LOGZILLA DOCUMENTATION

Search Types

Three ways to query LogZilla events: the main query bar with severity and host filters, widget data search, and direct URL entry

LogZilla Dashboards · Generated June 11, 2026 · logzilla.ai/docs/using-the-dashboard/search-types

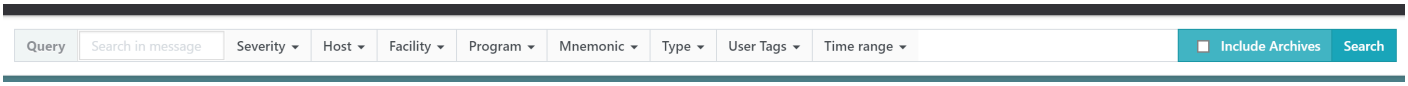
The Search Results page will provide a list of events matching the criteria set by one of:

- The Main Query Bar
- Widget Data Search
- Direct URL Entry

Main Query Bar

The Query Bar provides an easy-to-use interface for setting filters on queries. For syntax on text matching, please refer to the Search Syntax (https://www.logzilla.ai/docs/using-the-dashboard/search-syntax) help document.

Main Query Bar



Users may also set more filtering criteria using the query bar such as:

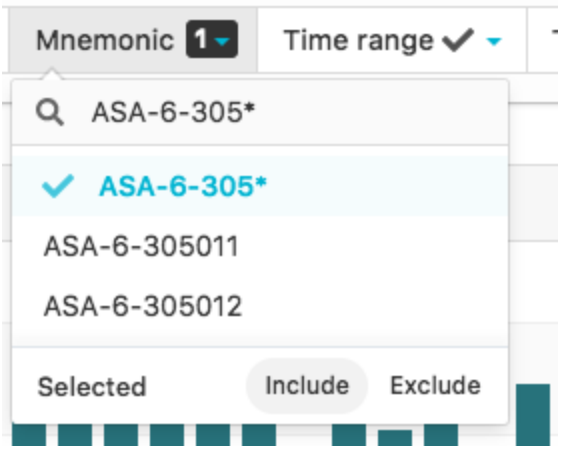
- Severity
- Host
- Facility
- Program
- Cisco Mnemonics
- Time Range
- Status (Actionable, Non-Actionable, Unknown)
- User Tag

Each dropdown provides a list of recently seen entries. Wildcards may be used to search for any unlisted entries in the dropdown.

In the example below, the search results would return all events matching `ASA-6-305*`.

Note that after typing `ASA-6-305*` (case-sensitive) users must **select the wildcard pattern typed in** as seen below in the screenshot (indicated by the blue check mark).

Query Bar Filter Example

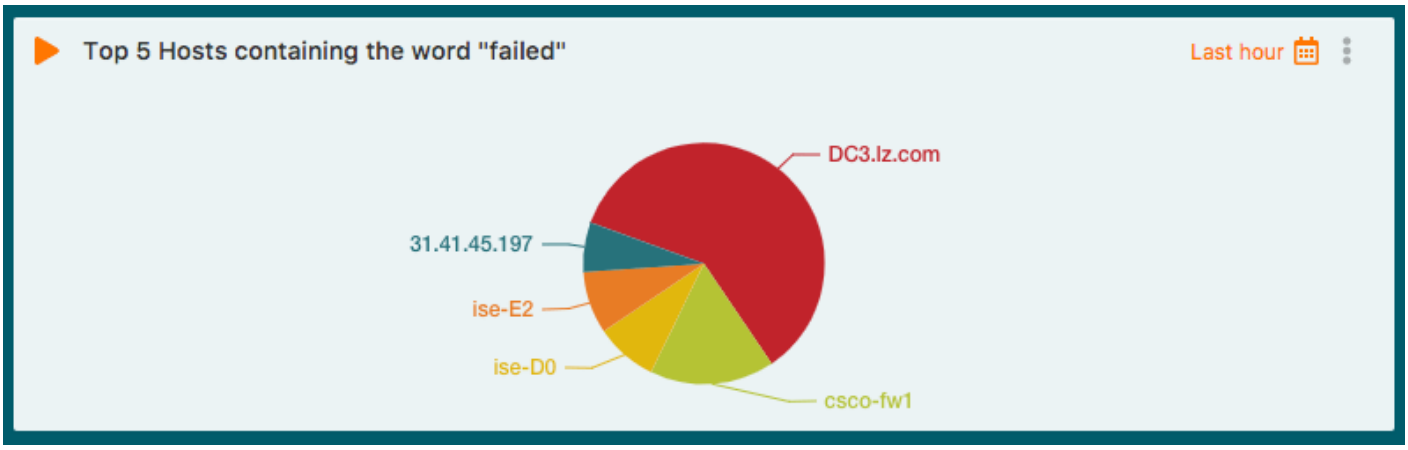


Widget Data Search

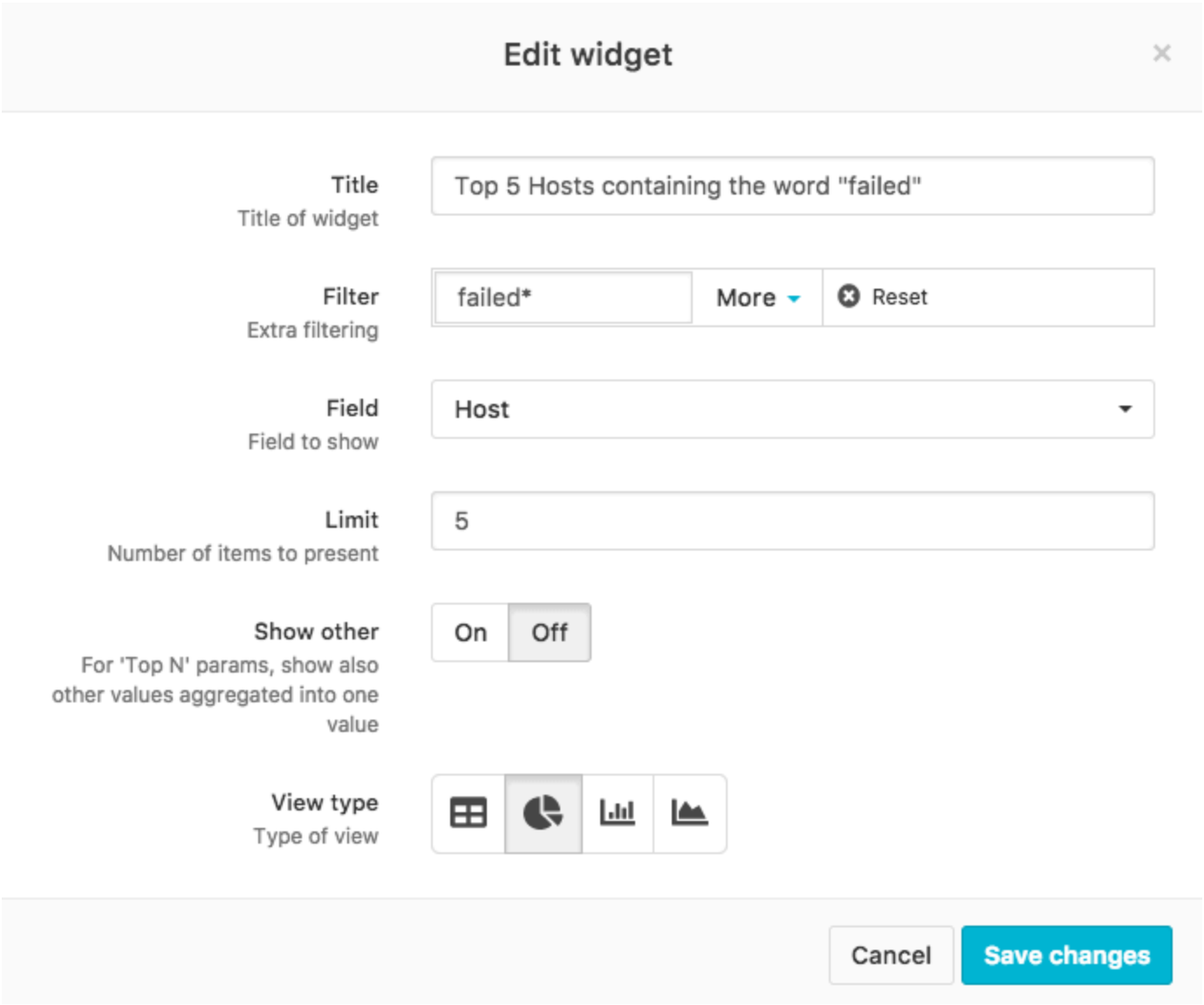
All widgets have an option to perform a search of the data contained in the widget itself. This allows users to perform searches without having to manually enter all of the filter criteria set in that widget.

For example, the widget below has a filter set for showing only the Top 5 hosts which contain the word `failed` in the message.

Top 5 Widget With Filters

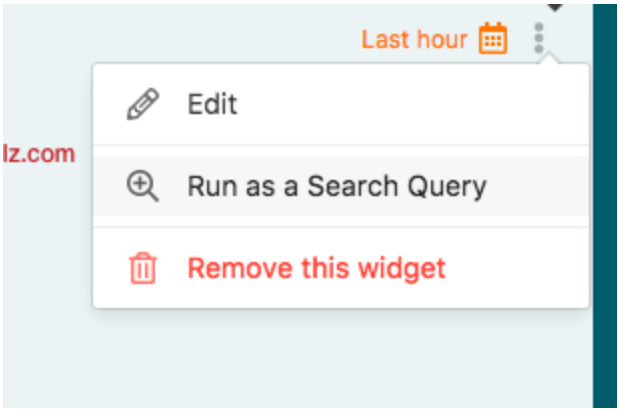


Settings For The Widget Above



To search for all events contained in that widget, simply select the widget handle, then click **Run as Search Query**.

Query From Widget



Direct URL Entry

LogZilla also allows direct searching via the browser's URL by typing the query string along with any desired filter criteria, for example: `http://logzilla.company.com/search?{querystring}`

Usage

- The `search` call must start with a question mark, i.e.: `/search?msg=foo`
- It may contain keys with or without values separated by an `=` (equal) sign or pairs separated by ampersand.
- If multiple values for a single parameter are present in the URL (e.g.: `/search?facility=USER&facility=KERN`), the requested search for these two items will return results for `either` of the two filters (boolean `OR`).

Example

```
http://logzilla.company.com/search?msg=successful%20auth&facility=USER&severity=Info&time_range=2017-12-13T00:00~14T00:00
```

URL Query String Parameters

What "array" means here

- **array**: Multiple values (strings) can be provided for that parameter. The search will match events where the field equals any of those values (boolean OR).
- **How to pass it in the URL**: Repeat the same key multiple times.
  - Example: multiple facilities
    - `/search?facility=USER&facility=KERN`
  - Example: multiple severities
    - `/search?severity=Error&severity=Warning`
- **Effect**: Matches entries with any of the listed values.
  - E.g., `/search?facility=USER&facility=KERN` returns events with facility USER OR KERN.
- **Applies to**: `facility`, `host`, `mnemonic`, `program`, `severity` (each can be a single `string` or an `array<string>` via repeated params).

- **Full example**:

```
/search?msg=successful%20auth&facility=USER&facility=KERN&severity=Info&severity=Error
```

`msg`

Type: `string`

Search terms are encoded as a Uniform Resource Identifier (URI)(https://tools.ietf.org/html/rfc3986) component (`encodeURIComponent()` (https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/encodeURIComponent) function or equivalent) supporting mixed-mode search syntax (https://www.logzilla.ai/docs/using-the-dashboard/search-syntax) searches.

`facility`

Type: `string` **or** `array<string>`

Facility keywords (case-insensitive) are defined in RFC 3164 (https://tools.ietf.org/html/rfc3164#section-4.1.1).

Supported Facility Values

Keyword	Description
KERN	Kernel messages
USER	User-level messages
MAIL	Mail system
DAEMON	System daemons
AUTH	Security/authorization messages (note 1)
SYSLOG	Messages generated internally by syslogd
LPR	Line printer subsystem
NEWS	Network news subsystem
UUCP	UUCP subsystem
CLOCK	Clock daemon (note 2)

Keyword	Description
AUTHPRIV	Security/authorization messages (note 1)
FTP	FTP daemon
NTP	NTP subsystem
AUDIT	Log audit (note 1)
ALERT	Log alert (note 1)
CRON	Clock daemon (note 2)
LOCAL0	Local use 0 (local0)
LOCAL1	Local use 1 (local1)
LOCAL2	Local use 2 (local2)
LOCAL3	Local use 3 (local3)
LOCAL4	Local use 4 (local4)
LOCAL5	Local use 5 (local5)
LOCAL6	Local use 6 (local6)
LOCAL7	Local use 7 (local7)

These values may also be found in the LogZilla API on the server at `/api/dictionaries/facility`

```
GET /api/dictionaries/facility
```

host

Type: `string` or `array<string>`

Hostname or IP address of the device.

mnemonic

Type: `string` or `array<string>`

Cisco mnemonic.

NOTE: Mnemonics should be passed without the `%` prefix as the `%` is a reserved character for URI encoding. e.g.: `SYS-5-CONFIG_I` instead of `%SYS-5-CONFIG_I`

program

Type: `string` or `array<string>`

Name of the source program/process.

severity

Type: `string` or `array<string>`

Severity name (case-insensitive) as defined in RFC 5424 (https://tools.ietf.org/html/rfc5424#section-6.2.1).

Supported Severity Values

Name	Description
`Emergency`	System is unusable
`Alert`	Action must be taken immediately
`Critical`	Critical conditions
`Error`	Error conditions
`Warning`	Warning conditions
`Notice`	Normal but significant condition
`Info`	Informational messages
`Debug`	Debug-level messages

These values may also be found in the LogZilla API on the server at `/api/dictionaries/severity`

```
GET /api/dictionaries/severity
```

time_range

Type: `string` or `start:iso8601~end:iso8601`

Default: `last_1_hours`

Option 1: Time range preset
Use relative time range preset as defined in the API on the server at `/api/dictionaries/time_range`.

Preset	Description
`last_1_minutes`	Last minute
`last_1_hours`	Last hour
`last_6_hours`	Last 6 hours
`today`	Today
`yesterday`	Yesterday
`last_3_days`	Last 3 days
`last_7_days`	Last week
`last_30_days`	Last 30 days

Fetch list from API

```
GET /api/dictionaries/time_range
```

Option 2: Date time range
Searches within a specific time range using combined ISO 8601 (https://www.w3.org/TR/NOTE-datetime) date/time representation of start and end times, should contain a tilde character (`~`) as the separator (*basic format* is `YYYY-MM-DDTHH:mm:ss.sss~YYYY-MM-DDTHH:mm:ss.sssZ`). If any elements are missing from the end value, they are assumed to be the same as the starting value.

Examples
Cross-day range:

```
2017-12-01T18:00~2018-01-03T01:00
```

Searches from December 1, 2017 at 6:00 PM through January 3, 2018 at 1:00 AM.

Date range (same time each day):

```
2017-11-04~06
```

Searches from November 4, 2017 at 12:00 AM through November 6, 2017 at 12:00 AM.

Same day, different times:

```
2017-08-04T08:00:00~11:00
```

Searches on August 4, 2017 from 8:00 AM through 11:00 AM.

`sort`

Type: `string`

Default: `-last_occurrence`

Name of the field to sort by (`first_occurrence`, `last_occurrence` or `counter`). Prefixing with a negative sign reverses the order.