PingIdentity™

PingFederate®

SQL Password Credential Validator v1.3.4

User Guide

© 2005-2018 Ping Identity ® Corporation. All rights reserved.

PingFederate SQL Password Credential Validator User Guide
Version 1.3.4
March 2018

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

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

Trademarks
Ping Identity, the Ping Identity logo, PingFederate, PingID, PingOne, PingConnect, and PingEnable are registered trademarks of Ping Identity Corporation ("Ping Identity"). All other trademarks or registered trademarks are the property of their respective owners.

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

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

Contents

Purpose

This user guide is intended for use by PingFederate clients, who would like the ability to leverage a password credential validator against a SQL database.

Prerequisites

This document assumes that you already have the following installed and configured:

- A functional PingFederate environment, version 8.4+
- JDK version 8+
- A pre-configured SQL datastore with driver
- At least one IdP adapter for use as the primary form of authentication, so that it can be configured to leverage the SQL Password Credential Validator (PCV)
- At least one SP connection that can be configured with that IdP adapter as a primary form of authentication

Installation

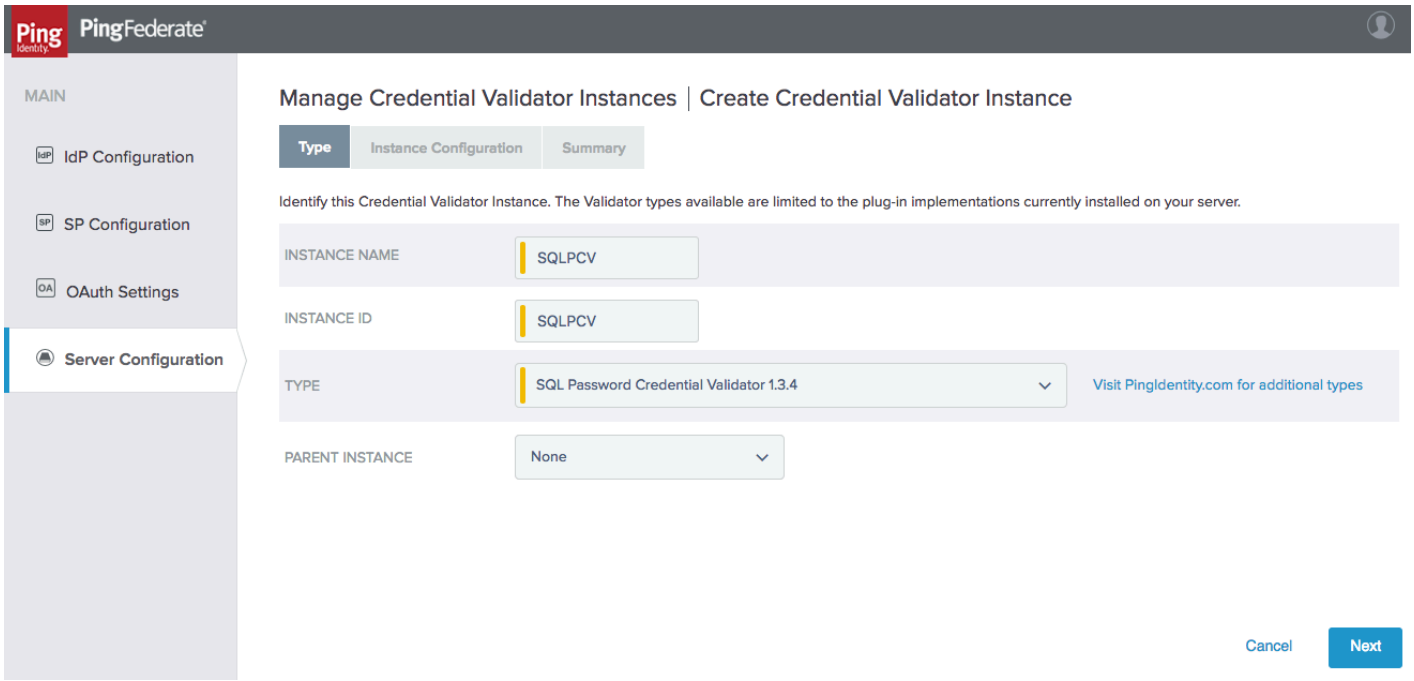
1. From the /dist folder in *pf-sql-password-credential-validator-1.3.4.zip*, copy the noted file to the following directory in your PingFederate:
   - <PingFederateInstall>/pingfederate/server/default/deploy/
     - pf-sql-password-credential-validator-1.3.4.jar
     - commons-dbutils-1.7.jar
2. Repeat step 1 on other clustered engine nodes.
3. Start or restart PingFederate.

Configuration

Configuring the SQL Password Credential Validator

Please note: this example is using Microsoft SQL Server for the SQL database.

1. Log into the PingFederate admin console and click **Password Credential Validators** under **Server Configuration >> Authentication**.
2. Ensure that a pre-configured SQL datastore has already been configured.
3. Click **Create New Instance…**
4. Enter the **Instance Name** and **Instance Id**, choose **SQL Password Credential Validator 1.3.4** and click **Next**.



5. Input the required information, and click **Next**.

SQL PCV using Regular Statement Processing Type:

PingFederate

MAIN

- IdP Configuration
- SP Configuration
- OAuth Settings
- Server Configuration

Manage Credential Validator Instances | Create Credential Validator Instance

Type | Instance Configuration | Summary

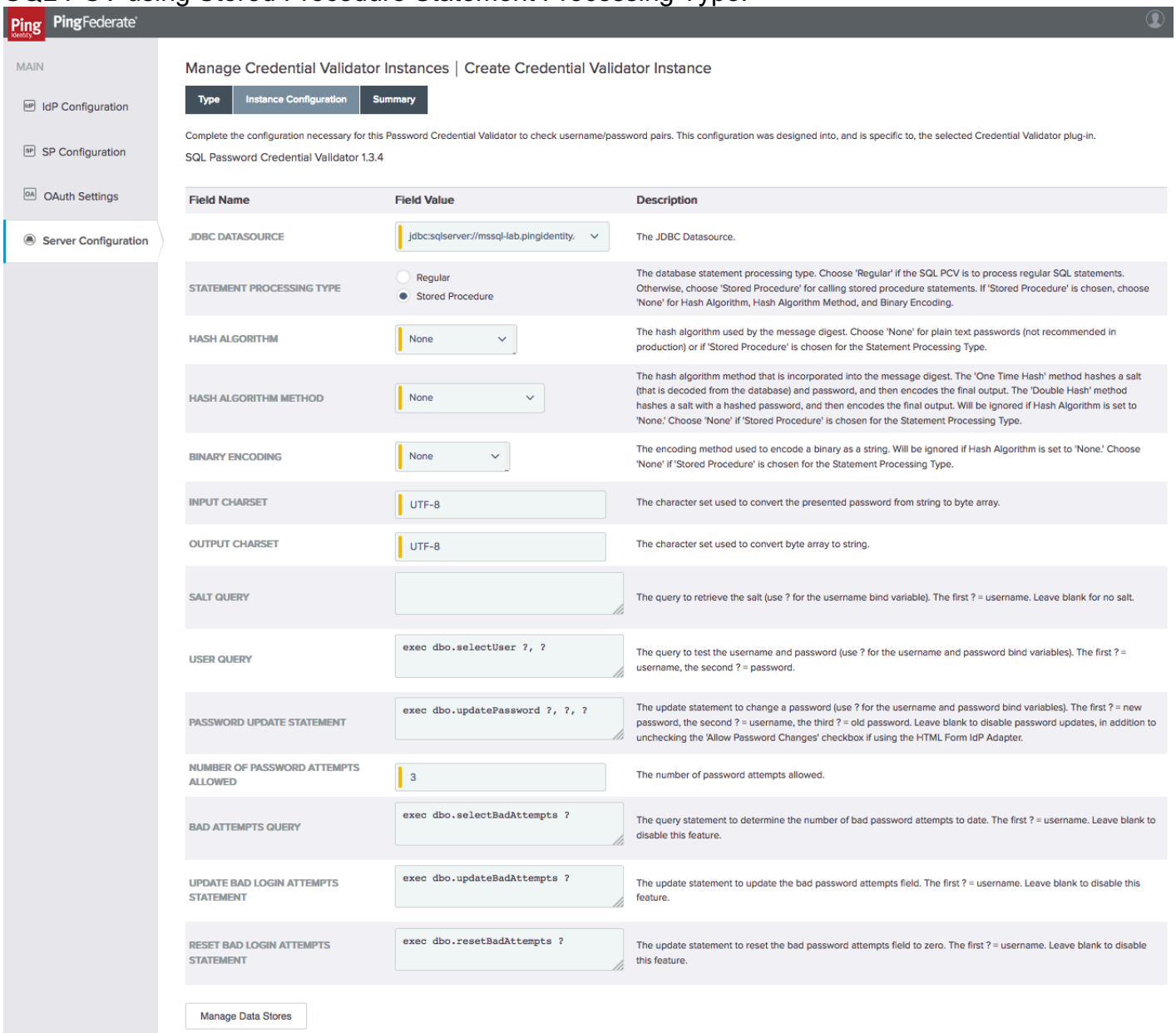
Complete the configuration necessary for this Password Credential Validator to check username/password pairs. This configuration was designed into, and is specific to, the selected Credential Validator plug-in.

SQL Password Credential Validator 1.3.4

Field Name	Field Value	Description
JDBC DATASOURCE	jdbc:sqlserver://mssql-lab.pingidentity.	The JDBC Datasource.
STATEMENT PROCESSING TYPE	● Regular ○ Stored Procedure	The database statement processing type. Choose 'Regular' if the SQL PCV is to process regular SQL statements. Otherwise, choose 'Stored Procedure' for calling stored procedure statements. If 'Stored Procedure' is chosen, choose 'None' for Hash Algorithm, Hash Algorithm Method, and Binary Encoding.
HASH ALGORITHM	SHA-256	The hash algorithm used by the message digest. Choose 'None' for plain text passwords (not recommended in production) or if 'Stored Procedure' is chosen for the Statement Processing Type.
HASH ALGORITHM METHOD	Double Hash	The hash algorithm method that is incorporated into the message digest. The 'One Time Hash' method hashes a salt (that is decoded from the database) and password, and then encodes the final output. The 'Double Hash' method hashes a salt with a hashed password, and then encodes the final output. Will be ignored if Hash Algorithm is set to 'None.' Choose 'None' if 'Stored Procedure' is chosen for the Statement Processing Type.
BINARY ENCODING	Hex	The encoding method used to encode a binary as a string. Will be ignored if Hash Algorithm is set to 'None.' Choose 'None' if 'Stored Procedure' is chosen for the Statement Processing Type.
INPUT CHARSET	UTF-8	The character set used to convert the presented password from string to byte array.
OUTPUT CHARSET	UTF-8	The character set used to convert byte array to string.
SALT QUERY	`select salt from dbo.users where username = ?`	The query to retrieve the salt (use ? for the username bind variable). The first ? = username. Leave blank for no salt.
USER QUERY	`select username from dbo.users where lower(username) = ? and password = ?`	The query to test the username and password (use ? for the username and password bind variables). The first ? = username, the second ? = password.
PASSWORD UPDATE STATEMENT	`update dbo.users set password = ? where username = ? and password = ?`	The update statement to change a password (use ? for the username and password bind variables). The first ? = new password, the second ? = username, the third ? = old password. Leave blank to disable password updates, in addition to unchecking the 'Allow Password Changes' checkbox if using the HTML Form IdP Adapter.
NUMBER OF PASSWORD ATTEMPTS ALLOWED	3	The number of password attempts allowed.
BAD ATTEMPTS QUERY	`select badattempts from dbo.users where username = ?`	The query statement to determine the number of bad password attempts to date. The first ? = username. Leave blank to disable this feature.
UPDATE BAD LOGIN ATTEMPTS STATEMENT	`update dbo.users set badattempts = badattempts + 1 where username = ?`	The update statement to update the bad password attempts field. The first ? = username. Leave blank to disable this feature.
RESET BAD LOGIN ATTEMPTS STATEMENT	`update dbo.users set badattempts = 0 where username = ?`	The update statement to reset the bad password attempts field to zero. The first ? = username. Leave blank to disable this feature.

Manage Data Stores

SQL PCV using Stored Procedure Statement Processing Type:



6. Review the configuration on the summary page, and click **Done**.

SQL PCV using Regular Statement Processing Type:

Ping Identity PingFederate

MAIN

- IdP Configuration
- SP Configuration
- OAuth Settings
- Server Configuration

Manage Credential Validator Instances | Create Credential Validator Instance

Type | Instance Configuration | Summary

Password Credential Validator configuration summary.

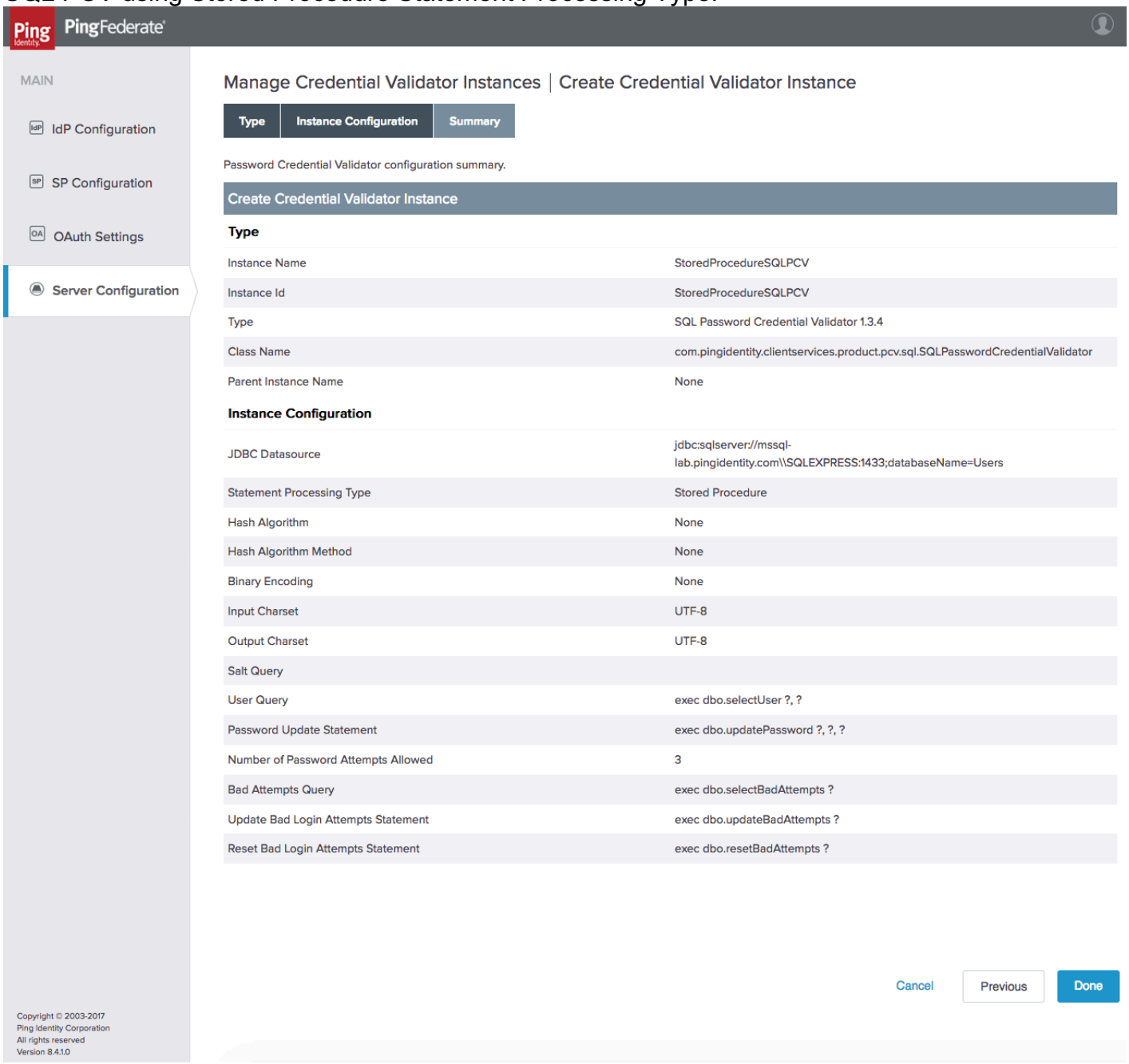
Create Credential Validator Instance

Type	
Instance Name	SQLPCV
Instance Id	SQLPCV
Type	SQL Password Credential Validator 1.3.4
Class Name	com.pingidentity.clientservices.product.pcv.sql.SQLPasswordCredentialValidator
Parent Instance Name	None
Instance Configuration	
JDBC Datasource	jdbc:sqlserver://mssql-lab.pingidentity.com\\SQLEXPRESS:1433;databaseName=Users
Statement Processing Type	Regular
Hash Algorithm	SHA-256
Hash Algorithm Method	Double Hash
Binary Encoding	Hex
Input Charset	UTF-8
Output Charset	UTF-8
Salt Query	select salt from dbo.users where username = ?
User Query	select username from dbo.users where lower(username) = ? and password = ?
Password Update Statement	update dbo.users set password = ? where username = ? and password = ?
Number of Password Attempts Allowed	3
Bad Attempts Query	select badattempts from dbo.users where username = ?
Update Bad Login Attempts Statement	update dbo.users set badattempts = badattempts + 1 where username = ?
Reset Bad Login Attempts Statement	update dbo.users set badattempts = 0 where username = ?

Cancel | Previous | Done

Copyright © 2003-2017
Ping Identity Corporation
All rights reserved
Version 8.4.1.0

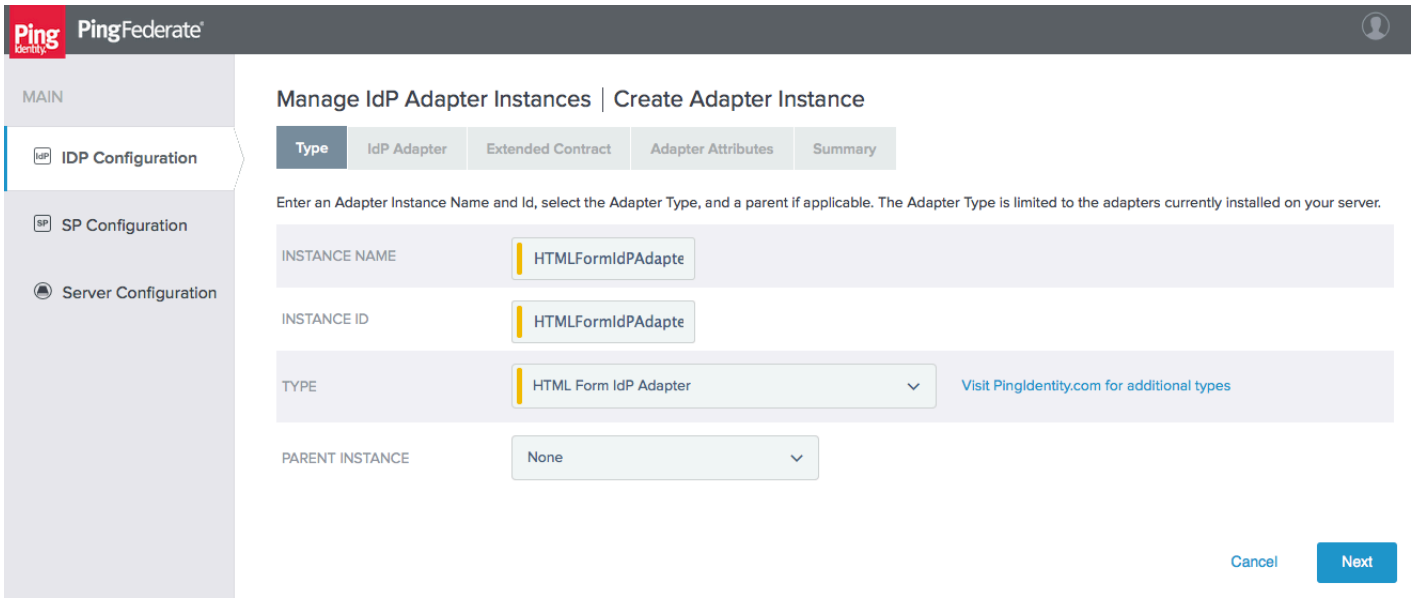
SQL PCV using Stored Procedure Statement Processing Type:



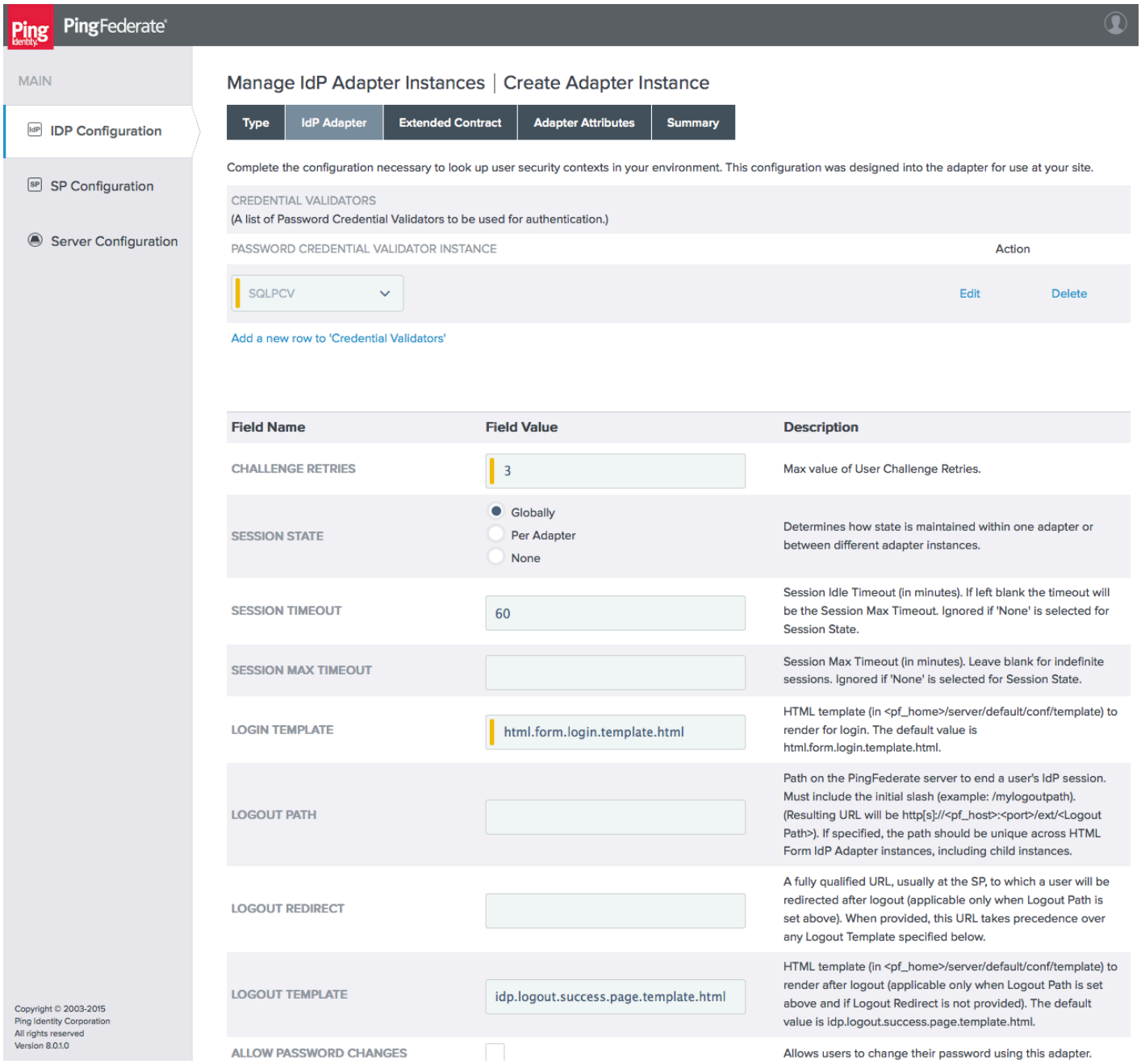
7. Click **Save**.

Configuring an IdP Adapter to Leverage the SQL Password Credential Validator

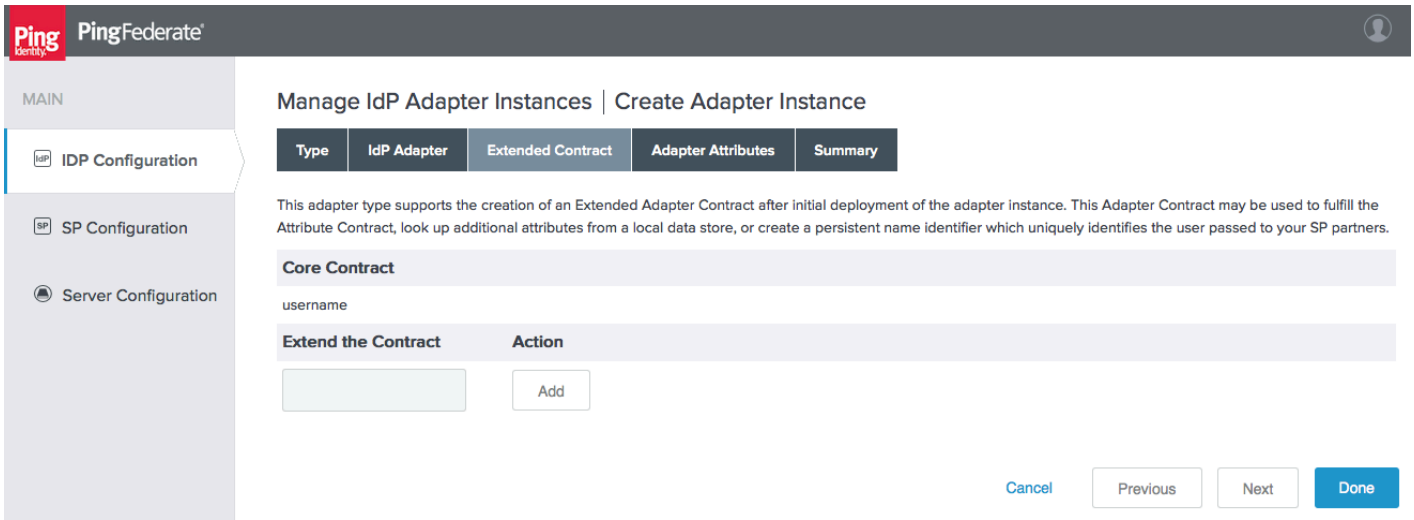
1. Click **Adapters** under **IdP Configuration >> Application Integration Settings**.
2. Click **Create New Instance…**
3. Enter the **Instance Name** and **Instance Id**, choose the adapter type (e.g., HTML Form IdP Adapter), and click **Next**.



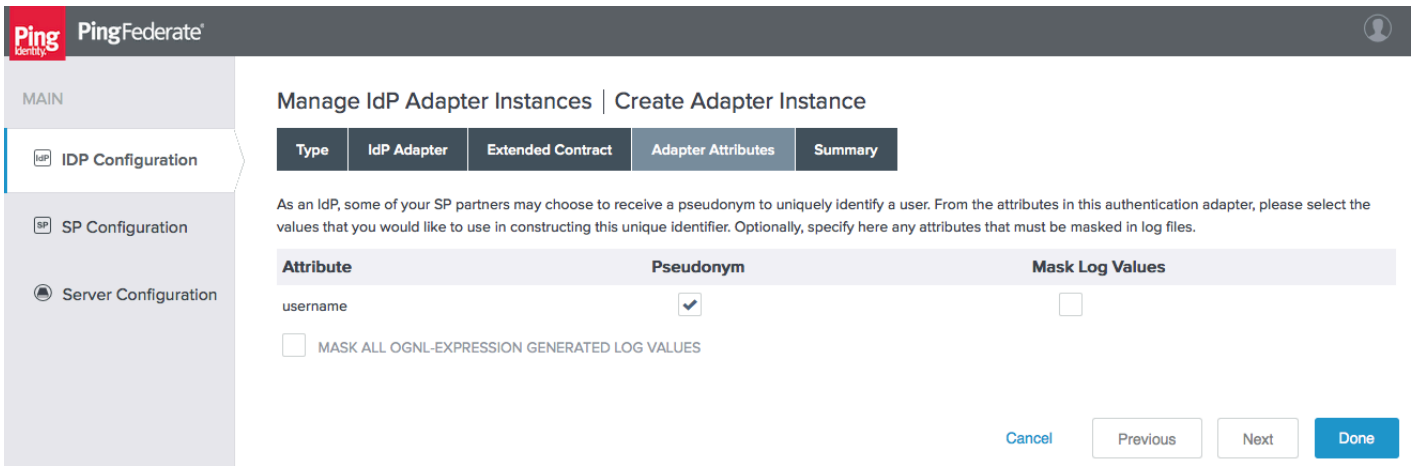
4. Click **Add a new row to 'Credential Validators.'**
5. Select the SQL Password Credential Validator created, and click **Update** (should change to **Edit** after updated).



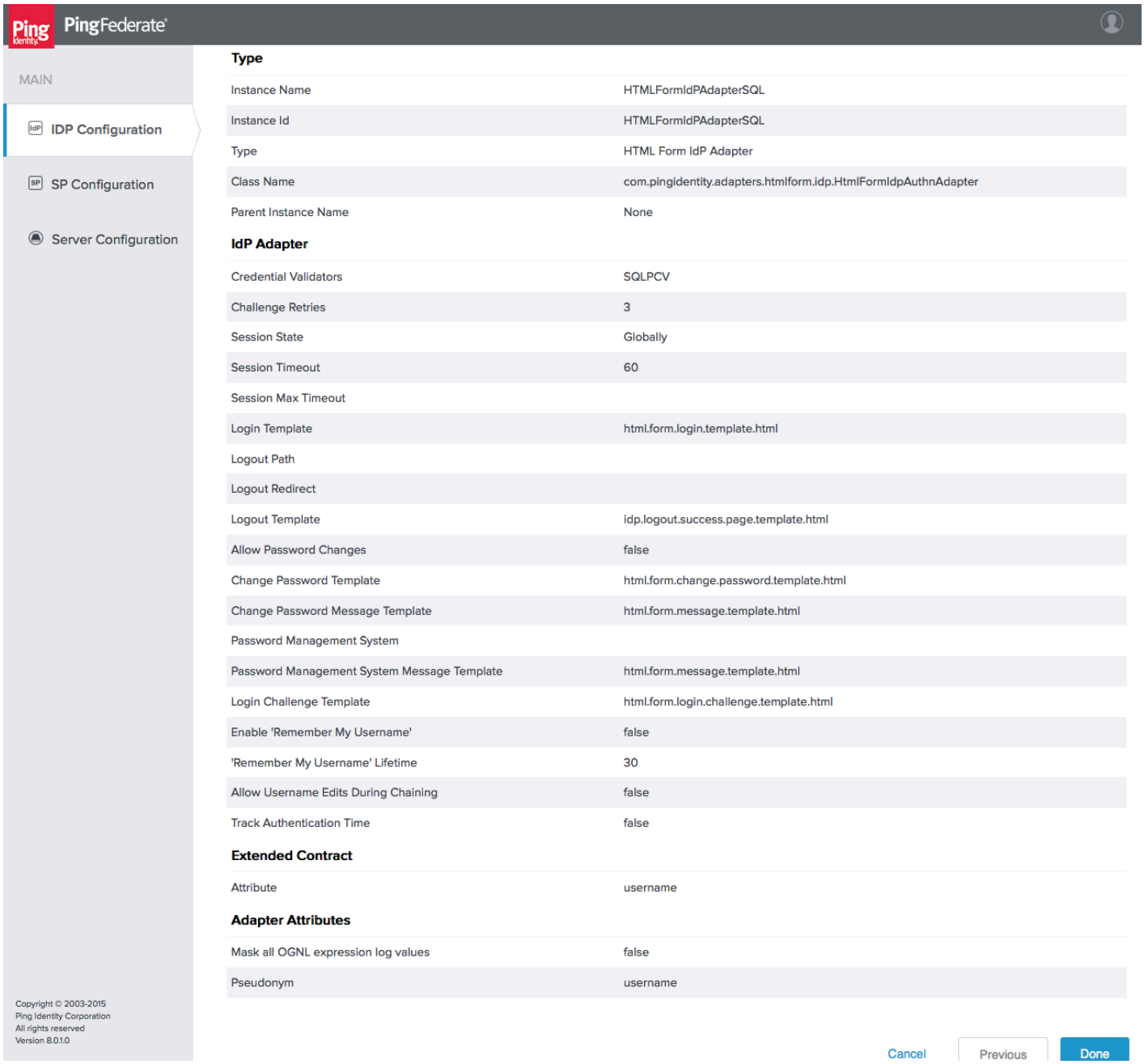
6. Modify the other configuration parameters on that page if needed, and click **Next**.
7. Extend the contract if needed, and click **Next**.



8. Select the **Pseudonym**, and click **Next**.



9. Review the configuration on the summary page, and click **Done**.

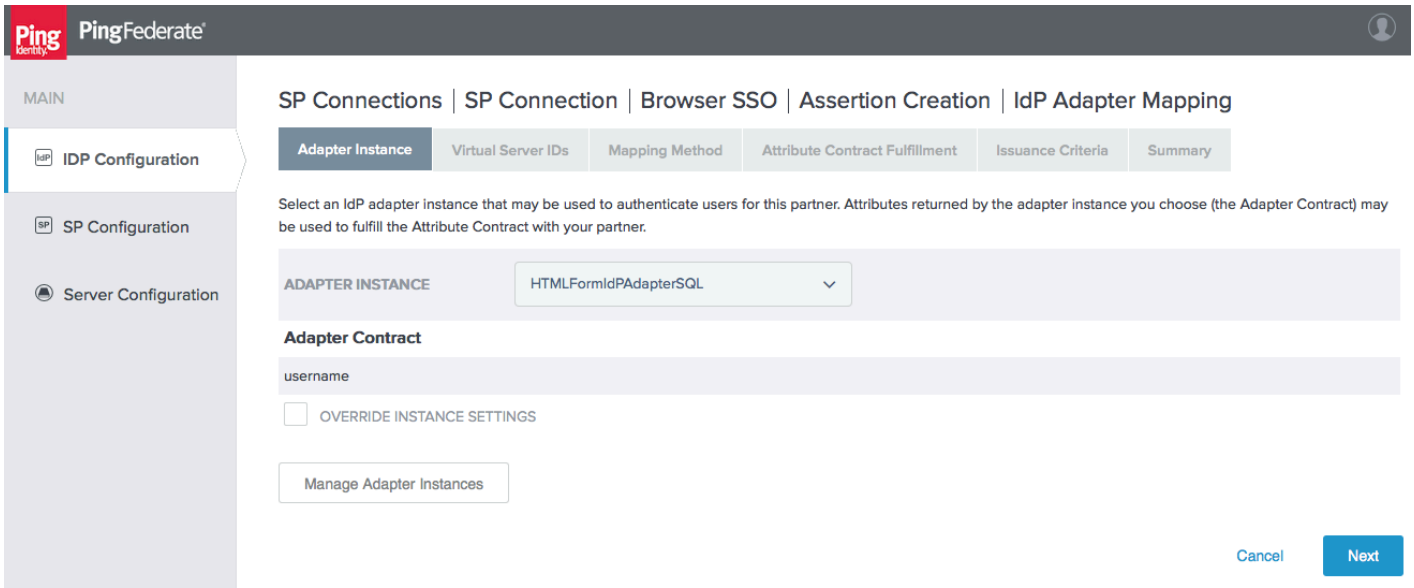


Type	
Instance Name	HTMLFormIdPAdapterSQL
Instance Id	HTMLFormIdPAdapterSQL
Type	HTML Form IdP Adapter
Class Name	com.pingidentity.adapters.htmlform.idp.HtmlFormIdpAuthnAdapter
Parent Instance Name	None
IdP Adapter	
Credential Validators	SQLPCV
Challenge Retries	3
Session State	Globally
Session Timeout	60
Session Max Timeout	
Login Template	html.form.login.template.html
Logout Path	
Logout Redirect	
Logout Template	idp.logout.success.page.template.html
Allow Password Changes	false
Change Password Template	html.form.change.password.template.html
Change Password Message Template	html.form.message.template.html
Password Management System	
Password Management System Message Template	html.form.message.template.html
Login Challenge Template	html.form.login.challenge.template.html
Enable 'Remember My Username'	false
'Remember My Username' Lifetime	30
Allow Username Edits During Chaining	false
Track Authentication Time	false
Extended Contract	
Attribute	username
Adapter Attributes	
Mask all OGNL expression log values	false
Pseudonym	username

10. Click **Save**.

Configuring the SP Connection to Leverage the SQL PCV-integrated Adapter

1. Click the SP connection to be modified, which should be located under **IdP Configuration >> SP Connections**.
2. Under **Assertion Creation**, click on **IdP Adapter Mapping**, and click **Map New Adapter Instance**.
3. Choose the adapter that was configured with the SQL Password Credential Validator, and click **Next**.

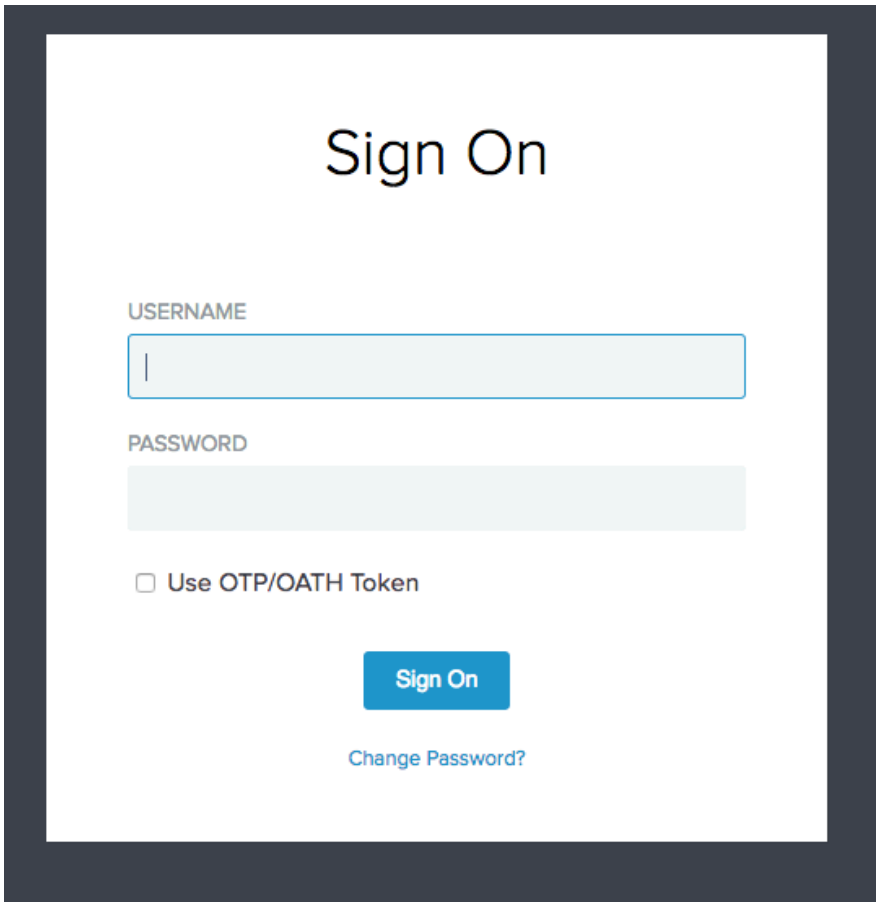


4. Select the appropriate Adapter Contract Assertion Mapping, and click **Next**.
5. Configure the Attribute Contract Fulfillment, and click **Next**.
6. Configure the appropriate Issuance Criteria (optional), and click **Next**.
7. Review the configuration on the summary page, and click **Done**.
8. Click **Done**.
9. Click **Done**.
10. Click **Save**.

Testing

Primary Test Case

1. Open a browser and go to the IdP login form chosen as the primary form of authentication. In this example, the HTML Form IdP Adapter leveraging the SQL PCV was chosen.



2. Log in using the test credentials.
3. If authorized into the target destination, authentication was a success.

Other Test Cases

Please note: For all test cases below, please make sure to log out, clear browser data, close and re-open the browser.

1. Repeat the primary test case as defined above, but with a test user that had too many bad login attempts (test this only if this use case is being used – i.e., if a query has been entered in the Update Bad Login Attempts Statement in the SQL Password Credential Validator configuration). Verify the number of bad login attempts in the SQL database.

2. Repeat the primary test case as defined above, but with a test user that is updating his/her password (test this only if this use case is being used – i.e., if the 'Allow Password Changes; checkbox has been checked in the adapter). Verify the password change by logging in with that password.

Logging

To enable various logging modes for the SQL Password Credential Validator, add the following in the relevant sections in <PingFederateInstall>/pingfederate/server/default/conf/log4j2.xml.

```
<Logger name="com.pingidentity.clientservices.product.pcv.sql " level="[ DEBUG | INFO |
```

```
WARN | ERROR ]" />
```

To enable logging for the SQL Password Credential Validator in a separate file, add the following in the relevant sections in <PingFederateInstall>/pingfederate/server/default/conf/log4j2.xml.

```
<RollingFile name="SQLPCV" fileName="${sys:pf.log.dir}/sqlpcv.log"
 filePattern="${sys:pf.log.dir}/sqlpcv.%d{yyyy-MM-dd}.log"
 ignoreExceptions="false">
        <PatternLayout>
                <!-- Uncomment this if you want to use UTF-8 encoding instead of system's
                default encoding.
                <charset>UTF-8</charset> -->
                <pattern>%d %m%n</pattern>
        </PatternLayout>
        <Policies>
                <TimeBasedTriggeringPolicy />
        </Policies>
</RollingFile>

<Logger name="com.pingidentity.clientservices.product.pcv.sql"
level="[ DEBUG | INFO | WARN | ERROR ]" additivity="false" includeLocation="true"/>
        <appender-ref ref="SQLPCV" />
</Logger>
```

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