PingOne Risk Integration Kit

Contents

PingOne Risk Integration Kit

The PingOne Risk Integration Kit allows PingFederate to communicate with PingOne Risk for risk-based authentication.

By sending transaction information and an optional device profile to PingOne when a user signs on, PingFederate can get a security risk assessment for the sign-on event. Including the risk assessment in your PingFederate authentication policy allows you to dynamically adjust the user's authentication requirements each time they sign on.

Components

- PingOne Risk IdP Adapter
  - When a user signs on through PingFederate, the adapter sends the transaction information to PingOne Risk, and retrieves a risk evaluation and other information about the user's current and previous transactions.
- Template and script files
  - When a user signs on through PingFederate and device profiling is enabled, these files create a device profile for the adapter to send to PingOne Risk.

Intended audience

This document is intended for PingFederate administrators.

If you need help during the setup process, see the following resources:

- *PingOne Risk* on the Ping Identity site
- *PingOne Risk* in the PingOne documentation
- The following sections of the PingFederate documentation:
  - *Managing IdP adapters*
  - *Authentication Policies*

System requirements

- PingFederate 10.2 or later
- To allow PingFederate to make outbound HTTPS connections, you might need to allow the following host names in your firewall:
  - https://api.pingone.com, https://api.pingone.asia, or https://api.pingone.eu
  - https://auth.pingone.com, https://auth.pingone.asia, or https://auth.pingone.eu
- A PingOne Risk license

  (If you don't have a license, you can create a trial account in *Creating an organization and environment in PingOne*)

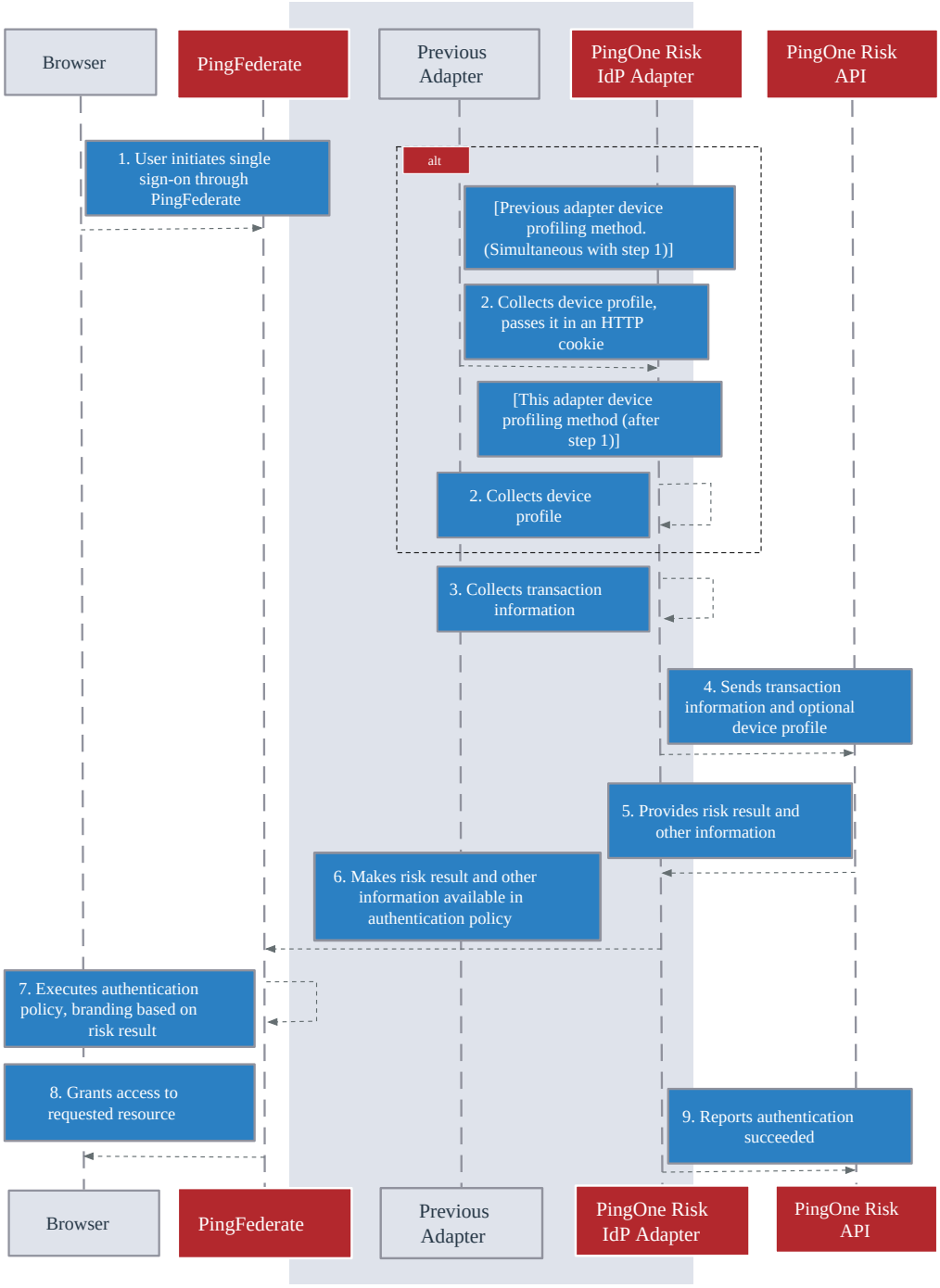
Overview of PingOne Risk

PingOne Risk evaluates the level of security risk for a user sign-on event based on a device profile and transaction information. For more information, see *PingOne Risk* in the PingOne documentation.

Overview of the SSO flow

With the PingOne Risk Integration Kit, PingFederate includes PingOne Risk in the sign-on flow.

The following figure shows how PingOne Risk is integrated into the sign-on process:



Description

1. A user initiates the sign-on process by requesting access to a protected resource.
2. When device profiling is enabled, one of the following occurs (depending on the device profiling method):
   - An adapter that is earlier in the authentication flow runs a script that creates a device profile. The script passes the device profile to the PingOne Risk IdP Adapter in a series of HTTP cookies.
   - The PingOne Risk IdP Adapter creates a device profile.
3. The PingOne Risk IdP Adapter collects transaction information, such as the user's IP address.
4. The adapter sends the transaction information and optional device profile to PingOne Risk.

5. PingOne Risk returns a JSON payload with the risk result and other information, such as the IP reputation, to the adapter.
6. The PingOne Risk IdP Adapter makes the risk result and other information available in the PingFederate authentication policy.
7. PingFederate executes the authentication policy, which branches based on the risk result provided by the adapter.
8. PingFederate returns the resource that the user requested.
9. The adapter notifies PingOne Risk whether authentication ultimately succeeded. This helps PingOne Risk evaluate subsequent sign-on attempts.

Device profiling methods

In addition to the basic transaction information, PingOne Risk can include an optional device profile in its risk evaluation. There are different methods for collecting the device profile depending on your preferences and whether users are authenticating directly through PingFederate or through the PingFederate authentication API.

Complete steps for each device profiling method are available when you reach the *Integrating device profiling* on page 8 part of the setup process.

Fingerprint JS and Signals SDK

Signals SDK is the preferred way to get device profiling and is recommended for use in the PingOne Risk Integration Kit 1.3 and later. For backwards compatibility, Fingerprint JS is also supported. Either option can be used for device profiling and can be used with each of the following scenarios.

> ⓘ **Note:**
>
> Some resource files might need to be updated if using Fingerprint JS.

Direct authentication mode - Captured by the PingOne Risk IdP Adapter

The PingOne Risk IdP Adapter inserts a page into the sign-on flow that shows a spinner animation while it collects the device profile. The page appears in the sign-on flow in the order set by your PingFederate authentication policy, typically after the first-factor sign-on page.

This method does not require modifications to any other pages, but adds a wait time to the sign-on process while the device profile is collected. The length of the wait depends on your environment.

Direct authentication mode - Captured by a previous adapter

You add a device profiling script to a page that appears earlier in the sign-on flow than the PingOne Risk IdP Adapter. The script creates the device profile and stores it in a series of HTTP cookies. When the PingOne Risk IdP Adapter is triggered by the PingFederate authentication policy, it picks up the device profile from the cookies. The adapter sends the device profile to PingOne Risk along with the transaction information.

By adding the device profiling script to your first-factor authentication page, for example, the device profile is created while the user types in their credentials. This can reduce the perceived wait time during sign on.

You can integrate the device profiling script into any page that meets the following criteria:

- The page appears before the PingOne Risk IdP Adapter in the sign-on flow.
- The page is hosted in the same domain as your PingFederate server. This is allows HTTP cookies to pass the transaction information to the PingOne Risk IdP Adapter. You might be able to work around this requirement by consolidating your domains with a reverse proxy server.

- You have access to the page to add the script.

Authentication API mode - Captured by your web application first-factor sign-on page
You add a device profiling script to your web application sign-on page. The script creates the device profile and stores it in a series of HTTP cookies. When the PingOne Risk IdP Adapter is triggered by the PingFederate authentication policy, it picks up the device profile from the cookies. The adapter sends the device profile to PingOne Risk along with the transaction information.

Setup

To configure the PingOne Risk Integration Kit, follow these steps.

1. *Deploying the integration files* on page 7
2. *Integrating device profiling* on page 8
3. *Configuring an adapter instance* on page 15
4. *Using custom risk predictors* on page 21
5. *Adding risk level results to your authentication policy* on page 23.
6. *Configuring PingFederate to forward IP addresses* on page 28.

Deploying the integration files

To get started with the integration, deploy the PingOne Risk Integration Kit files to your PingFederate directory.

Before you begin

Make sure you have completed the following tasks:

1. Create an environment in PingOne. For more information, see *Creating an environment*.
2. Create a custom risk policy in PingOne Risk. For more information, see *Risk policies*.
3. Connect PingFederate to PingOne Risk. For more information, see *Connecting PingFederate to PingOne*.

Procedure

1. Download the PingOne Risk Integration Kit `.zip` archive from the **Add-ons** tab of the *PingFederate downloads page*.
2. Stop PingFederate.
3. If you are upgrading an existing deployment, back up your customizations and delete earlier versions of the integration files.
   a. Back up any PingOne Risk Integration Kit files that you customized in `<pf_install>/pingfederate/server/default/conf/`.
   b. Delete `pf-pingone-risk-management-adapter-<version>.jar` from `<pf_install>/pingfederate/server/default/deploy`.

> ⓘ **Note:**
>
> If you are upgrading and not moving to Signals SDK, you must update the template HTML and JS files.

4. Extract the `.zip` archive and merge the contents of the `dist` directory with your `<pf_install>/pingfederate/server/default/deploy` directory.

5. If there is more than one version of the `pf-authn-api-sdk-<version>.jar` file in your `<pf_install>/pingfederate/server/default/lib` directory, delete all but the latest version of the file.
6. If you backed up any customized files, modify the new files with your customizations.
7. Start PingFederate.
8. If you operate PingFederate in a cluster, repeat steps 2-7 for each engine node.

Integrating device profiling

Device profiles are optional. If you want to send a device profile to PingOne Risk, complete one of the steps below.

About this task
For detailed descriptions of each method, see *Device profiling methods* on page 6.

Procedure

Do one of the following.

Method	Steps
Direct authentication mode - Captured by the PingOne Risk IdP Adapter.	No extra steps are necessary. Continue to *Configuring an adapter instance* on page 15.
Direct authentication mode - Captured by a previous adapter, such as the HTML Form Adapter.	Complete the steps in *Adding device profiling to a browser-based authentication page* on page 8.
Authentication API mode - Captured by your web application first-factor sign-on page.	Complete the steps in *Adding device profiling to a browser-based authentication page* on page 8.

ⓘ **Tip:**

For help with the authentication API, see *Authentication API* and *Exploring the Authentication API* in the PingFederate documentation.

Adding device profiling to a browser-based authentication page

There are two ways for device profiling data to be collected; by the PingOne Risk IdP Adapter, or by a previous adapter such as the HTML Form Adapter. Getting the device profile on the HTML Form Adapter page reduces the perceived wait times for the user. For more information, see *Device profiling methods* on page 6.

Adding device profiling to a browser-based authentication page using the PingOne Risk (Signals) SDK with integration kit 1.3.1

About this task

You can adapt these instructions to add device profiling to any page, such as the HTML Form Adapter or your web application. The page must meet the criteria listed in *Device profiling methods* on page 6.

ⓘ **Important:**

PingOne Risk Integration Kit version 1.3.1 must be deployed before any changes can be made on the HTML side. SDK version 5.2 and later requires adapter version 1.3.1 or later.

Procedure

1. If you are modifying an external web application, copy the following files from the integration `.zip` archive to a location that your page can access.
   - `pingone-risk-profiling-signals-sdk.js`
   - `pingone-risk-management-embedded.js`
   - `signals-sdk-<version>.js`
2. Optional: Edit the `pingone-risk-profiling-signals-sdk.js` file and add your PingOne environment ID.

```
function profileDevice(callback) {
    // Initialize the SDK
    // replace <envid> with the PingOne console > Environment >
 Environment ID value
    onPingOneSignalsReady(function () {
        _pingOneSignals.initSilent({
            envId: "<envid>",
            behavioralDataCollection: false,
            deviceAttributesBlackList: []
        }).then(function ()
```

3. Add the following external script references to the sign-on page:

> **Important:**
>
> The scripts must be added in the following order.

```
<script type="text/javascript" src="signals-sdk-<version>.js"></script>
<script type="text/javascript" src="pingone-risk-profiling-signals-
sdk.js"></script>
<script type="text/javascript" src="pingone-risk-management-
embedded.js"></script>
```

4. Optional: Customize the device profile cookie name prefix to suit your environment.
   a. Open `pingone-risk-management-embedded.js` for editing.
   b. On the following line, change the value to a name of your choosing:

```
var cookieNamePrefix = "pingone.risk.device.profile";
```

   c. Save the file.
5. When you complete the steps in *Configuring an adapter instance* on page 15, follow the instructions to set the **Device profiling method** to **Captured by a previous adapter**. Update the **Cookie Name Prefix** field if you customized it above.

Adding device profiling to a browser-based authentication page using the PingOne Risk (Signals) SDK and integration kit 1.3 or earlier

About this task

You can adapt these instructions to add device profiling to any page, such as the HTML Form Adapter or your web application. The page must meet the criteria listed in *Device profiling methods* on page 6.

Procedure

1. If you are modifying an external web application, copy the following files from the integration `.zip` archive to a location that your page can access.
   - `pingone-risk-profiling-signals-sdk.js`
   - `pingone-risk-management-embedded.js`
   - `signals-sdk.js`
2. Edit the `pingone-risk-profiling-signals-sdk.js` file and add your PingOne environment ID.

> ⓘ **Important:**
>
> The PingOne Risk SDK won't work if the PingOne environment ID is missing.

```
function profileDevice(callback) {
    // Initialize the SDK
    // replace <envid> with the PingOne console > Environment >
 Environment ID value
    onPingOneSignalsReady(function () {
        _pingOneSignals.initSilent({
            envId: "<envid>",
            deviceAttributesBlackList: []
        }).then(function ()
```

3. Add the following external script references to the sign-on page:

> ⓘ **Important:**
>
> The scripts must be added in the following order.

```
<script type="text/javascript" src="signals-sdk.js"></script>
<script type="text/javascript" src="pingone-risk-profiling-signals-
sdk.js"></script>
<script type="text/javascript" src="pingone-risk-management-
embedded.js"></script>
```

4. Optional: Customize the device profile cookie name prefix to suit your environment.
   a. Open `pingone-risk-management-embedded.js` for editing.
   b. On the following line, change the value to a name of your choosing:

   ```
   var cookieNamePrefix = "pingone.risk.device.profile";
   ```

   c. Save the file.
5. When you complete the steps in *Configuring an adapter instance* on page 15, follow the instructions to set the **Device profiling method** to **Captured by a previous adapter**. Update the **Cookie Name Prefix** field if you customized it above.

Adding device profiling to a browser-based authentication page using Fingerprint JS

About this task

You can adapt these instructions to add device profiling to any page, such as the HTML Form Adapter or your web application. The page must meet the criteria listed in *Device profiling methods* on page 6.

> ⓘ **Note:**

The PingOne Risk (Signals) SDK is the preferred way to get device profiling and is recommended for use in the PingOne Risk Integration Kit 1.3 and later.

Procedure

1. If you are modifying an external web application, copy the following files from the integration `.zip` archive to a location that your page can access.
   - `fingerprint2-<version>.min.js`
   - `pingone-risk-management-profiling.js`
   - `pingone-risk-management-embedded.js`
2. Add the following external script references to the sign-on page:

   Important:

   The scripts must be added in the following order.

```
<script type="text/javascript" src="fingerprint2-<version>.min.js"></
script>
<script type="text/javascript" src="pingone-risk-management-
profiling.js"></script>
<script type="text/javascript" src="pingone-risk-management-
embedded.js"></script>
```

3. Edit the `pingone-risk-management-embedded.js` file to use Fingerprint JS according to the comments in the file.
4. Optional: Customize the device profile cookie name prefix to suit your environment.
   a. Open `pingone-risk-management-embedded.js` for editing.
   b. On the following line, change the value to a name of your choosing:

```
var cookieNamePrefix = "pingone.risk.device.profile";
```

   c. Save the file.
5. When you complete the steps in *Configuring an adapter instance* on page 15, follow the instructions to set the **Device profiling method** to **Captured by a previous adapter**. Update the **Cookie Name Prefix** field if you customized it above.

Adding device profiling to an authentication API-based application

If you are using the PingFederate authentication API and want to avoid using HTTP cookies, modify your application to collect the device profile and send it to the authentication API.

Adding device profiling to an authentication API-based application using the PingOne Risk (Signals) SDK and integration kit 1.3.1

About this task

Use the following instructions to add device profiling to a web application using the PingOne Risk (Signals) SDK. For more information about the authentication API, see *Authentication API* in the PingFederate documentation or *PingOne Risk Native SDKs* in the PingOne Native SDK documentation.

Important:

PingOne Risk Integration Kit version 1.3.1 must be deployed before any changes can be made on the HTML side. SDK version 5.2.1 or later requires adapter version 1.3.1 or later.

Procedure

1. Implement the Signals SDK by following the steps in *PingOne Risk Native SDKs*.
2. Configure functions to format the device profile and submit it to the authentication API. Use the following code sample as a guide.

> ⓘ **Note:**
>
> The `pingone-risk-profiling-signals-sdk.js` assumes you are using the web SDK. If using a mobile device and you want to send the `deviceProfile` check the *SDK documentation* and use accordingly.

```
function onCompletion(flowId, deviceProfile) {
 submitDeviceProfile(flowId, deviceProfile);
}

function submitDeviceProfile(flowId, deviceProfile) {
 var myHeaders = new Headers();
 myHeaders.append("X-XSRF-Header", "test");
 myHeaders.append("Content-Type", "application/
vnd.pingidentity.submitDeviceProfile+json");
 var raw = JSON.stringify({
     "signalsSdkDeviceProfile": deviceProfile
 });
 var requestOptions = {
     method: 'POST',
     headers: myHeaders,
     body: raw,
     redirect: 'follow'
 };
 fetch("https://pf_host:pf_port/pf-ws/authn/flows/" + flowId,
 requestOptions)
     .then(response => response.text())
     .then(result => console.log(result))
     .catch(error => console.log('error', error));
}
```

When the authentication API is in the `DEVICE_PROFILE_REQUIRED` state, your application should `submitDeviceProfile` with the value from the `profileDevice(callback)` method.

3. When you complete the steps in *Configuring an adapter instance* on page 15, set the **Device profiling method** to **Captured by previous adapter**.

Adding device profiling to an authentication API-based application using the PingOne Risk (Signals) SDK and integration kit 1.3 or earlier

About this task

Use the following instructions to add device profiling to a web application using the PingOne Risk (Signals) SDK. For more information about the authentication API, see *Authentication API* in the PingFederate documentation or *PingOne Risk Native SDKs* in the PingOne Native SDK documentation.

Procedure

1. Implement the Signals SDK by following the steps in *PingOne Risk Native SDKs*.

2. Configure functions to format the device profile and submit it to the authentication API. Use the following code sample as a guide.

> ⓘ **Note:**
>
> The `pingone-risk-profiling-signals-sdk.js` assumes you are using the web SDK. If using a mobile device and you want to send the `deviceProfile` check the *SDK documentation* and use accordingly.

```
function onCompletion(flowId, deviceProfile) {
 submitDeviceProfile(flowId, deviceProfile);
}

function submitDeviceProfile(flowId, deviceProfile) {
 var myHeaders = new Headers();
 myHeaders.append("X-XSRF-Header", "test");
 myHeaders.append("Content-Type", "application/
vnd.pingidentity.submitDeviceProfile+json");
 var raw = JSON.stringify({
     "signalsSdkDeviceProfile": deviceProfile
 });
 var requestOptions = {
     method: 'POST',
     headers: myHeaders,
     body: raw,
     redirect: 'follow'
 };
 fetch("https://pf_host:pf_port/pf-ws/authn/flows/" + flowId,
 requestOptions)
     .then(response => response.text())
     .then(result => console.log(result))
     .catch(error => console.log('error', error));
}
```

When the authentication API is in the `DEVICE_PROFILE_REQUIRED` state, your application should `submitDeviceProfile` with the value from the `profileDevice(callback)` method.

3. When you complete the steps in *Configuring an adapter instance* on page 15, set the **Device profiling method** to **Captured by previous adapter**.

Adding device profiling to an Authentication API-based application using Fingerprint JS

About this task

Use the following instructions to add device profiling to a web application using Fingerprint JS. For more information about the authentication API, see *Authentication API* in the PingFederate documentation.

> ⓘ **Note:**
>
> The PingOne Risk (Signals) SDK is the preferred way to get device profiling and is recommended for use in the PingOne Risk Integration Kit 1.3 and later.

Procedure

1. Copy the following files from the integration `.zip` archive to your web server.
   - `fingerprint2-<version>.min.js`
   - `pingone-risk-management-profiling.js`

2. Add the following external script references to the sign-on page:

> ⓘ **Important:**
>
> The scripts must be added in the following order.

```
<script type="text/javascript" src="fingerprint2-<version>.min.js"></
script>
<script type="text/javascript" src="pingone-risk-management-
profiling.js"></script>
```

3. Configure your application to call the `profileDevice(onCompletion)` function. This creates the device profile.
4. Configure functions to format the device profile and submit it to the authentication API. Use the following code sample as a guide.

```
function onCompletion(flowId, components) {
    var deviceProfile = transformComponentsToDeviceProfile(components);
    submitDeviceProfile(flowId, deviceProfile);
}
function submitDeviceProfile(flowId, deviceProfile) {
    var myHeaders = new Headers();
    myHeaders.append("X-XSRF-Header", "test");
    myHeaders.append("Content-Type", "application/
vnd.pingidentity.submitDeviceProfile+json");
    var raw = JSON.stringify({
        "deviceProfile": deviceProfile
    });
    var requestOptions = {
        method: 'POST',
        headers: myHeaders,
        body: raw,
        redirect: 'follow'
    };
    fetch("https://pf_host:pf_port/pf-ws/authn/flows/" + flowId,
 requestOptions)
        .then(response => response.text())
        .then(result => console.log(result))
        .catch(error => console.log('error', error));
}
```

When the authentication API is in the `DEVICE_PROFILE_REQUIRED` state, your application should `submitDeviceProfile` with a value structured as follows:

```
{
   "deviceProfile":{
      "userAgent":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:82.0)
 Gecko/20100101 Firefox/82.0",
      "language":"en-CA",
      "colorDepth":24,
      "deviceMemory":null,
      "hardwareConcurrency":12,
      "screenResolution":[
         3440,
         1440
      ],
      "availableScreenResolution":[
         3440,
         1417
      ],
      "timezoneOffset":480,
```

```
      "timezone":"America/Los_Angeles",
      "sessionStorage":true,
      "localStorage":true,
      "indexedDb":true,
      "addBehaviour":false,
      "openDatabase":false,
      "cpuClass":null,
      "platform":"MacIntel",
      "plugins":[
      ],
      "webgl":[
         "
   [...]
   WEBGL_draw_buffers;WEBGL_lose_context",
         "webgl aliased line width range:[1, 1]",
         "webgl aliased point size range:[1, 8191]",
         [...]
         "webgl fragment shader low int precision rangeMin:24",
         "webgl fragment shader low int precision rangeMax:24"
      ],
      "webglVendorAndRenderer":"ATI Technologies Inc.~AMD Radeon Pro 560X
 OpenGL Engine",
      "adBlock":false,
      "hasLiedLanguages":false,
      "hasLiedResolution":false,
      "hasLiedOs":false,
      "hasLiedBrowser":false,
      "touchSupport":[
         "0",
         "false",
         "false"
      ],
      "fonts":[
         "Andale Mono",
         "Arial",
         [...]
         "Wingdings",
         "Wingdings 2",
         "Wingdings 3"
      ],
      "audio":"35.7383295930922"
   }
}
```


5. When you complete the steps in *Configuring an adapter instance* on page 15, set the **Device profiling method** to **Captured by this adapter**.

Configuring an adapter instance

To get started with the integration, deploy the PingOne Risk Integration Kit files to your PingFederate directory.

Procedure

1. In the PingFederate administrative console, go to **Authentication**# **Integration**# **IdP Adapters**. Click **Create New Instance**.

2. On the **Type** tab, set the basic adapter instance attributes.
   a. In the **Instance Name** field, enter a name for the adapter instance.
   b. In the **Instance ID** field, enter a unique identifier for the adapter instance.
   c. From the **Type** list, select **PingOne Risk IdP Adapter**. Click **Next**.
3. Optional: On the **IdP Adapter** tab, in the **Additional User Attributes** section, configure additional information to send to PingOne Risk.

   > ⓘ **Important:**
   >
   > A mapping is required for the **Session ID**, **Client Platform** and **Client Action** fields in **Client Attributes** table.

   The attributes are looked up in following order:

   - Chained input parameters (or API payload in the context of AuthN API)
   - Tracked parameters
   - Cookie

   a. Click **Add a new row to 'Additional User Attributes'**.
   b. In the **Incoming Attribute Name** field, enter the name of an attribute from any authentication source that appears earlier in your PingFederate authentication policy than the PingOne Risk IdP Adapter.
   c. From the **PingOne Risk Attribute** list, select the PingOne attribute that you want to populate.
   d. In the **Action** column, click **Update**.
4. Optional: On the **IdP Adapter** tab, map any custom risk predictors.

   For more information, see *Using custom risk predictors* on page 21.
5. Optional: On the **IdP Adapter** tab, in the **PingOne Risk API Response Mappings** section, map attributes from PingOne Risk Evaluation API response to the attribute contract. These attributes become available in your PingFederate authentication policy.
   a. Click **Add a new row to 'PingOne Risk API Response Mappings'**.
   b. In the **Local Attribute** field, enter a name of your choosing for an attribute.
   c. In the **PingOne Risk API Attribute Mapping** field, enter the JSON Pointer syntax for the source PingOne attribute as shown in *JSON Pointer syntax reference* on page 16.
   d. In the **Action** column, click **Update**.
   e. To add more attributes, repeat steps a-d.
6. On the **IdP Adapter** tab, configure the adapter instance by referring to *PingOne Risk IdP Adapter settings reference* on page 19. Click **Next**.
7. On the **Actions** tab, test your connection to PingOne Risk. Resolve any issues that are reported, and then click **Next**.
8. On the **Extended Contract** tab, add any attributes that you included in the **PingOne Risk API Response Mappings** section of the **IdP Adapter** tab. Click **Next**.
9. On the **Adapter Attributes** tab, set pseudonym and masking options as shown in *Set pseudonym and masking options* in the PingFederate documentation. Click **Next**.
10. On the **Adapter Contract Mapping** tab, configure the contract fulfillment details for the adapter as shown in *Define the IdP adapter contract* in the PingFederate documentation. Click **Next**.
11. On the **Summary** tab, check and save your configuration. Click **Save**.

JSON Pointer syntax reference

JavaScript Object Notation (JSON) Pointer defines a syntax for identifying a specific value within a JSON payload. Using the sample payload and JSON Pointer examples below, identify the attributes that you want to use to populate your attribute contract.

For a complete technical description of JSON Pointer syntax, see *JavaScript Object Notation (JSON) Pointer* on ietf.org.

Example PingOne Risk JSON payload


```
{
    "_links": {
     "self": {
      "href": "https://api.pingone.com/v1/environments/fe561fe9-e374-4de8-
be4c-0e38b3b83356/riskEvaluations/8d482f35-4ae5-4ecc-b9ff-db696a01f39f"
     },
     "environment": {
      "href": "https://api.pingone.com/v1/environments/fe561fe9-e374-4de8-
be4c-0e38b3b83356"
     },
     "event": {
      "href": "https://api.pingone.com/v1/environments/fe561fe9-e374-4de8-
be4c-0e38b3b83356/riskEvaluations/8d482f35-4ae5-4ecc-b9ff-db696a01f39f/
event"
     }
    },
    "id": "8d482f35-4ae5-4ecc-b9ff-db696a01f39f",
    "environment": {
     "id": "fe561fe9-e374-4de8-be4c-0e38b3b83356"
    },
    "createdAt": "2022-09-16T20:49:04.029Z",
    "updatedAt": "2022-09-16T20:49:04.029Z",
    "event": {
     "completionStatus": "IN_PROGRESS",
     "ip": "209.133.96.106",
     "flow": {
      "type": "AUTHENTICATION"
     },
     "user": {
      "id": "john",
      "name": "John DeMock",
      "type": "EXTERNAL",
      "groups": [
       {
        "name": "dev"
       },
       {
        "name": "sre"
       }
      ]
     },
     "origin": "PF_RISK_IK",
     "customProp": {
      "": {
       "nested": [
        "abcd",
        3.14
       ]
      }
     },
     "myNumber": 8.1,
     "isManaged": "No"
    },
    "riskPolicySet": {
     "id": "368a8459-3f46-4301-bc1e-f4214eb4a5f5",
     "name": "IK-2406"
    },
    "result": {
     "value": "Unmanaged device!",
     "level": "HIGH",
     "score": 1.8918918918918919,
```

```
      "source": "OVERRIDE",
      "type": "VALUE"
     },
     "details": {
      "ipAddressReputation": {
       "score": 0,
       "level": "LOW"
      },
      "anonymousNetworkDetected": false,
      "country": "united states",
      "state": "georgia",
      "city": "atlanta",
      "impossibleTravel": false,
      "ipVelocityByUser": {
       "level": "LOW",
       "threshold": {
        "source": "MIN_NOT_REACHED"
       },
       "velocity": {
        "distinctCount": 1,
        "during": 3600
       },
       "type": "VELOCITY"
      },
      "userLocationAnomaly": {
       "level": "HIGH",
       "reason": "New transaction location",
       "status": "NEW_LOCATION",
       "type": "USER_LOCATION_ANOMALY"
      },
      "userVelocityByIp": {
       "level": "LOW",
       "threshold": {
        "source": "MIN_NOT_REACHED"
       },
       "velocity": {
        "distinctCount": 1,
        "during": 3600
       },
       "type": "VELOCITY"
      },
      "geoVelocity": {
       "level": "LOW",
       "type": "GEO_VELOCITY"
      },
      "userRiskBehavior": {
       "reason": "Not enough information to assess risk score",
       "status": "NOT_AVAILABLE",
       "type": "USER_RISK_BEHAVIOR"
      },
      "ipRisk": {
       "level": "LOW",
       "type": "IP_REPUTATION"
      },
      "userBasedRiskBehavior": {
       "reason": "Not enough information to assess risk score",
       "status": "IN_TRAINING_PERIOD",
       "type": "USER_RISK_BEHAVIOR"
      },
      "anonymousNetwork": {
       "level": "LOW",
       "type": "ANONYMOUS_NETWORK"
      },
      "myNumber": {
```

```
       "level": "HIGH",
       "reason": "Attribute ${event.myNumber} is 8.1.",
       "attribute": "${event.myNumber}",
       "value": 8.1,
       "type": "MAP"
      },
      "nested": {
       "reason": "Not enough information to assess risk score",
       "status": "NOT_AVAILABLE",
       "attribute": "${event.customProp.nested}",
       "type": "MAP"
      },
      "isManaged": {
       "level": "HIGH",
       "reason": "Attribute ${event.isManaged} is \"No\".",
       "attribute": "${event.isManaged}",
       "value": "No",
       "type": "MAP"
      }
     }
}
```

JSON Pointer syntax

Description	JSON Pointer	Example value
Minimum Risk Module Policy Score	/details/policies/ MIN_RISK_MODULE/score	`501`
Minimum Risk Module Policy Level	/details/policies/ MIN_RISK_MODULE/level	`MEDIUM_RISK`

ⓘ **Note:**

To populate an attribute with the entire JSON response, leave the **PingOne Risk API Attribute Mapping** field blank.

PingOne Risk IdP Adapter settings reference

Field descriptions for the PingOne Risk IdP Adapter configuration screen.

Standard fields

Field	Value	Description
PingOne Environment	*<PingOne Connection>*	Your PingOne Environment. Create connections in **System**# **External Systems**# **PingOne Connections**. This field is blank by default.

Field	Value	Description
PingOne Risk Policy	*<Default Risk Policy>*	The risk policy used by PingOne for the risk evaluation. Overrides the environment and global default policy selections. This list is populated when you select a PingOne Environment.

Advanced fields

Field	Value	Description
Include Device Profile		When enabled, PingFederate will include a device profile in the risk evaluation. This option is selected by default.
Device Profiling Method	▪ Captured by this adapter (default) ▪ Captured by a previous adapter	When **Include Device Profile** is enabled, this setting determines whether the adapter will collect the device profile itself, or look for a device profile provided by another adapter that appears earlier in the authentication flow. If you manually added the device profiling script to an authentication page, select **Captured by a previous adapter**. Otherwise, select **Captured by this adapter**.
Device Profiling Timeout	`5000` (default)	The amount of time in milliseconds that PingFederate waits for the device profiling script to collect device details. Applies only if **Device Profiling Method** is **Captured by this adapter**.
Cookie Name Prefix	`pingone.risk.device.profile`	The name prefix of the cookie(s) that contains the device profile captured by a previous adapter. Applies only if **Device Profiling Method** is **Captured by a previous adapter**. The default value is `pingone.risk.device.profile`.
Failure Mode	▪ Continue with fallback policy decision ▪ Fail (default)	When PingOne Risk is unavailable or an error occurs, this setting determines whether the user's sign-on attempt should fail or continue with a pre-determined policy decision.

Field	Value	Description
Fallback Policy Decision Value	`MEDIUM` (default)	The risk result (for example, `MEDIUM`, `HIGH`, or unknown) to use in the authentication policy when the PingOne Risk service is unavailable or an error occurs, and **Failure Mode** is set to **Continue with fallback policy decision**.
Risk Policy Override		The attribute that contains the Risk policy ID to use when doing evaluations. This field is blank by default.
Device Profiling Script	▪ Fingerprint JS ▪ Signals SDK (default)	The script used to gather the client device profile.
API Request Timeout	`2000`	The amount of time in milliseconds that PingFederate allows when establishing a connection with PingOne Risk or waiting for a response to a request. A value of 0 disables the timeout. The default value is `2000`.
Proxy Settings		Defines proxy settings for outbound HTTP requests. The default value is **System Defaults**.
Custom Proxy Host		The proxy server hostname to use when **Proxy Settings** is set to **Custom**. This field is blank by default.
Custom Proxy Port		The proxy server port to use when **Proxy Settings** is set to **Custom**. This field is blank by default.

Using custom risk predictors

The PingOne Risk IdP Adapter lets you use attributes from your PingFederate authentication flow as risk predictors in PingOne Risk.

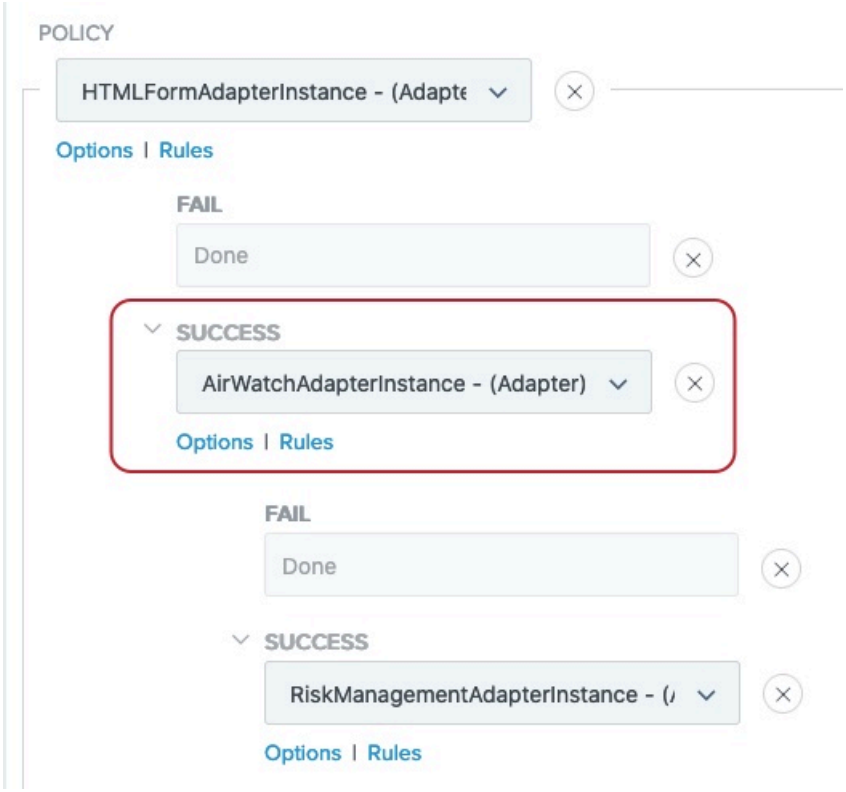
About this task

For an overview of risk predictors, see *Risk predictors* in the PingOne documentation.

The example in the steps below shows how to include the device security state from a mobile device management (MDM) service in the PingOne Risk risk evaluation.

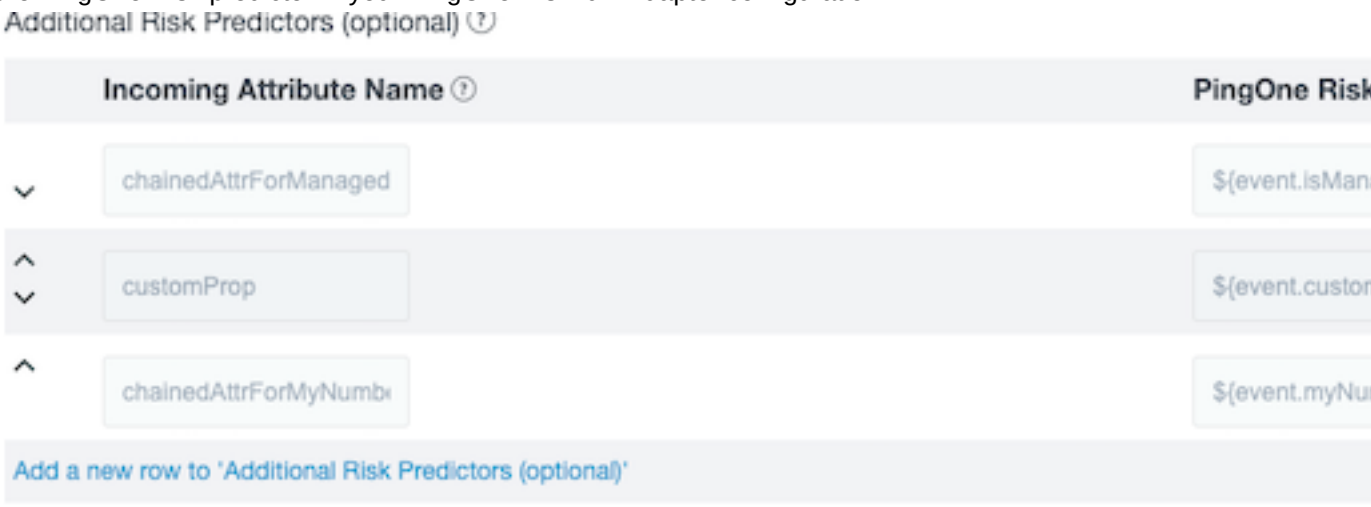
Procedure

1. Make the predictor available as an attribute in your PingFederate authentication policy.



   a. Add the source of the predictor data to your authentication policy.
   MDM example: Add a mobile device management adapter. On the **Extended Contract** tab of the configuration for that adapter instance, the attribute that holds the security state of the user's device is called `ComplianceStatus`.
   b. Later in the flow, add the PingOne Risk IdP Adapter that you configured in *Configuring an adapter instance* on page 15.
2. In PingOne Risk, add the risk predictor and include it in your risk policy. For help, see *Risk predictors* in the PingOne documentation.
   MDM example: Add a predictor with the JSON pointer `${event.ComplianceStatus}`.

3. In the **Risk Predictors** table of your PingOne Risk IdP Adapter configuration, map the predictor attribute from your PingFederate authentication policy to the JSON pointer you defined in PingOne Risk. For help, see *Configuring an adapter instance* on page 15.
   MDM example: Map the PingFederate `chainedAttForManaged` attribute to the PingOne Risk predictor in your PingOne Risk IdP Adapter configuration.



   During the authentication flow, the PingOne Risk IdP Adapter gets the predictor attribute from the PingFederate authentication policy and passes it to PingOne Risk. Next, PingOne Risk compares the value to the risk levels you defined and includes it in the risk evaluation.

Adding risk level results to your authentication policy

By modifying your PingFederate authentication policy to include the risk evaluation from PingOne Risk, you can dynamically change authentication requirements based on security risk level.

Before you begin

The steps in this topic assume that an HTML Form Adapter exists for login purposes. For more information on creating an HTML Form Adapter for login, see *Configuring an HTML Form Adapter instance*.

About this task

These steps are designed to help you add to an existing authentication policy. For general information about configuring authentication policies, see *Authentication API* in the PingFederate documentation.

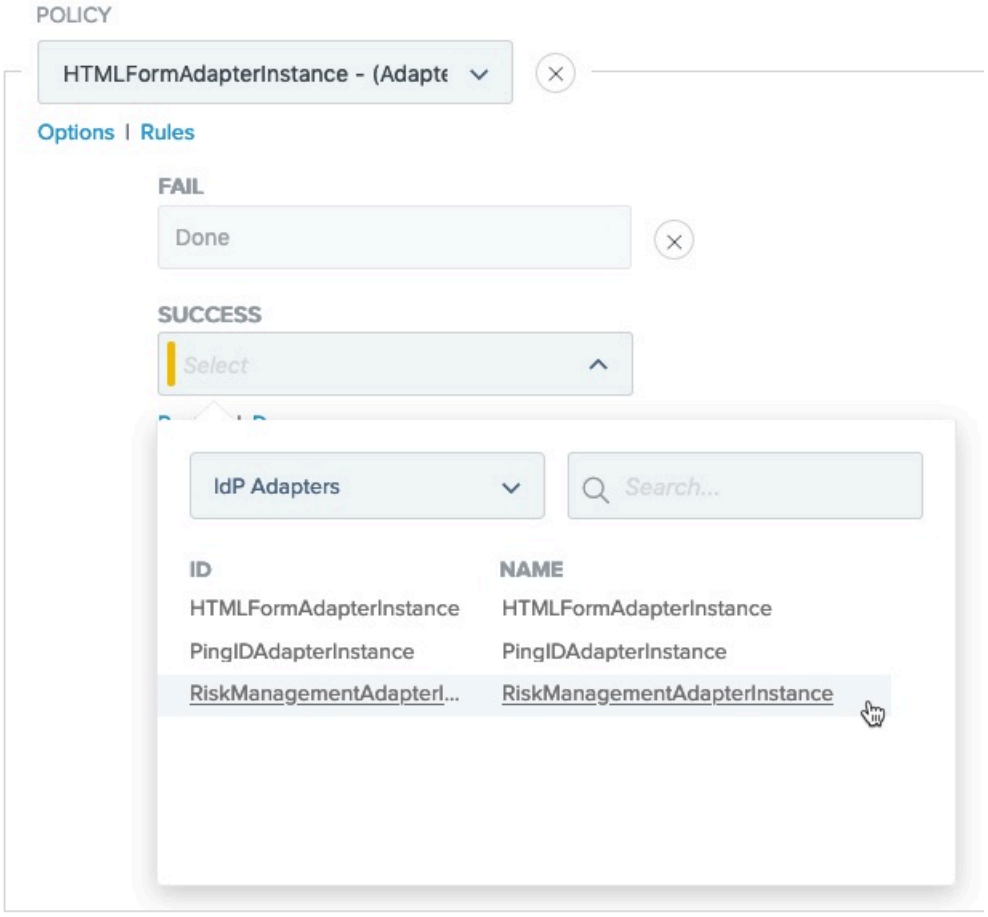
For new deployments, we recommend that you allow for a training period. To do this, configure your policy to pass traffic through the PingOne Risk IdP Adapter and continue regardless of the risk evaluation result. When you are ready to end the training period, adjust your authentication policy as described below.

> ⓘ **Important:**
>
> When the authentication flow finishes, PingFederate informs PingOne Risk whether the user ultimately succeeded or failed. This is an important consideration when designing your authentication flow.
>
> For example, a user receives a risk evaluation of `HIGH`, but ultimately completes the PingFederate authentication policy successfully. Based on that success, PingOne Risk now considers the user authenticated and lowers the risk evaluation to `MEDIUM` or `LOW` on the next attempt.

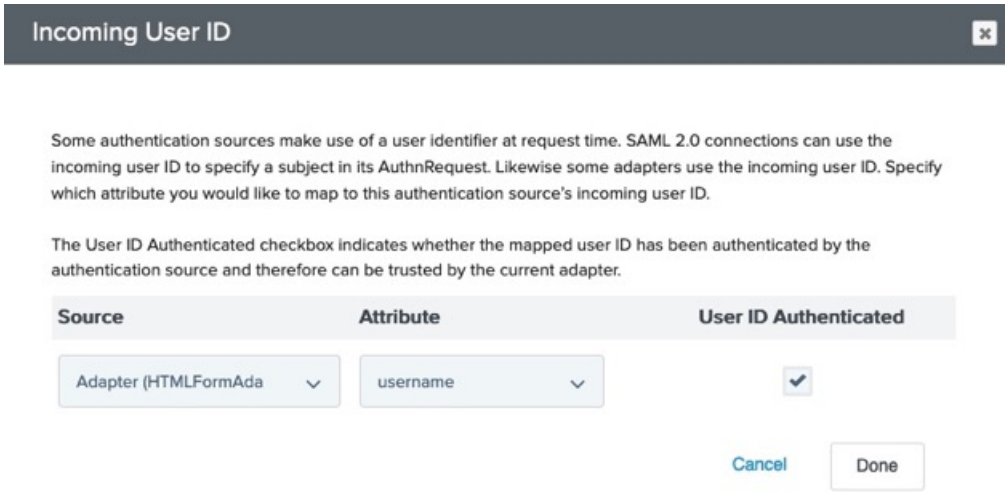
Procedure

1. On the PingFederate administrative console, go to **Policies**# **Authentication**# **Policies**# **Policies**.

2. Select the **IdP Authentication Policies** check box.
3. Open an existing authentication policy, or click **Add Policy**. See *Defining authentication policies* in the PingFederate documentation.
4. In the **Policy** area, from the **Select** list, select a PingOne Risk IdP Adapter instance.

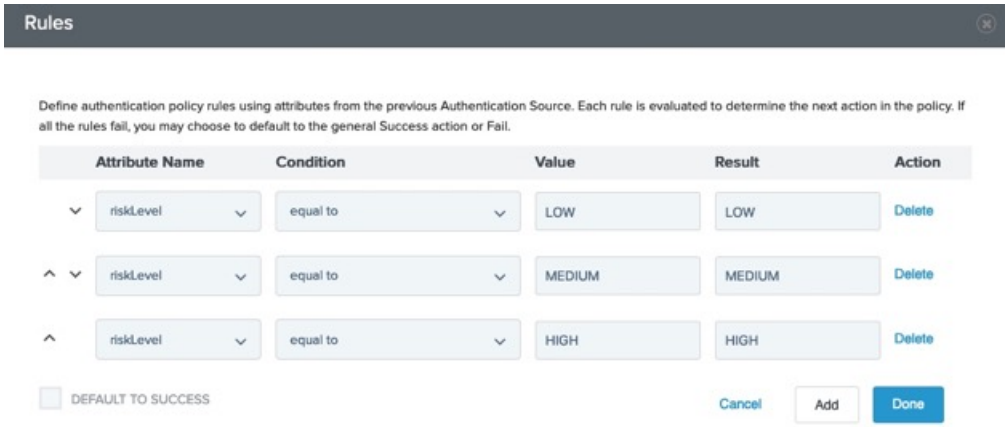


5. Map the user ID into the PingOne Risk IdP Adapter instance.



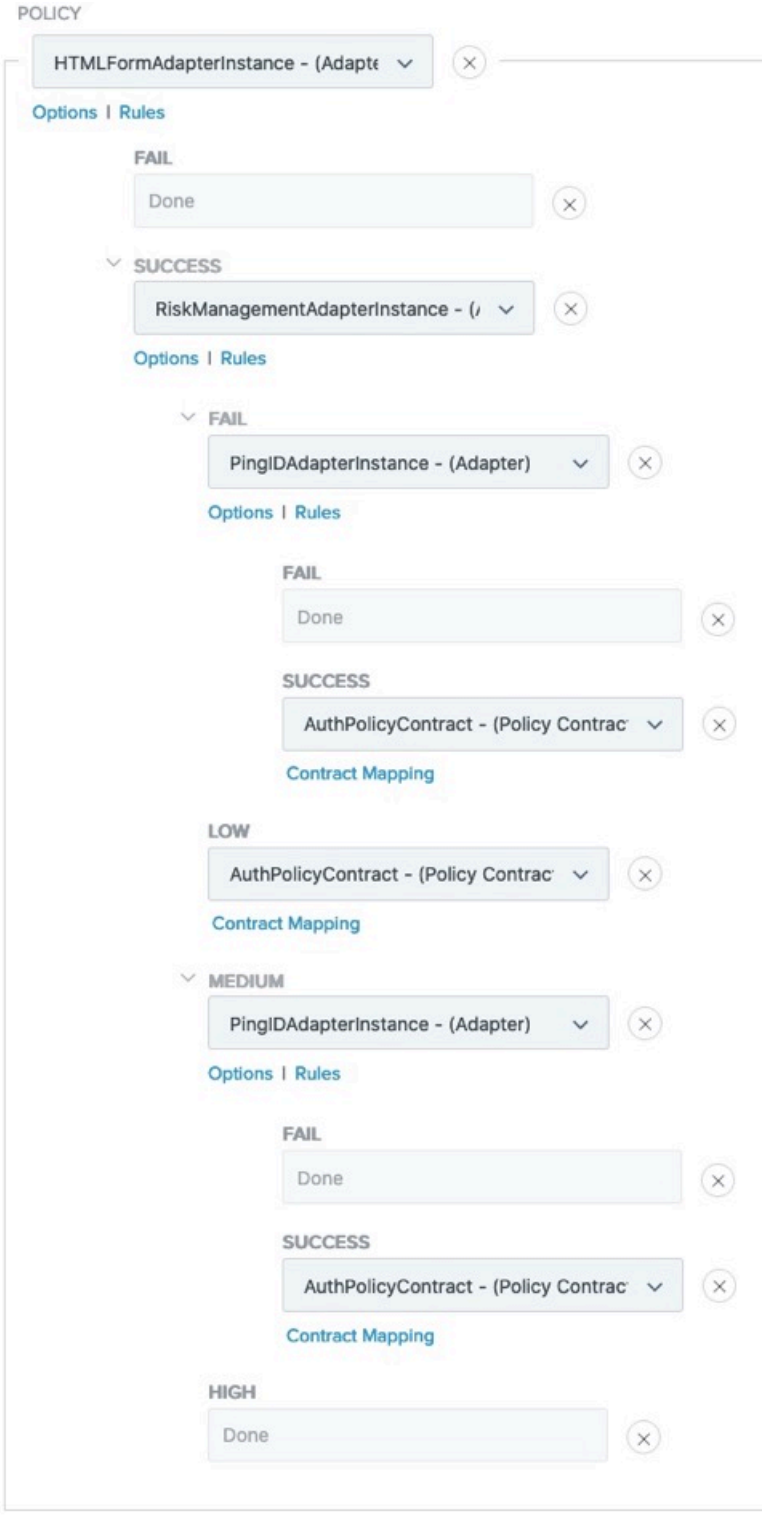
   a. Under the PingOne Risk IdP Adapter instance, click **Options**.
   b. On the **Options** dialog, from the **Source** list, select a previous authentication source that collects the user ID.
   c. From the **Attribute** list, select the user ID. Click **Done**.

6. Define policy paths based on risk results.



a. Under the PingOne Risk IdP Adapter instance, click **Rules**.
b. On the **Rules** dialog, from the **Attribute Name** list, select **riskLevel** or **riskValue**.
c. From the **Condition** list, select **equal to**.
d. In the **Value** field, if you selected **riskLevel**, enter `LOW`, `MEDIUM`, or `HIGH`. If you selected **riskValue**, enter one of the risk values that you configured in PingOne.
e. In the **Result** field, enter a name. This appears as a new policy path that branches from the authentication source.
f. If you want to add more policy paths, click **Add** and repeat steps b-e.
g. Optional: Clear the **Default to success** check box.
h. Click **Done**.

7. Complete the authentication policy.
   a. Configure each of the policy paths.
   b. Optional: Allow users continue to sign on by satisfying stricter authentication requirements when PingOne Risk is unreachable or returns an error. Do one of the following:
      - In your PingOne Risk IdP Adapter instance, set the **Failure mode** as shown in *PingOne Risk IdP Adapter settings reference* on page 19.
      - In your authentication policy, set the **Fail** outcome of the PingOne Risk IdP Adapter instance to point to a second authentication factor, as shown in the example below.



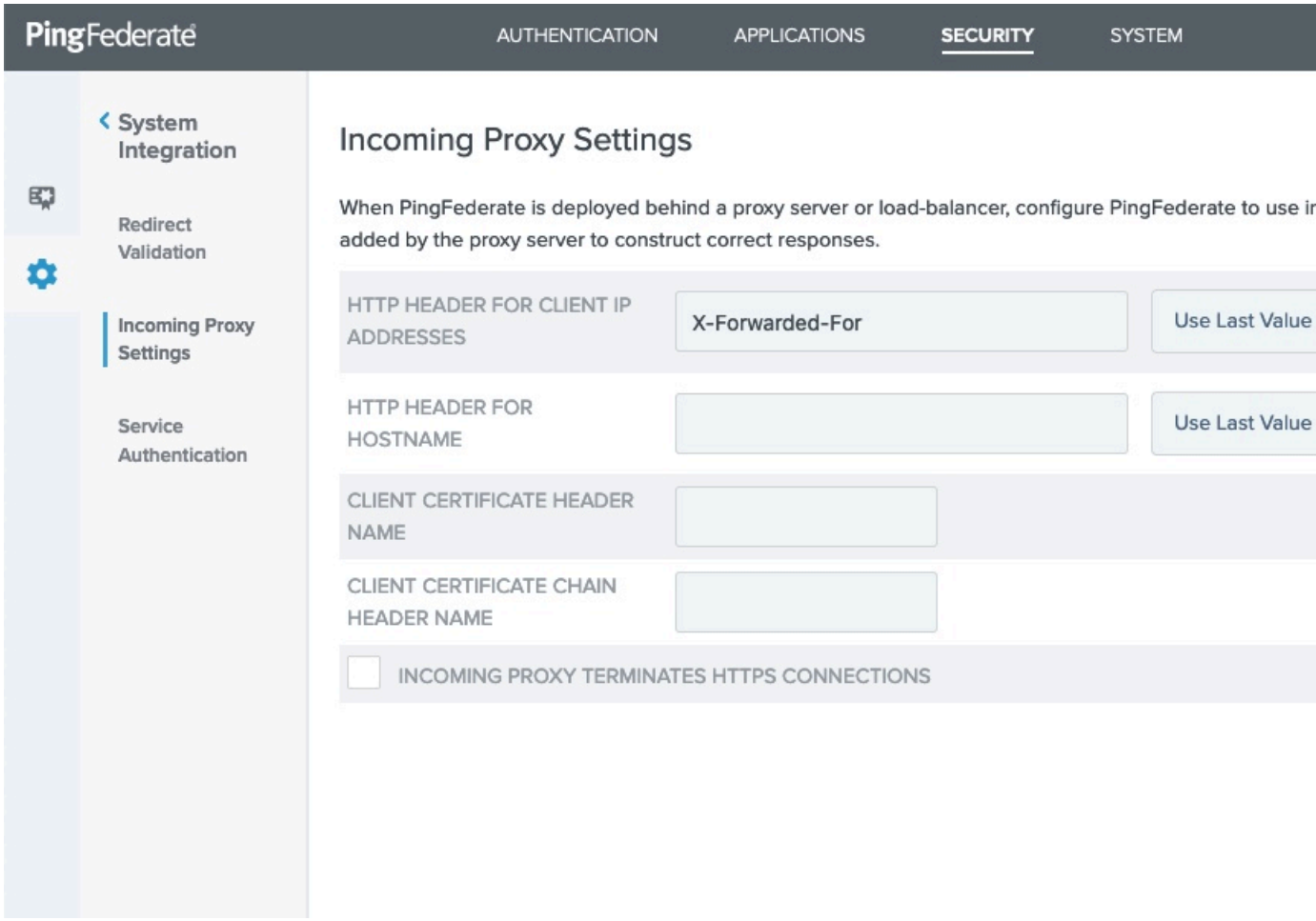
8. Click **Done**.
9. On the **Policies** window, click **Save**.

Configuring PingFederate to forward IP addresses

The PingOne Risk service needs the client's public-facing (rather than local) IP address for the risk assessment.

Procedure

1. On the PingFederate administrative console, go to **Security**# **System Integration**# **Incoming Proxy Settings**.
2. In the **HTTP Header for Client IP Addresses** field, enter `X-Forwarded-For`.
3. Leave **Use Last Value** selected.
4. Click **Save**.



Troubleshooting

Enabling debug logging

To help with troubleshooting or monitoring, you can turn on activity logging for PingFederate, the PingOne Risk IdP Adapter, or both.

About this task

This task is optional. You can use logging for troubleshooting or analytics.

For general information about logging, see *Enabling debug messages and console logging* in the PingFederate documentation.

Procedure

1. Open the `<pf_install>/pingfederate/server/default/conf/log4j2.xml` file for editing.
2. To log activity for PingFederate and all adapters:
   a. Find the following section in the file.

```
<AsyncRoot level="INFO" includeLocation="false">
 <!-- <AppenderRef ref="CONSOLE" /> -->
 <AppenderRef ref="FILE" />
</AsyncRoot>
```

   b. Change `INFO` to `DEBUG`.

      The following code snippet shows `DEBUG` in bold for visibility.

```
<AsyncRoot level="DEBUG" includeLocation="false">
 <!-- <AppenderRef ref="CONSOLE" /> -->
 <AppenderRef ref="FILE" />
</AsyncRoot>
```

   c. Optional: To see the adapter activity in the console as well as the log file, remove the comment tags (`<!--` and `-->`) that surround the `CONSOLE` line.

```
<AsyncRoot level="INFO" includeLocation="false">
 <AppenderRef ref="CONSOLE" />
 <AppenderRef ref="FILE" />
</AsyncRoot>
```

3. If you want to log activity relating to the PingOne Risk IdP Adapter, do one of the following.

> ⓘ **Tip:**
>
> You can use this information with a third-party log analysis tool to monitor for important events, such as when a sign-on event has a high-risk risk evaluation.

   - To log activity for the PingOne Risk IdP Adapter as well as its HTTPS and component activity, add the following line.

```
<Logger name="com.pingidentity.adapters.pingone.risk" level="DEBUG"/>
```

   - To log activity for the adapter's HTTPS activity and other components but not the adapter itself, add the following line.

```
<Logger name="com.pingidentity.adapters.pingone.risk.shade"
 level="DEBUG"/>
```

   - To log activity for the PingOne Risk IdP Adapter but not its HTTPS or component activity, add the following lines.

```
<Logger name="com.pingidentity.adapters.pingone.risk" level="DEBUG"/>
<Logger name="com.pingidentity.adapters.pingone.risk.shade"
 level="INFO"/>
```

4. Save the file.

Troubleshooting information

The following information addresses technical situations that you might encounter after setting up the PingOne Risk Integration Kit.

Situation	Information
Include Device Profile is selected in the adapter configuration, but the device profile does not affect the risk result or appear in the response from PingOne Risk.	When this setting is enabled, it is possible for an error to prevent the device profile from reaching PingOne Risk. Because PingOne Risk considers the device profile to be optional, it still successfully returns a risk evaluation to the adapter. The adapter logs a warning in the PingFederate error log about the missing device profile and returns a "success" result to the authentication policy. As a result, the process succeeds but no device profile information is available. To address the problem generating or sending the device profile, review the steps in *Integrating device profiling* on page 8. Make sure you have completed the correct set of steps (authentication page vs. web application) and completed the steps exactly as described.

Release notes

Changelog

The following is the change history for the PingOne Risk Integration Kit.

PingOne Risk Integration Kit 1.3.1 — March 2023

- Updated SDK version.

PingOne Risk Integration Kit 1.3 — September 2022

- Changed the product name to PingOne Risk Integration Kit.
- Added the ability to set a **Policy ID** as a chained attribute and use it to set the risk policy.
- Added Signals SDK Javascript and mechanism to capture device profiling.
- Added support for **Risk Score** as an attribute returned by the adapter.
- Fixed an issue that caused page expired issues when the target application did not respond during within the Risk Timeout.

PingOne Risk Integration Kit 1.2.1 — February 2022

- Fixed an issue that caused an error when users attempted the get risk predictors action.

PingOne Risk Management Integration Kit 1.2 — September 2021

- Added support for including third-party risk predictors in risk evaluations. For more information, see *Using custom risk predictors* on page 21.
- Improved the **PingOne Risk Policy** list by clarifying that the default selection uses the default PingOne Risk policy.
- Improved the clarity of the tooltips in the adapter configuration.
- Improved the device profiling script by excluding low-value attributes, such as `fonts`, `touchSupport`, `webgl` and `audio`. This reduces the size of the device profile cookie by up to 86% without impacting the quality of the risk evaluations.

PingOne Risk Management Integration Kit 1.1 — March 2021

- Added support for pre-populating adapter settings based on the selected PingOne environment. Available in PingFederate 10.2 or later.
- Fixed an issue that caused an error when the adapter received an unexpected attribute type.
- Fixed an issue that skipped device profiling in Internet Explorer 11 when "Captured by this device" was selected.

PingOne Risk Management Integration Kit 1.0 — December 2020

- Initial release.
- Added support for getting risk results and transaction information from PingOne Risk.
- Added the ability to send a device profile and transaction information to PingOne Risk.
- Added the ability to add device profiling into any browser-based authentication source.
- Added support for risk models based on user behavior, IP reputation, geovelocity, and anonymous network detection.

- Added the ability to map any PingOne Risk API response attribute to an attribute in the PingFederate authentication policy.
- Added authentication success/failure reporting to PingOne Risk.
- Added the ability to test the connection to PingOne Risk.
- Added support for the *JavaScript Widget for the PingFederate Authentication API*.
- Added support for the *PingFederate authentication API*.
- Added settings for API connection and request timeouts.
- Added settings to override the PingFederate system-default proxy settings.

Known issues and limitations

The following are known issues or limitations for the PingOne Risk Integration Kit.

Known issues

There are no known issues.

Known limitations

There are no known limitations.

Download manifest

The following files are included in the PingOne Risk Integration Kit `.zip` archive:

- `Legal.pdf` – copyright and license information

- `dist` – contains the integration files
  - `deploy` – contains the Java libraries
    - `pf-pingone-risk-management-adapter-<version>.jar` – JAR file that contains the PingOne Risk IdP Adapter.
  - `conf` – contains the HTML template that presents the PingOne sign-on form.
    - `language-packs` – contains files with customizable user-facing messages
      - `pingone-risk-management-messages.properties` – a variable file that customizes the messages that appear on the template file.
    - `template` – contains user-facing HTML template files
      - `pingone-risk-management-template.html` – a sign-on redirect page used with the "captured by this adapter" device profiling method. Runs scripts to show a spinner animation and create the device profile.
      - `assets` – contains functional scripts and files used by the template
        - `css` – contains CSS files for the templates.
          - `pingone-risk-management.css` – a CSS file that customizes the appearance of the template files.
          - `end-user/<version>/end-user.css` – a CSS file that customizes the appearance of the template files.
        - `fonts/end-user/icons` – contains template icons
        - `images` – contains template image files
          - `ping-logo.svg` – an image file with company branding
          - `spinner.svg` – an image file used in a spinner animation
        - `scripts` – contains script files used to collect and send information
          - `fingerprint2-<version>.min.js` – a JavaScript script that collects information and creates a device profile. Used when device profiling is enabled.
          - `pingone-risk-profiling-signals-sdk.js` - A JavaScript script that wraps the device profiling when using the Signals SDK. Used with both device profiling methods when device profiling is enabled.
          - `pingone-risk-management-profiling.js` – a JavaScript script that wraps the device profiling script. Used with both device profiling methods when device profiling is enabled.
          - `pingone-risk-management-embedded.js` – a JavaScript script that can be embedded in any authentication page. Initiates the device profiling script and stores the device profile in a cookie. Used with the "captured by a previous adapter" device profiling method.
  - `lib/pf-authn-api-sdk-<version>.jar` – a JAR file that contains the PingFederate Authentication API SDK