Protect Foundations - PingFederate Integration Guide

PingOne Protect

Field	Value
Version	1.0
Date	2026-04-01
Owner	Partner Delivery Architects
Intended Audience	Technical Consultants
Distribution	Internal/Partner

Related Delivery Kit Assets

- **Protect Foundations - Getting Started**
- **Protect Foundations - Fundamentals**
- **Protect Foundations - Best Practices**
- **Protect Foundations - DaVinci Integration Guide**
- **Protect Foundations - PingAM / AIC Integration Guide**
- **Protect Foundations - Delivery Roadmap Template**
- **Protect Foundations - Delivery Playbook**

Table of Contents

Protect Foundations - PingFederate Integration Guide

This guide describes how to install and configure the PingOne Protect Integration Kit for PingFederate so PingFederate can call PingOne Protect for risk-based authentication and use the results in authentication policies. It assumes you've completed **Protect Foundations - Getting Started** and **Fundamentals** (Protect enabled, worker app created, initial risk policy in place).

This guide focuses on implementation within PingFederate. For overall delivery flow, follow the **Protect Foundations - Delivery Playbook**.

---

1. Overview & Objectives

Goal
Enable PingFederate to:

- Send transaction and (optionally) device profiling data to PingOne Protect at sign-on time.
- Receive a risk evaluation (score, level, details) and use it to branch the PF authentication policy.

What this guide covers

- Kit components and where they fit in PF.
- Kit deployment to a PF environment.
- High-level configuration of:
    - PingOne (environment, Protect, worker app, risk policies).
    - PF (adapter, provider, device profiling, IP forwarding, policy tree usage).
- A basic validation and troubleshooting approach.

2. Prerequisites

Before you start, you should have:

PingFederate

- Version 11.3 or later (PingOne Protect Integration Kit 1.0+).
- Administrative access to PF console and filesystem.

PingOne

- An organization and environment created in PingOne.
- PingOne Protect added to that environment and at least one risk policy (default is fine initially).
- A worker application with appropriate roles (Environment Admin + Identity Data Admin or equivalent) and its Environment ID, Client ID, Client Secret recorded.

Network

PingFederate must be able to reach PingOne APIs and authentication endpoints for the correct region.

Examples:

- US: https://api.pingone.com, https://auth.pingone.com
- EU: https://api.pingone.eu, https://auth.pingone.eu
- APAC: https://api.pingone.asia, https://auth.pingone.asia

Ensure the region matches the PingOne environment being used. Misalignment will result in failed authentication and risk evaluation calls.

Licensing

- PingOne Protect (or PingOne Risk) license for the target environment.

Ensure the **Protect Foundations - Getting Started** and **Fundamentals** have been completed before beginning this integration.

3. Integration Kit Components

The PingOne Protect Integration Kit adds the following to PingFederate:

PingOne Protect IdP Adapter

- Called in PF's authentication policy.
- Sends **transaction information** (user, IP, application, etc.) to PingOne Protect.
- Receives a **risk result** (level, score, details of current and previous transactions).

PingOne Protect Provider + SDK

- For more detailed guidance on SDKs, see section 6.4.1 for a how-to guide.

Device Profiling Templates & Scripts

- HTML/templates and JS assets used when **device profiling** is enabled.
- Create a **device profile** PF sends alongside transaction data to Protect.

You can deploy the adapter only, the provider only, or both, depending on the pattern you follow (see the official kit docs for pattern diagrams).

In most implementations, the IdP Adapter is required, while the Provider/SDK is strongly recommended for accurate device and behavioural risk detection.

---

4. Deploying the Integration Files

This is a PF filesystem operation.

1. **Download the kit**
   - From PingFederate Downloads → Add-ons → PingOne Protect Integration Kit or the Ping Identity Marketplace.

2. **Stop PingFederate** (all nodes in a cluster).

3. **Copy the JAR**
   - Place `pf-pingone-protect-adapter-<version>.jar` into:
     - `<pf_install>/pingfederate/server/default/deploy`

4. **Copy configuration files**
    - Into `<pf_install>/pingfederate/server/default/conf`, including:
        - Language pack: e.g. `pingone-protect-messages.properties` (to `language-packs`).
        - Template files, JS assets for device profiling and SDK injection (per kit instructions).
    - For newer kit versions, ensure device-profiling scripts are updated with any required flags (e.g. `universalDeviceIdentification: true`).

5. **Start PingFederate** and confirm the console shows:
    - PingOne Protect IdP Adapter type available.
    - PingOne Protect Provider available under appropriate sections.

For clusters, replicate these changes to each node (or use your standard deployment mechanism).

After deployment, verify the adapter and provider are visible in the PF admin console before proceeding to configuration.

---

5. Configuring PingOne

Most of this is already done if you followed **Protect Foundations - Fundamentals**, but verify:

Protect service enabled

- Environment → Overview → Services → PingOne Protect present.

Risk policy ready

- At least one policy (default or custom) configured and active for the target journeys.

Worker app configured

Applications → Applications → Worker app for PF integration with:

- Roles (Env Admin + Identity Data Admin, or equivalent).
- Client ID, Client Secret, Environment ID captured for PF configuration.

Optionally document:

Which policy will be used for:

- Authentication
- Registration
- Recovery
- High-risk transactions

6. Configuring PingFederate

This section focuses on implementation patterns. For conceptual understanding of Protect components, refer to the **Protect Foundations - Fundamentals** guide.

6.1 High-Level Flow

For a simple username/password login with Protect:

1. User hits PF login page (HTML Form Adapter).
2. Protect provider/SDK collects device data (if using device profiling).
3. PF invokes PingOne Protect IdP Adapter:
    - Sends transaction data (+ SDK payload).
    - Receives risk level, score, and details.
    - Ensure that risk evaluations are consistently executed and their outcomes are reflected in authentication policy decisions.
    - Incomplete or inconsistent handling of evaluation results can impact model feedback and overall effectiveness.The PingOne Protect IdP Adapter must execute before any risk-based branching or mitigation logic is applied.
    - All authentication policy decisions (e.g. allow, step-up, deny) must be based on the result returned by this adapter.
4. PF authentication policy branches on risk result:
    - Low: normal path / reduced MFA.
    - Medium: step-up.
    - High: deny / strong step-up.

Where available, use recommendedAction and predictor context in addition to risk level when defining authentication policy branching.

Avoid relying solely on riskLevel; incorporate scenario-specific mitigation (e.g. bot detection, VPN usage).

See the **Protect Foundations - Best Practices** for recommended response patterns.

6.2 Configure Connection to PingOne

Follow PF's standard pattern for connecting to PingOne (via the PingOne Integration Kit or direct OAuth client), but at minimum:

- Ensure PF can obtain an access token using the worker app credentials so the adapter can call the Protect APIs.

6.2.1 PingOne Credentials → PingFederate Configuration (Walkthrough)

This section walks through how PingOne worker application credentials are used within PingFederate to enable PingOne Protect integration.

1. Obtain credentials from PingOne

In PingOne:

- Navigate to Applications → Applications → Worker Application
- Locate the worker app created for PingFederate integration

Record the following values:

- **Environment ID**
- **Client ID**
- **Client Secret**
- **Region** (e.g. auth.pingone.com, .eu, .asia)

These credentials represent the PingFederate "identity" when calling PingOne Protect APIs.

2. Configure credentials in PingFederate

In PingFederate:

- Navigate to **Identity Provider** → **Adapters**
- Create or edit the **PingOne Protect IdP Adapter**

Configure:

- Environment ID → from PingOne
- Client ID → from worker application
- Client Secret → from worker application
- Region → must match PingOne environment

These values allow the adapter to:

- Obtain an access token from PingOne
- Call PingOne Protect APIs for risk evaluation

3. Verify connectivity

After saving the adapter:

- Use any available **t**est connection functionality (if supported by the kit), or
- Trigger a test authentication flow

Confirm:

- PingFederate successfully obtains a token
- No authentication or API errors (401 / 403) occur
- Risk evaluations are created in PingOne

4. Verify components after startup

After deploying the Integration Kit and starting PingFederate confirm the admin console shows:

- **PingOne Protect IdP Adapter** under *Identity Provider → Adapters*
- **PingOne Protect Provider** available for use with HTML Form Adapters

Do not proceed with integration until these components are visible.

Important

(If your customer is already using PF with PingOne MFA or PingOne SSO, you may be reusing that connection.)

Ensure the configured endpoints match the region of the PingOne environment. Using incorrect regional endpoints will result in authentication or API failures.

6.3 Configure the PingOne Protect IdP Adapter

1. In PF admin console, go to Identity Provider → Adapters.

2. Add a new adapter of type PingOne Protect IdP Adapter (kit component).

3. Configure key settings (names may differ slightly by kit version):
    - **PingOne environment details** (Environment ID, region).
    - **Client credentials** (Client ID/Secret of the worker app).
    - **Risk policy ID** (optional – if blank, adapter uses default policy).
    - **Attribute contract** to expose:
        - Risk level (e.g. `riskLevel`).
        - Risk score (`riskScore`).
        - Recommended action (`recommendedAction`).
        - Any extra details you want to use in policy.

At a minimum, ensure riskLevel is exposed and available for policy branching.

4. Map contract attributes appropriately so the PF authentication policy can branch on them.

Save the adapter and test connection (if the kit provides a test button) to verify PF can contact PingOne.

6.4 Configure the Protect Provider (Optional but Recommended)

If you want **device profiling / bot detection** via the provider:

1. Ensure the kit templates/scripts are deployed as per section 4 and official docs.
2. In PF, go to the HTML Form Adapter configuration and:
    - Enable the **PingOne Protect provider** integration.
    - Reference your PF templates that include the provider's JS and HTML for device profiling.

When enabled:

- The quality and timing of Signals SDK or device profiling implementation directly impacts the accuracy of risk evaluations. Poor or incomplete implementation can lead to missing or low-quality signals.
- Ensure Signals SDK or device profiling is implemented early in the user interaction and given sufficient time to collect data before risk evaluation is triggered.
- The provider injects the Signals (Protect) SDK and collects device data during form interaction.
- The SDK payload is passed to the IdP adapter (or evaluated directly if using the provider-only pattern).

For more advanced device profiling options (e.g., `universalDeviceIdentification` and multi-form coverage), use the **Integrating device profiling** section of the official docs.

For most CIAM use cases, enabling the provider/SDK should be considered standard.

Device profiling must begin during user interaction on the login page and complete before the PingOne Protect IdP Adapter is invoked.

6.4.1 Implementing Device Profiling & Signals (Protect) SDK in PingFederate

Use this when device profiling and behavioural data collection are required in PingFederate authentication flows, using the PingOne Protect Integration Kit (Protect Provider + Signals SDK).

1. Confirm compatibility before deployment

- Verify the PingFederate version meets the minimum requirement for your Integration Kit (typically 11.3+ for current versions)
- Download the PingOne Protect Integration Kit for PingFederate that matches your PF version from the PingFederate Add-ons or Marketplace

If device profiling or bot detection is required, confirm the Integration Kit version explicitly supports:

- PingOne Protect Provider
- Signals (Protect) SDK JavaScript assets provided by the Integration Kit

2. Deploy required integration components

- Stop PingFederate (all engine and admin nodes in a cluster)
- Copy the adapter JAR:
    - Place the Integration Kit JAR (for example pf-pingone-protect-adapter-<version>.jar) into:
      <pf_install>/pingfederate/server/default/deploy
- Copy configuration assets into:
  <pf_install>/pingfederate/server/default/conf

Including:

- Language pack (for example pingone-protect-messages.properties) → language-packs
- HTML templates for device-profiling-enabled login forms
- JavaScript assets used by those templates (including Signals SDK files)

Repeat on all nodes, or use your standard PingFederate configuration deployment process

3. Verify adapter and provider availability

- Start PingFederate
- In the admin console, confirm:
    - PingOne Protect IdP Adapter **is available under *Identity Provider → Adapters***
    - PingOne Protect Provider is available for use with HTML Form Adapters

Do not proceed until required components are visible, If not visible:

- Re-check JAR placement
- Verify configuration file paths
- Restart all nodes

4. Enable device profiling in the authentication flow

- Open your HTML Form Adapter (or equivalent login adapter) configuration
- Enable the PingOne Protect Provider for that form
    - This associates the provider with the login templates and injects the Signals (Protect) SDK
- Ensure device-profiling templates and scripts from the Integration Kit are applied to the actual login flow in use (not just sample templates)
- Ensure device profiling scripts execute before primary authentication completes
- Device profiling must begin on the login page before authentication is completed; do not defer SDK execution to a later step in the flow

For bot detection:

- Enable the provider under CAPTCHA / Risk Provider configuration (if applicable)
- Ensure all relevant login forms (identifier-first and HTML) reference the provider

5. Configure adapter and provider integration

Configure the PingOne Protect IdP Adapter with:

- Environment ID
- Region (e.g. auth.pingone.com, .eu, .asia)
- Client ID / Client Secret
- Optional Risk Policy ID (leave blank to use default)

Define the adapter attribute contract to expose at least:

- riskLevel
- riskScore
- recommendedAction

In the IdP Authentication Policy:

- Use the HTML Form Adapter first (with Protect Provider enabled)
- Add the PingOne Protect IdP Adapter after credential capture but before MFA / step-up

Ensure the HTML Form Adapter using the Protect Provider sits in the same authentication policy branch that leads to the Protect IdP Adapter

- This ensures the Signals SDK payload is available when the adapter calls PingOne Protect

6. Validate end-to-end behaviour

Execute a test authentication through the profiled login form

Confirm:

- The PingOne Protect IdP Adapter executes without error
- PingFederate successfully obtains an access token and calls PingOne Protect
- Risk evaluations are created in PingOne Protect
- Device-related predictors are evaluated when profiling is enabled

In PingOne:

- Verify evaluations appear in the Protect dashboard
- Confirm device-related predictors (e.g. new device, bot detection) are evaluated once sufficient traffic is present

In PingFederate:

Create simple policy rules

- LOW → normal path
- HIGH → deny or step-up

Confirm correct branching behaviour across test scenarios

- If device-related predictors consistently show “not evaluated”, re-check SDK execution timing, template usage, and provider configuration

7. Common Pitfalls

- **Incorrect kit / PF version pairing**
  - Verify compatibility before deployment
- **Provider or adapter not visible after startup**
  - Check JAR placement, configuration files, and restart sequence
- **Templates or scripts not applied to active login flow**
  - Ensure the production HTML Form Adapter references the correct templates and provider
- **Device profiling scripts executed too late**
  - Ensure scripts load before authentication completes and allow time for interaction
- **Region or environment mismatch**
  - Ensure region endpoints and Environment ID match the PingOne environment

6.5 Forwarding Client IP

To ensure IP-based predictors (IP reputation, anonymous network, etc.) are accurate, configure PF to forward the true client IP (not just WAF/proxy):

- Use headers like `X-Forwarded-For` or `X-Real-IP` based on your network topology.
- Configure PF’s Runtime Settings to treat the correct header as the client IP.
- Verify in Protect’s events that the IP matches expectations.

6.6 Authentication Policy Wiring

In IdP Authentication Policies:

6.6.1 Insert the login step

- Insert your HTML Form adapter (if used) at the start of the policy.

6.6.2 Add the PingOne Protect IdP Adapter

- Add the PingOne Protect IdP Adapter to the policy flow:
    - After the user has entered identifier (and optionally password)
    - Before MFA or other high-friction steps
- The PingOne Protect IdP Adapter must be executed before any branching logic or step-up decisions.
- Authentication policy rules must use the adapter's output (e.g. riskLevel, recommendedAction) as the basis for all risk-driven decisions.

6.6.3 Define policy rules (how decisions are made)

PingFederate authentication policies evaluate rules based on adapter output attributes.

- Each rule defines:
    - A condition (e.g. riskLevel = HIGH)
    - An outcome (e.g. route to MFA, allow, deny)
- Rules are evaluated top-down, and the first matching condition determines the path taken.

6.6.4 Implement baseline branching

Start with simple risk-level based rules:

- If riskLevel = LOW → proceed to standard path / minimal MFA
- If riskLevel = MEDIUM → route to MFA or step-up authentication
- If riskLevel = HIGH → route to deny or strong mitigation flow

6.6.5 Refine with recommendedAction (optional but recommended)

Where available, use recommendedAction or predictor-specific attributes to refine behaviour:

- BOT_MITIGATION → route to CAPTCHA or bot mitigation flow
- Other actions → apply scenario-specific controls
- Ensure more specific conditions (e.g. recommendedAction) are evaluated before general riskLevel rules where required.

6.6.6 Implementation guidance

- Start with simple riskLevel branching before introducing more complex logic
- Ensure rule ordering reflects desired behaviour (specific → general)
- Confirm all possible outcomes (allow, step-up, deny) are handled in the policy

7. Validation & Testing

7.1 Basic Connectivity

Confirm PF starts successfully after kit deployment, trigger a test login and verify that:

- The PingOne Protect adapter executes.
- No errors appear in PF logs about contacting PingOne.

7.2 Validate Risk Evaluations

1. Perform several test logins (ideally from different IPs/devices).

2. In PingOne:

   Open the Threat Protection / Protect dashboard, confirm you see risk evaluations for the environment and that:

- Risk levels and scores are populated.
- Predictor details appear as expected.

3. If you can, simulate risky scenarios (e.g., Tor/VPN IPs, new devices) to observe High or Medium results.

7.3 Validate Policy Branching

- Create simple PF policy rules (e.g., branch on `riskLevel = HIGH`).
- Use known scenarios to drive:
  - LOW → standard login path.
  - HIGH → deny or forced MFA.
- Confirm user experience matches the configured branches.

Confirm that evaluation results are consistently returned and used to drive policy decisions across all scenarios.

If risk-based branching behaves unexpectedly, verify adapter execution order, attribute mapping, and policy wiring before investigating risk policies.

8. Troubleshooting

If issues arise:

No risk events in Protect
Check:

- PF logs for adapter/provider errors.
- Worker app credentials (Client ID/Secret, Environment ID).
- Network egress to `api/auth.pingone.*` endpoints.

Adapter errors
Enable debug logging in PF and the integration kit's logging categories (per official kit docs).

Device profiling not working
Confirm:

- Scripts and templates from the kit are deployed and referenced by the HTML Form adapter.
- Browser developer tools show the SDK JS loading without errors.
- The Integrating device profiling steps have been followed.

Predictors showing "not evaluated"
Verify:

- Required inputs (IP, SDK payload, custom attributes) are present in evaluations.
- Integration steps for those predictors (Headers, SDK, third-party feeds) are complete.

For more in-depth scenarios and test cases, refer to **Protect Foundations – Delivery Playbook** for PS patterns.