MSDF SDCOE Ed-Fi Transcript Prototype: Deployment Options Guide

Revision July 2021

San Diego County Office of Education, Innovation and ITS (datascience@innovatesd.net)
Application development: Leapfrog Technology, Inc. (DBA Eduphoric)

Introduction

This document talks about the various configuration options that were added during Phase 1.3. The configuration options enable running the core functionality of the SDCOE with minimum need to configure external api and services. In this iteration, an option to disable the need for blockchain and an option to enable file based verification of a transcript has been added. The Microsoft authentication mechanism is also optional now and one user can authenticate using both Microsoft login and Basic/Local login mechanism.

The upcoming sections describe each of the configuration options in detail.

Note: There are sections in this document where a URL, domain name, user name, password, IP address or email address are highlighted in light orange. This signifies where each person or organization installing this application will substitute their own information.

Configuration changes in transcript-api

This section talks about the configuration parameters that are needed after Phase 1.3 in the Transcript-api module. The module resides in a subdirectory called `transcript-api`

Turning the Digital Signing On or Off

The functionalities to toggle the digital signing are configured via transcript-api module. All the functionalities to sign the PDF to ensure the tamper free pdf can be configured in the transcript-api module. This module contains the code for REST API for transcript API and depends on modules like pdf-utils, email utils etc for other functionalities.

In order to toggle the Digital Signature, we need to update the environment file (.env) file and rebuild the transcript-api module.

```
# Verification
IS_DIGITAL_SIGNING_ENABLED=${SDCOE_IS_DIGITAL_SIGNING_ENABLED}
```

Possible values for the above env variables are tabulated below:

IS_DIGITAL_SIGNING_ENABLED	true	Digital signing is enabled
	false	Digital signing is disabled

Note: The transcript generated while the digital signing disabled (i.e false), will not be validated.

After updating the desired configuration value, rebuild the module and start the server.

```
$ cd transcript-api
# Installing the dependencies
$ yarn
# making production build
$ yarn build
```

Starting the server using PM2

```
$ pm2 start dist/index.js --name transcript-api
```

If this works correctly, one should be able to point a browser to the URL and port specified in the .env file.

Signing and Verification modes

Configure Verification Mode

In the transcript-api module before signing the pdf, the mode to sign the document can be configured in the environment variable. The signing mode is configured with the following variable. For both "did" and "file" based signing modes, the "proof-metadata", added to the pdf , is signed using the same private key. But the signed pdf can only be verified using the mode specified during the signing process. So a pdf signed using "file" mode is *verified as false* when an attempt to verify it is made using a blockchain (did) method.

```
# Pass either did or file
DIGITAL_SIGNING_MODE=${SDCOE_DIGITAL_SIGNING_MODE}
```

Possible values for the above env variables are tabulated below:

DIGITAL_SIGNING_MODE	did	DID based signing
	file	File based signing

Configure Verification Key

After adding digital signing mode in the above step, we need to either set up Ethereum related configuration or a simple URN for the verification key.

DID related Configuration
Explained in the Windows Deployment Documentation. Refer to the security section in the Windows Deployment Documentation.

Updated env variable

```
# Security: DID and URN key
VERIFICATION_KEY_DID=${SDCOE_VERIFICATION_KEY_DID}
VERIFICATION_KEY_URN=${SDCOE_VERIFICATION_KEY_URN}

# RSA: Path to file containing the signer private key
SIGNING_KEY_URN=${SDCOE_SIGNING_KEY_URN}
```

For **file mode** of verification, all the DID related configuration can be skipped. One should only add the Verification/Signing Key URN for the file mode.

The SDCOE Signing/Verification Key Format and illustrative example is shown below

urn:sdcoe-keypair:L2hvbWUvc2Rjb2UtZGF0YS9wdWJsaWMta2V5LnB1Yg==#created-20210714	
urn:	prefixed with urn: to match the did: prefix pattern in the blockchain counterpart
sdcoe-keypair:	URN identifier
L2hvbWUvYWFrcml0c3ViZWRpL0Rlc2t0b3Avc2Rjb2UtZGF0YS9wdWJsaWMta2V5LnB1Yg==	base64 encoded path to public/private key
#created-20210714	created date as identifier

Base64 private/public key path configuration

Public Key Path

L2hvbWUvc2Rjb2UtZGF0YS9wdWJsaWMta2V5LnBlbQ==
Decoded File path: /home/sdcoe-data/public-key.pem

So, the verification key URN is

urn:sdcoe-keypair:L2hvbWUvc2Rjb2UtZGF0YS9wdWJsaWMta2V5LnBlbQ==#created-20210720

Private Key Path

L2hvbWUvc2Rjb2UtZGF0YS9wcml2YXRlLWtleS5wZW0==
Decoded File path: /home/sdcoe-data/private-key.pem

So, the signing key URN is

urn:sdcoe-keypair:L2hvbWUvc2Rjb2UtZGF0YS9wcml2YXRlLWtleS5wZW0==#created-20210720

The URN pattern should match exactly as illustrated in above examples.

After updating the desired configuration value, rebuild the module and start the server using PM2.

Note: Use an online tool to base64 encode/decode the path to the keypair.

- Base64 Encode
  https://www.base64encode.org/
- Base64 Decode
  https://www.base64decode.org/

Auth Mode Switching

This section talks about the additional configuration for authentication mode needed to set up the transcript-api after Phase 1.3

Configuration changes in transcript-api

The switching of the authentication mode is configured in the transcript-api module and this configuration is served from the config endpoint i.e https://yourdomainorip/api/config. The client side application will render the login component based on the configuration value.

Auth Mode switching environment variables

```
# Login methods: Login methods and Microsoft auth
IS_LOCAL_AUTH_ENABLED=${SDCOE_LOCAL_AUTH_ENABLED}
IS_MICROSOFT_AUTH_ENABLED=${SDCOE_MICROSOFT_AUTH_ENABLED}
```

Possible values for the both env variables are tabulated below:

IS_LOCAL_AUTH_ENABLED	true	Enable the Local Auth method
	false	Disable the Local Auth method

IS_MICROSOFT_AUTH_ENABLED	true	Enable the Basic Auth method
	false	Enable the Basic Auth method

We should at least enable one of the authentication modes. Enabling both the authentication mode will support both ways to login to the system.

Example response from the config endpoint

```
// https://yourdomainorip/api/config

{
  "isLocalAuthEnabled": "true",
  "isMicrosoftAuthEnabled": "true"
}
```

For basic/local authentication mode, we should add additional configuration to the transcript-api module. The SALT value for password hashing and Reset password link should be configured as below:

Salt Factor

SALT_FACTOR	Numeric value	Salt factor support only numeric value
	Example: 10	Default suggested value is 10.

Redirect URL

```
# Password Reset Link
APP_RESET_PASSWORD_LINK=${SDCOE_APP_RESET_PASSWORD_LINK}
```

The redirect URL should be configured as the reset link of the frontend application. The redirect link is sent in email when the user requests to change his/her password.

```
# Password Reset Link
APP_RESET_PASSWORD_LINK=https://yourdomainorip/forgot-password
```

After updating the desired configuration value, rebuild the module and start the server using PM2.

Web client config

There is an update regarding the naming of the environment variable for the Microsoft based authentication login URI in web-client. We have added a `REACT_APP` prefix to the previous name.

```
# Microsoft Redirect URI
REACT_APP_MICROSOFT_LOGIN_URI=${SDCOE_REACT_APP_LOGIN_URI}
```

The authentication configurations are exposed as a response of an API endpoint service. We call an API endpoint `https://yourdomainorip/api/config` to fetch the authentication configurations while visiting the landing page for the first time. Based on the response we get, we set our authentication mode on our application either to basic authentication or microsoft authentication or both.

Detailed scenarios/cases for an application based on the response configurations are presented below.

isLocalAuthEnabled	true	Enable the Local Auth method i.e. password based authentication
	false	Disable the Local Auth method

isMicrosoftAuthEnabled	true	Enable the Microsoft based authentication method
	false	Disable the Microsoft based authentication method

isLocalAuthEnabled	true	Redirect to login page after clicking on login from header Hide login with Microsoft button on login page
isMicrosoftAuthEnabled	false	

isLocalAuthEnabled	true	Redirect to login page on clicking login from header Show password based login & Login with Microsoft option
isMicrosoftAuthEnabled	true	

isLocalAuthEnabled	false	Redirect to Microsoft login page on clicking login from header
isMicrosoftAuthEnabled	true	

Adding User

The user table schema has been updated in the sdcoe database.
Add new columns are

1. password { type: string } *email-password based login*
2. has_logged_in { type: integer } *reset password for first login*
3. is_active { type: integer } *to temporarily disable access*

Simply run the INSERT query in the database console or any database management tool.

Insert Query

```
INSERT INTO users(user_id, first_name, last_name, email_address, role, CDS_code)
VALUES(md5(random()::text || clock_timestamp()::text)::uuid, ?,?,?,?,?);
```

Example Query

```
# Adding Staff User
INSERT INTO users(user_id, first_name, last_name, email_address, role)
VALUES(md5(random()::text || clock_timestamp()::text)::uuid,'Tester', 'Staff',
'staff_user@gmail.com', 'staff');

# Adding District User (CDS Code is required)
INSERT INTO users(user_id, first_name, last_name, email_address, role, SELECT * FROM
CDS_code) VALUES(md5(random()::text || clock_timestamp()::text)::uuid, 'Tester',
'District', 'district_user@gmail.com', 'district', '37679830000000');
```

Sending a reset link to user to reset password

After adding a user to the database, we need to send the password reset link to change the password. Now, after resetting the password, the user can access the system.

In order to send the reset password link to the user's email, run the following command in the transcript-api module

```
$ cd sdcoe-transcript/transcript-api
$ yarn send-reset-link --email useremail@example.com
```

Example Script

```
$ cd sdcoe-transcript/transcript-api
$ yarn send-reset-link --email staff_user@youraccount.onmicrosoft.com
```

Now, the user staff_user@youraccount.onmicrosoft.com will receive an email with the password reset link. They can update the password and use it for new sessions.

Configuration changes in Web Application

This section talks about the additional configuration needed to set up the web application after Phase 1.3

Switching Verification Feature ON/OFF

For turning the verification off and on feature, we have configured it on our frontend application by adding an environment variable based on which the application enables/disables the verification tab.

```
# Enable/Disable Verification API i.e true or false
REACT_APP_IS_VERIFICATION_ENABLED=${SDCOE_REACT_APP_VERIFICATION_ENABLED}
```

Possible values for the above env variables are tabulated below:

REACT_APP_IS_VERIFICATION_ENABLED	true	Validate transcript tab enabled
	false	Validate transcript tab disabled

Credentials

Some of the demo credentials to the application deployed at Linux server.

Demo credentials	
person+district@yourdomain.net	Reset password from email received
person+staff@yourdomain.net	Reset password from email received

If the reset token is expired, one can send the reset link to the user by running the following command in the transcript-api module.

```
$ cd sdcoe-transcript/transcript-api
$ yarn send-reset-link --email person+district@yourdomain.net

$ yarn send-reset-link --email person+staff@yourdomain.net
```

Demo Credentials

Staff User	
Email Address: staff@yourdomain.com	Password: password
District User	
Email Address: district@yourdomain.com	Password: password