**105.1 Настройка и использование командной оболочки**

Студент должен уметь настроить под себя оболочку, а также изменять глобальные и пользовательские профили.

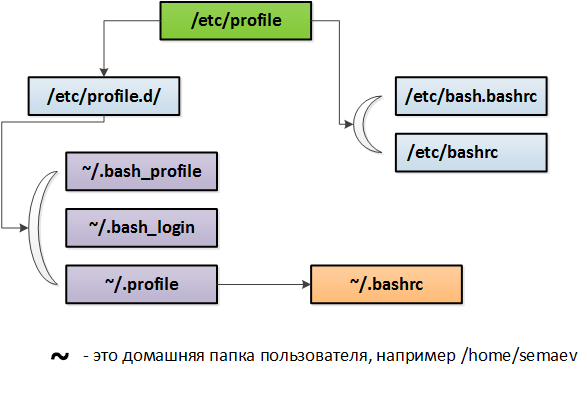
**Изучаем**:

* Установку переменных во время входа в систему, или при запуске оболочки;
* Создание bash-сценариев для часто используемых цепочек команд;
* Управление структурой каталогов для новых пользователей;
* Корректные пути поиска команд.

**Термины и утилиты:**

* .
* source
* /etc/bash.bashrc
* /etc/profile
* env
* export
* set
* unset
* ~/.bash\_profile
* ~/.bash\_login
* ~/.profile
* ~/.bashrc
* ~/.bash\_logout
* function
* alias
* lists

Вся работа пользователя с серверами и тонкая настройка клиентов Linux осуществляется в командной оболочке. Командная оболочка представляет собой настраиваемое пространство для работы пользователя с системой. Существует несколько популярных оболочек, но мы будем рассматривать используемую по умолчанию в Ubuntu оболочку bash.



В ***/etc/profile*** находится глобальный профиль, который загружает все из директории ***/etc/profile.d/*** , настройки глобального пользователя и необходимые файлы из домашней папки текущего пользователя ***/home/username*** (или ***~***).

Содержимое профилей это, как правило, набор переменных и некоторые индивидуальные настройки пользователей. В домашней папке пользователя в разных дистрибутивах находятся разные файлы (см. рисунок), в частности в ubuntu за настройки профиля пользователя отвечает файл ***~/.bashrc***

То есть когда осуществляется вход в систему **bash** читает и вызывает команды из файла ***/etc/profile***, если этот файл существует. После чтения этого файла, он смотрит следующие файлы в следующем порядке: ***~/.bash\_profile***, ***~/.bash\_login*** и ***~/.profile (~/.bashrc)***, читает и вызывает команды из первого, который существует и доступен для чтения.

При выходе bash читает и выполняет команды из файла ***~/.bash\_logout***

Если оболочка запускает не для входа в систему, то bash читает и исполняет команды из файлов ***/etc/bash.bashrc*** и ***~/.bashrc***, если они существуют.

Директория ***/etc/skel/***представляет собой каталог, содержимое которого будет скопировано в профиль каждого вновь создаваемого пользователя.

\_\_\_\_

При работе с bash используются псевдонимы и функции:

**Псевдоним (alias) –** текст для вызова команды с ключами, например*:*

***la = ‘ls -A’*** *(команда la будет вызывать команду ls с ключом ”–A”)*

**Функция** - текст для вызова скрипта из нескольких команд, например*:*

***function Hello() {*** *(имя функции – “Hello”)*

***echo “Hello, dear, I am awake for:”;*** *(вывести текст в кавычках)*

***uptime –p; }*** *(вывести результат команды “uptime -p”)*

Псевдонимы и функции можно просматривать и редактировать в файле ***~/.bashrc***

\_\_\_\_

Функции можно передавать аргументы, для этого используются символы $1 (первый аргумент) и т.д., например:

***Function showlog () {*** *(имя функции – “showlog”)*

***Date > $1.txt;*** *(отправить результат команды date в файл по имени первого аргумента)*

***Grep $1 /var/log/auth.log | tail -n $2 >> $1.txt;*** *(отсортировать по первому аргументу содержимое файла /var/log/auth.log и дописать последние строки по второму аргументу в файл созданный на предыдущем этапе)*

***}*** *(конец функции)*

То есть команда ***showlog semaev 3*** создаст в текущем каталоге файл *semaev.txt* и запишет в него последние *три* строчки из файла /var/log/auth.log, содержащие текст «*semaev*».

\_\_\_\_

В Linux есть два вида переменных:

* **Переменные –** работают в пределах текущей оболочки;
* **Переменные среды** – работают так же во всех дочерних процессах;

Команды:

**set** – вывод всех переменных и функций;

**unset** – удаление переменных;

**export** – превращает переменную в переменную среды;

**env** – выводит переменные среды;

**PATH** – переменная содержащая путь к исполняемым файлам.

Дополнение:

В bash есть команда **source** (ее также можно заменить обычной точкой), которая может запускать скрипты в рамках текущий консоли.

То есть ***./script.sh*** запускает новый процесс командного интерпретатора, указанного в первой строчке скрипта, при этом процесс-родитель останавливает свою работу, на момент выполнения скрипта.

А команда ***source ./script.sh*** (или ***. ./script.sh***) запустит процесс при помощи текущего интерпретатора (в текущей оболочке).

Существует переменная ***list***, которой можно присваивать значение массива данных (что может пригодиться в скриптах), например:

***list=(apple mango banana)*** *(задать значение переменной в виде последовательности текста)*

***echo ${list[1]}*** *(вывести значение первой ячейки массива в переменной list: будет выведено слово mango, так как нумерация начинается с ноля)*