Red Hat 3scale API Management 2.4

Quickstart

This guide will help you get up and running to boost your API with 3scale in no time at all.

Last Updated: 2019-06-03

Red Hat 3scale API Management 2.4 Quickstart

This guide will help you get up and running to boost your API with 3scale in no time at all.

Legal Notice

Abstract

This guide provides the basic information to help users get started with Red Hat 3scale API Management 2.4.

Table of Contents

CHAPTER 1. LAUNCHING THE API PROGRAM

This guide helps you to get started with boosting the API with 3scale.

It will cover the following key steps to launch the API:

1. **Secure** the API.
2. **Configure** the API access policies with application plans.
3. **Engage** your developers with a Developer Portal.
4. **Go Live**.

You can choose from the following three paths to launch the API:

- Prototype
  Completion time: Less than an hour.

  Goal*: Complete an end-to-end integration of 3scale with a simple public API.

  Recommended for: The prototype path is recommended to get the fastest possible overview of how to integrate 3scale and to get an appreciation of 3scale's end-to-end capabilities. You must do this before going through the basic path. If you have successfully completed the onboarding wizard in the Admin Portal, you can skip this path and go to the next one.

- Basic
  Completion time: Less than one week.

  Goal: Complete all implementation steps to launch your API in production.

  Recommended for: If you want to go live with your API in production and you have limited time, the basic path will cover most of your needs.

- Advanced
  Completion time: Several weeks.

  Goal: Optional extras after you have completed the basic path. Advanced control of your API. Deeper customization of the Developer Portal.

  Recommended for: If you have a more complex requirement or if you have covered the basic path already, you may be ready to consider advanced options. If you want to use a custom domain and email (in SaaS), they have a long lead time, so you should follow the steps in the go live section to get them set up quickly.

The timing guidelines depend on the complexity of your API and the resources you plan to dedicate to the effort. You will spend most of your time on refining your API and preparing content for your developer portal. If you already have a stable API and content for documentation, you can go live within a week.

1.1. HOW TO SECURE, CONTROL, AND PROMOTE YOUR APIS

You can follow through the **prototype**, **basic**, and **advanced** paths individually from end to end, or you can also choose to perform some steps from the three different paths according to your needs. Each path is independent, but they build on top of each other.

1.1.1. Prototype

1.1.1.1. Secure your API

Assuming your API is publicly accessible (for Saas) or reachable from your 3scale AMP installation (for on-premises), you can prototype the 3scale access control layer within a few minutes.

The Echo API will serve as en example of a public API. It is a simple API that accepts any path and returns information about the request (path, request parameters, headers, etc.) in the response body. It is accessible at the following URL: https://echo-api.3scale.net

1. Verify that your API is reachable (after the security layer is in place, you can hide or restrict access to the backend host). Example: https://echo-api.3scale.net/v1/fast/track.
2. Navigate to **[Your_API_name] > Integration > Configuration**
3. Before you set up your own API, verify that you can make a test call using the default parameters.
4. After verification, enter the private base URL; example: **https://echo-api.3scale.net:443**.
5. Enter the URL path for a valid GET request to make a test call; example: **/v1/fast/track**, and then click **Update & Test Staging Environment**
    1. Copy the cURL statement, which includes the **user_key** as the default credential to make calls from the command line:

    ```
    curl "https://api-2445581407825.staging.apicast.io:443/v1/fast/track?
    user_key=287d64924e6120d215b1000ac07c063b"
    ```

    You can make different calls. For example try another endpoint, adding the same **user_key**.

    NOTE

    You can get the API keys from the application details page of one of the developer accounts.

    Your 3scale access control layer will now only allow authenticated calls through to your backend API.

1.1.1.2. Configure your API access policies with application plans

In the preceding steps, you ensured that only authenticated calls are allowed through to your API. In this section you will apply policies to differentiate the rate limits.

In 3scale, *applications* define the credentials to access your API. An application is always associated with one *application plan* that determines the access policies. Applications are stored within *developer accounts*. In the basic 3scale plans only a single application is allowed; but, in the higher plans, multiple applications per account are allowed.

In this example, you add a policy to the Echo API used in the preceding section.

1. Navigate to **[Your_API_name] > Applications > Application Plans**

2. In the 'Application plans' section, go to the *basic* application plan to edit one of the plans that was generated by the sample data after installing or signing up for 3scale.

3. Select *limits* in the *hits* row, and create a new usage limit of 3 per hour.

4. Find one of your sample applications, by navigating to **[Your_API_name] > Applications > Listing**. Ensure that the application is set to the *basic* plan. If not, *change plan* on the application details page.

5. Use the credentials for this application and repeat the previous sample call at least 3 times.

You have now successfully defined more restrictive access polices for all the applications on the basic plan.

1.1.1.3. Engage your developers with a developer portal

For the prototype, you do not need to create any documentation content. It is usually enough to check that the workflows will meet your requirements. For example, while the API is in development and testing, you may want to disable the full self-service workflow:

1. From your Admin Portal, navigate to **Audience** space and click **Visit Portal** link in the **Developer Portal** menu.

2. Create a test signup and walk through all the steps.

3. Usually self-service is enabled by default. To change it, go to **Audience > Accounts > Usage Rules** and click the *account approval required* checkbox.

4. Repeat the test signup walkthrough and verify that you need to approve the account in the Admin Portal before the user can log in.

You can now successfully customize workflows for your developer portal.

1.1.2. Basic

1.1.2.1. Secure your API

For a full production implementation, you need to make some fundamental decisions about how to structure your API and implement integration with 3scale.

You have the choice of several authentication modes for API traffic. Consult the guide on the available options and configure the settings.

IMPORTANT

After you set it, you should not switch auth modes again because it can easily invalidate existing credentials.

You also have the choice of several deployment options for the API traffic manager layer . APIcast, the NGINX based API gateway, is the favorite amongst 3scale customers due to its combination of ease of configuration and performance. You can use APIcast hosted which is great to get started quickly, but comes with volume limits and additional latency. Or, you can deploy it on your own servers for the best performance and completely unrestricted traffic volume.

APIcast hosted

1. Follow the onboarding wizard after you log in to your Admin Portal for the first time.
2. Continue iterating on your API configuration (such as refining access policies) until you have reached a version you are happy with for production.
3. Promote your APIcast configuration to the production gateway.

APIcast self-managed

1. Follow the guide for APIcast for more details than in the *prototype* steps. This guide also covers some ground to configure the API access policies.
2. Set up a test installation of your API gateway on your OpenShift servers.
3. Continue iterating on your API configuration (such as refining access policies) until you have reached a version that you are happy with for production.
4. Promote your APIcast configuration to the production gateway.

1.1.2.1.1. Configure your API access policies with application plans

In the preceding section, you ensured that only authenticated calls are allowed through to your API. In this section you will apply policies to differentiate rate limits.

In 3scale, *applications* define the credentials to access your API. An application is always associated with one *application plan* that determines the access policies. Applications are stored within *developer accounts*. In the basic 3scale plans, only a single application is allowed. In the higher plans, multiple applications per account are allowed.

In *prototype*, you can only control access based on overall hits on your API. The flexibilty of 3scale is realized after you start using custom methods and metrics to create more sophisticated tiers for your application plans and for deeper analytic insight to your API. For a brief background, see the analytics guide.

IMPORTANT

- The mapping between your API structure and methods or metrics in 3scale is logical. You can report the usage to 3scale, if you can define a consistent rule. You must determine the level of detail. Generally, it is good to aim for 5-20 methods/metrics.
- The values reported to 3scale can only be incremented. You cannot set absolute values or decrement the counters.
- After adding any new methods or metrics to 3scale, it is important to add the new system names to your integration point (API gateway or code plugin).
- You can make changes, such as rate limits, at runtime without redeploying.

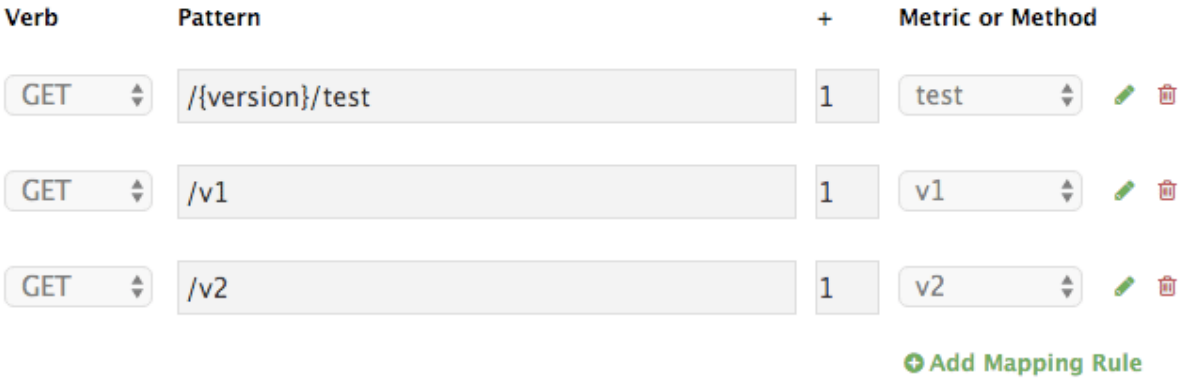
In this example, to add polices to the application plan of the Echo API, take the following steps:

1. Find the API you want to work on.
2. In the 'Application Plans' section, select *basic* to edit one of the plans that was generated automatically after signing up to 3scale/deploying your instance.

3. If you have a rate limit for *hits*, remove it.
4. Add a *new method* to the plan under the *hits* metric with the system name "test".
5. Set a rate limit for the test method to 5 per hour.
6. Add two *new metrics* with system names "v1" and "v2".
7. Under the v2 metric, disable access by clicking on the *enabled* column. This has the same effect as setting a rate limit of zero.

APIcast deployment

1. Go to **[Your_API_name] > Integration > Configuration**
2. Expand the mapping rules section and add the following mappings:



NOTE

the default mapping for "/" has been removed. If still used, it will lead to double-counting of hits.

Code plugin deployment

1. Follow the instructions and examples in your plugin library to add usage for custom methods and metrics to your 3scale authorization and reporting calls.
2. Ensure a mapping from the URL structure to the custom method, "test".
3. Ensure a mapping from the URL to the custom metrics "v1" and "v2".
4. Test the calls using application credentials associated with the basic plan.
   - Calls will be allowed:

     ```
     curl "https://api-2445581407825.staging.apicast.io:443/v1/test?
     user_key=287d64924e6120d215b1000ac07c063b"
     ```

     After 5 calls, the calls will start to get rejected. This is because of the limit set for the test method.

   - Calls will be rejected because v2 is not allowed in the *basic* plan:

     ```
     curl "https://api-2445581407825.staging.apicast.io:443/v2/test?
     user_key=287d64924e6120d215b1000ac07c063b"
     ```

- Calls will be rejected because there is no mapping rule set for missing:

```
curl "https://api-2445581407825.staging.apicast.io:443/missing?
user_key=287d64924e6120d215b1000ac07c063b"
```

- These calls will be allowed for NGINX (depending on how you have implemented the mapping for your plugin). For the following call, it will be up to your application to return a 404 not found response. To avoid this, refine the mapping:

```
curl "https://api-2445581407825.staging.apicast.io:443/noversion/test?
user_key=287d64924e6120d215b1000ac07c063b"
```

This basic concept gives you all the flexibility you need to define your API tiers. It is important to decide early on what you want to use for your custom methods and metrics. Whenever you make changes to the system names, you must redeploy the changes as described in the *secure your API* section.

1.1.2.2. Engage your developers with a developer portal

The developer portal guide contains information to complete a developer portal. Consider writing your content in Textile or Markdown. Following are optional steps that you may want to consider:

- Configure ActiveDocs to bring interactive capabilities to your documentation and make it easier for developers to explore.
- Add a favicon.
- Add your Google Analytics tracker code by editing *partial* in your CMS called *analytics*.
- Configure your signup workflows.
- Customize your email addresses (doc for SaaS) and the email template content.

1.1.3. Advanced

1.1.3.1. Secure your API

Advanced authentication mode: OpenId Connect

Secure your APIs using the APIcast integration with OpenID Connect for Red Hat Single Sign-On (RH-SSO). Applications in the Red Hat 3scale API Management Platform are synchronized with the Identity Provider (IdP), in this case RH-SSO. Currently, this is an end-to-end supported solution. It covers the main OAuth 2.0 flows: Authorization code, Resource password owner, Client credentials, and Implicit grant.

Code plugin deployment

Almost all 3scale customers find performance to be fine. But, if you want to turbo-charge your API, you can cache authorization calls to 3scale using any caching library that you are comfortable with.

1.1.3.2. Configure your API access policies with application plans

In the preceding section, you ensured that only authenticated calls are allowed through to your API. In this section, you apply policies to differentiate rate limits.

In 3scale, *applications* define the credentials to access your API. An application is always associated with one *application plan* that determines the access policies. Applications are stored within *developer accounts*. In the basic 3scale plans, only a single application is allowed. In the higher plans, multiple applications per account are allowed.

Alerts may be configured to send notifications by email or to the web consoles: . Go to your API Settings page: **[Your_API_name] > Integration > Settings** . Go to the Alerts section on the page. Here, you can configure the alerts that you want as a percentage of your rate limit levels.

3scale gives you the flexibility to decide whether to make rate limits soft (even calls above the limits are allowed through) or hard (calls are rejected before hitting your application). With the code plugin, you consciously need to decide which type to implement. On the other hand, APIcast defines hard limits by default. These can be customized in the Lua file to avoid rejecting over-limit calls.

1.1.3.3. Engage your developers with a developer portal

After you have completed the basic path, follwoing are the two advanced areas to explore for the developer portal:

- Liquid markup provides tags and drops that provide direct access to system objects and allow you to introduce dynamic rendering of developer portal pages.
- All 3scale system pages can be customized. This is for advanced users because the HTML is complex. Ultimately, you can customize virtually any page of your developer portal. Usually the default pages will be perfectly fine with some CSS changes.

1.2. GO LIVE

Following is the final checklist before the public launch of your API.

NOTE

Raise request for the custom domain and email as soon as possible because they have a long lead time.

1. Set up a custom domain* (SaaS documentation).
2. Optionally, set up a custom outbound email address* (SaaS documentation).
3. Remove the developer portal access code from **Audience > Developer Portal > Domains & Access**.

Following are some extra points for consideration:

- Add pricing to generate revenue directly from your API (only available for SaaS accounts).
- Use insight from your API analytics (under *analytics* in your Admin Portal) to refine your application plans.