Single Sign-on Overview Guide

1/24/2017 Blackbaud NetCommunity 7.1 Single Sign-on Overview US

SSO-2016

Contents

Single Sign-on Overview

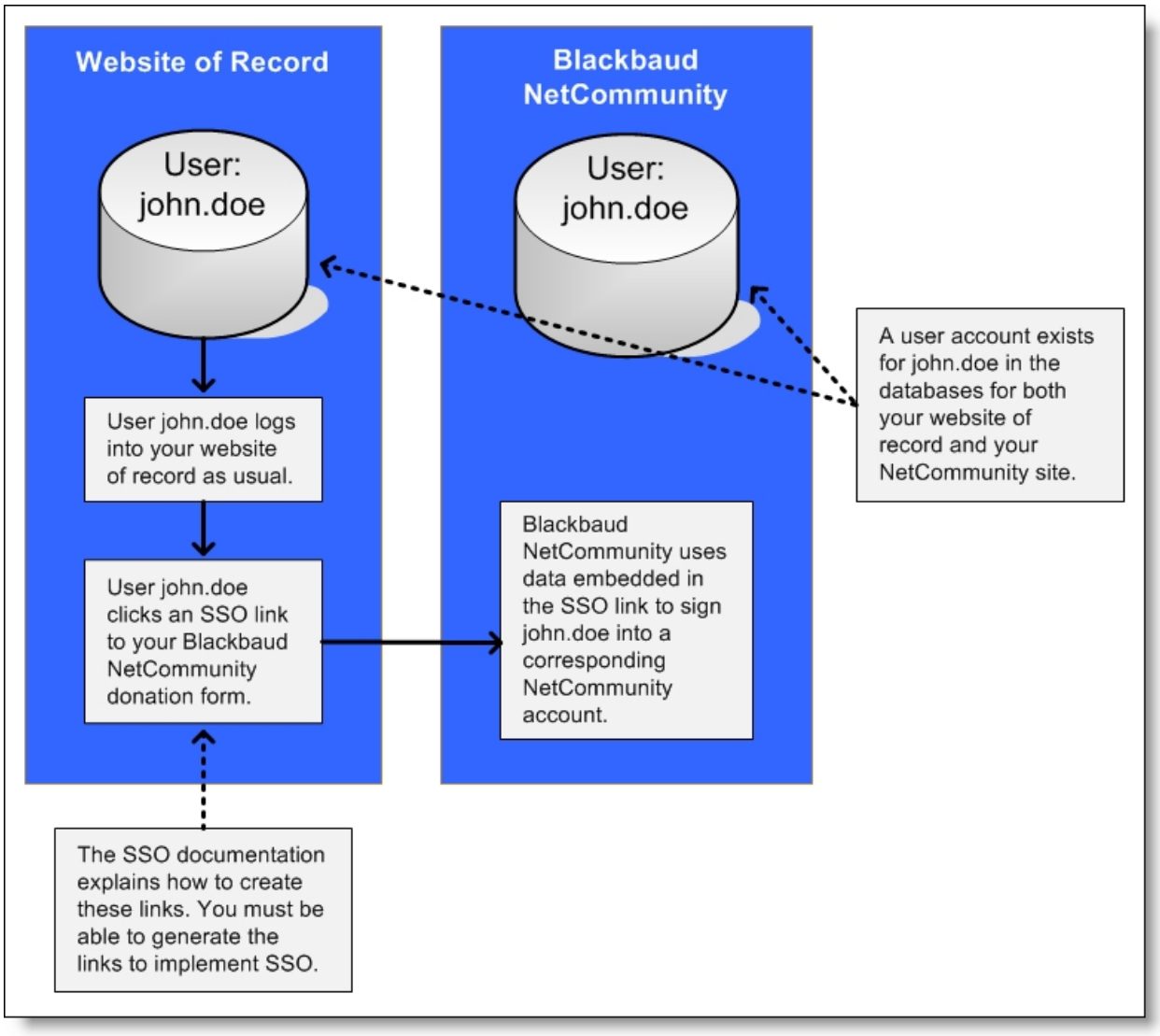
Blackbaud NetCommunity provides two options for single sign-on (SSO) to allow website users to log in to both your Blackbaud NetCommunity site and another third-party site without the need to log in twice.

- An incoming single sign-on option allows website users to sign in to a third-party site and automatically log in to your Blackbaud NetCommunity site as well. For information about this option, see Incoming Single Sign-on on page 4.
- An outgoing single sign-on option allows website users log in to your Blackbaud NetCommunitysite and automatically log in to a third-party website as well. For information about this option, see Outgoing Single Sign-on on page 13.

Incoming Single Sign-on

The incoming single sign-on (SSO) feature in Blackbaud NetCommunity allows website users to transparently sign in to their Blackbaud NetCommunity accounts without re-entering their login information. This incoming SSO functionality allows users who are authenticated via another system ("Website of Record") to be transparently signed in to their corresponding Blackbaud NetCommunity accounts. This option is useful for the scenario where the source of registrations on your website is a system other than Blackbaud NetCommunity.

The following illustration outlines how the incoming single sign-on option works in Blackbaud NetCommunity.



Single Sign-on Requirements

For the single sign-on option to work, website users must have two user accounts.

1. User must have accounts in your local website's system ("Website of Record").
2. Users must have corresponding accounts for your Blackbaud NetCommunity website.

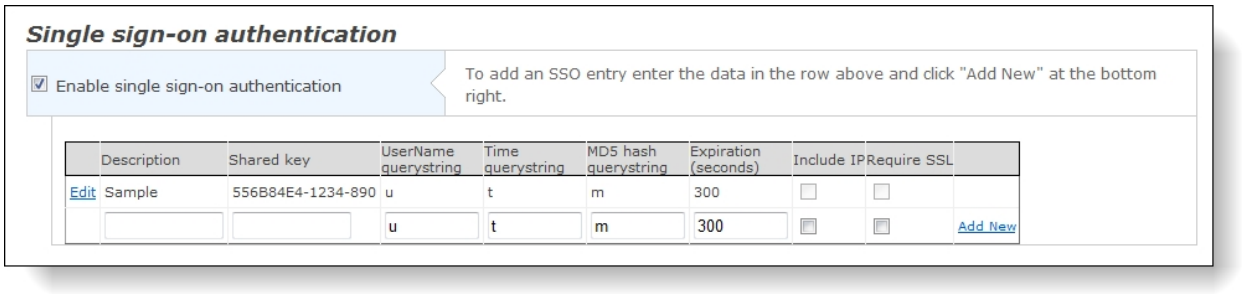
When users register for the first time on the "Website of Record," corresponding Blackbaud NetCommunity user accounts may not exist. To handle this scenario, Blackbaud NetCommunity provides a web service to create user accounts on the fly. For information about this web service, see Use Web Service Calls to Create Users on page 10.

Configure Incoming Single Sign-on

Blackbaud NetCommunity contains a built-in model for incoming single sign-on that you can use to authenticate requests that come in from other systems against Blackbaud NetCommunity user accounts. If you know the usernames for Blackbaud NetCommunity users, single sign-on allows you to grant them access to their accounts and bypass the manual login process.

This model uses the concept of a secret key known only to the Blackbaud NetCommunity website and the "Website of Record." The key is encrypted onto the URL with some additional information including an expiration timer. This timer value states how long the URL will be accepted from the time it was created.

To set up a calling system, you create a secret key in *Site & settings*. From *Administration*, click **Sites & settings** and then select a site in the site hierarchy on the left. On the Settings tab, scroll down to the **Single sign-on authentication** option and select **Enable single sign-on authentication**.



In the grid that appears under **Single sign-on authentication**, you create the shared key to set up a calling system.

Item	Description
Description	User-defined text to help you to keep track of this entry.
Shared Key	A secret key that only you and the incoming system know. This can be any string value, but a long combination of letters and numbers is a safe bet.
Querystring Parameter names	The variable names of the three parameters that the "Website of Record" passes to Blackbaud NetCommunity via the URL. For information about the parameters, see Implement Incoming Single Sign-on on page 7.
Expiration	The number of seconds after the <time> Querystring Parameter that the URL expires. Blackbaud recommends setting this value to 300 (five minutes). After the time elapses, the link expires and the page must be refreshed to get a new, valid link. This is an important security feature to prevent SSO links from being shared.
Include IP	Select this checkbox if the hash contains the known caller's IP address. This is optional but can be used as an additional security measure to ensure that the SSO link is only valid for a specific user's IP address.

For a procedure that walks you through these single sign-on authentication settings, see the Sites Settings section of the Administration Guide.

Recommended Usage

When you use incoming single sign-on for Blackbaud NetCommunity, the authentication of users is handled by the inbound query string URL rather than individual user passwords. If all access to restricted content in Blackbaud NetCommunity is granted in this fashion, then your website users don't need to remember Blackbaud NetCommunity credentials. Instead, all traffic is routed to your Blackbaud NetCommunity site from the "Website of Record" through custom hyperlinks.

In this scenario, you should limit access to certain parts and fields for users on your Blackbaud NetCommunity site.

- Do not let users register or log in directly through the User Login part.
- Do not let users edit their login credentials through the Change User ID/Password part.
- Do not include the **Username**, **Password**, or **Confirm Password** fields in the User Login section on the Profile Form part.

These parts and fields should be not be accessed by Blackbaud NetCommunity users because any changes to their Blackbaud NetCommunity usernames or passwords would leave the credentials out of sync with the "Website of Record." By removing the ability to register or update usernames and passwords in Blackbaud NetCommunity, you can ensure that users only register programmatically through the custom User Registration web handler.

Implement Incoming Single Sign-on

To implement incoming single sign-on, you must be able to generate specially formatted hyperlinks in your "Website of Record." This requires the ability to write and deploy custom code to the "Website of Record." Implementing single sign-on requires that your users click a specially formatted hyperlink, and the URL for the hyperlink must use the following format.

```
<a
href="http://yourNC.org/netcommunity/page.aspx?pid=123&u=<username>&t=<time>&m
=<md5 hash string>">
SSO Link Text
</a>
```

Note: Keep in mind that you must link to a Blackbaud NetCommunity page with a User Login part in order to honor the single sign-on link.

Make sure to replace each <tag> in the "href" property with the appropriate values outlined in this table.

Querystring Replacement	Description
<username>	The Blackbaud NetCommunity username to log the user in as.

Querystring Replacement	Description
<time>	The current epoch time. (The time since Jan. 1, 1970 in seconds)
<md5 hash string>	An MD5 hash of the hash string. The hash string can be one of two values depending on whether or not you want your IP in the hash. • Without IP: hashstr = sharedkey + <username> + <time>; • With IP: hashstr = sharedkey + <username> + <ip> + <time>; The sharedkey is specified in *Site & settings* in Blackbaud NetCommunity. The above only shows the value of the hash string. After the string is built you must apply an md5 hash to it.

Code Samples for Incoming Single Sign-on

The following code samples can be used to programmatically generate outbound URLs from your "Website of Record" that will authenticate users to NetCommunity.

Tip: Please keep in mind that these samples are a just examples to provide a good starting point for building your single sign-on solution. Please be aware that supporting and troubleshooting the code that you create is not part of your maintenance and support agreement with Blackbaud.

Sample Code for PHP

```
<?php
/* make_sso_url
* $sharedkey = the key that is specified in Sites & settings
* $username = the Blackbaud NetCommunity username to log in with
* $url = the URL of the page with the User Login part that users should be directed to
* $ip = the IP address the user should be from
* $includeip = whether or not to include ip address in the hash
 */


  function make_sso_url($sharedkey
                        ,$username
                        ,$url
                        ,$ip
                        ,$includeip)
{
    $time = time();


      if ($includeip) {
         $hashString = $sharedkey . $username . $ip . $time;
```

```
    } else {
      $hashString = $sharedkey . $username . $time;
    }


    return $url . "&t=" . $time . "&u=" . $username . "&m=" . md5($hashString);
}


?>
```

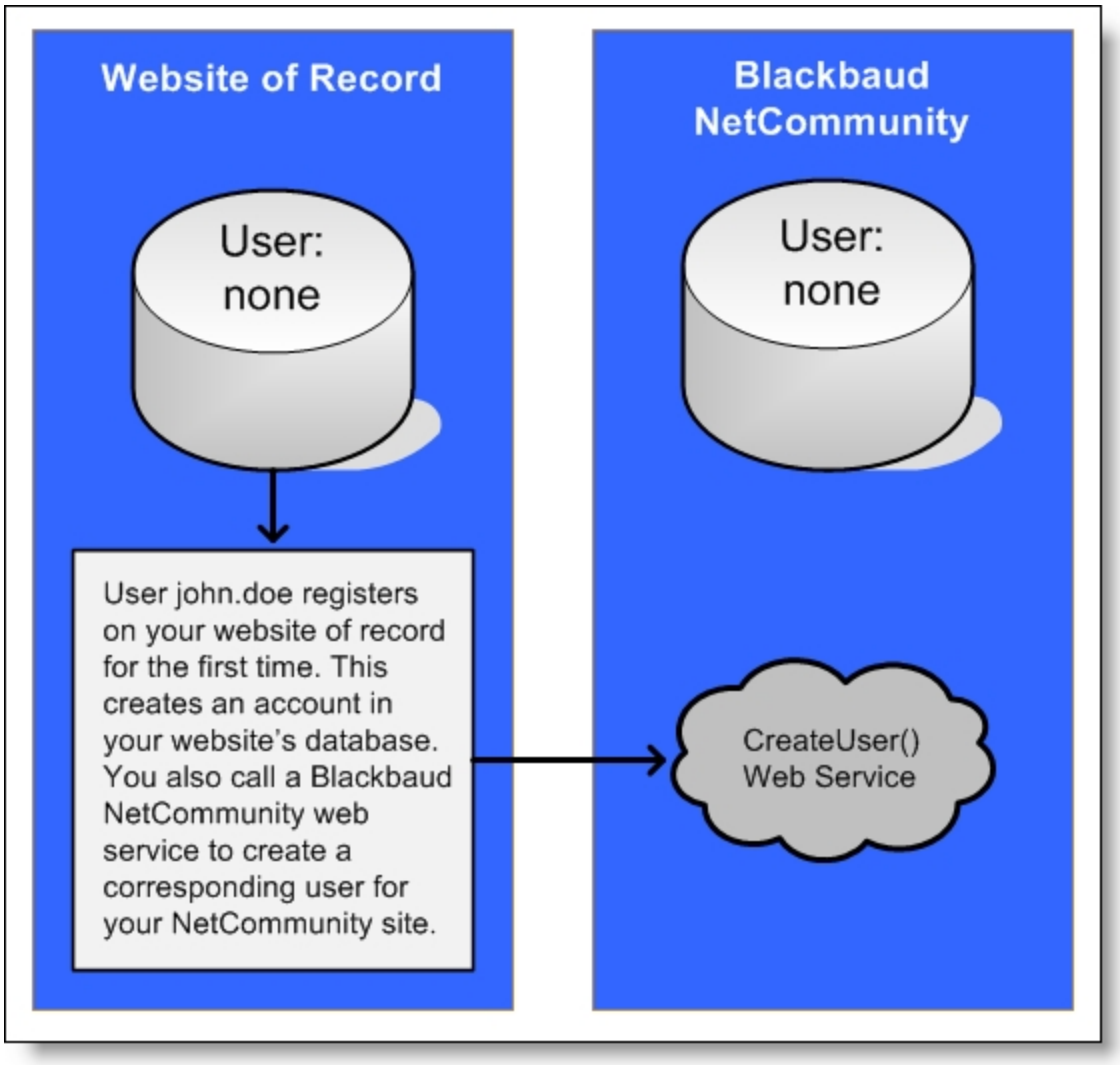
Sample Code for C#

```
    public String make_sso_url( String url
                              ,String sharedkey
                              ,String username
                              ,String ip
                              ,bool useip)
    {


        //variable declarations
       long    time;
        String  hash_string= "";
       String sso_url = "";



        //get the time
        long time = (DateTime.UtcNow.Ticks - EPOCH_BASE); //time in ticks
      time = time / 10000000; //time in seconds


        if (useip)
        {
          hash_string=sharedkey+username + ip + time; //hash string is ip is used
       }
        else
        {
          hash_string = sharedkey + username+time;      //hash string if ip is not used
      }


        //build the SSO URL
       sso_url =   url
                + "&t=" + time
                 + "&u=" + username
                + "&m=" + EncodeToMD5(hash_string);
```

```
        return sso_url;

    }

    public string getMD5(string originalString)
    {

        //Declarations
        String  md5String;
        Byte[]  originalBytes;
        Byte[]  encodedBytes;
        MD5     md5;            //note: MD5 is an abstract class



        //instantiate MD5CryptoServiceProvider
        md5 = new MD5CryptoServiceProvider();


        //convert originalString to bytes
        originalBytes = ASCIIEncoding.Default.GetBytes(originalString);


        //make the hash
        encodedBytes = md5.ComputeHash(originalBytes);


        //convert the bytes back to a string
        md5String = BitConverter.ToString(encodedBytes);


        // replace "-" with " "
        md5String = md5String.Replace("-", " ");


        return md5String;
    }
```

Use Web Service Calls to Create Users

Blackbaud NetCommunity is capable of creating users on the fly via a web service call. This is useful when users register for accounts on the "Website of Record" for the first time and need corresponding Blackbaud NetCommunity user accounts.



This web service is implemented via a Blackbaud NetCommunity custom handler and can be accessed on your Blackbaud NetCommunity installation via a URL like:

https://www.mydomain.org/components/custom.ashx?handler=Blackbaud.Web.Content.Core.Extensions.API.Users.RecordNewUserHandler,Blackbaud.Web.Content.Core

Note: This web service can only be accessed via https.

Using the web service is identical to a user registering through Blackbaud NetCommunity and generates a sign-up transaction unless the transaction is suppressed via a parameter. The web service returns the integer Blackbaud NetCommunity UserID of the created user. If a user with the supplied username already exists, the UserID of the existing user is returned. If the http output of the web service is anything other than an integer value, an error has occurred and will be returned as the http output of the web service.

Note: While the custom handler can create user accounts and generate signup transactions, it cannot assign new users to roles or create their constituent records in The Raiser's Edge. To assign users to roles in Blackbaud NetCommunity, you can edit their user accounts in *Users & security*. For

information, see the Users & Security Guide. To create constituent records, you must process the signup transactions in The Raiser's Edge. For information, see the Blackbaud NetCommunity and The Raiser's Edge Integration Guide.

User Registration Web Service Parameters

The web service expects to find the following parameters in the http POST. Parameter names are case sensitive.

Warning: For the custom handler to function properly, you must have Blackbaud NetCommunity 6.41 patch 26 or a later release.

Parameter Name	Description
AdminUsername	Required. The web service requires a valid username for a Blackbaud NetCommunity user account with the Supervisor rights that are necessary to create new user accounts.
AdminPassword	Required. The web service requires a valid password for the Blackbaud NetCommunity user account that you specify in the **AdminUsername** parameter.
FirstName	Required. The first name of the target user.
LastName	Required. The last name of the target user.
EmailAddress	Required. The email address of the target user.
Password	Required. The password for the user. Blackbaud recommends a random string of at least eight characters. The user never has to use this password if Blackbaud NetCommunity is accessed via SSO.
ConfirmPassword	Required. This must match the password parameter.
ConstituentId	Optional. A Raiser's Edge constituent ID that is used to create a link between the sign-up transaction and a constituent record.
SkipSignupTransaction	Optional. If set to value "true", the sign-up transaction is not created.
Username	Required. The Blackbaud NetCommunity username of the target user. This should be a value that can be recreated from known user information in the "Website of Record" so that the proper Blackbaud NetCommunity username can later be inserted into SSO links.

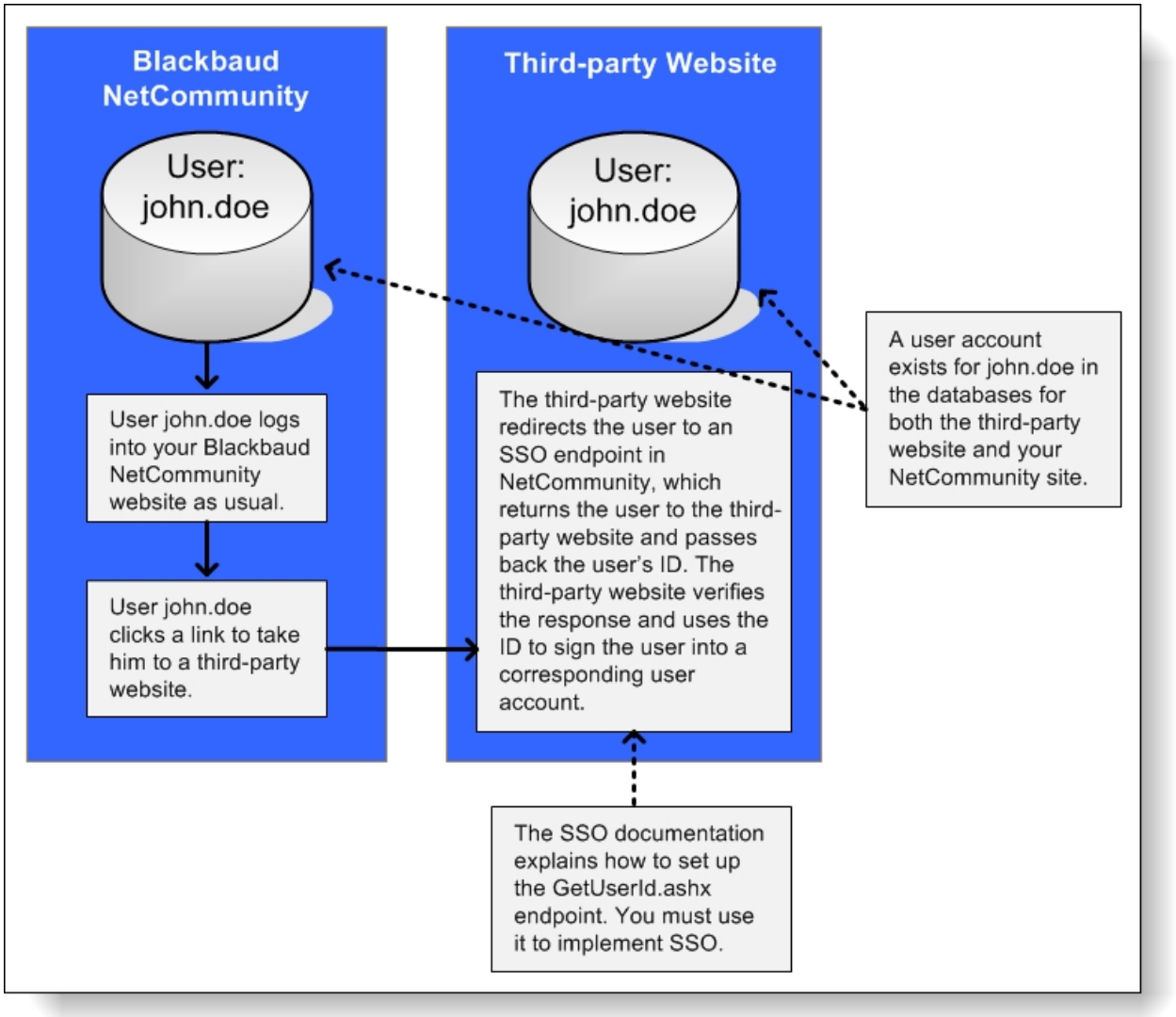
Note: The required fields are sent to The Raiser's Edge via a sign-up request just like all standard Blackbaud NetCommunity user registrations. However, the new user registration email is not sent.

Outgoing Single Sign-on

The outgoing single sign-on (SSO) feature in Blackbaud NetCommunity allows website users to transparently sign in to their accounts on an external system without re-entering their login information. This outgoing SSO functionality allows users who are authenticated via Blackbaud NetCommunity to be transparently signed in to corresponding accounts on a third-party site.

To implement outgoing single sign-on between Blackbaud NetCommunity and an external system, you must be able to access the GetUserID.ashx endpoint in the Blackbaud NetCommunity API. To determine user IDs, the external system accesses the endpoint with a return URL in the query string. The endpoint redirects website users to the return URL and uses several query string parameters to securely determine user IDs. For information about the endpoint, see GetUserId.ashx Endpoint on page 14.

The following illustration outlines how the outgoing single sign-on option works in Blackbaud NetCommunity.



GetUserId.ashx Endpoint

The GetUserId.ashx endpoint provides the ability for an external system to determine the Blackbaud NetCommunity user ID of the end user. It can optionally enforce that the end user login to Blackbaud NetCommunity.

This endpoint should be accessed with a return URL in the query string. The end user will be redirected to the return URL with several query string parameters appended to securely determine the current user ID. If the user is not logged in and the optional require login parameter isn't supplied, then no query string information will be added to the redirect URL.

The signature parameter in the response can be used to verify the authenticity of the request, so that a user cannot browse directly to your redirect URL with a forged userid parameter. The signature built and validated with the GetUserId.ashx private key that is specified in Blackbaud NetCommunity on the API tab in *Sites & settings*.

Usage:

- Request:
  - Parameters
    - Redirect – specifies the redirect URL.
    - RequireLogin (optional) – a value of 1 indicates that the user should be forced to login before the redirect.
  - Example:
    - http://www.yourNetCommunitySite.org/components/GetUserID.ashx?redirect=http://www.thirdpartydomain.com
    - http://www.yourNetCommunitySite.org/components/GetUserID.ashx?redirect=http://www.thirdpartydomain.com&requireLogin=1
- Response:
  - Parameters
    - UserId – the BBIS user ID of the current logged in user.
    - TS – time stamp of when the redirect was created. This is created using this string format: http://msdn.microsoft.com/en-us/library/az4se3k1.aspx#Roundtrip
    - Sig – this is a signature to verify the authenticity of the user id. It is created by taking an MD5 hash of the userId, time stamp, and private key appended together in that order.
  - Example:
    - http://www.thirdpartydomain.com/?userid=1&ts=2011-05-27T09%3a20%3a41.5068885-04%3a00&sig=83197991befbfd4af31979ba9928622b
    - http://www.thirdpartydomain.com – Not logged in.

Note: It is the responsibility of the third-party domain to actually parse the GetUserID.ashx return URL, validate the hashed site key in the query string, and actually use the returned userID to log that user into their site. Implementation details will vary based on the platform of the third-party domain.