Use Cases

June 30, 2025

USE CASES

Table of Contents

Use Cases Overview

Learn how to configure Ping products with each other and with selected third-party products to enable you to run your security business better. If you need additional, detailed information for a Ping product, follow the links in the guides or go to the Ping Documentation home page.

Ping Identity Use Cases

- Getting Started Guides
- Best Practice Guides
- Standards and Protocols Use Cases
- Customer Use Cases
- Workforce Use Cases
- Single Sign-on Use Cases
- Multi-factor Authentication Use Cases
- Data and Application Security Use Cases
- Directory Use Cases
- Developer API Use Cases

PingOne Advanced Identity Cloud Use Cases

- Administration
- Authentication
- Customization
- Data (identity) management
- Reports
- Self-service
- Third-party integration

Other Use Cases

- Ping Identity configuration guides
- PingOne DaVinci use cases
- PingAccess use cases
- PingOne use cases
- PingOne API use cases

Getting Started Guides

Use case	Description
Getting started with PingAccess	Use this procedure to guide you in setting up PingAccess.
Getting Started with PingFederate	Use this workflow to guide you in setting up PingFederate.
Getting started with PingID core settings	Learn how to configure and use core PingID settings, including a detailed walkthrough of the advanced options.
Performing a near-zero downtime PingFederate upgrade	Use a near-zero downtime upgrade process to reduce PingFederate's downtime to minutes, minimizing the impact on your users.

Getting started with PingAccess

Use this procedure to guide you in setting up PingAccess. Learn more in PingAccess documentation.

Before you begin

Component

- PingAccess 6.3

Do the following:

- Set up a Ping Identity account username and password.
- If you do not have a PingAccess license, contact sales@pingidentity.com.
- To get familiar with how PingAccess works, see Introduction to PingAccess (page 41).

Steps

1. Download PingAccess from PingAccess downloads.
2. Install PingAccess (page 53).
    1. Start PingAccess.
    2. Access the PingAccess administrative console.
3. Import the PingAccess license (page 348)
4. Sign on and accept the end-user license agreement (EULA).
5. Configure the PingAccess token provider (page 520).

> **Note**
>
> A token provider is used to provide identity information to PingAccess, which can be used to grant or deny access to an application. You can use PingFederate as the token provider.

6. Configure the PingAccess virtual hosts (page 254).

Getting Started with PingFederate

Use this workflow to guide you in setting up PingFederate.

Component

PingFederate 10.3

Prerequisites

- If you do not have a PingFederate license, contact sales@pingidentity.com.
- Make sure your environment meets the system requirements (page 109).

Key terms and concepts

You can find more information on the following terms and concepts in Introduction to PingFederate (page 26) and its subtopics.

Identity provider (IdP)

A trusted provider that issues authentication assertions to grant access to other resources.

Service provider (SP)

A provider that receives authentication assertions from an IdP and grants or denies resource access.

WS-Trust Security Token Service (STS)

A protocol for systems and applications to use when requesting a service to issue, validate, and exchange security tokens.

OAuth 2.0

A protocol for securing application access to protected resources by issuing access tokens to clients of Representational State Transfer (REST) APIs, and non-REST APIs.

Browser-based SSO

Enables users to securely authenticate with multiple applications and websites by logging in only once.

Downloading and installing PingFederate

Steps

1. Download PingFederate.
2. Install Pingfederate (page 108).
3. Start PingFederate (page 106) and then open the administrative console (page 158).

The first time you open the administrative console, PingFederate guides you through the setup wizard (page 159).

4. Familiarize yourself with the PingFederate administrative console (page 158).

   The PingFederate user interface consists of menus, windows, and tabs.

Additional information

After you finish setting up PingFederate, you can begin the following tasks:

- Create an IdP adapter (page 678).

  An IdP adapter is used to look up session information and provide user identification to PingFederate.

- Create an SP connection (page 411).

  As an IdP, you manage connection settings to support the exchange of federation-protocol messages (SAML, WS-Federation, or WS-Trust) with an SP or STS client application at your site.

- Create an SP adapter (page 667).

  An SP adapter is used to create a local-application session for a user in order for PingFederate to provide SSO access to your applications or other protected resources.

- Create an IdP connection (page 678)

  As a Service Provider, you manage connection settings to support the exchange of federation-protocol messages (OpenID Connect, SAML, WS-Federation, or WS-Trust) with an IdP, OAuth client, OpenID Provider (OP), or STS client application at your site.

- You can download the PingFederate Security Hardening Guide for security-related best practices.

  > **Note**
  >
  > This requires a Ping Identity account.

- Integrate PingFederate with a supported hardware security module (HSM) (page 168).

  Standards such as the Federal Information Processing Standard (FIPS) 140-2 require the storage and processing of all keys and certificates on a certified cryptographic module.

Learn more in PingFederate documentation.

Getting started with PingID core settings

Learn how to configure and use core PingID settings, including a detailed walkthrough of the advanced options.

This guide covers only the configuration of the core PingID settings. For tasks such as configuring an identity provider, branding, PingID policies, and device pairing settings, see the PingID Administration Guide.

Before you begin

You must:

- Have a web browser with connectivity to the Internet.
- Have a PingID instance that needs to be newly configured or that needs a configuration change.

Configuring PingID quick setup

Configure PingID with the most common settings.

About this task

To get your PingID instance up and running quickly:

Steps

1. In your PingOne environment, click the **Overview** tab.
2. From the **Services** section, click the PingID icon.
3. Go to **Setup → PingID → Configuration**.

   > **Note**
   >
   > If you're using a trial mode of PingID, you see a notice at the top of the **PingID Settings** page indicating that your features are limited.
   > To upgrade your license, contact sales@pingidentity.com.

4. In the **Admin Message** field, enter contact information to aid users experiencing trouble.

   Example:

   For example: `For support with PingID, contact helpdesk@mycompany.com.`

5. To give your users time to get acquainted with the addition of multi-factor authentication (MFA), in the **Mandatory Enrollment Date** section, enter a date 30 days from the current date.
6. To allow users to self-enroll an MFA device the first time they're prompted for MFA, in the **Self-enrollment During Authentication** section, click **Enable**.
7. To allow your users to enroll multiple devices, in the **Maximum Allowed Devices** field, enter a number greater than 1.
8. To prevent your users from having to select an MFA device every time they authenticate, in the **Device Selection** section, click **Default to Primary**.
9. To allow users to disconnect their mobile device without administrator approval, in the **Device Management** section, select the **Allow users to unpair and change devices using the mobile app** checkbox.
10. To allow users to manage their devices in the PingID console, in the **Device Management** section, select the **Allow users to manage their devices on the web** and **Enable device management for users with no paired devices** checkboxes.
11. To avoid sending an email to a user every time a new MFA device is added, in the **Email Notification for New Devices** section, click **Disable**.

12. To maintain the 40-second MFA challenge timeout default, in the **New Request Duration** section, click **Default**.
13. To allow users to use a one-time passcode (OTP) on their mobile app if a push notification doesn't reach their device, in the **One-time Passcode Fallback** section, click **Enable**.
14. To require a mobile device to attempt a push notification before the user can use an OTP, in the **Direct Passcode Usage** section, click **Disable**.
15. To allow the use of a device's native biometrics for an MFA challenge, in the **Device Biometrics** section, click **Enable**.
16. To enable both iOS and Android devices, in the **Enable On** section, select the **iOS** and **Android** checkboxes.
17. To prevent accidental automatic MFA approvals when FaceID is enabled, in the **Face ID Consent** section, click **Enable**.
18. To require users to unlock their device before approving an MFA challenge, in the **Authentication While Device is Locked** section, click **Disable**.
19. To enable the most common MFA types, in the **Alternate Authentication Methods** section, select the **Enable** and **Pairing** checkboxes for **SMS**, **Voice**, and **Email**.
20. To allow users to provide their own phone numbers and email addresses when enrolling a device, clear the **Pre-populate** and **Restrict** checkboxes for **SMS**, **Voice**, and **Email**.
21. To maintain English as the supported language, in the **Local Language for Voice Calls** section, click **Disable**.
22. To prevent abuse of SMS or Voice services, set the **Daily Used SMS/Voice Limit** field to `15` and the **Daily Unused SMS/Voice Limit** field to `10`.
23. To use the default PingID SMS and voice provider, in the **Twilio Account** section, click **Ping Identity**.
24. To use English for all SMS messages, in the **Local Language for SMS** section, click **Disable**.
25. To enable the simple use of the Desktop OTP application:
    1. In the **Desktop Security PIN** section, click **Disable**.
    2. In the **Use Proxy for Desktop** section, click **Disable**.
26. To enable a simple configuration for security keys:
    1. In the **Resident Key** section, click **Not Required**.
    2. In the **User Verification** section, click **Preferred**.
27. To allow enforcement of MFA policies that you define, in the **Enforce Policy** section, click **Enable**.
28. To enforce policies specifically for the PingID Windows login agent, in the **Enforce Policy for Windows Login**, click **Enable**.
29. If you are using a trial version of PingID, to prevent your users from being locked out if your trial expires, in the **Evaluation** section, click **Allow single sign-on without PingID**.

PingID advanced setup

Use this topic for a description each of the settings in the PingID configuration and how to use them.

When you are ready to customize your configuration beyond the recommended defaults, use the following tables to determine the settings that best meet your business and technical needs.

Support

Section	Description
Admin Message	The end user sees the **Admin Message** field when a multi-factor authentication (MFA) challenge is issued. The message should provide directions on getting help if the user has trouble signing on. For example: `In the event of difficulty, contact the Helpdesk at helpdesk@mycompany.com.` This field is optional.

Enrollment

Section	Description
Mandatory Enrollment Date	The **Mandatory Enrollment Date** section specifies the last date an end user can choose not to enroll a device in PingID. When users are presented with an MFA challenge for the first time, they are prompted to enroll a device in the PingID service. This option allows existing users a grace period before requiring enrollment in PingID. **Note** • Until the specified mandatory enrollment date, a **Not Now** button is shown on the enrollment page, allowing the user to bypass both the enrollment process and the MFA challenge. • On and after the mandatory enrollment date, the **Not Now** button is not shown and the user must enroll a device in the PingID service to authenticate.
Self-Enrollment During Authentication	The **Self-Enrollment During Authentication** section specifies whether the end user is presented with the built-in PingID enrollment process during the user's first MFA challenge: • Selecting **Enable** allows users to self-register a device with PingID and is the appropriate choice for most organizations. This is the default value. • Selecting **Disable** results in an error when a user without a registered device is prompted for MFA. Select this option if your organization needs to implement custom business processes as part of the user's first MFA challenge. **Note** If **Disable** is selected, the **Admin Message** field should direct the user to the custom enrollment process of the organization. For example: `To enroll a device in PingID, visit http://mycompany.com/registerMFA.`

Devices

Section	Description
Maximum Allowed Devices	The **Maximum Allowed Devices** setting specifies the maximum number of devices each user can enroll in PingID for MFA challenges. This provides a fallback in the event that a primary device is lost, stolen, or damaged. It also allows organizations to create policies that require a specific device to be used in different MFA challenges. **Note** Each additional device that a user enrolls increases the attack surface for that user. Organizations should balance user convenience with security when choosing a value for the **Maximum Allowed Devices** section. The default value is `5`.
Device Selection	The **Device Selection** option specifies whether a user's primary device is used as the default for MFA. This option is shown when **Maximum Allowed Devices** is greater than 1. • If **Default to Primary** is selected, the primary device is always used unless a PingID policy overrides this setting. If the primary device doesn't respond to the MFA challenge in time, the user is prompted to select an alternate device if more than one is enrolled. This is the default setting and ensures a smooth, fast MFA experience for end users. • If **Prompt User to Select** is selected, the user is prompted to choose an enrolled device every time they receive an MFA challenge. **Note** Choosing **Prompt User to Select** generates additional user activity during MFA challenges.

Section	Description
Device Management	The **Device Management** section has three options: • **Allow Users to Unpair and Change Devices Using the Mobile App** lets users unpair a mobile device from within the PingID mobile application. It also allows the user to move their PingID enrollment from one mobile device to another. This is the default selection. **Note** This doesn't allow users to add new devices, only to move from one mobile device to another. If all mobile devices are issued by the organization to its users, it is recommended that this selection be cleared. This prevents users from removing their company-issued mobile device from PingID. If individuals are allowed to use personal mobile devices, select **Allow Users to Unpair and Change Devices Using the Mobile App**. • **Allow Users to Manage Their Devices on the Web** lets users manage their MFA devices in the devices section of the PingOne dock. • **Enable Device Management for Users with No Paired Devices** lets users with no paired devices manage their devices in the **Devices** section of the PingOne dock. If this option is disabled, users must self-enroll during authentication before they can access the **Devices** section. **Note** This option requires that you select the **Allow Users to Manage Their Devices on the Web** checkbox.
Email Notification For New Devices	The **Email Notification for New Devices** section specifies whether PingID sends an email notification to the end user when a new MFA device is enrolled for their account: • Selecting **Disable** doesn't notify the end user when new devices are registered for their account, and they aren't able to report fraudulent activity. This is the default value. • Selecting **Enable** sends the end user an email when a new MFA device has been registered for their account. The email shows the type of device that was registered and a link to report to the PingID administrator if the action was fraudulent.

Mobile App Authentication

Section	Description
New Request Duration	The **New Request Duration** setting defines the maximum amount of allowed time for an MFA challenge to reach a device before timing out as well as the total amount of time allowed for an MFA response before timeout: • If **Default** is selected, the amount of time for an MFA challenge to reach a device before timing out is 25 seconds, and the total amount of time allowed for an MFA response before timeout is 40 seconds. This means that the user has 15 seconds to respond to a challenge after receiving it. **Note** Timeouts for the **Default** value apply to all MFA challenges. • If **Global** is selected, **Device Timeout** and **Total Timeout** settings are displayed. For the **Global** value, values for **Device Timeout** and **Total Timeout** apply to all MFA challenges. This is the recommended choice for most organizations. • The **Device Timeout** setting defines how much time is allowed for an MFA challenge to reach a device before timeout and allows the organization to override the default of 25 seconds. • The **Total Timeout** setting defines how much time is allowed for an MFA challenge response before timeout and allows the organization to override the default of 40 seconds. ◦ If **Advanced** is selected, options for Web single sign-on (SSO), API, SSH, and VPN MFA challenges are presented, allowing the organization to set **Device Timeout** and **Total Timeout** for each different type of MFA challenge. **Note** For the **Global** or **Advanced** settings, you must set **Total Timeout** to at least 15 seconds greater than the **Device Timeout** value.
One-Time Passcode Fallback	This allows the organization to configure whether the end user can use a one-time passcode (OTP) within the PingID mobile application to complete an MFA challenge if the mobile push notification times out: • The **Enable** value is selected by default, and allows the user to use the OTP on their mobile device if a mobile push notification is not completed before timeout. **Tip** This is useful when a mobile device doesn't have data connectivity or has poor coverage. • The **Disable** value doesn't allow the use of an OTP in the event of a mobile push notification timeout. The user can only retry the mobile push and not use an OTP.

Section	Description
Direct Passcode Usage	If **One-Time Passcode Fallback** is set to **Enable**, the **Direct Passcode Usage** option is displayed. **Direct Passcode Usage** configures whether the end user can use an OTP to complete an MFA challenge before a mobile push notification times out: • If **Disable** is selected, the user can enter an OTP if a mobile push notification times out but not before. This is the default value. • If **Enable** is selected, the user is presented with a **Use Code** button during a mobile push notification, which allows the user to enter an OTP.

Section	Description
Device Biometrics	The **Device Biometrics** section determines whether the PingID mobile app can use the native biometric capabilities of the mobile device, such as fingerprint authentication or face recognition: • If **Disable** is selected, the PingID mobile application won't use the native biometric capabilities of the mobile device, and the end user must always swipe on their mobile device to complete MFA. • If **Enable** is selected, the PingID mobile app can use the mobile device's native biometric capabilities, such as fingerprint authentication or face recognition, depending on the device's capabilities. When **Enable** is selected, the **Enable On** and **Face ID Consent** sub-settings display: ◦ **Enable On** determines whether device biometrics can be used on Apple devices, Android devices, or both. If neither **Apple** nor **Android** is selected, **Device Biometrics** is set to **Disable**. ◦ **Face ID Consent** determines whether the user must consent to Face ID before a push notification is approved. If **Face ID Consent** is disabled, PingID will automatically approve a push notification if the end user is looking at their phone with the PingID app open. If **Face ID Consent** is enabled, a notification prompts the end user to confirm they wish to authenticate with Face ID. This option prevents end users from automatically approving MFA challenges without being able to review whether the challenge is valid. • If **Required** is selected, the PingID mobile app only accepts the biometric capabilities of the mobile device, as opposed to also presenting the swipe option. Enrolled mobile devices must have biometric capabilities configured for the device. If biometric capabilities are not configured or are disabled for the PingID mobile application, the user receives an error on their mobile device during MFA and cannot complete the MFA challenge. When selected, the **Notification Actions** option displays. **Notification Actions** determines can complete MFA challenges from mobile notification banners: ◦ If **Disable** is selected for **Notification Actions**, the user cannot act upon an MFA challenge from a notification banner and is required to unlock the phone and open the PingID application to complete MFA. ◦ If **Enable** is selected for **Notification Actions**, the user can take action on an MFA challenge from a notification banner. For a full description of the behavior of Apple and Android devices for the **Disable** and **Enable** options of **Notification Actions**, review the **Cases Matrix for iPhone iOS 8+ Devices** and **Cases Matrix for Android 5.0+ Devices** tables in Configuring authentication for the PingID mobile app.

Section	Description
Authentication While Device is Locked	The **Authentication While Device is Locked** section determines whether the PingID mobile application presents the swipe option over the Android lock screen. Enabling this setting streamlines the user experience on Android devices, but also makes it easier for a fraudulent MFA approval. Organizations should weigh the user experience against the weaker security footprint when configuring this setting: • If **Disable** is selected, the user must unlock their Android device before completing MFA. • If **Enable** is selected, the PingID application presents a swipe request over the Android device's lock screen. **Note** This setting applies only to versions of Android older than Android Q. As of Android Q, application notifications are no longer allowed over the device's lock screen.

Alternate Authentication Methods

The following table shows the options for SMS, Voice, Email, YubiKey, Desktop, Security Key, OATH Tokens, and FIDO2 Biometrics.

Option	Description
Enable	Selecting the **Enable** checkbox of the corresponding item enables the use of that type of device for MFA challenges within PingID. **Note** If the **Enable** checkbox is cleared, that device type is not supported for your organization within PingID, and the user is unable to register such a device.
Pairing	Selecting the **Pairing** checkbox of the corresponding authentication method allows device pairing for that method. This checkbox is automatically selected when an authentication method is enabled. Disabling pairing is useful to phase out a specific method of authentication without blocking existing users from authenticating. **Note** When pairing is disabled, devices that are already paired are not affected, and the corresponding method is still available as a backup authentication method.

Option	Description
Pre-Populate	The **Pre-Populate** checkbox tells PingID to retrieve a value from an associated identity repository for that authentication type. To use the **Pre-Populate** setting, you must have an identity repository configured in PingOne or have the appropriate attributes configured within PingOne if you are using the internal PingOne directory. For more information, see Identity providers and Configuring the phone number attribute in PingOne. **Note** • For **SMS** and **Voice**, the value retrieved is a phone number, while for **Email** the value is an email address. • Pre-population of selected values into PingID occurs at the time the user goes through registration. • Pre-population of values does not occur if the user registers using the Windows login adapter, SSH adapter, or VPN adapter. • Pre-population of values does not mean that the corresponding devices are registered, only that values for the selected devices are pre-populated for user convenience. • The **Pre-Populate** checkbox cannot be changed unless the corresponding device type is enabled.
Restrict	The **Restrict** checkbox is enabled for any factor which has the **Pre-Populate** value selected. If **Restrict** is selected, the user cannot change the pre-populated value for that device. For example, if **Pre-Populate** and **Restrict** are selected for **SMS**, a phone number is pre-populated from the integrated identity repository, and the user cannot change that phone number.
Backup Authentication	The **Backup Authentication** checkbox specifies whether the selected device factor can be used in the event that a user is unable to use a registered device. The types of devices that an organization enables for alternate authentication methods should be determined by the amount of control an organization wants to have over their user's MFA devices as well as the impact of that device on the organization's security footprint: • If **Backup Authentication** is selected for a device factor, a value for that factor is retrieved from the integrated identity repository and is presented to the user as an option for the **Forgot Your Device?** button during an MFA challenge. • You can select the **Backup Authentication** checkbox even if the corresponding device type is not enabled as a primary factor. In this scenario, the device type is not available for registration, but can still be used to assist a user complete an MFA challenge if the user indicates that their registered MFA devices are unavailable. **Note** SMS, email, and voice factors are less secure than other alternate methods, such as the PingID mobile application.

Section	Description
Voice	The **Local Language for Voice Calls** setting allows voice calls, if enabled as a factor, to be performed in a language local to the end user when using web-based SSO. The local language is determined by the language specified in the user's browser: • If set to **Disable**, voice calls are always in English. • If set to **Enable**, voice calls are in the local language as determined by the web browser. If the browser's local language isn't supported, the call will be in English. **Note** Because Windows login, SSH, and VPN don't use a browser, voice calls are always in English for those authentication types. For a list of supported languages, see PingOne language support.

Section	Description
SMS/Voice	SMS and voice MFA challenges are performed utilizing Twilio. The **Twilio Account** section allows the organization to choose whether to use Ping Identity's Twilio account or to use the organization's own Twilio account: • Selecting **Ping Identity** configures PingID to use the Ping Identity Twilio account. **Note** Using the Ping Identity Twilio account incurs per-transaction charges. • Selecting **Custom** configures PingID to use your organization's Twilio account for SMS and voice MFA challenges: ◦ Selecting **Custom** displays text boxes for **Account SID** and **Auth Token**, which the organization must provide from its Twilio account. ◦ **Organization Numbers** are displayed when the Twilio account has been verified by clicking **Verify Account**. **Note** One or more of the **Organization Numbers** must be selected, and PingID uses one of the selected numbers as the originating number for SMS and voice MFA challenges. ◦ **Fallback to Default Account** determines whether PingID falls back to the Ping Identity Twilio account in the event of an error using the organization's unique account: ■ Selecting **Disable** configures PingID not to fall back to the Ping Identity Twilio account and will cause SMS and voice MFA challenges to fail in the event of an error with the organization's unique Twilio account. ■ Selecting **Enable** configures PingID to use the Ping Identity Twilio account if an error occurs using the organization's unique Twilio account. The **Daily Used SMS/Voice Limit** and **Daily Unused SMS/Voice Limit** sections specify how many SMS or voice calls a user can receive each day. This prevents abuse of the SMS and voice service. • The **Daily Used SMS/Voice Limit** field specifies the number of SMS or voice authentication requests a user can receive and respond to each day. The default value is `15` for the licensed version of PingID and `5` for the trial version. • The **Daily Unused SMS/Voice Limit** field specifies the number of SMS or voice authentication requests a user can receive and not respond to each day. The default value is `10` for the licensed version of PingID and `5` for the trial version. For more information, see SMS and voice usage limits.

Section	Description
Desktop	Ping Identity provides a desktop application for Windows and Mac which presents an OTP for use during MFA challenges. This application should not be confused with the PingID integrated Windows login adapter. The **Desktop** section is only visible if **Desktop** has been enabled in the **Alternate Authentication Methods** section. To provide an additional layer of protection for the desktop application, the **Desktop Security PIN** setting determines whether a PIN is required to unlock the desktop application. The PIN for the desktop application is uniquely set by each user: • If **Disable** is selected, no PIN code is required to unlock the application. • If **4-Digit** is selected, a four-digit PIN is required to unlock the desktop application. • If **6-Digit** is selected, a six-digit PIN is required to unlock the desktop application. • The **Use Proxy for Desktop** setting allows an organization to configure the desktop application to use an enterprise proxy for internal and internet communication. ◦ Selecting **Disable** configures PingID to support only non-proxied desktop applications. ◦ Selecting **Enable** configures PingID to support proxied desktop applications. **Note** Organizations should consider the security implications of using a desktop-based application for retrieving OTPs, as having the application on the same desktop on which MFA is initiated might reduce the security value of the MFA challenge.
Security Key	The **Security Key** section contains two options: • **Resident Key** determines whether a private key will be stored on the authenticator to enable passwordless authentication. • **User Verification** determines whether end users are preferred or required to authenticate using a security key that supports a user verification interaction. **Note** If **Resident Key** is set to **Required**, **User Verification** is automatically required.

Policy

Setting	Description
Enforce Policy	The **Enforce Policy** setting is a master on-off switch for PingID authentication policies: • If **Disable** is selected, no PingID policies are processed during MFA challenges. • If **Enable** is selected, PingID policies are processed during MFA challenges. For more information on creating policies or to view the PingID documentation on policies, see Authentication policy.

Setting	Description
Enforce Policy for Windows Login	The **Enforce Policy for Windows Login** setting tells PingID whether to process PingID policies specifically for the Windows login adapter: • If **Disable** is selected, PingID policies are not processed for Windows login MFA challenges. • If **Enable** is selected, PingID policies are processed for Windows login MFA challenges.

Evaluation

If you are running a trial of PingID, the **Evaluation** section is visible. After you purchase PingID, the **Evaluation** section is no longer displayed.

Setting	Description
Expiration Policy	The **Expiration Policy** setting determines how PingID behaves when an organization's PingID trial has expired: • The **Allow Single Sign-on Without PingID** setting configures PingID to automatically approve any MFA challenges without prompting the user. This value effectively sets PingID to pass through all requests without evaluating them. • The **Decline Single Sign-on Attempts Using PingID** configures PingID to automatically deny any MFA challenges. This value prevents users from being able to complete authentication processes where PingID is integrated.

Performing a near-zero downtime PingFederate upgrade

If you cannot incur a full PingFederate outage when you upgrade, you can perform a near-zero downtime upgrade.

What it is

Best practice when possible is to schedule downtime when upgrading PingFederate. During a scheduled downtime upgrade, PingFederate is unavailable for the entire duration of the upgrade. This can create a significant user impact in some scenarios, such as if your company is globally based, and your users need access to resources 24 hours a day.

Although there is no officially supported method for zero downtime PingFederate upgrades at this time, if you cannot incur a full outage, you can perform a near-zero downtime upgrade. A near-zero downtime upgrade is a method of upgrading that can reduce the PingFederate downtime to minutes, minimizing the impact to your users.

> **Important**
>
> This document outlines how to execute the upgrade without incurring service downtime or unexpected results caused by mixed version nodes communicating on the cluster communication ports. It does not cover all the upgrade considerations, pre-upgrade planning, pre-upgrade tasks, or post-upgrade tasks required to be successful. All documentation links refer to the latest version.

During a near-zero downtime upgrade of the runtime engines, you essentially split the cluster into two using the cluster auth password, then bring it back together into one cluster at the end.

The success of this process largely depends on isolating traffic from engine nodes and changing the cluster password so that the nodes on different versions are never joined to the cluster concurrently. A portion of your cluster must be capable to handle 100% of traffic while the rest is upgraded.

The Upgrade Utility does not affect your existing source deployment. It creates a new upgraded installation side by side.

> **Note**
>
> Some requests could fail during the upgrade if states and references generated on nodes won't replicate or might not be available to the full cluster. If that occurs, the user must authenticate again.

What you'll need

- Have persistent authentication sessions enabled. This can reduce disruption during the upgrade, as sessions will be stored in external storage and not only in node memory. Learn more in Sessions and Defining a datastore for persistent authentication sessions.
- Have all proper license files ready if you are going to a new major version.
- Arrange for access to your load balancer.
- Schedule your upgrade during off hours to minimize impact on your business.
- Verify that a portion of your cluster is capable of handling 100% of traffic, or the traffic expected during off hours, while you are upgrading the rest of your cluster.
- Review the upgrade documentation that's specific to the version to which you are upgrading as well as for any releases between your current production release and that version to plan your pre and post-upgrade tasks. Learn more about upgrade specifics in Upgrading PingFederate in the PingFederate documentation.

What you'll do

To perform your upgrade, you'll:

1. Upgrade the admin node.
2. Upgrade the first half of the engine nodes.
3. Upgrade the remaining engine nodes.
4. Restore the user traffic flow to all of the engine nodes.

Step 1: Upgrade the admin node

Because it doesn't serve any user traffic, you'll begin your upgrade on the admin node.

1. On the admin node, stop the PingFederate service.
2. Run the upgrade utility command.

Learn more in Upgrading PingFederate installations.

- If you're upgrading a Windows install, change the file service properties in the upgraded node's `<pf_install>/pingfederate/sbin/wrapper/PingFederateService.conf` to a unique name so that you can start either the new version or the old version if you need to roll back.
- If you're upgrading a Linux install, modify the `init` script to point to the new location.

3. Before you start up the service for the new version, change the `pf.cluster.auth.pwd` in the `run.properties` file so that it cannot communicate with the nodes that have not been upgraded yet.
4. Start the new admin node service.
5. Sign on to the PingFederate administrative console, upload the new license if needed, and validate that your configuration is correct.

Step 2: Upgrade the first half of the engine nodes

Next, you'll upgrade the first half of your engine nodes while the remaining servers continue to process requests.

1. Stop the traffic flow to 50% of your runtime servers.
2. Stop the PingFederate service on the first set of engine nodes that you are upgrading.
3. Run the upgrade utility command..
    - If you're upgrading a Windows install, change the file service properties in the upgraded node's `<pf_install>/pingfederate/sbin/wrapper/PingFederateService.conf` to a unique name so that you can start either the new version or the old version if you need to roll back.
    - If you're upgrading a Linux install, modify the `init` script to point to the new location.
4. Before you start up the service for the new version, change the `pf.cluster.auth.pwd` in the `run.properties` file to the new password that you used on the upgraded admin node.
5. Start the engine nodes service and validate that they are working as expected by pointing the hosts file on a client to resolve the PingFederate base URL to each upgraded node.

    The admin node should see these upgraded nodes joined to the new cluster.

Step 3: Upgrade the remaining engine nodes

After you upgrade the first half of your engine nodes, you'll upgrade the rest of the engine nodes while the nodes that you upgraded in step 2 process requests.

1. After validating that the load balancer works as expected, switch all user traffic flow to the upgraded nodes.
2. Stop traffic to the remaining nodes.
3. Stop the PingFederate service on the remaining engine nodes to be upgraded.

4. Run the upgrade utility command:
    - If you're upgrading a Windows install, change the file service properties in the upgraded node's `<pf_install>/pingfederate/sbin/wrapper/PingFederateService.conf` to a unique name so that you can start either the new version or the old version if you need to roll back.
    - If you're upgrading a Linux install, modify the init script to point to the new location.

    > **Important**
    >
    > Do not change the `pf.cluster.auth.pwd` in the `run.properties` file on these nodes until you validate that they work as expected.

5. Start the engine nodes service and validate that they are working as expected by pointing the hosts file on a client to resolve the PingFederate base URL to each upgraded node.
6. After you've successfully validated that the nodes are working correctly, change the `pf.cluster.auth.pwd` in the `run.properties` file to the new password that you used on the upgraded admin node, then restart the engine nodes service.

    The admin node should see all of the engine nodes joined to the new cluster.

    > **Note**
    >
    > This is when the downtime in near-zero downtime occurs.

Step 4: Restore user traffic flow

After you've upgraded all of your nodes and validated that they're working correctly, you'll restore the user traffic flow to all of them.

What's next

Learn more about your upgraded version in the PingFederate documentation.

Best Practice Guides

Use case	Description
Best Practices: Session Management	Session management is the process of managing user sessions in a web application. A session is a series of interactions between users and a web application that take place over a period of time.
Best Practices: Elevated Rights for PingDirectory	This document provides an overview of Ping Identity's recommendations for management of elevated rights in PingDirectory.
Best Practices: Performance Testing PingDirectory	PingDirectory ships with several tools that you can use for performance testing.
Best Practices: PingDirectory Operational Support	This document contains recommendations and best practices for the PingDirectory application onboarding process. Additionally, this document provides recommendations on supporting processes that could be used in conjunction with application integration.
Best Practices: PingFederate SAML Signing Certificates	The following reference guide details the best practices for managing PingFederate Security Assertion Markup Language (SAML) signing certificate settings, depending on your partners' preferences.
Best Practices: Performance Testing for PingFederate	This document provides an overview as well as general guidelines related to performance testing methodology for testing a PingFederate server prior to that system entering a customer production environment.
Best Practices: Journey to Passwordless	Learn about how passwordless authentication reduces friction for users.

Best Practices: Session Management

Session management is the process of managing user sessions in a web application. A session is a series of interactions between users and a web application that take place over a period of time.

When sessions are well-managed, users can securely interact with the application and exchange sensitive information without having to frequently re-authenticate. The type of session management that organizations use depends on the sensitivity of the information being exchanged:

- Short-lived sessions last as long as the user interacts with the application. Sessions end when the user signs out of the application or when the session lifetime limit is reached.
- Long-lived sessions keep users signed on to the application even if they leave. These sessions store session IDs on user devices, which allows users to reopen an application and use it without needing to re-authenticate, and are most often used on mobile applications.

While long-lived sessions often provide users with a better experience, it can become a security risk if someone else obtains access to the device and the session is still active.

The challenge is finding the right balance between keeping application sessions safe and providing users with the best possible experience. If a session timeout is too short, it can frustrate users because they'll be required to sign on again, but if it's too long, sensitive information can be exposed that hackers can acquire. Failure to find this balance can either result in users abandoning their sessions and not returning to the application, or sessions being attacked, both of which can result in losing customers and revenue.

The specific challenges you might face depend on the type of application you're protecting. For example, with workforce applications, because you understand who your users are and where they're located, configuring application sessions might seem to be a simple task. However, when employees travel and occasionally work from different locations, session configuration becomes more complicated.

With retail applications, users are not always authenticated until purchases and other transactions occur, so it's even more difficult to determine if a returning user is the same person. You can use long-lived cookies with unique values that identify specific visits and returns, but many users don't want to be tracked and remove the cookies. Additionally, because other users might reside in locations where cookies aren't allowed, relying on persistent cookies is not always possible.

Fortunately, there are a wide variety of ways to configure retail and workforce application sessions to ensure that authentication occurs at the appropriate time and place, using methods deemed appropriate for the risk level detected.

Key findings

You can prevent the most common types of session attacks by ensuring that session IDs and session cookies are protected:

- Session IDs are unique identifiers that the web applications create and assign to users for the duration of their visit. The session ID remains the same for a period of time, but a new one should be created for each stage of the session.
- Session cookies are files that contain the session ID. When users initially sign on to an application, a session ID and a session cookie containing that ID are created and sent to the user's browser to provide access. The browser then sends the cookie with every request to the server, which verifies the session ID and retrieves the requested object. Session cookies are temporarily stored on the user's device during a session and are typically destroyed when the session ends.

Session cookies are different from persistent cookies because persistent cookies exist after users close their browsers. Persistent cookies are used to recognize users and their devices, track their activity, display personalized ads, and create a better browsing experience by showing users other items that might interest them based on their browsing activities. The most common types of attacks, which are session hijacking attacks, man-in-the-middle attacks, and fixation attacks, occur when either the session ID or session cookies have been compromised.

Session hijacking

Session hijacking occurs when attackers eavesdrop on network traffic and steal or predict the target's session ID, which enables them to impersonate the user, gain access to their sensitive information, and commit fraud and theft.

In this diagram, the attacker uses sniffer tools to obtain valid session IDs.

+ image::rpf1695396852062.png[alt="A diagram showing the attacker intercepting the session ID as the user is interacting with the application.",role="border-no-padding"]

Then, attackers use these session IDs to access the application by impersonating the user.

+ image::zkp1695396742226.png[alt="This diagram shows the attacker using the session ID to access the application.",role="border-no-padding"]

It is especially easy for attackers to eavesdrop on open, unencrypted wireless networks, such as the free WiFi offered at coffee shops and other businesses. Laptops or mobile devices broadcast a request to the WiFi device in the room that receives the signal, but these broadcasts are also visible to any other device in the room, including eavesdropping attackers.

Man-in-the-middle attacks

Man-in-the-middle attacks occur when attackers impersonate either the user or the application and make it appear as though normal communication is in progress. Their goal is to steal sensitive information, such as sign-on credentials, credit card numbers, and financial account details.

First, they find a way to impersonate the original connection, then they communicate with the user, and finally, they access user accounts. These types of attacks can be compared to your mailman opening your bank statement, obtaining your account information, resealing the envelope, and delivering it to your door.

+ image::rau1695396896143.png[alt="This diagram shows the attacker impersonating the original connection.",role="border-no-padding"]

Fixation attacks

Fixation attacks occur when attackers steal valid session IDs that have not yet been authenticated. Attackers send users a link that contains the session ID and tricks them into clicking on it. When they authenticate with what they think is the application, the attacker uses the same session ID to access user accounts.

+ image::iek1695396954692.png[alt="This diagram shows the attacker sending the user a link. The user clicks on the link to access what they think is the application. The attacker uses that session to access the application.",role="border-no-padding"]

Recommendations

Because the ways in which application sessions are handled can affect the user experience, it’s important to get it right. You want to keep session data safe from attackers, but you also want to ensure that users enjoy interacting with your applications and aren’t unnecessarily interrupted during their journeys with authentication requests. The more your users enjoy their experiences, the more likely they are to become loyal customers, have stronger emotional connections to your brand, and refer other customers, which will ultimately increase your revenue.

For example, most people understand why financial institutions require multi-factor authentication (MFA) and have shorter timeout sessions and expirations, and they appreciate the fact that protection efforts are in place. However, many other types of organizations are not concerned with protecting sensitive information until purchases and other transactions are made or relationships are established. They are focused on engaging their users and making applications easy to access and use, which usually means that their sessions are much longer-lived and more complicated to protect.

While no solution is perfect and some level of risk will always exist, we’ve created a list of best practice planning and implementation recommendations, and provided some session configuration suggestions, to help you keep your sessions as secure as possible.

Planning and implementation recommendations

Many tend to overlook the importance of the planning process when large amounts of time, effort, and energy can be saved if session management is appropriately planned and implemented. We recommend the following:

- Include the appropriate stakeholders in your decision-making processes. Often, identity and access management (IAM) administrators and engineers begin defining session parameters without consulting others who will be affected by configuration decisions, such as product managers, marketing and support representatives, and application owners.

  It's best to work closely with these stakeholders from the beginning to ensure that you all understand the current and future state of user journeys and that you're all involved in the decisions made regarding these journeys. Consider using an orchestration tool to help you visualize each path users can take.

- Review process flows to ensure that authentication occurs at the appropriate time and place. Consider the sensitivity of the data involved in each interaction and determine what type of security is needed.

  There are many ways you can configure application sessions to best meet your needs. For example, you can create rules that drive authentication and determine what users can see and do in an application, or you can map re-authentication methods to actions taken based on risk tolerance and levels of identity assurance.

- Continually monitor your sessions to determine whether they can still be trusted. Track as many interactions as you can and obtain data feeds from as many systems as possible. This data can help you identify patterns in normal behavior so that you can more easily detect abnormal activity. You can use API detection solutions to identify suspicious activity, monitor user behavior, and compare it to known patterns. You might also consider using telemetry monitoring to help you detect irregularities.

- Use risk-based authentication, which requires users to provide additional authentication information if risk signals are detected. The risk level is determined by several unique factors including location, time of day, device and browser information, IP address, user information, and the context of the request.

- Determine what will happen when abnormal activity is detected. Will you end the session? Require users to re-authenticate? Require users to authenticate using a different method? Map out these remediation processes during the planning process and test them before implementation. Orchestration tools are also helpful in mapping out these processes.

- Use passkeys, if appropriate. Passkeys are a safer and easier alternative to passwords. With passkeys, users can sign on to apps and websites with a biometric sensor (such as a fingerprint or facial recognition), PIN, or pattern, freeing them from having to remember and manage passwords.

After you and your team implement your session configurations, the real work begins. Continue to review the data you collect and revisit your authentication processes on a regular basis. The threat landscape is constantly evolving, and it's important that you evolve with it. Technologies change quickly and new attack techniques will continue to emerge, so the session configurations that you establish today might work well, but might not work quite as well tomorrow.

Session configuration recommendations

Hijacking attacks, man-in-the-middle attacks, and fixation attacks most often occur when either the session ID or session cookies have been compromised. To protect them as much as possible:

Use HTTPS instead of HTTP

When HTTPS is used, all communication between the user's browser and the application is encrypted, which means that even if an attacker manages to intercept the traffic, they cannot read or tamper with the data. HTTPS also provides authentication, which ensures that users are communicating with the real application and not a fake application set up by an attacker. To protect cookies, we also recommend enabling the following attributes:

- The `Secure` attribute so that they will only be sent over HTTPS.
- The `HttpOnly` attribute so that JavaScript cannot access the cookies, which prevents attackers from stealing them using cross-site scripting (XSS).

Create secure session IDs and cookies

To ensure that attackers can't predict session IDs, it's considered an industry best practice to randomly generate session IDs that are a unique combination of letters and numbers at least 128 bits (16 bytes) long. This ID is simply an identifier and shouldn't contain sensitive information. When the session ID is created, a session cookie is also created to store the ID. It's a good idea to store session IDs in a different cookie from other sensitive information, such as a username. That way, even if an attacker manages to obtain the cookie, they won't be able to hijack a session without knowing the username. It's also a good idea to encrypt session IDs so that attackers can't read them without having the encryption key.

Limit the number of simultaneous sessions per user

If an attacker were to gain access to a user's account, they could do whatever they want. However, if you limit the number of simultaneous sessions available to each user, the application will only be available from one device at a time. This doesn't mean that users can only access the application from one device. It means that users can only be signed on to one device at a time.

Regenerate session IDs

To protect session IDs, we recommend regenerating them after users sign on to the application because it makes it much more difficult for hackers to exploit session IDs. It's also important to regenerate session IDs when users' privileges change. If a hacker hijacks a user's session, they will have access to the user's account with all of the privileges that the user has. However, if the session ID is regenerated after a significant privilege change, the hacker will no longer have a valid session ID, which makes it more difficult for them to access the user's account and escalate their privileges within the application. There are two ways to regenerate session IDs:

- The first is to invalidate the old session ID and create a new one. This approach is the most secure, but it can cause problems if the user has multiple tabs open because they will be signed out of all of them.
- The second is to change the secret key associated with the session ID. This approach is less secure but provides a better user experience because it doesn't sign the user out of other open tabs.

Destroying the session ID when users sign off is also highly recommended, as you invalidate the session ID and make it much more difficult for an attacker to hijack the session.

Invalidate all open sessions when passwords change

If an attacker obtains a user's password, they could use it to access the user's account. However, if all open sessions are invalidated when the password change occurs, the attacker will be locked out of the session.

Expire sessions based on user inactivity and risk tolerance

The longer sessions last, the more vulnerable they can become. The amount of time sessions should last depends on the sensitivity of the information exchanged between the user and the application and the level of identity assurance. For example, an e-commerce retail company might not have any concern about their users being signed on for long periods of time, especially if the user is just browsing and hasn't authenticated with the application. However, most financial institutions don't allow their users to be inactive for longer than 15 or 20 minutes before the session expires because the risk of fraud and theft increases.

Related links

To learn more about session management and authentication, see:

- PingAccess server-side session management configuration
- PingFederate session configuration
- PingOne Platform API Reference

To learn more about monitoring solutions, see:

- PingOne Protect
- PingOne Protect API Reference
- PingOne Authorize

Best Practices: Planning your upgrade

Upgrading your software is essential to maintaining a secure environment that responds to your business needs. This planning guide is intended to supplement your internal upgrade protocols.

If you have questions about your upgrade, post them to our Support Community for expert answers from other Ping users. Tag your questions with the "Upgrade" topic. If you're new to the community, check out Getting Started with Ping Community.

If you have an urgent upgrade request, please open a support ticket in the Support portal.

If this is your first time upgrading Ping products, or if your environment has grown in complexity, we recommend connecting with your account team to ensure you have the right resources in place. Ping Identity and our partners are available to consult on upgrades or even perform them for you as part of a Professional Services engagement.

> **Note**
>
> Ping Identity products are designed using open standards and can be upgraded in any order. However, we recommend that you upgrade one product at a time. Use the following templates to create a separate upgrade plan for each Ping product you use.

Upgrade planning guide

Use the following checklist to assess the scope of the upgrade process before beginning.

Before you begin

Note the product name, current version, and upgrade version for each product that you're upgrading.

Product name	Current version	Upgrade version

Steps

1. Validate upgrade scope requirements:

Pre-upgrade task	Assignees	Completion Date
Identify the number of deployed environments.		
Identify the number of nodes (admin plus runtime) per deployed environment.		
Read the upgrade guide and release notes for your target upgrade version. **Note** If you skip versions (for example, if you're upgrading from version 2.0 to version 5.0) we recommend reviewing the upgrade guides, release notes, and system requirements for versions 3.0 and 4.0, as well as version 5.0, to ensure that your upgrades go smoothly.		
Review the differences between current and target versions. Note any incompatibilities or dependencies.		

Pre-upgrade task	Assignees	Completion Date
Read the tuning guides for your target upgrade version to ensure that you're aware of significant changes and new tuning recommendations, especially when Java versions have been upgraded.		
Review the new JDK defaults and make and relevant changes.		
Collect and review the deployment requirements, use cases, and architecture and design documents.		
Accept the changes and features from the release notes.		
Obtain a new license key for upgraded version.		

2. Identify external services that might be affected by the upgrade:

External Service Type	External Service Name
Directories	
Databases	
Application integrations	

External Service Type	External Service Name

3. Assess upgrade readiness.

   Do you have the team and tools to complete this upgrade successfully? If not, please reach out to your Ping Account Team to discuss resources.

Upgrade process

This upgrade process is recommended for all Ping products.

Steps

1. Identify the key contacts:

Ping primary contacts	Contact phone or email
Account executive	
Ping Technical Support	◦ North America: 1-855-355-PING (7464) ◦ EMEA: 44 0 808 196 0788 ◦ APJ: 61 1800 370 672

Ping primary contacts	Contact phone or email
Other	

Internal primary contacts	
Project manager	
Technical team	
Technical team	
Additional stakeholders	

2. Review the release notes for the upgrade version:

Pre-upgrade task	Completion date	Assignee
Review release notes for upgrade. **Note** If you skip versions (for example, if you're upgrading from version 2.0 to version 5.0) we recommend reviewing the upgrade guides, release notes, and system requirements for versions 3.0 and 4.0, as well as version 5.0, to ensure that your upgrade goes smoothly.		

Pre-upgrade task	Completion date	Assignee
Perform any pre-upgrade tasks.		
Define your rollback plan.		
Review the post-upgrade requirements and note any applicable tasks.		

3. Perform a trial upgrade:

Trial upgrade task	Completion date	Assignee
Schedule a trial upgrade on a duplicate image or staging environment.. **Note** If your staging upgrade environment differs significantly from your production upgrade environment, making a copy of the production environment (for PingFederate and PingAccess, at a minimum, the admin server and one runtime server) can help you find any anomalies that you need to address prior to a full production upgrade.		
Review the upgrade log. Note the anomalies, differences, and results.		

Trial upgrade task	Completion date	Assignee
Review your custom templates, compare them to the original versions, and determine if these customizations need to be carried over to the templates in the new version. This review is especially important if adapter templates were added or updated.		
Migrate your custom `.jar` files.		
Migrate your HTML templates.		
Migrate your velocity templates.		
Verify upgrade reliability according to your company standards.		
Create the timeline and tasks for your production upgrade.		

4. Perform your production platform upgrade:

Production upgrade task	Completion date	Assignee
Create the upgrade task list for your production environment.		

Production upgrade task	Completion date	Assignee
Schedule your upgrade to minimize downtime.		
Notify Ping support of scheduled upgrade. Create a case in the Support Portal with the type = "Upgrade" and include important details such as the date and time of your move, which products you are upgrading, the old and new version, and helpful architectural details.		

General upgrade best practices

These tips will help you avoid common pitfalls, no matter which Ping product you are upgrading.

1. If you are migrating to a new server platform/operating system, copy the engines and consoles from the original servers to the new ones.
2. If your new environment needs unlimited Java Cryptography Extension (JCE):
    - Install the new Java and unlimited JCE on the new servers.
    - Upgrade the old console installations on the new servers.
    - Upgrade the old engines on the new servers.
    - Update the `run.properties` file to reflect new IP addresses on the new servers.
3. Configure load balancers for the new clusters:
    - Validate that the new configuration is working by updating a few host files on desktops using the new load balancer address.

      > **Note**
      >
      > Reference adapters and OAuth aren't very testable in any scenario other than on the cutover platform.

    - Switch DNS to the new clusters after you've successfully unit tested the configurations.

Product-specific upgrade guides

These links go to the upgrade pages and release notes for each Ping Identity product. These pages are updated with each product release.

PingFederate

- PingFederate release notes
- PingFederate upgrade guide
- PingFederate system requirements
- PingFederate performance tuning guide

PingAccess

- PingAccess release notes
- PingAccess upgrade guide
- PingAccess zero-downtime upgrade guide
- PingAccess system requirements
- PingAccess performance tuning guide

PingDirectory suite of products

- PingDirectory suite of products release notes
- PingDirectory and PingDirectoryProxy server upgrade guide
- Delegated Admin upgrade guide
- PingDataSync upgrade guide
- PingDirectory system requirements
- PingDirectory performance tuning guide

PingAuthorize

- PingAuthorize release notes
- PingAuthorize upgrade guide
- PingAuthorize system requirements

PingCentral

- PingCentral release notes
- PingCentral upgrade guide
- PingCentral system requirements

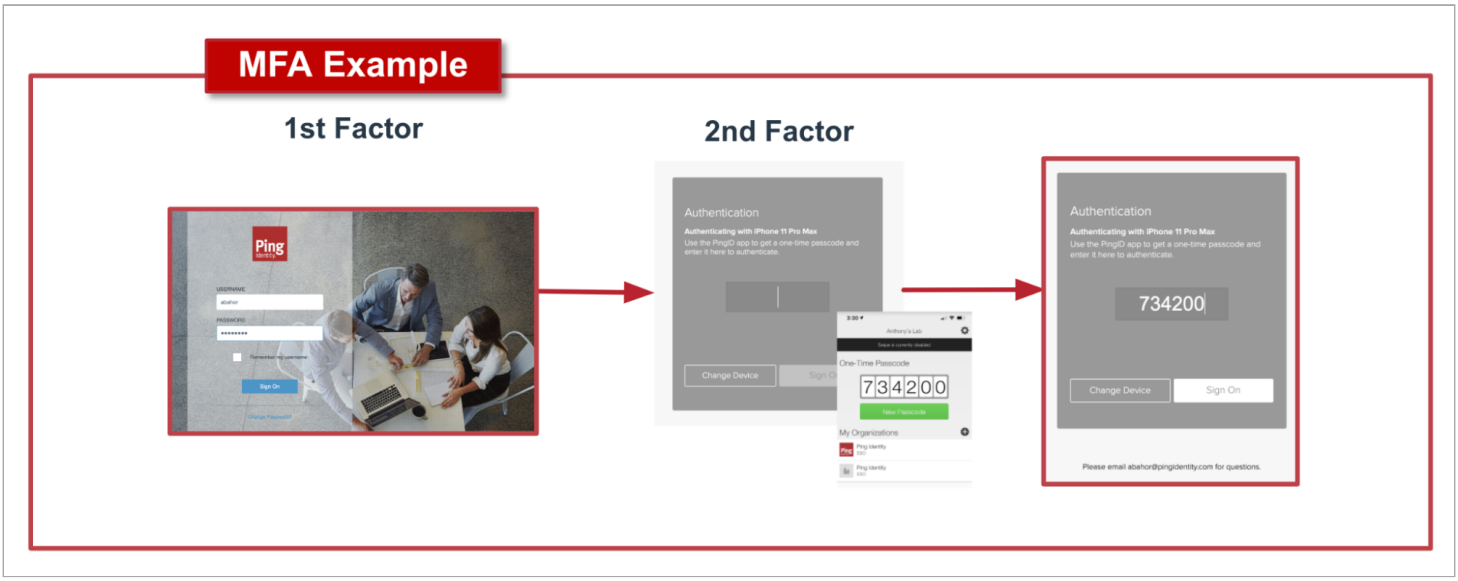
Best Practices: Journey to Passwordless

Learn about how passwordless authentication reduces friction for users.

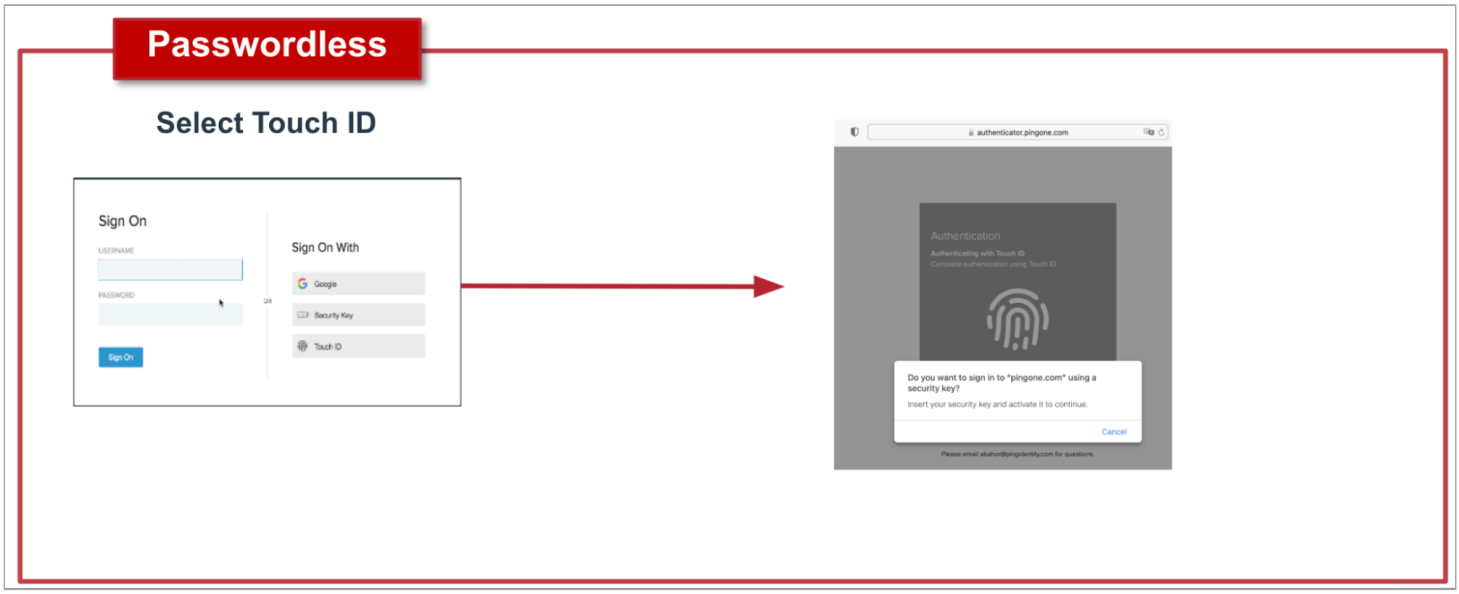
Your enterprise is more than likely taking advantage of using multi-factor authentication (MFA). This enables step-up authentication by providing a second factor with authentication. The second factors have multiple methods an administrator can configure, which include but are not limited to:

- Authenticator applications
- Email
- SMS
- Voice
- One-time passcodes (OTPs)
- Hard tokens
- FIDO

When using PingID, you might have a similar experience using first and second factors together as shown in the following image.



The goal is to reduce passwords and to evolve the experience into a frictionless experience, as seen in the following image showing the passwordless experience using Touch ID.



The number of steps for a passwordless experience decreases compared to the MFA experience:

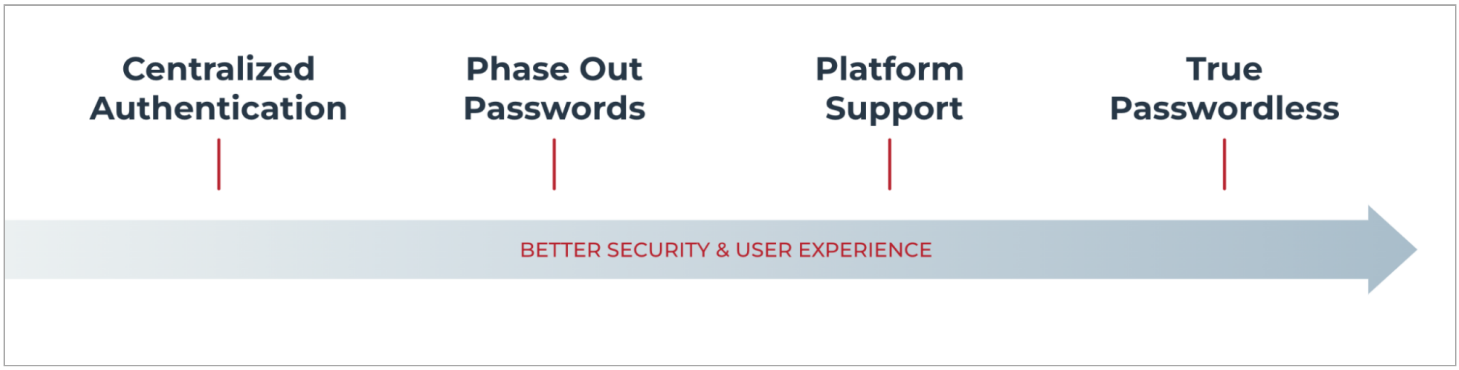
- Reduce footprint:
  - Single sign-on (SSO) and MFA
  - Authentication authority
  - Standards
  - Risk signals
- Reduce friction:
  - First factor FIDO
  - Continuous AuthN
  - Zero login

You can balance security and experience by managing risk.

- Passwordless has many different definitions, depending on who you ask.
- Passwordless boils down to either reducing passwords or eliminating them altogether.
- People agree that, when done right, passwordless offers a better experience and better security compared to traditional sign-on experiences.

Workforce passwordless journey

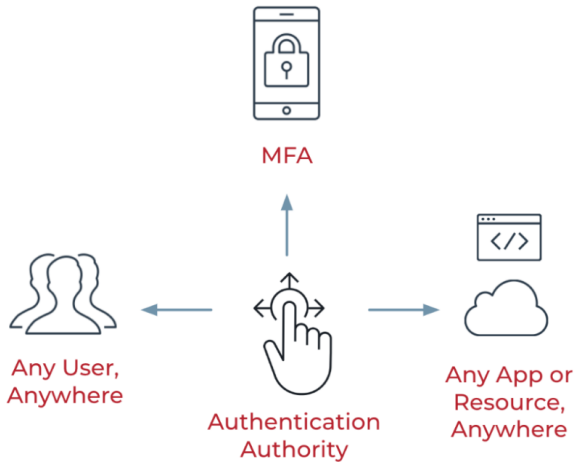
The journey to passwordless is comprised of many phases and goals.



The following diagrams show each step of the passwordless journey and its goals.

Step #1 Goals:

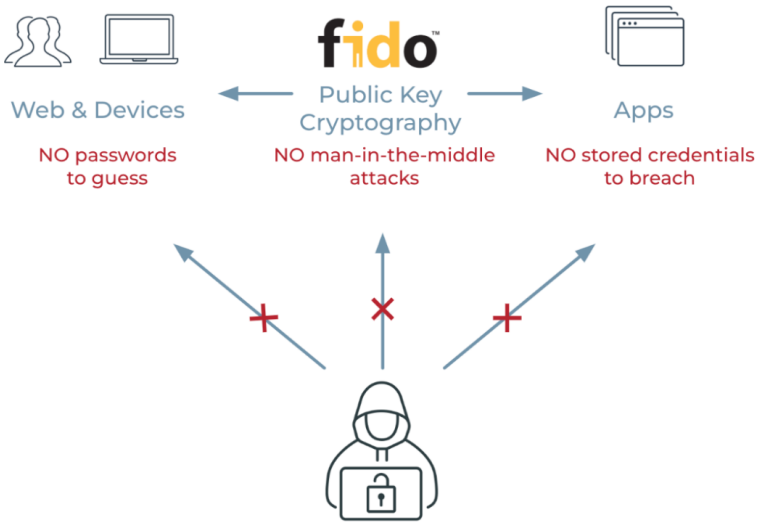
- Centralize SSO & MFA on an Authentication Authority Foundation
- Add Risk-Based MFA



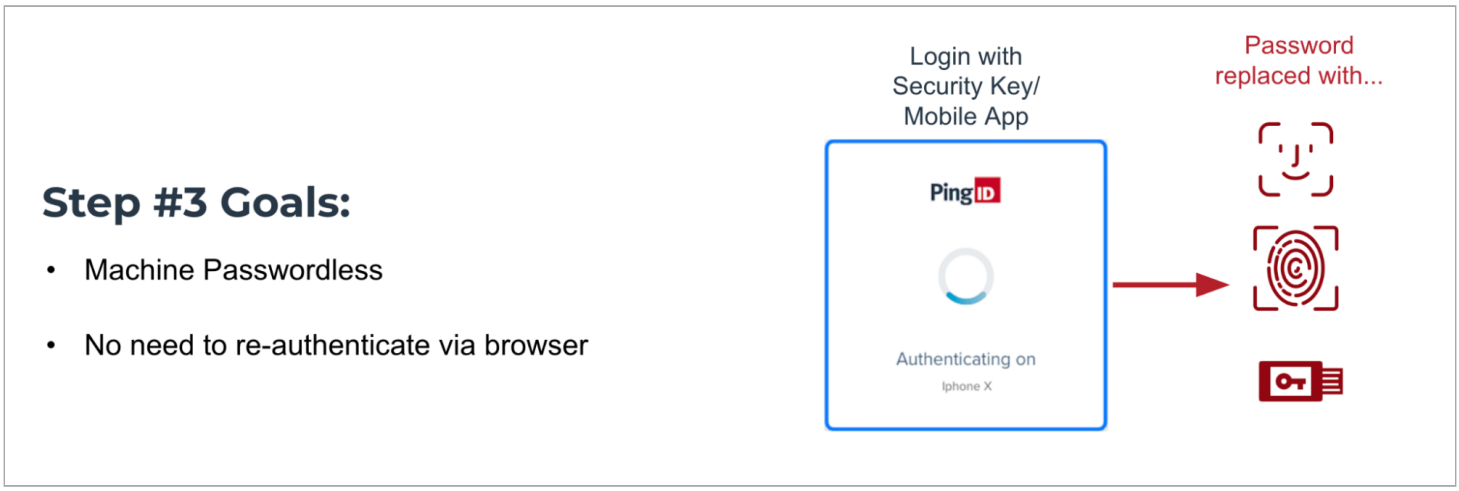
The goals of step 1 are to centralize SSO and MFA on an authentication authority foundation.

Step #2 Goals:

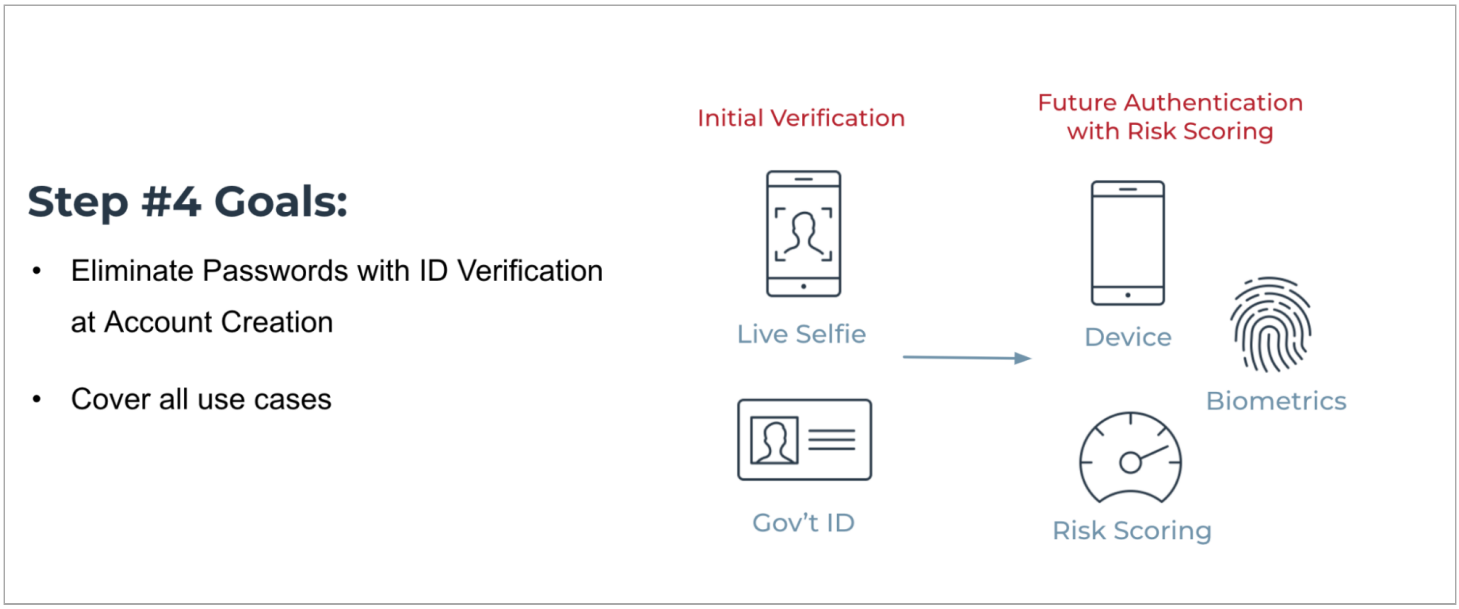
- Reduce use of passwords and rely on stronger, more convenient method
- Rely on risk and adaptive authentication, enforce the right method at the right time.



The goals of step 2 are to reduce password use and rely on adaptive authentication to enforce the right methods as needed.



The goals of step 3 are to implement machine passwordless control and remove the need for re-authentication with a browser.



The goals of step 4 are to eliminate passwords with ID verification at account creation and to cover all use cases.

Planning the workforce passwordless journey

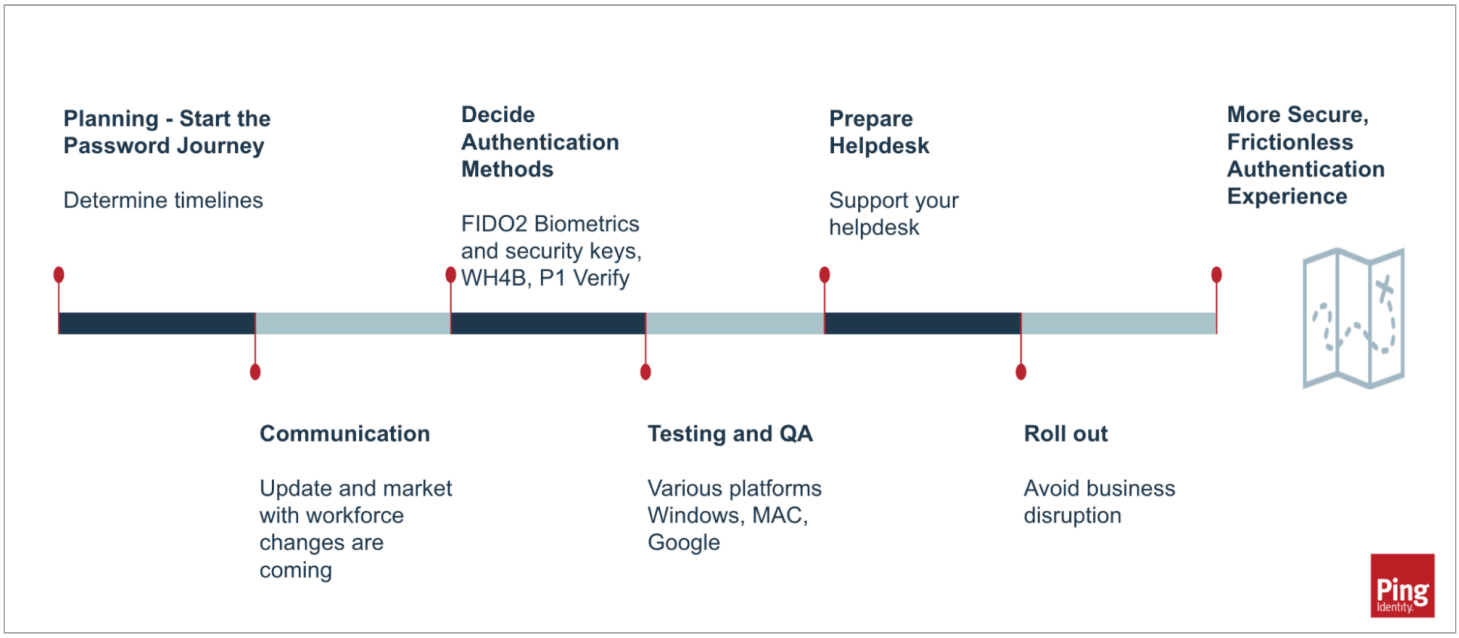
Passwordless authentication provides technologies, such as FIDO security keys, FIDO biometrics, and Ping Risk, that eliminate the use of passwords.

Because lost and stolen passwords create security risks, passwordless authentication helps mitigate security risks associated with passwords and removes friction from the authentication process.

The following planning diagram breaks the implementation process apart into separate phases:

1. Planning the journey
2. Communicating the upcoming changes
3. Deciding on authentication methods

4. Testing the implementation
5. Preparing your help desk for passwordless
6. Rolling out the passwordless experience



Change management

The most important step to begin discussing and planning with your organization. This helps to define the change that is coming, prepare for communication, and begin working with the internal channels that the enterprise is shifting to passwordless.

As an administrator, understand you are eliminating friction in the authentication process and improving security posture by eliminating passwords. Achieving frictionless authentication requires introducing friction at the start of the process, such as when you're planning change management, training, and implementing a deployment. The end result is a frictionless authentication experience.

When working with many internal teams through your change management processes there will be friction introduced in the beginning until this is complete. This is the part where you introduce more friction before going frictionless.

1. Understand that you are aiming to mitigate password risks currently present and introducing a frictionless experience, improved end user experience.
2. Take advantage of biometrics and FIDO2 compliant devices.
3. Deploy adaptive authentication with passwordless using intelligent behavior analysis of user activity to determine authentication requirements:
    - Low-risk authentication requests
    - High-risk authentication requests that require re-authentication or block

When dealing with change management, understand what devices and technologies are available for the organization. Work with the departments that handle inventory and the preferred choice of software used on workstations, and list what is available today for testing. Ensure internal teams are aligned as you prepare for implementation.

The different changes you have planned not only determine your timelines and goals, but affect the tools and strategies that you will need going forward. Team alignment helps leaders identify whether they must provide any additional resources before rolling out passwordless authentication.

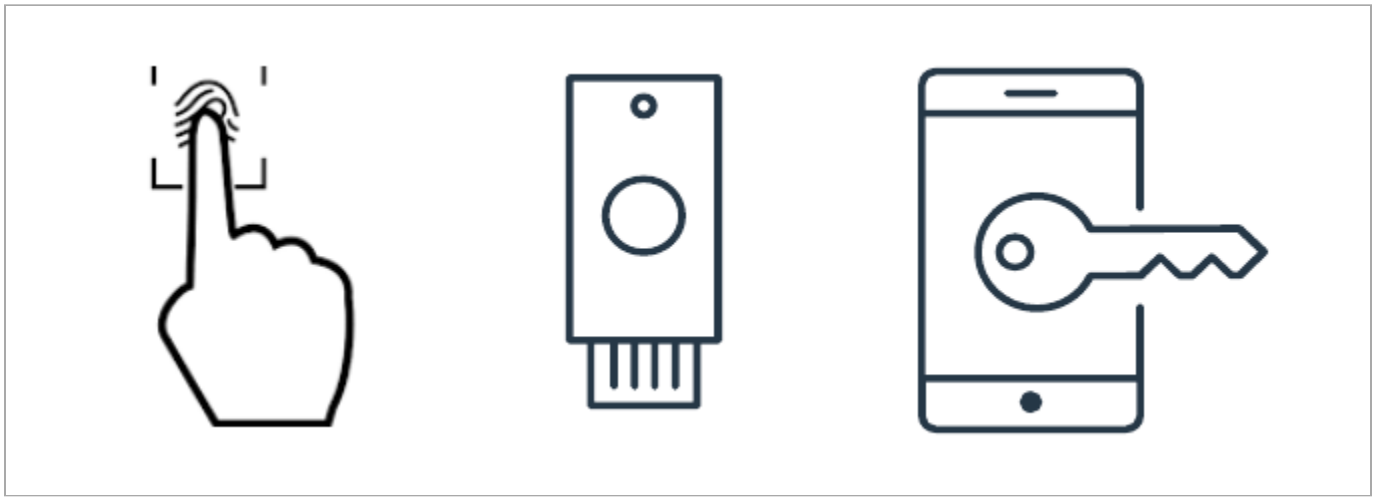
What is being used with MFA today? • Are registered devices FIDO compliant? • If not, begin introducing FIDO compliant devices allowed to be registered.	What is your current inventory? • Google, Apple, Microsoft, and so on • Safari, Chrome, Edge • Mac and Windows • Windows Hello
Default timelines • Make sure they're realistic • Determine testing period • Determine rollout period • Determine backup methods	Test with user groups • Seek opinions and feedback • Hold regular product management meetings

Communication

When planning this step, understand and determine who are the internal stakeholders including who will own what from the help desk, other departments, vendors, and application teams:

- Communicate timelines.
- Design marketing material and knowledge management articles.
- Establish the guidelines for these changes.
- Documentation and KMs.
- Prepare and train the help desk.
- Create a communication plan to update Workforce changes that are coming in the future.

Decide passwordless methods



As you decide on passwordless methods, keep the following in mind:

- Does your organization use Windows, Apple, Linux, or a combination of those?
- Are you using MFA?
- Ensure backup authentication methods are in place, such as:
  - TouchID, FaceID, FIDO2 compliant Security Key
  - Phone, tablet, laptop, and similar hardware.

> **Note**
>
> Always register more than one device or technology. Biometric authentication is slightly more secure than a PIN, but if biometric authentication fails, it can be set up to fail back to PIN.

- What adaptive authentication capabilities does your organization use, such as PingOne Protect?

Rethink policies and processes

When you rethink your authentication policies, consolidate them to simplify.

When you rethink procedures, consider:

- Help desk policies
- Lost device policies
- How to test for your use cases

Testing and QA

Before you plan for rollout, make sure you perform sufficient testing on different browsers and operating systems to prevent delays in implementation.

To test:

- Take into account third-party implementers for the standard.

  Chrome, Safari, Edge, and Firefox might have different requirements and behaviors that you must take into account.

- Involve diverse departments to help choose authentication factors and to test.
- Register a device on different operating systems.

  Some devices require updating the operating system. Updating the operating system for security requirements could create a delay with your planned timeline.

- Continue to test whenever browser and operating system updates occur.

Planning rollout

As you plan for rollout, consider the following priorities:

- Prevent disrupting business.
- What groups or apps are already using MFA?

  Start here.

- Rollout to end users.

  What departments and groups are best-equipped for the first wave?

- Deploy in a small volume, then ramp it up:
  - 10 users with 5 or 10 applications
  - Increase to 100 users for 10 apps and so on
  - Increase to 500 users with 10 apps and so on

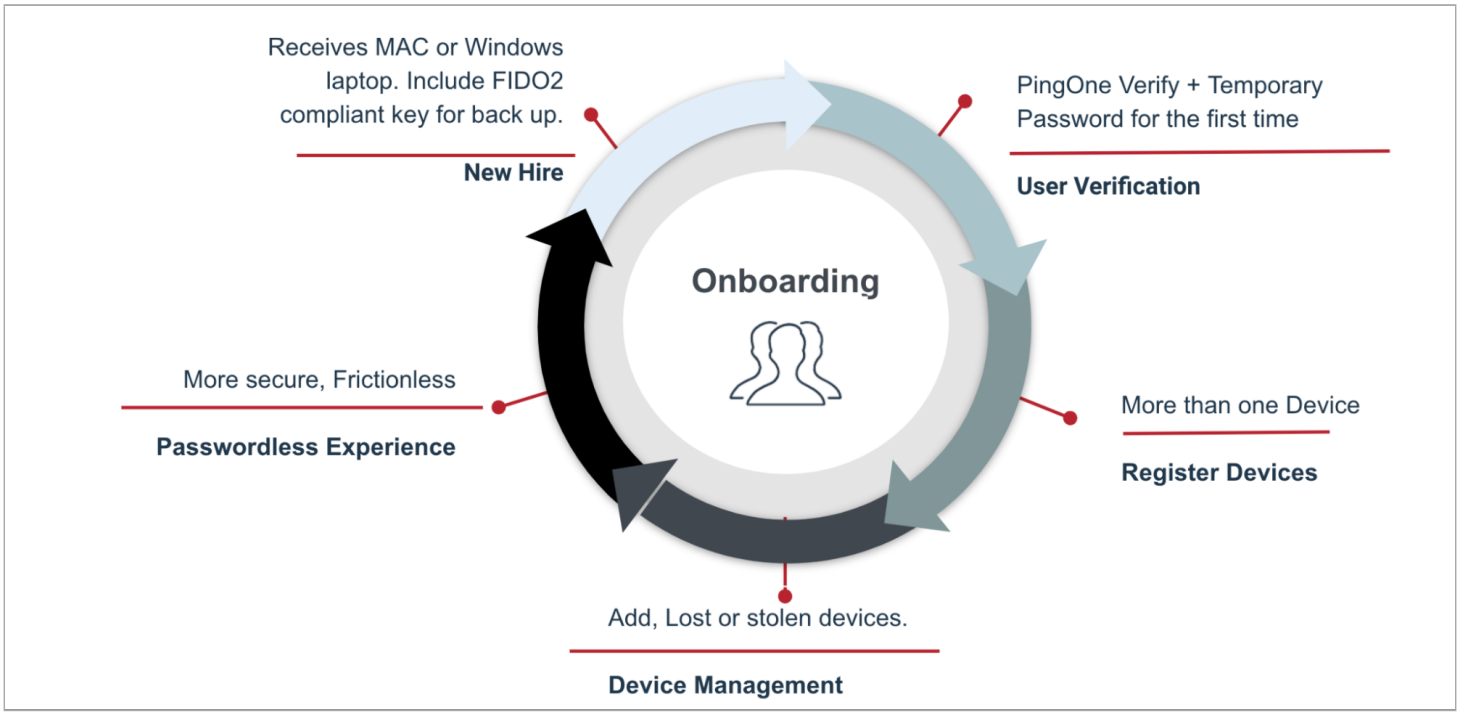
Onboarding

As you prepare to onboard individuals with your passwordless authentication experience, consider the following:

- Self-service is the goal.
- Who is your user population?
  - What groups are best-equipped to be first adopters?
  - What considerations do you need to take into account for global users?
    - Not all can use MFA today.
    - Might not own smartphones.
  - What considerations do you need to take into account for seasonal employees?
    - You might not want to give them a security key.
    - You might need a policy for these groups.

- Are you using MFA?
  - What are those devices?
  - Is more than one registered?
  - How many of those would fall into biometric, security keys, Windows Hello, or others?

The following diagram shows the onboarding cycle.



1. A new hire receives their Mac or Windows laptop. Include a FIDO2 compliant key for back up.
2. The user undergoes verification for the first time through PingOne Verify and a temporary password.
3. The user registers their devices, adding more than one device.
4. The user manages their devices, including adding device and reporting lost or stolen devices.
5. The user moves into a more secure, frictionless experience through passwordless authentication.

Lost devices

How are you handling lost or stolen devices? The following can all help you plan for what to do about these devices.

> **Note**
>
> Remember to have more than one device registered. For example, have an iPhone, iPad, and a laptop registered.

- User verification.
- PingID mobile application.
- What's easier to get back on network?

FIDO keys should be the second-to-last resort.

- HelpDesk can come into play through using a temporary method.

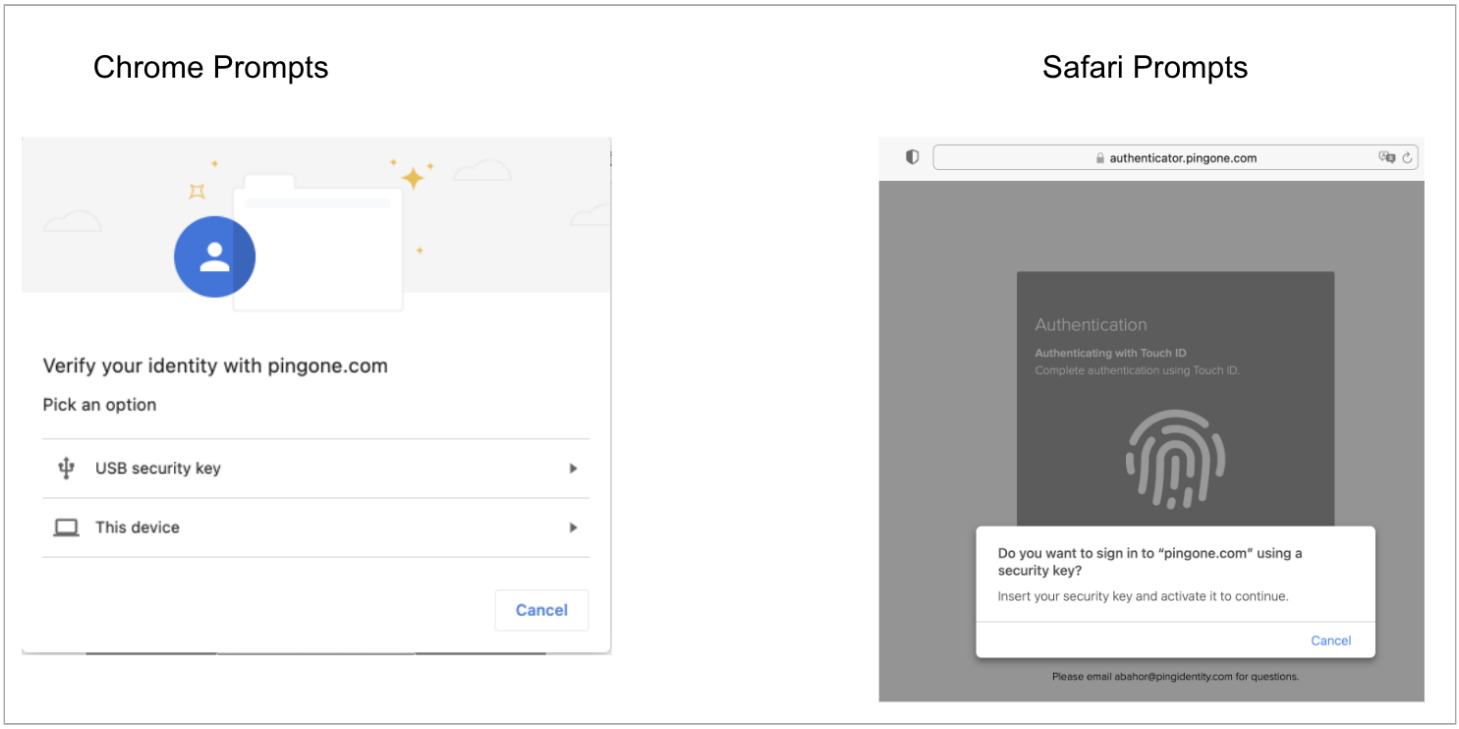
  The PingID desktop application with a lost or stolen Yubikey or phone should be your last resort.

Lessons learned from the workforce passwordless journey

- Passwordless takes time:
  - Having an MFA deployment prior to adoption into passwordless will ease the process.
  - Setting up and testing takes time. There is a high upfront cost with time, planning, and testing. You need to understand the different behavior when interacting with different software and then test and determine what's best for your users before rolling out passwordless. Stay with the guidelines you've set to simplify the onboarding and support process moving forward.
- You are now in the hands of the implementers for the standard:
  - Chrome, Safari, Edge, and Firefox might have different behaviors that you must take into account.
  - Client browser or OS updates could change the experience.
  - Using different browsers requires multiple registrations.
- User experience:
  - Format for selecting registered devices
- User verification and adaptive authentication:
  - Risk-based authentication

Web browsers

Different clients will provide a slightly different passwordless experience, as shown in the following example of the differences between a prompt in Chrome and a prompt in Safari.



> **Note**
>
> FIDO must be registered twice if you are using two clients. Your organization should support at least two clients in case one encounters an issue from an update or similar.

Deploying products

Before you begin

You must have:

- FIDO2 capable Security Key

  The example below leverages Yubikey 5.

- PingID Adapter 2.8.
- A browser that supports WebAuthn.
- PingFederate 10.2 or later, which provides native support for adding a passwordless authentication flow icon for the HTML Form Adapter along with only showing the **Security Key** button on the HTML Form when the browser in use supports WebAuthn.

About this task

This document makes the following assumptions:

- The organization has a functioning HTML Form IdP Adapter that passwordless authentication processes can be added to.
- A new PingFederate authentication policy is being built as opposed to adding to an existing authentication policy, which should also be possible.

- The authentication policy built within the example is at a global level as opposed to an application-specific one. An admin determines the authentication flow that will be best suited for passwordless experience.

> **Note**
>
> As of this writing, Firefox for Mac does not support PIN code user verification, resulting in the registered security key that has an associated PIN not being recognized.

PingID passwordless use cases

Windows login - passwordless makes it possible for users to sign on to their Windows computer without a password, using only one of the PingID authentication methods, such as the PingID mobile app.

Windows passwordless login

Learn more in Integrating PingID with Windows login (passwordless) in the PingID documentation.

> **Note**
>
> In the initial version of Windows login - passwordless, the only supported authentication method is the PingID mobile app 1.15 or later.

Consider the following into account before setting up Windows login - passwordless:

- For users to use the passwordless login, they must already have a device that has been paired with PingID.
- Windows login - passwordless includes support for Run as Admin.
- Windows login - passwordless includes support for remote desktop (RDP). If you plan on using RDP, you must install Windows login - passwordless on both the accessing client and the remote computer.

Overview of Windows passwordless login

These are the main steps the administrator must do to set up the PingID integration with passwordless Windows login:

1. Create a new environment in PingOne and connect it to your existing PingID account.
2. Configure identity store provisioners.
3. Create an issuance certificate in PingOne.
4. Create an authentication policy in PingOne.
5. Create and configure a passwordless Windows login application in PingOne.
6. Generate a Key Distribution Center (KDC) certificate if necessary.
7. Install the Windows login - passwordless integration software on the individual Windows client computers.

Setting up Windows passwordless login

You can use Windows login - passwordless so that users can sign on to their Windows computer without a password.

Before you begin

To set up and use the PingID integration for passwordless Windows login, the following system requirements must be met:

- Microsoft Active Directory is running on Windows Server 2016 or later
- Users' computers must be running Windows 10 (64-bit), and must support TPM 2.0.

You must have:

- Admin rights for the domain controller
- A PingOne account
- A PingID account

Users must have the PingID mobile app installed on their devices and must have already paired the device.

Creating a PingOne environment and connecting it to a PingID account

About this task

Create a new environment in PingOne and connect it to an existing PingID account (to allow syncing of the PingID data) or to a newly-created PingID account.

> **Note**
>
> You must create a new PingOne environment even if you have an existing environment because you cannot connect a PingID account to an existing PingOne environment.

Steps

1. In the PingOne admin console, click **Add Environment**.
2. Select **Build your own solution**.
3. Hover over the **PingOne SSO** element and click **Select**.
4. Hover over the **PingID** element and click **Select**.
5. Click **Next**.
6. When you are presented with the two options for PingID, you can either:

   Choose from:

   - Connect to an existing PingID account.

     After you select this option, enter the credentials that you use for the PingID account.

   - Create a new PingID account.
7. Click **Next**.
8. Enter a name for the new environment.
9. Select the relevant license.

10. Click **Finish**.

Configuring identity store provisioners

About this task

To use passwordless Windows login, user attributes must be mapped to attributes in PingOne.

If you have been using PingFederate with the PingID connector for user provisioning, you must make the transition to using PingFederate with the PingOne Provisioning connector for user provisioning.

You can find more information on using this integration in Provisioning connector in the PingOne Integration Kit documentation.

> **Note**
>
> When mapping attributes, keep in mind that the `ObjectSID` attribute must be mapped to a unique attribute in PingOne. You can find more information on passing binary attributes in Passing binary attributes to PingOne in the PingOne Intergration Kit documentation..

Creating an issuance certificate in PingOne

About this task

The PingID Windows login - passwordless solution uses certificate-based authentication (CBA), so a certificate is required for each user that will be signing on. This requires that you create an issuance certificate in PingOne and then publish the certificate.

Steps

1. Create an issuance certificate in PingOne.

   Learn more in Adding a certificate and key pair in the PingOne documentation.

2. Publish the issuance (CA) certificate to Active Directory (AD):

   ```
   certutil -dspublish -f  <CA certificate filename>  NTAuthCA
   ```

3. To verify that the certificate was published, run the following command and make sure that you see the CA certificate in the list:

   ```
   certutil -viewstore "ldap:///CN=NTAuthCertificates,CN=Public Key
   Services,CN=Services,CN=Configuration,DC=<domain name>"
   ```

4. Import the CA certificate in the Group Policy Management Console (GPMC) to publish the CA certificate to end users' computers:

   1. Open the Group Policy Management Console (GPMC).
   2. Locate the relevant domain.
   3. Locate the group policy that you'll be using.

4. In the **Public Key Policies** section, select **Trusted Root Certification Authorities** and import the CA certificate.

Creating an authentication policy (Windows passwordless)

Steps

1. In the PingOne admin console, open the environment you are using for Windows login - passwordless.
2. Click the **Identities** icon.
3. Click **Attributes**.
4. In the list of attributes, locate the PingOne attribute that you mapped to `ObjectSID`.
5. Click the **Pencil** (✎) icon to edit the attribute properties.
6. Select the **Enforce Unique Values**checkbox. Confirm the choice if prompted to do so.
7. Click **Save**.
8. Click the **Experiences** icon.
9. Click **Authentication Policies**.
10. Click **Add Policy**.

    Result:

    The policy definition page opens.
11. Enter a name for the policy.
12. For **Step Type**, select **Windows Login Passwordless**.
13. In the **Match Attributes** list, select the attribute that you mapped to `ObjectSID`.

    > **Note**
    >
    > This list includes any attributes that you have specified as unique by selecting the **Enforce Unique Values** option.
14. **Optional:** Select the **Offline Mode** option if you want to allow users to sign on when PingOne or PingID are not available.
15. Click **Save**.

Creating and configuring a passwordless Windows login application in PingOne

About this task

After creating the authentication policy, you can now create the application for passwordless Windows login:

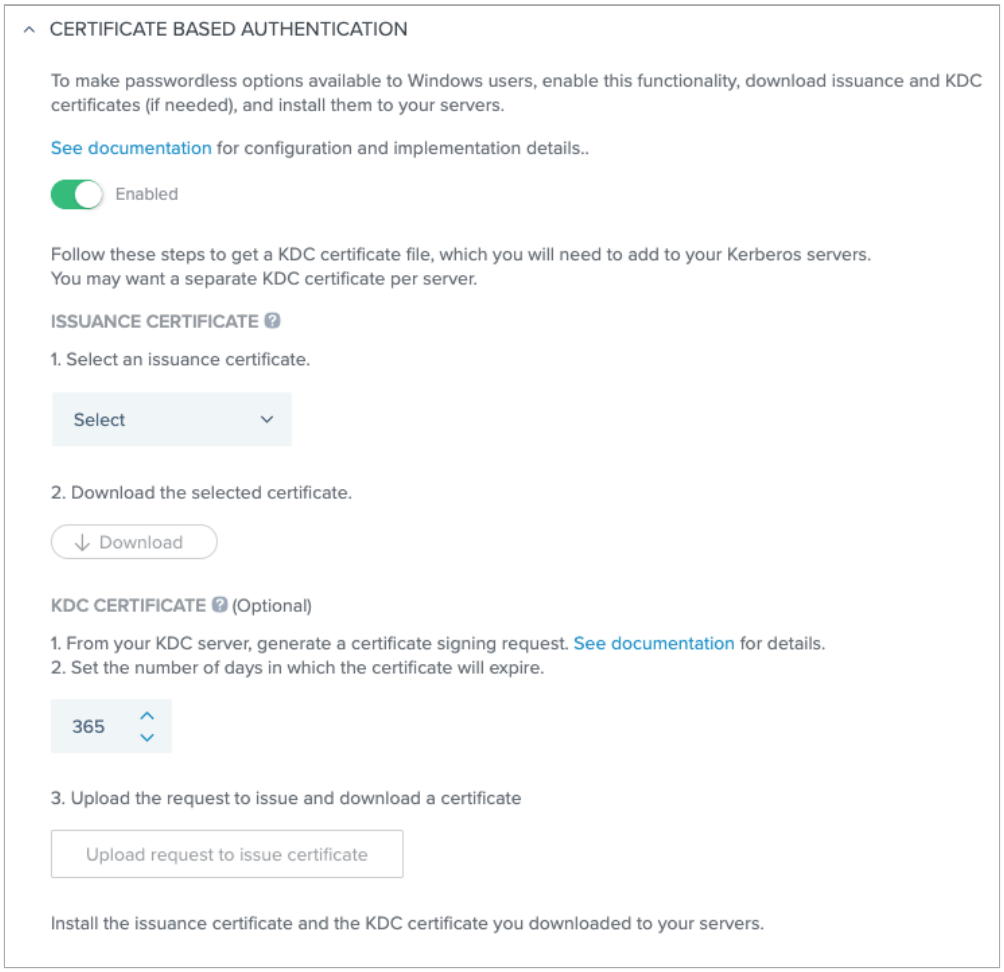
Steps

1. Go to the PingOne admin console and open the environment that you are using for Windows login - passwordless.
2. Click the **Connections** icon.

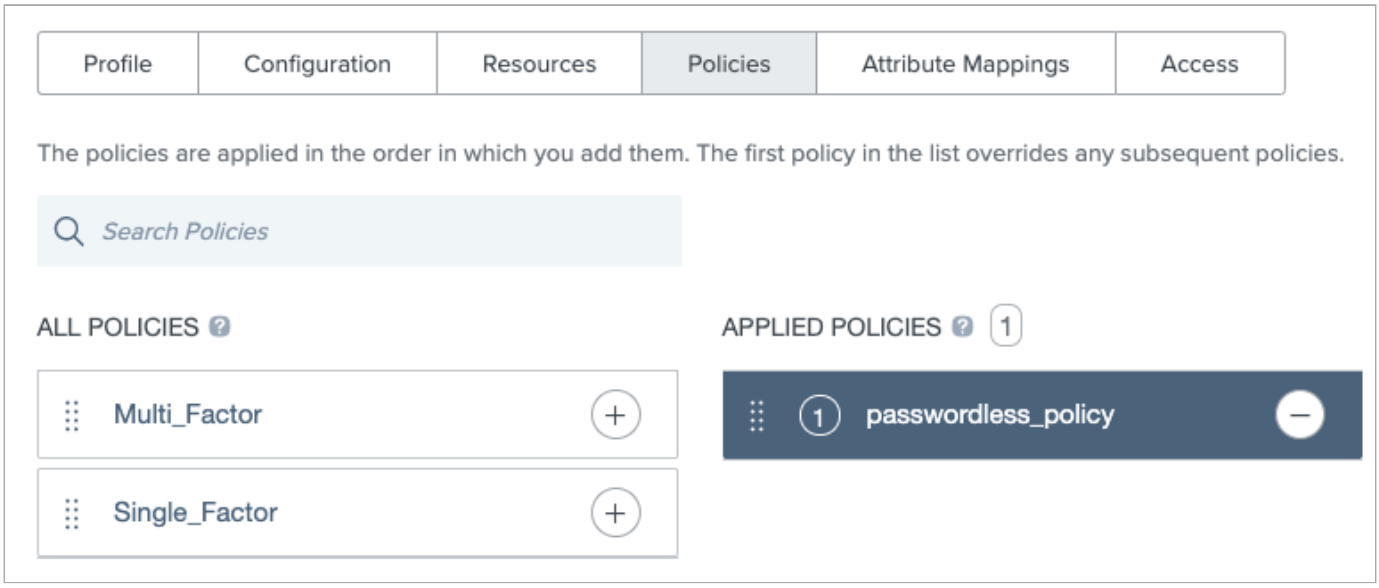
3. Click **Applications**.
4. Click the **+** icon to add a new application.
5. For the **Application Type**, select **Native App**.
6. Click **Configure**.
7. Enter a name and description for the application. **Click Next**.
8. Enter the redirect URL, `winlogin.pingone.com://callbackauth`, and then click **Save and Continue**.

> **Note**
>
> You can skip the **Grant Resource Access** and **Attribute Mapping** steps.

9. In the **Certificated Based Authentication** section, click the **Enabled** toggle.



10. Select an existing issuance certificate.
11. Go to the application's **Policies** tab and drag the passwordless policy that you created from the **All Policies** list to the **Applied Policies** list.



Generating a KDC certificate

About this task

If there is not yet a certificate for the KDC server that you will be using, you will need to generate one.

> **Note**
>
> The KDC certificate is used as part of the Kerberos PKINIT mutual authentication mechanism. If you already have a KDC certificate installed on your Active Directory Domain Controllers, you don't need to perform this task

Steps

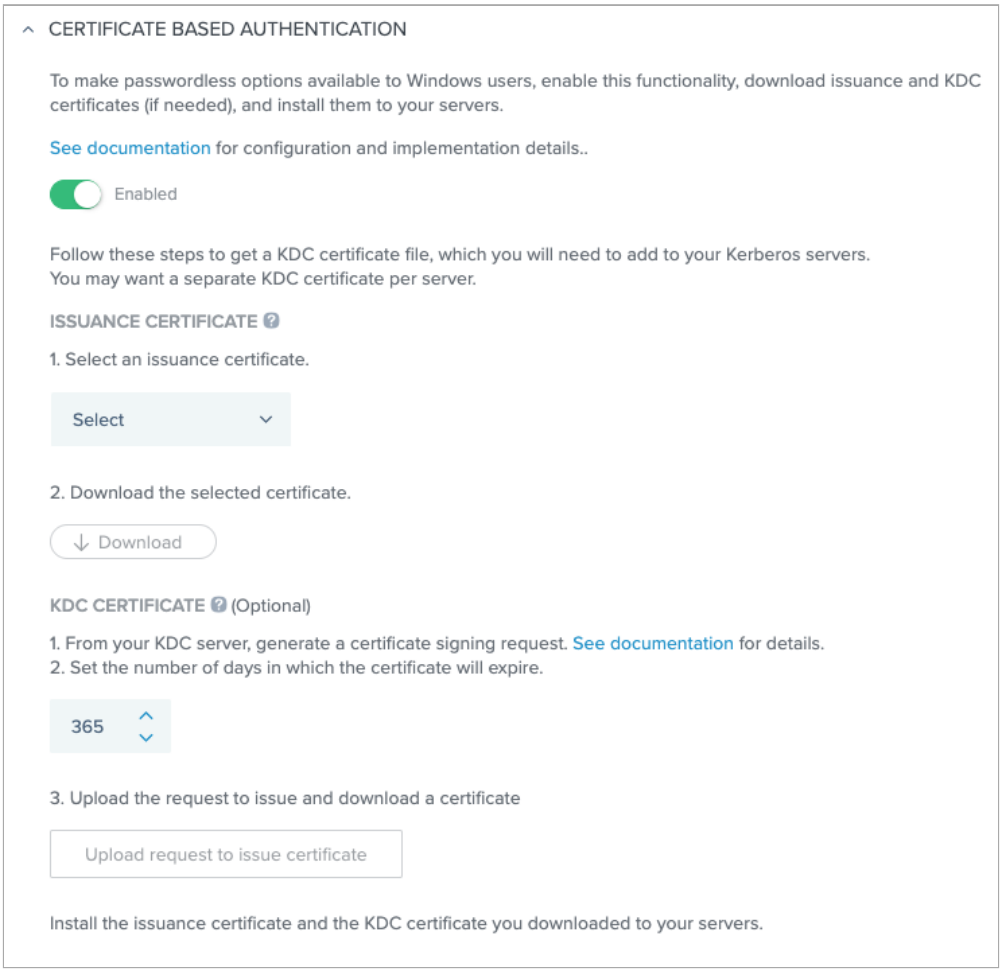
1. Create an `.inf` file containing the following information:

```
[newrequest]
      subject = "CN=<hostname>"
      KeyLength = 2048
      MachineKeySet = TRUE
      Exportable = FALSE
      RequestType = PKCS10
      SuppressDefaults = TRUE
      [Extensions]
      ;Note 2.5.29.17 is the OID for a SAN extension.
      2.5.29.17 = "{text}"
      continue = "dns=<DNS hostname>"
```

> **Note**
>
> For more information on the contents of `.inf` files for the **`certreq`** command, see Certreq in the Microsoft documentation.

2. Generate a certificate signing request from your KDC server by running `certreq -new '<path to the .inf file>' 'kdc.req'`.

3. In the PingOne admin console, open the application that you created for passwordless Windows login.
4. Click the **Configuration** tab of the application.
5. Scroll down to the **Certificate Based Authentication** section.



6. For the KDC certificate signing request that you created previously with the `certreq` command:
    1. Set the number of days until the certificate should expire.
    2. Click **Upload request** and **Issue Certificate** to have the certificate issued.

> **Note**
>
> The KDC certificate does not have to be signed by the issuance certificate that you created with PingOne. Any valid certification path will work.

7. Install the KDC certificate on your server:

```
certreq -accept -machine -f  <KDC certificate filename>
```

Installing the Windows login - passwordless integration on client computers

Before you begin

- To use the Windows login - passwordless feature, users' computers must be running Windows 10 and must support TPM 2.0.
- The first time that a user carries out passwordless Windows login, they must be online and connected to the organizational network because certificate enrollment requires a connection to Active Directory. Afterward, there is no need for a connection to the network, and authentication can be carried out online or offline for as long as the certificate is valid.

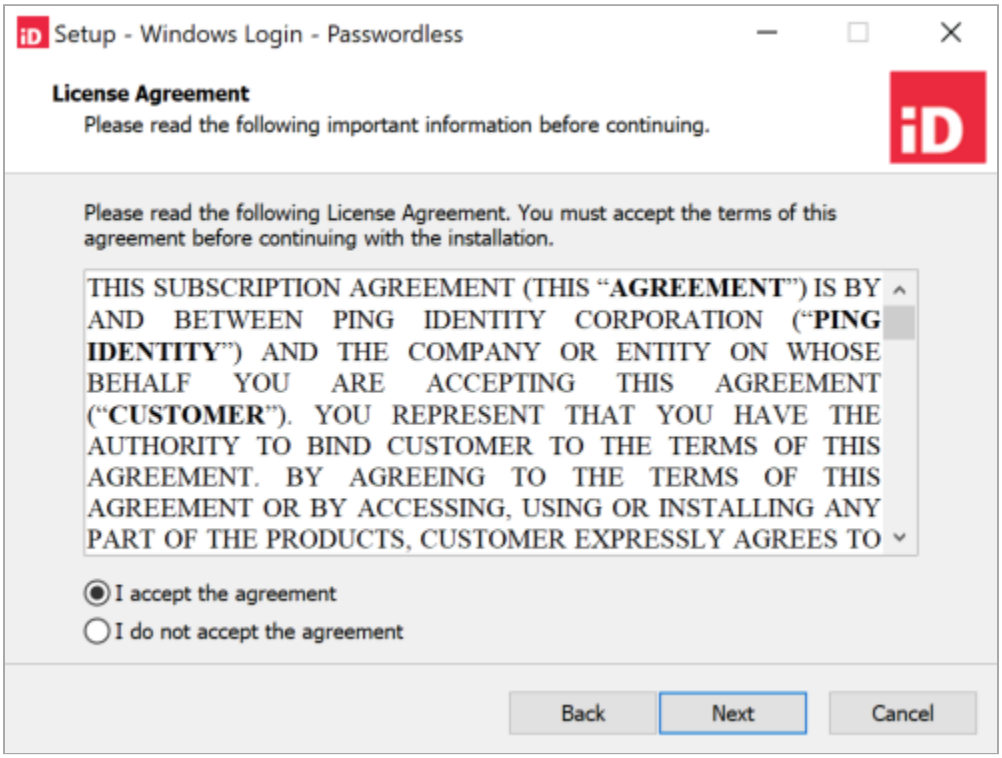
About this task

To install the integration for Windows login - passwordless on your users' computers using the UI-based method:

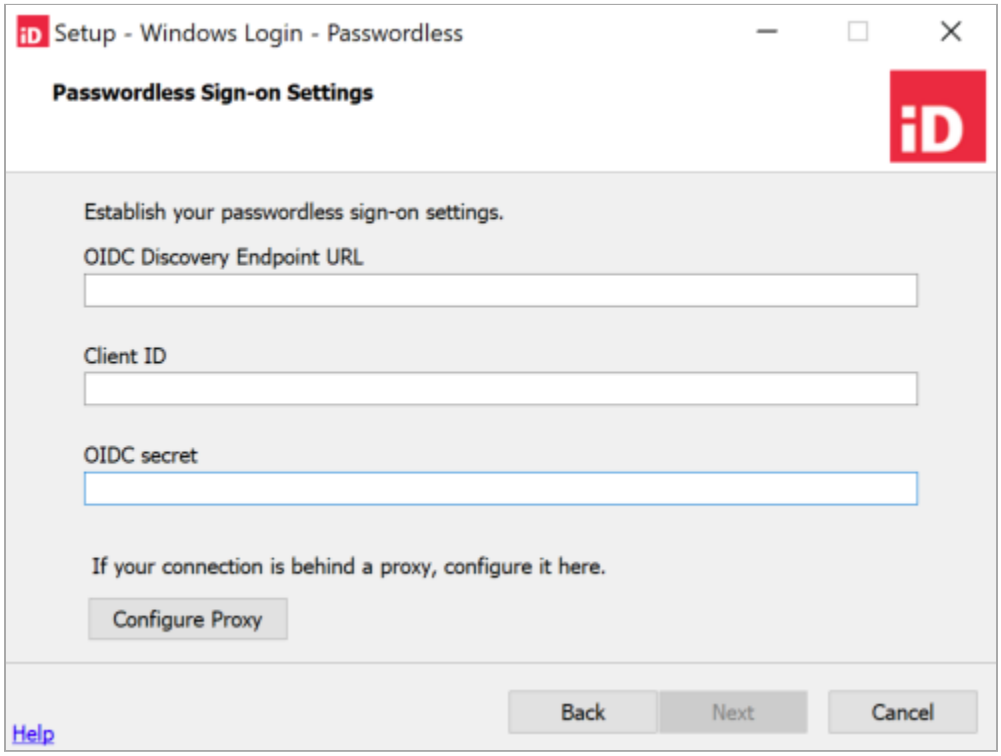
Steps

1. Run the provided executable, and when the welcome page is displayed, click **Next**.

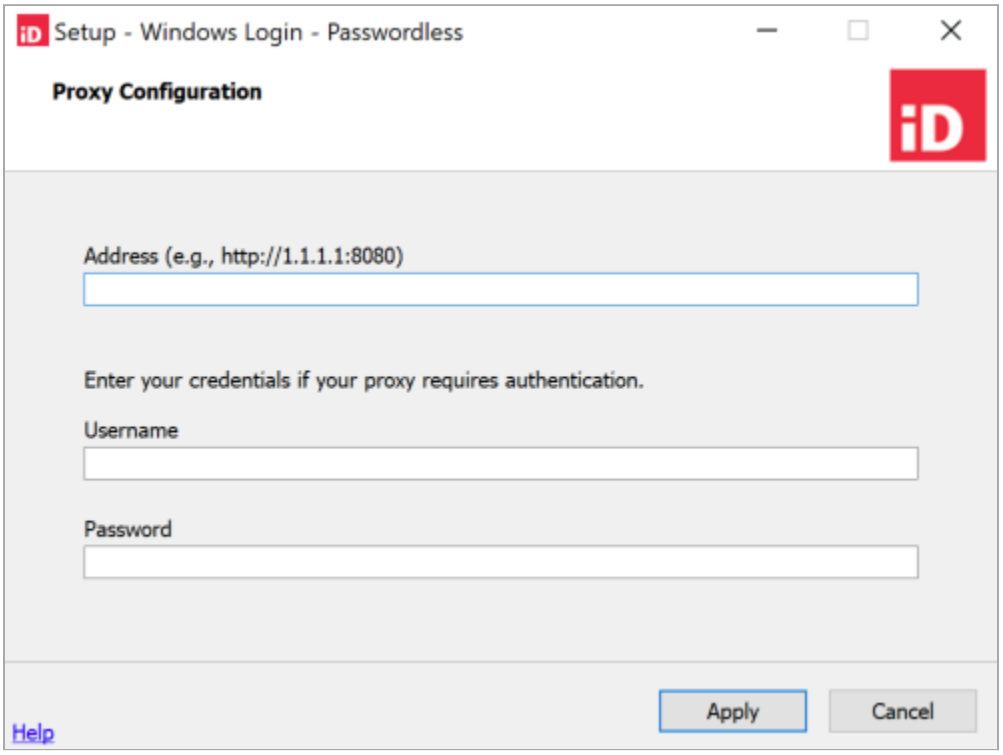
2. Accept the license agreement and click **Next**.



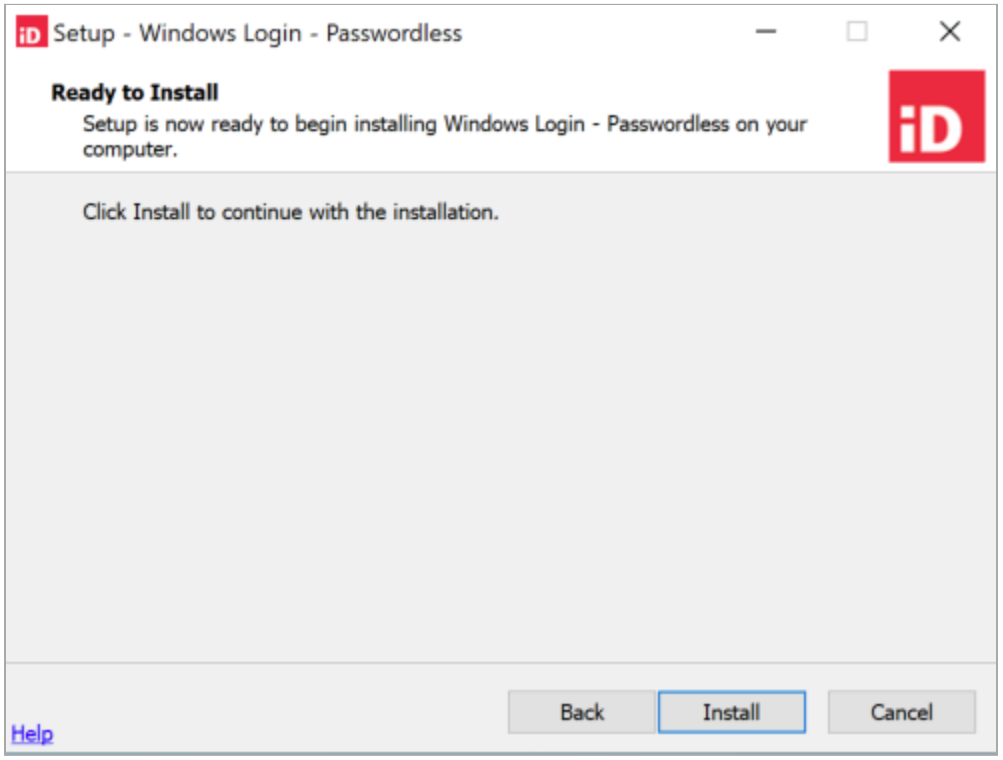
3. The settings that must be entered on the **Passwordless Sign-on Settings** page should be copied from the **Configuration** tab of the application that you created for Windows login - passwordless in PingOne. If your organization uses a proxy, click **Configure Proxy**. Otherwise, click **Next**.



4. If you clicked **Configure Proxy** in the previous step, enter the proxy information, click **Apply**, and when you are returned to the **Passwordless Sign-on Settings** page, click **Next**.



5. When the **Ready to Install** page is open, click **Install** to start the installation.



Using the PowerShell script for setting up Windows login - passwordless

About this task

You can use the `Configure-Passwordless.ps1` PowerShell script to quickly perform the steps required to set up Windows login - passwordless.

> **Note**
>
> Only use this for purposes such as informal testing or demonstrations. Do not use for a production instance.

Steps

- Run `Configure-Passwordless.ps1`.

  The script carries out the following steps:

  - Creates and installs the CA certificate, also to the group policy
  - Sets `externalId` to be a unique attribute
  - Creates the authentication policy
  - Creates and configures the passwordless Windows login application
  - Creates a KDC certificate: request creation, issuing of certificate from request, installation of certificate

    You can download the script from GitHub.

Troubleshooting Windows login - passwordless

If you encounter any issues with Windows login - passwordless, review the information that is recorded in the log files and the event information that is displayed in the **Audit** window in PingOne.

You can find detailed activity information regarding Windows login - passwordless in the log files that are located in the `logs` folder under the folder that you specified during installation (the default location is `C:\Program Files\Ping Identity\PingID\Windows Passwordless\logs`). To include a greater level of detail in the log files, contact customer support for instructions on how to set the logging level to **Debug**.

> **Note**
>
> For some of the log files, there is no mechanism to limit the file size. You shouldn't leave the logging at **Debug** level for an extended period of time.

The **Audit** window in PingOne includes information on events, such as certificate creation and user authentication. You can find more information in Audit section in the PingOne documentation.

Best Practices: Elevated Rights for PingDirectory

This document provides an overview of Ping Identity's recommendations for management of elevated rights in PingDirectory.

PingDirectory is designed as a security product and provides a number of mechanisms that can be used to provide fine-grained control over the elevated rights that can be assigned to an administrator or service account.

Capabilities

Privileges

PingDirectory has a number of defined privileges that are used for fine-grained control of privilege.

The capabilities of the Directory Manager account that is created by default during a PingDirectory install is granted by assignment of privileges. This default Directory Manager account itself does not possess any special privileges or capabilities beyond those assigned by privileges. Any account in the directory that is assigned the same privileges as that default Directory Manager will have exactly the same level of access as that Directory Manager account.

The privileges assigned to the Directory Manager account can be removed (or the account itself can be deleted) without impacting the functionality of the directory.

Privilege name	Root privilege	Privilege description
`audit-data-security`	Yes	Provides the ability to audit the security of data in any backend. The user still needs access control permission to perform the requested operation
`backend-backup`	Yes	Provides the ability to perform a backup of one or more backends with the server online via the tasks interface. The user still needs access control permission to perform the requested operation.
`backend-restore`	Yes	Provides the ability to perform a restore of a backend with the server online through the tasks interface. The user still needs access control permission to perform the requested operation.
`bypass-acl`	Yes	Provides the ability to bypass all access control evaluation for any type of operation. **Note** Users with the `bypass-acl` privilege can still be subject to other restrictions, such other privileges that might be required to process a particular operation.
`bypass-pw-policy`	No	Provides the ability to exempt an administrator from certain types of password policy evaluation when performing an operation against another user.
`bypass-read-acl`	No	Provides the ability to bypass all access control evaluation, but only for bind, compare, and search operations. Normal access control evaluation is still performed for add, delete, extended, modify, and modify DN operations.
`collect-support-data`	Yes	Allows the requester to invoke the collect-support-data tool using an administrative task or extended operation.

Privilege name	Root privilege	Privilege description
`config-read`	Yes	Provides the ability to perform search and compare operations in the server configuration. These operations are still subject to access control restrictions.
`config-write`	Yes	Provides the ability to perform add, delete, and modify operations in the server configuration. These operations are still subject to access control restrictions.
`disconnect-client`	Yes	Provides the ability to terminate an arbitrary client connection. The user still needs access control permission to perform the requested operation.
`exec-task`	No	Allows the requester to schedule an exec task.
`file-servlet-access`	Yes	Indicates that the requester may be permitted access to the content exposed by file servlet instances that require this privilege.
`jmx-notify`	No	Provides the ability to subscribe to receive JMX notifications.
`jmx-read`	No	Provides the ability to perform read operations using JMX.
`jmx-write`	No	Provides the ability to perform write operations using JMX.
`ldif-export`	Yes	Provides the ability to perform LDIF export operations with the server online through the tasks interface. The user still needs access control permission to perform the requested operation.
`ldif-import`	Yes	Provides the ability to perform LDIF import operations with the server online through the tasks interface. The user still needs access control permission to perform the requested operation.
`lockdown-mode`	Yes	Provides the ability to cause the server to enter and leave lockdown mode or to access the server while it is in lockdown mode. The user still needs access control permission to perform the requested operation.
`manage-topology`	Yes	Provides the ability to manage a topology of server instances, including adding servers to and removing servers from a topology. The user still needs access control permission to perform the requested operation.
`metrics-read`	Yes	Provides the ability to search or retrieve data in the metrics backend. The user still needs access control permission to perform the requested operation.
`modify-acl`	Yes	Provides the ability to modify access control rules. The user still needs access control permission to perform the requested operation.

Privilege name	Root privilege	Privilege description
`password-reset`	Yes	Provides the ability to change another user's password. The user still needs access control permission to perform the requested operation.
`permit-externally-processed-authentication`	No	Provides the ability for the requester to issue a bind request that uses the `UNBOUNDID-EXTERNALLY-PROCESSED-AUTHENTICATION` Simple Authentication and Security Layer (SASL) mechanism.
`permit-forwarding-client-connection-policy`	No	Provides the ability to request that an operation be processed using a specified client connection policy.
`permit-get-password-policy-state-issues`	Yes	Provides the ability for the requester to issue a bind request that includes the get password policy state issues request control. The bind request must also include the retain identity request control.
`privilege-change`	Yes	Provides the ability to alter the set of privileges assigned to an individual user or to change the set of privileges that can be automatically assigned to root users.
`proxied-auth`	No	Provides the ability to request that an operation be processed using an alternate authorization identity, such as using the proxied authorization or intermediate client request control or using a SASL authorization identity.
`server-restart`	Yes	Provides the ability to request a server restart using the tasks interface. The user still needs access control permission to perform the requested operation.
`server-shutdown`	Yes	Provides the ability to request a server shutdown using the tasks interface. The user still needs access control permission to perform the requested operation.
`soft-delete-read`	Yes	Provides the ability to retrieve, compare, modify, delete, or undelete soft-deleted entries. The user still needs access control permission to perform the requested operation.
`stream-values`	Yes	Provides the ability to use the stream directory values extended operation to obtain a list of all entry DNs or unique attribute values or to use the stream proxy values extended operation to obtain information from the global index. The user still needs access control permission to perform the requested operation.
`third-party-task`	Yes	Provides the ability to invoke a third-party task in the server. The user still needs access control permission to perform the requested operation.

Privilege name	Root privilege	Privilege description
`unindexed-search`	Yes	Provides the ability to perform an expensive unindexed search in a local DB backend. The user still needs access control permission to perform the requested operation.
`unindexed-search-with-control`	No	Provides the ability to perform an unindexed search if the request also includes the permit unindexed search request control.
`update-schema`	Yes	Provides the ability to alter the server schema. The user still needs access control permission to perform the requested operation.
`use-admin-session`	Yes	Provides the ability to use an administrative session to request that operations be processed in a dedicated thread pool.

Privileges are granted to an account by adding the desired privilege to the accounts `ds-privilege-name` attribute.

This attribute can be explicitly populated, populated with a virtual attribute, or a combination of the two.

Privileges allow us to grant accounts the ability to perform basic administrative tasks, such as `server-shutdown`, without needing to grant more powerful privileges, such as `bypass-aci`.

Access control instructions (ACI)

ACIs are used to define the level of access an account can have to entries and attributes in the directory. By default, PingDirectory is configured with an implicit deny, so read/write access to entries must be explicitly granted.

Client connection policy

A number of different client connection policies may be defined on a server. A client connection policy can, among other things, determine:

- Which branches of the directory are accessible
- Allow operation types (for example, search, add, delete, and modify)
- Allowed filter types
- Search size and time restrictions
- Attributes to be excluded from search results
- Attributes that cannot be included in search filters
- Attributes that cannot be modified (even if ACIs would normally allow)

Which client connection policy applies to an authenticated account can be determined by:

- Included/excluded IP address or IP range
- Connection type (HTTPs, LDAPs, LDAP)
- Location of authenticated account in the directory

- Group membership of authenticated account
- Attribute value contained in the authenticated account
- Authenticated account privileges

A common use of client connection policies is creating a connection policy for insecure LDAP communications where only accounts in specific groups are allowed to connect insecurely and can only see a limited number of entries and attributes.

ACI best practices

Best practices for the definition of ACIs in a PingDirectory environment include:

- Default to deny
  - Don’t define any global permissive ACIs for all users.
  - Ensure that unless explicitly granted, all security decisions are a deny.
- Group-based ACI
  - All ACIs should be assigned through group membership.
  - Do not assign ACIs to specific user accounts.
- Implement ACI identifiers and document thoroughly

Group based

Ping Identity highly recommends using a set of standardized, group-based ACIs to grant permissions to the directory. Managing the application of ACIs through group membership provides a number of benefits:

- Easier to understand
  - You only need to look at group membership to determine what a user has access to. You don’t need to decode ACIs to see which ACIs apply to a user.
- Simplified auditing (just look at group membership)
- Greatly simplifies the granting of rights
- Reduces the total number of ACIs
  - ACIs always get evaluated, so keeping the number of defined ACIs relatively small can help performance.
- Reduce risk and ACI proliferation
  - Application-specific or user-specific ACIs can be difficult to maintain. It can be difficult to know if an ACI is safe to remove or if it grants too many rights to an application. Group-based ACIs transform this issue into a group management issue that administrators are much better equipped to deal with.
  - Creating generic, reusable ACIs greatly reduces or eliminates the need to create custom, application-specific ACIs for each new application or unique use case.

Example

For a specific user organizational unit (OU) in the directory, you could create a set of group-based ACIs that grant the following:

- Read/Search access to non-sensitive attributes
- Read/Search access to sensitive attributes, such as SSN or date of birth.
- Write access to non-sensitive attributes
- Write access to sensitive attributes
- Create permission for specific entry types (create for `inetOrgPerson` or `groupOfUniqueNames`)
- Write access to specific attribute (for example, `uniqueMember` grants the ability to manage group membership separately from the ability to create/delete groups)
- Import/Export rights (to allow for `moddn` operations)
- Delete permission for specific entry types

This might initially look like a lot of ACIs to create for each OU in your directory. However, these ACIs greatly reduce or eliminate the need for the creation of future ACIs, as 95% of your conceivable use cases for granting of permissions can now be accomplished through group membership.

PingDirectory allows for the use of nested groups, so it is possible to create a Level1 Help Desk group and nest it into multiple ACI groups across multiple OUs to simplify administration.

Use implicit deny, not explicit

Using the above methodology means an entry will by default have no access to any entries in the directory. Any access granted to an account will be an accumulation of explicitly granted permissions. Starting from a position of no privilege and only having permissive ACIs greatly simplifies the evaluation of privileges when troubleshooting or verifying that an account has the correct rights.

One of the biggest issues with a deny ACI is that the deny is being implemented as a security control to remove access that would otherwise be granted. This implies that the account in question would by default have access to data it should not unless action is explicitly taken to deny that access.

Having only permissive ACIs means that mistakes in granting rights to a user will usually result in a user not having enough access. Using a deny ACI opens us up to the possibility of a mistake granting a user rights they should not have.

If you don't grant a user enough privileges, the user will usually let you know. Detection of this type of mistake is usually easy (the user is motivated to let you know) and poses low security risk to the organization.

If you grant a user or account too many permissions, no one will likely notice. Even if a user or application owner notices, they might not consider this an issue and are unlikely to report it. Detection of this type of mistake is very difficult and can pose a high security risk to the organization.

Use names with identifiers in ACIs and document

When an ACI has been in place for an extended period of time it might not be a simple process to determine why that ACI exists, what exactly it does, and if it's safe to modify or delete it. This presents a number of auditing and supportability concerns and can produce a directory whose security stance cannot be fully determined.

It is highly recommended when creating ACIs that each ACI contain a descriptive name with a unique identifier that can be cross-referenced with a version document containing:

- ACI Unique Identifier
- ACI High Level Description
- ACI Business Case
- Change order/request id used to implement the ACI
- Responsible party
- Parties to be informed if ACI needs to be deleted or changed

For example, we might have an ACI that looks like:

```
(target="ldap:///cn=changelog")(targetscope="subtree")(targetattr="*||+")(version
          3.0; acl "GL-SYNC-1: Allow Read access to changelog backend";allow(read,search,compare)
          groupdn="ldap:///cn=SyncReadGrp,ou=Groups,ou=Administrators,dc=example,dc=com";)
```

With a corresponding entry in the ACI document similar to the following.

GL-SYNC-1

Description	Grants read access to cn=changelog for the PingDataSync service account so it can monitor for changes to user data
Business case	CRM Unit needs to monitor for new customer entries in the directory so they can retrieve application generated unique identifiers and associate those with customer entries in the CRM database
Change order	Implemented 12/24/2020 under CR23423
Request ID	Requested 6/15/2020 with REQ 1231254
Area owner	US Customer Relations
Responsible parties	Manager – Alice Smith (bob.smith@company.com) BA – Bob Jones (bob.jones@company.com)

Privileges best practices

Privileges are assigned to an entry through the population of the `ds-privilege-name` attribute with a list of the privileges that entry should have.

Ongoing maintenance and auditing of privilege assignment can be challenging if privileges are assigned through direct population of the `ds-privilege-name` attribute. Ping Identity does not recommend direct population of this attribute except in special cases.

Ping Identity recommends the use of group-membership based virtual attributes to populate privileges.

For example, to assign the `pwd-reset` privilege a virtual attribute would be created similar to:

```
dsconfig create-virtual-attribute \
 --name ADM-Password-Reset-Priv --type constructed \
 --set enabled:true --set attribute-type:ds-privilege-name \
 --set enabled:true --set attribute-type:ds-privilege-name \
 --set group-dn:cn=ADM-PasswordReset,ou=groups,ou=admins,dc=example,dc=com \
 --set value-pattern:password-reset
```

Using this virtual attribute, an account can be granted the password reset privilege by adding the user to the `ADM-PasswordReset` group.

Exception

You might encounter a potential bug with applications that heavily use Proxy Auth privileges where security context changes multiple times over a single connection. This behavior is typically limited to applications such as PingFederate and Siteminder. An existing connection that's heavily used for Proxy Auth might forget what privileges it has unless they are explicitly assigned to the entry's `ds-privilege-name` attribute.

Client connection policy best practices

Client connection policies can be used to control what a client can see or do at a very high level. You can use client connection policies to enforce a certain level of security that is not available through other mechanisms.

There are a number of common use cases for client connection policies:

- Allow exceptions to requirement for secure connections based on a service account's group membership
- Limit the subtrees viewable or accessible to a client
    - Useful when storing both employee and customer data on the same server to logically isolate the branches from each other for most applications
- Restrict access or disallow access to PingDirectory based on client IP address (for example, don't allow adds/mod/writes from the Internet even if the account has ACIs granting those permissions)
- Place restrictions on allowable search filters
- Enforce limits on poorly behaving applications or applications that are yet to be vetted by the directory administrators
- The calculation of expensive virtual attributes can be restricted so that they only occur over a specific connection policy

The client connection policy associated with a particular connection can change over time. The client connection policy will be determined:

- When the connection is initially established
- After any successful Bind operation
- After a StartTLS request is received

Not a replacement for ACIs

Care should be taken not to rely too heavily on connection policies to enforce security that can be addressed at a lower level by ACIs or Privileges.

If you don't want a particular user to have access to `ou=Customers`, for example, you could create a connection policy that prevents that user from seeing `ou=Customers`. However, ACIs should also be created to ensure that the user does not have access to those entries if a client connection policy mistake is made.

> **Important**
>
> Creating new client connection policies can easily introduce issues in the client connection policy evaluation order and process a more permissive policy before the more restrictive policies.

Define unauthenticated/insecure policy

When clients connect to PingDirectory, the initial connection is almost always unauthenticated.

If you are deleting the default client connection policy (or modifying it), make sure that there is at least one client connection policy that allows for a connection to transition from an unauthenticated state to an authenticated state or to allow an unauthenticated connection to send a StartTLS control.

Set restrictive client connection criteria

The evaluation order defined for a client connection policy will determine which client connection policy a client receives if it meets the criteria for more than one client connection policy. This has the potential to create unexpected scenarios if a valid client connection state is not considered or tested during the design of the client connection criteria and client connection policies.

Best practice is to implement mutually exclusive client connection criteria where possible to reduce or eliminate reliance on the evaluation order index when a connection's client connection policy is determined.

Examples

The following is a sample list of group based ACIs that might be created for an OU that contains employee accounts:

```
dn: ou=people,ou=internal,dc=example,dc=com

aci: (target =
"ldap:///ou=people,ou=internal,dc=example,dc=com")(targetattr = "* || +") (version 3.0; acl "IP1 read internal people
ou"; allow
(search,read,compare)
(groupdn="ldap:///cn=ADM-InternalPeopleRead,ou=groups,ou=admins,dc=example,dc=com");)

aci: (target =
"ldap:///ou=people,ou=internal,dc=example,dc=com")(targetattr != "userpassword || authpassword") (version 3.0; acl
"IP2 write internal people"; allow (write) (groupdn="ldap:///cn=ADM-
InternalPeopleUpdate,ou=groups,ou=admins,dc=example,dc=com");)

aci: (target = "ldap:///ou=people,ou=internal,dc=example,dc=com")(targetattr = "userpassword || authpassword")
(version 3.0; acl "IP3 password update internal people"; allow (write)
(groupdn="ldap:///cn=ADM-InternalPeoplePwdReset,ou=groups,ou=admins,dc=example,dc=com");)

aci: (target = "ldap:///ou=people,ou=internal,dc=example,dc=com") (version 3.0; acl "IP4 add internal people"; allow
(add)
(groupdn="ldap:///cn=ADM-InternalPeopleAdd,ou=groups,ou=admins,dc=example,dc=com");)

aci: (target = "ldap:///ou=people,ou=internal,dc=example,dc=com") (version 3.0; acl "IP5 delete internal people";
allow (delete)
(groupdn="ldap:///cn=ADM-InternalPeopleDel,ou=groups,ou=admins,dc=example,dc=com");)

aci: (target = "ldap:///ou=people,ou=internal,dc=example,dc=com") (version 3.0; acl "IP6 move branch internal
people"; allow
(import,export)
(groupdn="ldap:///cn=ADM-InternalPeopleModDN,ou=groups,ou=admins,dc=example,dc=com");)
```

In this example, if we wanted to give the help desk the ability to search for user accounts, read user accounts, and reset passwords, we would place the help desk users into the following groups (either directly or, more likely, by creating a help desk group and nesting it into these groups):

- `ADM-InternalPeopleRead`
- `ADM-InternalPeoplePwdReset`

With descriptive group names, this makes the determination of a user's rights to the directory intuitively obvious.

One further item to note is that the ACI that grants write access to `userPassword` (IP3) will need to be used in conjunction with the `password-reset` privilege before the help desk user can reset an employee's password. In the Privileges best practices section, there is an example virtual attribute that is used to assign the `password-reset` privilege to users that are direct or indirect members of the group `ADM-PasswordReset`.

To enable help desk users to reset the passwords of employees, those help desk users would need to be added to `ADM-InternalPeoplePwdReset`, which grants write access to `userPassword`. Then you would need to nest the `ADM-InternalPeoplePwdReset` group into the `ADM-PasswordReset` group that is used by that virtual attribute, which will grant those help desk users the `password-reset` privilege.

Best Practices: Performance Testing PingDirectory

PingDirectory ships with several tools that you can use for performance testing.

Performance testing

The following table explains what each tool does.

Tool name	Description
`searchrate`	Test search performance
`authrate`	Test authentication performance
`modrate`	Test modification and write performance

Creating test entries

Before testing, you should create some entries to test with. The easiest way to do this is by creating a template that can be used with the `make-ldif` utility.

1. Create a template file called `templateTest.tmp`:

```
define suffix=dc=example,dc=com
define maildomain=example.com
define numusers=5001

branch: ou=PerfTest,[suffix]
subordinateTemplate: person:[numusers]

template: person
rdnAttr: uid
objectClass: top
objectClass: person
objectClass: organizationalPerson
objectClass: inetOrgPerson
givenName:  <first>
sn:  <last>
cn: {givenName} {sn}
initials: {givenName:1}<random:chars:ABCDEFGHIJKLMNOPQRSTUVWXYZ:1>{sn:1}
employeeNumber:  <sequential:0>
uid: user.{employeeNumber}
mail: {uid}@[maildomain]
userPassword: Password_123
telephoneNumber:  <random:telephone>
homePhone:  <random:telephone>
pager:  <random:telephone>
mobile:  <random:telephone>
street:  <random:numeric:5>  <file:streets>  Street
l:  <file:cities>
st:  <file:states>
postalCode:  <random:numeric:5>
postalAddress: {cn}${street}${l}, {st} {postalCode}
description: This is the description for {cn}.
```

2. To create an LDIF file that can be used to create test users, run `make-ldif` with the template file:

```
bin/make-ldif --templatefile templateTest.tmp --ldiffile testUser.ldif
```

3. To create the `testuser` organizational unit (OU) and the test users in the directory, apply the LDIF:

```
bin/ldapmodify -a -f testUser.ldif
```

Searchrate testing

Now you can run **`searchrate`**. Running this utility on the same server hosting the directory being tested will have some impact on performance results.

```
bin/searchrate --hostname [server name] --port [LDAP port] \
--bindDN "cn=directory manager" \
--bindPassword [directory manager password] \
--baseDN dc=example,dc=com \
--scope sub --filter "(uid=user.[1-5000])" \
--attribute givenName --attribute sn --attribute mail \
--numThreads 10
```

The output will look similar to:

```
      Recent      Recent         Recent      Recent       Overall     Overall
Searches/Sec  Avg Dur ms   Entries/Srch  Errors/Sec  Searches/Sec  Avg Dur ms
------------ ------------  ------------ ------------ ------------ ------------
    9703.655       0.204          1.000       0.000      8261.414       0.239
    9867.418       0.201          1.000       0.000      8796.509       0.225
```

Increasing the thread count will improve throughput for lower values. Higher thread counts will have diminishing returns on performance.

Authrate testing

You should test authentication rate. **`Authrate`** testing will be similar to **`searchrate`** testing.

The following command issues a search request to find a user and then a bind request to authenticate that user:

```
bin/authrate --hostname [server name] --port [LDAP port] \
--bindDN "cn=directory manager" --bindPassword [password] \
--baseDN dc=example,dc=com --scope sub --filter "(uid=user.[1-5000])" \
--credentials Password_123 --numThreads 10
```

The thread count should be varied to get an idea of how thread count (connection count) will impact performance. Test results will look similar to:

```
    Recent        Recent       Recent      Overall      Overall
 Auths/Sec    Avg Dur ms   Errors/Sec    Auths/Sec   Avg Dur ms
------------ ------------ ------------ ------------ ------------
  4131.452         0.482        0.000     3634.848        0.547
  4097.089         0.486        0.000     3789.004        0.525
  3829.309         0.520        0.000     3799.079        0.524
```

Modrate testing

The `modrate` tool tests the rate at which the directory can process modify operations. The arguments and output format will be similar to the other rate tools.

```
bin/modrate --hostname [server name] --port [LDAP port] \
--bindDN "cn=directory manager" \
--bindPassword [directory manager password] \
--entryDN "uid=user.[1-5000],ou=perftest,dc=example,dc=com" \
--attribute description --valueLength 12 \
--numThreads 10
```

Output will look similar to:

```
     Recent       Recent       Recent      Overall      Overall
   Mods/Sec   Avg Dur ms   Errors/Sec     Mods/Sec   Avg Dur ms
------------ ------------ ------------ ------------ ------------
   6505.814        1.530        0.000     6505.811        1.530
   8270.312        1.206        0.000     7387.366        1.349
   9295.419        1.073        0.000     8023.173        1.242
```

> **Note**
>
> Remember, varying the thread or connection count will impact performance results.

Best Practices: PingDirectory Operational Support

This document contains recommendations and best practices for the PingDirectory application onboarding process. Additionally, this document provides recommendations on supporting processes that could be used in conjunction with application integration.

Schema dictionary

The schema dictionary is an often overlooked portion of running a well-managed directory. It can be time-consuming to create from scratch, but you'll get that time investment back.

Attributes

A schema dictionary does not need to contain all of the attributes that you have defined in the schema. It should only contain those attributes that will get used by applications and other consumers of the directory and whose documentation is beneficial. You don't need to thoroughly document operational attributes or attributes that are never used.

Syntax

A schema dictionary should contain a thorough description of the attribute syntax and format in both the directory and in any data sources that feed that data into the directory.

For example, for `HREmployeeID` you might have something like this:

```
HREmployeeid
Directory: Multi-valued, Case-Insensitive Unicode, Min. Length 5,
Max. Length 90
Peoplesoft: Single-valued, VARCHAR(30)
ContractorDB: Single-valued, NUM(45)
```

It can save a lot of time if you can get your developers to look at this document when they write their code. Having an attribute that is always a five-digit number can be dangerous if the source definition is VARCHAR. Documenting the real data format prevents developers from using an implied data format and hard coding something that will cause problems later.

Example

Let's say you have an attribute called `HREmployeeID` that is used as a user's uid. In this example, it has always been a six-digit number. Then you hire your millionth employee. You're faced with what could be a difficult decision: how many of your application entitlement databases are using `HREmployeeID` as the key field? How many of these have that field defined as being six characters long because IDs are never longer than that? And how many of those have defined that field as an integer because it's always been an integer? Keeping the six character length and moving to alphanumeric will probably break some of your apps, but so will moving to a seven-digit integer.

The schema dictionary provides an easy means of preventing programmers and database administrators from making simple mistakes like this. It's not going to solve all of your issues, but most coders, database administrators, and application architects will read the document.

It's easy for a programmer to make a mistake with implied formats. Recovering from that mistake after it's been in production for a few years is hard. Just making sure app designers are aware that many of your attributes are case-insensitive all by itself will fully return your time investment.

Ownership and change process

Most of the data that's stored in your directory is not owned or managed by you, and you're probably the only person in your organization who knows that.

Adding some verbiage into the schema dictionary as to who the owner is for each attribute and what the change process is for that attribute will save you a lot of time.

This is also an opportunity to keep your directory clean. On a regular basis, you should reach out the data owners for each attribute and have them certify the information that you have recorded about that attribute.

If you can't find an owner for an attribute, you should remove it from your directory (after giving advance notice first to your app owners, especially the ones that might be using that attribute). The default owner of an unowned attribute that remains in your directory is you, which can lead to issues such as governance difficulties and audit compliance if you don't have accurate information about the attribute.

Once a year, you should provide attribute owners with a list of groups and accounts in the directory that have read and write access to their attributes. It's critical that access decisions to potentially sensitive attribute information are approved by the owners of that data and not by an area (the directory support team) that does not own that data.

Attribute metadata

Directory users are constantly getting themselves into trouble by thinking they understand what the attributes stored in your directory mean.

For example, with a `streetAddress` attribute, you know that a user has at least two business addresses in HR: a physical location and a physical mailing address. And you know that about 5% of the time, those two are different. You also know which of those two got mapped to `streetAddress` in our directory, but not everyone using the attribute is aware of that.

The schema dictionary is a great place to store information about your attributes that your users need. Initially, you won't have much metadata to worry about, but as identities get more complicated and start coming from a larger and larger number of disparate sources, the volume of metadata (and the importance of having it easily available) will grow.

Metadata can include:

- A description of what the attribute is
- Level of assurance
- Data classification
- Appropriate usage (for example, `streetAddress` sourced from Database A was collected under a EULA that prohibits usage for advertising)

> **Note**
>
> To understand more about metadata documentation, you should reading up on Master Data Management (MDM) methodologies. If you've been doing IDM for a while, you'll realize fairly rapidly that IDM is essentially MDM for identity.

Application onboarding

About the only time an application team will feel motivated to answer your questions and provide documentation about their application is while they're waiting for you to approve creation of their service account. While they're waiting, here is some of the data you will want to collect:

- Expected SLAs

  You'll want to collect data about how responsive they expect the directory to be. This can be used later as business requirements or justification when you want to request more server resources.

- Change windows

Knowing the change windows for your consuming applications can help you identify the best time to perform maintenance. When asking the application areas for this, it should be made clear that staying in their change windows will not be guaranteed and is just best effort.

- Financial impact

  How much revenue an application generates and how much money is lost when it is not available is very useful information to have when you are building a business case for funding and resources.

  - Revenue generated per year
  - Revenue lost for 1 minute / 10 minute / 30 minute / 1 hour / full day outage.

- Service account password change process

  Application support teams rotate into and out of their areas all of the time. If you ever get into a situation where you want to mandate a password change after an application has been in place for more than a couple of years you might discover that no one knows how to safely change that password.

- Contact information

  This can be useful to have documented somewhere if you notice that a specific application is creating issues in the directory (especially if the issues are only moderately impactful and don't rise to the level of an incident).

  You should send an email to all of your application contacts a few times a year so that you can keep the contact list updated.

Application profiling

PingDirectory provides the ability to filter the data written to access logs based on connection criteria (see the appendix for a configuration example). This provides us with a mechanism to create a custom access log that only contains operations performed by specific accounts.

This can be used with new or existing applications to collect data about how the application behaves. The bin/summarize-access-log utility can be used against this application-specific log file to generate an overview of the types of searches that are run by the application, index utilization, errors, and an operation response time histogram.

Having this information documented provides a useful comparison if the application begins to experience production issues and can greatly simplify troubleshooting.

Schema modification

The ability to extend the schema should not be delegated to any groups outside of the directory administrators.

If an application area needs to extend the schema they will need to document their requirements. Their documentation should include:

- Attribute names
- objectclass definitions
- Single or multi-valued
- Attribute syntaxes

- Attribute indexing requirements
- Who owns the data in the attribute
- How the data is stored in the system of record (if the directory will not be the system of record)

The application area might need some help and guidance to answer these questions.

Before extending the schema, the data owners associated with this new data should be contacted and their approval granted and documented.

JSON attributes

If an application's data storage requirements cannot easily be met by a defined, structured schema (for example, largely non-homogeneous data), consider creating application specific JSON attributes for the application.

Because JSON attributes can contain any JSON-formatted set of data, they're an ideal candidate for storing complex data and relationships that cannot be easily defined or stored in a traditional hierarchical data model.

Because there could be little enforcement as to what gets populated, care should be taken with JSON attributes, and their usage should be reviewed on an annual basis.

Optionally, you can place restrictions on JSON attributes to restrict things such as allowable keys, must and may keys, and syntaxes. You can find more information in Configuring JSON attribute constraints for details on how to implement JSON restrictions and JSON key indexing.

Best Practices: PingFederate SAML Signing Certificates

The following reference guide details the best practices for managing PingFederate Security Assertion Markup Language (SAML) signing certificate settings, depending on your partners' preferences.

Component

PingFederate 10.1

What are the best practices for signing certificate administration?

There is no cryptographic difference between self-signed or certificate authority (CA)-signed certificates that use the same algorithm and key-length. From a security perspective, they're the same, and many customers use self-signed certificates for SAML signing.

The following are some benefits to using self-signed certificates:

- You can set a longer lifetime, decreasing the amount of update maintenance required.
- You can use PingFederate's automatic certificate rotation feature: a new key and certificate pair are periodically generated based on your policy, and active connections in PingFederate using the policy rotate to the new certificate without intervention.
- If your partners have the ability to monitor and automatically reload your metadata, they are automatically updated. For partners that cannot monitor a metadata URL that you provide, it can require coordination to update.

For more information on managing certificates and certificate rotation, see Manage digital signing certificates and decryption keys in the PingFederate Server documentation.

Although there is no security benefit, your partners might require you to use a certificate issued from a Certificate Authority (CA) for trust reasons. Self-signed certificates do not have a trust chain. If the owner of the CA-signed certificate can have their recipient partners agree to an anchored trust model, this can be a great way to ease the administrative burden. In an anchored trust model, the recipient partner validates that the signing certificate was issued by an expected issuer they trust and that the subject distinguished name matches what is expected. Because the recipient partners validate the trust chain and the expected values, rather than the specific certificate, when a newly-issued certificate is used for signing that meets the same criteria, it passes validation.

Choosing between an anchored or unanchored trust model depends on what your partners support. There is no benefit to partitioning separate certificates for internal or external partners, unless the intention is to use a stronger algorithm or key-length for external partners. One certificate is acceptable if it meets your organization's security guidelines.

For more information on whether you need a CA-signed certificate for SAML signatures, see Do I need a trusted CA-signed certificate for SAML signatures? in Ping Identity's support community.

When all of your partners accept a self-signed signing certificate

1. Create a new self-signed certificate in PingFederate.
2. Set the validity period for as long as your security team allows. For example, three to five years.
3. Configure the certificate for certificate rotation with a creation buffer threshold of at least six months before the activation buffer threshold.

   > **Note**
   >
   > While the next new certificate approaches its expiration, you have six months to coordinate the new certificate transition with partners.

4. Send your partners the new certificate and document which ones can add it as a secondary decryption key or which ones need a coordinated cutover.

   > **Tip**
   >
   > Inventory any unused service provider connections.

5. Plan to stagger your cutovers in advance of the existing certificate expiry.

   > **Tip**
   >
   > Schedule your cutovers so that they all do not transition on the same day. This gives you time to test and troubleshoot while managing any amount of connections and minimizes the impact on the users of those applications.

6. Schedule enough downtime for you and partner to update the changing connection on both sides and to test and roll back, if needed. This ensures your users are not surprised that they cannot access an application and helps them plan accordingly.

> **Note**
>
> Depending on your certificate rotation policy time settings, a new certificate is created before the current certificate expires, which gives you time to coordinate again.

7. When the activation buffer threshold is reached, all of your partner connections in PingFederate using that certificate are automatically updated to use the new one, and you do not need to edit each connection yourself.

When your partners require a signing certificate issued by a trusted Certificate Authority

1. Obtain a new CA-issued certificate with a validity period for as long as the issuer allows.
2. Send your partners the new certificate and document which ones can add it as a secondary or which ones need a coordinated cutover.

> **Tip**
>
> Take an inventory of any unused service provider connections.

3. Discuss with your partners if they can and want to use the anchored trust model.

> **Tip**
>
> Document which partners support and configure the anchored trust model. The next time you need to update the CA-issued certificate, there is no action needed from your partners.

4. Plan to stagger your cutovers in advance of the existing certificate expiry.

> **Tip**
>
> Schedule the cutovers so that they all do not transition on the same day. This gives you time to test and troubleshoot while managing any amount of connections and minimizes the impact on the users of those applications.

5. Schedule enough downtime for you and partner to update the changing connection on both sides and to test and roll back, if needed. This ensures your users are not surprised that they cannot access an application and can plan accordingly.

> **Note**
>
> Certificate rotation is not possible with CA-signed certificates, so when the new certification expires, repeat this process. You need to manually update each connection in PingFederate and coordinate with your partners that do not use the anchored trust model.

Best Practices: Performance Testing for PingFederate

This document provides an overview as well as general guidelines related to performance testing methodology for testing a PingFederate server prior to that system entering a customer production environment.

Many PingFederate deployments are still rolled out into production without any performance and scalability testing. This document shows you how Ping Professional Services approaches performance testing and can assist in identifying and thinking through potential bottlenecks so that you can deploy your servers with confidence.

Audience

This is intended as a starting point for anyone interested in learning or expanding their knowledge of PingFederate performance testing. Performance testing should always be undertaken with a consistent methodology and well-known tooling. This article will not make you a seasoned performance tester because that can take many years of experience. Instead, the basic skills explained here will give you some familiarity with the testing techniques and approaches Ping Identity takes when approaching such an important topic.

Why do performance testing?

Low-performing PingFederate applications do not deliver their intended benefits to an enterprise. Slow authentication performance or minimal scalability result in a net loss of time and money. If PingFederate is not delivering in a highly performant manner, this reflects poorly on all of the architects and consultants that delivered the solution to the customer.

Lesson 1: Load generation

Before you embark on your performance testing journey, you must ensure there is sufficient hardware for load/traffic generation.

A common misconception of performance testing efforts is that a load generator is a piece of software that can generate massive amounts of load on very little hardware. Always keep in mind that the load generator must have sufficient access to hardware in order to generate the load required in order to test an enterprise-scale system.

A customer could purchase a load generator such as Loadrunner and incorrectly assume that they can install it on a couple of modestly-sized PingFederate instance and expect it to generate thousands of concurrent users.

Remember, load generators are software, and they should be treated as such. All software is bound by resource use and design. Everything has performance limitations.

For anyone doing performance testing, size your client hardware like you would your target system. For testing PingFederate, the testing hardware should almost always overpower the PingFederate servers because PingFederate transactions are very fast. If your load testing system is not able to generate requests at a sufficiently high rate, you can mistakenly assume that you've hit the peak of PingFederate's capability, when in reality your load generator is not able to sufficiently stress PingFederate.

Make sure that if you or your customer is starting a performance and load testing endeavor, you are using adequate testing hardware.

Use a large system to test a small system. Use a 12-core load generator to performance test a quad-core PingFederate system. Make sure your load generator has at least two to three times the performance of the test system. You need power to test power. The faster the response times are for your load tests, the more powerful the hardware that you need to pump through those requests.

Ask yourself:

- Are you connected to a network segment that can handle that load/amount of network traffic?
- What about proxy servers that can introduce delay and have scalability issues themselves? Could that be interfering with your test?

Lesson 2: Results validation

Don't use intrusive validation in your test cases. For example, when you have a test case, you want to make sure that the validation of the actual result is as lightweight as possible. When you validate the result, you want to make sure that you pick the simplest and fastest way to do so.

For example, if you get a sign on page back from a request and you want to validate whether the sign on page is there, don't look through the entire body of the HTML to make sure it's exactly the same as the previous one that came in. Instead, look for specific key information in the request that comes back. If you are testing identity provider (IdP)-initiated single sign-on (SSO), use a simple, lightweight adapter on the service provider (SP) side that returns a simple, small, static HTML page.

If you have too heavy of a validation approach, it slows you down overall. This in turn means you might not really understand where the bottleneck is located.

Lesson 3: Warm up your server

As far as the backend goes, tune and warm up the system. When you deploy PingFederate out-of-the-box on a server with 8 GB RAM/multi-core system, understand it is not fully tuned. Make sure that you go through some sort of tuning process so that PingFederate can use the resources that are available to it.

Don't test a cold system because cold systems don't yield typical results. Don't just restart a PingFederate server and immediately hit it with load, as this is not a valid approach. Java by nature improves over time with the just-in-time (JIT) compilation. Hot spots in the code are compiled rather than interpreted after a certain amount of time and if they are really hot, they can be inlined. You want to make sure that you warm the system up.

A server is typically running all the time, and it is reasonable to say that JIT compilation will have occurred. Because you want to make sure that you test a system that would otherwise be in that state, warm it up before you test it.

Lesson 4: Things to keep in mind when monitoring performance

Performance monitoring or resource monitoring is an act of non-intrusively collecting or observing performance data from an operating or running application. As with any Java application, enterprise applications are affected by garbage collection performance.

In contrast, performance profiling is an act of collecting performance data from an operating or running application that might be intrusive on application throughput or responsiveness.

Profiling is rarely done in production environments. You want to avoid the use of profiling system, as they will affect your test.

Do not use excessive logging or harsh monitoring tools that affect the overall performance of the product. Just attaching JConsole monitoring to the system can affect runtime performance and give you results that are not usable because the system is spending part of its time responding to requests from the monitoring tools. Doing this results in additional load on the system that you are not yet accounting for.

CPU

For an application to reach its highest performance or scalability, it needs to not only take full advantage of the CPU cycles available to it, but also to use them in a non-wasteful manner. Making efficient use of CPU cycles can be challenging for multithreaded applications running on multiprocessor and multicore systems. Additionally, it is important to note that an application that can saturate CPU resources does not necessarily imply it has reached its maximum performance or scalability. To identify how an application is utilizing CPU cycles, monitor CPU utilization at the operating system level with tools such as perfmon or typeperf (command-line tool) or System Manager, with top being the command-line example.

Linux has vmstat, which shows combined CPU utilization across all virtual processors. Vmstat can optionally take a reporting interval, in seconds, as a command-line argument. If no reporting interval is given to vmstat, the reported output is a summary of all CPU use data collected since the system has last been booted. When a reporting interval is specified, the first row of statistics is a summary of all data collected since the system was last booted.

Disk

PingFederate disk operations/disk I/O should be monitored for possible performance issues. Application logs write important information about the state or behavior of the application as various events occur. Disk I/O utilization is the most useful monitoring statistic for understanding application disk usage because it is a measure of active disk I/O time. Disk I/O utilization along with system or kernel CPU utilization can be monitored using iostat.

When monitoring applications for an extended period of time, such as several hours or days, or in a production environment, many performance engineers and system administrators of Linux systems use sar to collect performance statistics. With sar, you can select which data to collect, such as user CPU utilization, system or kernel CPU utilization, number of system calls, memory paging, and disk I/O statistics. Data collected from sar is usually looked at after the fact, as opposed to while it is being collected.

Observing data collected over a longer period of time can help identify trends that may provide early indications of pending performance concerns. You can find additional information on what performance data can be collected and reported with sar in the Linux sar man pages.

In summary, just remember the “observer effect” and that seeing is changing.

Lesson 5: Did the test return expected results?

Do you get the expected results when running your tests? A performance test is based on a functional test and therefore must perform functionally first. The key metrics for performance and reliability result analysis are response time and throughput. Generally speaking, these two metrics will always be the most important, and they are offset by resource use.

You want to look at:

1. What is the response time for a given request?
2. How much data can you push through the system?
3. How many requests can you process of a given type?

After you have this information, some pass and fail criteria can be handed down from product managers. If product managers have specific criteria that must be met for those response time targets and resource use targets, those must be taken into account by the performance tester, and those are usually the ones that have to be met before you can consider something to be ready for production.

Lesson 6: Tools of the trade

A useful tool that is HTTPS based is Apache JMeter, which you can use to send HTTPS requests to any given PingFederate server.

Tune JMeter for optimal performance where appropriate. JMeter is just software and is subject to the same resource constraints as any application. The PingFederate Capacity Planning Guide has recommendations on this, so be sure to review it.

Lesson 7: Understand and tune the infrastructure

PingFederate is only one part of the data center infrastructure, and it has a great deal of reliance on other systems performing properly.

Ensure PingFederate is the only application running on the test systems, or at least be aware of the fact that other applications might be running on the PingFederate server. Network latency between test systems can affect results, so ensure that network infrastructure is robust, and don't take for granted bandwidth or latency. Be aware of any firewalls or proxy servers because these can cause issues with data transfer latencies between systems. Before conducting performance tests, make sure that you have a uniform configuration across all PingFederate servers. A non-uniform configuration often manifests itself as a single cluster member that has higher CPU use and a higher latency because it is garbage collecting more frequently.

Lastly, don't forget disk latency. Make sure that if you must log information for audit purposes or other reasons, you are writing to a fast disk.

Summary

This document has given general guidelines related to performance testing methodologies for testing a PingFederate service prior to running in a real production environment. It's shown some of the common reasons why failure to do effective performance testing can lead to a system that does not perform up to expectations.

Our next installment of this series will move into a discussion of how to set up an actual performance test and give some common examples of scripts and tools used for that.

Standards and Protocols Use Cases

Use case	Description
Changing the federation protocol in Office 365 from WS-Federation to SAML2P	Office 365 can use either SAML2P or WS-Federation to authenticate passive profiles or web-based clients. This task details changing the federation protocol configuration of your Office 365 domain from WS-Federation to SAML2P.
Configuring browsers for Kerberos and NTLM	The PingFederate Integrated Windows Authentication (IWA) adapter uses the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) for Kerberos and NTLM authentication.
Integrated Windows Authentication Group Policy browser settings	Apply browser settings using Group Policy.
Providing a persistent SAML NameID format in PingFederate	Use a custom SAML NameID format by defining a hidden attribute in the PingFederate attribute contract.
Using OpenSSL s_client commands to test SSL connectivity	Test SSL connectivity with `s_client` commands to check whether the certificate is valid, trusted, and complete.

Changing the federation protocol in Office 365 from WS-Federation to SAML2P

Office 365 can use either SAML2P or WS-Federation to authenticate passive profiles or web-based clients. This task details changing the federation protocol configuration of your Office 365 domain from WS-Federation to SAML2P.

Before you begin

- Connect PowerShell to Office 365

About this task

Change the federation protocol from WS-Federation to SAML2P in Office 365 using PowerShell.

Steps

1. Sign on to Office 365 PowerShell as an administrator.

```
PS C:\Users\Administrator> Connect-MsolService
```

2. Show current settings.

```
PS C:\Users\Administrator> Get-MsolDomainFederationSettings -domainName  Office 365 domain name  | Format-List
*

ExtensionData                   : System.Runtime.Serialization.ExtensionDataObject
ActiveLogOnUri                  : https://pf1.pinggcs.com:9031/idp/sts.wst
FederationBrandName             : Ping Identity
IssuerUri                       :  Office 365 domain name
LogOffUri                       : https://pf1.pinggcs.com:9031/idp/prp.wsf
MetadataExchangeUri             : https://pf1.pinggcs.com:9031/pf/sts_mex.ping?
PartnerSpId=urn:federation:MicrosoftOnline
NextSigningCertificate          :
PassiveLogOnUri                 : https://pf1.pinggcs.com:9031/idp/prp.wsf
PreferredAuthenticationProtocol : WsFed
SigningCertificate              : MIICX...
```

3. Save the settings to a variable.

```
PS C:\Users\Administrator> $saml = Get-MsolDomainFederationSettings -DomainName  Office 365 domain name
```

> **Tip**
>
> Save the old settings to a file for easy recovery.
>
> ```
> PS C:\Users\Administrator> Get-MsolDomainFederationSettings -DomainName  Office 365 domain name  |
> Export-Clixml dfs-pf-wsfed.xml
> ```

4. Update the variable to use SAML2P endpoints for the passive profile.

```
PS C:\Users\Administrator> $saml.PassiveLogOnUri = "https://pf1.pinggcs.com:9031/idp/SSO.saml2"
PS C:\Users\Administrator> $saml.LogOffUri = "https://pf1.pinggcs.com:9031/idp/startSLO.ping"
```

5. Disable SSO from the domain.

```
PS C:\Users\Administrator> Set-MsolDomainAuthentication -DomainName  Office 365 domain name  -Authentication
Managed
```

6. Use `Set-MsolDomainAuthentication` to set the `$saml` variable to enable federation.

```
PS C:\Users\Administrator> Set-MsolDomainAuthentication -DomainName  Office 365 domain name  -
FederationBrandName $saml.FederationBrandName -Authentication Federated -PassiveLogOnUri $saml.PassiveLogOnUri
-ActiveLogOnUri $saml.ActiveLogonUri -SigningCertificate $saml.SigningCertificate -IssuerUri $saml.IssuerUri -
LogOffUri $saml.LogOffUri -PreferredAuthenticationProtocol "SAMLP"
```

7. Review the results.

```
PS C:\Users\Administrator> Get-MsolDomainFederationSettings -domainName  Office 365 domain name  | Format-List
*

ExtensionData                   : System.Runtime.Serialization.ExtensionDataObject
ActiveLogOnUri                  : https://pf1.pinggcs.com:9031/idp/sts.wst
FederationBrandName             : Ping GCS
IssuerUri                       :  Office 365 domain name
LogOffUri                       : https://pf1.pinggcs.com:9031/idp/startSLO.ping
MetadataExchangeUri             : https://pf1.pinggcs.com:9031/pf/sts_mex.ping?
PartnerSpId=urn:federation:MicrosoftOnline
NextSigningCertificate          :
PassiveLogOnUri                 : https://pf1.pinggcs.com:9031/idp/SSO.saml2
PreferredAuthenticationProtocol : Samlp
SigningCertificate              : MIICX...
```

8. Save the new settings to a different file.

```
PS C:\Users\Administrator> Get-MsolDomainFederationSettings -DomainName  Office 365 domain name  | Export-
Clixml dfs-pf-samlp.xml
```

Troubleshooting

For troubleshooting, see the following to restore the federation protocol settings back to WS-Federation from SAML2P:

1. Restore the saved settings to a variable.

```
PS C:\Users\Administrator> $wsfed = Import-Clixml dfs-pf-wsfed.xml
```

2. Disable SSO from the domain.

```
PS C:\Users\Administrator> Set-MsolDomainAuthentication -DomainName  Office 365 domain name  -Authentication
Managed
```

3. Use `Set-MsolDomainAuthentication` to enable WS-Federation using the `$wsfed` variable.

```
PS C:\Users\Administrator> Set-MsolDomainAuthentication -DomainName  Office 365 domain name  -
FederationBrandName $wsfed.FederationBrandName -Authentication Federated -PassiveLogOnUri
$wsfed.PassiveLogOnUri -ActiveLogOnUri $wsfed.ActiveLogonUri -SigningCertificate $wsfed.SigningCertificate -
IssuerUri $wsfed.IssuerUri -LogOffUri $wsfed.LogOffUri -PreferredAuthenticationProtocol "WSFED"
```

Configuring browsers for Kerberos and NTLM

The PingFederate Integrated Windows Authentication (IWA) adapter uses the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) for Kerberos and NTLM authentication.

You can find IWA adapter system requirements in the IWA documentation.

Read the following sections for instructions specific to the browsers you want to configure.

Configuring Apple Safari

Safari on Windows supports SPNEGO with no further configuration. SPNEGO supports Kerberos if the computer is domain-joined and logged in by a domain user, otherwise SPNEGO negotiates NTLM.

Safari on Mac OS X supports SPNEGO with Kerberos if Mac OS is joined to Active Directory (AD), otherwise SPNEGO negotiates NTLM.

For information on joining Mac OS to AD, see Integrate Active Directory.

Configuring Microsoft Edge

Before configuring Microsoft Edge for Kerberos and NTLM, determine whether you have the legacy or Chromium version.

Legacy

To configure Microsoft Edge (Legacy), see Kerberos Adapter does not work for Edge Browsers in Windows 10 for SSO in the Ping Identity Knowledge Base.

Chromium

To configure Microsoft Edge (Chromium), see Kerberos unconstrained double-hop authentication with Microsoft Edge (Chromium) in the Microsoft documentation.

Configuring Internet Explorer and Google Chrome on Windows for Kerberos and NTLM

Add sites to the trusted zone to enable the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO).

About this task

By default, any IWA authentication request originating from an Internet host is not allowed. The default setting only allows clients to automatically provide credentials to hosts within the intranet zone. Sites are considered to be in the intranet zone if the connection was established using a Universal Naming Convention path (for example, `pingsso`), the site bypasses the proxy server, or host names don't contain periods (for example, `http://pingsso`).

Most PingFederate single sign-on (SSO) connections use the fully qualified domain name, so they won't be categorized as being in the intranet zone. Configure the browser to trust the host by adding the PingFederate hostname to the trusted sites zone.

The default setting, **Automatic logon with current user name and password**, uses Kerberos if available and NTLM if not. The setting **Prompt for user name and password** only uses NTLM.

If Internet Explorer Enhanced Security Configuration is enabled, a login prompt overrides the automatic login behavior. This prompt allows Kerberos and NTLM functionality, however it does not use the cached credentials from the user login.

To configure Internet Explorer and Google Chrome to support SPNEGO:

Steps

1. From the Control Panel, go to **Network and Internet → Internet Options → Security**.

2. Click **Trusted Sites**, then click **Custom Level**.
3. Under User Authentication, select**Automatic logon with current user name and password**. Click **OK**.
4. On the **Security** tab, click **Trusted Sites**, then click **Sites**.
5. In the **Add this website to the zone** field, enter the PingFederate server's hostname and click **Add**. Click **Close**.

> **Note**
>
> You can include an asterisk in front of the domain suffix to trust any host name within the AD domain (for example,**.ADdomain.pingidentity.com*).

Result

SPNEGO supports Kerberos if the computer is domain-joined and logged in with an AD user account.

SPNEGO negotiates NTLM on non-domain-joined computers. You are prompted for credentials, for which you would enter *<ADdomain>**<username>* and the password.

> **Note**
>
> The NetBIOS domain name (*<ADdomain>* in the above example) must qualify the username if:
>
> - The computer is not joined to an AD domain, or
> - There are multiple AD domains or forests and you are authenticating over a cross-domain trust.

You can add the PingFederate URL to the local intranet zone as an alternative to adding it to the trusted sites zone. Reasons for this vary based on the network design of the environment, but setting **Automatic logon with current user name and password** for the trusted sites zone implies that negotiate/authorization credentials might be sent in requests to sites outside of the intranet zone.

Configuring Google Chrome on Mac OS for Kerberos and NTLM

Authorize hosts for the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO) using the terminal.

About this task

SPNEGO works on Chrome without configuration, but only negotiates NTLM. To enable Kerberos, you must authorize host or domain names for SPNEGO protocol message exchanges. Do this from Terminal or by joining Mac OS to AD. For information on joining Mac OS to AD, see Integrate Active Directory. For iOS, only NTLM via SPNEGO has been tested. Kerberos has not been verified.

Configure `AuthServerWhitelist` from the Terminal:

Steps

1. Within your Mac OS Terminal, run **`kinit`** to get an initial ticket-granting ticket from your Kerberos domain controller to request service tickets for the IWA adapter.

```
>kinit  <joe@ADdomain.com>
joe@ADdomain.com's Password:  <YourPassword>
```

2. Go to the Chrome directory and start Chrome with the `AuthServerWhitelist` parameter.

```
>cd  </Applications/Google Chrome.app/Contents/MacOS>
>./"Google Chrome" --auth-server-whitelist="<*.addomain.com>"
```

> **Note**
>
> Some services require delegation of the users identity. By default, Chrome does not allow this. The `AuthNegotiateDelegateWhitelist` policy points Chrome to a server to delegate credentials. Add this parameter to the above command by specifying `--auth-negotiate-delegate-whitelist="*.adexample.com"`.

Result:

This setting persists every time Chrome is launched.

3. Run `kinit` every 10 hours for Chrome to request service tickets for the IWA adapter.

Configuring Mozilla Firefox for Kerberos and NTLM

Configure a list of trusted sites for the Simple and Protected GSS-API Negotiation Mechanism (SPNEGO).

About this task

Firefox rejects all SPNEGO challenges from any web server by default. You must configure a whitelist of sites permitted to exchange SPNEGO messages with the browser.

Steps

1. In the Firefox address bar, enter `about:config`. Click **I accept the risk!**
2. Search for the following preferences:
   - `network.negotiate-auth.trusted-uris`
   - `network.automatic-ntlm-auth.trusted-uris`
3. Double-click each of the preferences and enter any host or domain names in the **Enter string value** field, separated by commas. Click **OK**.

> **Note**
>
> You can add a period in front of the domain suffix to trust any hostname within the domain (for example, *.adexample.pingidentity.com*).

Result

SPNEGO supports Kerberos if the computer is joined to Active Directory (AD) and logged on with a domain user account, otherwise SPNEGO negotiates NTLM.

Firefox on Mac OS supports both Kerberos and NTLM if the computer is joined to AD, otherwise only NTLM negotiates.

Integrated Windows Authentication Group Policy browser settings

Apply browser settings using Group Policy.

The PingFederate Integrated Windows Authentication (IWA) adapter supports the Kerberos and NTLM authentication protocols, though some browsers must be configured to use them. These settings are configured manually on each computer, except in a Windows Enterprise environment, where they are managed by Group Policy.

Within Group Policy there are two configuration subsets available: policies and preferences. Policies and preferences are applied through Group Policy Objects (GPOs) for the following browser configurations:

- Internet Explorer
- Google Chrome
- Mozilla Firefox

Policies typically configure system-specific settings such as Windows operating system, security, and software settings. When the GPO falls out of scope, Group Policy Preference settings remain the same. In Policies, the defined settings supersede the local system or user settings. When they fall out of scope, the local settings revert to the previous settings.

Providing a persistent SAML NameID format in PingFederate

Use a custom SAML NameID format by defining a hidden attribute in the PingFederate attribute contract.

Before you begin

You must have the following product versions:

- PingFederate 10.3

About this task

Some SAML federation partner software requires a SAML NameID format of `urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`. Provide this format by using `SAML_NAME_FORMAT`.

Steps

1. In PingFederate, go to **Applications → SP Connections**.
2. In the **SP Connections** list, select your connection.
3. Click the **Browser SSO** tab, and then click **Configure Browser SSO**.
4. Click the **Assertion Creation** tab, and then click **Configure Assertion Creation**.
5. Click the **Attribute Contract** tab.
6. Extend the contract using the following table as a guide.

Attribute Contract	Subject Name Format
`SAML_SUBJECT`	**urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified**

Attribute Contract	Subject Name Format
`SAML_NAME_FORMAT`	**urn:oasis:names:tc:SAML:1.1:attrname-format:unspecified**

7. Click **Next**.
8. Click the **Authentication Source Mapping** tab and then click **Map New Adapter Instance**.
9. On the **Adapter Instance** tab, in the **Adapter Instance** list, select your adapter. Click **Next**.
10. On the **Mapping Method** tab, leave the default settings and click **Next**.
11. On the **Attribute Contract Fulfillment** tab, fulfill the contract using the following table as a guide.

Attribute Contract	Source	Value
`SAML_SUBJECT`	`Adapter`	`username`
`SAML_NAME_FORMAT`	`Text`	`urn:oasis:names:tc:SAML:2.0:nameid-format:persistent`

12. Click **Next** until you reach the **Summary** tab. Click **Save**.

Result

This produces a `SAML_SUBJECT` similar to the following example.

```
<saml:Subject>
<saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:persistent">joe</saml:NameID>
<saml:SubjectConfirmation Method="urn:oasis:names:tc:SAML:2.0:cm:bearer">
```

The new `SAML_NAME_FORMAT` value overrides the original SAML NameID.

Related links

- Assertions and Protocols for the OASIS Security Assertion Markup Language

Using OpenSSL s_client commands to test SSL connectivity

Test SSL connectivity with `s_client` commands to check whether the certificate is valid, trusted, and complete.

Before you begin

Install OpenSSL software from http://www.openssl.org/.

Steps

1. In the command line, enter `openssl s_client -connect` *<hostname>* `:<port>`.

   Result:

This opens an SSL connection to the specified hostname and port and prints the SSL certificate.

2. Check the availability of the domain from the connection results.

   The following table includes some commonly used **`s_client`** commands. For more information, see OpenSSL s_client commands man page in the OpenSSL toolkit.

   To view a complete list of **`s_client`** commands in the command line, enter `openssl -?`.

Command Options	Description	Example
`-connect`	Tests connectivity to an HTTPS service.	`openssl s_client -connect pingfederate.<YourDomain>.com:443`
`-showcerts`	Prints all certificates in the certificate chain presented by the SSL service. Useful when troubleshooting missing intermediate CA certificate issues.	`openssl s_client -connect <hostname>:<port> -showcerts`
`-tls, -dtls1`	Forces TLSv1 and DTLSv1 respectively.	`openssl s_client -connect <hostname>:<port> -tls1`
`-cipher`	Forces a specific cipher. This option is useful in testing enabled SSL ciphers. Use the **`openssl ciphers`** command to see a list of available ciphers for OpenSSL.	`openssl s_client -connect <hostname>:<port> -cipher DHE-RSA-AES256-SHA`

3. Troubleshooting:

   For troubleshooting connection and SSL handshake problems, see the following:

   - If there is a connection problem reaching the domain, the OpenSSL **`s_client -connect`** command waits until a timeout occurs and prints an error, such as `connect: Operation timed out`.
   - If you use the OpenSSL client to connect to a non-SSL service, the client connects but the SSL handshake doesn't happen. `CONNECTED (00000003)` prints as soon as a socket opens, but the client waits until a timeout occurs and prints an error message, such as `44356:error:140790E5:SSL routines:SSL23_WRITE:ssl handshake failure:/SourceCache/OpenSSL098/OpenSSL098-47.1/src/ssl/s23_lib.c:182:`.

     After disabling a weak cipher, you can verify if it has been disabled or not with the following command.

     ```
     openssl s_client -connect google.com:443 -cipher EXP-RC4-MD5
     ```

Related links

- SSL Labs

Customer Use Cases

Use case	Description
Authenticating with social media providers	You can authenticate using social media providers as an external identity provider (IdP) using PingOne and PingFederate
Customizing SSO user sign-on windows in PingFederate	You can customize the default single sign-on (SSO) end user sign-on window for PingFederate.
Enabling MFA for your application	Enable an authentication policy that includes multi-factor authentication (MFA) for your applications in PingOne.
Obtaining logging data from PingOne	Learn how to obtain logging data from PingOne.
Setting up an agent in PingAccess	Learn how to set up an agent integration for PingAccess applications.
Setting up an OIDC application in PingFederate	Create a new OAuth or OpenID Connect (OIDC) application in PingFederate.
Setting up and customizing sign-on windows in PingOne	Customize the sign-on window in PingOne to match your company's desired branding and themes.
Setting up password recovery in PingOne	Learn how to set up password recovery for an application using PingOne.
Setting up password reset in PingOne	Learn how to customize the user's sign-on experience by enabling self-service management, such as change password and password reset, in the PingFederate administrative console when using the company's HTML Form sign-on page.
Setting up PingDataSync between Active Directory and PingOne	Learn how to configure PingDataSync for Microsoft Active Directory (AD) to PingOne in a Windows environment.
Setting up PingDataSync between PingDirectory and PingOne	Learn how to set up PingDataSync between PingDirectory and PingOne using installation commands for Linux.

Authenticating with social media providers

You can authenticate using social media providers as an external identity provider (IdP) using PingOne and PingFederate.

Authenticating with social media providers using PingOne

Using an external IdP allows linked users to authenticate using the credentials provided by the external IdP. Learn more about using external IdPs with PingOne in Identity providers.

Learn more about how to add the following social media providers as an external IdP using PingOne:

- Amazon
- Apple
- Facebook
- Google
- LinkedIn
- Twitter
- Yahoo

Authenticating with social media providers using PingFederate

Ping Identity provides several login integration kits that allow PingFederate to coordinate single sign-on (SSO) by using third-party services as IdPs.

Component

PingFederate 10.3

Additional information

Learn more in the following topics:

- Amazon Login Integration Kit
- Apple Login Integration Kit
- Facebook Login Integration Kit
- Google Login Integration Kit
- LinkedIn Login Integration Kit
- Twitter Login Integration Kit

Customizing SSO user sign-on windows in PingFederate

You can customize the default single sign-on (SSO) end user sign-on window for PingFederate. Learn more about customizable user-facing pages in IdP user-facing pages in the PingFederate documentation.

Component

PingFederate 10.1 and later

Configuration instructions

Read the following sections for instructions specific to the configuration you are performing.

Formatting the default sign-on window template

Steps

1. To view the various HTML files that PingFederate presents to the end user, in your PingFederate server open the `<pf_install>/pingfederate/server/default/conf/template` folder.
2. Format the default sign-on window template.
    1. Open the `html.form.login.template.html`.

       This document contains the default sign-on window.
    2. Customize the content of the sign-on window.

       > **Note**
       >
       > The `html.form.login.template.html` file contains information in comments to assist you in customizing the content of the web page.
       > Remember to save a backup or copy of the file before editing.
3. Save the `html.form.login.template.html` file.
4. If you changed the name of the `html.form.login.template.html` file, you must update it in the PingFederate administrative console.
    1. In the PingFederate administrative console, go to **Authentication > Integration > IdP Adapters**.
    2. Click the adapter that you want to modify.
    3. On the **IdP Adapter** tab, scroll to the bottom and click **Show Advanced Fields**.
    4. In the **Login Template** field, enter the updated `html` login template.
    5. Save your changes.

Using CSS to customize SSO user sign-on windows in PingFederate

Edit the CSS in the `main.css` directory file to set up a custom background for your single sign-on (SSO) end user sign-on window in PingFederate.

About this task

If you are familiar with CSS, you can edit the `main.css` directory file to customize the background of your PingFederate SSO sign-on window as desired.

If you are running PingFederate in a cluster, you must make these changes on each runtime node.

Steps

1. Open the `<pf_install>/pingfederate/server/default/conf/template/css/main.css` file.
2. Edit the `main.css` file.

   > **Note**
   >
   > Remember to create a copy or backup of the `main.css` file in case you need to revert to the default settings.

   1. To locate the `html` rule set, open the file and search for `html {`.

      Example:

      ```
      html {
        height: 100%;
        background-color: #f7f7f7;
        background-repeat: no-repeat;
        background-attachment: fixed;
        background-color: #3d454d;
        background-color: rgba(61, 69, 77, 0.9) \9;
        /*IE9 hack */
        background-image: radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d 80%);
        background-image: -o-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d 80%);
        background-image: -ms-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d 80%);
        background-image: -moz-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d
      80%);
        background-image: -webkit-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d
      80%);
        -webkit-background-size: cover;
        -moz-background-size: cover;
        -o-background-size: cover;
        background-size: cover;
      }
      ```

   2. Edit the CSS as desired.
3. Save the `main.css` file.

Setting up a custom background for the SSO user sign-on window in PingFederate

Edit the `main.css` directory file to set up a custom background for your single sign-on (SSO) end user sign-on window in PingFederate.

About this task

This procedure is applicable for PingFederate 8.4.x and later.

If you are unfamiliar with CSS, you can use this workaround to edit the `main.css` file in the PingFederate directory server. Perform the following changes to specify the desired PingFederate server image file for your custom background.

If you are running PingFederate in a cluster, you must make these changes on each runtime node.

Steps

1. Select the image you want to use as a custom background and save it as `<pf_install>/pingfederate/server/default/conf/template/images/custombackground.png`.

   > **Note**
   >
   > To update the image later, repeat step 1 or update the filepath in the `main.css` file to point to a different image.

2. Open the `<pf_install>/pingfederate/server/default/conf/template/assets/css/main.css` file.

3. Edit the `main.css` file.

   > **Note**
   >
   > Remember to create a copy or backup of the `main.css` file in case you need to revert to the default settings.

   1. To locate the `html` rule set, open the file and search for `html {`.

      Example:

      ```
      html {
        height: 100%;
        background-color: #f7f7f7;
        background-repeat: no-repeat;
        background-attachment: fixed;
        background-color: #3d454d;
        background-color: rgba(61, 69, 77, 0.9) \9;
        /*IE9 hack */
        background-image: radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d 80%);
        background-image: -o-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d 80%);
        background-image: -ms-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d 80%);
        background-image: -moz-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d
      80%);
        background-image: -webkit-radial-gradient(40% 40%, circle cover, rgba(61, 69, 77, 0.9) 30%, #3d454d
      80%);
        -webkit-background-size: cover;
        -moz-background-size: cover;
        -o-background-size: cover;
        background-size: cover;
      }
      ```

   2. Replace the declaration block for the `html` rule set with the following declaration block.

```
html {
  height: 100%;
  background-color: #f7f7f7;
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-image: url(../images/custombackground.png);
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-position: center;
  background-size: cover;
}
```

4. Save the `main.css` file.

   Result:

   The background image points to the `<pf_install>/pingfederate/server/default/conf/template/images/custombackground.png` file.

Enabling MFA for your application

Enable an authentication policy that includes multi-factor authentication (MFA) for your applications in PingOne.

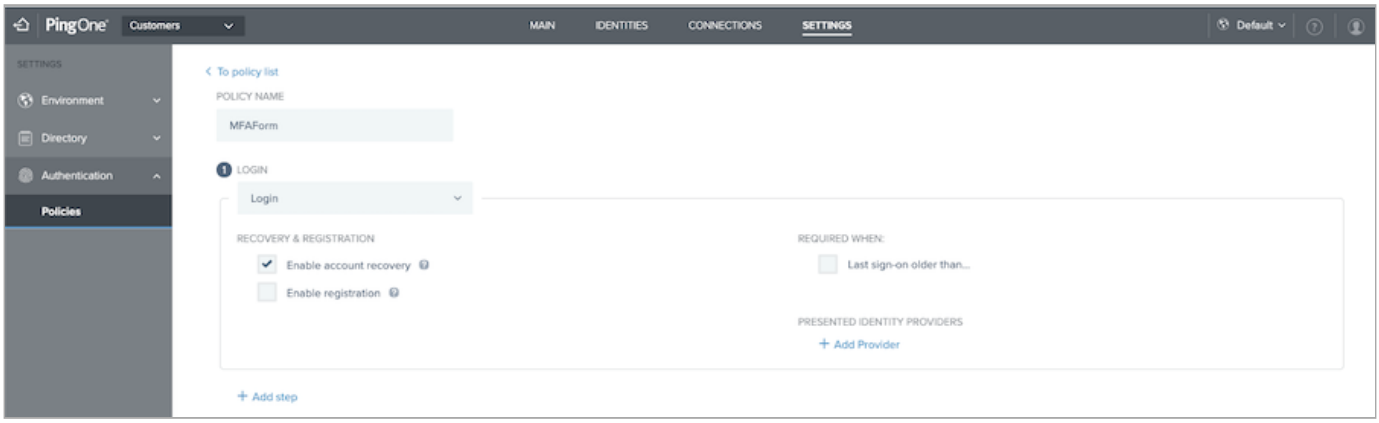
Before you begin

- Register for a PingOne tenant.
- To use a custom application for MFA, the application must already be configured.
- Configure an application connection.

Steps

1. In the PingOne dashboard, click **Settings**.
2. Go to **Authentication > Policies**, and then click **+Add Policy**.
3. Select an option for the first **Step Type**.

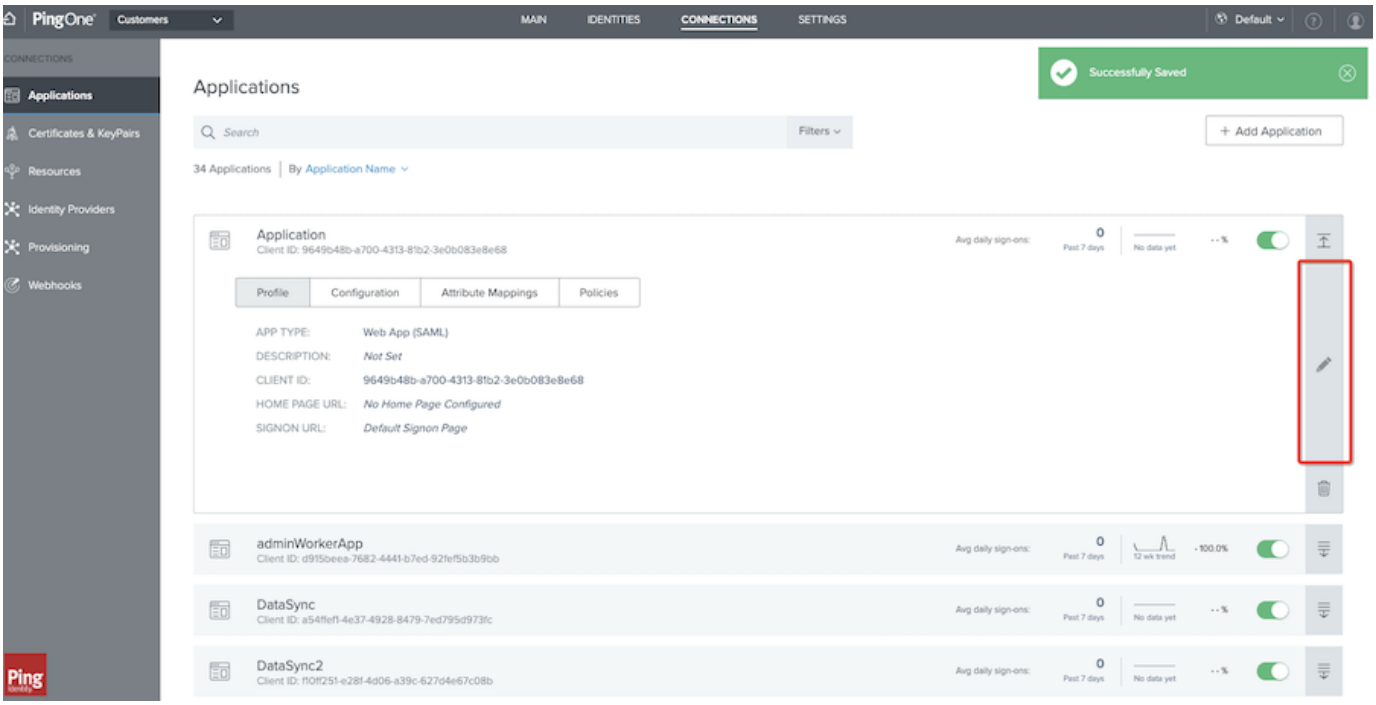
   There are multiple options depending on how you would like your user experience to be. For this example, **Login** was selected.



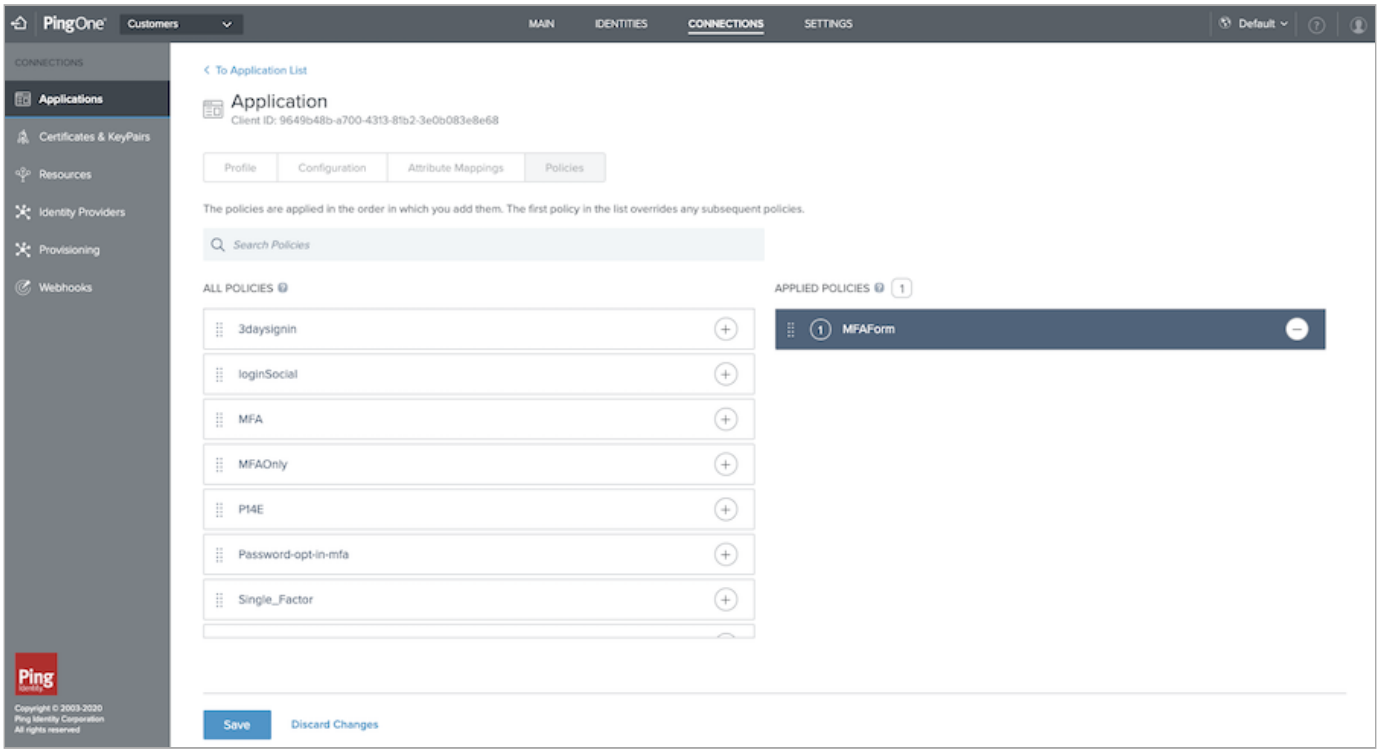
4. Click **+Add Step**, and then select **Multi-factor Authentication**.
5. Select the methods you want to enable for your users.

   If you have created a mobile application for MFA, you will have an option to select the appropriate application to associate to this login logic.
6. Select pertinent rules to be evaluated when a user is processed through this policy.

   Learn more about these options in Adding a multi-factor authentication step.
7. Review your selections, and then click **Save**.
8. Click **Connections**.
9. Select the desired connection you want to add your new policy to and click the **Pencil** (✎) icon.



10. Click the **Policies** tab.
11. Drag and drop your policy from the **All Policies** list to the **Applied Policies** list.

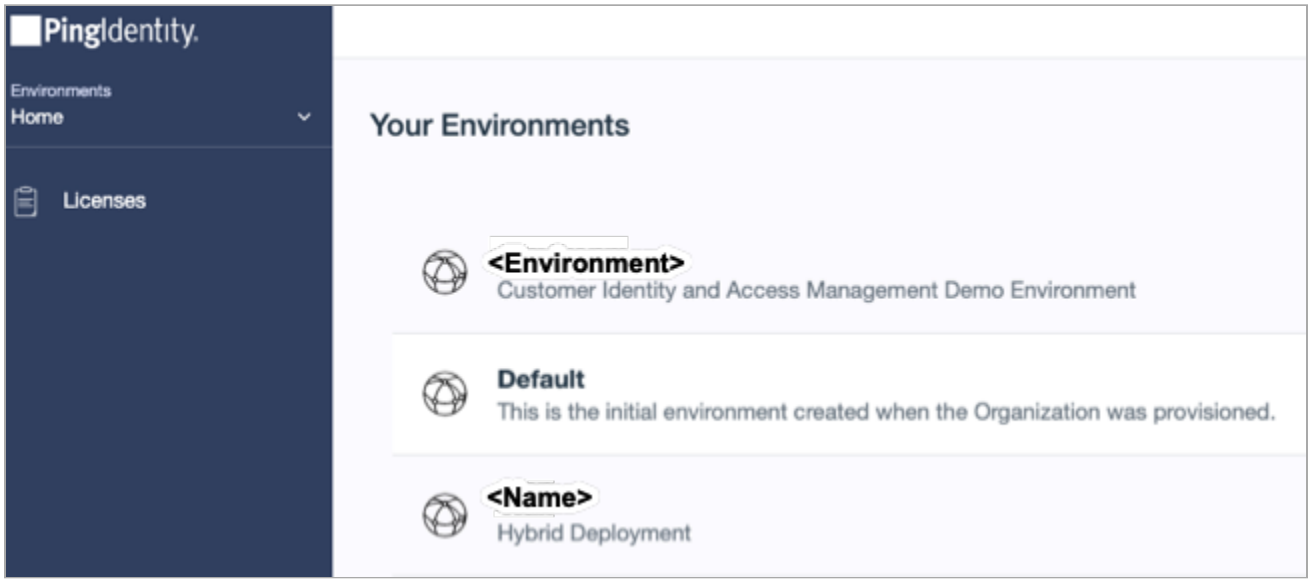


12. Click **Save**.

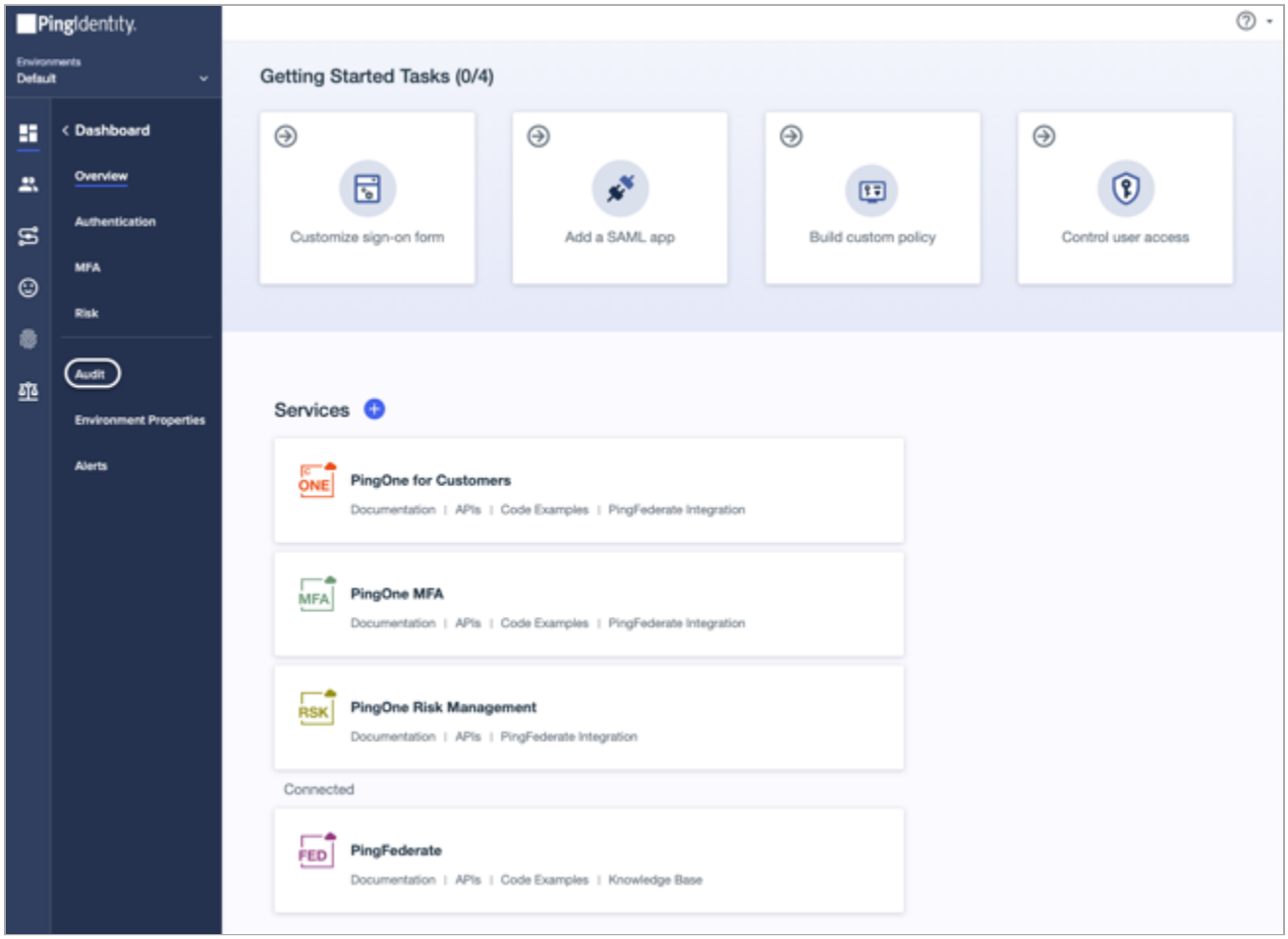
Obtaining logging data from PingOne

Steps

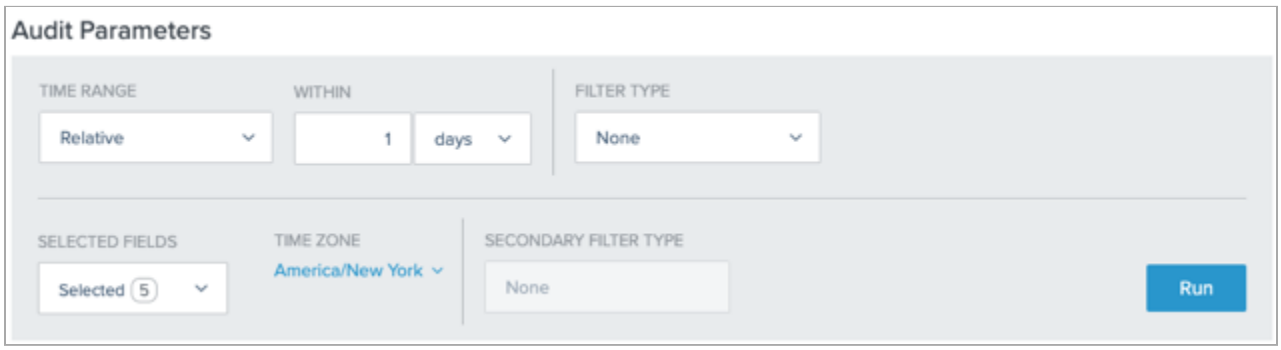
1. Sign on to PingOne and select your environment.



2. In the left navigation pane, click **Audit**.



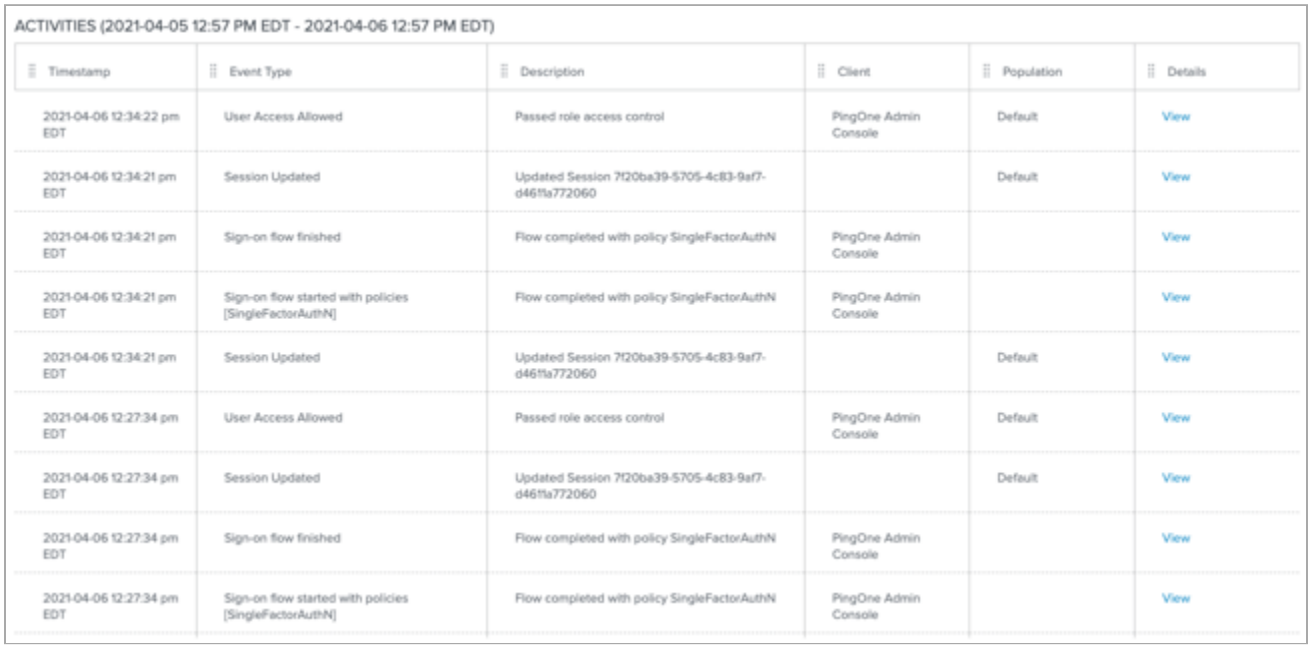
3. In the **Audit Parameters** section, adjust the audit parameters fields as needed.



4. To update the report, click **Run**.

   Result:

   The audit report is created.



ACTIVITIES (2021-04-05 12:57 PM EDT - 2021-04-06 12:57 PM EDT)

Timestamp	Event Type	Description	Client	Population	Details
2021-04-06 12:34:22 pm EDT	User Access Allowed	Passed role access control	PingOne Admin Console	Default	View
2021-04-06 12:34:21 pm EDT	Session Updated	Updated Session 7f20ba39-5705-4c83-9af7-d461fa772060		Default	View
2021-04-06 12:34:21 pm EDT	Sign-on flow finished	Flow completed with policy SingleFactorAuthN	PingOne Admin Console		View
2021-04-06 12:34:21 pm EDT	Sign-on flow started with policies [SingleFactorAuthN]	Flow completed with policy SingleFactorAuthN	PingOne Admin Console		View
2021-04-06 12:34:21 pm EDT	Session Updated	Updated Session 7f20ba39-5705-4c83-9af7-d461fa772060		Default	View
2021-04-06 12:27:34 pm EDT	User Access Allowed	Passed role access control	PingOne Admin Console	Default	View
2021-04-06 12:27:34 pm EDT	Session Updated	Updated Session 7f20ba39-5705-4c83-9af7-d461fa772060		Default	View
2021-04-06 12:27:34 pm EDT	Sign-on flow finished	Flow completed with policy SingleFactorAuthN	PingOne Admin Console		View
2021-04-06 12:27:34 pm EDT	Sign-on flow started with policies [SingleFactorAuthN]	Flow completed with policy SingleFactorAuthN	PingOne Admin Console		View

> **Note**
>
> If no fields are selected, the audit report only contains an empty **Details** column.

Example

The **Details** column contains a **View** link showing the JSON representation of the audit entry, as shown in the following example.

> **Note**
>
> All unique identifiers in this example are intentionally blocked.

```
{
 "_links": {
   "self": {
     "href": "https://api.pingone.com/v1/environments/429f5783-0f16-432f-b726-88223c380ab0/activities/979c1096-
a693-4920-a2c6-62e34ff74dfe"
   }
 },
 "id": "9*******-a***-4***-a***-6**********",
 "recordedAt": "2021-04-06T16:27:34.783Z",
 "createdAt": "2021-04-06T16:27:34.803Z",
 "correlationId": "f******-2***-4***-9***-6**********",
 "actors": {
   "client": {
     "id": "b*******-4***-4***-9***-0***********",
     "name": "PingOne Admin Console",
     "environment": {
       "id": "4*******-0***-4***-b***-8**********"
     },
     "href": "https://api.pingone.com/v1/environments/4*******-4*******-0***-4***-b***-8**********/applications/
b*******-4***-4***-9***-0**********",
     "type": "CLIENT"
   },
   "user": {
     "id": "7*******-5***-4***-9***-d**********",
     "name": "m******p********@pingidentity.com",
     "environment": {
       "id": "4*******-0***-4***-b***-8**********"
     },
     "population": {
       "id": "4*******-0***-4***-b***-8**********"
     },
     "href": "https://api.pingone.com/v1/environments/4*******-0***-4***-b***-8**********/users/
7*******-5***-4***-9***-d**********",
     "type": "USER"
   }
 },
 "action": {
   "type": "USER.ACCESS_ALLOWED",
   "description": "User Access Allowed"
 },
 "resources": [
   {
     "type": "USER",
     "id": "7*******-5***-4***-9***-d**********",
     "name": "matthewpollicove@pingidentity.com",
     "environment": {
       "id": "4*******-0***-4***-b***-8**********"
     },
     "population": {
       "id": "4*******-0***-4***-b***-8**********"
     },
     "href": "https://api.pingone.com/v1/environments/4*******-0***-4***-b***-8**********/users/
7*******-5***-4***-9***-d**********"
   }
 ],
 "result": {
```

```
    "status": "SUCCESS",

    "description": "Passed role access control"
  }
}
```

Next steps

If you need to connect the audit data to an external application, such as Splunk, learn more in Monitoring activity with Splunk.

Audit parameter fields

The following table describes the fields and options available in the **Audit Parameters** section of PingOne.

Field	Option	Description
Time Range	**Specific**	Set the time range to a specific date range.
	Relative	Set the time range to a period relative to the current time.
Filter Type **Note** You must select a **Filter Type** before the **Secondary Filter** field is available.	**Resource ID**	Find activities by resource ID.
	Correlation ID	Find activities by correlation ID. When an HTTP request is received by PingOne, it is assigned a correlation ID. You can use the correlation ID to associate HTTP responses with messages in the event log.
	Event type	Find activities by event type. Select an event type. If **Risk** is enabled, the following parameters are available: • **Risk Evaluation Created** • **Risk Evaluation Updated** • **Risk Policy Created** • **Risk Policy Deleted** • **Risk Policy Updated**
	User ID (Actor)	Find activities that were performed by a particular user by user ID.
	Username (Actor)	Find activities that were performed by a particular user by username.

Field	Option	Description
	Client (Actor)	Select a client to find activities that were performed by that client. The list of clients can vary depending on your configuration.
	Resource population	Select a population to find activities that were performed in resources within a particular population.
	Resource type	Select a resource to find activities that were performed on a particular type of resource.
	Population	Find activities that were performed on a particular population.
	User	Find activities that were performed on a particular user.
	Application	Find activities that were performed by a particular client application.
Selected Fields	**Timestamp**	The date and time of the event. The format is: MM/DD/YYYY HH:mm:ss.
	Event name	A unique identifier for the event.
	Description	A brief description of the event.
	Client	The client that performed the event.
	User identity	The user for which the event was performed.
	Population	The population in which the event was performed.
	Resource type	The type of resource for which the event was performed.

Setting up an agent in PingAccess

Learn how to set up an agent integration for PingAccess applications.

Component

- PingAccess 8.2
- PingAccess 6.3

Configuring an agent for PingAccess

Before you begin

- Install PingAccess with either a PingFederate, PingOne, or OpenID Connect (OIDC) token provider configured.

Steps

1. Sign on to the PingAccess admin console.
2. Go to **Applications > Agents** and click **+ Add Agents**:
    1. In the **Name** field, enter a name for the agent.
    2. In the **PingAccess Host** field, enter a host name and port number.
    3. To retrieve the `agent.properties` file to use during the agent installation process, click **Save & Download**.

    > **Note**
    >
    > In most deployments the host name and port must match the `agent.http.port` value in the PingAccess `run.properties` file (default `3030`).
    > The host name and port of the PingAccess server is where this agent sends requests.

3. Go to **Applications > Applications**.
4. To connect the newly created agent with a PingAccess application, click **+ Add Application**:
    1. In the **Name** field, enter an application name.
    2. In the **Context Root** field, enter an appropriate value.
    3. In the **Virtual Host(s)** section, click **+ Create**.
    4. Create a virtual host that matches the agent server's host and port values that users will access, then click **Save**.
5. In the `Web Session` section, click **+ Create** to create a new web session:
    1. In the **Name** field, enter a name for the web session.
    2. In the **Audience** field, enter the names of the applications using this web session.
    3. In the **Client ID** and **Client Secret** fields, enter the OIDC Login Type values from the PingAccess token provider.
    4. Click **Save**.

    Result:

    You return to the **New Application** page.

6. In the **Web Identity Mapping** section, click **+ Create** to create a new mapping:
    1. In the **Name** field enter an identity mapping name.
    2. In the **Type** list, select **Header Identity Mapping**.
    3. In the **Attributes** section, click **Exclusion List**.
    4. In the **Header Name Prefix** field, enter a prefix pattern the agent application will expect.

       Example:

       If your user was Ping, your prefix header for the username field would be "ping-" and then it would say ping-username.
    5. In the **Attribute to Header Mapping** section, click **+ Add Row**.
    6. In the **Subject Attribute Name** list, select the attribute that corresponds to the user's subject value.
    7. In the **Header Name** field, enter a header name.
    8. Click **Save**.

       Result:

       You return to the **New Application** configuration page.
7. In the **Destination** list, select **Agent**.
8. In the **Agent** list, select the agent you created earlier.
9. Select the **Enabled** checkbox and click **Save**.

Configuring a token provider

Steps

- For the PingAccess token provider that you're using, add the virtual host's redirect URI to the OAuth client selected for the web session of the created application.

  Example:

  https://*<virtualhostname>*:*<virtualhostnameport>*/pa/oidc/cb

Installing the agent

Steps

1. Download the appropriate agent installation file from the PingAccess Add-Ons Downloads page.
2. Configure the PingAccess agent installation.

   You can find installation instructions for each agent type in PingAccess Agents and Integrations (page 362).
3. Copy the `agent.properties` file that you downloaded previously into the PingAccess agent installation directory.

> **Note**
>
> The properties file must be named `agent.properties`.

4. To deploy the new PingAccess agent configuration to the desired resources, restart the web or API service that you just installed the agent on.

Setting up an OIDC application in PingFederate

Create a new OAuth or OpenID Connect (OIDC) application in PingFederate.

Before you begin

Component

- PingFederate 10.1

Do the following:

- Install PingFederate.
- Configure an access token manager (page 550).
- Configure an OIDC policy (page 572).
- Map authentication sources to persistent grants (page 528).
- Map persistent grants to access token attribute contract (page 549).

Steps

1. In the PingFederate administrative console, go to **Applications > OAuth > Clients**, and click **Add Client**.
2. In the **Client ID** field, provide a client ID.

   The client ID is a unique identifier and cannot have the same ID as another OAuth client.

   > **Note**
   >
   > You cannot change a client ID after it is set.

3. In the **Name** field, enter a name for the client.

   The name value is a descriptive name displayed for end users that indicates the purpose of the client.

4. In the **Description** field, provide a description that gives additional detail on the use of the client.
5. Select a **Client Authentication** method:

   Choose from:

   - **None**

- **Client TLS Certificate**

    1. In the **Issuer** list, select the certificate for a trusted issuer if the client should expect certificates from a specific issuer. If certificates from any issuer should be allowed, select **Trust Any**.
    2. In the **Subject DN** field, enter the subject DN for the certificate or extract it from a file by clicking **Choose File**, selecting an appropriate certificate, and then clicking **Extract**.

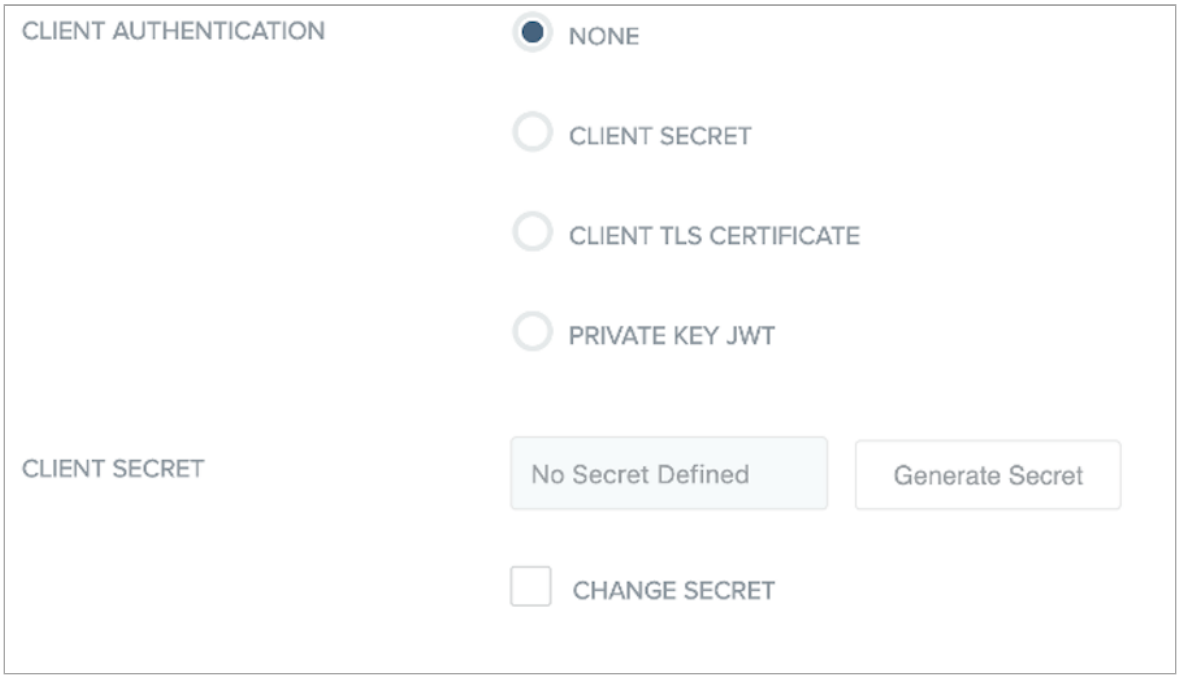
- **Private Key JWT**

    1. Select the **Replay Prevention** checkbox if the client should require a unique JSON web token (JWT) for each request.
    2. From the **Signing Algorithm** list, select the specific algorithm for the incoming JWT, or to allow any supported signing algorithm to be used, select **Allow Any**.

- **Client Secret**

    > **Note**
    >
    > If selected as the authentication method, you must provide a client secret. If you select another method for authentication, you don't need a client secret.

    1. Select the **Change Secret** checkbox.
    2. In the **Client Secret** field, enter a client secret, or to have a random value provided, click **Generate** secret. Make a copy of this value because it won't be visible after the client is saved.



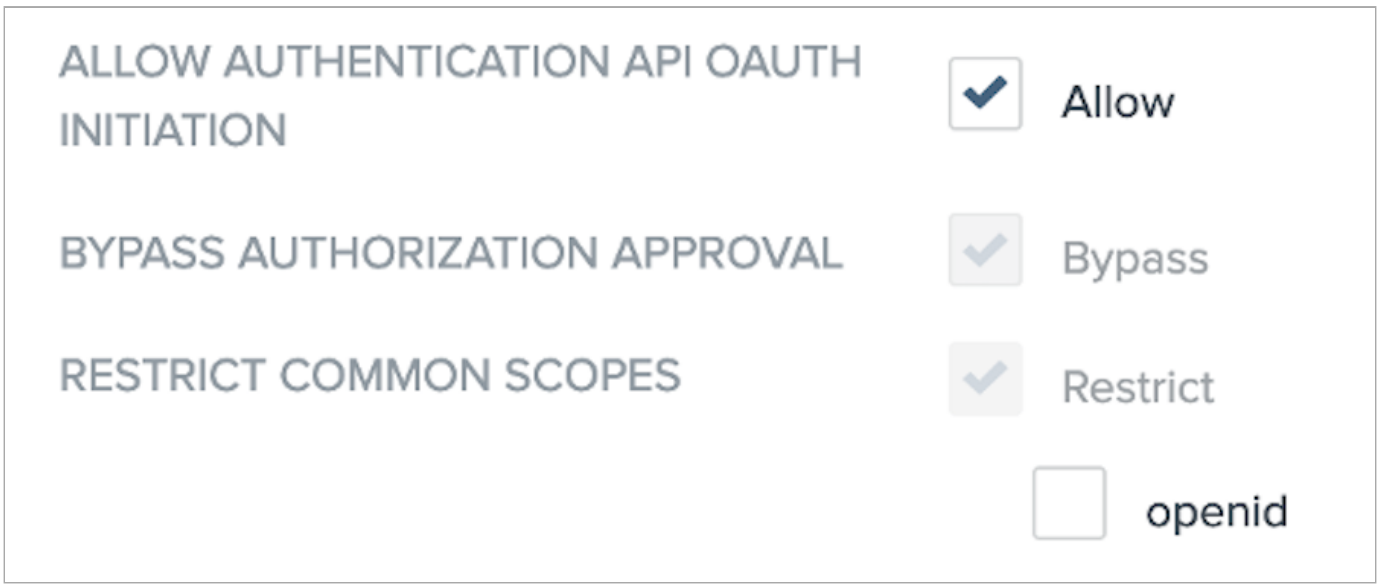
6. If the client requires all requests to be signed, select the **Require Signed Requests** checkbox.
7. If you selected **Require Signed Requests** checkbox, select the expected signing algorithm or select **Allow Any**.

8. If you selected **Private Key JWT** as the authentication method, or if you selected the **Require Signed Requests** checkbox, complete either the **JWKS URL** or **JWKS** fields:
    1. To use a JSON web key set (JWKS) URL, enter the URL of the JWKS.
    2. To provide the JWKS directly, copy and paste the contents of the JWKS in the **JWKS** field.
9. If the client will be configured to support the OAuth authorization code or implicit flows, in the **Redirection URI** section, enter at least one URI.

    The redirection URI values specify the valid redirection locations that an application might request post authorization. The redirection URI value must be a fully qualified URL. Wildcards can be used to allow redirection into any sub-path. Make redirection URIs as restrictive as possible.
    1. In the **Redirection URI** field, enter a value.
    2. Click **Add**.
    3. Repeat steps 9a and 9b for each valid redirection URI.
10. In the **Logo URL** field, enter a fully qualified URL.

    This is the URL for the logo image that displays on the **User Grant Authorization** and **Revocation** pages.
11. If the client will use the PingFederate authentication API for authentication and will require an experience that doesn't use HTTP redirections, select the **Allow Authentication API OAuth Initiation** checkbox.

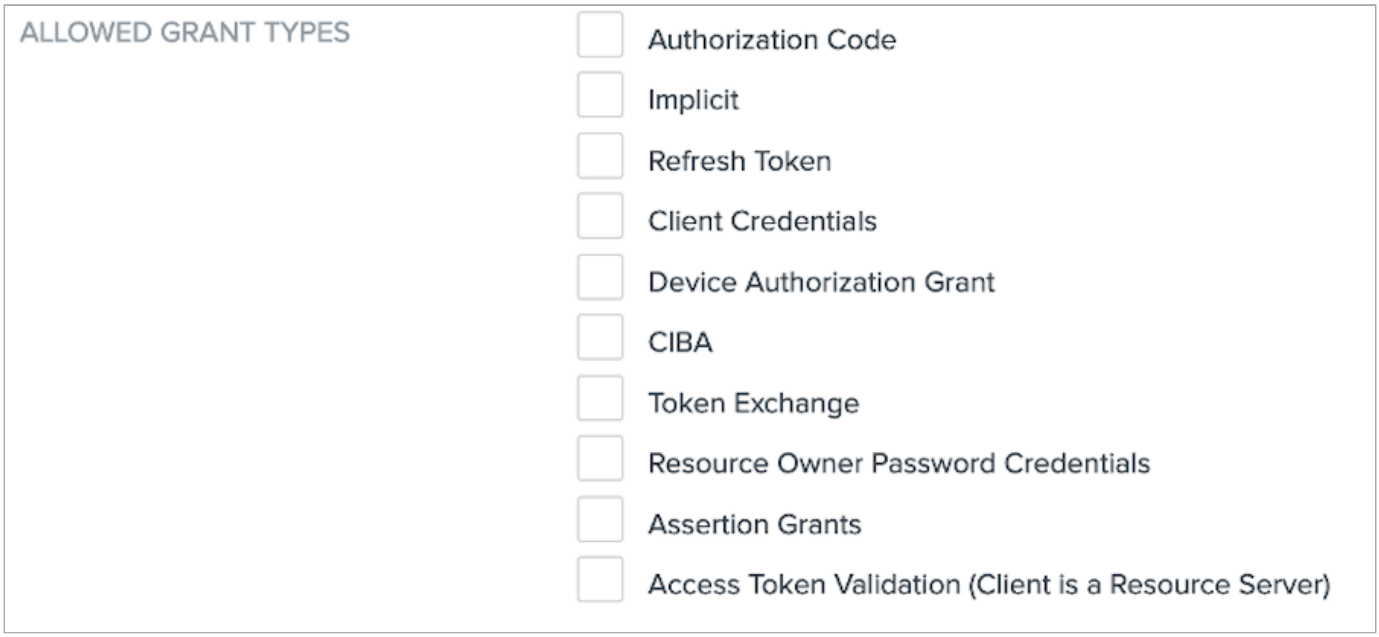
    Note
    - The **Bypass Authorization Approval** checkbox is automatically selected and can't be changed because this flow doesn't support the user-facing grant consent page.
    - The **Restrict Common Scopes** checkbox is automatically selected and can't be changed because this flow doesn't support the user-facing grant consent page.
    - Any configured **Common Scopes** and the default openID scope is displayed and can be selected as valid scopes for the client.



12. If any exclusive scopes are defined and the client should be allowed to use them, select the **Exclusive Scopes** checkbox.

Any defined exclusive scopes are displayed and can be selected as available to this client.

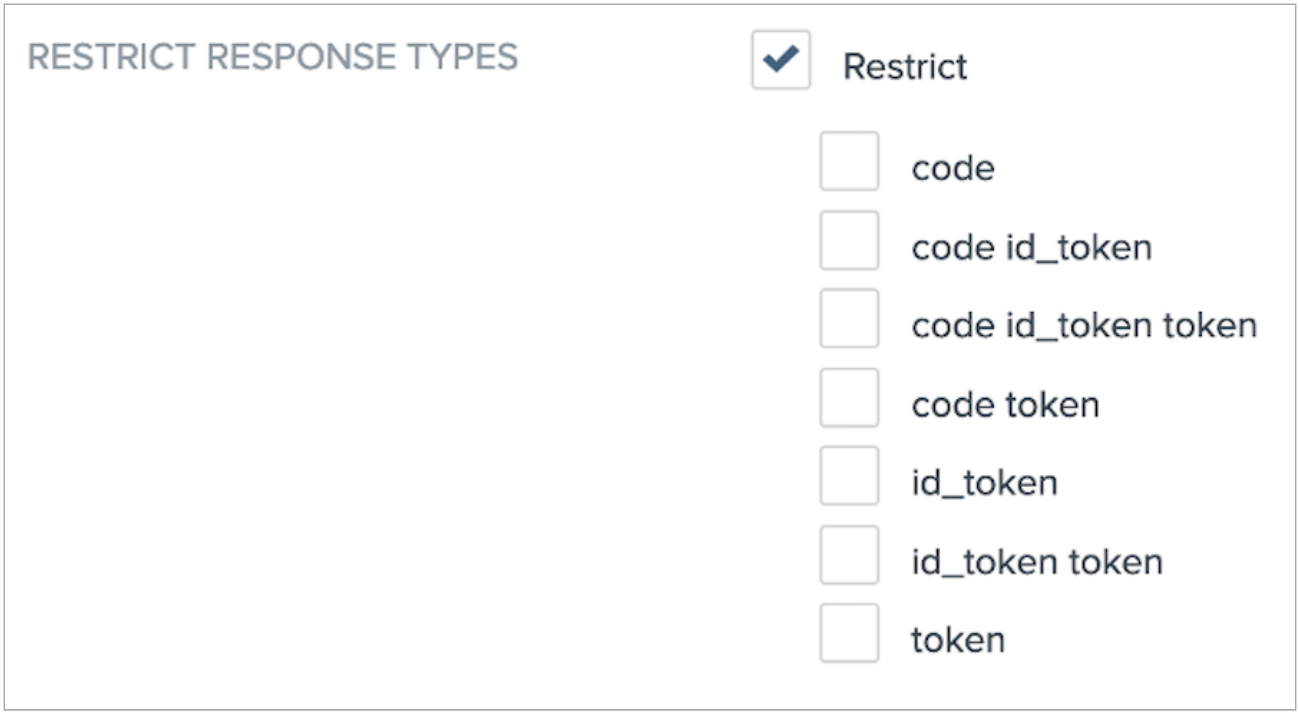
13. In the **Allowed Grant Types** list, select at least one option.



Learn more about each grant type in Grant Types.

14. If the client should restrict the available response types requested by the application, select the **Restrict Response Types** checkbox.

    1. In the list of available response types, select at least one option.



15. In the **Default Access Token Manager** list, select the default access token manager (ATM) for this client.

The Default ATM can either be the default ATM configured in **Access Token Mappings** or a specific ATM.

16. **Optional:** For resource server clients that won't receive a specific request for an ATM, select the **Validate Against All Eligible Access Token Managers** checkbox to instruct PingFederate to validate access tokens against all available ATMs.
17. If the client should require PKCE, select the **Require Proof Key for Code Exchange (PKCE)** checkbox.

> **Note**
>
> This checkbox isn't displayed unless **Authorization Code** is selected under **Allowed Grant Types**.

18. Override the global setting for the **Persistent Grants Max Lifetime** set in **System > OAuth Settings > Authorization Server Settings**:

    Choose from:

    - To use the global setting (default), click **Use Global Setting**.
    - For grants that should not have an expiration, click **Grants Do Not Expire**.
    - To enter a custom duration for this client's grants, click the radio button below **Grants Do Not Expire**.

19. Override the global setting for the **Refresh Token Rolling Policy** set in **System > OAuth Settings > Authorization Server Settings**.

    The client can override the global setting.

    Choose from:

    - To use the global setting (default), click **Use Global Setting**.
    - To tell the client to never roll the refresh token, click **Don't Roll**.
    - To tell the client to roll the refresh token, click **Roll**.

20. Change the default option for the token signing algorithm by selecting a different value in the **ID Token Signing Algorithm**list.
21. If the content of the ID token should be encrypted, then in the **ID Token Key Management Encryption Algorithm** list, select the algorithm that will be used to encrypt the content encryption key.
22. (Optional) If an **ID Token Key Management Encryption Algorithm** is selected, then in the **ID Token Content Encryption Algorithm** list, select the value of the encryption algorithm used to encrypt the plaintext content of the ID token.
23. To have the client use a specific OIDC policy, in the **Policy** list, select a specific value.

    By default, the client uses the OIDC policy configured as default in **Applications > OAuth > OpenID Connect Policy Management**.

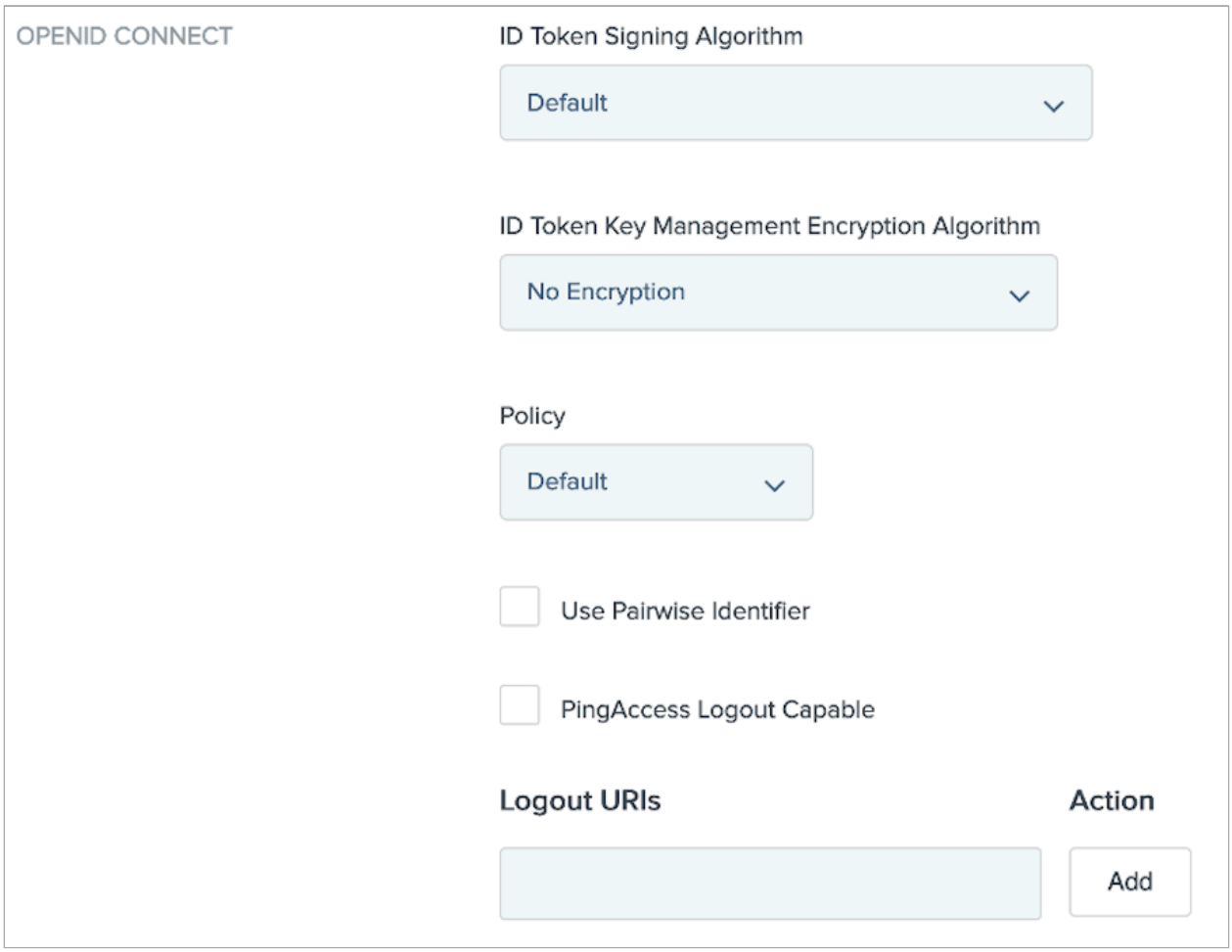
24. To enable pairwise pseudonymous identifiers for open banking support, select the **Use Pairwise Identifier** checkbox.
    - In the **Sector Identifier URI** field, enter a single HTTPS URI.
25. To send logout requests to an OIDC endpoint in PingAccess as part of the logout process, select the **PingAccess Logout Capable** checkbox.

> **Note**
>
> This checkbox only displays if the **Track User Sessions for Logout** option is selected in **System > OAuth Settings > Authorization Server Settings**.

26. Enter a fully qualified URI in the **Logout URIs** field and click **Add** for each endpoint desired.

> **Note**
>
> The **Logout URIs** field is displayed only if the **Track User Sessions for Logout** option is selected in **System > OAuth Settings > Authorization Server Settings**.

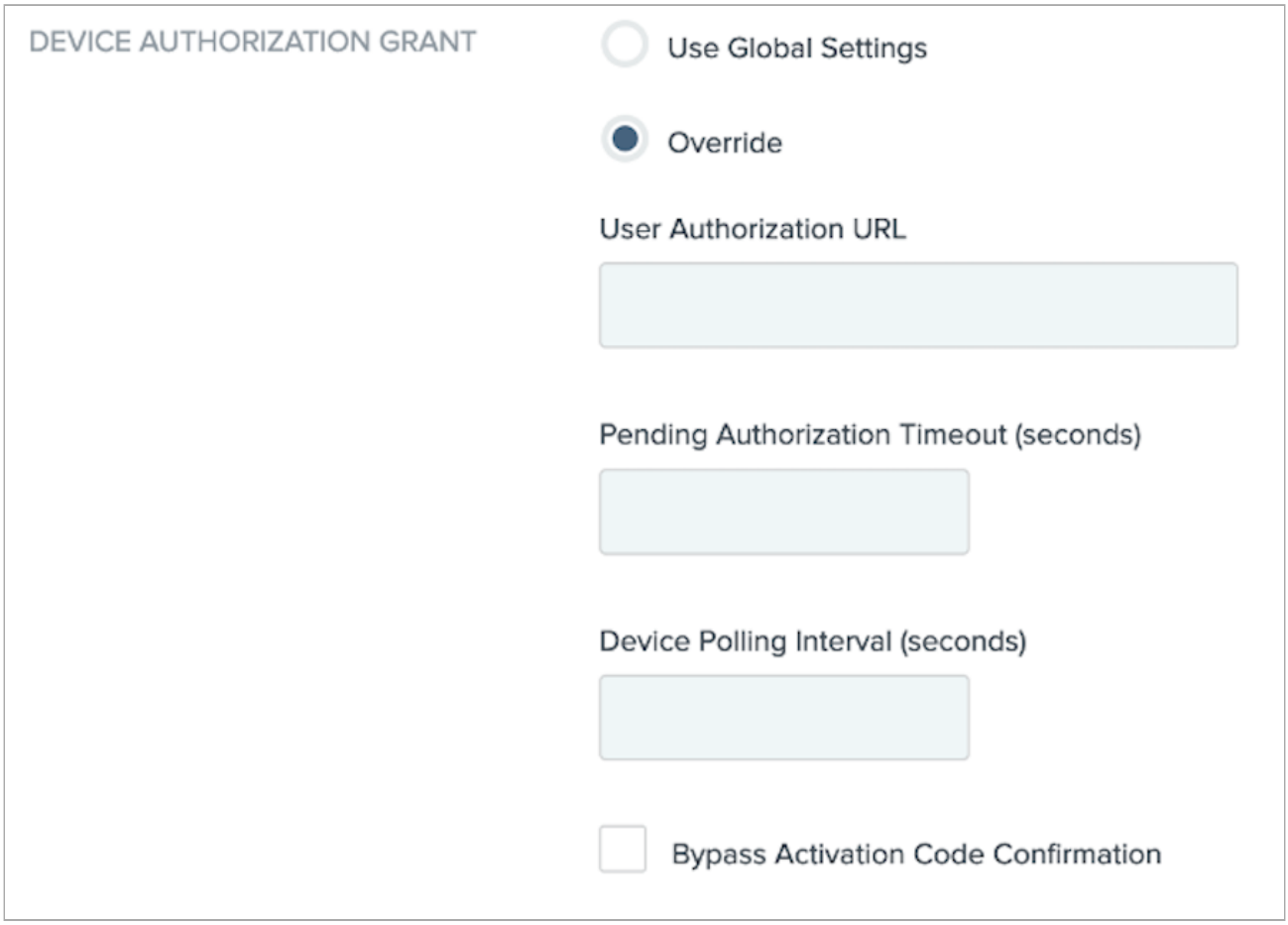
PingFederate sends logout requests to each relying party listed.



27. To allow the client to use the back-channel session revocation API, select the **Allow Access to Session Revocation API** checkbox.
28. To allow the client to use the session management API, select the **Allow Access to Session Management API** checkbox.
29. Override the global values set by clicking **Override** in **System > OAuth Settings > Authorization Server Settings**.

> **Note**
>
> The **Device Authorization Grant** settings section is only displayed If **Device Authorization Grant** is selected in **Allowed Grant Types**.

1. To allow the client to override the global value for the user authorization URL, in the **User Authorization URL** field, enter a fully qualified URL.
2. To allow the client to override the global value for an activation code, in the **Pending Authorization Timeout (seconds)** field, specify a timeout value.
3. To allow the client to override the global value for the device's polling interval to the PingFederate token endpoint, in the **Device Polling Interval (seconds)** field, enter an interval value.
4. To allow the client to override the global setting for bypassing the confirmation of an activation code, select the Bypass **Activation Code Confirmation** checkbox.



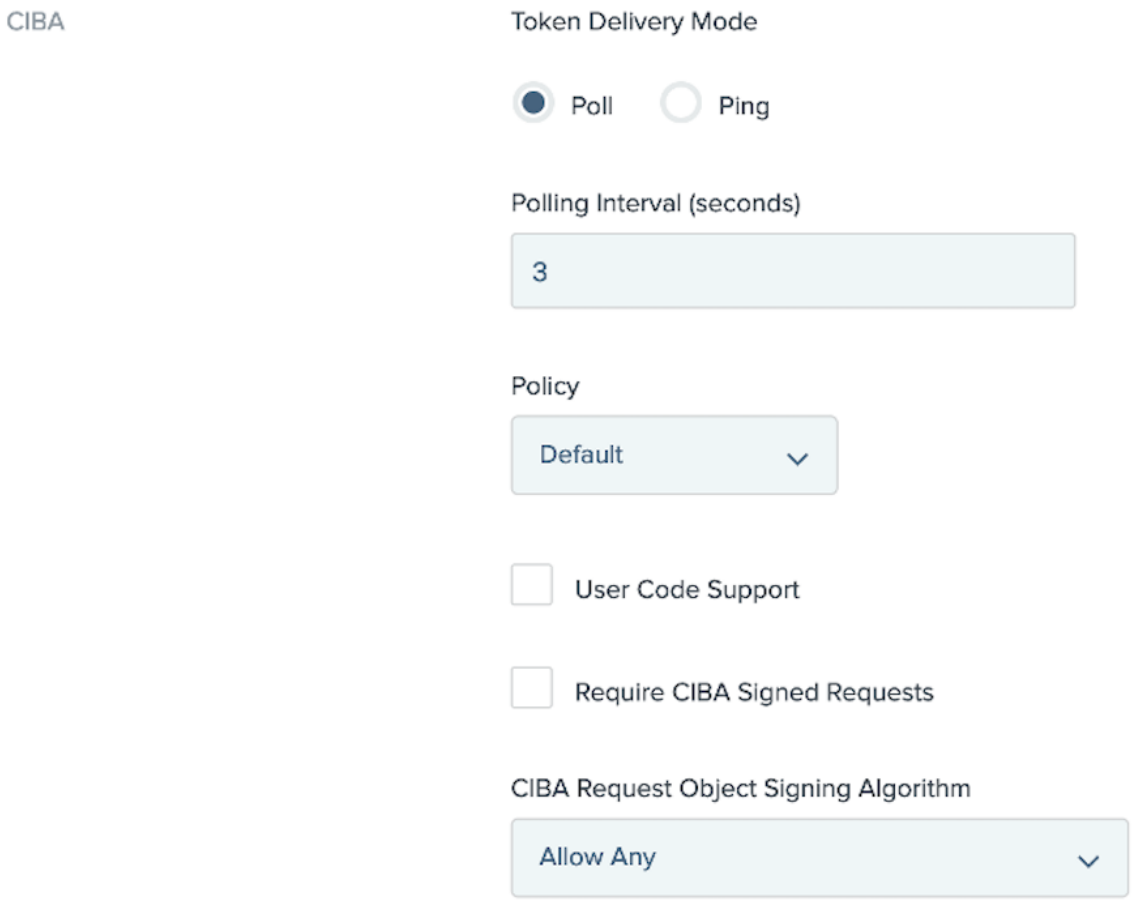
30. Configure the following CIBA settings:

> **Note**
>
> The **CIBA** section is displayed only if **CIBA** is selected in **Allowed Grant Types**.

1. In the **Token Delivery Method** field, specify token delivery method for the client:
   - If the client can check authorization results at the token endpoint, select **Poll**.
   - If the client expects a callback when authorization results are available, select **Ping**.
2. If you selected **Ping**, in the **Notification Endpoint** field that displays, enter the client notification endpoint for PingFederate to provide call back messages for this client.
3. If you selected **Poll**, to override the default polling interval, in the **Polling Interval** field, specify a value..

   The default polling interval for **Poll** clients is 3 seconds.
4. To override the default CIBA request policy configured in **Applications > OAuth > CIBA > Request Policies**, in the **Policy** list, select a different request policy.
5. To enable support for the user code in the client, select the **User Code Support** checkbox.
6. To require CIBA signed requests, select the **Require CIBA Signed Requests**checkboxx.
7. To select a specific algorithm, in the **CIBA Request Object Signing Algorithm** list, select a specific value.

   By default, the client supports any signing algorithm for the CIBA request object.



31. To override the default processor policy configured in **Applications** > **Processor Policies**, in the **Processor Policy** list, select a specific process policy.

    If **Token Exchange**is selected in **Allowed Grant Types**, the **Token Exchange** section is displayed.

32. If any extended properties are defined in **System** > **Extended Properties**, click **Next** to continue to the **Extended Properties** options for the client.

33. Click **Save**.

Setting up and customizing sign-on windows in PingOne

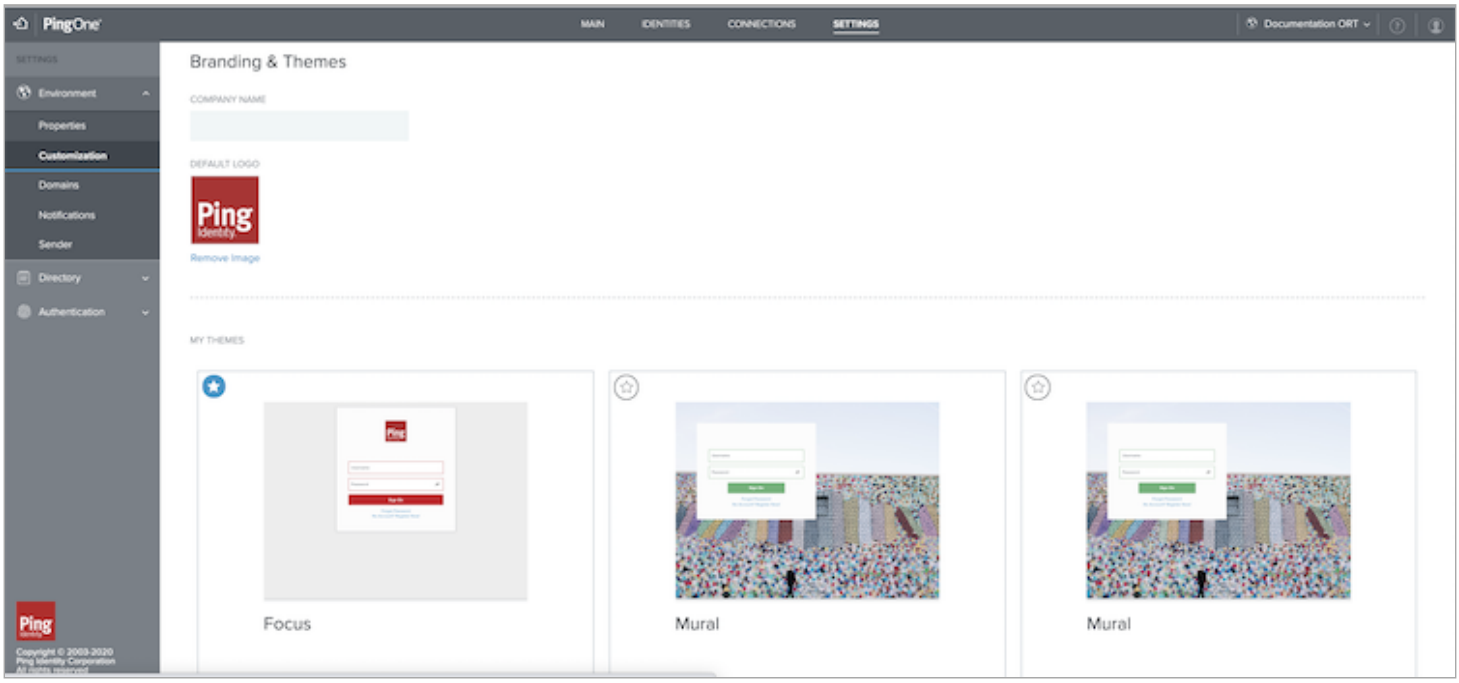
Customize the sign-on window in PingOne to match your company's desired branding and themes.

Before you begin

Collect any logos, background images, and specific color Hex values to match your company branding.

About this task

The **Branding & Themes** window provides the ability to customize logos, colors, and layout of all end user-facing pages.



Steps

1. In the PingOne admin console, go to **Settings > Customization**.
2. In the **Company Name** field, enter a company name.
3. Upload a desired image for your logo.

   If there is already a default logo, click **Remove Image** before uploading a new logo.

   1. In the **Default Logo** field, click the **Plus** (+) icon to select an image file.
   2. Select the desired image and click **Open**.

      The max size is 2.0 MB. Your image file must be a JPEG, JPG, GIF, or PNG.
4. For additional customization, change the theme.

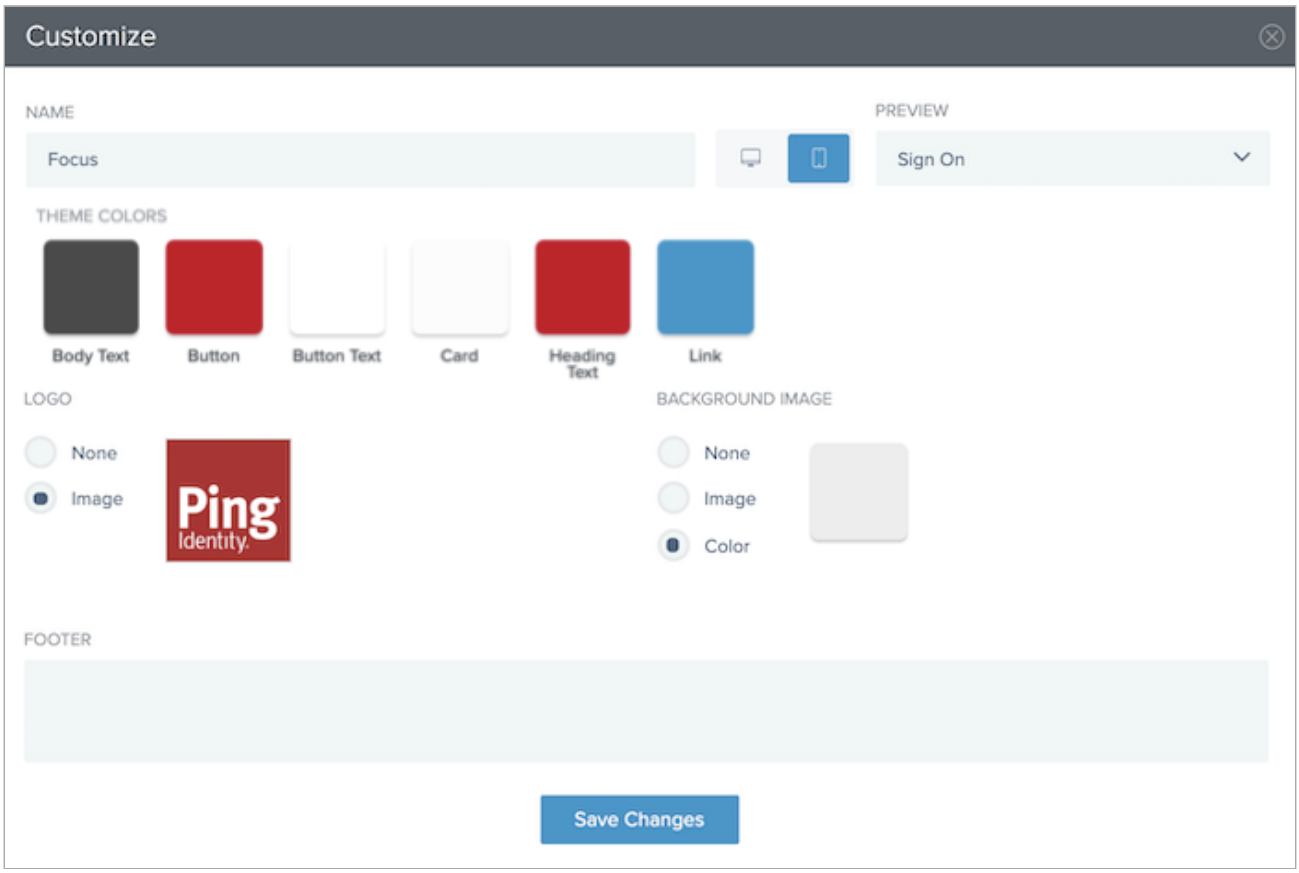
   1. In the **My Themes** section, click the **Plus** (+) icon.
   2. Click to select the desired theme from the available options.
   3. Use the editor to further customize the appearance.

      > **Note**
      >
      > All customization changes will appear in real-time.
5. To open the editor and customize the theme, click the desired theme.

   Choose from:

   - In the **Name** field, enter a new name.
   - From the **Preview** list, select a different preview display.

- In the **Theme Colors** field, select a new color for the theme.
- In the **Logo** field, remove or add a new image logo.
- In the **Background Image** field, remove, update, or change the color of the background image.
- In the **Footer** field, enter a footer.

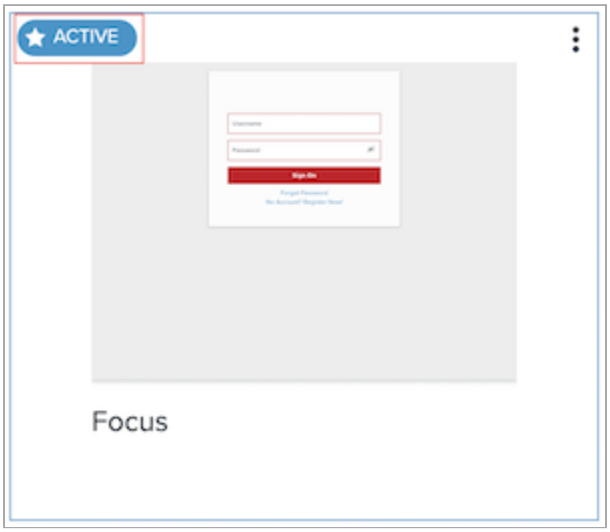


6. To toggle between the desktop and mobile view, click the **Desktop** (🖵) or **Mobile** (📱) icon.

7. To confirm your changes, click **Save Changes**.

> **Note**
>
> Add multiple themes as needed.

8. To activate a theme, click the **Star** (★) icon.



Setting up password recovery in PingOne

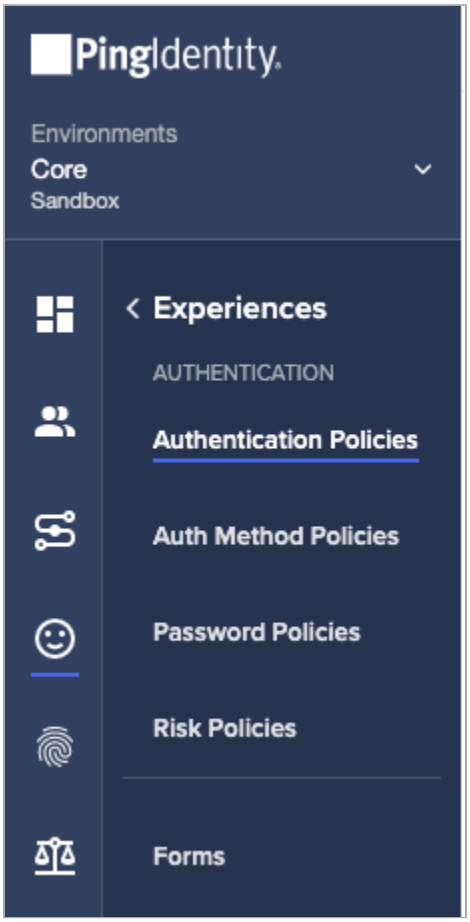
Learn how to set up password recovery for an application using PingOne.

About this task

Set up account recovery by modifying an application's **Single_Factor** policy in the PingOne administrative console or Unified Product Administrator.

Steps

1. Sign on to PingOne and go to **Experiences > Authentication Policies**.

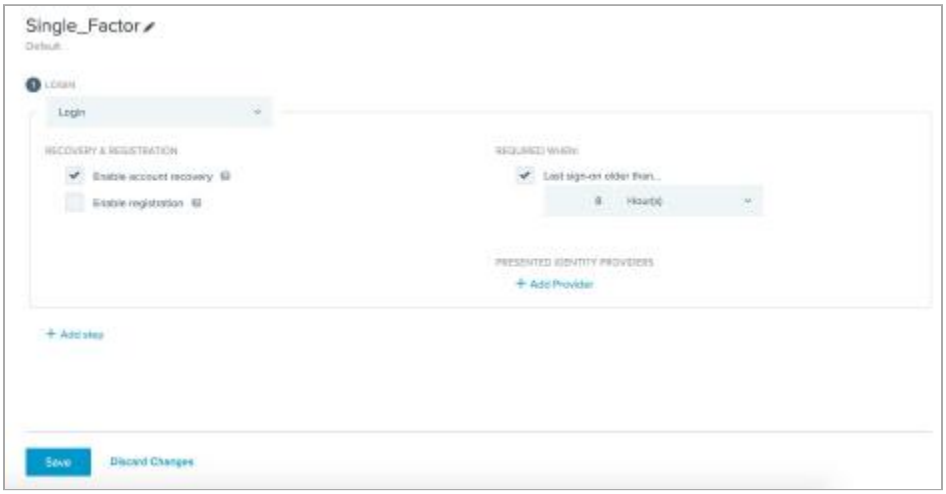


Result:

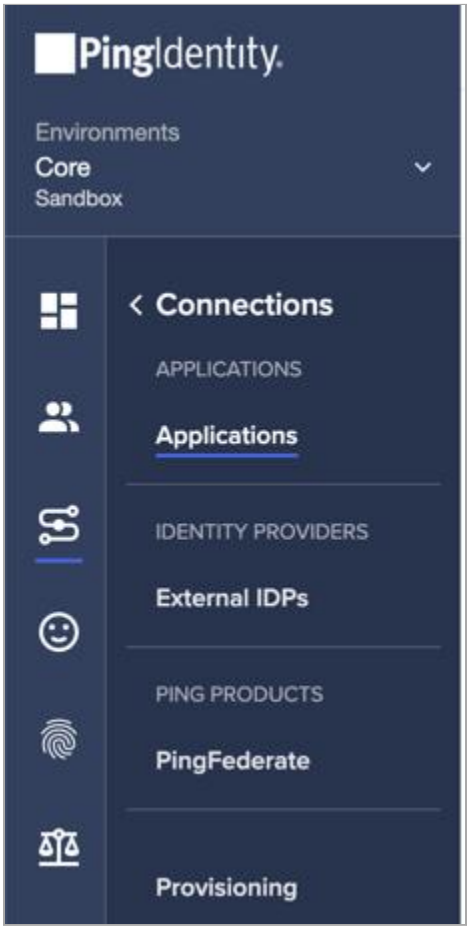
The **Policies** page lists your default **Single_Factor** and **Multi_Factor** policies.

2. To modify the **Single_Factor** policy, click the **Expand** icon and then click the **Pencil** (✎) icon.

3. In the **Recovery & Registration** section, select the **Enable account recovery** checkbox. Click **Save**.



4. To view the connection policies of an application, go to **Connections** > **Applications**.



Result:

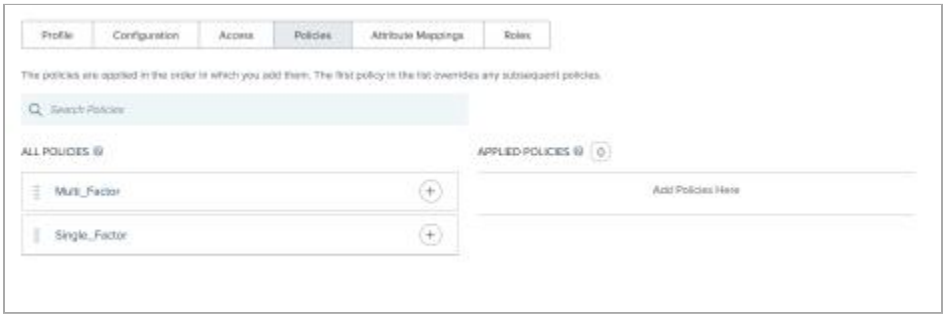
The **Applications** page lists all of your existing applications.

5. Locate the application that contains the connection policy you want to modify, and click the **Expand** icon.
6. To modify the application's policies, click the **Policies** tab and then click the **Pencil** (✎) icon.

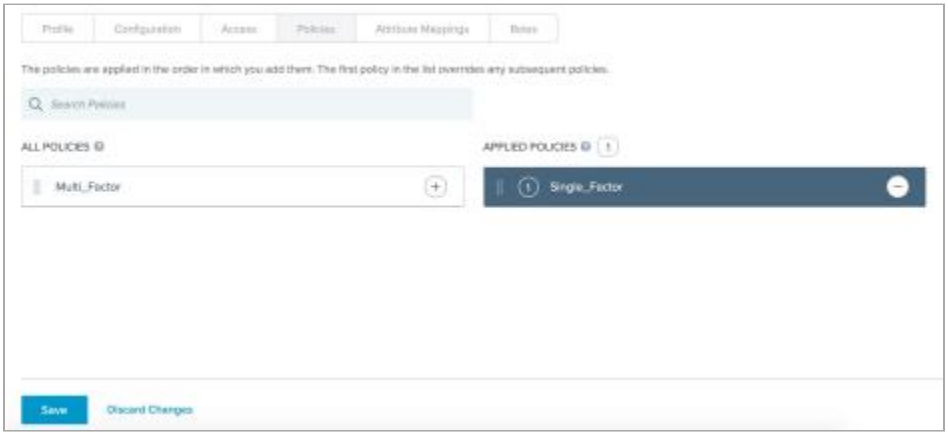
7. To add the **Single_Factor** policy to **Applied Policies**, choose one of the following ways:

   Choose from:

   - Drag the **Single_Factor** policy from **All Policies** to **Applied Policies**.
   - Click the **+** icon in the **Single_Factor** policy.



8. Click **Save**.



Setting up password reset in PingOne

Learn how to customize the user's sign-on experience by enabling self-service management, such as change password and password reset, in the PingFederate administrative console when using the company's HTML Form sign-on page.

Components

- PingOne
- PingFederate 10.1

Overview of changing and resetting passwords

The change password capability is helpful when a user knows their password and wants to change it. The password reset capability is helpful when a user forgets their password and wants to use another factor, such as PingDirectory, to authenticate and change their password. This guide covers how to successfully configure password reset and enable change password in the HTML Form Adapter and password credential validator (PCV) framework in PingFederate. PingFederate provides the following password reset methods for self-service password reset:

- Email one-time link
- Email one-time passcode
- Text message
- PingID

Each method requires additional configuration.

> **Note**
>
> Self-service password reset using the authentication policy method in PingFederate isn't covered in this topic. Learn more about the authentication policy method and configuration steps in Configuring self-service account recovery (page 643) in the PingFederate Server documentation.

Before you begin

1. Create an LDAP datastore source connection in PingFederate using LDAPS.
2. Create a service provider (SP) connection in PingFederate.
3. Add PingFederate as an identity provider (IdP) to PingOne and configure PingID.
4. Create an HTML Form Adapter and PingID IdP adapter in PingFederate.
5. Create a PCV in PingFederate.

Setting up an LDAPS datastore connection in PingFederate

About this task

The self-service password reset capability relies on the LDAP connection to your directory server and the Username PCV to query the required attributes for the chosen reset method.

PingFederate supports the following datastores:

- PingDirectory
- Microsoft Active Directory
- Oracle Unified Directory
- Oracle Directory Server out-of-the-box

> **Note**
>
> This task covers specific configuration settings for this use case. Learn more in Configuring an LDAP Connection (page 871).

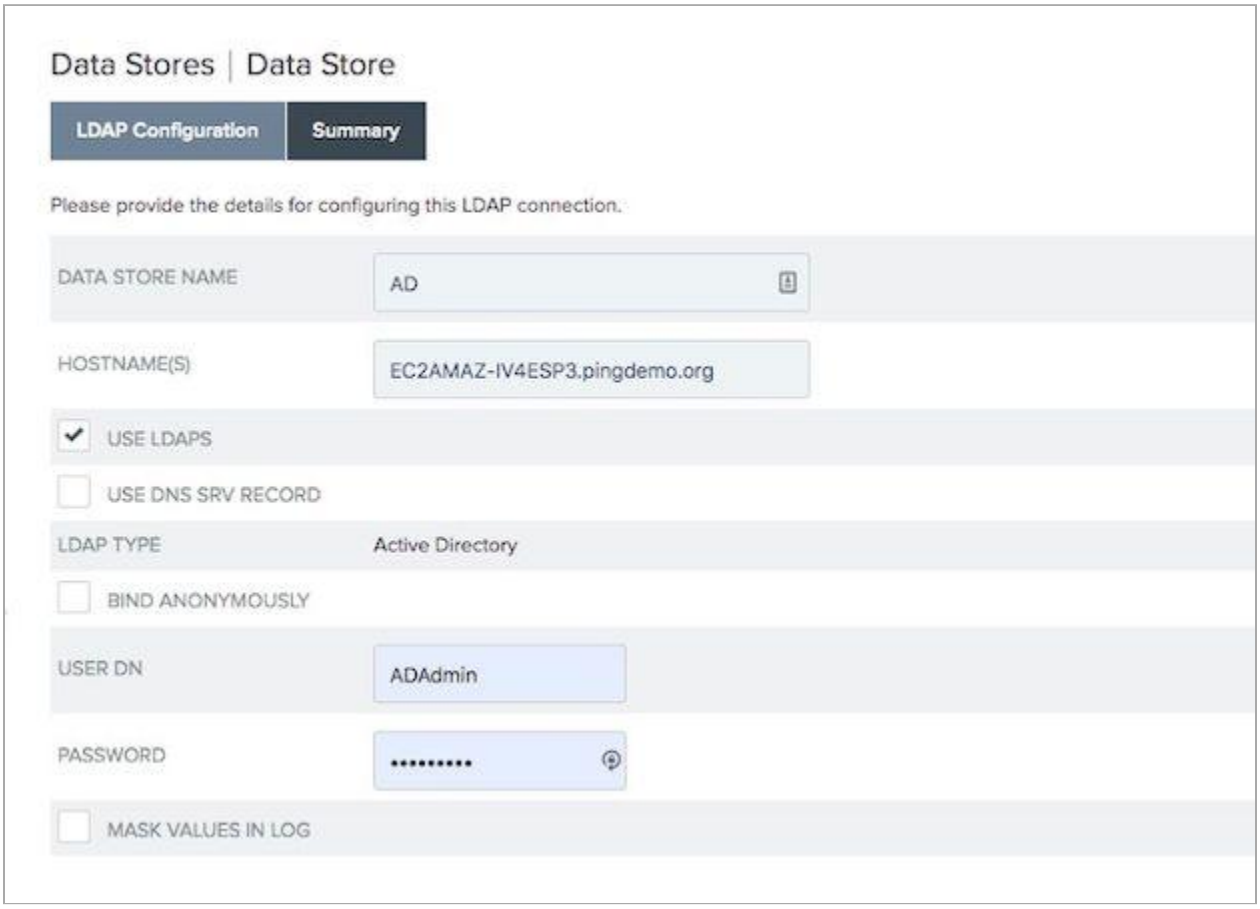
Steps

1. Go to **System > Data & Credential Stores > Data Stores**, and click **Add New Data Store**.
2. On the **Data Store Type** tab, in the **Data Store Name** field, enter a name for the datastore.
3. In the **Type** list, select **Directory (LDAP)**. Click **Next**.

   > **Note**
   >
   > For an Active Directory (AD) datastore, you must issue a certificate from your internal certificate authority (CA) and import it. Follow these substeps to complete the process:

   1. For an AD datastore, go to **Security > Trusted CAs**, and click **Import**.
   2. On the **Import Certificate** tab, click **Choose File** and upload the relevant file. Click **Next.**
   3. On the **Summary** tab, click **Save**.
4. Go to the **LDAP Configuration** tab:
   1. Select the **Use LDAPS** checkbox.

      > **Note**
      >
      > PingFederate assumes port 389 when the **Use LDAPS** checkbox is cleared and assumes port 636 when this checkbox is selected. If you are using the default port of 636, you don't have to specify it in the **Hostname(s)** field.



2. Enter the user attributes in the **User DN** and **Password** fields.

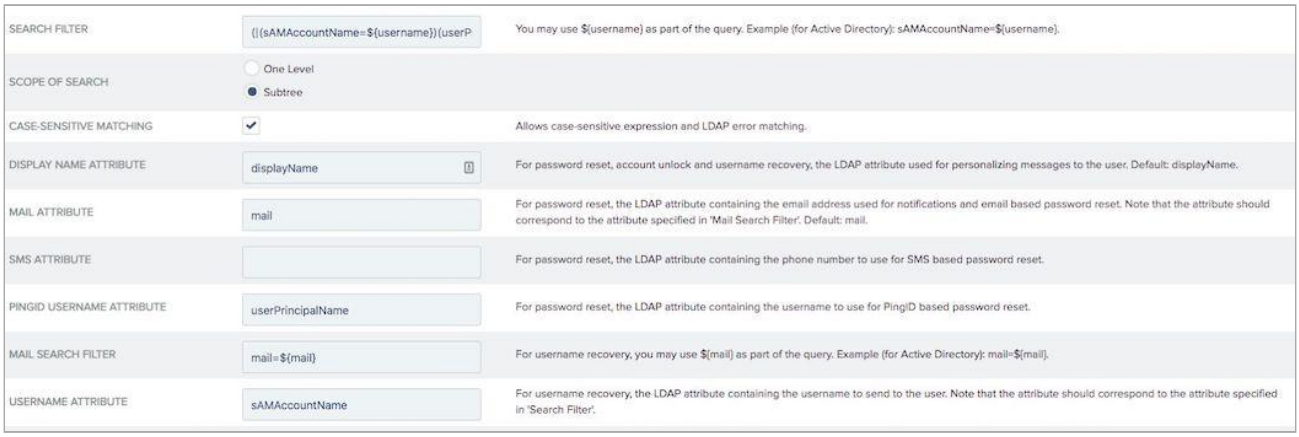
> **Note**
>
> - If the **Password Reset Type** is **PingID** , the user attribute that passes to PingID/> during password reset must be the attribute that is associated with the PingID/> account in PingOne.
> - For an AD datastore, the default user attribute is `sAMAccountName` . This does not have to be the attribute you enter into the username field on the account recovery page.

3. Enter the attributes you want to use to query in the **Search Filter** field.

   The **Search Filter** field, commonly used for Office 365 connections, allows you to enter `sAMAccountName` or `userPrincipleName` .

   For example, `(|(sAMAccountName=${username})(userPrincipalName=${username}))` .

> **Tip**
>
> If the **Password Reset Type** is **PingID**, use a search filter that searches with multiple attributes. You can enter either attribute into the fields, and it passes the username attribute you set in your PCV.
> To view or modify this user attribute:
>
> 1. Go to **System > Data & Credential Store > Password Credential Validators > Password Credential Validators**, and select the relevant PCV instance.
> 2. On the **Instance Configuration** tab, edit the **PingID Username Attribute** field.
>
> This is the attribute used for a PingID password reset type.



5. Click **Next**.
6. Configure the remaining LDAP settings as needed.

   Learn more about the settings in Configuring an LDAP connection (page 871) and Setting advanced LDAP options (page 874).
7. On the **Summary** tab, click **Save**.

Configuring an HTML Form Adapter instance for password reset

Before you begin

Make sure you have configured an LDAP datastore connection in PingFederate to connect to your application to enable self-service password reset.

This task covers specific configuration steps. You can find comprehensive instructions in Setting up an LDAP connection in PingFederate.

About this task

An HTML Form Adapter instance is used to validate a user authentication session with a PCV and an LDAP datastore connection. This authentication mechanism allows you to customize a user's sign-on experience, such as:

- Enabling self-service password reset
- Account unlock
- Notifying users with password expiration information
- Localizable template files

To create or modify an HTML Form Adapter instance with a password credential validator (PCV) and an LDAP datastore connection for self-service password management:

Steps

1. Go to **Identity Provider > IdP Adapters** and choose an HTML Form Adapter:

   Choose from:

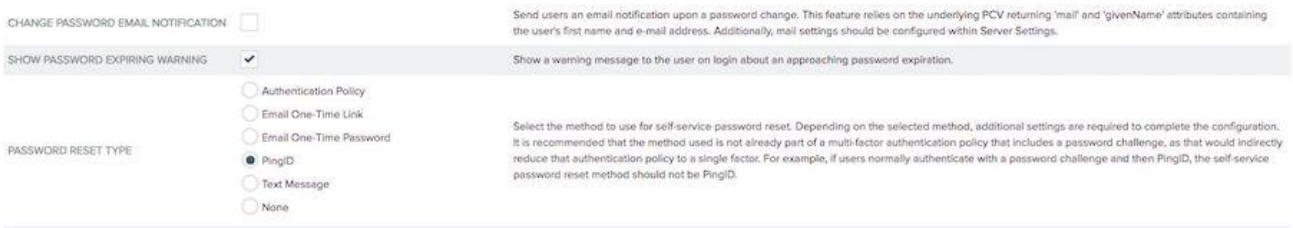
   - In the **Instance Name** list, reuse an existing HTML Form Adapter.

- Click **Create New Instance** to create one.

2. Go to the **IdP Adapter** tab:
    1. Click **Add New Row to 'Credential Validators'** and add the PCV that's linked to your LDAP connection. Click **Update**.
    2. Select the **Allow Password Changes** checkbox.

> **Important**
>
> You must select the **Allow Password Changes** checkbox to enable password reset. If you don't enable this setting, your changes can't be saved.

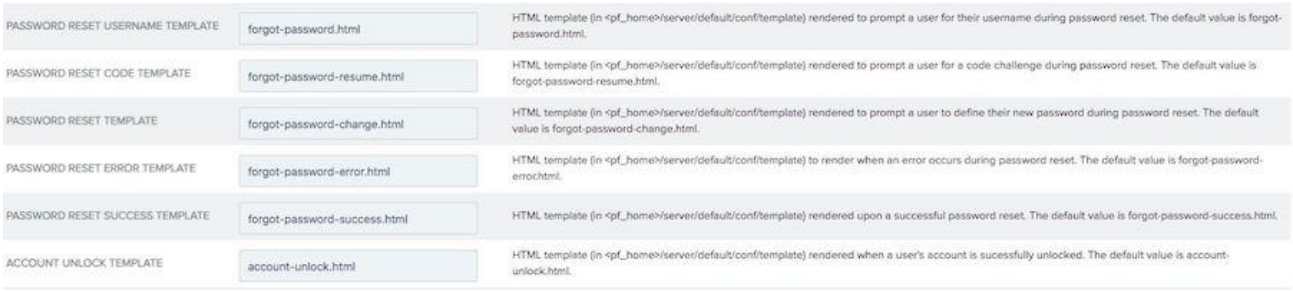
    3. **Optional:** To send the user an email when their password is changed, select the **Change Password Email Notification** checkbox.
    4. **Optional:** To alert the user with an approaching password expiry message at sign on, select the **Show Password Expiring Warning** checkbox.
    5. In the **Password Reset Type** row, click the password reset method that you want to use.



    6. To allow a user with a locked account to unlock the account using the password reset function, select the **Account Unlock** checkbox.

3. To edit the templates for the HTML pages for password reset:
    1. Click **Show Advanced Fields**.
    2. Edit the relevant template fields as needed with the appropriate HTML template.

> **Note**
>
> If you modify and rename a template, make sure to update the template name of that specific template.



4. For the **PingID** password reset type, in the **PingID Properties** field, import your PingID properties file from PingOne.

   This is the same file you used to setup your PingID adapter in PingFederate.

5. Configure the remaining settings as needed. Click **Next**.

   You can find more information about the settings in Configuring an HTML Form Adapter instance (page 288) and HTML Form Adapter advanced fields (page 298).

6. On the **Summary** tab, click **Save**.

Result

You have successfully created an instance of the HTML Form Adapter with the self-service password reset capability. When a user signs on through this adapter instance, the sign-on page displays the **Change Password?** and **Trouble Signing On?** options.

Resetting a password using various methods

You can use several methods to configure password reset. Click the following tabs to see instructions for each method.

Setting up PingDataSync between Active Directory and PingOne

Learn how to configure PingDataSync for Microsoft Active Directory (AD) to PingOne in a Windows environment.

Before you begin

Components

- PingOne
- PingDataSync

You must:

- Install PingDataSync.
- Have the hostname for the AD instance.
- Have the port for the AD instance.

  With AD, this is 389 or 636. If you're not planning to work with passwords, you should keep everything on 389. Steps for working with SSL over port 636 are not a part of this guide.

- Have the AD Admin ID (For example, cn=administrator, cn=users, dc=mydomain, dc=com).
- Have your PingOne Environment ID, Client ID, and Client Secret from your designated PingOne Worker App.

> **Note**
>
> Use the Client ID and Client Secret from the PingOne Worker App that will manage the operation. Learn more about creating and maintaining Worker Apps in Adding an application in the PingOne documentation.

About this task

Setting this configuration primarily uses the `dsconfig.bat` tool.

> **Note**
>
> Although the steps for this configuration are shown in a Windows environment, you can configure this in Linux or Docker with the correct networking configuration in place.

This task uses the following naming conventions:

- PingDataSync Server references: “server” + Application.

  For example, `serverAD` or `serverP1`` .

- PingDataSync objects: object name + source + “to” + destination.

  For example, `mapADtoP1` , `pipeADtoP1` .

Steps

1. To create an external server in PingDataSync, open a terminal window and run the following command.

   > **Note**
   >
   > Make sure to replace the bracketed fields with the values for the administrative user.

   ```
   C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
   create-external-server ^
   --server-name serverAD ^
   --type active-directory ^
   --set server-host-name:<hostname or IP>  ^
   --set server-port:389 ^
   --set bind-dn:<your bind DN> ^
   --set password:<password> ^
   --set connection-security:none ^
   --set key-manager-provider:null ^
   --trustAll ^
   --no-prompt
   ```

   This step defines the connection from PingDataSync to the AD server.

   > **Note**
   >
   > The `--trustAll` and `--no-prompt` parameters bypass any potential certificate issues and suppress prompts or inputs from executing **`dsconfig`** .

2. To create the sync source, specify the starting point for the synchronization process with the following command.

   ```
   C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
   create-sync-source ^
   --source-name sourceAD ^
   --type active-directory ^
   --set base-dn:<your base DN> ^
   --set server:serverAD ^
   --trustAll ^
   --no-prompt
   ```

3. To create the sync destination, run the following command.

```
C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
create-sync-destination ^
--destination-name destinationP1 ^
--type ping-one-customer ^
--set api-url:https://api.pingone.com/v1 ^
--set auth-url:https://auth.pingone.com/<your environment ID>/as/token ^
--set environment-id:<your environment ID> ^
--set oauth-client-id:<your OAuth client ID> ^
--set oauth-client-secret:<your client secret> ^
--trustAll ^
--no-prompt
```

> **Note**
>
> Because you're using PingOne as a destination, you don't need to create an external server reference. Everything is done through the API.

4. Create the attribute map:

    1. Create the map object with the following command.

    ```
    C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
    create-attribute-map ^
    --map-name mapADtoP1 ^
    --trustAll ^
    --no-prompt
    ```

    There are three types of mappings that you can make after you define a map:

    ### *Direct*

    All the contents from the source attribute are mapped to the destination attribute with no changes, for example, `mail` to `email`.

    ### *Constructed*

    The value of the destination attribute is constructed by various means with the simplest use case being a user defined string, for example, `resourceType` to `"user"`.

    ### *JSON Attribute mapping*

    JSON mappings hold a JSON representation of a complex attribute. PingOne specifically uses JSON representation for concepts, such as addresses and name information. These attributes in PingOne are case-sensitive. For example, `Address.street` doesn't work, but `address.streetAddress` does.

    > **Note**
    >
    > The following mappings are suggestions for what works. Your installations might require different mappings.

2. Create the direct attribute mappings.

Mapping	Command

`sAMAccountName` to `accountID`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name accountID ^`
	`--type direct ^`
	`--set from-attribute:samaccountname ^`
	`--trustAll ^`
	`--no-prompt`

`mobile` to `mobilePhone`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name mobilePhone ^`
	`--type direct ^`
	`--set from-attribute:mobile ^`
	`--trustAll ^`
	`--no-prompt`

`mail` to `email`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name email ^`
	`--type direct ^`
	`--set from-attribute:mail ^`
	`--trustAll ^`
	`--no-prompt`

`telephoneNumber` to `primaryPhone`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name primaryPhone ^`
	`--type direct ^`
	`--set from-attribute:telephoneNumber ^`
	`--trustAll ^`
	`--no-prompt`

Mapping	Command

`title` to `title`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name title ^`
	`--type direct ^`
	`--set from-attribute:title ^`
	`--trustAll ^`
	`--no-prompt`

`employeeNumber` to `externalID`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name externalID ^`
	`--type direct ^`
	`--set from-attribute:employeeNumber ^`
	`--trustAll ^`
	`--no-prompt`

`sAMAccountName` to `username`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name username ^`
	`--type direct ^`
	`--set from-attribute:samaccountname ^`
	`--trustAll ^`
	`--no-prompt`

3. Create constructed attribute mappings.

Mapping	Command

`population`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name population ^`
	`--type constructed ^`
	`--set value-pattern:{{"P1People":"name"}} ^`
	`--trustAll ^`
	`--no-prompt`

Mapping	Command

`resourceType`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-attribute-mapping ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name resourceType ^`
	`--type constructed ^`
	`--set value-pattern:user ^`
	`--trustAll ^`
	`--no-prompt`

4. Create JSON attribute maps:

   - To create the `name` attribute, run the following command.

     ```
     C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
     create-attribute-mapping ^
     --map-namemapADtoP1 ^
     --mapping-name name ^
     --type json ^
     --trustAll ^
     --no-prompt
     ```

     Note

     The PingOne name attribute holds information about the identity's name — first name, last name, and formatted (display name).

   - To create the `address` attribute, run the following command.

     ```
     C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
     create-attribute-mapping ^
     --map-name mapADtoP1 ^
     --mapping-name address ^
     --type json ^
     --trustAll ^
     --no-prompt
     ```

     Note

     The PingOne address attribute holds address information and maps to a number of different fields.

5. Create JSON attribute mappings.

Mapping	Command

`sn` to `name.family`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name name ^`
	`--field-name family ^`
	`--set json-type:string ^`
	`--set from-attribute:sn ^`
	`--trustAll ^`
	`--no-prompt`

`givenName` to `name.given`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name name ^`
	`--field-name given ^`
	`--set json-type:string ^`
	`--set from-attribute:givenName ^`
	`--trustAll ^`
	`--no-prompt`

`cn` to `name.formatted`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name name ^`
	`--field-name formatted ^`
	`--set json-type:string ^`
	`--set from-attribute:cn ^`
	`--trustAll ^`
	`--no-prompt`

`l` to `address.locality`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name address ^`
	`--field-name locality ^`
	`--set json-type:string ^`
	`--set from-attribute:l ^`
	`--trustAll ^`
	`--no-prompt`

Mapping	Command

`postalCode` to `address.postalCode`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name address ^`
	`--field-name postalCode ^`
	`--set json-type:string ^`
	`--set from-attribute:postalCode ^`
	`--trustAll ^`
	`--no-prompt`

`st` to `address.region`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name address ^`
	`--field-name region ^`
	`--set json-type:string ^`
	`--set from-attribute:st ^`
	`--trustAll ^`
	`--no-prompt`

`street` to `address.streetAddress`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name address ^`
	`--field-name streetAddress ^`
	`--set json-type:string ^`
	`--set from-attribute:street ^`
	`--trustAll ^`
	`--no-prompt`

`c` to `address.countryCode`	`C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^`
	`create-json-attribute-mapping-field ^`
	`--map-name mapADtoP1 ^`
	`--mapping-name address ^`
	`--field-name countryCode ^`
	`--set json-type:string ^`
	`--set from-attribute:c ^`
	`--trustAll ^`
	`--no-prompt`

5. Create the sync pipe with the following command.

```
C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
create-sync-pipe ^
--pipe-name pipeADtoP1 ^
--set started:true ^
--set sync-source:sourceAD ^
--set sync-destination:destinationP1 ^
--trustAll ^
--no-prompt
```

Creating the sync pipe provides the object that is directly used by PingDataSync and continues to bring the PingDataSync objects together.

6. Create the sync class with the following command.

```
C:\<Ping>\<PingDataSync>\bat\dsconfig.bat ^
create-sync-class ^
--pipe-name pipeADtoP1 ^
--class-name classADtoP1 ^
--set attribute-map:mapADtoP1 ^
--set "include-filter:(objectClass=user)" ^
--set auto-mapped-source-attribute:-none- ^
--set destination-correlation-attributes:username ^
--set replace-all-attr-values:true ^
--set creates-as-modifies:true ^
--trustAll ^
--no-prompt
```

The sync class brings the remaining objects together and is directly linked to the sync pipe.

7. To test the PingDataSync connection between AD and PingOne, run the `resync -p pipeADtoP1` command.

> **Note**
>
> If the sync encounters any errors, examine the `C:\<Ping>\<PingDataSync>\logs\tools\re-sync-failed-DNs.log` file.

Setting up PingDataSync between PingDirectory and PingOne

Learn how to set up PingDataSync between PingDirectory and PingOne using installation commands for Linux.

Before you begin

You must have:

- PingDataSync
- PingDirectory
- PingOne

You must:

- Install PingDataSync.
- (Optional) Note the following values in a plain text file for easy copy and paste to the command line:
  - Implementation suffix
  - Host name for the PingDirectory instance
  - PingDirectory port
  - PingDirectory starting point
  - PingDirectory filter
  - PingDirectory Admin ID
  - PingDirectory Admin password
  - PingOne Population ID
  - PingOne Environment ID
  - WorkerApp Client ID
  - WorkerApp Client Secret

> **Note**
>
> Use the Client ID and Client Secret from the PingOne Worker App that will be managing the operation. Learn more about creating and maintaining Worker Apps in Adding an application in the PingOne documentation.

- (Optional) Use SSO for the PingAuthorize Administrative Console.

This allows administrative users to single sign-on (SSO) to the PingData admin console from PingOne.

Steps

1. To create an external server, run the following command:

```
/opt/<PingDataSync>/bin/dsconfig create-external-server --server-name serverPD_PDtest --type ping-identity-ds --set server-host-name:localhost --set server-port:11389 --set bind-dn:<your bind DN> --set password:<your password>  --set connection-security:none --set key-manager-provider:null --trustAll --no-prompt
```

> **Note**
>
> The `--type` parameter is different if you're using Active Directory or another Directory Server type.

2. To create a sync source, run the following command:

```
/opt/<PingDataSync>/bin/dsconfig create-sync-source --source-name sourcePD_PDtest --type ping-identity --set base-dn:ou=test,dc=p1,dc=lab --set server:serverPD_PDtest --trustAll --no-prompt
```

> **Note**
>
> Make sure that your `base-dn` indicates where you want to start in the directory tree.

3. To create a sync destination, run the following command:

```
/opt/<PingDataSync>/bin/dsconfig create-sync-destination --destination-name destinationPD-P1_PDtest --trustAll --no-prompt --type ping-one-customer --set api-url:https://api.pingone.com/v1 --set auth-url:https://auth.pingone.com/<your PingOne environment ID>/as/token --set environment-id:<your PingOne environment ID>  --set oauth-client-id:<your worker app client ID>  --set oauth-client-secret:<your worker app client secret>  --set default-population-id:<your PingOne population ID>
```

> **Note**
>
> Setting the population ID here avoids having to configure it in the attribute mapping section.

4. To create an attribute map, run the following command:

```
/opt/<PingDataSync>/bin/dsconfig create-attribute-map --map-name mapPDtoP1_PDtest --trustAll --no-prompt
```

There are three types of mappings that you can make after you define a map:

Direct

All the contents from the source attribute are mapped to the destination attribute with no changes, such as `mail` to `email`.

Constructed

The value of the destination attribute is constructed by various means, with the simplest use case being a user defined string, such as `resourceType` to `"user"`.

JSON Attribute mapping

JSON mappings hold a JSON representation of a complex attribute. PingOne specifically uses JSON representation for concepts, such as addresses and name information. These attributes in PingOne are case-sensitive. For example, `Address.street` doesn't work, but `address.streetAddress` does.

> **Note**
>
> The following mappings are suggestions for what works. Your installations will possibly require different mappings.

1. Create direct mappings.

> **Tip**
>
> This is easier to run as a **`dsconfig`** batch.

1. Create a `<PingDataSync>/directMapping.dsconfig` text file.
2. Place the following commands into your `directMapping` file:

```
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name
accountID --type direct --set from-attribute:uid --trustAll --no-prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name
mobilePhone --type direct --set from-attribute:mobile --trustAll --no-prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name email
--type direct --set from-attribute:mail --trustAll --no-prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name
primaryPhone --type direct --set from-attribute:telephoneNumber --trustAll --no-
prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name title
--type direct --set from-attribute:title --trustAll --no-prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name
externalID --type direct --set from-attribute:employeeNumber --trustAll --no-
prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name
username --type direct --set from-attribute:uid --trustAll --no-prompt
```

3. Run the batch with the following command:

```
/opt/<PingDataSync>/bin/dsconfig --trustAll --no-prompt --batch-file /opt/<Your
directMapping file name>.dsconfig
```

2. Create constructed attribute mappings with the following command:

```
/opt/<PingDataSync>/bin/dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --
mapping-name resourceType --trustAll --no-prompt --type constructed --set value-
pattern:user
```

3. Create JSON attribute maps.

> **Tip**
>
> This is easier to run as a `dsconfig` batch. The JSON maps are created as a subset of the attribute map that was just constructed and are populated in the following steps.

1. Create a `<PingDataSync>/jsonMap.dsconfig` text file.
2. Place the following commands in your `jsonMap` file:

```
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name name
--type json --trustAll --no-prompt
dsconfig create-attribute-mapping --map-name mapPDtoP1_PDtest --mapping-name
address --type json --trustAll --no-prompt
```

3. Run the batch with the following command:

```
/opt/<PingDataSync>/bin/dsconfig --trustAll --no-prompt --batch-file /opt/
jsonMap.dsconfig
```

4. Create JSON attribute mappings.

> **Tip**
>
> This is easier to run as a **`dsconfig`** batch.

1. Create a `<PingDataSync>/jsonMapping.dsconfig` text file.
2. Place the following commands in your `jsonMapping` file:

```
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name name --field-name family --set json-type:string --set from-
attribute:sn --trustAll --no-prompt
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name name --field-name given --set json-type:string --set from-
attribute:givenName --trustAll --no-prompt
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name name --field-name formatted --set json-type:string --set from-
attribute:cn --trustAll --no-prompt
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name address --field-name locality --set json-type:string --set from-
attribute:l --trustAll --no-prompt
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name address --field-name postalCode --set json-type:string --set from-
attribute:postalCode --trustAll --no-prompt
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name address --field-name region --set json-type:string --set from-
attribute:st --trustAll --no-prompt
dsconfig create-json-attribute-mapping-field --map-name mapPDtoP1_PDtest --
mapping-name address --field-name streetAddress --set json-type:string --set from-
attribute:street --trustAll --no-prompt
```

3. Run the batch with the following command:

```
/opt/<PingDataSync>/bin/dsconfig --trustAll --no-prompt --batch-file /opt/
jsonMapping.dsconfig
```

5. To create a SyncPipe, run the following command:

```
/opt/<PingDataSync>/bin/dsconfig create-sync-pipe --pipe-name  pipePDtoP1_PDtest --set started:true
--set sync-source:sourcePD_PDtest --set sync-destination:destinationPD-P1_PDtest --trustAll --no-
prompt
```

6. To create a sync class, run the following command:

```
/opt/<PingDataSync>/bin/dsconfig  create-sync-class --pipe-name pipePDtoP1_PDtest --class-name
classPDtoP1_PDtest --set attribute-map:mapPDtoP1_PDtest --set "include-filter:
(objectClass=inetOrgPerson)" --set auto-mapped-source-attribute:-none- --set destination-
correlation-attributes:username --set replace-all-attr-values:true --set creates-as-modifies:true --
trustAll --no-prompt
```

7. Test the sync:

    1. Run the sync with the following command:

    ```
    /opt/<PingDataSync>/bin/resync -p pipePDtoP1_PDtest
    ```

    2. (Optional) If the sync results in any errors, examine the `/Ping/<PingDataSync>/logs/tools/re-sync-failed-DNs.log`.

    3. (Optional) If you receive an error that includes `Cannot connect because: The connection to server localhost:11389 was closed while waiting for a response to a bind request SimpleBindRequest(dn='cn=dmanager').`:

        1. In the PingDataSync admin console, go to **Configuration** > **External Servers** > **ServerPD_PDtest**.

        2. Update your password.

Workforce Use Cases

Use case	Description
Authenticating Azure AD tenants who don't have their own Azure account	Create a PingFederate workflow to authenticate users from different Microsoft tenants.
Configuring adaptive authentication in PingFederate	This document explains the conceptual information behind network-based adaptive authentication. It also provides instructions for creating a new selector and configuring an authentication policy to enable adaptive authentication.
Configuring an Active Directory datastore for PingFederate	In PingFederate, establish an Active Directory datastore connection for retrieving user attributes for outbound connections.
Configuring a SAML application	Configure a SAML application in PingFederate, PingOne, and PingOne for Enterprise.
Connecting PingFederate to a Microsoft SQL JDBC datastore with Windows authentication	Create a Microsoft SQL server Java Database Connectivity (JDBC)-connected datastore in PingFederate and configure it for Windows authentication.
Connecting PingFederate with Yahoo through OIDC	Learn how to connect PingFederate with your Yahoo developer account using OpenID Connect (OIDC).
Delegating all authentication to an external IdP	PingOne provides an authentication policy step that allows you to make an external identity provider (IdP) part of a PingOne authentication policy or delegate all authentication to that external IdP.
Enabling SLO for a PingAccess-protected application using PingFederate	Learn how to require a sign off of a PingAccess-protected application with PingFederate acting as token provider.
Integrating Pulse Connect Secure with PingFederate	Learn how to integrate Pulse Connect Secure with PingFederate for single sign-on (SSO).
Protecting a web application with PingAccess using PingFederate as the token provider	Configure a proof of concept to protect a web application from unwanted access using PingAccess with PingFederate as the token provider.
Protecting your VPN with PingID MFA	To improve network security posture and provide a true MFA experience to network resources, add PingID multi-factor authentication (MFA) to your VPN authentication ceremony.
Setting up a login form that validates credentials against AD in PingFederate	Configure a login form in PingFederate that validates credentials against Active Directory (AD).
Setting up an agent in PingAccess	Learn how to set up an agent integration for PingAccess applications.

Use case	Description
Setting up an authentication flow that includes MFA (PingFederate and PingID)	This configuration creates a service provider (SP) connection with a multi-factor authentication (MFA) flow using PingFederate and PingID.
Setting up an authentication flow that includes MFA (PingOne for Enterprise and PingID)	You can create an authentication flow that uses multi-factor authentication (MFA) with PingOne for Enterprise and PingID.
Setting up an OIDC application in PingFederate	Create a new OAuth or OpenID Connect (OIDC) application in PingFederate.
Setting up and testing a custom authentication policy	Authentication policies are used in PingFederate to implement complex authentication requirements. This document explains how to create a new custom authentication policy in PingFederate, and then test the policy.
Setting up Kerberos authentication in PingFederate	Set up a Kerberos authentication adapter in PingFederate for a seamless user authentication experience from a Windows machine to your applications.
Setting up password recovery in PingFederate	Learn how to set up PingFederate for self-service password reset and account recovery through an HTML Form Adapter.
Setting up passwordless authentication in PingOne	Learn how to set up passwordless authentication and eliminate the need for your users to enter a password. Passwordless authentication is a quick and easy configuration where end users sign on with a paired multi-factor authentication (MFA) device.
Setting up PingFederate as a FedHub	Configuring PingFederate as an identity bridge or FedHub (SAML Chaining) allows you to manage external identities and facilitate access to applications across the enterprise community.
Setting up your PingOne Dock	The PingOne Dock gives your users one-click, single sign-on (SSO) access to the applications and other service providers (SPs) you authorize them to use.
Updating a PingOne for Enterprise verification certificate on an unmanaged PingFederate identity bridge	If you use an unmanaged manual PingFederate connection as the identity provider (IdP) for PingOne for Enterprise, and your certificate is about to expire, you must update your signing certificate in PingFederate and your verification certificate in PingOne for Enterprise.

Authenticating Azure AD tenants who don't have their own Azure account

Create a PingFederate workflow to authenticate users from different Microsoft tenants.

If you use Microsoft Azure AD as an identity provider (IdP), a standard IdP connection won't authenticate users from other Azure tenants or from other Microsoft account types, such as outlook.com, live.com, or hotmail.com.

If you have users from these other tenants, you can authenticate them through the Azure Application Registration Portal and V2 endpoints.

Component

PingFederate 10.0

Creating an OIDC V2 app for AuthN

Register a new OpenID Connect (OIDC) application in the Azure App registration service.

Steps

1. In the Azure portal, go to **App registrations > New registration**.
2. Enter an application name and click **Create**.

   > **Tip**
   >
   > Give your application a name that identifies it and differentiates it from applications created through Azure AD, such as `PingAuthentication-V2` .

3. Under **Supported account types**, click **Accounts in any organizational directory and personal Microsoft accounts**.
4. Click **Register**.

   Result:

   The **Overview** tab provides the **Application (client) ID**. This is the **Client ID** for your PingFederate OIDC IdP connection.

5. Click **API permissions**.
6. Click **Add a permission > Microsoft Graph > Delegated permissions > Directory** and select the **Directory.Read.All** checkbox.
7. Click **Add permissions**.
8. (Optional) Click the **Branding** tab to customize the following:
   - Brand logo
   - Home page URL
   - Terms of Service URL
   - Privacy Statement URL

9. At the top of the page, click **Save**.

Viewing and updating the app in ADD dashboard

Verify and update the permissions in your v2 Azure application.

Steps

1. In the Azure portal, go to **App registrations**.
2. Click the V2 application that you created to open the **Overview** tab.
3. Click **Authentication**.

   Result:

   Under the **Supported account types** heading, you see `Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)`.
4. At the top of the page, click **Save**.

Creating an OpenID Connect IdP connection in PingFederate

Steps

1. In PingFederate, go to **Authentication** > **Integration** > **IdP Connections** and click **Create Connection**.
2. On the **Connection Type** tab, select **Browser SSO**.
3. In the **Protocol** list, select **OpenID Connect**.
4. Click **Next**.
5. On the **Connection Options** tab, click **Next**.
6. On the **General Info** tab, enter the following values:
   1. In the **Issuer** field, enter `https://login.microsoftonline.com/common` and click **Load Metadata**.

      Result:

      When you click **Load Metadata**, the **Issuer** field is updated with a metadata URL.
   2. Replace the *<tenant>* placeholder at the end of the URL with your Microsoft Tenant ID and add `/v2.0` to the end of the URL.

      > **Note**
      >
      > You can find your Tenant ID at **Azure Active Directory** > **Overview** in your Microsoft Azure account.
   3. Select the **Enable Additional Issuers** checkbox.
   4. In the **Connection Name** field, enter a plain-language identifier for the connection, for example a company or department name.

      This name is displayed in the connection list in the administration console.

5. In the **Client ID** field, enter the **Application (client) ID** value found in the **App registrations** menu in Azure AD.
6. Click **Next**.

7. On the **Additional issuers** tab, select the **Accept All issuers (Not Recommended)** checkbox and click **Save**.
8. On the **Browser SSO** tab, click **Configure Browser SSO**.
    1. On the **User-Session Creation** tab, click **Configure User-Session Creation**
    2. Choose one of the **Identity Mapping** tab options:
        - Click **Account Mapping** if you plan to pass end-user claims to the target application through a service provider (SP) adapter instance or an authentication policy contract if your PingFederate server is a federation hub that bridges an OpenID provider to an SP.
        - Click **Account Linking** if your target application requires account linking.
        - Click **No Mapping** if you plan to pass end-user claims to the target application through an authentication policy contract in an SP authentication policy.
9. Delete the attributes that are unnecessary to your application in the **Attribute Contract** menu generated by the issuer metadata in step 5.

    Troubleshooting:

    You are likely to encounter attribute-related errors when testing your connection. If this occurs, review the `server.log` file to see what attributes or claims are sent to Azure and delete the unnecessary attributes from your attribute contract.
10. (Optional) On the **Target Session Mapping** menu, click **Map New Adapter Instance** to map end-user claims to the target application through an SP adapter instance or an authentication policy contract.

    Learn more in Managing target session mappings (page 659).
11. On the **Summary** tab, review the **User Session Creation** settings and click **Save**.
12. On the **Protocol Settings** tab, click **Configure Protocol Settings**.
13. On the **OpenID Provider Info** tab, enter the following values.

Field	Value
Authorization Endpoint	`https://login.microsoftonline.com/common/oauth2/v2.0/authorize`
Token Endpoint	`https://login.microsoftonline.com/common/oauth2/v2.0/token`
User Info Endpoint	`https://graph.microsoft.com/oidc/userinfo`
JWKS URL	`https://login.microsoftonline.com/common/discovery/v2.0/keys`

14. When you have finished configuring the identity provider (IdP) connection, copy the **Redirect URI** from the **Activation & Summary** tab and add it to your V2 application.
    1. In your Azure account, go to **App registrations**.

2. Click the application you want to connect.
3. Click **Authentication > Add a platform > Web**.
4. Paste the redirect URI into the **Enter the redirect URI of the application** field.
5. Select both the **Access Tokens** and **ID Tokens** checkboxes.
6. Click **Configure**.

Result

You can now authenticate users with non-Azure Microsoft accounts.

Configuring adaptive authentication in PingFederate

This document explains the conceptual information behind network-based adaptive authentication. It also provides instructions for creating a new selector and configuring an authentication policy to enable adaptive authentication.

Network-based adaptive authentication is useful when PingFederate must authenticate users differently based on their network location. A typical application of this use case is when users must authenticate differently, depending on whether they are accessing a service from the organization's internal network or from the internet. For example, an organization might want to use Kerberos to authenticate internal users to provide a seamless single sign-on (SSO) experience while presenting a sign-on page for external users.

Network-based adaptive authentication is achievable on all supported versions of PingFederate. The examples shown make use of PingFederate 10.1. All capabilities are offered out-of-the-box and no additional or custom components are required to implement this solution.

Component

PingFederate 10.1

Creating a new selector

Selectors and authentication sources can be conditionally chained together in paths to form policies.

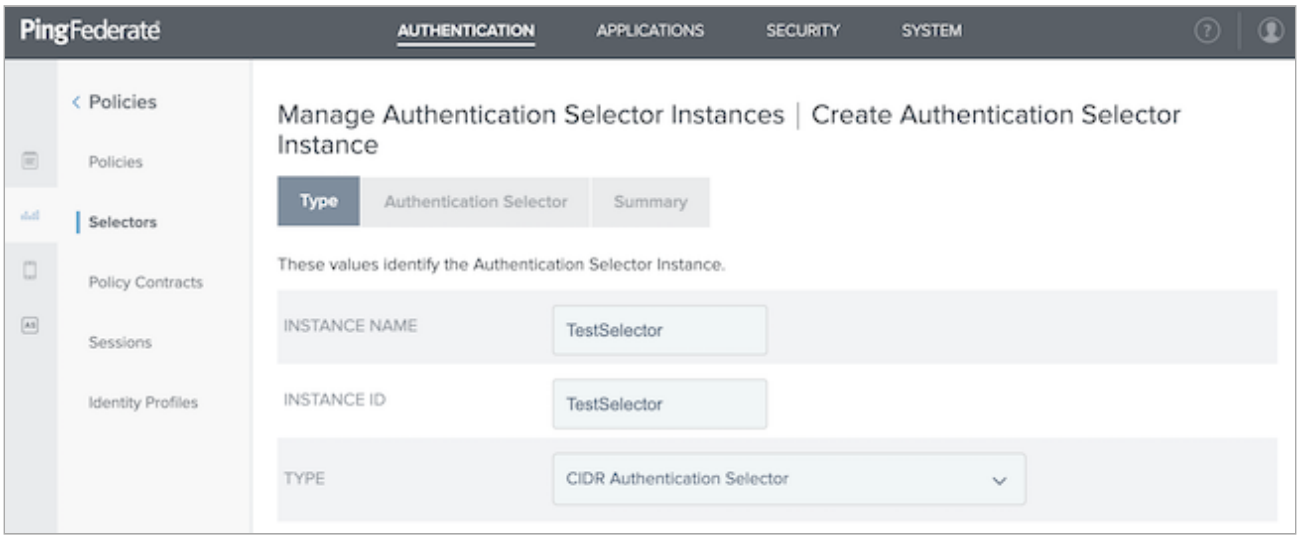
Before you begin

- PingFederate must determine if a user is inside your internal network. You must know CIDR network ranges that identify your internal network.
- Upon identifying the network location of your user, you must know how you intend to authenticate your user in each case.
    - Configure authentication adapters, such as the Kerberos adapter and the HTML form adapter, along with their dependencies (Kerberos Realms and password credential validators (PCVs), respectively).
- Define an authentication policy contract to allow the outcome of the authentication process to be mapped into your SAML connections or OAuth environment.

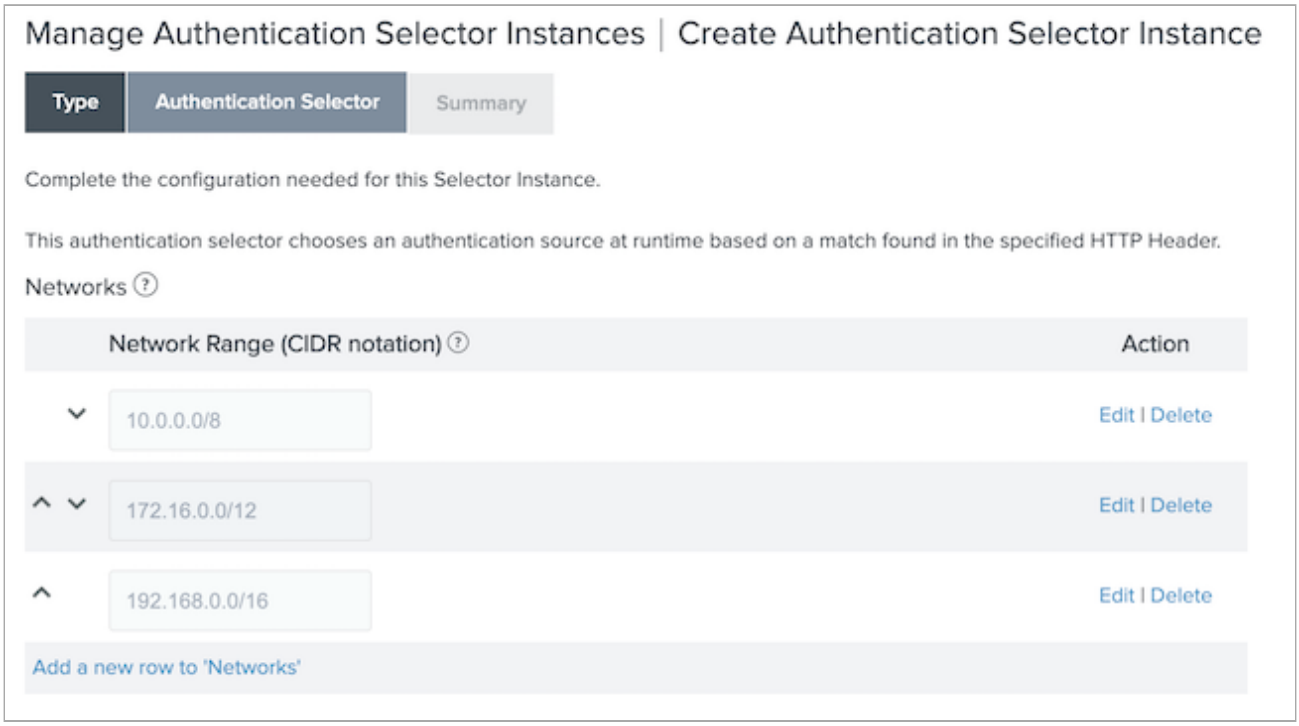
Steps

1. In the PingFederate administrative console, go to **Authentication > Policies > Selectors**.

2. To create a new selector, click **Create New Instance**.
3. Configure the **Type** window.
    1. In the **Instance Name** field, enter an instance name.
    2. In the **Instance ID** field, enter the instance ID.
    3. In the **Type** list, select **CIDR Authentication Selector**.
    4. Click **Next**.



4. Configure the **Authentication Selector** window.
    1. Click **Add a new row to 'Networks'**.
    2. In the **Network Range (CIDR notation)** field, enter the CIDR ranges that identify your internal network address ranges.



3. To save your network, click **Update**.
4. **Optional:** In the **Result Attribute Name** field, enter an attribute name.
5. Click **Next**.

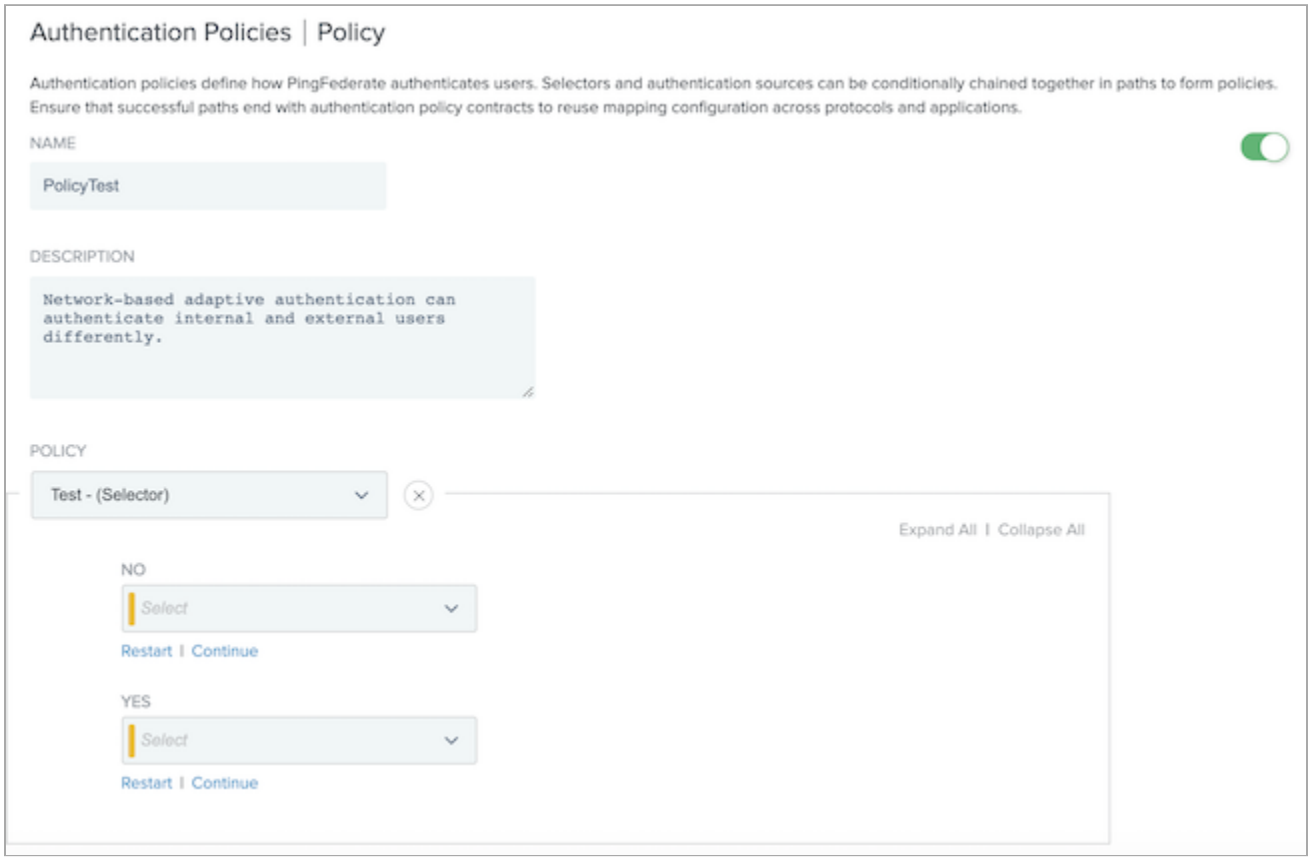
5. On the **Summary** window, click **Done**.
6. Click **Save**.

Configuring the authentication policy

Authentication policies define how PingFederate authenticates users.

Steps

1. In the PingFederate administrative console, go to **Authentication > Policies > Policies**.
2. To create a new policy, click **Add Policy**.
3. Configure the **Policy** window.

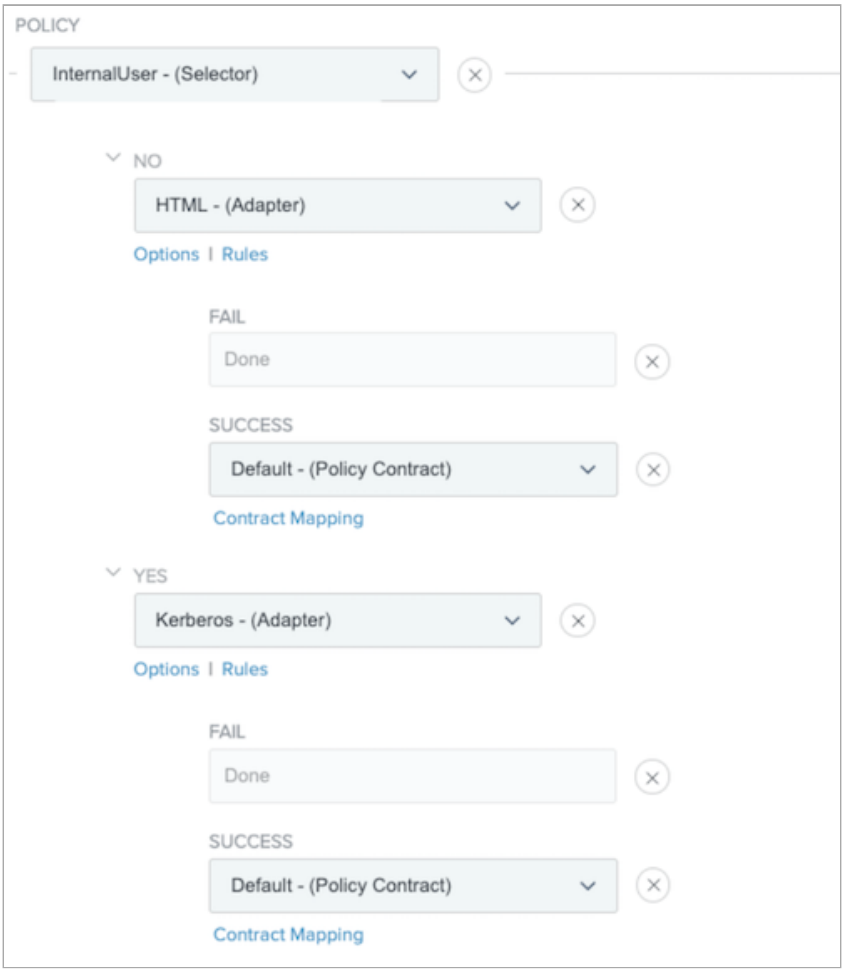


1. In the **Name** field, enter a name.
2. In the **Description** field, enter a description.
3. From the **Policy** list, go to **Selectors** and choose your previously created selector.

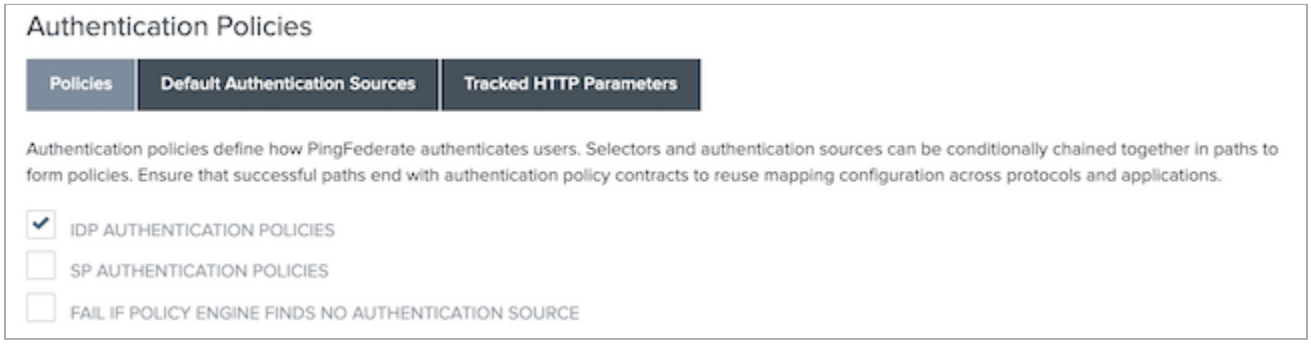
   Result:

   After choosing your selector, additional fields display that require you to identify which authentication adapters to use for internal and external users.

4. From the additional lists that display, configure the authentication adapters to be used for internal and external users.



5. Click **Done**.
6. Click **Save**.
7. To enable the network-based adaptive authentication policy, go to **Authentication > Policies > Policies** and select the **IDP Authentication Policies** checkbox.



Next steps

- Map the policy contract you used after completing the adaptive authentication within your SAML connections, OAuth persistent grants, or both.
- You can hierarchically organize the policy to appear earlier or later in the **Policy** list.

> **Note**
>
> To configure PingFederate with multiple authentication policies or specify the order in which they are presented, go to **Authentication > Policies > Policies**.

Configuring an Active Directory datastore for PingFederate

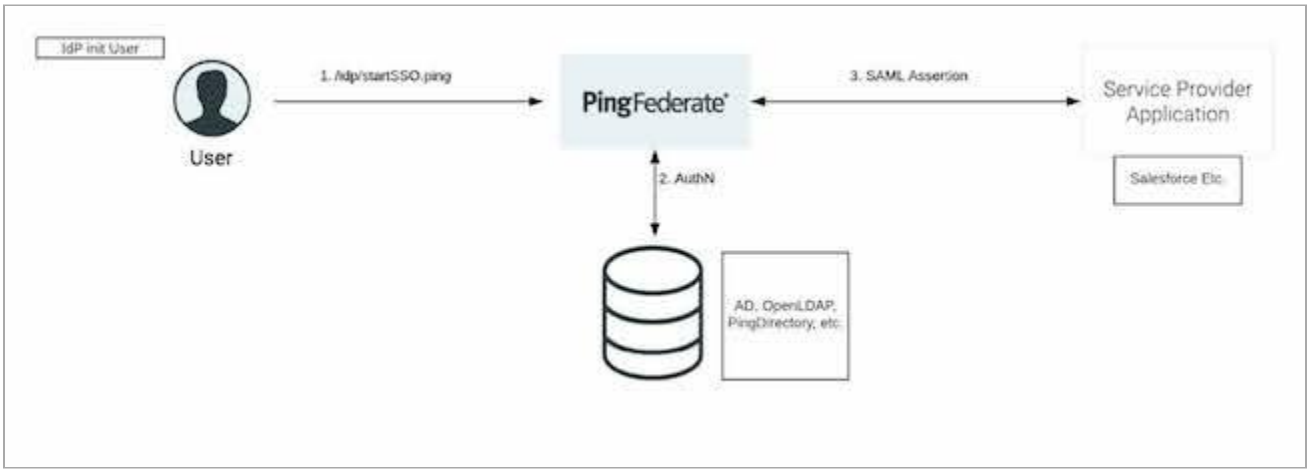
In PingFederate, establish an Active Directory datastore connection for retrieving user attributes for outbound connections.

Component

PingFederate 10.1

Processing steps

Almost every customer using PingFederate as an identity provider (IdP) has at least one connection to a datastore. A datastore connection allows PingFederate to retrieve user attributes for outbound connections. Active Directory is the most common data source used to connect to PingFederate.



1. The user initiates single sign-on (SSO) and activates PingFederate.
2. The user enters credentials in the htmlForm page. PingFederate query's the connected datastore for authentication.
3. A SAML assertion is sent to the service provider containing the select attributes for SSO.

Configuring an Active Directory datastore

In PingFederate, configure a datastore connection to allow PingFederate, the identity provider (IdP), to retrieve user attributes for outbound connections.

Before you begin

Your administrator account associated with Active Directory must be configured in the directory and have read permissions to the organizational unit where user attribute searches are done.

About this task

This topic details specific tasks for configuring an Active Directory datastore connection. Learn more in Datastores (page 125) in the PingFederate Server documentation.

Steps

1. From the PingFederate admin console, go to **System > Data Stores**. Click **Add a New Data Store**.

   Result:

   The **Data Store** window configuration opens.

2. On the **Data Store Type** tab:
   1. In the **Name** field, enter a name.
   2. From the **Type** list, select **Directory (LDAP)**.
   3. Click **Next**.
3. On the **LDAP Configuration** tab:
   1. In the **Hostname(s)** field, enter the hostname for the configuration. Click **Add**.

      This is the hostname of the domain controller.

      > **Note**
      >
      > The **Hostname(s)** field entry can rely on network naming to route to the closest domain controller. For example, `pingdemo.com` resolves to `dc1.pingdemo.com`.
      > Alternatively, you can define domain controllers explicitly, separated by a space. For example, `dc1.pingdemo.com dc2.pingdemo.com`. This creates a failover to each domain controller. If it does not find the user in the first directory, it then queries the second and so on.

   2. In the **User DN** field, enter the distinguished name (DN).

      This is used as the domain name of the service account used to query the directory.

   3. In the **Password** field, enter a password.

      This is the password of the service account.

   4. Select the **Use DNS SRV Record** checkbox.

      SRV records are not required for this configuration, but you can use them.

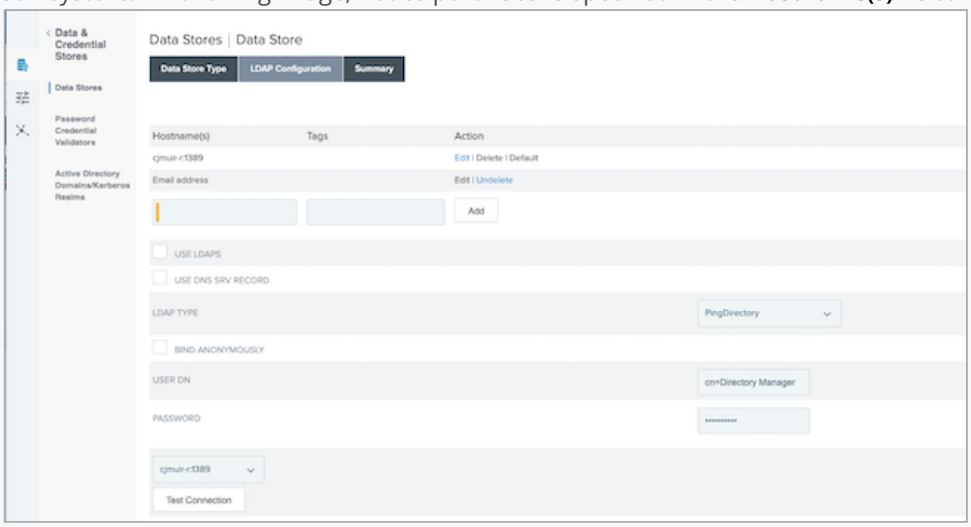
   5. Choose whether to enable the **Use LDAPS** checkbox.
      - Select the **Use LDAPS** checkbox.

        The configuration assumes port 636 if the LDAPS option is selected.

      - Clear the **Use LDAPS** checkbox.

        The configuration assumes port 389 if the LDAPS option is cleared.

> **Note**
>
> If you are running your directory on another port, you must state this in the **Hostname(s)** field as shown in the image below, and have the Active Directory public certificate uploaded in your trusted keystore. In following image, notice port 1389 is specified in the **Hostname(s)** field.



6. Click **Next**.
7. On the **Summary** tab, click **Save**.

   Result:

   The **Data Store** configuration window closes. You are directed back to the **Data Stores** window where you can manage all your datastore connections.

Configuring a SAML application

Configure a SAML application in PingFederate, PingOne, and PingOne for Enterprise.

Read the following sections for instructions for each product.

Configuring a SAML application in PingFederate

Configure a SAML application in PingFederate.

Before you begin

Component

- PingFederate 10.1

Make sure you have the following:

- A datastore connection
- A configured password credential validator (PCV)

- A configured identity provider (IdP) adapter.
- An IdP digital signing certificate

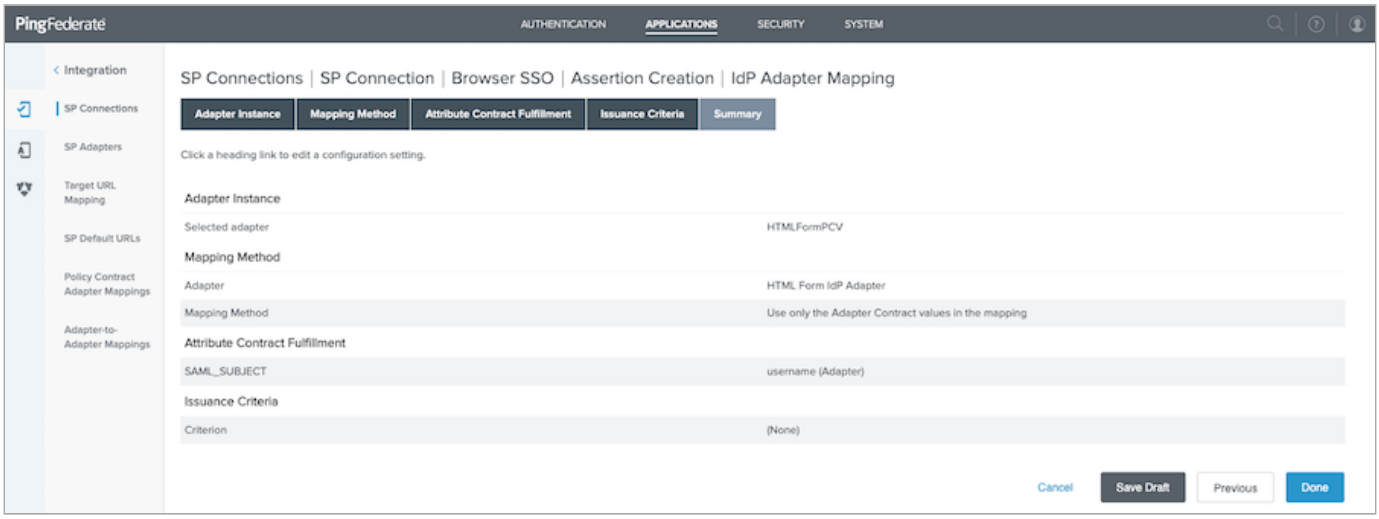
Steps

1. In the PingFederate administrative console, go to **Applications > Integration > SP Connections**.
2. Click **Create Connection**.
3. On the **Connection Template** tab, click **Do not use a template for this connection**. Click **Next**.
4. On the **Connection Type** tab, select the **Browser SSO Profiles** checkbox.
5. In the **Protocol** list, select **SAML 2.0**. Click **Next**.
6. On the **Connection Options** tab, leave the **Browser SSO** checkbox selected, and then click **Next**.
7. On the **Import Metadata** tab, import service provider (SP) metadata, pull from a URL, or enter the data manually. Click **Next**.

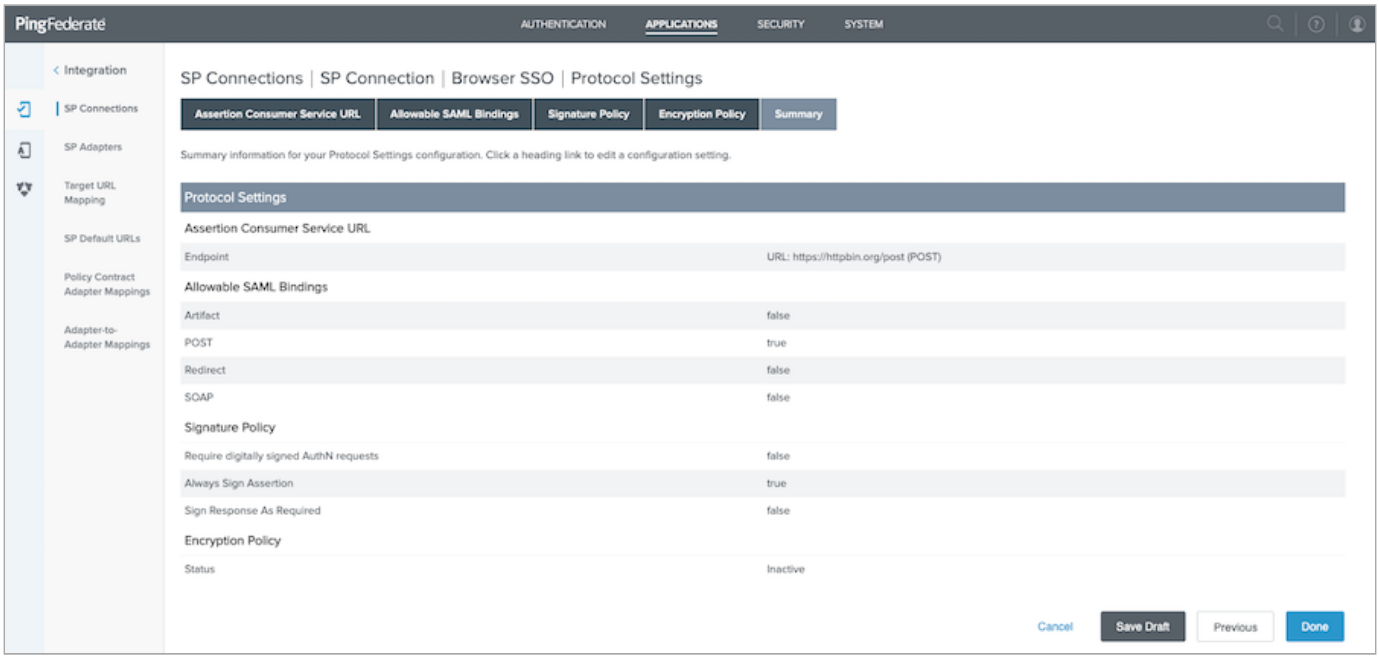
   In this example, we assume that SP metadata is provided.
8. On the **General Info** tab, provide a **Connection Name** if needed and review the information. Click **Next**.

   > **Note**
   >
   > **Entity ID** and **Base URL** should be provided by the SP.
9. On the **Browser SSO** tab, click **Configure Browser SSO**.
10. On the **SAML Profiles** tab, select the **IdP-Intitiated SSO** and **SP-Initiated SSO** checkboxes. Click **Next**.
11. On the **Assertion Lifetime** tab, leave the default entries, and then click **Next**.
12. On the **Assertion Creation** tab, click **Configure Assertion Creation**.
13. On the **Identity Mapping** tab, click **Standard**. Click **Next**.
14. On the **Attribute Contract** tab, ensure that whatever attributes you need for the SP are defined here. Click **Next**.
15. On the **Authentication Source Mapping** tab, click **Map New Adapter Instance**.
16. On the **Adapter Instance** tab, from the **Adapter Instance** list, select your previously configured HTML form adapter. Click **Next**.
17. On the **Mapping Method** tab, leave the default selection, and then click **Next**.
18. On the **Attribute Contract Fulfillment** tab, from the **Source** list for **SAML_SUBJECT**, select **Adapter**.
19. From the **Value** list, depending on what the SP is expecting, select **mail** or **uid**.
20. Define any other mappings as needed. Click **Next**.

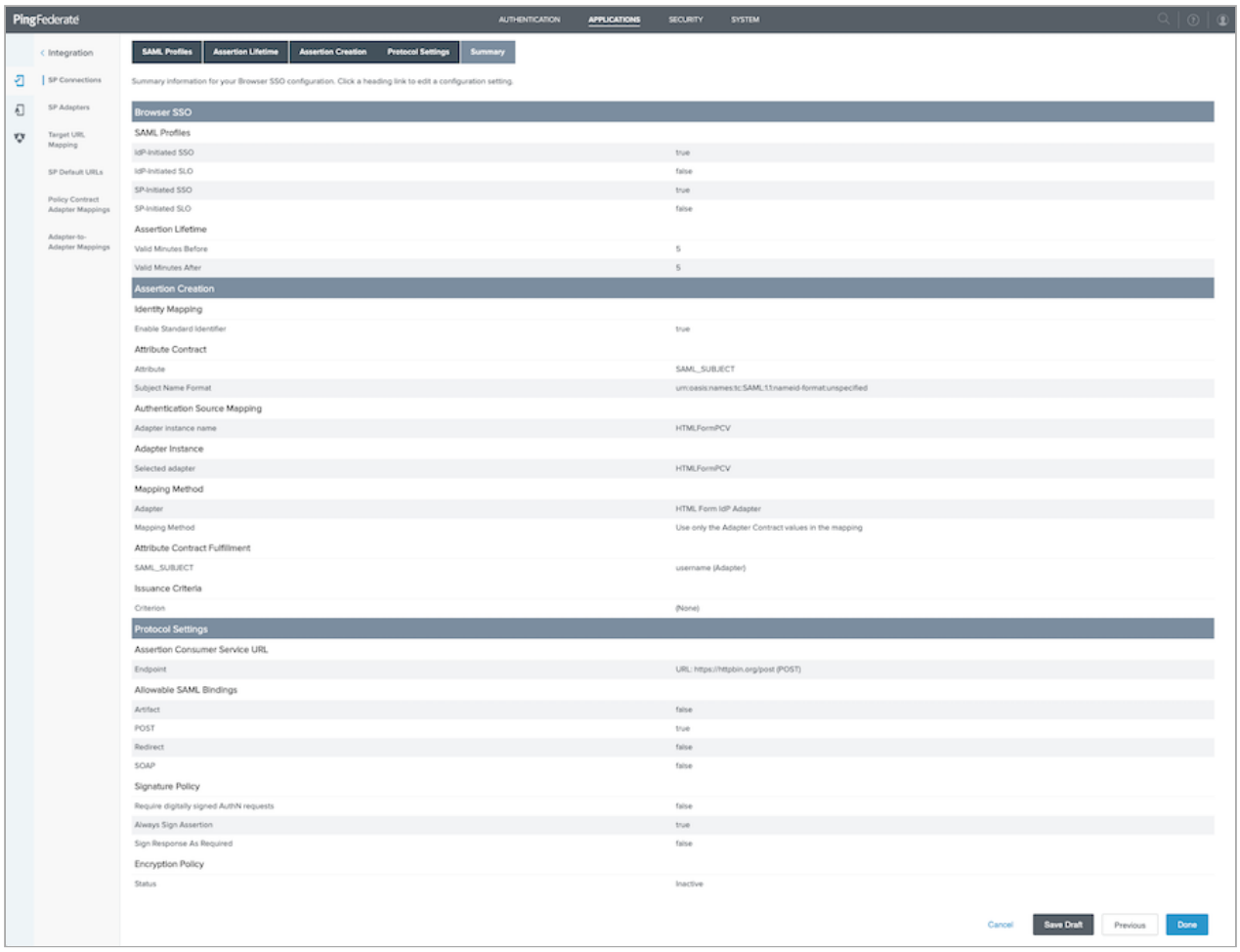
    You can leverage hard-coded “Text” for sending values to the SP connection.
21. On the **Issuance Criteria** tab, click **Next**.
22. On the **Summary** tab, review your entries, and then click **Done**.



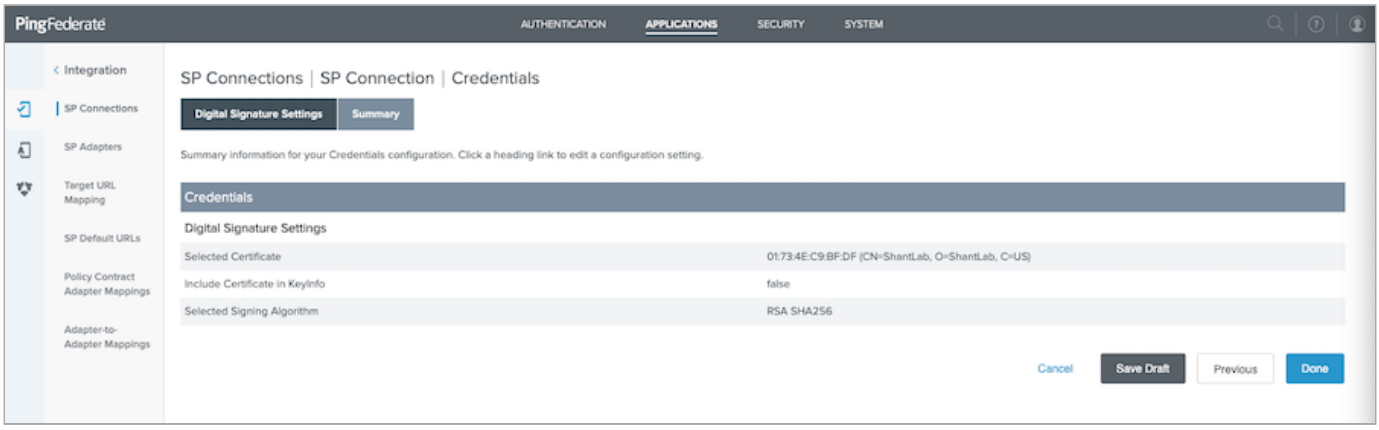
23. On the **Authentication Source Mapping** tab, click **Next**.
24. On the **Summary** tab, review your entries, and then click **Done**.
25. On the **Assertion Creation** tab, click **Next**.
26. On the **Protocol Settings** tab, click **Configure Protocol Settings**.
27. On the **Assertion Consumer Service URL** tab, ensure you see an entry for your SP based on the metadata that you uploaded. Click **Next**.
28. On the **Allowable SAML Bindings** tab, **POST** should be selected. Click **Next**.
29. On the **Signature Policy** tab, click **Always Sign the SAML Assertion**. Click **Next**.
30. On the **Encryption Policy** tab, click **None**. Click **Next**.
31. On the **Summary** tab, review your entries, and then click **Done**.



32. On the **Protocol Settings** tab, click **Next**.
33. On the **Summary** tab, review your entries, and then click **Done**.

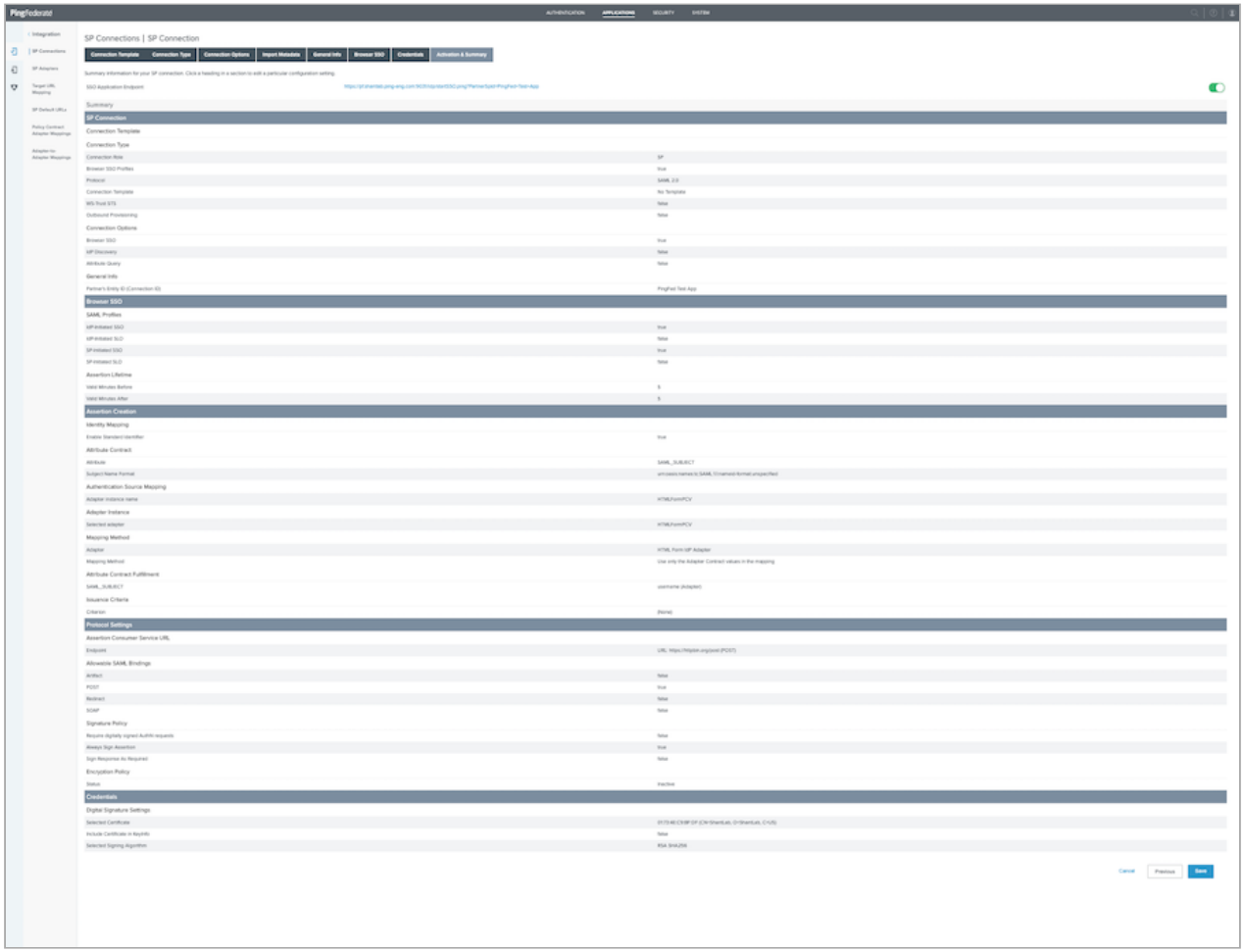


34. On the **Browser SSO** tab, click **Next**.
35. On the **Credentials** tab, click **Configure Credentials**.
36. On the **Digital Signature Settings** tab, from the **Signing Certificate** list, select your organization's default signing certificate that you previously created.
37. Select the **Include the Certificate in the Signature <KeyInfo> Element** check-box. Click **Next**.
38. On the **Summary** tab, review your entries, and then click **Done**.



39. On the **Credentials** tab, click **Next**.
40. On the **Activation & Summary** tab, click the toggle to enable the connection, and then scroll to the bottom and click **Save**.

    The connection status is enabled when the toggle is green. You must click **Save** or your work will be lost.



===Next steps

Click on the SP connection that you just created and copy the **SSO-URL** link. Start a private browsing session and test your connection using the **SSO-URL** link.

Configuring a SAML application in PingOne

Configure a SAML application in PingOne.

About this task

In the following configuration, values will vary depending on the identity provider (IdP) requirements.

> **Note**
>
> Some application settings can only be configured after the application is created. Learn more in Editing an application.

Steps

1. Go to **Applications** > **Applications**.
2. Click the + icon.
3. Create the application profile by entering the following:
   - **Application name**: A unique identifier for the application.
   - **Description** (optional): A brief characterization of the application.
   - **Icon** (optional): A graphic representation of the application. Use a file up to 1MB in JPG, JPEG, GIF, or PNG format.
4. For **Application Type**, select **SAML Application**.
5. Click **Configure** and specify the details of the connection between the application and PingOne.

   You can enter the values manually, or import them from a file or URL.

   Choose from:

   - Import the configuration details from an XML metadata file. Select **Import Metadata**. Click **Select a File** and then select an XML metadata file on your file system. Click **Open**.

     The configuration values are populated based on the information in the metadata file.

     > **Note**
     >
     > If the metadata file does not specify all the configuration values, you must enter the missing values manually.

   - Import the configuration details from a metadata URL. Select **Import from URL**. Enter the URL and then click **Import**.

     > **Note**
     >
     > The URL must be a valid absolute URL.

     The configuration values are populated based on the information from the URL.

   - Enter the configuration details manually. In the **ACS URLs** field, enter the Assertion Consumer Service (ACS) URLs. You must specify at least one URL, and the first URL in the list is used as the default.

In the **Entity ID** field, enter the service provider entity ID used to look up the application. The Entity ID is a required property and is unique within the environment.

6. Click **Save**.

Next steps

After the application is created, you can edit the application settings, configure application policies, and control application access. Learn more in Editing an application - SAML, Applying authentication policies to an application, and Application access control.

Configuring a SAML application in PingOne for Enterprise

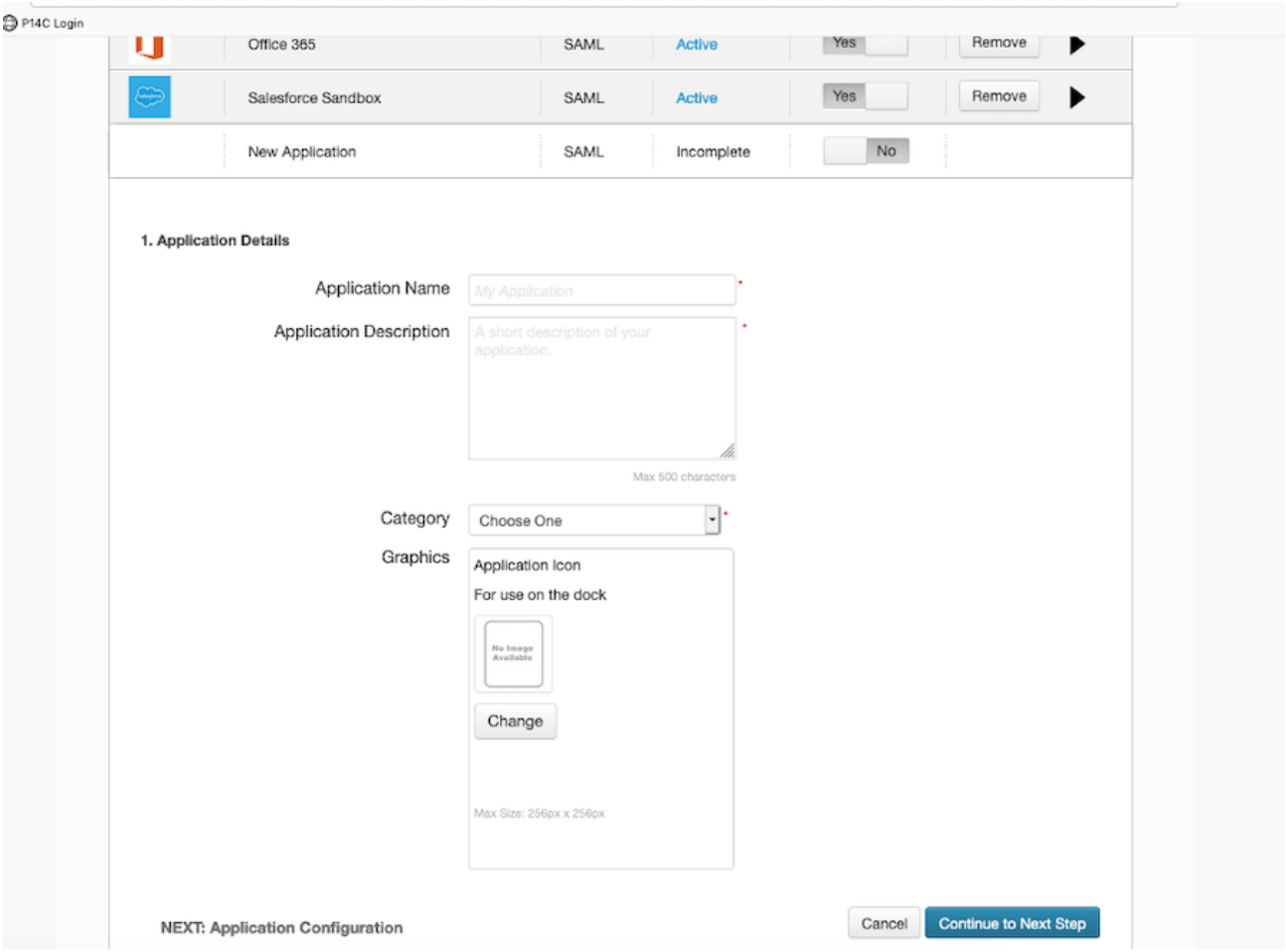
Configure a SAML application in PingOne for Enterprise.

Before you begin

If you do not have the service provider's (SP) single sign-on (SSO) URL for the application, generally a SAML application that already exists in your organization, you must configure the necessary SAML settings for the application to add it to PingOne for Enterprise.

Steps

1. In the PingOne for Enterprise dashboard, go to **Applications > My Applications > SAML**.
2. Click **Add Applications > New SAML Application**.
3. In the **Application Details** section, complete the following required fields:
    - **Application Name**
    - **Application Description**
    - **Category**



4. Click **Continue to Next Step**.
5. In the **Application Configuration** section, provide the SAML configuration details for the application.
    1. From the **Signing Certificate** list, select the signing certification you want to use.
    2. In the **SAML Metadata**field, click **Download** to retrieve the SAML metadata for PingOne for Enterprise.

       This supplies the PingOne for Enterprise connection information to the application.
    3. In the **Protocol Version** field, select the SAML protocol version appropriate for your application.
    4. In the **Upload Metadata** section, click **Choose File** to upload the application's metadata file.

       > **Note**
       >
       > The **ACS URL** and **Entity ID** will then be supplied for you. If you don't upload the application metadata, you'll need to enter this information manually. When manually assigning an entity ID, the value must be unique unless you are assigning the entity ID value for a private managed application, an application that is supplied and configured by a PingOne for Enterprise administrator, rather than an SP. When applications are supplied by an SP, entity ID values are required to be unique to ensure against possible identifier conflicts with the IdP ID for the application.

    5. In the **Application URL** field, enter an appropriate URL.

This is required by some applications as the target URL. It is used in IdP-initiated SSO for a deep-linking purpose. The application URL is passed in the RelayState parameter by the IdP.

6. In the **Single Logout Endpoint** field, enter the URL to which the service will send the SAML single logout (SLO) request using the **Single Logout Binding Type** that you select.
7. In the **Single Logout Response Endpoint** field, enter the URL to which your service sends the SLO response.
8. In the **Single Logout Binding Type** field, select the binding type, **Redirect** or **POST**, to use for SLO.
9. In the **Primary Verification Certificate** field, click **Choose File** to upload the primary public verification certificate to use for verifying the SP signatures on SLO requests and responses.
10. In the **Secondary Verification Certificate** field, click **Choose File** to upload the secondary verification certificate if available.

    The secondary verification certificate is used if the primary verification certificate fails to validate a signature.
11. Select the **Encryption Assertion** checkbox.

    If selected, the assertions PingOne for Enterprise sent to the SP for a multiplexed application are encrypted. You can also use this option for your managed applications. Available for SAML 2.0 applications only.

    Selecting this option displays the information needed to encrypt the assertion:

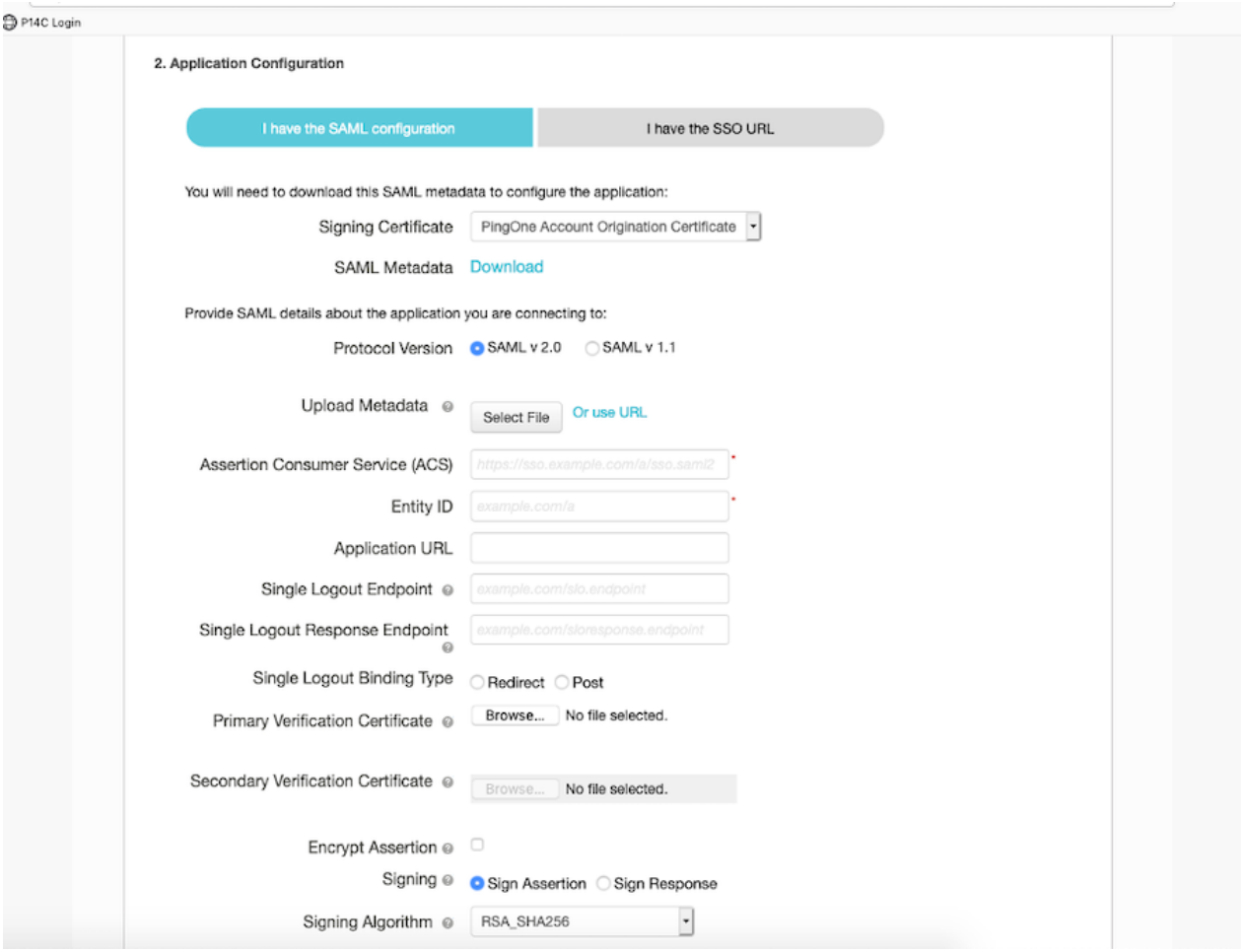
    - **Encryption Certificate**: Upload the certificate to use to encrypt the assertions.
    - **Encryption Algorithm**: Choose the algorithm to use for encrypting the assertions. We recommend **AES_256** (the default), but you can select **AES_128** instead.
    - **Transport Algorithm**: The algorithm used for securely transporting the encryption key. Currently, **RSA-OAEP** is the only transport algorithm supported.

    > **Note**
    >
    > If an encryption certificate is included in the metadata you upload, this option is automatically enabled. The entry for **Encryption Certificate** shows the name of the certificate and the entry for **Encryption Algorithm** is set to **AES_256**.
12. In the **Signing** field, select either to sign the SAML assertion or to sign the SAML response.

    If the **Encryption Assertion** checkbox has been selected, choose to sign the response. This provides a significant increase in security.
13. In the **Signing Algorithm** list, select the desired algorithm or use the default value.
14. Select the **Force Re-authentication** checkbox.

    If selected, users having a current, active SSO session will be re-authenticated by the identity bridge to establish a connection to this application.
15. Select the **Force MFA** checkbox.

    If selected, users are required to use multi-factor authentication (MFA) as defined by your authentication policy each policy each time they access the application. You'll need to have an authentication policy in place to use this setting. Learn more in Create or update an authentication policy.



6. Depending on your requirements, complete the remaining entry fields. Click **Continue to Next Step**.

   The remaining entry fields are optional, depending on your requirements.

7. In the **SSO Attribute Mapping** section, modify or add any attribute mappings as necessary for the application.

   In most cases, the default attribute mappings are sufficient. These mappings assign your identity repository attributes to the attributes provided by the SP for the application. For each application attribute, you can:

   - Click the **Required** checkbox to designate an attribute or attributes as required by the application.
   - In the **Application Attribute** field, enter an identity repository attribute.
   - In the **Identity Bridge Attribute or Literal Value** field, select an identity repository attribute from the list.
   - Select the **As Literal** checkbox, and then enter a literal value to assign.
   - Click **Advanced**, and then enter any additional attributes required by the application. You then have all of the choices above when configuring the attribute.

8. When finished modifying or adding any additional attributes, click **Continue to Next Step**.

9. In the **Group Access** section, make the new application available to your users by assigning the groups authorized to use the application.

   1. Click **Add** for each group you want to authorize to use the application.

All members of the selected group or groups will be able to use the application. When the application supports user provisioning, user provisioning to this application is also enabled for members of the assigned groups.

10. Click **Continue to Next Step**.
11. In the **Review Setup** section, review the application connection information.

    Some of this information might be needed by the SP to complete the SSO configuration for the application. In particular, you can download the PingOne for Enterprise signing certificate or the PingOne for Enterprise SAML metadata, which has the certificate embedded.

12. **Optional:** To change any of the configuration settings, click **Edit**.
13. Click **Finish**.

Result

The new SAML application is added to your **My Applications** list. Go to **Users → User Groups** to see the application you've added is now authorized for use by the selected group or groups.

Connecting PingFederate to a Microsoft SQL JDBC datastore with Windows authentication

Create a Microsoft SQL server Java Database Connectivity (JDBC)-connected datastore in PingFederate and configure it for Windows authentication.

If your organization primarily uses a Microsoft Windows platform, you can have your PingFederate nodes on Windows servers, and you can use Microsoft SQL Server for your databases. One example use case for this type of datastore is storing OAuth grants in a clustered environment.

High availability requirements for this database should follow your organization's procedures and are outside the scope of this document. Any database maintenance tasks are also not addressed in this document.

Component

PingFederate 9.3 or later

Before you begin

You must have:

- An SQL server on the network, accessible from the PingFederate nodes on its assigned port

  > **Note**
  >
  > Port 1433 is the default port for SQL server. You can test connectivity to the `server:port` with the telnet command line utility.

- Access to a database on the server with the correct tables

  Work with the database administrator to determine an appropriate name for your database, such as "PingFederate".

> **Note**
>
> For storing OAuth grants, you can find the table creation scripts (access-grant-sqlserver.sql and access-grant-attribute-sqlserver.sql) in `<pf_install>/pingfederate/server/default/conf/access-grant/sql-scripts` .

- A user account in the Active Directory (AD) domain you can use as a service account

  It does not need any special domain privileges, but it receives local permissions on your PingFederate nodes.

  Work with your database administrators to ensure the user account in the AD has permissions to access and write to the database.

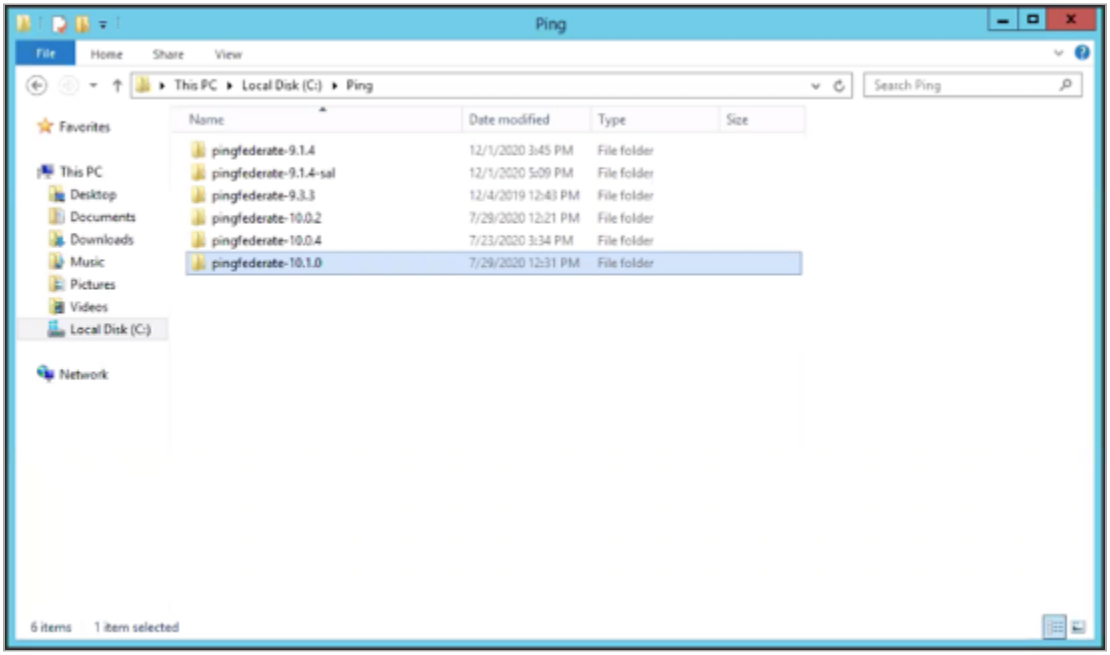
Adding a new user

About this task

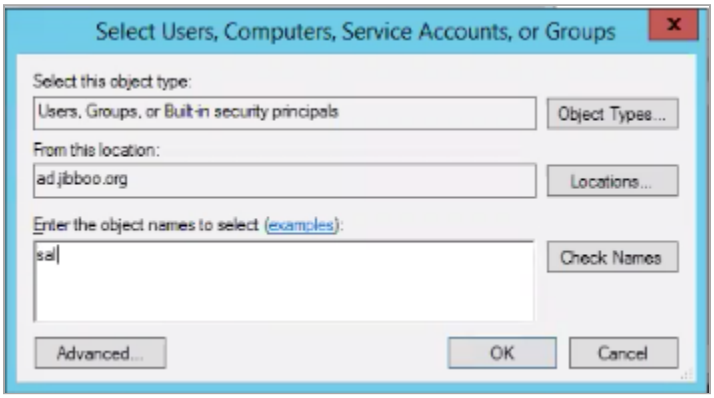
Perform these steps on each PingFederate node.

Steps

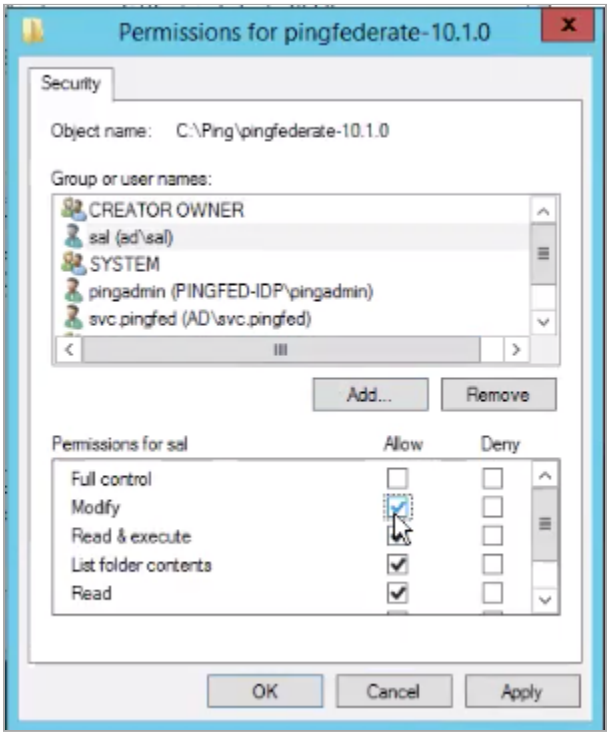
1. On your Windows machine, open the folder where PingFederate is installed.



2. Right-click the folder and select **Properties**.
3. Click the **Security** tab.
4. To add a user, click **Edit** and **Add**.
5. Add the user account and click **OK**.



6. From the user account **Permissions** section, select the **Modify** checkbox.



7. To save your changes, click **Apply**.

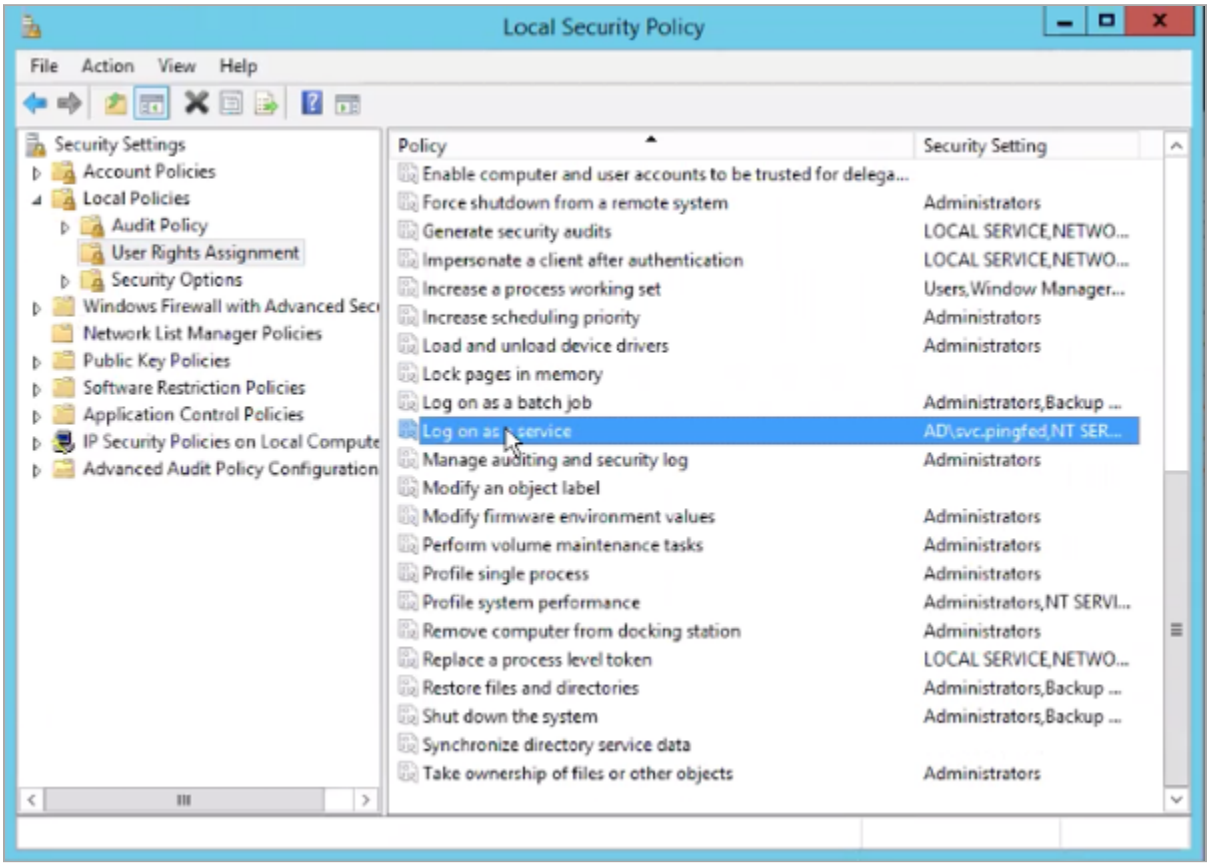
Assigning the log on as a service policy to the new user

About this task

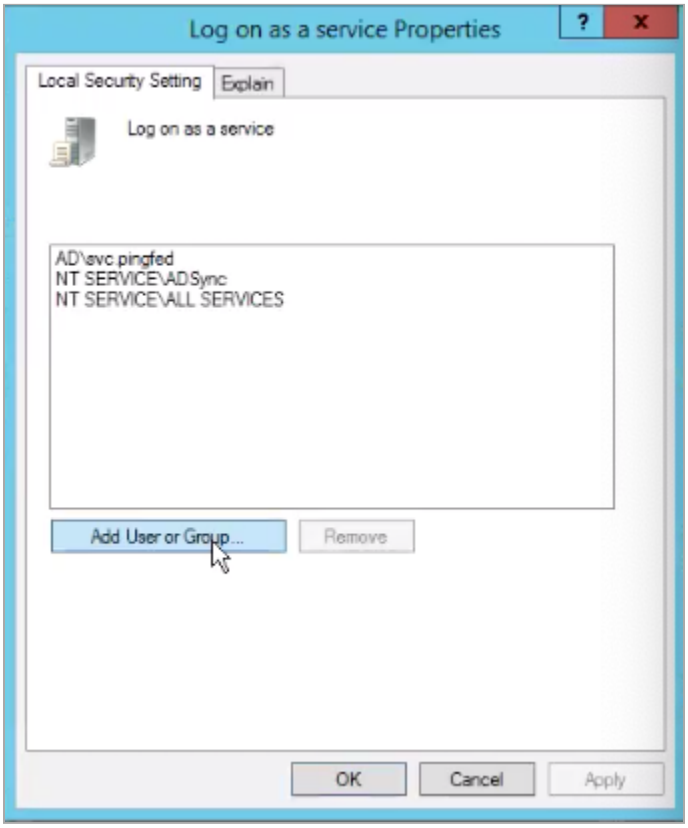
Perform these steps on each PingFederate node.

Steps

1. On your Windows machine, open the **Local Security Policy** menu. You can search for "local security policy" from the Windows Start menu.
2. From the **Local Security Policy** window, go to **Local Policies > User Rights Assignment > Log on as a service**.



3. In the **Log on as a service Properties** window, click **Add User or Group...**.



4. Add the user information as needed.
5. Click **Apply**, and then click **OK**.

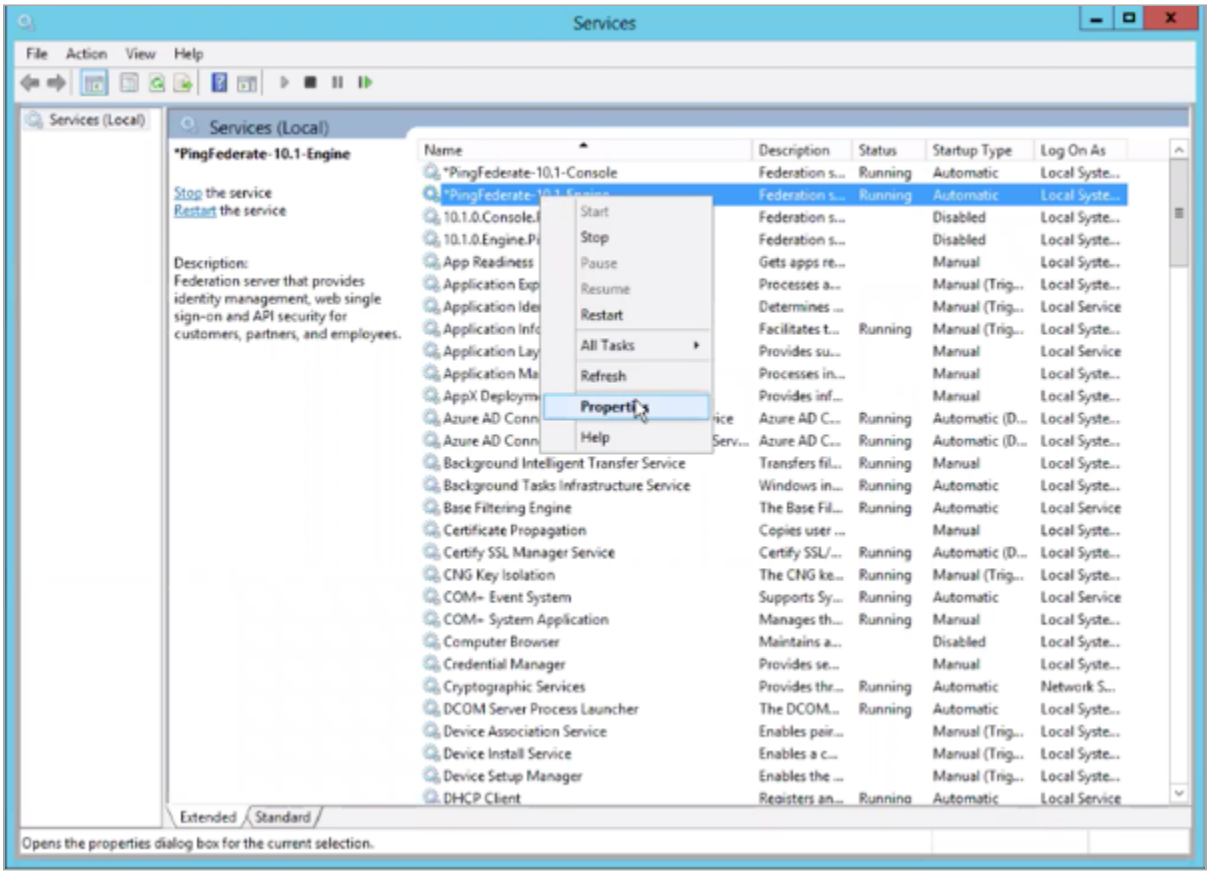
Editing the sign-on tab for the PingFederate service

About this task

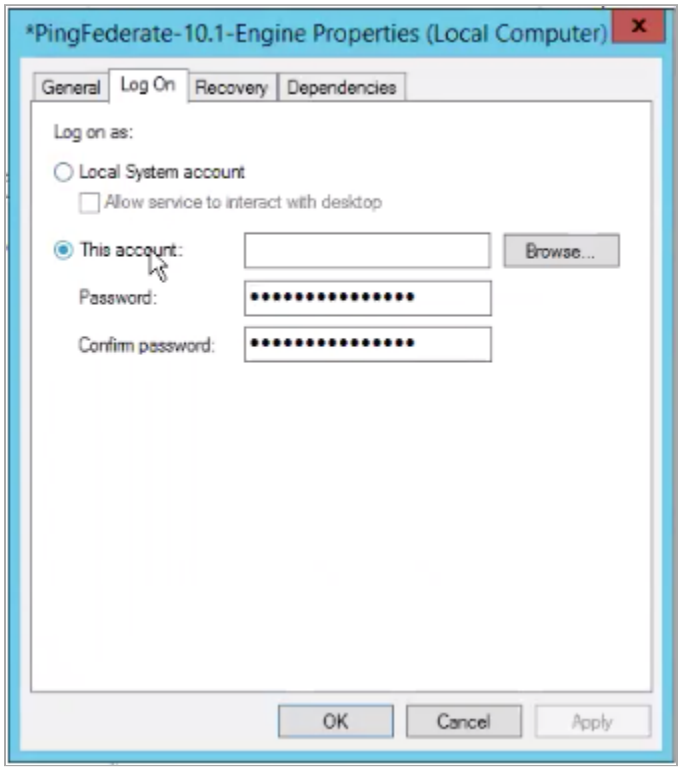
Perform these steps on each PingFederate node.

Steps

1. On your Windows machine, open the **Services** menu.
2. Right-click the PingFederate service that is running on your machine and select **Properties**.



3. On the **Log On** tab, click **This account**.

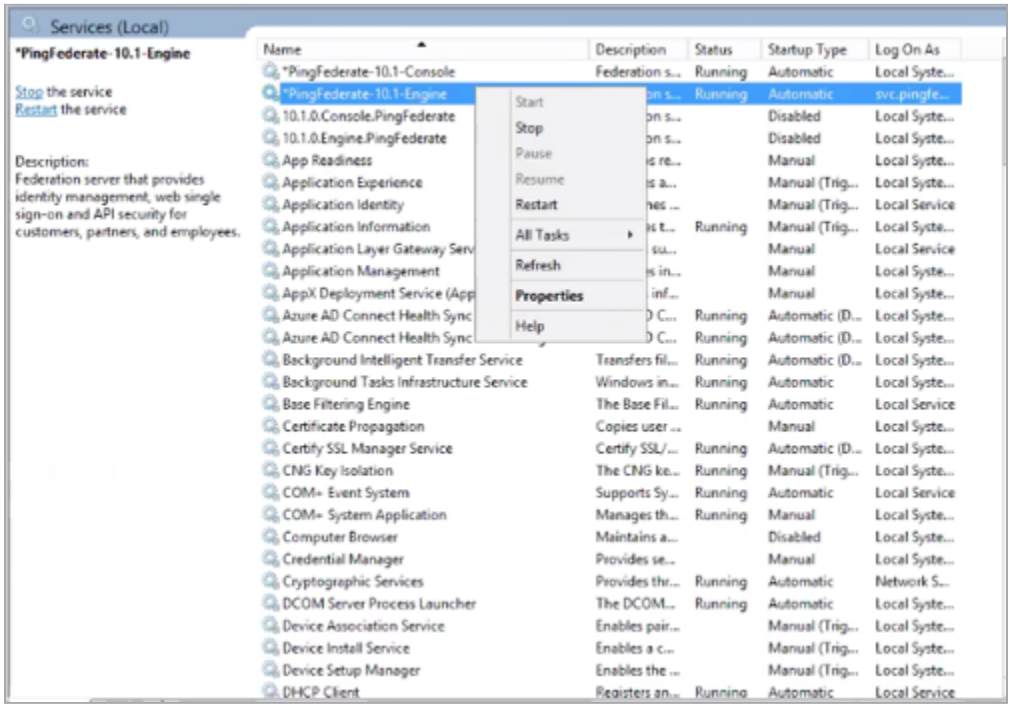


4. In the **This account** field, enter the entire UPN name of the account, such as `svc.pingfed@<your-domain>`.

5. Enter a password for the account.

6. To save your changes, click **Apply**, and then click **OK**.

7. To restart the service, right-click the PingFederate service and select **Restart**.



Deploying the required JDBC driver files and DLLs

About this task

Perform these steps on each PingFederate node.

Steps

1. Go to .microsoft.com/en-us/sql/connect/jdbc/release-notes-for-the-jdbc-driver?view=sql-server-ver15//[Microsoft release notes] and download the correct `.zip` file for your Java version.

   Package 6.4 is known to work properly.

2. Extract the files and find the `.jar` file that corresponds to your Java version.
3. Place the `.jar` file in `<pf_install>/pingfederate/server/default/lib`
4. In the `auth` folder of your JDBC driver download folder, find the appropriate folder for your Java virtual machine (JVM) version (x32 or x64-based), and copy the DLL file to your `/windows/system32` folder.
5. Restart the PingFederate service.

Creating the JDBC datastore connection in PingFederate

Steps

1. From the PingFederate administrative console, go to **System > Data Stores** and click **Add New Data Store**.
2. On the **Data Store Type** tab, in the **Name** field, enter a name.
3. In the **Type** list, select **Database (JDBC)**. Click **Next**.
4. On the **Database Config** tab, in the **JDBC URL** field enter `jdbc:sqlserver://<databaseservername>;<databaseName=databasename>;integratedSecurity=true`.
5. Click **Add**.

   > **Note**
   >
   > Use the fully qualified domain name for your server. Port `1433` is the default port for SQL server. If you are using port `1433`, then you can omit it from the JDBC URL. For any non-standard ports, specify them in the URL. `integratedSecurity=true` makes the connection use Windows authentication. Without that, it attempts SQL authentication.

6. In the **Driver Class** field, enter `com.microsoft.sqlserver.jdbc.SQLServerDriver`.
7. Complete the **Username** and **Password** fields with the same service account credentials from step 1.
8. In the **Validate Connection SQL** field, enter `SELECT getdate()`.

   > **Note**
   >
   > `SELECT getdate()` is used to re-establish the JDBC connection if it gets disconnected.

9. Click **Next**, and then click **Save**.

10. Replicate the cluster configuration.

Testing the newly created external datastore

Steps

1. Follow the steps in Configure external databases for grant storage.
2. Issue a grant in PingFederate.

> **Note**
>
> If you do not have an OAuth client application readily available, you can use the OAuth Playground authorization code flow to obtain a code and exchange it for an access and refresh token.

3. Work with the database administrator (if necessary) to view the tables in the database that were created by the script.

   Result:

   You should see an entry for the newly issued grant.

Connecting PingFederate with Yahoo through OIDC

Learn how to connect PingFederate with your Yahoo developer account using OpenID Connect (OIDC).

> **Note**
>
> Yahoo no longer supports OpenID2 and migrated to OIDC.

Component

PingFederate 10.3

Creating an OIDC app in your Yahoo developer account

Before you begin

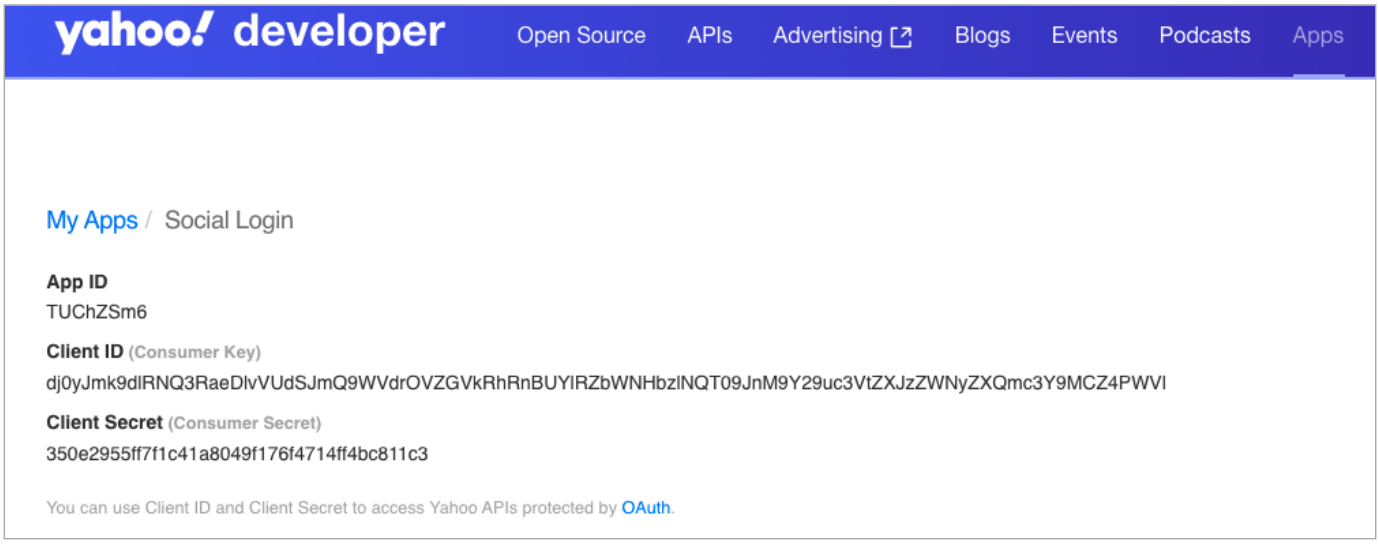
- Go to developer.yahoo.com and create a developer account.

About this task

In your Yahoo developer account, create an OIDC app and obtain the **Client ID** and **Client Secret**.

Steps

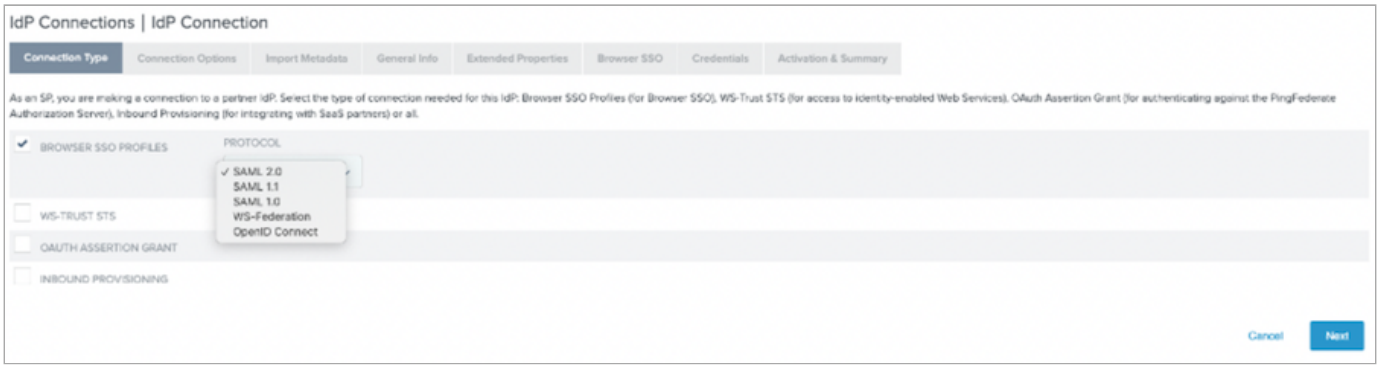
1. Sign on to your Yahoo developer account and go to **Apps > Create an App**.
2. Create an application with OpenID Connect permissions.
3. Copy the **Client ID** and **Client Secret**.



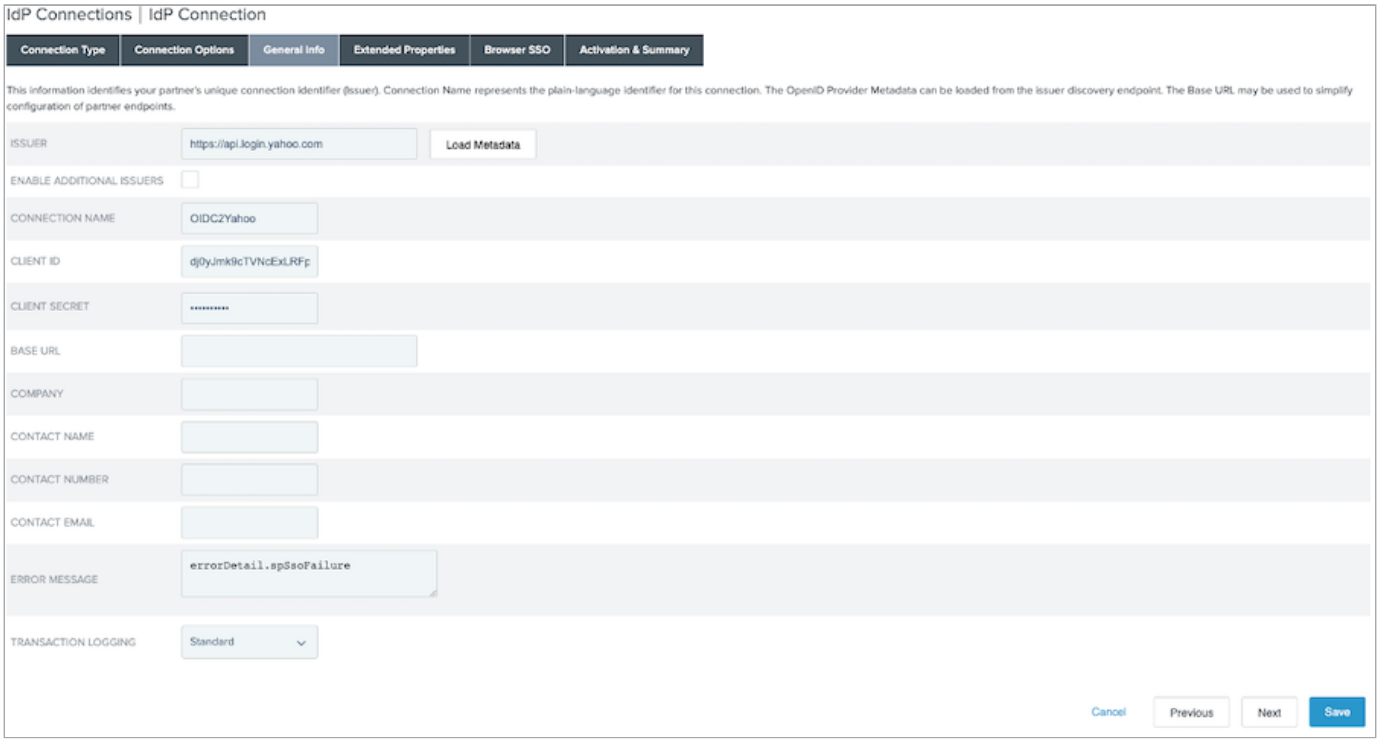
Creating an OIDC type IdP connection

Steps

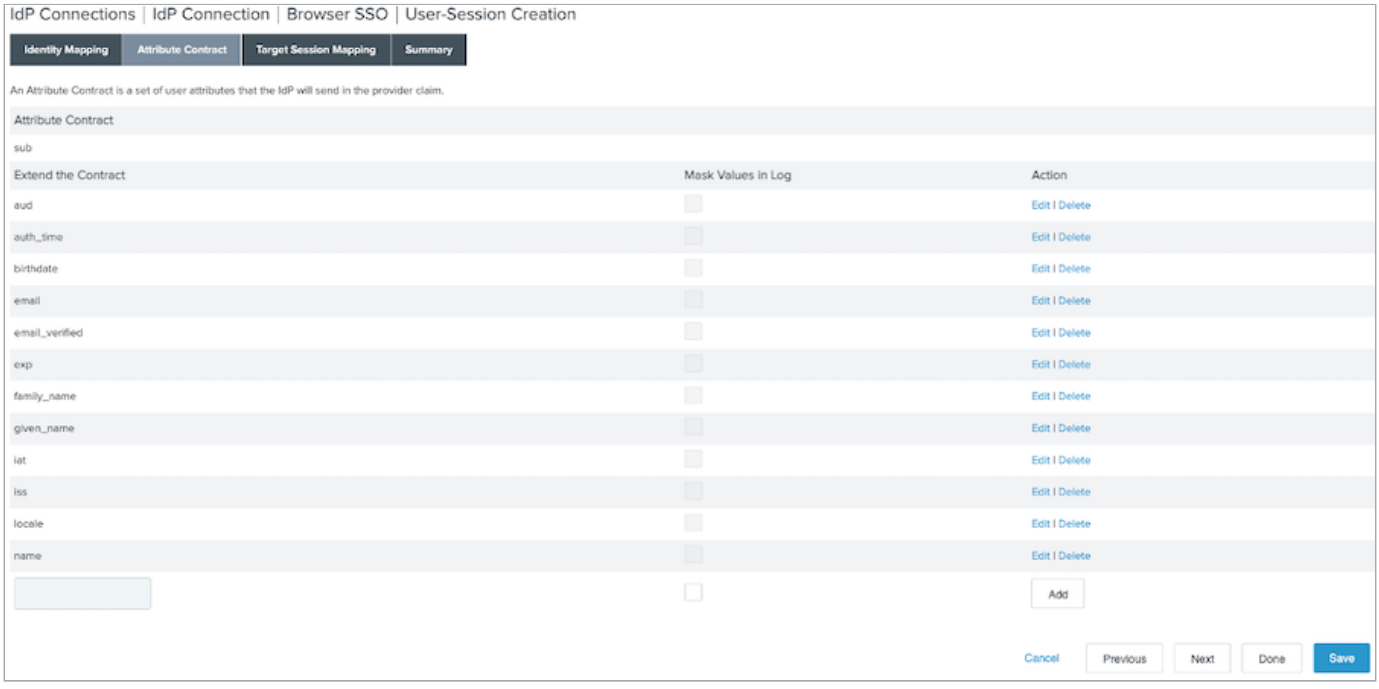
1. Sign on to PingFederate and go to **Authentication > Authorization > IdP Connections**. Click **Create Connection**.
2. On the **Connection Type** tab, select the **Browser SSO** checkbox, and in the **Protocol** list, select **SAML 2.0**. Click **Next**.



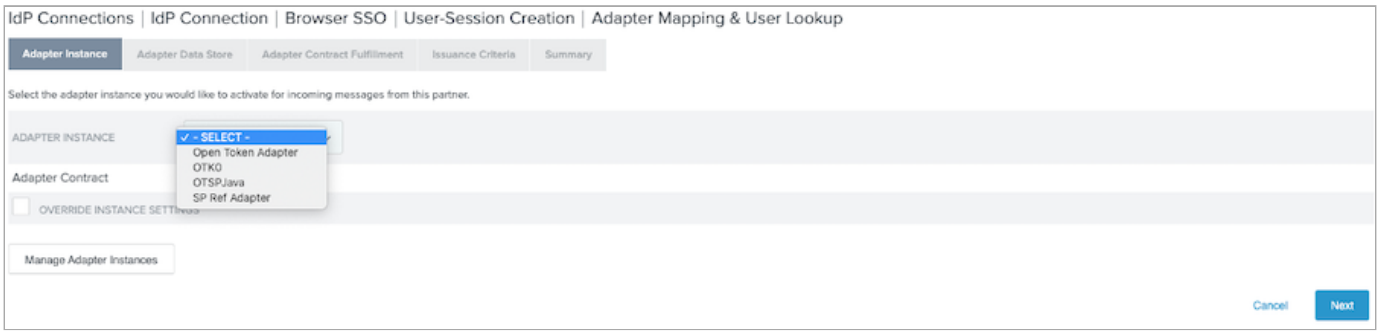
3. On the **Connection Options** tab, select the **Browser SSO** checkbox. Click **Next**.
4. On the **General Info** tab, in the **Issuer** field, enter `https://api.login.yahoo.com`.
5. In the **Client ID** and **Client Secret** fields, enter the values copied earlier from your Yahoo OIDC app.
6. Click **Load Metadata**. Click **Next**.



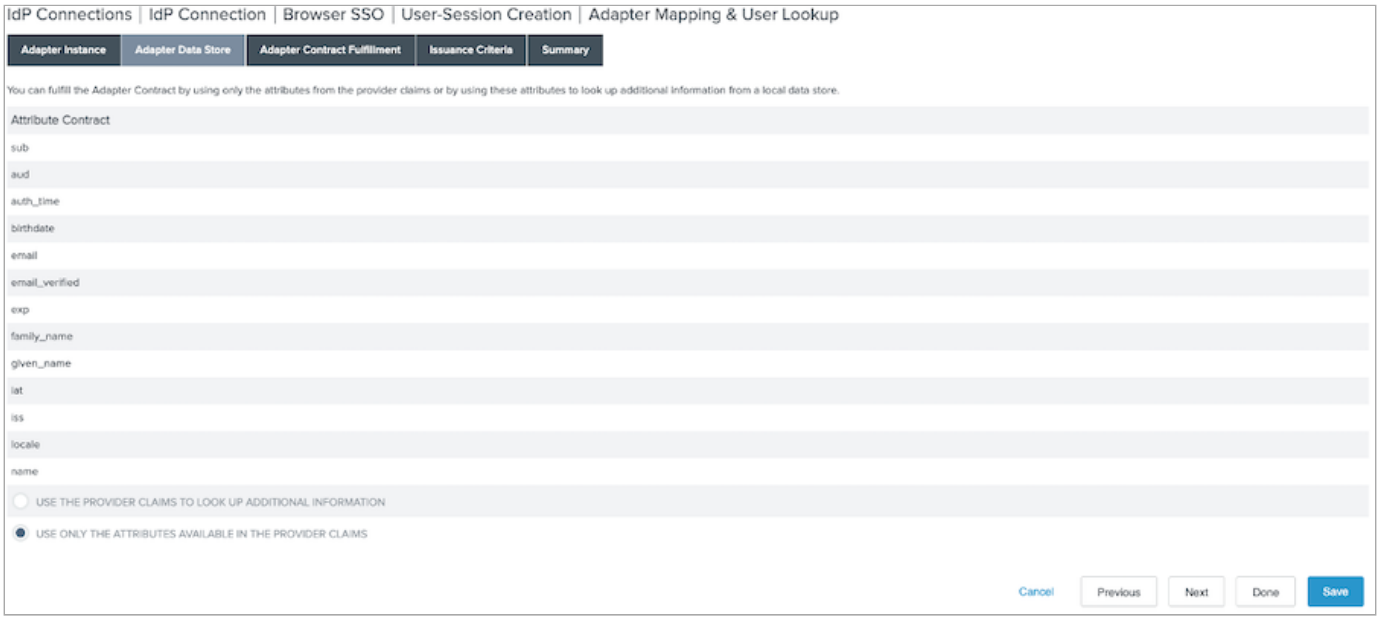
7. On the **Extended Properties** tab, click **Next**.
8. On the **Browser SSO** tab, click **Configure Browser SSO**.
9. On the **User Session Creation** tab, click **Configure User-Session Creation**.
10. On the **Identity Mapping** tab, select **Account Mapping**. Click **Next**.
11. On the **Attribute Contract** tab, leave the default values selected. Click **Next**.



12. On the **Target Session Mapping** tab, click **Map New Adapter Instance**.
13. On the **Adapter Instance** tab, in the **Adapter Instance** list, select **Open Token** adapter. Click **Next**.

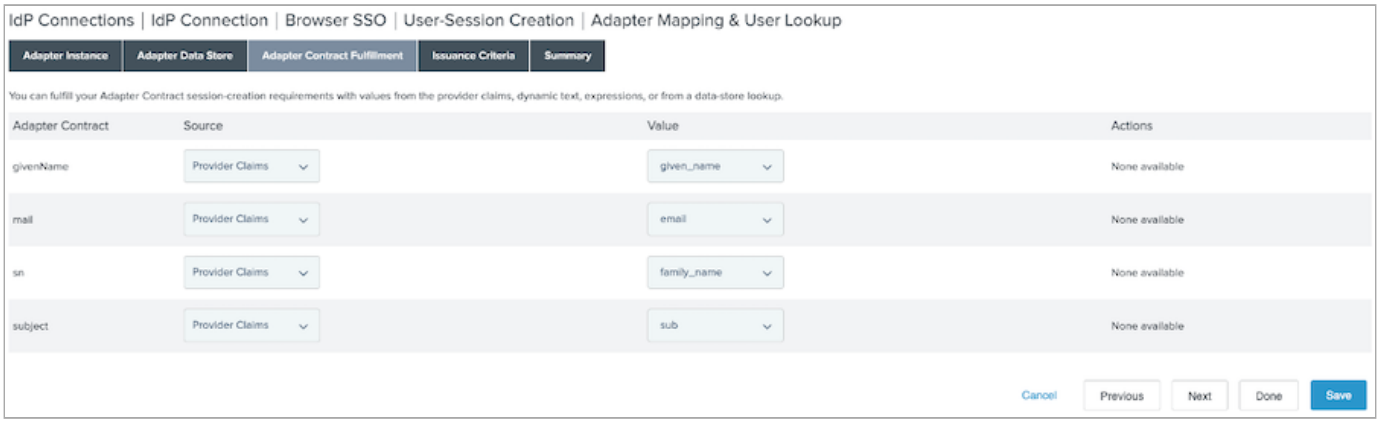


14. On the **Attribute Data Store** tab, leave the default values selected. Click **Next**.

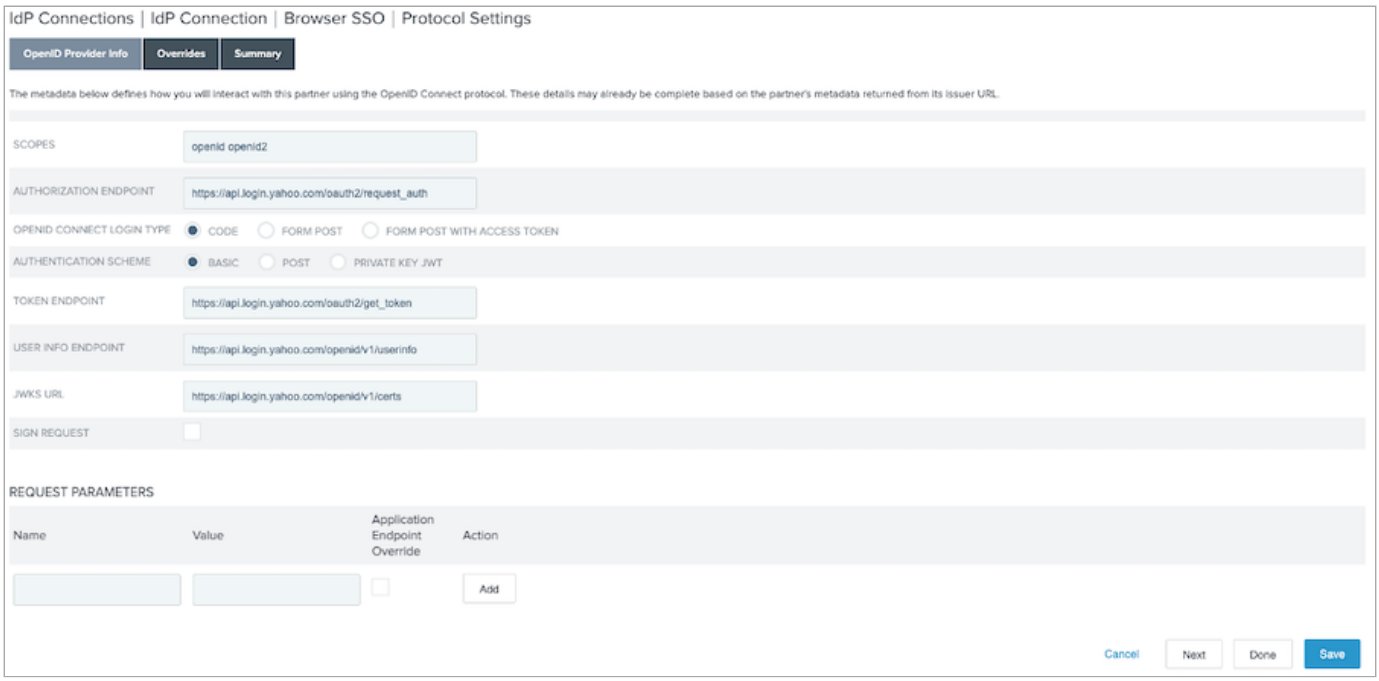


15. On the **Adapter Contract Fulfillment** tab, map the values as follows. Click **Next**.

Attribute	Source	Value
givenName	Provider Claims	given_name
mail	Provider Claims	email
sn	Provider Claims	family_name
subject	Provider Claims	sub



16. On the **Issuance Criteria** tab, click **Next**.
17. On the **Summary** tab, review your entries and click **Done**.
18. On the **User Session Creation** tab, click **Next**.
19. On the **Protocol Settings** tab, click **Configure Protocol Settings**.
20. On the **OpenID Provider Info** tab, review the information and click **Next**.

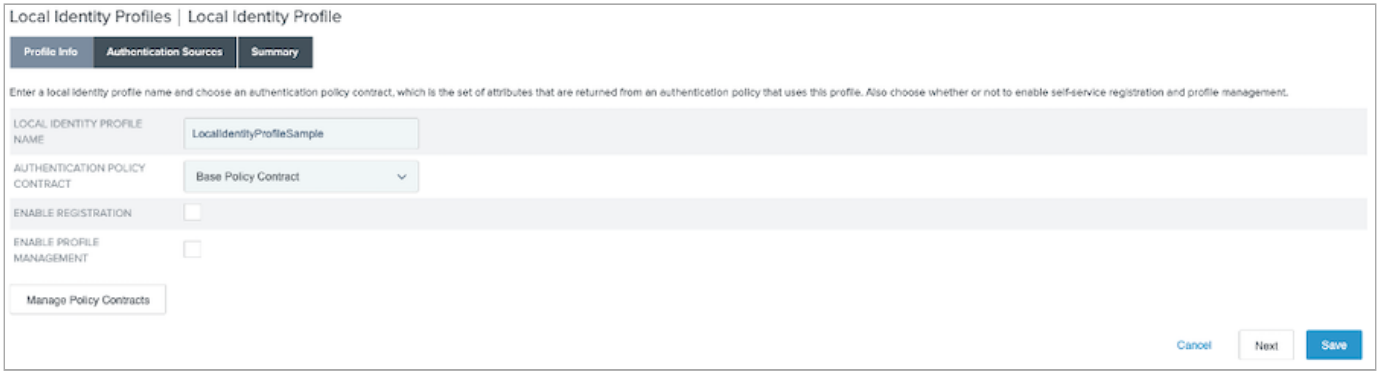


21. On the **Overrides** tab, enter a **Default Target URL**. Click **Next**.
22. On the **Summary** tab, review your entries and click **Done**.
23. On the **Protocol Settings** tab, click **Next**.
24. On the **Summary** tab, review your entries and click **Done**.
25. On the **Activation and Summary** tab, click the toggle to activate the connection. Click **Save**.

Creating a local identity profile

Steps

1. Go to **Authentication > Policies > Local Identity Profiles** and click **Create New Profile**.
2. On the **Profile Info** tab, choose an existing policy contract or create a new one. Click **Next**.



3. On the **Authentication Sources** tab, in the empty field next to the **Add** button, enter `Yahoo`. Click **Add**.

4. Click **Save**.

Creating an HTML form IdP adapter

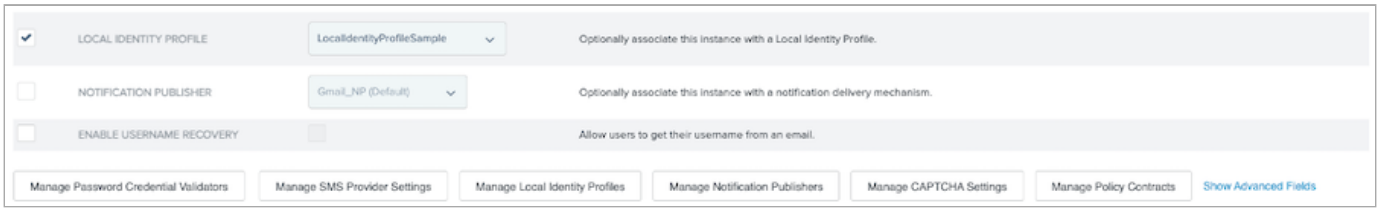
About this task

Create an HTML form IdP adapter to include the newly created LIP.

Steps

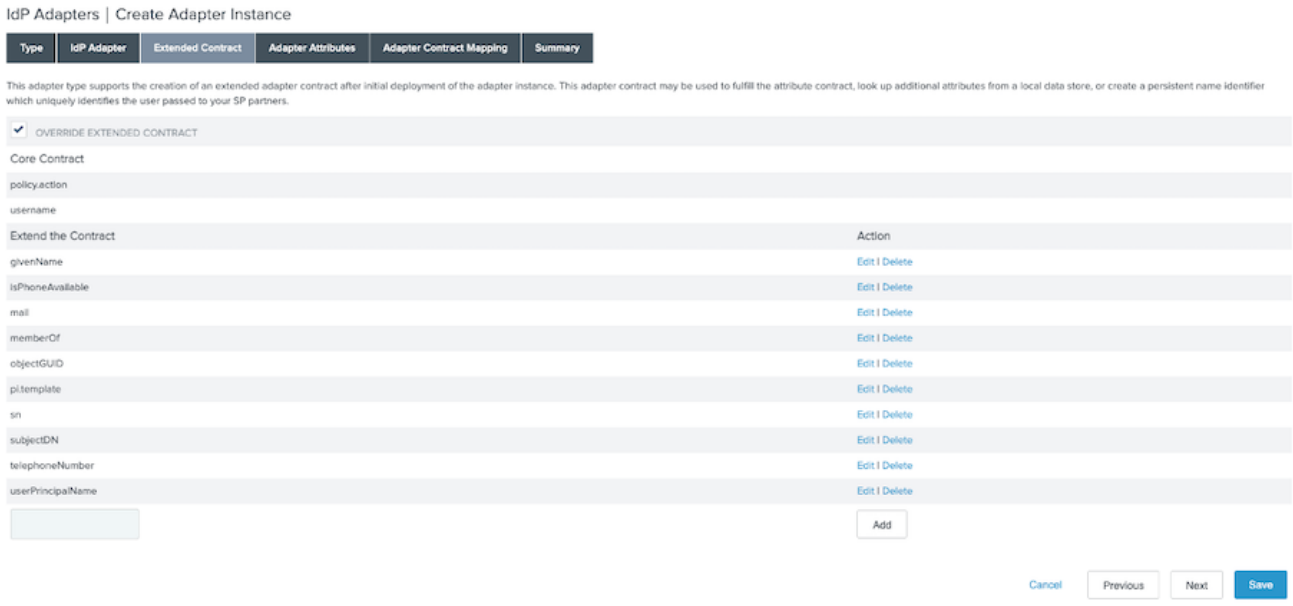
1. Go to **Authentication > Integration > IdP Adapters** and click **Create New Instance**.
2. On the **Type** tab, enter a **Instance Name** and **Instance ID**, and in the **Type** list, select **HTML From IdP Adapter**. Click **Next**.

3. On the **IdP Adapter** tab, select the **Local Identity Profile** checkbox and select the newly-created LIP in the list. Click **Next**.

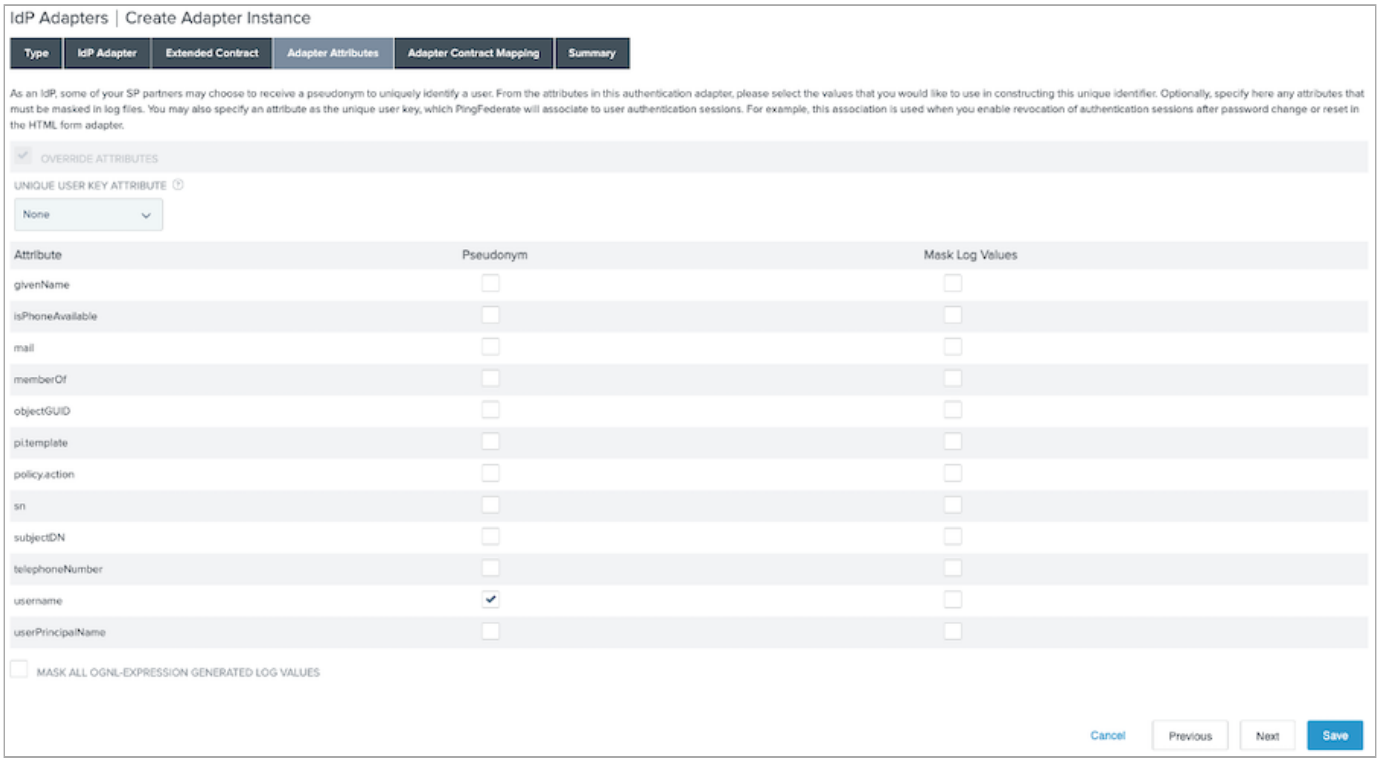


4. On the **Extended Contract** tab, add all desired attributes. Click **Next**.

    1. To add an attribute, enter the name in the empty field and click **Add**.



5. On the **Adapter Attributes tab**, in the **username** row, select the **Pseudonym** checkbox. Click **Next**.



6. On the **Adapter Contract Fulfillment** tab, configure the contract as follows. Click **Next**.

Attribute	Value
IsPhoneAvailable	`#this.get("telephoneNumber")== null? false:#this.get("telephoneNumber").toString().equalsIgnoreCase("")?false:true`
telephoneNumber	`telephoneNumber`
mail	`mail`
policy.action	`policy.action`
givenName	`givenName`
objectGUID	`objectGUID`
memberOf	`memberOf`
pi.template	`{ "name": "strong_authentication"."variables": { "logourl"."https//www.logosurfer.com/wp-content/uploads/2018/03/kohls-log_0.png"."currency": "USD"."recipient": "Charlie Parker" }}`
sn	`sn`
userPrincipalName	`userPrincipalName`

Attribute	Value
subjectDN	`subjectDN`
username	`username`

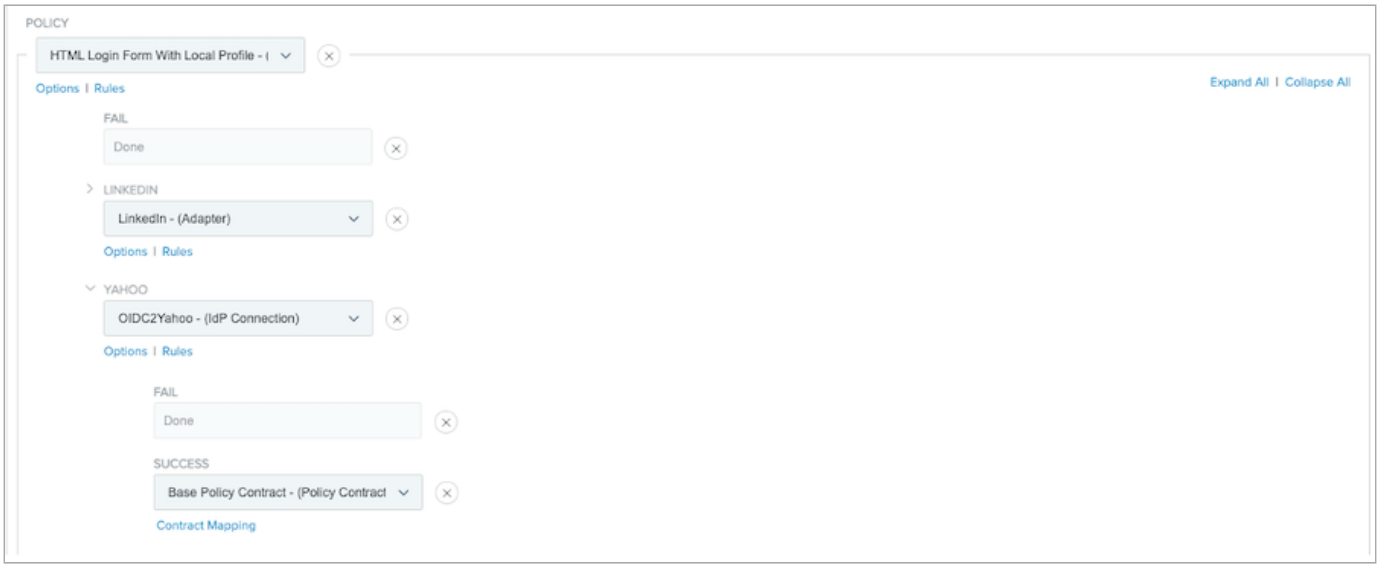
7. On the **Summary** tab, review your entries. Click **Save**.

Creating a policy to fulfill the policy contract chosen in the LIP

Steps

1. Select the HTML form adapter that you created earlier and click **Rules**.
2. Add **Yahoo** as a rule:
    1. From the **Attribute Name** list, select **policy.action**.
    2. From the **Condition** list, select **equal to**.
    3. In the **Value** field, enter `Yahoo`.
    4. In the **Result** field, enter `Yahoo`.
    5. Click **Done**.

    The rest of the values are optional.
3. Under the **Yahoo** branch, in the **Policy** list, select the IdP connection that you created earlier.
4. In the **Success** list, select the policy contract that you used in the LIP. Click **Contract Mapping**.



5. On the **Contact Fulfillment** tab, configure the following attributes.

Attribute	Value
UPN	`name`
Email	`email`
Group Membership	`grp`
Object GUID	`objectguid`
subject	`sub`
First Name	`given_name`
DN	`dn`
Last Name	`family_name`

6. On the **Summary** tab, click **Done**.

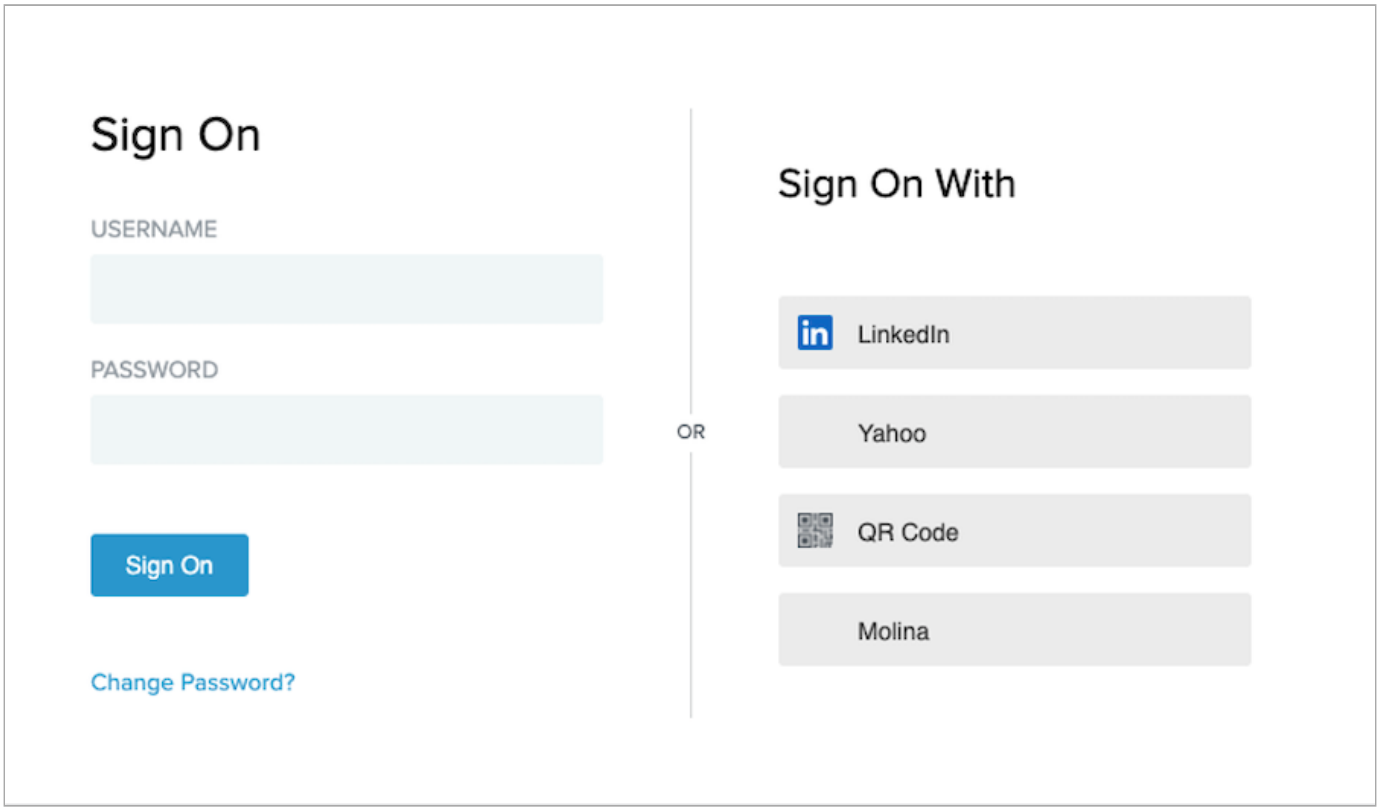
Testing the configuration

Steps

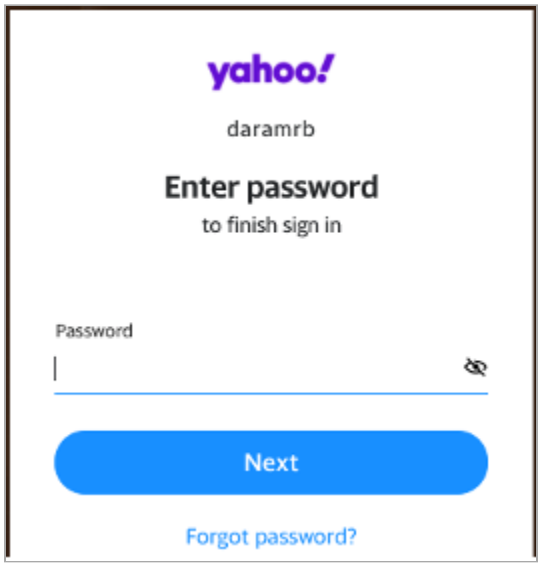
1. Launch an application that satisfies the newly-created policy.

   Result:

   A sign-on window opens.



2. In the **Sign On With** section, click **Yahoo** to go to the Yahoo sign-on page.



3. Enter your password and click **Next**.

Result

After signing on, you are taken to the end application.

Delegating all authentication to an external IdP

PingOne provides an authentication policy step that allows you to make an external identity provider (IdP) part of a PingOne authentication policy or delegate all authentication to that external IdP.

Before you begin

You must have:

- An external IdP defined in your PingOne tenant
- An authentication policy that specifies that external IdP as the only step

Configuring an external IdP

Before you begin

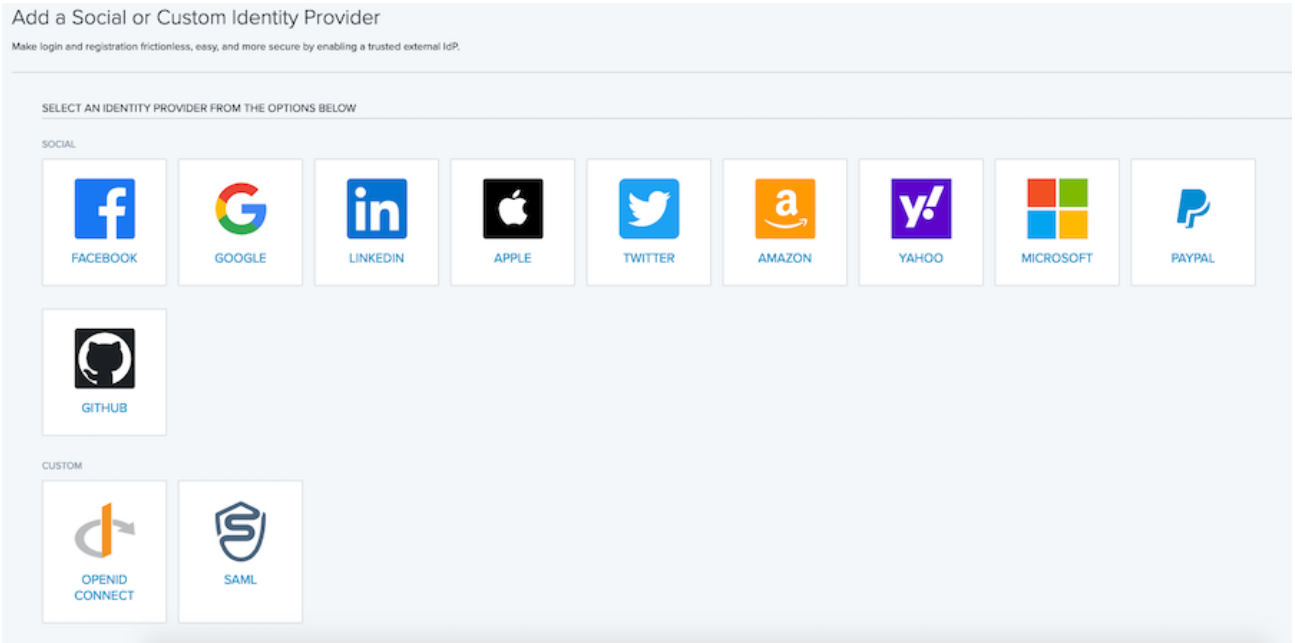
- If you want to use OpenID Connect (OIDC), you must configure an OIDC client in PingFederate.
- If you want to use SAML, you must configure a SAML service provider (SP) in PingFederate.

Steps

1. In your PingOne tenant, go to **Integrations > External IDPs** and click **Add Provider**.
2. Go to **Add a Social or Custom Identity Provider > Select an Identity Provider from the Options Below > Custom** and click either:

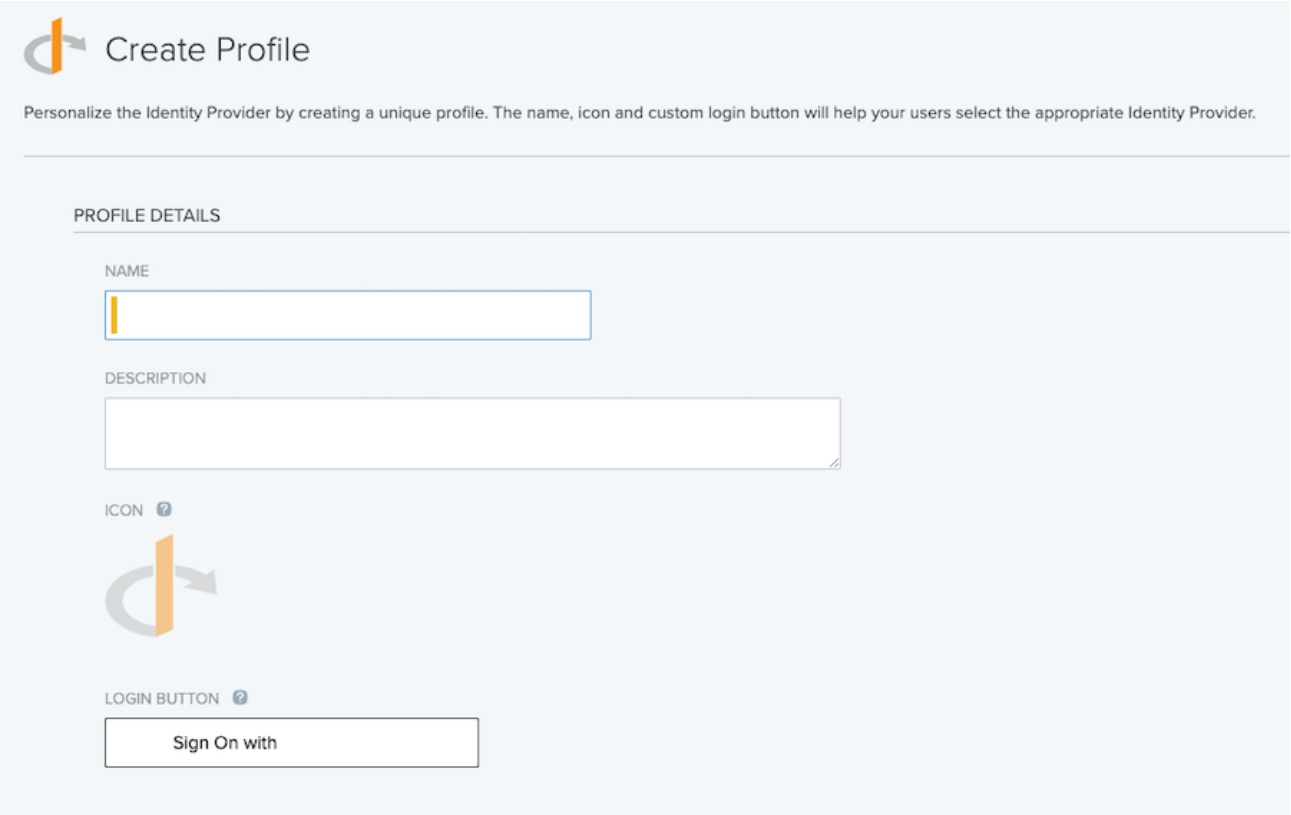
   Choose from:

   - **OpenID Connect**
   - **SAML**



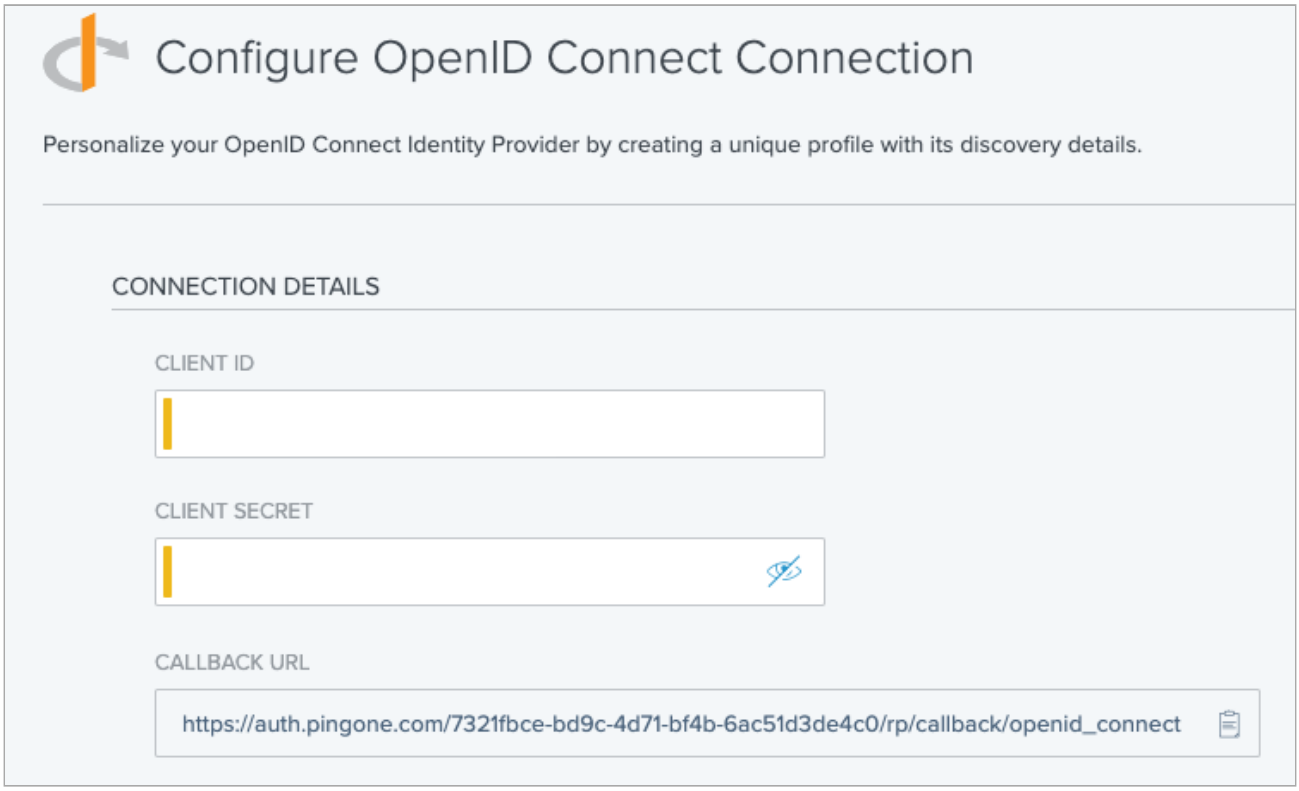
3. If you clicked **OpenID Connect**:

    1. In the **Create Profile**window, in the **Name** field, specify a name for the IdP (used only in the PingOne console) and click **Continue**.



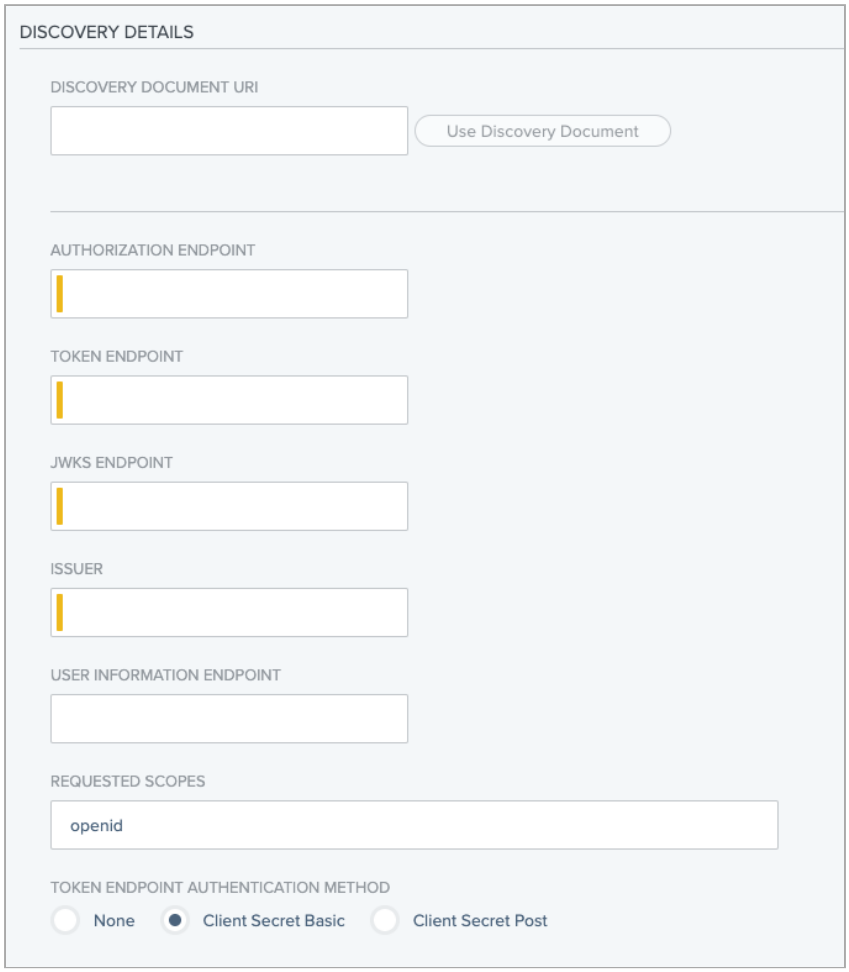
    2. In the **Connection Details** section, in the **Client ID** and **Client Secret** fields, enter the client ID and client secret from the external IdP.

> **Note**
>
> This must be an auth-code client.

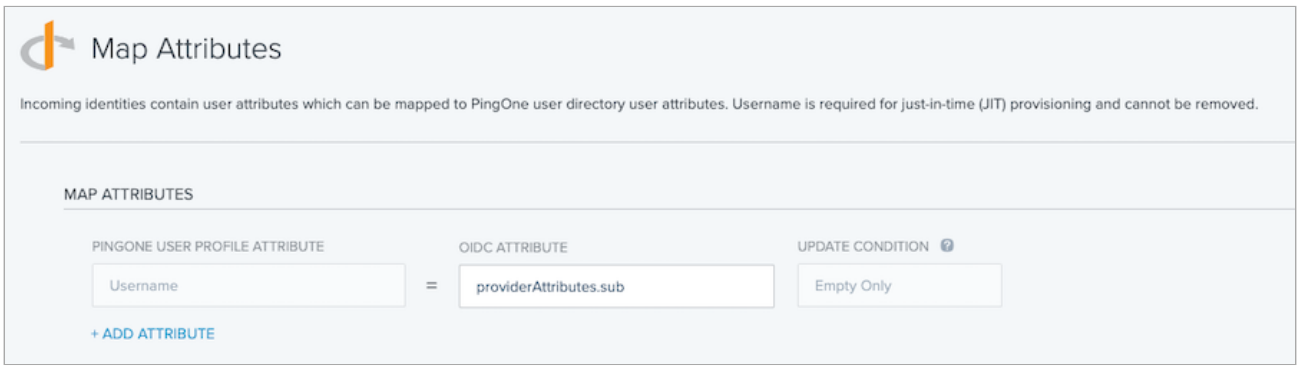


3. In the **Discovery Details** section, you can provide the OpenID well-known endpoint in the **Discovery Document** section to pre-populate all values.

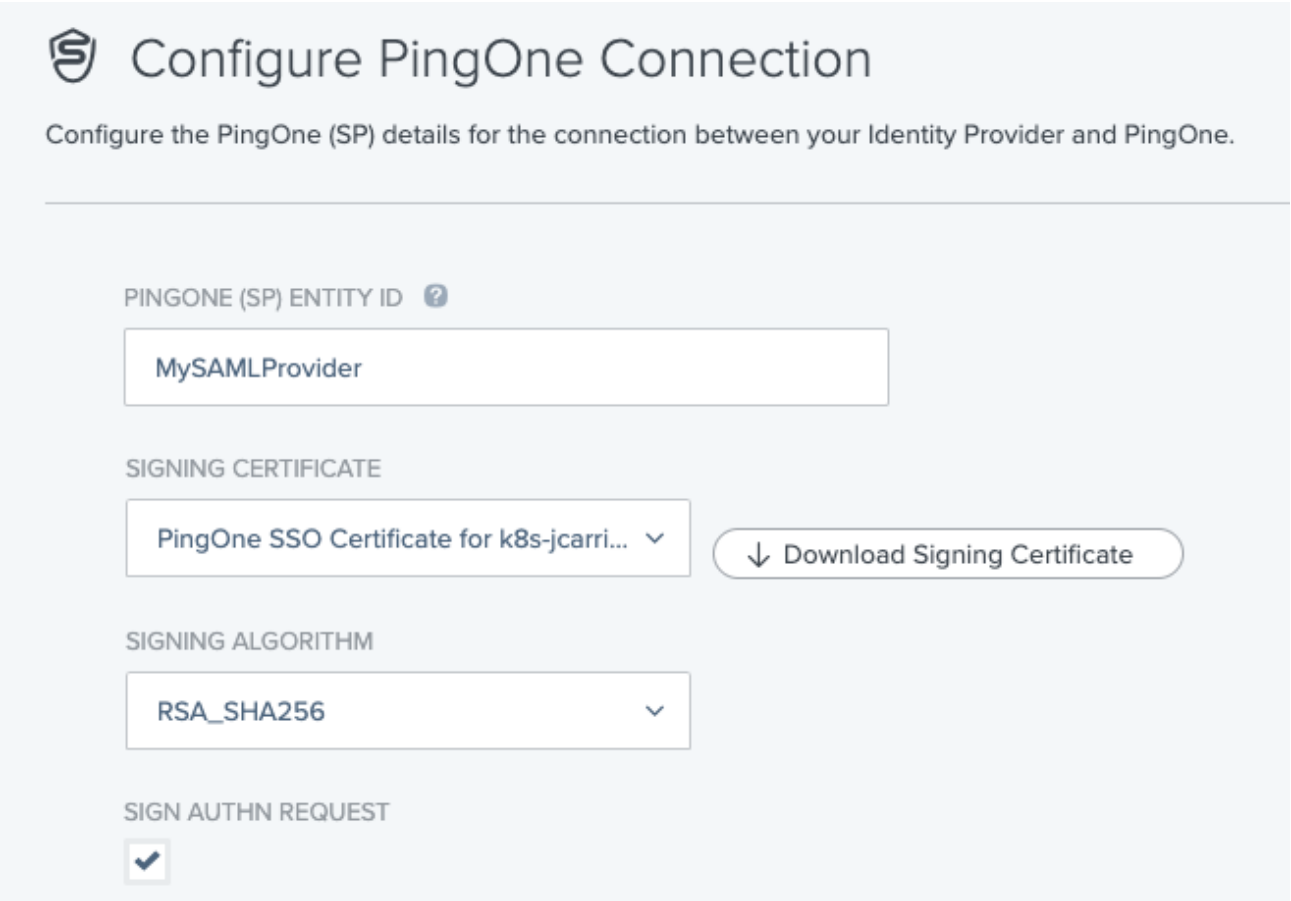
   If the OpenID well-known endpoint isn't available, you must manually enter all the required values.



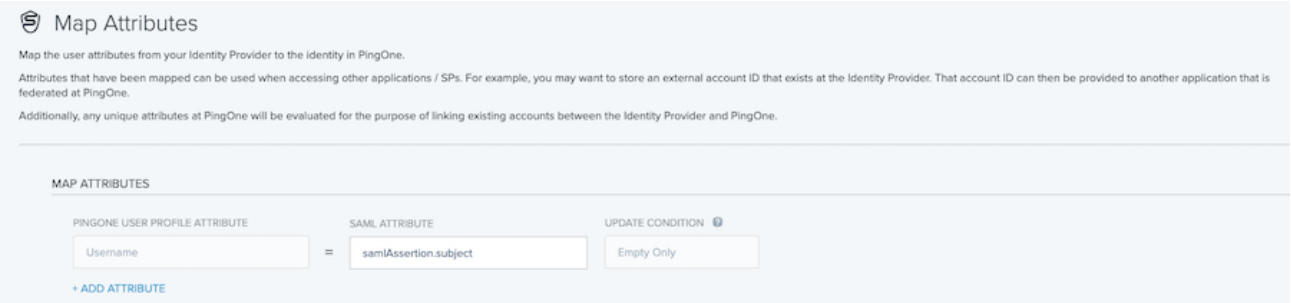
4. Click **Save and Continue**.
5. In the **Map Attributes** window, map incoming values as needed, and then click **Save and Finish**.



4. If you clicked **SAML**:
    1. In the **Create Profile** window, in the **Name** field, specify a name for the IdP (used only in the PingFederate console) and click **Continue**.
    2. In the **Configure PingOne Connection** section, choose the signing certificate for SP-initiated SAML authentication requests and click **Continue**.



3. In the **Configure IDP Connection** window, import data or provide the values, and then click**Save and Continue**.
4. In the **Map Attributes** window, map incoming values as needed, and then click **Save and Finish**.



5. **Optional:** To support just-in-time (JIT) creation, edit the newly created external IdP:

   If a user who doesn't exist in PingOne is redirected from the external IdP, PingOne can perform a JIT creation of an account for that user in PingOne.

   1. Click **Registration**.
   2. In the **Population** list, select the population into which new users should be JIT provisioned.
   3. Click **Save**.

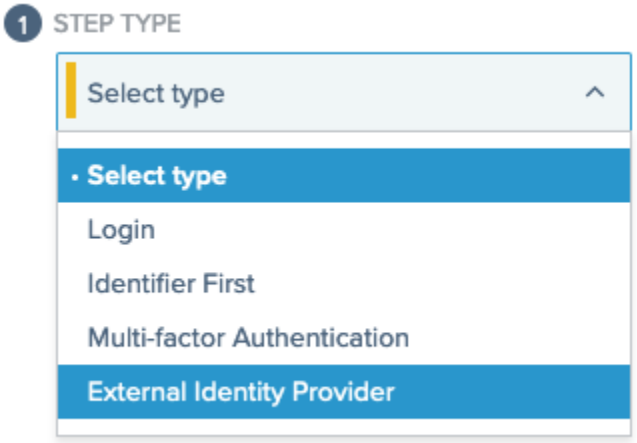
6. Enable the external IdP you created.

Creating an external IdP authentication policy

Steps

1. In your PingOne tenant, go to **Authentication > Authentication** and click **Add Policy**.

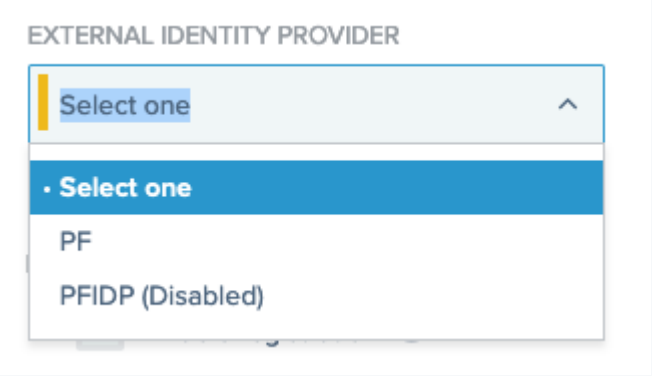
2. In the **Policy Name** field, enter a unique policy name.
3. In the **Step Type** list, select **External Identity Provider**.



4. In the **External Identity Provider** list, select the external IdP you want to delegate to.

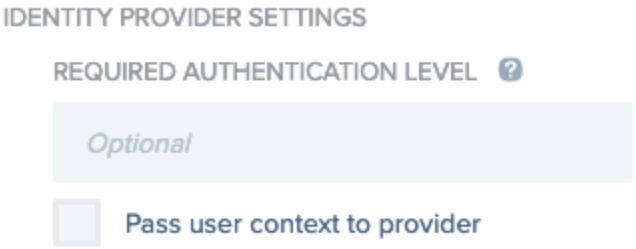
Note

Disabled external IdPs are marked as such.



5. **Optional:** In the **Required Authentication Level** field, specify an authentication context to request from the IdP.

   For example, if you were using PingFederate you could use a selector on the incoming context to determine authentication policy flows.



6. Click **Save and Continue**.

Next steps

Depending on how you want to use it, you can configure this policy as the default or assign it to specific applications. After calling an app that has this policy assigned, users are automatically sent to the external IdP for authentication.

After a successful return from the external IdP:

- If the user doesn't exist in PingOne, the user is created.
- If user does exist in PingOne, the user is prompted for linking and then passed to their respective application.

Enabling SLO for a PingAccess-protected application using PingFederate

Learn how to require a sign off of a PingAccess-protected application with PingFederate acting as a token provider.

There are multiple scenarios for signing off a user. As an administrator, you might require one or more sign-off flows because requirements for each application can differ.

This guide provides information for the following use cases:

- Global single logout (SLO)
- Per application or partial logout

Each PingAccess sign-off flow is user-initiated, not system-initiated. PingAccess checks if a session is revoked and, if the existing session is found to be revoked, redirects to the web. PingAccess doesn't revoke sessions. It checks the revocation status by sending a GET request to that endpoint.

Learn more in Authorization endpoint (page 1,069) and Session Revocation API endpoint (page 1,156).

Components

- PingAccess 6.3
- PingFederate 10.3

PingAccess Logout endpoint

PingAccess responds differently depending on whether the sign off is successful or unsuccessful.

- For successful sign offs:
  1. PingAccess responds to the `/pa/oidc/logout.png` request with `Set-Cookie: PA.ACE_ws=;`.
  2. The `/pa/oidc/logout.png` endpoint clears the ID token from the browser containing the PingAccess cookie.

     Unless you select **Use single-logout** (SLO) for the token provider, the `/pa/oidc/logout.png` endpoint clears the cookie only from the requested host/domain, and the cookie might still exist in requests bound for other hosts/domains.

     > **Note**
     >
     > If you select the **Use Single-Logout** option when configuring the token provider, the `/pa/oidc/logout.png` endpoint also sends a logout request to the token provider, which completes a full SLO flow.

- For unsuccessful sign offs:
  1. PingAccess responds to the same `/pa/oidc/logout.png` request without clearing the `PA.ACE_ws;` cookie.
  2. The user is directed back to the PingAccess-protected application page.
  3. If the application reads and finds the `PA.ACE_ws;` cookie present, it doesn't redirect to PingFederate for authentication.

PingAccess can only clear the sessions for which the corresponding cookie was sent in the request to the `/pa/oidc/logout` resource. If PingFederate or the authentication authority can maintain different sessions for each set of apps, you can use SLO to sign off of all sessions in each set. To initiate the end sessions sign off in specific domains, call the `/pa/oidc/logout.png` endpoint used by SLO.

Learn more in Server-side session management configuration (page 112) in the PingAccess solutions documentation.

Configuring PingAccess partial logout

Learn how to require termination of a user's session per application or by a partial logout protected in the PingAccess administrative console.

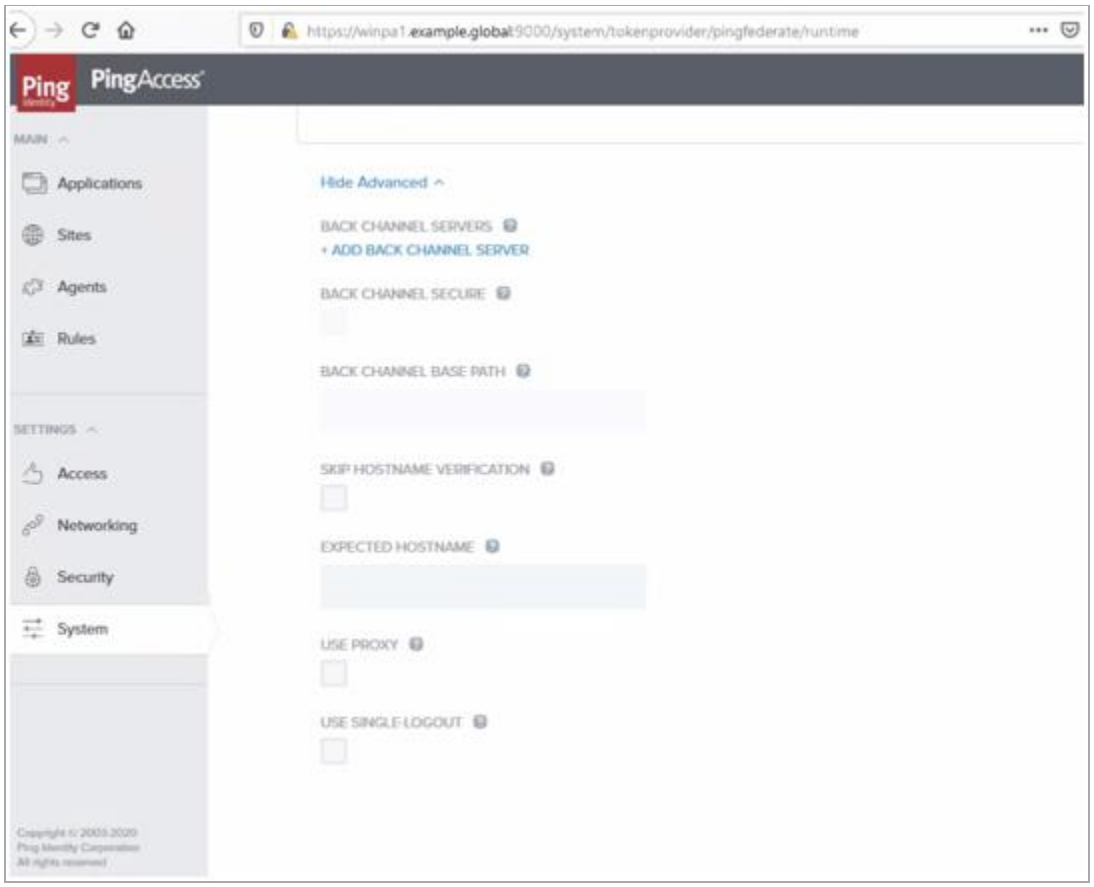
Before you begin

You must:

- Execute the identity provider (IdP) adapter's logout endpoint. Learn more in Html Form Adapter Logout Configuration.
- Have experience with PingFederate and PingAccess.

Steps

1. In the PingAccess administrative console:
    1. Go to **Settings > Token Provider > Runtime > Show Advanced Settings**.
    2. Clear the **Use Single-Logout** checkbox.



    3. Click **Save**.
2. In the PingFederate administrative console:
    1. Go to **Authentication > Integration > IdP Adapters > Manage Adapter Instances** and then select the relevant IdP adapter instance.

Result:

The **Create Adapter Instance** page opens.

2. To show the logout related fields, go to the **IdP Adapter > Show Advanced Fields**.
3. In the **Logout Path** field, enter the path with the PingAccess endpoint.

   You can enter any valid path string.

> **Note**
>
> This value must start with a "/" character. For example, if you enter `/mylogoutpath`, then the logout path is `/ext/mylogoutpath`. Don't use a path already used by another adapter, such as `/ext/pickup` or `/ext/dropoff`.

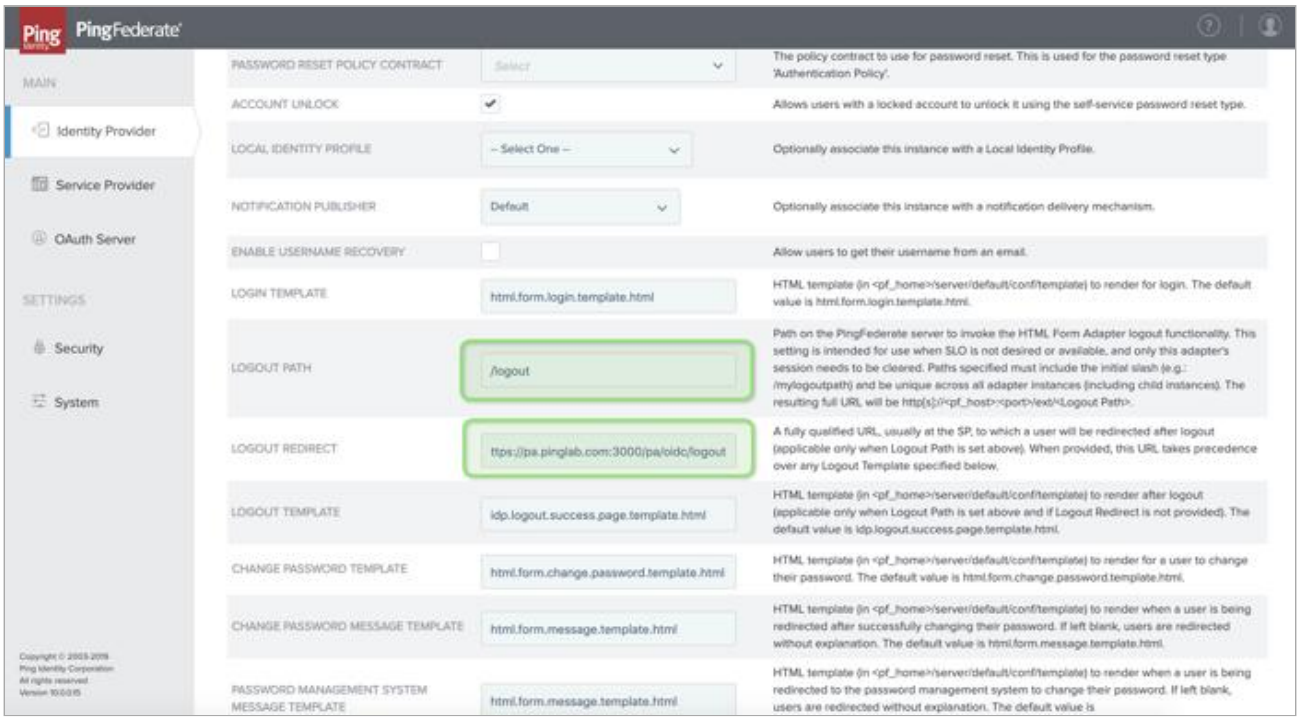
> **Tip**
>
> Use an alphanumeric string to minimize the risk of using an invalid value in this field.

4. In the **Logout Redirect** field, enter the URL that PingFederate uses to redirect the user after sign off.

   The default **Logout Redirect** value is `https://<pingaccessServer>:3000/pa/oidc/logout`.

5. For PingFederate to display a page using a template, in the **Logout Template** field, enter the name of the template file.
6. In the **Logout Path** field, enter a path with the PingAccess endpoint.

   The default **Logout Path** value is `<pf_install>/server/default/conf/template/idp.logout.success.page.template.html`.



7. Click **Done**.

Configuring global single logout

Learn how to revoke global sessions with single logout (SLO) in the PingFederate administrative console.

Before you begin

Make sure the identity provider (IdP) adapters have their **Session State** set to **Globally** in PingFederate.

About this task

To revoke global sessions with SLO:

Steps

1. In the PingFederate administrative console, go to **Applications > Integrations > Default URLs > SP Default URLs**.
2. To allow `TargetResource` as a redirect URI in PingFederate, enter and edit the URL in the **Provide The Default URL You Would Like to Send The User to When Single Logout (SLO) Has Succeeded** field.

   The `TargetResource` is the landing page PingFederate directs the user to after logout, for example, `http://pf01.pinglab.com:9331/idp/startSLO.ping?TargetResource=://pa01.pinglab.com:3000/PingAccessQuickstart/`.

   > **Note**
   >
   > The `TargetResource` must be an allowed redirect URI in PingFederate.

3. Click **Save**.

Result

PingFederate automatically redirects to the PingAccess logout endpoint `pa/oidc/logout`.

Configuring PingFederate for PingAccess single logout

Learn how to configure PingFederate for user-initiated PingAccess single logout (SLO) so that PingFederate knows to add the Subresource Integrities (SRIs) to the revocation list if SLO is initiated.

About this task

There are two ways to implement Server-Side Session Management:

- PingAccess can reject a PingAccess cookie associated with a PingFederate session that has been invalidated as a result of an end-user-driven logout.
- The end-user can initiate a logout from all PingAccess issued web sessions using a centralized sign off.

PingAccess can only clear the sessions for which the corresponding cookie is sent in the request to the `/pa/oidc/logout` resource. If PingFederate, as the authentication authority, can maintain different sessions for each set of apps, you can use SLO to sign off of all sessions in each set. Call the `/pa/oidc/logout.png` endpoint used by SLO to initiate the end sessions sign off in specific domains.

SLO is done by redirecting to the standard SLO location, which is configured in the `run.props` file. PingAccess does not revoke the user's session. The user is directed to the `pa.oidc.logout.redirectURI` URI when they sign off using OpenID Connect and the PingFederate SLO endpoint.

Learn more in Configuration file reference (page 171) and OpenID Connect endpoints (page 158).

Steps

1. In the PingFederate administrative console, go to **Applications > OAuth > Clients > Client Management**, and select the relevant client.

   Result:

   The **Client** page opens.

2. To enable PingFederate to add the SRIs to the revocation list if SLO is initiated, in the **OpenID Connect** section, select the **PingAccess Logout Capable** checkbox.

3. Click **Save**.

Result

PingFederate uses the `logout.png` endpoint `/pa/oidc/logout.png` to initiate a sign off from PingAccess in conjunction with the SLO functionality. This endpoint terminates the PingAccess tokens across domains.

Learn more in Configuring PingFederate for user-initiated single logout (page 113).

Integrating Pulse Connect Secure with PingFederate

Learn how to integrate Pulse Connect Secure with PingFederate for single sign-on (SSO).

Component

PingFederate 10.3

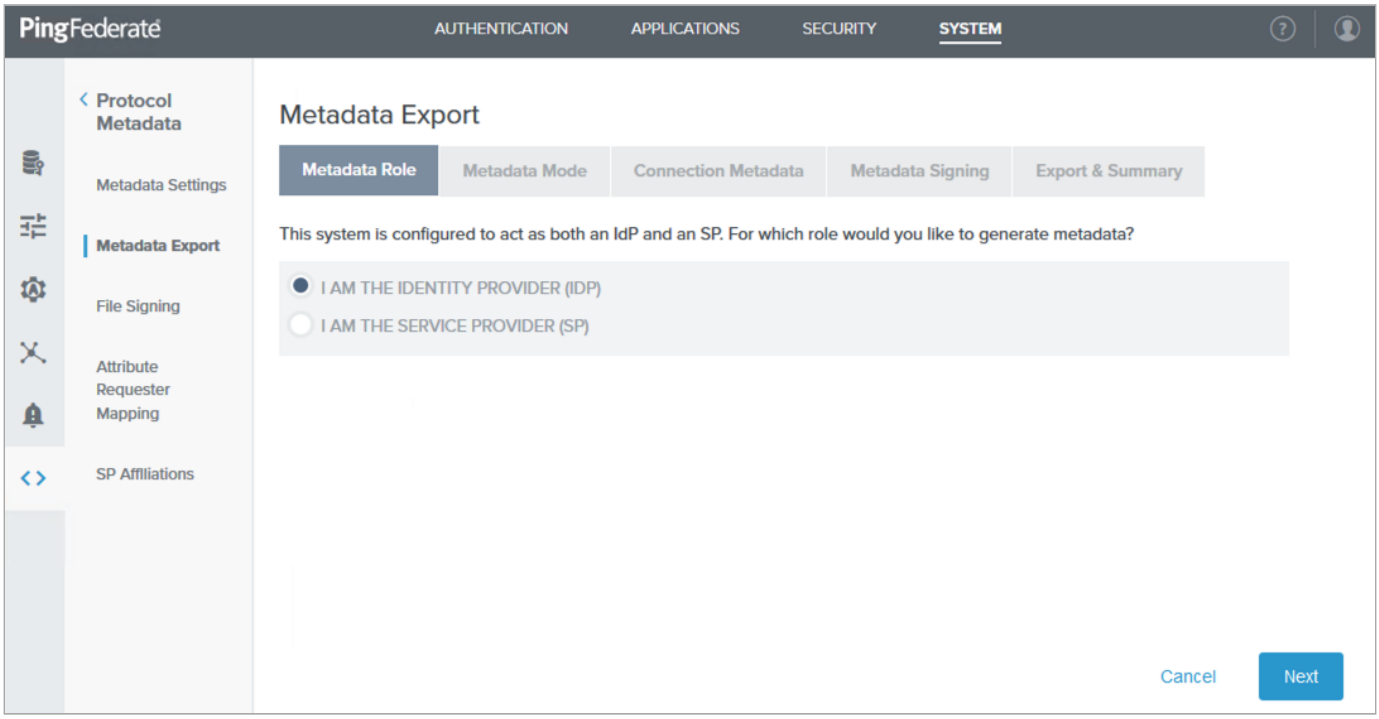
Before you begin

- Configure a PingFederate data store. Learn more in Datastores.
- Configure a PingFederate Password Credential Validator.
- Configure a PingFederate HTML Form Adapter.
- Configure a Pulse Connect Secure authentication realm for your users.
- Configure a Pulse Connect Secure sign-on policy for your users.

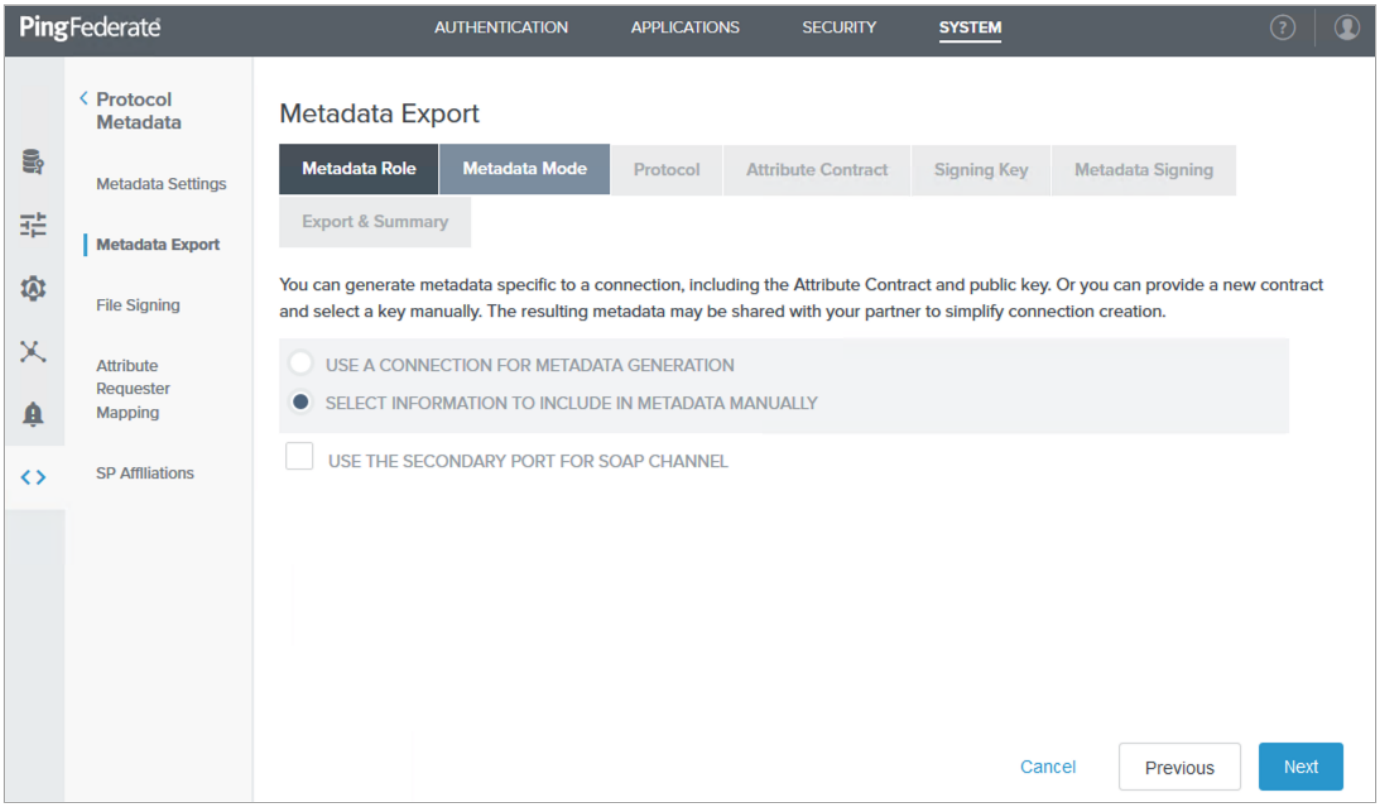
Exporting SAML metadata from PingFederate

Steps

1. Sign on to the PingFederate administrative console and go to **System → Protocol Metadata → Metadata Export**.
2. On the **Metadata Role** tab, select **I am the Identity Provider (IdP)**, and then click **Next**.



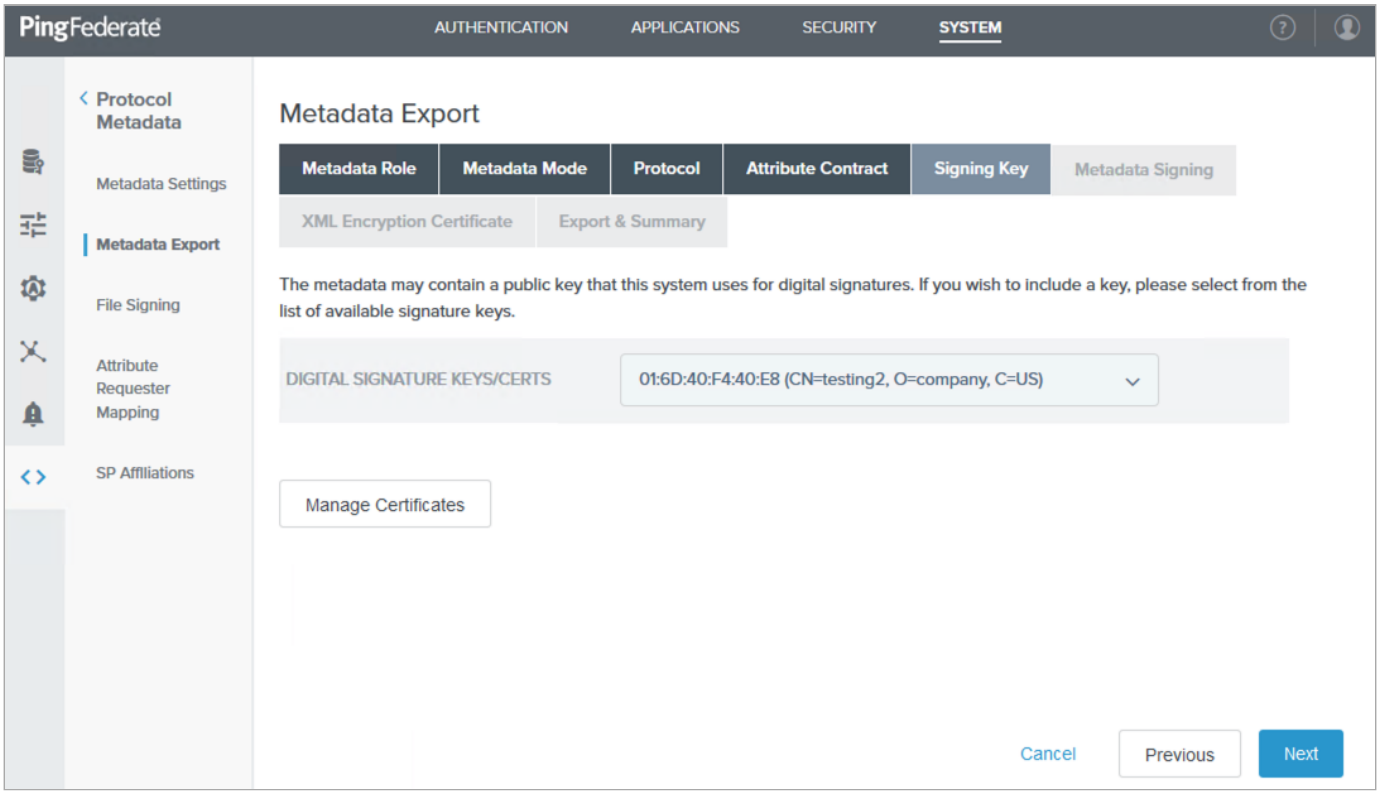
3. On the **Metadata Mode** tab, select **Select Information to Include in Metadata Manually**, and then click **Next**.



4. On the **Protocol** tab, click **Next** until you reach the **Signing Key** tab, accepting the default values.

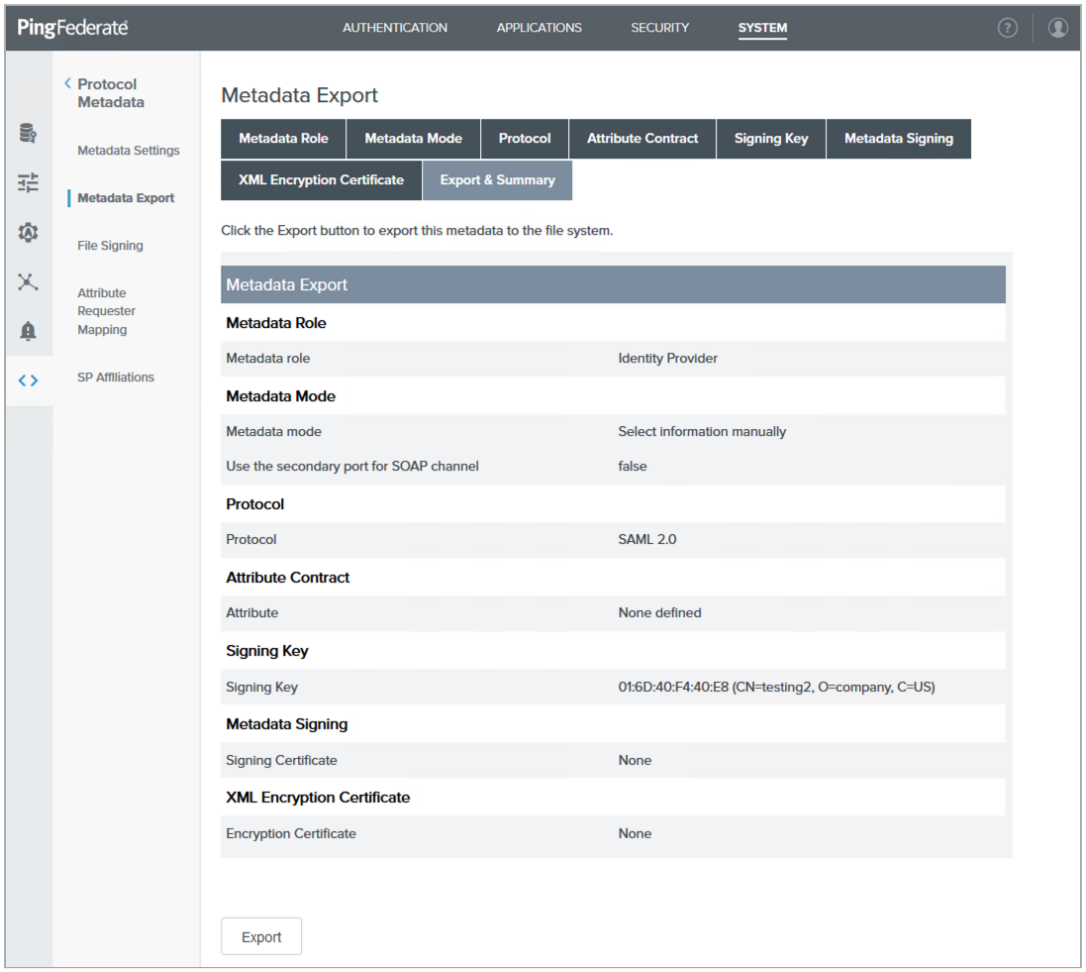
5. On the **Signing Key** tab, select an available signing key from the **Digital Signature Keys/Certs** list, and then click **Next**. If none are available, click **Manage Certificates** to create a signing key, and then follow the on-screen instructions.

> **Important**
>
> Although you can use a self-signed certificate, a CA-signed certificate is recommended.



6. Click **Next** until you reach the **Export & Summary** tab, accepting the default values on the **Metadata Signing** and **XML Encryption Certificate** tabs.

7. On the **Export & Summary** tab, click **Export** and save the `metadata.xml` file. You will upload this file to Palo Alto Networks NGFW in the next step.



Exporting the signing certificate from PingFederate

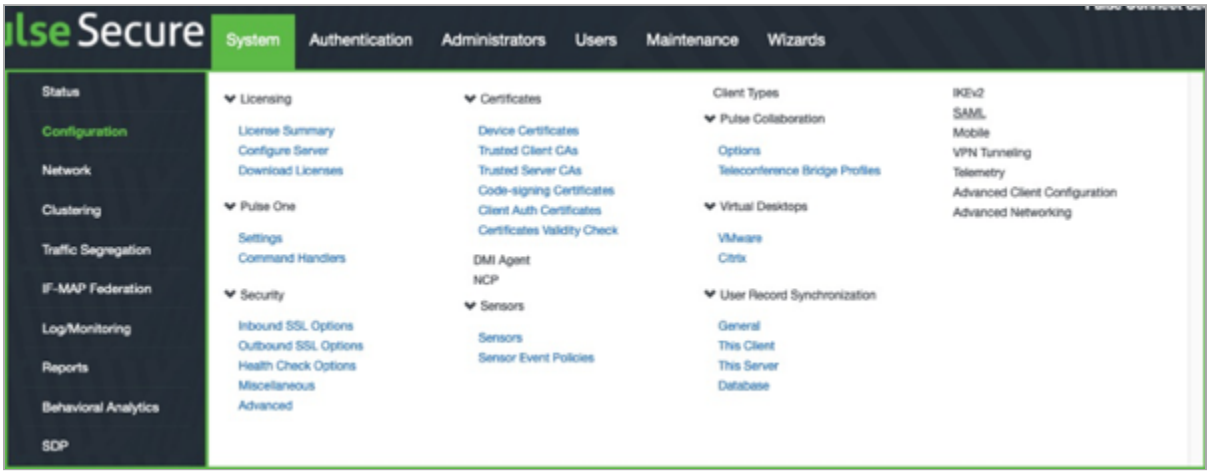
Steps

1. Sign on to the PingFederate administrative console.
2. Go to **Security > Signing & Decryption Keys & Certificates**.
3. In the row of the certificate that you want to use to sign SAML assertions to Pulse Connect Secure, in the **Select Action** list, select **Export**.
4. On the **Export Certificate** tab, click **Certificate Only**. Click **Next**.
5. On the **Export & Summary** tab, click **Export** and save the file.
6. Click **Done**.

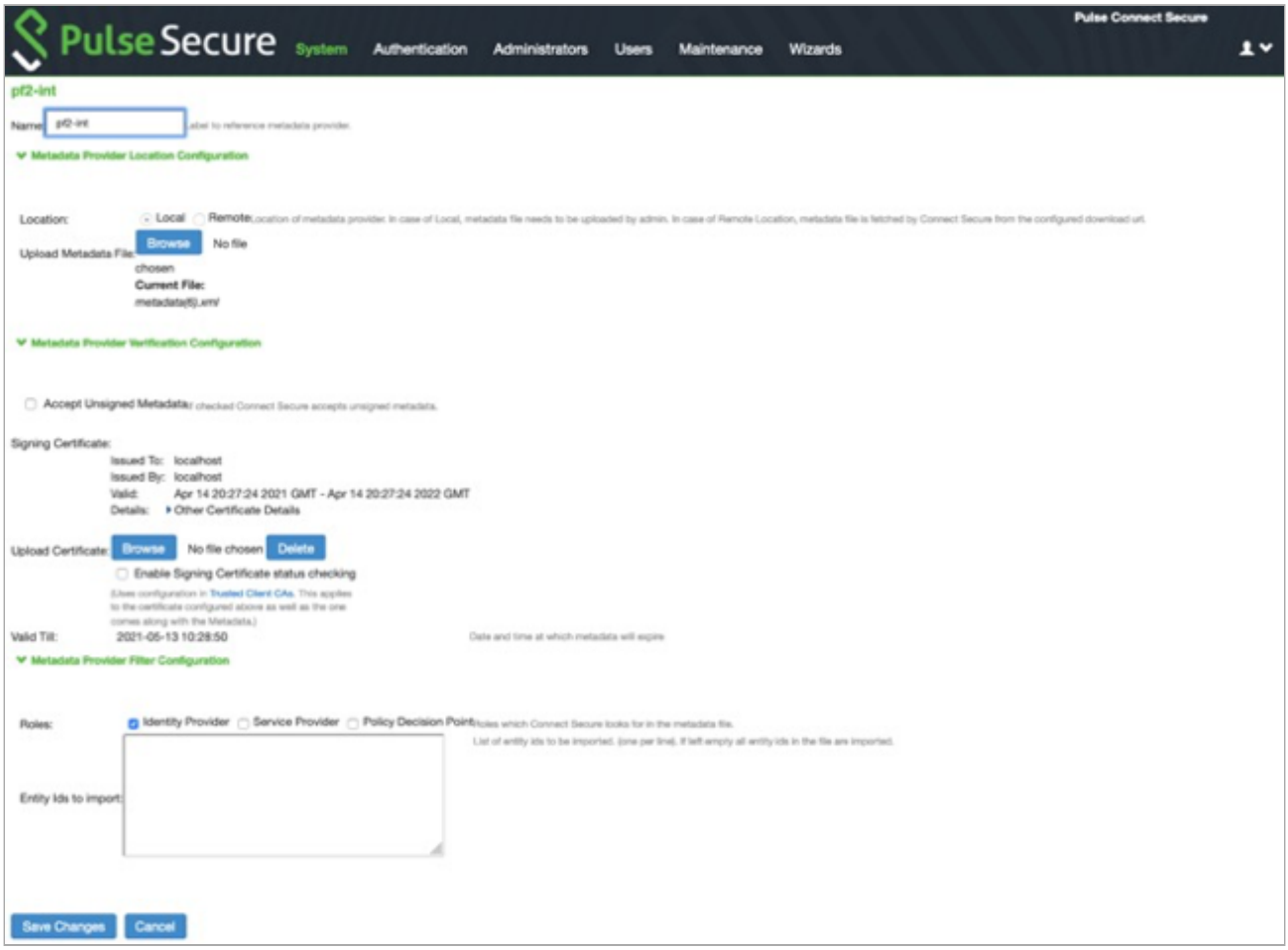
Configuring SAML integration with PingFederate in Pulse Connect Secure

Steps

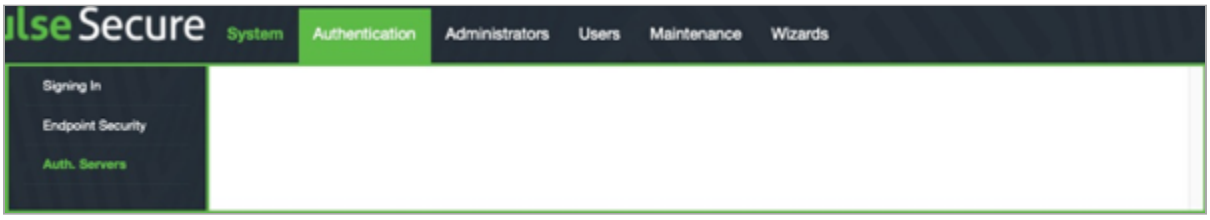
1. In the Pulse Connect Secure administrative interface, go to **System > Configuration > SAML**.



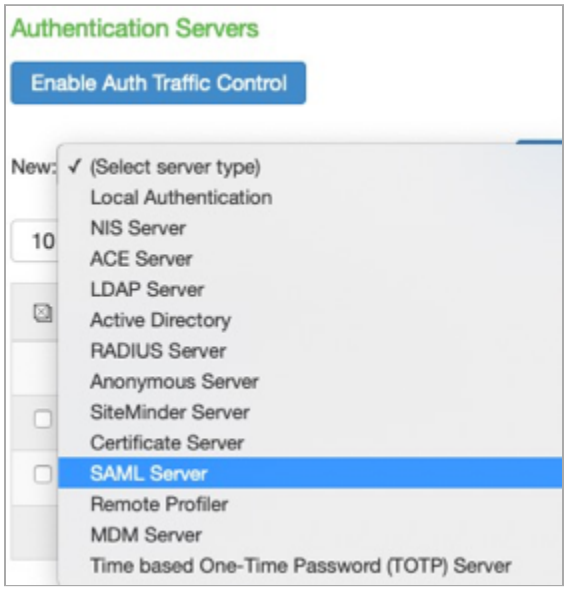
2. Click **New Metadata Provider**.
3. Configure the new metadata provider:
    1. In the **Name** field, enter a name.
    2. In the **Location** field, select **Local**.
    3. In the **Upload Metadata File** field, click **Browse** and import the metadata file you saved in Configuring SSO for GlobalProtect VPN with PingFederate
    4. In the **Signing Certificate** field, click **Browse** and select the certificate file you saved in the previous topic Exporting the signing certificate from PingFederate.
    5. In the **Roles** field, select the **Identity Provider** checkbox.
    6. Click **Save Changes**.



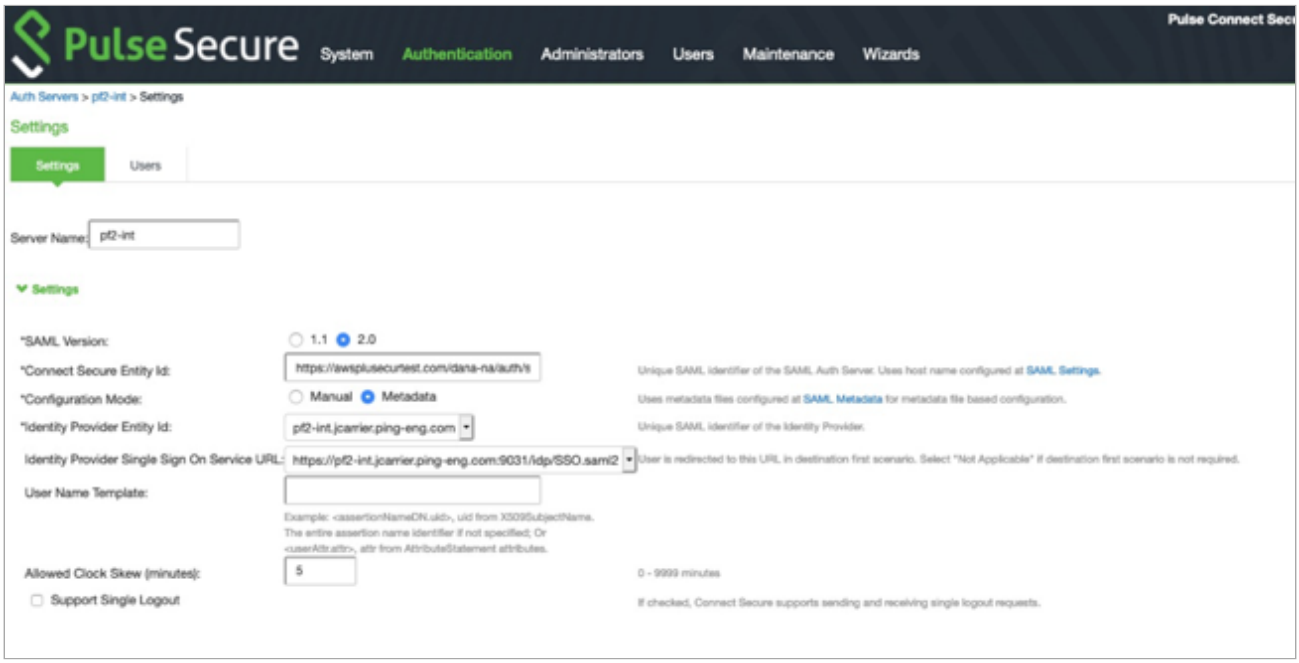
4. In the Pulse Connect Secure administrative interface, go to **Authentication > Auth Servers**.



5. In the list, select **SAML Server** and then click **New Server**.



6. Configure the new server:
    1. Enter a **Server Name**.
    2. For **SAML Version**, click **2.0**.
    3. For **Configuration Mode**, click **Metadata**.
    4. In the **Identity Provider Entity ID** list, select the identity provider (IdP) that you created in the previous steps.
    5. In the **Identity Provider Single Sign On Service URL** list, select the appropriate SSO URL.

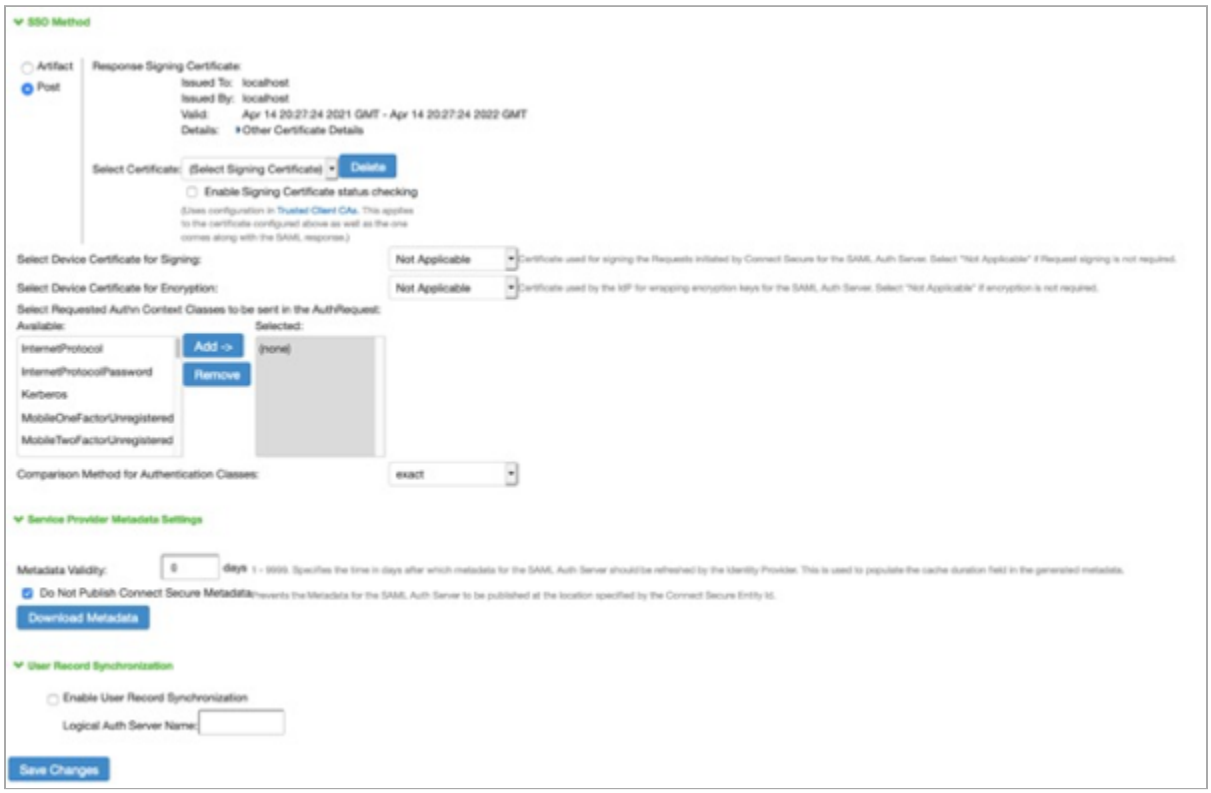


    6. In the **SSO Method** section, click **POST**.
    7. In the **Select Certificate** list, select the signing certificate you created previously.

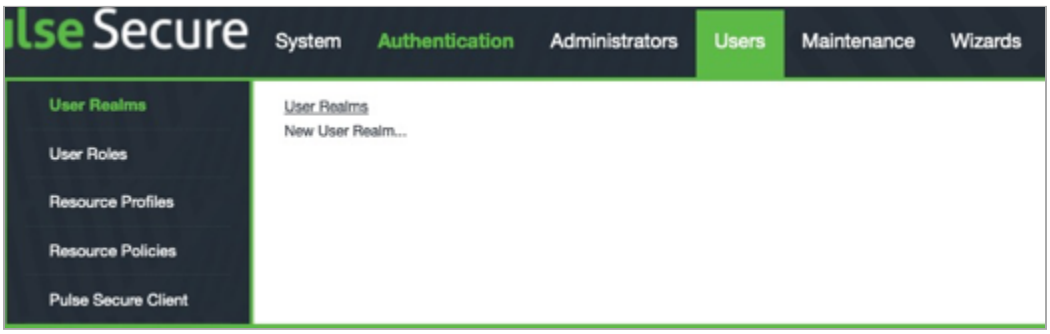
8. In the **Metadata Validity** field, enter any non-zero value.

> **Note**
>
> You must populate the **Metadata Validity** field even though it won't be used.

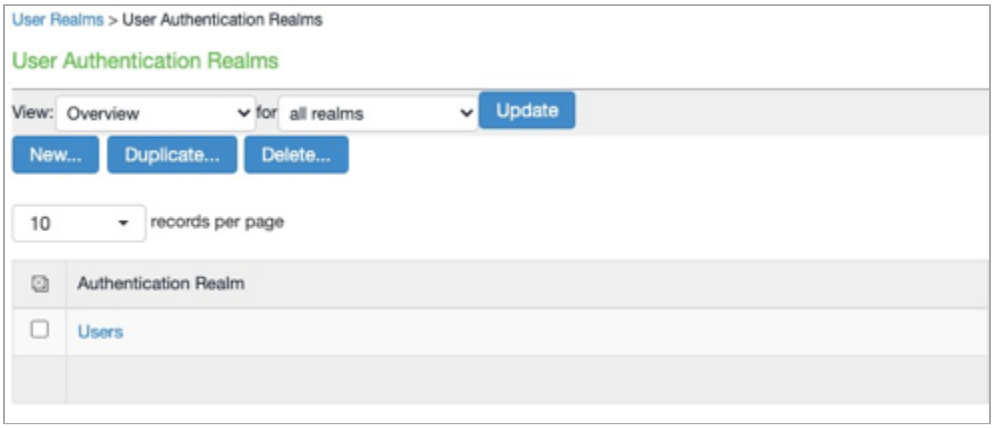
9. Select the **Do Not Publish Connect Secure Metadata** checkbox.
10. Click **Save Changes**.



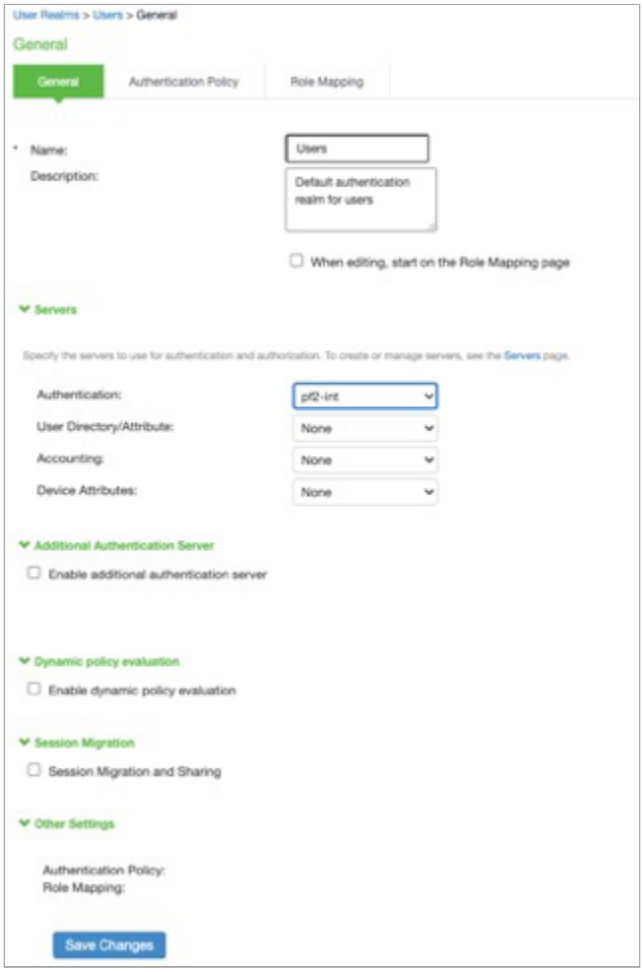
11. Click **Download Metadata** and save the file.
12. In the Pulse Connect Secure administrative interface, go to **Users > User Realms**.



13. Select the authentication realm for your user population.



14. In the **Authentication** list, select the IdP that you configured.



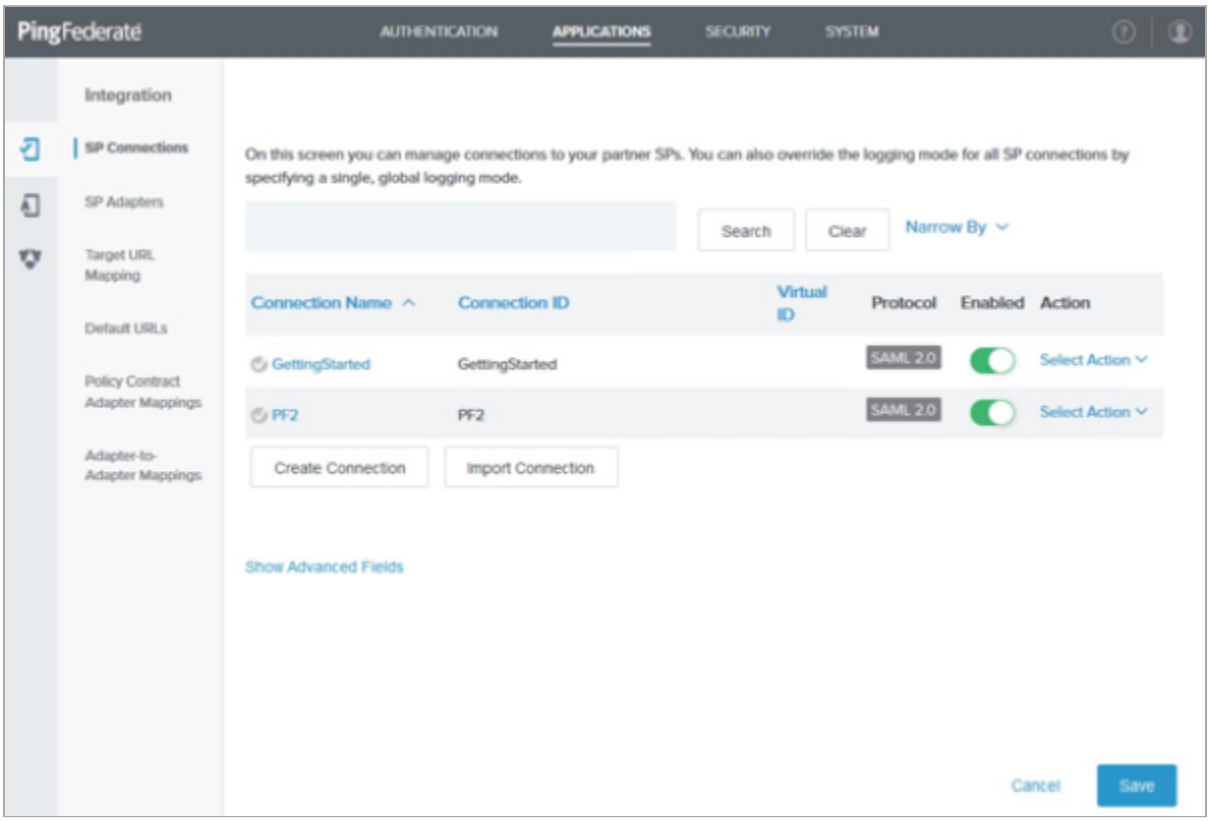
15. Click **Save Changes**.

Configuring SAML integration with Pulse Connect Secure in PingFederate

Steps

1. In the PingFederate administrative console, go to **Applications > Integration > SP Connections**.

2. Click **Create Connection**.



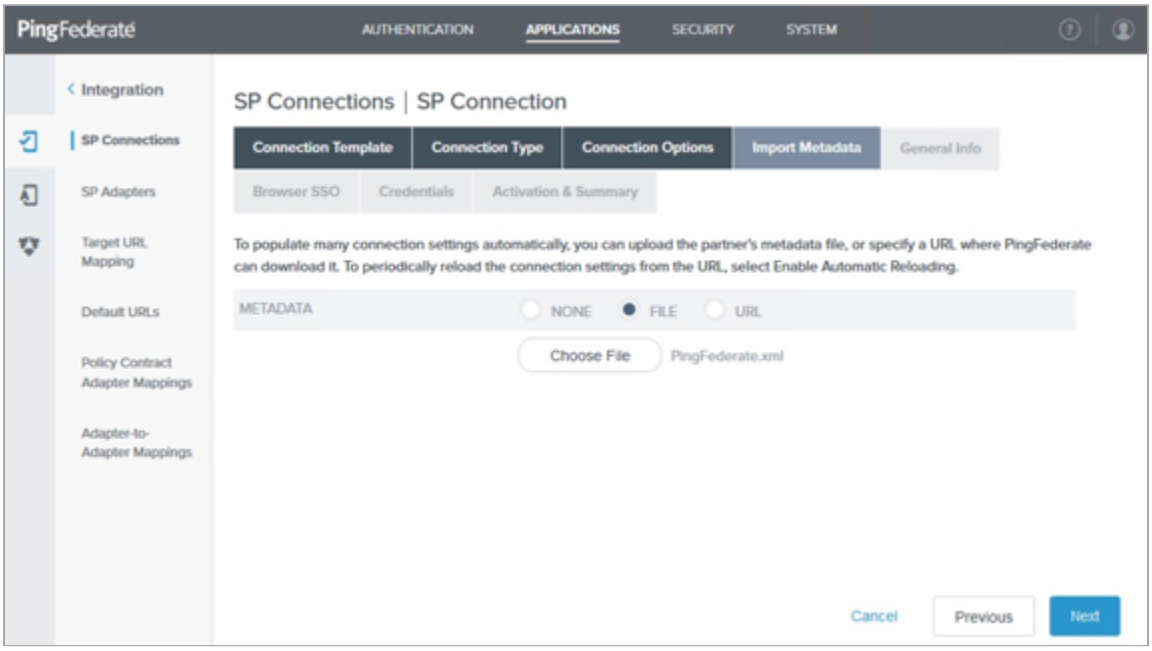
3. On the **Connection Template** tab, click **Do not use a template for this connection**. Click **Next**.

4. On the **Connection Type** tab, select the **Browser SSO Profiles** checkbox.

5. In the **Protocol** list, select **SAML 2.0** and click **Next**.

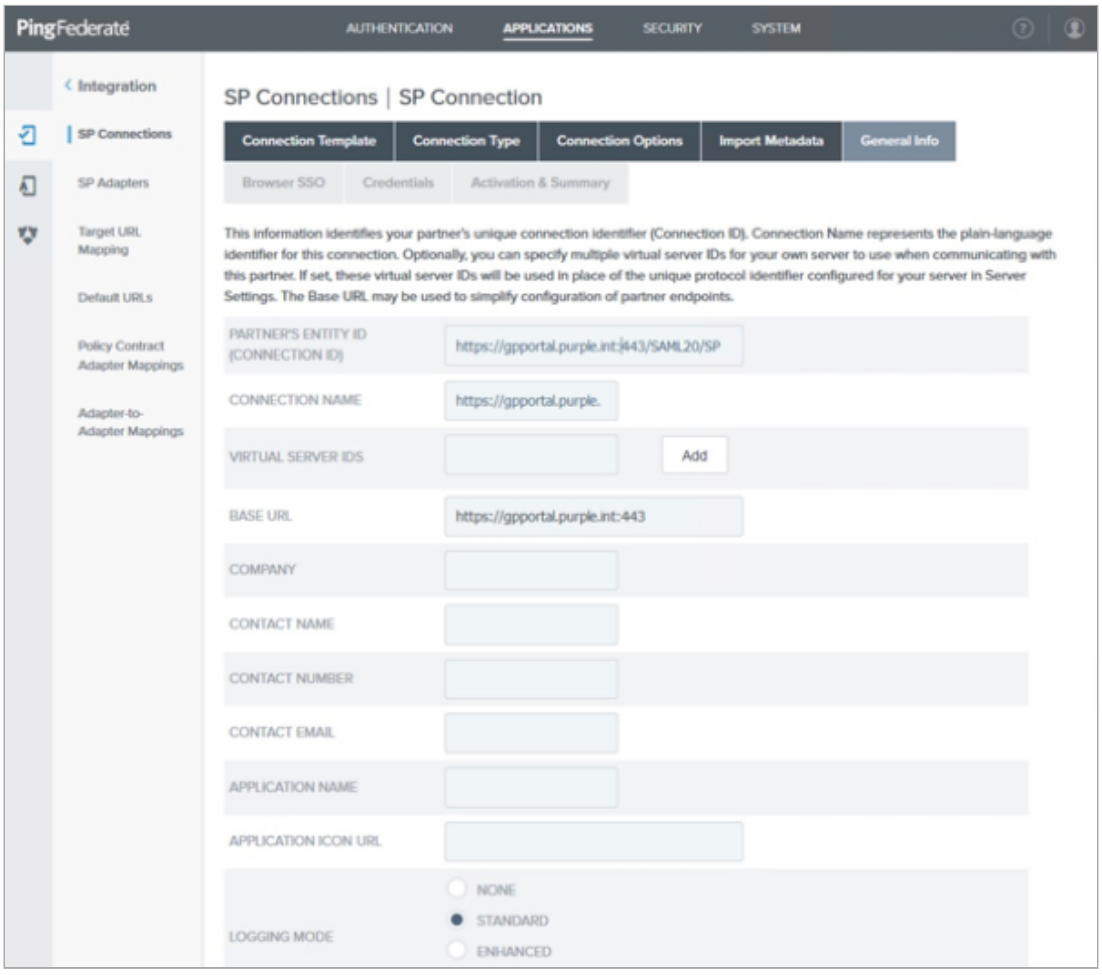
6. On the **Connection Options** tab, click **Next**.

7. On the **Import Metadata** tab, click **File** and then choose the metadata file that you downloaded previously. Click **Next**.

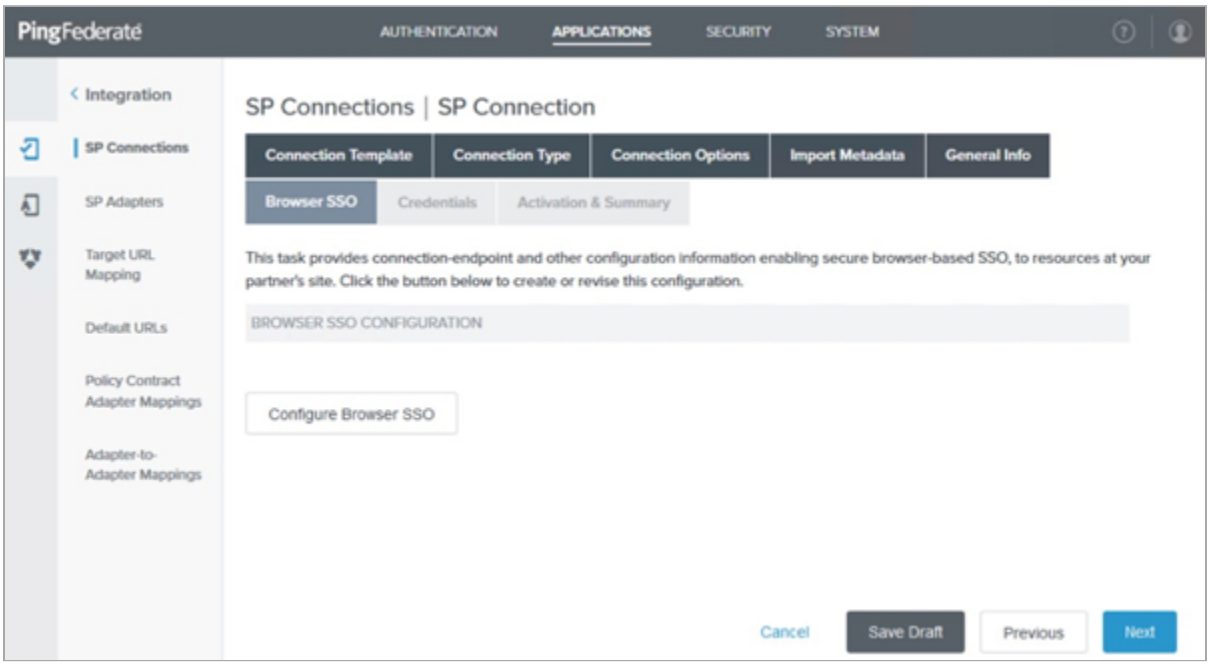


8. On the **Metadata Summary** tab, review the **EntityID** field and click **Next**.

9. On the **General Info** tab, review the imported **Base URL** field, then click **Next**.



10. On the**Browser SSO** tab, click **Configure Browser SSO**.

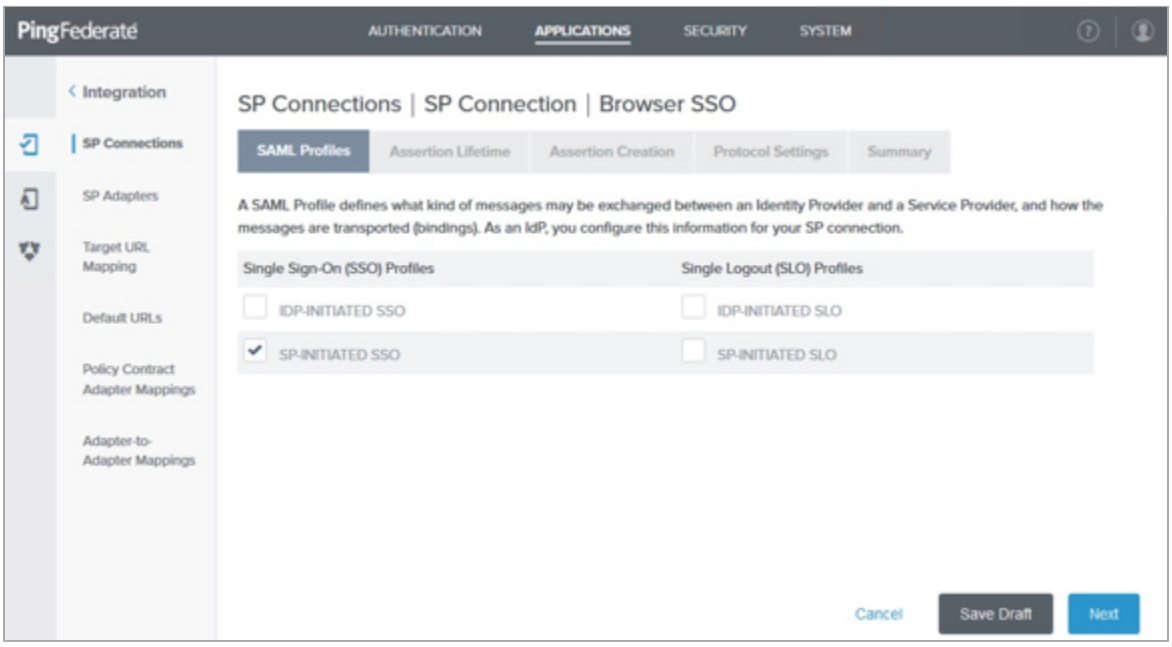


Result:

The tabs for the **Browser SSO** section display.

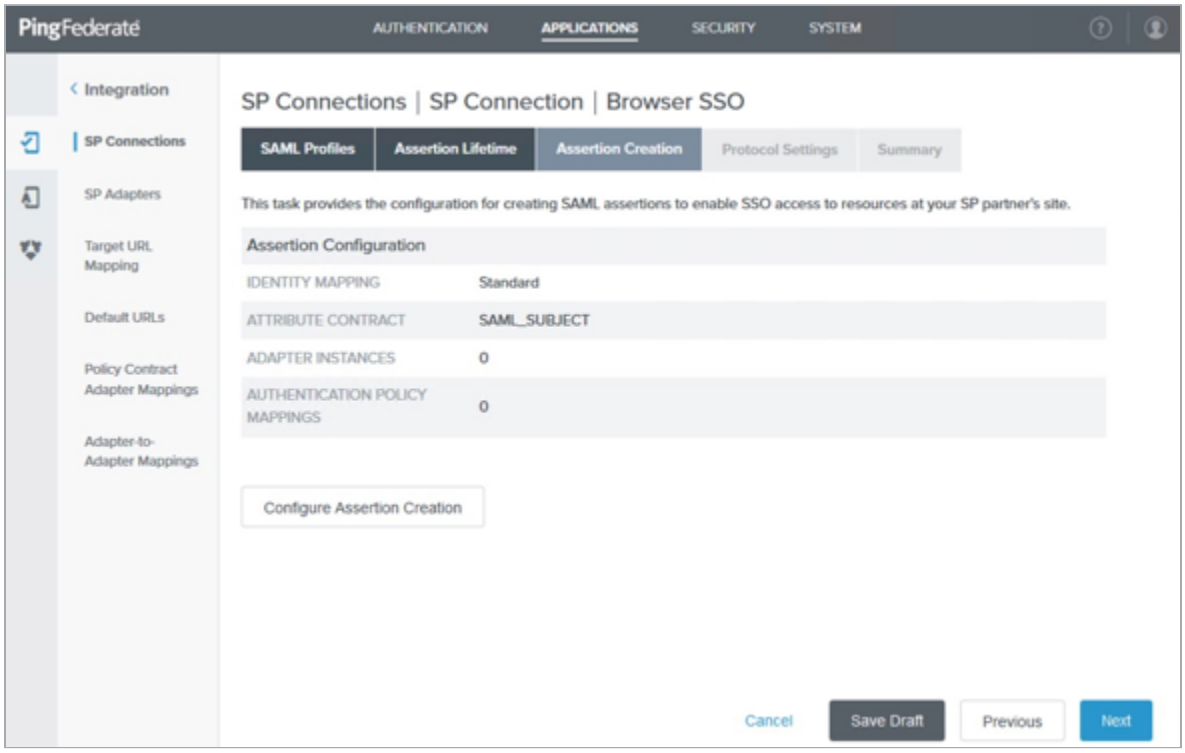
11. Configure the browser SSO:

    1. On the **SAML Profiles** tab, select the **SP-Initiated SSO** checkbox. Click **Next**.



    2. On the **Assertion Lifetime** tab, accept the default values and click **Next**.

    3. On the **Assertion Creation** tab, click **Configure Assertion Creation**.

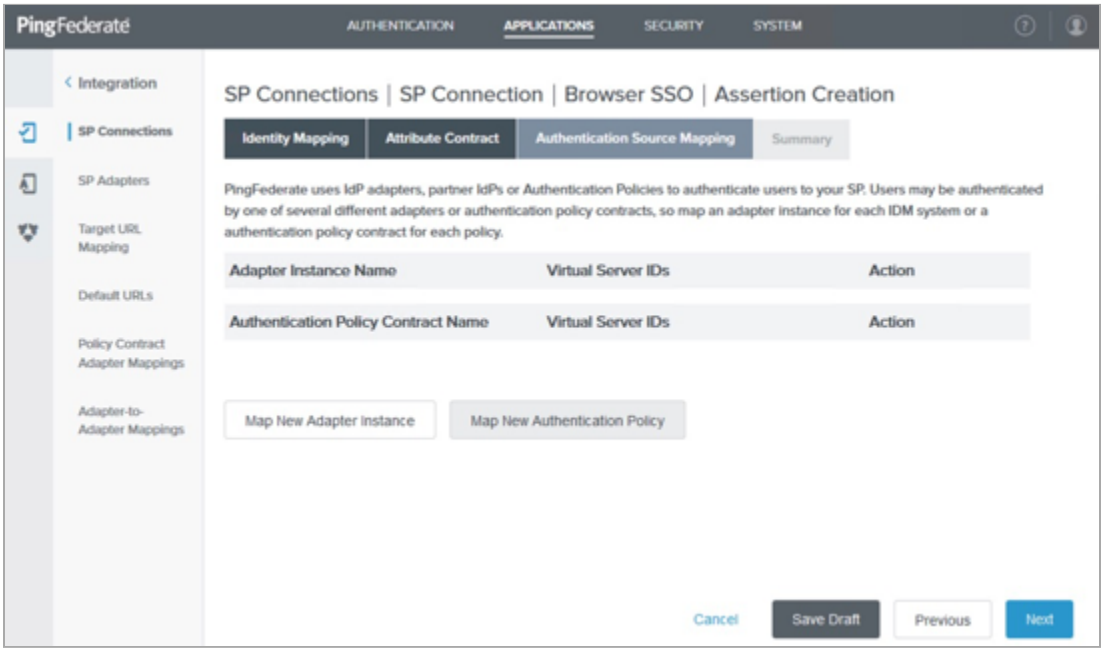


Result:

The tabs for the **Assertion Creation** section display.

12. Configure the assertion creation:

    1. On the **Identity Mapping** tab, click **Next**.
    2. On the **Attribute Contract** tab, click **Next**.
    3. On the **Authentication Source Mapping** tab, click **Map New Adapter Instance**.

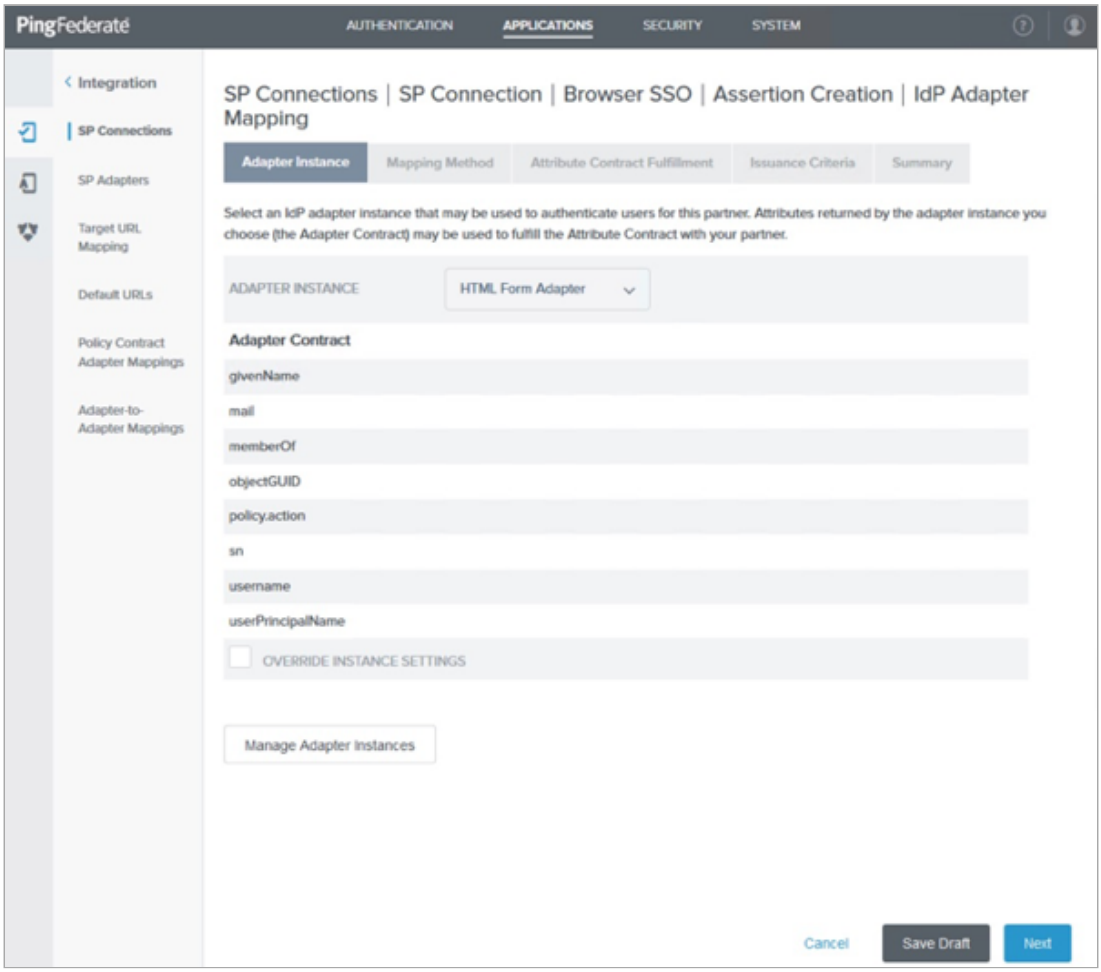


Result:

The tabs for the **IdP Adapter Mapping** section display.

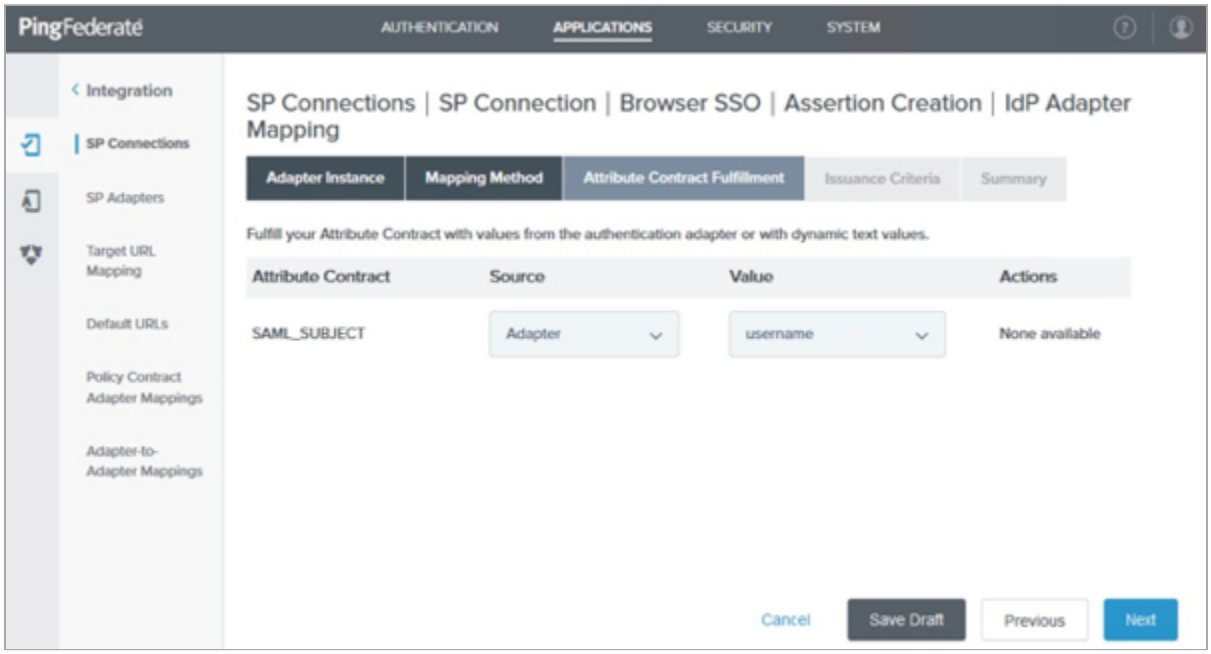
13. Configure the IdP adapter mapping:

    1. On the **Adapter Instance** tab, select the HTML form adapter that you created. Click **Next**.



    2. On the **Mapping Method** tab, click **Next**.

    3. On the **Attribute Contract Fulfillment** tab, in the **Source** list select **Adapter** and in the **Value** list select **username**. Click **Next**.



4. On the **Issuance Criteria** tab, click **Next**.
5. On the **Summary** tab, click **Done**.

   Result:

   You return to the **Assertion Creation** section.

14. On the **Authentication Source Mapping** tab, click **Next**.
15. On the **Summary** tab, click **Done**.

    Result:

    You return to the **Browser SSO** section.

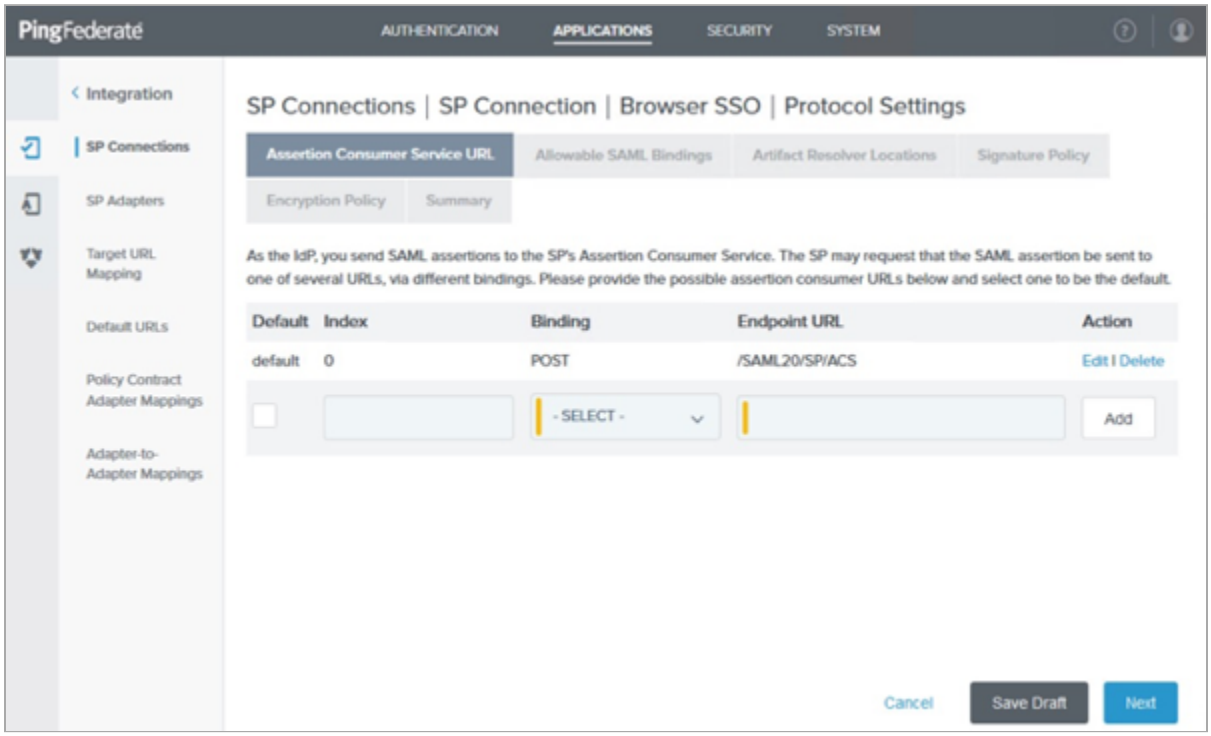
16. On the **Assertion Creation** tab, click **Next**.
17. On the **Protocol Settings** tab, click **Configure Protocol Settings**.

    Result:

    The tabs for the **Protocol Settings** section display.

18. Configure the protocol settings:

    1. On the **Assertion Consumer Service URL** tab, review the **Endpoint URL** value. Click **Next**.



2. On the **Allowable SAML Bindings** tab, ensure that **POST** and **REDIRECT** are the only values checked. Click **Next**.
3. On the **Signature Policy** tab, click **Next**.
4. On the **Encryption Policy** tab, click **Next**.
5. On the **Summary** tab, click **Done**.

   Result:

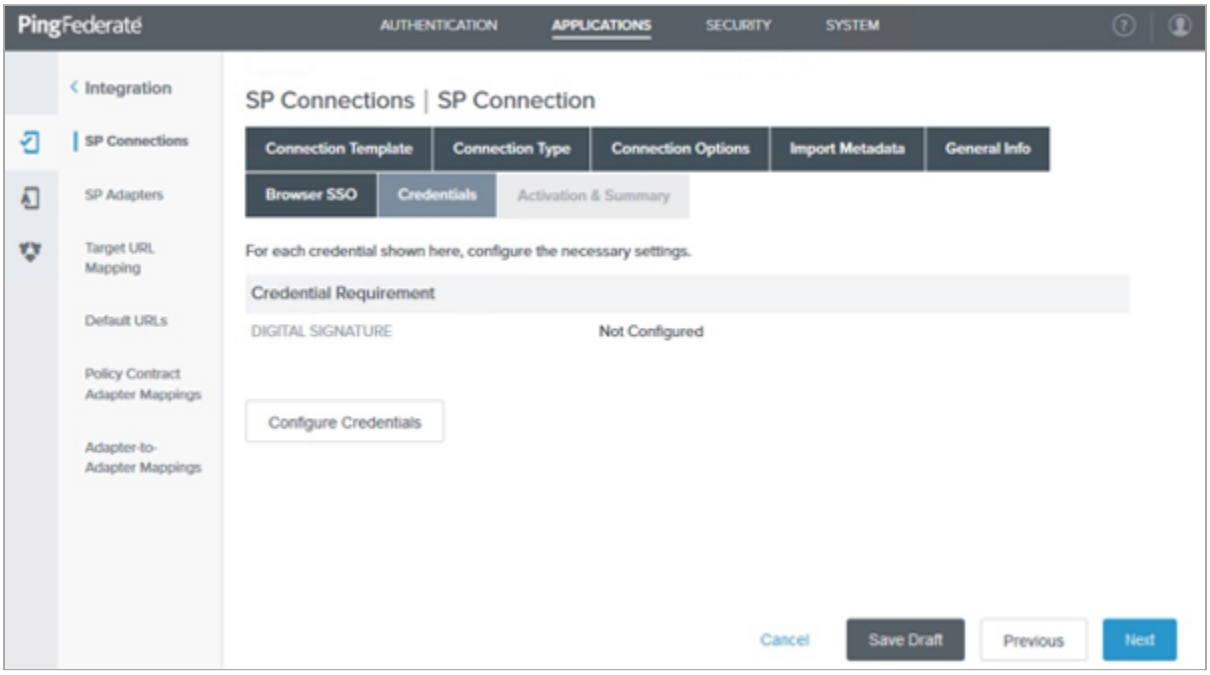
   You return to the **Browser SSO** section.

19. On the **Protocol Settings** tab, click **Next**.
20. On the **Summary** tab, click **Done**.

    Result:

    You return to the **SP Connection** section.

21. On the **Browser SSO** tab, click **Next**.
22. On the **Credentials** tab, click **Configure Credentials**.

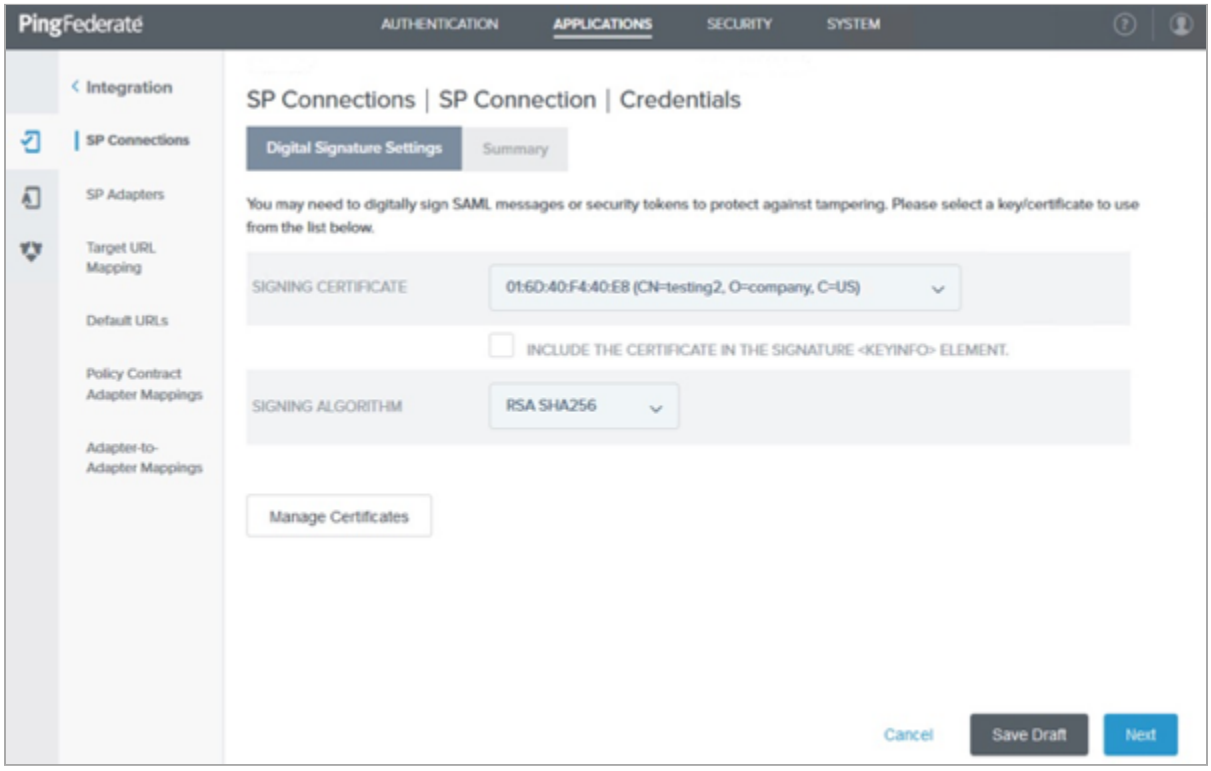


Result:

The tabs for the **Credentials** section display.

23. Configure the credentials:

    1. On the **Digital Signature Settings** tab, select the **Signing Certificate** that you chose in Exporting the signing certificate from PingFederate. Click **Next**.



    2. On the **Summary** tab, click **Done**.

Result:

You return to the **SP Connection** section.

24. On the **Credentials** tab, click **Next**.
25. On the **Activation & Summary** tab, click **Save**.