Linux Shell

д-р Филип Андонов

26 септември 2022 г.

Съдържание

- Какво е Linux
- Какво е обвивка (shell)
- Команди
- Работа с файлове
- Работа с директории
- Работа с процеси
- Мрежови инструменти
- grep
- sed

Операционната система е набор от програми, които координират работата на компютърния хардуер и софтуер. Функциите, които осигурява, най-общо са:

- вход-изход
- команден интерпретатор
- Управление на файловете
- Среда за разработване на програми
- Управление на процеси
- Защита на информацията
- Комуникации
- Управление на потребителите

Вход-изход

Входно-изходните операции са съществени за работата на всеки компютър. Те позволяват на компютъра да съхранява и извлича данни от периферни устройства. Всяка ОС трябва да осигурява някаква форма на вход-изход.

Обвивка за работа с потребителя

Независимо дали е команден интерпретатор или графична обвивка, всяка ОС, която не е предназначена единствено за вградени системи, има механизъм за приемане на команди от потребителя, които да бъдат изпълнявани.

Управление на файловете

Управлението на файловете ви позволява да организирате данните във файлове и директории и да ги манипулирате на това ниво - създаване, изтриване, преместване, копиране, определяне на метаданни и други.

Среда за разработване на програми

Всяка ОС предлага програмен интерфейс, през който програмите да взаимодействат с нея.

Управление на процеси

Всяка ОС позволява изпълнението на програми. Независимо дали е еднозадачна или многозадачна, стартирането, управлението и спирането на процеси и съпътстващите действия са работа на ОС.

Защита на информацията

Защитата на една ОС касае много неща - паметта, файловете, достъпа до изпълние на команди и други.

Комуникации

Работа на всяка модерна ОС е да осигури способността на компютъра да комуникира с други устройства през различни интерфейси и към различни видове мрежи.

Управление на потребителите

Съвременните ОС осигуряват регистрация на потребители, които да имат права до различни ресурси на машината - памет, процесори, файлове, устройства, приложения.

BASH

- BASH(Bourne Again SHell) е една от многото съществуващи обвивки, които позволяват работа с ОС през команден интерпретатор.
- Написан като свободен заместител на стандартния Bourne Shell (/bin/sh)
- Той се казва така, защото оригинално е написан от Стив Борн за UNIX системи
- Съдържа всички функции на оригиналния шел плюс някои подобрения.
- Тъй като е свободен софтуер, се използва от повечето Линукс дистрибуции.

Команди

В Bash можем да извикваме командите като ги напишем и натиснем ENTER, или да им подаваме аргументи, за да конкретизираме какво да правят и върху какво. Обикновено синтаксисът на една команда може да съдържа следните спецификатори:
command [-argument] [-argument] [–argument] [file]

Получаване на помощ

Команда man Един от начините да получим информация за дадена команда е да прочетем нейните инструкции, наречени накратко man pages. За да го постигнем, просто извикваме командата man и ѝ подаваме като аргумент името на командата, за която искаме да научим повече.
Например: man ls
За да търсим за конкретна дума в инструкцията, използваме обратна наклонена черта. За да излезем от инструкцията, натискаме клавиша "Q".

Получаване на помощ

Понякога не знаем коя е командата, която ни трябва. Тогава можем да потърсим за дума в текстовото описание на инструкциите с ключ -k

man -k permission

Резултатът ще изглежда така:

access (2) - check user's permissions for a file

chmod (2) - change permissions of a file

Съдържание на директория

Команда ls – извежда съдържанието на директория.

опция	действие
-l	детайлно представяне на съдържанието
-a	не пропуска файлове, започващи с точка
-S	сортира записи по размер, най-големите първо
-r	сортира в обратен ред
-R	рекурсивно преминаване през под-директориите
-t	сортира по време на промяна, най-новите първо

Съдържание на файл

Команда cat - копира, сливайки, всеки файл или стандартния вход (означен с '-' и стойност по подразбиране) в стандартния изход. Използва се най-вече за бърз преглед на текстов файл. Също може да се използва за създаването на файл с пренасочване към файл.
cat > file

Изтриване

Команда rm - изтрива файл. Очаква аргумент името на файла за изтриване.

опция	действие
-i	пита преди да изтрие файл
-r, -R, –recursive	рекурсивно премахва директории

Копиране

Команда cp - копира файлове и директории
Синтаксис:
cp [опции] <източник> <дестинация>

опция	действие
-i	пита преди да запише върху съществуващ файл
-r, -R, –recursive	рекурсивно копира директории

Синоними

Команда ln - създава синоним на файл

Синтаксис:

ln OPTION... -T TARGET LINK_NAME - създава синоним на TARGET с име LINK_NAME
ln OPTION... TARGET - създава синоним на TARGET в текущата директория
ln OPTION... TARGET... DIRECTORY - създава псевдоними на всички TARGET в ДИРЕКТОРИЯ
ln OPTION... -t DIRECTORY TARGET... - създава псевдоними на всички TARGET в ДИРЕКТОРИЯ

По подразбиране са hard links. За да са символични, трябва да се използва –symbolic За hard links всяка TARGET трябва да съществува.

Синоними

Твърдите връзки (hard links) са копия на файла. Дори оригиналът да бъде изтрит, копието съдържа оригиналното съдържание.
Символните връзки са като указател към оригиналния файл. Ако оригиналът бъде изтрит, символната връзка сочи към несъществуващ файл.

Работа с директории

Команда cd - промяна на директория
Променя текущата работна директория с тази, подадена като аргумент

Работа с директории

Команда mkdir - създава директория.
С помощта на опцията -p или –parents създава и родителска директория.

Работа с директории

Команда rmdir - изтриване на директория.
Тази команда изтрива самата директория, ако тя няма съдържание.

Команда pwd - показва текущата директория.
Разпечатва пълното име на текущата работна директория.

Работа с процеси

С помощта на командата top можем да видим стартираните Линукс процеси и използваните ресурси. По подразбиране сортира по колона %CPU. Ако искаме да променим това, можем да използваме следните означения.

- P - сортира по %CPU
- M - сортира по %MEM
- N - сортира по PID
- T - сортира по TIME+

Команда ps

Показва информация относно активните процеси в момента. По подразбиране ps показва всички процеси с ID на потребителя като на текущия потребител и асоциирани със същия терминал като този, от който е извакана. Показва идентификатора на процеса PID, терминала, общото време процесора и времето на изпълнение. Резултатът не е сортиран. С опцията -A, която е еквивалентна на -e, можем да видим всички процеси.

Права за достъп до файловете

Всеки файл има набор от следните права.
R - права за четене
W - права за писане
X - права за изпълнение
Ако изпълним командата `ls -l` ще видим че те се повтарят три пъти. Това касае собственика, групата и всички други. Ако някой режим не е разрешен, на съответното място има знак за минус.

Права за достъп до файловете

Команда chmod - промяна на режимите за достъп.
Синтаксисът е chmod режим име_на_файл

собственик			група			други		
r	w	x	r	w	x	r	w	x
4			4			4		
	2			2			2	
		1			1			1

Права за достъп до файловете

Команда chown - смяна на собственика.
Синтаксисът е chown потребител файл.
Команда chgrp - смяна на групата.
Синтаксисът е chgrp група файл.

Търсене

Командата find претърсва поддърво от директории. Може да търси файлове и директории и да изпълнява действия върху тях. Позволява търсене по име, име на директория, дата създаване, дата на промяна, собственик и права за достъп. Синтаксисът е:
find [where to start][expression determines what to find] [-options] [what to find]
Например следния ред търси за пайтън файлове от текущата директория рекурсивно.
find ./ -name "*.py"

Търсене

Нека потърсим всички файлове с права за четене и писане на собственика и групата, и с права за четене за всички останали: find ./ -perm 664

Стандартен вход и стандартен изход

По подразбиране входа на UNIX/Linux командите е клавиатурата и изходът е екрана. Това всъщност са стойностите по подразбиране на стандартния вход и стандартния изход.

Пренасочване

Въпреки, че Linux изходът на командите е предназначен за стандартния изход, той лесно може да се пренасочи към файл. Например ако искаме да запишем всички текущо стартирани процеси във файл, достатъчно е да напишем
ps -A > processes

Пренасочване

Не бива да забравяме, че това пренасочване е деструктивно по отношение на файла. Ако такъв файл съществува, неговото съдържание ще бъда заменено с пренасочения изход.

```
ps -A > processes
echo overwrite > processes
cat processes
overwrite
```

Пренасочване

За да добавяме, вместо да презаписваме, можем да използваме $> >$.

```
date > current_date.txt
date "+%A»> current_date.txt
cat current_date.txt
пт май 20 14:34:44 EEST 2022
петък
```

Конвейер

Друг начин за манипулиране на входа и изхода е като вържем изходът на една програма да е входът на друга с помощта на конвейер.

```
cat BSD|wc -l
26
```

Променливи на средата

Обвивката позволява дефиниране на променливи и присвояването им на стойности. Името започва с буква(малка или главна), или със символ за подчертаване. В себе си може да съдържа цифри. Интервали не са разрешени. Следните са примери за валидни имена:

- i5
- length
- Input_file
- HOMEDIR
- _cflag

Променливи на средата

Следните не са валидни имена на променливи на обвивката.

- 5i
- .length
- file name
- променлива

Присвояването на стойност става като се изпише името на променливата, последвана от равенство, последвана от стойност, без интервали.
Length = 80

Променливи на средата

Можем да извеждаме стойности на променливите с командата echo. За да получим обаче достъп до стойността, трябва да напишем пред името ѝ знакът за долар.
echo PATH ще изведе низа PATH
echo $PATH ще изведе стойността на променливата PATH

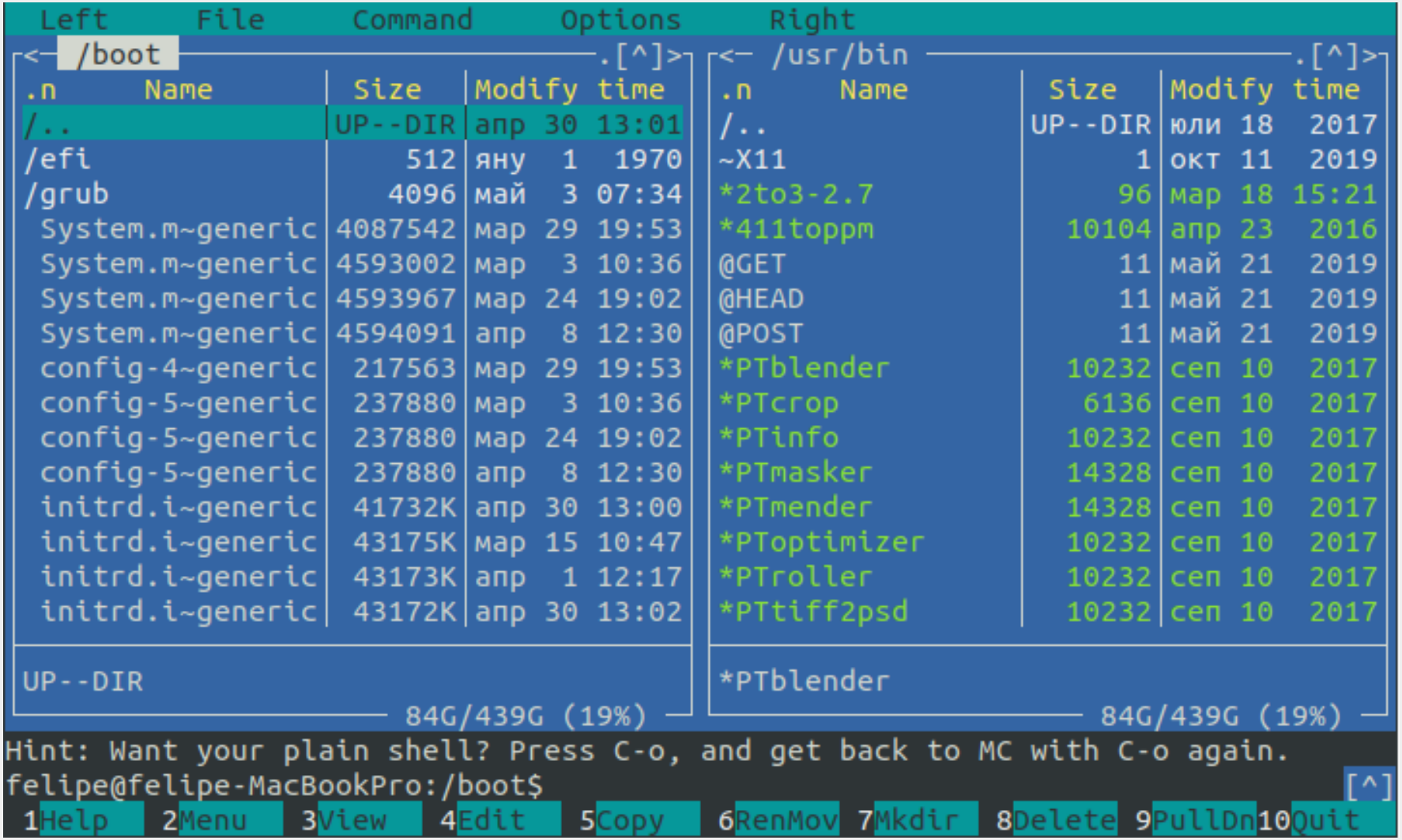
Сравняване на файлове

Командата diff сравнява съдържанието на файлове ред по ред и извежда разликите им. Синтаксисът е
diff <options> file1 file2

a -> означава, че нещо е било добавено c -> означава, че нещо е било променено d -> означава, че нещо е било изтрито < Редове от първия файл > Редове от втория файл

Midnight commander

GNU Midnight Commander е файлов мениджър за конзолата, вдъхновен от Norton Commander. с командата mc го стартирате.



Командата ifconfig(interface configutaror) се използва за инициализиране на интерфейс, задаване на IP адрес на интерфейс и изключване на интерфейс. Също често се използва да видим зададените IP адреси, зададени на даден интерфейс.
С аргументи име на интерфейс можем да видим данни конкретно за него, като IP и MAC адреси и други.
ifconfig eth0

Командата ping (Packet INternet Groper) позволява да видим дали има връзка между две свързани към мрежа (локална или глобална) устройства. Ping използва протокола ICMP (Internet Control Message Protocol), за да комункира с други устройства като изпраща пакет ICMP ECHO_REQUEST до целта и чака за отговор. Трябва да се има предвид, че някои устройства в мрежата блокират ICMP заявките с firewall. Употребата е:
ping ip_address
или
ping hostname

Командата traceroute е инструмент за диагностициране на мрежови проблеми. Показва броя на хостовете, през които минава един пакет по пътя си. Често се използва след като ping е показала загуба на пакети. Също така traceroute показва времето до всеки хост в мрежата.

Командата netstat(Network Statistic) се използва за определяне на мрежовите връзки, рутиращите таблици и други настройки и статистики. Без параметър показва списъка с отворени сокети.
С параметър -l показва всички сокети, отворени за слушане, които по подразбиране не се показват.
С параметър -i показва списък на всички мрежови интерфейси на машината.
С параметър -r показва рутиращата информация. Това означава конфигурираните пътища за пращане на пакети.
С параметър -a се показват всички сокети, независимо от тяхното състояние.

Инструментът telnet се свързва с целта през протокола telnet. Той позволява двупосочна, интерактивна текстово-базирана комуникация между две машини. Синтаксисът е:
telnet host_machine

Командата wget е част от инструментите на проекта GNU за изтегляне на файлове от уеб-сървъри. Синтаксисът е:
wget https://somedomain/file
Това изтегля файла в текущата директория. С параметър -O можем да запишем файла под друго име, а с -P в друга директория.

grep

Командата grep е много удобна, когато искаме да открием един или няколко файла, съдържащи определен низ от символи. Синтаксисът ѝ е следният:
grep pattern files
Всеки ред от всеки файл, който съдържа в себе си низа pattern, се извежда на екрана. Ако чрез командата са специфицирани повече от един файла, всеки ред е предшестван от името на файла, който го съдържа.

grep copyright *
Apache-2.0: "Licensor"shall mean the copyright owner or entity authorized by
Apache-2.0: the copyright owner that is granting the License.
Apache-2.0: copyright notice that is included in or attached to the work
Artistic: "Copyright Holder"is whoever is named in the copyright or
Artistic: copyrights for the package. Artistic:under the copyright of this Package, but belong to whoever generated
BSD:1. Redistributions of source code must retain the above copyright
BSD:2. Redistributions in binary form must reproduce the above copyright

grep

Можем да използваме регулярни изрази, за да търсим например дума независимо дали започва с малка или главна буква.

```
grep [tT]he BSD
Copyright (c) The Regents of the University of California.
modification, are permitted provided that the following conditions
1. Redistributions of source code must retain the above copyright
notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above
copyright
notice, this list of conditions and the following disclaimer in the
documentation and/or other materials provided with the
distribution.
3. Neither the name of the University nor the names of its
contributors
```

grep

Понякога ни интересуват само имената на файловете, които съдържат даден низ, без самите редове. Тогава може да се използва опцията -l.

```
grep -l copyright *
Apache-2.0
Artistic
BSD
CC0-1.0
GFDL
GFDL-1.2
GFDL-1.3
GPL
GPL-1
GPL-2
GPL-3
LGPL
LGPL-2
```

grep

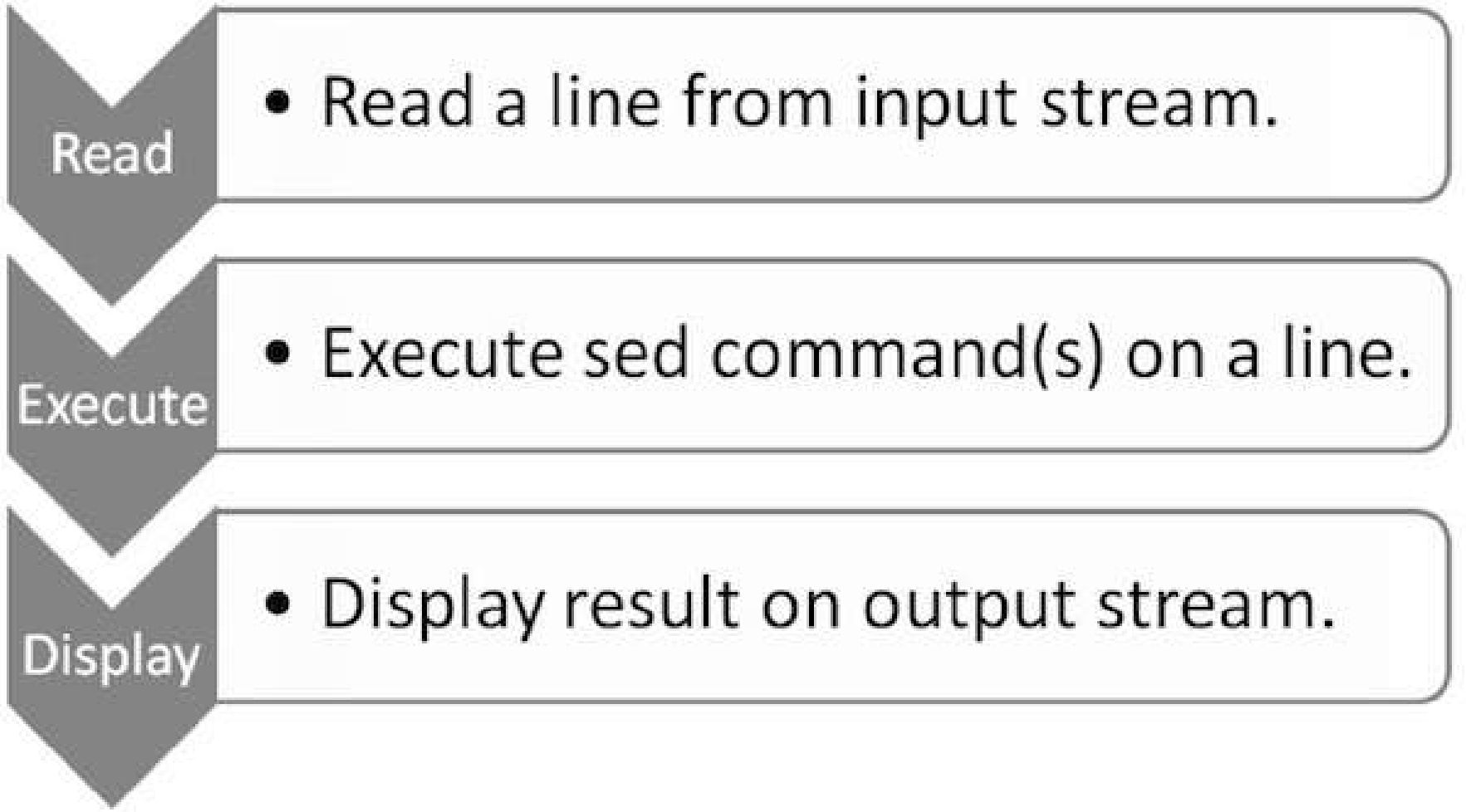
Много често се използва конвейр, за да се подаде текст към grep, и да се открият редовете, които съдържат даден низ. Например нека искаме да убием процеса gedit и затова ни трябва неговия номер. Можем да направим нещо такова:

```
ps -A | grep gedit
30434 ? 00:06:39 gedit
```

Поточен редактор sed

Командата sed дава възможност за еднопасово редактиране на файл. Командата бива приложена към всеки ред от файла, освен ако не са изрично посочени определени редове. Резултатът от sed се изпраща на стандартния изход, който може да бъде пренасочен.
Синтаксисът е:
sed [опции] команда [файл]

Поточен редактор sed



cd /usr/share/common-licenses/
sed ” BSD
Не подаваме команда и затова файлът се извежда.
sed ’p’ BSD ще разпечатва всеки ред по два пъти. Веднъж от явното извикване на командата, веднъж от поведението по подразбиране. sed -n ’p’ BSD - така подтискаме разпечатването по подразбиране.

Адресирането става като се подава номер на ред в командата.
Например sed -n '1p' BSD ще изведе само първия ред на файла.
sed -n '1,5p' BSD ще разпечата първите пет реда.
Можем да кажем действието да се извършва само с всеки n-ти ред с тилда. Например за да печатаме всеки втори ред ще напишем така:
sed -n '1~2p' BSD

Поточен редактор sed

Една от най-честите употреби е заместването на текст. Синтаксисът тук е като на регулярните изрази в grep. sed 's/old_word/new_word/' файл.
Например sed 's/the/ze/' BSD ще замени the със ze. Не трябва да се забравя, че sed заменя първото срещане на низа на всеки ред. Ако искаме да го прави на всички места в редовете, трябва да добавим g така: sed 's/the/ze/g' BSD
Ако искаме да заменяме всяко второ срещане на всеки ред, ще направим така: sed 's/the/ze/2' BSD
За да не бъде търсенето чувствително към регистъра, използваме i sed 's/the/ze/i' BSD

Поточен редактор sed

Друга честа употреба и изтриването на текст. Синтаксисът е: sed ’d’ файл. Пример:
sed ’1,3d’ BSD

Поточен редактор sed

Въпреки че sed не променя входния файл по подразбиране, това може да бъде променено с подаването на опцията -i.
Трябва да се внимава с тази настройка, защото с нея оригиналния файл бива презаписан. sed -i '1d' BSD.backup

Поточен редактор sed

Можем да подаваме няколко команди наведнъж на sed с помощта на опцията -e. Нека изтрием първи, трети и пети ред.
sed -e '1d' -e '3d' -e '5d' BSD

Поточен редактор sed

С командата w можем да пишем в друг файл. Така няма нужда да пренасочваме изхода.
sed -n '2~2 w BSD-alternate' BSD

Поточен редактор sed

Командата c се използва за промяна на ред.
Например
sed '1 c Copyleft' BSD
ще промени първия ред, започващ с Copyright с текста Copyleft.

Поточен редактор sed

С командата i можем да вмъкваме ред преди дадена позиция.
sed '1 i Read carefully:' BSD

Поточен редактор sed

Можем да правим субституции на символи с командата y, означаваща translate.
echo "LEET HACKERS sed 'y/EHST/3457/'

Поточен редактор sed

С командата e можем да изпълняваме външни команди, например ако имаме съвпадение на условието. Например нека сменим думата this с текущата дата:
sed '/this/ e date' BSD

Поточен редактор sed

Когато сме използвали регулярен израз не знаем с какво точно е съвпаднал. Ако искаме да го укажем, трябва да имаме начин да кажем "това, с което е съвпаднал". Това става със символа за амперсанд.
echo "LEET HACKERS sed 's/[A-Z]*/(&)/'

Задачи

- Създайте директория във вашата домашна директория, в която да експериментирате.
- В нея създайте под-директории и файлове.
- Преименувайте някои от тях.
- Изтрийте директория, която има файлове и директории в нея.
- Отидете в домашната си директория и оттам копирайте файл от една от под-директориите в първата ви създадена такава.
- Отидете в друга директория.
- Преименувайте няколко файла.
- Преместете и преименувайте файл едновременно.

Задачи

- Използвайте командата grep, за да откриете всички работещи процеси на вашата система, които принадлежат на използвания от вас браузер.
- В директорията /usr/share/common-licenses използвайте sed за да подмените всяко срещане на думата "package независимо от регистъра, с "PACKAGE".
- Изтрийте всички редове, съдържащи "PACKAGE"

Благодаря за вниманието!

Въпроси?