

- 1. 下载插件
  - 2. 部署
  - 3. 配置 systemd
  - 4. 配置文件
  - 5. 启动服务
  - 6. 在 alertmanager 中配置
  - 7 测试
- 传送门

## 1. 下载插件

1 下载钉钉插件 prometheus-webhook-dingtalk

二进制下载地址: <https://github.com/timonwong/prometheus-webhook-dingtalk/releases>

```
curl -o prometheus-webhook-dingtalk.tgz -L
https://github.com/timonwong/prometheus-webhook-
dingtalk/releases/download/v2.1.0/prometheus-webhook-dingtalk-2.1.0.linux-
amd64.tar.gz
```

## 2. 部署

将二进制包解压后, 放到合适的位置。

这里放在的 /usr/local/ 下, 命名为 prometheus-webhook-dingtalk

```
tar -xf prometheus-webhook-dingtalk.tgz
mv prometheus-webhook-dingtalk-2.1.0.linux-amd64 /usr/local/prometheus-webhook-
dingtalk
```

## 3. 配置 systemd

命令行启动项说明

配置项	含义
<code>--web.listen-address=:8060</code>	程序监听端口, 默认 8060
<code>--web.enable-lifecycle</code>	支持通过发送 HTTP 请求, 热更新配置文件
<code>--config.file=config.yml</code>	指定配置文件路径
<code>--log.level=info</code>	日志级别 [debug, info, warn, error]
<code>--log.format=logfmt</code>	日志输出格式 [logfmt, json]
<code>--web.enable-ui</code>	可以使用 <a href="http://ip:8060/ui">http://ip:8060/ui</a> 打开测试模板的web界面
<code>--version</code>	输出版本信息

```
[Unit]
Description=The prometheus webhook dingtalk
After=network-online.target
Wants=network-online.target

[Service]
WorkingDirectory=/usr/local/prometheus-webhook-dingtalk
ExecStart=/usr/local/prometheus-webhook-dingtalk/prometheus-webhook-dingtalk --
config.file=config.yml --web.enable-lifecycle --web.enable-ui

KillSignal=SIGQUIT

Restart=always

RestartPreventExitStatus=1 6 SIGABRT

TimeoutStopSec=5
KillMode=process
PrivateTmp=true
LimitNOFILE=1048576
LimitNPROC=1048576

[Install]
WantedBy=multi-user.target
```

`config.yml` 是相对于 `WorkingDirectory` 指定的目录

## 4. 配置文件

在部署包里有个示例文件 `config.example.yml`，可以复制一份名为 `config.yml` 之后进行配置。

```
cp config.example.yml config.yml
```

有如下配置项可供配置。

```
## 请求超时时间
# timeout: 5s

## 为了从头开始编写模板，请取消对以下行的注释
#no_builtin_template: true

## 自定义模版文件路径
#templates:
# - contrib/templates/legacy/template.tpl

## 您也可以使用 `default_message` 覆盖默认模板
## The following example to use the 'legacy' template from v0.3.0
#default_message:
#  title: '{{ template "legacy.title" . }}'
#  text: '{{ template "legacy.content" . }}'

## Targets, 以前被称为 "profiles"
targets:
```

```
webhook1:
  url: https://oapi.dingtalk.com/robot/send?access_token=xxxxxxxxxxx
  # 钉钉机器人安全设置方式: 加签
  secret: SEC000000000000000000000000
webhook2:
  # 这个没有使用 secret 进行安全认证, 就需要下钉钉机器人那里设置自定义关键词或者IP地址
  (段) 认证方式。
  url: https://oapi.dingtalk.com/robot/send?access_token=xxxxxxxxxxx
webhook_legacy:
  url: https://oapi.dingtalk.com/robot/send?access_token=xxxxxxxxxxx
  # 自定义的模版内容
  message:
    # Use legacy template
    title: '{{ template "legacy.title" . }}'
    text: '{{ template "legacy.content" . }}'
webhook_mention_all:
  url: https://oapi.dingtalk.com/robot/send?access_token=xxxxxxxxxxx
  # @ 所有人
  mention:
    all: true
webhook_mention_users:
  url: https://oapi.dingtalk.com/robot/send?access_token=xxxxxxxxxxx
  # @ '156xxx8827' 和 '189xxx8325'
  mention:
    mobiles: ['156xxx8827', '189xxx8325']
```

secret 是钉钉自定义机器人的加密方式的一种。

具体参考[钉钉官方文档](#)

## 5. 启动服务

```
systemctl enable --now webhook-dingtalk.service
```

```
ss -ntal | grep 8060
```

## 6. 在 alertmanager 中配置

将如下内容添加到 Alertmanager 的配置文件中。

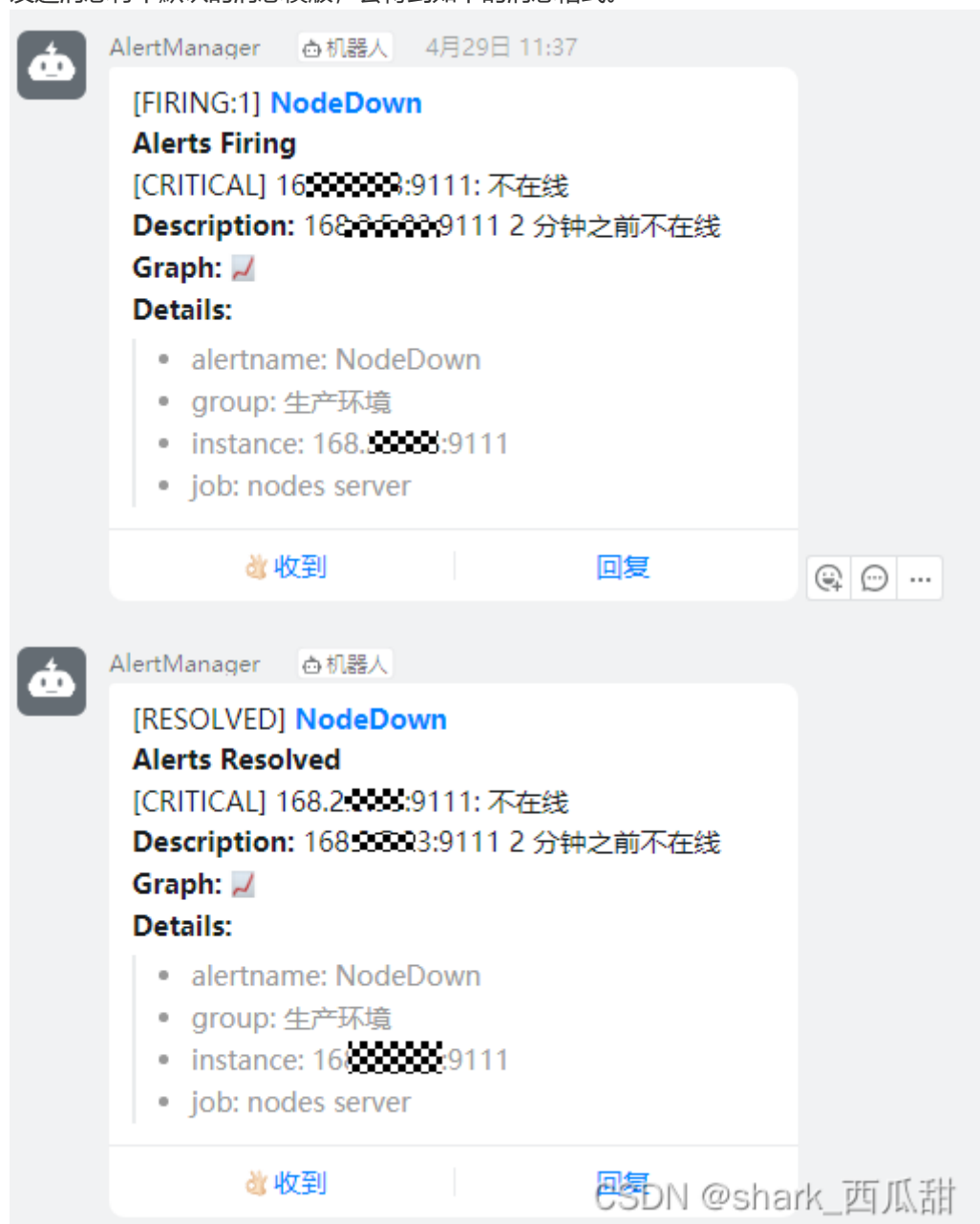
```
receivers:
- name: 'web.hook'
  webhook_configs:
  - url: 'http://钉钉插件的IP:8060/dingtalk/webhook1/send'
```

webhook1 是钉钉插件配置文件中 targets 定义的其中一个。

## 7 测试

尝试在某个被监控的节点, 停止 node-exporter 服务。等接收到告警通知后, 再启动 node-export, 观察是否收到告警恢复的信息。

发送消息有个默认的消息模版，会得到如下的消息格式。



The screenshot displays two Slack messages from a bot named 'AlertManager' (机器人) sent on April 29th at 11:37. The first message is titled '[FIRING:1] NodeDown Alerts Firing' and contains the following information: '[CRITICAL] 168.2.5.3:9111: 不在线', 'Description: 168.2.5.3:9111 2 分钟之前不在线', 'Graph: [line graph icon]', and 'Details: alertname: NodeDown, group: 生产环境, instance: 168.2.5.3:9111, job: nodes server'. The second message is titled '[RESOLVED] NodeDown Alerts Resolved' and contains similar information: '[CRITICAL] 168.2.5.3:9111: 不在线', 'Description: 168.2.5.3:9111 2 分钟之前不在线', 'Graph: [line graph icon]', and 'Details: alertname: NodeDown, group: 生产环境, instance: 168.2.5.3:9111, job: nodes server'. Both messages have '收到' (Received) and '回复' (Reply) buttons at the bottom.

AlertManager 机器人 4月29日 11:37

[FIRING:1] **NodeDown**  
**Alerts Firing**  
[CRITICAL] 168.2.5.3:9111: 不在线  
**Description:** 168.2.5.3:9111 2 分钟之前不在线  
**Graph:** [line graph icon]  
**Details:**

- alertname: NodeDown
- group: 生产环境
- instance: 168.2.5.3:9111
- job: nodes server

收到 | 回复

AlertManager 机器人

[RESOLVED] **NodeDown**  
**Alerts Resolved**  
[CRITICAL] 168.2.5.3:9111: 不在线  
**Description:** 168.2.5.3:9111 2 分钟之前不在线  
**Graph:** [line graph icon]  
**Details:**

- alertname: NodeDown
- group: 生产环境
- instance: 168.2.5.3:9111
- job: nodes server

收到 | 回复

CSDN @shark\_西瓜甜

如何自定义通知模版样式，请看下篇文章。

## 传送门

[上一篇: 9-云原生监控体系-Alertmanager-发送管理警报](#)

[下一篇: 11-云原生监控体系-Alertmanager-通知模版](#)