[Clustering] How to set up a monitoring server

1 Monitoring Server Introduction

Prometheus is an open-source systems monitoring and alerting toolkit. This article will explain how to set up Prometheus, AlertManager, and Grafana for your Gateway.

2 Prometheus configuration

2.1 Download Prometheus

- Go to `https://github.com/prometheus/prometheus/releases/tag/v2.45.2` and download the suitable Prometheus.

2.2 Configure Prometheus

- Create `PrometheusService.xml` at `C:\Program Files\Prometheus`.
- Copy the below information to `prometheus.yml` and enter your actual information, then save the file.

```
<service>
  <id>Prometheus</id>
  <name>Prometheus</name>
  <description>Prometheus</description>
  <workingdirectory>C:\Program Files\Prometheus</workingdirectory>
  <executable>./prometheus.exe</executable>
  <arguments>--config.file=prometheus.yml --storage.tsdb.path=Data</arguments>
  <log mode="roll" />
  <onfailure action="restart" />
</service>
```

- Create `PrometheusService.xml` at `C:\Program Files\Prometheus`.
- Copy the below information to `prometheus.yml` and enter your actual information, then save the file.

```
global:
```

```
  scrape_interval:     15s # By default, scrape targets every 15 seconds.

  # Attach these labels to any time series or alerts when communicating with
  # external systems (federation, remote storage, Alertmanager).
  external_labels:
    monitor: 'codelab-monitor'
# data will keep 3days
alerting:
  alertmanagers:
    - static_configs:
      - targets: ["alert_manager_host:9093"]
rule_files:
- rules.yml
# A scrape configuration containing exactly one endpoint to scrape:
# Here it's Prometheus itself.
scrape_configs:
  # The job name is added as a label `job=<job_name>` to any timeseries scraped from this config.
  - job_name: 'prometheus'
    # Override the global default and scrape targets from this job every 5 seconds.
    scrape_interval: 15s
    static_configs:
      - targets: ['127.0.0.1:9090']
  - job_name: 'node_exporter'
    scrape_interval: 15s
    static_configs:
      - targets: ['node_host1:9182','node_host2:9182',...]
  - job_name: 'postgres-exporter'
    scrape_interval: 15s
    static_configs:
      - targets: ['pg_host:9187']
  - job_name: 'caddy'
    scrape_interval: 15s
    static_configs:
      - targets: ['caddy_host:2019']
  - job_name: 'onpremise'
    scrape_interval: 15s
    static_configs:
      - targets: [host1:9980,host2:9980]
  - job_name: 'relay'
    scrape_interval: 10s
    static_configs:
      - targets: [host1:5681,host2:5681]
  - job_name: 'relay_frontend'
    scrape_interval: 10s
    static_configs:
      - targets: [host1:5682,host2:5682]
```

Tips

- Replace `node_host1,2,3` ... with all the instance IPs on line 27
- Replace `pg_host` with your actual Database Server IP on line 31
- Replace `caddy_host` with your actual LoadBalancer Caddy IP on line 35
- Replace `host1,2` with your actual API Server IP (**Main Node**) on line 39
- Replace `host1,2` with your actual Relay Server (**Main Node** or **Extra Relay**) on line 43
- Replace `host1,2` with your actual Relay Frontend IP (**Main Node**) on line 47

- Create `rules.yml` at `C:\Program Files\Prometheus`.

- Copy the below information to `rules.yml` and enter your actual information, then save the file.

```
groups:
- name: service-rule
  rules:
  - alert: ServiceRule
    expr: up==0
    for: 30s
    labels:
      component: instance
      severity: 'emergency'
      resType: 'Service'
    annotations:
      summary: "service [{{$labels.job}}({{$labels.instance}})] down"
      description: "service [{{$labels.job}}({{$labels.instance}})] down"
- name: upstream-rule
  rules:
  - alert: CaddyUpstreamRule
    expr: caddy_reverse_proxy_upstreams_healthy == 0
    for: 30s
    labels:
      component: upstream
      severity: 'emergency'
      resType: 'upstream'
    annotations:
      summary: "upstream [{{$labels.job}}({{$labels.upstream}})] offline"
      description: "upstream [{{$labels.job}}({{$labels.upstream}})] offline"
```

- Copy `prometheus.exe` to `C:\Program Files\Prometheus`.
- Copy `WinSW.exe` to `C:\Program Files\Prometheus` and change the file name to `PrometheusService.exe`.
- Run Command Prompt as administrator, then run the below script to install Prometheus and start the service.

```
PrometheusService.exe install
PrometheusService.exe start
```

3 Install AlertManager

3.1 Download AlertManager

- Go to `https://github.com/prometheus/alertmanager/releases/tag/v0.26.0` and download the suitable AlertManager.

3.2 Configure AlertManager

- Create `AlertManagerService.xml` at `C:\Program Files\AlertManager`.
- Copy the below information to `AlertManagerService.xml`, then save the file.

```
<service>
  <id>AlertManager</id>
  <name>AlertManager</name>
  <description>AlertManager</description>
  <workingdirectory>C:\Program Files\AlertManager</workingdirectory>
  <executable>.\alertmanager.exe</executable>
  <arguments>--config.file=alertmanager.yml</arguments>
  <log mode="roll" />
  <onfailure action="restart" />
</service>
```

- Create `alertmanager.yml` at `C:\Program Files\AlertManager`.
- Copy the below information to `alertmanager.yml`, then save the file.

```
## Alertmanager config
global:
  resolve_timeout: 5m
  # smtp config
  smtp_from: "from email"
  smtp_smarthost: 'smtp host:port'
  smtp_auth_username: 'auth user'
  smtp_auth_password: 'auth pass'
  smtp_require_tls: true
templates:
  - 'templates/*.tmpl'
route:
  receiver: ops1
  group_wait: 30s
  group_interval: 5m
  repeat_interval: 1h
  group_by: [alertname]
  routes:
      - receiver: 'ops1'
        match:
          component: instance
      - receiver: 'ops2'
        match:
          component: upstream
receivers:
# ops group define
- name: ops1
  email_configs:
  - to: 'email'
    send_resolved: true
    headers: { Subject: " [alert]  {{ .CommonLabels.alertname }} " }
    html: '{{ template "instance.html" . }}'
- name: ops2
  email_configs:
  - to: 'email'
    send_resolved: true
    headers: {Subject: "[alert] {{.CommonLabels.alertname }}"}
    html: '{{ template "upstream.html" . }}'
```

Tips

- Replace `from email` with the sender email address on line 5
- Replace `smtp host` with the SMTP server address on line 6
- Replace `port` with the SMTP server port on line 6 (for example, 465, 587...)
- Replace `auth user` with the authentication username on line 7
- Replace `auth pass` with the authentication password on line 8
- If your SMTP server do NOT support TLS, please set `smtp_require_tls` to `false` on line 9
- Replace `email` with the alert email reception addresses on line 29 & 35

3.3 Create Template for AlertManager

- We provide two templates for your reference. You can follow the steps to add these templates to the Alertmanager.
- Create `instance.tmpl` at `C:\Program Files\AlertManager\templates`.

- Copy the below information to `instance.tmpl`, then save the file.

```
{{ define "instance.html" }}
{{- if gt (len .Alerts.Firing) 0 -}}
<h2>@Alerting</h2>
{{- range $index, $alert := .Alerts.Firing -}}
   <div>
    Severity:       {{ $alert.Labels.severity }} <br>
    AlertName:      {{ $alert.Labels.alertname }} <br>
    Instance:       {{ $alert.Labels.instance }} <br>
    Summary:        {{ $alert.Annotations.summary }} <br>
    Description:    {{ $alert.Annotations.description }} <br>
    StartsAt:       {{ $alert.StartsAt.Local.Format "2006-01-02 15:04:05" }} <br>
   </div>
   <br>
{{- end }}
{{- end }}

{{- if gt (len .Alerts.Resolved) 0 -}}
<h2>@Resolved</h2>
{{- range $index, $alert := .Alerts.Resolved -}}
   <div>
    Instance:       {{ $alert.Labels.instance }} <br>
    Summary:        {{ $alert.Annotations.summary }} <br>
    Description:    {{ $alert.Annotations.description }} <br>
    StartsAt:       {{ $alert.StartsAt.Local.Format "2006-01-02 15:04:05" }} <br>
    ResolveAt:      {{ $alert.EndsAt.Local.Format "2022-03-19 15:04:05" }} <br>
   </div>
   <br>
{{- end }}
{{- end }}
{{- end }}
```

- Create `upstream.tmpl` at `C:\Program Files\AlertManager\templates`.
- Copy the below information to `upstream.tmpl`, then save the file.

```
{{ define "upstream.html" }}
{{- if gt (len .Alerts.Firing) 0 -}}
<h2>@Alerting</h2>
{{- range $index, $alert := .Alerts.Firing -}}
   <div>
      Severity:       {{ $alert.Labels.severity }} <br>
      AlertName:      {{ $alert.Labels.alertname }} <br>
      Upstream:       {{ $alert.Labels.upstream }} <br>
      Summary:        {{ $alert.Annotations.summary }} <br>
      Description:    {{ $alert.Annotations.description }} <br>
      StartsAt:       {{ $alert.StartsAt.Local.Format "2006-01-02 15:04:05" }} <br>
   </div>
   <br>
{{- end }}
{{- end }}

{{- if gt (len .Alerts.Resolved) 0 -}}
<h2>@Resolved</h2>
{{- range $index, $alert := .Alerts.Resolved -}}
   <div>
      Upstream:       {{ $alert.Labels.upstream }} <br>
      Summary:        {{ $alert.Annotations.summary }} <br>
```

```
      Description:    {{ $alert.Annotations.description }} <br>
      StartsAt:       {{ $alert.StartsAt.Local.Format "2006-01-02 15:04:05" }} <br>
      ResolveAt:      {{ $alert.EndsAt.Local.Format "2006-01-02 15:04:05" }} <br>
   </div>
   <br>
{{- end }}
{{- end }}
{{- end }}
```

- Copy `alertmanager.exe` to `C:\Program Files\AlertManager`.
- Copy `WinSW.exe` to `C:\Program Files\AlertManager` and change the file name to `AlertManagerService.exe`.
- Run Command Prompt as administrator, then run the below script to install Prometheus and start the service.

```
AlertManagerService.exe install
AlertManagerService.exe start
```

4 Install Grafana

- Go to `https://dl.grafana.com/enterprise/release/grafana-enterprise-10.2.3.windows-amd64.msi` and download Grafana.

5 Access and Configuration on Monitoring Server

5.1 Prometheus

- Go to http://monitoring_ip:9090 to check the target status.

5.2 Alert Manager

- Go to http://monitoring_ip:9093 to check alertmanager.

5.3 Grafana

- Go to http://monitoring_ip:3000 and log in with username (admin) and password (admin).

5.3.1 Add data source

- Go to http://monitoring_ip:3000/datasources.
- Select 'Prometheus' as the data source.
- Fill in the **Prometheus server URL.**
- Save the settings.

5.3.2 Add dashboard template

- Go to http://monitoring_ip:3000/dashboards.
- Click New then select Import.
- Import via grafana.com
  - 14694 (windows_exporter)
  - 12485 (postgres_exporter)
- Select 'Prometheus' as the data source.
- Import the json.

Relay-1705027142557.json

Onpremise-1705027111214.json

Service Status-1692600935491.json

Caddy Upstream-1692599975490.json

Relay Frontend-1703745236774.json