Integrated Security Information Services Version 4
Developer's Guide

Document Revision 0020
Published: February 9, 2006

If you receive this document after August 2006, please visit http://mi6.ais.ucla.edu/isis/isis-4-developers-guide to retrieve the most current revision of this document.

NOTE: This document revision (0020) replaces all previous revisions of this document. All other revisions are now deprecated.

Abstract

This document provides detailed information on how to work with the ISIS4 common authentication service.

Contents

Getting Started

Welcome to the ISIS4 Developer's Guide. If you are a developer looking to use ISIS as the authentication mechanism for your application, this is the document for you.

ISIS4 Overview

Integrated Security Information Services (ISIS) is UCLA's common web authentication service. ISIS provides the following key functions:

- Provide user authentication via BOL ID/Password[1], QDB ID/Password[2], UID/PIN, and AIS's mainframe OASIS ID/Password via a secure common login server.
- Provide self-service user password change and reset[3].
- Enable single sign-on among campus web applications.
- Provide basic user identity, demographic, and role membership look-up information via the Attribute Service[4].

ISIS4 is built on a robust and highly scalable system architecture designed to enhance system security and to improve support for web single sign-on. Enabling your application to use ISIS for user authentication is straightforward. If your application meets the system requirements, using ISIS is a simple matter of redirecting the user to the ISIS Login Server (ILS) for user authentication, and upon successful authentication, querying the ISIS Web Service (IWS) to confirm the authentication and to retrieve user profile and session information.

Document Conventions

For the purpose of this document (and any discussions in the context of ISIS), we define the following terms:

- **Target Application** – Synonymous with *ISIS Enabled Application*.
- **Target** – Short for *Target Application.*
- **Target Administrator** – The application administrator is the person, typically the application developer, who signs into the ISIS administration utility to configure and manage the various target-specific preferences within ISIS.
- **ISIS Administrator** – The ISIS Administrator is basically the ISIS project team at AIS.

---

[1] Bruin Online (BOL) is UCLA's campus email and Internet service provider service. For more information, visit http://www.bol.ucla.edu.

[2] Query Database (QDB) is UCLA's data warehouse. See http://www.qdb.ucla.edu.

[3] ISIS4 supports password change and reset via each account's native password change/reset utilities. For additional information, visit https://i4w.ais.ucla.edu/ils/password.htm.

[4] This feature has not been completely implemented.

- **ISIS Administration Tool (IAT)** – A component of ISIS. The IAT is a web application used by the ISIS Administrator and the Application Administrator to manage ISIS configurations and settings.
- **ISIS Enabled Application** – An ISIS enabled application is any UCLA web application relying on ISIS for user authentication and single user sign-on. The rest of the document refers any ISIS Enabled Application as the "Target".
- **ISIS Login Server (ILS)** – A component of ISIS. The ILS is a web application for the users. It performs user authentication and password change and reset functions.
- **ISIS Web Service (IWS)** – A component of ISIS. The IWS is a set of web services used by ISIS enabled applications to query user session status, retrieve user attributes, and terminate user sessions.
- **User** – A user is a person who accesses campus applications that rely on ISIS for authentication.

System Requirements

The ISIS4 service is designed to support UCLA web application user authentication. In order to use ISIS, your application needs to meet the following requirements:

- Your application should be a web application.
- ISIS authentication is performed through a central login web application (ILS). Therefore, your application must be able to perform a "HTTP redirect" to redirect the user to ILS for authentication.
- Your application must supply an "authentication event handler" page/script/routine capable of accepting and processing the result of the user authentication attempt at ILS. For more details, see the Handling User Login section.
- Your application needs to be able to make SOAP web service calls. If the programming platform you use does not directly support SOAP, you can still implement workarounds by creating and parsing SOAP compliant XML messages directly within your code.
- Synchronize the clock on your server to the UCLA network time server. For more information, visit http://www.cts.ucla.edu/TimeSync.html.

Adding Your Application to ISIS

Before starting to use ISIS4, you need to sign up as a Target Administrator in ISIS. For details, please contact the ISIS project team (see Appendix A for contact information).

Architecture

ISIS4 is a collection of interconnected services. These components are:

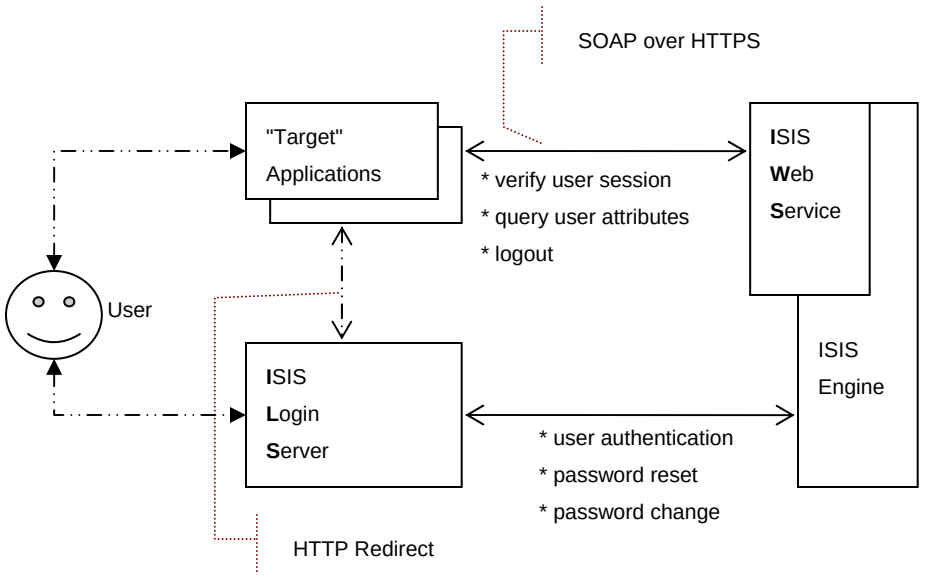

Figure 1: ISIS4 Architecture

- ISIS Login Server (ILS)
- ISIS Web Service (IWS)
- ISIS Administration Tool (IAT)
- ISIS Enabled Application (Target)

ISIS Login Server (ILS)

The ILS provides a secure, centrally managed, application-customizable login user interface. It is the only end-user-visible component of ISIS. All ISIS authentications are performed via the ILS[5].The ILS is responsible for the following tasks:

- **Centralized User Authentication** – The ILS provides a single, secure (SSL encryption), and robust login user interface. Within certain constraints, a target administrator can customize the login interface to conform to the look and feel of his/her application. For details on how to customize the ILS login interface, refer to the Working with the ISIS Administration Tool section in this document.

[5] Because of its close architectural ties with the UID system, URSA Online performs its own user authentication via the UID system and starts an ISIS session upon successful user authentication.

- **Self-service Password Change and Reset** – The ILS provides links to password change and reset websites for BOL and QDB accounts. Pending demand and available resources, password change and reset support for other ID types may become available in future service releases.
- **ISIS Enabled Application Directory** – The ILS provides a directory listing of ISIS enabled applications. This page is the default page displayed should a user hit the ILS without being directed from a source application. Listing an application in this directory is at the discretion of each application administrator.
- **ISIS Ticket Cookie Management** – Upon successful authentication, the ILS writes the ISIS ticket to the browser cookie cache. It also deletes obsolete ISIS tickets from the browser cookie cache where appropriate.

ISIS Web Service (IWS)

The IWS is the main programmatic interface to ISIS. It exposes a single sign-on and session management API to ISIS enabled applications. This API is implemented as a SOAP web service. The IWS provides the following functionalities:

- **GetSystemInfo** – The *GetSystemInfo* method returns basic system version and running configuration information.
- **VerifySession** – The V*erifySession* method performs the bulk of the ISIS web service functions:
  1. Return the user's session status, his/her UID (if available), his/her account ID's, and each account's status (suspended, active, authenticated, etc.).
  2. "Extend" the ISIS session's timeout by resetting the session's last visit timestamp.
  3. (Optional) If requested, return extended user demographic and role attributes from the Attribute Service.
- **Logout** – Ends the user's ISIS session.
- **StartSession**[6] – This is a reserved call for URSA use only.

ISIS Administration Tool (IAT)

The ISIS Administration Tool is a web application used by ISIS administrators and application administrators to register applications and manage application settings such as ILS login page preference and server IP address access control list. Using the IAT, application administrators can directly manage their applications' ISIS settings.

[6] Because of its close architectural ties with the UID system, URSA Online 2002 performs its own user authentication via the UID system and starts an ISIS session upon successful user authentication.

To get started using the ISIS Administration Tool, contact the ISIS support team (see Appendix A for contact information). Also refer to the ISIS Administration Tool User's Guide for additional details.

ISIS Enabled Applications

An ISIS enabled application, or target application ("target" for short), refers to a campus web application using ISIS to perform user authentication. An application becomes an ISIS Enabled Application when it uses ISIS to perform user authentication. In addition, an ISIS Enabled Application participates in the ISIS single sign-on community by supporting user authentication via ticket verification. All ISIS Enabled Applications are integral components of this common authentication environment. ISIS's success depends on the cooperation from each participating application to adhere to some simple programming and application management guidelines.

Programming with ISIS4

ISIS4 provides a common framework to perform the following key user authentication and session management tasks:

- Handle the initial user login
- Identify User and Validate User Session Status
- Change and Reset user passwords
- Logout

This section describes the programming essentials you need to know to use the ISIS authentication service and make your application an ISIS enabled application.

Handling the Initial User Login

When a user visits your application, one of your first tasks is likely to perform user login. Doing so using ISIS involves performing the following sequence of events:

1. Determine if the user has been previously authenticated and has an active ISIS4 session. If the user has been authenticated, skip to step 3. Otherwise,
2. Perform user login using ISIS4.
3. Retrieve the user attribute data from some user data store to determine whether the user has permission to access the requested resource/page.
4. If the user has permission, deliver the requested resource/page to the user.

To exammine each step in more detail:

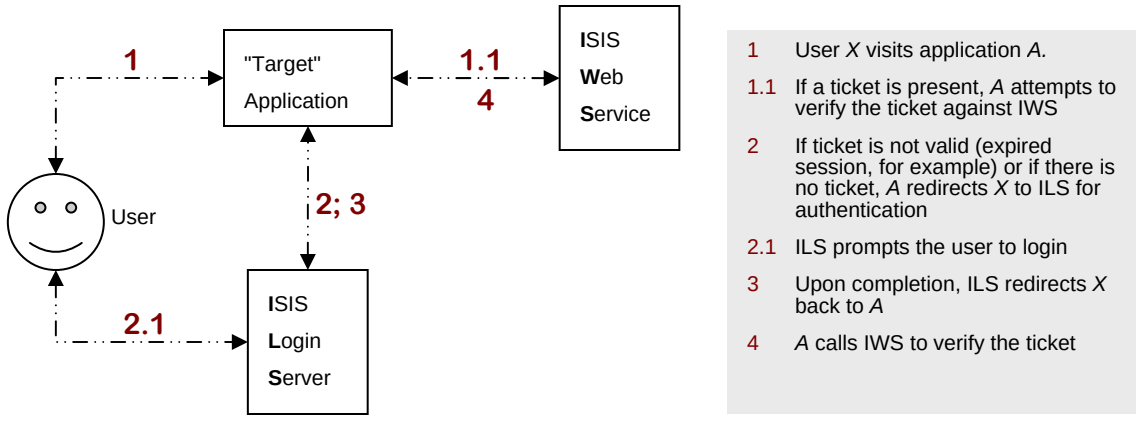

Figure 2: Typical ISIS User Authentication Scenario

Step 1: Determine the user's authentication and session status

When a user visits a target application, the target first determines whether the user has already logged in. This is done by checking the user's browser cookie cache for the presence of the ISIS ticket. If a ticket is present, call the IWS *VerifySession* web service (see Verify the ISIS Session) to find out if the ticket is valid and the session associated with the ticket is active.

The cookie is stored at the ucla.edu domain level. The parameter name is *edu.ucla.isis4.ticket.* (For detailed information on the ISIS ticket format, see Working with ISIS4 Ticket).

If there is no ISIS ticket in the browser cookie cache or the ticket turns out to be invalid, redirect the user to the ILS for authentication.

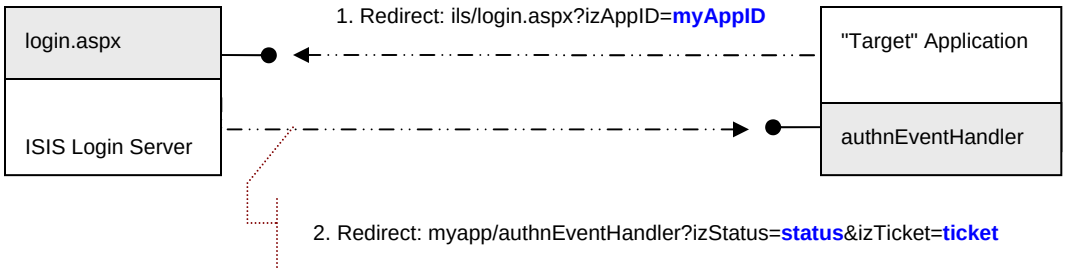

Figure 3: Login Redirect Details

Step 2: Authenticate the user

If the target determines that the user does not have a valid ISIS session, it redirects the user to the ISIS Login Server for login. ISIS4 performs user authentication via the ISIS Login Server (ILS) application hosted at https://i4w.ais.ucla.edu/ils/login.aspx.

When redirecting the user to the ILS, the target needs to submit its application ID as a HTTP query string parameter. Example 1 illustrates the redirect using the HTML Meta tag. Example 2 demonstrates the same redirect from an ASP page using VB script.

```
<html>
<head>
...
<meta http-equiv="refresh"
content="0;URL=https://i4w.ais.ucla.edu/ils/login.aspx?izAppID=myAppID">
</head>
...
</html>
```

Example 1: Redirect to ILS using HTML Meta tag

```
...

If Not IsUserAuthenticated Then
   Response.Redirect https://i4w.ais.ucla.edu/ils/login.aspx?izAppID=myAppID
End If
```

```
...
```

Example 2: Redirect to ILS from an ASP Page

Upon completion of the authentication process, the ILS returns the user back to the target by redirecting him/her to a pre-designated "authentication event handling URL" within the target. This URL, configurable via the ISIS administration tool, should point to a dynamic script or a HTTP request processor such as an ASP page or a Perl program. It needs to parse the query string in order to determine the outcome of the user's authentication attempt and act accordingly. ISIS4 passes two common parameters in the query string: *izTicket* and *izStatus*. *izTicket* contains the ISIS ticket if the authentication was successful. *izTicket* returns nothing if the authentication is not successful. *izStatus* contains the result status code of the authentication request. A third parameter, *izTargetUrl*, is a pass-through parameter for use by the target to pass any values it needs during the redirect operations. It is an optional parameter and will only appear if the target passes an *izTargetUrl* parameter in its redirect to the login page. See the ILS Reference section for a complete description of these parameters.

Example 3 shows a sample ASP VB script snippet processing these parameters.

```
<!-- #include file="IsisWebService.vbs" -->

<%
' -- In case you are wondering, the "IsisWebService.vbs" include file is the actual
' -- module containing code that'd actually call the ISIS web service. There are
' -- several possible implementations of that module and will be discussed
' -- separately.

Dim ticket
Dim authnStatus
Dim iws

If Request.QueryString("izStatus") <> "" Then
  authnStatus =  Request.QueryString("izStatus")
End If

If Request.QueryString("izTicket") <> "" Then
  authnStatus =  Request.QueryString("izTicket")
End If

Select Case authnStatus
    Case "S"   'authentication successful

        ' call IWS's VerifySession web service to verify ticket
        ' in this example, we assume an ASP VB class named IsisWebService
        ' has been created and that the web service call is wrapped within
        ' a method called verifySession.
        Set iws = New IsisWebService()
        If iws.VerifySession(ticket, Request.ServerVariables("remote_addr")) Then
            ' session is valid
            Set iws = Nothing
            ' <todo>
            ' deliver user requested resource
            ' </todo>
        Else
            Set iws = Nothing
            ' <todo>
            ' Handle invalid ticket error.
            ' </todo>
```

```
        End If
    Case "L", "R"   'authentication unsuccessful. Account locked out

        ' <todo>
        ' perform application specific stuff to handle account lockout
        ' </todo>

    Case "F", "U"   'authentication unsuccessful. User cancelled request

        ' <todo>
        ' perform application specific stuff to clean up
        ' </todo>

    Case "E"   'Unable to perform authentication due to system error

        ' <todo>
        ' perform application specific stuff to handle ISIS outage,
        ' possibly fail over to back up authentication scheme or
        ' allow user limited unauthenticated access to your site.
        ' </todo>

    Case Else

        ' <todo>
        ' handle unexpected ILS response
        ' </todo>

End Select
```

Example 3: ASP authentication event handler

If the user's authentication attempt is successful, ILS starts the user ISIS session just prior to returning the user to your application. The *izTicket* parameter contains the session ticket corresponding to that session. To complete the authentication sequence, the target must call the V*erifySession* web service to verify that the ticket returned via the redirect has not been tampered with. In addition to being a security check, the *VerifySession* web service also returns user identity and demographic information. Continue to step 3 for more details.

Step 3: Retrieve user identity and demographic data

Once the user has successfully authenticated, a target may need to retrieve the user's identity and possibly demographic and campus attribute data in order to determine whether the authenticated user has access to the target application. For most targets, this may be a two part process. First, call the ISIS *VerifySession* web service to retrieve user identity and common demographic data. Then if necessary, the target may query its own access control database to determine the user's specific access.

Because the HTTP Redirect operation is not a secure message exchange mechanism, ISIS does not return user identity data to the target during the redirect. Instead the target must query the IWS web service directly. Specifically, the target should call the *VerifySession* web service. For details, see the Verifying the ISIS Session section later in this document.

IMPORTANT: Even if a target does not need to know the user's identity, it should still call *VerifySession* to make sure that the ticket returned via redirect has not been tampered with.

Step 4: Deliver the user requested resource/page

If a user authenticates successfully and passes the authorization check (VerifySessoin and any target-specific access control checks), the target then delivers the user-requested resource/page to the user. The specific implementation of how this is done will differ in each target. A likely scenario might be to redirect the user from the authentication event handler URL to the particular page he/she requested using HTTP redirect.

Managing/Verifying the ISIS Session

Handling the initial user login is only one part of managing the ISIS session life cycle. Once a user has authenticated, a target should periodically query ISIS to validate the user's session status by calling the *VerifySession* web service method. This query allows a target to:

- Validate that the ISIS session represented by the ISIS ticket is active.
- Check the user's individual account status (suspended, active, authenticated, etc.).
- Retrieve user identity, demographic, and campus attribute data

Passing sensitive user data using HTTP redirect is unreliable and undesirable. Instead ISIS4 exchanges all sensitive data with the target applications using a set of SOAP XML Web Service over HTTPS[7]. Specifically, ISIS4 exposes a single web service method call, *VerifySession*, to perform the outlined tasks.

The *VerifySession* call is perhaps the most critical and the most used function in ISIS4. When a target receives a user's ISIS ticket (either via the ILS redirect or from the browser's cookie cache), it calls *VerifySession* to confirm that the user's session ticket is valid and that its associated session is alive. If the user is new to the target, the target uses *VerifySession* to retrieve user identity and other attribute data in order to determine user access and to retrieve profile data. Depending on the target's implementation, it may end up calling *VerifySession* at the start of every page request.

For most web application development environments, calling a SOAP web service should be relatively straight forward. The specific implementation depends on the technology used by a target. Appendix E lists a number of SOAP and web service related resources. Additional resources are available online at the AIS Technology Infrastructure Group web site (http://www.ais.ucla.edu/tig).

Change and Reset user passwords

ISIS4 leverages the password change and reset functions of each account type to perform user password change and reset. https://i4w.ais.ucla.edu/ils/password.htm lists these password management tools. To provide a link to password change and reset functions from within a target application, link to this URL.

---

[7] If you are not familiar with SOAP, XML, or web service, please contact the ISIS development team. We will be glad to help. We will also make various learning and reference resources available online.

Logout

When a user is done and wants to logout, the target should call the ISIS *logout* web service as part of its logout process. Calling *logout* terminates the user's ISIS session.

Design and Coding Considerations

ISIS is a shared resource. Its success and smooth operation depend on the cooperation of each participating application. When developing application to use ISIS's features, please observe these ground rules and guidelines:

General Rules

- *The application should be a web application.* ISIS is designed to support web applications using HTTP/HTTPS. It requires the application to render its user interface in a web browser. This means that the application should render HTML pages and support standard HTTP operations such as HTTP redirect.
- Call the VerifySession web service. Always call the *VerifySession* web service to validate the ticket returned by ILS. In addition, call *VerifySession* whenever you need to revalidate the user's session credentials. DO NOT call *VerifySession* more than once per web page delivery. There is no reason to do so.
- Provide a "Logout" function/link/button in the application's user interface. It seems obvious, but it's an often neglected feature in web applications. Clicking this function should trigger the application to call ISIS's *Logout* web service method. It should also log the user out of your application.

Working with the ISIS4 Ticket

ISIS4 user sessions are represented by a string token called a ticket. The ISIS4 ticket is a key component of the ISIS architecture. It is a part of the unique token used to identify an authenticated ISIS4 user session. The ISIS4 ticket is a variable length string ranging from 32 to 128 characters.

One way to obtain this ticket is to grab it from the query string during the redirect to the application's authentication event handler page. This ISIS ticket is also stored in the user's browser cookie cache at the *.ucla.edu* domain level under a parameter named *edu.ucla.isis4.ticket*. The browser cookie cache is the official location of the ISIS4 ticket. Whenever you need to access the ISIS4 ticket, always check the user's cookie cache[8].

Because the ISIS4 ticket is the session identifier for an ISIS4 session, it is important that you follow the following guidelines:

- *Do not modify the ISIS4 ticket in anyway.* The ISIS4 ticket is stored as a browser cookie in the user's browser cookie cache. The cookie is stored at the *.ucla.edu* domain level. Treat this cookie as a read-only value. Do not overwrite the data. Do not append your own cookies with the cookie name *edu.ucla.isis4.ticket*. Do not delete the ISIS4 ticket cookie when the user logs out. Instead, call the *Logout* web service method. Let the ISIS Server components (specifically, ILS) perform the ISIS4 cookie clean up tasks.

[8] Because the ticket is stored in the .ucla.edu domain, if your web application is not served through an URL with a *.ucla.edu host name, you will not be able to read this ticket from the browser's cookie cache.

- *Do not assume or hardcode the ISIS4 ticket size.* As of this release, the ISIS4 ticket is a 64 character string. However, do not assume that is the length of a ticket. The ticket length could fluctuate from 32 to 128 characters in the future without notice. This is particularly important if you are programming in a language such as C where typical string declaration (char array) requires a pre-determined length.
- *Encode the ISIS4 ticket where appropriate.* The ISIS4 ticket can potentially contain characters that are not HTTP URL safe. If you need to transmit the ticket as part of the HTTP GET request for any reason, make sure you encode it before attaching it to the URL. By the way, this rule also applies if you store the ticket in external stores such as SQL servers. Apply the appropriate encoding methods when working with the ISIS4 ticket.

Working with the ILS

There are a few simple rules when working with the ISIS4 login server:

- *Do not create your own login page.* Redirect authentication requests to the ISIS Login Server (ILS). Do not attempt to "screen scrape" the ILS login page to perform your own user authentication.
- *Do not frame the ILS pages.* Do not attempt to place any of the ILS pages within a frame or any other container window within your application.
- *Implement code to handle all possible return codes from ILS*. It may be tempting to handle only the successful login case and ignore the rest by redirecting the user back to ILS. That, in fact, is the incorrect behavior. Make sure you understand the meaning of each return code passed in the *izStatus* parameter and handle each accordingly.

Working with the IWS

The ISIS Web Service runs on a redundant, scalable, and high performance hardware platform. It is, nonetheless, a shared resource. Be considerate, don't abuse the service:

- Create error-free request XML messages. It sounds obvious, but bad requests waste precious server resources. Make sure your code always generates syntactically correct and semantically meaningful requests.
- Pass in all requested information. The V*erifySession* web service requires, among other things, the user's browser IP address. This is a required field. Validate the format of that IP address and report the real user IP address to ISIS.
- When using V*erifySession*, set the "returnExtendedAttributes" parameter to "True" only if you need that information. There is a considerable performance overhead to query that information. As a rule of thumb, request the user attribute information only once at the start of the user's session in your application. Store that information in your application's session state instead of re-querying it from ISIS on every page.
- If you do retrieve the extended user attributes, note that the information you receive is not public information. Do not re-transmit or display the data to unauthorized parties. Do not store the data beyond the length of the user session in your application.

- Call *Logout* only if the user explicitly chooses to end his/her ISIS session by clicking the logout button/link. Remember: the fact that the user is done with your application does not necessarily mean he or she is done with all of his/her other work.
- Set timeouts in your application when making IWS calls. When you call any IWS function, if possible, set a short call timeout value (say, 3 seconds), in your HTTP request. This protects your application from being impacted by any unlikely performance delays in IWS.

Supporting Single Sign-on

A major feature of ISIS is its support for single web application sign-on. When developing an ISIS enabled application, remember that even if a user is accessing your application for the first time, he or she may have already been authenticated from another ISIS enabled application. Do not automatically redirect the user to the ISIS login server. To ensure the best user experience, do the following when a user hits your application:

- *Always check the browser cookie cache for the presence of the ISIS ticket cookie before you direct a user to ILS for authentication.* The ticket is stored in the parameter *edu.ucla.isis4.ticket* at the ucla.edu domain. If a ticket is present, call the V*erifySession* ISIS web service to check the status of the session. Redirect the user to the ISIS login server (ILS) for authentication only if the user session is not active or if the ISIS ticket cookie is not present.
- *Support all authentication types.* Unless your application has special security requirements, do not redirect the user to ILS for authentication simply because he or she has not authenticated using your preferred login ID. Accept the session as long as it is active with at least one of the ISIS supported login types. If your application has special security requirements requiring you to prompt for a specific login type, contact the ISIS development team.
- *Call the VerifySession web service often.* In a single sign-on environment, the user may log out of ISIS via another application in between visits to your application. Call *VerifySession* periodically to make sure the user is still logged in. Remember: among other things, the *VerifySession* web service call extends the user's session timeouts within ISIS. Even if you maintain your own session state, call the *VerifySession* every once in a while to extend the user's ISIS session.

Technical Reference

ILS Reference

Login Server

URL: https://i4w.ais.ucla.edu/ils/login.aspx

Input Query String Parameters

The following are query string parameters you need to/can specify when redirecting the users to the login server:

Parameter Name	Description
izAppId	*Required.* Submit your application's application ID using this parameter. You must specify the correct application ID when redirecting to the login server in order for ILS to load the proper login page preference for your application. For example, if your application id is edu.ucla.ais.sar, the redirect URL would look like: https://i4w.ais.ucla.edu/ils/login.aspx?izAppId=edu.ucla.ais.sar
izTargetUrl	*Optional.* This optional parameter gives your application the ability to pass application-specific data through ILS to your event handling page. ILS does not process the content of this parameter. Instead, it passes the entire string back to you using the same parameter name (izTargetUrl) when redirecting to your authentication event handler page. A typical use of this URL might be to embed a particular URL in your application (other than default) you'd like to send the user to after he or she authenticates For example: https://i4w.ais.ucla.edu/ils/login.aspx?izAppId=edu.ucla.ais.sar&izTargetUrl=https%3a%2f%2fsar.ais.ucla.edu%2fpage3.cgi NOTE: Make sure to URL encode your parameter data when submitting it using the izTargetUrl parameter.

Output Query String Parameters

Upon completion of the authentication processing, ILS returns the user to your application by redirecting him/her to your ISIS authentication event handler page. During this redirect, ILS attaches the following query string parameters in the URL:

Parameter Name	Description
izStatus	*Required.* The izStatus parameter contains the result of the user's authentication request. You should always check this parameter first

	before processing others. The valid values for this parameter are: **S** – Authentication Successful. The user successfully authenticated. **F** – The user cancelled his/her login attempt. A return code of "F" indicates that the user had attempted to login unsuccessfully at least once during this session. **U** – The user cancelled his/her login attempt. A return code of "U" indicates that the user never attempted to login during this session. **L** – The user's account is locked out either due to consecutive failed login attempts or other prior account suspension conditions. **R** – The user's login attempt failed because the account is in an "reset needed" state. **E** – ISIS was unable to authenticate the user because of system failure.
izTicket	*Optional.* If the user successfully authenticates, ILS returns the user's new ISIS session ticket in this parameter. NOTE: You can also find the ticket in the user's browser cookie cache
izTargetUrl	*Optional.* (see ILS input query string parameters) If you specified an izTargetUrl when redirecting the user to ILS, you'll receive it back here. Otherwise, this parameter is empty.

Password Management

ISIS4 leverages the password management utilities provided by each account type to provide self-service password change and reset (if available). To find the available password change and reset options for account ISIS accepts for user authentication, visit https://i4w.ais.ucla.edu/ils/password.htm.

IWS Reference

The ISIS4 Web Service (IWS) is a SOAP web service with 4 methods:

- **GetSystemInfo** – The *GetSystemInfo* method returns basic system version and running configuration information.
- **VerifySession** – The V*erifySession* method performs the bulk of the ISIS web service functions:
    4. Return the user's session status, his/her UID (if available), his/her account ID's, and each account's status (suspended, active, authenticated, etc.).
    5. "Extend" the ISIS session's timeout by resetting the session's last visit timestamp.

6. (Optional) If requested, return extended user demographic and role attributes from the Attribute Service.

- **Logout** – Ends the user's ISIS session.
- **StartSession**[9] – This is a reserved call for URSA use only.

The precise/technical description of the ISIS Web Service can be found as a WSDL document at https://i4w.ais.ucla.edu/iws/v4.asmx?WSDL (Check Appendix B for WSDL listing for testing and other alternate environments). The remainder of this section provides sample illustrations of these SOAP request/response messages.

VerifySession

The following example illustrates a sample HTTP POST request message for the *VerifySession call*. The **placeholders** shown need to be replaced with actual values:

```
POST /iws/v4.asmx HTTP/1.1
Host: myapp.ucla.edu
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://isis.ais.ucla.edu/ws/VerifySession"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <wsConsumerCredential xmlns="http://isis.ais.ucla.edu/ws/">
      <password>string</password>
      <id>string</id>
    </wsConsumerCredential>
  </soap:Header>
  <soap:Body>
    <VerifySession xmlns="http://isis.ais.ucla.edu/ws/">
      <ticket>string</ticket>
      <userIpAddr>string</userIpAddr>
      <returnExtendedAttributes>boolean</returnExtendedAttributes>
    </VerifySession>
  </soap:Body>
</soap:Envelope>
```

Here is the matching successful/normal HTTP response from the *VerifySession* call:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
```

[9] Because of its close architectural ties with the UID system, URSA Online 2002 performs its own user authentication via the UID system and starts an ISIS session upon successful user authentication.

```
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <VerifySessionResponse xmlns="http://isis.ais.ucla.edu/ws/">
      <iwsResponse action="[Start|Verify|Logout]" hasErrors="boolean"
                   transactionId="long">
        <sessionInfo status="[Uninitialized|Active|ActiveWithCachedCredential
                              |ActiveWithMultipleCredentials|Expired]">
          <userAttributes>
            <attribute name="string" value="string"/>
            <attribute name="string" value="string"/>
          </userAttributes>
          <ticket>string</ticket>
          <uclaId>string</uclaId>
          <accounts>
            <account loginId="string"
                     type="[BruinOnline|ACF2|QDB|UID|CommonLogon]"
                     status="[Unknown|Active|Authenticated
                              |AuthenticatedWithCachedCredential|NotAuthenticated
                              |AccountSuspended|AccountExpired
                              |PasswordResetRequired]" />
            <account loginId="string"
                     type="[BruinOnline|ACF2|QDB|UID|CommonLogon]"
                     status="[Unknown|Active|Authenticated
                              |AuthenticatedWithCachedCredential|NotAuthenticated
                              |AccountSuspended|AccountExpired
                              |PasswordResetRequired]" />
          </accounts>
        </sessionInfo>
        <errorInfo count="int">
          <errors>
            <error errorCode="int" detail="string" />
            <error errorCode="int" detail="string" />
          </errors>
        </errorInfo>
      </iwsResponse>
    </VerifySessionResponse>
  </soap:Body>
</soap:Envelope>
```

There may be 0 or more <attribute> xml elements.

There may be 0 or more <account> xml elements

There may be 0 or more <error> xml elements.

Logout

The following example illustrates a sample SOAP request and response pair for the *logout* call. The **placeholders** shown need to be replaced with actual values.

Request message:

```
POST /iws/v4.asmx HTTP/1.1
Host: myapp.ucla.edu
Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "http://isis.ais.ucla.edu/ws/Logout"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <wsConsumerCredential xmlns="http://isis.ais.ucla.edu/ws/">
```

```
      <password>string</password>
      <id>string</id>
    </wsConsumerCredential>
  </soap:Header>
  <soap:Body>
    <Logout xmlns="http://isis.ais.ucla.edu/ws/">
      <ticket>string</ticket>
      <userIpAddr>string</userIpAddr>
    </Logout>
  </soap:Body>
</soap:Envelope>
```

Response message:

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xmlns:xsd="http://www.w3.org/2001/XMLSchema"
               xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <LogoutResponse xmlns="http://isis.ais.ucla.edu/ws/">
      <iwsResponse action="Logout" hasErrors="boolean" transactionId="0">
        <sessionInfo status="Expired">
        </sessionInfo>
        <errorInfo count="int">
          <errors>
            <error errorCode="int" detail="string" />
            <error errorCode="int" detail="string" />
          </errors>
        </errorInfo>
      </iwsResponse>
    </LogoutResponse>
  </soap:Body>
</soap:Envelope>
```

There may be zero or more <error> xml elements.

Appendix A: Contact Information

ISIS Project Team:

Manager:	Albert Wu albertwu@ucla.edu +1 310 825 1933
Developer(s):	Xiaoling Zhang xzhang@ais.ucla.edu +1 310 794 4973
Web Site:	http://mi6.ais.ucla.edu/isis/
AIS Help Desk:	+1 310 206 6951

Appendix B: WSDL for ISIS Web Service

You can find the latest Web Service Description Language document for the ISIS4 web service at the following URL's:

WSDL for the production environment: https://i4w.ais.ucla.edu/iws/v4.asmx?WSDL

WSDL for the test environment: http:/isisdev1.tig.ucla.edu/iws/v4.asmx?WSDL

Appendix C: ISIS4 Exceptions

The following table lists ISIS exceptions you may receive when calling the ISIS Web Service. These exceptions are reported in the <errorInfo> section inside the <IwsResponse> tag.

NOTE: Though very unlikely, it is possible that you may encounter generic transport level errors such as SOAP exceptions, HTTP protocol errors, and TCP/IP socket errors. They are not listed here, but you should check for and handle those errors.

Exceptions You'll likely Encounter

These exceptions are the "public" exceptions you may see in the IwsResponse. Check for them in your code.

Isis Web Service Exceptions

Error Number	Description
700999	Session Start up Failed.
700001	An unspecified error occurred. This may be due to invalid input data format and size. Please double check your input.
702010	No entry found in the access control list store.
704001	Permission Denied. The current security credential does not have permission to perform this operation.
704002	Application Authentication Failed. The web service consumer's security credential is invalid.
704999	(not in active use) User login failed.
705001	IWS encountered unexpected invalid ISIS session state while attempting to load session information.
705002	IWS encountered unexpected invalid ISIS session state while attempting to start session.

Session Exceptions

Error Number	Description
600003	Unknown error occurred while saving session info to store.
600007	An error occurred while calculating session state.
600100	Unable to create a Ticket.
601001	An error occurred while reading from the configuration file.
602001	A database error has occurred. Unable to retrieve session from data store.

602002	A database error has occurred. Unable to update session data store.
602003	Unknown database error occurred while saving session info to store.
602004	Incorrect login id. Session update failed.
602999	An unspecified database error has occurred.
604001	Permission Denied. The current security context does not have permission to perform this operation.
604005	Security Violation: The supplied login ID does not belong to the current session user.
604010	Security Violation: ISIS is unable to verify the authenticity of the ISIS ticket or the supplied user IP address.
605001	Duplicate Session ID. Session creation failed.

Exceptions You'll Most Likely Never See

These are internal ISIS exceptions. You should never see these, but they are listed here for completeness.

Account Exceptions

Error Number	Description
200010	Placeholder method only. Function not yet implemented.
200001	An error occurred while reading from the configuration file.
201001	No QDB server address defined in config file.
201002	No BOL authentication server address defined in config file.
201003	No Acf2 authentication server address defined in config file.
202010	A database error occurred while reading UID Cached Table.
202011	A database error occurred while performing UID authentication.
203001	Cannot write to BOL request stream.
203002	Cannot read from BOL response stream.
203003	Unable to connect to any of the BOL authentication servers. Authentication attempt failed.
203004	Unable to connect to any of the QDB servers. Authentication attempt failed.
203005	Cannot write to ACF2 authentication server request stream.
203006	Cannot read from Acf2 authentication server response stream.

203007	Unable to connect to any of the Acf2 authentication servers. Authentication attempt failed.
204001	Permission Denied. The current security context does not have permission to perform this operation.
204100	Acf2 authentication server denied the authentication request because of an access violation.
204101	UID authentication server denied the authentication request because of an access violation.
204102	UID cache server denied the authentication request because of an access violation.
205001	BOL Server returned an unknown response.
205002	Invalid BolAuthenticationStatus enum.
205003	Invalid QdbAuthenticationStatus enum.
205004	Unexpected UID Authentication Status.
205008	Undefined Acf2 authentication response.

Attribute Service Exceptions

Error Number	Description
500002	An unexpected error occurred while instantiating an object of AttributeService class.
500020	Method not yet implemented.
500003	An unexpected error occurred in [AttributeService].GetUId.
502001	A database error has occurred. Unable to query the Attribute server.
504005	A security violation occurred while attempting to retrieve user UID from the Attribute Service.
505001	An error occurred while querying the Attribute Service.
506001	A valid login id is not provided in GetUID.
506002	A proper login type is not provided in GetUID.
506005	The user account cannot be “Nothing” when getting the user's credentials.
506010	A valid UID is needed in order to get the user's attributes.
506015	Attribute service query returned no result.

Configuration Exceptions

Error Number	Description
800010	Hey! The service type is not supported.
801001	An error occurred while reading the configuration file.

Appendix D: Changes from Previous Versions

There are two older versions of ISIS in current use. The first and far more common is the C API version referred to as ISIS 2. Most applications using ISIS use this version. The other is ISIS 3. ISIS 3 is a relatively new web service based version used by a handful of applications. ISIS 3 shares a similar architecture to ISIS4.

If you are using ISIS 3, much of the information described in this document will be familiar to you. However, please study this document carefully as there are some significant implementation differences between ISIS 3 and 4.

If you are using ISIS 2, there are several significant changes to the ISIS architecture. These changes require modifications to your software. The following is a list of key changes:

- **Centralized Logon Server** – ISIS4 requires all applications to redirect users to a common login server for user authentication. This is perhaps one of the most significant changes for ISIS 2 developers. For security reasons, ISIS4 does not provide an open API for an application to perform user authentication. Instead, you must redirect the user to the ISIS Login Server (ILS) to login.
- **Changes in Server Certificate Requirement -** Since the application no longer hosts the login page, server digital certificate is not required for your application. However, you may still want to install it if you need to secure your application data over the wire.
- **Web Service Interface –** A new SOAP web service interface replaces the old C socket API and the COM component. Depending on your experience with web services, this new interface may take a little getting used to. We will work with everyone directly to ensure a smooth transition.
- **Cookie based Ticket Exchange –** In previous ISIS versions, there was not a defined mechanism for passing an ISIS ticket from one application to the next to support single sign-on. ISIS4 specifies a cookie based ticket exchange mechanism among applications to unify and improve support for single sign-on. NOTE: ISIS4 has a new ticket format. For more details, see Working with ISIS4 Ticket.
- **Security Changes –** ISIS4 adds a slew of new security and auditing features. One of these is a centralized, comprehensive authentication activity logging from ILS. Another is the tracking of end user IP addresses. While none of these measures can completely halt attempts to compromise systems, the combination of these security measures[10] should provide an effective deterrent. To help make ISIS as secure as possible, we ask that you continue to log user access in your application. In addition, synchronize your server's system clock with the UCLA time server hosted by CTS. It will make cross-log security analysis much easier.

[10] For security reasons, this document does not disclose all of the security measures in place. If you are interested in finding out more about security in ISIS, please contact the ISIS team at AIS directly.

- **Distributed Administration –** ISIS4 has a brand new administration utility. It enables application administrators and developers to register new servers and configure their settings within ISIS directly.

Appendix E: Additional Links and Resources

Web Service and SOAP Information

"Top Ten FAQ's for Web Services"
http://www.oreillynet.com/lpt/a//webservices/2002/02/12/webservicefaqs.html

"Understanding SOAP"
http://msdn.microsoft.com/webservices/understanding/webservicebasics/default.aspx?pull=/library/en-us//dnsoap/html/understandsoap.asp

Simple Object Access Protocol (SOAP) 1.1 Specification
http://www.w3.org/TR/SOAP/

"Making SOAP out of Java"
http://www.fawcette.com/Archives/premier/mgznarch/javapro/2001/04apr01/prs0104/prs0104.asp

"Leveraging XML with ColdFusion"
http://download.macromedia.com/pub/devnet/downloads/sybex_cfdhb-4029c07.pdf

"SOAP::Lite for Perl"
http://www.soaplite.com/

Additional Protocols and Standards Resources

Naming and Addressing: URIs, URLs, …
http://www.w3.org/Addressing/

HTTP – Hypertext Transfer Protocol
http://www.w3.org/Protocols/

HTTP Made Really Easy
http://www.jmarshall.com/easy/http/

Appendix F: Document Change Log

Changes in Revision 0020:

- Updated the ISIS Support site URL.
- Refreshed the document expiration date.

Changes in Revision 0019:

- Minor style change.
- Removed mentions of RACF authentication since it is no longer supported.

Changes in Revision 0018:

- Created the Document Change Log.
- Corrected the incorrect ISIS ticket parameter in the "Support Single Sign-on" section.