PingFederate®

Syncplicity Provisioner v2.0.0

User Guide

PingFederate Syncplicity Provisioner User Guide
Version 2.0.0
March 2018

Ping Identity Corporation
1001 17th Street, Suite 100
Denver, CO 80202
USA

Phone: 877.898.2905 (+1 303.468.2882 outside North America)
Fax: 303.468.2909   E-mail: info@pingidentity.com
Web Site: http://www.pingidentity.com

Trademarks

Disclaimer

This document is proprietary and not for general publication. It may be provided ad hoc for informational purposes only, and the information herein is subject to change without notice. Ping Identity does not provide any warranties and specifically disclaims any liability in connection with this document.

Note that Ping Identity may not provide support for any sample configurations provided in this document. The variability inherent among security environments prevents full testing and support for all possible platform configurations. If you need special assistance or would like to inquire about implementation or support programs, please contact Ping Identity Global Client Services (http://support.pingidentity.com).

PingFederate - Syncplicity Provisioning: User Guide

The Syncplicity SaaS Provisioner is intended for use by a PingFederate customer who would like to to integrate outbound provisioning with Syncplicity as the service provider.

1.0 Prerequisites

1. This document assumes you already have a PingFederate environment installed and functional (i.e. you already have at least one IDP Adapter configured for user login).
   - PingFederate 9.x+
   - JDK version 8+
2. This document assumes you already determined the Data Store that contains the user accounts which you want to be Provisioned into the Syncplicity application.
3. If this is your first time using Outbound Provisioning with PingFederate: you need to create a new Database where PingFederate can store information about accounts it needs to provision/synchronize over to your Service Provider (see https://documentation.pingidentity.com/pingfederate/pf80/index.shtml#concept_configuringOutboundProvisioningSettings.html#concept_configuringOutboundProvisioningSettings).
   - PingFederate has been tested using Hypersonic, Oracle, and MySQL databases as internal provisioning data stores. However, any relational database should work as well; adaptable setup scripts used for Hypersonic, Oracle, and MySQL are provided in the directory: `<pf_install>/pingfederate/server/default/conf/provisioner/sql-scripts`
4. You will need to create an Administrator user account in that Provisioning Database so PingFederate can use that account to update the data.
5. If your PingFederate environment is running as a cluster, you will likely want failover capabilities for the

provisioning feature as well. This requires that all PingFederate nodes have connectivity to the Provisioning Database.

2.0 Installation of the Components

1. Download the pf-syncplicity-saas-provisioner-2.0.0.zip file onto your PingFederate server.
2. Extract the contents of the zip package to a location of your choice.
3. Copy the following files into the *<pingfederate root>\pingfederate\server\default\deploy* directory
    - pf-syncplicity-saas-provisioner-2.0.0.jar
4. Restart PingFederate.
5. If you are running PingFederate in a cluster, you will have to repeat Steps 1-4 for each PingFederate server.

3.0 PingFederate Configuration

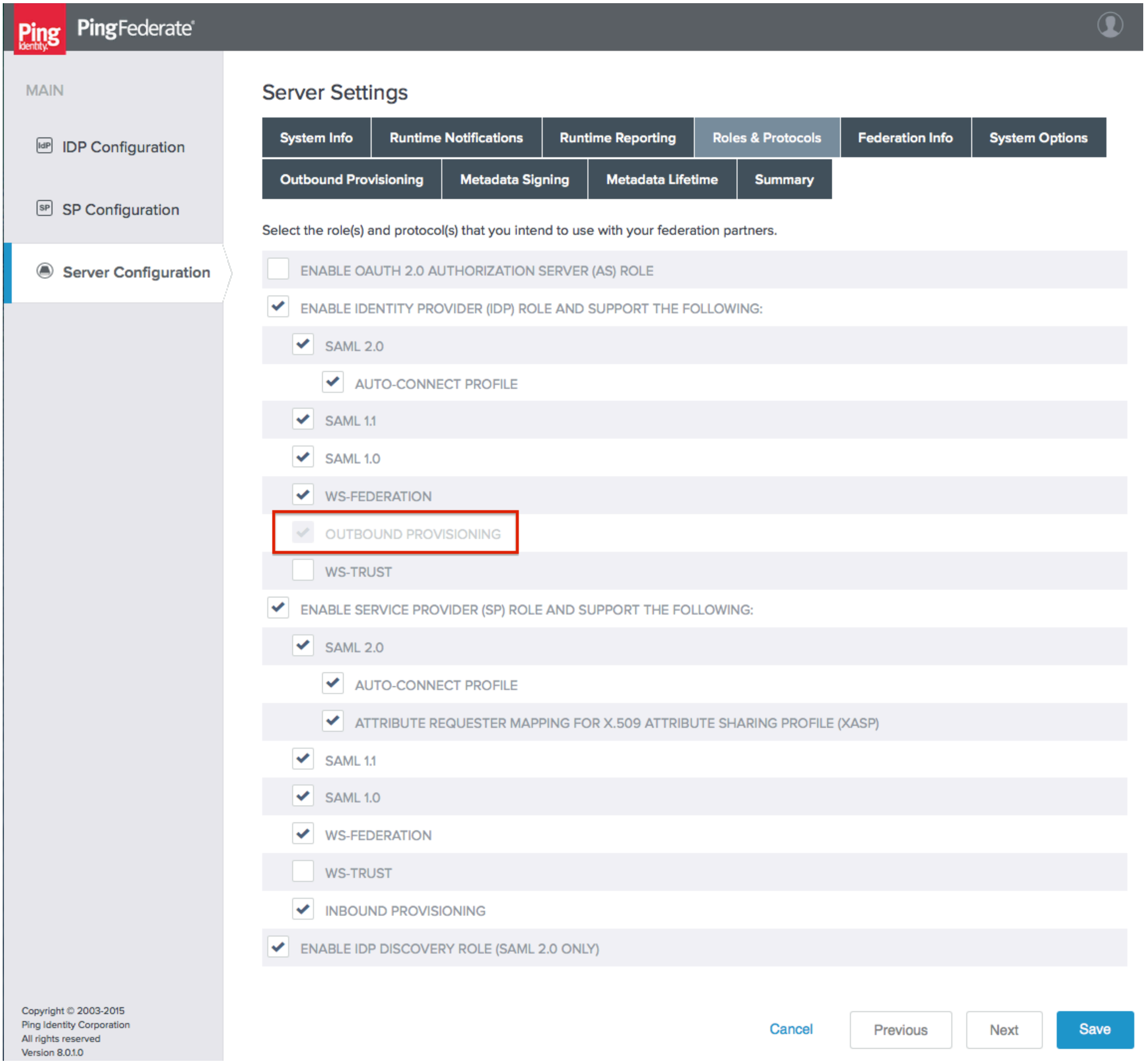
If you have never enabled Outbound Provisioning in this PingFederate environment, please complete the steps in section 3.1 and 3.2. Otherwise you may skip 3.1 and go directly to section 3.2.

3.1 Enable Outbound Provisioning

1. Login to your **PingFederate admin console.**

2. Go to **Server Settings >> Roles & Protocols.**

3. Enable **Outbound Provisioning** under the IdP role.



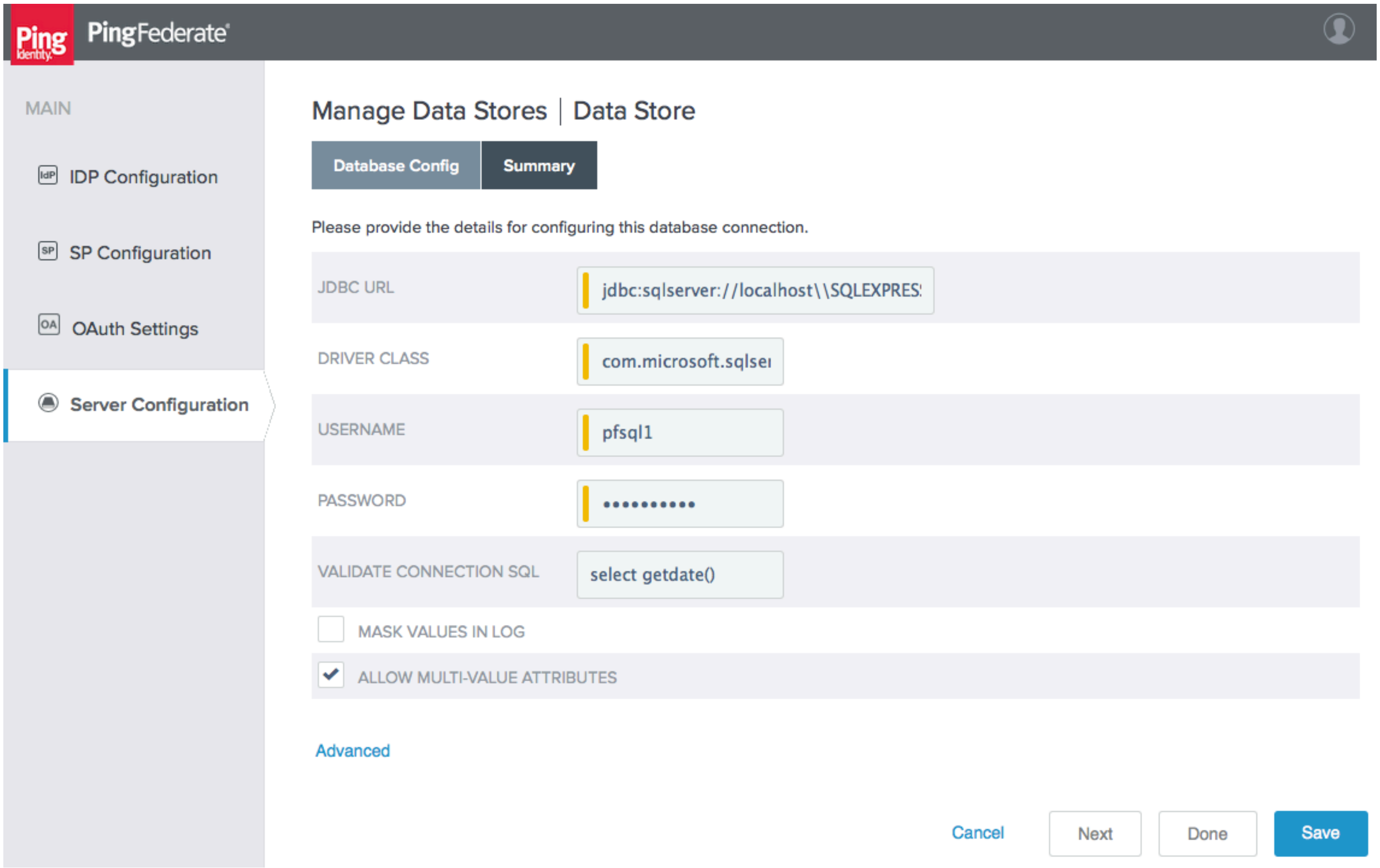
Server Settings

Select the role(s) and protocol(s) that you intend to use with your federation partners.

- ☐ ENABLE OAUTH 2.0 AUTHORIZATION SERVER (AS) ROLE
- ☑ ENABLE IDENTITY PROVIDER (IDP) ROLE AND SUPPORT THE FOLLOWING:
  - ☑ SAML 2.0
    - ☑ AUTO-CONNECT PROFILE
  - ☑ SAML 1.1
  - ☑ SAML 1.0
  - ☑ WS-FEDERATION
  - ☑ OUTBOUND PROVISIONING
  - ☐ WS-TRUST
- ☑ ENABLE SERVICE PROVIDER (SP) ROLE AND SUPPORT THE FOLLOWING:
  - ☑ SAML 2.0
    - ☑ AUTO-CONNECT PROFILE
    - ☑ ATTRIBUTE REQUESTER MAPPING FOR X.509 ATTRIBUTE SHARING PROFILE (XASP)
  - ☑ SAML 1.1
  - ☑ SAML 1.0
  - ☑ WS-FEDERATION
  - ☐ WS-TRUST
  - ☑ INBOUND PROVISIONING
- ☑ ENABLE IDP DISCOVERY ROLE (SAML 2.0 ONLY)

4. Click **Save.**

5. **Select the Provisioning Data Store** you wish to use from the drop down list – **or** – if you have not configured a Provisioning Data Store, click **Manage Data Stores…**
   a. Click Add New Data Store.
   b. Select Database and click Next.
   c. Complete the required fields for JDBC URL, Driver Class, and the Username/Password for an account with Administrator permissions to the database. For further details on Data Store configuration go here: https://documentation.pingidentity.com/pingfederate/pf80/index.shtml#concept_managingDataStores.html#concept_managingDataStores



Manage Data Stores | Data Store

Please provide the details for configuring this database connection.

JDBC URL: jdbc:sqlserver://localhost\\SQLEXPRES

DRIVER CLASS: com.microsoft.sqlsei

USERNAME: pfsql1

PASSWORD: ••••••••••

VALIDATE CONNECTION SQL: select getdate()

☐ MASK VALUES IN LOG

☑ ALLOW MULTI-VALUE ATTRIBUTES

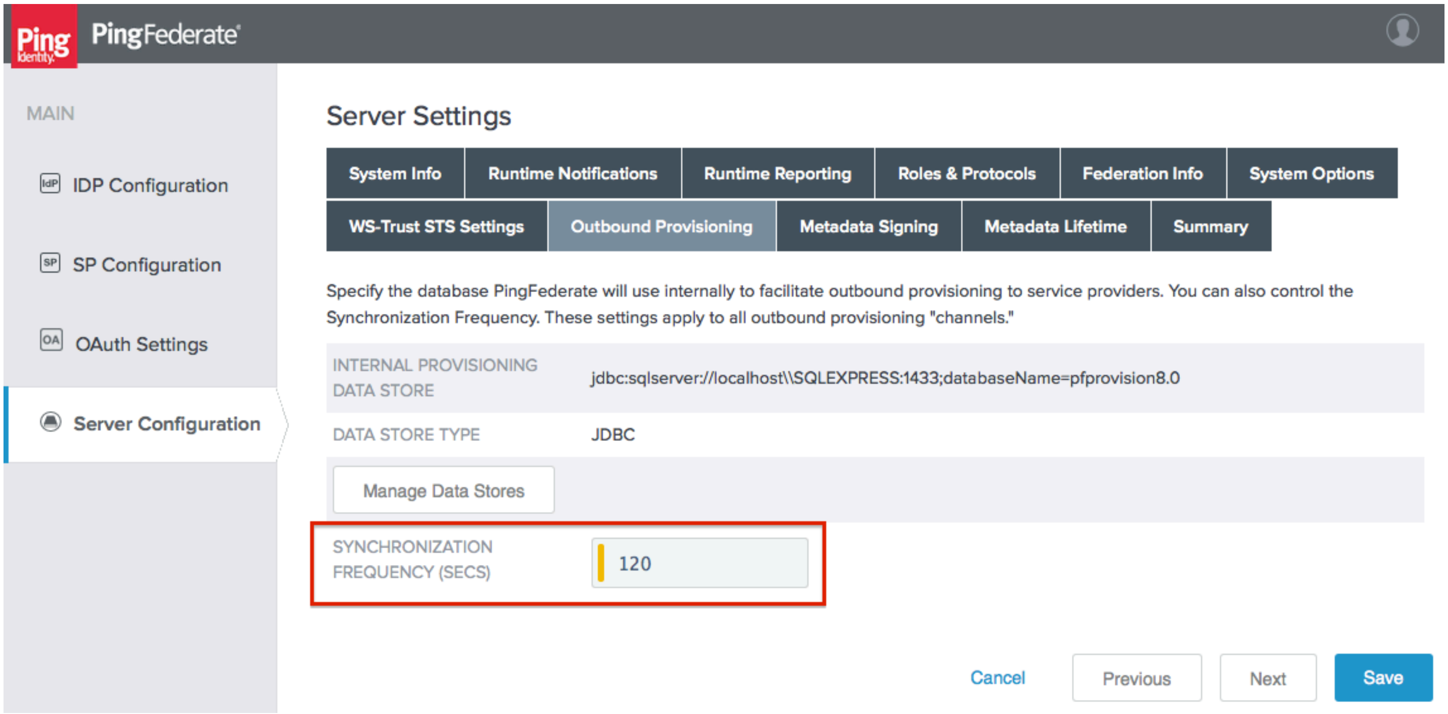
d. Click Next.

e. Click **Done**.

f. Click **Save**.

g. On the Outbound Provisioning page, you should now select your new Data Store from the drop down list.

6. Select the **Synchronization Frequency** at which you want changes to be monitored (i.e. If you add a new user, do you want this to be synchronized to your service providers within 60 seconds or more, etc.).



Server Settings

Specify the database PingFederate will use internally to facilitate outbound provisioning to service providers. You can also control the Synchronization Frequency. These settings apply to all outbound provisioning "channels."

INTERNAL PROVISIONING DATA STORE	jdbc:sqlserver://localhost\\SQLEXPRESS:1433;databaseName=pfprovision8.0
DATA STORE TYPE	JDBC
SYNCHRONIZATION FREQUENCY (SECS)	120

7. Click **Save.**

8. If you are running PingFederate in a cluster, you should **replicate** your changes now.

9. Next, open the **run.properties** file located in your <pf_install>/pingfederate/bin folder.

10. Update the **pf.provisioner.mode** setting to STANDALONE or FAILOVER
    a. Note: If you run PingFederate as a cluster, you'll likely want FAILOVER capabilities for this Provisioning feature to ensure there are no delays in user synchronization to your Service Providers.

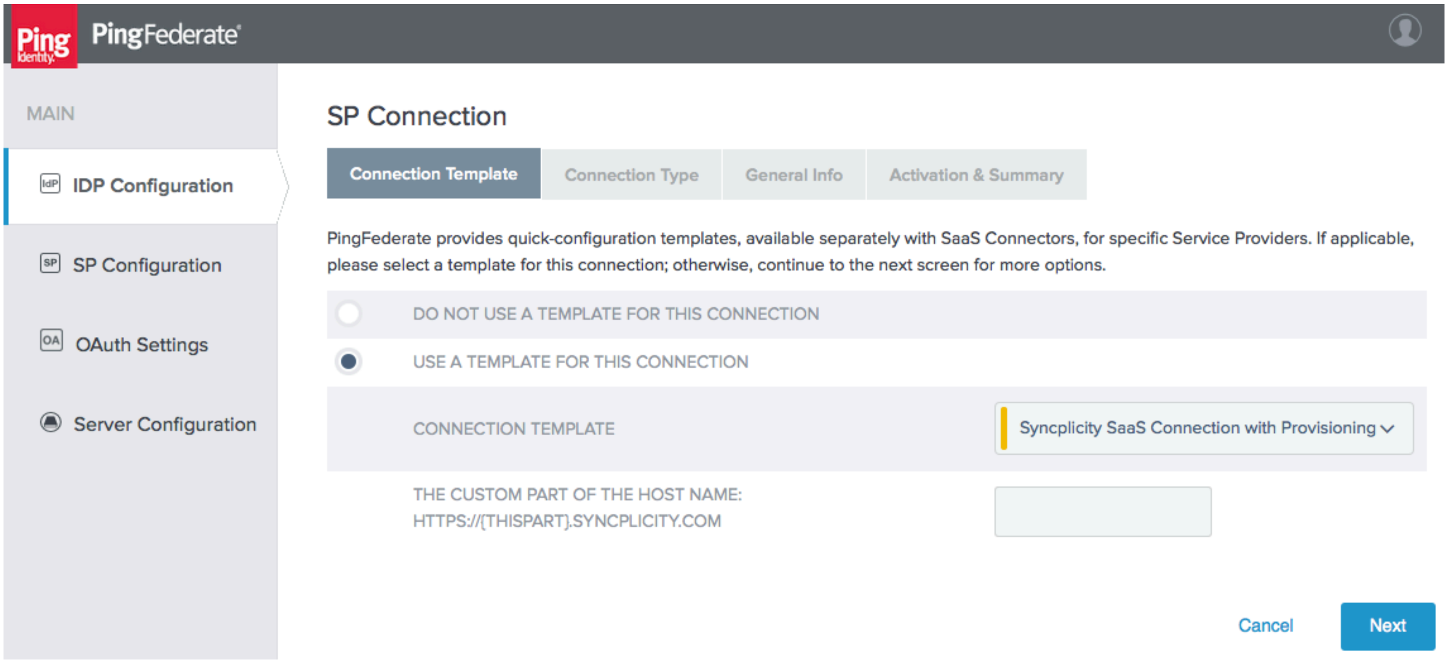
11. If you selected FAILOVER:
    a. Ensure you specify a unique provisioner.node.id for each PingFederate node.
    b. Specify a value for provisioner.failover.grace.period . This means if the current provisioning node is down for that amount of seconds, the next node will take over provisioning.

12. **Save** the file.

13. If you are running PingFederate in a cluster you must repeat steps 9-12 on each node.

3.2 Create the SP Connection with Provisioning

1. Login to your PingFederate Administrator Console.
2. Under **IdP Configuration >> SP Connection >>** click **Create New.**
3. On the **Connection Template** page
    a. Select **Use a template for this connection**
    b. For Connection Template select **Syncplicity SaaS Connection with Provisioning**



SP Connection

PingFederate provides quick-configuration templates, available separately with SaaS Connectors, for specific Service Providers. If applicable, please select a template for this connection; otherwise, continue to the next screen for more options.

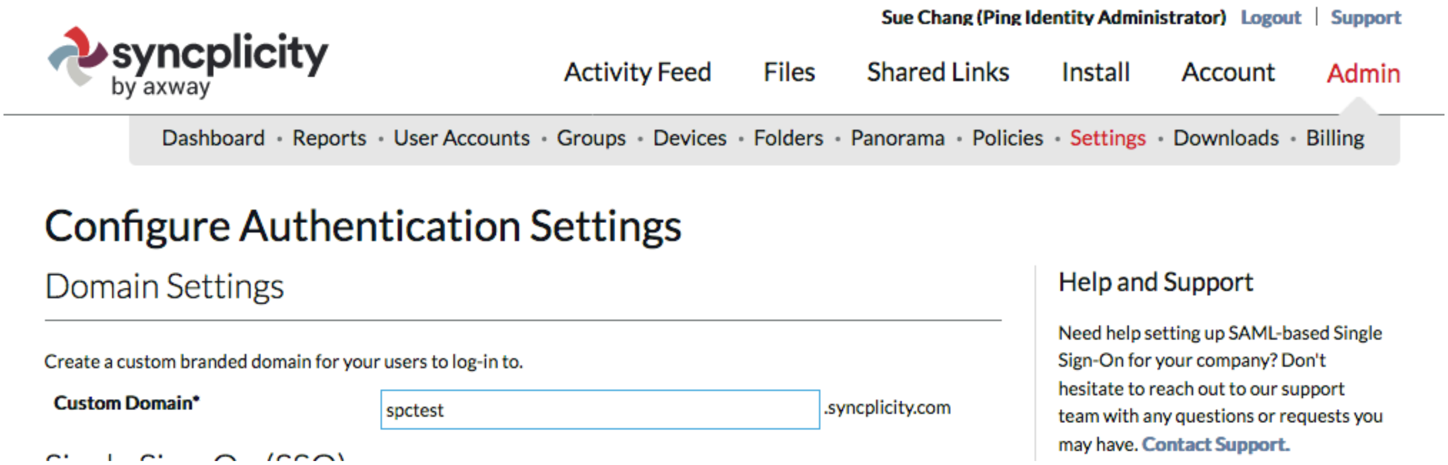
○ DO NOT USE A TEMPLATE FOR THIS CONNECTION

● USE A TEMPLATE FOR THIS CONNECTION

CONNECTION TEMPLATE: Syncplicity SaaS Connection with Provisioning

THE CUSTOM PART OF THE HOST NAME: HTTPS://{THISPART}.SYNCPLICITY.COM

c. For **the custom part of the hostname**, you'll have to get the value by completing the following steps:
   i. Login to Syncplicity with your administrator account*:* https://my.syncplicity.com/LoginWithPassword.aspx
   ii. Go to admin >> settings >> Custom domain and single sign-on.
   iii. Copy the **Custom Domain** value and paste it into PingFederate (if the Syncplicity field was empty, you may set it at this time and click Save Changes in Syncplicity so we can proceed).


syncplicity by axway
Sue Chang (Ping Identity Administrator) Logout | Support
Activity Feed
Files
Shared Links
Install
Account
Admin
Dashboard • Reports • User Accounts • Groups • Devices • Folders • Panorama • Policies • Settings • Downloads • Billing

Configure Authentication Settings

Domain Settings

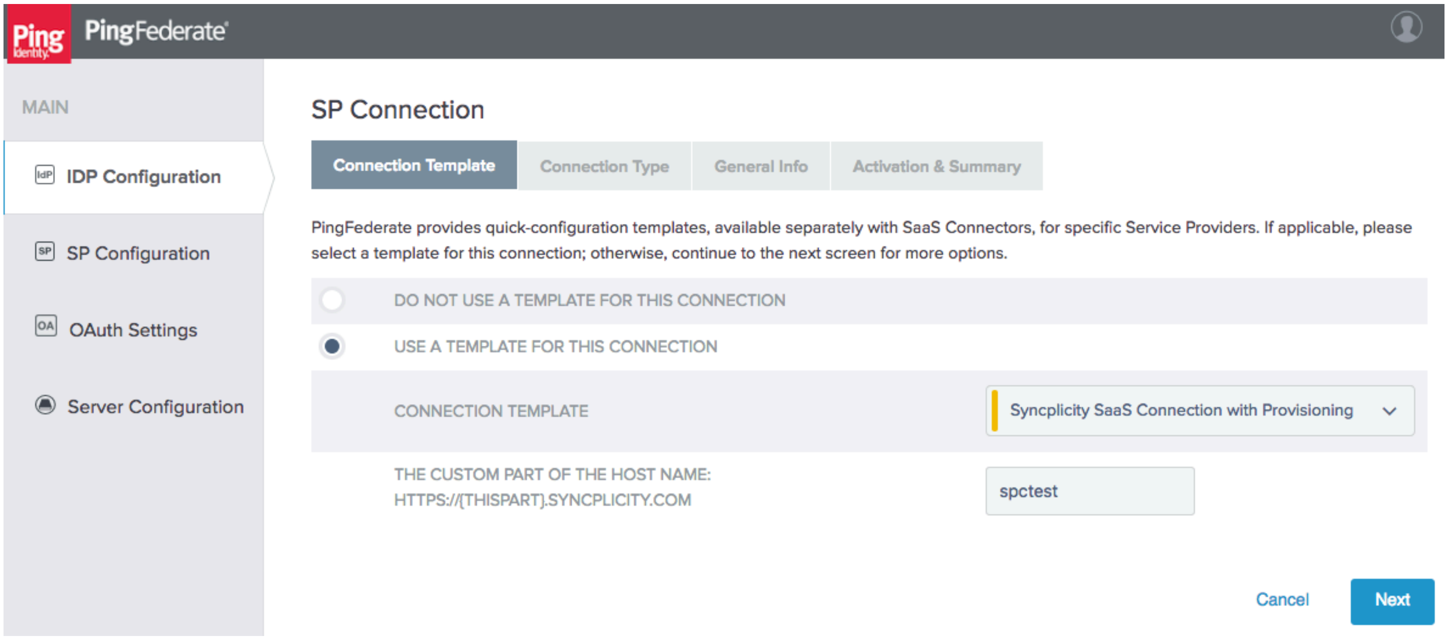
Create a custom branded domain for your users to log-in to.

Custom Domain* spctest .syncplicity.com

Help and Support

Need help setting up SAML-based Single Sign-On for your company? Don't hesitate to reach out to our support team with any questions or requests you may have. Contact Support.

d. Click **Next**.



SP Connection

PingFederate provides quick-configuration templates, available separately with SaaS Connectors, for specific Service Providers. If applicable, please select a template for this connection; otherwise, continue to the next screen for more options.

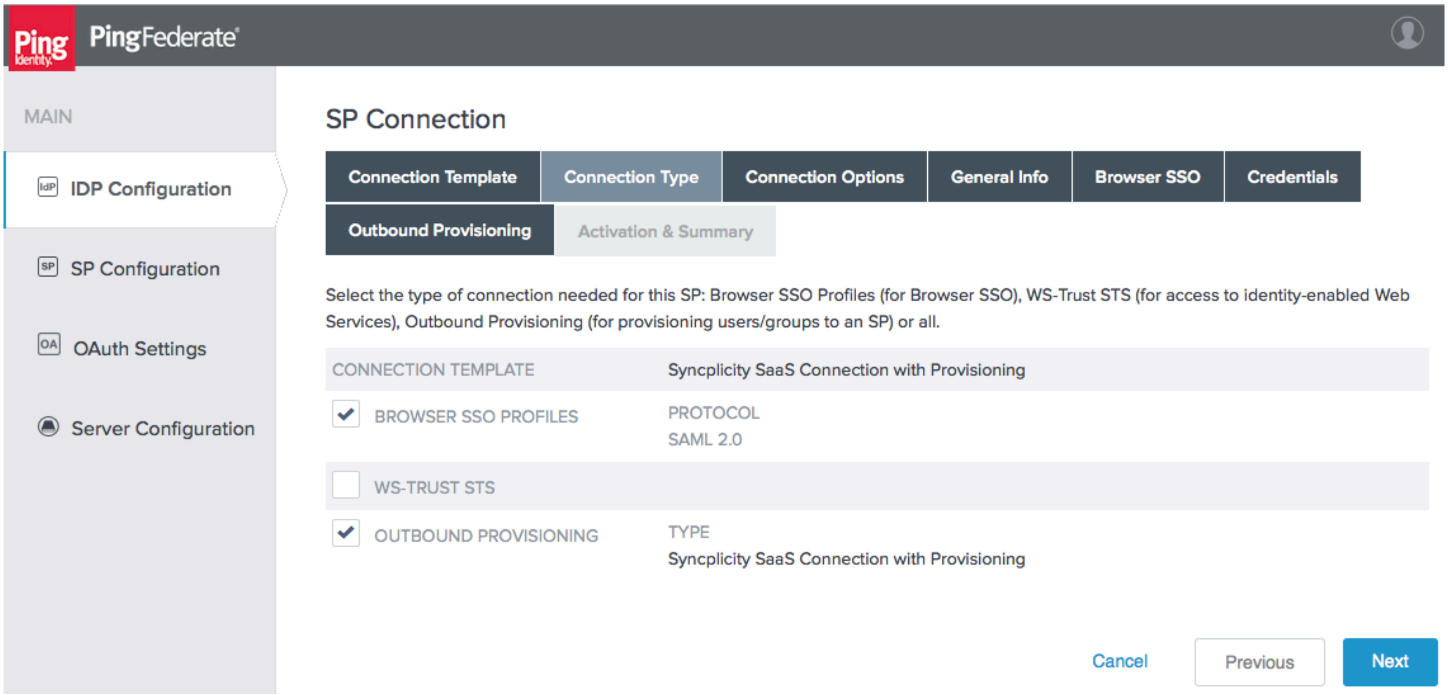
○ DO NOT USE A TEMPLATE FOR THIS CONNECTION

● USE A TEMPLATE FOR THIS CONNECTION

CONNECTION TEMPLATE: Syncplicity SaaS Connection with Provisioning

THE CUSTOM PART OF THE HOST NAME: HTTPS://[THISPART].SYNCPLICITY.COM: spctest

4. On the **Connection Type** page, leave Browser SSO Profiles and Outbound Provisioning enabled and click **Next**.



SP Connection

Select the type of connection needed for this SP: Browser SSO Profiles (for Browser SSO), WS-Trust STS (for access to identity-enabled Web Services), Outbound Provisioning (for provisioning users/groups to an SP) or all.

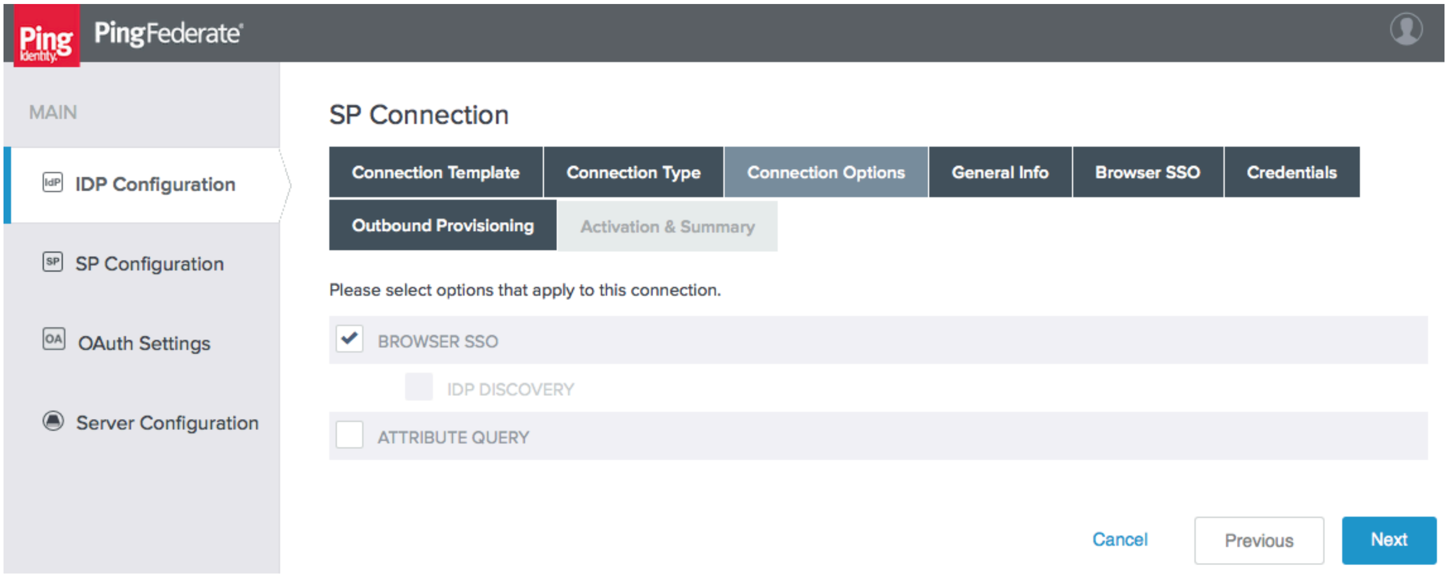
CONNECTION TEMPLATE: Syncplicity SaaS Connection with Provisioning

☑ BROWSER SSO PROFILES — PROTOCOL: SAML 2.0

☐ WS-TRUST STS

☑ OUTBOUND PROVISIONING — TYPE: Syncplicity SaaS Connection with Provisioning

5. On the **Connection Options** page, leave Browser SSO enabled, click **Next**.



SP Connection

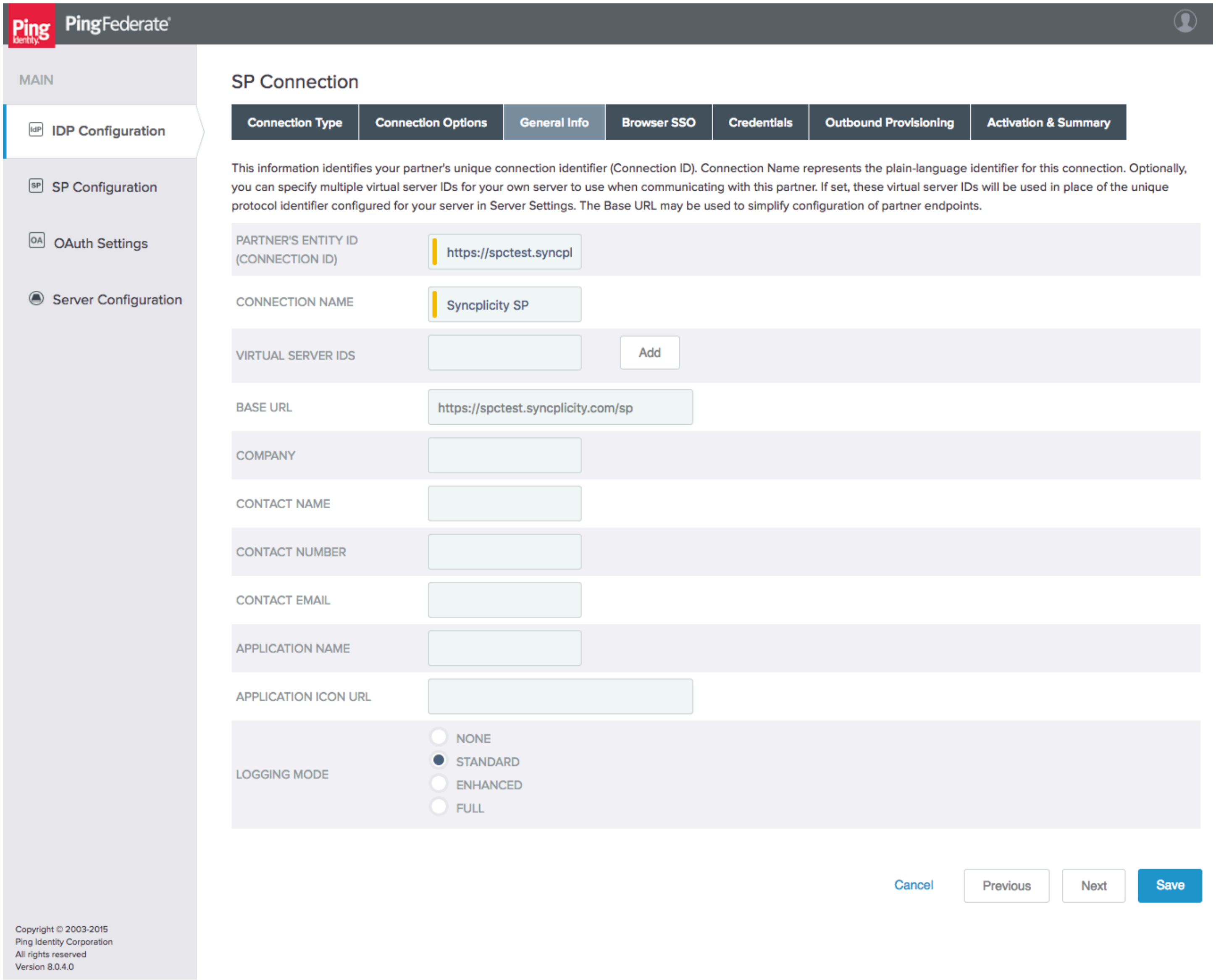
Please select options that apply to this connection.

☑ BROWSER SSO

□ IDP DISCOVERY

□ ATTRIBUTE QUERY

6. On the **General Info** page
    a. Notice your Entity ID should have been pre-populated, copy that value and paste it as the **Connection Name** as well.
    b. The BaseURL should have also been pre-populated.
    c. Click **Next**.



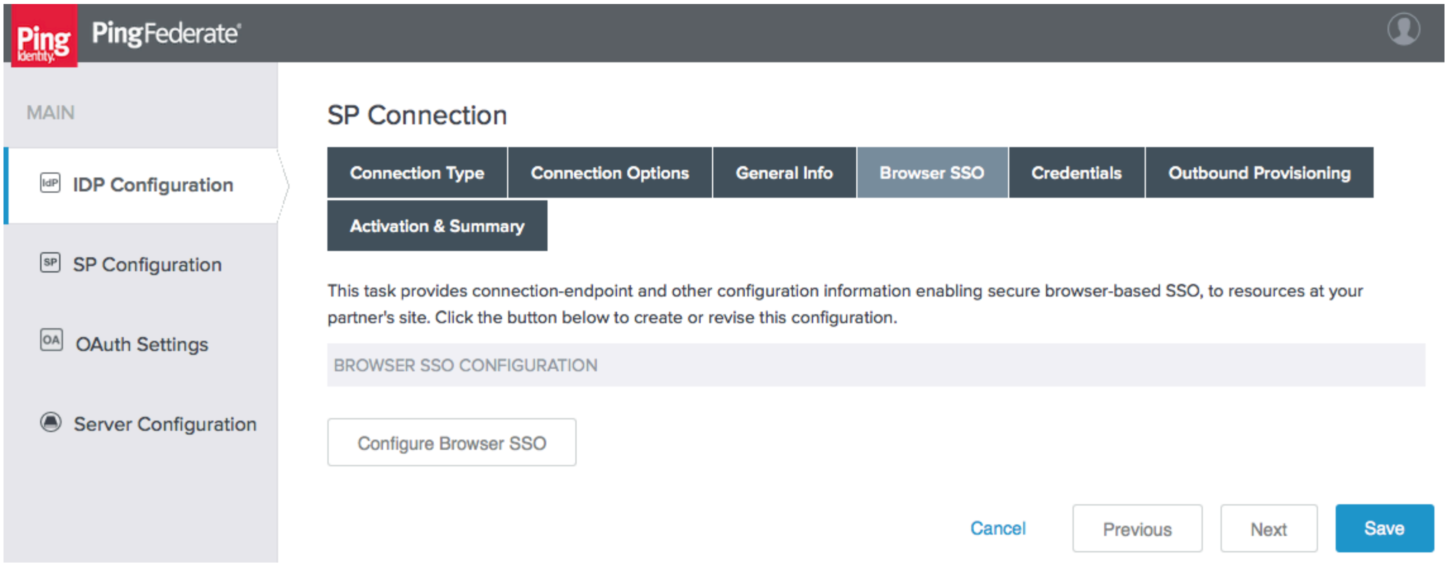
SP Connection

This information identifies your partner's unique connection identifier (Connection ID). Connection Name represents the plain-language identifier for this connection. Optionally, you can specify multiple virtual server IDs for your own server to use when communicating with this partner. If set, these virtual server IDs will be used in place of the unique protocol identifier configured for your server in Server Settings. The Base URL may be used to simplify configuration of partner endpoints.

PARTNER'S ENTITY ID (CONNECTION ID)	https://spctest.syncpl
CONNECTION NAME	Syncplicity SP
VIRTUAL SERVER IDS	
BASE URL	https://spctest.syncplicity.com/sp
COMPANY	
CONTACT NAME	
CONTACT NUMBER	
CONTACT EMAIL	
APPLICATION NAME	
APPLICATION ICON URL	
LOGGING MODE	○ NONE ● STANDARD ○ ENHANCED ○ FULL

Copyright © 2003-2015
Ping Identity Corporation
All rights reserved
Version 8.0.4.0

7. On the **Browser SSO** page, click **Configure Browser SSO**.



SP Connection

This task provides connection-endpoint and other configuration information enabling secure browser-based SSO, to resources at your partner's site. Click the button below to create or revise this configuration.

BROWSER SSO CONFIGURATION

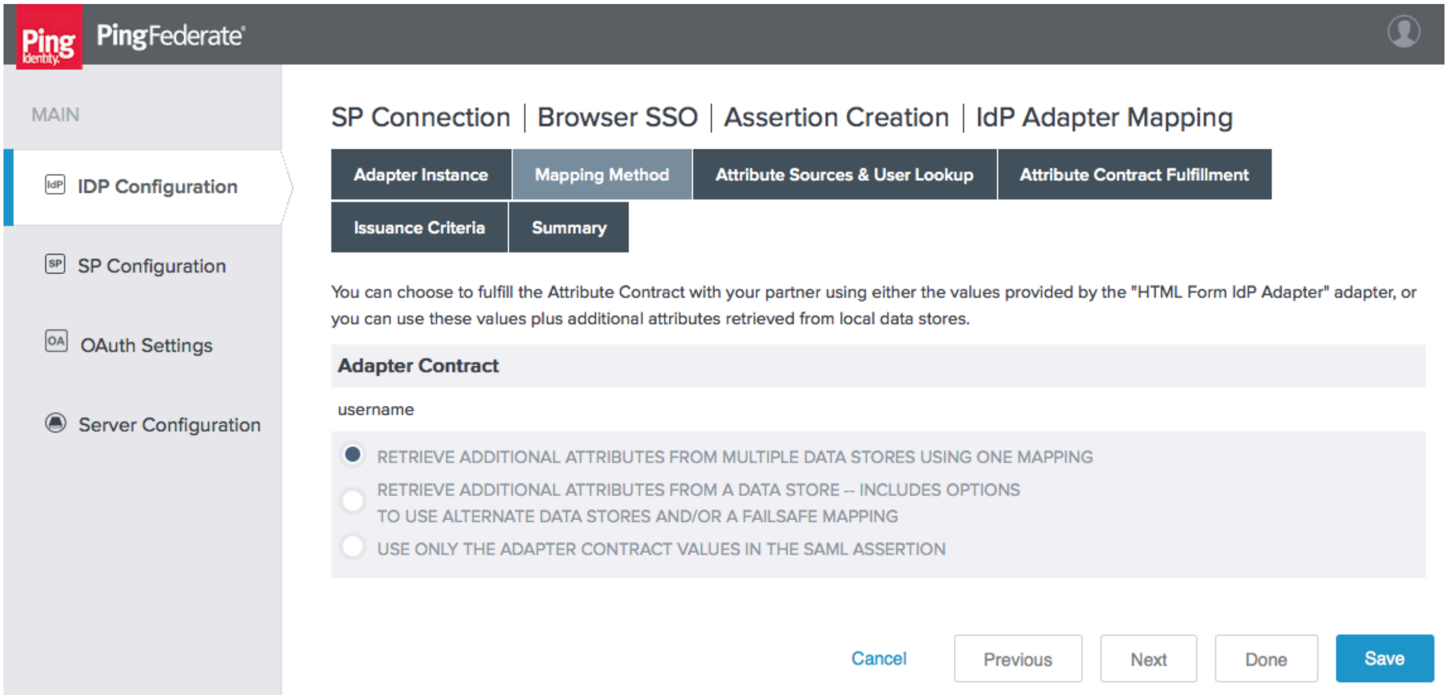
8. On **Assertion Creation** page, click **Configure Assertion Creation**.

9. On **IdP Adapter Mapping**, click **Map New Adapter Instance…**

10. Select an **existing IDP Adapter** (like HTML Form) and click **Next**.
    a. If you have not created an IDP Adapter yet, you will have to follow the prompts to create one before being able to proceed.

11. On the **Assertion Mapping** page..
    a. Select "**Use only the Adapter Contract values in the SAML assertion**" if your IDP Adapter already contains the Syncplicity username. Click **Next**.

    b. Otherwise, select "**Retrieve additional attributes from multiple data stores using one mapping**". Click **Next**.
        i. In this example we will use this option because our IDP Adapter is only giving us username=sAMAccountName and we want to use the mail attribute as the Syncplicity username.



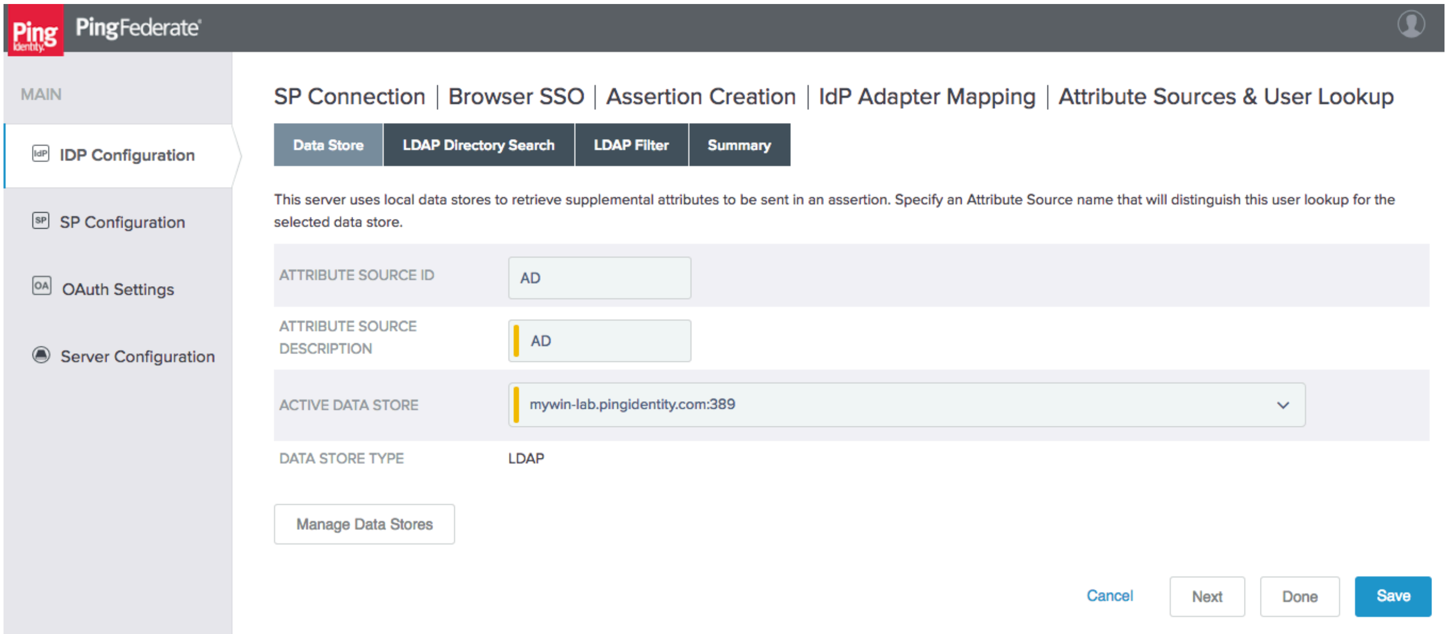
You can choose to fulfill the Attribute Contract with your partner using either the values provided by the "HTML Form IdP Adapter" adapter, or you can use these values plus additional attributes retrieved from local data stores.

Adapter Contract

username

● RETRIEVE ADDITIONAL ATTRIBUTES FROM MULTIPLE DATA STORES USING ONE MAPPING
○ RETRIEVE ADDITIONAL ATTRIBUTES FROM A DATA STORE -- INCLUDES OPTIONS TO USE ALTERNATE DATA STORES AND/OR A FAILSAFE MAPPING
○ USE ONLY THE ADAPTER CONTRACT VALUES IN THE SAML ASSERTION

        ii. On the **Attribute Sources & User Lookup** page, click **Add Attribute Source…**
        iii. On the **Data Store** page, provide an Attribute Source Id, Attribute Source Description, select the relevant Data Store and click **Next**.

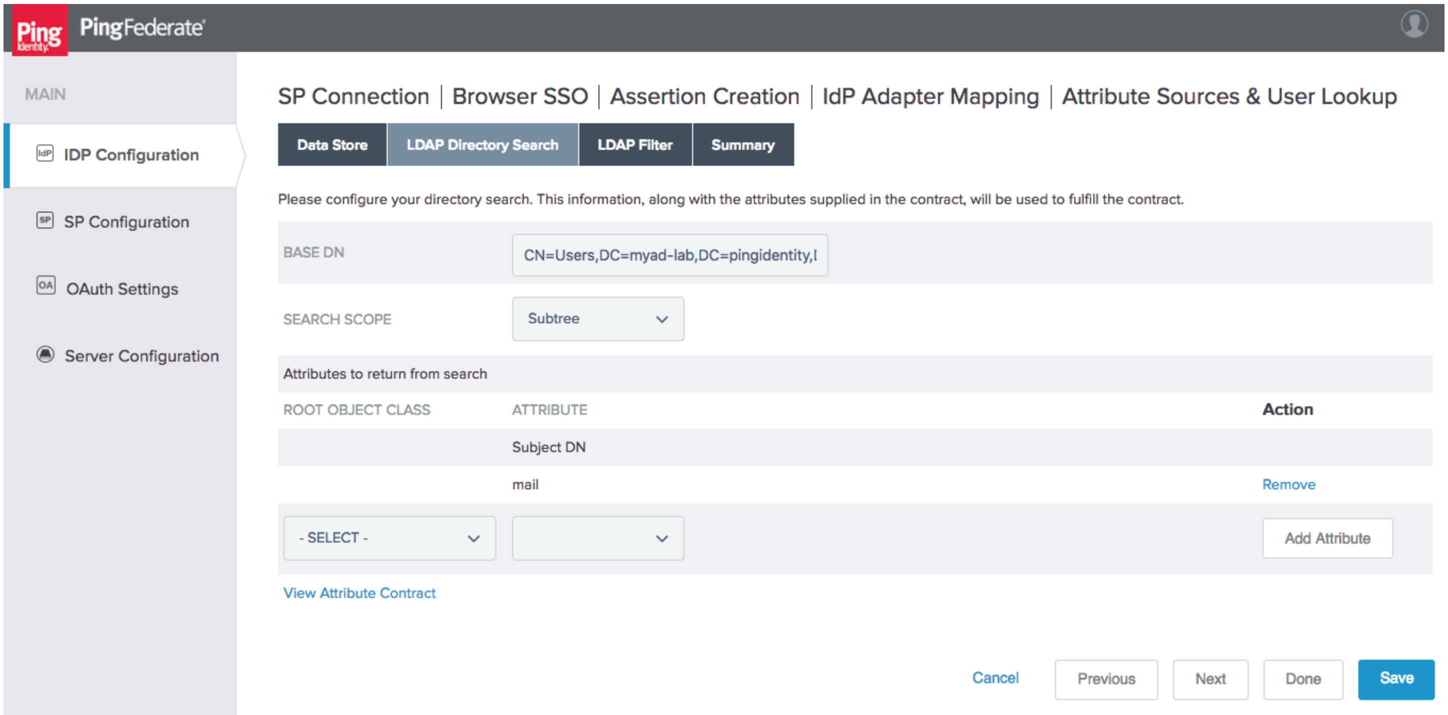


This server uses local data stores to retrieve supplemental attributes to be sent in an assertion. Specify an Attribute Source name that will distinguish this user lookup for the selected data store.

ATTRIBUTE SOURCE ID	AD
ATTRIBUTE SOURCE DESCRIPTION	AD
ACTIVE DATA STORE	mywin-lab.pingidentity.com:389
DATA STORE TYPE	LDAP

iv. On the **LDAP Directory Search** page, fill in the Base DN and LDAP attributes you want to use, then click **Next**.

- In this example we wanted to use the mail attribute, so to add it you must select <Show All Attributes> under ROOT OBJECT CLASS, and then under ATTRIBUTE you should be able to select "mail" from the list. Once the attribute is selected click Add Attribute to add it.



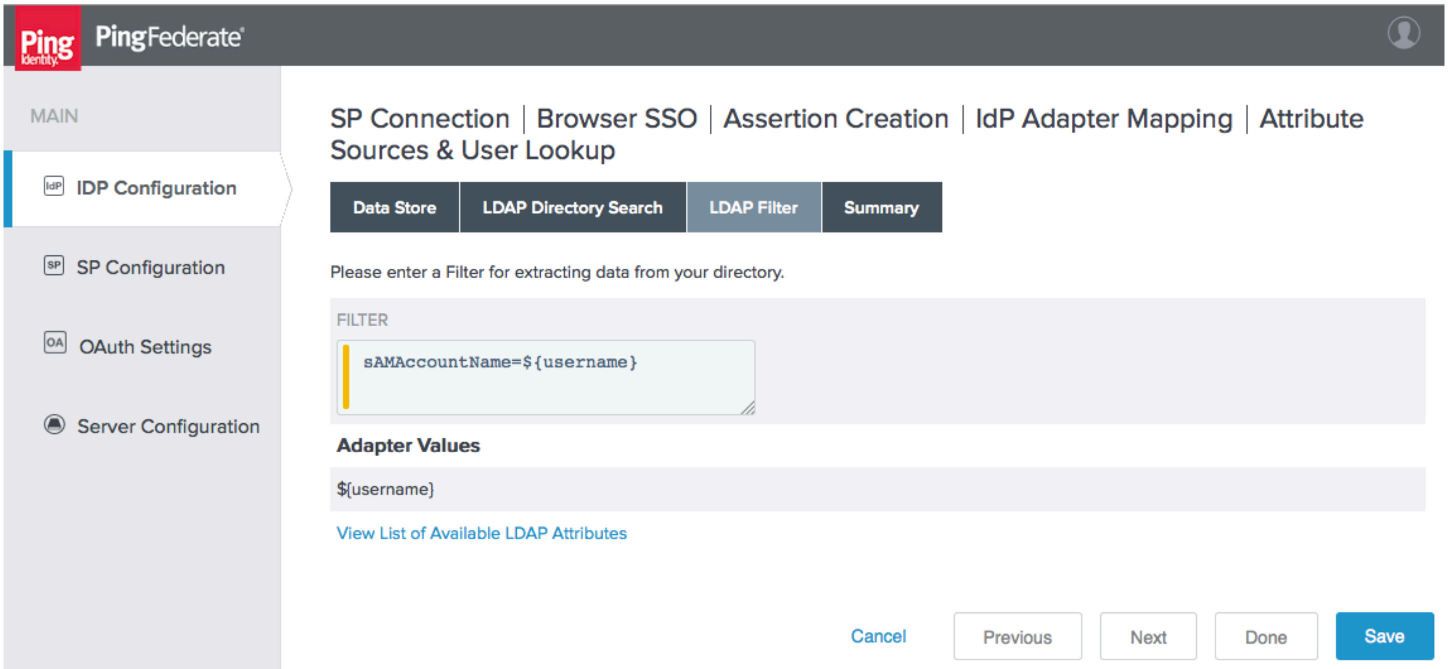
Please configure your directory search. This information, along with the attributes supplied in the contract, will be used to fulfill the contract.

BASE DN	CN=Users,DC=myad-lab,DC=pingidentity,I
SEARCH SCOPE	Subtree

Attributes to return from search

ROOT OBJECT CLASS	ATTRIBUTE	Action
	Subject DN	
	mail	Remove

v. On the **LDAP Filter** page, fill in the Filter field and click **Next**.



SP Connection | Browser SSO | Assertion Creation | IdP Adapter Mapping | Attribute Sources & User Lookup

Please enter a Filter for extracting data from your directory.

FILTER

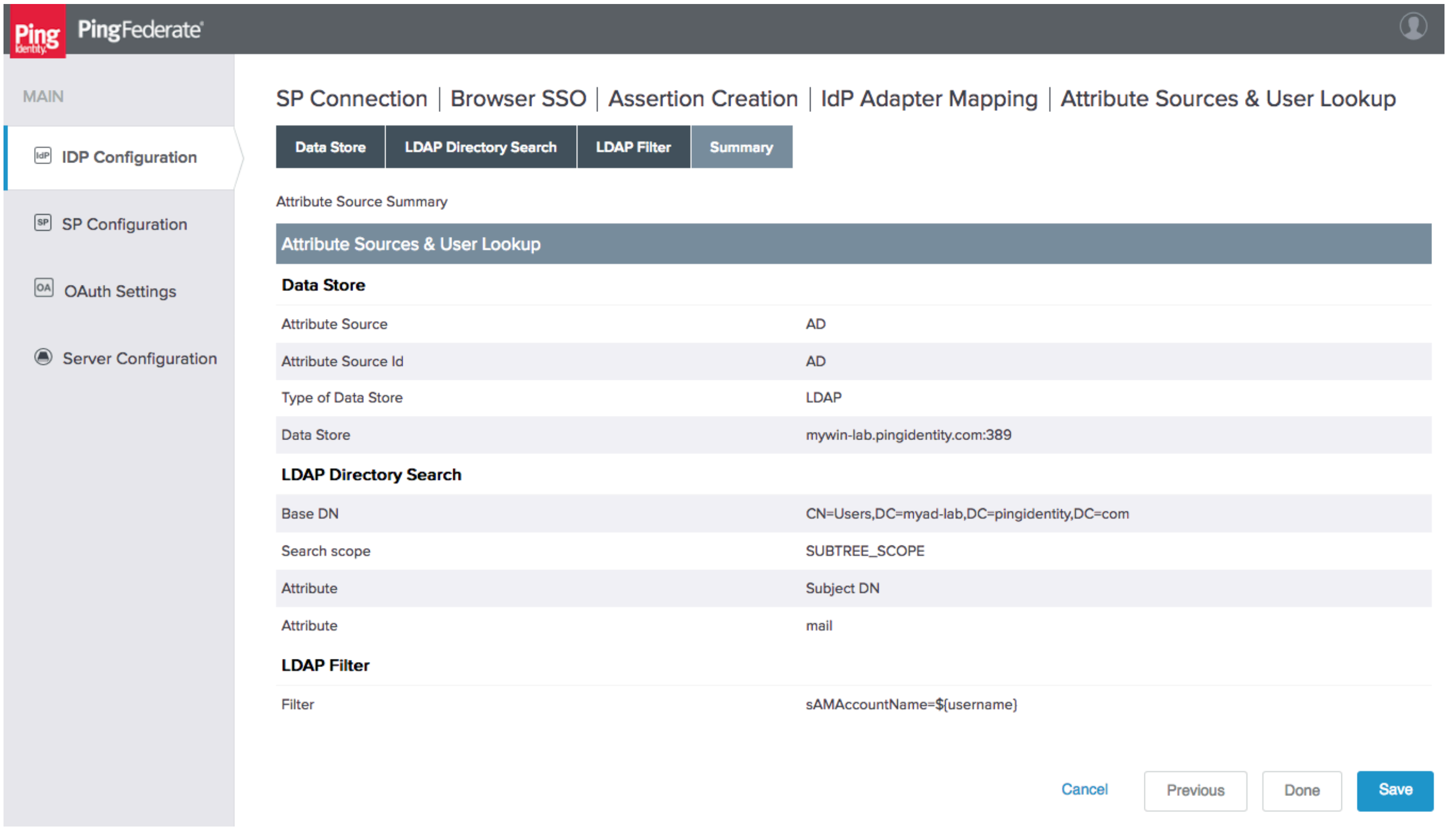
```
sAMAccountName=${username}
```

Adapter Values

${username}

View List of Available LDAP Attributes

vi. On the **Summary** page, click **Done**.

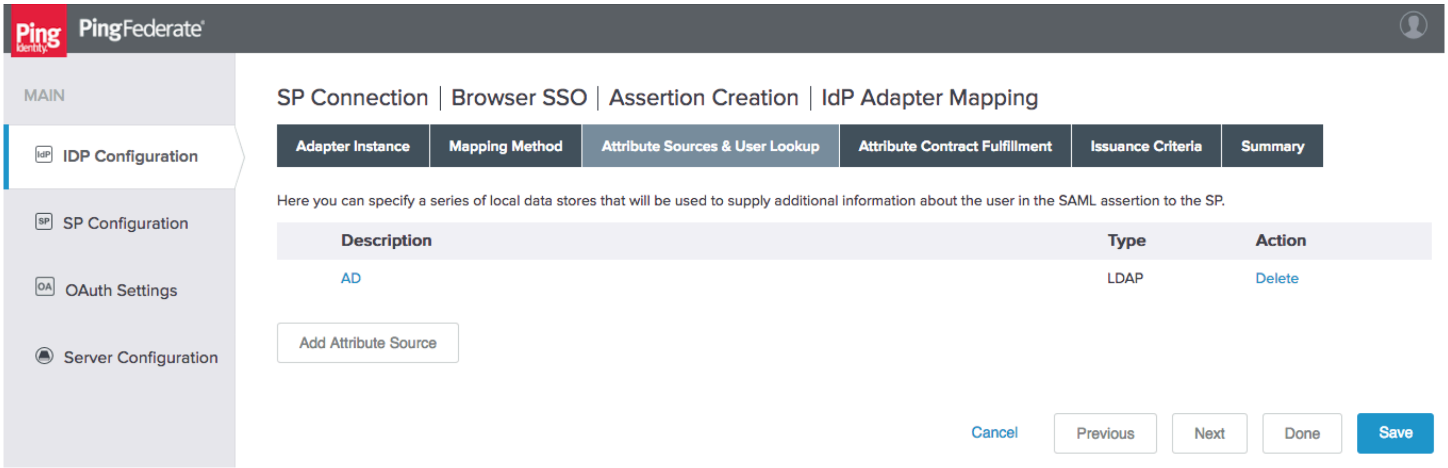


SP Connection | Browser SSO | Assertion Creation | IdP Adapter Mapping | Attribute Sources & User Lookup

Attribute Source Summary

Attribute Sources & User Lookup	
Data Store	
Attribute Source	AD
Attribute Source Id	AD
Type of Data Store	LDAP
Data Store	mywin-lab.pingidentity.com:389
LDAP Directory Search	
Base DN	CN=Users,DC=myad-lab,DC=pingidentity,DC=com
Search scope	SUBTREE_SCOPE
Attribute	Subject DN
Attribute	mail
LDAP Filter	
Filter	sAMAccountName=${username}

vii. On the **Attribute Sources & User Lookup** page, click **Next**.



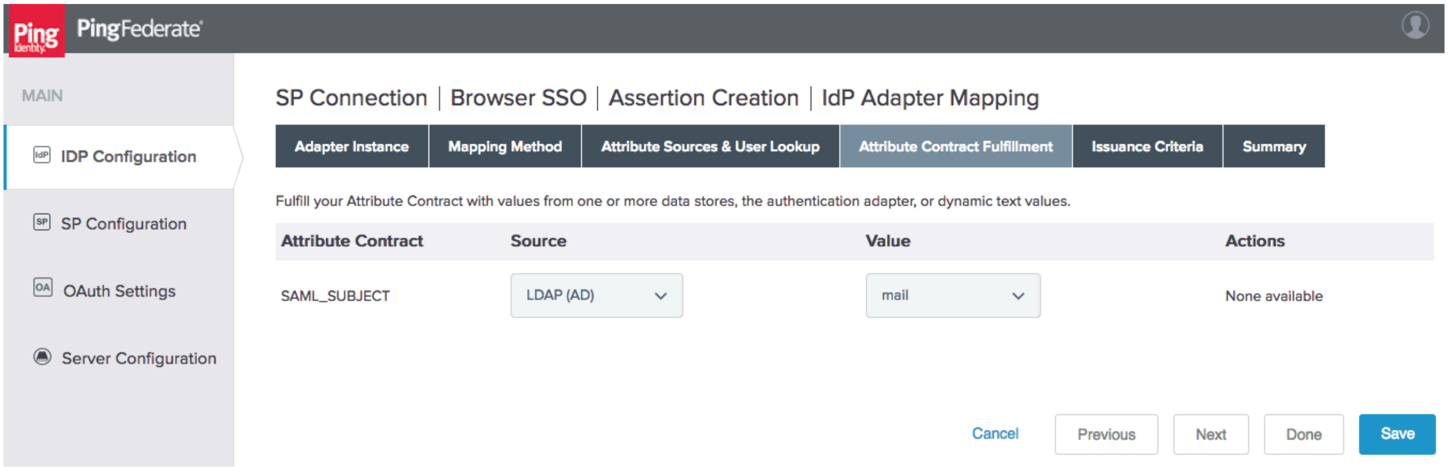
Here you can specify a series of local data stores that will be used to supply additional information about the user in the SAML assertion to the SP.

Description	Type	Action
AD	LDAP	Delete

12. On the **Attribute Contract Fulfillment** page, for SAML_SUBJECT

a. Select the relevant Source and Value for the Syncplicity username

b. Click **Next**.

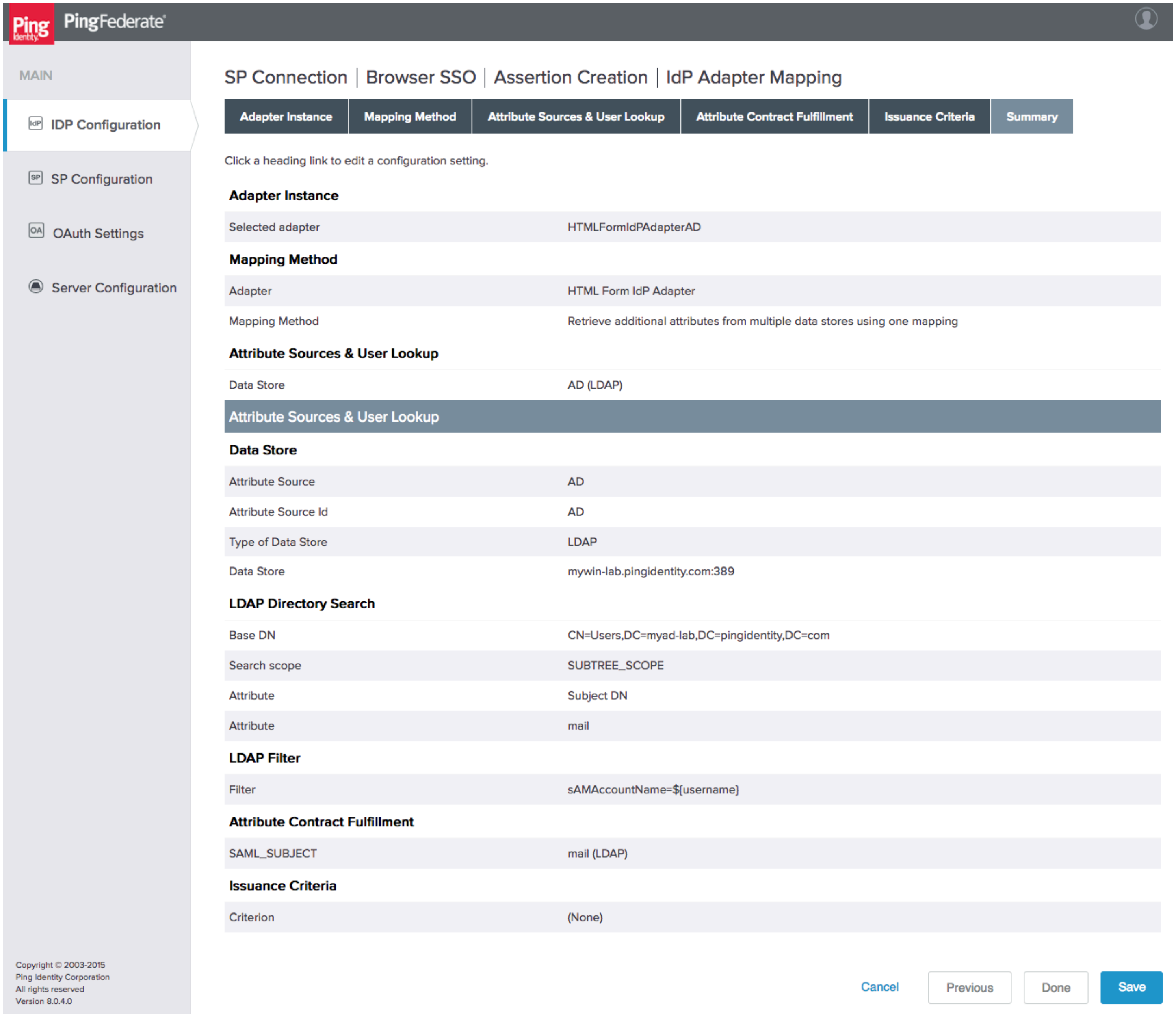


Fulfill your Attribute Contract with values from one or more data stores, the authentication adapter, or dynamic text values.

Attribute Contract	Source	Value	Actions
SAML_SUBJECT	LDAP (AD)	mail	None available

13. On the **Issuance Criteria** page, click **Next**.

14. On the Summary page, click **Done**.



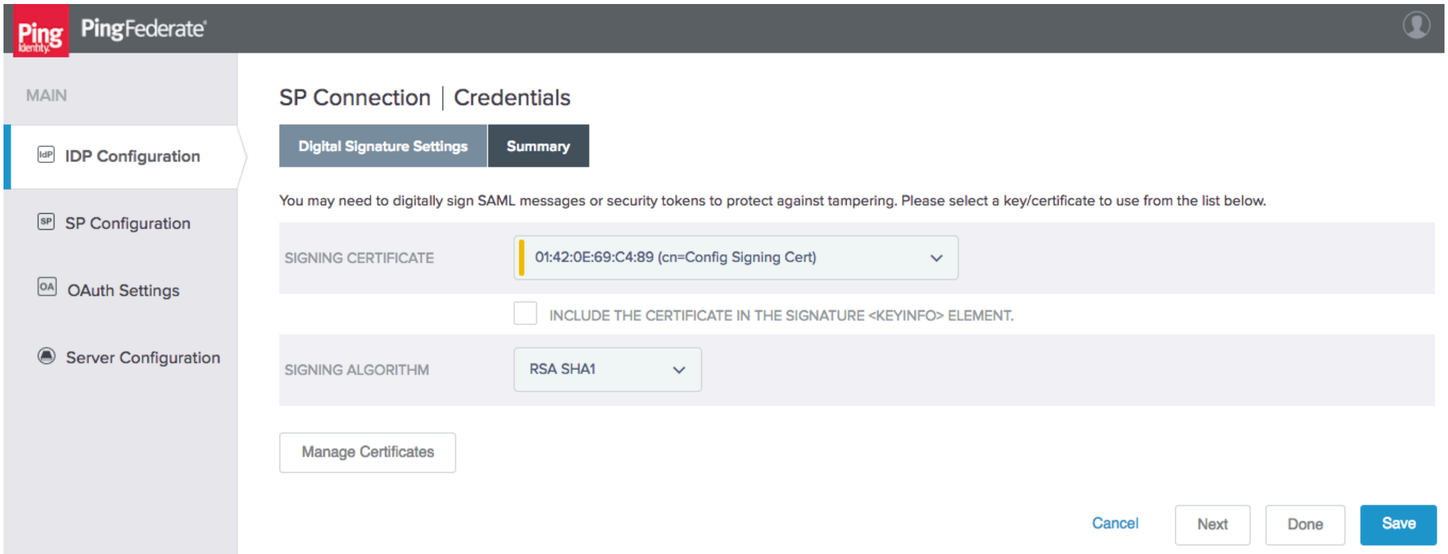
SP Connection | Browser SSO | Assertion Creation | IdP Adapter Mapping

Adapter Instance
Mapping Method
Attribute Sources & User Lookup
Attribute Contract Fulfillment
Issuance Criteria
Summary

Click a heading link to edit a configuration setting.

Adapter Instance	
Selected adapter	HTMLFormIdPAdapterAD
Mapping Method	
Adapter	HTML Form IdP Adapter
Mapping Method	Retrieve additional attributes from multiple data stores using one mapping
Attribute Sources & User Lookup	
Data Store	AD (LDAP)
Attribute Sources & User Lookup	
Data Store	
Attribute Source	AD
Attribute Source Id	AD
Type of Data Store	LDAP
Data Store	mywin-lab.pingidentity.com:389
LDAP Directory Search	
Base DN	CN=Users,DC=myad-lab,DC=pingidentity,DC=com
Search scope	SUBTREE_SCOPE
Attribute	Subject DN
Attribute	mail
LDAP Filter	
Filter	sAMAccountName=${username}
Attribute Contract Fulfillment	
SAML_SUBJECT	mail (LDAP)
Issuance Criteria	
Criterion	(None)

15. On the IdP Adapter Mapping page click **Done**.
16. On the Assertion Creation page, click **Done**.
17. On the Browser SSO page, click **Next**.
18. On the Credentials page, click **Configure Credentials**.
19. Select the **Signing Certificate** which your IDP will use, and click **Done**.



SP Connection | Credentials

You may need to digitally sign SAML messages or security tokens to protect against tampering. Please select a key/certificate to use from the list below.

SIGNING CERTIFICATE: 01:42:0E:69:C4:89 (cn=Config Signing Cert)

□ INCLUDE THE CERTIFICATE IN THE SIGNATURE <KEYINFO> ELEMENT.

SIGNING ALGORITHM: RSA SHA1

20. On the Credentials page, click **Next**.
21. On the Outbound Provisioning page, click **Configure Provisioning**.
22. On the **Configure Channels >>Target page**

a. Notice the Service URLs are pre-populated for the first 3 fields.

b. For **- Behavior of the Provisioner. When the provisioner believes the user has been removed or disabled -** you have two choices, please select one:

   i. Selecting “Disable User” will result in the user account being disabled in Syncplicity

   ii. Selecting “Delete User” will result in the user account being deleted in Syncplicity

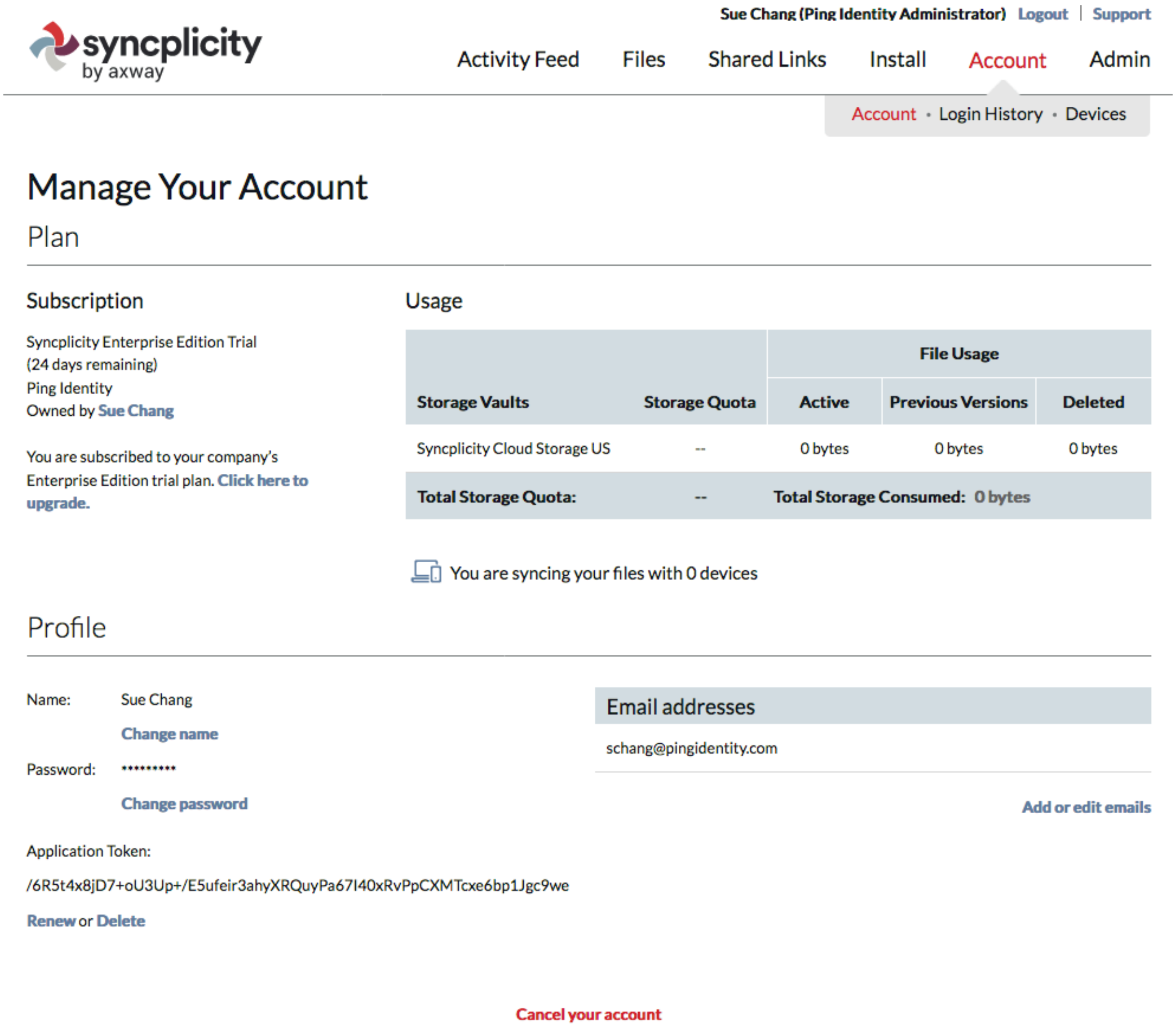
   Note: A deprovisioning event is triggered by removing the user from the target provisioning group or by disabling their account after they were already a member of the provisioning group.

c. For **- The Application Token for performing Authentication -** copy the value from Syncplicity:

   i. Login to Syncplicity with your administrator account*:* *https://my.syncplicity.com/LoginWithPassword.aspx*

   ii. Go to the Account >> Account >> Application Token (bottom left-hand corner)

   iii. If you don’t already have an Application Token listed there, click Create to create one, then copy that value. If the Application Token is not visible on the Account page, please contact Syncplicity Support: support@syncplicity.com


syncplicity
by axway
Sue Chang (Ping Identity Administrator) Logout | Support
Activity Feed
Files
Shared Links
Install
Account
Admin
Account • Login History • Devices

Manage Your Account

Plan

Subscription

Syncplicity Enterprise Edition Trial
(24 days remaining)
Ping Identity
Owned by Sue Chang

You are subscribed to your company's Enterprise Edition trial plan. Click here to upgrade.

Usage

		File Usage		
Storage Vaults	**Storage Quota**	**Active**	**Previous Versions**	**Deleted**
Syncplicity Cloud Storage US	--	0 bytes	0 bytes	0 bytes
Total Storage Quota:	--	**Total Storage Consumed: 0 bytes**		

You are syncing your files with 0 devices

Profile

Name: Sue Chang
Change name

Password: •••••••••
Change password

Email addresses

schang@pingidentity.com

Add or edit emails

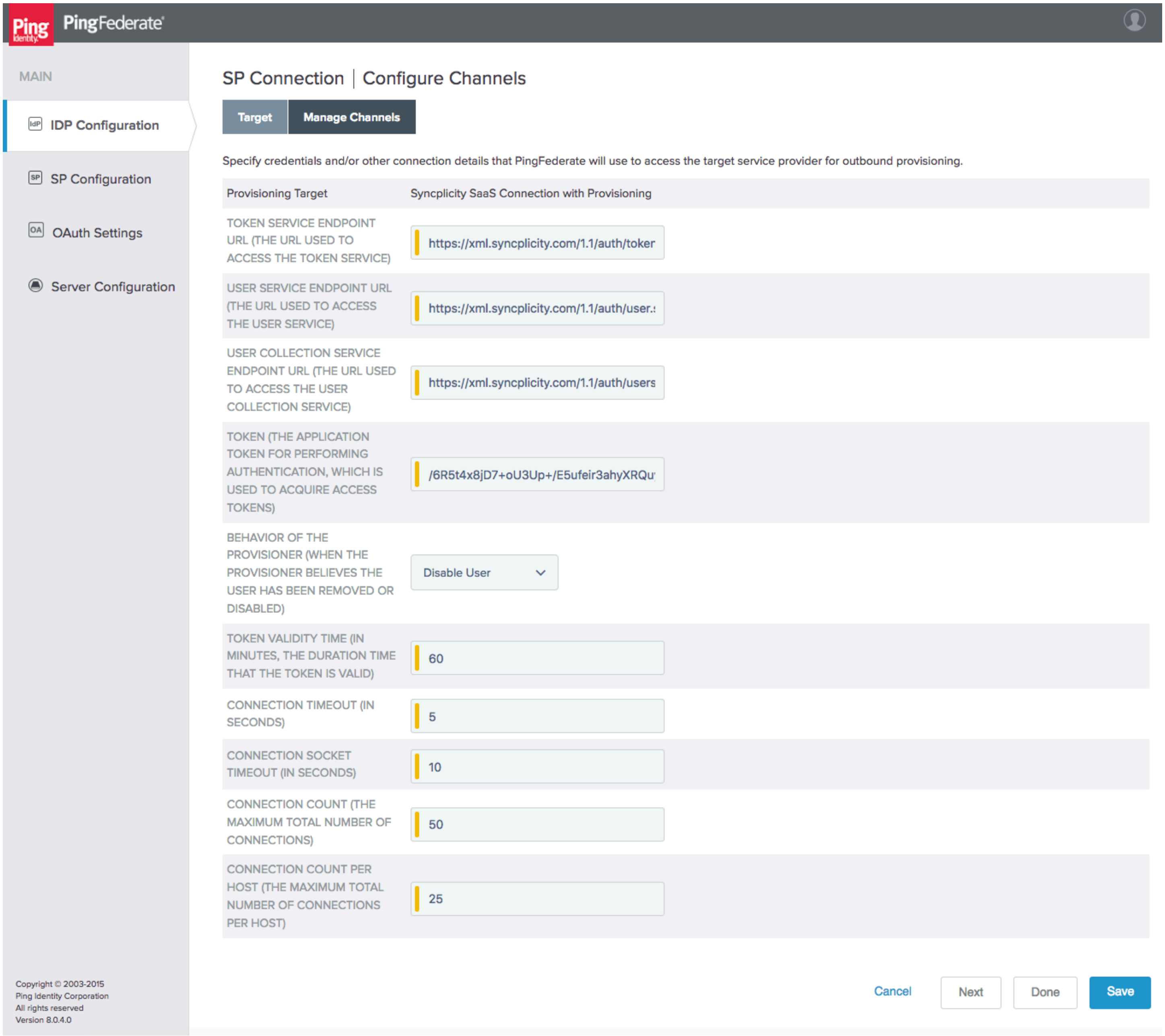
Application Token:

/6R5t4x8jD7+oU3Up+/E5ufeir3ahyXRQuyPa67l40xRvPpCXMTcxe6bp1Jgc9we

Renew or Delete

Cancel your account

d. Click **Next** when you are done.



SP Connection | Configure Channels

Specify credentials and/or connection details that PingFederate will use to access the target service provider for outbound provisioning.

Provisioning Target	Syncplicity SaaS Connection with Provisioning
TOKEN SERVICE ENDPOINT URL (THE URL USED TO ACCESS THE TOKEN SERVICE)	https://xml.syncplicity.com/1.1/auth/toker
USER SERVICE ENDPOINT URL (THE URL USED TO ACCESS THE USER SERVICE)	https://xml.syncplicity.com/1.1/auth/user.:
USER COLLECTION SERVICE ENDPOINT URL (THE URL USED TO ACCESS THE USER COLLECTION SERVICE)	https://xml.syncplicity.com/1.1/auth/users
TOKEN (THE APPLICATION TOKEN FOR PERFORMING AUTHENTICATION, WHICH IS USED TO ACQUIRE ACCESS TOKENS)	/6R5t4x8jD7+oU3Up+/E5ufeir3ahyXRQu'
BEHAVIOR OF THE PROVISIONER (WHEN THE PROVISIONER BELIEVES THE USER HAS BEEN REMOVED OR DISABLED)	Disable User
TOKEN VALIDITY TIME (IN MINUTES, THE DURATION TIME THAT THE TOKEN IS VALID)	60
CONNECTION TIMEOUT (IN SECONDS)	5
CONNECTION SOCKET TIMEOUT (IN SECONDS)	10
CONNECTION COUNT (THE MAXIMUM TOTAL NUMBER OF CONNECTIONS)	50
CONNECTION COUNT PER HOST (THE MAXIMUM TOTAL NUMBER OF CONNECTIONS PER HOST)	25

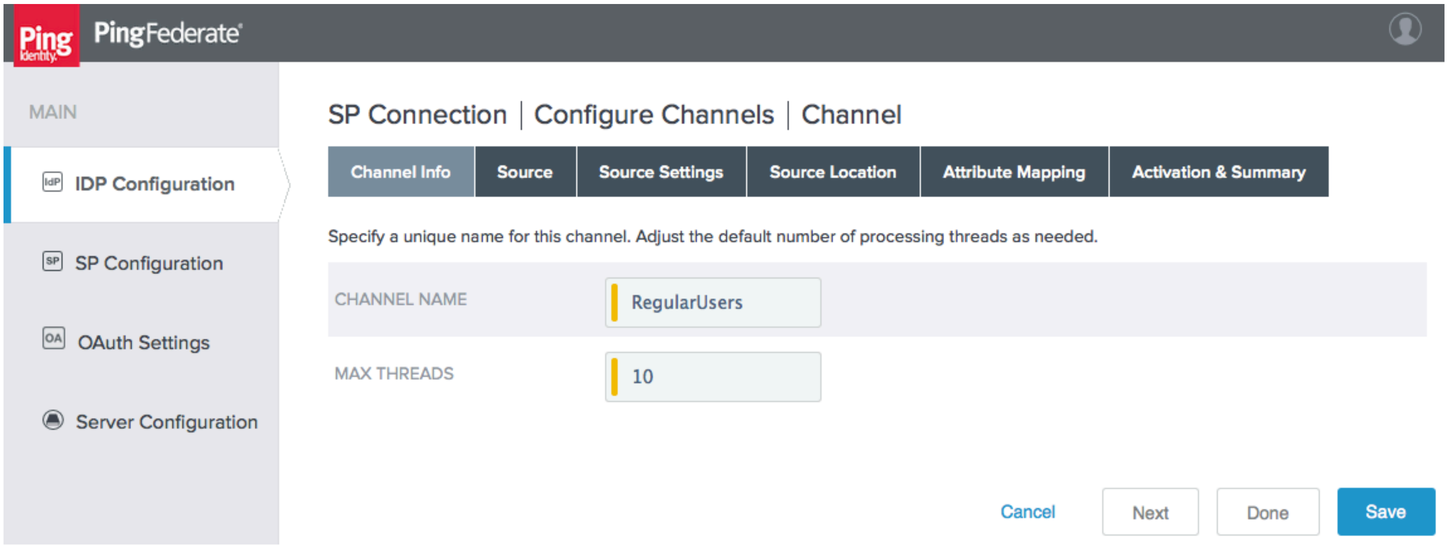
Token Validity Time should be adjusted to how long the token should be valid during the processing of life cycle management events by the provisioner. After the duration validity time has been reached, the provisioner will request a new token to be created from Syncplicity. As of October 2015, the duration of a Syncplicity token is 90 minutes. So, if the **Token Validity Time** has been configured for 60 minutes, the token will expire after 60 minutes, regardless if the Syncplicity token is set to expire in another 30 minutes. Setting the **Token Validity Time** to 0 or more than 90 minutes will automatically default to the Syncplicity token duration time of 90 minutes. The recommended setting for **Token Validity Time** is 60 minutes.

Please also note that the **Connection Count** and **Connection Count Per Host** configuration fields will need to be adjusted to your PingFederate environment (i.e., maximum number of threads the server can handle, etc.).

23. On the **Manage Channels** page, click **Create**...

24. On the **Channel Info** page

a. For **Channel Name** type in a name of your choosing.
b. For **Max Threads**, choose the number of threads you want for that channel.
c. Click **Next**.



SP Connection | Configure Channels | Channel

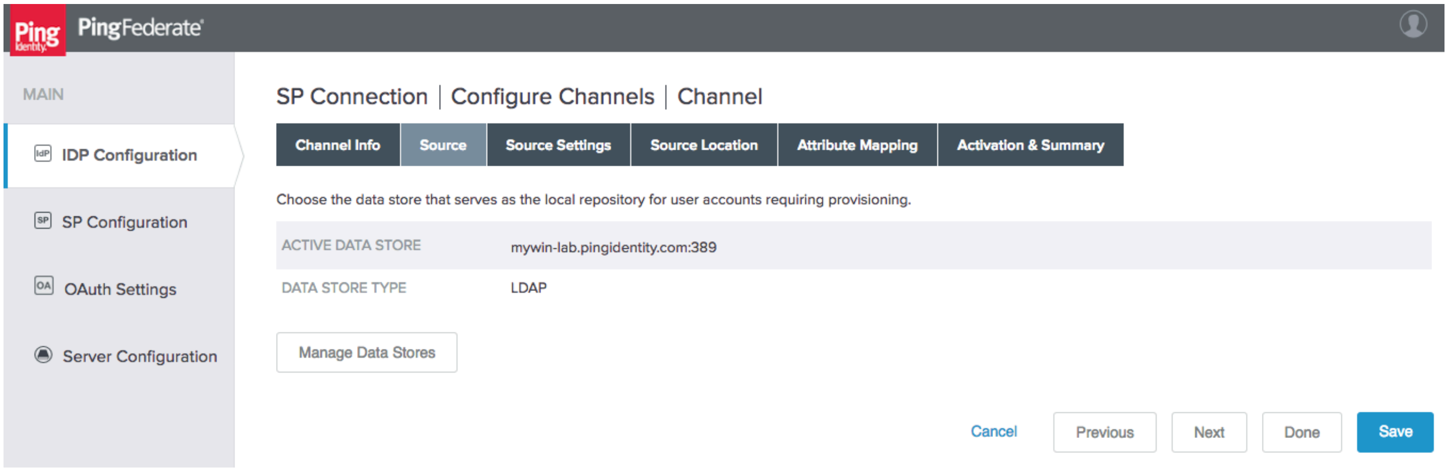
Specify a unique name for this channel. Adjust the default number of processing threads as needed.

CHANNEL NAME: RegularUsers

MAX THREADS: 10

25. On the **Source** page

a. For **Active Data Store** select the Data Store which contains the accounts you want to provision in Syncplicity.
b. Click **Next**.



SP Connection | Configure Channels | Channel

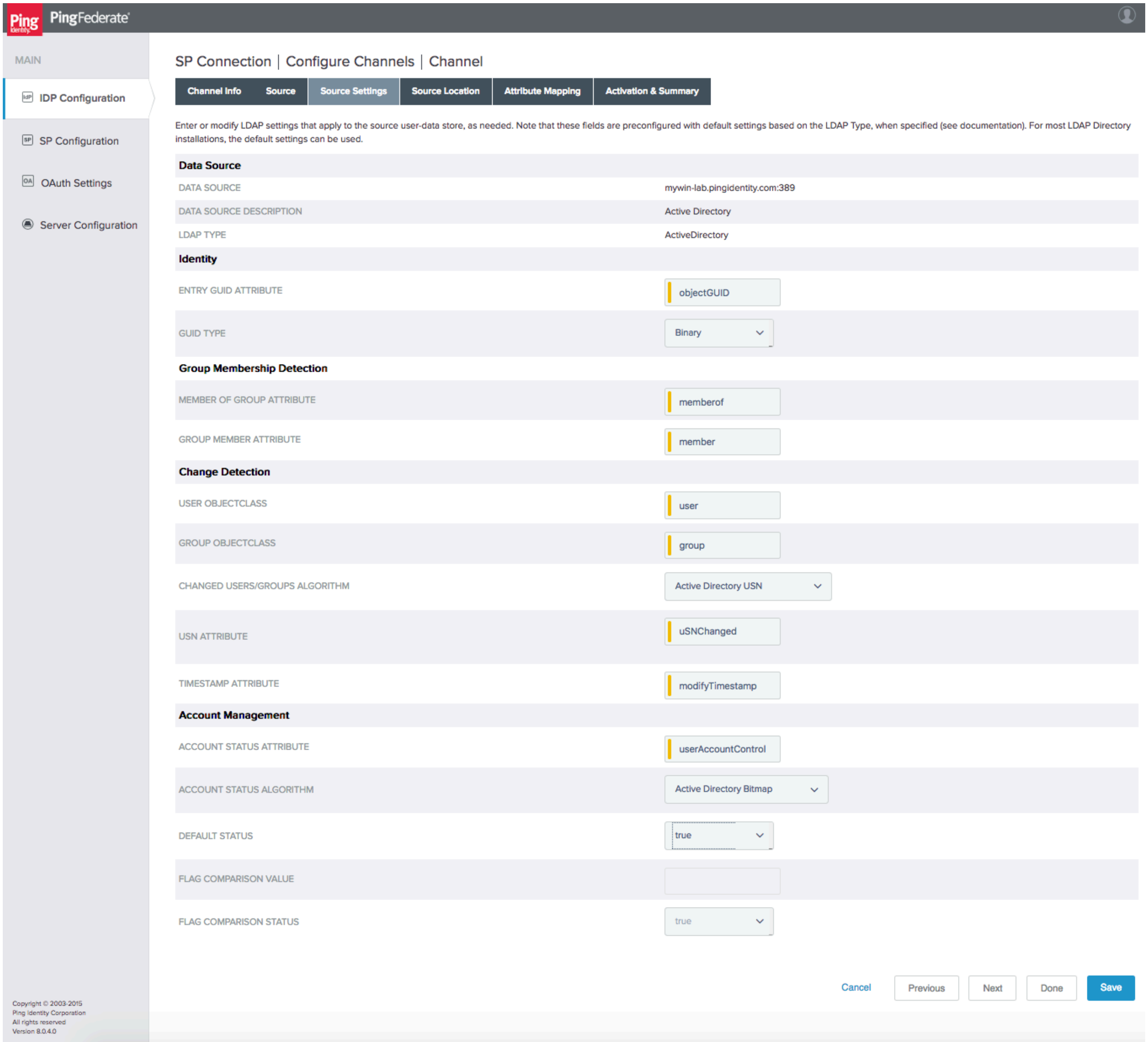
Choose the data store that serves as the local repository for user accounts requiring provisioning.

ACTIVE DATA STORE: mywin-lab.pingidentity.com:389

DATA STORE TYPE: LDAP

26. On the **Source Settings** page

a. Review the configuration and make any edits if needed, then click **Next**.



SP Connection | Configure Channels | Channel

Enter or modify LDAP settings that apply to the source user-data store, as needed. Note that these fields are preconfigured with default settings based on the LDAP Type, when specified (see documentation). For most LDAP Directory installations, the default settings can be used.

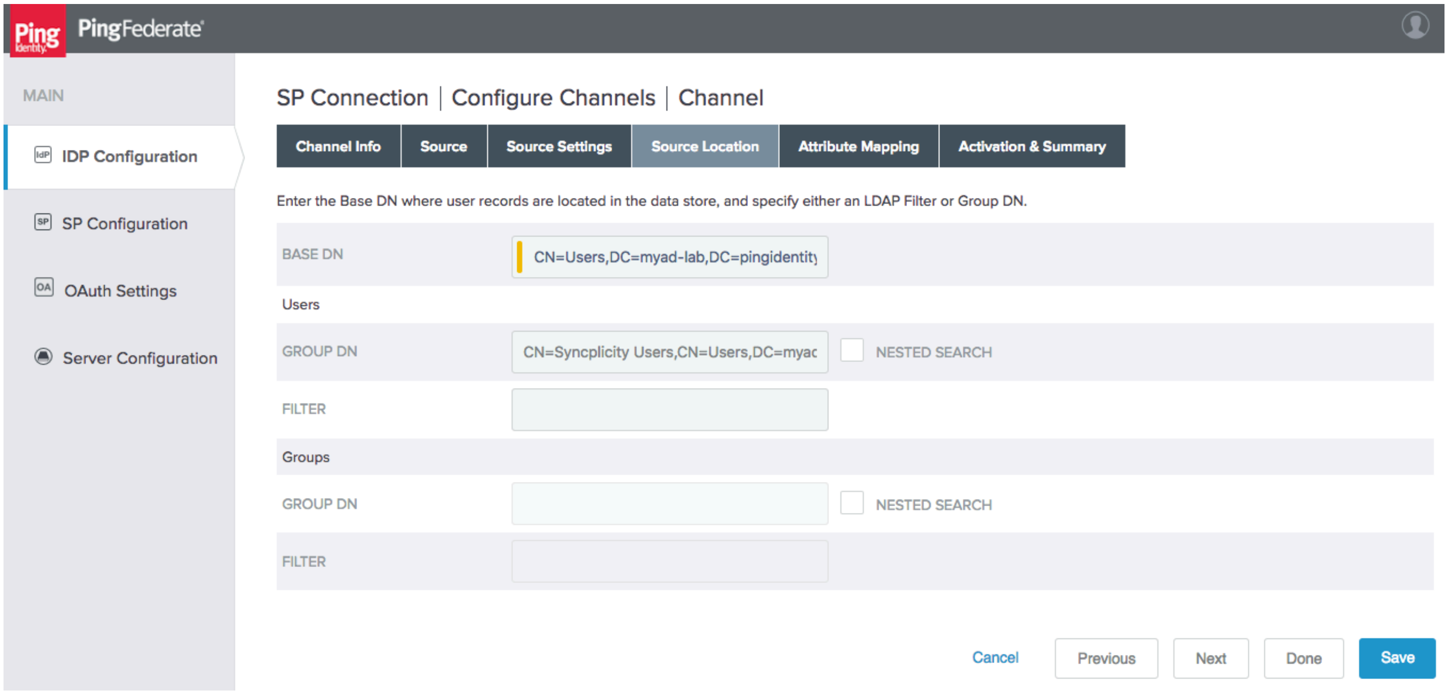
Data Source	
DATA SOURCE	mywin-lab.pingidentity.com:389
DATA SOURCE DESCRIPTION	Active Directory
LDAP TYPE	ActiveDirectory
Identity	
ENTRY GUID ATTRIBUTE	objectGUID
GUID TYPE	Binary
Group Membership Detection	
MEMBER OF GROUP ATTRIBUTE	memberof
GROUP MEMBER ATTRIBUTE	member
Change Detection	
USER OBJECTCLASS	user
GROUP OBJECTCLASS	group
CHANGED USERS/GROUPS ALGORITHM	Active Directory USN
USN ATTRIBUTE	uSNChanged
TIMESTAMP ATTRIBUTE	modifyTimestamp
Account Management	
ACCOUNT STATUS ATTRIBUTE	userAccountControl
ACCOUNT STATUS ALGORITHM	Active Directory Bitmap
DEFAULT STATUS	true
FLAG COMPARISON VALUE	
FLAG COMPARISON STATUS	true

27. On the **Source Location** page

a. Specify the **Base DN** where the user accounts exist.

b. Under **Users** you can either specify an AD group that contains the user accounts you want to provision in Syncplicity or you can use an LDAP Filter to query the users based on some criteria you decide.

c. Click **Next**.



Enter the Base DN where user records are located in the data store, and specify either an LDAP Filter or Group DN.

BASE DN: CN=Users,DC=myad-lab,DC=pingidentity

Users

GROUP DN: CN=Syncplicity Users,CN=Users,DC=myac □ NESTED SEARCH

FILTER:

Groups

GROUP DN: □ NESTED SEARCH

FILTER:

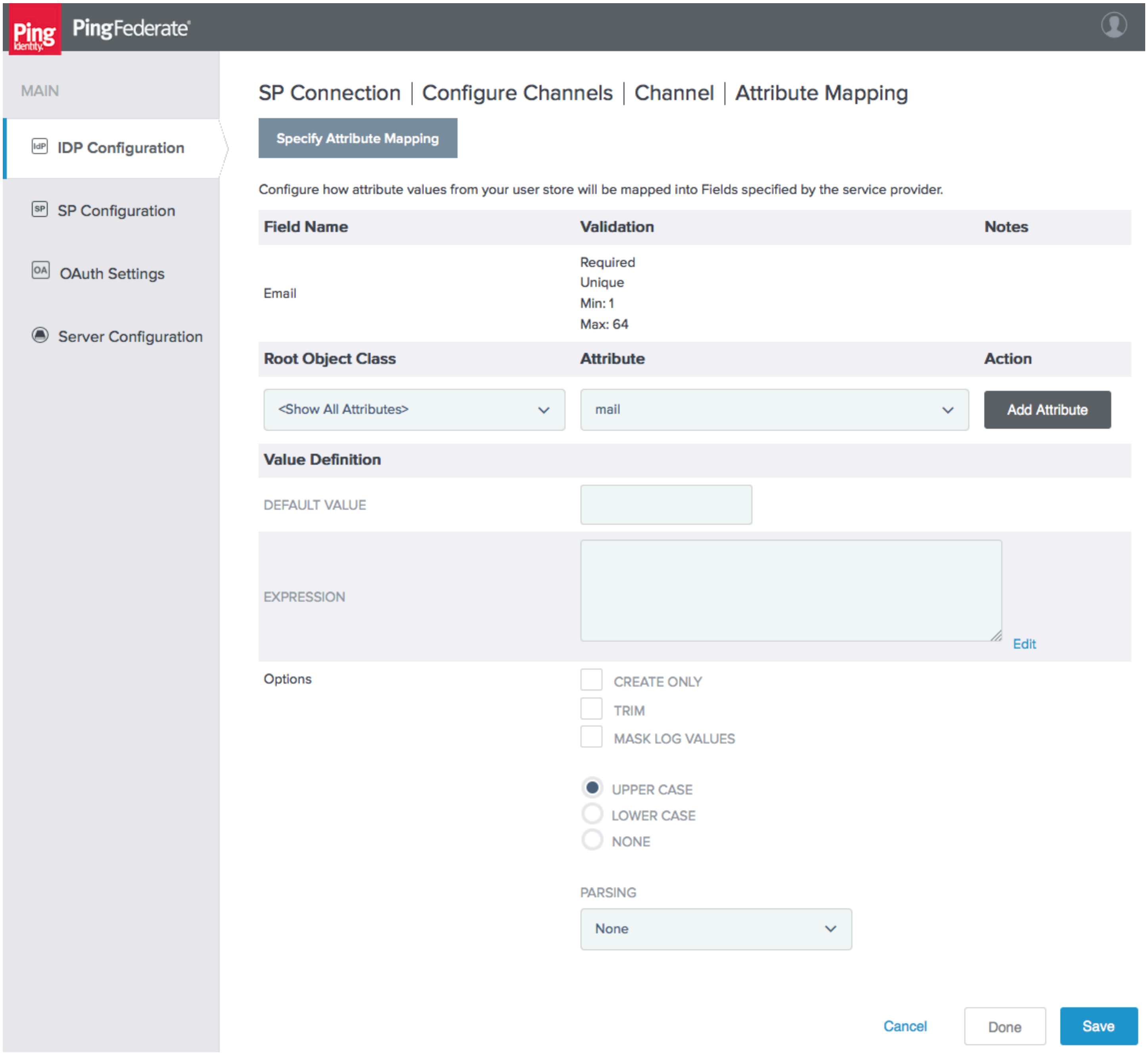
28. On the **Attribute Mapping** page

a. For **Email**, **First Name** and **Last Name**:

   i. Click the **Edit** link next to each attribute.

   ii. Map it to the relevant attribute from your data store (e.g. Email = mail).

   iii. Click **Add Attribute** and then click **Done**.



SP Connection | Configure Channels | Channel | Attribute Mapping

Configure how attribute values from your user store will be mapped into Fields specified by the service provider.

Field Name	Validation	Notes
Email	Required Unique Min: 1 Max: 64	

Root Object Class	Attribute	Action
<Show All Attributes>	mail	Add Attribute

Value Definition

DEFAULT VALUE:

EXPRESSION:

Options

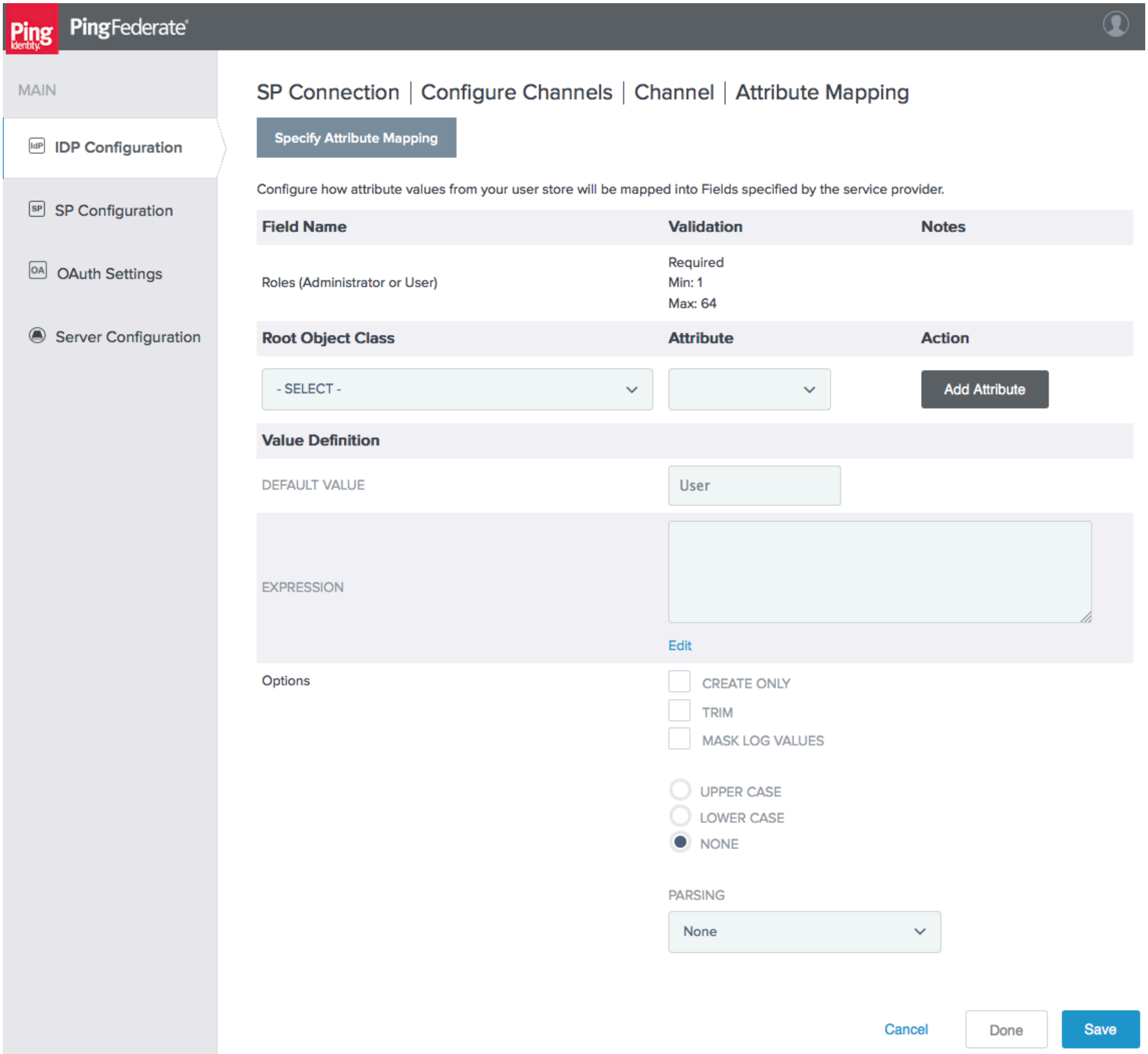
□ CREATE ONLY
□ TRIM
□ MASK LOG VALUES

● UPPER CASE
○ LOWER CASE
○ NONE

PARSING: None

b. For **Roles -** use the Default Value of “**User**” which represents regular users in Syncplicity.

   i. If you need to also provision users with the Administrator role, you should create a second Provisioning Channel to target your Syncplicity Administrator User Group and then set this attribute value to “**Administrator**”. *Note: It is important that users are only a member of either the Syncplicity regular user group, or only a member of the Syncplicity administrator user group. If they are a member of both groups, the Provisioner will only process their account based on the first group they joined.*



SP Connection | Configure Channels | Channel | Attribute Mapping

Specify Attribute Mapping

Configure how attribute values from your user store will be mapped into Fields specified by the service provider.

Field Name	Validation	Notes
Roles (Administrator or User)	Required Min: 1 Max: 64	

Root Object Class	Attribute	Action
- SELECT -		Add Attribute

Value Definition

DEFAULT VALUE: User

EXPRESSION

Edit

Options

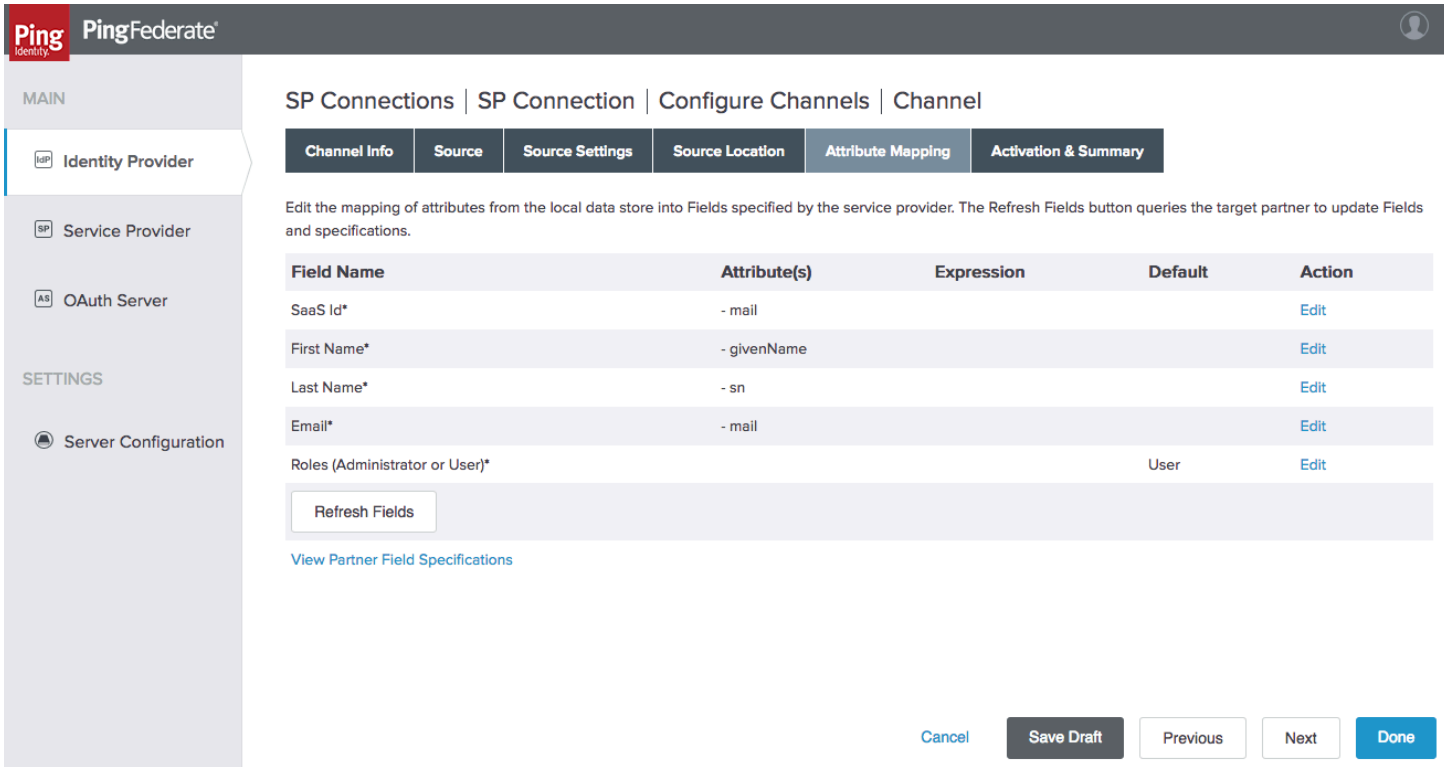
□ CREATE ONLY
□ TRIM
□ MASK LOG VALUES

○ UPPER CASE
○ LOWER CASE
● NONE

PARSING: None

Cancel   Done   Save

c. Once you're finished mapping the attributes, click **Next.**

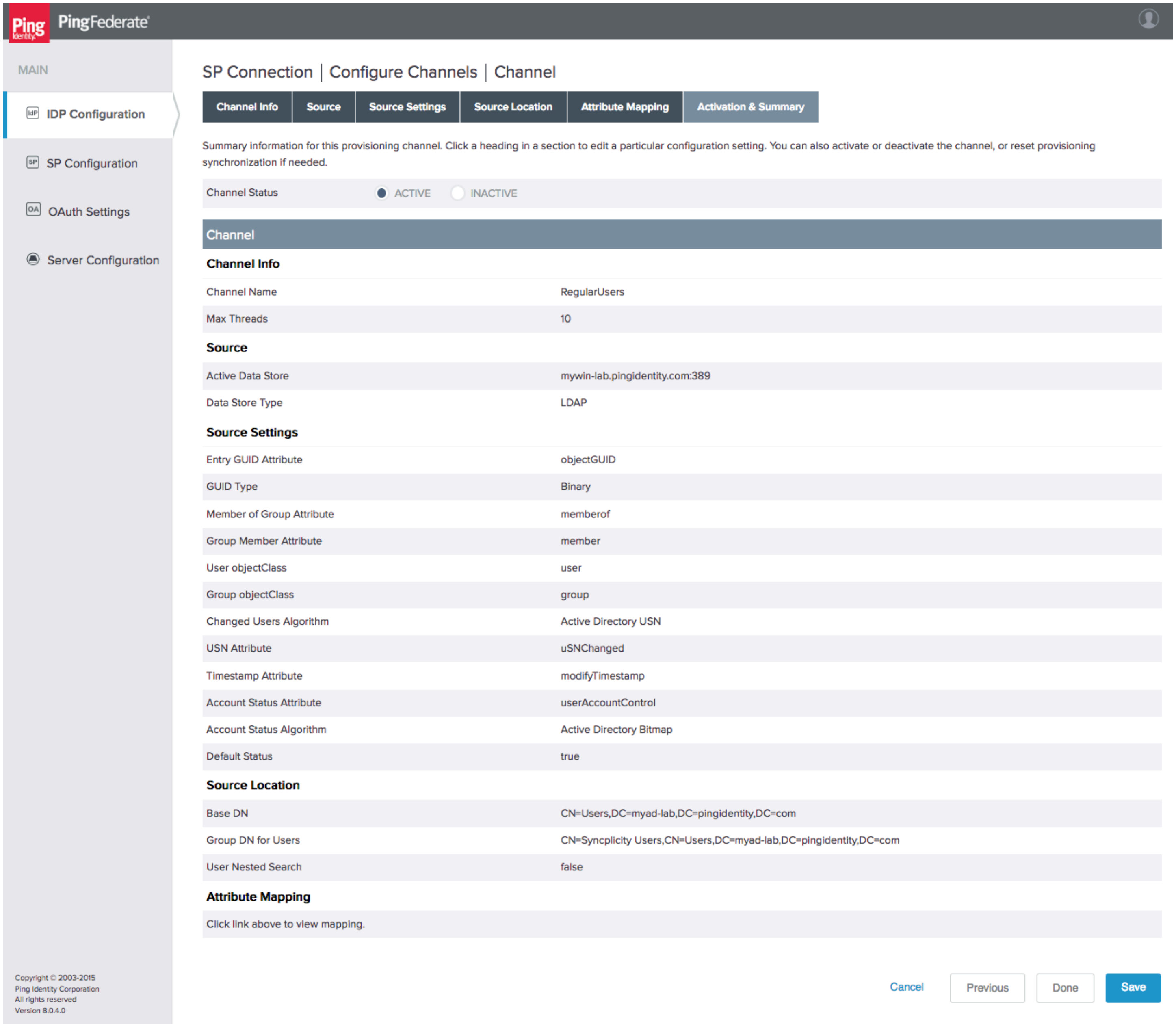


SP Connections | SP Connection | Configure Channels | Channel

Edit the mapping of attributes from the local data store into Fields specified by the service provider. The Refresh Fields button queries the target partner to update Fields and specifications.

Field Name	Attribute(s)	Expression	Default	Action
SaaS Id*	- mail			Edit
First Name*	- givenName			Edit
Last Name*	- sn			Edit
Email*	- mail			Edit
Roles (Administrator or User)*			User	Edit

29. On the **Activation & Summary** page, click **Active**, then click **Done**.



SP Connection | Configure Channels | Channel

Summary information for this provisioning channel. Click a heading in a section to edit a particular configuration setting. You can also activate or deactivate the channel, or reset provisioning synchronization if needed.

Channel Status ● ACTIVE ○ INACTIVE

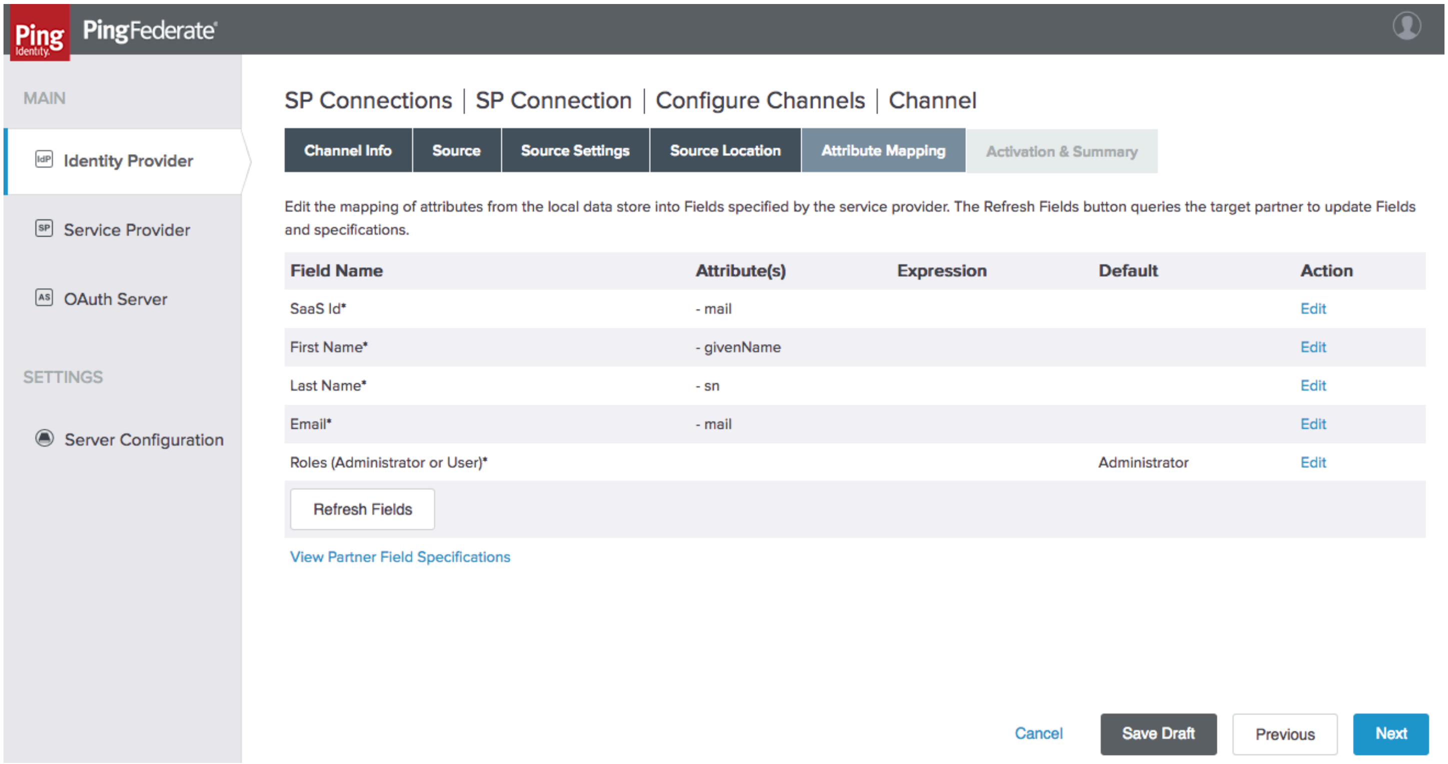
Channel	
Channel Info	
Channel Name	RegularUsers
Max Threads	10
Source	
Active Data Store	mywin-lab.pingidentity.com:389
Data Store Type	LDAP
Source Settings	
Entry GUID Attribute	objectGUID
GUID Type	Binary
Member of Group Attribute	memberof
Group Member Attribute	member
User objectClass	user
Group objectClass	group
Changed Users Algorithm	Active Directory USN
USN Attribute	uSNChanged
Timestamp Attribute	modifyTimestamp
Account Status Attribute	userAccountControl
Account Status Algorithm	Active Directory Bitmap
Default Status	true
Source Location	
Base DN	CN=Users,DC=myad-lab,DC=pingidentity,DC=com
Group DN for Users	CN=Syncplicity Users,CN=Users,DC=myad-lab,DC=pingidentity,DC=com
User Nested Search	false
Attribute Mapping	
Click link above to view mapping.	

Copyright © 2003-2015
Ping Identity Corporation
All rights reserved
Version 8.0.4.0

30. On the **Manage Channels** page..

a. If you need to provision Syncplicity users with the Administrator Role then please repeat steps 23-29 to add a second provisioning channel for SyncplicityAdminUsers.

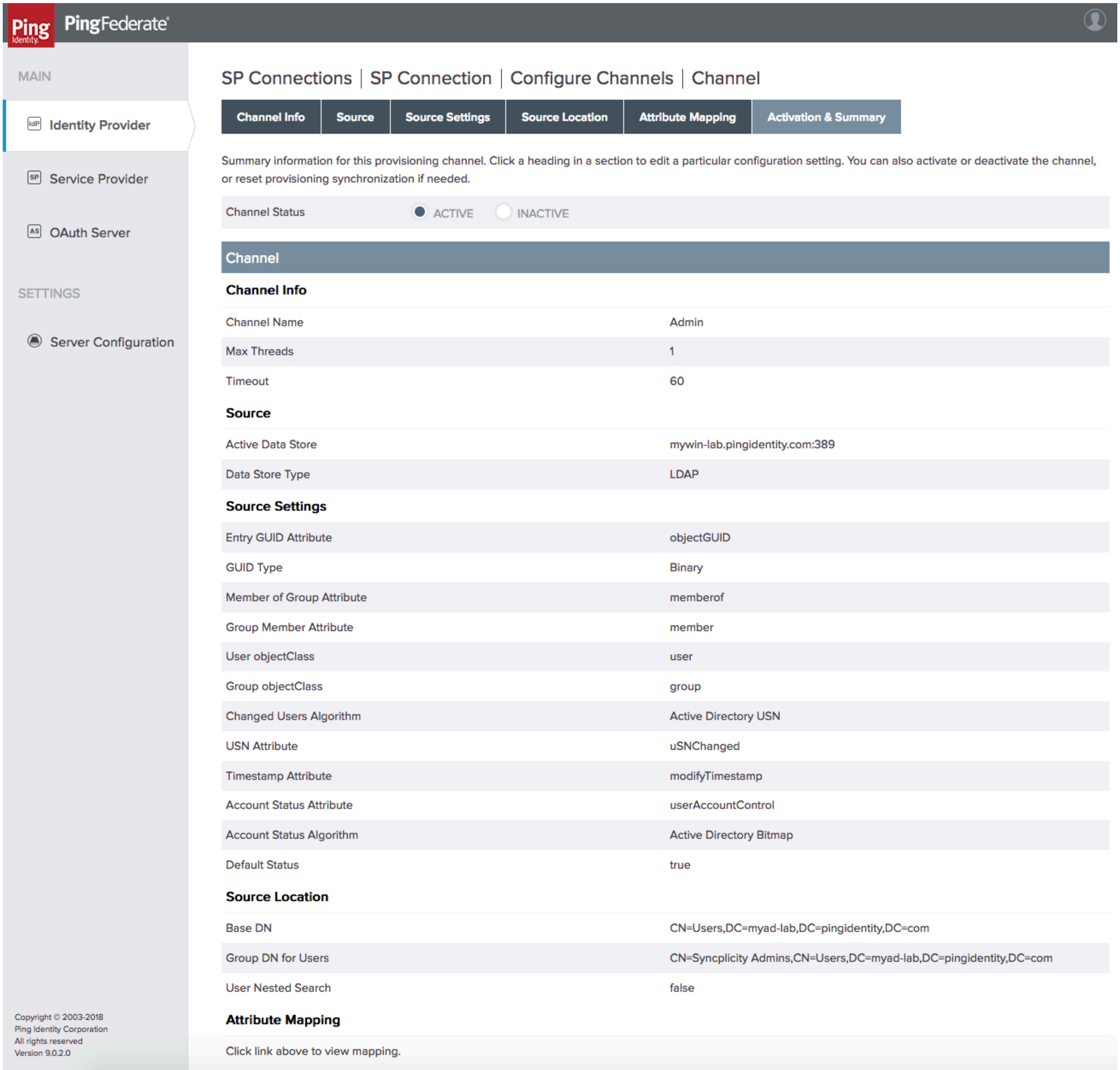
b. Click **Done** when you are finished adding channels**.**



SP Connections | SP Connection | Configure Channels | Channel

Edit the mapping of attributes from the local data store into Fields specified by the service provider. The Refresh Fields button queries the target partner to update Fields and specifications.

Field Name	Attribute(s)	Expression	Default	Action
SaaS Id*	- mail			Edit
First Name*	- givenName			Edit
Last Name*	- sn			Edit
Email*	- mail			Edit
Roles (Administrator or User)*			Administrator	Edit



SP Connections | SP Connection | Configure Channels | Channel

Summary information for this provisioning channel. Click a heading in a section to edit a particular configuration setting. You can also activate or deactivate the channel, or reset provisioning synchronization if needed.

Channel Status ● ACTIVE ○ INACTIVE

Channel

Channel Info	
Channel Name	Admin
Max Threads	1
Timeout	60
Source	
Active Data Store	mywin-lab.pingidentity.com:389
Data Store Type	LDAP
Source Settings	
Entry GUID Attribute	objectGUID
GUID Type	Binary
Member of Group Attribute	memberof
Group Member Attribute	member
User objectClass	user
Group objectClass	group
Changed Users Algorithm	Active Directory USN
USN Attribute	uSNChanged
Timestamp Attribute	modifyTimestamp
Account Status Attribute	userAccountControl
Account Status Algorithm	Active Directory Bitmap
Default Status	true
Source Location	
Base DN	CN=Users,DC=myad-lab,DC=pingidentity,DC=com
Group DN for Users	CN=Syncplicity Admins,CN=Users,DC=myad-lab,DC=pingidentity,DC=com
User Nested Search	false
Attribute Mapping	
Click link above to view mapping.	

Copyright © 2003-2018
Ping Identity Corporation
All rights reserved
Version 9.0.2.0

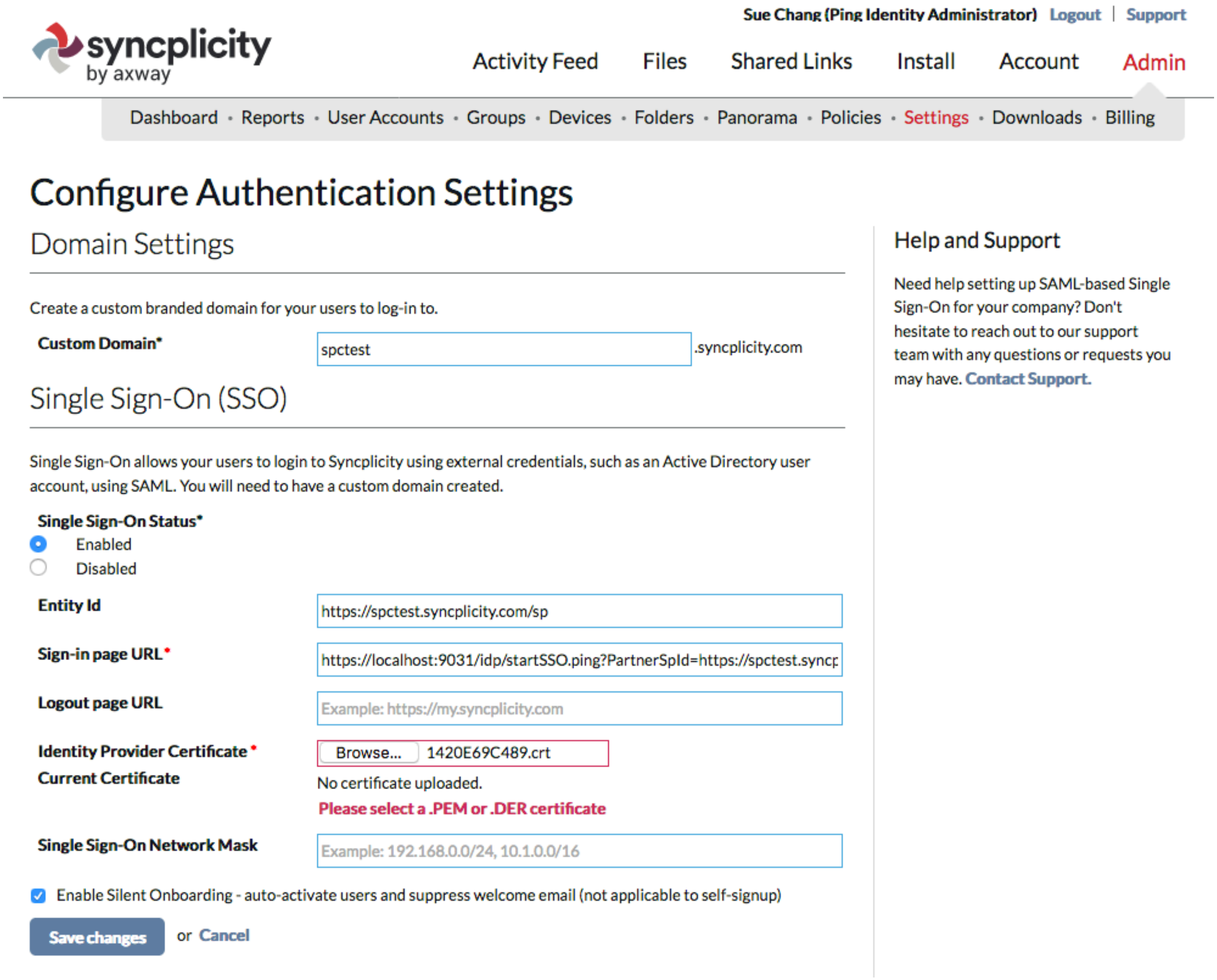
31. On the **Outbound Provisioning** page, click **Next**.
32. On the **Activation & Summary** page, click **Active** then click **Save**.

4.0 Syncplicity SSO Configuration

1. **Login to Syncplicity** with your Administrator account.
2. Go to **admin >> settings >> Custom domain and single sign-on**.
3. Under Single Sign-On (SSO) >> **Single Sign-On Status >>** select the option for **Enabled**.
4. For **Entity ID**, copy/paste the Entity ID from your PingFederate environment.
5. For **Sign-in page URL**, enter your PingFederate IDP SSO URL.
6. For Identity Provider Certificate, select the public key Signing Certificate from your PingFederate.

   a. Note: Syncplicity requires a .PEM or .DER certificate file.

7. Enable the checkbox for “**Enable Silent Onboarding**” if you prefer this setting.
8. Click **Save** Changes (you should then get a confirmation message saying Domain and Single Sign-On Settings have been updated successfully).



Configure Authentication Settings

Domain Settings

Create a custom branded domain for your users to log-in to.

Custom Domain*: spctest .syncplicity.com

Single Sign-On (SSO)

Single Sign-On allows your users to login to Syncplicity using external credentials, such as an Active Directory user account, using SAML. You will need to have a custom domain created.

Single Sign-On Status*
● Enabled
○ Disabled

Entity Id: https://spctest.syncplicity.com/sp

Sign-in page URL*: https://localhost:9031/idp/startSSO.ping?PartnerSpId=https://spctest.syncp

Logout page URL: Example: https://my.syncplicity.com

Identity Provider Certificate*: Browse... 1420E69C489.crt

Current Certificate: No certificate uploaded.

Please select a .PEM or .DER certificate

Single Sign-On Network Mask: Example: 192.168.0.0/24, 10.1.0.0/16

☑ Enable Silent Onboarding - auto-activate users and suppress welcome email (not applicable to self-signup)

Save changes or Cancel

Help and Support

Need help setting up SAML-based Single Sign-On for your company? Don't hesitate to reach out to our support team with any questions or requests you may have. Contact Support.

Tip: *Once you enable SSO on the site, everyone will be prompted to go through SSO login (including your administrator accounts). If you need to bypass the SSO login while troubleshooting SSO, you can login to Syncplicity with your Administrator credentials via this direct link:* [*https://my.syncplicity.com/LoginWithPassword.aspx*](https://my.syncplicity.com/LoginWithPassword.aspx)

5.0 Testing & Verification

1. In order to test, please open a new browser window to your SP-Initiated Login URL.
   a. That would be in the format https://<your custom domain>.syncplicity.com (our example would be https://csa-pfdemo.syncplicity.com).
2. You can then proceed to login with a user account that is a member of the AD group you configured for

Syncplicity Provisioning.

3. In a separate browser window, you can go to the same SP-initiated Login URL, but try to login with a user account who is not a member of the Syncplicity Provisioning AD Group. You should receive a Login Error from Syncplicity indicating you do not have an account.

Note: The following scenarios have been tested, and you are welcome to try these on your environment as well:

Scenario 1: Provisioning Users

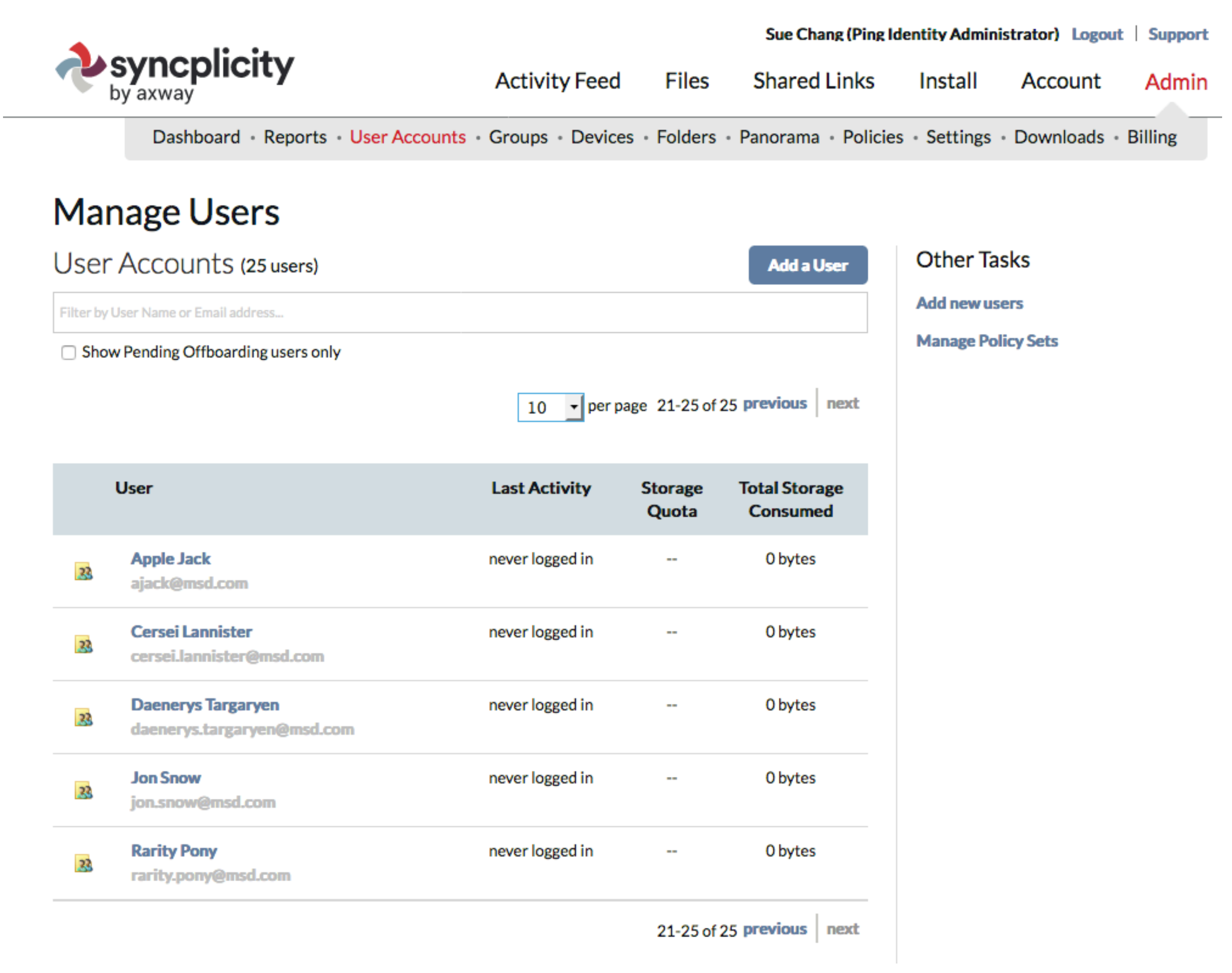
The PingFederate system is configured with a specific AD group for Provisioning to Syncplicity. All accounts in that group should be created in Syncplicity, you can confirm this by adding a user to the group and then checking the PingFederate provisioner.log and the User accounts in Syncplicity.

Provisioner.log (<pf install directory>\pingfederate\log)

```
14:04:25,644 DEBUG [SyncplicitySaasProvisioner] getUser() :: BEGIN
14:04:25,644 DEBUG [SyncplicitySaasProvisioner] getUser() (before) :: guid :: null :: username ::
rarity.pony@merck.com
14:04:25,644 DEBUG [SyncplicitySaasProvisioner] getUser() (after) :: guid :: rarity.pony@merck.com
:: username :: rarity.pony@merck.com
14:04:25,644 DEBUG [UserManager] getUser() :: PROCESSING A GET USER REQUEST FOR ::
rarity.pony@merck.com :: BEGIN
14:04:25,644 DEBUG [TokenManager] isTokenExpired() :: token expiration set by Syncplicity was set
at :: Mon Feb 26 15:28:23 PST 2018
14:04:25,645 DEBUG [TokenManager] isTokenExpired() :: it is now :: Mon Feb 26 14:04:25 PST 2018
14:04:25,645 DEBUG [TokenManager] isTokenExpired() :: token was created at :: Mon Feb 26 13:58:23
PST 2018
14:04:25,645 DEBUG [TokenManager] isTokenExpired() :: token expiration set by the Syncplicity
Provisioner was set at :: Mon Feb 26 14:58:23 PST 2018
14:04:25,645 DEBUG [TokenManager] getToken() :: token has not expired yet, reusing existing token
14:04:25,645 DEBUG [TokenManager] getToken() :: token response's token ::
FOn05cX1OF757jXzj5TqfZHMZHp0aumzjh0MuDYv6vhv8B6BwQN9bXnX4HM6qImR
14:04:25,645 DEBUG [UserManager] getUser() :: getting an access token for the get user request
14:04:25,645 DEBUG [UserManager] processRequest() :: active connections in the pool :: 1
14:04:25,645 DEBUG [GetUserRequest] GetUserRequest - execute() :: processing get user request for
:: rarity.pony@merck.com
14:04:25,645 DEBUG [GetUserRequest] GetUserRequest - execute() :: releasing connection
14:04:25,645 DEBUG [GetUserRequest] BaseRequest - execute() :: performing GET request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40merck.com
14:04:25,645 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: applying authentication mechanism
14:04:25,645 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: releasing connection
14:04:26,017 DEBUG [GetUserRequest] BaseRequest - execute() :: received status code :: 403 ::
while performing GET request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40merck.com with the following response
body :: {"StatusCode":403,"StatusDescription":"Not A Business Admin"}
14:04:26,018 DEBUG [GetUserRequest] BaseRequest - execute() :: releasing connection
14:04:26,018 DEBUG [UserManager] processRequest() :: finished processing :: GET USER transaction
request, and received status code :: 403
14:04:26,018 DEBUG [UserManager] processRequest(); :: releasing connection
14:04:26,018 DEBUG [UserManager] getUser() :: get user request is not ok, status code is :: 403 ::
for saas ID :: rarity.pony@merck.com
14:04:26,018 DEBUG [UserManager] getUser() :: releasing connection
14:04:26,018 DEBUG [UserManager] getUser() :: PROCESSING A GET USER REQUEST FOR ::
rarity.pony@merck.com :: END
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] getUser() :: did not get user, user is null
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] getUser() :: END
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] createUser() :: BEGIN
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] createUser() :: SaaS user data ::
        accountEnabled = true
```

```
       FirstName = [Rarity]
       Roles = [User]
       LastName = [Pony]
       SaasId = [rarity.pony@merck.com]
       EmailAddress = [rarity.pony@merck.com]
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(2) BEGIN
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (saas
user data) :: null
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (user)
:: null
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID
(attribute) :: rarity.pony@merck.com
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(3) BEGIN
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (saas
user data) :: null
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (user)
:: null
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID
(attribute) :: rarity.pony@merck.com
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(3) END
14:04:26,018 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(2) END
14:04:26,018 DEBUG [UserManager] createUser() :: PROCESSING A CREATE USER REQUEST FOR ::
rarity.pony@merck.com :: BEGIN
14:04:26,018 DEBUG [TokenManager] isTokenExpired() :: token expiration set by Syncplicity was set
at :: Mon Feb 26 15:28:23 PST 2018
14:04:26,018 DEBUG [TokenManager] isTokenExpired() :: it is now :: Mon Feb 26 14:04:26 PST 2018
14:04:26,018 DEBUG [TokenManager] isTokenExpired() :: token was created at :: Mon Feb 26 13:58:23
PST 2018
14:04:26,018 DEBUG [TokenManager] isTokenExpired() :: token expiration set by the Syncplicity
Provisioner was set at :: Mon Feb 26 14:58:23 PST 2018
14:04:26,018 DEBUG [TokenManager] getToken() :: token has not expired yet, reusing existing token
14:04:26,018 DEBUG [TokenManager] getToken() :: token response's token ::
FOn05cX1OF757jXzj5TqfZHMZHp0aumzjh0MuDYv6vhv8B6BwQN9bXnX4HM6qImR
14:04:26,018 DEBUG [UserManager] createUser() :: getting an access token for the create user
request
14:04:26,019 DEBUG [UserManager] processRequest() :: active connections in the pool :: 1
14:04:26,019 DEBUG [CreateUserRequest] CreateUserRequest - execute() :: processing create user
request for :: rarity.pony@merck.com
14:04:26,021 DEBUG [CreateUserRequest] CreateUserRequest - execute() :: releasing connection
14:04:26,021 DEBUG [CreateUserRequest] BaseRequest - execute() :: performing POST request on ::
https://xml.syncplicity.com/1.1/auth/users.svc/
14:04:26,021 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: applying authentication mechanism
14:04:26,021 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: releasing connection
14:04:26,475 DEBUG [CreateUserRequest] BaseRequest - execute() :: received status code :: 200 ::
while performing POST request on :: https://xml.syncplicity.com/1.1/auth/users.svc/ with the
following response body ::
[{"EmailAddress":"rarity.pony@merck.com","FirstName":"Rarity","LastName":"Pony","Disabled":false,"
Company":{"Name":"Ping Identity"},"Roles":[],"AccountType":7}]
14:04:26,475 DEBUG [CreateUserRequest] BaseRequest - execute() :: releasing connection
14:04:26,475 DEBUG [UserManager] processRequest() :: finished processing :: PROVISION transaction
request, and received status code :: 200
14:04:26,475 DEBUG [UserManager] processRequest(); :: releasing connection
14:04:26,475 DEBUG [UserManager] createUser() :: create user request is ok, status code is :: 200
:: for saas ID :: rarity.pony@merck.com
14:04:26,475 DEBUG [UserManager] createUser() :: response body ::
[{"EmailAddress":"rarity.pony@merck.com","FirstName":"Rarity","LastName":"Pony","Disabled":false,"
Company":{"Name":"Ping Identity"},"Roles":[],"AccountType":7}]
14:04:26,475 DEBUG [UserManager] createUser() :: releasing connection
14:04:26,475 DEBUG [UserManager] createUser() :: PROCESSING A CREATE USER REQUEST FOR ::
rarity.pony@merck.com :: END
14:04:26,475 DEBUG [SyncplicitySaasProvisioner] createUser() :: END
```

Syncplicity >> admin >>user accounts



Manage Users

User Accounts (25 users)

□ Show Pending Offboarding users only

10 per page 21-25 of 25 previous | next

User	Last Activity	Storage Quota	Total Storage Consumed
Apple Jack ajack@msd.com	never logged in	--	0 bytes
Cersei Lannister cersei.lannister@msd.com	never logged in	--	0 bytes
Daenerys Targaryen daenerys.targaryen@msd.com	never logged in	--	0 bytes
Jon Snow jon.snow@msd.com	never logged in	--	0 bytes
Rarity Pony rarity.pony@msd.com	never logged in	--	0 bytes

21-25 of 25 previous | next

Scenario 2: Deprovisioning Users

You can test deprovisioning by either removing a test user from the Syncplicity Provisioning AD group or by deactivating the user account in Active Directory.

Provisioner.log (<pf install directory>\pingfederate\log)

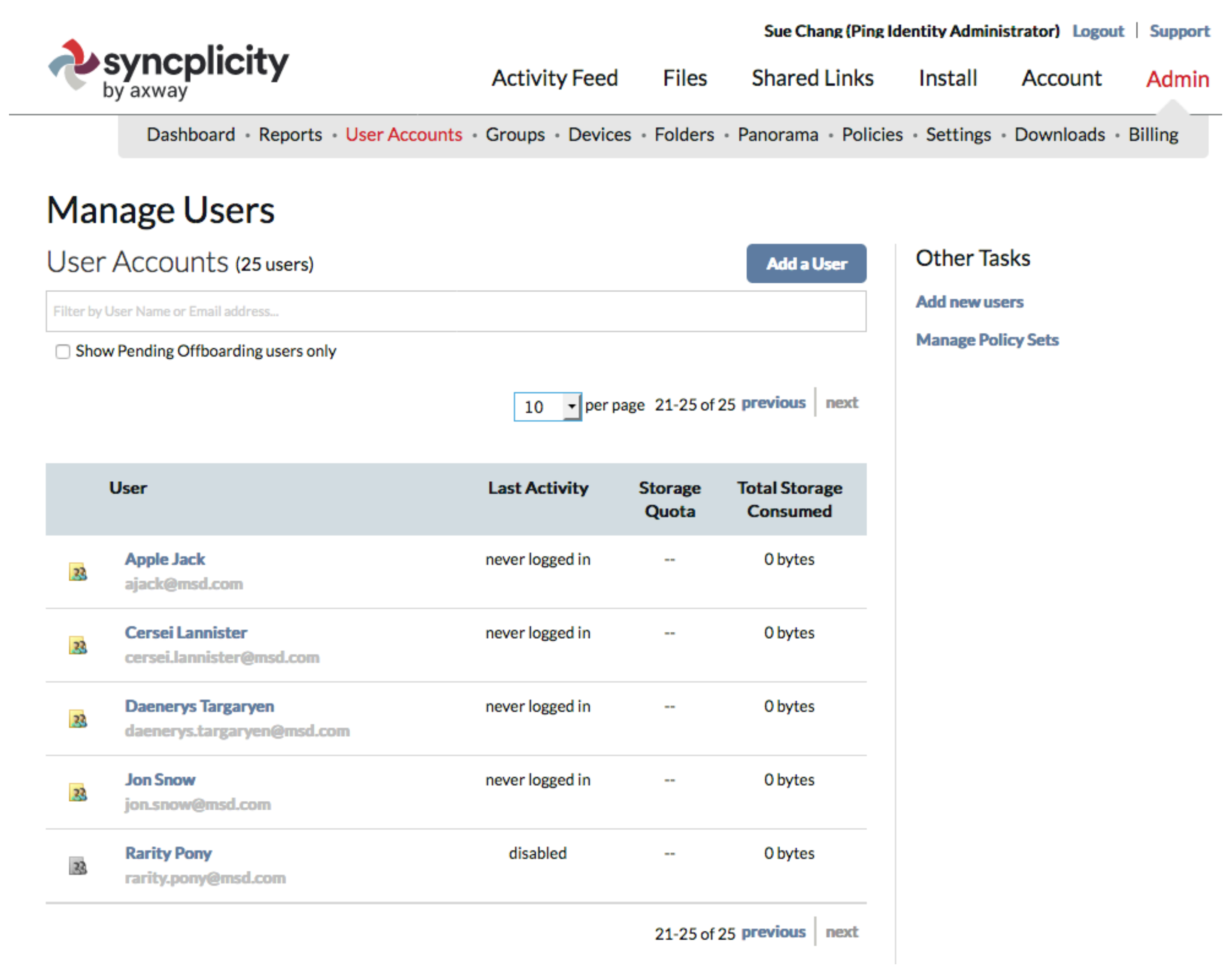
```
14:42:17,056 DEBUG [SyncplicitySaasProvisioner] getUser() :: BEGIN
14:42:17,056 DEBUG [SyncplicitySaasProvisioner] getUser() (before) :: guid :: rarity.pony@msd.com
:: username :: rarity.pony@msd.com
14:42:17,056 DEBUG [SyncplicitySaasProvisioner] getUser() (after) :: guid :: rarity.pony@msd.com
:: username :: rarity.pony@msd.com
14:42:17,056 DEBUG [UserManager] getUser() :: PROCESSING A GET USER REQUEST FOR ::
rarity.pony@msd.com :: BEGIN
14:42:17,056 DEBUG [TokenManager] isTokenExpired() :: token response is null, returning
isTokenExpired as true
14:42:17,056 DEBUG [TokenManager] getToken() :: token has expired, getting new token
14:42:17,056 DEBUG [TokenManager] initToken() :: BEGIN
```

```
14:42:17,056 DEBUG [TokenManager] isTokenExpired() :: token response is null, returning
isTokenExpired as true
14:42:17,056 DEBUG [CreateTokenRequest] CreateTokenRequest - execute() :: releasing connection
14:42:17,056 DEBUG [CreateTokenRequest] CreateTokenRequest - execute() :: executing token creation
request
14:42:17,056 DEBUG [CreateTokenRequest] BaseRequest - execute() :: performing GET request on ::
https://xml.syncplicity.com/1.1/auth/tokens.svc/
14:42:17,056 DEBUG [ApplicationTokenCredential] ApplicationTokenCredential -
applyAuthenticationMechanism() :: applying authentication mechanism
14:42:17,056 DEBUG [ApplicationTokenCredential] ApplicationTokenCredential -
applyAuthenticationMechanism() :: releasing connection
14:42:17,437 DEBUG [CreateTokenRequest] BaseRequest - execute() :: received status code :: 200 ::
while performing GET request on :: https://xml.syncplicity.com/1.1/auth/tokens.svc/ with the
following response body ::
[{"Id":"z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30","Duration":5399.9822188,
"User":{"Id":"ebb99234-4b68-403f-8e85-
419884aa401a","EmailAddress":"","FirstName":"Sue","LastName":"Chang","Company":{"Id":"bc096d16-
f58b-41f6-8142-859553e89b46","Name":"Ping Identity","AccountType":3},"AccountType":14}}]
14:42:17,437 DEBUG [CreateTokenRequest] BaseRequest - execute() :: releasing connection
14:42:17,437 DEBUG [TokenResponse] fromHttpResponse() :: response status code is ok [200] :: with
body ::
[{"Id":"z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30","Duration":5399.9822188,
"User":{"Id":"ebb99234-4b68-403f-8e85-
419884aa401a","EmailAddress":"","FirstName":"Sue","LastName":"Chang","Company":{"Id":"bc096d16-
f58b-41f6-8142-859553e89b46","Name":"Ping Identity","AccountType":3},"AccountType":14}}]
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: JSON string ::
[{"Id":"z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30","Duration":5399.9822188,
"User":{"Id":"ebb99234-4b68-403f-8e85-
419884aa401a","EmailAddress":"","FirstName":"Sue","LastName":"Chang","Company":{"Id":"bc096d16-
f58b-41f6-8142-859553e89b46","Name":"Ping Identity","AccountType":3},"AccountType":14}}]
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: received status code :: 200 for token
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: time before request (used for setting
expected token expiration) :: 1519684937437
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: status is ok and received a JSON array when
requesting token
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: Received [1] tokens from attempt to acquire
tokens.
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: Received
[[{"User":{"Company":{"Id":"bc096d16-f58b-41f6-8142-859553e89b46","AccountType":3,"Name":"Ping
Identity"},"FirstName":"Sue","Id":"ebb99234-4b68-403f-8e85-
419884aa401a","LastName":"Chang","AccountType":14,"EmailAddress":""},"Duration":5399.9822188,"Id":
"z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30"}]] tokens from parse.
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: token response status code has been set to
:: 200
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: token response token has been set to ::
z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30
14:42:17,437 DEBUG [TokenResponse] TokenResponse() :: expected token expiration set to :: Mon Feb
26 16:12:16 PST 2018
14:42:17,437 DEBUG [TokenResponse] TokenResponse(); :: JSON object is not null, user object has
been set
14:42:17,437 DEBUG [TokenResponse] fromHttpResponse() :: releasing connection
14:42:17,437 DEBUG [TokenManager] initToken() :: user from token response is not null
14:42:17,437 DEBUG [TokenManager] initToken() :: releasing connection
14:42:17,437 DEBUG [TokenManager] initToken() :: END
14:42:17,437 DEBUG [TokenManager] getToken() :: token response's token ::
z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30
14:42:17,437 DEBUG [UserManager] getUser() :: getting an access token for the get user request
14:42:17,437 DEBUG [UserManager] processRequest() :: active connections in the pool :: 1
14:42:17,437 DEBUG [GetUserRequest] GetUserRequest - execute() :: processing get user request for
:: rarity.pony@msd.com
14:42:17,438 DEBUG [GetUserRequest] GetUserRequest - execute() :: releasing connection
14:42:17,438 DEBUG [GetUserRequest] BaseRequest - execute() :: performing GET request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com
14:42:17,438 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: applying authentication mechanism
14:42:17,438 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: releasing connection
14:42:17,534 DEBUG [GetUserRequest] BaseRequest - execute() :: received status code :: 200 ::
```

```
while performing GET request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com with the following response
body :: {"Id":"e31c32b7-1da1-4d36-85fa-
396ac614ad63","EmailAddress":"rarity.pony@msd.com","FirstName":"Rarity","LastName":"Pony","Roles":
[],"AccountType":7,"Status":2,"CreatedDateUtc":"\/Date(1519682666237)\/"}
14:42:17,534 DEBUG [GetUserRequest] BaseRequest - execute() :: releasing connection
14:42:17,534 DEBUG [UserManager] processRequest() :: finished processing :: GET USER transaction
request, and received status code :: 200
14:42:17,534 DEBUG [UserManager] processRequest(); :: releasing connection
14:42:17,534 DEBUG [UserManager] getUser() :: get user request is ok, status code :: 200 :: for
saas ID :: rarity.pony@msd.com
14:42:17,534 DEBUG [UserManager] getUser() :: response body :: {"Id":"e31c32b7-1da1-4d36-85fa-
396ac614ad63","EmailAddress":"rarity.pony@msd.com","FirstName":"Rarity","LastName":"Pony","Roles":
[],"AccountType":7,"Status":2,"CreatedDateUtc":"\/Date(1519682666237)\/"}
14:42:17,534 DEBUG [UserManager] getUser() :: releasing connection
14:42:17,534 DEBUG [UserManager] getUser() :: PROCESSING A GET USER REQUEST FOR ::
rarity.pony@msd.com :: END
14:42:17,534 DEBUG [SyncplicitySaasProvisioner] getUser() :: got user, user exists
14:42:17,534 DEBUG [SyncplicitySaasProvisioner] getUser() :: END
14:42:17,534 DEBUG [SyncplicitySaasProvisioner] updateUser() :: BEGIN
14:42:17,535 DEBUG [SyncplicitySaasProvisioner] updateUser() :: existing user data ::
        accountEnabled = true
        SaasId = [rarity.pony@msd.com] :: updated user data ::      accountEnabled = false
        FirstName = [Rarity]
        Roles = [User]
        LastName = [Pony]
        SaasId = [rarity.pony@msd.com]
        EmailAddress = [rarity.pony@msd.com]
14:42:17,535 DEBUG [SyncplicitySaasProvisioner] updateUser() :: Triggering an update for ::
rarity.pony@msd.com (SaaS ID), rarity.pony@msd.com (SaaS Username)
14:42:17,535 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(3) BEGIN
14:42:17,535 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (saas
user data) :: rarity.pony@msd.com
14:42:17,535 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(3) END
14:42:17,535 DEBUG [UserManager] updateUser() :: PROCESSING AN UPDATE USER REQUEST FOR ::
rarity.pony@msd.com :: BEGIN
14:42:17,535 DEBUG [TokenManager] isTokenExpired() :: token expiration set by Syncplicity was set
at :: Mon Feb 26 16:12:16 PST 2018
14:42:17,535 DEBUG [TokenManager] isTokenExpired() :: it is now :: Mon Feb 26 14:42:17 PST 2018
14:42:17,535 DEBUG [TokenManager] isTokenExpired() :: token was created at :: Mon Feb 26 14:42:16
PST 2018
14:42:17,535 DEBUG [TokenManager] isTokenExpired() :: token expiration set by the Syncplicity
Provisioner was set at :: Mon Feb 26 15:42:16 PST 2018
14:42:17,535 DEBUG [TokenManager] getToken() :: token has not expired yet, reusing existing token
14:42:17,535 DEBUG [TokenManager] getToken() :: token response's token ::
z3Rd0l7PRgOcIQoUqBiFXo1OJfv3YfVIbBnELte0mkl3tUWRy2jKj67W5gs0ak30
14:42:17,535 DEBUG [UserManager] updateUser() :: getting an access token for the update user
request
14:42:17,535 DEBUG [UserManager] processRequest() :: active connections in the pool :: 1
14:42:17,535 DEBUG [UpdateUserRequest] UpdateUserRequest - execute() :: processing update user
request for :: rarity.pony@msd.com
14:42:17,535 DEBUG [UpdateUserRequest] UpdateUserRequest - execute() :: releasing connection
14:42:17,535 DEBUG [UpdateUserRequest] BaseRequest - execute() :: performing PUT request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com
14:42:17,535 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: applying authentication mechanism
14:42:17,535 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: releasing connection
14:42:17,665 DEBUG [UpdateUserRequest] BaseRequest - execute() :: received status code :: 200 ::
while performing PUT request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com with the following response
body ::
{"EmailAddress":"rarity.pony@msd.com","FirstName":"Rarity","LastName":"Pony","Disabled":true,"Comp
any":{"Name":"Ping Identity"},"Roles":[],"AccountType":7,"Active":true}
14:42:17,665 DEBUG [UpdateUserRequest] BaseRequest - execute() :: releasing connection
14:42:17,665 DEBUG [UserManager] processRequest() :: finished processing :: UPDATE transaction
request, and received status code :: 200
14:42:17,665 DEBUG [UserManager] processRequest(); :: releasing connection
```

```
14:42:17,665 DEBUG [UserManager] updateUser() :: update user request is ok, status code is :: 200
:: for saas ID :: rarity.pony@msd.com
14:42:17,665 DEBUG [UserManager] updateUser() :: releasing connection
14:42:17,665 DEBUG [UserManager] updateUser() :: PROCESSING AN UPDATE USER REQUEST FOR ::
rarity.pony@msd.com :: END
14:42:17,665 DEBUG [SyncplicitySaasProvisioner] updateUser() :: Update processed for ::
rarity.pony@msd.com (SaaS ID), rarity.pony@msd.com (SaaS Username)
14:42:17,665 DEBUG [SyncplicitySaasProvisioner] updateUser() :: END
14:42:17,666 DEBUG [JdbcUserStore] Setting SaaS GUID for user in JDBC store: channel=4,
dsGUID=b96ae5d22be29949bc946bcb4d397bb6, saasGuid=rarity.pony@msd.com []
14:42:17,928 DEBUG [JdbcUserStore] Clearing user in JDBC store: channel=4,
dsGUID=b96ae5d22be29949bc946bcb4d397bb6 []
14:42:18,185 DEBUG [JdbcUserStore] Setting SaaS Identity for user in JDBC store: channel=4,
dsGUID=b96ae5d22be29949bc946bcb4d397bb6,
saasIdentity=eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2Iiwia2lkIjoiZ2FCQkR5UzdlZCIsInZlcnNpb24iO
iI4LjAuNC4wIn0..QZKhX76a05HwfJXjNEfMNQ.qKt-
kWUaqAV7hPTXxJ0pJvegbAz1NlPbdE4LIqWRAyM.UHZDuqGATKRji7pf42H3zw []
14:42:18,444 INFO  [SystemOut] 14:42:18,444 INFO  [UserActivityLogger] Updated:
rarity.pony@msd.com
```

Syncplicity >> admin >>user accounts


syncplicity
by axway
Sue Chang (Ping Identity Administrator) Logout | Support
Activity Feed
Files
Shared Links
Install
Account
Admin
Dashboard • Reports • User Accounts • Groups • Devices • Folders • Panorama • Policies • Settings • Downloads • Billing
Add a User
Filter by User Name or Email address...
Other Tasks
Add new users
Manage Policy Sets
10
per page 21-25 of 25 previous | next

Manage Users

User Accounts (25 users)

□ Show Pending Offboarding users only

User	Last Activity	Storage Quota	Total Storage Consumed
Apple Jack ajack@msd.com	never logged in	--	0 bytes
Cersei Lannister cersei.lannister@msd.com	never logged in	--	0 bytes
Daenerys Targaryen daenerys.targaryen@msd.com	never logged in	--	0 bytes
Jon Snow jon.snow@msd.com	never logged in	--	0 bytes
Rarity Pony rarity.pony@msd.com	disabled	--	0 bytes

21-25 of 25 previous | next

Scenario 3: Reactivating a De-provisioned User Account

In order to reactivate the Syncplicity account for the deprovisioned user from the above example, we just need to reactivate his AD account or re-add him to the AD group (depending on how you chose to remove him).

We can confirm this was successful by checking the following:

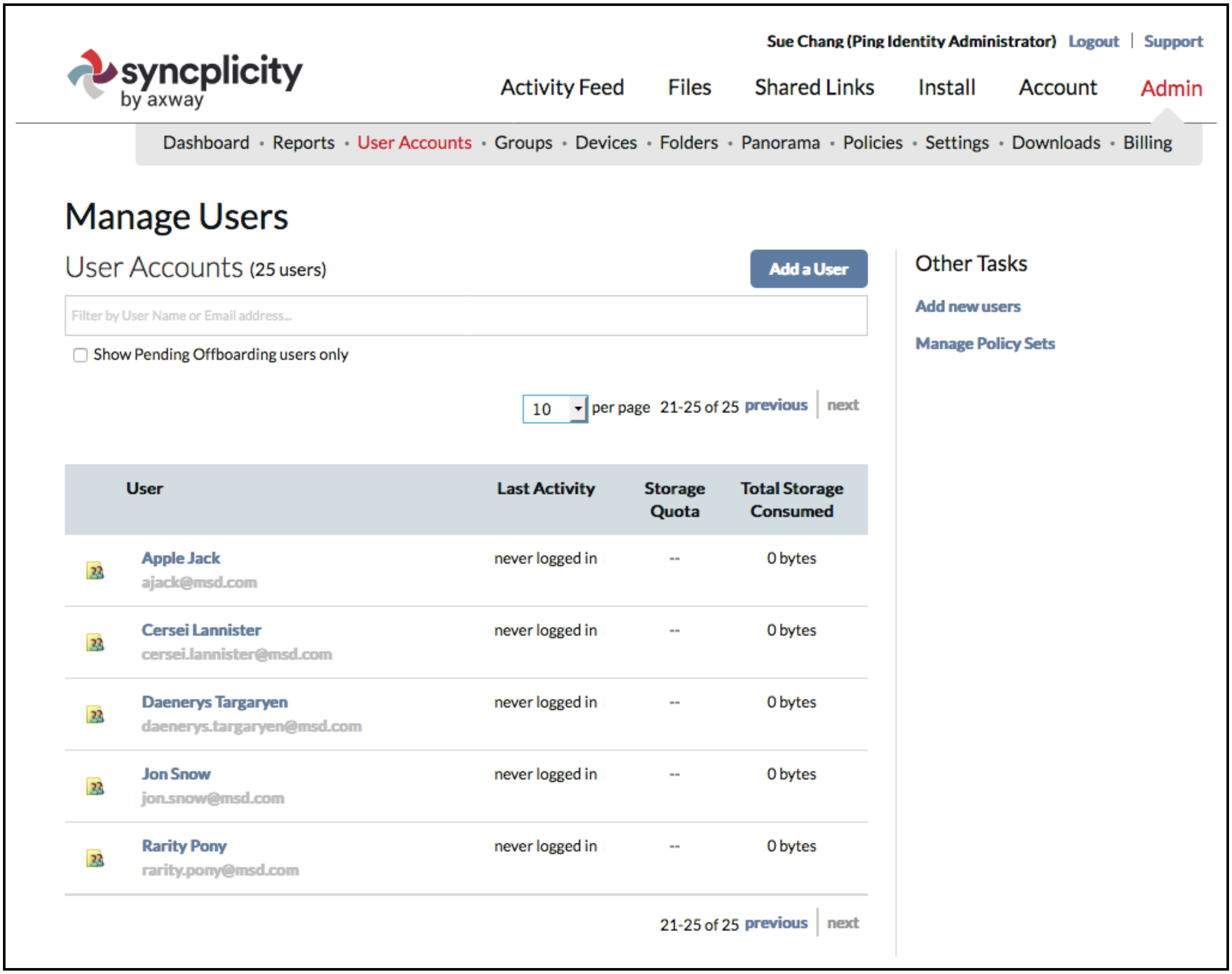
Provisioner.log (<pf install directory>\pingfederate\log)

```
14:52:44,136 DEBUG [SyncplicitySaasProvisioner] getUser() :: BEGIN
14:52:44,136 DEBUG [SyncplicitySaasProvisioner] getUser() (before) :: guid :: null :: username ::
rarity.pony@msd.com
14:52:44,136 DEBUG [SyncplicitySaasProvisioner] getUser() (after) :: guid :: rarity.pony@msd.com
:: username :: rarity.pony@msd.com
14:52:44,136 DEBUG [UserManager] getUser() :: PROCESSING A GET USER REQUEST FOR ::
rarity.pony@msd.com :: BEGIN
14:52:44,137 DEBUG [TokenManager] isTokenExpired() :: token response is null, returning
isTokenExpired as true
14:52:44,137 DEBUG [TokenManager] getToken() :: token has expired, getting new token
14:52:44,137 DEBUG [TokenManager] initToken() :: BEGIN
14:52:44,137 DEBUG [TokenManager] isTokenExpired() :: token response is null, returning
isTokenExpired as true
14:52:44,158 DEBUG [CreateTokenRequest] CreateTokenRequest - execute() :: releasing connection
14:52:44,158 DEBUG [CreateTokenRequest] CreateTokenRequest - execute() :: executing token creation
request
14:52:44,158 DEBUG [CreateTokenRequest] BaseRequest - execute() :: performing GET request on ::
https://xml.syncplicity.com/1.1/auth/tokens.svc/
14:52:44,158 DEBUG [ApplicationTokenCredential] ApplicationTokenCredential -
applyAuthenticationMechanism() :: applying authentication mechanism
14:52:44,158 DEBUG [ApplicationTokenCredential] ApplicationTokenCredential -
applyAuthenticationMechanism() :: releasing connection
14:52:44,573 DEBUG [CreateTokenRequest] BaseRequest - execute() :: received status code :: 200 ::
while performing GET request on :: https://xml.syncplicity.com/1.1/auth/tokens.svc/ with the
following response body ::
[{"Id":"w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8","Duration":5399.9939232,
"User":{"Id":"ebb99234-4b68-403f-8e85-
419884aa401a","EmailAddress":"","FirstName":"Sue","LastName":"Chang","Company":{"Id":"bc096d16-
f58b-41f6-8142-859553e89b46","Name":"Ping Identity","AccountType":3},"AccountType":14}}]
14:52:44,573 DEBUG [CreateTokenRequest] BaseRequest - execute() :: releasing connection
14:52:44,574 DEBUG [TokenResponse] fromHttpResponse() :: response status code is ok [200] :: with
body ::
[{"Id":"w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8","Duration":5399.9939232,
"User":{"Id":"ebb99234-4b68-403f-8e85-
419884aa401a","EmailAddress":"","FirstName":"Sue","LastName":"Chang","Company":{"Id":"bc096d16-
f58b-41f6-8142-859553e89b46","Name":"Ping Identity","AccountType":3},"AccountType":14}}]
14:52:44,574 DEBUG [TokenResponse] TokenResponse() :: JSON string ::
[{"Id":"w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8","Duration":5399.9939232,
"User":{"Id":"ebb99234-4b68-403f-8e85-
419884aa401a","EmailAddress":"","FirstName":"Sue","LastName":"Chang","Company":{"Id":"bc096d16-
f58b-41f6-8142-859553e89b46","Name":"Ping Identity","AccountType":3},"AccountType":14}}]
14:52:44,574 DEBUG [TokenResponse] TokenResponse() :: received status code :: 200 for token
14:52:44,574 DEBUG [TokenResponse] TokenResponse() :: time before request (used for setting
expected token expiration) :: 1519685564574
14:52:44,580 DEBUG [TokenResponse] TokenResponse() :: status is ok and received a JSON array when
requesting token
14:52:44,580 DEBUG [TokenResponse] TokenResponse() :: Received [1] tokens from attempt to acquire
tokens.
14:52:44,581 DEBUG [TokenResponse] TokenResponse() :: Received
[[{"User":{"Company":{"Id":"bc096d16-f58b-41f6-8142-859553e89b46","AccountType":3,"Name":"Ping
Identity"},"FirstName":"Sue","Id":"ebb99234-4b68-403f-8e85-
419884aa401a","LastName":"Chang","AccountType":14,"EmailAddress":""},"Duration":5399.9939232,"Id":
"w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8"}]] tokens from parse.
14:52:44,581 DEBUG [TokenResponse] TokenResponse() :: token response status code has been set to
:: 200
```

```
14:52:44,581 DEBUG [TokenResponse] TokenResponse() :: token response token has been set to ::
w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8
14:52:44,581 DEBUG [TokenResponse] TokenResponse() :: expected token expiration set to :: Mon Feb
26 16:22:43 PST 2018
14:52:44,585 DEBUG [TokenResponse] TokenResponse(); :: JSON object is not null, user object has
been set
14:52:44,585 DEBUG [TokenResponse] fromHttpResponse() :: releasing connection
14:52:44,585 DEBUG [TokenManager] initToken() :: user from token response is not null
14:52:44,585 DEBUG [TokenManager] initToken() :: releasing connection
14:52:44,585 DEBUG [TokenManager] initToken() :: END
14:52:44,585 DEBUG [TokenManager] getToken() :: token response's token ::
w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8
14:52:44,585 DEBUG [UserManager] getUser() :: getting an access token for the get user request
14:52:44,586 DEBUG [UserManager] processRequest() :: active connections in the pool :: 1
14:52:44,586 DEBUG [GetUserRequest] GetUserRequest - execute() :: processing get user request for
:: rarity.pony@msd.com
14:52:44,586 DEBUG [GetUserRequest] GetUserRequest - execute() :: releasing connection
14:52:44,586 DEBUG [GetUserRequest] BaseRequest - execute() :: performing GET request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com
14:52:44,586 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: applying authentication mechanism
14:52:44,586 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: releasing connection
14:52:44,682 DEBUG [GetUserRequest] BaseRequest - execute() :: received status code :: 200 ::
while performing GET request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com with the following response
body :: {"Id":"e31c32b7-1da1-4d36-85fa-
396ac614ad63","EmailAddress":"rarity.pony@msd.com","FirstName":"Rarity","LastName":"Pony","Roles":
[],"AccountType":7,"Status":1,"CreatedDateUtc":"\/Date(1519682666237)\/"}
14:52:44,682 DEBUG [GetUserRequest] BaseRequest - execute() :: releasing connection
14:52:44,682 DEBUG [UserManager] processRequest() :: finished processing :: GET USER transaction
request, and received status code :: 200
14:52:44,682 DEBUG [UserManager] processRequest(); :: releasing connection
14:52:44,682 DEBUG [UserManager] getUser() :: get user request is ok, status code :: 200 :: for
saas ID :: rarity.pony@msd.com
14:52:44,682 DEBUG [UserManager] getUser() :: response body :: {"Id":"e31c32b7-1da1-4d36-85fa-
396ac614ad63","EmailAddress":"rarity.pony@msd.com","FirstName":"Rarity","LastName":"Pony","Roles":
[],"AccountType":7,"Status":1,"CreatedDateUtc":"\/Date(1519682666237)\/"}
14:52:44,682 DEBUG [UserManager] getUser() :: releasing connection
14:52:44,682 DEBUG [UserManager] getUser() :: PROCESSING A GET USER REQUEST FOR ::
rarity.pony@msd.com :: END
14:52:44,683 DEBUG [SyncplicitySaasProvisioner] getUser() :: got user, user exists
14:52:44,683 DEBUG [SyncplicitySaasProvisioner] getUser() :: END
14:52:44,683 DEBUG [SyncplicitySaasProvisioner] updateUser() :: BEGIN
14:52:44,683 DEBUG [SyncplicitySaasProvisioner] updateUser() :: existing user data ::
        accountEnabled = true
        SaasId = [rarity.pony@msd.com] :: updated user data ::      accountEnabled = true
        FirstName = [Rarity]
        Roles = [User]
        LastName = [Pony]
        SaasId = [rarity.pony@msd.com]
        EmailAddress = [rarity.pony@msd.com]
14:52:44,683 DEBUG [SyncplicitySaasProvisioner] updateUser() :: Triggering an update for ::
rarity.pony@msd.com (SaaS ID), rarity.pony@msd.com (SaaS Username)
14:52:44,683 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(3) BEGIN
14:52:44,684 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (saas
user data) :: null
14:52:44,684 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID (user)
:: null
14:52:44,684 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData() :: saas ID
(attribute) :: rarity.pony@msd.com
14:52:44,684 DEBUG [SyncplicitySaasProvisioner] populateUserFromSaasUserData(3) END
14:52:44,684 DEBUG [UserManager] updateUser() :: PROCESSING AN UPDATE USER REQUEST FOR ::
rarity.pony@msd.com :: BEGIN
14:52:44,684 DEBUG [TokenManager] isTokenExpired() :: token expiration set by Syncplicity was set
at :: Mon Feb 26 16:22:43 PST 2018
14:52:44,684 DEBUG [TokenManager] isTokenExpired() :: it is now :: Mon Feb 26 14:52:44 PST 2018
14:52:44,684 DEBUG [TokenManager] isTokenExpired() :: token was created at :: Mon Feb 26 14:52:43
```

```
PST 2018
14:52:44,684 DEBUG [TokenManager] isTokenExpired() :: token expiration set by the Syncplicity
Provisioner was set at :: Mon Feb 26 15:52:43 PST 2018
14:52:44,684 DEBUG [TokenManager] getToken() :: token has not expired yet, reusing existing token
14:52:44,684 DEBUG [TokenManager] getToken() :: token response's token ::
w8lCBe28tmOWA6cUtf8i6Fc4aUtNO9ifMWaVnCZaDIiGM581jW6tz5zAk0DznOj8
14:52:44,684 DEBUG [UserManager] updateUser() :: getting an access token for the update user
request
14:52:44,685 DEBUG [UserManager] processRequest() :: active connections in the pool :: 1
14:52:44,685 DEBUG [UpdateUserRequest] UpdateUserRequest - execute() :: processing update user
request for :: rarity.pony@msd.com
14:52:44,688 DEBUG [UpdateUserRequest] UpdateUserRequest - execute() :: releasing connection
14:52:44,688 DEBUG [UpdateUserRequest] BaseRequest - execute() :: performing PUT request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com
14:52:44,688 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: applying authentication mechanism
14:52:44,688 DEBUG [SyncplicityAccessToken] SyncplicityAccessToken -
applyAuthenticationMechanism() :: releasing connection
14:52:44,818 DEBUG [UpdateUserRequest] BaseRequest - execute() :: received status code :: 200 ::
while performing PUT request on ::
https://xml.syncplicity.com/1.1/auth/user.svc/rarity.pony%40msd.com with the following response
body ::
{"EmailAddress":"rarity.pony@msd.com","FirstName":"Rarity","LastName":"Pony","Disabled":false,"Com
pany":{"Name":"Ping Identity"},"Roles":[],"AccountType":7,"Active":true}
14:52:44,818 DEBUG [UpdateUserRequest] BaseRequest - execute() :: releasing connection
14:52:44,818 DEBUG [UserManager] processRequest() :: finished processing :: UPDATE transaction
request, and received status code :: 200
14:52:44,818 DEBUG [UserManager] processRequest(); :: releasing connection
14:52:44,818 DEBUG [UserManager] updateUser() :: update user request is ok, status code is :: 200
:: for saas ID :: rarity.pony@msd.com
14:52:44,818 DEBUG [UserManager] updateUser() :: releasing connection
14:52:44,818 DEBUG [UserManager] updateUser() :: PROCESSING AN UPDATE USER REQUEST FOR ::
rarity.pony@msd.com :: END
14:52:44,818 DEBUG [SyncplicitySaasProvisioner] updateUser() :: Update processed for ::
rarity.pony@msd.com (SaaS ID), rarity.pony@msd.com (SaaS Username)
14:52:44,818 DEBUG [SyncplicitySaasProvisioner] updateUser() :: END
14:52:44,818 DEBUG [JdbcUserStore] Setting SaaS GUID for user in JDBC store: channel=4,
dsGUID=b96ae5d22be29949bc946bcb4d397bb6, saasGuid=rarity.pony@msd.com []
14:52:45,080 DEBUG [JdbcUserStore] Clearing user in JDBC store: channel=4,
dsGUID=b96ae5d22be29949bc946bcb4d397bb6 []
14:52:45,342 DEBUG [JdbcUserStore] Setting SaaS Identity for user in JDBC store: channel=4,
dsGUID=b96ae5d22be29949bc946bcb4d397bb6,
saasIdentity=eyJhbGciOiJkaXIiLCJlbmMiOiJBMTI4Q0JDLUhTMjU2Iiwia2lkIjoiZ2FCQkR5UzdlZCIsInZlcnNpb24iO
iI4LjAuNC4wIn0..mditAx9Pn30oIlCeyqbt1Q.BNoKHIbvkqCc151Han9-Cekh3yx2-
Yxwp4xecvrYoek.lp3Qm8V9LQj4cfxbtyQSoQ []
14:52:45,606 INFO  [SystemOut] 14:52:45,606 INFO  [UserActivityLogger] Updated:
rarity.pony@msd.com
```

Syncplicity >> admin >>user accounts


syncplicity
by axway
Sue Chang (Ping Identity Administrator) Logout | Support
Activity Feed
Files
Shared Links
Install
Account
Admin
Dashboard • Reports • User Accounts • Groups • Devices • Folders • Panorama • Policies • Settings • Downloads • Billing
Add a User
Filter by User Name or Email address...
Other Tasks
Add new users
Manage Policy Sets
10 per page
previous | next

Manage Users

User Accounts (25 users)

□ Show Pending Offboarding users only

21-25 of 25

User	Last Activity	Storage Quota	Total Storage Consumed
Apple Jack ajack@msd.com	never logged in	--	0 bytes
Cersei Lannister cersei.lannister@msd.com	never logged in	--	0 bytes
Daenerys Targaryen daenerys.targaryen@msd.com	never logged in	--	0 bytes
Jon Snow jon.snow@msd.com	never logged in	--	0 bytes
Rarity Pony rarity.pony@msd.com	never logged in	--	0 bytes

21-25 of 25

Scenario 4: Update Syncplicity profile due to user Name Change

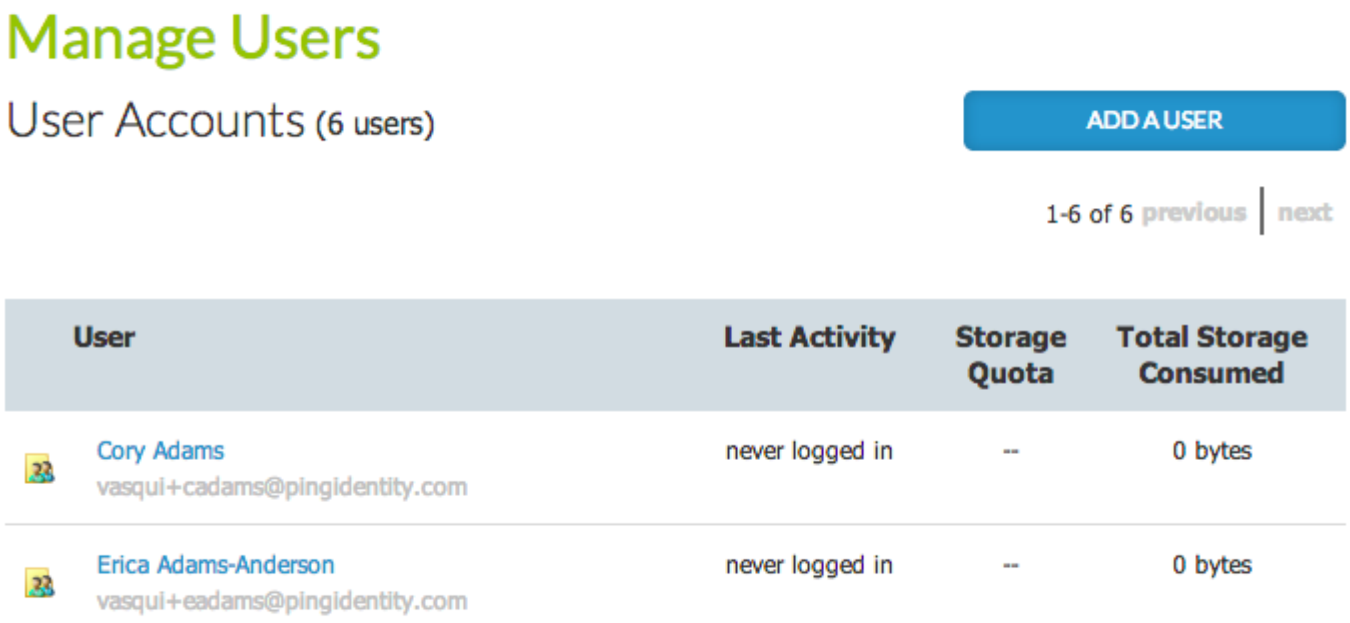
In this scenario we created a user (Erica Adams) as a member of the AD Provisioning group.
Then we changed her last name in Active Directory to Adams-Anderson and confirmed the name change was reflected in the Syncplicity profile within minutes.

Provisioner.log (<pf install directory>\pingfederate\log)

```
12:48:54,756 DEBUG [SpringLdapDirectoryDriver] Getting group modify time for:
CN=SyncplicityUsers,CN=Users,DC=corp,DC=partnera,DC=com; using attribute: modifyTimestamp
Channel: channel1
12:48:54,756 DEBUG [GenericUserProvisioningChannel]         not needed.  Channel: channel1
12:48:54,756 DEBUG [GenericUserProvisioningChannel]  Collecting changed users...  Channel:
channel1
12:48:54,772 DEBUG [JdbcUserStore] Updating user in JDBC store: channel=1,
saasUsername=vasqui+eadams@pingidentity.com, hash=0264ef92f53ac38ad6e7018a40c47fa1,
saasIdentity=EmailAddress&true&EmailAddress=vasqui%2Beadams%40pingidentity.com&FirstName=Erica&LastName=Adams-Anderson&Roles=BLANK&AccountType=7, dsGUID=c66c0cd64c5a8c458bebc51fb380f96d  Channel:
channel1
```

```
12:48:54,788 DEBUG [SaasProvisioningEventTracker] User provisioning event during cycle
1381430934756 in channel 1 @ Thu Oct 10 12:48:54 MDT 2013 type: 'UPDATE_USER_INTERNAL' successful:
'true' source id: 'c66c0cd64c5a8c458bebc51fb380f96d' target id: 'vasqui+eadams@pingidentity.com'
Channel: channel1
12:48:54,788 DEBUG [AttributeSyncCallback] User marked for update, GUID:
c66c0cd64c5a8c458bebc51fb380f96d  Channel: channel1
12:48:54,788 DEBUG [PagingLdapTemplate] Simple Paged Search: Page size: 500, Result size: 1
Channel: channel1
12:48:54,788 DEBUG [GenericUserProvisioningChannel]  Clearing dirty user records...  Channel:
channel1
12:48:55,616 DEBUG [JdbcUserStore] Clearing user in JDBC store: channel=1,
dsGUID=c66c0cd64c5a8c458bebc51fb380f96d  Channel: channel1
12:48:55,647 INFO  [UserActivityLogger] Updated: vasqui+eadams@pingidentity.com  Channel: channel1
12:48:55,647 DEBUG [SaasProvisioningEventTracker] User provisioning event during cycle
1381430934756 in channel 1 @ Thu Oct 10 12:48:55 MDT 2013 type: 'UPDATE' successful: 'true' source
id: 'c66c0cd64c5a8c458bebc51fb380f96d' target id: 'vasqui+eadams@pingidentity.com'  Channel:
channel1
```

Syncplicity >> admin >>user accounts



Manage Users

User Accounts (6 users)

User	Last Activity	Storage Quota	Total Storage Consumed
Cory Adams vasqui+cadams@pingidentity.com	never logged in	--	0 bytes
Erica Adams-Anderson vasqui+eadams@pingidentity.com	never logged in	--	0 bytes

Scenario 5: Update Syncplicity profile due to user Email Change

In this scenario we created a user (Erica Adams) as a member of the AD Provisioning group. Then we changed her email in Active Directory (from erica.adams@pingidentity.com to eadams@pingidentity.com), and confirmed the email change was reflected in the Syncplicity profile within minutes.

Server.log (<pf install directory>\pingfederate\log)

```
2015-10-06 14:39:08,704  DEBUG
[com.pingidentity.clientservices.product.provisioner.SyncplicitySaasProvisioner] updateUser() ::
```

```
Update triggering an update on + (eadams@pingidentity.com (SaaS GUID), eadams@pingidentity.com
(SaaS Username))
2015-10-06 14:39:08,704  DEBUG
[com.pingidentity.clientservices.product.provisioner.SyncplicitySaasProvisioner]
populateUserFromSaasUserData() :: guid (saas user data):: erica.adams@pingidentity.com
2015-10-06 14:39:08,704  DEBUG
[com.pingidentity.clientservices.product.provisioner.SyncplicitySaasProvisioner] updateUser() ::
SaaS GUID is being updated :: erica.adams@pingidentity.com (before), eadams@pingidentity.com
(after)
2015-10-06 14:39:08,704  DEBUG
[com.pingidentity.clientservices.product.provisioner.token.TokenManager] Testing tokenExpiration
2015-10-06 14:39:08,813  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] UpdateUserRequest()
:: user saas id :: erica.adams@pingidentity.com
2015-10-06 14:39:08,813  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UserManager] Connections in Pool: 1
2015-10-06 14:39:08,813  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] execute() :: user
saas id :: erica.adams@pingidentity.com
2015-10-06 14:39:08,813  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] getEndpointURL() ::
saasIdToQuery :: erica.adams@pingidentity.com
2015-10-06 14:39:08,813  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] createRequestURL() ::
url :: https://xml.syncplicity.com/1.1/auth/user.svc/erica.adams%40pingidentity.com
2015-10-06 14:39:08,813  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] execute() :: url ::
https://xml.syncplicity.com/1.1/auth/user.svc/erica.adams%40pingidentity.com
2015-10-06 14:39:08,844  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] execute() ::
performing PUT on https://xml.syncplicity.com/1.1/auth/user.svc/erica.adams%40pingidentity.com
2015-10-06 14:39:08,954  DEBUG
[com.pingidentity.clientservices.product.provisioner.user.UpdateUserRequest] execute() :: received
status code :: 200
2015-10-06 14:39:08,954  DEBUG
[com.pingidentity.clientservices.product.provisioner.SyncplicitySaasProvisioner] updateUser() ::
Update processed on + (eadams@pingidentity.com (SaaS GUID), eadams@pingidentity.com (SaaS
Username))
```

6.0 Troubleshooting

Error [The role was not specified as "Administrator" or "User"] in Provisioner.log

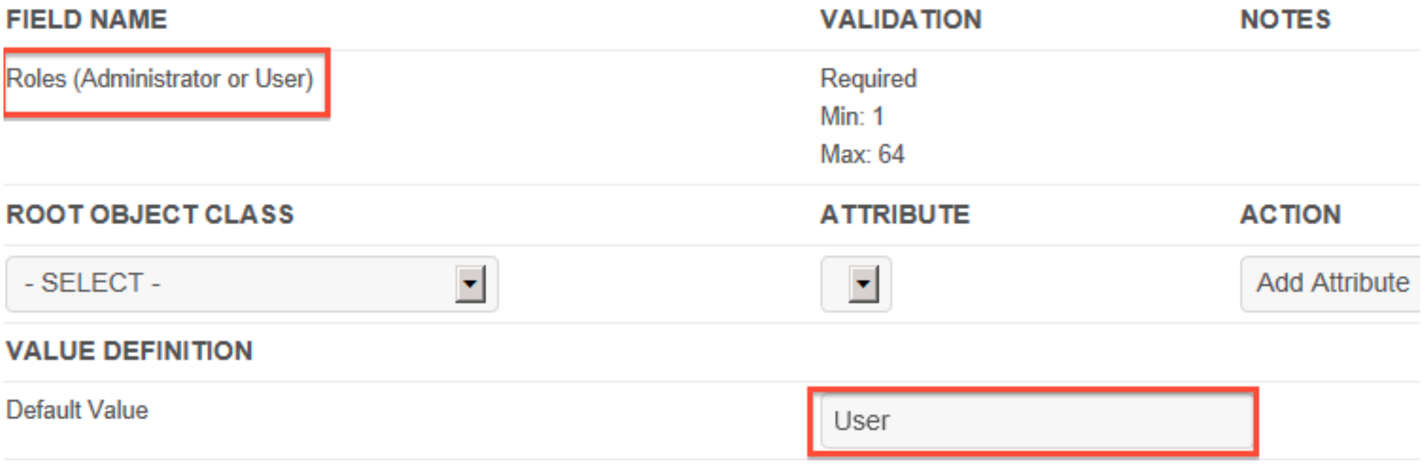
Issue: This error can occur if you failed to specify Administrator or User as the default value for the Role attribute in the Attribute Mapping for the Syncplicity Provisioning channel.

Solution:

1. Login to your PingFederate Administrator console and open the Syncplicity SP Connection
    a. Go to the Outbound Provisioning page, click Configure Provisioning
    b. On the Configure Channels >> Manage Channels page, click on your Syncplicity channel (if you have more than one you'll want to repeat this process for each).
    c. Click on Attribute Mapping

d. For the “Roles (Administrator or User)” field, click the Edit link (under Action)

e. Set the Default value to either User or Administrator then click Done



FIELD NAME	VALIDATION	NOTES
Roles (Administrator or User)	Required Min: 1 Max: 64	

ROOT OBJECT CLASS	ATTRIBUTE	ACTION
- SELECT -		Add Attribute

VALUE DEFINITION

Default Value	User

f. On the Attribute Mapping page, click Done

g. On the Manage Channels page, click Done

h. On the Outbound Provisioning page, click Save

i. On the Activation & Summary page, click Save

j. You will then need to re-trigger the action for the user that was encountering the error to ensure they get provisioned properly at Syncplicity (you do that by adding or removing them from the provisioning group). If you forget to do this, you’ll continue to see the error in the log.

Provisioner.log sample

```
14:35:10,037 ERROR [AbstractSaasPluginBase] Exception in doCreateUser(). The role was not
specified as "Administrator" or "User"  Channel: channel1
14:35:10,037 DEBUG [SaasProvisioningEventTracker] User provisioning event during cycle
1384378509396 in channel 1 @ Wed Nov 13 14:35:10 MST 2013 type: 'CREATE' successful: 'false'
source id: '606937bb1ab98742bba016a433730a22' target id: 'vasqui+dadams@pingidentity.com' cause:
'com.pingidentity.provisioner.sdk.SaasPluginException: The role was not specified as
"Administrator" or "User"'  Channel: channel1
14:35:10,037 ERROR [SynchronousSaasCallback] Cannot process resource:
vasqui+dadams@pingidentity.com  Channel: channel1
com.pingidentity.provisioner.saas.SaasException:
com.pingidentity.provisioner.sdk.SaasPluginException: The role was not specified as
"Administrator" or "User"
       at
com.pingidentity.provisioner.sdk.internal.AbstractSaasPluginBase.doCreateUser(AbstractSaasPluginBa
se.java:118)
       at com.pingidentity.provisioner.saas.BaseSaasPlugin.updateUser(BaseSaasPlugin.java:165)
       at
com.pingidentity.provisioner.channel.synchronous.SynchronousSaasUserRunner.run(SynchronousSaasUser
Runner.java:28)
       at java.lang.Thread.run(Thread.java:722)
Caused by: com.pingidentity.provisioner.sdk.SaasPluginException: The role was not specified as
"Administrator" or "User"
       at
com.pingidentity.clientservices.syncplicity.provisioner.SyncplicitySaaSProvisioner.populateUserFro
mSaasUserDate(SyncplicitySaaSProvisioner.java:150)
       at
com.pingidentity.clientservices.syncplicity.provisioner.SyncplicitySaaSProvisioner.createUser(Sync
plicitySaaSProvisioner.java:116)
       at
com.pingidentity.provisioner.sdk.internal.AbstractSaasPluginBase.doCreateUser(AbstractSaasPluginBa
se.java:104)
       ... 3 more
```

```
14:35:10,349 DEBUG [JdbcUserStore] Clearing user in JDBC store: channel=1,
dsGUID=c66c0cd64c5a8c458bebc51fb380f96d  Channel: channel1
```

Error [403] in Provisioner.log

Issue: This error can occur if your Application Token is misconfigured.

Solution:

2. Please login to Syncplicity and go to the Account page
    a. Copy the Application Token value listed under Account
3. Login to your PingFederate Administrator console and open the Syncplicity SP Connection
    a. Go to the Outbound Provisioning page, click Configure Provisioning
    b. On the Configure Channels page, click on Target
    c. Verify the Application Token value here matches what you copied from Syncplicity in step 1.

Provisioner.log sample

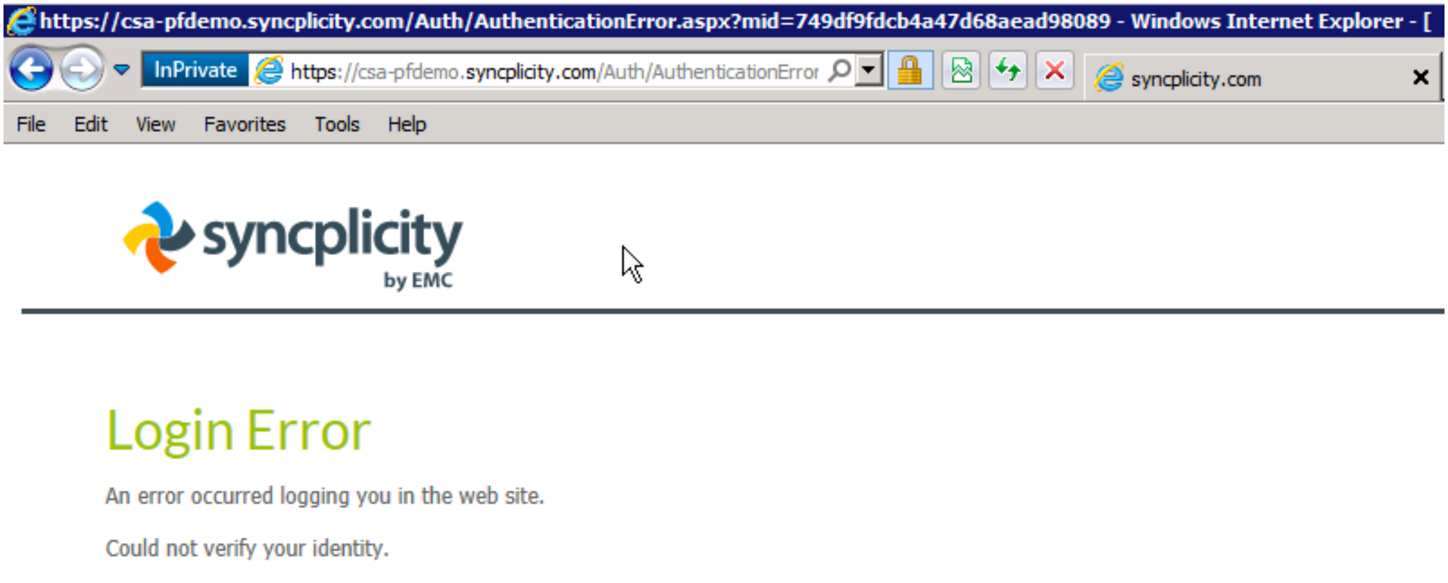
```
09:31:52,670 DEBUG [SpringLdapDirectoryDriver] Getting group modify time for:
CN=SyncplicityUsers,CN=Users,DC=corp,DC=partnera,DC=com; using attribute: modifyTimestamp
Channel: channel1
09:31:52,670 DEBUG [GenericUserProvisioningChannel]         not needed.  Channel: channel1
09:31:52,670 DEBUG [GenericUserProvisioningChannel]   Collecting changed users...  Channel:
channel1
09:31:52,670 DEBUG [GenericUserProvisioningChannel]   Clearing dirty user records...  Channel:
channel1
09:31:52,749 ERROR [SynchronousSaasCallback] Cannot process resource:
vasqui+asmith@pingidentity.com  Channel: channel1
com.pingidentity.provisioner.sdk.SaasPluginException: response with error code [403] with body
       at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenResponse.fromHttpResponse(Token
Response.java:111)
       at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenManager.initToken(TokenManager.
java:51)
       at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenManager.getToken(TokenManager.j
ava:27)
       at
com.pingidentity.clientservices.syncplicity.provisioner.user.UserManager.getUser(UserManager.java:
37)
       at
com.pingidentity.clientservices.syncplicity.provisioner.SyncplicitySaaSProvisioner.getUser(Syncpli
citySaaSProvisioner.java:224)
       at
com.pingidentity.provisioner.sdk.internal.AbstractSaasPluginBase.doGetUser(AbstractSaasPluginBase.
java:127)
       at com.pingidentity.provisioner.saas.BaseSaasPlugin.updateUser(BaseSaasPlugin.java:149)
       at
com.pingidentity.provisioner.channel.synchronous.SynchronousSaasUserRunner.run(SynchronousSaasUser
Runner.java:28)
       at java.lang.Thread.run(Thread.java:722)
09:31:52,811 ERROR [SynchronousSaasCallback] Cannot process resource:
vasqui+bsmith@pingidentity.com  Channel: channel1
com.pingidentity.provisioner.sdk.SaasPluginException: response with error code [403] with body
       at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenResponse.fromHttpResponse(Token
Response.java:111)
       at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenManager.initToken(TokenManager.
java:51)
        at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenManager.getToken(TokenManager.j
ava:27)
        at
com.pingidentity.clientservices.syncplicity.provisioner.user.UserManager.getUser(UserManager.java:
37)
        at
com.pingidentity.clientservices.syncplicity.provisioner.SyncplicitySaaSProvisioner.getUser(Syncpli
citySaaSProvisioner.java:224)
        at
com.pingidentity.provisioner.sdk.internal.AbstractSaasPluginBase.doGetUser(AbstractSaasPluginBase.
java:127)
        at com.pingidentity.provisioner.saas.BaseSaasPlugin.updateUser(BaseSaasPlugin.java:149)
        at
com.pingidentity.provisioner.channel.synchronous.SynchronousSaasUserRunner.run(SynchronousSaasUser
Runner.java:28)
        at java.lang.Thread.run(Thread.java:722)
09:31:52,874 ERROR [SynchronousSaasCallback] Cannot process resource: vasqui@pingidentity.com
Channel: channel1
com.pingidentity.provisioner.sdk.SaasPluginException: response with error code [403] with body
        at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenResponse.fromHttpResponse(Token
Response.java:111)
        at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenManager.initToken(TokenManager.
java:51)
        at
com.pingidentity.clientservices.syncplicity.provisioner.token.TokenManager.getToken(TokenManager.j
ava:27)
        at
com.pingidentity.clientservices.syncplicity.provisioner.user.UserManager.getUser(UserManager.java:
37)
        at
com.pingidentity.clientservices.syncplicity.provisioner.SyncplicitySaaSProvisioner.getUser(Syncpli
citySaaSProvisioner.java:224)
        at
com.pingidentity.provisioner.sdk.internal.AbstractSaasPluginBase.doGetUser(AbstractSaasPluginBase.
java:127)
        at com.pingidentity.provisioner.saas.BaseSaasPlugin.updateUser(BaseSaasPlugin.java:149)
        at
com.pingidentity.provisioner.channel.synchronous.SynchronousSaasUserRunner.run(SynchronousSaasUser
Runner.java:28)
        at java.lang.Thread.run(Thread.java:722)
```

Syncplicity Login Error

Issue: This error can occur if your username has not yet been provisioned with a user account in Syncplicity.

Solution:

1. Verify the user is a member of the Active Directory security group or AD Filter that you specified to target Syncplicity users for provisioning.
2. Check the provisioner.log to see if there were any errors attempting to provision this user account.
3. Have the user re-try once the provisioning task is completed successfully.



https://csa-pfdemo.syncplicity.com/Auth/AuthenticationError.aspx?mid=749df9fdcb4a47d68aead98089 - Windows Internet Explorer - [

Login Error

An error occurred logging you in the web site.

Could not verify your identity.

User’s Name in Syncplicity changes after SSO login

Known Issue: Once a user is Provisioned in Syncplicity and they login via SSO for the first time, their First Name in Syncplicity gets changed to their email address value.

7.0 Logging

To enable various logging modes for the Syncplicity Provisioner, add the following in the logger section (near the bottom) to <PingFederateInstall>/pingfederate/server/default/conf/log4j2.xml:

```
<Logger name="com.pingidentity.clientservices.product.provisioner" level="[ DEBUG | INFO | WARN | ERROR ]" />
```

Log4J Standard Levels:

Standard Level	intLevel
OFF	0
FATAL	100
ERROR	200
WARN	300
INFO	400
DEBUG	500
TRACE	600
ALL	`Integer.MAX_VALUE`

Reduced logging begins at the first level to more detailed at the bottom.

Detailed training on using Log4j in PingFederate can be found at:
https://ping.force.com/Support/PingIdentityKnbSearchHome?searchText=log4j