MPass Integration Guide

The e-Government Center
Government of Moldova

Confidential

Change Log

Version	Date	Description
1.0	01.05.2015	Final version ready for distribution
1.1	03.06.2015	Documented support for SAML Metadata and HTTP Redirect binding
1.2	17.02.2016	Changed IsCitizen attribute to IsResident, as IDNPs are applicable to residents
1.3	02.12.2016	Removed OfficialAddress and CurrentAddress attributes as they are not structured and not authentic. Removed NotificationChannel attribute as it is not useable.
1.4	12.11.2018	Added X-Frame-Options recommendation when processing LogoutRequest.
1.5	16.07.2020	Added AdministeredLegalEntity attribute. Updated service owner contacts. Changed staging environment address.

CONFIDENTIAL

Table of Contents

CONFIDENTIAL

1. Executive summary

MPass is a reusable governmental shared platform-level service the main scope of which is to offer secure authentication using a variety of authentication methods and provide information for further authorization decisions by the integrating systems. MPass enables a secure government-wide single sign-on (SSO) as well as single logout (SLO) for residents of Republic of Moldova, so that they don't have to remember multiple credentials for different services and not requiring them to visit or register in some other way directly with the service provider.

This document describes the technical interfaces exposed by MPass for information systems that will use MPass as authentication and authorization information provider. Its target audience is the development teams for those information systems.

The document contains the relevant information required for a complete understanding of MPass from the integration point of view. It contains integration-related technical details, security considerations, as well as describing the process of testing integration security.

This document is also accompanied by a .NET sample that exemplify the main interaction scenarios, i.e. performing SSO and SLO.

CONFIDENTIAL

2. Glossary of terms

Term	Definition
GET	A standard HTTP request method
IDNP	Personal identifier number (unique in Republic of Moldova)
IDNO	Organization identifier number (unique in Republic of Moldova)
POST	A standard HTTP request method
SAML	Secure Assertion Markup Language
SLO	Single logout
SSO	Single sign-on

CONFIDENTIAL

3. Introduction

3.1. Scope and target audience

This document describes the technical interfaces exposed by MPass for information systems that will use MPass as authentication and authorization information provider. Its target audience is the development teams for those information systems.

The details related to various authentication methods, such as using authorized client certificates, mobile signature, username/password, etc. provided by MPass are out of scope of this document.

3.2. Structure of this document

This document contains the relevant information required for a complete understanding of MPass from the integration point of view. It is also accompanied by samples that exemplify integration scenarios using certain technologies.

The recommended reading sequence are the following chapters:

- System context
- Interaction scenarios
- Integration development
- Security considerations

The remaining chapters are for reference purpose.

3.3. Notations

This document contains several notation styles; the following details the styles that have a degree of significance beyond the purpose of communicating information:

Yellow Highlighted Text – Text that is highlighted in yellow irrespective of font attributes (font type, italics, bold, underlined, etc.) means that the text is waiting clarification or verification.

Red Bold Text – Text that is red in color and bold, defines an important piece of information that must be read.

Italic Bold Text – Text that is bold and italic detail actual information or scripts that need to be executed, created, and copied from or to.

CONFIDENTIAL

4. Organizational context

4.1. Service owner

Service Owner	
Organization	e-Governance Agency of Moldova
General Point of Contact	
E-mail	office@egov.md
Technical Point of Contact	
Name	MPass Support
E-mail	support.mpass@egov.md

CONFIDENTIAL

5. System context

5.1. General system capabilities

MPass is a reusable governmental shared platform-level service the main scope of which is to offer secure authentication using a variety of authentication methods and provide information for further authorization decisions by the integrating systems. MPass enables a secure government-wide single sign-on (SSO) as well as single logout (SLO) for residents of Republic of Moldova, so that they don't have to remember multiple credentials for different services and not requiring them to visit or register in some other way directly with the service provider.

MPass is used as intermediary between various information systems and various authentication methods. Authentication methods differ significantly from the integration point of view, exposing various APIs that might involve direct user interaction through the browser to enter some additional data and/or access user's cryptographic device or interact with cryptographic devices that are not directly connected to user's PC. MPass integrates with these identity and authentication providers, hides the differences and exposes a single unified and secure interface to information systems that require authentication functionality.

Depending on requesting information system, MPass will provide various attributes about the authenticated user identity. These attributes might come from user certificate used for authentication, user's profile in MPass or external authentic systems. Using the provided identity attribute values, integrating systems can further perform authorization decisions inside their session established for the authenticated identity.

For actual authentication, MPass exposes web pages that guide the user through authentication method, method specific data input, authentication progress and result pages.

5.2. Service dependencies

MPass depends on the digital identity providers, so its availability and performance is directly influenced by the availability and performance of the services delivered by the providers.

5.3. Protocols and standards

MPass is using SAML v2.0 standard protocol and format for authentications. The following table contains a comprehensive list of references to standard specifications.

CONFIDENTIAL

SAML v2 Specification	Abstract
SAML Core	This specification defines the syntax and semantics for XML-encoded assertions about authentication, attributes, and authorization, and for the protocols that convey this information.
SAML Bindings	This specification defines protocol bindings for the use of SAML assertions and request-response messages in communications protocols and frameworks.
SAML Profiles	This specification defines profiles for the use of SAML assertions and request-response messages in communications protocols and frameworks, as well as profiles for SAML attribute value syntax and naming conventions.
SAML Authn Context	This specification defines a syntax for the definition of authentication context declarations and an initial list of authentication context classes for use with SAML.
SAML Metadata	This specification defines profiles for the dynamic exchange of SAML metadata among system entities regarding identifiers, binding support and endpoints, certificates and keys, and so forth.
SAML Security Considerations	This non-normative specification describes and analyzes the security and privacy properties of SAML.
SAML 2.0 Errata	This document lists approved errata to the SAML V2.0 OASIS Standard.

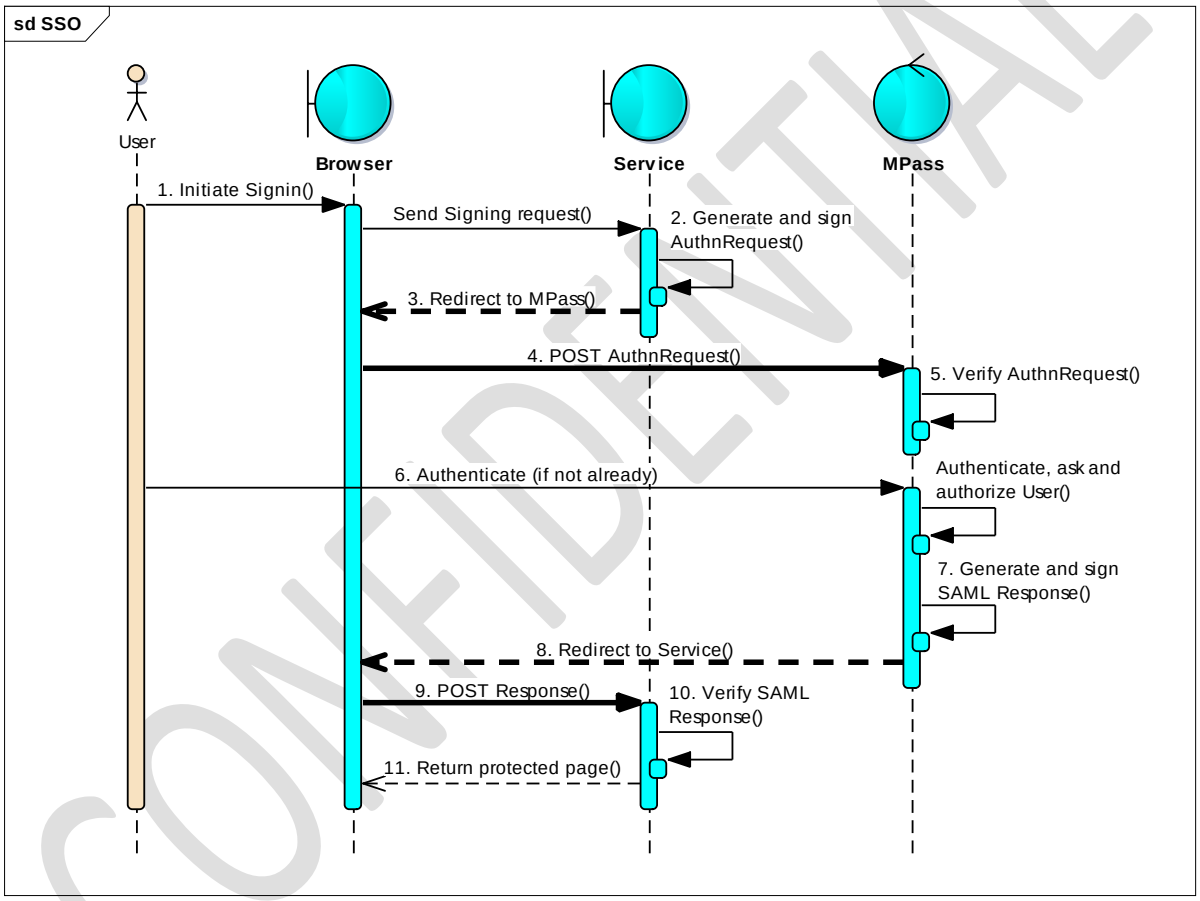
CONFIDENTIAL

6. Interaction scenarios

6.1. Authentication Process

The most important integration scenario with MPass is user authentication.

During this process, if the user is already authenticated, MPass session is not expired and authentication is not forced, user is not requested to proof its identity again. This is actually how single sign-on (SSO) is implemented.



CONFIDENTIAL

Here is the description of authentication process using MPass:

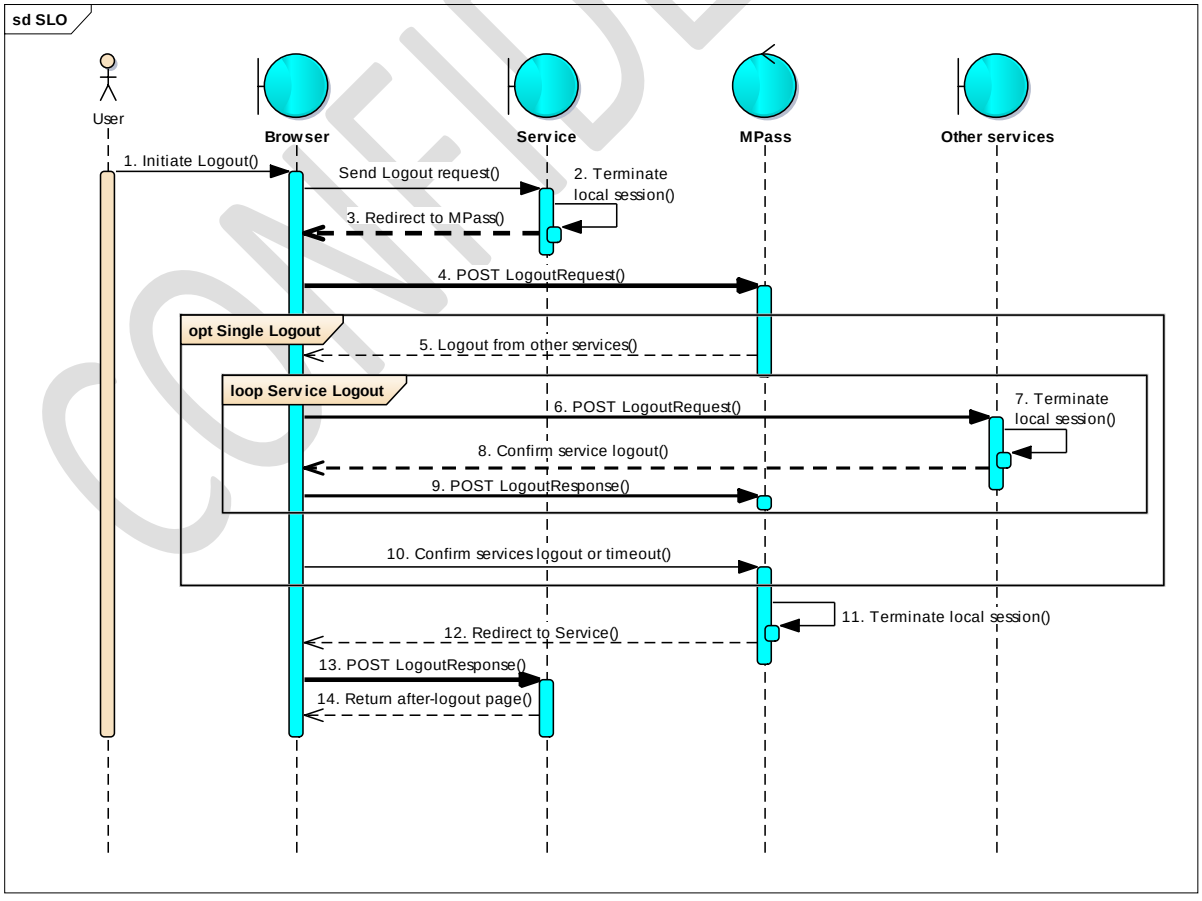
1. The user accesses some protected service resource or explicitly chooses to authenticate in the service. The Browser sends this request to the Service on behalf of user.
2. The Service generates an AuthnRequest (authentication request) and signs it using its private key. See AuthnRequest structure description for details.
3. The signed AuthnRequest is returned to the Browser in a special redirection page.
4. The Browser posts (using HTTP POST method) the request to MPass.
5. MPass verifies incoming AuthnRequest and the properties of service registration.
6. If user is not already authenticated or the authentication is forced, MPass interacts with

the user for authentication, authorization and requests user's consent to provide its identity attributes if needed.

7. MPass generates and signs a SAML Response with the result of authentication. Note that if AuthnRequest verification fails or user explicitly cancels or refuses the authentication, the SAML Response will be generated with an unsuccessful status. See Response structure description for details.
8. The signed Response is returned to the Browser in a special redirection page.
9. The Browser posts (using HTTP POST method) the request to Service.
10. The Service verifies the Response and creates its own session/cookie or handles the Response is any other specific way. For details on the correct way of this verification process, please refer to Security considerations.
11. The Service serves the protected resources to the now authenticated user until its local session expires or the user explicitly request logout (see below).

6.2. Logout Process

Because users can login into many services during an MPass session, from security point of view SSO is not fully implemented without a proper SLO (Single logout). Integrating services MUST implement both.



CONFIDENTIAL

Here is the description of logout process using MPass:

1. The user explicitly requests to logout. Its Browser submits this request to the Service.
2. The Service terminates its local session of the user, i.e. user will have to authenticate again to further access any protected resources.
3. The Service generates and signs a LogoutRequest and returns this request to the browser in a special redirection page.
4. The Browser posts (using HTTP POST method) the request to MPass.
5. If during user's MPass session, user has authenticated in other services, MPass generates and signs a LogoutRequest for each such service, returning them all to the Browser.
6. The Browser posts these requests to respective services.
7. Upon LogoutRequest receipt, each service validates the request, then terminates its local session of the user, i.e. user will have to authenticate again to further access service protected resources.
8. Each service then generates and signs a LogoutResponse to confirm the logout result and returns this response to the Browser in a special redirection page. Note that for correct logout processing when using HTTP POST, services must return the following header in HTTP response:

   ```
   X-Frame-Option: allow-from https://mpass.gov.md
   ```

9. The Browser sends all the resulted responses to MPass.
10. MPass is informed on results after all participating services confirm the logout or after a timeout (to handle the case for services that cannot confirm the logout).
11. MPass then terminates its local session of the user, i.e. user will have to authenticate again to access its MPass profile.
12. MPass generates and signs a LogoutResponse and returns it to the Browser.
13. The Browser posts (using HTTP POST method) the response to Service.
14. Finally, after handling the resulting LogoutResponse, the Service is free to return to the user any page that fits the needs.

CONFIDENTIAL

7. Integration development

This chapter describes the process of developing an integration.

7.1. Service registration

Before being able to interact with MPass, a Service must be registered accordingly in MPass. To perform such a registration, please generate a self-signed or provide any existing certificate file (in .cer file format) to Service owner.

For security reasons, Service test and production environments MUST use a different certificate and corresponding private keys MUST be kept as confidential as possible. MPass does not require access to Service private keys for integration.

7.2. Returned attributes

After a successful authentication and user consent (if required), MPass generates and returns a SAML Response with authenticated identity attributes. The list of the returned attributes is configurable as part of Service registration.

The following table contains the list of standard attributes.

Attribute Name	Type	Description
NameIdentifier	string (128)	Username or IDNP. This is a special attribute and it should be returned as NameID (i.e. SubjectAttribute) in SAML.
IsResident	Boolean	Specifies whether the user is verified to be a resident of Republic of Moldova.
FirstName	string (64)	User's first name or given name.
LastName	string (64)	User's last name or surname.
BirthDate	string (10)	User's birth date in "yyyy-MM-dd" format (e.g. "1990-12-31").
Gender	integer	User's gender. Allowed values: • 0 – Unspecified • 1 – Male • 2 – Female
EmailAddress	string (64)	User's e-mail address.
MobilePhone	string (16)	User's mobile phone number.

CONFIDENTIAL

HomePhone	string (16)	User's home phone number.
Language	string (2)	User's preferred language. Allowed values: "ro", "ru", "en".
AdministeredLegalEntity	string (512)	The name and identifier of the companies (zero or more) the user is administering in the following format: "Legal Entity Name IDNO" Notice that the IDNO is after the last space of the name.
IDNO	string (13)	User's organization identifier. This attribute is only available if the authentication was performed using an instrument which includes this value in the certificate.
CompanyName	string (128)	User's organization or company name. This attribute is only available if the authentication was performed using an instrument which includes this value in the certificate.

A Service can have custom attributes created (usually used for authorization purpose, such as Role, Permissions, etc.), assigned to identities and returned as part of the same SAML Response with values corresponding to the authenticated identity.

Please identify the set of required attributes (including custom attribute names and values) to be returned by MPass during the design phase of the Service and specify them as part of Service registration.

7.3. Network access

Because MPass interface is exposed to public, there is no need for special network configuration or access control list modifications. A developer can integrate with MPass using its local development machine and use a localhost address for AssertionConsumerServiceURL in AuthnRequest.

Note that, for security reasons, a localhost address is not accepted in MPass production environment.

7.4. Authentication methods

MPass provides several authentication methods. All strong authentication methods require a strong authentication instrument, which means that the private key of the person that authenticates is generated and held on special devices. It is in the integrator responsibility to obtain such a secure device from available providers.

Weak authentication methods (such as username/password) are discouraged and usually not enabled for any systems in production environment.

CONFIDENTIAL

7.5. System environments

There are 2 services environments available: a testing and a production environment.

Environment	SSO URL	SLO URL
Testing	https://mpass.staging.egov.md/login/saml	https://mpass.staging.egov.md/logout/saml
Production	https://mpass.gov.md/login/saml	https://mpass.gov.md/logout/saml

Integrations MUST be developed and tested within the testing environment only. To ensure high availability, no performance, security or any other kind of tests are allowed on production environment.

7.6. SAML Metadata

MPass exposes SAML metadata, conformant with SAML Metadata specification at the following URL:

Environment	Metadata Index	SAML Metadata URL
Testing	https://mpass.staging.egov.md/meta	https://mpass.staging.egov.md/meta/saml
Production	https://mpass.gov.md/meta	https://mpass.gov.md/meta/saml

The index page also includes links to MPass certificate used to sign SAML messages as Identity Provider.

CONFIDENTIAL

8. Security considerations

The following security considerations must be taken into account while developing an integration with MPass.

8.1. Authentication

MPass performs integrating systems authentication by verifying the signature of requests, which MUST be signed by using the private key of the requesting system.

MPass uses X509 certificates for system registration and actual signature verification.

Important. The description of the process of installing, registering or explicitly trusting the certificate and any certificate chain in the operating system or framework used by the integrating information systems has to be done accordingly, is specific to that environment and it's out of the scope of this document.

Integrating systems MUST accept and handle only valid MPass messages (SAML Response, LogoutResponse, as well as SAML SLO LogoutRequest).

The following checks MUST be performed for a correct and complete MPass message verification:

1. Verify SAML message signature and/or SAML assertion signature.
2. Check whether the SAML message is not expired (using some absolute configurable timeout and taking into account some expected time differences between servers' clocks) of the following SAML attributes, depending on your needs:
    a. @IssueInstant – always verify;
    b. Response/Assertion/AuthnStatement/@AuthnInstant – verify when you need to see how long ago the user has authenticated;
    c. Response/Assertion/Conditions/@NotOnOrAfter – verify when SAML assertion is signed;
    d. Response/Assertion/AuthnStatement/@SessionNotOnOrAfter – verify in special cases.
3. Check SAML message destination and/or audience:
    a. @Destination – always verify;
    b. Response/Assertion/Subject/SubjectConfirmation /SubjectConfirmationData/@Recipient – verify when SAML assertion is signed;
    c. Response/Assertion/Conditions/AudienceRestriction/Audience – always verify.
4. Check SAML responses to see if they have been initiated by your Service:
    a. Response/@InResponseTo – always verify;

CONFIDENTIAL

b. Response/Assertion/Subject/SubjectConfirmation/SubjectConfirmationData/@InResponseTo – verify when SAML assertion is signed.

5. Always check SAML response status code for Success in Response/Status/StatusCode/@Value attribute. The response MUST be handled as invalid whenever this code has a different value.

For more details and a comprehensive understanding of all security considerations, please refer to the following SAML standards document sections: [SAML Core, 3.2], [SAML Bindings, 3.5.5], [SAML Profiles, 4.1.4.3] and [SAML Security Considerations, 6.4].

8.2. Authorization

MPass does not perform actual authorization of the integrating systems or users by itself. MPass is a convenient user management instrument and acts as what is commonly referred to as Policy Information Point (PIP) and Policy Retrieval Point (PRP).

On user authentication MPass can return custom authorization attributes as configured in service registration. Actual authorization decision must be taken by integrating systems based on the set of returned attributes for the authenticated identity. This means that the systems act as actual Policy Decision Point (PDP) and Policy Enforcement Point (PEP), i.e. the actual decision, its correctness and enforcement is the responsibility of the integrating systems.

8.3. Confidentiality

To maintain the confidentiality of the attributes related to authenticated identities, integrating systems MUST use some encryption mechanism. Considering that the integrating systems are web-based, this means that all interactions with user's browser during authentication and logout MUST be performed using HTTPS (TLS).

Note that, although MPass can encrypt and sign SAML Assertions, this feature shall not be considered as enough for maintaining confidentiality.

8.4. Integration review and audit

The security of MPass integrating systems heavily depends on the security of the integration. Please review the provided Test cases which include security related tests to thoroughly test the security of the integration.

CONFIDENTIAL

9. Integration testing

9.1. Test cases

Here are some basic test cases that could be added to the test suite of the systems that integrates with MPass.

Test Case ID	TC_FUNCT_01		
Description	Service initiated authentication		
Initial Conditions	User not authenticated into the Service and MPass		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass, no errors shown	**Pass / Fail**
2	Authenticate in MPass	Browser redirected back to the Service as logged in	**Pass / Fail**

Test Case ID	TC_FUNCT_02		
Description	Single sign-on through MPass		
Initial Conditions	User not authenticated in the Service User authenticated directly in MPass		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back (with or without authentication consent) to the Service as logged in, no errors shown	**Pass / Fail**

CONFIDENTIAL

Test Case ID	TC_FUNCT_03		
Description	Aborted authentication		
Initial Conditions	User not authenticated into the Service and MPass		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass for authentication	**Pass / Fail**
2	Cancel the authentication in MPass	Browser redirected back to the Service without authentication and no errors are shown by the Service	**Pass / Fail**

Test Case ID	TC_FUNCT_04		
Description	Service initiated logout		
Initial Conditions	User authenticated into Service via MPass		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Logout" button/link of the Service	Browser redirected to MPass and redirected back (with or without authentication consent) to the Service as logged out, no errors shown	**Pass / Fail**
2	Access any Service protected resource	Access to resource is denied and/or user is redirected to MPass for authentication	

CONFIDENTIAL

Test Case ID	TC_FUNCT_05		
Description	MPass initiated logout (i.e. single logout)		
Initial Conditions	User authenticated into Service via MPass		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Logout" link in MPass	After performing single sign-out, MPass shows that the user is not authenticated	**Pass / Fail**
2	Access any Service protected resource	Access to resource is denied and/or user is redirected to MPass for authentication	

Test Case ID	TC_SEC_01		
Description	Check SAML Response signature validation		
Initial Conditions	User not authenticated into Service, but authenticated in MPass Only the following option is checked in SAML Advanced Options: "Do not sign SAML Response"		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back to the Service without successful authentication, as SAML Response is not signed	**Pass / Fail**

CONFIDENTIAL

Test Case ID	TC_SEC_02		
Description	Check SAML Response signature validation certificate		
Initial Conditions	User not authenticated into Service, but authenticated in MPass Only the following option is checked in SAML Advanced Options: "Use compatible certificate for signing"		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back to the Service without successful authentication, as SAML Response is signed with invalid certificate	**Pass / Fail**

Test Case ID	TC_SEC_03		
Description	Check SAML Response is not expired		
Initial Conditions	User not authenticated into Service, but authenticated in MPass No option is checked in SAML Advanced Options Service server clock changed to several hours in the future		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back to the Service without successful authentication, as SAML Response is expired	**Pass / Fail**

CONFIDENTIAL

Test Case ID	TC_SEC_04		
Description	Check SAML Response is not too new		
Initial Conditions	User not authenticated into Service, but authenticated in MPass Only the following option is checked in SAML Advanced Options: "SAML Response IssueInstant is specified in local time, instead of UTC"		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back to the Service without successful authentication, as SAML Response is expired (2 or 3 hours in the future for Moldova time zone)	**Pass / Fail**

Test Case ID	TC_SEC_05		
Description	Check if SAML Response Destination is validated		
Initial Conditions	User not authenticated into Service, but authenticated in MPass Only the following option is checked in SAML Advanced Options: "Do not specify Destination in SAML Response"		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back to the Service without successful authentication, as SAML Response/@Destination is not specified	**Pass / Fail**

CONFIDENTIAL

Test Case ID	TC_SEC_06		
Description	Check if SAML Response InResponseTo is checked for		
Initial Conditions	User not authenticated into Service, but authenticated in MPass Only the following option is checked in SAML Advanced Options: "Do not specify InResponseTo in SAML Response"		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass and redirected back to the Service without successful authentication, as SAML Response/@InResponseTo is not specified	**Pass / Fail**

Test Case ID	TC_SEC_07		
Description	Check if SAML Response InResponseTo is validated		
Initial Conditions	User not authenticated into the Service and MPass No option is checked in SAML Advanced Options		
Step	**Task**	**Expected Result**	**Actual Result**
1	Access the "Login" button/link of the Service	Browser redirected to MPass for authentication	**Pass / Fail**
2	Abort user's session in the Service (restart the server or delete it from session store) so that the generated AuthnRequest/@ID is lost	User session aborted	**Pass / Fail**
3	Authenticate in MPass	Browser redirected back to the Service without successful authentication, as SAML Response/@InResponseTo is now invalid	**Pass / Fail**

CONFIDENTIAL

10. SAML Reference

10.1. Overview

MPass currently supports only one standard way of integration for SSO and SLO: SAML 2.0 via HTTP POST binding. Although MPass also supports HTTP Redirect binding, its usage is not recommended.

To completely understand how to integrate with MPass, **please read [SAML Binding, 3.5]**.

The above reference describes how to correctly generate, encode and submit requests, as well as how to handle errors and responses.

MPass uses AuthnRequest/Response and LogoutRequest/LogoutResponse pairs to implement SSO and SLO.

For a detailed description of security related aspects, **please read [SAML Security Considerations, 6.4]**.

10.2. SAML Structures

This section describes all the involved SAML structures in details.

10.2.1. AuthnRequest structure

An authentication request, referred to as AuthnRequest, is generated by Service and submitted to MPass via user's browser to direct him for authentication.

Here is an example of AuthnRequest generated by .NET sample.

```
<saml2p:AuthnRequest ID="_92a43de2-b6ab-460c-abe5-8580f828bf76" Version="2.0"
IssueInstant="2014-10-20T08:26:26.9933782Z"
Destination="https://mpass.staging.egov.md/login/saml"
AssertionConsumerServiceURL="http://localhost:50341/Account/Acs"
xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml2:Issuer>http://sample.testmpass.gov.md</saml2:Issuer>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"><!-- ... --></Signature>
  <saml2p:NameIDPolicy AllowCreate="true" />
</saml2p:AuthnRequest>
```

The following table describes structure attributes.

Element or @Attribute	Type	Value	Description
@ID	any unique string	required	Generated by Service. This value will be returned back by MPass in Response/@InResponseTo attribute and can

CONFIDENTIAL

			be checked in order to stop replay attacks.
@Version	Constant	2.0	This value is required and must be set to "2.0".
@Destination	URL	required	Must be set to MPass SSO URL.
@AssertionConsumerServiceURL	URL	optional, default value: set in Service settings	This URL is used by MPass to submit the SAML Response.
@ProtocolBinding (not present in above example)	URI	optional, default: urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST	URI reference that identifies the SAML protocol binding to be used when returning the Response. Although MPass supports HTTP POST and Redirect binding, Redirect binding is not recommended.
@ForceAuthn (not present in above example)	bool	optional, default: false	When set to "true", MPass will required the user to authenticate even if user already has a valid and authenticated MPass session.
@IsPassive (not present in above example)	bool	optional, default: false	When set to "true", MPass will not interact with the user and submit the SAML Response with current user authentication status.
Issuer	string	required	Represents the issuer of this AuthnRequest, usually as an URL.
Signature	Xml	required	Contains the signature of this AuthnRequest, applied using the private key of the Issuer.
NameIDPolicy	Xml	required	MPass currently ignore this policy element, but all Services are required to include it (exactly as in example above, including AllowCreate attribute) for compatibility.

10.2.2. Response structure

MPass is generating a successful (i.e. user authenticated) or failed (i.e. user failed or cancelled the authentication) SAML Response in response to AuthnRequest. The Response is submitted to the Service via user's browser in the same way as the request. Before handling the response, Service must validate it accordingly.

CONFIDENTIAL

Here is an example of successful SAML Response returned by MPass in response to the above AuthnRequest.

```
<samlp:Response ID="_7722d8dc-3401-4e16-b789-8c4db923ea86"
InResponseTo="_7b874d06-2b14-4dbe-b177-3a70140a5b66" Version="2.0" IssueInstant="2014-
10-20T08:39:48.786Z" Destination="http://localhost:50341/Account/Acs"
Consent="urn:oasis:names:tc:SAML:2.0:consent:prior"
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml:Issuer>http://devmpass.gov.md</saml:Issuer>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"><!-- ... --></Signature>
  <samlp:Status>
    <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
  </samlp:Status>
  <saml:Assertion Version="2.0" ID="_b3ef6fcb-8db0-44dc-9a96-a3ca008ec55c"
IssueInstant="2014-10-20T08:39:48.786Z">
    <saml:Issuer>http://devmpass.gov.md</saml:Issuer>
    <saml:Subject>
      <saml:NameID>admin</saml:NameID>
      <saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
        <saml:SubjectConfirmationData NotOnOrAfter="2014-10-20T08:49:48.786Z"
Recipient="http://localhost:50341/Account/Acs"
InResponseTo="_7b874d06-2b14-4dbe-b177-3a70140a5b66" />
      </saml:SubjectConfirmation>
    </saml:Subject>
    <saml:Conditions NotOnOrAfter="2014-10-20T08:49:48.786Z">
      <saml:AudienceRestriction>
        <saml:Audience>http://sample.testmpass.gov.md</saml:Audience>
      </saml:AudienceRestriction>
    </saml:Conditions>
    <saml:AuthnStatement AuthnInstant="2014-10-20T08:38:19.703Z"
SessionIndex="c5b3376a-a437-4b9c-addf-a3ca008e5883"
SessionNotOnOrAfter="2014-10-20T08:48:19.697Z">
      <saml:SubjectLocality Address="127.0.0.1" />
      <saml:AuthnContext>
<saml:AuthnContextClassRef>urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTran
sport</saml:AuthnContextClassRef>
      </saml:AuthnContext>
    </saml:AuthnStatement>
    <saml:AttributeStatement>
      <saml:Attribute Name="FirstName">
        <saml:AttributeValue>Admin</saml:AttributeValue>
      </saml:Attribute>
      <saml:Attribute Name="LastName">
        <saml:AttributeValue>Adminovich</saml:AttributeValue>
      </saml:Attribute>
    </saml:AttributeStatement>
  </saml:Assertion>
</samlp:Response>
```

The following table describes structure attributes.

Element or @Attribute	Type	Value	Description
@ID	any unique string	required	Generated by MPass, provided for traceability purposes.
@InResponseTo	string	required	This value is set to the incoming AuthnRequest/@ID. Service can use it for traceability or check it to stop authentications by 3rd parties even with a

			stolen Service private key.
@Version	constant	2.0	This value is required and must be set to "2.0".
@IssueInstant	datetime	required, value in UTC	The instant of time when this Response was generated. To minimize the risk of replays, Service must not accept old responses.
@Destination	URL	required	The URL of the destination for this Response. Can be checked by Service to stop Responses aimed for a different Service or deployment environment.
@Consent	URI	optional	Additional information regarding user consent. Can be set to explicit (the consent was explicitly given during the authentication) or prior (the consent was persisted during previous authentication).
Issuer	string	required	Represents the issuer of this Response, which is obviously one of the MPass instances (dev, test or production environment).
Signature	xml	required	Contains the MPass signature of this Response, applied using MPass the private key. Service must verify this signature.
Status	xml	required	Contains Response status details.
Status/StatusCode /@Value	string	required	Specifies the top-level status of this Response. Possible values are Success (authentication succeeded), Requestor (failure on Service side), Responder (failure on MPass or user side) and VersionMismatch. See [SAML Core, 3.2.2.2].
Status/StatusCode /StatusCode/@Value	string	optional	Second-level status code that specifies the reason of status. See [SAML Core, 3.2.2.2].
Status/StatusMessage	string	optional	Additional human-readable message related to Response status.
Assertion	xml	included in successful Response	SAML Assertion that includes the details of the authentication, authentication conditions and authenticated identity. If required by Service, MPass can sign this similarly to the whole Response.
Assertion/@ID	any unique string	required	Specifies the ID of this Assertion.

CONFIDENTIAL

Assertion/@IssueInstant	datetime	required, value in UTC	The instant of time when this Assertion was generated.
Assertion/Issuer	string	required	Represents the issuer of this Assertion, which is obviously one of the MPass instances (dev, test or production environment).
Assertion/Subject	xml	required	Contains details of authenticated identity (i.e. its subject) and authentication confirmation.
Assertion/Subject /NameID	string	required	Authenticated identity name identifier. This is usually the IDNP of the authenticated resident, but it can also be an unique username of MPass user.
Assertion/Subject /SubjectConfirmation /SubjectConfirmationData /@Recipient	URL	required	Specifies the expected recipient of this Assertion. Same as Response/@Destination. Required by SAML standard.
Assertion/Subject /SubjectConfirmation /SubjectConfirmationData /@InResponseTo	string	required	Same as Response/@InResponseTo. Required by SAML standard.
Assertion/Conditions	xml	required	Contains conditions under which this Assertion must be considered.
Assertion/Conditions /@NotOnOrAfter	datetime	required, value in UTC	Specifies the time until this Assertion can be considered as valid.
Assertion/Conditions /AudienceRestriction /Audience	string	required, could be many	Specifies the expected audience for this Assertion. Service must check for match with its AuthnRequest/Issuer. Note that an Assertion can have many audiences.
Assertion/AuthnStatement	xml	required	Contains authentication details.
Assertion/AuthnStatement /@AuthnInstant	datetime	required, value in UTC	Specifies the exact time instant of authentication.
Assertion/AuthnStatement /@SessionIndex	string	required	Specifies the ID of the MPass session under which the identity was authenticated. Must be used in LogoutRequest/SessionIndex to specify the exact session to log out from.
Assertion/AuthnStatement /@SessionNotOnOrAfter	datetime	required, value in UTC	Specifies the time until MPass session is valid.

CONFIDENTIAL

Assertion/AuthnStatement /SubjectLocality /@Address	IP address	optional	Specifies the IP address of the client that performed the authentication as seen by MPass.
Assertion/AuthnStatement /AuthnContext /AuthnContextClassRef	string	optional	Specifies the instrument or method used for authentication. See [SAML Core, 2.7.2.2] and [SAML Authn Context, 3.4].
Assertion /AttributeStatement	xml	required	Contains authoritative attribute values (i.e. claims) that represent the, are assigned or associated with authenticated identity.
Assertion /AttributeStatement /Attribute/@Name	string	required	The name of the provided attribute.
Assertion /AttributeStatement /Attribute/AttributeValue	string or xml	optional, could be many	Contains the value of the attribute. Note that some attributes can naturally have multiple values (e.g. identity Role in some Service).

10.2.3. LogoutRequest structure

A LogoutRequest is generated either by a Service (as a result of user explicit request to logout) or MPass (in order to perform single logout, as a result of another service requesting user logout in the same authentication session or direct user logout in MPass UI).

Here is an example of LogoutRequest generated by .NET sample.

```
<saml2p:LogoutRequest ID="_bdeed8e2-5c79-4d5b-8f93-2620f1219753" Version="2.0"
IssueInstant="2014-10-20T08:51:44.8184811Z"
Destination="http://devmpass.gov.md/logout/saml"
xmlns:saml2p="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:saml2="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml2:Issuer>http://sample.testmpass.gov.md</saml2:Issuer>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"><!-- ... --></Signature>
  <saml2:NameID>admin</saml2:NameID>
  <saml2p:SessionIndex>ae94d066-67ba-4296-b149-a3ca0091b24e</saml2p:SessionIndex>
</saml2p:LogoutRequest>
```

The following table describes structure attributes.

Element or @Attribute	Type	Value	Description
@ID	any unique string	required	Generated by MPass/Service, provided for traceability purposes.
@InResponseTo	string	required	This value is set to the incoming LogoutRequest/@ID. The requesting party can use it for traceability or check it to stop authentications by 3rd parties even with a stolen party's private key.

CONFIDENTIAL

@Version	constant	2.0	This value is required and must be set to "2.0".
Issuer	string	Required	Represents the issuer of this LogoutResponse, usually as an URL.
Signature	xml	Required	Contains the signature of this LogoutRequest, applied using the private key of the Issuer.
NameID	string	Required	Contains the username (usually IDNP) of the logged in user, as previously returned by MPass in Response/Assertion /Subject/NameID.
SessionIndex	string	Required	Contains the ID of the session that was previously returned by MPass in Response/Assertion /AuthnStatement/@SessionIndex.

10.2.4. LogoutResponse structure

A LogoutResponse is returned as a response to a LogoutRequest, either by MPass (in response to a Service request) or a service (as a response for single logout request from MPass).

Here is an example of successful LogoutResponse returned by MPass in response to the above LogoutRequest.

```
<samlp:LogoutResponse ID="_34650a27-f348-41cc-a2ef-e481d89a727c"
InResponseTo="_bdeed8e2-5c79-4d5b-8f93-2620f1219753" Version="2.0"
IssueInstant="2014-10-20T08:52:13.207Z"
Destination="http://localhost:50341/Account/AfterLogout"
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml:Issuer>http://devmpass.gov.md</saml:Issuer>
  <Signature xmlns="http://www.w3.org/2000/09/xmldsig#"><!-- ... --></Signature>
  <samlp:Status>
    <samlp:StatusCode Value="urn:oasis:names:tc:SAML:2.0:status:Success" />
  </samlp:Status>
</samlp:LogoutResponse>
```

The following table describes structure attributes.

Element or @Attribute	Type	Value	Description
@ID	any unique string	required	Generated by MPass or Service. This value will/must be returned back by MPass/Service in LogoutResponse/@InResponseTo attribute and can be checked in order to stop replay attacks.
@Version	constant	2.0	This value is required and must be set to "2.0".

@IssueInstant	datetime	required, value in UTC	The instant of time when this LogoutResponse was generated. To minimize the risk of replays, requesting party must not accept old responses.
@Destination	URL	required	The URL of the destination for this LogoutResponse. Can be checked by requesting party to stop Responses aimed for a different party or deployment environment.
Issuer	string	required	Represents the issuer of this LogoutResponse, usually as an URL.
Signature	xml	required	Contains the signature of this LogoutResponse, applied using the private key of the Issuer.
Status	xml	required	Contains LogoutResponse status details.
Status/StatusCode /@Value	string	required	Specifies the top-level status of this LogoutResponse. Possible values are Success (logout succeeded), Requestor (failure on requesting side), Responder (failure on responder side) and VersionMismatch. See [SAML Core, 3.2.2.2].
Status/StatusCode /StatusCode/@Value	string	optional	Second-level status code that specifies the reason of status. See [SAML Core, 3.2.2.2].
Status/StatusMessage	string	optional	Additional human-readable message related to LogoutResponse status.

CONFIDENTIAL

11. Samples

11.1. Sample SAML messages

This guide must be accompanied by sample SAML messages in the following files:

1) AuthnRequest.xml – sample AuthnRequest generated by .NET sample;
2) Response.xml – sample of a successful SAML Response generated by MPass in response to the above AuthnRequest;
3) Response - Cancelled.xml – sample of failed SAML Response generated by MPass as a result of user authentication cancellation;
4) LogoutRequest.xml – sample LogoutRequest generated by .NET sample;
5) LogoutResponse.xml – sample of a successful LogoutResponse generated by MPass in response to the above LogoutRequest.

11.2. .NET Sample

This document is accompanied by a .NET integration sample. If the ZIP archive is encrypted (for e-mail filtering pass-through purposes), the password is "mpass" (without quotes).

11.2.1. Software requirements

The sample is build using Visual Studio 2013 based on ASP.NET 4.5, MVC 5.2 using C# and NuGet package manager. Although there are no 3rd party libraries or licenses required to build the sample, NuGet requires an internet connection while downloading packages for the first build.

11.2.2. Sample overview

After opening the MPass.Sample.sln, please go to Web.config and modify the appropriate configuration elements in applicationSettings section. All of the settings related to MPass integration are placed in this section.

You'll find the important code in AccountController. Just run the sample and follow the code.

SAML message handling is implemented in SamlMessage helper class. Feel free to copy in your own solution for easier integration.

Please observe the following:

- How AccountController.Login builds an AuthnRequest, signs it and submits it to MPass;
- How AccountController.Acs handles the Response and uses SamlMessage. LoadAndVerifyLoginResponse to verify the resulting response;
- How AccountController.Logout builds and submits a LogoutRequest and

CONFIDENTIAL

AccountController.AfterLogout handles LogoutResponse;

- How AccountController.SingleLogout handles LogoutRequest and generates a LogoutResponse for MPass;
- How Redirect view is used to submit a SAML request as well as a SAML response;
- How OutputCacheAttribute is applied to Login and Logout actions to prevent caching, as recommended by [SAML Binding, 3.5.5.1];
- SamlMessage throws ApplicationException on any verification failure and this error is not handled by AccountController on the purpose to let you finish a correct error handling implementation, according to your Service needs;
- RelayState is returned as part of response submission and can contain any value useful for your Service (note that according to [SAML Binding, 3.5.3], RelayState must not exceed 80 bytes in length);
- Comments provided in the source code for further integration customization specific to your Service.

CONFIDENTIAL