Implementing 3-D Secure 2.2 authentication using the SDK

Correct authentication via 3-D Secure 2.2 on a merchant's page can be implemented using the SDK.

The 3DSWebSDK is a lightweight JavaScript that allows merchants to easily invoke *3DS Method* and *Challenge Request* messages for browser-based transactions.

Installing the Web SDK

To install 3DSWebSDK, you need to import the ready-made JavaScript into your enterprise's HTML page.

Installation

```
<!DOCTYPE html>
<html>
    <head>
        ...
        <script src="nca-3ds-web-sdk.js" type="text/javascript" />
        ...
    </head>
    <body>
    ...
    </body>
</html>
```

The script will attach the 3DSWebSDK to the JavaScript window object.

The 3DSWebSDK object contains the following operations:

Operation	Description
init3DSMethod	Creates an HTML structure and attaches a form with a single input (*threeDSMethodData*) and automatically submits it to the *threeDSMethodUrl*.
createIframeAndInit3DSMethod	Creates an iFrame with an HTML structure and attaches a form with a single input (*threeDSMethodData*) and automatically submits it to the *threeDSMethodUrl* and attaches the frame to the container. If specified, a callback will be executed when the frame is loaded. Additionally, an optional timeout duration and timeout callback can be specified to handle cases where the ACS does not complete the 3DS Method in time.
init3DSChallengeRequest	Creates an HTML structure and attaches a form with a single input (*creq*) and automatically submits it to the *acsUrl*.
createIFrameAndInit3DSChallengeRequest	Creates an iFrame with an HTML structure and attaches a form with a single input (*creq*) and automatically submits it to the *acsUrl* and attaches the frame to the container. If specified, a callback will be executed when the frame is loaded. Additionally, an optional timeout duration and timeout callback can be specified to handle cases where the ACS page is not loaded in time.
getBrowserData	Collects browser-specific data (*browserJavaEnabled*, *browserJavascriptEnabled*, *browserLanguage*, *browserColorDepth*, *browserScreenHeight*, *browserScreenWidth*, *browserTZ*, *browserUserAgent*) can be used initially to populate the corresponding parameters in the payment request via the silentpay interface.

Technical documentation

Documentation

```
/**
 * Attach all methods to window.
 */
```

```
let nca3DSWebSDK = {};

/**
 * Creates an iframe, attach it to the rootContainer and submit 3DS Method form.
 *
 * @param threeDSMethodUrl - a FQDN endpoint to submit the 3DS Method request
 * @param threeDSMethodData - Base64-encoded 3DS Method Data value
 * @param frameName - name of the frame container. if not set it will be set to 'threeDSMethodIFrame'
 * @param rootContainer - the container where the iframe will be attached to.
 *                        If not set defaults to the JavaScript document.body object
 * @param onFrameLoadCallback - callback function attached to the iframe.onload event
 * @param timeoutSeconds - timeout in seconds (default: 0 = no timeout)
 * @param onFrameTimeoutCallback - callback function called when timeout occurs
 * @throws {Error} - throws error if there is a validation error
 * @returns {HTMLIFrameElement} - returns the generated iframe element
 */
nca3DSWebSDK.createIframeAndInit3DSMethod = createIframeAndInit3DSMethod;

/**
 * Initiates a 3DS Method request and submits the form the 3DS Method URL. It will automatically hide the
container
 * when initiating a 3DS Method request.
 *
 * @param threeDSMethodUrl - a FQDN endpoint to submit the 3DS Method request
 * @param threeDSMethodData - Base64-encoded 3DS Method Data value.
 * @param container - the iframe container where the form will be attached to. The container must have the
'name'
 *                    attribute set
 * @throws {Error} - throws error if there is a validation error
 * @returns {HTMLIFrameElement} - the container
 */
nca3DSWebSDK.init3DSMethod = init3DSMethod;

/**
 * Initiates a 3DS Challenge request and submits the form the ACS URL.
 *
 * @param acsUrl - the FQDN URL to submit the Challenge Request
 * @param creqData - Base64-encoded Challenge Request
 * @param container - the iframe container where the form will be attached to. The container must have the
'name'
 *                    attribute set
 * @throws {Error} - throws error if there is a validation error
 * @returns {HTMLIFrameElement} - the container
 */
nca3DSWebSDK.init3DSChallengeRequest = init3DSChallengeRequest;

/**
 * Creates an iframe, attach it to the rootContainer and submits 3DS Challenge Request.
 * @param acsUrl - the FQDN URL to submit the Challenge Request
 * @param creqData - Base64-encoded Challenge Request
 * @param challengeWindowSize - EMVCo assigned window size.
 *                              '01' -> 250px x 400px,
 *                              '02' -> 390px x 400px,
 *                              '03' -> 500px x 600px,
 *                              '04' -> 600px x 400px,
 *                              '05' -> Full screen, or full container content
 * @param frameName - name of the frame container. if not set it will be set to 'threeDSCReqIFrame'
 * @param rootContainer - the container where the iframe will be attached to.
 *                        If not set defaults to the JavaScript document.body object
 * @param onFrameLoadCallback - callback function attached to the iframe.onload event
 * @param timeoutSeconds - timeout in seconds (default: 0 = no timeout)
 * @param onFrameTimeoutCallback - callback function called when timeout occurs
 * @throws {Error} - throws error if there is a validation error
 * @returns {HTMLIFrameElement} - returns the generated iframe element
 */
nca3DSWebSDK.createIFrameAndInit3DSChallengeRequest = createIFrameAndInit3DSChallengeRequest;


/**
 * Collects browser-specific data.
 *
```

```
 * @returns {Object} An object containing various properties about the user's browser and device.
 * @property {boolean} browserJavaEnabled - Whether Java is enabled in the browser.
 * @property {boolean} browserJavascriptEnabled - Always true; indicates JavaScript is functioning.
 * @property {string} browserLanguage - The browser's preferred language setting (e.g., 'en-US').
 * @property {number} browserColorDepth - Screen color depth in bits per pixel.
 * @property {number} browserScreenHeight - Total height of the screen in pixels.
 * @property {number} browserScreenWidth - Total width of the screen in pixels.
 * @property {string} browserTZ - Time zone of the browser in IANA format (e.g., 'America/New_York').
 * @property {string} browserUserAgent - The full User-Agent string of the browser.
 */
nca3DSWebSDK.getBrowserData = getBrowserData;
window.nca3DSWebSDK = nca3DSWebSDK;
```

init3DSMethod Request - Example

init3DSMethod

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <script src="nca-3ds-web-sdk.js" type="text/javascript"></script>
    <title>Init 3DS Method - Example</title>
</head>
<body>
<!-- The first example shows how to generate a form and attach to an already defined iframe -->
<iframe name="iframeContainer" id="iframe"></iframe>

<!-- In the second example it will generate an iframe and attach to the #frameContainer element -->
<div id="frameContainer"></div>

<script type="text/javascript">
    iframe = document.getElementById('iframe');
    frameContainer = document.getElementById('frameContainer');
    nca3DSWebSDK.init3DSMethod('http://example.com', 'base64-encoded-3dsmethod-request', iframe);

    nca3DSWebSDK.createIframeAndInit3DSMethod(
        'http://example.com',
        'base64-encoded-3dsmethod-request',
        'threeDSMethodIFrameContainer',
        frameContainer,
        () => {
            console.log('Iframe loaded, form created and submitted');
        }
    );
</script>

</body>
</html>
```

init3DSMethod Request with Timeout - Example

init3DSMethod with Timeout

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <script src="nca-3ds-web-sdk.js" type="text/javascript"></script>
    <title>Init 3DS Method with Timeout - Example</title>
</head>
<body>
<div id="methodFrameWithTimeout"></div>

<script type="text/javascript">
    const methodContainer = document.getElementById('methodFrameWithTimeout');

    // 3DS Method with 10 second timeout
    nca3DSWebSDK.createIframeAndInit3DSMethod(
        'http://example.com/3dsmethod',
        'base64-encoded-3dsmethod-request',
        'methodIFrameTimeout',
        methodContainer,
        () => {
            console.log('3DS Method completed successfully');
        },
        10, // 10 second timeout
        () => {
            console.log('3DS Method timed out after 10 seconds');
        }
    );
</script>

</body>
</html>
```

init3DSChallengeRequest - Example

init3DSChallengeRequest

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <script src="nca-3ds-web-sdk.js" type="text/javascript"></script>
    <title>Init 3DS Challenge Request - Example</title>
</head>
<body>
<!-- This example will show how to initiate Challenge Reqeuests for different window sizes. -->
<div id="frameContainer05"></div>

<script type="text/javascript">
    container05 = document.getElementById('frameContainer05');

    nca3DSWebSDK.createIFrameAndInit3DSChallengeRequest(
        'http://localhost:8080/acs/challenge',
        'base64-encoded-challenge-request',
        '05',
        'threeDSCReq05',
        container05,
        () => {
            console.log('Iframe loaded, form created and submitted');
        }
    );
</script>

</body>
</html>
```

init3DSChallengeRequest with Timeout – Example

init3DSChallengeRequest with Timeout

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <script src="nca-3ds-web-sdk.js" type="text/javascript"></script>
    <title>Init 3DS Challenge Request with Timeout - Example</title>
</head>
<body>
<div id="challengeFrameWithTimeout"></div>

<script type="text/javascript">
    const challengeContainer = document.getElementById('challengeFrameWithTimeout');

    // 3DS Challenge with 60 second timeout
    nca3DSWebSDK.createIFrameAndInit3DSChallengeRequest(
        'http://localhost:8080/acs/challenge',
        'base64-encoded-challenge-request',
         '03',
        'challengeIFrameTimeout',
        challengeContainer,
        () => {
            console.log('Challenge authentication completed successfully');
        },
        60, // 60 second timeout
        () => {
            console.log('Challenge authentication timed out after 60 seconds');
        }
    );
</script>

</body>
</html>
```

To implement 3-D Secure authentication methods on merchant pages, you can use a ready-made SDK.

Back on top