

# Managing Legacy Single Sign-on (SSO)

Collaboration Suite 7.5.5

Updated May 26, 2017

---

Medicity, Inc. has prepared this document for sole use by Medicity employees, licensees, and customers. The information contained herein is the property of Medicity, Inc. and shall not be copied, photocopied, translated, or reduced to any electronic or machine readable form, either in whole or in part, without prior written approval from Medicity, Inc. All materials are confidential and proprietary.

© 2017 by Medicity, Inc., a wholly owned subsidiary of Aetna. All Rights Reserved.

Medicity and the Medicity logo are trademarks and registered names. Community Health Record (CHR), Medicity Collaboration Suite, and Organize are unregistered trademarks of Medicity, Inc. Microsoft, Windows, and Windows Internet Explorer are registered trademarks of Microsoft Corporation. Adobe and Acrobat are registered trademarks of Adobe Systems, Inc. All other trademarks or registered trademarks are the property of their respective owners.

Software Version: 7.5.5

DV 1.9

Medicity, Inc.  
The Medicity Building  
257 East 200 South  
Suite 1300  
Salt Lake City, UT 84111

Phone 801.322.4444 | Fax 801.322.4413

[www.medicity.com](http://www.medicity.com)

# Table of Contents

**Overview..... 5**

    Supported Sign-On Modes ..... 5

        Impersonation Authentication (IA) .....5

        User Based Authentication (UA).....6

    Embedding Options ..... 6

**Chapter 1   Configuring Single Sign-on in CHR..... 7**

    Accessing Single Sign-On Maintenance ..... 7

    Creating a New SSO Impersonation Account..... 9

    Editing an SSO Impersonation Account.....11

    Utilizing the Transaction Log .....12

**Chapter 2   Support for External Applications..... 15**

    Utilizing HTTP GET .....15

        Expected Parameters in Request Query String .....15

        Expected Parameters in Payload .....16

        Patient Context Data in Payload.....17

    Using HTTP POST .....18

    Setting up SSO Support in the CHR .....20

**Frequently Asked Questions ..... 22**

**Appendix A: SSO Test Client and Payload Simulator ..... 26**



# Overview

This document explains how to create Single Sign-On (SSO) accounts for users of the Medicity Collaboration Suite. The Collaboration Suite (CS) supports SSO requests from external applications at both the organization level and the user level. A single SSO service account can now address all IA and UA requests for an organization or entity.

This chapter includes the following sections:

- Supported Sign-On Modes
- Embedding Options

## Supported Sign-On Modes

The Collaboration Suite supports two sign-on modes from external applications for authenticating users. The purpose in having two modes is so that when authenticating, the system can attach a specified security profile to the SSO request. The two modes are Impersonation Authentication (IA) and User Based Authentication (UA).

## Impersonation Authentication (IA)

Also known as the Single-Sign-On (SSO) mode, this mode is used when you need to create a single profile that grants rights to a specific, unique group. For example, you might create an Impersonation Authentication role for your emergency room (ER) users. You could use this SSO role to launch the Patient Search and Chart component of Community Health Record (CHR) for ER users. Or you might create a SSO profile for an in hospital pharmacy. This IA profile might take a pharmacy user directly to the Prescription Monitoring Program functionality.

A single user account must be set up inside the Collaboration Suite for each unique IA profile. These profiles can only be created by a System Administrator. This user profile (or account) can be given rights to access all clinical data across all data sources with no restrictions, or it can be subject to specific rules depending on your need. Since IA profiles are shared, all AI users will have the same access rights once they are logged in to the Collaboration Suite. The UserFirstName and UserLastName arguments passed in will be used inside the application for logging and display purposes (so you know who is accessing the application).

**Note:** User information is not logged if transaction logging is disabled. See the section, *Utilizing the Transaction Log* on page 12 for more information about this feature.

## User Based Authentication (UA)

This is similar to the authentication process on the CHR login page, except we are using SSO Authentication in place of the user's password. The external application will pass along any CHR User's Credential (User's Login) to authenticate. The CHR will then authenticate the external application under the provided user account information. This model enables external systems to use existing CHR user accounts for authentication rather than using the SSO "user" account.

## Embedding Options

In previous versions of ProAccess, the patient's chart data (standing alone) could be embedded in another application. When utilized, ProAccess would display only that patient's chart without the option to search additional chart data. Starting with Community Health Record (CHR) 7.4.2, embedding is once again available.

# Chapter 1

## Configuring Single Sign-on in CHR

Single Sign-On is a web based client side application. Collaboration Suite System Administrators can use this tool to create new SSO accounts, edit existing SSO accounts, or delete existing accounts. This section explains these tasks.

This chapter includes the following topics:

- Accessing Single Sign-On Maintenance
- Creating a New SSO Impersonation Account
- Editing an SSO Impersonation Account
- Using the Transaction Log

## Accessing Single Sign-On Maintenance

Before you can use the Medicity Collaboration Suite (MCS), you must first access the application Log In page. You can access the Login page at [https://medicity.com/medicity/login](#).

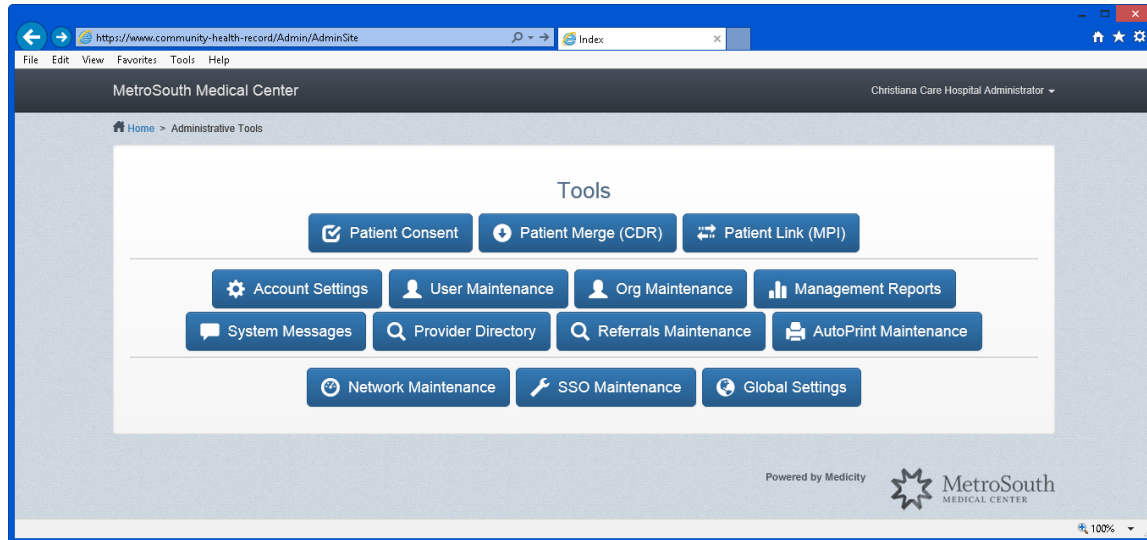
If you are a CS System Administrator, the application displays the administrator Home page when you first log in. The page contains links to the various administrative tools in the application. And it also displays System Message. You will click the SSO Maintenance button to access the Single Sign-On Maintenance page.

To access single sign-on maintenance:

1. In your browser address bar, enter [https://medicity.com/medicity/login](#).  
*The browser displays the MCS Login page.*
2. Enter your username in the **Username** field; then enter your password in the **Password** field.

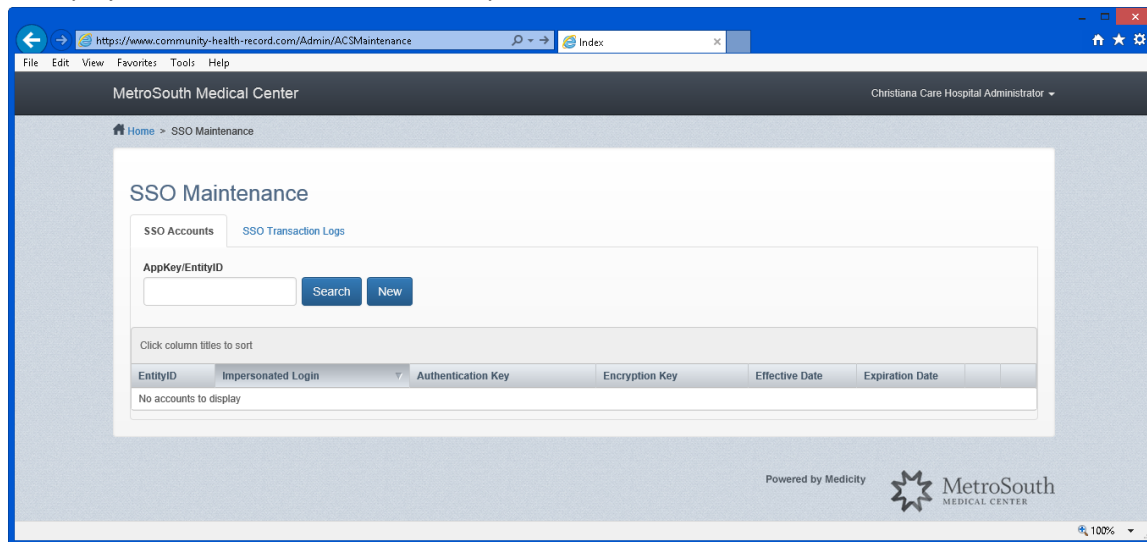
3. Click the **Login** button.

*After verifying your username and password, CS displays the Home or Administrative Tools page.*



4. Click the **SSO Maintenance** button.

*CS displays the SSO Maintenance Search panel.*





# Creating a New SSO Impersonation Account

You can create or generate new Single Sign-On impersonation account as-needed. If you have not already accessed the SSO Maintenance page, do this first.

To create a new SSO impersonation account:

1. Click the **New** button.  
*CS displays the SSO New/Edit page.*

The screenshot shows a web browser window with the URL <https://www.community-health-record.com/Admin/ACSMaintenance>. The page title is "MetroSouth Medical Center" and the user is logged in as "Christiana Care Hospital Administrator". The main content area is titled "SSO Maintenance" and has two tabs: "SSO Accounts" and "SSO Transaction Logs". The "New SSO" form is displayed with the following fields and values:

Field	Value
Entity ID	
Impersonated Login (IA only)	
Authentication Key	91ebc684-8427-4697-9636-f0abc1fa6131
Encryption Key	779ae615-4f78-4f05-920e-e8c2b342eb0d
Effective Date	05/21/2017
Expiration Date	05/21/2018

Buttons: Save, Save & Exit, Cancel, Generate (for keys), and a calendar icon for dates.

2. Complete the fields displaying on the New/Edit SSO user page.  
For more information about the SSO New/Edit page fields, see Table 1.

3. Click the **Save & Exit** button.

*The Collaboration Suite saves the new impersonation account, and then it re-displays the SSO Maintenance Search panel.*

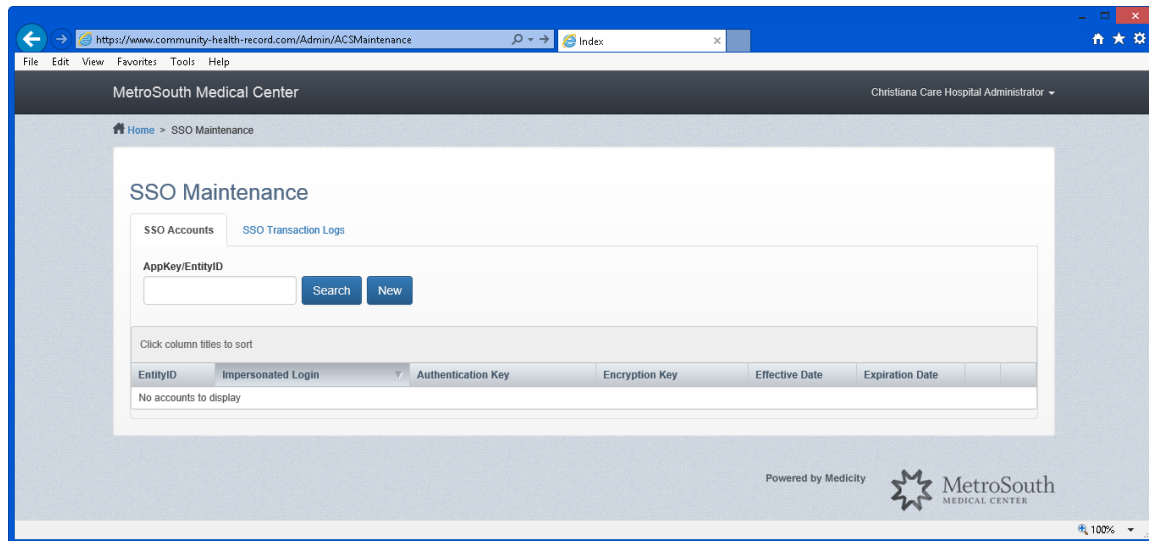


Table 1: SSO New/Edit Page Fields

Label	Options
<b>EntityID(new only)</b>	Enter a unique Entity ID in this field. For UA, this could be the ACO name, an ORG name, or a facility name. For IA, this ID might identify a certain user group (example: City Center Hospital Pharmacy).  <b>Note:</b> The application will display an error message if you attempt to save a new SSO entry using an existing EntityID. And CS will not save a new entry until the issue is corrected.
<b>ImpersonatedLogin (IA only)</b>	Enter an existing CHR Username in this field. If left empty, that application will enter the string "(UA)" in this field. Any Username value entered into this field will be validated against all existing CHR user records. If no match is found, an error message will pop up and the entry will not be saved.
<b>AuthenticationKey</b>	This is an auto generated field. You may click the Generate Key button to the right of the field to generate a new authentication key.
<b>EncryptionKey</b>	An auto generated field. User can click the Generate Key button at the right end of the field to generate a new encryption key.
<b>Effective Date-Time</b>	Enter the date that you want the keys to become active. By default, the Collaboration Suite will enter the current date in this field.
<b>Expiration Date-Time</b>	Enter the date that you want the SSO Impersonation Login account to expire. By default, this is set to expire one year from the current date.

# Editing an SSO Impersonation Account

A Collaboration Suite System Admin user can edit any SSO impersonation account. The system administrator will use EntityID or ImpersonatedLogin ID to search for the account. You may use partial entries to do a search.

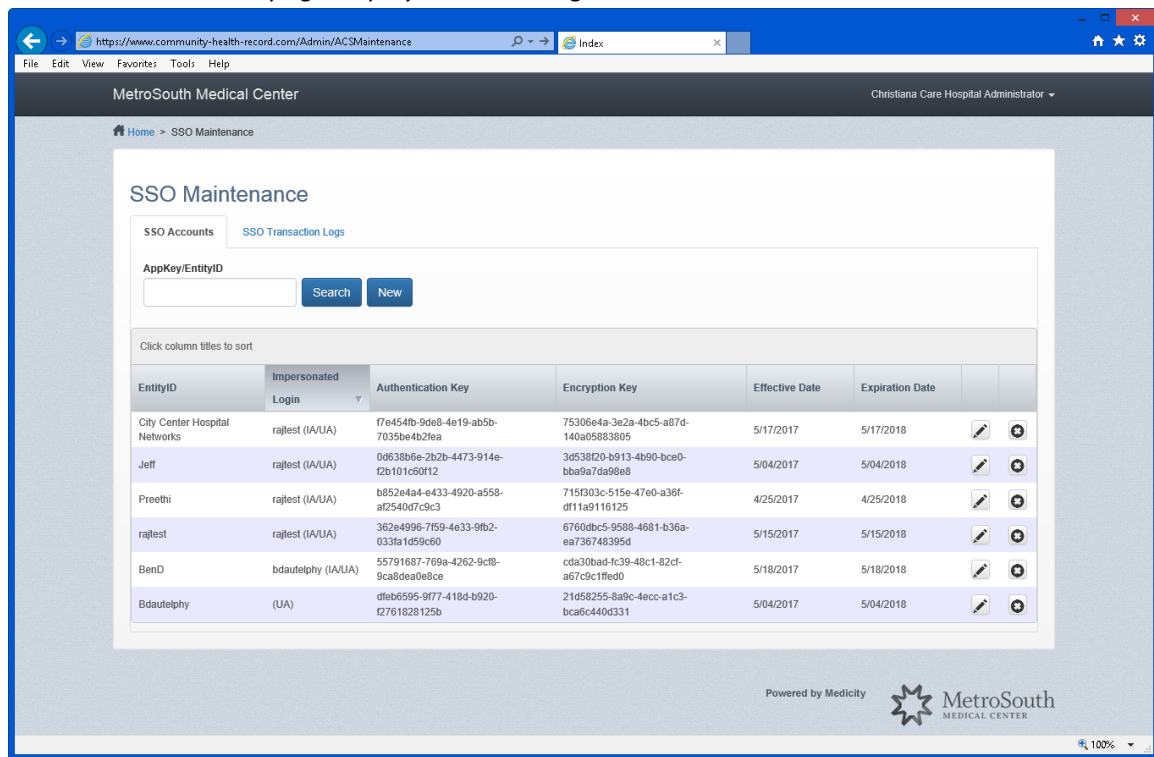
## To edit an SSO impersonation account:

1. Enter the impersonation Entity ID or AppKey in the **Search By AppKey/EntityID** field.

**Note:** If search button is clicked while no value is entered into Search textbox, all existing SSO entries will be returned.


2. Click the **Search** button.

*The SSO Maintenance page displays all matching results in the List SSO Entities list.*




3. Do either of the following:

### To edit an impersonation account:

- a) Select  (the Edit button) to the right of the impersonation account that you want to edit. *CS displays the account record on the SSO New/Edit page.*
- b) Edit the SSO impersonation account fields as needed.

### To delete an impersonation account:

- a) Click  (the Delete button) to the right of the impersonation account that you want to delete. *The CS displays a deletion confirmation dialog box.*
- b) Click the **OK** button. *The CS deletes the impersonation account and then redisplays the Search SSO panel.*

### To edit an impersonation account:

For more information about the SSO New/Edit page fields, see Table 1.

- c) Click the **Save & Exit** button.  
*The Collaboration Suite saves the updates and then re-displays the SSO Maintenance Search panel.*

### To delete an impersonation account:

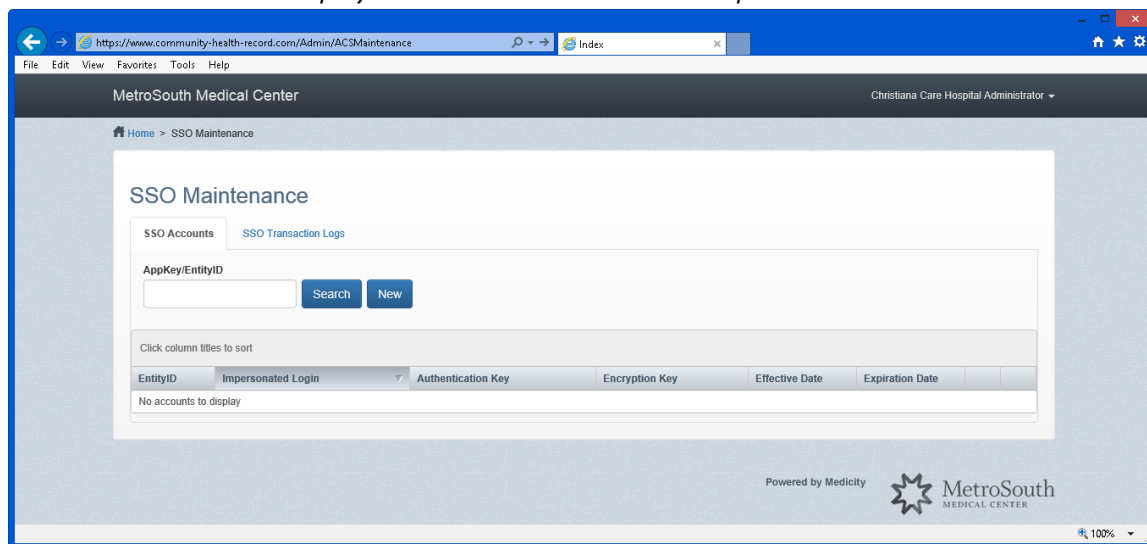
## Utilizing the Transaction Log

System admin user can use the Single Sign-On Transaction Log to troubleshoot failed SSO login events. The transaction log can be used two different ways. If you enable the transaction log, then the Collaboration Suite (CS) will track all impersonation logins. If you disable transaction logging, then CS will track only failed impersonation login attempts. You can also delete logged entries.

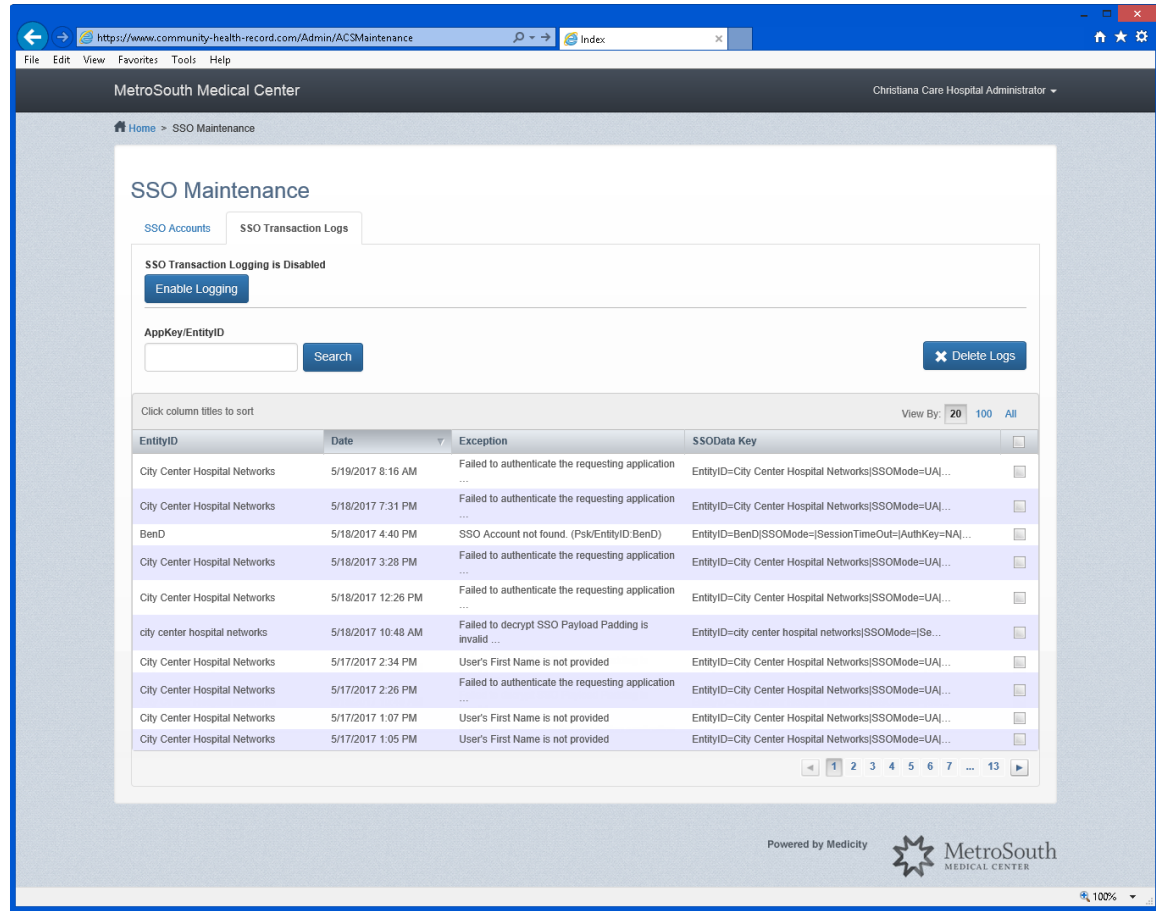
**Note:** While the Transaction Log is an effective tool for troubleshooting failed events, this log is not the most effective tool for auditing all logins. If you or your organization needs to monitor or audit CHR login events, we recommend using the Login Reports. These reports can be found in your CHR Management Reports utility.

### To Utilize the Transaction Log:

1. If you have not already access SSO Maintenance, do this first.  
*The Collaboration Suite displays the SSO Maintenance Search panel.*



- Click the **SSO Transaction Logs** tab.  
*CS displays the Transactions Log list.*



- Do either of the following:

#### To view an SSO Transaction:

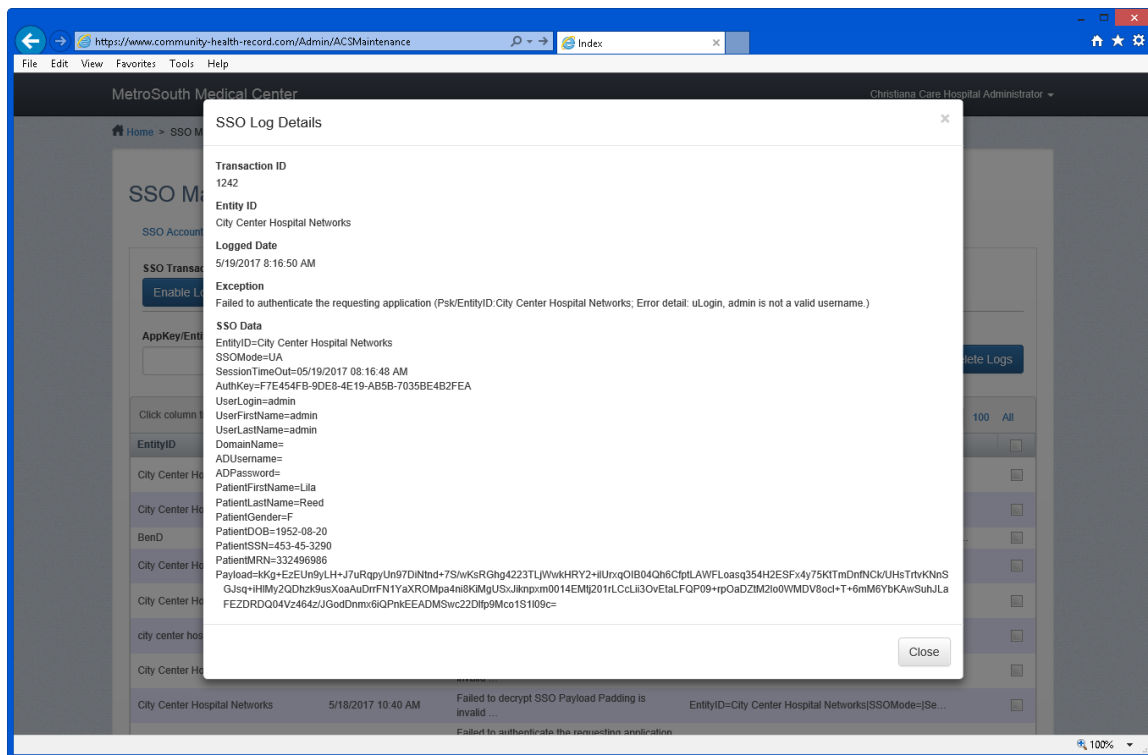
Click any of the transaction events.  
*SSO Maintenance displays the full event log (see .Figure 1: SSO Log Details Panel on 14).*

#### To delete a logged event:

- Select each transaction event that you wish to delete, or click the Select All option to delete all events.
- Click the **Delete Logs** button.  
*CS deletes the logged event.*

**Note:** Once executed, the logged event is permanently deleted. Exercise caution when using the Delete Logged event feature.

Figure 1: SSO Log Details Panel





# Chapter 2

## Support for External Applications

Community Health Record accepts Single Sign-on (SSO) requests from external applications using two payload request models. Your external applications can access CHR SSO using isEmbedded as a payload argument (HTTP GET). Or you can access CHR using an HTTP Post Request. This chapter describes the requirements for both.

This chapter includes the following sections:

- Utilizing HTTP GET
- Using HTTP POST
- Setting up SSO Support in the CHR

### Utilizing HTTP GET

External applications can launch CHR by issuing an HTTP (or HTTPS) GET Request using the following URL:

<http://ProAccessBsseURL/ACS/SSO?psk=BASE64EncodedKey&payload=BASE64EncodedPayload>

For production, a secure channel needs to be utilized:

<https://ProAccessBsseURL/ACS/SSO?psk=BASE64EncodedKey&payload=BASE64EncodedPayload>

### Expected Parameters in Request Query String

CHR expects two parameters in the query string during HTTP GET payload request:

- **“psk” Parameter (private system key):** There are three unique keys used by CHR in Single-Sign-on implementation: **AppKey/IdpEntityID**, **EncryptionKey**, and **AuthenticationKey**. The **psk** parameter will hold the value of the **AppKey/IdpEntityID**. This key is used by CHR to identify the external application. The value should be Base64 Encoded when it's transmitted to CHR. Each external application should have a different set of keys and will be provided per implementation.

**Note:** A user's authentication key is part of the payload. However, the encryption key is used to decrypt the payload data, and you should never transmit this over the wire.

- **“Payload” Parameter:** Used with HTTP Get only, this is the encrypted data which consists of user information, patient context information, and

**Security Note:** You may use HTTP or HTTPS for GET (either will work). The app key is 64-bit encoded, and it is used only for database look-up. For the payload itself, we use 128 bit AES encryption. The payload contains the user authentication key and patient PHI/PII. But again, this is encrypted utilizing industry standard 128 bit AES.

the authentication. CHR uses data from the payload to authenticate the user and to load the requested patient data in context.

An example of an unencrypted payload is shown below:

```
ssoMode=UA|sTime=1/10/2014 5:34:33 PM|uLogin=jbaker|uKey=a2db0dc4-9b0f-e111-bfcd-001a4ba8bfd0|fName=Joe|lName=Baker|pFName=Walton|pLName=Bender|pGender=|pDOB=|pSSN=|pM RN=|isEmbedded=
```

## Expected Parameters in Payload

To initiate the HTTP GET Single Sign-on of a user from an external application, CHR expects to receive the following parameters in the payload:

Table 2: SSO Mode, Authentication Key and User's Information

Parameter	Name	Description
ssoMode	SSO Mode	Valid values for this parameter are as follows: <ul style="list-style-type: none"> <li>IA (Impersonation Authentication)</li> <li>UA (User Base Authentication)</li> </ul>
sTime	Session Start Time	ProAccess will check DateTime when the URL is created. This is a UTC DateTime and it is placed inside the payload data. ProAccess will compare this DateTime to the current DateTime when it processes the request from an external system. The time span allowed is between 30-60 seconds depending on the configuration.
uLogin	User Log-in Credential	In an <b>Impersonation Authentication (IA)</b> scenario this value is ignored since we get the SSOAccount from the psk and log only that user, however, there must always be a ProAccess Database user with the same Login as this SSOAccount's ImpersonatedLogin or the request will fail.  In a <b>User Base Authentication (UA)</b> scenario, this will be the CHR user login credential (the user that is actually logged in). The fName and lName values will be overwritten with the login's name in the CHR Database.
uKey	User's Authentication Key	This will be the generated "AuthenticationKey" for all HTTP GET single-sign-on scenarios. This is created along with the other three keys at set-up time. They are stored in the SSOAccounts table in the pad database. Any Admin can create and delete an SSOAccount. They can provide the keys for any SSOAccount. Once CHR decrypts the payload it will verify that the request sent the same



		AuthenticationKey as is stored in the database, thus authentication the user.
<b>fName</b>	User's First Name	Authenticated user's first name. This will be logged so we know who is actually logged in.
<b>lName</b>	User's Last Name	Authenticated user's last name. This will also be logged.
<b>isEmbedded</b>	Indicate whether ProAccess is embedded (true or false).	Applies to v7.4.2 or higher. Must be 'true' or 'false'

## Patient Context Data in Payload

For SSO to open CHR with patient context, the following parameters will also be required. These parameters are required in addition to those provided in the previous topic. If these parameters are not included in the payload, then the CHR will open to the default screen for clinical users (assuming this functionality is enabled) and to the admin page for non-clinical users.

**Note:** If there is more than one match for the requested patient, CHR will return a list of patients. From here, a user would identify and select the desired patient.

Table 3: Patient Context Data in Payload

Parameter	Name	Description
<b>pfName</b>	Patient First Name	This is the patient's first name.
<b>pLName</b>	Patient Last Name	This is the patient's last name.
<b>pGender</b>	Patient Gender	This is the patient's gender. The following values are valid: <ul style="list-style-type: none"> <li>• M: Male</li> <li>• F: Female</li> <li>• U: Undifferentiated</li> </ul>
<b>pDOB</b>	Patient DOB	This is the patient's date of birth. A short date format should be used: MM/DD/YYYY.
<b>pSSN</b>	Patient SSN	This is the patient's social security number. Dashes are acceptable, e.g. 111-22-3333 or 111223333.
<b>pMRN</b>	Patient MRN	This is the patient's medical record number.

The payload will be encrypted using AES algorithm and the encryption key will be provided at the time of implementation. We included encoding and encryption options in the SSO Client API/.dll.

## Using HTTP POST

In CHR 7.4.x, external applications may use an HTTP Post (or HTTPS POST) request for SSO. Since with an HTTP Post request the Client Application uses HTTPS, the payload need not be encrypted. However, for security HTTP Post requests (or arguments) do require the Signature parameter, which is a SHA Hash. This is explained below.

**Security Note:** HTTP POST uses only a hash signature for security; so CHR can verify that the request was initiated by someone who has the keys and has the hash algorithm. There is no encryption here, so HTTP POST is as secure as your SSL.

### NEW in CHR 7.5.5

In CHR 7.5.5, we added Active Directory (AD) support to Legacy SSO. This feature requires ProAccess Database (PAD) matching, and the payload must contain the AD username currently in use. In addition, the feature does require a modification to an internal BORG setting.

**Note:** If you or your organization would like to utilize this new AD feature, please contact your Medicity Account Representative.

Arguments for the POST include the following:

Table 4: Patient Context Data in HTTP Post

Parameter	Name	Description
<b>Psk</b>	Private System Key	This will be the same as with HTTP Get (AppKey/IdpEntityID) except it will now be an HTTP Post argument. It must Base64 encoded for both GET and POST.
<b>Signature</b>	Signature	Is a SHA512 hash of a string with most of the payload parameters, salted with both the AuthKey and the EncyptionKey. (For convenience, we've added a GetSignature method to the Client API - and code. We also added the GenerateSignaturePreHashed.)
<b>SSOMode</b>	SSO Mode	Valid values for this parameter are as follows: <ul style="list-style-type: none"> <li>• IA (Impersonation Authentication)</li> <li>• UA (User Base Authentication)</li> </ul>
<b>SessionTimeout</b>	Session Time Out	This is a UTC DateTime telling CHR how long a Signature is valid until for the given HTTP Post. This time can be as far into the future as needed, but we recommend only 30-60 seconds ahead for security reasons. The Signature

		is only valid no more than 30-60 seconds after the Session Time Out, depending on configuration.
<b>Domain</b>	Active Directory Domain Name	This is the Active Directory (AD) Domain Name for the client.
<b>User</b>	In-Network Username	This is the in-network username for the user connecting to CHR.
<b>Password</b>	In-Network User Password	The in-network password for the CHR user.
<b>UserLogin</b>	CHR User Log-in Credential	<p>In an <b>Impersonation Authentication (IA)</b> scenario, in practice, this value is ignored since we get the SSOAccount from the psk and log only that user, however, there must always be a ProAccess Database user with the same Login as this SSOAccount's ImpersonatedLogin or the SSO attempt will fail.</p> <p>In a <b>User Base Authentication (UA)</b> scenario, this will be the ProAccess user login credential (the user that is actually logged in). The UserFirstName and UserLastName parameters will be ignored and replaced with the login's name stored in CHR Database.</p>
<b>UserFirstName</b>	User's First Name	The First name displayed for IA. For UA, CHR uses the First Name in the login in the ProAccess Database
<b>UserLastName</b>	User's Last Name	The Last name displayed for IA. For UA, CHR uses the Last Name of the login in the ProAccess Database
<b>PatientFirstName</b>	Patient First Name	The Patient's First Name to search. Only used if a PatientLastName is provided
<b>PatientLastName</b>	Patient Last Name	The Patient's Last Name to search.
<b>PatientGender</b>	Patient Gender	<p>The Patient's Gender to search. Valid values are as follows:</p> <ul style="list-style-type: none"> <li>• M: Male</li> <li>• F: Female</li> <li>• U: Undifferentiated</li> </ul>
<b>PatientDOB</b>	Patient Date of Birth	This is the patient's date of birth. A short date format should be used: MM/DD/YYYY.
<b>PatientSSN</b>	Patient SSN	This is the patient's social security number. Dashes are acceptable, e.g. 111-22-3333 or 111223333.
<b>PatientMRN</b>	Patient MRN	This is the patient's medical record number.

**Note:** Be aware that Payload and IsEmbedded are not supported Post Arguments. An example of a pre-hashed signature is as follows:

```
SSOMode={0}|SessionTimeout={1}|Domain={2}|User={3}|Password={4}|UserLogin={5}|UserFirstName={5}|UserLastName={6}|PatientFirstName={7}|PatientLastN  
ame={8}|PatientGender={9}|PatientDOB={10}|PatientSSN={11}|PatientMRN={12}
```

## Setting up SSO Support in the CHR

If **Impersonation Authentication (IA)** is your preferred SSO method, your organization must define your **Single-Sign-On-User (SSOU)** IA profiles in CHR before SSO can be utilized. You can create a single SSOU IA profile. You can create multiple SSOU IA profiles each with different rights. Or you can create specific, unique SSOU IA profiles for individual organization or practice. As a general rule, we recommend that you keep it your SSOU AI profiles as simple as possible.

Implementation will assist you in setting up your initial SSOU IA profile or profiles. After the initial setup, Admins can create or modify Single-Sign-On accounts from the SSO Maintenance page. It is here, on the SSO Maintenance page that you will obtain the key value sets (private system key/idpEntityID; the encryption key/user's authentication key). You will use these keys to construct and encrypt the **payload** or the **Signature** (depending on HTTP GET or POST).

If **User Base Authentication (UA)** is used, you do not need to create a **Single-Sign-On User (ssouser)** in CHR. Instead, the user's login info will be passed to CHR during the SSO login process. In this situation, SSO will require that the user profile exists in CHR. Otherwise the SSO attempt will fail.



# Frequently Asked Questions

Below, you will find a list of several of the most common questions that we receive regarding Single Sign-on. You will also find the answers for each question.

**Question:** This guide talks about creating Impersonation Authentication (IA) accounts. Is that all the SSO tool handles?

**Answer:** *No. You create an IA and User Based Authentication (UA) account the exact same way. IA is used if you desire to have users authenticate into the same account, specified as the ImpersonatedLogin. Even if an ImpersonatedLogin is specified, the SSOAccount can still use UA, you just have to specify the user to be used in the payload*

**Question:** How do we create UA accounts?

**Answer:** *See above.*

**Question:** Both IA and UA must match an existing CHR user ID to work, correct?

**Answer:** *This question requires a nuanced answer. CHR validates IA credentials against existing Collaboration Suite (CS) usernames during the SSO Impersonation Account creation. UA does not validate credentials during SSO account creation. However, since UA requires an actual Collaboration Suite username during login, UA validates the username used during a log-in event.*

**Question:** Any other action items that must be completed when using UA?

**Answer:** *If you are asking if the System Administrator needs to do anything else in the Collaboration Suite for setup, the answer is no.*

**Question:** What happens after an IA/UA account is created?

**Answer:** *The Collaboration Suite provides the client with the Authentication Key and the Encryption Key. These two keys are the primary items the client needs to complete the SSO setup.*

**Question:** What are the next steps for an end user?

**Answer:** *The client needs to configure their application for SSO.*

**Question:** I realize it would be different for each external application, but do we have an example?

**Answer:** *It is completely different for each application depending on the system used. However, an organization's technical development team should know what to do next. Also, these external application vendors should provide specs that walk a user through SSO configuration. At the end of the day, their system is simply creating the Payload for SSO integration.*

**Question:** There must be something else than I have to do?

**Answer:** *Possibly. Your technical support team should have all of the information that they need from here, however.*

**Question:** How does transaction logging work with SSO?

**Answer:** *The transaction log can be used two different ways. If you enable the transaction log, then the Collaboration Suite (CS) will track all impersonation logins. If you disable transaction logging, then CS will track only failed impersonation login attempts.*

**Question:** Are there auditing tools or a Management Report for SSO?

**Answer:** *The application contains the SSO Transaction Log. You should not disable the Transaction Log if you want to track all SSO activity. If you do disable Transaction Logging, CHR will track only failed events. Please note that the SSO Transaction Log is a troubleshooting tool. For auditing, you will utilize the Login Reports found in the Management Reports utility. The Login Reports will identify when an SSO user has accessed CHR just as it would for a traditional user.*

**Question:** Can I export the Transaction Log as a report?

**Answer:** *At this time, the SSO transaction log may be viewed only in CHR.*

**Question:** What does SSO actually integrate? Meaning, from an end user perspective, what can I see/do? What functions are enabled? Are there any restrictions?

**Answer:** *If IA is used, then the login access will mirror the IA user credentials. If UA is utilized, then SSO uses the users CS credentials. Once CHR has opened, all functionality is identical to the user name or user role's access as through a normal login had been performed.*

**Question:** Does the SSO user have full unrestricted access to all records, regardless of consent status?

**Answer:** *No. The SSO user will always be constrained based upon their rights. If Impersonation Authentication (IA) is the organizations preferred SSO method, then you define rights for each IA profile created. And if User Based Authentication (UA) is used, CHR simply uses the defined rights for the individual user.*

**Question:** My organization resides in an Opt-Out state. How does SSO work for AI users in an Opt-Out environment?

**Answer:** *The best way to think about Impersonation Authentication (IA) is to look at the profile as if it is single user that multiple users utilize to access the application. As such, rights would work exactly as you expect for a traditional user. So if an individual accesses CHR using IA SSO, and if the user resides in a default Opt-Out state, then CHR handles rights the same as it would for a regular user (n this scenario). Or in other words, the same rules that govern viewing rights, AAR, and express consent would apply.*

**Question:** I was told that I can include a link or tab within my HIS or EMR that would allow me to bypass the CHR log in screen? Is this true?

**Answer:** Yes.

**Question:** Will every features/functions work in our UI after this initial launch?

**Answer:** *Again, a nuanced answer is needed. All features/functions will work according to the IA or UA user credentials.*

**Question:** Is there any reason the Entry ID field isn't just called "name"?

**Answer:** *No. EntityID was clearer than the previous Psk. We could have changed it to 'Name', but that is not as clear as EntityID, since it is the primary key column in the database table.*

**Question:** Is embedding supported in CHR v7.4.2 (formerly known as ProAccess)?

**Answer:** Yes.

**Question:** If Community Health Record (CHR) is embedded in our application, can we use the payload package to bring the current patient in to focus.

**Answer:** *For v7.4.2 or higher, yes. Your internal development team must read through the SSO technical documentation to implement this feature. isEmbedded must be 'true'.*

**Question:** Can we use Active Directory?

**Answer:** *Yes. However, this feature requires ProAccess Database (PAD) matching, and the payload must contain the AD username currently in use. In addition, the feature requires a modification to an internal BORG setting.*





# Appendix A: SSO Test Client and Payload Simulator

The following SSO test client tool simulates SSO integration. This tool is quite useful for integration teams to simulate a real SSO payload. Please work with you Medicity Integration Team and representative for details and access to the tool in pre-production environments.

Figure 2: SSO Test Client and Payload Simulator

SSO Mode:	<div>Impersonation Authentication (IA) ▼</div>		
Embedded :	<input checked="" type="checkbox"/>		
Appkey :	<div>City Center Hospital Networks</div>		
Authentication Key :	<div>58B31C5E-5485-483D-88F4-ED7F85E2D5B3</div>		
EncryptionKey :	<div>C11065D0-AD20-42A8-827F-87B9ABCD58C</div>		
URL :	<div>http://zgen-qa-web/proaccess/acs</div>		

User Context	Patient Context		
User Name :	<div>ssouser</div>	Patient Last Name :	<div>Doe</div>
User Family Name :	<div>Doe</div>	Patient First Name :	<div>John</div>
User GivenName :	<div>John</div>	Patient MiddleName :	<div></div>
		Patient Gender :	<div>Male</div>
		Patient Date Of Birth :	<div>01/10/1999</div>
		Patient SSN :	<div>123456789</div>
		Patient MRN :	<div>A812D8392</div>
		Patient Account :	<div>793457812001</div>

Show Payload

Perform SSO

Lower Case Encryption Key: c11065d0-ad20-42a8-827f-87b9abcbd58c

Encryption Log

SHA512 Hash:

```
4abTxYFrhEi7+dWUCJAZQfNSBIKHs3dtItX4sCxM4ruDdWuPZK8rcme8R2Men7Qn2dRipESAffgbhpgp2YiTEw==
key= xYFrhEi7+dWUCJAZQfNSBIKH (key = hash.Substring(4, 24))
IV= 4abTs3dtItX4sCxM (IV = hash.Substring(0, 4) + hash.Substring(28, 12))
```

Payload: ssoMode=IA|sTime=12/7/2016 4:26:47 PM|uLogin=ssouser|uKey=58b31c5e-5485-483d-88f4-  
 ed7f85e2d5b3|fName=John|lName=Doe|pFName=John|pLName=Doe|pGender=Male|pDOB=01/10/1999|pSSN=123456789|pMRN=A812D8392|isEmbedded=True  
 URL: http://zgen-qa-web/proaccess/acs?psk=Y2l0eSBjZW50ZXIgaG9zcGl0YWwgYmV0d29ya3M%  
 3d&payload=Xj4j501IEtVYg5%2fu5eGofV4BuGn2qoEPb53yUM4%2bS4Sib1woP%  
 2bGMcFOFbVHfVlnrFNHj1go6utvnPP%2bpGaIFAV1rkrkvmHz0pn5lEnudTW0nuF%2bSibtJF2mbmzevi3Hajz5cOMI6%  
 2b6guO8PwbV1iVEjEDm0aG0up8hu2NvZMkzpqjN95xm6uFUxWhvxVfyTOleSKY3obevZjWyT%  
 2fnoSApwEHQm5wNteZkQrASJfIX8E44xQXKzLOktWNhIDPUILCv5JY050QeTAdczL08Qo1PU%2ffaQ4JastD%2bT3k%

