nfs服务

作者：牟建波（1353429820@qq.com）

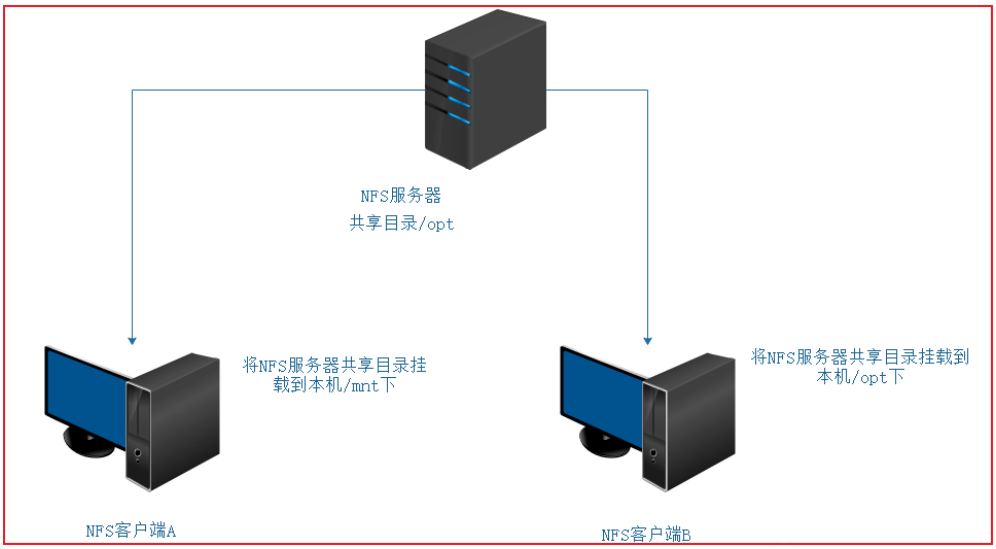
时间：2025-05-31

描述：日常自学笔记

1.NFS基本介绍

CentOS Stream 9中的NFS（网络文件系统）是一种允许不同计算机通过网络共享文件和目录的协议，使得远程主机能够像本地磁盘一样访问和操作文件

- 它基于客户端-服务器架构，NFS服务器将本地文件系统共享给客户端，客户端可以通过网络挂载这些共享目录，像访问本地文件一样进行操作
- NFS主要应用于Linux/Unix系统之间的文件共享，虽然也支持跨操作系统，但其兼容性一般较差



2.NFS相关配置

需求：在 `node2服务器` 中的根目录下，有一个 `/export/software` 目录，该目录下存储了大量的软件包，由于各个服务器都需要解压使用，为了节省空间，需要将此目录共享出来，提供给各个服务器进行使用，同时也方便各个服务器放置共享的安装包

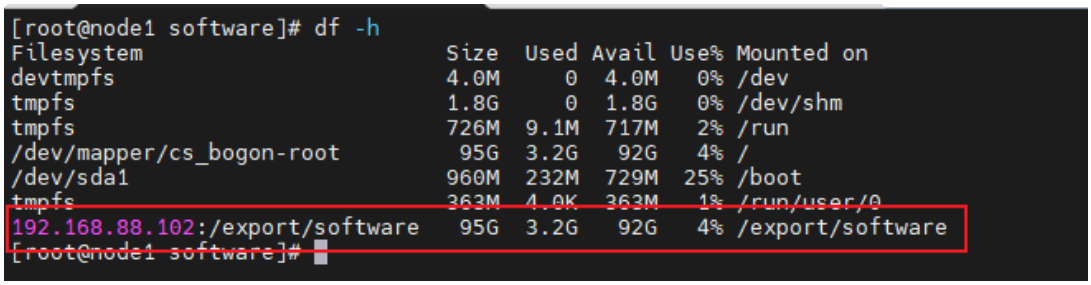
服务器规划：

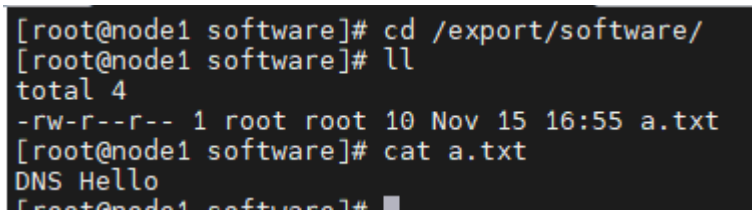
- node2：作为NFS服务器端
- node1：作为客户端，获取共享目录信息

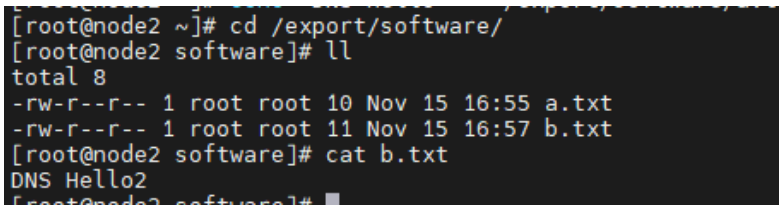
代码块

```
# 1.安装NFS服务器
dnf install -y nfs-utils     # 客户端、服务器端都按照
# 执行开放端口
firewall-cmd --zone=public --add-service=nfs --permanent
firewall-cmd --reload

# 2.在node2上创建共享目录
mkdir -p /export/software     # 创建需要共享的目录
vim /etc/exports              # NFS中设置指定需要共享的目录
# 添加以下内容
/export/software    *(rw,sync,no_root_squash)
# 参数说明
1./export/software：这个是要共享的目录路径
2.*：表示所有客户端都可以访问这个共享目录
3.rw：表示客户端可以对共享目录进行读写操作，默认是只读权限ro
4.sync：表示NFS服务器会同步写入数据。若选择async则为异步，先缓存在内存再同步磁盘
5.no_root_squash：NFS默认情况下，客户端root用户会被映射到一个普通用户root squashing，
目的是增强安全性，防止远程root用户拥有过高的权限。使用no_root_squash选项，不再进行映射，
意味着客户端的root用户可以直接访问共享目录中的所有文件，拥有与NFS服务器本地root相同的权限

# 3.服务启动并设置开机启动
systemctl start nfs-server
systemctl enable --now nfs-server

# 4.验证是否成功共享：服务端执行exportfs
[node2]
exportfs

# 5.挂载使用
[node1]
mkdir -p /export/software
mount -t nfs 192.168.88.102:/export/software /export/software
df -h
```

```
34
35  # 6.测试：服务器里添加内容，客户端能不能看见
36  [node2]
37  echo 'DNS Hello' >> /export/software/a.txt
38  [node1]
39  cat /export/software/a.txt
40
41  # 7.测试：客户端里添加内容，服务器能不能看见
42  [node1]
43  echo 'DNS Hello2' >> /export/software/b.txt
44  [node2]
45  cat /export/software/b.txt
```







3.NFS工作原理

在NFS早期版本中， 负责数据传输工作，主要采用rpcbind工具， 其底层采用的就是RPC远程过程调用协议

> RPC（Remote Procedure Call Protocol）： 远程过程调用协议， 它是一种通过网络从远程计算机程序上请求服务，不需要了解底层网络技术的协议

当安装NFS的时候, 由于该服务依赖了很多其他的服务,主要包括:

> rpcbind（RPC 服务绑定） TCP/UDP 111
>
> nfs-server（NFS 服务） TCP 2049（NFS 主要端口）
>
> mountd（NFS 挂载守护进程） 动态分配，通常是 20048/tcp 和 20048/udp
>
> statd（NFS 锁定守护进程）动态分配，通常是 TCP/UDP 32765-32766

问题一: RPC与NFS如何通讯呢?

代码块

```
答：客户端要与服务端进行数据传输，必须知道服务端 NFS 的具体端口号。然而，在 NFS 服务的早期版本（如 NFSv3），NFS 服务的一些组件（如 mountd，nlockmgr 等）使用动态分配的端口，导致客户端无法直接确定端口号。为了协调这些动态端口，RPC 框架通过 rpcbind 服务（监听在固定端口 111）实现以下功能：

动态端口管理：
RPC 统一管理服务端各 NFS 服务组件（mountd、nfsd 等）的动态端口，并将这些端口信息记录在 rpcbind 中。
当客户端发起请求时，客户端首先与 rpcbind 通信，查询目标 NFS 服务对应的实际端口号。

客户端与服务端的通信流程：
客户端向服务端的 rpcbind 服务（监听端口 111）发送查询请求，指定目标服务（如 mountd）。
rpcbind 返回目标服务的实际端口号。
客户端随后通过查询到的端口与 NFS 的具体服务进行通信，完成挂载、文件读写等操作。

固定端口的引入（在 NFSv4 中）：
NFSv4 对架构进行了简化，将所有功能整合在一个服务中，并固定使用 TCP 2049 端口。
客户端无需再依赖 rpcbind 动态查询端口，直接与服务端通信。
```

问题二：RPC是如何指定NFS服务的端口的呢?

代码块

```
1. 首先当NFS启动后，就会随机的使用一些端口，然后NFS就会向RPC去注册这些端口
2. RPC记录下这些端口，并且RPC会开启111端口，等待客户端RPC的请求
3. 客户端发送请求，那么服务器端的RPC就会将之前记录的NFS端口信息告知客户端
4. 客户端获取NFS服务器端的端口信息，就会以实际端口进行数据的传输了


补充说明
在 NFSv4 中，整个机制变得更加简单：

所有 NFS 功能整合到一个服务上，并使用固定的 TCP 2049 端口。
不再需要 rpcbind 管理动态端口，客户端可以直接与服务端通信，无需额外的端口查询步骤。
```