

1. NAT 和 PAT 的区别

IP 地址耗尽促成了 CIDR 的开发 但是 CIDR 开发的主要目的是为了有效的使用现有的 Internet 地址

根据 RFC1631 开发的 NAT 却可以在多重的 Internet 子网中使用相同的 IP 地址 用来减少注册 IP 地址的使用

NAT 分为: 静态 NAT 动态 NAT 端口 NAT

静态 NAT: 内部网络中的每个主机都被永久的映射成外部网络中的某个合法地址

动态 NAT: 在外部网络中定义了一系列的合法地址 采用动态分配的方法映射到内部网络

PAT: 是人们比较熟悉的一种转换方式 PAT 普遍应用接入设备中 它可以将中小型的网络隐藏在合法的 IP 地址后面

PAT 它将内部连接映射到外部网络中的一个单独的 IP 地址上 同时在该地址上加上一个由 NAT 设备选定的 TCP 端口号 也就是采用 portmultiplexing 技术 或改变外出数据的源 port 的技术将多个内部 IP 地址映射到同一个外部地址

(网络地址转换是一个 Internet 工程任务组标准 用于允许专用网络上的多台 PC 共享单个或全局路由的 IPv4 地址)

IPv4 地址不足是经常部署 NAT 的一个主要原因 WindowsXP 和 WindowsMe 中的"Internet 连接共享"及许多 Internet 网关设备都使用 NAT, 尤其是在通过 DSL 或电缆调制解调器连接宽的情况下

NAT 对于解决 IPv4 地址耗费问题尽管很有效 但毕竟属于临时性的解决方案 这种 IPv4 地址占用问题在亚洲及世界其他一些地方已比较严重 但日渐成为北美地区需要关注的问题这就是人们为什么长久以来一直关注使用 IPv6 来克服这个问题的原因所在 除了减少所需的 IPv4 地址外 由于专用网络之外的所有主机都通过一个共享的 IP 地址来监控通信 因此 NAT 还为专用网络提供了一个隐匿层 NAT 与防火墙或代理服务器不同 但它确实有利于安全)

2. NAT 使用的几种情况

1. 连接到 Internet 但却没有足够的合法地址分配给内部主机
2. 更改到一个需要重新分配地址的 ISP
3. 有相同的 IP 地址的两个 internet 合并
4. 想支持负载均衡

3. NAT 四种用法

1. translation inside local address 将内部地址一对一的转换成外部地址
2. over loading inside global address 将内部地址多对一转换成外部地址 通过地址加端口号的方式区别不同的本地地址
3. tcp load distributing 提供多个或利用率高主机之间进行负载分担的方法
4. handling over lapping network NAT 地址转化是指每个内网地址都被转换成 IP 地址+源端口的形式 这需要公网 IP 地址为多个

PAT 由于 IP 地址不足 就会出现内网地址被转换成 IP 地址+端口的形式 这样的公网 IP 地址通常是一个

5. 若一台计算机的内存为 128M 则交换分区的大小是多少
256M

6. 软链接和硬链接的区别

硬链接不会创建新的 **inode** 只是给原文件多加了一个文件名 不能跨分区 删除原文件后 另一个文件还能使用

软链接创建新的 **inode** 相当于重新创建了一个文件 可以跨分区 删除原文件后 另一个文件不能再使用

7. ps aux 中的 VSZ 代表什么意思 RSS 代表什么意思

VSZ 虚拟内存集 进程占用的虚拟内存空间

RSS 物理内存集 进程占用实际物理内存空间

8. 保存当前磁盘分区的分区表 dd 命令是个强大的命令 在复制的同时进行转换 请写出具体的命令

`dd if=/dev/sda of=./mbr.txt bs=1M count=512`

9. 6 块 300G 的硬盘做 raid5 新的设备容量是多大

由于 **raid5** 磁盘阵列兼顾了安全性和速度 所以有一块磁盘要被用作安全备份区

6 块 300G 的磁盘实际使用到的只有 5 块 $5 \times 300G = 1500G$

10. 显示网络接口常见命令 查看服务端口的命令

显示网络接口 `ifconfig`

查看服务端口 `netstat`

11. 简述描述 linux 系统下源代码编译方式安装软件的大致步骤

1. 部署基础环境: 防火墙 关闭 **selinux** 安装编译器
2. 下载 解压软件包
3. 配置安装路径
4. 编译
5. 安装

12. 实时抓取并显示当前系统中 tcp80 端口的网络数据信息 请写出完整操作命令

`tcpdump -n tcp port 80 (tcpdump -i ens33 'tcp port 80')`

13. kernel 调参

1. `/proc/sys/net/core/rmem_max` 最大的 TCP 数据接收缓冲
2. `/proc/sys/net/core/wmem_max` 最大的 TCP 数据发送缓冲
3. `/proc/sys/net/ipv4/tcp_timestamps` 时间戳在(请参考 RFC1323)TCP 的包头增加 12 个字节
4. `/proc/sys/net/ipv4/tcp_sack` 有选择的应答
5. `/proc/sys/net/ipv4/tcp_window_scaling` 支持更大的 TCP 窗口.如果 TCP 窗口最大超过 65535(64KB) 必须设置该数值为 1
6. `rmem_default` 默认接收窗口大小
7. `rmem_max` 接收窗口的最大大小
8. `wmem_default` 默认发送窗口大小
9. `wmem_max` 发送窗口的最大大小

14. Linux (CentOS7)中的服务自启动添加方法有哪些

1.systemctl enable 服务名

2.vim/etc/rc.localsystemctlstart 服务名

3.vim/etc/fstabssystemctlstart 服务名

15. 请解释下 Telnet 和 SSH 的区别

Telnet: 不安全 没有对传输的数据进行加密 容易被监听还有遭受中间人攻击 telnet 不能压缩传输数据 所以传输慢

ssh: 对数据进行了 RSA 加密 安全性高 ssh 传输数据是经过压缩的 所以数据传输速度比较快

16. 进程和线程的区别

线程: 线程来源与进程 线程模式需要事先启

线程模式并发量高 相对与进程模式来说没有进程模式稳定 共享内存 不支持 php

进程: 进程稳定性高 线程稳定性低

进程独占内存 线程共享进程的内存一个进程可以有多个线程

17. dhcp 客户端是哪个地址来申请一个新的 IP 地址

dhcp 客户端是用使用广播地址来向全网广播发送广播包来寻找 dhcp 服务器的

源地址是 0.0.0.0 广播地址是 255.255.255.255

客户端没有 IP 地址才需要申请 所以它只能用 0.0.0.0 来代替它的 IP 向服务器申请 IP 地址

18. Mysql 优化做了哪些

MySQL 内核优化

MySQL 配置参数优化 此优化需要进行压力测试来进行参数调整

对 SQL 语句以及表优化

19. 主从同步 主服务器宕机怎么处理

可用 keepalived 做主从 主服务器宕机可用切到优先级高的服务器 让优先级高的服务器做主服务器

20. 主从同步异常怎么解决

设置 MySQL 双一模式, 将需要同步的数据保存到磁盘中

如果 innodb_flush_log_at_trx_commit 设置为 0

log buffer 将每秒一次地写入 log file 中 并且 log file 的 flush(刷到磁盘)操作同时进行该模式下在事务提交的时候 不会主动触发写入磁盘的操作

如果 innodb_flush_log_at_trx_commit 设置为 1

每次事务提交时 MySQL 都会把 log buffer 的数据写入 log file 并且 flush(刷到磁盘)中去

如果 innodb_flush_log_at_trx_commit 设置为 2

每次事务提交时 MySQL 都会把 log buffer 的数据写入 log file.但是 flush(刷到磁盘)操作并不会

同时进行 该模式下,MySQL 会每秒执行一次 flush(刷到磁盘)操作

sync_binlog 的默认值是 0

像操作系统刷其他文件的机制一样 MySQL 不会同步到磁盘中去而是依赖操作系统来刷新 binary log

当 sync_binlog =N (N>0)

MySQL 在每写 N 次 二进制日志 binary log 时 会使用 fdatsync()函数将它的写二进制日志 binary log 同步到磁盘中去

21. MySQL 用了什么引擎

Innodb 引擎 MyIASM 引擎

22. 主从原理

从库生成两个线程 I/O 线程 SQL 线程

I/O 线程 请求主库的 binlog 并将得到的 binlog 日志写到 relay log 文件中 主库会生成一个 log dump 线程 用来给从库 I/O 线程传 binlog

SQL 线程 会读取 relay log 文件中的日志 并解析成具体操作 来实现主从的操作一直 而最终影响数据一致性

23. 我们都知道 DNS 既采用了 TCP 协议,又采用了 UDP 协议,什么时候采用 TCP 协议? 什么时候采用 UDP 协议? 为什么要这么设计

dns 有两种情况 一种是区域传输 一种是域名解析

1.区域传输时 一个区中主 DNS 服务器从自己本机的数据文件中读取该区的 DNS 数据信息 而辅助 DNS 服务器则从区的主 DNS 服务器中读取该区的 DNS 数据信息 传输协议是 TCP

2.域名解析时 使用 UDP 传输 不用经过 TCP 三次握手 这样 DNS 服务器负载更低 响应更快 但是当域名解析的反馈报文长度超多 512 字节时 将不能使用 UDP 协议进行解析 此时必须使用 TCP

24. 我们都知道 FTP 协议有两种工作模式 说明他们的大概的一个工作流程

模式: 主动模式 (ActiveFTP) 和被动模式 (PassiveFTP)

1、在主动模式下: FTP 客户端随机开启一个大于 1024 的端口 N 向服务器的 21 号端口发起连接 然后开放 N+1 号端口进行监听 并向服务器发出 PORTN+1 命令 服务器接收到命令后会用其本地的 FTP 数据端口 (通常是 20) 来连接客户端指定的端口 N+1, 进行数据传输 客户端使用 1024 以上的端口连接服务器的 21 号端口,并且会事先开启一个 1024+N 的端口准备让服务器来连接,服务器得知客户端准备的端口后,用 20 号端口与客户端连接 创建数据链路时 server 端处于主动去连

接客户端的状态固称为主动传输模式

2、在被动模式下: FTP 客户端随机开启一个大于 1024 的端口 N 向服务器的 21 号端口发起连接 同时会开启 N+1 号端口。然后向服务器发送 PASV 命令 通知服务器自己处于被动模式 服务器收到命令后 会开放一个大于 1024 的端口 P 进行监听 然后用 PORTP 命令通知客户端自己的数据端口是 P 客户端 收到命令后 会通过 N+1 号端口连接服务器的端口 P 然后在两个端口之间进行数据传输 客户端使用 1024 以上的端口连接服务器的 21 号端口,服务器接到连接请求后,开启一个 1024+M 的端口等待客户端来连接,再通知客户端这个等待连接的端口

号是 1024+M,客户端接到服务器端的信息后 开启新的一个 1024+N 的端口与 1024+M 进行连接,创建数据连接时,server 端处于被客户端连接的状态,固称为被动传输模式 vsftp 默认是根据客户端不同而使用不同模式,windows 客户端则使用主动,linux 客户端则使用被动可以用 pasv_enable=NO 来强制使用主动

总的来说

主动模式的 FTP 是指服务器主动连接客户端的数据端口

被动模式的 FTP 是指服务器被动地等待客户端连接自己的数据端口

25. keepalived 的工作原理是什么，使用的是什麼协议？

keepalived 是以 VRRP 协议为实现基础的，VRRP 全称 Virtual Router Redundancy Protocol，即虚拟路由冗余协议。虚拟路由冗余协议，可以认为是实现路由器高可用的协议，即将 N 台提供相同功能的路由器组成一个路由器组，这个组里面有一个 master 和多个 backup，master 上面有一个对外提供服务的 vip（该路由器所在局域网内其他机器的默认路由为该 vip），master 会发组播，当 backup 收不到 vrrp 包时就认为 master 宕掉了，这时就需要根据 vrrp 的优先级来选举一个 backup 当 master。这样的话就可以保证路由器的高可用了。

26. nginx 的七层代理有什么好处？

请求如果直接发到同步处理的后端，那么从收到请求到把响应发出去的这段时间中，一个进程的资源就被占用了。在慢链接的情况下，这个进程除了处理之外，大多数时间基本耗费在等待上，而 nginx 有异步非阻塞模型，它可以通过基于事件的方式同时处理和维持多个请求，而后端只需要做逻辑计算，节约了等待时间去处理更多的请求。动态页面的 IO 性能不好，nginx 可以将请求缓存下来，再将完整的请求转发给后端服务器做处理，减少后端服务器的等待时间。

27. nginx 代理负载均衡的调度算法有哪些 具体实现时的现象是什么

1.轮询（默认）：每个请求按时间顺序逐一分配到不同的后端，如果后台某台服务器宕机，自动剔除故障系统，使用户访问不受影响，这种方式简便，成本低，但是可靠性低，负载均衡不均衡，适用于图片服务器集群和纯静态页面服务器集群。

2.weight（权重）：weight 的值越大分配到访问概率越高，主要用于后端每台服务器性能不均衡的情况下，或者仅仅为在主从的情况下设置不同的权值，达到合理有效的利用主机资源。

3.IP_HASH（访问 IP）：每个请求按访问的哈希结果分配，使来自同一个 IP 的访问固定一台后端服务器，并且可以有效解决动态网页存在的 session 的共享问题。

4.FAIR（第三方）：比 weight、ip_hash 更加智能的负载均衡算法，fair 算法可以根据页面大小和加载时间长短智能的进行均衡负载，也就是根据后端服务器的响应时间来分配请求，响应时间短的优先分配。nginx 本身不支持 fair，如果需要这种调度算法，则需要安装 upstream_fair 模块。

5.URL_HASH（第三方）：按访问的 URL 的哈希结果来分配请求，使每个 URL 定向到一台后端服务器，可以进一步提高后端缓存服务器的效率。这种调度算法需要安装 nginx 的 hash 软件包

28. nginx upstream 的容错机制是怎么样的（有台机器挂掉了，nginx 怎么处理的）

nginx 收到客户端的请求，将请求根据调度算法转发给后台服务器，后台服务器访问被拒绝或者返回错误信息。nginx 将其暂停一段时间，在这段时间内不再将请求转发给该服务器，并将请求转发给后台的其他服务器响应

29.LVS 的三种工作模式是什么？分别是如何工作的

NAT 模式（VS-NAT）：LVS 将客户端发来的数据包的目的地址转换成其中一台 RS（real server）的 IP 地址，由 RS 处理数据并返回给 LVS，LVS 再把数据包的源 IP 改为自己的 IP，目的地址 IP 改为客户端的 IP 地址发送给客户端。

IP 隧道模式（VS-TUN）：将客户端发来的数据包封装一个新的目的 IP 头标记，通过 IP 隧道转发给 RS，RS 收到后，先把数据包的头解开，还原数据包，处理后直接返回给客户端，不需要再经过 LVS。

DR 模式（VS-DR）：客户端发送请求到 VIP，LVS 将请求报文的目标 MAC 地址改为 RS 的 MAC 地址，将请求转发给 RS，而 RS 响应后的处理结果直接返回给客户端。

29. LVS nginx haproxy 的优缺点

LVS 优点：具有很好的可伸缩性、可靠性、可管理性。抗负载能力强、对内存和 CPU 资源消耗比较低。工作在四层上，仅作分发，所以它几乎可以对所有的应用做负载均衡，且没有流量的产生，不会受到大流量的影响。

缺点：软件不支持正则表达式处理，不能做动静分离，如果 web 应用比较庞大，LVS/DR+KEEPALIVED 实施和管理比较复杂。相对而言，nginx 和 haproxy 就简单得多

nginx 优点：工作在七层之上，可以针对 http 应用做一些分流的策略。比如针对域名、目录结构。它的正则规则比 haproxy 更为强大和灵活。对网络稳定性依赖非常小。理论上能 PING 就能进行负载均衡。配置和测试简单，可以承担高负载压力且稳定。nginx 可以通过端口检测到服务器内部的故障。比如根据服务器处理网页返回的状态码、超时等。并且可以将返回错误的请求重新发送给另一个节点，同时 nginx 不仅仅是负载均衡器/反向代理软件。同时也是功能强大的 web 服务器，可以作为中层反向代理、静态网页和图片服务器使用

缺点：不支持 URL 检测，仅支持 HTTP 和 EMAIL，对 session 的保持，cookie 的引导能力相对欠缺

Haproxy 优点：支持虚拟主机、session 的保持、cookie 的引导；同时支持通过获取指定的 url 来检测后端服务器的状态。支持 TCP 协议的负载均衡；单纯从效率上讲比 nginx 更出色，且负载均衡策略非常多

缺点：扩展性能差；添加新功能很费劲，对不断扩展的新业务很难对付

30.MySQL 的主从复制过程是同步的还是异步的？

主从复制的过程是异步的复制过程，主库完成写操作并计入 binlog 日志中，从库再通过请求主库的 binlog 日志写入 relay 中继日志中，最后再执行中继日志的 sql 语句。