e.2.- Sample authentication script (using JWT private key)

Below is an example of authentication script.

private_key_jwt authentication script

```
#!/bin/bash
PRIVATE_KEY_ES256=$2
CLIENT_ID=$1
SCOPE="das-api/office-exchange"
ISSUER="https://www5.wipo.int/am/oauth2"

# https://logindev.wipo.int/am/oauth2/.well-known/openid-
configuration
OIDC_CONFIG_JSON=$(curl -sS -k "${ISSUER}/.well-known/openid-
configuration")

# Generic way to obtain the token endpoint
TOKEN_ENDPOINT=$(printf '%s' ${OIDC_CONFIG_JSON} | jq -r ".
token_endpoint")

#echo $TOKEN_ENDPOINT

UTC_TIME=$(date -u +%s)
EXP_TIME=$(expr "$UTC_TIME" + 1000)

JSON='{'
JSON=${JSON}$(printf '"iss":"%s"' ${CLIENT_ID})
JSON=${JSON}$(printf ,"sub":"%s"' ${CLIENT_ID})
JSON=${JSON}$(printf ,"aud":"%s"' ${TOKEN_ENDPOINT})
JSON=${JSON}$(printf ,"exp":%s' ${EXP_TIME})
JSON=${JSON}'}'

JSON_HEADER_B64=$(printf '{"alg":"ES256","typ":"JWT"}' | jq .
-cj | base64 -w0 | tr -d '\n=' | tr '+/' '-_')
# echo $JSON_HEADER_B64
JSON_PAYLOAD_B64=$(printf $JSON | jq . -cj | base64 -w0 | tr -
d '\n=' | tr '+/' '-_')
JSON_SIGNATURE_ASN1_B64=$(printf '%s.%s' $JSON_HEADER_B64
$JSON_PAYLOAD_B64 | openssl dgst -sha256 -sign
"${PRIVATE_KEY_ES256}" | openssl asn1parse -inform DER |
base64 -w0)
JSON_SIGNATURE_HEX=$(printf $JSON_SIGNATURE_ASN1_B64 | base64
-d | sed -n '/INTEGER/p' | sed 's/.*INTEGER\s*://g' | sed -z
's/[^0-9A-F]//g')
JSON_SIGNATURE_B64=$(printf $JSON_SIGNATURE_HEX | xxd -p -r |
base64 -w0 | tr -d '\n=' | tr '+/' '-_')

JWT_ASSERTION=$(printf '%s.%s.%s' $JSON_HEADER_B64
$JSON_PAYLOAD_B64 $JSON_SIGNATURE_B64)
echo
#echo $JWT_ASSERTION

echo


curl --insecure --location --request POST "${TOKEN_ENDPOINT}"
\
--header "Content-Type: application/x-www-form-urlencoded" \
--data-urlencode "grant_type=client_credentials" \
--data-urlencode "scope=${SCOPE}" \
--data-urlencode "client_assertion_type=urn:ietf:params:oauth:
client-assertion-type:jwt-bearer" \
--data-urlencode "client_assertion=${JWT_ASSERTION}"
```

Information on how to execute the sample scripts

e.3.- Executing the sample scripts under linux

e.4.- Executing the sample scripts via docker

The output of the script is as follows:

private_key_jwt authentication output


```
{
  "access_token": "eyJ0eXAiOiJKV1QiLCJraWQiOiJmVWRmbEJSa3c5bm1
tejcrL3BmMWM5d2RYdXc9I.......vvwFVnq8c8QArKsMmgBw",
  "scope": "das-api/das-access",
  "token_type": "Bearer",
  "expires_in": 3599
}
```


Access_token attributes like signature, validity, audience and scopes must be verified by the client, similarly DAS API must also verify the access_token and must additionally check if the client id (=sub claim) is authorized. DAS API must maintain the whitelisted clients

---

access_token payload


```
{
  "sub": "das-api-auth",
  "cts": "OAUTH2_STATELESS_GRANT",
  "auditTrackingId": "142b3081-d3c7-422c-b8d4-65869065f348-
54991",
  "iss": "https://logindev.wipo.int:443/am/oauth2",
  "tokenName": "access_token",
  "token_type": "Bearer",
  "authGrantId": "nJ8nhylC8Kx9DY8l2SHlopwCfbg",
  "aud": "das-api-auth",
  "nbf": 1622454953,
  "grant_type": "client_credentials",
  "scope": [
    "das-api/das-access"
  ],
  "auth_time": 1622454953,
  "realm": "/",
  "exp": 1622458553,
  "iat": 1622454953,
  "expires_in": 3600,
  "jti": "roG8mqa8Z1Z3F00Md20vUoyhA0I"
}
```


New DAS exchange API