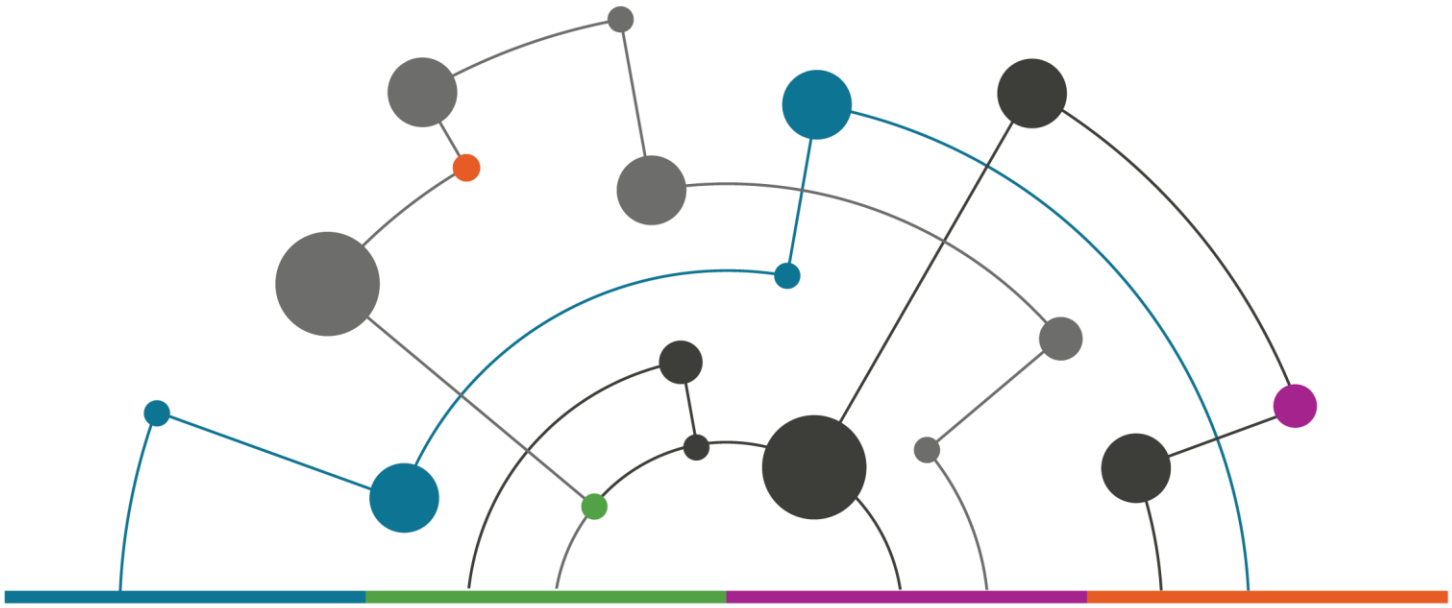
Shared Services Canada | Services partagés Canada

Canada

Technology Guide for Entrust GetAccess

Government of Canada Credential Federation

Revision 1.2 2018-07-19



Powering Technology for the Government of Canada

Revision History

Date	Version	Description
Jan 19, 2012	0.1	Peer review draft
Jan 20, 2012	0.2	First release posted on the forum
Feb 21, 2012	0.3	Improvements related to language passing.
Apr 2, 2012	0.4	Updated to latest Server patch; Removed the ":443" metadata recommendation; Referenced CA certs now available on the forum.
Apr 25, 2012	0.5	Corrected location of some config settings and changed GAEncryptID to false to accommodate browser issues.
June 8, 2102	0.6	Fixed typos in example URLs
Nov 23, 2012	0.7	Fixed typo in logout URL
Jun 11, 2013	1.0	Updated for SHA-256, referenced the forum for CRL URLs. Reference and Branding updates.
April 6, 2018	1.1	Updated for GetAccess 9.0. Removed dependencies on IES and ASI.
July 19, 2018	1.2	New branding template. Updated secure cookie config.

Contents

1. Introduction 4
1.1 Purpose 4
1.2 Scope 4
1.3 Audience 4
1.4 References 4
2. Prerequisites 4
2.1 Documentation Requirements 4
2.2 Software Requirements 5
3. Installation 5
3.1 Installing the GetAccess Repository and Server 5
3.2 Installing the GetAccess Runtime 6
4. Configuration 6
4.1 Initial Configuration 6
4.2 Configuring GetAccess for a new CSP 7
4.3 Producing SP Metadata 10
4.4 Connectivity Testing 11
5. Application Integration 12
5.1 Configuring the Auto-Federation Extension 12
5.2 Implementing the Login flow 13
5.2.1 Initiating Authentication 13
5.2.2 Obtaining Assertion Data 18
5.2.3 Handling Authentication Failures 19
5.3 Logout 20
5.3.1 Initiating Logout 20
5.3.2 Post-logout 20
6. Preparing for Production 20
6.1 Security Hardening 20
6.2 Performance and High Availability 21
A. Glossary of Acronyms 22

1. Introduction

1.1 Purpose

The purpose of this document is to help Government of Canada departments and agencies to successfully integrate their program's on-line service applications with the shared credential and authentication services available within the Government of Canada Credential Federation (GCCF) using Entrust GetAccess.

1.2 Scope

This document provides information to assist departments and agencies with installing and configuring the Entrust GetAccess SAML product to integrate with the GCCF.

1.3 Audience

This document is primarily targeted toward application architects, web designers and developers who will be responsible for integrating an on-line service application into the Federation. It may also be of value to other technical stakeholders such a security, network and infrastructure architects.

1.4 References

[CATS]	Cyber Authentication Technology Solutions Interface Architecture and Specification Version 2.0: Deployment Profile
[GCCF-AIG]	Government of Canada Credential Federation - Application Integration Guide V1.3

2. Prerequisites

2.1 Documentation Requirements

This document makes frequent references to the following documents which are available for download from the Entrust Trusted Care web site:

ENTRUST GETACCESS™ SERVER 9.0 PLANNING AND INSTALLATION GUIDE

Document issue: 4.0, Date of Issue: June 2017

ENTRUST GETACCESS™ SERVER 9.0 SYSTEM ADMINISTRATION GUIDE

Document issue: 8.0, Date of Issue: May 2018

ENTRUST GETACCESS™ RUNTIME 9.0 ADMINISTRATION GUIDE

Document issue: 3.0, Date of Issue: October 2017

ENTRUST GETACCESS SERVER 9.0 BUSINESS ADMINISTRATION GUIDE

Document issue: 2.0, Date of Issue: June 2017

ENTRUST GETACCESS SERVER 9.0 PROGRAMMING GUIDE

Document issue: 2.0, Date of Issue: June 2017

2.2 Software Requirements

In order to install and configure GetAccess as described in this document, the following Entrust Software components are required:

ENTRUST GETACCESS SERVER 9.0

ENTRUST GETACCESS RUNTIME 9.0 (APPROPRIATE VERSION FOR YOUR WEB SERVER)

In addition, Entrust had published the following mandatory and recommended patches at the time this document was created:

ENTRUST GETACCESS SERVER 9.0 PATCH 207443 (OR HIGHER)

ENTRUST GETACCESS RUNTIME 9.0 PATCH (LATEST FOR YOUR PLATFORM)

These files can all be downloaded from the Entrust Trusted Care web site.

Finally, Shared Services Canada has developed a software toolkit that includes a number of small utilities and GetAccess extensions to facilitate implementing certain GC-specific capabilities. This toolkit can be downloaded from the CATS GCconnex Group at:

https://gcconnex.gc.ca/file/view/32673702

3. Installation

3.1 Installing the GetAccess Repository and Server

Note: In order to install the GetAccess server, a valid licence key is required from Entrust.

1. Follow the instructions for installing the GetAccess repository that can be found in Chapter 3 of the GetAccess Server Planning and Installation Guide.
2. Follow the instructions for installing the GetAccess server that can be found in Chapter 4 of the GetAccess Server Planning and Installation Guide.
3. Follow the instructions in Chapter 7 of the GetAccess Planning and Installation Guide to install the most recent GetAccess server patch (207409 or later).

3.2 Installing the GetAccess Runtime

1. Follow the instructions in the GETACCESS™ RUNTIME 9.0 ADMINISTRATION GUIDE to install the GetAccess Runtime on your web server(s). Note that at least one web server must be configured as an Access Portal; this will be the web server that exchanges SAML messages with the Credential Service Providers (CSPs) and therefore it must be accessible by your users from the Internet.

   Tip: The Runtime installer will examine your web server configuration to determine the host and port names of your virtual servers and automatically configure GetAccess to support them. Ensuring that your web server is fully configured before installing the Runtime will avoid the need for manual configuration of GetAccess later.

2. Follow the instructions in the applicable patch installation notes to install the most recent Patch for the GetAccess Runtime.

   Tip: If the firewall between your web server and your GetAccess infrastructure server is configured to automatically disconnect idle TCP connections after a period of time, the release notes for the GetAccess Runtime patch contains instructions for ensuring that this does not create problems in your deployment.

3. After installing the first Runtime, perform the "Post Installation Task" described at the end of CHAPTER 4 of the GETACCESS SERVER PLANNING AND INSTALLATION GUIDE.

4. Configuration

Before you begin the configuration, review Chapters 2 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE to familiarize yourself with GetAccess configuration concepts.

4.1 Initial Configuration

In CHAPTER 10 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE, under "Common configuration tasks for both the IDP and the SP" (page 396) follow the instructions described in the following two steps:

1. Enabling SAML 2.0
2. Configuring the Runtime

Caution: The Runtime has its own configuration file and console that is separate from the server configuration file and console. When performing step 2, be sure to launch the configuration console from the bin directory of your GetAccess Runtime installation (e.g. GetAccess/Runtime/IIS6.0/bin), not from the bin directory of the GetAccess Server (i.e. GetAccess/bin).

Your Client Integration Team will assist you in obtaining two PKI certificates from the government of Canada Internal Credential Management Service. These two certificates will be used by GetAccess to digitally sign your SAML requests, and to decrypt assertions received from Credential Service Providers.

In order to import these two keys into GetAccess, they must each be in their own PKCS#12 or PEM format file.

3. Follow the instructions in CHAPTER 10 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE, under "Importing certificates using GaKeyTool" (page 484) to import each certificate into GetAccess. Be sure to name each certificate with an alias that reflects its usage (e.g. "sp:signing", "sp:decryption"). You will need to re-enter these alias names when configuring GetAccess to use each CSP.
4. Using the GetAccess Configuration Console, add the complex setting <global> <gaSaml2> and adjust the following child settings:
5. Ensure the simple setting <gaSamlAttributeAuthoritySettings> <gaEnabled> is set to **false**.
6. Ensure the simple setting <gaSamlAttributeRequesterSettings> <gaEnabled> is set to **false**.
7. Ensure the simple setting <gaSamlIDPSSOSettings> <gaEnabled> is set to **false**.
8. Ensure the simple setting <gaSamlSPSSOSettings> <gaEnabled> is set to **true**.
9. Change the value of the simple setting <gaSamlStandardRelayStateCache><gaCacheIdleTime> from 1802 to 1200.
10. Change the value of the simple setting <gaSamlStandardRelayStateCache><gaCacheTTL> from 28800 to 1200.
11. Open or create the complex setting <gaSamlSPResourceSecondaryDomainList>.
12. Under this new complex setting, edit the <gaDomain> simple setting and change it to the name of the common cookie domain e.g.: "**.fjgc-gccf.gc.ca**"

4.2 Configuring GetAccess for a new CSP

Your Client Integration team will provide you with a SAML metadata XML file containing the configuration setting needed to configure GetAccess for each CSP's SAML interface.

1. In CHAPTER 10 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE, under "Common configuration tasks for both the IDP and the SP", follow the instructions described under "Exporting and importing SAML 2.0 metadata" / "To import SAML 2.0 metadata" (page 399-400) to import the CSP metadata. When prompted for the local entity ID in *step 5*, enter the

unique SAML entity ID that has been agreed upon between your department or agency and Shared Services Canada.

Importing a CSP's metadata will create a new <gaSamlEntityDescriptor> complex setting under <global><gaSaml2> in your configuration. Your configuration will contain one <gaSamlEntityDescriptor> for each CSP to which you are connected. You can determine which <gaSamlEntityDescriptor> pertains to which CSP by looking at the value of the <gaPartnerEntityID> simple setting that will contain the Entity ID of the CSP. If you wish, you can create the <gaDisplayName> simple setting under each <gaSamlEntityDescriptor> and set it to a more recognizable name (e.g. "GCKey" or "SecureKey Concierge").

> *Caution: The GetAccess installer may have created an initial default <gaSamlEntityDescriptor> entry. If you see an extra <gaSamlEntityDescriptor> that does not correspond to the metadata of a CSP that you have loaded, then you should delete it.*

In order to securely connect to the CSP's SOAP services, you will also need to load the public root certificate (plus any chain certificates) of the certificate authority that issued the CSP's SSL certificate into the GetAccess keystore. These have been posted to CATS GCconnex Group (https://gcconnex.gc.ca/gcforums/topic/view/37122067) for your convenience.

2. Follow the instructions in CHAPTER 12 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE, under "Importing certificates using GaKeyTool" (page 484) to import the CA root certificate into GetAccess. Use a meaningful alias name that includes the name of the CA, for example: "EntrustCA".

Using the GetAccess Configuration Console, adjust the following settings under <global> <gaSaml2>:

Under the CSP's <gaSamlEntityDescriptor> <gaSamlIDPSSODescriptor>:

3. Create the simple setting <gaAssertionConsumerServiceBinding> and set it to urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST.
4. Create the simple setting <gaAuthnRequestServiceBinding> and set it to urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect.
5. Set the simple setting <gaSignSSORequest> to true.
6. Create the simple setting <gaVerifyAssertion> and set it to ALWAYS.

Under the CSP's <gaSamlEntityDescriptor>:

7. Create the complex setting <gaSAMLSecuritySettings>

Under the <gaSAMLSecuritySettings> you just created:

8. Change the simple setting <gaSoapServerAuthenticationEnabled> to true.
9. Create the simple setting <gaSoapServerCACertAlias> and set it to the alias name you gave to the CA root certificate you imported in step #2 above.

Next you need to configure GetAccess to support each level of assurance supported by the particular CSP you are adding. To determine which levels of assurance the CSP Supports you can examine the CSP's metadata in a browser or text editor. Start by searching the metadata for the string "urn:oasis:names:tc:SAML:attribute:assurance-certification". The levels of assurance will be listed in a series of <AttributeValue> elements immediately following this element.

Under the CSP's <gaSamlEntityDescriptor> <gaSamlIDPSSODescriptor>:

10. For each level of assurance listed in the CSP's metadata, create a complex setting of <gaSamlAuthnContextMapping>. Then, under this complex setting:
    a. Create the simple setting <gaAuthLevel> and set it to an integer value reflecting the level of Assurance (e.g. 2).
    b. Create the simple setting <gaAuthMethod> and set it to a short name reflecting the level of assurance (e.g. "CATSLoA2").
    c. Create the simple setting <gaAuthnContext > and set it to the URI value that appears in the metadata (e.g. "urn:gc-ca:cyber-auth:assurance:loa2").

Finally, to specify one level of assurance to be the default, create a <gaAuthMethod> simple setting back under the CSP's <gaSamlEntityDescriptor> <gaSamlIDPSSODescriptor> and set it to the short name that reflects the level of assurance that you expect to use most often (e.g. "CATSLoA2").

Under the CSP's <gaSamlEntityDescriptor> <gaSamlIDPSSODescriptor> <gaSamlSSODescriptor>:

11. Create the simple setting <gaAllowIDFederation> and set it to true.
12. Create the simple setting <gaEncryptID> and set it to false.
13. Create the simple setting <gaManageNameIDServiceEnabled> and set it to false.
14. Create the simple setting <gaSingleLogoutServiceBinding> and set it to urn:oasis:names:tc:SAML:2.0:bindings:HTTP-Redirect.
15. Create the simple setting <gaSignSLORequest> and set it to true.
16. Create the simple setting <gaSignSLOResponse> and set it to true.
17. Create the simple setting <gaVerifySLORequest> and set it to ALWAYS.
18. Create the simple setting <gaVerifySLOResponse> and set it to ALWAYS.

Under the CSP's <gaSamlEntityDescriptor> <gaSamlIDPSSODescriptor> <gaSamlSSODescriptor> <gaSamlNameIDFormat> <gaSamlNameIDFormatMapping>:

19. Verify that the simple setting <gaNameIDFormat> is set it to urn:oasis:names:tc:SAML:2.0:nameid-format:persistent.

Under the CSP's <gaSamlEntityDescriptor> <gaSamlIDPSSODescriptor> <gaSamlSSODescriptor> <gaSamlRoleDescriptor> <gaSamlKeyDescriptor>:

20. Create the simple setting <gaDecryptionPKCertAlias> and set it to the alias that you chose for your decryption certificate (e.g. "sp:decryption").
21. Create the simple setting <gaSigningPKCertAlias> and set it to the alias that you chose for your signing certificate (e.g. "sp:signing").
22. Verify that the simple setting <gaEncryptionMethod> is set it to http://www.w3.org/2001/04/xmlenc#aes128-cbc.
23. Create the simple setting <gaSignatureAlgorithm> and set it to http://www.w3.org/2001/04/xmldsig-more#rsa-sha256.
24. To enable CRL checking, create the simple setting <gaCrlUrl> and set it to the value appropriate for the CSP. The appropriate CRL URL for each CSP is available on the CATS GCconnex Group: https://gcconnex.gc.ca/gcforums/topic/view/37122067

    Note: Your departmental firewall(s) must be configured to allow port 389 LDAP connections to the ICM directory server from the server hosting your GetAccess infrastructure.

25. By default, GetAccess will cache the CRL entries that it retrieves from the LDAP directory for up to 5 minutes. In order to reduce unnecessary load on the directory servers, Shared Services Canada requests that relying parties configure their CRL caches with a timeout of at least one hour. This can be configured in GetAccess by creating the simple setting <gaCrlTimeout> and setting its value to 60.

When done, save your configuration settings, exit the Configuration Console and re-start your GetAccess infrastructure.

4.3 Producing SP Metadata

In order for the CSPs to recognize your service provider, you must generate a SAML metadata file and provide it to your GCCF Client Implementation Team.

PWGSC has developed a custom GetAccess metadata generation template that excludes certain elements that are not part of the CATS specification. This template is provided as a file named "saml.metadata.template.xml" contained in the GetAccess toolkit ZIP file. To install this custom template, back up the saml.metadata.template.xml file located in the *<GA_Root>/config* directory of your GetAccess server installation (e.g. by renaming it to "saml.metadata.template.xml.bak") and then replace it with the version from the ZIP file.

In CHAPTER 10 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE, under "Common configuration tasks for both the IDP and the SP", follow the instructions described under "Exporting and importing SAML 2.0 metadata" / "To export SAML 2.0 metadata" (page 395-396) to export your service provider metadata.

26. When prompted "Does information exist for this partner in the GetAccess configuration file?" enter "y".
27. When prompted "Enter partner entity ID:", enter the CSP service's unique entity ID. (This can be copied and pasted from the CSP XML metadata file. Look for the entityID attribute of the <EntityDescriptor> element).
28. When prompted "Enter service URL prefix", enter the HTTPS address of the web server on which you have installed the GetAccess runtime configured as an Access Portal. For example: https://*myserver.mydepartment*.gc.ca

The CATS interface specification requires that all metadata files be digitally signed. Entrust GetAccess does not support the signing of metadata, however Shared Services Canada has developed a small standalone program that will apply a CATS-compliant XML signature. This program requires that you have a Java 6 runtime environment on your desktop and can be obtained from CATS GCconnex Group at https://gcconnex.gc.ca/file/view/32306451.

4.4 Connectivity Testing

You can prepare your GetAccess installation for connectivity testing at the same time that your SP metadata is being validated and configured into the CSP service:

By Default, GetAccess ships with the SAML logout and login links on its web pages disabled. For instructions on how to enable these links, refer to CHAPTER 10 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE:

1. Under "Enabling Web Browser SSO on the IDP and SP" follow the instructions described under "To enable the SAML Login link" on pages 448 and 449.
2. Under "Enabling SLO on the IDP and SP" follow the instructions described under "To enable the SAML Logout link" on pages 457 and 458.

In order to test interoperability, you will need to create a new GetAccess user account that you can then link to a CSP credential. To do this:

3. Log in to GetAccess as "websecadm" (the default password is "changeme").
4. Click on the "GetAccess Admin" link. The GetAccess administration tool should appear. If it does not, you may need to disable the pop-up blocker in your web browser.
5. Click on "Users" -> "Create", fill out the Personal Information and User Login and Password fields and click "Save".

6. Close the administration tool window and click the Log Out" button on the main GetAccess page.

Your Client Implementation Team will notify you once your SP metadata has been accepted by and configured into the CSP service. At this point you are now ready to test the integration:

7. Click on the "SAML Login" link at the bottom of the GetAccess login screen.
8. If you are prompted with a list of configured IDPs, select the one that you wish to test.
9. The CSP Login screen should appear. Proceed through the CSP login process.
10. At this point, GetAccess should display "ALERT: You are in the process of performing an ID Federation from an IDP to this SP.", Click on "Ok" to continue.
11. The GetAccess login screen will appear with the message "You must log in locally at the Service Provider to complete SAML ID Federation". Enter the username and password you created in step 6 above and click OK.
12. GetAccess should display a Welcome page. Click on "Ok" to continue.

In order to test single sign on:

13. Click on the "Log Out" button. This will log you out of GetAccess but not out of CSP.
14. Click the "SAML Login" link; you should be immediately logged in again.

In order to test single logout:

15. Click on the "SAML Log Out" button. This will log you out of GetAccess and out of CSP.
16. Click the "SAML Login" link. This time CSP will prompt you to log in again.
17. Log in to CSP as before. This time you will be logged into GetAccess immediately since a GetAccess user has already been linked to your CSP credential.

5. Application Integration

5.1 Configuring the Auto-Federation Extension

GetAccess requires every user to have an account in the GetAccess repository. As illustrated during the test above, when a new user authenticates with your GetAccess installation via the CSP service for the first time, GetAccess will, by default, prompt them for a GetAccess username and password that must have been previously set up.

The Auto-Federation extension developed by SSC automates the creation of an anonymous GetAccess user account for new users the first time they authenticate, eliminating the need to pre-assign a GetAccess username and password and avoiding the GetAccess login prompt.

The Auto-Federation extension is part of the GetAccess toolkit available from the technical forum. The toolkit ZIP file contains both the deployable *gaExtension.jar* file as well as the source code.

If you wish to modify or enhance the behaviour of the extension, you can refer to CHAPTER 2 of the PROGRAMMING GUIDE. You may also want to download the ADMINISTRATION SERVICE INTERFACE SDK from the Entrust web site. The SDK contains sample source code together with build scripts that can be used to re-compile the extensions.

To install the extension, copy the *gaExtension.jar* file to the *<GA_Root>/lib* directory of your GetAccess server installation.

To enable the extension, use the GetAccess Configuration Console to:

1. Create the complex setting <services> <INFRASTRUCTURE> <global> <gaExtensions>.
2. Under ...<gaExtensions> <onSamlBeforeUserLookup>, create the simple setting <gaExtensionName> and set it to ca.gc.pwgsc.cats.gatoolkit.AutoFederateExtension.
3. Under ...<gaExtensions> <onSamlBeforeUserLookup>, set the simple setting <gaExtensionEnabled> to true.
4. Re-start your GetAccess infrastructure.

To test auto-federation:

5. Go to the CSP web site and register for a second CSP Credential.
6. Open the GetAccess login screen and click on the “SAML Login” link.
7. Select an IDP if prompted.
8. The CSP Login screen should appear. Proceed through the CSP login process.
9. A new GetAccess user will be automatically created and linked to the new CSP credential. This new user will be logged in to GetAccess immediately without any further prompts.

5.2 Implementing the Login flow

5.2.1 Initiating Authentication

Once you have configured and tested GetAccess connectivity to CSP, the next step is to configure it to protect your application. Refer to the ENTRUST GETACCESS SERVER 9.0 BUSINESS ADMINISTRATION GUIDE for instruction on how to configure GetAccess to protect your application’s resources.

There are 8 steps to the process of logging a user in through GetAccess and a CSP in order to provide access to protected application resources:

1. The user selects their session language.
2. The user selects their preferred credential service from the options presented on your CSP chooser page.

3. The common domain language cookie must be written with the user's session language.
4. GetAccess sends the user with an authentication request to the selected CSP.
5. The user logs in to the CSP and the CSP returns an authentication response to GetAccess.
6. GetAccess reads the user's PAI (and Issuer) from the Assertion in the authentication response.
7. The common domain language cookie must be read to determine if the user changed their session language while authenticating.
8. The user is redirected to the application.

There are at least two possible development approaches for implementing this process:

a) You can develop a set of customized GetAccess login page templates to implement the language selection "splash" page and CSP selection "chooser" page, or
b) You can implement the language selection and CSP chooser pages in your application and explicitly invoke GetAccess from your chooser page to perform the SAML steps.

Both approaches will work equally well. The choice of which approach to take is mostly a matter of technology preference, and whether you wish to maintain the login flow pages as part of your application or as part of your GetAccess configuration. Information about how to create custom GetAccess login templates can be found in Chapter 20 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE.

The rest of this section will provide information on how to implement the 4 main functions that are required: Intercepting unauthenticated users, setting the language cookie, invoking GetAccess to perform an authentication request, and then reading the language cookie when the user returns from the CSP.

5.2.1.1 Intercepting Unauthenticated Users

Normally a user will arrive at your web site, select their preferred language on a splash page, view your (unprotected) home page and navigate their way to your application through the links and pages provided. In the event that a user attempts to access a protected resource directly (e.g. through a bookmark) they need to be intercepted and redirected into the login flow instead. This interception is performed by the GetAccess Runtime module on your web server. For each Runtime on each web server there is a "Login Challenge URL" configuration setting accessible from the "Servers" section in the GetAccess administration application that specifies where an unauthenticated user should be sent if they attempt to directly access a protected resource.

By default, the Login Challenge URL points to "https://servername/auth/Login" which is the invocation URL for the standard GetAccess login flow. If you wish to direct the unauthenticated user to your application login flow instead you can change this parameter to point to any unprotected application URL. For example:

a) You could enter the URL of your language selection “splash” page in order to force the user to select their session language and then proceed to the application normally via your home page, or
b) You could enter the URL of your language cookie service that could implement logic to check to see if the user has already selected a language and then redirect them to either the splash page or the language specific home page (or perhaps the chooser page) depending on the presence or value of the cookie.

When the GetAccess runtime forwards the user to the Login Challenge URL that you specify, it will append a “GAURI” query parameter that contains the protected URL that the user was attempting to access. This may be useful if you want to implement some dynamic content in your login pages or if you want to eventually send the user directly to the protected page they were trying to access.

5.2.1.2 Setting the language cookie

There are example implementations of web applications for reading and writing the common domain language cookie available on the GCCF technical forum. One of these, a Java Servlet implementation developed by Shared Services Canada provides extra features including support for the GetAccess custom login template mechanism. This implementation should work on any Java Application server that supports version 2.4 or later of the Java servlet specification, including the Apache Tomcat Application server that is used by the GetAccess infrastructure. (Please refer to Entrust’s Tech-Note TN 5172 if you are considering running your language service on the GetAccess infrastructure).

If a user has multiple browser windows or tabs open, all of these will share the same language cookie in the browser’s memory. Because of this, reading and writing the cookie too often may create a poor user experience. For example, if the user changes languages in one browser tab, they wouldn’t normally expect that to have an impact the language displayed in other tabs. To avoid these types of problems is best to write the language cookie once at the last possible moment before transferring the user to the CSP.

The easiest way to do this is to have the Login buttons on your CSP chooser page (which is language specific) call your cookie service to write the language cookie before invoking GetAccess to initiate the SAML request. The following sections provide examples on how to do this.

5.2.1.3 Invoking GetAccess SAML Authentication

Once a user arrives at your CSP chooser page, you need to be able invoke GetAccess to send a SAML authentication request to whichever CSP they choose. This is quite easy as GetAccess provides a means for an application to explicitly invoke the authentication request process by having the browser send a request to a particular URL. This URL accepts certain query

parameters that can be used to specify the SAML Entity ID of the CSP to be used as well as other options such as the requested authentication context (i.e. desired level of assurance), forceAuthn and isPassive attributes, etc. An additional query parameter is also supported that provides the destination URL indicating where to redirect the Individual's browser after the authentication response has been processed.

The URL is: https://*servername*/GetAccess/Saml/SSO/Init

…where *servername* is the name of your web server with the GetAccess runtime installed and configured as an Access Portal.

Supported Query Parameters:

Name	Value / Use
GA_SAML_IDP	SAML EntityID of the CSP to be sent the authentication request
GAURI	Target URL for browser redirect after authentication response processing
GA_SAML_FORCE_AUTHN	"true" or "false" / To set forceAuthn in the AuthnRequest
GA_SAML_IS_PASSIVE	"true" or "false" / To set isPassive in the AuthnRequest
GA_SAML_AC_COMPARISON	"exact" / Authentication context comparison.
GA_SAML_AC_CLASS_REF	Authentication Context (Level of Assurance)

Example:

Here is some HTML to implement a chooser page login button that will send the user to the CSP with entity ID "https://idp.entityid.com" with a level 2 authentication request specifying forced authentication (i.e. not allowing single sign-on), and then forward the user to the application page at "https://servername.mydemt.gc.ca/Enrol/Confirm.aspx" when they return:

```
<form method="GET"
action="https://servername.mydept.gc.ca/GetAccess/Saml/SSO/Init?GA_SAML_IDP=h
ttps://idp.entityid.com&GAURI=https://servername.mydept.gc.ca/Enrol/Confirm.a
spx&GA_SAML_FORCE_AUTHN=true&GA_SAML_AC_COMPARISON=exact&GA_SAML_AC_CLASS_REF
=urn:gc-ca:cyber-auth:assurance:loa2">
      <input type="submit" value="Login"/>
</form>
```

If you want to call your cookie service to set the language cookie first then the form action would look something like:

```
action="https://app-dept.fjgc-
gccf.gc.ca/GACS/?l=en&t=https://servername.mydept.gc.ca/GetAcces
s/Saml/SSO/Init?GA_SAML_IDP=https://idp.entityid.com%26GAURI=htt
ps://servername.mydept.gc.ca/Enrol/Confirm.aspx%26GA_SAML_FORCE_
AUTHN=true%26GA_SAML_AC_COMPARISON=exact%26GA_SAML_AC_CLASS_REF=
urn:gc-ca:cyber-auth:assurance:loa2"
```

5.2.1.4 Reading the language cookie

Users are able to change their session language while authenticating at the CSP. Therefore, you MUST read the session language from the common domain language cookie when they return before you display any unilingual application pages. It is best to read the language cookie once at the earliest possible moment after receiving the response from the CSP.

The easiest way to do this is to have GetAccess send the user to your language cookie service when they return, and then have your language cookie service redirect them to appropriate application page once it has read the cookie.

> *Tip: By default, when the user returns from the CSP, GetAccess will refuse to forward them to any URL in a secondary domain that is not listed under the <gaSaml2><gaSamlSPResourceSecondaryDomainList> configuration setting. You should ensure that the common domain (e.g. ".fjgc-gccf.gc.ca") is one of the domains listed (see the final two steps back in section 4.1).*

To implement this, simply modify the GAURI query parameter that you pass to GetAccess/Saml/SSO/Init on your chooser page to go through your language cookie service instead. For example, if the URL of your cookie service is https://myapp-mydept.fjgc-gccf.gc.ca/GACS and the application destination URL is the same as in the above example then the query parameter would look something like:

```
GAURI=https://myapp-mydept.fjgc-
gccf.gc.ca/GACS/?t=https://servername.mydept.gc.ca/Enrol/Confirm
.aspx
```

5.2.1.5 Putting it all together

The result of combining the previous three techniques results in a single hyperlink (or form target) URL that redirects the user to set the language cookie, then redirects the user to the CSP for authentication, then redirects the user to read the language cookie when they return, then finally redirects them to your application. This "quadruple redirection" requires some careful encoding of the various URLs that are nested within the query strings of other URLs to make sure that the right query parameters are consumed by the right code. Fortunately, the W3C rules for

parsing URLs are rather forgiving for most of the special characters, only the "&" query parameter separator needs careful attention. Assuming that you have configured a default level of assurance for the CSP (See step 10 in section 4.2 earlier) and the ultimate destination URL for the application requires two query parameters appParm1 and appParm2, here is an example that shows how all of this fits together:

https://app-dept.fjgc-gccf.gc.ca/GACS/?l=eng&t=https://servername.mydept.gc.ca/GetAccess/Saml/SSO/Init?GA_SAML_IDP=https://idp.entityid.com%26GAURI=https://myapp-mydept.fjgc-gccf.gc.ca/GACS/?t=https://servername.mydept.gc.ca/Application?appParm1=x%2526appParm2=y

As highlighted in the example, "&" must be encoded to %26 at the first level of nesting, and then double encoded to %2526 at the second level of nesting.

5.2.2 Obtaining Assertion Data

In order to integrate your enrolment, mapping and program applications with GetAccess, you need to configure the GetAccess Runtime to make the NameID (which contains the PAI) and the NameQualifier (the unique identifier of the CSP that issued the PAI) available to your applications via secure HTTP headers.

Get Access can pass the Name Id and Attributes to your application as XML strings (via <gaApplicationData> settings NAME_ID and ATTRIBUTES respectively). However, if you want to avoid parsing this XML in your application, SSC provides a "SamlCookieExtension" that will pass the NameId value (aka the PAI) and qualifier.

The SamlCookieExtension as provided by SSC depends on the AutoFederation extension to do the actual XML parsing. If you have installed and configured the AutoFederationExtension, then you can configure the SamlCookieExtension as follows:

Using the GetAccess Configuration Console, under <services> <INFRASTRUCTURE> <global> <gaExtensions>:

9. Under ...<gaExtensions> <onSuccessLogin>, create the simple setting <gaExtensionName> and set it to ca.gc.pwgsc.cats.gatoolkit.SamlCookieExtension.
10. Under ...<gaExtensions> < onSuccessLogin>, set the simple setting <gaExtensionEnabled> to true.
11. Re-start your GetAccess infrastructure.

The following table contains the names of the actual HTTP headers that will be set.

> *Important Note:*
>
> *All of these header values have been URL (percent) encoded. Your application must URL decode them before use.*

Header	Value
SCSAML_ISSUER_NAME	The name qualifier (SAML EntityID of the credential provider) as a URL Encoded string. You must use this value to distinguish Name Id's from different providers, since they are not necessarily unique. Populated by the SamlCookieExtension.
SCSAML_NAME_ID_VALUE	The value of the SAML Name ID (i.e. the PAI) as a URL Encoded string. Populated by the SamlCookieExtension.
SCSAML_NAME_ID	The full SAML Name ID element as an XML string. Enabled by configuring <gaApplicationData> to NAME_ID.
SCSAML_ATTRIBUTES	The complete list of SAML Attributes as an XML string. Enabled by configuring <gaApplicationData> to ATTRIBUTES.

5.2.3 Handling Authentication Failures

If a user is unable to authenticate while at the CSP, or if they choose to cancel, the CSP will return an authentication response to GetAccess with an unsuccessful status code. When this happens, GetAccess will treat this as an error condition and display an error page. Be default, this page displays a generic message such as "An error has occurred. Please contact your system administrator. General Saml Partner Error."

In order to handle these situations more elegantly, you can customize the "Warning Template.html" file that is used to produce this page. You will find two versions of this file under the Templates/Auth subdirectory of your GetAccess server installation directory. The English one will be in the en_US directory and the French one in the fr_FR directory.

> *Tip: By default GetAccess is unaware of the user's session language stored in the common domain language cookie so it will normally use the English version. Nevertheless you should customize both templates in case it does use the French one.*

One simple approach is to change the HTML in the warning template to perform an instant client-side redirect to an application page of your choosing via your language cookie service. For example:

```
<html>
   <head>
      <meta http-equiv="refresh"
            content="0;URL='https://app-dept.fjgc-
gccf.gc.ca/GACS/?t=https://app.dept.gc.ca/Sorry.aspx'" />
   </head>
</html>
```

This way you have control over the user experience while honouring the user's session language.

5.3 Logout

5.3.1 Initiating Logout

GetAccess provides a means for an application to explicitly invoke the SAML Logout process by having the browser send a request to the following URL:

https://*servername*/GetAccess/Saml/SLO/Init

Where *servername* is the name of your web server with the GetAccess Runtime you have configured as an Access Portal.

5.3.2 Post-logout

Upon completion of the logout request, GetAccess will forward the user to a URL that you can specify in the configuration setting <services><INFRASTRUCTURE><gaLogoutURL>. GetAccess will pass a "MESSAGE" query parameter to this URL with a value of "SamlLogoutSuccessful" if the SAML global logout succeeded. The logout page at this location MUST check this value. If GetAccess returns any "MESSAGE" value other than "SamlLogoutSuccessful" then an error occurred during the global logout process and the user may still be logged to an active session at another site. In order to safeguard the user's privacy your logout page should inform the user that they may not be fully logged out and advise them to close their browser.

6. Preparing for Production

In preparation of "Going Live" with your GetAccess deployment, you should consider enabling additional GetAccess security and high availability features.

6.1 Security Hardening

Chapter 17 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE provides information on changing default passwords, disabling unused administration accounts and enabling security features such as secure session cookies and TLS encryption between GetAccess components. Reading the sections on firewalls is especially important. If you have a firewall sitting between your web servers and the GetAccess infrastructure and that firewall is configured to disconnect idle TCP connections then GetAccess will not work reliably unless you make certain configuration changes.

6.2 Performance and High Availability

Once you have hardened your deployment you may also wish to do performance tuning or enable high availability. These features are described in chapters 14 and 16 of the GETACCESS SERVER SYSTEM ADMINISTRATION GUIDE.

A. Glossary of Acronyms

CATE	Client Acceptance Test Environment
CATS	Cyber Authentication Technology Solutions
CBS	Credential Broker Service
CIOB	Chief Information Officer Branch, Treasury Board Secretariat
CSP	Credential Service Provider
GC	Government of Canada
GCBCS	Government of Canada Branded Credential Service
GCCF	Government of Canada Credential Federation
GCCFMB	Government of Canada Credential Federation Management Board
GCCFO	Government of Canada Credential Federation Operator
HTTP	Hypertext Transfer Protocol
ICM	Internal Credential Management
LDAP	Lightweight Directory Access Protocol
LoA	Level of Assurance
MBUN	Meaningless But Unique Number
MITS	Operational Security: Management of Information Technology Security
OASIS	Organization for the Advancement of Structured Information Standards
PAI	Persistent Anonymous Identifier
SSC	Shared Services Canada
RP	Relying Party
PWGSC	Public Works and Government Services Canada

SAML	Security Assertion Markup Language
SOAP	Simple Object Access Protocol
SP	Service Provider
SSL	Secure Sockets Layer
TBS	Treasury Board Secretariat
TLS	Transport Layer Security
URI	Uniform Resource Identifier (see http://tools.ietf.org/html/rfc3986)
URL	Uniform Resource Locator (see http://tools.ietf.org/html/rfc3986)
WAM	Web Access Management
XML	Extensible Markup Language