Synology®

Synology SSO Server

Development Guide

Synology®

Synology Inc.
3F-3, No. 106, Chang-An W. Rd. Taipei 103, Taiwan

Table of Contents

Chapter 1

Introduction

Synology DSM SSO Server is based on the OAuth 2 protocol. We provide the JavaScript SDK for 3rd party development. SSO Server JavaScript SDK script will be installed automatically after SSO Server installation.

Chapter 2

Javascript SDK

DSM JavaScript SDK Script Location

http://DSM_IP_OR_HOSTNAME:5000/webman/sso/synoSSO-1.0.0.js

Usage

Initialization

SYNOSSO.init

SYNOSSO.init is used to initialize SYNOSSO SDK. You need to call SYNOSSO.init before calling any other SYNOSSO APIs.

Function parameters of SSOSYNO.init:

Key	Value	Description
oauthserver_url	**string**	The URL of the DSM where SSO Server is installed.
app_id	**string**	APP ID registered on the DSM SSO Server
redirect_uri	**string**	Redirect URI registered on the DSM SSO Server.
callback	**Javascript function object**	User defined callback for handling login query/login response.
domain_name(optional)	**string**	Windows AD domain name of SSO client. Ex: "MYDOMAIN.COM"
ldap_baseDN(optional)	**string**	LDAP baseDN of SSO client. Ex: "dc=myldap,dc=com"

***Directory service related options are for directory service checking. If one of these options is provided, SSO Server will validate if this directory service is the same as DSM that SSO Server belongs to.**

Example:

```
SYNOSSO.init({
oauthserver_url: 'http://10.13.20.131:5000',
  app_id: '153fcb35b01571b49cb0adca3a4bda40',
  redirect_uri: 'http://10.13.20.130/relay.html', //redirect url have to be the same as the
one registered in SSO server, and can be a plain text html file.
  callback: authCallback
});
```

Authentication

SYNOSSO.login();

After calling SYNOSSO.login, a login popup window containing a dialog for SSO will appear. SYNOSSO.login has no arguments and will call the callback registered in SYNOSSO.init after the user logs in successfully.

Example:

SYNOSSO.login();

Response:

Response of Callback registered in SYNOSSO.init():

Key	Value	Description
Status	**String: "login"/"not_login"/ERR_STRING**	Show status of this user on SSO Server.

Key	Value	Description
Access_token	**string**	Access token returned from SSO Server after this user logs in successfully.

If the user already login SSO Server

```
response:{
  status: 'login',
  access_token: 'ABCDE'
}
```

If the user didn't login SSO Server

```
response:{
  status: 'not_login'
}
```

If any unexpected error occurred.

```
response:{
  status: 'ERR_STRING'
}
```

*** For ERR_STRING, please refer to Chapter 6 for more details.**

Logout

```
SYNOSSO.logout(function(){
    //do something after logout.
});
```

Function parameters of SSOSYNO.logout:

Key	Value	Description
callback	**Javascript function**	The callback which will be called after the user logs out from SSO Server.

SYNOSSO.logout has a callback which will be called after user logs out from SSO Server.

- Before a user logs out from your application, call **SYNOSSO.logout**, and this method will log out this user from SSO Server.
- **SYNOSSO.init** must be called before **SYNOSSO.logout**.
- **SYNOSSO.logout** only logs out the user from SSO Server and will not affect login status of the user in others applications.

Response of Callback of SYNOSSO.logout has no arguments.

Chapter 3

Manual Flow

Step1: Bring the user to *http://[DSM Oauth Server:5000]/webman/sso/SSOOauth.cgi* with the following query string parameters:

- app_id : APP ID registered on DSM SSO Server.
- redirect_uri : Redirect URI registered on DSM SSO Server.
- scope: Currently, SSO server only provide “user_id” scope which means limited user information for Single-Sign On.
- state(optional) : Use to protect CSRF.

Then the login window will show up, waiting for the user to input username/password.

Ex:
SSO Server: 10.13.20.254
SSO Client: 10.13.22.128

http://10.13.20.254:5000/webman/sso/SSOOauth.cgi?app_id=a5a78d55b7d30dab1b3067d26bc4

9e49&scope=user_id&redirect_uri=http://10.13.22.128/sso_redirect_relay.html

Step2: User logs in to SSO Server

Step3: After logging in successfully, the user will be redirected back to the redirect URI which this app registered on SSO Server with following hash values:

- access_token: The access token which will be used to exchange user information.
- State(optional): If you provide the state at Step1, the exact same state will be returned.

Ex:
http://10.13.22.128/sso_redirect_relay.html#access_token=58322f3eaaG7t69030edH2bcdee08brWc6250eba&state=fabc21cf

Chapter 4

Exchange User Information

To exchange for user's information

1 You need to use an accesstoken to get user_id and user_name

2 Go to endpoing: *http://[DSMOauthServer:5000]/webman/sso/SSOAccessToken.cgi* with these query string parameters:

- action: "exchange"
- access_token: "ABCDE"
- app_id: "asfsf sdfsdf3e"

Example:

curl http://[DSMOauthServer:5000]/webman/sso/SSOAccessToken.cgi?action="exchange"&access_token="ABCDE"&app_id="asfsffsdfsdf3e"

Response:

If the token is correct:

```
{
        success: true,
        data:{
            user_id:1024,
            user_name:john
    }
}
```

If any unexpected errors occurred:

```
{
    success: false,
       error: 'ERR_STRING'
}
```

Example Code

Javascript SDK Examples

Fontpage.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>Test App 1</title>
        <meta name="viewport" content="width=device-width,
initial-scale=1.0">
        <meta name="description" content="">
        <meta name="author" content="">
    </head>
    <body>
        <div class="container">
         <div class="form-signin">
          <h1 class="form-signin-heading">Test App 1</h1>
          <h2 class="form-signin-heading">Please sign in via Synology
Oauth </h2>
          <button id="login-button">SSO Login</button>
        </div>

        </div>
    </body>

    <script type="text/javascript" src="jquery-2.1.1.min.js"></script>
    <script type="text/javascript"
src="http://10.13.20.254:5000/webman/sso/synoSSO-1.0.0.js"></script>
  <script>
        //SYNOSSO Javascript SDK don't depend on jQuery!
        //SSO Server: 10.13.20.254
        //SSO Client: 10.13.20.130
        $ (function(){
              SYNOSSO.init({
                  oauthserver_url: 'http://10.13.20.254:5000',
                  app_id: '153fcb35b01571b49cb0adca3a4bda40',
                  redirect_uri: 'http://10.13.20.130/ssorelay.html',
//redirect URI have to be the same as the one registered in SSO server, and
should be a plain text html file
                  callback: authCallback
                  })
                    function authCallback(response){
                  console.log("client side");
                  if('not_login' === response.status) { //user not
login
                            console.log (response.status);
                  } else if('login' === response.status) {
                            console.log (response.status);
                            console.log (response.access_token);
```

```
                        alert("access token: "+
response.access_token);
                        $.ajax ({ url : '/login_backend.php' ,
                                cache: false,
                                type: 'GET',
                                data:{
                                        accesstoken:
response.access_token
                                },
                                error: function(xhr){
                                        alert("ajax error");
                                        //deal with errors
                                },
                                success: function(response){
                                        alert("success");
                                        //deal with success
                                }
                        });
                } else {
                        alert("error");
                        //deal with errors;
                }
        }
        var login_button = document.getElementById("login-button");
        login_button.addEventListener('click' , SYNOSSO.login);
    })()
  </script>
</html>
```

Login_backend.php

```php
<?php
session_start();
$accesstoken = $_GET['accesstoken'];

function httpGet ($url)
{
	$ch = curl_init();

	curl_setopt($ch, CURLOPT_URL, $url);
	curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
	curl_setopt($ch, CURLOPT_HEADER, false);
	curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);//for testing,
ignore checking CA
	curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
	$output=curl_exec($ch);

	curl_close($ch);
	return $output;
}
//SSO Server: 10.13.20.254:5000
$url_str =
"http://10.13.20.254:5000/webman/sso/SSOAccessToken.cgi?action=exchange&access_token=".$accesstoken
;

$resp = httpGet($url_str);
$json_resp = json_decode($resp, true);
if($json_resp["success"] == true){
	$userid = $json_resp["data"]["user_id"];
	$_SESSION["user_id"] = $userid;
	//login success
} else {
	//not login, redirect to frontpage.html
}
?>
```

Error String

ERR_STRING

- server_error - SSO server error.
- parameter_error - Parameter error when SYNOSSO.init.
- invalid_app_id - APP_ID error.
- invalid_redirect_uri - Redirect URI error.
- invalid_directory_service - Different directory service between SYNOSSO.init and DSM SSO Server.
- invalid_token - Invalid SSO access token.
- unknown_error - Other unexpected errors.