

Filters

To create any filter via the API, you must create or update a record for your domain with the answers you plan to use. You must also define any metadata that would be required for the filters you want to create. Finally, you specify the resource record to which to apply these changes. You create via a `PUT` request, and you update via a `POST` request.

Example

Suppose that you want to create an A record with two answers on the `pulsar.example.com` domain that uses the Pulsar Performance Sort filter. In the first step of this process, we often recommend that you start by defining backup logic, such as basic geographical steering, in order to have backup logic if Pulsar does not have enough data to make a decision. To do this, you would:

1. Define the `zone`, `domain`, and the record `type`.
2. Define at least one `answer` within the record.
3. Add geographic metadata using the `meta` object for the answer to define the backup filter.

So far, your command would read as follows:

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d
'{"zone":"example.com","domain":"pulsar.example.com","type":"A",
"answers":[{"answer":["1.1.1.1"],"meta":{"georegion":["US-EAST"]}},
{"answer":["9.9.9.9"],"meta":{"georegion":["US-WEST"]}},{"filters":[{"filter":"geotarget_re
gional"}]
https://api.nsonet.net/v1/zones/example.com/pulsar.example.com/A
```

From here, you would add definitions for the Pulsar metadata, filters, and their associated thresholds. Changes are showcased in the following example. We have:

4. Added Pulsar metadata to both answers.
5. Defined filters for the 9.9.9.9 answer.

The result should look something like this:

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"example.com","domain":"pulsar.example.com","type":"A",
"answers":[{"answer":["1.1.1.1"],"meta":{"georegion":["US-EAST"],
"pulsar":{"job_id": "luxw9ir"}},
{"answer":["9.9.9.9"],"meta":{"georegion":["US-WEST"],
"pulsar":{"job_id": "luxxo37"}}}], "filters":[{"filter":"geotarget_regional"}, {"filter":"pulsar_performance_sort"}]}' https://api.nsonet.net/v1/zones/example.com/pulsar.example.com/A
```

Your performance metric is latency, so you decide to sort in ascending order. The changes are highlighted in the following sample:

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"example.com","domain":"pulsar.example.com","type":"A",
"answers":[{"answer":["1.1.1.1"],"meta":{"georegion":["US-EAST"],
"pulsar":{"job_id": "luxw9ir"}}, {"answer":["9.9.9.9"],
"meta":{"georegion":["US-WEST"], "pulsar":{"job_id": "luxxo37"}}}],
"filters":[{"filter":"geotarget_regional"},
{"filter":"pulsar_performance_sort","config":{"sort_descending": false}}]}' https://api.nsonet.net/v1/zones/example.com/pulsar.example.com/A
```

Pulsar Availability Sort

NOTE

You should set up a proper fallback strategy to ensure that other filters in NSI's Filter Chain can intelligently route traffic when there isn't sufficient performance telemetry. You should set up your Filter Chain to include these filters before each Pulsar filter. For example, you could include any of the geotarget filters to fall back on a geotargeting strategy. If you are distributing traffic across endpoints like CDNs or distributing traffic across a geographically dispersed infrastructure, use the `shuffle` or `cost` filters to better balance the lowest-cost traffic across your endpoints.

The following parameter is common to all of the Pulsar availability filters:

PARAMETER	TYPE	DESCRIPTION
<code>job_id</code>	string	Required. The ID of the Pulsar job to be associated with this record.

By default, answers are sorted from lowest to highest value. If there is not enough performance data available to make a decision, answers are passed through unchanged, and these decisions are logged as “insufficient” when you [view Pulsar decision data](#).

Each answer is associated with a specific `job_id`, which is tied to a probe that gathers performance and availability telemetry that is used to rate the answer.

The Pulsar Availability Sort filter requires the `pulsar` input metadata field, which is a custom data structure. It must be a dictionary, nested inside a list. You can find the fields of the dictionary above.



If there is not enough availability data, possible answers are passed through unchanged, and fallback logic is used. In these instances, the query is then marked as having “insufficient data” and is added to the associated category within Pulsar’s reporting dashboard. To learn more about insufficiency reporting, [click here](#) and scroll to the **Decisions** header. To retrieve decision insufficiencies via API, visit the NSI API Documentation [endpoint](#).



Copy Code

Define a Pulsar Availability Sort filter for a record



```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"","domain":"","child_domain":"","type":"","record_type":"","answers":[{"answer":["9.9.9.9"],"meta":{"pulsar":["job_id": "job_id"]}}],"filters":[{"filter":"pulsar_availability_sort"}]' https://api.nsonet.net/v1/zones/:zone/:domain/:record_type
```

Example Request:



```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"example.com","domain":"pulsar.example.com","type":"A","answers":[{"answer":["1.1.1.1"],"meta":{"georegion":["US-EAST"],"pulsar":["job_id": "luxw9ir"]}, {"answer":["9.9.9.9"],"meta":{"georegion":["US-WEST"],"pulsar":["job_id": "luxxo37"]}}],"filters":[{"filter":"geotarget_regional"}, {"filter":"pulsar_availability_sort"}]' https://api.nsonet.net/v1/zones/example.com/pulsar.example.com/A
```



Hide Code Examples



Pulsar Performance Sort

NOTE

You should set up a proper fallback strategy to ensure that other filters in NSI's Filter Chain can intelligently route traffic when there isn't sufficient performance telemetry. You should set up your Filter Chain to include these filters before each Pulsar filter. For example, you could include any of the geotarget filters to fall back on a geotargeting strategy. If you are distributing traffic across endpoints like CDNs or distributing traffic across a geographically dispersed infrastructure, use the `shuffle` or `cost` filters to better balance the lowest-cost traffic across your endpoints.

The following parameters are common to Pulsar performance filters:

PARAMETER	TYPE	DESCRIPTION
<code>job_id</code>	string	Required. The ID of the Pulsar job to be associated with this record.
<code>bias</code>	string	Only applies to the Pulsar Performance Stabilize and Pulsar Performance Sort filters. The bias to be associated with this answer. When Pulsar is comparing different latencies, it will apply this modifier to this answer to prioritize or de-prioritize it. Must take the form of a mathematical operator (one of <code>+</code> , <code>-</code> , or <code>*</code>) followed by a number which must be a positive float. The <code>+</code> and <code>-</code> operators add or subtract the specified number of milliseconds to the answer's latency, and the <code>*</code> operator scales the answer's latency by the specified amount.

By default, answers are sorted from lowest to highest value. You can set the `sort_descending` parameter to `true` to sort in ascending order to account for certain metrics that require answers to be sorted from highest to lowest. If there is not enough performance data available to make a decision, answers are passed through unchanged, and these decisions are logged as “insufficient” when you [view Pulsar decision data](#).


Sorting by ascending answers (worst to best) is best when your sort criteria is latency, because you want to send end users to endpoints that have lower latency values. On the other hand, if your performance measurement is defined by bandwidth and MBps, sorting in descending order would be more advantageous, because you would be sorting from the best value to the worst value.

Each answer is associated with a specific `job_id`, which is tied to a probe that gathers performance and availability telemetry that is used to rate the answer.

The Pulsar Performance Sort filter requires the `pulsar` input metadata field, which is a custom data structure. It must be a dictionary, nested inside a list. You can find the fields of the dictionary above.


If there is not enough availability data, possible answers are passed through unchanged, and fallback logic is used. In these instances, the query is then marked as having “insufficient data” and is added to the associated category within Pulsar’s reporting dashboard. To learn more about insufficiency reporting, [click here](#) and scroll to the **Decisions** header. To retrieve decision insufficiencies via API, visit the NSI API Documentation [endpoint](#).

PARAMETER	TYPE	DESCRIPTION
<code>sort_descending</code>	boolean	If enabled, the filter bases its sorting behavior on the worst-performing answer instead of the best, then sorts the remaining answers accordingly.



```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"","domain":"","child_domain":"","type":"","record_type","answers":[{"answer":["1.1.1.1"],"meta":{"pulsar":["job_id":"","luxw9ir"]}],{"answer":["9.9.9.9"],"pulsar":["job_id":"","jobID"]}]},{"filters":[{"filter":"","pulsar_performance_sort","config":{"sort_descending": true}}]}' https://api.nsonet.net/v1/zones/:zone/:domain/:record_type
```

Example Request:



```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"","example.com","domain":"","pulsar.example.com","type":"","A","answers":[{"answer":["1.1.1.1"],"meta":{"pulsar":["job_id":"","luxw9ir"]}],{"answer":["9.9.9.9"],"pulsar":["job_id":"","luxxo37"]}]},{"filters":[{"filter":"","pulsar_performance_sort","config":{"sort_descending": true}}]}' https://api.nsonet.net/v1/zones/example.com/pulsar.example.com/A
```

Example Request:

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"example.com","domain":"pulsar.example.com","type":"A", "answers":[{"answer":["1.1.1.1"],"meta":{"georegion":["US-EAST"],"pulsar":["job_id":"luxw9ir"]}, {"answer":["9.9.9.9"],"meta":{"georegion":["US-WEST"],"pulsar":["job_id": "luxxo37"]}], "filters":[{"filter":"geotarget_regional"}, {"filter":"pulsar_performance_sort","config":{"sort_descending": false}}]}' https://api.nsonone.net/v1/zones/example.com/pulsar.example.com/A
```

 Hide Code Examples

Pulsar Performance Stabilize

NOTE

You should set up a proper fallback strategy to ensure that other filters in NSI's Filter Chain can intelligently route traffic when there isn't sufficient performance telemetry. You should set up your Filter Chain to include these filters before each Pulsar filter. For example, you could include any of the geotarget filters to fall back on a geotargeting strategy. If you are distributing traffic across endpoints like CDNs or distributing traffic across a geographically dispersed infrastructure, use the `shuffle` or `cost` filters to better balance the lowest-cost traffic across your endpoints.

The following parameters are common to all of the Pulsar performance filters:



PARAMETER	TYPE	DESCRIPTION
job_id	string	Required. The ID of the Pulsar job to be associated with this record.
bias	string	Only applies to the Pulsar Performance Stabilize and Pulsar Performance Sort filters. The bias to be associated with this answer. When Pulsar is comparing different latencies, it will apply this modifier to this answer to prioritize or de-prioritize it. Must take the form of a mathematical operator (one of $+$, $-$, or $*$) followed by a number which must be a positive float. The $+$ and $-$ operators add or subtract the specified number of milliseconds to the answer's latency, and the $*$ operator scales the answer's latency by the specified amount.

By default, this filter determines which answer has the best performance relative to the other answers that have also passed through to this filter. After determining the best or worst performance in each of the answers, the filter then compares its defined performance threshold to the other responses. If there is not enough performance data available to make a decision, answers are passed through unchanged, and these decisions are logged as “insufficient” when you [view Pulsar decision data](#).

If you sort by the default behavior, the answer that has the best performance based on the threshold is discarded. This can be useful if, for example, you are setting a threshold for latency, where a higher value can slow the flow of traffic. Conversely, your threshold could be based on bandwidth. If you sort in descending order, the worst-performing answer will be discarded.

For example, assume that you have three endpoints: Endpoint A, Endpoint B, and Endpoint C. You set the latency threshold to 50%. Endpoint A has a response time of 160ms, Endpoint B has a response time of 110ms, and Endpoint C has a response time of 100ms. Of these three endpoints, Endpoint A is dropped, because it had a higher latency. Endpoints B and C are passed along in the order that they were passed into the filter.

Use other filters, such as geographic filters or shuffle filters before this filter as a fallback to adequately balance traffic.

Set the stabilization threshold in the filter definition. If you want to set a different threshold for an answer, define it in the `answer` metadata.

The Pulsar Performance Stabilize filter requires the `pulsar` input metadata field, which is a custom data structure. It must be a dictionary, nested inside a list. You can find the fields of the dictionary above.

PARAMETER	TYPE	DESCRIPTION
sort_descending	boolean	If enabled, the filter bases the cutoff on the best-performing answer instead of the worst, then removes all answers below that cutoff instead of above. The cutoff will be the Pulsar value of the best answer, minus whatever amount is specified by the <code>stabilization_threshold</code> , instead of adding. The default value is <code>false</code> .
stabilization_threshold	string	Required. Specifies a threshold above or below the percentage that you specify.



Define a Pulsar Performance Stabilize filter and its configuration for a record

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"","domain":"","child_domain":"","type":"","answers":[{"answer":["9.9.9.9"],"meta":{"pulsar":{"job_id":"","job_id"}}],"filters":[{"filter":"pulsar_performance_stabilize", "config": {"stabilization_threshold": ":threshold_value"}}]}' https://api.nstone.net/v1/zones/:zone/:domain/:record_type
```

Example Request:

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":"example.com","domain":"pulsar.example.com","type":"A", "answers":[{"answer":["1.1.1.1"],"meta":{"country":["US"],"pulsar":["job_id": "1uxw9ir"}],{"answer":["9.9.9.9"],"meta":{"country":["CA"],"pulsar":["job_id": "1uxxo37"}]}],"filters":[{"geotarget_country"}, {"filter": "pulsar_performance_stabilize", "config": {"stabilization_threshold": "10"}}]' https://api.nsonone.net/v1/zones/example.com/pulsar.example.com/A
```

 Hide Code Examples

Pulsar Availability Threshold

NOTE

You should set up a proper fallback strategy to ensure that other filters in NSI's Filter Chain can intelligently route traffic when there isn't sufficient performance telemetry. You should set up your Filter Chain to include these filters before each Pulsar filter. For example, you could include any of the geotarget filters to fall back on a geotargeting strategy. If you are distributing traffic across endpoints like CDNs or distributing traffic across a geographically dispersed infrastructure, use the `shuffle` or `cost` filters to better balance the lowest-cost traffic across your endpoints.



The following parameter is common to all of the Pulsar availability filters:

PARAMETER	TYPE	DESCRIPTION
job_id	string	Required. The ID of the Pulsar job to be associated with this record.

The Pulsar Availability Threshold filter determines which answers have the best availability based on a specified threshold. Answers with the most availability (based on a defined availability threshold) are kept. Answers that do not meet this stabilization threshold are removed. The availability threshold that you set is the value used to determine which answers should be removed.

For example, assume that you have three endpoints (Endpoint A, Endpoint B, and Endpoint C). You set the threshold to 95%. Endpoint A has an availability of 90%, Endpoint B has an availability of 97%, and Endpoint C has an availability of 100%. Endpoint A will be removed, because it's available less than 95% of the time. Endpoints B and C are passed through in the existing order they were passed into this filter.

If there is not enough availability data, possible answers are passed through unchanged, and fallback logic is used. In these instances, the query is then marked as having “insufficient data” and is added to the associated category within Pulsar’s reporting dashboard. To learn more about insufficiency reporting, [click here](#) and scroll to the **Decisions** header. To retrieve decision insufficiencies via API, visit the NSI API Documentation [endpoint](#).

Use other filters, such as geographic filters or shuffle filters before this filter as a fallback to adequately balance traffic.

Set the threshold in the filter definition. If you want to define different thresholds for answers, set the threshold for each answer in the answer metadata.



The Pulsar Availability Threshold filter requires the `pulsar` input metadata field, which is a custom data structure. It always must be a dictionary, nested inside a list. The dictionary is defined above.

PARAMETER	TYPE	DESCRIPTION
threshold	string	Required. Specifies a threshold above or below the percentage that you specify.



Define a Pulsar Availability Threshold filter for an answer

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone":":zone","domain":":child_d  
omain","type":":record_type", "answers":[{"answer":["9.9.9.9"],"meta":{"pulsar":["job_  
id": ":job_id"]}], "filters":[{"filter": "pulsar_availability_threshold", "config": {"threshol  
d": ":threshold_value"]}]}' https://api.nsonet.net/v1/zones/:zone/:domain/:record_typ  
e
```

Example Request:

```
$ curl -X PUT -H "X-NSONE-Key: $API_KEY" -d '{"zone": "example.c  
om","domain": "pulsar.example.com","type": "A", "answers":[{"answe  
r": ["1.1.1.1"], "meta": {"pulsar": {"job_id": "luxw9ir"}}, {"answer": ["9.9.9.  
9"], "meta": {"pulsar": {"job_id": "luxxo37"}}], "filters": [{"filter": "shuffl  
e"}, {"filter": "pulsar_availability_threshold", "config": {"threshold": "0.9  
5"}}]}' https://api.nsonet.net/v1/zones/example.com/pulsar.exempl  
e.com/A
```