

# Labels in Prometheus Alerts: Think Twice Before Using Them

Learn how to write alerting rules and configure the Prometheus alertmanager to send concise, easy-to-understand notifications.



by Elena Morozova MVB · Oct. 18, 18 · Performance Zone · Tutorial



Like (3)



Comment (0)



Save



Tweet



3,102 Views

Sensu is an open source monitoring event pipeline. [Try it today.](#)

In this post, we will look at how to write alerting rules and how to configure the [Prometheus](#) alertmanager to send concise and easy to understand notifications.

Some alerts may contain labels and others may not. For example, here is an Instance Down alert with the labels ( `{{ $labels.instance }}` and `{{ $labels.job }}` ) and another one without labels:

```
1 groups:
2 - name: example
```



```

3 rules:
4 - alert: InstanceDownLabels
5   expr: up
6   for: 5m
7   labels:
8     severity: page
9   annotations:
10    summary: "Instance {{ $labels.instance }} down"
11    description: "{{ $labels.instance }} of job {{ $labels.job }} has been down for"
12 - alert: InstanceDownNoLabels
13   expr: up
14   for: 5m
15   labels:
16     severity: critical
17   annotations:
18    summary: "Instance down"
19    description: "Something has been down for more than 5 minutes."

```

Let's create a Slack receiver. We can do this by using an example from the Prometheus [documentation](#):

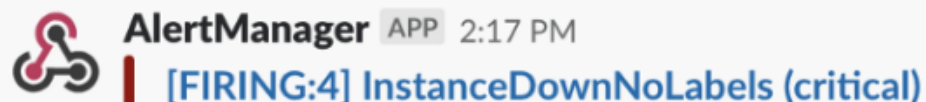
```

1 - name: 'team-x'
2   slack_configs:
3     - channel: '#alerts'
4       text: "<!channel> \nsummary: {{ .CommonAnnotations.summary }}\ndescription: {{

```

This receiver config says we want to get a notification with a common summary and common description.

But with these settings, our Slack notifications looks like this:



@channel

summary: Instance down

description: Something has been down for more than 5 minutes.

[FIRING:4] InstanceDownLabels (page)

@channel

summary:

description:

The first alert, `InstanceDownNoLabels`, looks good. But why are the summary and description empty for `InstanceDownLabels`?

This happens because every time series is uniquely identified by its *metric name* and a set of labels and every unique combination of key-value label pairs represents a new alert for this time series.

We used variables like `{{ $labels.instance }}` and `{{ $labels.job }}` in our description and summary, and as a result, there is no common value for them.

We can try ranging over all received alerts (see [example](#)):

```
1 - name: 'default-receiver'
2   slack_configs:
3     - channel: '#alerts'
4       title: "{{ range .Alerts }}{{ .Annotations.summary }}\n{{ end }}"
5       text: "{{ range .Alerts }}{{ .Annotations.description }}\n{{ end }}"
```

But with this config, our Slack notifications look like this:



**AlertManager** APP 2:49 PM

Instance down

Instance down

Instance down

Instance down

Something has been down for more than 5 minutes.

Something has been down for more than 5 minutes.

Something has been down for more than 5 minutes.

Something has been down for more than 5 minutes.

Instance distributor:8081 down

Instance ingester:8082 down

Instance manager:8080 down

Instance localhost:9090 down

distributor:8081 of job distributor has been down for more than 5 minutes.

ingester:8082 of job ingester has been down for more than 5 minutes.

manager:8080 of job manager has been down for more than 5 minutes.

localhost:9090 of job prometheus has been down for more than 5 minutes.

Now the alerts are not blank, but the first one contains duplicates in the Slack title and text.

One solution is checking both cases. For example, we can create our own template and save it in a file, my.tmpl:

```
1 {{ define "slack.my.title" -}}  
2     {{- if .CommonAnnotations.summary -}}
```

```

3      {{- .CommonAnnotations.summary -}}
4      {{- else -}}
5      {{- with index .Alerts 0 -}}
6          {{- .Annotations.summary -}}
7      {{- end -}}
8  {{- end -}}
9  {{- end }}
10 {{ define "slack.my.text" -}}
11     {{- if .CommonAnnotations.description -}}
12         {{- .CommonAnnotations.description -}}
13     {{- else -}}
14         {{- range $i, $alert := .Alerts }}
15             {{- "\n" -}} {{- .Annotations.description -}}
16         {{- end -}}
17     {{- end -}}
18 {{- end }}

```

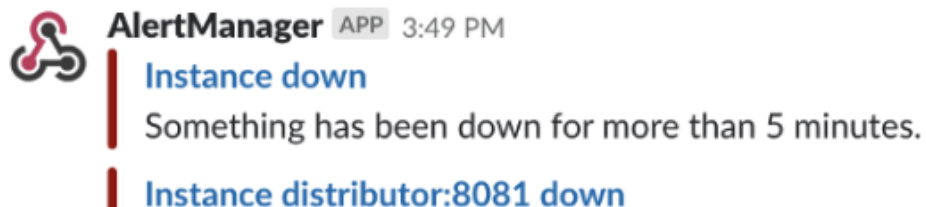
And if we use this custom template for Slack notifications:

```

1 - name: 'default-receiver'
2   slack_configs:
3     - channel: '#alerts'
4       title: '{{ template "slack.my.title" . }}'
5       text: '{{ template "slack.my.text" . }}'
6   templates:
7     - 'my.tmpl'

```

Our Slack messages look like this now:



```
distributor:8081 of job distributor has been down for more than 5 minutes.  
ingester:8082 of job ingester has been down for more than 5 minutes.  
manager:8080 of job manager has been down for more than 5 minutes.  
localhost:9090 of job prometheus has been down for more than 5 minutes.
```

Now we have all needed information without duplicates. To make our template look nice, we use minuses - before and after left and right delimiters {{ and }}. See [go text/template](#) documentation. For new lines we use {{ "\n" }}.

For the title, we check if there is a common summary and use it; otherwise, we use the summary from the first alert to keep a summary short.

For the text, we use the common description if one exists (not empty) or we range over all alerts and print the description for each of them. But there might be a lot of different values for labels and a lot of different descriptions. It is a good idea to add some limit for them, for example, only the first 10 descriptions:

```
1 {{- range $i, $alert := .Alerts -}}  
2     {{- if lt $i 10 -}}  
3         {{- "\n" -}} {{- index $alert.Annotations "description" -}}  
4     {{- end -}}  
5 {{- end -}}
```

The same applies to alerting rules with {{\$value}} inside an annotation.

## Conclusion

To get proper notifications, we need to make sure that our metrics, alerts, and receiver match each other. In particular, if we use labels or values in a field, we

should expect to have different values of this field, and our templates need to deal with that. By contrast, if a field is static (doesn't contain any labels, value), there is a common value across all alerts for this rule.

Of course, we can use the same approach for other receivers like email, PagerDuty, OpsGenie, etc. For example, an email message for the same alert as above with a custom template looks like this:



## InstanceDownLabels

distributor:8081 of job distributor has been down for more than 5 minutes.  
ingester:8082 of job ingester has been down for more than 5 minutes.  
manager:8080 of job manager has been down for more than 5 minutes.  
localhost:9090 of job prometheus has been down for more than 5 minutes.

Thanks, the Weaveworks team

To disable these notifications, adjust the [Settings](#).



See the configs for receivers with the < [tpl\\_string](#) > format.

Happy monitoring!

---

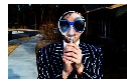
Sensu: workflow automation for monitoring. [Learn more—download the whitepaper.](#)

---

## Like This Article? Read More From DZone



**Creating Visualizations in Grafana**



**Prometheus Monitoring With Grafana**




**Alert Fatigue, Part 4: Alert Consolidation**



**Free DZone Refcard  
Introduction to Docker Monitoring**

Topics: [PERFORMANCE](#) , [PROMETHEUS](#) , [MONITORING](#) , [ALERTING](#) , [TUTORIAL](#)

 **Like (3)**    **Comment (0)**    **Save**    **Tweet**

 **3,102 Views**

Published at DZone with permission of Elena Morozova , DZone MVB. [See the original article here.](#) 

Opinions expressed by DZone contributors are their own.



# Performance Partner Resources

Open source resources for monitoring Chef deployments



Sensu

[Whitepaper] Workflow automation for monitoring



Sensu

[How to] Run Nagios as time-series data with Sensu, InfluxDB and Grafana



Sensu