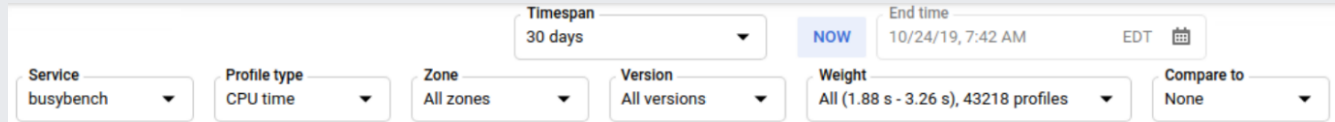


The menus or buttons displayed in the following screenshot determine the set of profiles that you can analyze using the Profiler interface:



The screenshot shows a horizontal toolbar with several filter controls. From left to right: a 'Service' dropdown menu set to 'busybench'; a 'Profile type' dropdown menu set to 'CPU time'; a 'Zone' dropdown menu set to 'All zones'; a 'Version' dropdown menu set to 'All versions'; a 'Timespan' dropdown menu set to '30 days'; a 'NOW' button with a blue background; an 'End time' field set to '10/24/19, 7:42 AM' with an 'EDT' label and a calendar icon; a 'Weight' dropdown menu set to 'All (1.88 s - 3.26 s), 43218 profiles'; and a 'Compare to' dropdown menu set to 'None'.

Each time you modify one of those menus or buttons, Stackdriver Profiler performs the following actions:

1. It identifies the set of profiles that meet the settings.
2. If more than 250 profiles are available, it randomly selects 250. Otherwise, all available profiles are selected.
3. It creates a single profile and then draws the flame graph.

Because changing a menu or button setting might change the profiles selected by Profiler, your flame graph might change.

You use the **Timespan** menu, **Now** button, and **End time** menu to control the range of time for which profiling data is displayed.

By default, the time fields have the following settings:

- **Timespan** is set to 7 days.
- **Now** button is shaded by a blue background.
- **End time** contains the time when Profiler was started and cannot be modified.

With these settings, Profiler analyzes profiles captured in the previous 7 days.

To set the timespan, click the **Timespan** down arrow, and then select an option from the list. Your choices range from 10 minutes to 30 days, the limit of the retention period for profile data.

To update the end time to the current time, click **Now**. The background of this button toggles between blue and white. In either case, a single click updates the end time field to the current time.

To set the end time, do the following:

- If the **End time** text isn't changeable, as shown in the following image, click **Now**:

The screenshot shows a control panel with a 'Timespan' dropdown set to '7 days', a 'NOW' button, and an 'End time' field. The 'End time' field contains '2/4/19, 5:40 PM' and 'EST' with a calendar icon. The 'End time' text is greyed out, indicating it is not currently editable.

- In the **End time** field, enter a date and time, or use the calendar option to select a date:

The screenshot shows the same control panel as above, but the 'End time' field is now active, highlighted with a blue border. The 'End time' field contains '1/31/19, 12:00 PM' and 'EST' with a calendar icon. The 'NOW' button is now white, indicating it is no longer the active state.

Use the **Service** menu to select the service whose data you want to analyze. The service name is one of the values you (or the runtime environment) specify when you run an application with profiling enabled. For more information on service names, see the profiling guides for [Go](/profiler/docs/profiling-go#svc-name-and-version) (/profiler/docs/profiling-go#svc-name-and-version), [Java](/profiler/docs/profiling-java#service_name_and_version_arguments) (/profiler/docs/profiling-java#service_name_and_version_arguments), [Node.js](/profiler/docs/profiling-nodejs#svc-name-and-version) (/profiler/docs/profiling-nodejs#svc-name-and-version), or [Python](/profiler/docs/profiling-python#svc-name-and-version) (/profiler/docs/profiling-python#svc-name-and-version).

Use the **Profile type** menu to select the type of profiling data to analyze:

Profile type	Go	Java	Node.js	Python
CPU time	Y	Y		Y
Heap	Y	Y	Y	

Profile type	Go	Java	Node.js	Python
Allocated heap	Y			
Contention	Y			
Threads	Y			
Wall time		Y	Y	Y

Each profile type captures a different kind of information:

- **CPU time:** information about CPU usage.
- **Heap:** information about the memory allocated in the program's heap when the profile was collected.
- **Allocated Heap:** information about the total memory that was allocated in the program's heap, including memory that is freed and no longer in use.
- **Contention:** information about mutex usage.
- **Threads:** information about thread usage.
- **Wall time:** information about total time to run.

To restrict the analysis to instances of the service running in a specific [Compute Engine zone](/compute/docs/regions-zones) (/compute/docs/regions-zones), click the **Zone** down arrow, and then select the zone from the list.

The default setting for this field is **All zones**.

To restrict the analysis to a specific version of the named service, click the **Version** down arrow, and then select the version of interest. The service version is an optional value you (or the runtime environment) specify when an application is run with profiling enabled. For more information on service versions, see the profiling guides for [Go](#)

(/profiler/docs/profiling-go#svc-name-and-version), Java
(/profiler/docs/profiling-java#service_name_and_version_arguments), Node.js
(/profiler/docs/profiling-nodejs#svc-name-and-version), or

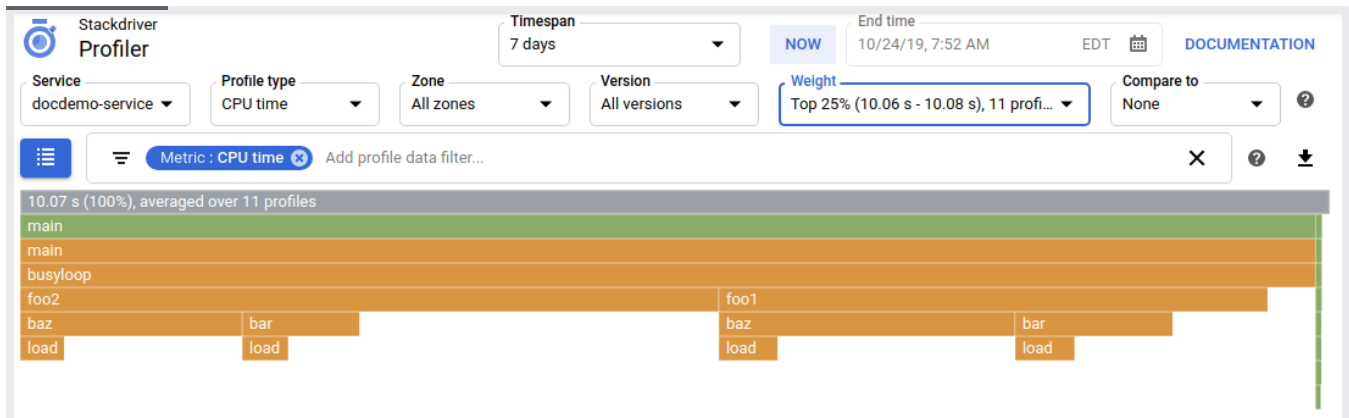
The default setting for this field is **All versions**.

To analyze only the profiles captured when the metric consumption was within a selected percentage of peak consumption, click the **Weight** down arrow, and make a selection from the menu. An example **Weight** menu is as follows:

Most rows in the **Weight** menu start with **Top** followed by a percentage. For example, **Top 5%** indicates that only profiles that were collected during the top 5% of metric consumption are available for analysis. The two values in the parentheses list the corresponding range of metric consumption. The last value is the number of profiles collected over this range. For the **Top 5%** row, 1 profile was collected.

The first row is the default setting for the **Weight** field. The word **All** indicates that all collected profiles, or equivalently 100% of the collected profiles, are available for analysis.

The following screenshot shows a weight-filtered graph:



The default setting for this field is all profiles.

To visually compare two profiles of the same type, taken from the same service with a project, that differ by a single user-defined attribute, use the **Compare To** feature. For example, if you deploy two versions of the same service, you can compare the profiles for these two versions.

For more information, see [Comparing profiles](/profiler/docs/comparing-profiles/) (/profiler/docs/comparing-profiles).

The default setting for this field is **None**.

- For information on using your pointer to change the display of the flame graph, see [Interacting with the flame graph](/profiler/docs/interacting-flame-graph/) (/profiler/docs/interacting-flame-graph).
- For information on how to suppress, or highlight, frames in the flame graph, see [Using filters](/profiler/docs/filtering-profiles/) (/profiler/docs/filtering-profiles).
- For information on focusing the graph on a single function, see [Focusing the graph](/profiler/docs/focusing-profiles/) (/profiler/docs/focusing-profiles).
- For information on comparing profiles collected by different deployments of your service, see [Comparing profiles](/profiler/docs/comparing-profiles/) (/profiler/docs/comparing-profiles).
- To download your profile data, see [Downloading profiles](/profiler/docs/downloading-profiles/) (/profiler/docs/downloading-profiles)

- For information on using the Profiler agent to collect profiling data for your services, see:
 - [Profiling Go applications \(/profiler/docs/profiling-go\)](/profiler/docs/profiling-go)
 - [Profiling Java applications \(/profiler/docs/profiling-java\)](/profiler/docs/profiling-java)
 - [Profiling Node.js applications \(/profiler/docs/profiling-nodejs\)](/profiler/docs/profiling-nodejs)
 - [Profiling Python applications \(/profiler/docs/profiling-python\)](/profiler/docs/profiling-python)
 - [Profiling applications running outside Google Cloud \(/profiler/docs/profiling-external\)](/profiler/docs/profiling-external)