

## Please Come In: Social Login for SAS® Web Applications

Michael Dixon, Selerity

### ABSTRACT

For customers providing SAS reporting to the public the ability to use a Social login opens several possibilities for providing richer services. Instead of everybody using a generic Guest access and being limited to a common subset of reports or other functionality, previously unknown users can seamlessly login and access SAS Web content while SAS Administrators can continue to apply best-practice security. This paper focuses on integrating Google Sign-In, Microsoft Account Sign-In and Facebook Sign-In as alternative methods to log in from the SAS Logon Manager, as well as registering any new users in the SAS Metadata automatically.

### INTRODUCTION

The use of social authentication has become ubiquitous in modern application development. Research consistently shows that end users prefer the use of social identities vs. traditional email / registration approaches to user authentication. Of the 90%+ users that prefer social authentication, over 97% of those users either prefer Facebook or Google as their social authentication provider (login radius, 2016).

The use of social authentication in modern application design improves usability, democratizes access, utilises leading security design principles including OAuth2, multi-factor authentication and encryption, and provides an integration point between disparate applications and platforms including Cloud. In the context of SAS Web Application access, allowing users to authenticate using their social login credentials has benefits for both the user and the SAS environment owner.

Benefits for the User	Benefits for the SAS Environment Owner
No effort to register/request initial access	No need to create new users
Familiar login/access process	No user administration overhead, e.g. resetting passwords.
Ability for users to personalize their SAS web application preferences	Encourages more interaction due to the ability for the user to personalize their experience
Control over their own login, e.g. User can cancel their authority for the SAS environment to use their social credentials	As social login providers require a valid email address, the user object in SAS will always contain a valid email address.

**Table 1. Social Login Benefits**

Many social login services utilize OAuth (Open Authorization) which is an open standard for token-based authentication and authorization on the Internet. "OAuth, which is pronounced "oh-auth," allows an end user's account information to be used by third-party services, such as Facebook, without exposing the user's password" (Tech Target, 2017).

The steps presented in this paper have been tested using SAS 9.4M3 and SAS 9.4M4 on both Windows and Linux.

### PREPARATION

The first step in enabling social login is to register your SAS environment with the social login provider(s) of your choice. The process is similar for most providers, and examples of the process for registering with Google, Microsoft and Facebook are below.

Once you have successfully registered with a social login provider you will receive a Client ID, which is then used in the OAuth JavaScript SDK to allow your SAS environment to use that provider.

## REGISTERING WITH GOOGLE

Open the Google+ API Dashboard of the Google Cloud Platform™ by navigating to <https://console.cloud.google.com/apis/api/plus/overview> in your browser and then logging in using an existing Google Account. Click the **Create Project** button at the top right of the screen and follow the prompts to create a new Project. Once you have a new Project, click the **Enable** button to turn on the Google+ API, allowing social login for Google.

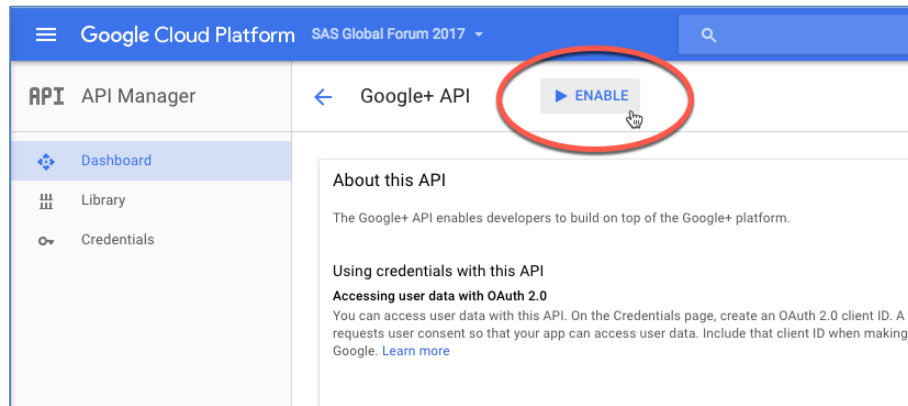


Figure 1. Enabling Google+ API

Once you have enabled the Google+ API you will need to create some OAuth credentials by clicking the **Credentials** item on the left, followed by clicking the **Create Credentials** button and choosing **OAuth client ID**. You will then be prompted to configure the consent screen. Fill in the details as required.

A screenshot of the Google OAuth consent screen configuration form. The form is titled 'Credentials' and has a sub-header 'OAuth consent screen'. It contains several input fields: 'Email address' (with a dropdown menu), 'Product name shown to users' (with a text input), 'Homepage URL (Optional)' (with a text input), 'Product logo URL (Optional)' (with a text input), 'Privacy policy URL' (with a text input), and 'Terms of service URL' (with a text input). There is also a section for 'Domain verification' with a text input. A 'Save' button is at the bottom left, and a 'Cancel' button is at the bottom right. A small icon of a laptop and a smartphone is shown on the right side of the form.

Figure 2. Google OAuth Consent Screen

Click **Save** after you have entered the details for the OAuth consent screen and you will then be asked what type of application you are creating. Select **Web application** and then complete the details presented. The two URIs you need to enter are:

- Authorized JavaScript origins
  - This is the plain URL of your SAS environment, e.g. <https://sgf2017.seleritycloud.com.au>. If your web applications are running on a non-standard port make sure that you include the `:<port>` component.
- Authorized redirect URIs
  - This is the URL that equates to the `/SASLogon/login` location of your SAS environment

**Create client ID**

**Application type**

- ☒ Web application
- ☐ Android [Learn more](#)
- ☐ Chrome App [Learn more](#)
- ☐ iOS [Learn more](#)
- ☐ PlayStation 4
- ☐ Other

**Name**

SAS Social Login Example

**Restrictions**  
Enter JavaScript origins, redirect URIs, or both

**Authorized JavaScript origins**  
For use with requests from a browser. This is the origin URI of the client application. It can't contain a wildcard (http://\*.example.com) or a path (http://example.com/subdir). If you're using a nonstandard port, you must include it in the origin URI.

https://sgf2017.seleritycloud.com.au  
http://www.example.com

**Authorized redirect URIs**  
For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorization code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

https://sgf2017.seleritycloud.com.au/SASLogon/login  
http://www.example.com/oauth2callback

**Create** **Cancel**

**Figure 3. Google Create Client ID**

Clicking **Create** will present you with your OAuth credentials for Google. Make note of the Client ID string as you will need to enter this value into the OAuth JavaScript SDK.

## REGISTERING WITH MICROSOFT

Navigate to the Microsoft Application Registration Portal at <https://apps.dev.microsoft.com> and sign in with a Windows Live/Microsoft Account.

Click the **Add App** button for the **Converged applications** section. You will be prompted to enter a name for your Application. Enter a valid name and click **Create application**.

Take note of the Application Id shown on the Registration screen. This is used as the Client ID in the JavaScript OAuth SDK.

Microsoft Application Registration Portal Provide Feedback Michael

## SGF 2017 Registration

This application will be registered in the Azure Active Directory instance used to manage your michael.dixon@selerity.com.au account.

**Properties** [Learn More](#)

**Name**

SGF 2017

**Application Id**

ea-b1-5-4-8-8-f-f6

**Figure 4. Microsoft Application Id**

On the Registration screen that appears, click the **Add Platform** button and then choose **Web** as the platform. You will be prompted to enter a Redirect URI, which equates to the `/SASLogon/login` URL of your SAS environment.

**Figure 5. Microsoft Redirect URI**

Click **Save** at the bottom of the Registration Page to complete the process for Microsoft.

## REGISTERING WITH FACEBOOK

Navigate to the Facebook developers site at <https://developers.facebook.com/apps> and login with your Facebook credentials. Once logged in click the **+ Add a New App** button at the top right. Enter a Display Name, Contact Email and Category for your application and then click the **Create App ID** button.

**Figure 6. Facebook Create a New App ID**

The Product Setup screen appears next and you should click on the **Get Started** button next to the Facebook Login item. On the Client OAuth Settings page enter the OAuth redirect URI which equates to the `/SASLogon/login` URL of your SAS environment and click the **Save Changes** button at the bottom right.

**Figure 7. Facebook Client OAuth Settings**

Click on the **Settings** item under the **Dashboard** item on the left and then click the **+ Add Platform** button.

The screenshot shows the 'Facebook Add Platform' form in the SAS Global Forum 2017 dashboard. The left sidebar has a 'Settings' item with a red circle containing the number '1'. The main form area has a header with 'APP ID: 1650230675280609' and a 'View Analytics' link. The form fields are: App ID (1650230675280609), App Secret (masked with dots and a 'Show' button), Display Name (SAS Global Forum 2017), Namespace (empty), App Domains (empty), Contact Email (michael.dixon@selerity.com.au), Privacy Policy URL (https://www.selerity.com.au/privacy-policy/), Terms of Service URL (https://www.selerity.com.au/terms-of-service/), App Icon (1024 x 1024, with a placeholder image), and Category (Productivity). At the bottom right of the form is a '+ Add Platform' button with a red circle containing the number '2'.

**Figure 8. Facebook Add Platform**

Choose **Website** from the list of platform types and then enter the plain URL of your SAS environment. If your SAS environment URL is on a non-standard port make sure that you add :<port> to the URL.

Facebook is slightly different to Google and Microsoft in that by default it will be in *development mode* and only the person that created the Facebook App can login using social login. To make social login available to anyone you must click the **App Review** item on the left and then click the Make public toggle button.

Make note of the App ID value for your Facebook App as this will be used as the Client ID in the JavaScript OAuth SDK.

## CONFIGURING SAS FOR SOCIAL LOGIN

Configuring SAS to allow social logins for web application access involves some updates to Metadata, modifying the SAS Logon Manager web application and installing an OAuth SDK within the SAS Logon Manager. The steps below show how this process is achieved for a Visual Analytics environment where members of the public are encouraged to use their social login to access several Visual Analytics Reports, but can easily be expanded upon for other web applications and functionality.

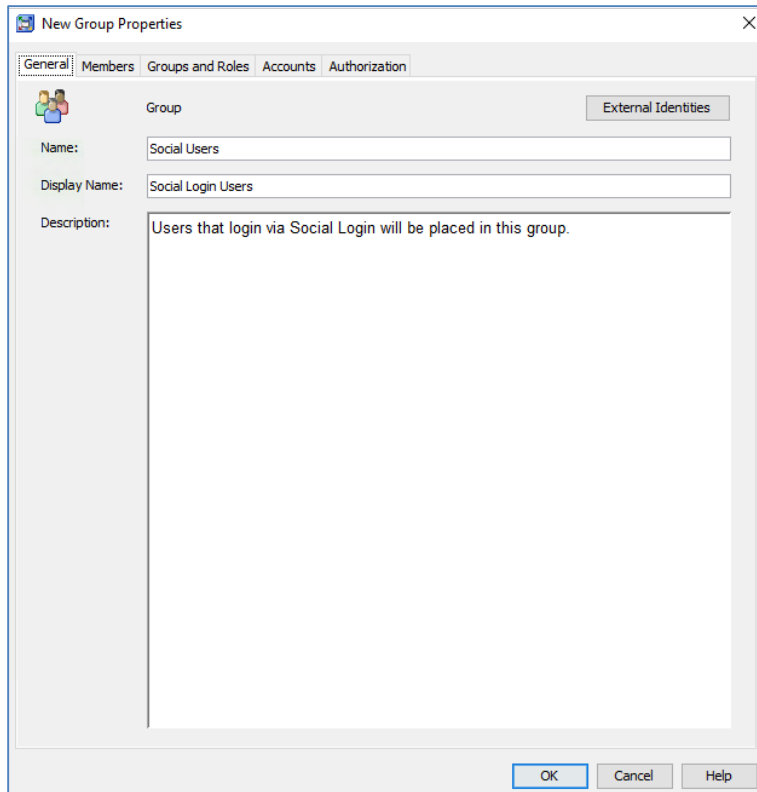
### SAS METADATA

The changes to Metadata are all done using the SAS Management Console. Start the SAS Management Console and log in using an unrestricted user account such as sasadm@saspw.

#### Add New Group for Social Logins

All users that access your environment using a social login will be automatically added to a specific group in Metadata. You will need to create this group before enabling social login. This can be done in SAS Management Console by right-clicking on the User Manager plugin in Management Console and choosing **New -> Group**. Enter the following properties for the new group:

- Name: Social Users
- Display Name: Social Login Users
- Description: Users that login via social login will be placed in this group.



**Figure 9. New SAS Metadata Group for Social Login Users**

Click on the **Groups and Roles** tab of the new group and add the **Visual Analytics: Report Viewing** role. This will restrict social login users to only being able to view existing reports in Visual Analytics. You may choose other Roles or Groups to allow your social login users to access the functionality you desire.

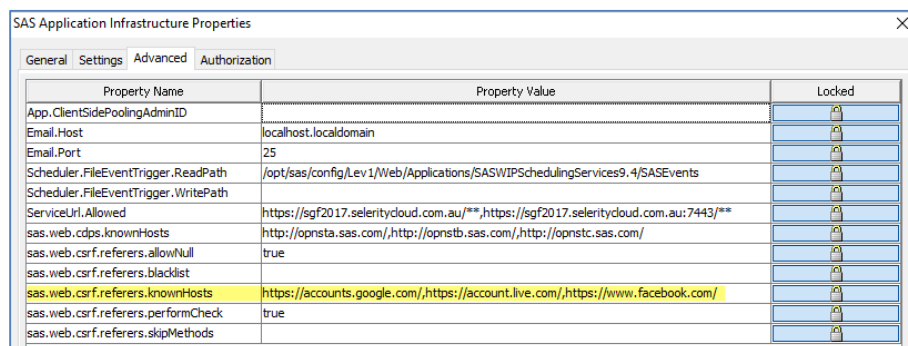
### Update the Allowed Referrers Setting

To allow a social login provider to *call back* to your SAS environment with details about who is trying to login you will need to modify the `sas.web.csrf.referers.knownHosts` setting of your environment. In SAS Management Console, expand **Application Management** and **Configuration Manager**. Right-click on the **SAS Application Infrastructure** object and select **Properties**. On the **Advanced** tab modify or add the property called `sas.web.csrf.referers.knownHosts`. This property contains a comma separated list of sites allowed to link into your environment and is used by SAS to protect against Cross-Site Request Forgery (CSRF) attacks. Depending on which provider(s) you want to use, add one or more of the following sites to this property:

Provider	Site to add
Google	<a href="https://accounts.google.com/">https://accounts.google.com/</a>
Microsoft	<a href="https://account.live.com/">https://account.live.com/</a>
Facebook	<a href="https://www.facebook.com/">https://www.facebook.com/</a>

**Table 2. Referrer Sites for `sas.web.csrf.referers.knownHosts`**

An example of what this property looks like when all providers are added is shown below.

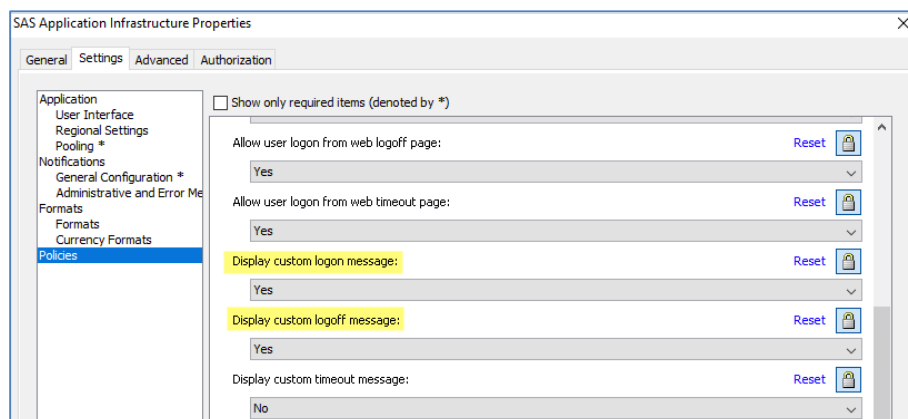


Property Name	Property Value	Locked
App.ClientSidePoolingAdminID		
Email.Host	localhost.localdomain	
Email.Port	25	
Scheduler.FileEventTrigger.ReadPath	/opt/sas/config/Lev1/Web/Applications/SASWIPSSchedulingServices9.4/SASEvents	
Scheduler.FileEventTrigger.WritePath		
ServiceUrl.Allowed	https://sgf2017.seleritycloud.com.au/**;https://sgf2017.seleritycloud.com.au:7443/**	
sas.web.cdps.knownHosts	http://opnsta.sas.com/http://opnsta.sas.com/http://opnsta.sas.com/	
sas.web.csrf.referers.allowNull	true	
sas.web.csrf.referers.blacklist		
<b>sas.web.csrf.referers.knownHosts</b>	<b>https://accounts.google.com/https://account.live.com/https://www.facebook.com/</b>	
sas.web.csrf.referers.performCheck	true	
sas.web.csrf.referers.skipMethods		

**Figure 10. Advanced Properties for SAS Application Infrastructure**

## Enable Custom Logon and Logoff Messages

Still within the SAS Application Infrastructure properties, go to the **General** tab and click on the **Policies** entry listed on the left. Locate the **Display custom logon message** and **Display custom logoff message** items and change both to **Yes**.



SAS Application Infrastructure Properties

General Settings Advanced Authorization

Application

- User Interface
- Regional Settings
- Pooling \*
- Notifications
- General Configuration \*
- Administrative and Error Me
- Formats
- Formats
- Currency Formats
- Policies**

☐ Show only required items (denoted by \*)

Allow user logon from web logoff page: Yes Reset

Allow user logon from web timeout page: Yes Reset

**Display custom logon message:** Yes Reset

**Display custom logoff message:** Yes Reset

Display custom timeout message: No Reset

**Figure 11. Enabling Custom Logon and Logoff Messages**

## SAS LOGON MANAGER WEB APPLICATION

The SAS Logon Manager handles all authentication services for accessing SAS web applications, providing a centralized location where we can modify or add extra authentication methods. For the method of enabling social login in this paper you will be carrying out tasks related to **Web Authentication** and **SAS Web Server Authentication** as found in the **SAS 9.4 Intelligence Platform: Middle-Tier Administration Guide, Fourth Edition** under the **Advanced Topics -> Enterprise Integration** chapter (SAS Institute Inc. 2016a, 2016b).

When modifying a SAS web application there are two places you can make changes, one that is *Live* and whose effect can be seen either immediately or after restarting the appropriate Web Application Server and the second being the *Source* that is used when rebuilding a web application and whose changes are only seen after rebuilding/redeploying. The steps below identify both locations, and you can make changes to either or both. If you choose to make changes to the *Live* location you only need to restart the SAS Web Application Server to see the effects. If you make changes only to the *Source* location you will need to use the SAS Deployment Manager to rebuild and then redeploy the SAS Web Infrastructure Platform before the changes will be seen.

**Note:** if you only make changes to the *Live* location then your changes will be lost if any hot fixes or maintenance are applied that affect the Web Infrastructure Platform.

## Configuring the Security Module for SAS Web Application Server

This modification will allow the SAS Logon Manager web application to accept the user name being returned in the HTTP header by the OAuth SDK.

Type	File Location
Source	SASHome/SASWebInfrastructurePlatform/9.4/Static/wars/sas.svcs.logon/META-INF/context.xml
Live	Levl/Web/WebAppServer/SASServer1_1/conf/Catalina/localhost/SASLogon.xml

**Table 3. Files for Configuring Security Module**

Add the following Valve entry to SASLogon.xml and/or context.xml:

```
<Valve className =  
"com.sas.vfabrictcsvr.authenticator.PrincipalFromRequestHeadersValve"/>
```

## Auto-Provisioning User Accounts

Starting with the third maintenance release of SAS 9.4, the ability to automatically provision new user accounts into the SAS Metadata was introduced. If the environment was configured for Web Authentication and the Web authentication mechanism has confirmed that users' credentials are valid then the system can automatically create a SAS Metadata User for that confirmed identity.

Make the following modifications to the deployment descriptor of the Logon Manager:

Type	File Location
Source	SASHome/SASWebInfrastructurePlatform/9.4/Configurable/wars/sas.svcs.logon/WEB-INF/web.xml.orig
Live	Levl/Web/WebAppServer/SASServer1_1/sas_webapps/sas.svcs.logon.war/WEB-INF/web.xml

**Table 4. Files for Modifying the Logon Manager Deployment Descriptor**

Add the following code to the **<filter>** section of web.xml.orig and/or web.xml:

```
<filter>  
  <filter-name>autoProvisioningFilter</filter-name>  
  <filter-class>  
org.springframework.web.filter.DelegatingFilterProxy</filter-class>  
</filter>
```

Add the following code to the **<filter-mapping>** section of web.xml.orig and/or web.xml:

```
<filter-mapping>  
  <filter-name>autoProvisioningFilter</filter-name>  
  <url-pattern>/login</url-pattern>  
</filter-mapping>
```

Make the following modifications to the spring security filter of the Logon Manager:

Type	File Location
Source	SASHome/SASWebInfrastructurePlatform/9.4/Static/wars/sas.svcs.logon/WEB-INF/spring-configuration/filters.xml
Live	Levl/Web/WebAppServer/SASServer1_1/sas_webapps/sas.svcs.logon.war/WEB-INF/spring-configuration/filters.xml

**Table 5. Files for Modifying the Logon Manager Spring Security Filter**

Add the following code before the directAuthenticationFilter entry:

```
<bean id="autoProvisioningFilter"  
class="com.sas.svcs.security.authentication.filters.UserAutoProvisioningFilter"
```



```
p:urlGenerator-ref="svcs.urlGenerator"
p:adminUser="@{server.admin.userid}"
p:adminPassword="@{server.admin.password}"
p:groupName="Social Users" />
```

The value for `p:groupName` in the above code should reflect the Name of the Metadata Group you have created that will contain new social login users.

## Install JavaScript OAuth SDK

The JavaScript OAuth SDK we are using is the **hello.js** SDK created by Andrew Dodson and found at <http://adodson.com/hello.js/> and on GitHub at <https://github.com/MrSwitch/hello.js>. Go to <https://adodson.com/hello.js/#install> and download the HelloJS JavaScript library. Copy the file you downloaded to the *Source* and/or *Live* locations listed below and rename the file to `hello.js`.

Type	File Location
Source	SASHome/SASWebInfrastructurePlatform/9.4/Static/wars/sas.svcs.logon/js/
Live	Levl/Web/WebAppServer/SASServer1_1/sas_webapps/sas.svcs.logon.war/js/

**Table 6. Location for hello.js JavaScript Library**

In the same locations, create a new file called `hello_login.js` and copy the following JavaScript code into this new file:

```
hello.on('auth.login', function(auth) {
    hello(auth.network).api('me').then(function(r) {
        xmlhttp=new XMLHttpRequest();
        var formData = new FormData();
        xmlhttp.open("POST", window.location.href);
        xmlhttp.onload = function () {
            if (xmlhttp.status == 200) {
                console.log('Authentication Successful');
                location.reload();
            }
        };
        xmlhttp.setRequestHeader("X-Remote-User", r.email);
        xmlhttp.send(formData);
    });
});
```

Also in this same location, create a file called `providers.js`. This file will contain the Client IDs obtained when registering with the social login providers. Use the following template to populate this file, replacing the Client ID placeholders as appropriate:

```
hello.init({
    google: '<YOUR-GOOGLE-CLIENT-ID>',
    windows: '<YOUR-MICROSOFT-CLIENT-ID>',
    facebook: '<YOUR-FACEBOOK-CLIENT-ID>'
}, {redirect_uri: '/SASLogon/login'});
```

## Customize the Logon and Logoff Pages

The final step is to use the custom Logon and Logoff messages to load the OAuth SDK and present social login buttons to end users.

Type	File Location
Source	SASHome/SASWebInfrastructurePlatform/9.4/Static/wars/sas.svcs.logon/WEB-INF/view/jsp/default/ui/logon_custom.jsp
Live	Levl/Web/WebAppServer/SASServer1_1/sas_webapps/sas.svcs.logon.war/WEB-INF/view/jsp/default/ui/logon_custom.jsp

**Table 7. Custom Logon Message File**

Replace the contents of the `logon_custom.jsp` file with the following:

```
<script src="/SASLogon/js/hello.js"></script>
<button class="btn-submit" style="width: 300px;"
onclick="hello('google').login({scope: 'email'})">
Sign in with Google</button><br/>
<button class="btn-submit" style="width: 300px;"
onclick="hello('windows').login({scope: 'email'})">
Sign in with Microsoft</button><br/>
<button class="btn-submit" style="width: 300px;"
onclick="hello('facebook').login({scope: 'email'})">
Sign in with Facebook</button><br/>
<script src="/SASLogon/js/hello_login.js"></script>
<script src="/SASLogon/js/providers.js"></script>
```

The button HTML code in bold above can be tweaked to give the logon buttons different styles if you wish.

Type	File Location
Source	SASHome/SASWebInfrastructurePlatform/9.4/Static/wars/sas.svcs.logon/WEB-INF/view/jsp/default/ui/logoff_custom.jsp
Live	Levl/Web/WebAppServer/SASServer1_1/sas_webapps/sas.svcs.logon.war/WEB-INF/view/jsp/default/ui/logoff_custom.jsp

**Table 8. Custom Logoff Message File**

Replace the contents of the `logoff_custom.jsp` file with the following:

```
Thank you for visiting!
<script src="/SASLogon/js/hello.js"></script>
<script>
hello.on('auth.login', function(auth) {
  hello(auth.network).logout({force:true});
  console.log("auth.logout: " + auth.network);
});
</script>
<script src="/SASLogon/js/providers.js"></script>
```

This code ensures that when a user chooses to log out of your SAS environment they also log out of their social login provider.

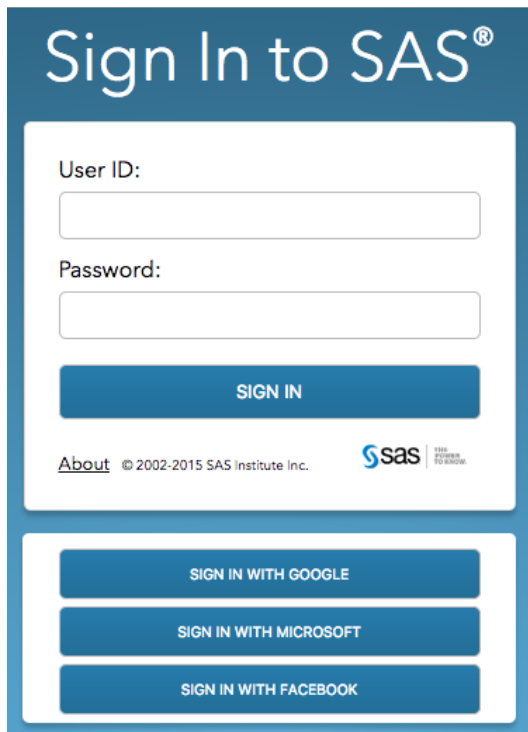
## ENABLING SOCIAL LOGIN

Once all the above changes have been implemented you can now enable your SAS environment to accept social login credentials.

If changes were made to the *Live* file locations, you can restart the SAS Web Application Server that serves the SAS Logon Manager web application, which is usually SASServer1.

If changes were made only to the *Source* file locations then you will need to use the SAS Deployment Manager to rebuild and then redeploy the SAS Web Infrastructure Platform.

Once the SAS Web Application Server has restarted, access the Logon page for your SAS Web Applications. Your Logon page should now look like this:



The image shows the SAS Sign In page. At the top, it says "Sign In to SAS®". Below this, there are two input fields: "User ID:" and "Password:". Below the password field is a blue "SIGN IN" button. At the bottom of the main form area, there is a link to "About" and copyright information "© 2002-2015 SAS Institute Inc." along with the SAS logo and the tagline "THE POWER TO KNOW". Below the main form, there are three additional buttons: "SIGN IN WITH GOOGLE", "SIGN IN WITH MICROSOFT", and "SIGN IN WITH FACEBOOK".

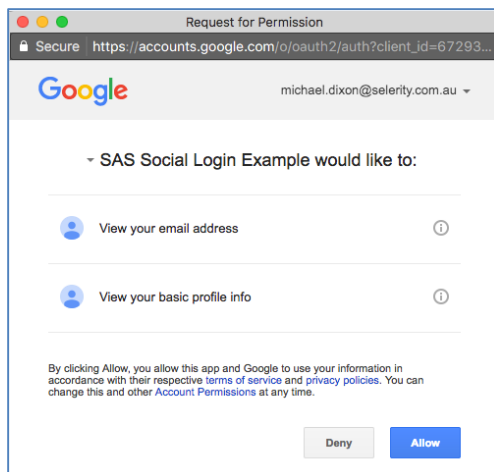
**Figure 12. SAS Logon Page with Social Login Enabled**

## USER LOGON EXPERIENCE

### GOOGLE

When you choose to sign in with your Google account, you will experience the following:

1. A prompt from Google asking for your email address and password
2. A prompt asking your permission to allow the SAS environment access your Google Account details (email address and basic profile information)



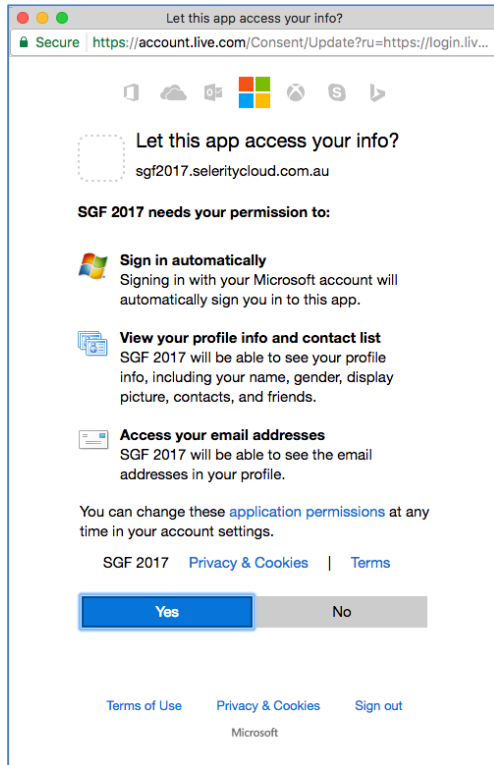
**Figure 13. Google User Security Prompt**

If you allow this access you are then forwarded through to the SAS web application as normal.

## MICROSOFT

When you choose to sign in with your Microsoft account, you will experience the following:

1. A prompt from Microsoft asking for your email address and password
2. A prompt asking your permission to allow the SAS environment access your Microsoft Account details (email address, profile information and contact list)



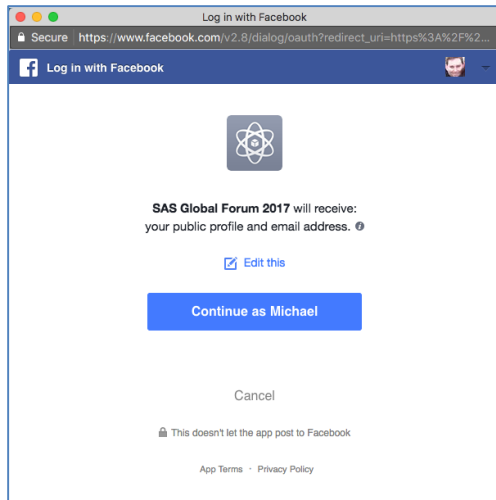
**Figure 14. Microsoft User Security Prompt**

If you allow this access you are then forwarded through to the SAS web application.

## FACEBOOK

When you choose to sign in with your Facebook account, you will experience the following:

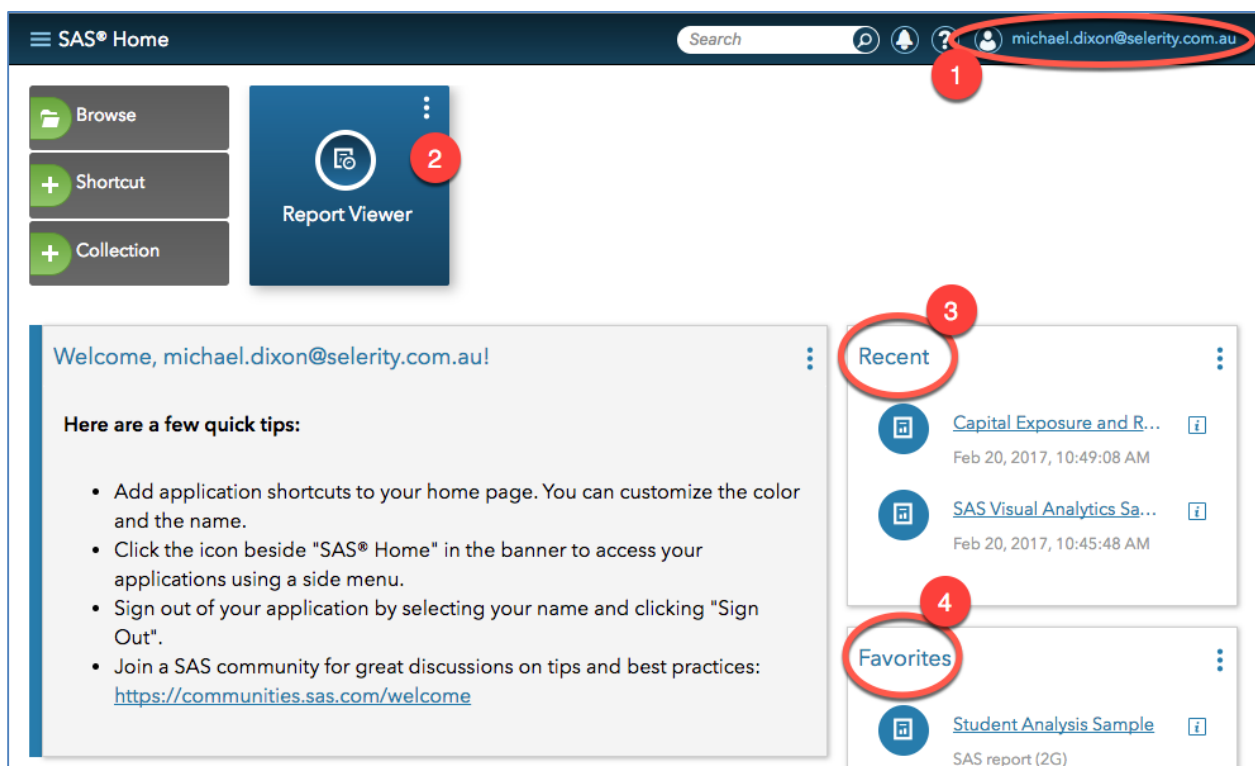
1. A prompt from Facebook for your email address and password
2. A prompt asking your permission to allow the SAS environment access your Facebook Account details (email address and public profile information)



**Figure 15. Facebook User Security Prompt**

If you allow this access you are then forwarded through to the SAS web application.

## SAS VISUAL ANALYTICS HUB EXAMPLE

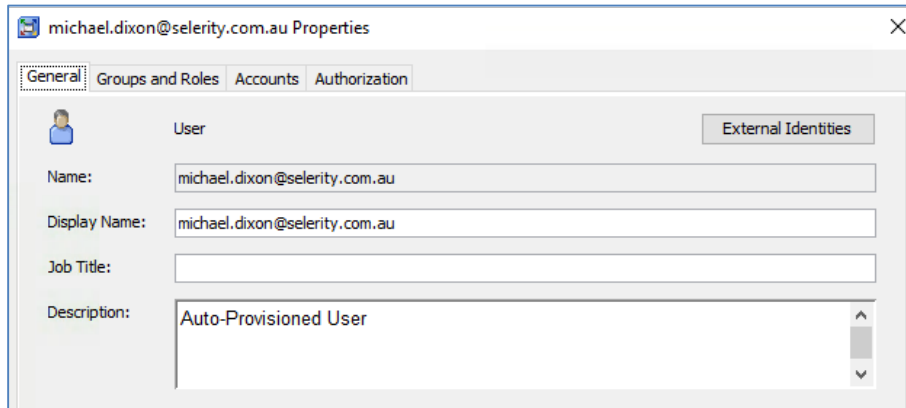


Using your Google account to log into the SAS Visual Analytics Hub the following items are apparent:

1. The system recognizes you by your Google account ID (email address) even though there was no SAS Metadata Identify created prior
2. You only have access to the Report Viewer due to security restrictions enforced by the **Social Login** Metadata group.
3. The items you have personally accessed appear in the **Recent** tile

- Any items you mark as a favorite appear in the **Favorites** tile without affecting any other users

Within the SAS Metadata, a new Identity (User) object was created:



The screenshot shows a window titled "michael.dixon@selerity.com.au Properties" with a close button in the top right corner. Below the title bar are four tabs: "General", "Groups and Roles", "Accounts", and "Authorization". The "General" tab is selected. On the left, there is a user icon and the label "User". To the right of the icon is a button labeled "External Identities". Below these are four input fields: "Name:" with the value "michael.dixon@selerity.com.au", "Display Name:" with the value "michael.dixon@selerity.com.au", "Job Title:" which is empty, and "Description:" with the value "Auto-Provisioned User".

When a user accesses your SAS environment using their social login for the very first time, a SAS Metadata User is created using the email address returned by the provider as their Name (User ID). They are also made a member of the Metadata Group you specified when configuring the system for social login. You can control what access and functionality social login users initially have by modifying the properties of that group in Metadata. For example, you can restrict users to only be able to access reports in a certain Folder in Metadata. This ensures that by default social login users cannot gain access to other areas where staff may be building internal reports, or areas that contain internal only data.

A useful side-effect of having the SAS Metadata User created with the email address as the User ID is that if a user has multiple social login accounts that use the same email address, they can use any provider to log into SAS and they will always be identified by SAS as the same Metadata User.

## CONCLUSION

As demonstrated, benefits of social authentication are available to modern SAS Web Applications allowing you to reach audiences far beyond a typical SAS Application's scope while still maintaining your company's best practices. In the age of Cloud computing, social authentication improves integration, usability and a lower total cost of ownership by de-coupling authentication and providing open and secure standards.

## REFERENCES

- SAS Institute Inc. 2016. *SAS® 9.4 Intelligence Platform: Middle-Tier Administration Guide, Fourth Edition*. "Auto-Provisioning User Accounts", page 220. Cary, NC: SAS Institute Inc. Available at <http://support.sas.com/documentation/cdl/en/bimtag/69826/PDF/default/bimtag.pdf>
- SAS Institute Inc. 2016. *SAS® 9.4 Intelligence Platform: Middle-Tier Administration Guide, Fourth Edition*. "Configuring the Security Module for SAS Web Application Server", page 262. Cary, NC: SAS Institute Inc. Available at <http://support.sas.com/documentation/cdl/en/bimtag/69826/PDF/default/bimtag.pdf>
- hello.js: Andrew Dodson. A client-side JavaScript SDK for authenticating with OAuth. Accessed Feb 2017. Available at <http://adodson.com/hello.js>
- Wikipedia. "Social login." Accessed Feb 2017. Available at [https://en.wikipedia.org/wiki/Social\\_login](https://en.wikipedia.org/wiki/Social_login)
- login radius. (2016). *Customer Identity Preference Trends Q2 2016*. Accessed Feb 2017. Available at <https://blog.loginradius.com/2016/08/customer-identity-preference-trends-q2-2016/>
- Tech Target. (2017). *What is OAuth?* Retrieved Feb 2017. Available at <http://searchmicroservices.techtarget.com/definition/OAuth>

## ACKNOWLEDGMENTS

I would like to thank Cameron Lawson for reviewing and contributing to this paper.

## RECOMMENDED READING

- *SAS® 9.4 Intelligence Platform: Middle-Tier Administration Guide, Fourth Edition*. Available at <http://support.sas.com/documentation/cdl/en/bimtag/69826/PDF/default/bimtag.pdf>
- *OAuth2 Simplified by Aaron Parecki*. Available at <https://aaronparecki.com/oauth-2-simplified/>

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Michael Dixon  
Selerity  
+61 2 8079 2906  
[michael.dixon@selerity.com.au](mailto:michael.dixon@selerity.com.au)  
[www.seleritysas.com](http://www.seleritysas.com)

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.