面试前总结

**服务器安装配置**

1.精简开机系统自启动 chkconfig

2.更改ssh服务器端远程登入配置，修改端口，不让root登录，sodu权限的给予。

3.调整linux系统的资源限制。用户的内存使用限制和进程个数，当系统受到攻击之后保障最小损失。

4.隐藏linux版本的信息

5.密码的安全策略。16位数字，随机生产，定时更换密码

6.清除/etc/passwd中的不必要的系统虚拟账号。

7..禁止linux系统被ping

8.linux系统最小化原则。安装服务，命令，用户，权限，授权

**mysql数据库**：

安装，创建索引，慢查询优化，高可用，主从复制，读写分离，分库分表。

①：配置文件的优化 ②：SQL语句的优化 ③：表结构的优化 ④：索引优化 ⑤：系统资源优化

读写分离：① mysql proxy ③ mycat ④ 程序代码实现

redis数据库：

支持持久化（将内存数据保存到磁盘），集群主从，

持久化：①半持久化RDB模式，使用快照备份。系统默认 ② 全持久化AOF模式

sentinel哨兵：redis自带的高可用==redis+keepalived

**zabbix监控：**

安装，自动发现，shell脚本安装agent端，邮件告警，企业微信告警，自定义脚本监控zabbix\_get获取脚本值，自定义监控mysql的主从，zabbix的分布式zabbix\_proxy。

zabbix使用自带的jmx模板对java程序进行监控：**监控端口，JVM，cpu，内存**

zabbix监控nginx：**需要在nginx的配置加上状态配置，然后使用zabbix\_get去获取nginx的连接上，处理数，拒绝数登信息。cpu，内存**

zabbix使用自带的mysql模板监控mysql：**语句的吞吐量，慢查询数量，连接数，cpu,内存**

**nginx服务：**

使用epoll异步非阻塞模型，kill -HUP pid 平滑重启；虚拟主机同端口不同域名，location的优先级，nginx的动静分离+静态缓存expires，upstram负载均衡+proxy\_pass反向代理，nginx的防盗链（照片，文件设置403返回），nginx的日志切割，nginx的https。 Varnish：web加速器

nginx负载均衡：7层负载。只能负载http服务

haproxy负载均衡：4,7层负载，效率比nginx高。可以负载mysql

LVS负载均衡：4层负载，效率比nginx高。可以负载mysql

LVS负载均衡：

①NET模式：调度器将请求报文的目的IP改成后端服务器的IP，服务端将数据返回给调度器--》用户，所以调度器存在瓶颈。

②DR模式：调度器将请求报文的目的IP改成后端服务器的MAC地址，服务器端直接返回数据给客户。

③TUN模式：属于加密的DR模式。

**linux性能优化**：

文件打开数，内核参数优化(优化TCP/IP【time\_wait，syn长度)，

vmstat，sar，iostat：检测是否为cpu瓶颈 <70% good

free，vmstat：检测是否为内存瓶颈 20% < free/total < 70% good

iostat：检测是否是磁盘I/O瓶颈 iowait <20

netstat，iftop：检测是否为网络瓶颈

**数据备份问题**：

100M以下使用mysqldump备份，会锁表。

100M以上使用xtrabackup备份

xtrabackup增量备份（周天进行全量备份，1-6增量备份）

**shell编程：**

先写需求文档，再根据方法写shell脚本

**自动化运维：**

1.平台 2. 告警 3. 维护日志 4. 优先级

**自动化运维工具：**

1.监控自动化：zabbix，普罗米修斯

2.配置变更检测自动化：ansible,pupet

3.提醒自动化：zabbix告警

4. 监控检测自动化

5. 维护报个生成自动化

**ansible工具：**

1. 使用模块：ping，shell，command，copy，yum，cron，user，service，playbook

2. 新能优化：因为ansible使用的是ssh通信，即优化ssh即可，①关闭秘钥检测 ②：关闭DNS的反向解析 ③：关闭facts缓存

**jenkins工具**：

CI/CD（持续集成/持续部署），Performance Plugin测试插件

jenkisn常用插件：git，maven，ssh，Git Parameter(git切换分支)，Role Strategy Plugin(权限管理)

jenkins的邮件通知。jenkins分布式，增加节点。jenkins+ansible高并发构建。

**高可用：**

keepalived+nginx主备：master端的vip接收用户请求，权重高；backup端权重低，一样的 VIP。当master端出现故障，脚本停掉master端的keepalived，VIP切换到backup端。

keepalived+nginx双主：master1和master2的keepalived均配置2个VIP，权重不一样。

当master1出现故障，脚本会停掉master1的keepalived，切换master1的VIP到master2。即master2用于2个VIP同时提供服务。

keepalived+redis主备，redis主备+sentinel哨兵：redis的高可用

自己搭建mysql：keepalived+mysql+mycat/mysql proxy 实现数据库的主从高可用读写分离

**docker：**

cgroups(资源管理)，namespace，rootfs(文件系统)，C/S模式，私有harbor仓库

传统虚拟机：硬件虚拟化操作系统，操作系统安装应用程序。

docker容器：硬件主机实现应用的虚拟化。

docker的特点：启动快，轻量级虚拟化，快速的交付，部署和扩容。

docker默认使用桥接网络，通过端口映射就行访问。

**kubernetes：服务和组件**

master节点：etcd(数据存储)，apiserver(通信接口)，controller manager(扩展控制)，scheduler(pod调度与master通信)

node节点：kubelet(pod的生命周期)，kube-proxy(负载和服务发现)

组件：pod(最小运行单位)，lable(标签)，namespace(命名空间)，RC(控制pod数量)，RS(控制pod数量)，Daemon Set(DS控制每个node运行指定的pod)，service(pod的虚拟映射)，endpoint(pod与外部通信)，volume(挂载)，deployment+RS+set image(k8s的重新发布)，CronJob(定时任务)