二

25 如何搭建一个高可用的 ZooKeeper 生产环境?

如何在生产环境中部署一个安全可靠的 ZooKeeper 运行环境，是每个 IT 技术人员都要掌握的知识。没有一个安全可靠的运行环境，无论开发的服务再怎么优秀，都无法为用户提供服务。因此，本课时的重点将聚焦在 ZooKeeper 生产环境下安装的相关知识和参数配置技巧上。

运行方式

首先，我们来介绍一下 ZooKeeper 服务的几种运行模式，ZooKeeper 的运行模式一般分为**单机模式、伪集群模式、集群模式**。其中单机模式和伪集群模式，在我们的日常开发中经常用到。

单机模式配置

在 ZooKeeper 的单机模式下，整个 ZooKeeper 服务只运行在一台服务器节点下。在 zoo.cfg 配置文件中，我们只定义了基本的 dataDir 目录和 clientPort 端口号等信息。

```
tickTime=2000

dataDir=/var/lib/zookeeper

clientPort=2181
```

伪集群模式配置

与单机模式相比，伪集群模式的意思是：**虽然 ZooKeeper 服务配置有多台服务器节点，但是这些集群服务器都运行在同一台机器上。** 通常伪集群服务器在配置的时候，每台服务器间采用不同的端口号进行区分，多用在本地开发或测试中。

如下面的代码所示，在配置伪集群的时候，我们将每台服务器的 IP 地址都指向 127.0.0.1，即本机地址，每台 ZooKeeper 对外提供服务的端口分别是 2223、3334、4445。

```
tickTime=2000
```

```
dataDir=/var/lib/zookeeper

clientPort=2181

sever.1=127.0.0.1:2222:2223

sever.2=127.0.0.1:3333:3334

sever.3=127.0.0.1:4444:4445
```

集群模式配置

集群模式在配置上与伪集群模式基本相同。不同之处在于配置服务器地址列表的时候，**组成 ZooKeeper 集群的各个服务器 IP 地址列表分别指向每台服务在网络中的实际 IP 地址。**

```
tickTime=2000

dataDir=/var/lib/zookeeper

clientPort=2181

sever.1=192.168.1.101:2222:2223

sever.1=192.168.1.102:3333:3334

sever.1=192.168.1.103:4444:4445
```

在 ZooKeeper 集群的三种模式中，单机模式和伪集群模式经常用于开发和测试中。而分别利用不同网络上的物理机器组成的 ZooKeeper 集群经常被我们作为生成系统的环境配置方式。

容器化部署

介绍完 ZooKeeper 服务器三种模式的配置方法后，接下来我们学习如何利用容器化技术来部署 ZooKeeper 集群。

首先，我们来了解一下什么是容器化技术。在我们前面的课程中，无论是在单机模式下在 ZooKeeper 数据模型中创建数据节点，还是在集群模式中，ZooKeeper 集群进行 Leader 节点选举，它们的实现都依赖于 ZooKeeper 服务部署在真实的物理机器上运行。

随着 IT 技术的发展，人们开始设想能否通过软件的方式，在一台机器上模拟出多台机器，突破单体物理机器的限制，利用一台物理机器的计算资源模拟出多台机器，为技术开发提供更加灵活和高效的环境。因此，有了我们比较熟悉的 VMware Workstation 等虚拟化技术软件。

利用该软件，我们可以在单一的桌面系统上，同时运行多个不同的操作系统。每个操作系统都可以看作独立的计算机。可以在不同的系统上进行程序开发、测试、服务部署等工作。虽然 VMware Workstation 为我们解决了系统资源虚拟化的问题，但是这种实现方式也有自身的缺点，比如每个虚拟机实例都需要运行客户端操作系统的完整副本以及其中包含的大量应用程序。从实际运行的角度来说，这会对物理机资源产生较大占用，也不利于整个虚拟系统的扩展和维护。

接下来我们要介绍的另一种容器化解决方式叫作 Docker，在实现容器化部署的同时，避免了 VMware Workstation 的上述问题。Docker 是一个开源的应用容器引擎，基于 Go 语言并遵从 Apache2.0 协议开源。与 VMware Workstation 相比，Docker 容器更加轻量化。在 Web 网站自动化部署、持续集成与发布等使用场景中具有广泛的应用。

本课时中，我们也使用 Docker 容器化技术来实现一个生产环境中的 ZooKeeper 集群部署案例。

使用 Docker 部署

安装 Docker

为了使用 Docker 容器技术部署我们的应用服务，首先，我们要在服务器上安装 Docker 软件。以 Linux 系统中的 CentOS 7 64 位版本为例。如下面的代码所示，通过 curl 命令使用官方安装脚本自动安装。curl 通过资源地址获取资源到本地进行安装。而国内服务器由于网络等原因可能无法访问默认的 Docker 资源服务器，因此这里采用的是国内阿里云的镜像资源服务器。

```
curl -fsSL https://get.docker.com | bash -s docker --mirror Aliyun
```

创建 Docer 服务器

安装完 Docker 后，接下来我们就开始部署 ZooKeeper 的集群环境。这里的集群环境仍然由三台 Linux 服务器组成。而与上面我们介绍的利用网络中三台实体机器不同，这三台服务器可以通过 Docker 的方式来创建。

如下面的代码所示，首先打开系统终端，输入 docker pull 获取需要的系统镜像文件，这里选择的是 3.6 版本的 ZooKeeper，当然我们也可以不指定具体版本号，系统会默认拉取最新版本的 ZooKeeper 。之后我们通过 docker run 命令来启动 ZooKeeper 镜像服务器。执行完这两个步骤，我们就拥有一台运行 ZooKeeper 服务的服务器了。

```
docker pull zookeeper:3.6
```

```
docker run -d --name=zookeeper1 --net=host zookeeper
```

配置 ZooKeeper 服务

创建完 ZooKeeper 服务器，接下来就要通过 zoo.cfg 文件来配置 ZooKeeper 服务。与部署在物理机器上不同，我们通过 docker exec 命令进入 Docker 创建的 ZooKeeper 服务器中，之后通过 vim 命令打开 zoo.cfg 文件进行相关配置。

```
docker exec -it zookeeper1 /bin/bash

vim /conf/zoo.cfg
```

多台服务器配置

按照上面介绍的方法，如果我们想搭建三台服务器规模的 ZooKeeper 集群服务，就需要重复上面的步骤三次，并分别在创建的三台 ZooKeeper 服务器进行配置。

不过在实际生产环境中，我们需要的 ZooKeeper 规模可能远远大于三台，而且这种逐一部署的方式不但浪费时间，在配置过程中出错率也较高。因此，这里介绍另一种配置方式，通过 Docker Compose 的方式来部署 ZooKeeper 集群。

Docker Compose 是用于定义和运行多容器 Docker 应用程序的工具。通过 Compose，你可以使用 YML 文件来配置应用程序需要的所有服务。然后，使用一个命令，就可以从 YML 文件配置中创建并启动所有服务。如下面的代码所示，我们创建了一个名为 docker-compose.yml 的配置文件。

```
version: '3.6'

services:

        zk1:

                image: zookeeper:3.6

                restart: always

                hostname: zk1

                container_name: zk1

                ports: - 2181:2181

                environment:
```

```
            ZOO_MY_ID: 1
            ZOO_SERVERS: server.1=zk1:2888:3888 server.2=zk2:2888:3888 server.
    zk2:
        image: zookeeper:3.6
        restart: always
        hostname: zk2
        container_name: zk2
        ports: - 2182:2181
        environment:
         ZOO_MY_ID: 1
         ZOO_SERVERS: server.1=zk1:2888:3888 server.2=zk2:2888:3888 server.
    zk3:
        image: zookeeper:3.6
        restart: always
        hostname: zk3
        container_name: zk3
        ports: - 2183:2181
        environment:
         ZOO_MY_ID: 1
         ZOO_SERVERS: server.1=zk1:2888:3888 server.2=zk2:2888:3888 server.
```

在这个文件中，我们将需要手工逐一创建的 ZooKeeper 服务器的创建过程，通过 docker-compose.yml 配置文件的方式进行了描述。在这个配置文件中，我们告诉 Docker 服务分别创建并运行三个 ZooKeeper 服务器，并分别将本地的 2181, 2182, 2183 端口绑定到对应容器的 2181 端口上。

Docker 容器化方式部署的服务默认情况下对外界隔离，默认的 Docker 容器内服务无法被外界访问，因此需要进行端口映射，将外部物理机器的端口映射到对应的 Docker 服务器端口，这样外界在对物理机器进行访问后，系统会自动映射该端口到对应的 Docker 服务上。

在 environment 节点下，我们配置了 ZooKeeper 集群需要的两个配置参数，分别是 ZOO_MY_ID 以及 ZooKeeper 集群的服务器列表 ZOO_SERVERS。ZOO_MY_ID 是

1-255 之间的整数，必须在集群中唯一。

启动服务

在编写完 docker-compose.yml 配置文件的相关信息后，接下来我们就启动 docker 创建 ZooKeeper 集群服务。如下面的代码所示，首先，我们打开系统终端，输入 docker-compose up 命令来启动服务器。之后终端会显示我们配置的三台服务器都成功启动。

```
docker-compose up

Name              Command               State           Ports

------------------------------------------------------------------------

zk1   /docker-entrypoint.sh zkSe ...   Up      0.0.0.0:2181->2181/tcp

zk2   /docker-entrypoint.sh zkSe ...   Up      0.0.0.0:2182->2181/tcp

zk3   /docker-entrypoint.sh zkSe ...   Up      0.0.0.0:2183->2181/tcp
```

访问服务

ZooKeeper 集群配置完成并成功启动后，我们可以通过客户端命令来访问集群服务。如下面的代码所示，通过 zkCli.sh -server 客户端命令来访问集群服务器。

```
zkCli.sh -server localhost:2181,localhost:2182,localhost:2183
```

总结

本课时我们介绍了 ZooKeeper 的三种部署方式，学习了在这三种部署方式下，zoo.cfg 的不同配置方式。之后介绍了什么是容器化技术，并重点介绍了目前最为流行的容器化技术 Docker。并利用 Docker 创建了三台 Linux 服务器，通过这三台服务器来部署 ZooKeeper 集群。

相比本课时的例子，在实际生产环境中，对于 ZooKeeper 的性能要求可能更高。为了满足性能的要求，我们可以在三台服务器的基础上对 Docker 服务器进行动态增加来满足性能要求，这也是本课时留给你的作业。

在扩展集群规模的时候，根据 ZooKeeper 集群中 Leader 节点的选举原则，整个 ZooKeeper 集群服务器在数量上，尽量采用奇数原则，从而满足当 Leader 节点选举时，能够最终产生大多数的投票结果，避免偶数服务器一直存在票数相等的问题，从而出现脑裂等问题。

上一页 下一页