Integrating Google Apps and Open Directory

v10.5

Randy Saeks

rsaeks@gmail.com

Table of Contents

Introduction

Google Applications for Education is one way schools and universities can provide an online collaborative setting for users. Modules for documents, email and calendaring are major draws to this environment. However, when adding 3rd party solutions to an existing infrastructure, issues surrounding seamless user provisioning and consolidated passwords emerge as potential hurdles to deployment, usage, and ultimately, adoption. From the IT standpoint one obstacle is user account management and maintenance. Viewed from the end-user perspective individuals are faced with a login and password for another system.

To resolve these two issues, Google provides an API and vendors have created solutions to leverage open technologies. One tool creates and manages user accounts in Google Apps from Open Directory. The second provides a web-based SAML Single Sign On solution.

A current limitation exists for users who will be using an email application, such as Mail.app. Since Mail.app does not use SAML authentication, the password for eMail applications will be the password created on initial account creation in Google Apps. To work around this drawback, have users change their Google Apps password to match their Open Directory password. Ensure this is done prior to enabling SSO or any password changes will be directed back to Open Directory and not Google Apps.

Requirements

Below are the requirements needed for the setup covered in this guide:

- Google Applications for Education Domain Registration
- Mac OS X Server v.10.5 running Open Directory
- VMware Virtual Machine Playback support
- SADA Systems Google Apps Provisioning Toolkit Virtual Machine

  (http://hosting.sadasystems.com/sadasystems/google_provisioning)
- SimpleSAMLphp

  (http://rnd.feide.no/simplesamlphp)

This guide is based on the most-recent release of Mac OS X Server as of publication, version 10.5.6.

LDAP User Import

The LDAP User import process is accomplished via SADA Systems Virtual Machine (VM) available at the provided URL. This VM allows a preconfigured system to be downloaded onto a computer and utilized with minimal modifications, mainly user-specific. This tool creates, modifies, deletes, or suspends accounts and its continuous running is not needed. That does mean, however, that the tool will need to be re-run when newly created network accounts need to be added to Google Apps.

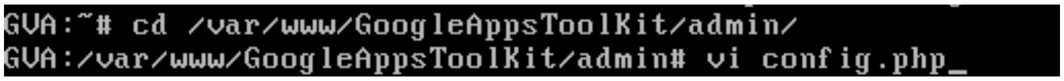
A table is provided with values in the configuration file that will / may need to be specified based on your environment. Additionally, a blank worksheet is included at the end of this document to allow you to keep track of your values.

Attribute	Value	Details
$domain = 'DOMAIN'	Your Google Apps domain	Set DOMAIN to the domain name of your Google Apps domain.
$admin = 'USER'	Login name	Set USER to a login name for a user with admin access to your Google Apps Domain.
$password = 'PASS'	Password	Set PASS to the password for a user with admin access to your Google Apps Domain.
$allow_account_deletion = 'yes, no'	yes, no	Setting this value to no will suspend accounts not found in the import. Setting to yes will delete accounts.
DEFINE('DB_TYPE', 'TYPE')	ldap	Setting TYPE to ldap will configure the toolkit to utilize LDAP connectivity.
DEFINE ('LDAP_SERVER','IP')	Address of an Open Directory Server	Set IP to a valid IP or DNS name of an Open Directory server to retrieve users.
DEFINE('LDAP_PORT' 'PORT')	Port of LDAP Server	Set PORT to the port LDAP is listening on. If a secure LDAP connection, set this to 636 and change IP in the above field to 'ldaps://IP/'
DEFINE('LDAP_BIND_RDN','DN')	Fully qualified name of a user to bind to your LDAP server	Set DN to the fully qualified name of a user to bind to LDAP. Ex: uid=someuser,cn=users,dc=your,dc=domain

DEFINE('LDAP_BIND_PASSWORD','LP')	Password for user specified above	Set LP to the password of the LDAP user you are using above.
DEFINE('LDAP_BASE_DN','BASE')	Directory location of users in LDAP	Set BASE to the LDAP base location of users. Ex: cn=users,dc=your,dc=domain
DEFINE('LDAP_FILTER', 'FILTER')	A filter to restrict objects returned	Set FILTER to a valid LDAP filter criteria. To test setup, you may wish to restrict importing to specific users. You can limit these via using (&(objectclass=person)(uidNumber=XX)) with XX corresponding to the uid of a SPECIFIC user.
DEFINE('LDAP_USERNAME','NAME')	LDAP attribute corresponding to what will be the users Google Apps login.	Set NAME to the LDAP attribute the user will use to login as to Google Apps. Typically, this will be **uid**. If you use another value, enter that value into NAME.
DEFINE('LDAP_DEFAULT_ PASSWORD','P')	Default LDAP password.	Set P to the default password given to all accounts. Since we will later add in steps to connect usernames and Open Directory passwords, this step is not important, but necessary. However, to protect accounts while setting up, use something secure.

Once you have downloaded and expanded the VM, open it in VMware.

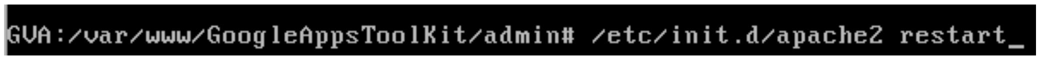
1. Change into the directory containing the configuration files to be edited. The command to change into this directory and edit the configuration file is displayed below. Feel free to use your favorite text editing application.



```
GVA:~# cd /var/www/GoogleAppsToolKit/admin/
GVA:/var/www/GoogleAppsToolKit/admin# vi config.php_
```

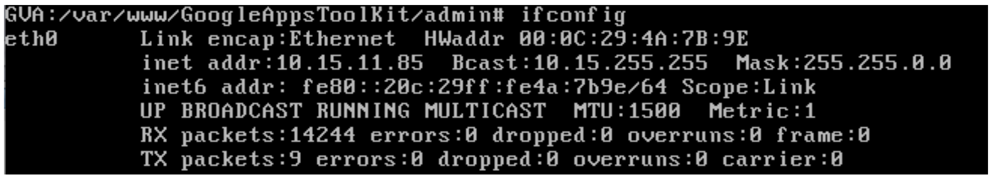
2. Enter in your site-specific values as outlined in the table above.

3. Restart apache.



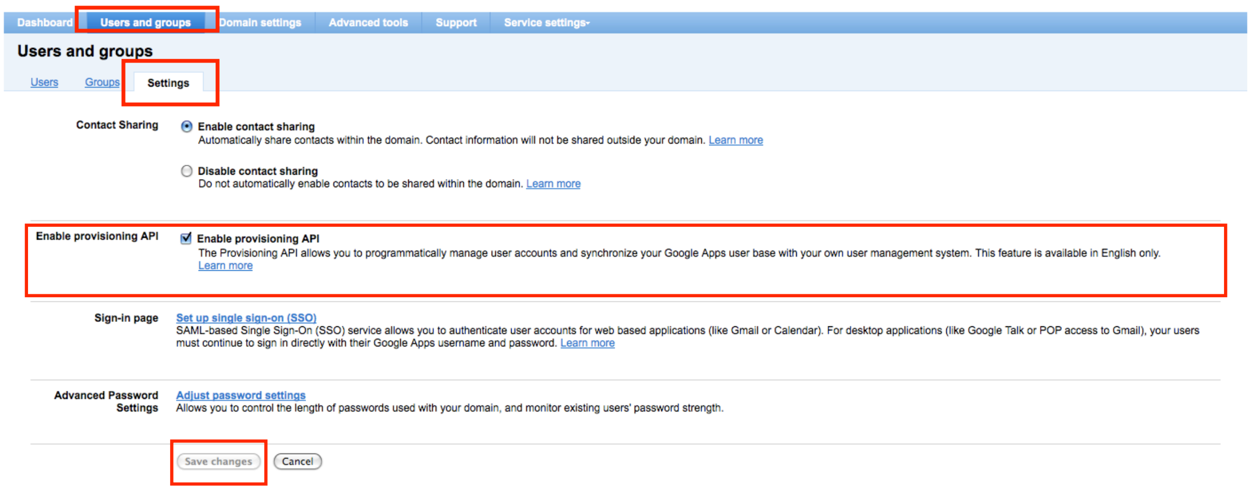
```
GVA:/var/www/GoogleAppsToolKit/admin# /etc/init.d/apache2 restart_
```

4. Obtain the IP address of the machine.



```
GVA:/var/www/GoogleAppsToolKit/admin# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:4A:7B:9E
          inet addr:10.15.11.85  Bcast:10.15.255.255  Mask:255.255.0.0
          inet6 addr: fe80::20c:29ff:fe4a:7b9e/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:14244 errors:0 dropped:0 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
```

5. Enable provisioning API Access in your Google Apps Domain. To do so, go to your Domain Administration Settings then to Users and Groups, and finally to settings. Enable the checkbox for the provisioning API, and click Save Changes.



Users and groups

Contact Sharing
[✓ Enable contact sharing]
Automatically share contacts within the domain. Contact information will not be shared outside your domain. Learn more

Disable contact sharing
Do not automatically enable contacts to be shared within the domain. Learn more

Enable provisioning API
[✓ Enable provisioning API]
The Provisioning API allows you to programmatically manage user accounts and synchronize your Google Apps user base with your own user management system. This feature is available in English only.
Learn more

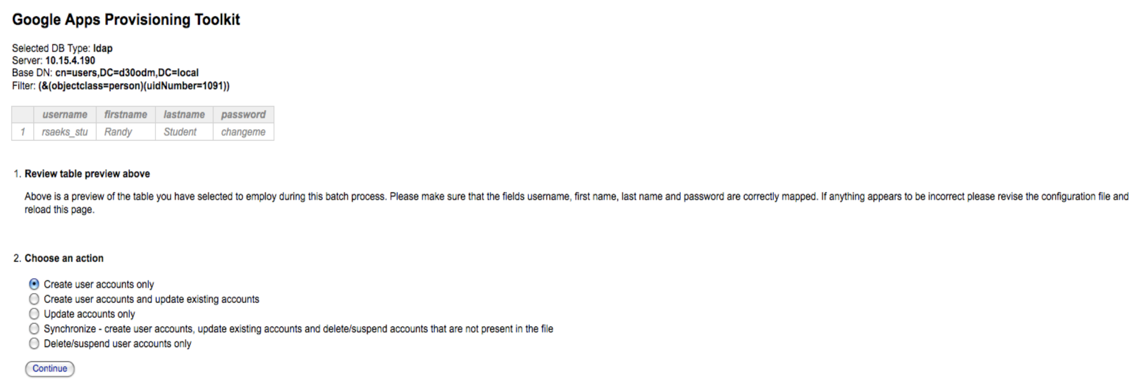
Sign-in page
Set up single sign-on (SSO)
SAML-based Single Sign-On (SSO) service allows you to authenticate user accounts for web based applications (like Gmail or Calendar). For desktop applications (like Google Talk or POP access to Gmail), your users must continue to sign in directly with their Google Apps username and password. Learn more

Advanced Password Settings
Adjust password settings
Allows you to control the length of passwords used with your domain, and monitor existing users' password strength.

6. Connect to the machine via the IP address obtained in step 4. The URL will be in the format:

   http://IPADDRESS/GoogleAppsToolKit/admin/

7. If the configuration is correct, you will see a screen similar to the one below:



Google Apps Provisioning Toolkit

Selected DB Type: ldap
Server: 10.15.4.190
Base DN: cn=users,DC=d30odm,DC=local
Filter: (&(objectclass=person)(uidNumber=1091))

	username	firstname	lastname	password
1	rsaeks_stu	Randy	Student	changeme

1. Review table preview above

Above is a preview of the table you have selected to employ during this batch process. Please make sure that the fields username, first name, last name and password are correctly mapped. If anything appears to be incorrect please revise the configuration file and reload this page.

2. Choose an action

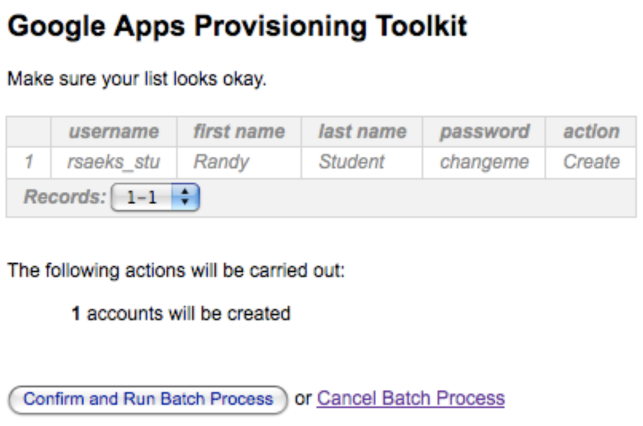
[✓ Create user accounts only]
Create user accounts and update existing accounts
Update accounts only
Synchronize - create user accounts, update existing accounts and delete/suspend accounts that are not present in the file
Delete/suspend user accounts only

8. Note in the above screen shot the information provided. This summarizes the location in LDAP being searched (Base DN) and the applied filter. Also, the information to be added will be displayed. In the above case, a specific user was selected based on their uidNumber being equal to 1091.

9. Step two of the process is to select an action to perform with the user import. At this point, clicking next will preview and create your batch job but not actually run it. A summary of the actions is provided below. Select the option best suited for your specific operation.

Action	Summary
Create user accounts only	This will add listed users to Google Apps. Any existing accounts will be retained and left untouched.
Create user accounts and update existing accounts	This will add new accounts to Google Apps. Existing accounts will be updated with displayed information.
Update accounts only	Updating the accounts will only change or reset settings on existing accounts. No new accounts will be created.
Synchronize - create user accounts, update existing accounts and delete/suspend accounts that are not present in the file	Synchronize will create new accounts based on displayed information and update any existing accounts and delete or suspend accounts not listed in the import file.
Delete/suspend user accounts only	If $allow_account_deletion = 'yes, no' is set to yes, this will delete accounts from your Google Apps user list that are not listed in the import file. If set to no, the account will be suspended and existing login information retained.

10. After selecting the action to be performed and clicking next, the Provisioning toolkit will scan Open Directory and generate a preview of the actions to be performed. This is the location where you can preview changes before they are applied. A sample summary screen is provided. This screen displays a summary of accounts to be created, deleted, or modified, as well as the user data corresponding to the account. Note the password being used is set by the directive DEFINE('LDAP_DEFAULT_PASSWORD','P') from the previous configuration, with P based on being set to changeme.



Google Apps Provisioning Toolkit

Make sure your list looks okay.

	username	first name	last name	password	action
1	rsaeks_stu	Randy	Student	changeme	Create

Records: 1-1

The following actions will be carried out:

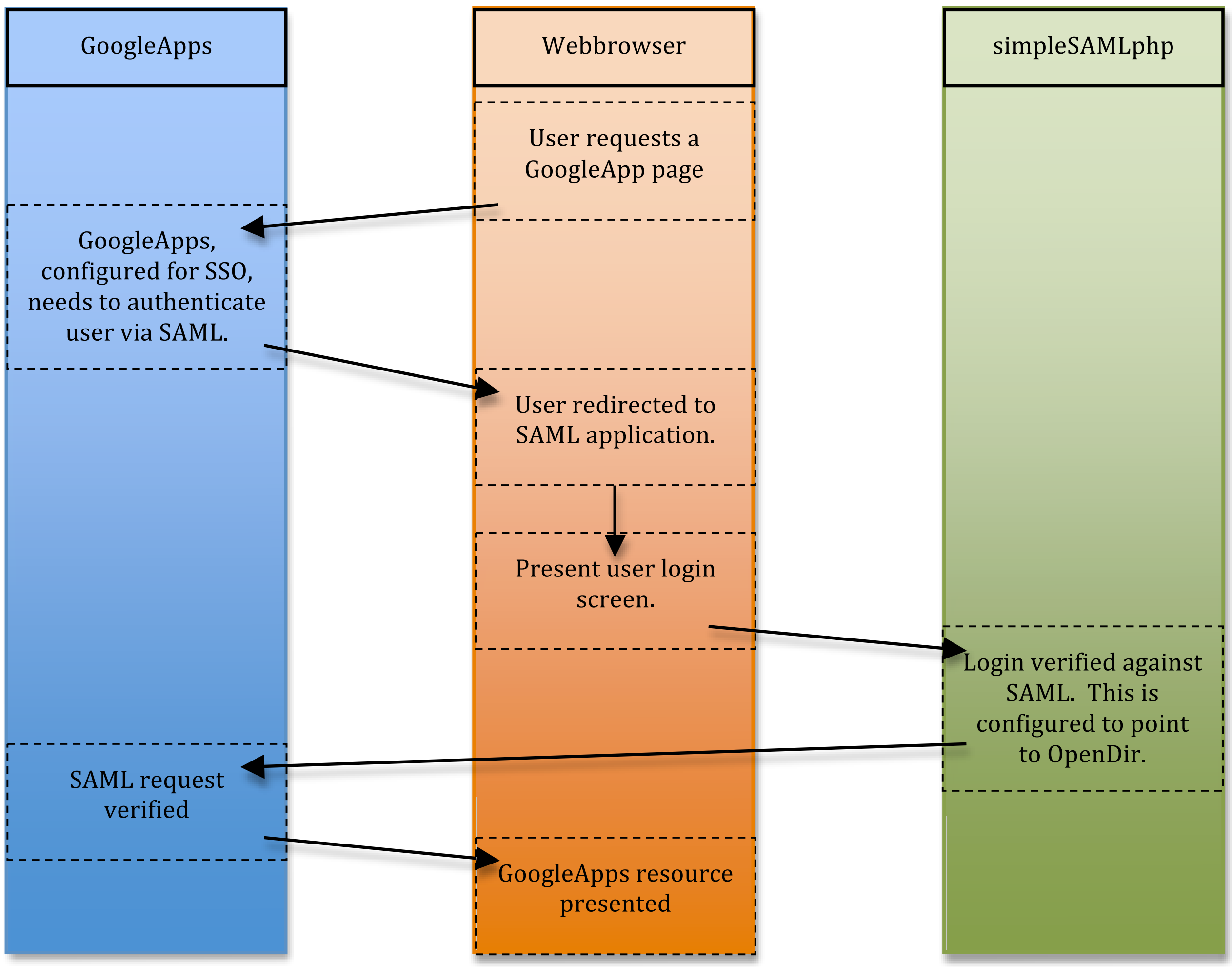
1 accounts will be created

11. Once you have completed your batch operations, you can power down the VM and disable provisioning API access in Google Apps.

At this point your Google Apps domain will have user accounts from Open Directory. To verify a new account was successfully created, login to your Google Apps domain with the newly created user. Remember, at this point the password for the user is set to the default provided in the configuration. The next step will be to install and configure a Web-based SAML Identity Provider. This will link Google Apps user account to your Open Directory system and allow for Open Directory passwords to be used on Google Apps.

Web-based Single-Sign On

Google Apps allows for a Single-Sign On (SSO) system to be leveraged simplifying user authentication to the Google services. Web-based SSO through SAML allows for a 3rd party Service Provider (Google Apps) to talk to an in-house Identity Provider (in this case simpleSAMLphp) which will provide authentication services. simpleSAMLphp in turn is configured to authenticate users to a specific LDAP directory. For the scope of this paper, the focus is on Open Directory. Below, the process is illustrated.


GoogleApps
Webbrowser
simpleSAMLphp
User requests a GoogleApp page
GoogleApps, configured for SSO, needs to authenticate user via SAML.
User redirected to SAML application.
Present user login screen.
Login verified against SAML. This is configured to point to OpenDir.
SAML request verified
GoogleApps resource presented

Installing simplesamlphp

Once you have downloaded simpleSAMLphp, extract it to your desktop.

1. Rename the folder to "simplesaml"
2. Open a terminal window and change to the root user. Once you have done this, create a simplesamlphp folder in /var.

```
cd /var
mkdir simplesamlphp
```

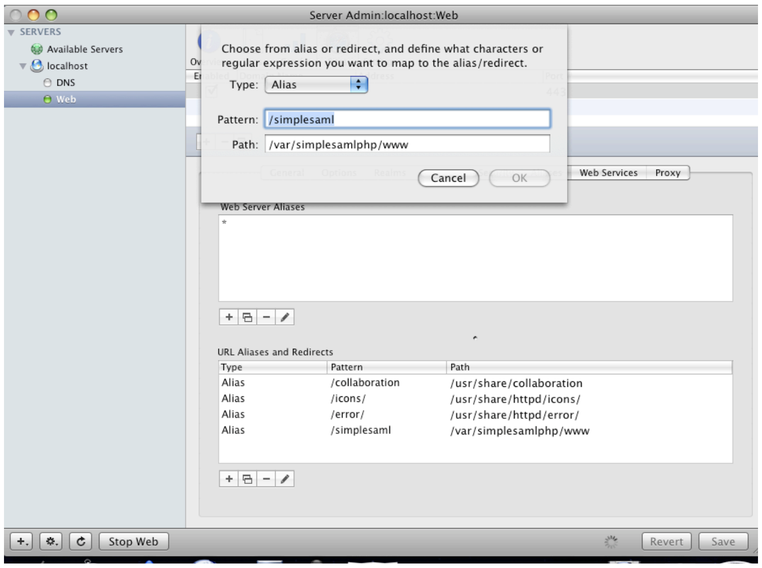
3. Copy the contents of the simplesamlphp folder on your desktop to /var/simplesamlphp.

```
cp -R /Users/USER/Desktop/simplesamlphp/* /var/simplesamlphp/
```

4. Copy the included sample configuration and meta-data templates into the production folder.

```
cd /var/simplesamlphp
cp -r config-templates/*.php config/
cp –r metadata-templates/*.php metadata/
```

5. Create a web Alias to /var/simplesamlphp/www. This can be done in Server Admin. To do so, open Server Admin and select the web service, then sites. To create the alias, click the plus sign under "URL Aliases and Redirects" and enter in the path above. You should consider allowing only https traffic to this machine, as it will be working with users and passwords.



Choose from alias or redirect, and define what characters or regular expression you want to map to the alias/redirect.

Type: Alias

Pattern: /simplesaml

Path: /var/simplesamlphp/www

Web Server Aliases

*

URL Aliases and Redirects

Type	Pattern	Path
Alias	/collaboration	/usr/share/collaboration
Alias	/icons/	/usr/share/httpd/icons/
Alias	/error/	/usr/share/httpd/error/
Alias	/simplesaml	/var/simplesamlphp/www

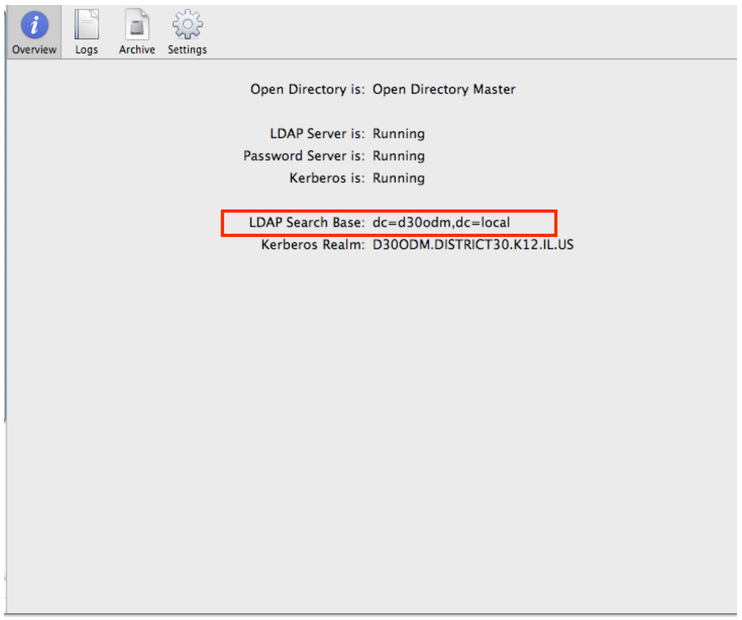
6. Enable the included php module. This can be accomplished in the web service settings of Server Admin, under modules. Place a checkbox in php5_module then save your settings and restart the web service.
7. Edit the provided config.php file and specify an administrative password. Open /var/simplesamlphp/config/config.php and change auth.adminpassword to something you would like to use.
8. Edit the config.php file and specify a value for secretsalt. This can be any random string and will be used to generate secure hashes. You can enter in a random string of letters and numbers.
9. Specify contact information in the fields technicalcontact_name and technicalcontact_email.
10. Disable using simplesamlphp as a service provider and enable using it as an identity provider. To make these changes, change the value of enable.saml20-sp to false and saml20-idp to true.
11. Save the changes to your file.
12. Enable LDAP support for simpleSAMLphp by running the following commands:

```
cd /var/simplesamlphp/modules/ldap
touch enable
```

At this point, you may test your install by navigating to http(s)://your.server.com/simplesaml. If the system is properly setup, you will see a screen for the web application. Additionally, the only item that should have a green check next to it is SAML 2.0 IdP. If you are able to view your page, you can now move onto configuring LDAP.

Configuring simpleSAMLphp for Open Directory

1. Open the ldap.conf file located in /var/simplesamlphp/config.
2. Edit the value of auth.ldap.dnpattern to specify the location in Open Directory to search for users. If you don't know this value, you can find it by using Server Admin and viewing the status of Open Directory. It will be located the Overview tab. This value should be in the format uid=%username%,cn=users,<u>dc=your,dc=domain</u>. The portion underlined is what is shown in the Overview Tab of Open Directory as the LDAP Search Base.



```
Open Directory is:  Open Directory Master

     LDAP Server is:  Running
 Password Server is:  Running
        Kerberos is:  Running

  LDAP Search Base:  dc=d30odm,dc=local
    Kerberos Realm:  D30ODM.DISTRICT30.K12.IL.US
```

3. Change the value of auth.ldap.hostname to match the hostname of your Open Directory Server.
4. In the event you require binding, change auth.ldap.search.username and auth.ldap.search.password to a valid user having access rights to perform an LDAP search.
5. Save the file.

Configuring simplesamlphp for Google Apps for Education

This process will configure Google Apps for Education to use a 3rd Party Identity Provider; in this case, simplesamlphp.

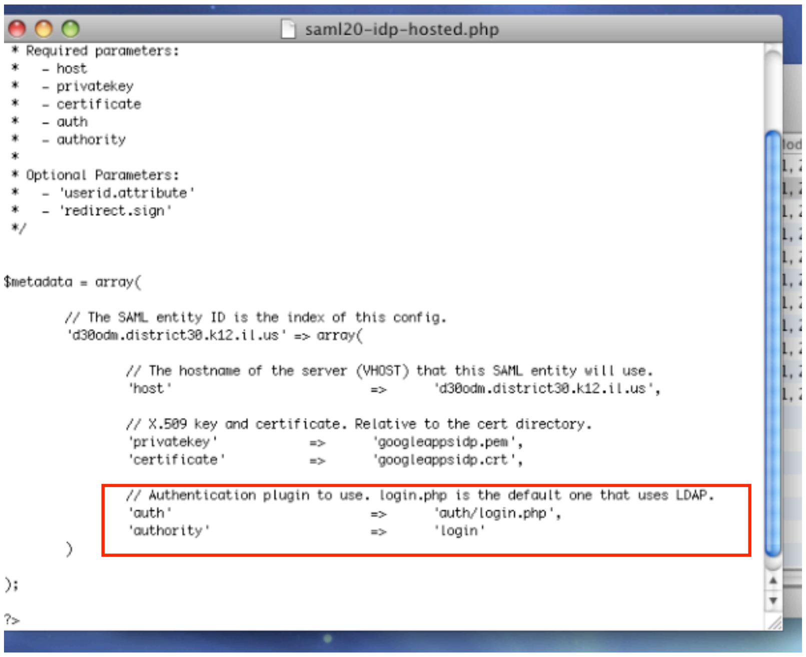
1. Generate SSL certificates for GoogleApps in terminal to use with simplesaml. Note: the commands are entered with a space between to reduce confusion. The final command is wrapped around lines and the "\" is not needed to be input to the command line.

```
cd /var/simplesamlphp/cert/

openssl genrsa –des3 –out googleappsidp.key 1024

openssl rsa -in googleappsidp.key -out googleappsidp.pem

openssl req -new -key googleappsidp.key -out googleappsidp.csr

openssl x509 -req -days 1095 -in googleappsidp.csr –signkey \
googleappsidp.key -out googleappsidp.crt
```

2. Edit the hosted metadata identity provider file to permit your server to honor Google App authentication requests. Open /var/simplesamlphp/metadata/saml20-idp-hosted.php and change:
    - __DYNAMIC:1__ to the hostname of your machine
    - host to the hostname of your machine
    - privatekey to googleappsidp.pem
    - certificate to googleappsidp.crt

3. Add a comma to the end of the line starting with 'auth' located near the bottom of the file (shown in the next page).
4. Add a line after the 'auth' line to read:

```
'authority'                    => 'login'
```



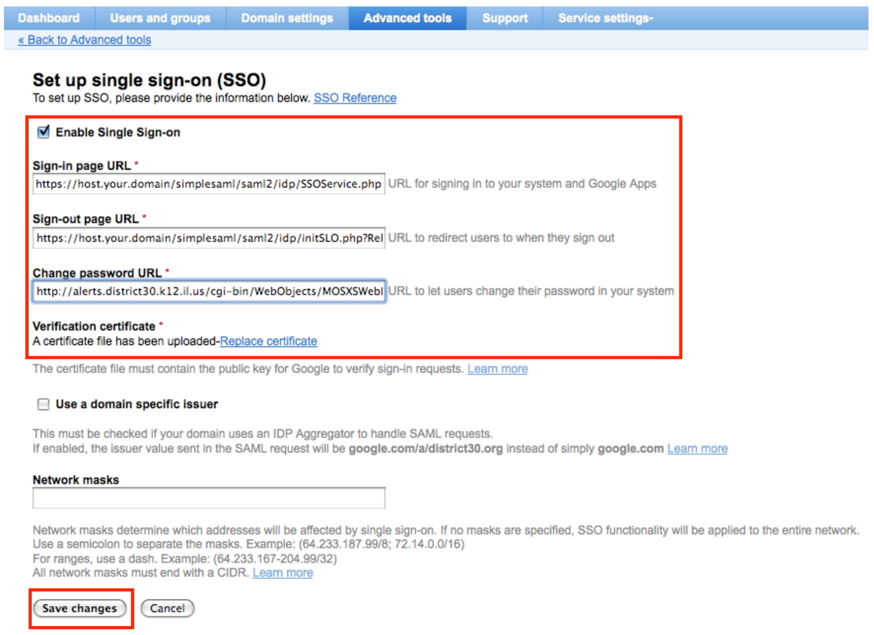
```
 * Required parameters:
 *    - host
 *    - privatekey
 *    - certificate
 *    - auth
 *    - authority
 *
 * Optional Parameters:
 *    - 'userid.attribute'
 *    - 'redirect.sign'
 */


$metadata = array(

        // The SAML entity ID is the index of this config.
        'd30odm.district30.k12.il.us' => array(

                // The hostname of the server (VHOST) that this SAML entity will use.
                'host'                          =>      'd30odm.district30.k12.il.us',

                // X.509 key and certificate. Relative to the cert directory.
                'privatekey'            =>      'googleappsidp.pem',
                'certificate'           =>      'googleappsidp.crt',

                // Authentication plugin to use. login.php is the default one that uses LDAP.
                'auth'                          =>      'auth/login.php',
                'authority'                     =>      'login'
        )

);

?>
```

5. Save your changes.
6. Edit the remote server provider metadata file to allow your server to respond to Google App authentication requests. Open /var/simplesamlphp /metadata/saml20-sp-remote.php. Near the bottom of the file, you will see an array for google.com. Under the field AssertionConsumerService change the default value of g.feide.no to your hosted domain.

Configuring Google Apps for Education to use SSO

Now that everything is configured, the final process is to enable Google Apps for Education to utilize your web-based Single-Sign On Solution.

1. Login to your Google Apps Domain, and select advanced tools. From there, select "Single Sign on"
2. Place a checkmark in the box "Enable Single Sign On".
3. Enter in the URL of your sign-in page. It should be in the format:



Set up single sign-on (SSO)
To set up SSO, please provide the information below. SSO Reference

[✓] Enable Single Sign-on

Sign-in page URL *
https://host.your.domain/simplesaml/saml2/idp/SSOService.php URL for signing in to your system and Google Apps

Sign-out page URL *
https://host.your.domain/simplesaml/saml2/idp/initSLO.php?Rel URL to redirect users to when they sign out

Change password URL *
http://alerts.district30.k12.il.us/cgi-bin/WebObjects/MOSXSWebl URL to let users change their password in your system

Verification certificate *
A certificate file has been uploaded-Replace certificate
The certificate file must contain the public key for Google to verify sign-in requests. Learn more

[ ] Use a domain specific issuer

This must be checked if your domain uses an IDP Aggregator to handle SAML requests.
If enabled, the issuer value sent in the SAML request will be **google.com/a/district30.org** instead of simply **google.com** Learn more

Network masks

Network masks determine which addresses will be affected by single sign-on. If no masks are specified, SSO functionality will be applied to the entire network.
Use a semicolon to separate the masks. Example: (64.233.187.99/8; 72.14.0.0/16)
For ranges, use a dash. Example: (64.233.167-204.99/32)
All network masks must end with a CIDR. Learn more

   https://host.your.domain/simplesaml/saml2/idp/SSOService.php

4. Enter in the URL of your sign-out page. It should be in the format:

https://host.your.domain/simplesaml/saml2/idp/initSLO.php?RelayState=/simplesaml/logout.php

5. Enter in the URL of your password change URL. If you do not have one, you can enter in a fake URL.
6. Under "Verification Certificate", select the googleappsidp.crt certificate created earlier. It will be located in /var/simplesamlphp/cert/. To copy the file to a folder viewable to the web browser, choose "Go" from the Finder menu and then "To

Folder". Enter in the above path and copy the .crt file to your desktop. Select the file and click Upload.

7. Click Save Changes.

IMPORTANT: To test your SSO implementation, there are two ways to accomplish this:

1. Enable Google Apps Single-Sign on from a test block of IPs by specifying a range in the field "Network Mask".

   - or-

2. Use a different test machine to login to your Google Apps Domain and stay logged in with your current session.

If you do not perform one of these two actions, you run the risk of being unable to login to you Google Apps domain and will need to contact Google support to turn off Single Sign On. This can occur due to the LDAP server not being reachable or an incorrect configuration of LDAP attribute mappings.

Additional Resources

Google Apps Provisioning Toolkit Worksheet

string	value	Your Value
$domain = ‘DOMAIN’	Your Google Apps domain	
$admin = ‘USER’	Login name	
$password = ‘PASS’	Password	
$allow_account_deletion = ‘yes, no’	yes, no	
DEFINE(‘DB_TYPE’, ‘TYPE’)	ldap	
DEFINE (‘LDAP_SERVER’,’IP’)	Address of an Open Directory Server	
DEFINE(‘LDAP_PORT’ ‘PORT’)	Port of LDAP Server	
DEFINE(‘LDAP_BIND_RDN’,’DN’)	Fully qualified name of a user to bind to your LDAP server.	
DEFINE(‘LDAP_BIND_PASSWORD’,’LP’)	Password for user specified above	
DEFINE(‘LDAP_BASE_DN’,’BASE’)	Directory location of users in LDAP	
DEFINE(‘LDAP_FILTER’, ‘FILTER’)	A filter to restrict objects returned.	
DEFINE(‘LDAP_USERNAME’,’NAME’)	LDAP attribute corresponding to what will be the users Google Apps login.	
DEFINE(‘LDAP_DEFAULT_ PASSWORD’,’P’)	Default LDAP password.	