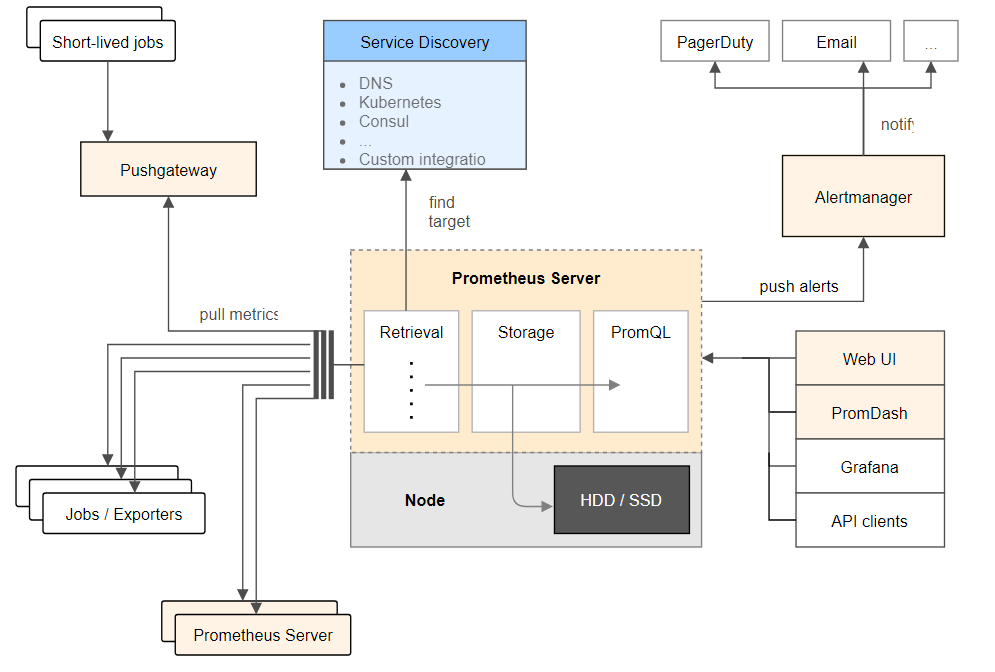
**prometheus数据采集agent编译部署**

**一、Prometheus部署架构图：**



**1、pull方式**

Prometheus采集数据是用的pull也就是拉模型,通过HTTP协议去采集指标，只要应用系统能够提供HTTP接口就可以接入监控系统，相比于私有协议或二进制协议来说开发、简单。

**2、push方式**

对于定时任务这种短周期的指标采集，如果采用pull模式，可能造成任务结束了，Prometheus还没有来得及采集，这个时候可以使用加一个中转层，客户端推数据到Push Gateway缓存一下，由Prometheus从push gateway pull指标过来。(需要额外搭建Push Gateway，同时需要新增job去从gateway采数据)

**3、安装prometheus**

cd /usr/local/src/

wget https://github.com/prometheus/prometheus/releases/download/v1.7.1/prometheus-1.7.1.linux-amd64.tar.gz

tar -zxf prometheus-1.7.1.linux-amd64.tar.gz

mv prometheus-1.7.1.linux-amd64 /data/app/

ln -s /data/app/prometheus-1.7.1.linux-amd64/ /data/app/prometheus

**4、启动prometheus**

./prometheus -config.file=prometheus.yml

1. **prometheus数据库指标采集agent**

**Msyql、redis、mongodb、postgresql采集数据库指标需要数据库用户及登录密码**

1. **编译环境go环境准备**

下载go(我的当前目录是/data/work)

$wget <https://studygolang.com/dl/golang/go1.10.1.linux-amd64.tar.gz>

$tar -xvf go1.10.1.linux-amd64.tar.gz

设置环境变量

$vim /etc/profile

添加

export GOROOT=/data/work/go

export GOPATH=/data/work/gopath

export PATH=$PATH:$GOROOT/bin:$GOPATH/bin

export GO111MODULE=on

export GOPROXY=https://goproxy.io

1. **编译及运行mysql\_expoerter**

（1）git clone git@github.com:percona/mysqld\_exporter.git

进入仓库目录

go build mysqld\_exporter.go

即可在当前路径生成mysqld\_exporter二进制文件了

（2）连接需监控数据库创建监控用户：

CREATE USER 'exporter'@'localhost' IDENTIFIED BY 'XXXXXXXX' WITH MAX\_USER\_CONNECTIONS 10;

GRANT PROCESS, REPLICATION CLIENT, SELECT ON \*.\* TO 'exporter'@'localhost';

新建agent读取密码文件

*$vim .my.cnf*   
[client]  
user=exporter  
password=Cds20180808!

host=127.0.0.1

port=3306

./mysql\_exporter -h查看帮助

启动agent ./mysqld\_exporter --config.my-cnf="/root/.my.cnf" --web.listen-address=0.0.0.0:9100

第二种启动方式

export DATA\_SOURCE\_NAME='root:root1@(localhost:3306)/'

./mysqld\_exporter --web.listen-address=0.0.0.0:9100

1. **编译及运行mongodb\_expoerter**

（1）git clone git@github.com:percona/mongodb\_exporter.git

进入仓库目录

go build mongdb\_exporter.go

即可在当前路径生成mongdb\_exporter二进制文件了

（2）连接需监控mongodb数据库，创建监控用户

db.getSiblingDB("admin").createUser({

user: "mongodb\_exporter",

pwd: "s3cr3tpassw0rd",

roles: [

{ role: "clusterMonitor", db: "admin" },

{ role: "read", db: "local" }

]

})

./mongdb\_exporter -h查看帮助

启动agent

./mongodb\_exporter --mongodb.uri=mongodb://admin:admin@127.0.0.1:27017/admin

--web.listen-address=0.0.0.0:9100

1. **编译及运行redis\_expoerter**

（1）git clone git@github.com:oliver006/redis\_exporter.git

进入仓库目录

go build redis\_exporter.go

即可在当前路径生成redis\_exporter二进制文件了

（2）./redis\_exporter -h查看帮助

启动agent ./redis\_exporter --web.listen-address=0.0.0.0:9100 --redis.addr=127.0.0.1:6379 --redis.password=111111

1. **编译及运行postgres\_expoerter**

（1）git clone git@github.com:wrouesnel/postgres\_exporter.git

进入仓库目录

go build postgres\_exporter.go

即可在当前路径生成postgres\_exporter二进制文件了

（2）

连接需监控pg数据库创建监控用户

CREATE USER postgres\_exporter PASSWORD 'password';ALTER USER postgres\_exporter SET SEARCH\_PATH TO postgres\_exporter,pg\_catalog;

-- If deploying as non-superuser (for example in AWS RDS), uncomment the GRANT-- line below and replace <MASTER\_USER> with your root user.-- GRANT postgres\_exporter TO <MASTER\_USER>;CREATE SCHEMA IF NOT EXISTS postgres\_exporter;GRANT USAGE ON SCHEMA postgres\_exporter TO postgres\_exporter;

CREATE FUNCTION get\_pg\_stat\_activity() RETURNS SETOF pg\_stat\_activity AS

$$ SELECT \* FROM pg\_catalog.pg\_stat\_activity; $$

LANGUAGE sql

VOLATILE

SECURITY DEFINER;

CREATE OR REPLACE VIEW postgres\_exporter.pg\_stat\_activityAS

SELECT \* from get\_pg\_stat\_activity();

GRANT SELECT ON postgres\_exporter.pg\_stat\_activity TO postgres\_exporter;

CREATE OR REPLACE FUNCTION get\_pg\_stat\_replication() RETURNS SETOF pg\_stat\_replication AS

$$ SELECT \* FROM pg\_catalog.pg\_stat\_replication; $$

LANGUAGE sql

VOLATILE

SECURITY DEFINER;

CREATE OR REPLACE VIEW postgres\_exporter.pg\_stat\_replicationAS

SELECT \* FROM get\_pg\_stat\_replication();

GRANT SELECT ON postgres\_exporter.pg\_stat\_replication TO postgres\_exporter;

./postgres\_exporter --help查看帮助

启动agent

export DATA\_SOURCE\_NAME=postgresql://postgres:111111@127.0.0.1:5432/postgres?sslmode=disable

./postgres\_exporter --web.listen-address=0.0.0.0:9100

1. **prometheus端配置抓取agent地址及端口**

在prometheus项目路径中找到prometheus.yaml，在scrape\_configs中添加job\_name配置pull的ip和端口

scrape\_configs:

# The job name is added as a label `job=<job\_name>` to any timeseries scraped from this config.

- job\_name: 'prometheus'

# metrics\_path defaults to '/metrics'

# scheme defaults to 'http'.

static\_configs:

- targets: ['localhost:9090']

- job\_name: mysql

static\_configs:

- targets: ['localhost:9104']

labels:

instance: db1

- job\_name: mongodb

static\_configs:

- targets: ['192.168.43.161:9001']

labels:

instance: db2