Implementation Guide

Make your MFA adaptive with actions templates

Background

Adaptive multi-factor authentication (MFA) reduces friction for legitimate users by assessing transaction risk with machine learning (ML) algorithms, so that known users in their usual stomping grounds are fast tracked onto your platform.

But, it takes time to build a risk engine from scratch, and getting MFA right can make the difference between building consumer trust, and a user abandoning your platform because there were too many steps to login.

To power Adaptive MFA, Okta CIC has ML confidence scoring available out-of-the-box to suit your risk assessment needs, in order to improve UX and security for all users who want to access your platform.

You can use this ML calculation with Actions, and create your own Adaptive MFA program that resolves blind spots that standalone MFA may miss, such as:

- How do you keep legitimate users' sessions uninterrupted but block unwanted traffic?
- When is it appropriate to present a second or third factor?
- What is considered foundational for keeping your platform safe with MFA?

In this post we are going to cover how to use Actions, and what Actions templates are available out of the box in order to hit the ground running when it comes to MFA implementation best practices.

As part of our extensibility framework, Actions are a drag-and-drop pro-code/no-code logic that you can customize for your own applications and integrations that start with Identity.

Actions lets you add code to vital points in the authentication pipeline with just javascript — and 2M+ npm modules at your disposal.

Actions templates teach you how to harness the power of Actions, and get to market faster than the competition, addressing common use cases that are vital for organizations today.

Template #1: Require MFA enrollment

Enrollment is a unique opportunity to give users a choice when it comes to authentication.

Based on a user's authentication preference, you decrease friction for them, and get them on board with your security posture.

Let's get started with the **Require MFA Enrollment** Action template.

Navigate to **Actions** > **Library** > **Build from Template**.

Here is the body of the template:

```
exports.onExecutePostLogin = async (event, api) => {
if (!event.user.multifactor?.length) {
api.multifactor.enable('any', { allowRememberBrowser: false });
}
};
```

What's really happening here: If there aren't any MFA factors enrolled, allow your user to enroll in any that you make available.

A template is just the beginning — Let's take a look at the event and api objects:

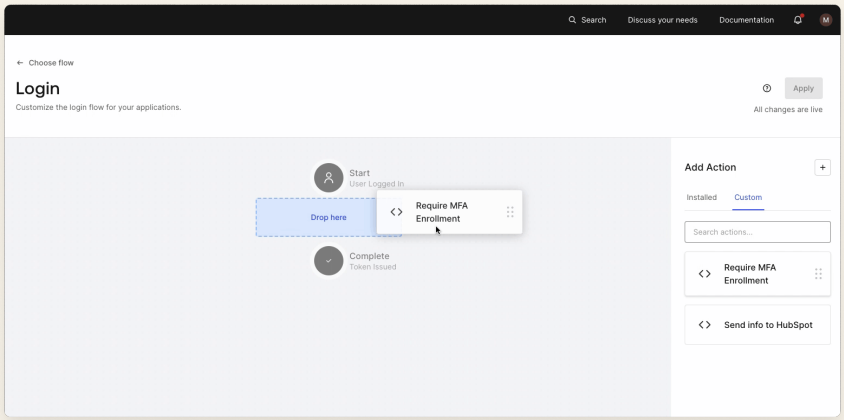
The event object has many different parameters, which includes data about the user, that you can use to customize your MFA requirements; in this case, we are polling the array of available MFA factors, `event.user.multifactor?.length`, and if there are none (!) enrolled, continue with enrollment.

Consider requiring or specifying different providers via the API object — factors include: `duo, google-authenticator,guardian`.

```
api.multifactor.enable(provider, options)
```

Options like `allowRememberBrowser` determines if the browser should be remembered, so that users can skip MFA later. This is an optional boolean, and the default is false. You can modify this option via the management API.

By deploying, then dragging and dropping your new action into the login flow (**Actions** > **Flows** > **Login**) and selecting **Apply**, your users now required to enroll in MFA:



Repeat the step above whenever you'd like to add an Action to a trigger in the authentication pipeline.

Getting adaptive with your MFA

Navigate to **Security** > **Multi-factor Authentication**, and select the factors you'd like to be available to your end users.

Scroll down to **Additional Options**, and toggle the option to **Customize MFA Factors using Actions**. This allows you to add your own Actions logic with our out-of-the-box Adaptive MFA ML intelligence.

Here are some primary pieces of information to consider about a user's transaction when coding to match your security playbooks:

- What conditions do I need my user to reauthenticate?
- How does their session information matter when it comes to conducting a given transaction?
- What corporate policy restrictions translate into application policies?

With these considerations in mind, let's walk through, step-by-step, how to implement Adaptive MFA with Actions templates.

Template #2: Trigger MFA when condition is met

This template makes use of our Adaptive MFA risk/confidence scoring — based on risk assessment, you can potentially keep bad actors out, but also build a security rapport with your customers to self-serve with a factor in the event that new or anomalous behavior is detected.

In this template, newDevice is the assessed condition for additional MFA prompts; you have the following risk assessment objects available to poll a confidence score:

- NewDevice
- ImpossibleTravel
- UntrustedIP
- PhoneNumber

You can even combine assessments to make a determination about the Action's outcome; for example, if impossible travel occurs, you can block the user's transaction altogether.

```
exports.onExecutePostLogin = async (event, api) => {
// Decide which confidence scores should trigger MFA, for more
information refer to
// https://auth0.com/docs/secure/multi-factor-authentication/adaptive-
mfa/customize-adaptive-mfa#confidence-scores
const promptConfidences = ['low', 'medium'];

// Example condition: prompt MFA only based on the NewDevice
// confidence level, this will prompt for MFA when a user is logging
in
// from an unknown device.
const confidence =
event.authentication?.riskAssessment?.assessments?.NewDevice
?.confidence;
const shouldPromptMfa =
confidence && promptConfidences.includes(confidence);

// It only makes sense to prompt for MFA when the user has at least
one
// enrolled MFA factor.
const canPromptMfa =
event.user.multifactor && event.user.multifactor.length > 0;
if (shouldPromptMfa && canPromptMfa) {
api.multifactor.enable('any', { allowRememberBrowser: true });
}
};
```

Template #3: Trigger MFA when the requesting IP is from outside a specific IP range

This template restricts access to a given application to say, a corporate network, and uses the ipaddr.js library to parse IPs, and, in this case, trigger a push notification via Guardian:

```
exports.onExecutePostLogin = async (event, api) => {
const ipaddr = require('ipaddr.js');

// get the trusted CIDR and ensure it is valid
const corp_network = event.secrets.TRUSTED_CIDR;
if (!corp_network) {
return api.access.deny('Invalid configuration');
}

// parse the request IP from and ensure it is valid
let current_ip;
try {
current_ip = ipaddr.parse(event.request.ip);
} catch (error) {
return api.access.deny('Invalid request');
}

// parse the CIDR and ensure validity
let cidr;
try {
cidr = ipaddr.parseCIDR(corp_network);
} catch (error) {
return api.access.deny('Invalid configuration');
}

// enforce guardian MFA if the IP is not in the trusted allocation
if (!current_ip.match(cidr)) {
api.multifactor.enable('guardian', { allowRememberBrowser: false });
}
};
```

Template #4: Require MFA once per session

This template does something a little different from the others.

Instead of keeping users out, this configuration helps you achieve silent authentication, which supports a user to go about their session from their usual browser stomping grounds without having to be prompted for MFA.

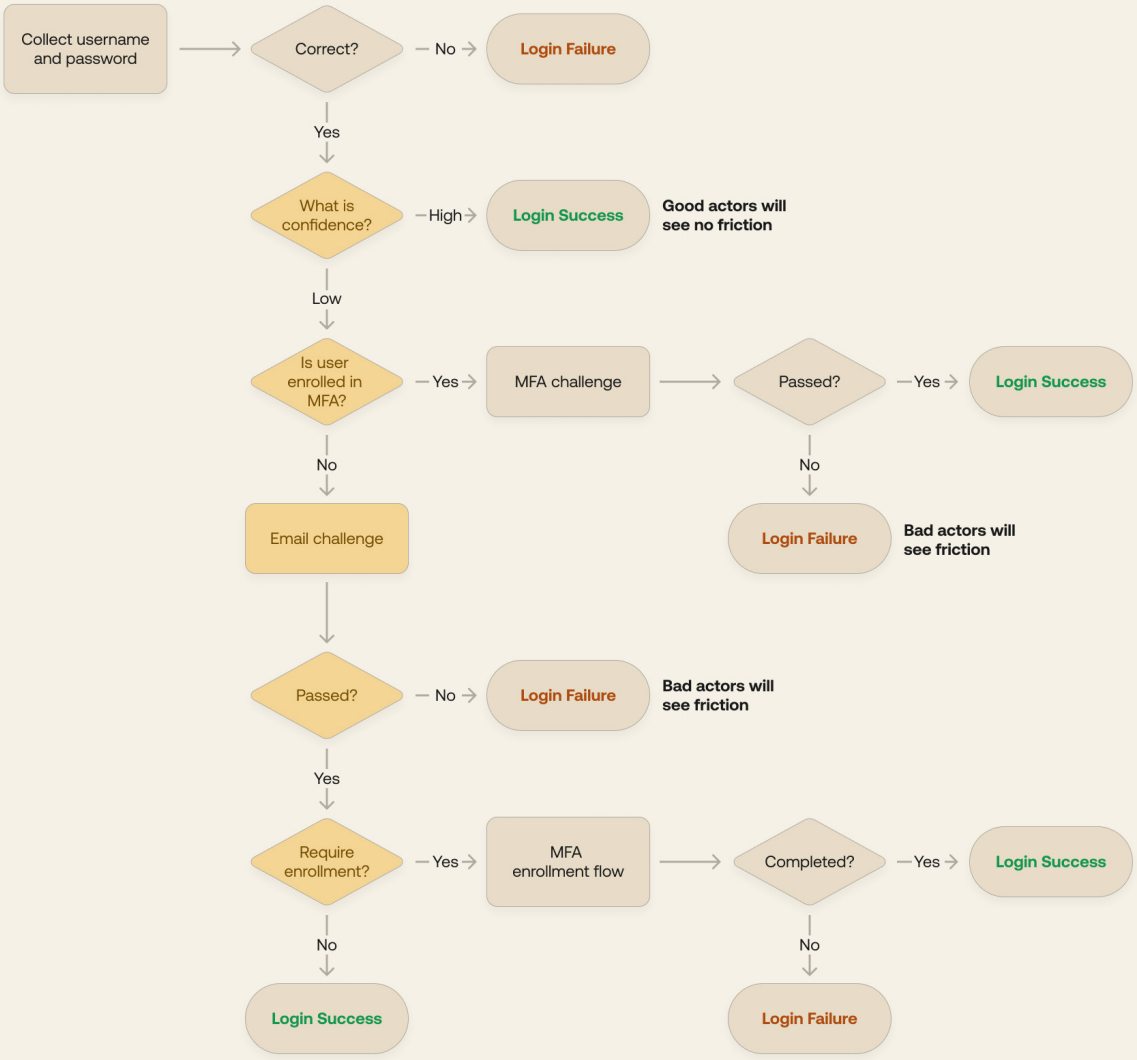
```
exports.onExecutePostLogin = async (event, api) => {
// if the array of authentication methods is valid and contains a
method named 'mfa', mfa has been done in this session already
if (
!event.authentication ||
!Array.isArray(event.authentication.methods) ||
!event.authentication.methods.find((method) => method.name === 'mfa')
) {
api.multifactor.enable('any');
}
};
```

Summary

Our templates covered how to enforce MFA on registration, outside a corporate network, per session, and the beginnings of an adaptive MFA implementation.

All of these templates power how our Universal Login functions in different authentication contexts, which means you can leave the UX to us.

With Actions, you can create an entire security flow to match your organization's security use cases, and also eliminate friction for legitimate users that are high on the confidence scale.



These templates are just the beginning.

Check out our other implementation guides:

- Flag sensitive transactions for step-up
- Collect FPD post-reg and post-login
- Prompt with personalized recommendations
- Customize UI for accessibility