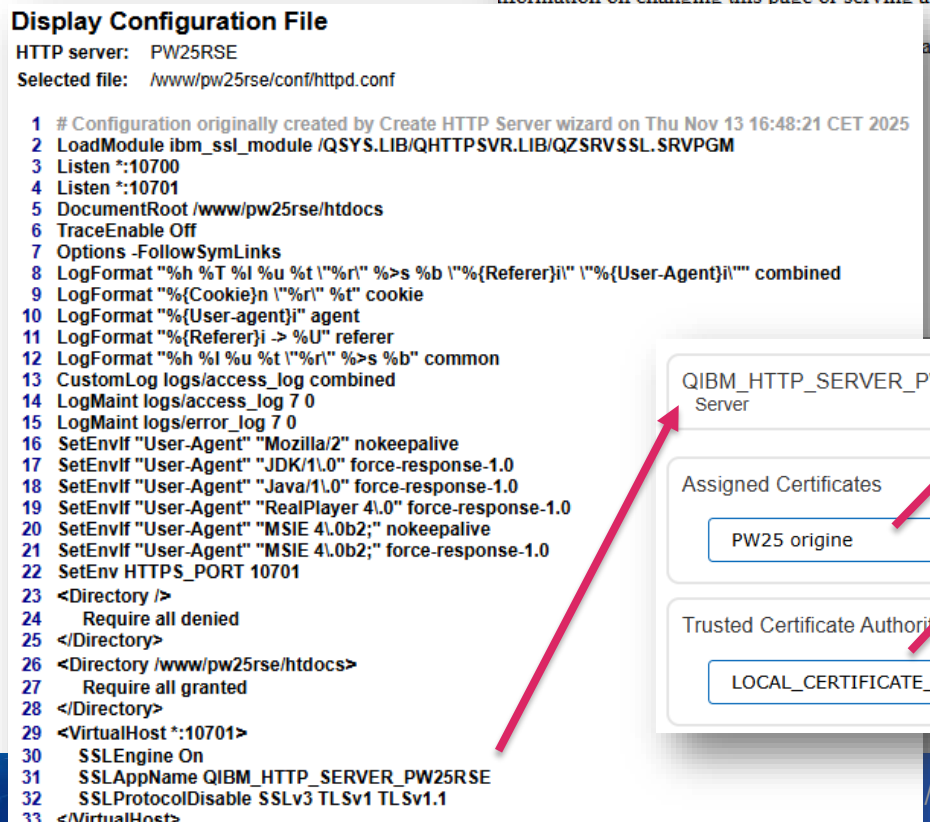
- Pour les APIs « Security Service »
  - L’utilisateur doit être
    - *ALLOBJ + *SECADM
  - Le mot de passe des magasins de certificat

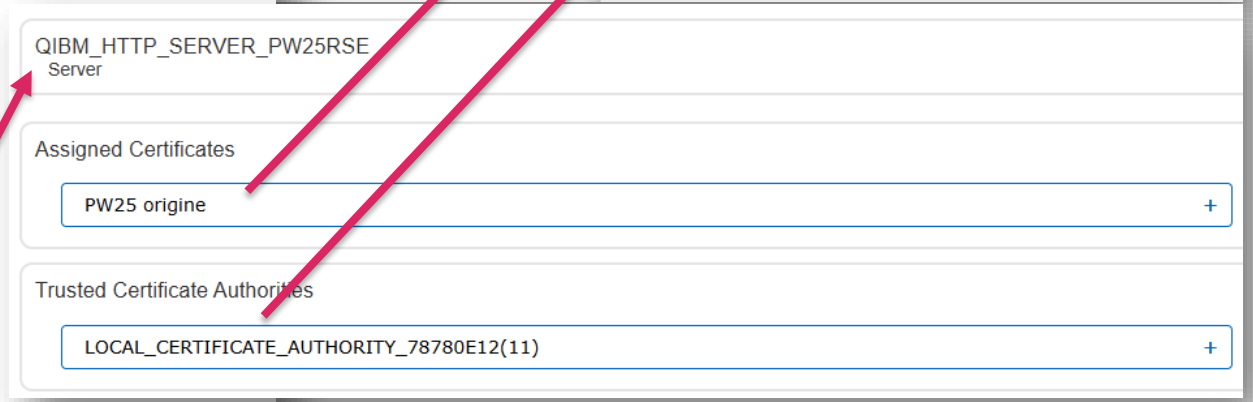
Scénario 1

- On a une instance HTTP Apache existante et sécurisée
  - Importer un certificat
  - L’associer à l’instance
  - Redémarrer

Scénario 1

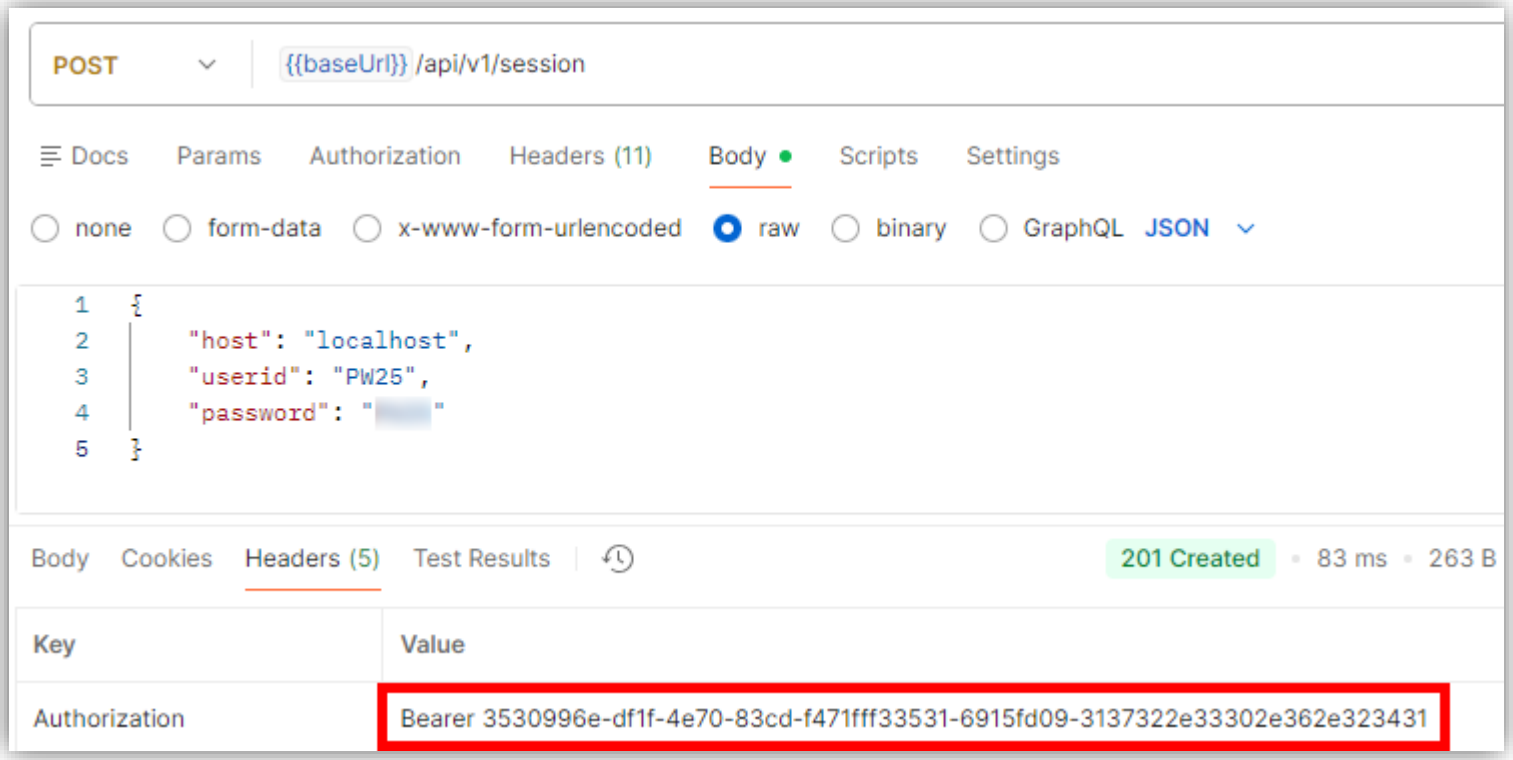
- Instance





Scénario 1

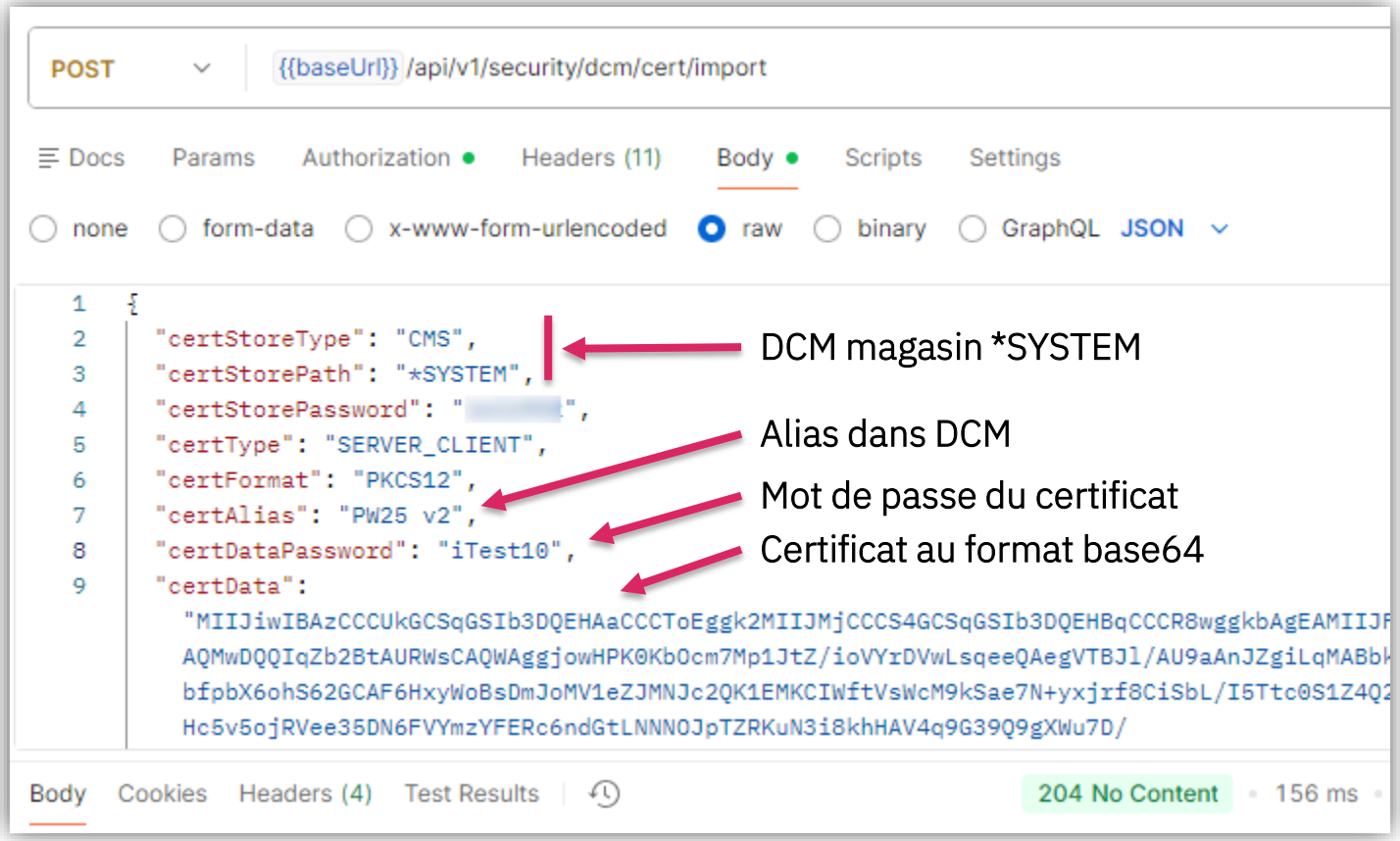
- Authentification



Scénario 1

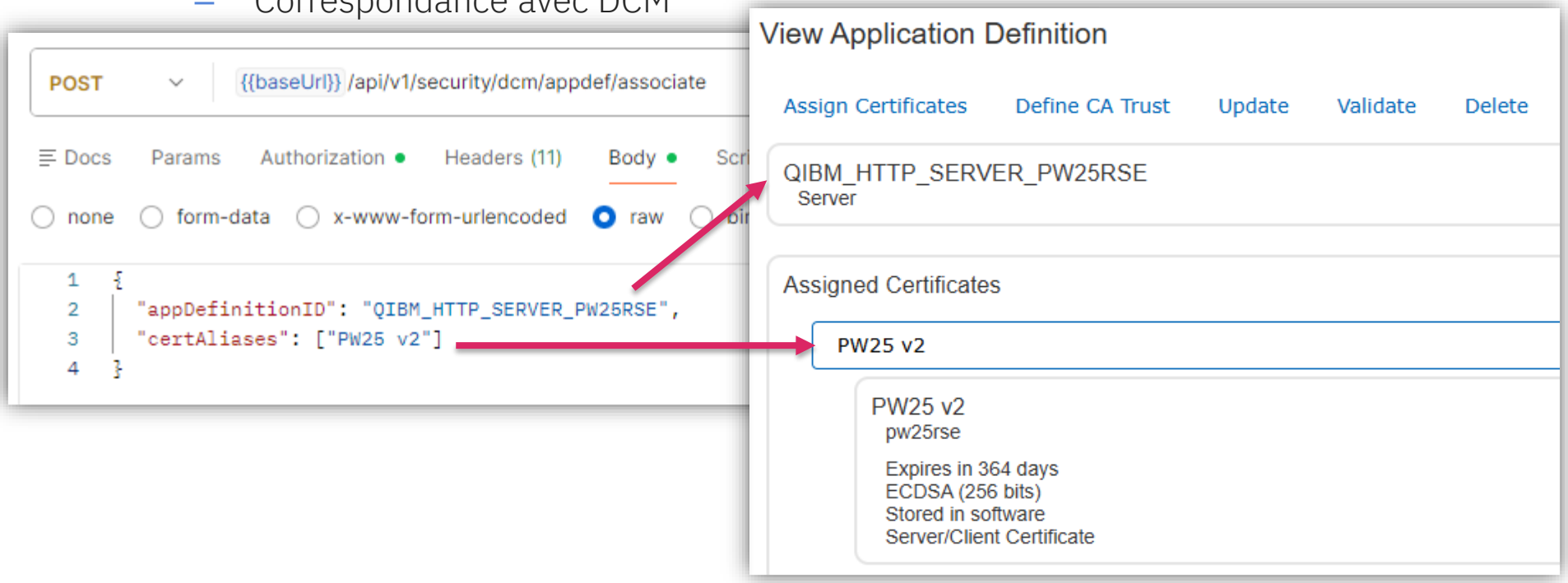
- Importer un certificat
  - Formats supportés :
    - PKCS12 si contient la clé privée (SERVER/CLIENT)
    - PEM (CRT) ou DER sinon (CA)
  - Dans notre cas, les CA existent, on peut importer directement le certificat

Scénario 1


POST
{{baseUrl}} /api/v1/security/dcm/cert/import
Docs
Params
Authorization
Headers (11)
Body
Scripts
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
{
"certStoreType": "CMS",
"certStorePath": "*SYSTEM",
"certStorePassword": "
",
"certType": "SERVER_CLIENT",
"certFormat": "PKCS12",
"certAlias": "PW25 v2",
"certDataPassword": "iTest10",
"certData":
"MIIJiwIBAzCCCUkGCSqGSIb3DQEHAaCCCToEggk2MIIJMjCCCS4GCSqGSIb3DQEHBqCCCR8wggkbAgEAMIIJF
AQMwDQQIqZb2BtAURWsCAQWAggjowHPK0KbOcm7Mp1JtZ/ioVYrDVwLsqeeQAegVTBJl/AU9aAnJZgiLqMABbk
bfpbX6ohS62GCAF6HxyWoBsDmJoMV1eZJMNJc2QK1EMKCIWftVsWcM9kSae7N+yxjrf8CiSbL/I5Ttc0S1Z4Q2
Hc5v5ojRVee35DN6FVYmzYFERc6ndGtLNNNOJpTZRKuN3i8khHAV4q9G39Q9gXWu7D/
DCM magasin *SYSTEM
Alias dans DCM
Mot de passe du certificat
Certificat au format base64
Body
Cookies
Headers (4)
Test Results
204 No Content
156 ms

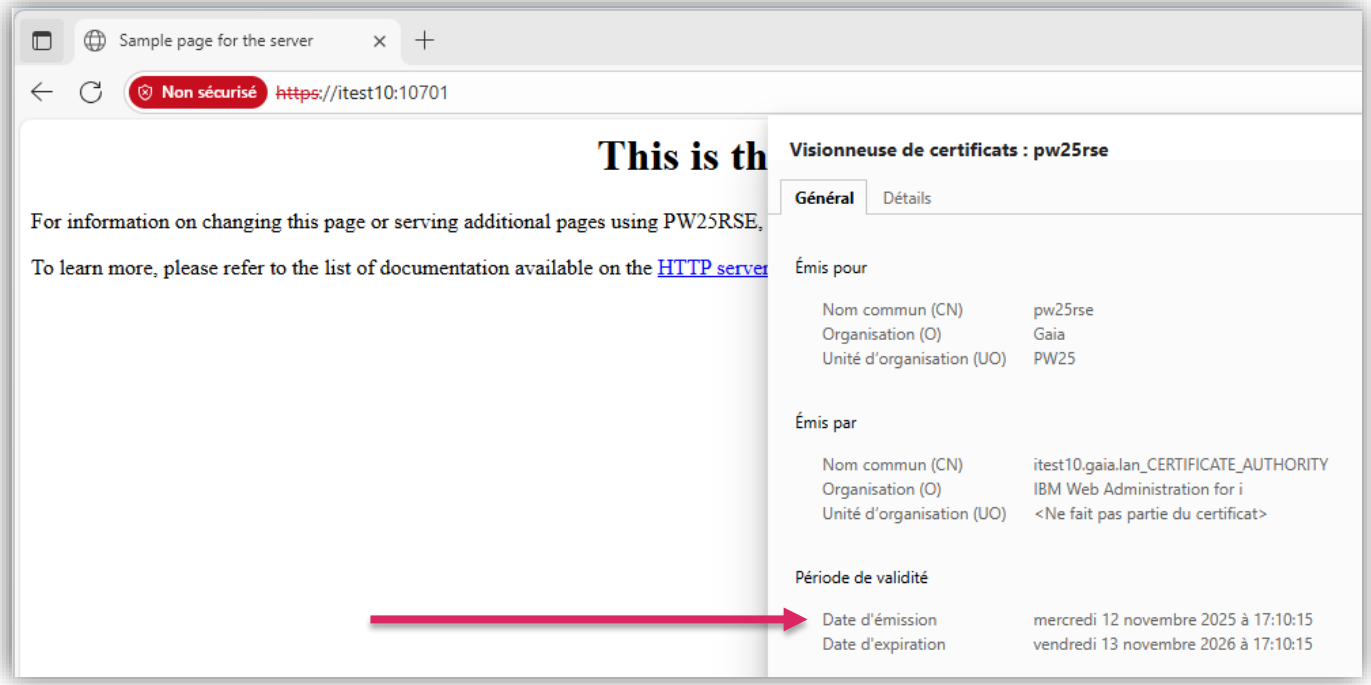
Scénario 1

- L’associer à l’instance
  - Correspondance avec DCM



Scénario 1

- Redémarrer
  - Nouveau certificat pris en compte

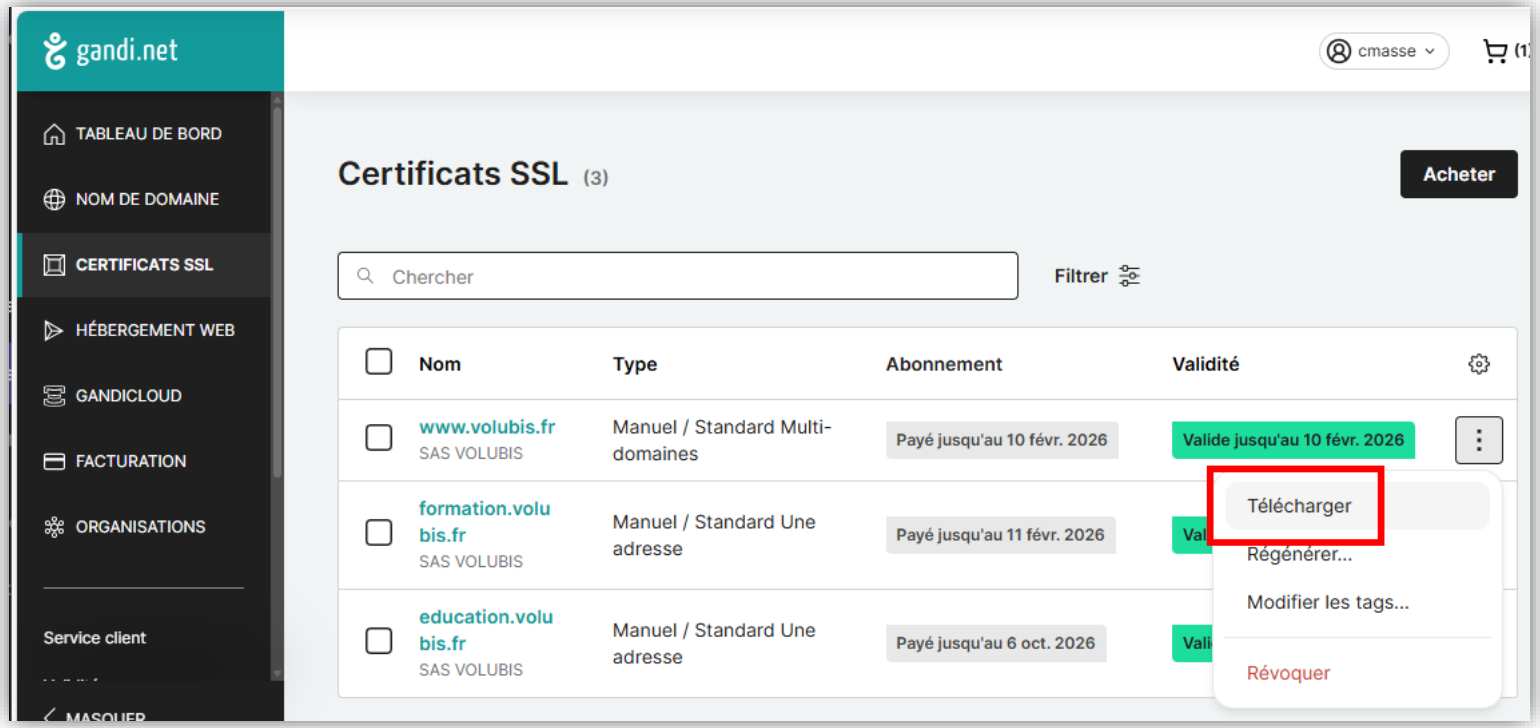


Scénario 2

- Importer un certificat d’une autorité publique
  - Importer le CA / certificat
  - Dissocier le certificat actuel
  - Associer le certificat importé

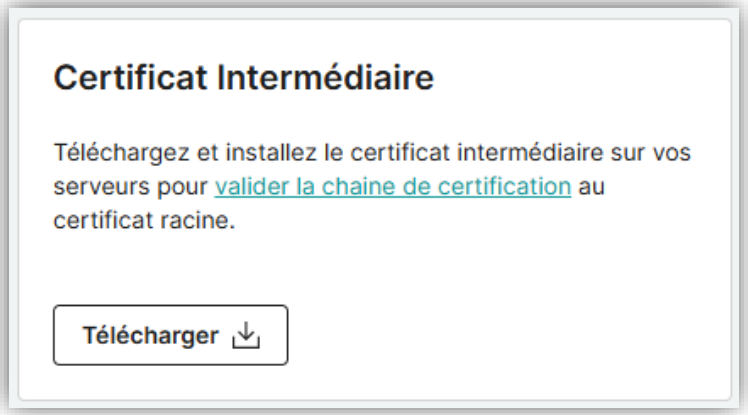
Scénario 2

- Récupérer le certificat + chaine de certification (CAs)



Scénario 2

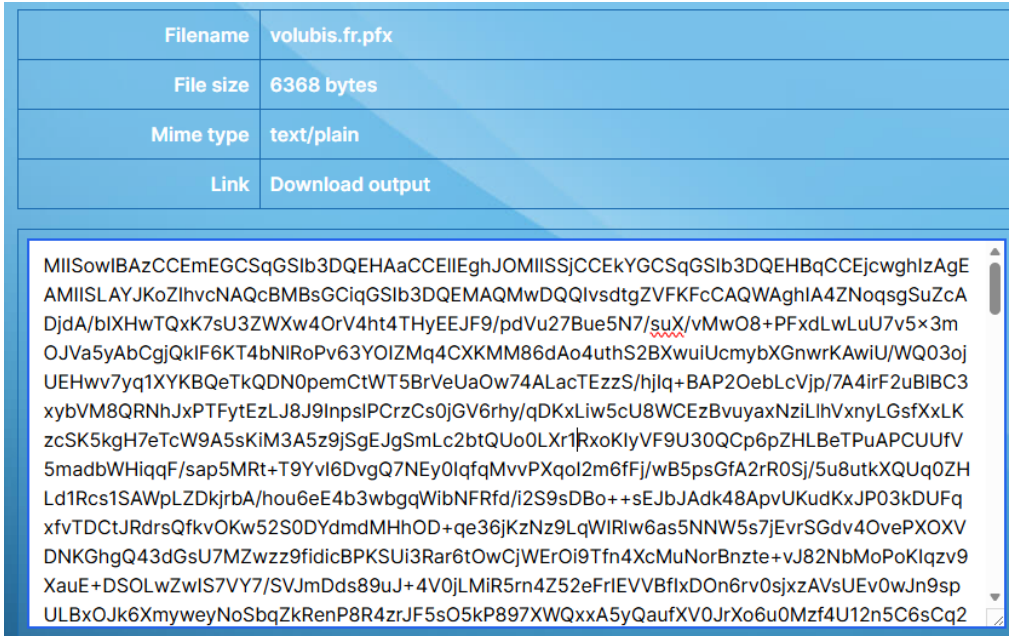
- Récupérer le certificat + chaine de certification (CAs)
  - Souvent une option pour récupérer les autorités (ici en PEM)



  - Il faut pouvoir télécharger avec la clé privée (PKCS12, pfx)
    - Protégé par un mot de passe
    - Contient les CA
    - Peut être considéré comme un magasin

Scénario 2

- Encoder le certificat en base 64
  - On a utilisé https://www.base64encode.org/fr/, https://www.base64encode.net/ ou https://emn178.github.io/online-tools/base64_encode.html
  - Mais aussi

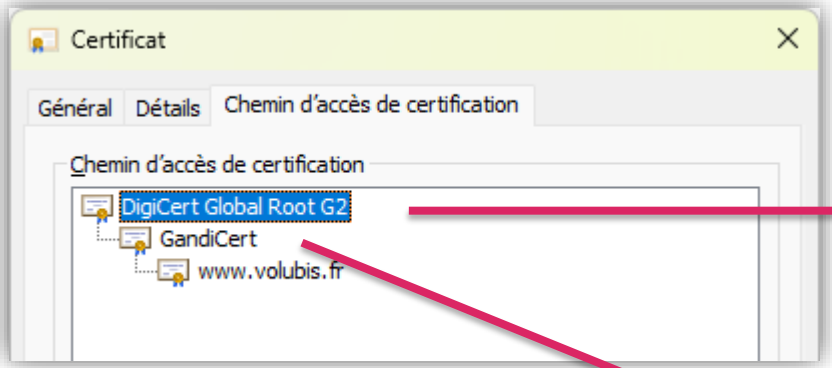



```
SELECT qsys2.base64_encode(line) FROM TABLE(QSYS2.IFS_READ_BINARY(PATH_NAME => '/home/NB/volubis.fr.pfx'));
```

```
00001
MIISowIBAzCCEmEGCSqGSIb3DQEHAaCCElIEghJOMIISSjCCEkYGCSqGSIb3DQEHBqCCEjcwghIzAgEAMIISLAYJKoZIhvcNA...
```

Scénario 2

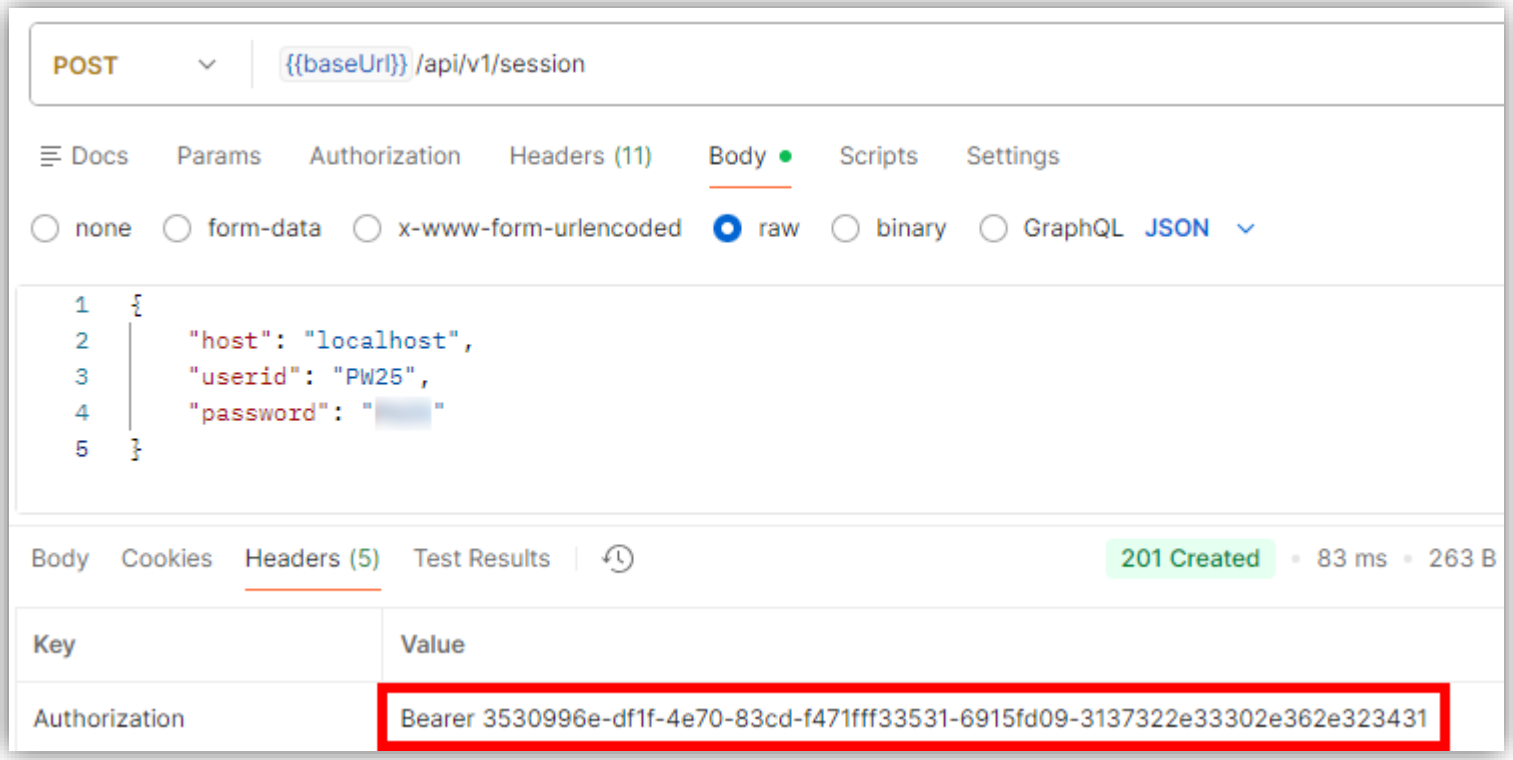
- Cas dans DCM



Absent

Scénario 2

- Authentification



Scénario 2

- Importer le pfx

POST {{baseUrl}} /api/v1/security/dcm/cert/import

Docs Params Authorization Headers (11) Body Scripts Settings

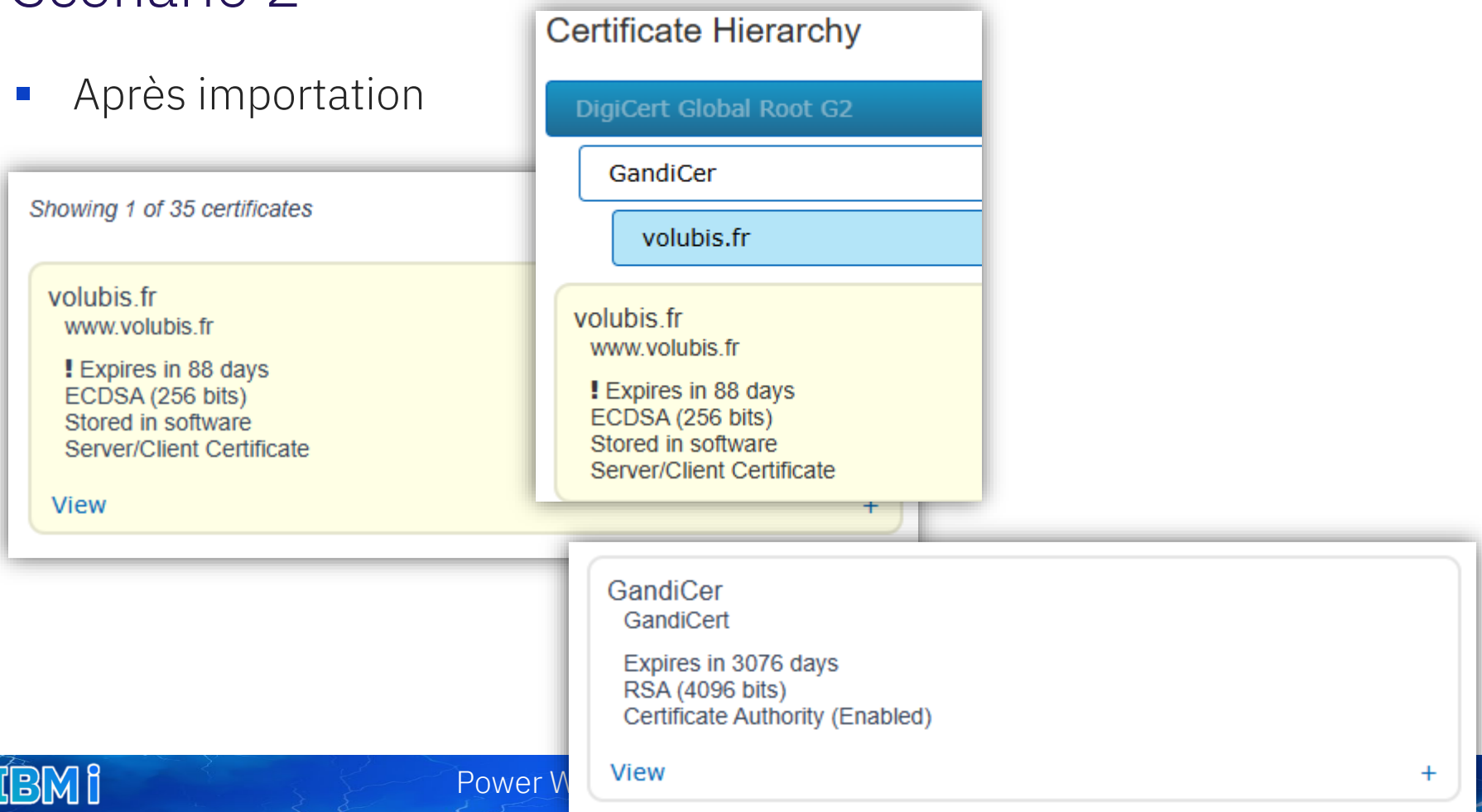
none form-data x-www-form-urlencoded raw binary GraphQL JSON


```
{
"certStoreType": "CMS",
"certStorePath": "*SYSTEM",
"certStorePassword": "[redacted]",
"certType": "SERVER_CLIENT",
"certFormat": "PKCS12",
"certAlias": "volubis.fr",
"certDataPassword": "[redacted]",
"certData": "MIISowIBAzCCEmEGCSqGSIb3DQEHAaCCElIEghJOMIISSjCCEkYGCSqGSIb3DQEHBqCCEjcwghIzAgEAMIISLAYJKoZIhvcNAQcBMBsGCiqGSIb3DQEMAQMwDQQIvsdtgZVFH
bIXHwTQxK7sU3ZWXw4OrV4ht4THyEEJF9/pdVu27Bue5N7/suX/vMwO8+PFxdLwLuU7v5x3mOJVa5yAbCgjQkIF6KT4bNlRoPv63YOIZMq4CXKMM86dAo4uthS2BXwuiUcmybXGnwrKAwiU/
WQ03ojUEHwv7yq1XYKBQeTkQDN0pemCtWT5BrVeUaOw74ALacTEzzS/hjIq+BAP2OebLcVjp/7A4irF2uBlBC3xybVM8QRNhJxPTFytEzLJ8J9InpslPCrzCs0jGV6rhy/
qDKxLiw5cU8WCEzBvuyaxNziLlhVxnyLGsfXxLKzcSK5kgH7eTcW9A5sKiM3A5z9jSgEJgSmLc2btQUo0LXr1RxoKIyVF9U30QCp6pZHLBeTPuAPCUUfV5madbWHiqqF/sap5MRt+T9YvI6D
wB5psGfA2rR0Sj/5u8utkXQUq0ZHLd1Rcs1SAWpLZDkjrbA/hou6eE4b3wbgqWibNFRfd/i2S9sDBo++sEJbJAdk48ApvUKudKxJP03kDUFqxfvTDCtJRdrsQfkvOKw52S0DYdmdMHhOD
+qe36jKzNz9LqWIRlw6as5NNW5s7jEvrSGdv4OvePXOXVDNKGhgQ43dGsU7MZwzz9fidicBPKSUi3Rar6tOwCjWErOi9Tfn4XcMuNorBnzte+vJ82NbMoPoKIqzv9XauE+DSOLwZwIS7VY7/
+4V0jLMiR5rn4Z52eFrIEVVBfIxDOn6rv0sjxzAVsUEv0wJn9spULBxOJk6XmyweyNoSbqZkRenP8R4zrJF5sO5kP897XWQxxA5yQaufXV0JrXo6u0Mzf4U12n5C6sCq2+2fHOJkAc
+BnEaLluMdVM3PczBKJfG3D55NyNgGCldDvTgut6z1Ne6M8reP1Tf+02GfSySKFuew7WXrV0rYtReas4h7ZQzFGx11YNz7+SUw3V1/+INSZFf1VQSKhHC+gOmveXJHbXr1yWnRGGNnqcbTae
fglZldKrVrc5dXwbeddRY2ihPiOmrBhiF0vZKS6gkLxRZupYYSgt8f7Oxai8b4lMSuTKVnfrNBiLYnLrKAUHbXTVpVRZFYu+DQ612f/zMPEZnZ6V56IIgp57KrMjjEM70MPJsCZ2FqVWF/
6qY6tlkmccYLwLIx59zCvZW4iRf06ltsz6Iw1a9XLlnS3mNGoOl+J70zzuIwOdC2TLDEmJXfgMIL06TXWsBuUyaFOBxfvd0tqvIedc0NRzxGmB+dvfuwT4c/EgF/+OTR/
NvQfoPd7x5edOxq2BFnLqRugxob7RDuGCxZICFS8GcAVuLFUWE66kyRyD7fmQOtIjEFDLx7vGCxDorgLRpkz7ire/
```


Body Cookies Headers (4) Test Results 204 No Content 270 ms

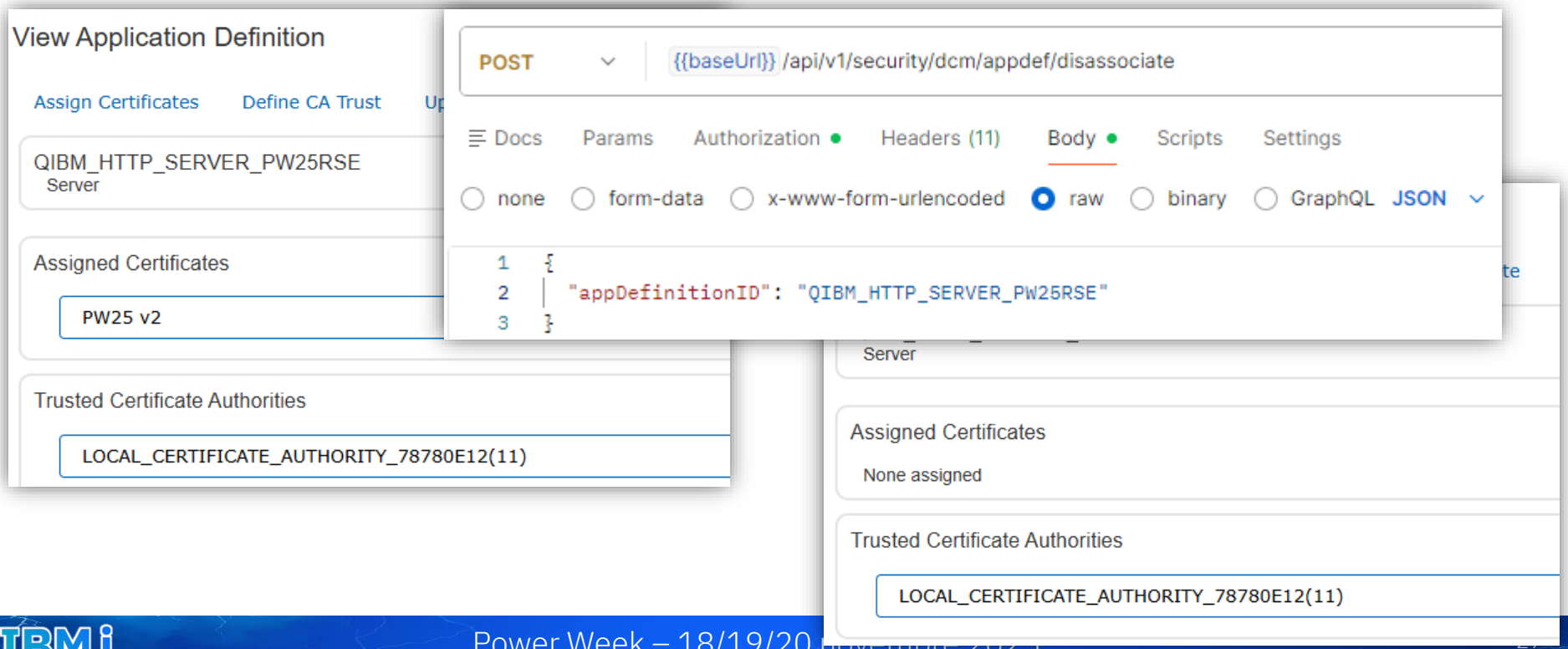
Scénario 2

- Après importation



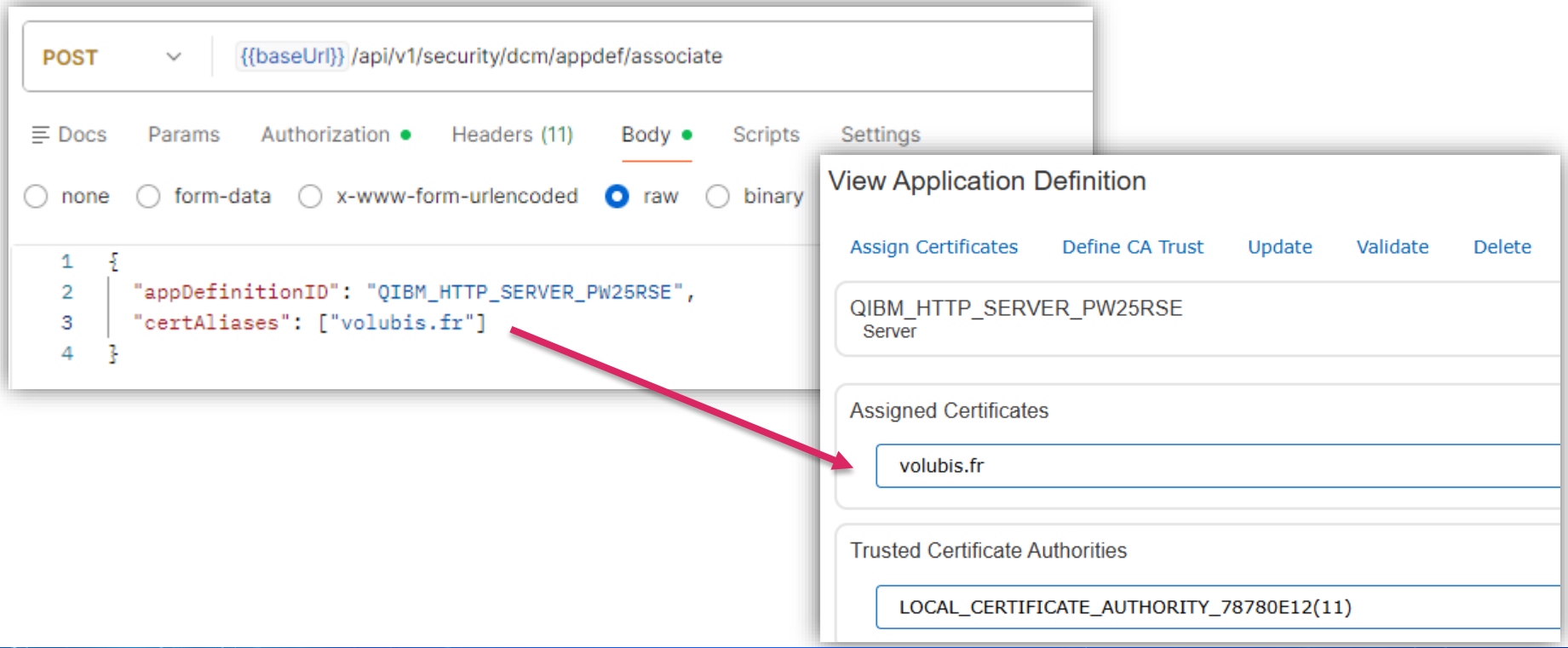
Scénario 2

- Dissocier le certificat



Scénario 2

- Associer le certificat



Scénario 3

- Pour automatiser, du code
  - SQL

```
-- Variables pour intégration
-------------------------------------------------------------------
create or replace variable nb.pw          varchar(10)  ;
create or replace variable nb.pwstore     varchar(10)  ;
create or replace variable nb.body        clob(1M)     ccsid 1208 ;
create or replace variable nb.bearer      varchar(200) ccsid 1208 ;

set nb.pw = ? ;
set nb.pwstore = ? ;
```

```
-- 1. Authentification
-------------------------------------------------------------------

set nb.body = '{"host": "localhost", "userid": "PW25", "password": "' concat nb.pw concat '"}' ;

set nb.bearer = (
  select JSON_VALUE(response_http_header, '$.Authorization' RETURNING varchar(200)) as bearer
  from table(qsys2.http_post_verbose(URL => 'https://itest10.gaia.lan:2012/rseapi/api/v1/session',
                                     REQUEST_MESSAGE => nb.body,
                                     OPTIONS => '{"headers":{"Content-Type":"application/json","Accept":"*/*"}}')) );
```

Scénario 3

- Pour automatiser, du code
  - SQL

```
-- 2. Import
set nb.body = json_object( 'certStoreType' value 'CMS',
                           'certStorePath' value '*SYSTEM',
                           'certStorePassword': nb.pwstore,
                           'certType': 'SERVER_CLIENT',
                           'certFormat': 'PKCS12',
                           'certAlias': 'volubis.fr',
                           'certDataPassword': 'volubis',
                           'certData' : (select qsys2.base64_encode(line)
                                           from table(qsys2.ifs_read_binary(PATH_NAME => '/home/NB/volubis.fr.pfx'))) ) ;

select *
  from table(qsys2.http_post_verbose(URL => 'https://itest10.gaia.lan:2012/rseapi/api/v1/security/dcm/cert/import',
                                     REQUEST_MESSAGE => nb.body,
                                     OPTIONS => '{"headers":{"Content-Type":"application/json",
                                                             "Accept":"*/*",
                                                             "Authorization":"' concat nb.bearer concat '"}}')) ;
```

Scénario 3

- Pour automatiser, du code
  - curl

```
PATH=/QOpenSys/pkgs/bin:$PATH
export PATH PASE_PATH

# 1. Authentification
curl -v -k \
     --header 'Content-Type: application/json' \
     --header 'Accept: */*' \
     --data '{"host": "localhost", "userid": "PW25", "password": "[redacted]"}' \
     --location --request POST 'https://itest10.gaia.lan:2012/rseapi/api/v1/session' \
     > /dev/null  2>curl.log

BEARER=$(grep -i '< Authorization:' curl.log | sed 's/< Authorization://g')
echo $BEARER
```

```
-bash-5.2$ changecert.sh
Bearer 7a0726a8-04ab-451d-b557-d2ae425340e6-69170057-3137322e33302e31342e3137
```

Remarques

- Pas (encore ?) toutes les fonctions de DCM
  - Impossible de créer un nouveau certificat
    - « Même pas un renew »

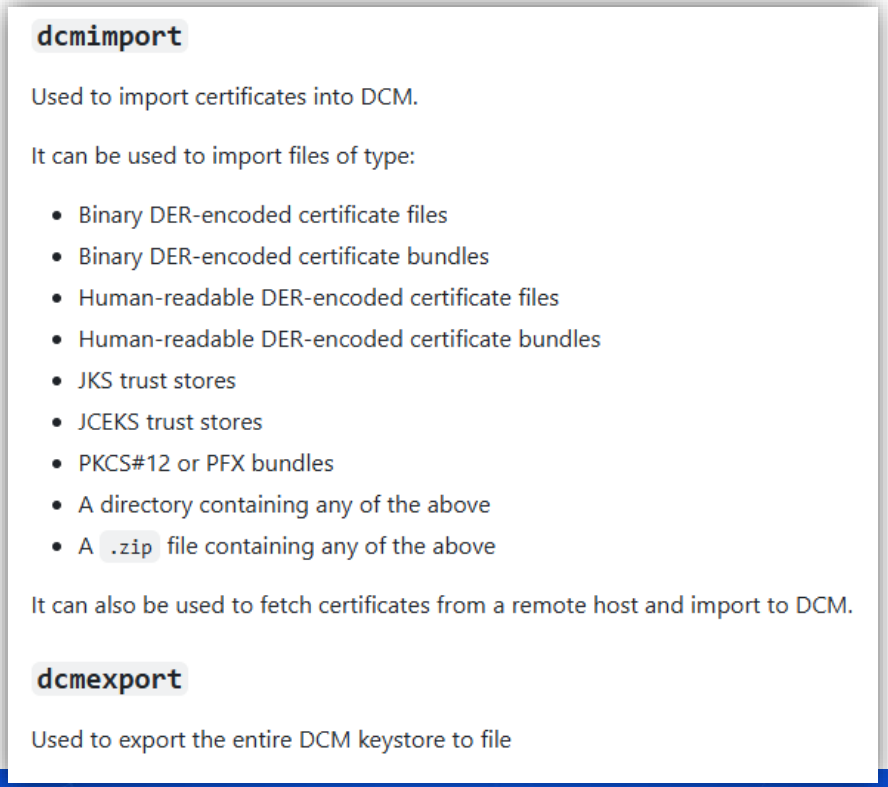
- Via SQL

```
15 select *
16   from table (
17       qsys2.certificate_info(certificate_store_password => '*NOPWD')
18     )
19   where validity_end < current date + 1 month
20   order by validity_end;
```

CERTIFICATE_LABEL	SERIAL_NUMBER	VALIDITY_START	VALIDITY_END
LOCAL_CERTIFICATE_AUTHORITY_78780E12(5)	6785139B	2025-01-12 14:22:35	2025-01-15 14:22:35
LOCAL_CERTIFICATE_AUTHORITY_78780E12(6)	67BC422F	2025-02-23 10:55:59	2025-02-26 10:55:59
itest10-services-2024	66E012CA06E120	2024-09-09 11:35:06	2025-09-10 11:35:06

Remarques

- https://github.com/ThePrez/DCM-tools



`dcmimport`

Used to import certificates into DCM.

It can be used to import files of type:

- Binary DER-encoded certificate files
- Binary DER-encoded certificate bundles
- Human-readable DER-encoded certificate files
- Human-readable DER-encoded certificate bundles
- JKS trust stores
- JCEKS trust stores
- PKCS#12 or PFX bundles
- A directory containing any of the above
- A `.zip` file containing any of the above

It can also be used to fetch certificates from a remote host and import to DCM.

`dcmexport`

Used to export the entire DCM keystore to file


MERCI