部署 Prometheus 基础环境

本文是在学习和配置 Prometheus 基础环境时的一些记录，包括 Prometheus, exporter,alertmanager,Grafana 和 Docker 部署。希望大家多多交流，分享您的经验，共同学习，一起进步。

如果有错误或疑问请告诉我，谢谢！ Email:265583#qq.com

目录

Table of Contents

Centos7 部署 Prometheu+Alertmanager+Grafana 监控系统

安装 Centos 7 系统

安装 Prometheus

```
# wget
https://github.com/prometheus/prometheus/releases/download/v1.7.1/prometheus-1.7.1.linux-amd64.tar.gz
# tar xvf prometheus-1.7.1.linux-amd64.tar.gz
# mv prometheus-1.7.1.linux-amd64 prometheus
# cp prometheus.yml prometheus.yml_bk
```

启动 Prometheus
nohup ./prometheus -config.file=prometheus.yml &
[1] 11625
可直接加载Prometheus配置而不停止服务方式让配置生效，在调试过程中，每次修改配置后执行该操作让配置生效更方便：
curl -X POST http://localhost:9090/-/reload
netstat -antl|grep 9090
tcp6 0 0 :::9090 :::* LISTEN
tcp6 0 0 ::1:9090 ::1:39254 TIME_WAIT
#安装完成后通过浏览器访问9090端口: http://192.168.100.22:9090/targets，至此，Prometheus安装完成。
为了启动方便，可将Prometheus服务设置为系统服务
vim /etc/systemd/system/Prometheus.service
[Unit]
Description=Prometheus Services.
Documentation=https://github.com/prometheus/prometheus
After=alertmanager.service

[Service]
EnvironmentFile=-/etc/alertmanager/template
User=root
ExecStart=/opt/prometheus/prometheus \
-config.file=prometheus.yml \
-storage.local.retention=4320h \
-alertmanager.url http://localhost:9093

Restart=on-failure
[Install]
WantedBy=multi-user.target
# systemctl enable Prometheus.service
# systemctl restart Prometheus.service

安装 Alertmanager

Alertmanager 是一个告警系统，通过设置规则，可以实现告警通知。

相关文档：
Prometheus Alertmanager报警组件
http://www.jianshu.com/p/239b145e2acc
https://github.com/prometheus/alertmanager

Prometheus监控 - Alertmanager报警模块
https://sagittariusyx.github.io/2016/03/07/prometheus-alertmanager/
https://github.com/prometheus/alertmanager

Alert template:

https://prometheus.io/blog/2016/03/03/custom-alertmanager-templates/

Sending alert notifications to multiple destinations
https://www.robustperception.io/sending-alert-notifications-to-multiple-destinations/

Alert tree:
https://prometheus.io/webtools/alerting/routing-tree-editor/

安装：

下载源文件并解压，编译，直接执行：
tar xvf alertmanager-0.7.1.linux-amd64.tar.gz
mv alertmanager-0.7.1.linux-amd64.tar.gz alertmanager
cd alertmanager
nohup ./alertmanager -config.file=/opt/alertmanager -config.file=simple.yml &

重启prometheus 服务：
./prometheus -config.file=prometheus.yml -alertmanager.url http://localhost:9093

也可以通过加载配置文件方式而不重启Alertmanager服务：
curl -X POST http://localhost:9093/-/reload

设置Alertmanager 系统服务
vim /etc/systemd/system/alertmanager.service
[Unit]
Description=Prometheus Alertmanager.
Documentation=https://github.com/prometheus/alertmanager
After=network.target

[Service]
EnvironmentFile=-/etc/alertmanager/template
User=root
ExecStart=/opt/alertmanager/alertmanager \
        -config.file=/opt/alertmanager/simple.yml \
        -storage.path=/var/lib/prometheus/alertmanager \
        $ALERTMANAGER_OPTS
ExecReload=/bin/kill -HUP $MAINPID
Restart=on-failure

[Install]
WantedBy=multi-user.target

systemctl enable alertmanager.service
systemctl restrart alertmanager.service

访问Alertmanager页面：http://192.168.100.22:9093/#/alerts

安装 Grafana

文档：
Configuration:
http://docs.grafana.org/installation/configuration/
https://prometheus.io/docs/visualization/grafana/#installing

Setting up Grafana for Prometheus
https://www.robustperception.io/setting-up-grafana-for-prometheus/

Sending alert notifications to multiple destinations
https://www.robustperception.io/sending-alert-notifications-to-multiple-destinations/
https://prometheus.io/docs/visualization/grafana/

安装
wget https://s3-us-west-2.amazonaws.com/grafana-releases/release/grafana-4.4.1-1.x86_64.rpm
yum localinstall grafana-4.4.1-1.x86_64.rpm

启动Grafana
/etc/init.d/grafana-server start
Starting grafana-server (via systemctl): [OK]

netstat -anp|grep 3000
tcp6 0 0 :::3000 :::* LISTEN 5337/grafana-server

# 将Grafana设置为系统服务
# mkdir -p /var/run/grafana
# chown grafana.grafana /var/run/grafana
# vim /etc/sysconfig/grafana-server, 添加:
PID_FILE_DIR=/var/run/grafan

vim /etc/systemd/system/grafana.service
[Unit]
Description=Grafana Services
Documentation=https://github.com/grafana/grafana
After=network.target
[Service]
EnvironmentFile=/etc/sysconfig/grafana-server
User=grafana
Group=grafana

```
Type=simple
WorkingDirectory=/usr/share/grafana
RuntimeDirectory=grafana
RuntimeDirectoryMode=0750
ExecStart=/usr/sbin/grafana-server                                  \
                    --config=${CONF_FILE}                    \
                    --pidfile=${PID_FILE_DIR}/grafana-server.pid  \
                    cfg:default.paths.logs=${LOG_DIR}          \
                    cfg:default.paths.data=${DATA_DIR}          \
                    cfg:default.paths.plugins=${PLUGINS_DIR}
LimitNOFILE=10000
TimeoutStopSec=20
UMask=0027

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

以上配置文件中的变量${CONF_FILE} 读取的是/etc/sysconfig/grafana-server中的内容

#配置文件变更后必须先reload

```
# systemctl daemon-reload
# systemctl restart grafana.service
# systemctl enable grafana.service
```

安装完毕后 通过默认管理员账号/密码:admin/admin访问Grafana:
http://192.168.100.22:3000/login

在Grafana里添加Prometheus数据源

1).Go to http://localhost:3000
2).Enter the username admin and password admin, and then click “Log In”.
3).Click “Data Sources” on the left menu
4).Click “Add new” on the top menu
5).Add a default data source of type Prometheus with http://localhost:9090 as the URL
6).Click "Add"

Add Dashboard

配置 Alertmanager

报警分两部分，报警条件规则文件默认放在Prometheus安装目录下，文件名为alert.rules。具体通知内容，例如邮件地址和通知人员设置在Alertmanager安装目录下

的simply.yml文件，以下是一些基础和常用配置，阈值和时间根据自己需求进行修改。

```
# alert.rules:

ALERT node_down
  IF up == 0 AND job="node"
  FOR 5m
  ANNOTATIONS {
    summary = "Node is down",
    description = "Node has been unreachable for more than 5 minutes.",
    severity = "warning"
  }

ALERT snmp_down
  IF up == 0 AND job="snmp"
  FOR 5m
  ANNOTATIONS {
    summary = "SNMP is down",
    description = "SNMP has been unreachable for more than 5 minutes.",
    severity = "warning"
  }

ALERT fs_at_80_percent
  IF hrStorageUsed{hrStorageDescr=~"/.+"} / hrStorageSize >= 0.8
  FOR 15m
  ANNOTATIONS {
    summary = "File system {{$labels.hrStorageDescr}} is at 80%",
    description = "{{$labels.hrStorageDescr}} has been at 80% for more than 15 Minutes.",
    severity = "warning"
  }

ALERT fs_at_90_percent
  IF hrStorageUsed{hrStorageDescr=~"/.+"} / hrStorageSize >= 0.9
  FOR 15m
  ANNOTATIONS {
    summary = "File system {{$labels.hrStorageDescr}} is at 90%",
    description = "{{$labels.hrStorageDescr}} has been at 90% for more than 15 Minutes.",
    severity = "average"
  }

ALERT disk_load_mostly_random_reads
  IF rate(diskIOReads{diskIODevice=~"sd[a-z]+"}[5m]) > 20 AND
     rate(diskIONReadX{diskIODevice=~"sd[a-z]+"}[5m]) /
rate(diskIOReads{diskIODevice=~"sd[a-z]+"}[5m]) < 10000
  FOR 15m
  ANNOTATIONS {
    summary = "Disk {{$labels.diskIODevice}} reads are mostly random.",
```

```
  description = "{{$labels.diskIODevice}} reads have been mostly random for the past 15
Minutes.",
  severity = "info"
 }

ALERT disk_load_mostly_random_writes
 IF rate(diskIOWrites{diskIODevice=~"sd[a-z]+"}[5m]) > 20 AND
   rate(diskIONWrittenX{diskIODevice=~"sd[a-z]+"}[5m]) /
rate(diskIOWrites{diskIODevice=~"sd[a-z]+"}[5m]) < 10000
 FOR 15m
 ANNOTATIONS {
  summary = "Disk {{$labels.diskIODevice}} writes are mostly random.",
  description = "{{$labels.diskIODevice}} writes have been mostly random for the past 15
Minutes.",
  severity = "info"
 }

ALERT disk_load_high
 IF diskIOLA1{diskIODevice=~"s|vd[a-z]+"} > 30
 FOR 15m
 ANNOTATIONS {
  summary = "Disk {{$labels.diskIODevice}} is at 30%",
  description = "{{$labels.diskIODevice}} Load has exceeded 30% over the past 15
Minutes.",
  severity = "warning"
 }

ALERT cpu_load_high
 IF ssCpuIdle < 70
 FOR 15m
 ANNOTATIONS {
  summary = "CPU is at 30%",
  description = "CPU Load has constantly exceeded 30% over the past 15 Minutes.",
  severity = "warning"
 }

ALERT linux_load_high
 IF laLoad1 > 50
 FOR 15m
 ANNOTATIONS {
  summary = "Linux Load is at 40",
  description = "Linux Load has constantly exceeded 40 over the past 15 Minutes.",
  severity = "average"
 }

ALERT if_operstatus_changed
 IF delta(ifOperStatus[15m]) != 0
```

```
  ANNOTATIONS {
    summary = "Port {{$labels.ifDescr}} changed status",
    description = "Port {{$labels.ifDescr}} went up or down in the past 15 Minutes",
    severity = "info"
  }

ALERT if_traffic_at_30_percent
  IF ifSpeed > 10000000 AND
     ifOperStatus == 1 AND
     rate(ifInOctets[5m]) > ifSpeed * 0.3
  FOR 15m
  ANNOTATIONS {
    summary = "Port {{$labels.ifDescr}} is at 30%",
    description = "Port {{$labels.ifDescr}} has had at least 30% traffic over the past 15
Minutes.",
    severity = "warning"
  }

ALERT if_traffic_at_70_percent
  IF ifSpeed > 10000000 AND
     ifOperStatus == 1 AND
     rate(ifInOctets[5m]) > ifSpeed * 0.7
  FOR 15m
  ANNOTATIONS {
    summary = "Port {{$labels.ifDescr}} is at 70%",
    description = "Port {{$labels.ifDescr}} has had at least 70% traffic over the past 15
Minutes.",
    severity = "average"
  }

# simply.yml

# 主要分三部分, Global部分设置发送邮件服务器信息，route设置规则和报警时间间隔等，receivers设置接收人。

global:
  # 设置发送邮件的地址和smtp信息
  smtp_smarthost: 'smtp.abc.com'
  smtp_from: 'prometheus@abc.com'
  smtp_auth_username: 'prometheus'
  smtp_auth_password: 'abcd’

route:
  receiver: 'team-X-mails'
  group_by: ['alertname']
  group_wait: 30s
  group_interval: 5m
```

```
  repeat_interval: 6h

inhibit_rules:
- source_match:
    severity: 'critical'
  target_match:
    severity: 'warning'
  # Apply inhibition if the alertname is the same.
  equal: ['alertname']

receivers:
- name: 'team-X-mails'
  email_configs:
  - to: 'support@abc.com'
     send_resolved: true

# 设置完毕后需要重新加载配置文件
```

配置 Prometheus 常用 exporter

node_export 安装

可采用直接下载安装和Docker容器方式安装两种

直接下载安装

```
#wget
https://github.com/prometheus/node_exporter/releases/download/v0.14.0/node_exporter-0.14.0.linux-amd64.tar.gz
# tar zxvf node_exporter-0.14.0.linux-amd64.tar.gz
# cd node_exporter-0.14.0.linux-amd64/
```

Docker方式安装

```
# docker pull prom/node-exporter
```

运行：

```
# docker run -d \
--net=host \
--restart=always \
--name node-exporter \
-p 9100:9100 \
-v "/proc:/host/proc" \
-v "/sys:/host/sys" \
-v "/:/rootfs" \
prom/node-exporter \
-collector.procfs /host/proc \
-collector.sysfs /host/sys \
-collector.filesystem.ignored-mount-points "^/(sys|proc|dev|host|etc)($|/)"
```

```
# docker ps -a
CONTAINER
ID      IMAGE                    COMMAND             CREATED         STATUS          PORTS
        NAMES
d5e20da8e3bd      hub.allyamall.com/node-exporter   "/bin/node_exporte..."   2 seconds
ago      Up 2 seconds                    node-exporter
[root@yiche-03 ~]# netstat -anp|grep 9100
tcp6      0     0 :::9100             :::*                LISTEN     6200/node_exporter
tcp6      0     0 192.168.100.29:9100    192.168.100.16:55336    TIME_WAIT   -
```

没有配置--net=host：

```
# netstat -anp|grep 9100
tcp6      0     0 :::9100             :::*                LISTEN     7687/docker-proxy
```

如果用docker方式安装，但没设置—net=host，在Prometheus/Grafana里将看不到网卡流量，netstat内容.

```
# netstat -anp|grep 9100
tcp6      0     0 :::9100             :::*                LISTEN     1470/node_exporter
```

设置node_exporter为系统服务

```
# vim /etc/systemd/system/node_export.service
[Unit]
Description=Prometheus NODE Exporter

[Service]
WorkingDirectory=/opt/projects/src/src/github.com/prometheus/node_exporter/
ExecStart=/usr/sbin/node_exporter $OPTIONS

[Install]
WantedBy=multi-user.target
# systemctl enable node_export.service
# systemctl restart node_export.service
```

#配置prometheus.yml，对应的node_exporter 端口为9100,例如:

```
#Node exporter
  - job_name: 'node'
    static_configs:
      - targets: ['127.0.0.1:9100']
```

snmp_exporter 安装, 通过 SNMP 监控网络设备

snmp相关的配置文件为/snmp_exporter/snmp.yml，可以通过设置oid方式具体监控网络设备，也可以直接用默认文件，监控网络设备流量。以下以监控h3c路由器为例。

Documents:
https://www.robustperception.io/snmp-monitoring-with-prometheus/
https://github.com/prometheus/snmp_exporter

SNMP Monitoring with Prometheus
https://www.robustperception.io/snmp-monitoring-with-prometheus/

基于Prometheus的分布式在线服务监控实践
https://zhuanlan.zhihu.com/p/24811652

安装：
wget
https://github.com/prometheus/snmp_exporter/releases/download/v0.4.0/snmp_exporter-0.4.0.linux-amd64.tar.gz
tar xzf snmp_exporter-0.4.0.linux-amd64.tar.gz
cd snmp_exporter-0.4.0.linux-amd64
#nohup ./snmp_exporter &
[1] 5201

netstat -anp|grep 9116
```
tcp6       0      0 :::9116                 :::*                    LISTEN      5201/./snmp_exporte
tcp6       0      0 127.0.0.1:9116          127.0.0.1:59030         TIME_WAIT   -
tcp6       0      0 127.0.0.1:9116          127.0.0.1:59026         TIME_WAIT   -
```

访问http://192.168.100.22:9116/，输入被监控的SNMP主机后可查看该主机信息。

在Prometheus.yml里配置snmp信息
vim /opt/promethens/peometheus.yml
For SNMP equipment

```
  - job_name: 'h3c'
    static_configs:
     - targets:
       - 192.168.100.1
    metrics_path: /snmp
    params:
      module: [default]
    relabel_configs:
      - source_labels: [__address__]
        target_label: __param_target
      - source_labels: [__param_target]
        target_label: instance
      - target_label: __address__
        replacement: 127.0.0.1:9116 #SNMP exporter
```

添加snmp_exporter为系统服务

```
# vim /etc/systemd/system/snmp_exporter.service
[Unit]
Description=Prometheus SNMP Exporter
After=network.target

[Service]
User=root
Group=root
WorkingDirectory=/opt/snmp_exporter/
ExecStart=/opt/snmp_exporter/snmp_exporter
Type=simple
[Install]
WantedBy=multi-user.target

# systemctl enable snmp_exporter.service
# systemctl restart snmp_exporter.service
```

BlackBox exporter 安装，通过 Ping,http,dns 进行监控

```
# 安装Blackbox exporter, 源码安装编译的时候可能会提示go路径的问题，所以选择直接安装。
# go get github.com/prometheus/blackbox_exporter
# go build github.com/prometheus/blackbox_exporter

#启动进程
# cd /opt/projects/bin
# ./blackbox_exporter -
config.file=/opt/projects/src/src/github.com/prometheus/blackbox_exporter/blackbox.yml
&
[1] 14756
INFO[0000] Starting blackbox_exporter (version=, branch=,
revision=)  source="main.go:153"
INFO[0000] Build context (go=go1.7, user=, date=)       source="main.go:154"
INFO[0000] Loaded config file                           source="main.go:71"
INFO[0000] Listening on :9115                           source="main.go:226"

# netstat -antp|grep 9115
tcp6       0      0 :::9115                 :::*                    LISTEN      14756/./blackbox_ex

# 浏览器访问：http://192.168.100.22:9115/
# 配置Prometheus.yml，通过ping监控网络设备
#Ping AP

  - job_name: 'ping'
```

```
    scrape_interval: 5s
    metrics_path: /probe
    params:
      module: [icmp] #ping
    static_configs:
    - targets: ['192.168.100.1', '192.168.100.11', '192.168.100.21', '192.168.100.31']
      labels:
        groups: 'H3C'
```

修改配置后需要重新加载配置文件

通过容器部署 Prometheus+Grafana+cAdvisor

目的：通过容器方式部署Prometheus+Grafana+cAdvisor,通过轻量式快速部署监控环境

Documents:

1). https://github.com/stefanprodan/dockprom
2). Prometheus/Docker/cAdvisor/Grafana集成监控
https://github.com/stefanprodan/dockprom
3). Grafana 高可用：（Grafana默认使用Sqlite3作为数据库，轻量级，但不支持分布式，如果要做HA需要用Mysql）
http://docs.grafana.org/tutorials/ha_setup/#how-to-setup-grafana-for-high-availability
4).Docker 集群监控平台---cAdvisor-InfluxDB-Grafana
https://jevic.github.io/2017/01/23/dockercAdvisordbgra-md/
http://www.2cto.com/net/201701/583599.html
5). 使用InfluxDB+cAdvisor+Grafana配置Docker监控
http://www.jianshu.com/p/d078d353d12f
6).grafana使用mysql存储
https://segmentfault.com/a/1190000008936411

步骤：

整个安装组件Prometheus,Grafana,Alertmanager, node_exporter,cAdvisor,全部为容器方式。

1). 从https://github.com/stefanprodan/dockprom 下载所有代码：

```
$ git clone https://github.com/stefanprodan/dockprom
$ cd dockprom
$ docker-compose up -d
```

docker-compose 命令是用来一次性启动管理多个容器的命令，对应的配置文件是

docker-compose.yml.
如果要单独启动某个容器，例如cAdvisor，可以执行：
```
# sudo docker run \
  --volume=/:/rootfs:ro \
  --volume=/var/run:/var/run:rw \
  --volume=/sys:/sys:ro \
  --volume=/var/lib/docker/:/var/lib/docker:ro \
  --publish=8080:8080 \
  --detach=true \
  --name=cadvisor \
  google/cadvisor:latest
```

安装完成后直接访问相应的网页和服务即可。

如果单独安装node-exporter,方法如下：
```
# docker pull prom/node-exporter
docker run -d \
--restart=always \
--name node-exporter \
-p 9100:9100 \
-v "/proc:/host/proc" \
-v "/sys:/host/sys" \
-v "/:/rootfs" \
prom/node-exporter \
-collector.procfs /host/proc \
-collector.sysfs /host/sys \
-collector.filesystem.ignored-mount-points "^/(sys|proc|dev|host|etc)($|/)"
```

#加--restart=always的目的是为了docker系统服务重启的时候容器可以自动重启。

Prometheus
http://192.168.100.111:9090

Grafana
http://192.168.100.111:3000
#默认用户名和密码：admin/changeme，添加默认数据库：
* Name: Prometheus
* Type: Prometheus
* Url: http://prometheus:9090
* Access: proxy

如果需要修改配置文件直接到dockprom 目录下进行修改即可，修改后重新加载配置生效：
```
# curl -X POST http://192.168.100.111:9090/-/reload
```

监控cAdvisor报警条件：

vim containers.rules

```
ALERT cAdvisor_down
  IF absent(container_memory_usage_bytes{name="cadvisor"})
  FOR 1m
  LABELS { severity = "critical" }
  ANNOTATIONS {
    summary= "cAdvisor containers down",
    description= "cAdvisor container is down for more than 1 minutes."
  }

ALERT cAdvisor_high_cpu
  IF sum(rate(container_cpu_usage_seconds_total{name="cadvisor"}[1m])) /
count(node_cpu{mode="system"}) * 100 > 10
  FOR 5m
  LABELS { severity = "warning" }
  ANNOTATIONS {
    summary= "cAdvisor high CPU usage",
    description= "cAdvisor CPU usage is {{ humanize $value}}%."
  }

ALERT cAdvisor_high_memory
  IF sum(container_memory_usage_bytes{name="cadvisor"}) > 1200000000
  FOR 5m
  LABELS { severity = "warning" }
  ANNOTATIONS {
     summary = "cAdvisor high memory usage",
     description = "cAdvisor memory consumption is at {{ humanize $value}}.",
  }
```

其它参考：

1). 容器部署完成后可以直接通过docker export导出为tar文件再导入到其它服务器。

导出：

```
$ docker ps -a
CONTAINER
ID      IMAGE                COMMAND             CREATED          STATUS          PORTS
      NAMES
4da6c4e8d580      prom/prometheus         "/bin/prometheus -..."   13 hours ago      Up 13
hours       0.0.0.0:9090->9090/tcp   prometheus
2cb99ce6b87f      prom/node-exporter      "/bin/node_exporte..."   13 hours ago      Up 13
hours       9100/tcp                 nodeexporter
40c02b10c25d      grafana/grafana         "/run.sh"           13 hours ago      Up 13
```

hours 0.0.0.0:3000->3000/tcp grafana
b2313d0b3247 prom/alertmanager "/bin/alertmanager..." 13 hours ago Up 13 hours 0.0.0.0:9093->9093/tcp alertmanager
6a4da9de8ebc google/cadvisor:v0.26.1 "/usr/bin/cadvisor..." 13 hours ago Up 13 hours 8080/tcp cadvisor
$ docker export 6a4da9de8ebc > cadvisor.tar
如果是导出镜像可以使用docker save,导入则是docker load.

导入：

cat cadvisor.tar | docker import - google/cadvisor:v0.26.1
sha256:cd45ecb65fe4ac1ebdae18baab23529c6597230374eccef27f165c6859f35167

docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
google/cadvisor v0.26.1 cd45ecb65fe4 18 seconds ago 58.1MB

2). Grafana 数据库存放路径：默认Grafana采用Sqlite3，轻量型数据库，但不支持分布式，如果需要HA可以使用Mysql替代。

#cd /var/lib/grafana
ll
total 1396
-rw-r--r-- 1 grafana grafana 1422336 Aug 15 15:03 grafana.db
drwxr-xr-x 6 grafana grafana 123 Jul 24 10:16 plugins
drwx------ 13 grafana grafana 105 Aug 14 08:31 sessions
[root@office-monitoring grafana]# du -sh *
1.4M grafana.db
16M plugins
24K sessions