Implementation guide for LSS

Table of contents

Version history

December 16th 2016	Version 2.0	TSS
August 31. 2016	Version 1.1	JGB
4th April	Version 1.1	MSP
28th March 2014	Version 1.0	MSP
13th March 2014	Version 0.93	BS
31th January 2014	Version 0.92	MSP
10th January 2014	Version 0.91	MSP
20th December 2013	Version 0.6	MSP
16th December 2013	Version 0.5	MSP

1 The purpose and audience of the document

The purpose of this document is to provide implementation guidelines for implementing support for LSS for NemID.

The document addresses developers at the LSS supplier organization, responsible for developing support of the LSS for NemID API in the relevant LSS product.

Summary of all documents in the LSS for NemID package:

General documentation

- Terms and concepts in LSS for NemID

Implementation documentation

- General technical specification
- LSS technical specification
- Implementation guide for LSS

Test documentation

- Testprocedures for LSS

Solution documentation

- Requirements feedback form for LSS

2 Introduction to LSS for NemID

For a general introduction to NemID and NemID for business, consult the current service provider package (TU-pakke) from Nets DanID.

For the rest of this document, knowledge of the general concept of NemID and NemID for business, as found in the current service provider package, is expected.

As of fall 2016 LSS for NemID support is included in the service provider package from Nets DanID.

As a supplier of LSS products, it is possible to integrate to the LSS for NemID. This makes it possible for employees in companies with the LSS, to use NemID for business from JavaScript enabled devices such as tablets, smartphones and ordinary computers with no need for specific plug-ins. Please note that service providers may or may not choose to support the LSS for NemID functionality in their services.

A LSS backend test stub which can be used as a local LSS backend in a test environment is available in the service provider package.

This document outlines the requirements for the LSS supplier's implementation of LSS for NemID.

The LSS supplier is required to consult also the service provider specific documentation.

3 Solution architecture

The purpose of the LSS for NemID is to provide the ability of employees of organisations with a LSS, to authenticate towards service providers and to digitally sign documents in formats Text, HTML, XML and PDF.

To support this functionality, the LSS supplier must implement authentication and signing capabilities towards their own LSS back-end.

The overall architecture is illustrated below.

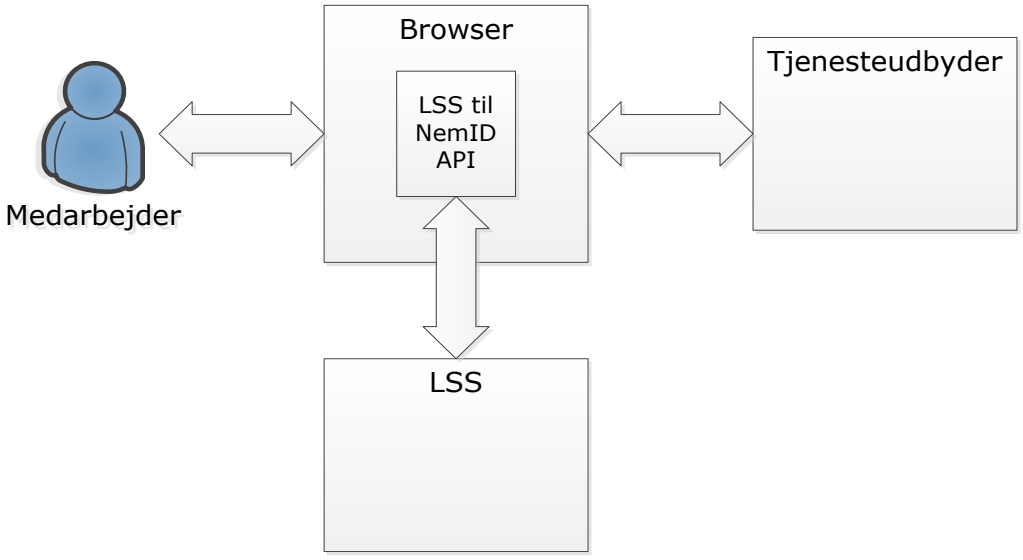

Fig 1: Overall architecture

3.1 Responsibilities of service providers and LSS suppliers

The responsibility and role of the service provider is the same as in existing NemID solutions.

The responsibility and role of the LSS supplier is to implement all the functionality required by the service provider, in order to provide authentication of the user and upon success return a XML-DSig message to the service provider.

1. The service provider either sends a challenge to be signed for authentication, or a document to be signed.

2. A login layout or the document to be signed is then presented to the user by the LSS supplier implementation.

3. The user authenticates towards the LSS, and the challenge or the document is signed and returned to the service provider as a XML-DSig Message.

4. The service provider validates the returned XML-DSig and extracts the required information from this.

Thus, it is the responsibility of the LSS supplier's implementation to present the text or document to be signed to the user. The implementation must support "What you see is what you sign" and apply the same restrictions to documents as is described in Nets DanID's service provider documentation.

3.2 LSS Client integration

The JavaScript LSS for NemID client is integrated with the service provider's page, using an *iframe* element, which enables a web page to allocate a segment of its area to another page. This is different from the Java applet client, where a Java applet was loaded as a page element.

Fig 2: The LSS Client (login)

An *iframe* element does not allow its content to expand beyond its borders, which requires allocation of sufficient room for every possible screen size, when it is created.

In short, the LSS Client must be setup with a fixed width and height.

Technical details on the UX, the flow, parameters etc. are provided in the technical documentation.

4 Implementation requirements

In order for service providers to offer uniform functionality to end users, regardless of which local signature server the organisation of the end user has implemented, common requirements are presented for the LSS implementation.

LSS implementations must fulfil the following requirements and document this through tests as specified in the document "Test procedures for LSS" as well as a report to Digitaliseringsstyrelsen, as specified in the document "Requirements feedback form for LSS".

Req. 1	The LSS supplier implementation must support both **authentication and signing**. For signing, the formats TEXT, HTML and XML must be supported. Support for signing of PDF is optional as specified in the technical specification.

Req. 2	The implementation must **follow the API** and technical specification provided in the document "Technical specification for LSS"

Req. 3	The implementation must be **tested successfully** following the specification provided in the document "Test specification for LSS". The test results must be communicated to Digitaliseringsstyrelsen in the template form provided in the document "Solution test template for LSS"

Req. 4	The implementation of the API by the LSS supplier must follow **common standards** for HTML, JavaScript etc. The solution as such, must be supported in standard browsers on Apple iOS and OSX, Linux, Android, Microsoft Windows and Windows Phone, without the need for installing extra software etc. The implementation should support the platforms supported by NemID as described here[1].

[1] https://www.nemid.nu/dk-da/support/faa_hjaelp_til_nemid/tekniske_krav/understoettede_programmer/internetbrowser/

Req. 5	The LSS supplier must describe conditions and prerequisites regarding the **performance** and **scalability** of the solution.

Req. 6	All requirements for organisations, implementing the suppliers LSS integration, must be communicated to Digitaliseringsstyrelsen in the "Requirement feedback form for LSS". This includes requirements for establishing a local DNS service for the bootstrap process and other technical requirements, in order for the solution to work. Requirements must also be communicated directly to customer organisations.

Req. 7	The LSS supplier must support the service providers and end users with the best possible error handling. This includes using common error codes and signaling under best practice as defined in the document "Technical specification for LSS". Error messages for end users must be meaningful to the user and suggest context dependent error remedy.

Req. 8	The LSS supplier must document in the "Requirement feedback form for LSS" how best security of the implemented solution is achieved by the organisation, hosting the LSS. This includes technical requirements, as well as awareness and training requirements. Protection of mobile devices, operations servers and operations environment must be described and required by the LSS supplier. Requirements must also be communicated directly to customer organisations.

Req. 9	The relevant requirements from the OCES certificate policy must be met by the LSS implementation.

Req. 10	The LSS implementation must be optimised for performance. Notably the API must be efficient with respect to - fast download - fast start up on device - efficient NemID operations with least network traffic - best possible performance experience for the user, avoiding blocking of GUI updates etc.

Req. 11	The LSS implementation must use standardised NemID dialogues and UX, following best practices and recommendations put forward in the LSS for NemID materials. The LSS supplier must document LSS for NemID UX in the "Solution test template for LSS" reporting.

Req. 12	The UX, implemented by the LSS supplier, should follow the usability recommendations put forward in the LSS for NemID documentation, otherwise the LSS supplier must test and document that the usability of the implemented UX is at least on par with the recommendations put forward in the LSS for NemID documentation. Where no recommendations exist, the supplier must ensure usability through own tests.

Req. 13	The LSS implementation must support best possible handicap accessibility.

Req. 14	Requirements from "Persondatalovens foreskrifter" on the protection of personal information and "Sikkerhedsbekendtgørelsen" must be met by the LSS implementation.

Req. 15	The LSS supplier must address functional or security related errors in a swift way, and provide updates and assistance to its customers, in order to keep the LSS installations as up to date as possible.

Req. 16	The LSS supplier must display the REQUESTISSUER parameter to the end user if CLIENTFLOW is logon.

Req. 17	In order not to log valid user-ids and passwords for NemID OTP, typed in by mistake by the user, exact values of user-id and password must not be logged in the LSS backend. Statistics about failed logins should be logged for supporting blocking after ex. 5 failed logons. The GUI for LSS shall be easy to distinguish by the user from the NemID keycard GUI. It is recommended that the LSS GUI contains the name og logo of the LSS organisation of the user.

Req. 18	The LSS supplier must perform a security review of its implementation and is recommended to perform a vulnerability scan according to the markets best practise. Specifically the LSS supplier should: - Consider using closures and anonymous functions for JavaScript - prevent cross site scripting - address risks of DOS attacks - address risk of cross site request forgery - address risk of replay of credentials - address risk of code injection - set eventual session cookies not accessible from javascript - ensure that ssl-for-nemid-server is accessible on HTTPS ONLY at the customers installations.

Req. 19	The code of the LSS backend implementation must be validated according to best practice. This includes the following steps - Code review - Javascript validation using JsLint/JsHint - Markup validation using 'http://validator.w3.org/' - Testing wrt. The procedures outline in the document 'LSS recommended testprocedures for LSS for NemID'

Req. 20	The UI of the LSS client must be scalable to a reasonable extend. The LSS client must be scalable down to the minimum size of 200x275 pixels

Req. 21	LSS implementations supporting API_VERSION 1.1.0.0 must implement issuance of certificates as described in general technical specification

5 Implementation recommendations

The following section describes recommendations for implementation of the LSS.

Rec. 1	The UX, implemented by the LSS supplier should be recognisable by the users as being another NemID / "Digital Signatur" solution. A layout, not deviating unnecessarily from the known layout of the existing NemID solutions, will help users recognise the solution and work flow. See the section "LSS visual guideline" for a visual proposal.

Rec. 2	The users should be able to find a reference to their local helpdesk inside the iframe. This information should be available in a similar manner across different LSS supplier's solutions. [Note: to be established] See the document "LSS visual guideline" for a visual proposal.

Rec. 3	For signing flows, it is highly recommended that the LSS-supplier utilize the entire allocated space inside the iframe in a dynamic way, in order for the service provider to be able to scale the frame according to the screen size of the user's device.

Rec. 4	The LSS supplier should display the content of the REQUESTISSUER parameter if CLIENTFLOW is signing.

Rec. 5	The LSS supplier should handle as many error situations as possible, before returning an error to the service provider.

Rec. 6	The LSS supplier should verify the parameter signature and audit log the subject serial number of the certificate passed in the SP_CERT parameter along with the timestamp.

Rec. 7	The LSS supplier could log the performed signatures.

Rec. 8	The LSS supplier should consider an alternative to cookies when keeping track of the users session. This is due to the fact that some browsers do not allow third party cookies..

6 LSS visual guideline

This section specifies the visual guidelines for the LSS implementation.

The example pictures shown in this section are taken from the LSS back-end test stub.

The first line 'TU Example' is the content of the REQUESTISSUER parameter. Some service providers has a long REQUESTISSUER string, which can be challenging to fit into the limited space.

The question mark button displays a small help message assisting the user in the flow. It is here one should fit in the reference to the local helpdesk.

The LSS back-end test stub gives the user the option of remembering his ID. This usually gives the user a good experience but can be challenging to implement given recommendation not to use cookies (Rec 8)

Fig 3: Logon UX with neutral "LSS implementation"

When implementing the password input field one must in mind that the certificate policy disallows caching of passwords.

In figure 3 an example of a user with multiple certificates is shown. Not all LSS suppliers allow users to have multiple certificates assigned to a single user. If the LSS supports multiple certificates per user, the user should be given the option to select which one of his valid certificates he wants to use. It is important to display sufficient information about the certificates to be able to distinguish them from one another.

In the LSS test stub the internal error messages are displayed between the certificate selection box and the login button to allow the user to get on faster by typing new and correct data as opposed to having to acknowledge a given error message. This is the recommended approach.

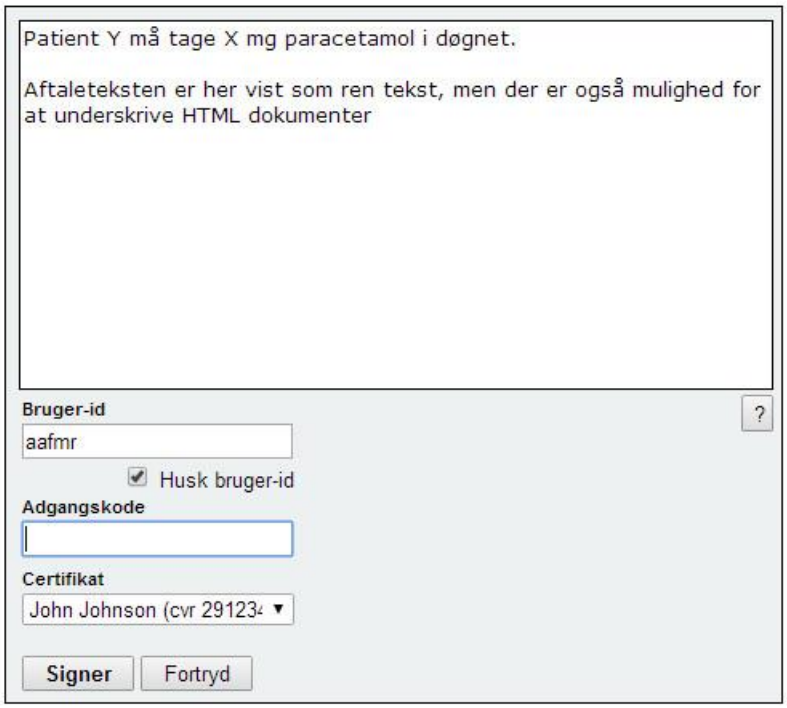

Fig4: Signing UX with neutral "LSS implementation"

For signing flows the most important requirement is the "what you see is what you sign", basically meaning that the document signed is presented correctly to the user. It is important that the LSS backend verifies with best effort that the document presented to the user corresponds to the document signed by the LSS Client and sent back to the service provider.

Another challenge is that the LSS Client is designed to support mobile platforms such as tablets and smartphones with limited screen size. It is recommended that LSS backend implementations support an option for the sign-text to be shown in full screen mode with extended zoom options.

7 Optimizing performance

In order to provide a fast and responsive user experience the LSS back-end implementation and setup should address several performance related issues states in this section.

The demo application and LSS back-end test stub provided in the LSS for NemID SP package contains a very light client consisting only of the required JavaScript enabling communication between the service provider and the LSS back-end. All CPU intensive tasks are done server side at either the LSS back-end or the service provider.

7.1 Scalability

The LSS back-end must be able to scale according to the load of concurrent LSS Clients.

The setup should be a high availability application able to respond quickly to each active LSS Client.

Bandwidth and CPU load should be considered for the setup for the expected number of concurrent user. Testing this during development is highly recommended.

The entire LSS back-end setup must be considered with respect to scalability - local DNS servers, deploy of trust to devices, bandwidth for users running over VPN, web servers and so forth.

7.2 Start-up time of the LSS Client

It is important that the LSS Client is loaded and ready for user interaction as fast as possible and that the interaction with the client is responsive.

It is recommended that the LSS back-end implementation ensures that the response time of the back-end is at a minimum and keeping the amount of data transferred to the client to a minimum.

Generally it should not take seconds before the LSS Client is ready for interaction.

7.3 Optimizing the user experience

It is highly recommended that the layout and interaction with the LSS Client is responsive that operations performed in the background or server side do not block for relevant information and feedback in the client.

Blocking operations should show some form of *working please wait* content to the user. The LSS test stub provided in the LSS for NemID SP package uses the *spinning wheel* also known from NemID to provide users with a known a recognizable experience.