



下载APP



40 | 软件部署实战（上）：部署方案及负载均衡、高可用组件介绍

2021-08-26 孔令飞

《Go 语言项目开发实战》

课程介绍 >



讲述：孔令飞

时长 10:33 大小 9.68M



你好，我是孔令飞。

接下来，我们就进入到这门课的最后一个模块，服务部署部分的学习。在这一模块中，我会带着你一步一步地部署一个生产级可用的 IAM 应用。

在 [03 讲](#) 中，我们快速在单机上部署了 IAM 系统，但这样的系统缺少高可用、弹性扩容等能力，是很脆弱的，遇到流量波峰、发布变更很容易出问题。在系统真正上线前，我们需要重新调整部署架构，来保证我们的系统具有负载均衡、高可用、弹性伸缩等核心运营能力。



考虑到你手中的系统资源有限，这一模块会尽量简单地展示如何部署一个相对高可用的 IAM 系统。按照我讲的部署方法，基本上可以上线一个中小型的系统。

在这一模块中，我会介绍两种部署方式。

第一种是传统的部署方式，基于物理机 / 虚拟机来部署，容灾、弹性伸缩能力要部署人员自己实现。第二种是容器化部署方式，基于 Docker、Kubernetes 来部署，容灾、弹性伸缩等能力，可以借助 Kubernetes 自带的功能来实现。

接下来的三讲，我们先来看下传统的部署方式，也就是如何基于虚拟机来部署 IAM 应用。今天我主要讲跟 IAM 部署相关的两个组件，Nginx + Keepalived 的相关功能。

部署方案

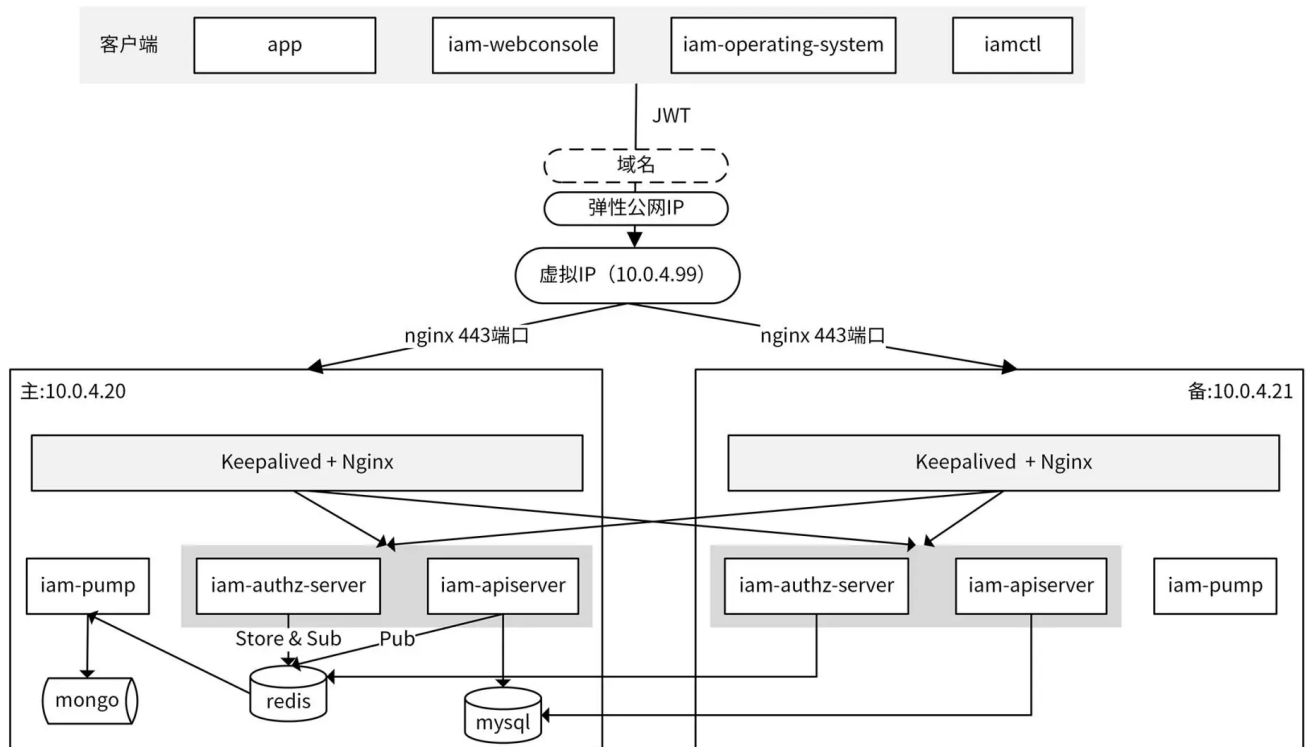
先来整体看下我们的部署方案。

这里，我采用 Nginx + Keepalived 来部署一个高可用的架构，同时将组件都部署在内网，来保证服务的安全性和性能。

部署需要两台物理机 / 虚拟机，组件之间通过内网访问。所需的服务器如下表所示：

VIP	内网IP	主机名	Nginx端口
10.0.4.99	10.0.4.20	VM-4-20-centos	80
10.0.4.99	10.0.4.21	VM-4-21-centos	80

两台服务器均为腾讯云 CVM，VIP（Virtual IP，虚拟 IP）为10.0.4.99。部署架构如下图所示：



这里我来具体介绍下图中的部署架构。部署采用的这两台 CVM 服务器，一主一备，它们共享同一个 VIP。同一时刻，VIP 只在一台主设备上生效，当主服务器出现故障时，备用服务器会自动接管 VIP，继续提供服务。

主服务器上部署了 iam-apiserver、iam-authz-server、iam-pump 和数据库 mongodb、redis、mysql。备服务器部署了 iam-apiserver、iam-authz-server 和 iam-pump。备服务器中的组件通过内网 10.0.4.20 访问主服务器中的数据库组件。

主备服务器同时安装了 Keepalived 和 Nginx，通过 Nginx 的反向代理功能和负载均衡功能，实现后端服务 iam-apiserver 和 iam-authz-server 的高可用，通过 Keepalived 实现 Nginx 的高可用。

我们通过给虚拟 IP 绑定腾讯云弹性公网 IP，从而使客户端可以通过外网 IP 访问内网的 Nginx 服务器（443 端口），如果想通过域名访问内网，还可以申请域名指向该弹性公网 IP。

通过以上部署方案，我们可以实现一个具有较高可用性的 IAM 系统，它主要具备下面这几个能力。

高性能：可以通过 Nginx 的负载均衡功能，水平扩容 IAM 服务，从而实现高性能。

具备容灾能力：通过 Nginx 实现 IAM 服务的高可用，通过 Keepalived 实现 Nginx 的高可用，从而实现核心组件的高可用。

具备水平扩容能力：通过 Nginx 的负载均衡功能，实现 IAM 服务的水平扩容。

高安全性：将所有组件部署在内网，客户端只能通过VIP:443端口访问 Nginx 服务，并且通过开启 TLS 认证和 JWT 认证，保障服务有一个比较高的安全性。因为是腾讯云 CVM，所以也可以借助腾讯云的能力再次提高服务器的安全性，比如安全组、DDoS 防护、主机安全防护、云监控、云防火墙等。

这里说明下，为了简化 IAM 应用的安装配置过程，方便你上手实操，有些能力，例如数据库高可用、进程监控和告警、自动伸缩等能力的构建方式，这里没有涉及到。这些能力的构建方式，你可以在日后的工作中慢慢学习和掌握。


接下来，我们看下这个部署方案中用到的两个核心组件，Nginx 和 Keepalived。我会介绍下它们的安装和配置方法，为你下一讲的学习做准备。

Nginx 安装和配置

Nginx 功能简介

这里先简单介绍下 Nginx。Nginx 是一个轻量级、高性能、开源的 HTTP 服务器和反向代理服务器。IAM 系统使用了 Nginx 反向代理和负载均衡的功能，下面我就来分别介绍下。

为什么需要反向代理呢？在实际的生产环境中，服务部署的网络（内网）跟外部网络（外网）通常是不通的，这就需要一台既能够访问内网又能够访问外网的服务器来做中转，这种服务器就是反向代理服务器。Nginx 作为反向代理服务器，简单的配置如下：

 复制代码

```
1 server {
2     listen      80;
3     server_name  iam.marmotedu.com;
4     client_max_body_size 1024M;
5
6     location / {
7         proxy_set_header Host $http_host;
8         proxy_set_header X-Forwarded-Host $http_host;
9         proxy_set_header X-Real-IP $remote_addr;
10        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
11     proxy_pass    http://127.0.0.1:8080/;
12     client_max_body_size 100m;
13 }
14 }
```

Nginx 的反向代理功能，能够根据不同的配置规则转发到不同的后端服务器上。假如我们在 IP 为 x.x.x.x 的服务器上，用上面说的 Nginx 配置启动 Nginx，当我们访问 `http://x.x.x.x:80/` 时，会将请求转发到 `http://127.0.0.1:8080/`。listen 80 指定了 Nginx 服务器的监听端口，`proxy_pass http://127.0.0.1:8080/` 则指定了转发路径。

Nginx 另一个常用的功能是七层负载均衡。所谓的负载均衡，就是指当 Nginx 收到一个 HTTP 请求后，会根据负载策略将请求转发到不同的后端服务器上。比如 iam-apiserver 部署在两台服务器 A 和 B 上，当请求到达 Nginx 后，Nginx 会根据 A 和 B 服务器上的负载情况，将请求转发到负载较小的那台服务器上。

这里要求 iam-apiserver 是无状态的服务。Nginx 有多种负载均衡策略，可以满足不同场景下的负载均衡需求。

Nginx 安装步骤


接下来，我就来介绍下如何安装和配置 Nginx。

我们分别在 10.0.4.20 和 10.0.4.21 服务器上执行如下步骤，安装 Nginx。


在 CentOS 8.x 系统上，我们可以使用 yum 命令来安装，具体安装过程可以分为下面 4 个步骤。

第一步，安装 Nginx：

```
1 $ sudo yum -y install nginx
```


 复制代码

第二步，确认 Nginx 安装成功：

 复制代码

```
1 $ nginx -v
2 nginx version: nginx/1.14.1
```

第三步，启动 Nginx，并设置开机启动：

 复制代码

```
1 $ sudo systemctl start nginx
2 $ sudo systemctl enable nginx
```

Nginx 默认监听80端口，启动 Nginx 前要确保80端口没有被占用。当然，你也可以通过修改 Nginx 配置文件/etc/nginx/nginx.conf修改 Nginx 监听端口。

第四步，查看 Nginx 启动状态：

 复制代码

```
1 $ systemctl status nginx
```

输出中有active (running)字符串，说明成功启动。如果 Nginx 启动失败，你可以查看/var/log/nginx/error.log日志文件，定位错误原因。

Keepalived 安装和配置

Nginx 自带负载均衡功能，并且当 Nginx 后端某个服务器故障后，Nginx 会自动剔除该服务器，将请求转发到可用的服务器，通过这种方式实现后端 API 服务的高可用。但是 Nginx 是单点的，如果 Nginx 挂了，后端的所有服务器就都不能访问，所以在实际生产环境中，也需要对 Nginx 做高可用。


业界最普遍采用的方法是通过 Keepalived 对前端 Nginx 实现高可用。Keepalived + Nginx 的高可用方案具有服务功能强大、维护简单等特点。

接下来，我们来看下如何安装和配置 Keepalived。

Keepalived 安装步骤

我们分别在10.0.4.20和10.0.4.21服务器上执行下面 5 个步骤，安装 Keepalived。

第一步，下载 Keepalived 的最新版本（这门课安装了当前的最新版本 2.1.5）：

 复制代码

```
1 $ wget https://www.keepalived.org/software/keepalived-2.1.5.tar.gz
```

第二步，安装 Keepalived：

 复制代码

```
1 $ sudo yum -y install openssl-devel # keepalived依赖OpenSSL，先安装依赖
2 $ tar -xvzf keepalived-2.1.5.tar.gz
3 $ cd keepalived-2.1.5
4 $ ./configure --prefix=/usr/local/keepalived
5 $ make
6 $ sudo make install
```

第三步，配置 Keepalived：


 复制代码

```
1 $ sudo mkdir /etc/keepalived # 安装后，默认没有创建/etc/keepalived目录
2 $ sudo cp /usr/local/keepalived/etc/keepalived/keepalived.conf /etc/keepalived
3 $ sudo cp /usr/local/keepalived/etc/sysconfig/keepalived /etc/sysconfig/keepal
```

Keepalived 的 systemd unit 配置，默认使用


了/usr/local/keepalived/etc/sysconfig/keepalived作为其

EnvironmentFile，我们还需要把它修改为/etc/sysconfig/keepalived文件。编辑/lib/systemd/system/keepalived.service文件，设置EnvironmentFile，值如下：

 复制代码

```
1 EnvironmentFile=-/etc/sysconfig/keepalived
```

第四步，启动 Keepalived，并设置开机启动：

 复制代码

```
1 $ sudo systemctl start keepalived
2 $ sudo systemctl enable keepalived
```

这里要注意，Keepalived 启动时不会校验配置文件是否正确，所以我们要小心修改配置，防止出现意想不到的问题。

第五步，查看 Keepalived 的启动状态：

 复制代码

```
1 $ systemctl status keepalived
```

输出中有active (running)字符串，说明成功启动。Keepalived 的日志保存在/var/log/messages中，你有需要的话可以查看。

Keepalived 配置文件解析

Keepalived 的默认配置文件为/etc/keepalived/keepalived.conf，下面是一个 Keepalived 配置：

 复制代码

```
1 # 全局定义，定义全局的配置选项
2 global_defs {
3 # 指定keepalived在发生切换操作时发送email，发送给哪些email
4 # 建议在keepalived_notify.sh中发送邮件
5     notification_email {
6         acassen@firewall.loc
7     }
8     notification_email_from Alexandre.Cassen@firewall.loc # 发送email时邮件源地址
9     smtp_server 192.168.200.1 # 发送email时smtp服务器地址
10    smtp_connect_timeout 30 # 连接smtp的超时时间
11    router_id VM-4-21-centos # 机器标识，通常可以设置为hostname
12    vrrp_skip_check_adv_addr # 如果接收到的报文和上一个报文来自同一个路由器，则不执行检
13    vrrp_garp_interval 0 # 单位秒，在一个网卡上每组gratuitous arp消息之间的延迟时间，
14    vrrp_gna_interval 0 # 单位秒，在一个网卡上每组na消息之间的延迟时间，默认为0
15 }
16 # 检测脚本配置
17 vrrp_script checkhaproxy
18 {
19     script "/etc/keepalived/check_nginx.sh" # 检测脚本路径
20     interval 5 # 检测时间间隔（秒）
```



```

21     weight 0 # 根据该权重改变priority，当值为0时，不改变实例的优先级
22 }
23 # VRRP实例配置
24 vrrp_instance VI_1 {
25     state BACKUP # 设置初始状态为'备份'
26     interface eth0 # 设置绑定VIP的网卡，例如eth0
27     virtual_router_id 51 # 配置集群VRID，互为主备的VRID需要是相同的值
28     nopreempt # 设置非抢占模式，只能设置在state为backup的节点上
29     priority 50 # 设置优先级，值范围0~254，值越大优先级越高，最高的为master
30     advert_int 1 # 组播信息发送时间间隔，两个节点必须设置一样，默认为1秒
31 # 验证信息，两个节点必须一致
32     authentication {
33         auth_type PASS # 认证方式，可以是PASS或AH两种认证方式
34         auth_pass 1111 # 认证密码
35     }
36     unicast_src_ip 10.0.4.21 # 设置本机内网IP地址
37     unicast_peer {
38         10.0.4.20 # 对端设备的IP地址
39     }
40 # VIP，当state为master时添加，当state为backup时删除
41     virtual_ipaddress {
42         10.0.4.99 # 设置高可用虚拟VIP，如果是腾讯云的CVM，需要填写控制台申请到的HAVIP地址。
43     }
44     notify_master "/etc/keepalived/keepalived_notify.sh MASTER" # 当切换到master状态
45     notify_backup "/etc/keepalived/keepalived_notify.sh BACKUP" # 当切换到backup状态
46     notify_fault "/etc/keepalived/keepalived_notify.sh FAULT" # 当切换到fault状态
47     notify_stop "/etc/keepalived/keepalived_notify.sh STOP" # 当切换到stop状态时
48     garp_master_delay 1 # 设置当切为主状态后多久更新ARP缓存
49     garp_master_refresh 5 # 设置主节点发送ARP报文的时间间隔
50     # 跟踪接口，里面任意一块网卡出现问题，都会进入故障(Fault)状态
51     track_interface {
52         eth0
53     }
54 # 要执行的检查脚本
55     track_script {
56         checkhaproxy
57     }
58 }

```

这里解析下配置文件，大致分为下面 4 个部分。

`global_defs`：全局定义，定义全局的配置选项。


`vrrp_script checkhaproxy`：检测脚本配置。

`vrrp_instance VI_1`：VRRP 实例配置。

`virtual_server` : LVS 配置。如果没有配置 LVS+Keepalived，就不用设置这个选项。这门课中，我们使用 Nginx 代替 LVS，所以无需配置 `virtual_server`（配置示例中不再展示）。

只有在网络故障或者自身出问题时，Keepalived 才会进行 VIP 切换。但实际生产环境中，我们往往使用 Keepalived 来监控其他进程，当业务进程出故障时切换 VIP，从而保障业务进程的高可用。

为了让 Keepalived 感知到 Nginx 的运行状况，我们需要指定 `vrrp_script` 脚本，`vrrp_script` 脚本可以根据退出码，判断 Nginx 进程是否正常，0 正常，非 0 不正常。当不正常时，Keepalived 会进行 VIP 切换。为了实现业务进程的监控，我们需要设置 `vrrp_script` 和 `track_script`：

 复制代码

```
1 vrrp_script checkhaproxy
2 {
3     script "/etc/keepalived/check_nginx.sh"
4     interval 3
5     weight -20
6 }
7
8 vrrp_instance test
9 {
10     ...
11     track_script
12     {
13         checkhaproxy
14     }
15     ...
16 }
```

这里，我介绍下上面配置中的一些配置项。

`script` : 指定脚本路径。

`interval` : 表示 Keepalived 执行脚本的时间间隔（秒）。

`weight` : 检测权重，可以改变 `priority` 的值。例如，-20 表示检测失败时，优先级 -20，成功时不变。20 表示检测成功时，优先级 +20，失败时不变。

总结

今天我主要讲了跟 IAM 部署相关的两个组件，Nginx + Keepalived 的相关功能。

我们可以基于物理机 / 虚拟机来部署 IAM 应用，在部署 IAM 应用时，需要确保整个应用具备高可用和弹性扩缩容能力。你可以通过 Nginx 的反向代理功能和负载均衡功能实现后端服务 iam-apiserver 和 iam-authz-server 的高可用，通过 Keepalived 实现 Nginx 的高可用，通过 Nginx + Keepalived 组合，来实现 IAM 应用的高可用和弹性伸缩能力。

课后练习

1. Keepalived 的主备服务器要接在同一个交换机上。思考下，如果交换机故障，如何实现整个系统的高可用？
2. iam-pump 是有状态的服务，思考下，如何实现 iam-pump 的高可用？

欢迎你在留言区与我交流讨论，我们下一讲见。

分享给需要的人，Ta 订阅后你可得 **24 元现金奖励**

👍 赞 1 💡 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 39 | 性能分析（下）：API Server 性能测试和调优实战

下一篇 特别放送 | 给你一份清晰、可直接套用的 Go 编码规范

专栏上新

陈天 · Rust 编程第一课

实战驱动，快速上手 Rust

早鸟优惠 **¥99** 原价¥129



陈天
Tubi TV 研发副总裁

精选留言 (5)

[写留言](#)**Realm**

2021-08-26

1 可以把交换机搞成堆叠模式，服务器分别接在两个不同的交换机上，可以减少单点故障。可以看看这篇文章<https://blog.51cto.com/netlt/2589364>

2 有状态一般是cookie或者session，负载均衡支持基于cookie的会话保持，开启后，相同cookie的http请求，始终打到某一个服务端...

[展开](#)

3

**kkgo**

2021-08-27

怎么做nginx负载均衡？

[展开](#)**daz2yy**

2021-08-27

Nginx 确保服务的高可用；

Keepalived 确保 Nginx 的高可用；

谁来保证 Keepalived 的高可用呢？感觉如果能确保 Keepalived 的高可用，那么去掉这一层，把保证 Keepalived 高可用的方式应用到保证 Nginx 的高可用上不是简单一些吗？

[展开](#)**XI 酥**

2021-08-26

老师，单台gin框架应该起多少goroutine性能才是最优的啊，100个1000个？，单台服务器起能抗住多少的并发啊？

[展开](#)**魔**

2021-08-26

对于如果交换机故障，如何实现整个系统的高可用？

其一采用备份交换机，当云解析到主交换机不同之后，把请求路由到备份交换机上，主交

换机和备份交换机都指向同一后台服务器

...

展开

