Python 3 (/appengine/docs/standard/python/migrate-to-python3).

Migrating Traffic

Python **2.7**/3 (/appengine/docs/standard/python3/migrating-traffic) | Java 8 (/appengine/docs/standard/java/migrating-traffic)/11 (/appengine/docs/standard/java11/ migrating-traffic) | PHP 5 (/appengine/docs/standard/php/migrating-traffic)/7 (/appengine/docs/standard/php7/ migrating-traffic) | Ruby (/appengine/docs/standard/ruby/ migrating-traffic) | Go 1.11 (/appengine/docs/standard/go111/ migrating-traffic) / 1.12+ (/appengine/docs/standard/go/ migrating-traffic) | Node.js (/appengine/docs/standard/nodejs/ migrating-traffic)

Traffic migration switches the request routing between the versions within a service of your application, moving traffic from one or more versions to a single new version.

For information about splitting traffic between two or more versions of your app, see Traffic Splitting (/appengine/docs/standard/python/splitting-traffic).

Before you begin

Before you can configure traffic to a version, ensure that your user account includes the required privileges (/appengine/docs/standard/python/access-control#primitive_roles).

Migrating traffic gradually

In the standard environment, you can choose to route requests to the target version, either immediately or gradually.

By default, warmup requests (/appengine/docs/standard/python/warmup-requests) are disabled and traffic is migrated immediately to a version.

You can also choose to enable warmup requests if you want the traffic gradually migrated to a version. If you immediately migrate traffic to a new version without any running instances then

you will experience a spike in latency for loading requests. Deploying a new version with the same name as an existing version causes an immediate traffic migration. All instances of the old version are immediately shut down. There will be a latency spike due to loading requests for the new version.

Gradual traffic migration traffic between versions running in the flexible environment is not supported. You mu
e traffic immediately to versions that are running in the flexible environment.

If warmup requests are enabled, you can migrate traffic between versions that reside in different environments only by specifying to migrate traffic immediately.

Adding warmup requests to your application

When warmup requests are enabled, traffic is migrated gradually by first sending a *warmup request* to new instances before those instances receive any user requests. Warmup requests improve user response time by allowing the version currently receiving traffic to handle those requests but the traffic migration to the new version can take a short amount of time while the new instances are created.

When warmup requests are not enabled, user requests are sent to those new instances before they have been created. Due to the delay caused by creating the new instances and loading application code, latency can occur for those user responses.

To avoid latency and enable warmup requests, include the `inbound_services` element in your configuration file before you deploy your application to App Engine.

For example, you include the following in your `app.yaml` file before deploying it to App Engine:

```
nd_services:
mup
```

For complete details about enabling warmup requests, see Configuring Warmup Requests to Improve Performance (/appengine/docs/standard/python/warmup-requests/configuring).

Migrating traffic to a new version

Consolegcloud (#gcloud)API (#api)

To migrate traffic in the Cloud Console, go to the Versions page:

Go to the Versions page (https://console.cloud.google.com/appengine/versions)

1. Select the version to which you want to migrate 100% of the traffic.
2. Click **Migrate traffic**.
3. Optional: When warmup requests are enabled your traffic is migrated gradually. To migrate traffic immediately, select the option under the **Show advanced options** section.

Last updated 2020-07-14 UTC.