回顾

正则表达式

^ $ [ ] [^] . \* \{n,m\} \{n,\} \{n\} \(\)

+ ? ( ) {n,m} | \b

sed

sed 选项 定址符 指令 被处理的文档

选项 -n -r -i 指令 p d s/ / /

练习一：

1，删除文件中每行的第二个、最后一个字符

sed 's/.//2;s/.$//' test

2，将文件中每行的第一个、倒数第1个字符互换

sed -r 's/^(.)(.\*)(.)$/\3\2\1/' test 使用正则找到首字符、中间字符、尾字符，然后替换的时候使用粘贴调换首尾字符

()可以实现保留功能 相当于复制 ，粘贴使用\1 \2 ..

3，删除文件中所有的数字

sed -i 's/[0-9]//g' test 找到任意的数字，都替换成空

4，为文件中每个大写字母添加括号

sed -r 's/([A-Z])/(\1)/g' test 找到任意大写字母，复制，粘贴的时候套上括号

练习二：

1，编写脚本，实现部署ftp服务，并实现匿名上传文件

实现匿名上传文件到ftp服务器中可能被影响的因素

1. 配置文件
2. 目录权限
3. 防火墙
4. selinux

#!/bin/bash

yum -y install vsftpd 装包

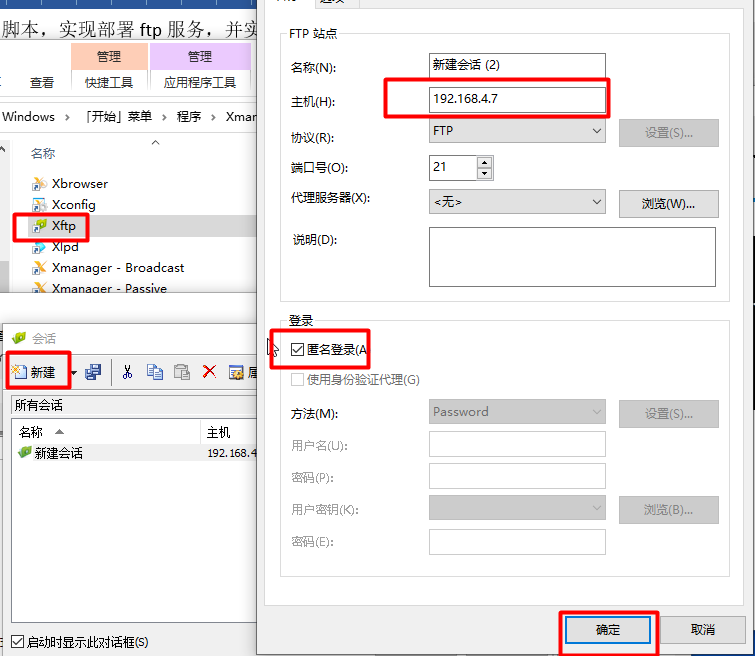
sed -i '/anon\_u/s/#//' /etc/vsftpd/vsftpd.conf 改配置，实现匿名上传文件

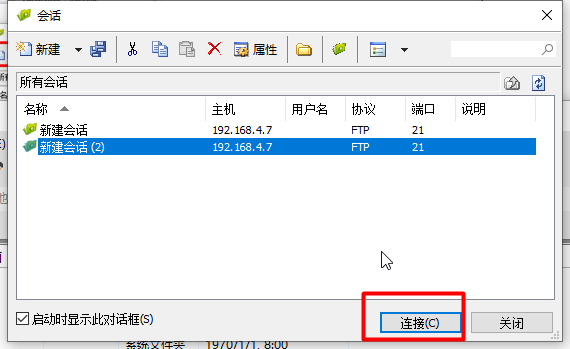
systemctl restart vsftpd 开服务

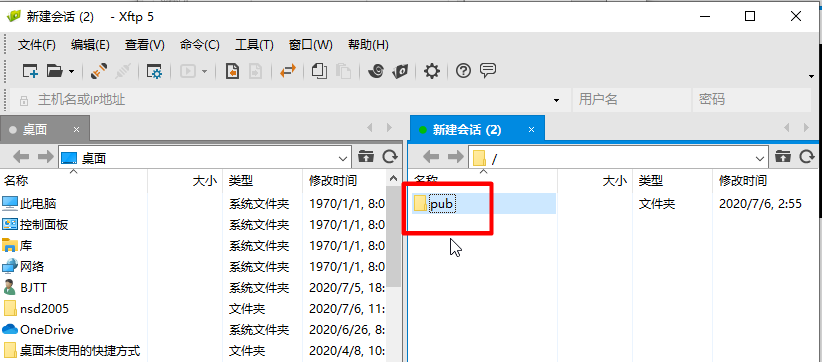
systemctl enable vsftpd 开机自启

chmod 777 /var/ftp/pub 放开目录权限

最后不要忘记关闭防火墙和selinux







双击进入pub目录，拖拽一个文本进去即成功

2，在sed 中使用a、i、c指令

a行下追加 i行上添加 c替换整行

sed 'a 666' test 在所有行下追加666

sed '1a 666' test 在第1行下追加666

sed '/ni/a 666' test 在有ni的行下追加666

sed 'i 666' test 在所有行上添加666

sed 'c 666' test 替换所有行为666

sed '1c 666' test 替换第1行为666

练习三：

找到系统中使用bash的账户，并按照“用户名 -- > 密码”的格式输出

#!/bin/bash

u=$(sed -n '/bash$/s/:.\*//p' /etc/passwd) 首先找到用户名

for i in $u 然后将用户交给for循环处理，

do

u1=$(grep $i /etc/shadow) 找到密码文件

u2=${u1#\*:} 掐头

u3=${u2%%:\*} 去尾

echo "$i --> $u3" 输出最终信息

done

grep 模糊搜索

sed 非交互式对文档进行修改

awk 精确搜索

练习四：使用awk

1，awk使用方式：

前置指令 | awk 选项 条件 指令

awk 选项 条件 指令 被处理文档

选项 -F 定义分隔符

指令 print

内置变量 $0 $1 $2 $3 … NR行号 NF列号

awk '{print}' test 输出所有行的内容

awk '{print $1}' test 输出所有行的第1列

awk '{print $2}' test 输出所有行的第2列

awk '{print $4}' test 输出所有行的第4列

awk '{print $0}' test 输出所有行的所有列

awk '{print $2,$0}' test 输出所有行的第2列，和所有列

awk '{print $2,$1}' test 输出所有行的第2列，第1列

awk '{print NR}' test 输出所有行的行号

awk '{print NF}' test 输出所有行都有几列

awk '/ni/{print}' test 找到有ni的行，并输出

awk '/ni/{print $1}' test 找到有ni的行，并输出第1列

awk '/ni/{print $2}' test 找到有ni的行，并输出第2列

head -5 /etc/passwd > user 创建素材

awk -F: '{print $1}' user 定义冒号作为分隔符，输出所有行的第1列

awk -F: '{print $3}' user 定义冒号作为分隔符，输出所有行的第2列

awk -F: '{print $1,$3}' user 定义冒号作为分隔符，输出所有行的第1列，第3列

awk -F[:/] '/root/{print $9}' user 使用冒号或斜线作为分隔符，输出有root的那行的第9列

awk -F: '/root/{print "abc"}' user 使用冒号作分隔符，找到有root的行，然后输出abc字符串，print输出的内容如果加了双引号，会被认为是常量

awk -F: '{print $1,$7}' user 输出第1列(用户名)，第7列(解释器)

awk -F: '{print $1,"的解释器是",$7}' user 常量加变量的混合输出

awk -F: '{print $1,"的ID号是",$3}' user

2，实际应用

提取网卡的流量信息

RX 接收流量 TX 发送流量

ifconfig eth0 | awk '/TX p/{print "eth0网卡 的发送流量是"$5"字节"}'

ifconfig eth0 | awk '/RX p/{print "eth0网卡 的接收流量是"$5"字节"}'

提取根分区剩余容量信息

df -h | awk '/\/$/{print "当前主机根分区剩余容量是"$4}'

提取安全日志信息，找出尝试登陆服务器的ip

awk '/Failed/{print $11}' /var/log/secure

练习五：awk处理时机

1，awk的任务安排

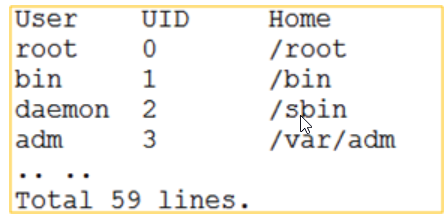
BEGIN{ } 执行1次

{ } 逐行 执行n次，n的次数由文档的行数决定

END{ } 执行1次

awk 'BEGIN{print "abc"}{print "abc"}END{print "abc"}' user

2，按类似格式输出信息：



tab 制表符,在一定程度上对文档进行排版, awk中使用"\t"

awk 'BEGIN{print "User\tUID\tHome"}' 表头

awk -F: '{print $1"\t"$3"\t"$6}' 内容

END{print "总计"NR"行"}' 结尾统计

awk -F: 'BEGIN{print "User\tUID\tHome"}{print $1"\t"$3"\t"$6}END{print "总计"NR"行"}' user 组合在一起实现信息的提取