You are looking at documentation for an older release. Not what you want? See the current release documentation (../current/index.html).

json

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Version: 3.0.2
- Released on: July 14, 2016
- Changelog (https://github.com/logstash-plugins/logstash-filter-json/blob/master/CHANGELOG.md#302)

Getting Help

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

For questions about the plugin, open a topic in the Discuss (http://discuss.elastic.co) forums. For bugs or feature requests, open an issue in Github (https://github.com/elastic/logstash). For the list of Elastic supported plugins, please consult the Elastic Support Matrix (https://www.elastic.co/support/matrix#show_logstash_plugins).

Description

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

This is a JSON parsing filter. It takes an existing field which contains JSON and expands it into an actual data structure within the Logstash event.

By default it will place the parsed JSON in the root (top level) of the Logstash event, but this filter can be configured to place the JSON into any arbitrary event field, using the `target` configuration.

This plugin has a few fallback scenario when something bad happen during the parsing of the event. If the JSON parsing fails on the data, the event will be untouched and it will be tagged with a `_jsonparsefailure` then you can use conditionals to clean the data. You can configured this tag with then `tag_on_failure` option.

If the parsed data contains a `@timestamp` field, we will try to use it for the event's `@timestamp`, if the parsing fails, the field will be renamed to `_@timestamp` and the event will be tagged with a `_timestampparsefailure`.

Synopsis

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

This plugin supports the following configuration options:

Required configuration options:

```
json {
    source => ...
}
```

Available configuration options:

Setting	Input type	Required
`add_field` (plugins-filters-json.html#plugins-filters-json-add_field)	hash (configuration-file-structure.html#hash)	No
`add_tag` (plugins-filters-json.html#plugins-filters-json-add_tag)	array (configuration-file-structure.html#array)	No
`enable_metric` (plugins-filters-json.html#plugins-filters-json-enable_metric)	boolean (configuration-file-structure.html#boolean)	No
`id` (plugins-filters-json.html#plugins-filters-json-id)	string (configuration-file-structure.html#string)	No
`periodic_flush` (plugins-filters-json.html#plugins-filters-json-periodic_flush)	boolean (configuration-file-structure.html#boolean)	No
`remove_field` (plugins-filters-json.html#plugins-filters-json-remove_field)	array (configuration-file-structure.html#array)	No
`remove_tag` (plugins-filters-json.html#plugins-filters-json-remove_tag)	array (configuration-file-structure.html#array)	No

Setting	Input type	Required
`skip_on_invalid_json` (plugins-filters-json.html#plugins-filters-json-skip_on_invalid_json)	boolean (configuration-file-structure.html#boolean)	No
`source` (plugins-filters-json.html#plugins-filters-json-source)	string (configuration-file-structure.html#string)	Yes
`tag_on_failure` (plugins-filters-json.html#plugins-filters-json-tag_on_failure)	array (configuration-file-structure.html#array)	No
`target` (plugins-filters-json.html#plugins-filters-json-target)	string (configuration-file-structure.html#string)	No

Details

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

`add_field`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is hash (configuration-file-structure.html#hash)
- Default value is `{}`

If this filter is successful, add any arbitrary fields to this event. Field names can be dynamic and include parts of the event using the `%{field}`.

Example:

```
filter {
  json {
    add_field => { "foo_%{somefield}" => "Hello world, from %{host}" }
  }
}
```

```
# You can also add multiple fields at once:
filter {
  json {
    add_field => {
      "foo_%{somefield}" => "Hello world, from %{host}"
      "new_field" => "new_static_value"
    }
  }
}
```

If the event has field `"somefield" == "hello"` this filter, on success, would add field `foo_hello` if it is present, with the value above and the `%{host}` piece replaced with that value from the event. The second example would also add a hardcoded field.

`add_tag`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is array (configuration-file-structure.html#array)
- Default value is `[]`

If this filter is successful, add arbitrary tags to the event. Tags can be dynamic and include parts of the event using the `%{field}` syntax.

Example:

```
filter {
  json {
    add_tag => [ "foo_%{somefield}" ]
  }
}
```

```
# You can also add multiple tags at once:
filter {
  json {
    add_tag => [ "foo_%{somefield}", "taggedy_tag"]
  }
}
```

If the event has field `"somefield" == "hello"` this filter, on success, would add a tag `foo_hello` (and the second example would of course add a `taggedy_tag` tag).

`enable_metric`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is boolean (configuration-file-structure.html#boolean)
- Default value is `true`

Disable or enable metric logging for this specific plugin instance by default we record all the metrics we can, but you can disable metrics collection for a specific plugin.

`id`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is string (configuration-file-structure.html#string)
- There is no default value for this setting.

Add a unique `ID` to the plugin configuration. If no ID is specified, Logstash will generate one. It is strongly recommended to set this ID in your configuration. This is particularly useful when you have two or more plugins of the same type, for example, if you have 2 grok filters. Adding a named ID in this case will help in monitoring Logstash when using the monitoring APIs.

```
output {
 stdout {
   id => "my_plugin_id"
 }
}
```

`periodic_flush`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is boolean (configuration-file-structure.html#boolean)
- Default value is `false`

Call the filter flush method at regular interval. Optional.

`remove_field`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is array (configuration-file-structure.html#array)
- Default value is `[]`

If this filter is successful, remove arbitrary fields from this event. Example:

```
filter {
  json {
    remove_field => [ "foo_%{somefield}" ]
  }
}
```

```
# You can also remove multiple fields at once:
filter {
  json {
    remove_field => [ "foo_%{somefield}", "my_extraneous_field" ]
  }
}
```

If the event has field `"somefield" == "hello"` this filter, on success, would remove the field with name `foo_hello` if it is present. The second example would remove an additional, non-dynamic field.

`remove_tag`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is array (configuration-file-structure.html#array)
- Default value is `[]`

If this filter is successful, remove arbitrary tags from the event. Tags can be dynamic and include parts of the event using the `%{field}` syntax.

Example:

```
filter {
  json {
    remove_tag => [ "foo_%{somefield}" ]
  }
}
```

```
# You can also remove multiple tags at once:
filter {
  json {
    remove_tag => [ "foo_%{somefield}", "sad_unwanted_tag"]
  }
}
```

If the event has field `"somefield" == "hello"` this filter, on success, would remove the tag `foo_hello` if it is present. The second example would remove a sad, unwanted tag as well.

`skip_on_invalid_json`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is boolean (configuration-file-structure.html#boolean)
- Default value is `false`

Allow to skip filter on invalid json (allows to handle json and non-json data without warnings)

`source`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- This is a required setting.
- Value type is string (configuration-file-structure.html#string)
- There is no default value for this setting.

The configuration for the JSON filter:

```
source => source_field
```

For example, if you have JSON data in the `message` field:

```
filter {
  json {
    source => "message"
  }
}
```

The above would parse the json from the `message` field

`tag_on_failure`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is array (configuration-file-structure.html#array)
- Default value is `["_jsonparsefailure"]`

Append values to the `tags` field when there has been no successful match

`target`

edit (https://github.com/logstash-plugins/logstash-filter-json/edit/master/lib/logstash/filters/json.rb)

- Value type is string (configuration-file-structure.html#string)
- There is no default value for this setting.

Define the target field for placing the parsed data. If this setting is omitted, the JSON data will be stored at the root (top level) of the event.

For example, if you want the data to be put in the `doc` field:

```
filter {
  json {
    target => "doc"
  }
}
```

JSON in the value of the `source` field will be expanded into a data structure in the `target` field.

NOTE

if the `target` field already exists, it will be overwritten!

« jdbc_streaming (plugins-filters-jdbc_streaming.html)

json_encode » (plugins-filters-json_encode.html)

Top Videos

- Elasticsearch Demo (https://www.elastic.co/webinars/getting-started-elasticsearch?baymax=default&elektra=docs&storm=top-video)
- Kibana 101 (https://www.elastic.co/webinars/getting-started-kibana?baymax=default&elektra=docs&storm=top-video)
- Logstash Primer (https://www.elastic.co/webinars/getting-started-logstash?baymax=default&elektra=docs&storm=top-video)

On this page