实验09-Kubernetes Monitor

一、实验目的

1、了解 Prometheus 和 HertzBeat;

- 2、掌握 Prometheus 的部署;
- 3、实现 Prometheus 监控 K8s 集群;
- 4、实现 HertzBeat 监控 K8s 集群。

二、实验学时

2 学时

三、实验类型

设计性

四、实验任务

1、完成 Prometheus 和 Grafana 的部署;

2、完成 Prometheus 监控 K8s 主机;

- 3、完成 Prometheus 监控 K8s 集群;
- 4、完成 HertzBeat 监控 K8s 集群;
- 5、完成监控数据可视化。

五、实验环境

1、硬件

本实验基于实验教学中心网络运维实验室服务器集群开展,每个实验小组分配集群中的1台物 理服务器作为实验基础平台,提供云计算资源。每个人配备计算机1台。(学生可根据自身情况 使用个人计算机)。

2、软件

Windows 操作系统,或 MacOS 操作系统。 安装最新版本的浏览器,建议使用 Edge、Chrome 等。

3、网络

计算机使用无线网络接入局域网,能够访问实验教学中心网络运维实验室服务器集群,并支持 对互联网的访问。

4、工具

无。

六、实验内容步骤

本实验需要VM1台,配置信息如表8-1所示。

表 8-1	虚拟机配置规划表
-------	----------

序号	虚拟机配置	操作系统配置
1	虚拟机名称: Cloud-K8s-Monitor CPU: 2核 内存: 2GB 硬盘: 40GB(系统盘)+100GB(数据存储) 网卡: Cloud-Platform-VM-Network	主机名: Cloud-K8s-Monitor 操作系统: openEuler 24.03 LT IP地址: 172.16.125.107 子网掩码: 255.255.255.0 网关: 172.16.125.1 DNS: 172.16.125.3

1、部署 Prometheus、Grafana 和 Alertmanager

步骤1:系统环境准备。创建数据目录并挂载、安装 Docker 环境。

```
1 #查看磁盘结构
2 [root@Cloud-K8s-Monitor ~]# lsblk
3 NAME
                    MAJ:MIN RM SIZE RO TYPE MOUNTPOINTS
4 sda
                      8:0
                           0 40G 0 disk
5 ⊢sda1
                      8:1
                           0 1M 0 part
6 \vdash sda2
                      8:2 0
                                1G 0 part /boot
7 └─sda3
                      8:3 0 19G 0 part
    └─openeuler-root 253:0 0 17G 0 lvm /
8
   └─openeuler-swap 253:1 0
                                 2G 0 1vm [SWAP]
9
10 sdb
                      8:16 0 100G 0 disk
11 sr0
                     11:0 1 1024M 0 rom
12 #格式化/dev/sdb
13 [root@Cloud-K8s-Monitor ~]# mkfs.ext4 /dev/sdb
14 mke2fs 1.47.0 (5-Feb-2023)
15 丢弃设备块:完成
16 创建含有 26214400 个块(每块 4k)和 6553600 个 inode 的文件系统
17 文件系统 UUID: 1522f744-18d2-450c-a555-40bc8c8ce7e3
18 超级块的备份存储于下列块:
          32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632,
19
  2654208.
20
          4096000, 7962624, 11239424, 20480000, 23887872
21
22 正在分配组表:完成
23 正在写入 inode表: 完成
24 创建日志 (131072 个块): 完成
25 写入超级块和文件系统账户统计信息:已完成
26 #创建数据目录并赋予权限
27 mkdir /data
28 chmod 777 /data
29 #进行挂载
30 mount /dev/sdb /data
31 #开机自动挂载
32 echo '/dev/sdb /data
                           ext4 defaults 0 0' >> /etc/fstab
33 #安装docker
34 yum-config-manager --add-repo http://mirrors.aliyun.com/docker-ce/linu
  x/centos/docker-ce.repo
35 sed -i 's/\$releasever/8/g' /etc/yum.repos.d/docker-ce.repo
36 yum install -y docker-ce docker-ce-cli containerd.io
37 cat > /etc/docker/daemon.json <<EOF
38 {
39
    "registry-mirrors": [
40
      "https://registry.cn-hangzhou.aliyuncs.com",
```

41	"https://hub.xdark.top",
42	"https://hub.littlediary.cn",
43	"https://dockerpull.org",
44	"https://hub.crdz.gq",
45	"https://docker.1panel.live",
46	"https://docker.mirrors.ustc.edu.cn",
47	"https://docker.m.daocloud.io",
48	"https://noohub.ru",
49	"https://huecker.io",
50	"https://dockerhub.timeweb.cloud",
51	"https://docker.1panel.dev",
52	"https://docker.unsee.tech",
53	"https://docker.1panel.live"
54]	
55 }	
56 EOF	
57 sys	temctl start docker
58 sys	temctl enable docker

步骤 2:编写 yaml 文件,部署 Prometheus、Grafana、Alertmanager

```
    (1)分别创建 /data/prometheus 、 /data/alertmanager 、 /data/grafana 目录作为 prometheus 、 alertmanager 、 grafana 服务的数据目录,并分别设置目 录 /data/prometheus 和 /data/grafana 的权限为777。
```

Shell
1 mkdir -p /data/{prometheus,alertmanager,grafana} 2 chmod 777 /data/prometheus 3 chmod 777 /data/grafana
(2)分别创建 /etc/prometheus 、 /etc/alertmanager 、 /etc/grafana 目录,并

在 /etc/prometheus 目录下创建 prometheus 服务的配置文件 prometheus.yml, 在 /etc/alertmanager 目录下创建 alertmanager 服务的配置文件 config.yml, 在 /etc/grafana 目录下创建 grafana 服务的配置文件 config.monitoring。

```
1 mkdir -p /etc/{prometheus,alertmanager,grafana}
2 vi /etc/prometheus/prometheus.yml
3 -----/etc/prometheus/prometheus.yml------
4 # 全局参数
5 global:
 6 scrape_interval: 15s # 设定抓取数据的周期,默认为1min
7 scrape_timeout: 15s # 设定抓取数据的超时时间,默认为10s
 8 evaluation interval: 30s # 设定更新rules文件的周期, 默认为1min
                        # 额外的属性, 会添加到拉取得数据并存到数据库中
 9
    external labels:
     monitor: 'codelab-monitor'
10
11
12 scrape_configs:
13 # 监控自身
14 - job_name: 'prometheus'
15 static_configs:
16
      - targets: ['localhost:9090']
17 # 监控alertmanager
18 - job_name: "alertmanager"
19
    static_configs:
      - targets: ["172.16.125.107:9093"]
20
21 -----/etc/prometheus/prometheus.yml------
22
23 vi /etc/alertmanager/config.yml
24 -----/etc/alertmanager/config.yml------
25 # 全局配置, 配置邮件服务器
26 global:
    smtp_from: '' # 配置发件人邮箱地址 可配置自己的邮箱
27
28
    smtp_smarthost: 'smtp.163.com:25' # SMTP 服务器地址与端口,也可配置qq邮箱。
    smtp_auth_username: '' # 配置发件人邮箱地址 可配置自己的邮箱
29
    smtp auth password: '' # 配置发件人邮箱密码 可配置自己邮箱的密码
30
31
    smtp_require_tls: false
32
    smtp_hello: '163.com'
33
34 # 路由配置
35 route:
    group_by: ['alertname'] # 根据告警名称分组
36
37
    group_wait: 5s
                          # 等待时间,等待同一组告警的时间
38 group_interval: 5s
                         # 每组告警之间的间隔时间
39 repeat_interval: 2m
                          # 重复发送间隔时间
40 receiver: 'email'
                         # 默认接收者设置为'email',即下方定义的'emai
  1'接收者
```

42 # 接收者配置 43 receivers: 44 - name: 'email' # 接收者名称 45 email_configs: 46 - to: '3183748324@qq.com' # 收件人邮箱地址 47 send_resolved: true # 是否发送已解决的告警 48 -----/etc/alertmanager/config.yml------49 50 vi /etc/grafana/config.monitoring 51 -----/etc/grafana/config.monitoring-----52 # grafana管理界面的登录用户密码,用户名是admin 53 GF_SECURITY_ADMIN_PASSWORD=admin#123456 54 # grafana管理界面是否允许注册,默认不允许 55 GF_USERS_ALLOW_SIGN_UP=false 56 -----/etc/grafana/config.monitoring-----

(3) 编写 docker compose 配置文件,创建 Prometheus +Grafana + Alertmanager 容器, docker compose 文件内容如下:

```
1 vi docker-compose.yml
 2 -----docker-compose.yml-----
 3 version: '3.3'
4 services:
    prometheus:
5
 6
       image: prom/prometheus:latest
 7
       container name: prometheus
 8
      restart: always
 9
      volumes:
         - /etc/localtime:/etc/localtime:ro
10
         - /etc/prometheus/:/etc/prometheus/
11
         - /data/prometheus:/prometheus
12
13
      command:
         - '--config.file=/etc/prometheus/prometheus.yml'
14
15
         - '--storage.tsdb.path=/prometheus'
16
         - '--web.console.libraries=/usr/share/prometheus/console_librarie
   s '
17
         - '--web.console.templates=/usr/share/prometheus/consoles'
18
        #热加载配置
         - '--web.enable-lifecycle'
19
20
        #api配置
21
        #- '--web.enable-admin-api'
        #历史数据最大保留时间,默认15天
22
         - '--storage.tsdb.retention.time=720d'
23
24
      networks:
25
         net:
26
           ipv4_address: 172.20.210.10
27
      links:
28
         - alertmanager
29
         - cadvisor
30
      expose:
31
         - '9090'
32
      ports:
         - 9090:9090
33
34
      depends_on:
35
         - cadvisor
36
37
    alertmanager:
38
       image: prom/alertmanager:v0.25.0
39
       container_name: alertmanager
40
       restart: always
41
      volumes:
```

42	<pre>- /etc/localtime:/etc/localtime:ro</pre>
43	<pre>- /etc/alertmanager/:/etc/alertmanager/</pre>
44	- /data/alertmanager:/alertmanager
45	command:
46	- 'config.file=/etc/alertmanager/config.yml'
47	- 'storage.path=/alertmanager'
48	networks:
49	net:
50	ipv4_address: 172.20.210.11
51	expose:
52	- '9093'
53	ports:
54	- 9093:9093
55	
56	cadvisor:
57	<pre>image: google/cadvisor:latest</pre>
58	container_name: cadvisor
59	restart: always
60	volumes:
61	<pre>- /etc/localtime:/etc/localtime:ro</pre>
62	- /:/rootfs:ro
63	- /var/run:/var/run:rw
64	- /sys:/sys:ro
65	<pre>- /var/lib/docker/:/var/lib/docker:ro</pre>
66	networks:
67	net:
68	ipv4_address: 172.20.210.12
69	expose:
70	- '8080'
71	ports:
72	- '8080:8080'
73	
74	grafana:
75	image: grafana/grafana:latest
76	container_name: grafana
77	restart: always
78	volumes:
79	<pre>- /etc/localtime:/etc/localtime:ro</pre>
80	- /data/grafana:/var/lib/grafana
81	 /etc/grafana/provisioning/:/etc/grafana/provisioning/
82	env_file:
83	<pre>- /etc/grafana/config.monitoring</pre>
84	networks:
85	net:

ipv4_address: 172.20.210.13 86 links: 87 - prometheus 88 89 ports: - 3000:3000 90 91 depends_on: 92 - prometheus 93 94 networks: 95 net: 96 driver: bridge 97 ipam: 98 config: - subnet: 172.20.210.0/24 99 100 -----docker-compose.yml------

(4) 执行 docker compose, 创建 Prometheus + Grafana + Alertmanager 容器服务。

Shell

```
1 docker compose -f docker-compose.yml up -d
```

安装完成后,用浏览器分别访问 Prometheus: http://IP:9090,和 Grafana: http://IP:3000, 如图 1-1 和 1-2 所示。

8 D A Prometheus Time Series Collectio x +		- 0 X
← C ▲ 不安全 172.16.125.107:9090/graph?g0.expr=&g0.tab=1&g0.display_mode=lines&g0.show_exemplars=0&g0.range_input=1h	● ☆ ♀ 点此搜索	s 😥 🖓 … 🥠
Prometheus Alerts Graph Status - Help		☆ € 0
Use local time Enable query history 😢 Enable autocomplete 🔮 Enable highlighting 🔮 Enable linter		
Q Expression (press Shift+Enter for newlines)		≅ 🛛 Execute
Table Graph		
< Evaluation time >		
No data queried yet		
		Remove Panel
Add Panel		

图 1-1 访问Prometheus

2 D Prometheus Time Series Collectio X 🧑 Grafana X +			- 0 ×
← C ▲ 不安全 172.16.125.107:3000/login		◎ ◎ ☆ ♀ 点比搜索	s 💓 😵 … 🍫
	ဖြို့ Documentation 💮 Support 🛱 Community Open Source Grafana v11.6.0 (d2fdff9	lec4)	

图 1-2 访问 Grafana

2、在K8s上部署采集器

(1) 部署 node-exporter

```
1 #创建命名空间devops
2 [root@k8s-master ~]# kubectl create namespace devops
3 #编写yaml文件,部署node_export
4 [root@k8s-master ~]# vi node_export.yaml
5 -----node_export.yaml-----
 6 apiVersion: apps/v1
7 kind: DaemonSet
8 metadata:
 9
    name: node-exporter
10
    namespace: monitoring
11
    labels:
12
       k8s-app: node-exporter
13 spec:
14
    selector:
15
       matchLabels:
16
           k8s-app: node-exporter
17
    template:
      metadata:
18
19
         labels:
20
           k8s-app: node-exporter
21
       spec:
22
         tolerations:
23
         - key: node-role.kubernetes.io/control-plane
24
           operator: Exists
25
           effect: NoSchedule
         - key: node-role.kubernetes.io/master
26
27
           operator: Exists
28
           effect: NoSchedule
29
         containers:
         - image: bitnami/node-exporter:1.7.0
30
31
           imagePullPolicy: IfNotPresent
32
           name: prometheus-node-exporter
33
           ports:
34
           - containerPort: 9100
35
             hostPort: 9100
36
             protocol: TCP
37
             name: metrics
38
           volumeMounts:
39
           - mountPath: /host/proc
40
             name: proc
41
           - mountPath: /host/sys
42
             name: sys
```

43	- mountPath: /host
44	name: rootfs
45	args:
46	<pre>path.procfs=/host/proc</pre>
47	path.sysfs=/host/sys
48	path.rootfs=/host
49	volumes:
50	- name: proc
51	hostPath:
52	path: /proc
53	- name: sys
54	hostPath:
55	path: /sys
56	- name: rootfs
57	hostPath:
58	path: /
59	hostNetwork: true # 使用宿主机网络和PID
60	hostPID: true
61	node_export.yaml
62	[root@k8s-master ~]# kubectl apply -f node_export.yaml
63	

(2) 部署 kube-state-metrics

```
1 vi kube-state-metrics.yaml
 2 -----kube-state-metrics.yaml-----
 3 apiVersion: apps/v1
 4 kind: Deployment
 5 metadata:
     name: kube-state-metrics
 6
     namespace: kube-system
 7
 8 spec:
 9 replicas: 1
10 selector:
11
       matchLabels:
12
         app: kube-state-metrics
13
   template:
      metadata:
14
15
         labels:
16
          app: kube-state-metrics
17
     spec:
         serviceAccountName: kube-state-metrics
18
19
        containers:
20
        - name: kube-state-metrics
21
           image: registry.cn-hangzhou.aliyuncs.com/zhangshijie/kube-state
   -metrics:v2.6.0
22
          ports:
23
          - containerPort: 8080
24
25 ---
26 apiVersion: v1
27 kind: ServiceAccount
28 metadata:
29
    name: kube-state-metrics
30
    namespace: kube-system
31 ---
32 apiVersion: rbac.authorization.k8s.io/v1
33 kind: ClusterRole
34 metadata:
     name: kube-state-metrics
35
36 rules:
37 - apiGroups: [""]
     resources: ["nodes", "pods", "services", "resourcequotas", "replicati
38
  oncontrollers", "limitranges", "persistentvolumeclaims", "persistentvol
  umes", "namespaces", "endpoints"]
39 verbs: ["list", "watch"]
```

```
40 - apiGroups: ["extensions"]
41 resources: ["daemonsets", "deployments", "replicasets"]
42 verbs: ["list", "watch"]
43 - apiGroups: ["apps"]
44 resources: ["statefulsets"]
45 verbs: ["list", "watch"]
46 - apiGroups: ["batch"]
    resources: ["cronjobs", "jobs"]
47
48 verbs: ["list", "watch"]
49 - apiGroups: ["autoscaling"]
    resources: ["horizontalpodautoscalers"]
50
51 verbs: ["list", "watch"]
52 ---
53 apiVersion: rbac.authorization.k8s.io/v1
54 kind: ClusterRoleBinding
55 metadata:
56
    name: kube-state-metrics
57 roleRef:
58
    apiGroup: rbac.authorization.k8s.io
59 kind: ClusterRole
60 name: kube-state-metrics
61 subjects:
62 - kind: ServiceAccount
    name: kube-state-metrics
63
64
    namespace: kube-system
65
66 ---
67 apiVersion: v1
68 kind: Service
69 metadata:
70 annotations:
71 prometheus.io/scrape: 'true'
72
    name: kube-state-metrics
73 namespace: kube-system
74
    labels:
75
      app: kube-state-metrics
76 spec:
77 type: NodePort
78
    ports:
79 - name: kube-state-metrics
80
      port: 8080
81
    targetPort: 8080
      nodePort: 31888
82
83
      protocol: TCP
```

```
84 selector:
85 app: kube-state-metrics
86 ------kube-state-metrics.yaml-----
87 [root@k8s-master ~]# kubectl apply -f kube-state-metrics.yaml
```

3、修改 Prometheus 配置文件,配置监控对象

Shell
1 #在配置文件prometheus.yml后添加监控任务
2 vi /etc/prometheus/prometheus.yml
3prometheus.yml
4 # 监控主机
5 - job_name: "K8sHost"
6 static_configs:
7 - targets: ["172.16.125.101:9100","172.16.125.102:9100","172.16.1
25.103:9100","172.16.125.104:9100","172.16.125.105:9100"]
8 # 监控K8s
9 - job_name: "K8s-Cluster"
10 honor_timestamps: true
<pre>11 metrics_path: /metrics</pre>
12 scheme: http
13 static_configs:
14 - targets: ["172.16.125.101:31888"]
15 metric_relabel_configs:
16 - target_label: cluster
17 replacement: K8s-Cluster
18
19 #重启prometheus容器
20 docker restart prometheus

使用浏览器访问 Prometheus 查看监控对象。

Prometheus Alerts Graph Status - H	ielp		÷ (0
Service Discovery			
	Q Filter by labels		
 K8s-Cluster (1 / 1 active targets) K8sHost (5 / 5 active targets) 			
 alertmanager (1 / 1 active targets) 			
 prometheus (1 / 1 active targets) 			
K8s-Cluster showless			
Discovered Labels		Target Labels	
address=*172.16.125.101:31888*		instance="172.16.125.101:31888"	
metrics_path_=="/metrics" scheme_="http"		Job="K8s-Cluster"	
scrape_interval="15s"			
job="K8s-Cluster"			
K8sHost show less			
Discovered Labels		Target Labels	
address=*172.16.125.101:9100*		instance="172.16.125.101:9100"	
metrics_path_="/metrics" scheme_="http"		job="K8sHost"	
_scrape_interval_="15s"			
job="K8sHost"			
_address*172.16.125.102:9100*		instance-"172.16.125.102:9100"	
metrics_pam_= /metrics		job= Kostiox	
and the second second second			Ψ.

图 9-1 prometheus监控数据

4、在 Grafana 进行数据可视化

步骤1:添加Prometheus数据源

选择左侧菜单"Connections"->"Data sources",点击右上角【add new data source】按钮,选择 prometheus 类型的数据源进行配置,在这里配置 prometheus 服务器的访问路径: http://172.16.125.107:9090,如图 9-2 所示,其他设置保存默认,最后测试并保存,如图 9-3 所示。

Ø	Grafana	Home > Connections > Data sources > prometheus-1	Q Search or jump to		📼 ctrl+k 🕂 👻 💿 📮 📀
③ □ ☆ 器		prometheus-1 Type: Prometheus	Type Prometheus	Alerting Supported	Explore data Build a dashboard
Ø		11 Settings			
<u>,8</u> ,	Drilldown 🧐 New!				
¢	Alerting Connections	Configure your Prometheus data source below Or skip the effort and get Prometheus (and Loki) as fully-managed, scalable, and hosted data sources from Grafana Labs with the free-fore			×
~	Add new connection Data sources Administration	Name O prometheus-1 Default			
		Before you can use the Prometheus data source, you must configure it below or in the config file. For detailed instructions, <u>view the documentation</u> Fields marked with * are required Connection			
		Prometheus server 0kL * ○ http://t/2.36.125.307.9090			
		Authentication Authentication methods Choose an authentication method to access the data source			
		Authentication method No Authentication			

图 9-2 配置数据源



图 9-3 添加 prometheus 数据源

步骤 2: 添加仪表盘

(1)选择左侧菜单"Dashboards"->"New"->"Import",输入仪表盘ID为11074并载入,如图9-4、图9-5所示。

Ø	Grafana	Home > Dashboards > Import dashboard	Q Search or jump to	🖽 ctrl+k	+ ~	0	ə 🕛
6 □ ☆		Import dashboard Import dashboard from file or Grafana.com					
88	Dashboards Playlists Snapshots Library panels Shared dashboards	Upload dashboard JSON file Drag and drop here or click to browse Accepted file types::_json, txt					
) (1) (1)		Find and import dashboards for common applications at <u>grafana.com/dashboards</u> & Load					
> © 0		Import via dashboard J50H model { "title": "Example - Repeating Dictionary variables", "uid": ".0HnE0N4Z", "anante: []					
0	Administration	Loud Cancel					

图 9-4 输入仪表盘ID

Ø	Grafana	Home > Dashboards > Import dashboard		Q Search or jump to	🖽 ctrl+k	+ ~	0 🖶	
□□☆器	Home Bookmarks Starred Dashboards	Import dashboard Import dashboard from file or Grafana.com Importing dashboard from Grafana.com						
	Playlists Snapshots Library panels	Published by Updated on	StarsL.cn 2023-07-07 15:40:39					
@ . <u>#</u>	Explore Drilldown ØNew!	Options Name						
\$ \$		Node Exporter Dashboard EN 20201010-StarsL.cn Folder Dashboards Unique identifier (UID)						
٢	Administration	The unique identifier (UII) of a dashboard carl. The U inique identify a dashboard beam nunique Graham carls. The U inique having consident UII of the accessing distributed with a chardwood will not break any local multicle of a dashboard will not break up to be the of a dashboard will not be to be the of a dashboard will not be to be to be the of a dashboard will not be to be tobe to						

图 9-5添加仪表盘

(2) 最后导入该仪表盘,数据展示如图 9-6 所示。



图 9-6 主机监控仪表盘

(3)按照上述步骤,添加仪表盘ID为15661并载入,最后导入该仪表盘,数据展示如图9-7 所示。



图 9-7 K8S集群监控仪表盘

💡 提醒:

可以到 Grafana 官网 https://grafana.com/grafana/dashboards/ 根据采集器的名称 或服务名称选择相应的仪表盘。可以下载仪表盘的 json 文件或复制仪表盘的 ID 用于 导入。

5、使用 HertzBeat 监控 K8s

步骤1: 部署 HertzBeat

Shell

```
1 docker run -d -p 1157:1157 -p 1158:1158 \
```

```
2 -v $(pwd)/data:/opt/hertzbeat/data \
```

```
3 -v $(pwd)/logs:/opt/hertzbeat/logs \
```

```
4 --restart=always ∖
```

5 -- name hertzbeat apache/hertzbeat

浏览器访问 http://172.16.125.107:1157/,默认账户密码 admin/hertzbeat,如图 9-8 所示、 9-9 所示。

	Hertz Beet 周用友好的开观实时监控系统		
开源、高	性能、分布式的	登入 HertzBeat	
实时监控		A admin •	
• 一处式收缩失敏通知 古		₽	
 易用友好,无需 Agent, 强大监控模版能力,自定 	为国行的运行,以18年5月,18日7月3日,1967年,2018年,1958年。 全页面操作,国标点一点就能监控告警。 义监控任何您想要的指标。	▼ 自动登录	
 高性能,采集器集群横向 自由的阈值规则,邮件钉 	扩展,支持多隔离网络监控,云边协同。 钉微值(飞扫短信等消息及时送达。	<u>.</u>	
• 提供强大的状态贝科建切	能,轻松问用尸悸还愿产品服务的实时状态。		
	Apache HertzBeat (incubating) v1.7.0 Copyright © 2025 Apache HertzBeat Licensed under the Apache License, Versio		

图 9-8 登录



图 9-9 HertzBeat 界面

步骤 2: 添加 K8s 监控

在左侧菜单栏选择"监控中心"->"新增监控"->"云原生监控"->"Kubernetes",配置监控 信息测试连接并添加,如图 9-10、图 9-11 所示。

🕅 Hertz Beät	≡ 0	Q. 搜索监控任务名称、HOST等				¢ ⊲ 8	🕸 💄 admin
主导航 〇 仪表盘 监控	 ② 仪表盘 监控中心是监持 您可以对监控3 ③ 常见问题 	目 监控中心 空源游管理入口,以列表的形式展示当前已添加的监控, 和选进行新潮,修改、删除,暂停监控,中入导出,批 1 [] 使用指南	Q 搜索特添加监控任务的类型: Linux, Redis				收起◇
 显 监控中心 目 自定义看板 区 监控模版 	○ 器新增	监控 ···· / 1 >	AUTO 操作系统监控	Ŭ Î			搜索
告警	₩ K8sC	luster 6.125.101	Web服务器运控 大数据监控 得在性物	ž			<u></u>
☑ 分组收敛 ♀ 告警抑制 ♀ 告警抑制	C Kuber	netes	应用服务监控 中间件监控	~			③ 3 minutes ago
 ◆ 告留辞款 查 告警中心 ☐ 消息通知 			数据库监控 自定义监控	v v			
高级 □ 状态页面 品 采集集群			AI大視型 服务器监控	v v			
			网络监控 应用程序监控	•			
 意 系统设置 ● 帮助中心 			云際生造控 ゆ Kubernetes	• > -			

图 9-10 选择Kubernetes

🕅 Hertz Beat	≡ ()	Q 捜索监控任务名称、HOST等				
	◎ 仪表盘	④ 新増 Kubernetes 监控				✓ 測试连接成功
土守肌	HertzBeat 通道 注意 A ・ 为了	过查询 Kubernetes ApiServer api 来对 kubernetes 的通用性能 ? 鉴控 Kubernetes 中的信息,则需要获取到可访问 Api Server	皆标(nodes、namespaces、pods、se 的授权 TOKEN,让平集请求获取到对F	ervices)进行采集监控。 前的信息,占击查看获取步	K.	
 ② (又表蓝 	② 常见问题	 回 使用指南 	3321X 10(U() 12.888 H3(10-403)/31		*0	
国由党ツ看板		* 目标Host:	172.16.125.101		0	
日本控模版		• 任务名称:	K8sCluster		0	
告警		* AniSen/er傑口:	6443			
🖽 阈值规则		Abserver wither -				
ダ 集成接入		* 认证方式:	Bearer Token			
☑ 分組收敛		* 认证Token:	eyJhbGciOiJSUzI1NilsImtpZCI6Inpl0	DEhSaGIIc2p0Zm1IMIZscWN	faHZtU1AtYkJHOTY	
せ 告警抑制						
<i>憂</i> 告警静默		采集器 ②:	默认系统调度-公共集群模式			
查 告警中心		监控周期 ⑦:	60 18			
☑ 消息通知		· #中标文 ③·	629	. /#		
高级		新足が立 0.	246	•	+	
□ 状态页面		绑定注解 ⑦:	键	: 值	+	
品 采集集群		描述备注 ②:				
♀ 标签管理						
冎 插件管理					0/100	
更多			3052	744 TO 116		
◎ 系统设置			26 IA	#8AC 4X7H		
● 帮助中心						

图 9-11 测试连接K8s集群

💡 提醒:

认证 Token 的获取方式可参考《实验 7-Kubernetes Cluster》使用 Token 添加已有 Kubernetes 集群到 Kuboard。复制脚本指令,直接粘贴到 kubernetes 集 群的 master 节点的命令行窗口执行,以获得 Token。

步骤3: 查看 K8s 监控数据

监控实时数据详情如图 9-12 所示,监控历史图表详情如图 9-13 所示。

🕅 HertzBeät	프 🌔 직搜索	监控任务名称、HOST等									۵ (⊲ 8	۵ 🙎
导航		/ ③ 监控详情 / Kubernete	5										
仪表盘	运 监控实时数据详情	ビ 监控历史图表详情								自动刷新 13	や 〇		
腔													
监控中心	监控任务信息				nodes						④ 采集时间:	16:23:29	
自定义看板 监控模版	K8sCluster		© 正常		E#	节点名称	节点就绪状态	CPU 容量	可分配 CPI	U 内存容i Mi	量 可分配 Mi	内存创建	时间
the second se	ID	498873169594880	HOST	172.16.125.101	- 1	k8s-master	True	2	2	3405.785	2 3305.7	852 2025	-03-21T
阈值规则 # #### \	端口	6443	监控周期	60s		k8s-worker1	True	2	2	3405.777	3 3305.7	773 2025	-03-21T
■ 成 度 へ	标签					k8s-worker2	True	2	2	3405.777	3 3305.7	773 2025	-03-21T
告警抑制					- 1	k8s-worker3	True	2	2	3405.785	2 3305.7	852 2025	-03-21T
告警静默	描述				- 1	k8s-worker4	True	2	2	3405.777	3 3305.7	773 2025	-03-21T
告警中心	创建时间	2025-04-10 16:00:08	编辑时间	2025-04-10 16:00:08	-								
消息通知 _及	namespaces			④ 采集时间: 16:	:23:29 🔀	pods						③ 采集时间:	16:23:32
状态页面	命名空(ii)	状态	创建时间		Pod名称	命名空间	状态	重启次数	主机IP	Pod IP	创建时间	启动
采集集群 	default	Active		2025-03-21T01:52:53Z	î.	nginx-depl	default	Running	Always	172.16.125	192.168.24	2025-03-21	2025-(
标立官理	devops	Active		2025-03-27T11:09:05Z		nginx-depl	default	Running	Always	172.16.125	192.168.10	2025-03-21	2025-(
зніт в +я	kube-node-lease	Active	Active			nginx-depl	default	Running	Always	172.16.125	192.168.19	2025-03-21	2025-(
系统设置	kube-public	Active		2025-03-21T01:52:53Z		nginx2-867	default	Running	Always	172.16.125	192.168.12	2025-03-21	2025-(
帮助中心	kube-system	Active		2025-03-21T01:52:53Z		nginx2-867	default	Running	Always	172.16.125	192.168.10	2025-03-21	2025-(
	kuboard	Active		2025-03-21T03:03:00Z		nginx2-867	default	Running	Always	172.16.125	192.168.19	2025-03-21	2025-(
	metallb-system	Active		2025-03-21T11:37:31Z	~	node-evnor	devone	Running	Alwave	172 16 125	172 16 125	2025-03-27	2025.1

图 9-12 监控实时数据



图 9-13 监控历史图表

七、实验讲解

本实验配置讲解视频,访问课程学习平台。

八、实验考核

实验考核为【实验随堂查】。

实验随堂查:每个实验设置 3-5 考核点,学生现场进行演示和汇报讲解。

1、考核点

考核点1:完成 Prometheus 监控平台的部署。(30分)

考核点 2: 使用 Grafana 可视化监控数据。(40分)

考核点 3: 使用 HertzBeat 监控 K8s。(30 分)

2、考核方式

以实验小组为单位进行考核,每个小组由1位同学进行实验成果汇报,小组其他成员回答教师 提问。根据汇报和答疑情况,对小组成员进行逐一打分。

由教师进行评分。