FEi Systems

Version 1.2

continuum™
THE ASAM CRITERIA DECISION ENGINE
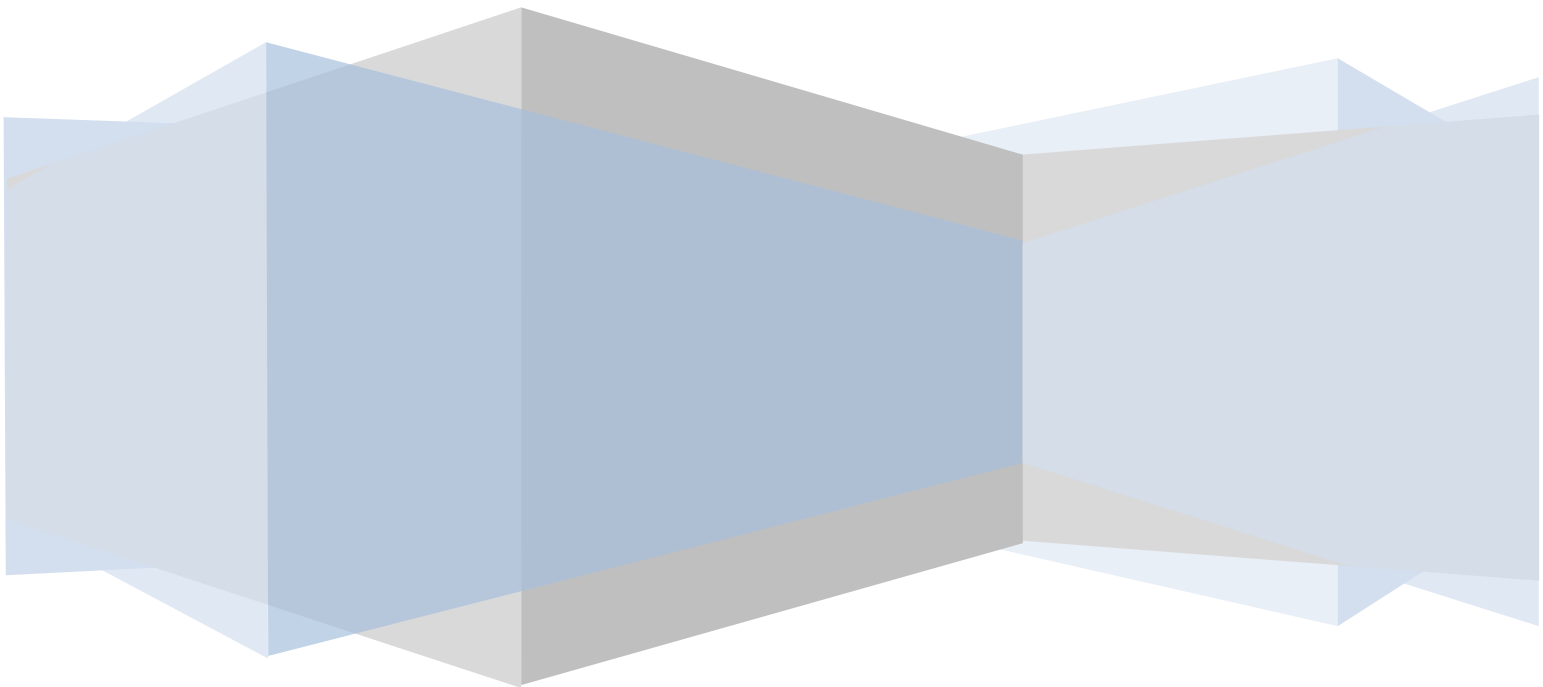
ASAM CS Single Sign-On

Table of Contents

1. Purpose

This document aims at providing a guide for integrating a system with an installation of Continuum, The ASAM Criteria Decision Engine™**.**

This document assumes you are familiar with core concepts such as:

- RESTful Services
- HTTP Requests
- Certificates

2. Single Sign-On Overview

The Continuum™ Integration API implements a simple and secure single sign on between a trusted web application and an installation of the ASAM CS that has been setup for integration mode. The Single Sign On API has been designed in a way to easily create a seamless experience for the user without having to have a federated security environment.

Using this implementation the remote application can provide links for their users that will automatically redirect them into Continuum™. This all works by making the remote application the trusted source for identifying the user and ensuring that user has access to the patients Continuum™ assessments. The user will not have to re-sign in with Continuum™ because the request to access the Continuum™ site will contain a digital signature verifying the request came from the remote application and that the request has not been tampered with in any way.

Before access can be granted to Continuum™, the client must send an HTTPS POST and if the authentication succeeds the client will be automatically redirected to the resource they are trying to access.

The URL for this service typically looks like this:

https://<server>/SingleSignOn/

There is a single route defined for the SingleSignOn that must include the following parameters. If you pass both a patientId and an assessmentId the system will open that assessment. If you do not pass an assessmentId it will take you to the patient list screen.

This service will be called by creating a Form post containing the parameters defined below. The parameters must be passed in the same order as was used in creating the token.

POST message Parameters:

Name	Description	Required
EhrId	The Id of the EHR you are coming from	Yes
OrganizationId	The Id of the Organization you are coming from	Yes
UserId	The Id of the user who is requesting the resource.	Yes

UserName	The name of the user who is requesting the resource.	Yes
UserEmail	The email of the user who is requesting the resource.	Yes
PatientId	The Id of the patient you are trying to access.	Yes
Timestamp	Timestamp the message was created. This must be within 1 minute of the current time for the single sign-on to be successful. The format should follow RFC 1123, "ddd, dd MMM yyyy HH':'mm':'ss 'GMT'"	Yes
AssessmentType	The type of assessment.	If AssessmentId
AssessmentId	The Id of the assessment being accessed. By specifying this Id the user will automatically be redirected to this Assessments Edit screen.	No
Token	This is the digitally signed SHA-1 hash of the current message. See below for instructions on creating this hash.	Yes

An example of the form POST content would be:

EhrId=1&OrganizationId=1&UserId=user-1&UserName=Fred
Jones&UserEmail=fred.jones@test.com&PatientId=patient-
1&Timestamp=Fri%2C+30+Oct+2015+17%3A51%3A02+GMT&Token=UGF0aWVudElkPXBhdGllbnQtMSZVc2
VySWQ9dXNlci0xJlVzZXJOYW1lPUZyZWQgSm9uZXMmVXNlckVtYWlsPWZyZWQuam9uZXNAdGVzdC5jb20m
VGltZXN0YW1wPTIwMDgtMTEtMDFUMTk6MzU6MDAuMDAwMDAwMC0wNzowMA==

3. Creating Token

The Token parameter is created by hashing the concatenation of the rest of the parameters and then digitally signing that hash and finally converting the signed bytes to a Base64 string. We require the SHA-1 algorithm for performing the hash.

You must have a valid API key for each organization and the API Key must be the last parameter. The names and values are case sensitive and must be posted in the same order as they are in the token.

The format of the concatenation of the parameters is "parameter-name=value" with an "&" in-between each parameter:

PatientId=123
&UserId=user-Id
&UserName=Fred Jones
&UserEmail=fred.jones@test.com
&Timestamp=Fri, 30 Oct 2015 17:51:02 GMT
&ApiKey=MKSL6EWVFQ3JZNDDJLLXRPVKPHOPQJBP9CF1O1SY2E

Then you would use that string as input to the SHA-1 algorithm to create the hash value of that string. And then you would sign that hash using a Certificate.

After signing the string, the ApiKey should be removed from the string.

Here is example C# code for performing the hashing and digital signature using a certificate:

```csharp
public static string SignWithCertificate(string text)
{
    // Open certificate store of current user
    var my = new X509Store(StoreName.My, StoreLocation.LocalMachine);
    my.Open(OpenFlags.ReadOnly);

    // Look for the certificate with specific subject
    var csp = my.Certificates.Cast<X509Certificate2>()
        .Where(cert => cert.Subject.Contains("CN=TrustedCert"))
        .Select(cert => (RSACryptoServiceProvider)cert.PrivateKey)
        .FirstOrDefault();

    // Hash the data
    var sha1 = new SHA1Managed();
    var data = Encoding.Unicode.GetBytes(text);
    var hash = sha1.ComputeHash(data);

    // Sign the hash
    var signedBytes = csp.SignHash(hash, CryptoConfig.MapNameToOID("SHA1"));
    return Convert.ToBase64String(signedBytes);
}
```

4. Release Notes

- V1.2

  o AssessmentType was added and is required if AssessmentId is specified.

  o AssessmentId is optional again.

- V1.1

  o UserEmail is no longer optional

  o AssessmentId is no longer optional, access to the patient list screen is no longer available.

  o Timestamp format was changed to use standard RFC 1123 "ddd, dd MMM yyyy HH':'mm':'ss 'GMT'"