Лекция 2.3 Текстовые файлы и потоки (часть 2).

Простое форматирование текста. Подготовка текста к печати. Замена символов табуляции на пробелы. Команды выбора, объединения строк. Замена символов с помощью команды tr. Способы объединения файлов. Получение дампа. Разделение файлов на части. Команда xargs. Сортировка sort.

Простое форматирование текста осуществляется командой **fmt**. Она форматирует текст так, чтобы каждая строка имела заданную равную длину (по умолчанию – 75 символов). Опция -w позволяет указать ширину текста в символах, опция -s означает только разделять строки, но не объединять их.

Текст можно **форматировать для печати**. Вывести в стандартный поток вывода подготовленный к печати текст позволяет команда pr.

pr +2 - вывести файл, начиная со второй страницы;

pr -2 - форматировать для печати в две колонки.

Иные опции:

-w – устанавливает ширину страницы (по умолчанию – 72 символа);

-o – создать отступ слева в заданное количество символов;

-d – двойной интервал.

Команда **expand** предназначена для **перевода символов табуляции в тексте в пробелы**. По умолчанию каждый символ табуляции заменяется на восемь пробелов, что можно изменить, указав опцию «-Число». Обратное преобразование пробелов в табуляцию осуществляет команда **unexpand**. Опция **unexpand** -t – установить позиции табуляции (если указывается только одно значение, то позиции табуляции буду расставлены через данный интервал); опция -a – заменить все пробелы, а не только ведущие в строках.

Команда **uniq** позволяет удалить все (за исключением одной) строки, встречающиеся более одного раза, из файла/потока ввода. Работает с предварительно отсортированными файлами. Полезные опции команды:

-c – посчитать количество повторов каждой строки;

-d – вывести только повторяющиеся строки;

-f<n> – пропустить n полей до определения уникальности;

-i – игнорировать регистр;

-u – вывести только неповторяющиеся строки;

Вывести дублирующиеся строки, сравнивать, начиная с 6го поля

cat file.txt

Привет Иванов И.И. MAR 2019 4.290

Кошка Петров В.В. FEB 2019 3e10

Кошка Петров В.В. FEB 2008 3e10

Алтын Friday B.W. MAY 1994 1.000

uniq -f5 -d file.txt

Кошка Петров В.В. FEB 2019 3e10

Заменой связки **sort + uniq** является команда **sort -u** (см. далее).

Команда **comm** позволяет сравнить два файла и вывести в три столбца 1) строки, которые есть только в первом файле; 2) строки, которые есть только во втором файле; 3) строки, которые есть в обоих файлах. Численная опция команды отменяет вывод соответствующего столбца. Команда работает только для отсортированных предварительно файлов. Если на вход команды поданы неотсортированные файла, команда не отрабатывает корректно (см. пример ниже, слово «Еж», присутствующее в обоих файлах, не отражается в третьем столбце):

cat 1.txt

Кошь

Мышь

Еж

cat 2.txt

Мышь

Кошь

Еж

comm 1.txt 2.txt

Кошь

Мышь

comm: данные файла 1 не отсортированы

comm: данные файла 2 не отсортированы

Еж

Кошь

Еж

Команда **tr** является чрезвычайно полезной для **удаления или замены символов** в строке. Опции команды:

-c/-C - использовать не указанное множество символов, а его дополнение до полного множества символов (т.е., инвертировать выбор символов);

-d - удалять символы из множества;

-s - заменить повторяющиеся символы из множества на один символ;

tr без опций заменяет встреченные в тексте элементы из первого множества символов на соответствующие элементы из второго множества символов.

tr [a-z] [A-Z] - заменить все строчные буквы на заглавные.

ps | tr -d '\n ' - удаление символов (перевод строки и пробел);

Вывод: **PIDTTYTIMECMD3410pts/700:00:00bash….**

echo 'boot' | tr -s [:alpha:] - исключение повторяющихся букв из слова;

Вывод: **bot**

tr -cs "[:alpha:]" "\n" <file1 >file2 - замена всех небуквенных символов на символ новой строки. Опция -s позволяет заменить идущие вплотную небуквенные символ только на один символ новой строки.

Команда **join** - **объединить два отсортированных файла по заданным полям**.

-t – разделитель полей;

-1 <n> -2 <m> – номера общих полей;

-a<n> – добавить в вывод непарные строки из файла n;

-v<n> – вывести только непарные строки из файла n.

cat 1.txt

drwxr-xr-x 2 root root 4096 Jan 9 21:18 mysql

cat 2.txt

2 10 20

join -1 2 -2 1 1.txt 2.txt

drwxr-xr-x 2 root root 4096 Jan 9 21:18 mysql 10 20

Команды **nl**, а также **cat -b**, позволяют осуществить нумерацию строк файла или текста в потоке вывода.

Команды **hexdump, xxd, od** позволяют получить двоичную, восьмеричную, десятичную или шестнадцатеричную репрезентацию файла.

Двоичная:

xxd -b file.txt

Восьмеричная:

hexdump -o file.txt

od -o file.txt

Шестнадцатеричная:

xxd file.txt

od -x file.txt

hexdump -C file.txt

Команда xxd имеет опцию -r, которая позволяет декодировать шестнадцатеричные данные при использовании этой же программы (в общем, стандартного формата записи с отступами) при кодировании.

Команда paste позволяет построчно соединить два файла:

paste <file1> <file2> - через [TAB];

paste -d ' ' <file1> <file2> - через пробел.

paste - - <file> вывести файл, соединив по две строки в одну

paste -s <file> - соединить все строки файла;

Разделение файлов на части осуществляется командой split. По умолчанию, команда записывается по 1000 строк текста в каждый создаваемый ей выходной файл xaa, xab, xac и т.д. Можно указать следующие опции команды:

-l<кол-во строк> – количество строк в частях файла;

-b<кол-во байтов> –количество байтов в частях файла;

-n – разрезать на n файлов;

-d – использовать численный суффикс x00, x01, …

csplit <file> <expression> - разделить файл на две части: до вхождения заданной строки и после нее; **{n}** в конце команды позволяет выполнить поиск шаблону expression и разделение n раз (на выходе – файлы xx00, xx01,…). Помимо {n}, можно использовать следующие шаблоны поиска:

целое_число - копировать файл до указанной строки, не включая ее;

/регулярное_выражение/ [смещение] - копировать до строки, включающей регулярное выражение, но не включать соответствующую строку.

%регулярное_выражение % [смещение] - пропустить строки до строки, включающей регулярное выражение, но не пропускать саму эту строку.

«смещение» - число строк, указанное в виде «+число» или «-число», на которое нужно сместиться после нахождения регулярного выражения.

xargs - выполнить команду, составив ее из входящих аргументов.

Часто используют связку xargs + find.

Пример:

поиск и удаление временных файлов (начинаются с «~»)

find ~ -type f -name '^~*' -exec rm -f {} \; - с опцией find -exec;

find ~ -type f -name ' ^~*' | xargs rm -f - применяя конвейер + xargs.

Сортировка строк в тексте осуществляется командой **sort.** Команда без опций осуществляется сортировку посимвольно в порядке возрастания (буквы - от «A» до «Z», цифры идут раньше букв, строчные буквы идут раньше прописных).

Опции команды:

-b, --ignore-leading-blanks - опустить ведущие пробелы;

-d, --dictionary-order - словарная сортировка (учитываются буквы, цифры и пробелы);

-f, --ignore-case - игнорировать регистр символов;

-g, --general-numeric-sort - сравнение по числовому значению в общей математической нотации (подходит как для сравнения чисел с плавающей запятой, так и целых чисел);

-h, --human-numeric-sort - сравнение размеров («5K», «2M»);

-i, --ignore-nonprinting - сортировать только по печатным символам;

-k, --key POS - сортировать по столбцу POS, а далее, - по столбцам после него. Варианты команды:

-k POS1,POS2 - сортировка по полям от POS1 до POS2, далее - с начала строки. Чтобы отключить сортировку с начала строки, используйте -s.

-k POS1.SYM1,[POS2.SYM2] - сортировка по полю POS1, начиная с символа SYM1 в поле, [до символа SYM2 в поле POS2, далее - с начала строки]. Чтобы отключить сортировку с начала строки, используйте -s.

-m, --merge - объединить заранее отсортированные файлы и не сортировать.

-M, -month-sort - сравнение дат по месяцам «JAN» < ... < «DEC».

-n, --numeric-sort - численное сравнение;

-r, --reverse - сортировка в обратном порядке;

-o, --output=ФАЙЛ Записать результат в ФАЙЛ вместо стандартного вывода;

-s, --stable - устойчивая сортировка (если две строки совершенно идентичны по всем полям, используемым для сортировки, то они гарантированно появятся в результатах сортировки в исходном порядке);

-t, --field-separator=РАЗД - использовать разделитель «РАЗД» для полей вместо пробела;

--parallel=N - ограничить количество потоков для сортировки до n (по умолчанию сортировка выполняется параллельно, число потоков равно число процессоров, но не больше восьми);

-u, --unique - удалить все идентичные строки, кроме первой (аналог uniq).

Примеры сортировки

cat file.txt

Привет Иванов И.И. March 2019 4.290

Кошка Петров В.В. February 2019 3e10

Кошка Петров В.В. February 2008 3e10

Алтын Friday B.W. May 1994 1.000

Сортировка по 4-му полю. После этого - сортировка по строке целиком (POSIX-свойство), поэтому Петров 2008 поднимается выше, чем Петров 2019.

sort -k4,4 file.txt

Кошка Петров В.В. FEB 2008 3e10

Кошка Петров В.В. FEB 2019 3e10

Привет Иванов И.И. MAR 2019 4.290

Алтын Friday B.W. MAY 1994 1.000

Устойчивая (stable) сортировка по 4-му полю. После сортировки по нему в остальном сохраняется исходный порядок строк (Петров 2019 выше Петров 2008)

sort -k4,4 -s file.txt

Кошка Петров В.В. FEB 2019 3e10

Кошка Петров В.В. FEB 2008 3e10

Привет Иванов И.И. MAR 2019 4.290

Алтын Friday B.W. MAY 1994 1.000

Сортировка со второй буквы первого слова

sort -k1.2 file.txt

Алтын Friday B.W. MAY 1994 1.000

Кошка Петров В.В. FEB 2008 3e10

Кошка Петров В.В. FEB 2019 3e10

Привет Иванов И.И. MAR 2019 4.290

Устойчивая сортировка со второй буквы первого слова до конца слова (без ",1" устойчивая сортировка не произойдет, так как сортировка выполнится до конца строки, и Петров 2008 поднимется выше Петров 2019)

sort -k1.2,1 -s file.txt

Алтын Friday B.W. MAY 1994 1.000

Кошка Петров В.В. FEB 2019 3e10

Кошка Петров В.В. FEB 2008 3e10

Привет Иванов И.И. MAR 2019 4.290

Сортировка по числу с плавающей запятой в 6-ом столбце

sort -k6 -g file.txt

Алтын Friday B.W. MAY 1994 1.000

Привет Иванов И.И. MAR 2019 4.290

Кошка Петров В.В. FEB 2008 3e10

Кошка Петров В.В. FEB 2019 3e10

Простая численная сортировка сортирует по числу до точки или буквы "e"

sort -k6 -n file.txt

Алтын Friday B.W. MAY 1994 1.000

Кошка Петров В.В. FEB 2008 3e10

Кошка Петров В.В. FEB 2019 3e10

Привет Иванов И.И. MAR 2019 4.290

Смена разделителя для сортировки. Сортировка по отчеству (W<B<И)

sort -t '.' -k2 file.txt

Алтын Friday B.W. MAY 1994 1.000

Кошка Петров В.В. FEB 2008 3e10

Кошка Петров В.В. FEB 2019 3e10

Привет Иванов И.И. MAR 2019 4.290

Лекция 2.4 Потоковые редакторы.

Потоковый редактор Sed (stream editor — потоковый редактор): буферы, замена, обратные ссылки, модификаторы, опции, удаление, печать, инвертирование выбора, запись, чтение. Потоковый редактор awk: шаблон, команды, встроенные переменные. Вычисления на awk. Написание скриптов awk. Предопределенные функции awk. Передача переменных из командной оболочки в awk и обратно.

awk и **sed** - две программы для Unix и Linux-систем, предназначенные для обработки текстовой информации. Их называют потоковыми редакторами, Текст поступает в виде потока на обработку, файл тоже можно передать на вход потокового редактора, и он будет обработан построчно, как текстовый поток.

Программы sed и awk можно использовать в качестве интерпретаторов файлов со скриптами. Однако самое распространенное применение этих потоковых редакторов - внутри однострочных скриптов («oneliner»), которые обычно вводятся напрямую с командной строки. Умение написать однострочный скрипт для произвольной обработки текста, данных ценится в кругу Linux-программистов и может считаться отчасти показателем умения пользователя работать с командной оболочкой и знания основных команд Linux.

Интерпретатор/потоковый редактор sed

Для многих целей sed сейчас заменен языком perl, но для простых преобразований в сценариях оболочки sed имеет довольно большую распространенность. Довольно мощный редактор, на sed можно написать даже тетрис (https://github.com/uuner/sedtris).

Наиболее часто используемой командой является **s** (замещение), которая заменяет один шаблон другим.

sed s/dog/pet/g in> out

заменит «dog» на «pet» в файле и выведет его в файл **out**; «**g**» означает «глобальную» замену, то есть, замену всех вхождений шаблона, а не только первого.

Для экранирования сложных шаблонов следует использовать одинарные кавычки:

echo "abccbd" | sed "s/a\([bc]*\)d/\1/g"

В данном примере заменяются все вхождения буквы «a», за которой следуют b либо c, а далее произвольное количество букв, завершавшиеся буквой «d» на захваченную группу

символов с номером 1 (\1). Для захвата группы символов используются круглые скобки, скобки экранированы обратным слэшем.

Теперь вы сможете самостоятельно разобрать, почему команда

echo "abccbd" | sed "s/a*\([bc]\)\(cb\)/\2\1/g"

выведет на экран

cbcd

Заметим, что в последнем примере «**d**», часть строки вне регулярного выражения, не подвергается преобразованиям.

GNU sed дает возможность трактовать скобки как регулярные выражения всегда, включив опцию «использовать регулярные выражения» «**-r**»:

echo "abccbd" | sed -r "s/a*([bc])(cb)/\2\1/g"

Пример: вывести те группы из файла /etc/groups, членом которых является пользователь root, при этом заменить все «:» на «_»:

sed -n /root/s/:/_/gp /etc/group

В этом примере показан комбинированный эффект опций **-n** – подавить вывод строк и **-p** – печать совпадающих с шаблоном строк.

Другие опции sed:

i – вставка строки перед заданной строкой;

a – добавление строки после заданной строки;

c – изменение всей строки на заданную (удаление + вставка строки);

q – выход без дальнейшей обработки строк;

e команда – выполнить команду;

f файл - обработать текст из файла;

= – команда печати номера строки.

Вставка перед второй строкой

who | sed '2i

новая строка'

Результат:

root tty1 Mar 13 17:23

новая строка

user tty7 Mar 13 17:04

Вставка после второй строки

who | sed '2a

новая строка'

Результат:

root tty1 Mar 13 17:23

user tty7 Mar 13 17:04

новая строка

Вывод номеров строк, содержащих «Привет»

sed '/Привет/=' file.txt

Другие примеры sed

sed '2,4d' - удалить вторую и четвертую строки текста.

Модификатор «!» инвертирует выбор:

sed '2,4 !d' - удалить все строки, кроме второй и четвертой.

Для использования переменной оболочки в sed экранируйте команды двойными кавычками, а не одинарными:

pattern=TO_DELETE

sed "/^$pattern/d" file.txt

Можно использовать и многие другие разделители для команд, например, точку с запятой. Это удобно, чтобы не экранировать слэш:

sed 's;/home/user;;g' file.txt

- удалит текст **/home/user** из файла **file.txt.**

Для удаления повторов используйте обратную ссылку внутри замены:

echo первый первый второй | sed 's_\([а-я]*\) \1_\1_'

Для захвата всего шаблона используют «&»:

echo 'цветы ландыша' sed 's/[а-я]*/(&)/g'

Результат:

(цветы) (ландыша)

Модификатор /i команды замены позволяет игнорировать регистр букв:

echo 'Цветы ландыша' sed 's/[а-я]*/(&)/gi'

Результат:

(Цветы) (ландыша)

Опция **-e** позволит выполнить несколько операций над одним текстом:

echo 'Цветы ландыша' sed -e 's/[а-я]*/(&)/g' -e 's/ы/ок/'

Результат:

(Цветок) (ландыша)

sed '2,$ s/кот/бык/i' file.txt - редактировать от второй строки до конца файл file.txt

sed '/Привет/,/Пока/ s/кот/бык/i' - редактировать от слова «Привет» до слова «Пока» файл file.txt.

sed '5,/^$/ d' - удалить текст от пятой до первой пустой строки.

awk — это командный интерпретатор, главным образом, предназначенный для обработки структурированных записей, содержащих текст.

При обработке производится разделение потока ввода на записи, каждая из которых содержит ряд полей. Каждый раз, когда интерпретатор awk встречает разделитель записей, он начинает новую запись. По умолчанию разделитель записи является символом новой строки. Это можно изменить. Внутри записи поля разделяются разделителем полей.

Работа awk заключается в следующем. Последовательно к каждой записи применяется ряд правил, записанных в основном теле скрипта awk (ограничивается фигурными скобками {...}). Кроме того, можно задать правила, применяемые перед началом обработки всех записей (с помощью BEGIN{…}) и после окончания обработки всех записей (с помощью END{}).

Преимущества и недостатки awk

+ прост в использовании, удобно использовать в командной строке для обработки текста/данных, имеющих структуру (разделенных на записи и поля);

- исполнение программ интерпретируемого языка (awk) медленнее, чем компилируемого (C);

- не является языком глобального назначения, не предназначен для интенсивных вычислений , т. е., язык - специализированный;

+/- все переменные находятся в глобальной области видимости за исключением параметров, переданных в функцию.

При вводе с консольной строки скрипт awk заключается между одинарными кавычками, а текст внутри скрипта заключают в двойные кавычки:

```
awk 'BEGIN {print "k"}'
# выводит k
```

Как показано в примере выше, скрипт awk может не принимать на вход текста, если используется BEGIN. Однако, основное назначение awk — обработка текста, и, поэтому, обычно можно видеть следующие типы (шаблоны) вызова awk:

Команды | awk '{команды}' – в цепочке конвейера

awk '{команды}' file.txt — для обработки file.txt

awk '{команды}' < file.txt — аналог предыдущей команды

awk -f file.awk file.txt — выполнить awk-скрипт из файла file.awk для обработки file.txt

./file.awk - выполнить исполняемый файл, содержащий awk-скрипт (файл file.awk)

Нижеприведенный скрипт построчно выводит в стандартный поток вывода (при нашем обучении — будет выводиться на экран) файл file.txt:

```
awk '{print $0}' file.txt
```

Интерпретатор awk имеет ряд **встроенных команд** для обработки текста. Самой частой используемой командой awk является print. В форме print или print $0 команда выводит в стандартный поток вывода текущую обрабатываемую запись. print $1 распечатает содержимое первого столбца записи, print $2 — второго, и так далее.

В awk также имеется ряд **встроенных переменных**. Общее число записей хранится в специальной переменной NR. Общее число столбцов в текущей записи — в переменной NF. Другие специальные переменные - разделитель полей для потока ввода FS и потока вывода OFS, разделитель записей в потоке ввода RS и в потоке вывода ORS, имя текущего обрабатываемого файла FILENAME и число записей в нем FNR (удобно использовать, если в качестве входных файлов для awk указано несколько имен файлов через пробел). Аналогично изменению переменной FS действует и параметр awk **-F.** OFMT представляет шаблон формата вывода (для форматирования чисел), его значение по умолчанию %.6g.

Выведем поля 3 и 4 с разделителем «=», использовав файл **/etc/passwd**, хранящего информацию о пользователях и аутентификации, где разделителем будем считать знак «:» :

awk -F':' 'BEGIN{OFS="=";} {print $3,$4;}' /etc/passwd

Результат:

0=0

1=1

2=2

...

Заметим, что все специальные переменные вводятся без знака $.

Самоконтроль: что выведет на экран команда **print $NF**?

Awk поддерживает **регулярные выражения**. Например, для выбора всех строк (записей), содержащих «honey», следует использовать однострочный скрипт (англ. one-liner)

awk '/honey/ {print}'

Регулярные выражение внутри шаблонов

- **/^a/** - поле начинается с **a**
- **/a$/** - поле кончается **a**
- **[abc]** - любой из символов **a**, **b** и **c**
- **[a-p]** - любой символ диапазона
- * - 0 или больше вхождений регулярного выражения
- + - 1 или больше вхождений регулярного выражения
- **?** - 0 или 1 вхождение регулярного выражения
- **ab|cd** - **ab** или **cd**

Чтобы искать текст, а не использовать регулярное выражение, применяйте экранирование:

- \+ - экранирует +

Фильтрация по выражению (выражение в отличие от шаблона включает оператор сравнения, в том числе, «~», который означает «содержит»).

$ awk '$3~0 {print}‘ file.txt

1 2 0 5

6 6 0 7

$ awk '$3!~0 {print}' file.txt

Петров Иван Иванович

1 2 1 5

Правило BEGIN выполняется только один раз, прежде чем будет прочитана первая входная запись. Аналогично, правило END выполняется только один раз, после того, как все данные прочитаны. Например,

$ awk '

> BEGIN {print "Ищем студента…"}

> /Petrov/ {++n}

> END {print "\"Petrov\" появляется в ",n, " записях." }' my_students.txt

Результат:

"Petrov" появляется в 3 записях.

Циклы и ветвление в awk

if (условие) {операторы} [else {операторы}]

while (условие) {операторы}

for (выражение; условие; выражение) {операторы}

for (индекс in имя_массива) {операторы}

Операторы:

exit — завершить исполнение скрипта;

next — перейти к обработке следующей записи;

break — завершение цикла;

continue — переход к следующей итерации цикла.

Вывод полей записи через одно:

awk '{ for(k=1; k<=NF; k+=2); {print $k}}'

Отбор записей, где первое поле больше 10:

awk '{ if ($1>10) {print}}'

Массивы в awk используют строковые индексы, а не числовые. Это позволяет обращаться к элементам не только с использованием чисел в качестве идентификатора элемента, как, например в языке C: a[5], но и с использованием, например, строки в качестве идентификатора. Поясним это на примере элемента массива a с идентификатором "k":

```
echo "" | awk '{a["k"]=10; print a["k"]}'
# выводит 10
```

Самоконтроль: напишите one-liner для вычисления корня из числа, заданного пользователем с использованием here document и awk.

При обращении к элементам многомерных массивов в awk используют несколько индексов, разделенных запятой:

```
#объявление массива размерностью 4x4
for (i=1; i<5; i++)
 for (j=1; j<5; j++)
        vec[i,j]=i+j;
```

Следующая функция транспонирует массив:

```
{
    if (max_nf < NF)
        max_nf = NF
    max_nr = NR
    for (x = 1; x <= NF; x++)
        vector[x, NR] = $x
}
END {
    for (x = 1; x <= max_nf; x++) {
        for (y = 1; y <= max_nr; y++)
            printf("%s ", vector[x, y])
        printf("\n")
    }
}
```

Примера ввода:

```
1 2 6
5 6 7
7 8 44
10 29 39
```

Вывод:

1 5 7 10

2 6 8 29

6 7 44 39

В awk предопределены следующие **встроенные функции** (табл. 7).

Таблица 7. Встроенные функции awk

sin(expr)	синус expr
cos(expr)	косинус expr
exp(expr)	возведение в степень expr
log(expr)	натуральный логорифм expr
sqrt(expr)	извлечение корня expr
int(expr)	целая часть числа
length(s)	длина строки s
printf(fmt, ...)	форматированный вывод по спецификации fmt (аналог функции языка C).
substr(s, m, n)	подстрока в n символов строки s, начинающаяся с m.
getline()	чтение следующей строки.
index(s1, s2)	номер позиции, с которой s1 совпадает с s2, иначе 0.
split(s, M, c)	строка s разбивается элементы массива M по разделителю c (по умолчанию FS=" "); функция возвращает число полей.

Пример: вывести только строки файла file.txt длиной более 20 символов

awk 'length($0) > 20' file.txt

Функция getline позволяет получить содержимое строки в переменную без какой-либо другой обработки, изменения указателей и переменной $0.

Например, при выполнении следующей команды

echo "2

3

4

5

6

7

8" | awk '{if ((getline t) > 0) { if ((getline t) > 0) {print(t)}; print $0}}'

мы получим вывод:

4 2 7 5

Как видно, значение большее нуля возвращается, если команда getline выполнилась успешно, и следующая строка для считывания существовала. Для последней строки, содержащей число «8», не существует следующей строки. Первое условие — ложно, и вывода последней строки не происходит.

Функция split позволяет разделить строку на части, посчитать число элементов, и сохранить результат в массив:

count = split(string, array_name, regexp);

Пример:

count = split("my sweet home", my_arr, / /);

В переменной count сохранится число элементов (count = 3), в массив my_arr будет записано три элемента (последний - «home»).

Возможно использование системных вызовов для вызова сторонних утилит из awk:

echo "22 23 24" | awk '{system("echo -n "$1" "$3"| wc -m")}'

Результат:

5

Данный скрипт выводит количеством символов в первом и третьем столбце введенного текста с учетом пробела между ними.

Литература к лекции 2.4

13. The GNU Awk User's Guide. - 2002. Режим доступа ftp://ftp.gnu.org/old-gnu/Manuals/gawk-3.1.0/html_chapter/gawk_toc.html#SEC_Contents Дата обращения 10.02.2018

Лекция 2.5 Работа с жесткими дисками и файловыми системами.

Устройство файловой системы. Хранение информации в файловой системе. Использование жестких связей и символических ссылок. Работа с жесткими дисками и файловыми системами. Физическая структура накопителя. Имена жестких магнитных дисков. Создание разделов при помощи fdisk. Создание файловой системы. Проверка целостности файловой системы. Монтирование файловых систем. Работа с разделом подкачки. Мониторинг дисковых ресурсов. Оптимизация производительности диска IDE. Резервное копирование. Команда dd. Архивирование файлов. Производительное копирование файлов при помощи утилиты rsync.

Устройство файловой системы покажем на примере файловой системы ext4. Вся область записи диска делится на **загрузочную запись и несколько разделов**. В каждом разделе содержатся **загрузочные блоки и группы блоков (цилиндров)** (рис. 20). Каждая группа блоков состоит из суперблока (информация о количестве блоков, о файловой системе), массива индексных дескрипторов и блоков данных (рис. 20).

Особые значения номеров inode (в файловой системе ext4):

1– bad block (испорченный сектор);

2– inode корневой файловой системы (/);

5 – загрузчик;

8 – журнал (скрытый файл).

stat - информация о метаданных файла;

ls -li - отобразить inode файла.

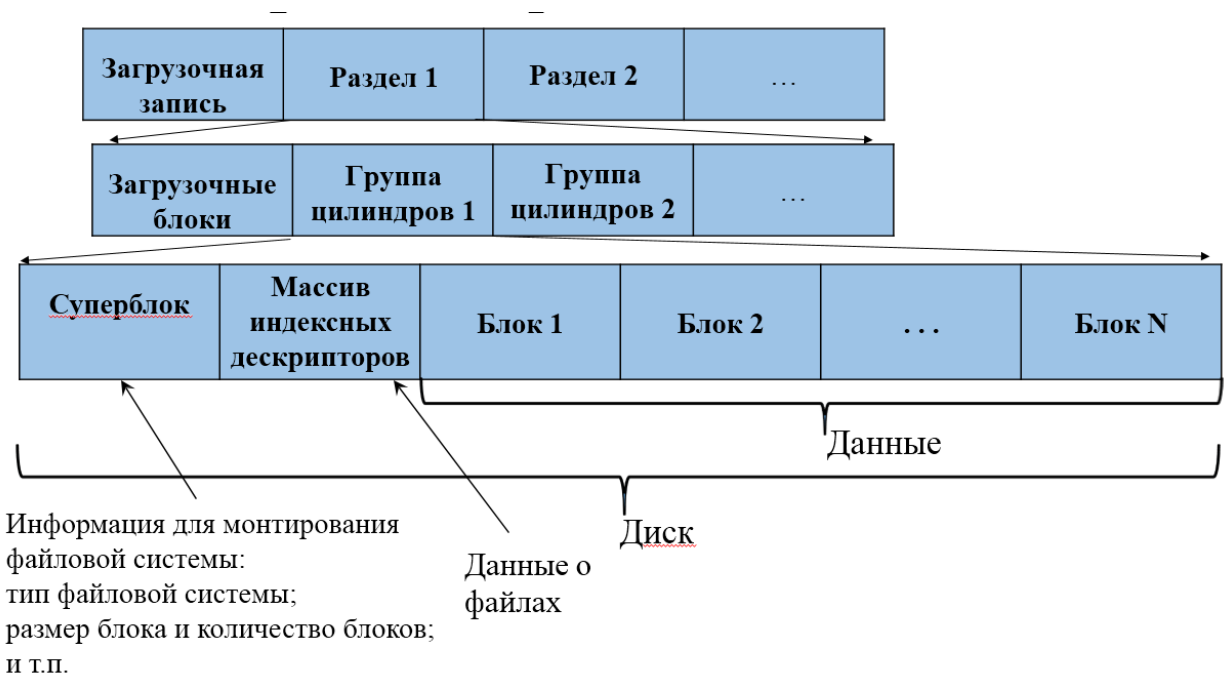

Рисунок 20. Устройство файловой системы ext4

Жесткая связь – разные имена для файла, имеющего один заданный inode.

ln <filename> <newname> - создать жесткую связь;

cp -l - создавать жесткие связи вместо копирования.

Символическая ссылка (мягкая связь) – указатель на другой файл.

ln -s <filename> <linkname> создать символическую ссылку;

cp -s - вместо копирования создавать символические ссылки.

При копировании файлов с использованием cp, автоматически копируются файлы по ссылкам.

cp -d - копировать сами ссылки, а не файлы

Устройство магнитного диска

Накопители на жестких магнитных дисках состоят из одного или нескольких магнитных дисков и магнитных головок (head) (рис. 21). Магнитная головка перемещается над поверхностью одного магнитного диска.

Магнитная поверхность диска разбита на дорожки (track). Дорожки одного диаметра на всех магнитных поверхностях образуют цилиндр (cyl), при этом справедлива формула

$$track = cyl \times head \quad (1)$$

Каждая дорожка разбита на секторы (sect), стандартный размер, которых 512 байт.

Объем диска в байтах:

$$capacity = cyl \times head \times sect \times 512 \quad (2)$$

Стандартное количество секторов в дорожке – 63.

Основные параметры диска (геометрия) – количества цилиндров cyl и головок head.

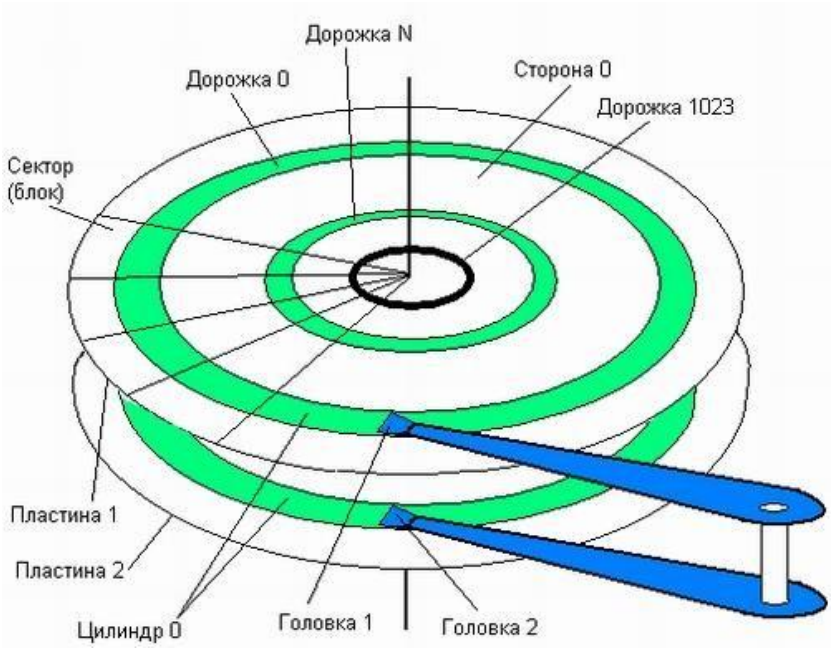

Рисунок 21. Устройство жесткого диска

/sbin/fdisk -l - получить сведения о геометрии диска.

Вывод команды:

Disk /dev/sda: 160.0 GB, 160041885696 bytes

255 heads, 63 sectors/track, 19457 cylinders, total 312581808 sectors

Units = sectors of 1 * 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disk identifier: ……

Твердотельные накопители (SSD) имеют более высокую стоимость за ГБ объема, меньшее число циклов перезаписи, но 1) намного более быстрые (скорость чтения), 2) бесшумные, нет механических частей, 3) легкие, 4) потребляют меньше энергии.

Диск делится на разделы (partitions) (максимум - 4 первичных), один из них может быть расширенным (extended) и содержать внутри любое количество логических разделов (logical partitions).

Имена жестких дисков SATA/SCSI

/dev/sda – Primary Master;

/dev/sdb – Primary Slave;

/dev/sdc – Secondary Master;

/dev/sdd – Secondary Slave.

Команда ls -l для файлов устройств вместо размера выводит **главный ID (major number) и вспомогательный ID (minor number)**. Главный ID– номер драйвера для данного типа устройств в Linux (8- SATA жесткий диск, 4 – терминал, 1 – псевдоустройство - /dev/null, /dev/random, dev/zero), вспомогательный – порядковый номер устройства данного типа (0 – для всего диска, 1,2,3,4 – для первичных разделов, 5-15 – для логических разделов).

Изменение разделов при помощи fdisk производится командой

fdisk <имя устройства>

Команды интерактивной утилиты fdisk:

m – список доступных команд;

q – завершение работы без сохранения изменений;

l – список доступных типов разделов;

d – удалить раздел;

n – создать раздел;

t – установка типа созданного раздела;

w – записать таблицу разделов;

p – напечатать таблицу разделов;

x – дополнительные функции (для экспертов).

Пример:

Command (m for help): n

Partition type:

p primary (3 primary, 0 extended, 1 free)

e extended

Select (default e): e

Select partition 5

First sector (102326202-485063864): 102326202 Last sector, +sectors or +size{K,M,G} (102326202-485063864, default 485063864):

Using default value 485063864

Partition 5 of type Linux and of size 195 GiB is set

Создание файловой системы может быть выполнено командой

mkfs <раздел>

Опции команды:

- t – тип ФС;
- c – проверка на bad blocks;
- b – установить размер блока (1024/2048/4096) в байтах.

Пример (создание файловой системы ext4)

 # mkfs -t ext4 /dev/sda6

Проверка целостности файловой системы

При сбое питания или другом сбое файловая система (ФС) может потерять целостность. При этом может произойти

- частичная потеря данных, над которыми производились операции;
- появление inode, которые указывают на уже освобожденные блоки, и, наоборот, появление блоков данных, на которые не ссылается ни один inode;
- другие искажения системы метаданных.

При запуске системы Ubuntu на экране можно видеть запись /dev/sda5 – clean или ошибки, как например «orphaned block…». Первая запись свидетельствует о флаге clean файловой системы (целостное состояние), вторая запись – о нарушении целостности ФС.

fsck <опции> <раздел> - проверить целостность ФС.

Проверять целостность можно только для отмонтированной ФС! Иначе, возможна полная потеря данных.

Утилиты восстановления целостности файловой системы не восстанавливают данные, а лишь возвращают ее к неповрежденному, непротиворечивому состоянию (consistency).

 # fsck -t ext3 /dev/sdb2 - проверить целостность файловой системы ext3;

ls -l /sbin/fsck.* - список утилит для проверки ФС.

Файловая система ext4, используемая часто для Ubuntu, поддерживает журналирование. **Журналирование** – инструмент поддержания целостности ФС. Восстановление по журналу позволяет избежать в большинстве случаев запуска fdisk после сбоя работы Ubuntu.

Как работает журналирование – на диске выделяется место для хранения информации об операциях с блоками. Все операции с диском происходят как транзакции, где в журнале записывается полная информация о транзакции. При внезапном сбое по журналу можно понять, какие этапы были сделаны, а какие – нет. Например, при перенесении файла на другой раздел, 1) сначала копируется содержимое, 2) устанавливается новый inode, 3) затем удаляется старый inode. При сбое после шага 2) файл останется в двух местах, если не ведется журнал.

Ext2 – ФС без поддержки журналирования (между тем, данные нетрудно восстановить утилитами восстановления данных). Ext3, Ext4 – ФС с журналированием. **Доступные режимы журналирования (ext3):**

Journal – как данные, так и метаданные, сохраняются в журнале.

Ordered - в журнале сохраняются только метаданные. Метаданные сохраняются в журнал после данных. Это - значение по умолчанию.

Writeback – аналог ordered, за исключением того, что метаданные сохраняются в журнал до или после того, как данные будут записаны на диск.

В Ext4 журналирование можно отключить (режим Off).

Схема отключения журналирования (стоит использовать только для SSD с малым ресурсом жизни, для флэш-накопителей с установленным Linux.), команды выполняются от суперпользователя (знак решетка в начале строки, выделенной жирным шрифтом):

демонтирование раздела

 # umount /dev/sdaX

отключение журнала

 # tune2fs -O ^has_journal /dev/sdaX

проверка диска

 # e2fsck -f /dev/sdaX

Ext3 поддерживает файлы до 2 TB, файловая система может быть до 32 TB. Ext4 поддерживает файлы до 16 TB, файловая система может быть до 1 EB (Exabyte = 1024 Petabyte = 1024*1024 TB).

Монтирование файловых систем

Монтирование – процесс подключения файловой системы, существующей на блочном устройстве, к корневой файловой системе.

Точка монтирования – директория, в которую монтируется устройство.

/mnt – директория для точек монтирования ФС внутренних накопителей;

/media – директория для точек монтирования ФС на съемных носителях.

Команда mount используется для монтирования ФС. Синтаксис

mount <опции> <устройство> <точка монтирования>

Команда без опций **mount** - вывести таблицу примонтированых устройств;

Отмонтировать устройства/разделы можно командами

umount <устройство>

umount <точка монтирования>

Примеры:

 # mkdir /mnt/new_disk

 # mount /dev/sda5 /mnt/new_disk

Отмонтировать корневую ФС можно

1) в однопользовательском режиме (уровень init = 1)

 # init 1

 # umount /home

 # umount /dev/sda

2) загрузившись с другой ОС, используя, например, установочный диск. При этом, для чтения ФС другая ОС должна иметь поддержку этой файловой системы. Поэтому, для чтения отмонтированной ФС ext4 удобно использовать Linux (Windows не работает с ext4).

Работа с разделом подкачки

Своппинг (подкачка) – перемещение неактивных страниц памяти из оперативной памяти на диск для экономии места в оперативной памяти (в Linux – исторически, чаще, это раздел диска, в Windows – файл pagefile.sys в корне диска). В Linux своп-раздел (или файл) также используется для гибернации (в Windows – файл hyberfil.sys в корне диска).

 # /sbin/swapon -s - вывести информацию об использовании подкачки.

Создание файла подкачки объемом 2 Гб:

dd if=/dev/zero of=swap.file bs=1k count=2097152

 # /sbin/mkswap swap.file

Создание раздела подкачки:

 # /sbin/mkswap -c <раздел>

Включить подкачку

 # swapon <файл> или <раздел>

Выключить подкачку

 # swapoff <файл> или <раздел>

Подкачка может нарушить безопасность при использовании зашифрованных данных, так как на диске могут оказаться незашифрованные данные из оперативной памяти!

Файл /etc/fstab содержит список файловых систем, монтируемых автоматически при загрузке, или монтируемых по требованию пользователя. В каждой строке присутствуют поля: имя монтируемого устройства (file system); точка монтирования (mount point); тип файловой системы (type); опции монтирования (options); опция резервного копирования (dump) с помощью утилиты dump: 1 – включено, 0 – выключено; порядок проверки раздела (pass, 0 - не проверяется, 1 - для корневого раздела, 2 - для остальных разделов).

<file system> <mount point> <type> <options> <dump> <pass>

proc /proc proc defaults 0 0

 # mount <точка монтирования> или <файл устройства> - монтирование при наличии записи в fstab.

Таблица 8. Опции монтирования файловых систем

Опция монтирования	Цель использования
defaults	опции rw, suid, dev, exec, auto, nouser, asynch
asynch	асинхронный режим ввода-вывода.

Опция монтирования	Цель использования
auto/noauto	включить/отключить автомонтирование во время загрузки
exec/noexec	разрешение/запрет на выполнение файлов
suid/nosuid	можно/нельзя устанавливать бит SUID
user/nouser	можно/нельзя монтировать обычному пользователю
ro	монтировать только для чтения
rw	Монтировать для чтения и записи

После изменения fstab применить изменения без перезагрузки можно данным образом:

mount -o remount раздел

Оптимизация производительности диска

hdparm <имя файла устройства> - вывести текущие параметры жесткого диска;

hdparm <опции> <имя файла устройства> - установить параметры жесткого диска.

Пример: вывод полной информации о диске

$sudo hdparm -i /dev/sda

/dev/sda:

Model=WDC WD10EZEX-08WN4A0, FwRev=01.01A01, SerialNo=.....

Другие опции команды

 # hdparm -t /dev/sda – тест скорости чтения с диска

 # hdparm -I /dev/sda – список опций для ускорения диска (использовать часть из них без опыта может быть опасно для диска!)

 # hdparm -M 128 /dev/sda – «тихий режим» (управление скоростью диска – число от 128 до 254)

Низкоуровневое поблочное копирование:

dd if=<входящий файл> of=<исходящий файл>

Опции:

count=<n> – количество блоков для копирования;

skip=<n> – число блоков, которые нужно пропустить с начала во входном файле;

seek=<n> – число блоков, пропускаемых в выходном файле до начала записи.

Резервное копирование всего диска целиком

 # dd if=/dev/sdX of=/dev/sdY bs=64K conv=noerror,sync,notrunc status=progress

bs - размер блока. По умолчанию 512 байт (осталось с 80-х годов). Лучше использовать значения около 1M;

noerror - продолжать работу, игнорируя все ошибки чтения. Поведение по умолчанию - dd должно останавливаться при любой ошибке;

sync - заполняет входные блоки с нулями, если есть ошибки чтения, поэтому данные в источнике и приемнике остаются синхронными, несмещенными;

status=progress - показывает статистику.

Чтобы сэкономить место, можно сжимать данные, созданные dd, с помощью gzip, например,

dd if=/dev/sdX | gzip -c> /mnt/removable_disk/image.img

Можно восстановить свой диск из образа с помощью:

gunzip -c /mnt/removable_disk/image.img | dd of=/dev/sdX

ddrescue - инструемент копирования информации с поврежденного диска. Копирование информации с такого диска выполняется в два прохода. Первый проход: скопировать каждый блок без ошибок чтения, а ошибки записать в 1.log.

 # ddrescue -f -n /dev/sdX /dev/sdY 1.log

Второй проход: три новых попытки чтения плохих блоков:

 # ddrescue -d -f -r3 /dev/sdX /dev/sdY 1.log

На новом диске следует исправить ошибки файловой системы

 # fsck -f /dev/sdY

или сначала попытаться **восстановить потерянные/частично поврежденные данные** (т.е., данные, которые утеряны из-за ошибок файловой системы, например, потери части inode) утилитами **testdisk, photorec, scalpel**.

Команда tar

tar <опции> <имя архива> <файлы и каталоги>

Извлечь файлы

tar -xvzf archive.tar.gz

Пояснение к команде:

x - извлечь файлы из архива

v - печатать имена распаковываемых файлов

z - файл является gzip-сжатым файлом

f - имя файла с архивом

j - bzip-сжатие

tar -xvjf archive.tar.bz2

Извлечь в заданную директорию

tar -xvzf archive.tar.gz -C /opt/folder/

Извлечь только заданные файлы

tar -xvz -f archive.tar.gz "./1.txt" "./2/3.txt"

Если вы хотите просто просмотреть содержимое tar-архива и не извлекать файлы, используйте параметр «-t».

Архивировать директорию (или файл)

tar -cvf my_archive.tar dir/

tar -czvf my_archive.tar.gz dir/

tar -cjvf my_archive.tar.bz dir/

Добавить файл в существующий несжатый архив (сжатый - надо распаковть и запаковать заново)

tar -rv -f my_archive.tar 555.txt

Резервное копирование с помощью tar папки ~/main_documents на примонтированный удаленный сервер

tar -cvz -f /mnt/installed_server/daily_docs_$(дата+%Y%m%d).tar.gz ~/main_documents

Команда rsync

rsync <опции> <откуда> <куда> - копирование файлов с синхронизацией

Опции:

-v – информировать о проводимых операциях;

-r – копировать данные рекурсивно (но не сохранять метки времени и права доступа);

-a – рекурсивное копирование, сохраняющее символические ссылки, права, метки времени;

-z – сжать данные файла;

-h – удобный формат чисел в выводе размер файлов;

-e протокол - использовать указанный протокол;

--partial - продолжить прерванную передачу файлов, а не начинать сначала;

--progress - показывать процент выполнения операций;

-P = --progress и --partial;

-u – пропуск файлов, имеющих в месте назначения более позднюю дату модификации.

Копирование директории на сервер и с сервера

rsync -avz dir/ user@10.0.1.214:

rsync -avz user@10.0.1.214:dir/ .

Копирование через защищенное соединение

rsync -avzhe ssh --progress user@8.8.8.8:dir/ .

Продолжить прерванную передачу файлов:

rsync -avzhe ssh -P user@8.8.8.8:dir/ .

Резервное копирование с помощью rsync (по умолчанию, старые файлы в destination получают тильду перед названием):

rsync -a -b source/ destination/

Залог сохранности данных – регулярное обязательное резервное копирование!

Места хранения резервных копий:

- Съемные носители;
- Удаленные машины / облачные сервисы.

Модуль 3. Администрирование серверных систем

Лекция 3.1. Управление программным обеспечением (ПО).

Системы управления программным обеспечением. Задачи управления ПО. Процесс управления программным обеспечением. Варианты установки ПО. Системы управления пакетами. Менеджеры пакетов rpm, yum, apt. Преимущества и недостатки системы управления программным обеспечением. Стандартные расположения установки программ. Конфликт пакетов, его разрешение. Сборка программного обеспечения из архивов с исходным кодом. Утилита configure, утилита make и файл сборки Makefile. Часто встречающиеся ошибки при компиляции из исходных кодов. Управление библиотеками.

Системы управления программным обеспечением служат для администрирования операционной системы Linux и других Unix-подобных систем. Это набор программного обеспечения, позволяющего устанавливать, настраивать, обновлять и удалять программы и их компоненты.

Задачи управления программным обеспечением – реализация удобного механизма взаимодействия программиста и пользователя, управление внутренними ресурсами системы, разработка и введение стандартов, то есть, администрирование.

Новое программное обеспечение можно установить с помощью пакетов, которые хранятся в репозиториях (рис. 22). **Репозиторий (хранилище пакетов, рис. 23)** – место хранения файлов, свободно распространяющихся в сети. В репозиториях программы хранятся уже скомпилированные с метафайлами, в которых описаны основные параметры и свойства программы, а также текущая версия установочного пакета. Такой способ установки имеет ряд минусов: не все разработчики программного обеспечения могут себе позволить содержать репозиторий, для установки требуется интернет. Однако безусловным плюсом является удобство работы с пакетами программного обеспечения и их установкой.

Второй способ, установка программ из исходных кодов, которые свободно распространяются в интернете, - способ довольно трудоемкий для начинающего пользователя. Рассмотрим его в лекции чуть позже.

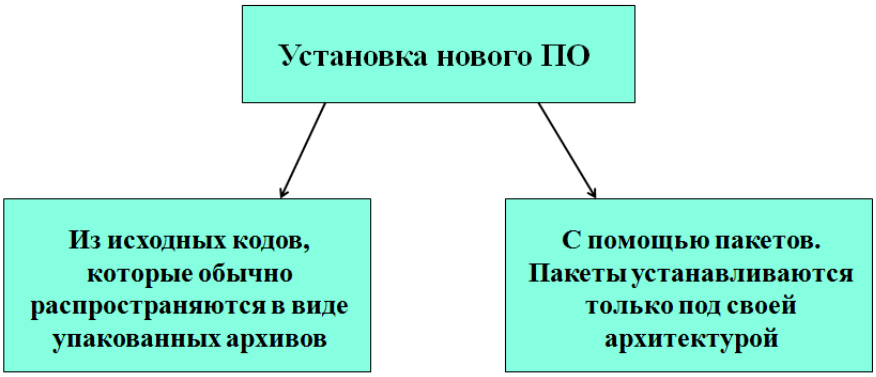

Рисунок 22. Схема вариантов установки программного обеспечения

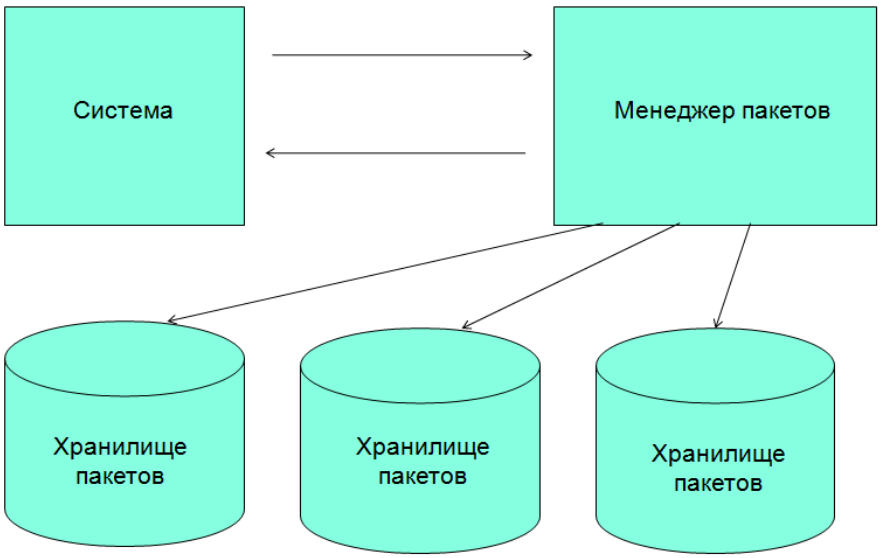

Рисунок 23. Схема работы менеджера пакетов

Начнем знакомство с **менеджерами пакетов**, и первым будет утилита под названием **rpm (RPM Package Manager)**, которая разрабатывалась для **Red Hat Linux**, позже она была перенесена и в другие дистрибутивы Linux.

Синтаксис утилиты: **rpm [режим опции] [пакет].rpm**

Режимы работы и основные опции:

-q – получение информации (запрос);

-i – установка;

-U –обновление;

-V – проверка пакетов;

-e – удаление.

-v – показать подробную информацию.

-i – получить информацию о пакете (**-qi**);

-I – список файлов пакета (**-qI**);

-c – список конфигурационных файлов пакета (**-qc**);

-a – список пакетов, установленных в системе;

--force – выполнять действие принудительно;

--nodeps – не проверять зависимости;

--test – проверить последствия удаления или установки пакета.

Yellowdog Ubdater, Modified (YUM) – консольный менеджер RPM-пакетов, представляющий собой оболочку для **RPM**, распространяется под лицензией **GNU**.

Основные команды **yum**:

#yum help – список команд и опций;

#yum list – список названий пакетов из репозиториев;

#yum list installed – список установленных пакетов.

#yum repolist – список подключенных репозиториев;

#yum install [пакет] – установить пакет;

#yum remove [пакет] – удалить пакет;

#yum update [пакет] – обновить пакет;

#yum info [пакет] – вывести информацию о пакете;

#yum search [строка] – найти пакет по строке в описание.

В современных дистрибутивах, основанных на RedHat, применяется обновленный. более быстрый по сравнению с yum, менджер пакетов **dnf**. Он имеет сходный синтаксис с yum.

Advanced Packaging Tool (apt) – утилита для управления ПО в операционных системах **Debian**, а так же системах основанных на **Debian**, таких как **Ubuntu** и **Mint**. Утилита имеет широкое распространение, есть как консольные версии, так и большое число графических оболочек, таких как **Synaptic** и **Adept**.

Синтаксис утилиты: **apt [опции] [команда] [пакет]**.

Полезные опции для apt:

-t – версия релиза для которой устанавливать пакет;

-f – выполнить операцию принудительно;

-o – строка конфигурации.

Основные команды apt:

download – скачать пакет без установки;

update – обновление информации о пакетах в репозиториях;

upgrade – обновление системы без удаления пакетов;

search – поиск пакетов в локальной базе;

show – просмотр информации о пакете;

install – установка пакета;

remove – удаление пакета;

purge – полное удаление пакета (вместе с конфигурационными файлами).

Рассмотрим и некоторые другие утилиты:

dselect – программа управления с графическим интерфейсом. Является старейшим фронтендом к **dpkg**. На сегодняшний момент практически полностью вытеснена **Advanced Packaging Tool**.

aptitude – оболочка с графическим интерфейсом для **Advanced Packaging Tool**.

dpkg – основное программное обеспечение системы управления пакетами **Debian**, а так же систем со схожей архитектурой.

dpkg -i [пакет].deb – установка пакета **.deb**.

Расположение установленных программ в файловой системе

Во время установки программного обеспечения, система управления распределяет файлы программы по следующим системным папкам:

/usr/bin – исполняемые файлы программ;

/usr/sbin – исполняемые файлы, запускаемые с правами администратора;

/usr/lib – библиотека программы;

/usr/share – остальные файлы программы.

/usr/local – ПО, собранное самостоятельно из исходных кодов.

С помощью команды **“dpkg -s [название пакета]**” можно получить список файлов и их местонахождение в системных папках.

Теперь разберём, как устанавливать программы с открытым исходным кодом. Этот способ установки программ подходит для компьютеров, которые не имеют доступа в Интернет. Обычно программы с исходным кодом распространяются в виде упакованных архивов, которые перед установкой необходимо распаковать, используя, например, следующие команды:

tar -Jxvf [пакет].tar.xz

tar -zxvf [пакет].tar.gz

Дальше запускается файл конфигурации:

./configure

Скрипт оболочки **configure** будет пытаться подобрать правильные значения переменных, использующихся в процессе компиляции. Эти переменные необходимы для создания **Makefile** в каждом каталоге пакета, набор заголовочных файлов, содержащих системные зависимости.

Переходим к установке:

make install

Удаление:

make uninstall – удаление установленных файлов (обратно make install)

make clean – удаление скомпилированных файлов в месте сборки (обычно, - очистка каталога build)

Пример Makefile с пояснениями приведен ниже.

```
# объявление переменной
# -c - собрать объектный файл. -Wall - включить все предупреждения
CFLAGS=-c -Wall
# цель по умолчанию (выполняется, если команда make вызвана без указания цели)
all: project
# это - цель сборки. После двоеточия указывают, от чего она зависит.
# Отсутствие зависимости позволяет не перестраивать весь проект при изменении одного файла с кодом.
project: 1.o 2.o 3.o
# в следующей строке обязателен отступ (TAB). Это команды (компиляции)
	$(CC) 1.o 2.o 3.o -o project
1.o: 1.cpp
	$(CC) $(CFLAGS) 1.cpp
2.o: 2.cpp
	$(CC) $(CFLAGS) 2.cpp
3.o: 3.cpp
	$(CC) $(CFLAGS) 3.cpp
# цель для очистки проекта (удаление скомпилированных файлов)
clean:
	rm -rf *.o project
```

Часто при компиляции из исходных кодов конфигуратор (**configure**) не может найти ту или иную библиотеку. Иногда названия библиотеки может не совпадать с названием пакета в репозитории. В таком случае можно 1) найти недостающую библиотеку

apt-cache search ключевое_слово

2) воспользоваться системами управления с графическим интерфейсом, например, **synaptic**, который выполнит поиск нужной библиотеки внутри пакетов.

Ошибки могут возникнуть при **конфликте пакетов друг с другом**, тогда для решения проблемы придётся вручную удалять наименее важные пакеты и проверить целостность оставшихся.

Исполняемые программы в Linux делятся на два типа:

1. Статически скомпонованные.

+уже содержат в себе все необходимые библиотечные функции.

+не требуется предварительная установка необходимых компонентов.

-занимают больше места по сравнению с динамически скомпонованными.

2. Динамически скомпонованные.

+занимают меньше места по сравнению со статически скомпонованными.

- требуют установку дополнительных компонентов (библиотек).

Расположение библиотек в Linux

Системное программное обеспечение стандартно устанавливается в следующих расположениях:

- исполняемые файлы

/bin, /sbin, /usr/bin, /usr/sbin

- библиотеки

/lib, /usr/lib, /lib64, /usr/lib64

Как правило, в 64-разрядных ОС Linux поддерживаются и 32-, и 64-разрядные программы. **Путь к библиотекам для 64-разрядных библиотек: /lib64, /usr/lib64**, а к 32-разрядным /lib, /usr/lib.

Пользовательское программное обеспечение может быть установлено в системные расположения, если оно предоставляется в системе как пакет (т.е., возможна его установка через менеджера приложений) или каталог /opt (традиционно, поскольку в прошлые десятилетия пользовательское ПО часто устанавливали с оптических дисков).

Библиотеки в свою очередь делятся на статические (static library) и динамические (shared library). Статические библиотеки используются на этапе сборки программы, а динамические во время её выполнения.

Каждая разделяемая библиотека Linux имеет расширение so.X.Y.Z. **X - версия публичного интерфейса (API)**, считается, что каждый раз, когда он меняется, **программа теряет обратную совместимость** (нельзя работать с новой версией, если программа использует стары интерфейс). **Y - ревизия**, обычно добавляет **новые функциональные возможности к программе**. При смене ревизии публичный **интерфейс не меняется**, то есть программы, работающие со старой версией, смогут работать и с новой. **Некоторые части интерфейса могут быть признаны устаревшими (deprecated), но не могут быть удалены**, так как это нарушит API. **Z - патч**, то есть обновленная версия программы, **исправляющая ошибки, улучшающая реализацию/работоспособность**. Патч, кроме того, не меняет никоим образом публичный интерфейс, и **не добавляет новые функции**, поэтому программа с патчем

и без не только обратно совместима, но и **прямо совместима** - старая версия может гарантированно быть использована вместо новой с точки зрения совместимости (как правило, старую версию использовать нежелательно с точки зрений наличия ошибок/пониженной безопасности/меньшей производительности).

Общие библиотеки:

/etc/ld.so.cache - файл, указывающий, где искать динамические библиотеки по умолчанию (общие библиотеки);

/etc/ld.so.conf - конфигурационный файл для генерации /etc/ld.so.cache;

ldconfig - утилита для генерации /etc/ld.so.cache.

LD_LIBRARY_PATH - переменная для добавления путей к библиотекам (в случае совпадения имен приоритетом обладают библиотеки, подключенные через LD_LIBRARY_PATH)

Ключевые системные библиотеки:

linux-vdso.so.1 - виртуальный динамический разделяемый объект - библиотека для быстрого использования системных вызовов;

/lib/ld-linux.so.2 и /lib64/ld-linux-x86-64.so.2 - исполняемый файл, отвечающий за определение необходимых динамических библиотек для запущенного приложения, и их связывание с приложением.

Иногда ошибки могут возникать при запуске программ (отсутствие библиотек)

Не найдена разделяемая библиотека: **Error loading shared library libopenblas.so.3: No such file or directory**

Решение:

$ ldd «исполняемый файл, использующий разделяемые библиотеки»

В списке библиотек ненайденные помечены будут как «Not Found».

$ sudo apt install apt-file

$ apt-file update

$ apt-file find [название ненайденной библиотеки]

$ sudo apt install [название пакета]

Централизованное управление программным обеспечением (ПО)

Puppet – кроссплатформенная клиент-серверная программа для централизованного управления ПО на множестве компьютеров. Используется для администрирования в крупных компаниях, позволяет легко настроить сеть и управлять ей на основе различных операционных систем.

Лекция 3.2 Системные журналы. Процесс загрузки и уровни выполнения.

Процесс загрузки и уровни выполнения. Конфигурирование службы syslog (system logger — система журналирования). Источники сообщений. Приоритеты. Ротация журналов. Последовательность процесса загрузки. Загрузчик grub (grand unified bootloader — унифицированный загрузчик с большими возможностями). Администрирование grub в Ubuntu Linux. Уровни выполнения — стандарт System V. Настройка автоматического запуска процессов инициализации. Запуск служб вручную. Остановка и перезагрузка системы.

В системах Linux существует 5 основных стадий загрузки ОС:

1. **BIOS** – отвечает за базовый ввод и вывод данных с устройства на устройство. Загружает главную загрузочную запись (**MBR = Master Boot Record)** - в BIOS всегда задан пользователем или по умолчанию порядок опроса дисков (Boot priority, boot sequence). Загрузка происходит с первого диска, на котором найдена действительная запись MBR.

2. **MBR** – основная загрузочная запись на диске (диск может быть **/dev/hdX** или **/dev/sdX**). Вызывает загрузчик ОС (в Ubuntu Linux - Grub). Имеет размер 512 байт.

Загрузочная запись состоит из кода загрузчика (446 байт), таблицы разделов и сигнатуры, указывающей на корректность загрузочной записи (всегда 55h AAh, «h» означает шестнадцатеричный код). Из-за ограниченного размера загрузочной записи таблица разделов может содержать не более четырех разделов по 16 байт. Каждый раздел имеет, поимоо других атрибутов, свой код: 82h - Linux swap, 83h - Linux, 85h - Linux extended (расширенный). Вместо одного из разделов может присутствовать расширенный раздел, который указывает на расширенную загрузочную запись. Каждая расширенная загрузочная запись (EBR), в свою очередь, содержит указатель на следующую EBR, таким образом, число разделов не ограничивается.

Если вместо BIOS используется более новый EFI, то вместо MBR используется таблица разделов GPT. Она состоит из 34 блоков по 512 байт, записанных симметрично с двух концов диска для дублирования и облегчения восстановления повреждений. Блоки имеют номер от 0 до 33, блок 0 соответствует MBR для совместимости.

3. **GRUB (Grand Unified Bootloader)** – загрузчик операционной системы под лицензией **GNU**. Позволяет выбирать загружаемую ОС (Windows, Linux и др.) и используемое ядро Linux среди установленных.

Возможности загрузчика Grub.

- Поддерживает работу с файлами.
- Поддерживает файловые системы: **ext2, ext3, ReiserFS, XFS, JFS, FAT32, FAT16, NTFS, ISO** и многие другие
- Загружает ядра **GNU/Linux, GNU/HURD, FreeBSD, SUN Solaris.**
- Осуществляет загрузку **Windows** (передаёт управление другим загрузчикам).
- Обладает встроенной командной оболочкой.
- Поддерживает загрузчик **EFI** (с версии 1.98).

- Защита паролем пунктов меню.
- Поддерживает загрузку через сеть по протоколу **TFTP** и **BOOTP**.

Администрирование Grub.

/boot/grub/menu.lst или **/boot/grub.conf** – последовательность команд, выполняемых **grub**. Эти файлы генерируются на основе пользовательского файла **/etc/default/grub**.

sudo update-grub - обновить **/boot/grub/menu.lst** и **/boot/grub.conf** на основе **/etc/default/grub.**

<c> – перейти в оболочку **grub** (во время загрузки).

<b> – продолжение загрузки.

Команды grub:

help – помощь.

geometry (hd0) – определить структуру **HDD**.

root (hd0, 1) – раздел, с которого читается загрузочная запись (**0 диск, 1 раздел**).

setup (hd0) – установить загрузчик в главную загрузочную запись.

quit – выйти из командной оболочки **grub**.

default – образ загрузки по умолчанию.

timeout – задержка в секундах перед загрузкой образа по умолчанию.

splashimage=(hd0,4)/boot/grub/splash.xpm.gz -- фоновое изображение при загрузке.

title – описание образа для загрузки.

initrd – имя образа начального электронного диска.

chainloader (file) – передать управление другому загрузчику (указать сектор).

Установка:

/sbin/grub-install – установить загрузчик grub из командной оболочки bash.

4. **Kernel** – ядро операционной системы. Запускает программу **/sbin/init**.

5. **Init** – система инициализации в **System V** (ОС Linux, выпущенная в 80-х годах компанией AT&T. Инициализация служб по образцу System V на долгие годы становилась стандартом инициализации ОС Linux), которая просматривает файл **/etc/inittab** для определения уровней выполнения и запускает остальные процессы. Команда **init** позволяет переключит уровень выполнения **runlevel**:

init 5

Runlevel – уровень запуска и выполнения служб.

0 — выключение системы;

1 — однопользовательский режим (только-root);

2 - обычно эквивалентен уровню 3. На некоторых системах (RedHat) не поддерживает NFS (см. лекцию 3.3);

3 — многопользовательский режим;

4 — в некоторых дистрибутивах, уровень графического сеанса;

5 — в большинстве дистрибутивов - уровень графического сеанса (в т.ч., для Ubuntu);

6 — перезагрузка системы.

/sbin/runlevel - узнать текущий уровень выполенния.

На конечном этапе загрузки происходит изменение runlevel от однопользовательской ОС (1) до многопользовательской ОС (3) с графическим интерфейсом (5). Серверные ОС остаются в режиме выполнения 3.

Уровни выполнения, журналирование с помощью syslog - это элементы концепции **System V (*"System Five"*)**. Сейчас в большинстве дистрибутивов операционной системы Linux они частично или полностью заменены службой **systemd**, но обычно могут быть использованы и установлены параллельно.

На каждом уровне выполнения при переходе на него, а также при завершении работы на нем, выполняются скрипты в каталогах **/etc/rc0.d - /etc/rc6.d** (это настроено в файле **/etc/inittab**), цифра соответствует уровню выполнения.

Для добавления в ОС собственной автозапускаемой программы **command1** используйте команду

sudo update-rc.d command1 start 70 2 3 4 5 . stop 30 0 1 6 .

70 и 30 - приоритет запуска (меньшее число означает более раннюю загрузку и/или завершение).

2 3 4 5 - уровни работы программы;

0 1 6 - уровни. где программа не будет запущена.

Скрипт должен быть создан на основе файла /etc/init.d/skeleton.

Для всех уровней загрузки автозапуск можно настроить путем внесения строки в файл **/etc/rc.local**.

Служба syslog

Syslog – стандарт отправки и регистрации сообщений о происходящих в системе событиях, использующихся в компьютерных сетях, работающих по протоколу **IP**.

syslogd – демон службы **syslog**.

/etc/syslog.conf – файл конфигурации службы **syslog**.

Все службы и системные приложения Linux раньше относились к четко определенной группе. Принадлежность к группе определялась в соответствии с задачами, решаемыми приложением. Если приложение обрабатывало почту, оно относилось к группе mail, если оно отвечало за вопросы безопасности системы, то относилось к группе security и т.д. В файле конфигурации **/etc/syslog.conf** описываются действия (*action*) при получении сообщений с

определенным приоритетом (*priority*), принимаемых от определенных групп (источников сообщений *facility*, рис. 24). Можно использовать шаблон глобальной подстановки «*» (*glob, wildcard*), указывая вместо конкретного названия, что подходит любой приоритет сообщения (см. ниже «Способы фильтрации сообщений»). Например, строка в файле конфигурации

mail.err /var/log/mail.err

означает прием сообщений группы «mail» с приоритетом «err» и запись их в файл /var/log/mail.err.

Отправить сообщение процесс может, например, использовав утилиту logger:

logger -p mail.err -t "MAIL DELIVERY FAILURE" 'Mail delivery failed to the following recepients: user@gmail.com'

-p - селектор (источник + приоритет);

-t - тема (заголовок) сообщения;

последним параметром утилите logger передается сам текст сообщения.

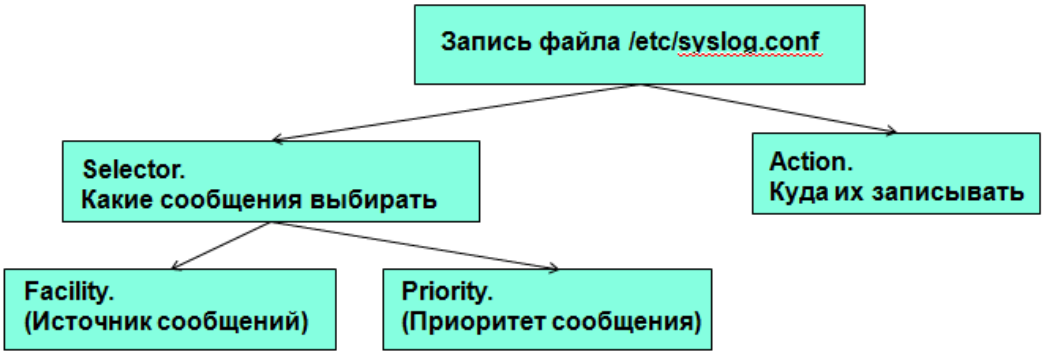

Рисунок 24. Схема конфигурации syslog

Источники сообщений:

/var/log – директория, основное место хранение лог-файлов (системных журналов).

/var/log/syslog или **/var/messages** – глобальный системный журнал, который содержит записи с момента запуска системы (запуск ядра, различных служб, обнаруженные устройства, интерфейсы и многое другое).

/var/log/auth.log или **/var/log/secure** – записи об авторизациях пользователей, включая неудачные попытки.

/var/log/dmesg – драйвера устройств.

/var/log/cron – сообщения служб **cron** и **at**.

/var/log/daemon –сообщения от демонов (служб).

/var/log/kern – сообщения ядра.

/var/log/lpr – сообщения службы печати.

/var/log/user – сообщения пользовательских программ.

/var/log/syslog – собственные сообщения службы **syslog**.

Приоритеты:

debug – отладочная информация.

info – информационное сообщение.

notice – основные события.

warning (warn) – предупреждение.

err (error) – ошибка.

crit – критическое событие.

alert – требуется немедленное вмешательство.

emerg (panic) – система не работоспособна.

Основные способы фильтрации сообщений:

[источник].* – все записи.

[источник].=[уровень] – только записи указанного уровня.

[источник].[уровень] – все записи не меньше указанного уровня.

[источник].![уровень] – все записи не меньше указанного уровня.

[источник].!=[уровень] – все записи, кроме указанного уровня.

Ротация системных журналов

В каталоге **/var/log** находятся архивы системных логов, так как эта информация может быть довольно громоздкой, имеется команда для ротации (**logrotate**) этих файлов, благодаря чему нет перемешивания новых и устаревших записей. Команда **logrotate** запускается автоматически, но есть возможность запустить её вручную. Команда нумерует файлы добавляя **“.[число]”** в конец файла, а также может удалять, сжимать, создавать и посылать по почте файлы журналов.

dmesg – беглый обзор информации о последней загрузки.

tail – последние записи в журнал.

more – постраничный просмотр.

less – просмотр и поиск информации.

logger – помещение в журнал своих сообщений.

Ротация журналов настраивается в общем файле /etc/logrotate и специфичных файлов для отдельных служб внутри директории /etc/logrotate.d. Доступны следующие опции:

weekly, dayly, monthly – периодичность ротации.

rotate <n> – кол-во хранимых старых журналов.

create 0666 root – с какими правами создавать новый журнал.

compress – сжимать файлы журналов.

notifempty – запрет ротации пустых журналов.

include <file> – включить файл или множество файлов.

mail – посылать журналы по e-mail.

size – максимальный размер журнала.

<имя журнала> {- индивидуальные настройки для журнала}

Пример ежедневной ротации лог-файла /var/log/filefire/main.log для некоторой псевдослужбы «filefire»:

```
/var/log/filefire/main.log {
    daily
    rotate 3
    postrotate
        filefire reload > /dev/null
    endscript
    compress
}
```

Запуск служб вручную (System V)

/sbin/service network или /etc/inid.d/network start – запуск службы network вручную.

/sbin/chkconfig -list – получить информацию о процессе начальной загрузки.

/sbin/chkconfig --level 2345 network on – включить исполнение network на уровнях 2345.

/sbin/chkconfig --add <служба> – добавить службу.

/sbin/chkconfig --del <служба> – удалить службу.

/sbin/chkconfig [level levels] on | off | reset – включить, выключить, перезапустить.

/sbin/runlevel или who -r – определить текущий уровень выполнения.

/sbin/init 1 – перейти на уровень выполнения 1.

Завершение работы и перезагрузка ОС

/sbin/init 0 или **halt** – немедленное завершение работы ОС.

/**sbin/init 6** или **reboot** – немедленная перезагрузка ОС.

halt -p или **poweroff** – завершение работы с отключением питания.

/sbin/shutdown [опции] [время] [сообщение] – команда выключения, перезагрузки, остановки, завершения сеанса локальных или удалённых компьютеров.

Опции shutdown:

-c – отменить начавшуюся остановку системы.

-a – создать файл **/fastboot** и не проверять файловую систему при загрузке.

-F – создать файл **/forcefsk** – проверить файловую систему при загрузке.

-h – остановка системы.

-r – перезагрузка системы.

-R – послать пользователям сообщение, но не перезагружать систему.

Примеры:

/sbin/shutdown -r 16:00 'Reboot at 16:00!' – перезагрузка в 16:00.

/sbin/shutdown -h now – остановка прямо сейчас.

/sbin/shutdown -h +10 – остановка через 10 минут.

Работа с systemd

systemd имеет два основных преимущества по сравнению с System-V: параллельный запуск служб при инициализации ОС, запуск служб по требованию.

В systemd единицей взаимодействия и настройки является юнит. Базовые виды юнитов:

service — управляет демонами;

socket — конфигурационный файл сокета;

device — конфигурационный файл с правилом udev для дерева устройств;

mount — отвечает за монтирование ФС;

automount — отвечает за автоматическое монтирование файловой системы.

Запуск служб вручную (systemd)

Запустить юнит:

systemctl start <unit>

Остановить юнит:

systemctl stop <unit>

Перезагрузить юнит:

systemctl restart <unit>

Перезагрузить конфигурацию юнита:

systemctl reload <unit>

Проверка статуса юнита:

systemctl status <unit>

Сделать юнит активным (можно запускать)

systemctl enable <unit>

Сделать юнит неактивным (нельзя запускать)

systemctl disable <unit>

Для отладки работы юнитов можно просмотреть только информацию об юнитах с проблемами:

systemctl --failed

Общий журнал расположен в /var/log/journal.

Вывести журналы, записанные с начала работы системы:

journalctl -b

Все сообщения утилиты systemd:

journalctl /usr/lib/systemd/system

Вывести журнал по PID процесса (используется при отладке работы юнитов):

journalctl _PID=ЧИСЛО

Создание юнитов

Системные юниты находятся в директории /usr/lib/systemd/system, а пользовательские юниты - в директории /etc/systemd/system.

Для юнитов создаются файлы конфигурации «.service». Юнит имеет обязательные секции «[Unit]» и «[Install]», а также «[Сервис]». При отсутствии файла юнита, но наличии файла System V с тем же именем без суффикса «.service», юнит будет создан «на лету». Также, для сервиса Test файлы с суффиксом «.conf» из каталога Test.service.d будут прочтены после разбора основного файла конфигурации юнита.

Пример файла юнита test1

[Unit]

Description=Test

[Service]

ExecStart=/usr/sbin/test1

[Install]

WantedBy=multi-user.target

Важные опции раздела [Unit]

Required= - настраивает зависимости в виде жестких требований. Если какой-либо юнит, на который опирается текущий юнит, не был запущен, то текущий юнит не будет запускаться.

Wants= - слабое требование. Юнит попробует стартовать, если какая-либо зависимость не была запущена.

Before =, After = - настройка зависимостей между юнитами.

Важные опции раздела [Install]

RequiredBy - обратная (какие юниты зависят от текущего) жесткая зависимость;

WantedBy - обратная мягкая зависимость;

sudo apt install syslog-ng - обеспечивает включение syslog в systemd наряду с journalctl.

В systemd для использования /etc/rc.local должен быть включен сервис rc-local.service:

systemctl status rc-local.service

Лекция 3.3 Сетевые службы Linux.

Службы сети. Удаленный доступ: SSH (secure shell — удаленное управление операционной системой по защищенному каналу посредством командной оболочки), VNC (virtual network computing — система удаленного доступа к рабочему столу). Передача файлов: FTP (file transfer protocol — протокол передачи файлов). Настройка FTP-сервера. Клиент FTP. Команда wget. Браузеры для консольной строки. Сетевая файловая система NFS (network file system). Система печати CUPS (common unix printing system — общая система печати UNIX). Совместная работа Windows и Linux компьютеров сети: пакет SAMBA.

Сетевая служба (*сервис, демон*) - специальный процесс (или взаимосвязанных ряд процессов) операционной системы, отвечающий за работу с локальными или удаленными соединениями заданного типа.

Сетевая служба создает **сокет** – пара ip-адрес + номер порта и привязывает сокет к данному порту в операционной системе. Говорят, что в таком случае служба выполняет **прослушивание порта**, т.е., ожидает запросов на установку соединения через этот порт. **Порт** - целое число, имеет номер от 0 до 65535. Номера портов от 1 до 1023 («общеизвестные порты») зарезервированы организацией IANA (Internet Assigned Numbers Authority) под стандартные службы. Порты с номерами от 1024 до 49151 также должны проходить регистрацию в организации. Это обеспечивает то, что никакие две службы не будут пытаться прослушать один порт. Порты с большими номерами являются динамическими.

Некоторые общеизвестные порты и протоколы, по которым осуществляется работа через эти порты.

- 20 – данные, передаваемые через протокол FTP;
- 21 – управляющая информация, передаваемая через протокол FTP;
- 22 – протокол удаленного шифрованного соединения SSH;
- 23 – удаленное соединение Telnet;
- 25 – почтовый протокол SMTP (отправка писем);
- 53 – протокол DNS;
- 80 – протокол HTTP;
- 110 – почтовый протокол POP3 (прием писем с сервера);
- 143 – почтовый протокол IMAP (прием и отправка заголовков писем);
- 443 – протокол HTTPS;
- 465 – шифрованный протокол SMTP;
- 993 – шифрованный протокол IMAP;
- 995 – шифрованный протокол POP3.

В файле /etc/services можно найти полный список сетевых служб. Здесь хранятся имя службы, номер порта, используемый протокол, псевдонимы службы.

netstat -an - получить список активных соединений.

Опции netstat:

-a – выводить спсиок портов;

-n – выводить информацию в виде чисел;

-t – только TCP-порты;

-u – только UDP-порты.

В выводе команды можно видеть следующие состояния соединения:

LISTEN (LISTENING) - идет прослушивания порта, создан сокет;

ESTABLISHED — установленные соединения.

Создание удаленного подключения telnet:

telnet имя_или_адрес_компьютера

В Unix было популярно использовать утилиты rsh/rlogin для входа на удаленный компьютер. В файле /etc/hosts.equiv – доверенные компьютеры, вход с которых осуществляется для заданных пользователей без пароля при входе чреез rsh/rlogin. Но эти утилиты не используют шифрование (как и telnet) и в настоящий момент их категорически **не следует применять**.

Связь с удаленным компьютером через зашифрованное соединение

Пакет OpenSSH:

- sshd – сервер службы SSH, прослушивает порт 22 (TCP);
- ssh – клиент службы SSH, позволяет начать сеанс связи с удаленным компьютером;
- scp – программа для удаленного копирования;

/etc/ssh/sshd_config – конфигурационный файл настройки сервера;

/etc/ssh/ssh_config – настройка клиентов ssh и scp.

ssh имя_пользователя@имя_или_адрес_компьютера - начать работу с удаленным компьютером. Разрешение имен в ip-адреса производится с использованием файла **/etc/hosts**.

Копирование файлов через ssh:

scp пользователь@компьютер:<откуда-копировать> <куда-копировать>

Пример:

scp 1.txt user@10.0.0.100:/home/user/shared_docs/

Традиционно аутентифицируются либо по паролю (менее безопасно), либо через связку «публичный» + «приватный» ключ.

Для аутентификации используются протоколы шифрования RSA (в примере) или DSA:

ssh-keygen -t rsa - создание пары ключей (открытый и закрытый) асимметричного шифрования. Публичный (открытый) ключ ~/.ssh/id_rsa.pub передается на все компьютеры, с

которыми должна осуществляться связь, его содержимое копируется как строка в файл авторизованных (которые разрешены для авторизации) ключей

cat id_rsa.pub >> ~/.ssh/authorized_keys;

Этот файл находится в директории .ssh в домашней папке пользователя, под которым вы будете входить на удаленном компьютере. На файл authorized_keys, должны быть выставлены права 600.

chmod 600 ~/.ssh/authorized_keys

На саму директорию .ssh - права 700:

chmod 700 ~/.ssh

Наконец, владельцем директории .ssh и файлов в ней должен быть тот пользователь, под которым вы будете входить.

Приватная часть ключа (закрытый ключ) ~/.ssh/id_rsa строго никому никогда не передается, это ваша личная конфиденциальная информация, которую вы вместо пароля используете для входов на удаленные машины.

При подозрении на то, что приватный ключ скомпрометирован (доступ к нему получили третьи лица) **следует немедленно создать новую пару ключей и удалить все старые публичные ключи на удаленных машинах**, чтобы третьи лица, завладевшие вашим утерянным приватным ключом, не смогли получит к ним доступ. **Аналогично следует поступать и при компрометации пароля пользователя** (установить новый пароль).

~/.ssh/authorized_keys – публичные ключи для безпарольного доступа.

~/.ssh/known_hosts – известные хосты, с которых подключение осуществляется без дополнительного подтверждения.

Протокол передачи файлов FTP

ftp < адрес_сервера> - синтаксис команды;

sftp < адрес_сервера> - ftp через **зашифрованное** соединение (через ssh). **Всегда используйте это команду вместо незашифрованоного ftp**;

Команды интерактивного режима ftp/sftp:

ftp> help – вызов справки;

ftp>? – аналог help;

ftp> ls – вывод списка файлов и директорий; **! ls** выполняет команды для локального, а не удаленного, компьютера;

ftp> cd 123 – перейти в каталог 123;

ftp> cdup – аналог cd .. в Linux

ftp> ascii – передача файлов в текстовом режиме ASCII;

ftp> binary – передача файлов в двоичном режиме (нельзя передавать двоичные данные в текстовом режиме – приводит к потере данных!);

ftp> get 1.jpg – загрузить файл 1.jpg с удаленного компьютера (без подтверждения перезаписи!)

ftp> put 1.jpg – загрузить файл 1.jpg на удаленный компьютер (без подтверждения перезаписи!)

ftp> mget *.txt - загрузить файлы с удаленного компьютера

ftp> mput *.txt - загрузить файлы на удаленный компьютер;

ftp> delete 1.txt - удалить файл 1.jpg с удаленного компьютера;

ftp> mdelete *.txt – удалить все текстовые файлы с удаленного компьютера.

Копирование файла в неинтерактивном режиме (аналог scp):

sftp user@10.0.0.100:docs/1.txt ~/

Утилита wget

wget <адрес_или_шаблон> - загрузка файлов с удаленного адреса;

Опции:

-c – возобновление загрузки в случае разрыва соединения;

-i <file> – взять адреса загружаемых объектов из файла;

-r – рекурсивная загрузка каталогов;

-l <num> – ограничить рекурсию до уровня num;

-x – создавать необходимую структуру каталогов;

--passiveftp – использовать пассивный режим ftp.

Пример использования:

wget -c http://releases.ubuntu.com/16.04.4/ubuntu-16.04.4-desktop-amd64.iso - загрузить образ диска Ubuntu 16.04 (система с 5-летней поддержкой до апреля 2021 года)

NFS (Network File System) – протокол сетевого доступа к файловым системам. Универсален, не зависит от ОС. Используется протокол RPC для обмена данными между клиентом и сервером. Ресурсы nfs можно монтировать при помощи команды mount.

/etc/exports - файл настройки nfs. Возможное содержимое файла:

/home/user/dir 10.1.12.0/24 (ro,async,all_squash,insecure) - предоставить каталог в режиме «только для чтения» для узлов сети 10.1.12.0/24.

sync/async – синхронный/асинхронный режим доступа - определяет, возможен ли ответ сервера до окончания записи на диск изменений, async может привести к потере данных при обрыве соединения;

all_squash – опция означает, что подключения будут выполнятся от имени анонимного пользователя.

sudo apt install nfs-kernel-server nfs-common - установка поддержки NFS в Ubuntu;

showmount -e - вывести список ресурсов, предоставленных в системе;

/etc/init.d/nfs-kernel-server restart - перезапуск службы nfs (Ubuntu);

exportfs -a - считать заново настройки /etc/exports;

mount -t nfs 10.5.122.221:/home/user /mnt/uhome - монтирование ресурса NFS;

Возможно записать в /etc/fstab информацию для удобства монтирования NFS-ресурсов через нажатие кнопки на панели дисковых устройств в Nautilus:

10.5.122.221:/home/user /mnt/uhome nfs user,rw,noauto 0 0

Служба печати CUPS

sudo apt install cups - установка CUPS через менеджер пакетов apt.

Служба обслуживания очереди заданий на печать имеет название cupsd. Она должна быть запущена для работы CUPS. Конфигурационные файлы CUPS находятся в директории /etc/cups. **Перед изменением создайте их резервные копии!**

- **cupsd.conf – основной файл настроек сервера.**
- printers.conf – описания и настройка принтеров.
- classes.conf – описания классов (групп принтеров).
- client.conf – индивидуальные настройки для клиентов (формат файлов схож с форматом сервера Apache). В данном файле должна быть строка **ServerName server.example.com:port с указанием адреса сервера печати.**

С помощью команды

sudo /etc/init.d/cups restart

осуществляется перезапуск службы cupsd после изменений конфигурации.

В основном файле конфигурации должны присутствовать следующие строки, которыми устанавливается прослушивание порта по умолчанию на сервере 10.0.10.129, разрешается удаленный доступ к средствам печати (Allow @LOCAL):

Listen 127.0.0.1:631

Listen 10.0.10.129:631

<Location />

Order allow,deny

Allow @LOCAL

</Location>

Веб-интерфейс для конфигурирования доступенпо адресу http://localhost:631. Также для конфигурирования существует утилита командной строки lpadmin.

Примеры:

lpadmin -p laser -o job-page-limit=10 - установить ограничение в 10 страниц;

lpadmin -p laser -u allow:stud1 - разрешить печать пользователю stud1;

lpadmin -p laser -o Resolution=600dpi - изменить разрешение печати;

lpoptions -l - получить текущие настройки.

Печать с помощью CUPS документа test.txt осуществляется командой

lp test.txt

Опции lp:

-d – имя принтера;

-h – имя узла сети;

-i – идентификатор задания;

-n – количество копий;

-q – приоритет задания;

-u – имя пользователя;

-H – дополнительные опции задания;

-P – список страниц для печати.

lp -d HP1 -n 3 1.txt - печать трех копий файла.

lpstat -a - состояние очереди печати;

lp -i HP-10 -H immediate - перемещение задания в очереди на первую позицию (-H immediate);

cancel - снять задание с печати.

Пакет SAMBA позволяет Linux-компьютеру работать в сети Windows, и предназначен для общего использования ресурсов: файлов и каталогов, а также устройств.

sudo apt install samba - установка SAMBA в Ubuntu Linux.

/etc/samba/smb.conf - основной файл настройки.

Для печати важны следующие **настройки smb.conf**:

workgroup = COMP

...

security = user

…

[printers]

….

browsable = yes

guest ok = yes

Для Windows-компьютеров следует добавить опцию

use client driver = yes

Применение изменений выполняется командами

sudo restart smbd

sudo restart nmbd

Пример части основного файла конфигурации SAMBA

[global]

netbios name = IKT-201

workgroup = COMP

security = user

guest account = nobody

[homes]

comment = Home directories

read only = no

[PUB]

comment = public share

path = /home/samba/pub

read only = no

guest ok = yes

[STUD]

path = /home/samba/student

browseable = no

valid users = student

Важные настройки в файле конфигурации:

- global – раздел общих настроек;
- printers – описывает методы подключения к принтерам;
- homes – настраивает доступ к домашним каталогам пользователей;
- PUB – настройки для публичного доступа;
- STUD – настройки для пользователя student.
- netbios name – имя компьютера в сети NetBIOS;
- workgroup – название рабочей группы NetBIOS;
- security – идентификация пользователей или через использования домена;
- share – права доступа проверяются при подключении к ресурсу, но после подключения к компьютеру, предоставляющему ресурс;
- user – права доступа при подключении к компьютеру, предоставляющему ресурсы;
- domain – данный компьютер является членом домена NT, аутентификация производится на контроллере домена (PDC);

- server – задает имя иного сервера аутентификации;
- guest account – имя анонимного пользователя;
- comment – необязательный комментарий;
- path – путь к разделяемому ресурсу, или к очереди печати принтера;
- read only – доступ только для чтения;
- guest ok – разрешение гостевого доступа;
- browseable – разрешение отображать ресурс в сетевом окружении.

Настройка SAMBA может быть осуществлена как редактированием файла smb.conf, так и с помощью утилит, например, SWAT.

Подключение через SAMBA

testparm -s | less - проверка правильности конфигурации smb.conf;

nmblookup <имя компьютера> - проверить разрешение NetBIOS-имен службой nmbd;

nmblookup -M -- - - определить, какой компьютер сети является мастер-браузером.

Мастер-браузер – компьютер, обслуживающий базу данных NetBIOS-имен компьютеров;

smbclient -L <имя-компьютера> - получить список smb-ресурсов на сервере;

smbclient //<имя-компьютера>/<имя-ресурса> - подключение к SAMBA-ресурсу (из ОС Linux)

smbclient //IKT-201/pub - аналог предыдущей команды;

net view \\IKT-201 - подключение к SAMBA-ресурсу (из ОС Windows)

/sbin/useradd -M -d /home/samba/stud1 stud1 - регистрация SAMBA-пользователя;

smbpasswd -a stud1 - установка пароля;

smbclient //IKT-201/STUD -U stud1 - указание пользователя при доступе;

/etc/samba/smbpasswd - база данных учетных записей SAMBA.

Монтирование файловых ресурсов SAMBA

Команда **mount.cifs (smbmount)** – монтирование файловых ресурсов протокола SMB/CIFS (реализована как демон, контролирующий события, происходящие во время монтирования и использования файловых ресурсов).

smbmount //IKT-201/pub /mnt/winshare -o username=nobody,password=' '

Разберем синтаксис команды:

//IKT-201 - имя компьютера

pub - имя ресурса

/mnt/winshare - точка монтирования

Команда **smbstatus** выводит список текущих подключений SAMBA.

Лекция 3.4 Сетевые средства Linux.

Настройка сетевого интерфейса из командной строки. Настройка маршрутизатора по умолчанию. Использование системы доменных имен (Domain Name System (DNS)). Поиск и устранение проблем в работе сети. Утилиты netstat, nmap. Проверка работы DNS. Утилита мониторинга трафика (поток сообщений в сети передачи данных) IPTraf. arp-кэш. Сетевой экран, его конфигурирование с помощью утилиты iptables. Антивирусная защита.

Адресация IPv4 (стандарт RFC 791, 1981 год)

IP - адрес- адрес компьютера в сети. Состоит из 32 бит (версия 4 стандарта). Из записывают в виде четырех десятичных чисел от 0 до 255 (по 8 бит), разделенных точкой, A.B.C.D, где A,B,C,D ∈ [0,255]. Пример: 192.168.10.10.

В IP-адресе можно выделить две части, разделенные точкой: **[адрес сети].[адрес узла].** Длина адреса сети разнится для сетей разных классов. Сети класса А имеют первые 8 бит, начинающиеся с 0. Это соответствует А ∈ [0,127], всего 128 сетей (рис. 25). При этом, 24 бита - произвольны, и соответствуют адресу узла (хоста, компьютера). Каждая сеть, соответственно включает 2^{24} - 2 узлов в сети (16 777 214). Число «2» отнимается, поскольку каждая сеть всегда включает широковещательный адрес (последний в сети, 127.255.255.255) и нулевой адрес, который означает саму сеть по себе и используется для маршрутизации.

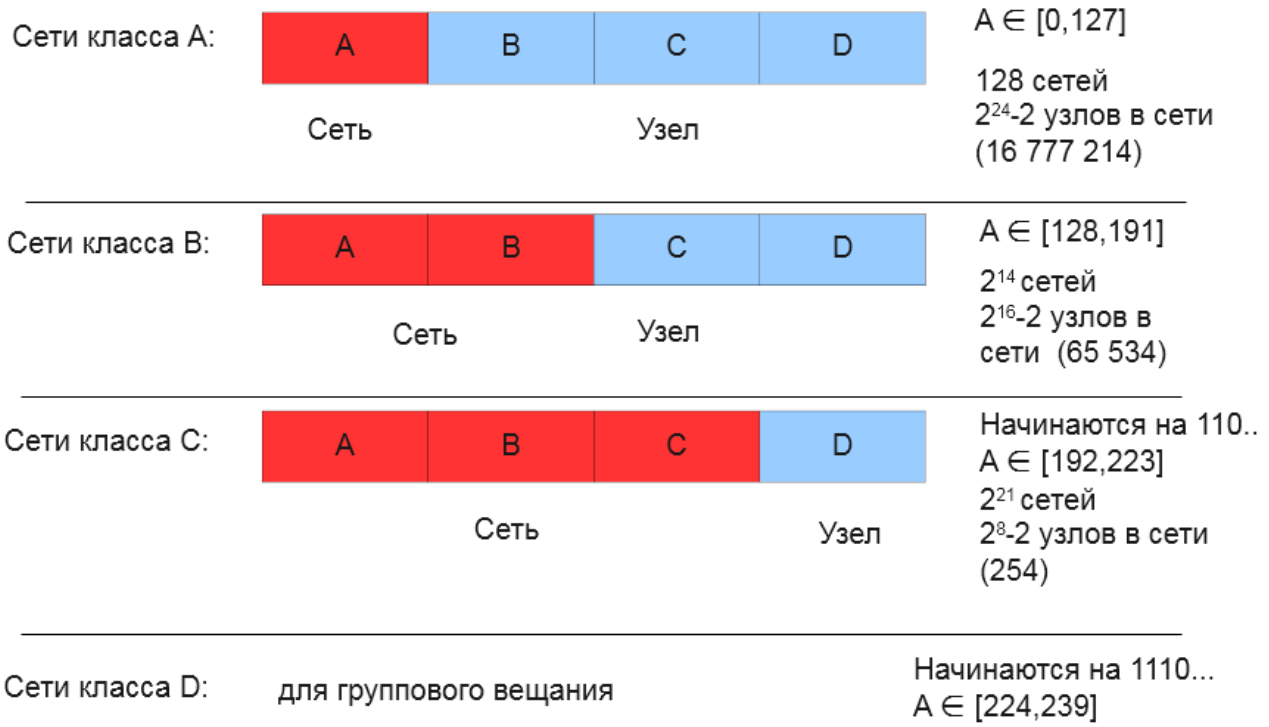

Рисунок 25. Схема адресации IPv4

Маска подсети определяет то, какая часть адреса является адресом сети. Там, где биты относятсяк адресу сети (а не узла), в маске содержатся единицы (табл. 9).

Таблица 9. Маска подсети

	В десятичном виде	**В двоичном виде**
IPадрес	192.168.111.**253**	1100 0000 1010 1000 0110 1111 **1111 1101**
Маска	255.255.255.**0**	1111 1111 1111 1111 1111 1111 **0000 0000**
Адрес сети	192.168.111.0	1100 0000 1010 1000 0110 1111 0000 0000

Маски для сетей различных классов:

Класс A: маска 255.0.0.0

Класс B: маска 255.255.0.0

Класс C: маска 255.255.255.0

Класс D: маска 255.255.255.255

Зарезервированные адреса

Адрес 127.0.0.1 сети класса A 127.0.0.0 — интерфейс «петля» Блоки адресов для локальных сетей:

Класс A: 10.0.0.0 - 10.255.255.255, одна сеть

Класс B: 172.16.0.0 - 172.31.255.255, шестнадцать сетей

Класс C: 192.168.0.0 -192.168.255.255, 256 сетей

Адресация IPv6 (RFC 2373)

Возможности IPv6:

- расширенное адресное пространство (128 бит);
- нет понятия класса сети;
- упрощенный формат заголовка IP-пакета;
- встроенная поддержка IPSec — создание виртуальных частных сетей с шифрованными каналами;
- поддержка IPMobile — для мобильных пользователей.

fe80:0000:0000:0000:0250:8bff:fe5f:7ceb - IP-адрес IPv6;

fe80::0250:8bff:fe5f:7ceb - сокращение нулей;

00:50:8B:5F:7C:EB, fe80::250:8bff:fe5f:7ceb - MAC-адрес и автоматически сконфигурированный для него IPv6 адрес.

Настройка сетевого интерфейса Ethernet

/sbin/ifconfig - посмотреть текущие настройки сети;

lsmod - убедиться, что загружен модуль ядра, /etc/modprobe.conf, связанный с сетью;

eth0, eth1, eth2... - имена сетевых интерфейсов;

ifconfig eth0 [ip-адрес] - просмотр базовой конфигурации устройства (интерфейса);

ifconfig -a - информация обо всех интерфейсах;

netstat -i - информация о приеме/передаче пакетов через все интерфейсы;

ping [ip-адрес] - проверка работоспособности сетевого интерфейса.

Настройка сетевого интерфейса в Ubuntu осуществляется через специальные проограммы, вызываемые, либо нажатием на значок в панели значков в верхнем правому углу экрана (рис. 26), либо в верхнем меню Параметры системы - Сеть (рис. 27).

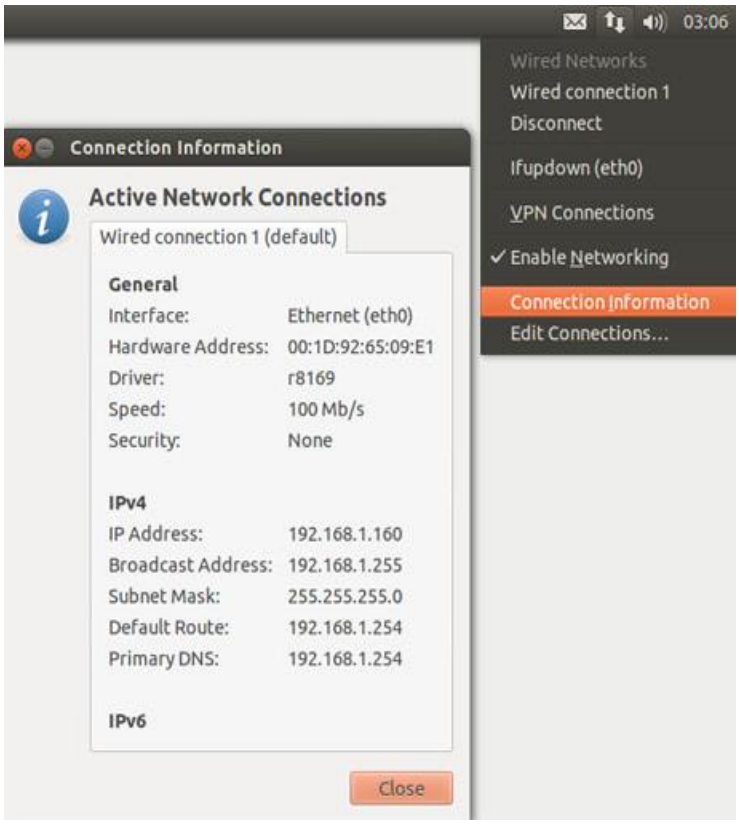

Рисунок 26. Информация о сетевых соединениях в Ubuntu

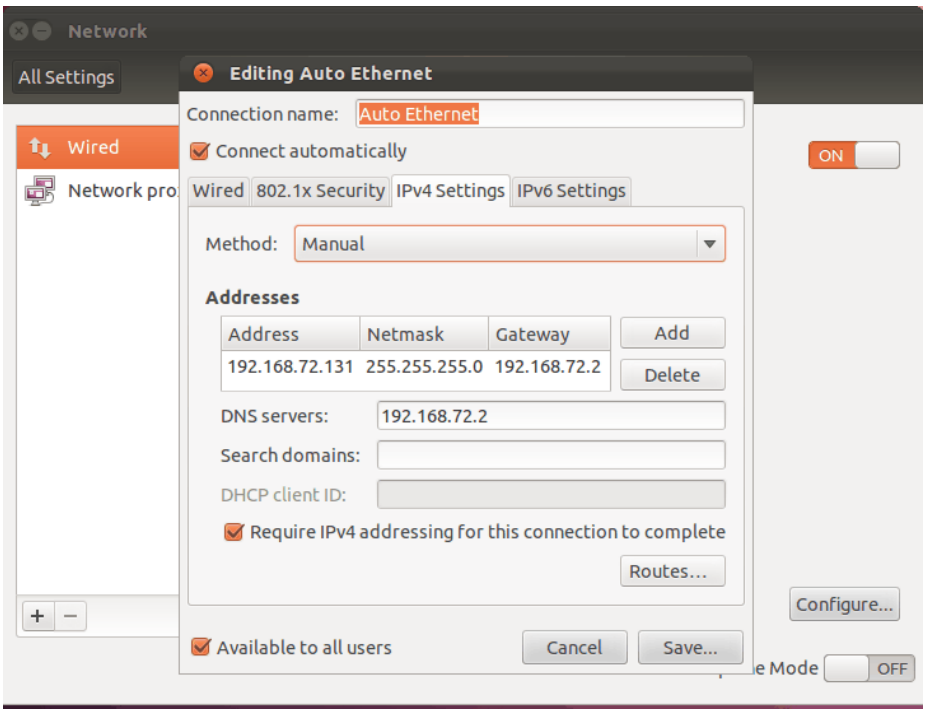

Рисунок 27. Управление сетями Network в Ubuntu

Настройка маршрутизатора по умолчанию

Маршрутизатор (*роутер*) — устройство, передающее IP-пакеты между различными сетями. Обычно при настройке подключения указывают маршрутизатор по умолчанию (default gateway).

В случае локальной сети или соединения точка-точка маршрутизацию настраивать не требуется. Для функционирования маршрутизации ядро должно поддерживать обмен пакетами между интерфейсами (forwarding).

netstat rn - получить текущую таблицу маршрутизации. Таблица маршрутизации содержит маршруты направления IP-пакетов в различные сети;

arp an - посмотреть кэш протокола ARP (преобразование IP-адресов в MAC-адреса). Для использования arp без указания пути директория /sbin должна быть в переменной $PATH;

route add default gw [ip-адрес] - назначить маршрутизатор по умолчанию

traceroute [ip-адрес] или [имя_домена] - утилита для построения маршрута между текущим компьютером и целевым (покажет все узлы, через которые будет проходить пакет). Позволяет отследить процесс прохождения пакетов через маршрутизаторы.

tracepath - аналог traceroute.

tracert - аналог команды traceroute, но для Windows.

Просмотр и смена сетевого имени

hostname - получить сетевое имя компьютера;

/etc/sysconfig/network - настройка имени компьютера в Red-Hat-подобных системах;

Настройка имени компьютера в Ubuntu (13.04 и выше)

1) Изменить имя хоста в файлах /etc/hosts и /etc/hostname

2) systemctl restart systemd-logind.service

Другой способ:

hostnamectl set-hostname new_host_name

Настройка разрешения имен

Разрешение имен - установление соответствия между парой «хост, домен» и IP-адресом хоста.

Способы разрешения имен:

- файл /etc/hosts — для небольших сетей
- использование службы DNS (Domain Name System)
- использование служб NIS, NIS+, LDAP — для корпоративных сетей

/etc/resolv.conf - настройка системы разрешения имен. В файле встречаются термины nameserver — сервер DNS, domain — имя домена, search — список доменов, которые нужно подставлять при поиске.

/etc/host.conf - тонкая настройка системы разрешения имен.

/etc/nsswitch.conf - информация о доменах и связанных с ними базах данных.

Поиск и устранение проблем с сетью

Основные причины проблем в работе сети:

- Неисправности в сетевых кабелях и сетевом оборудовании;
- Отсутствует драйвер для сетевой карты или установлен не тот драйвер;
- Неверно настроены IPадреса узлов сети;
- Засорен кэш ARP;
- Неверно указан маршрутизатор по умолчанию;
- Неверно настроена система разрешения имен;
- Избыточная блокировка фильтром IP пакетов (сетевым экраном).

/sbin/lspci - проверить работоспособность сетевой карты;

/sbin/modinfo - получение информации о модуле ядра;

/sbin/lsmod - список загруженных модулей;

/sbin/ifconfig, а также **netstat rn** - проверить, правильно ли настроен IP-адрес и маршрутизатор по умолчанию.

Алгоритм проверки работоспособности сети при помощи команды ping

1. Проверить работоспособность сетевой подсистемы в ядре:

ping 127.0.0.1

2. Проверить работоспособность сетевого адаптера:

ping [ip-адрес, присвоенный адаптеру]

3. Проверить работоспособность локальной сети:

ping [ip-адрес локального компьютера]

4. Проверить прохождение пакетов к внешнему сетевому адаптеру маршрутизатора по умолчанию;

ping [ip-адрес маршрутизатора]

5. Проверить прохождение пакетов по любому внешнему IP-адресу

ping [ip-адрес удаленного компьютера]

6. Проверить работу подсистемы разрешения имен

ping [имя домена]

Утилита netstat – информация о соединениях, и процессах, их открывших

netstat -tulpn | grep :80 - вывести информацию об открытых портах и отфильтровать только информацию о порте 80;

Nmap – утилита для исследования состояния компьютеров в сети (проверка открытых портов и т.п.). Применяется для проверки безопасности и корректности настройки сети.

nmap -sV -p 22,80,110,143,8080 192.168.0-10.1-255 - исследование состояния портов 22,80,110,143,8080 с попыткой определения открывшего порт приложения для ряда хостов от 192.168.0.1 до 192.168.10.255.

nmap -sP 192.168.0.0/16 -oG file.txt - пинг-сканирование всей локальной сети класса B для ряда хостов от 192.168.0.1 до 192.168.255.255 с выводом результатов в файл file.txt в удобном для программы grep формате.

Проверка кэша ARP

Ethernet работает с MAC-адресами, а не IP-адресами. Поэтому, для установки соединения компьютер выполняет широковещательный запрос (broadcast) по протоколу ARP. Целевой компьютер получает запрос и посылает ответный запрос, содержащий MAC-адрес. Кэш arp позволяет не перезапрашивать IP-адрес заново в течение некоторого времени (не превышающего, обычно, несколько часов)

/sbin/arp a - вывести содержимое кэша ARP (должно быть правильное сопоставление MAC-адресов и IP-адресов);

/sbin/arp d - удаление неправильных записей в кэше;

/sbin/arp s - установить запись вручную.

DNS (Domain Name System) — компьютерная система для получения информации о доменах. Основное употребление – перевод доменного имени в IP-адрес.

Для сайта disk.yandex.ru уровни доменов: III - disk, II - yandex, I -ru (доменная зона).

host - утилита для тестирования клиента;

dig - утилита для тестирования DNS-сервера;

nslookup - позволяет тестировать и сервер DNS, и клиента подсистему разрешения имен.

Утилита IPTraf - мониторинг сетевого трафика (рис. 28).

/var/log/iptraf - директория для хранения лог-файлов по умолчанию.

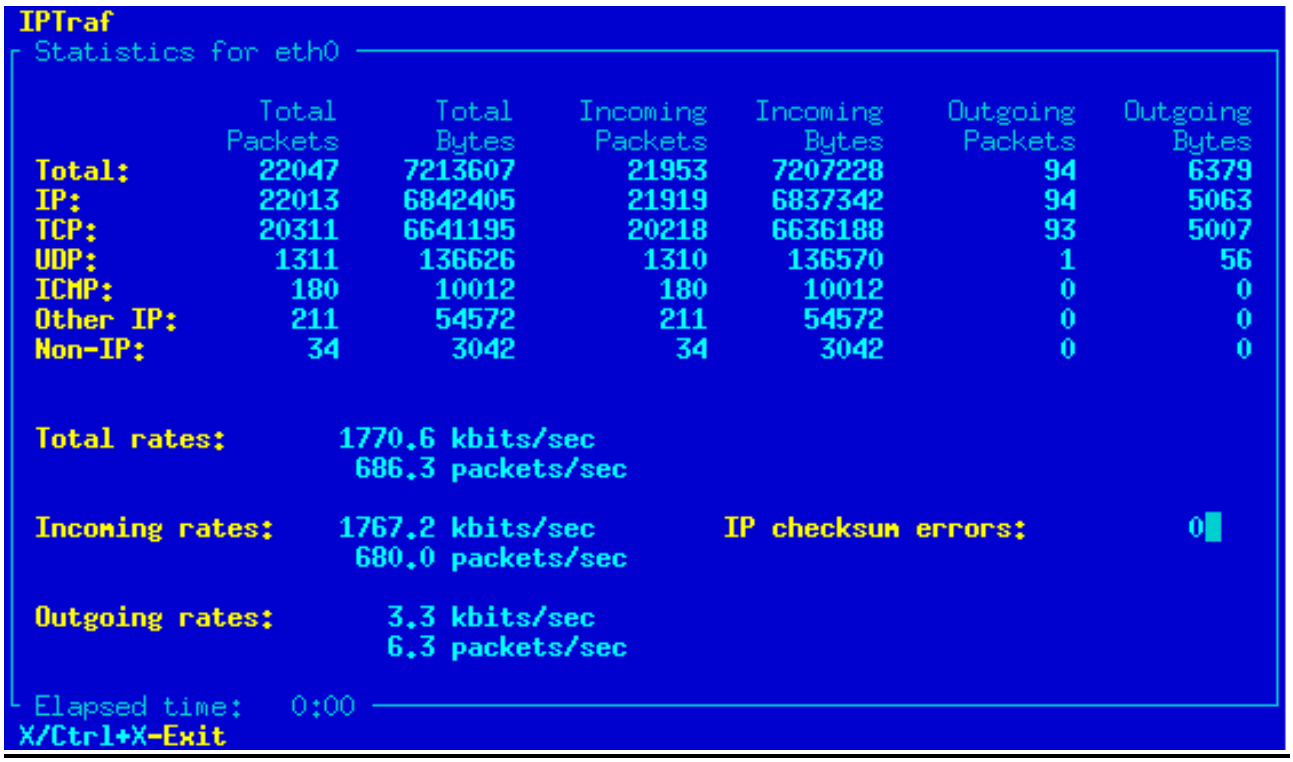

Рисунок 28. Окно утилиты IPTraf

Сетевой экран (*брандмауэр*, *firewall*) - программа, осуществляющая фильтрацию сетевого трафика на основе заданного набора правил. Популярный межсетевой экран, встроенный в ядро Linux с версии 2.4, - **netfilter**. Утилита для конфигурирования правил в netfilter называется **iptables**.

Основы конфигурирования с помощью iptables

Обмен трафиком между компьютерами происходит пакетами (информация разбивается на части опредленного размер, упаковывается и пересылается). Пакеты пропускаются через **цепочки, то есть, списки правил.**

Правила сопоставления пакетов

Каждый пакет проходит цепочку, начиная от первого правила. Как только найдено соответствие правилу, осуществляется переход (-j = --jump) к указанному действию (обычно - REJECT, ACCEPT, DROP).

Цепочки

Пять основных типов цепочек:

PREROUTING — предварительная обработка входящих пакетов.

INPUT — цепочка входящих пакетов с конкретным процессом-адресатом.

FORWARD — для перенаправляемых пакетов ().

OUTPUT — цепочка исходящих пакетов от локальных процессов.

POSTROUTING — пост-обработка исходящих пакетов.

Переадресуемые пакеты проходят три цепочки: PREROUTING - FORWARD - POSTROUTING, а не одну.

Стандартные действия, доступные во всех цепочках — ACCEPT (разрешить прохождение пакета), DROP (отклонить), QUEUE (передать на анализ внешней программе), и RETURN (вернуть в предыдущую цепочку).

Цепочки организованы в таблицы.

filter — основная таблица, используется по умолчанию если название таблицы не указано. Содержит цепочки INPUT, FORWARD, и OUTPUT.

iptables-save > /etc/network/iptables.rules - сохранить изменения iptables в файл.

В файл /etc/network/interfaces следует для интерфейса (устройства), к которому применяется данная политика фильтрации пакетов, указать строку

pre-up iptables-restore < /etc/network/iptables.rules

В версии Ubuntu 16.04 появился специальный пакет, упрощающий сохранение конфигурации iptables:

sudo apt install iptables-persistent

Сохранение и перезагрузка (применение) конфигурации iptables может быть прозведена командами

sudo /etc/init.d/iptables-persistent save

sudo /etc/init.d/iptables-persistent reload

Примеры использования iptables

iptables -L -n -v --line-numbers - просмотреть таблицы;

iptables -I INPUT 3 -s 205.205.205.205 -j DROP - вставить в цепочку INPUT правило под номером 3, блокирующее (-j DROP) прием пакетов от источника (-s = --source) 205.205.205.205.

iptables -A INPUT --source 127.0.0.1 --jump ACCEPT

iptables -A INPUT --jump FURTHER_ANALYSIS

Эти строки добавляют в конец цепочки INPUT два правила первое правило «пропустить все пакеты от 127.0.0.1», а второе правило — отправить (оставшиеся пакеты) на анализ в цепочку FURTHER_ANALYSIS.

iptables -A INPUT -i lo -j ACCEPT

iptables -A OUTPUT -o lo -j ACCEPT - разрешить трафик для интерфейса lo (внутренний трафик интерфейса локальная петля, lo - стандартное название интерфейса)

iptables -A INPUT --i eth0 -p tcp --dport 9000 -j LOG --log-prefix "Port 9000 incoming blocked! "

iptables -A INPUT -i eth0 -p tcp --dport 9000 -j DROP - правило в предыдущей строке позволяет отправить информацию о пакетах, пришедших на порт 9000 через устройство eth0 в /var/log/messages, а последнее правило блокирует прием этих пакетов;

host -t a имя_домена - найти ip-адрес для домена;

whois ip-адрес | grep CIDR - найти диапазон ip-адресов, используемых доменом и поддоменами.

На основе информации о диапазоне адресов, можно заблокировать выход на определенный домен и его поддомены

iptables -A OUTPUT -o eth0 -d 205.205.205.0/24 -j DROP - запретить исходящие соединения в заданную подсеть;

iptables -A INPUT -p icmp --icmp-type echo-request -j DROP - замаскировать компьютер, скрыв его от ping-запросов;

iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT - разрешить входящих трафик, если соединение с удаленным копьютером было инициализировано первично локальным компьютером.

iptables -A INPUT -p tcp -s 205.205.205.0/24 --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT - разрешить соединения по порту 22 (стандартно, используется службой ssh) только с компьютеров подсети 205.205.205.0/24 (в предположении, что политика по умолчанию для входящих пакетов - DROP);

Механизм NAT (Network Address Translation) - IP-адрес локального компьютера подменяется сетевым экраном или маршрутизатором подменяется на имеющийся внешний IP-адрес (например, маршрутизатора) для обмена пакетами с удаленным компьютером, для каждого соединения используется отдельный порт. Пакеты, поступившие от удаленного пакета по этому порту, передаресуются обратно инициировавшему сеанс связи локальному компьютеру.

Существуют **антивирусные программы для Linux**. Популярный антивирус - **ClamAV**. Управляется из командной строки.

Установка:

sudo apt install clamav clamav-daemon или **sudo yum install clamav**

Ообновление базы данных вирусов

sudo freshclam

Сканирование без очистки с оповещением при нахождении вирусов

sudo clamscan -r /home/user --bell

Сканирование с автоматическим удалением инфицированных файлов

sudo clamscan --infected --remove -r /home/user