

Кафедра ЮНЕСКО по новым информационным технологиям

# Операционная система Linux.

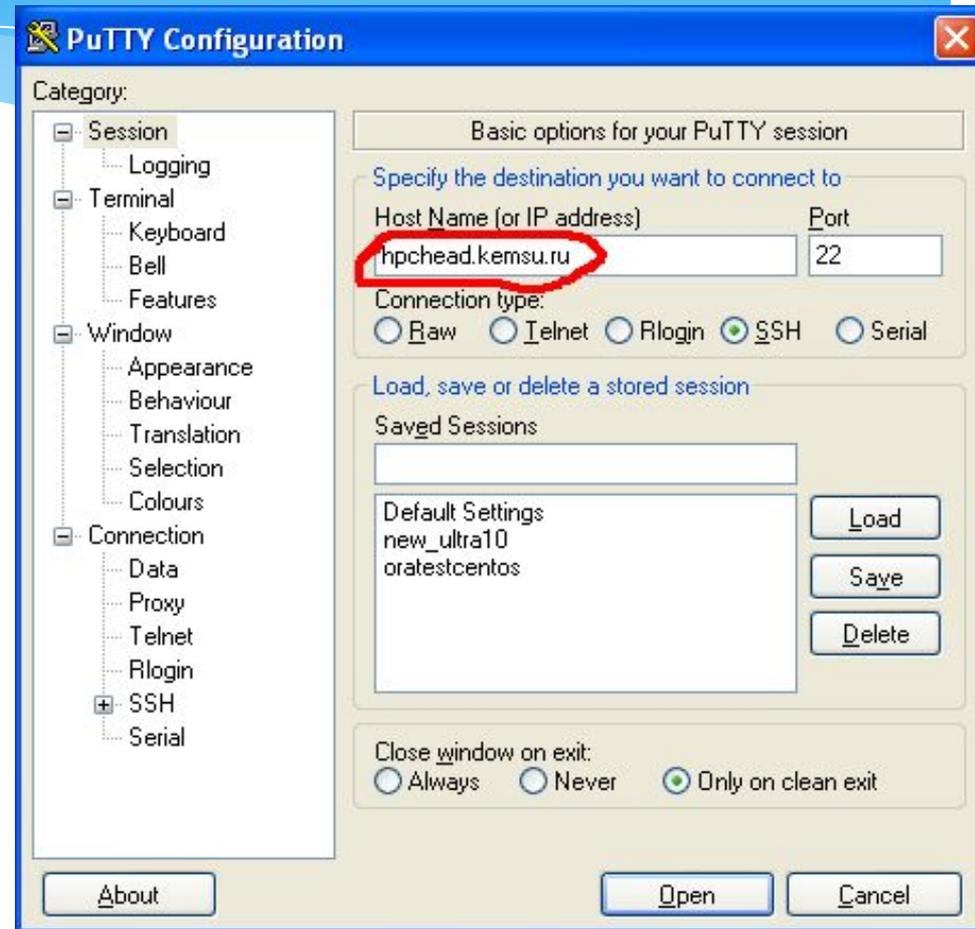
Лабораторная работа №1:  
Знакомство с операционной  
системой

# Авторизация в ОС

- 1) Запустить putty
- 2) В поле «Host Name»  
вписать  
***hpchead.kemsu.ru***
- 3) Кнопка «Open»

Login: stud

Password: stud1234

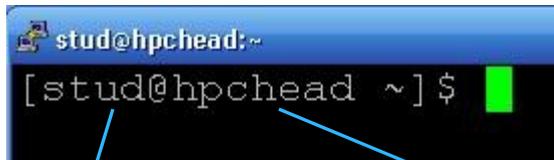


# Пользователи ОС

2 типа пользователей:

\* Обычные пользователи

\* root – администратор, суперпользователь



```
stud@hpchead:~  
[stud@hpchead ~]$
```

Имя пользователя

Имя компьютера (hostname)

# Общий вид команд в Linux

`$ имя_команды [опции]... [параметры]...`

Примеры:

`$ ls -la /home` // «-la» - опции, «/home» - параметр

`$ iptables -L` //отсутствуют параметры

`$ cat file.txt` //отсутствуют опции

# Подготовка к работе

- \* `$ mkdir M15...` // создание папки для группы
- \* `$ cd M15...` // вход в папку
- \* `$ mkdir Ivanov` // создание собственной папки (вместо «*Ivanov*» - Ваша фамилия)
- \* `$ cd Ivanov` // вход в собственную папку
- \* `$ touch Ivanov.txt` // создание файла (вместо «*Ivanov*» - Ваша фамилия)

# Получение справки

Команды **man** и **info**:

**\$ whatis *command\_name*** // краткая справка по команде

**\$ man *command\_name*** // подробная справка по команде

**\$ info *command\_name*** // подробная справка по команде

Для выхода из просмотра справки нажать «q»

Примеры:

**\$ whatis dir**

**\$ man ls**

**\$ man -k compress** // поиск в файлах справки whatis слова «compress»

# Несколько консолей (виртуальных терминалов)

<Ctrl>+<Alt>+<Fn>, где  $n=1, \dots, 6$  – переключение на консоль № $n$  (не получится при работе в *putty*)

Каждая консоль – отдельный рабочий стол со своими окнами или отдельная командная строка

# История команд

- \* Клавиша <Up> - вызов предыдущей команды
- \* **\$ history** – вывод истории команд
- \* **\$ !73** – вывод команды №73. Для того, чтобы ее выполнить просто нажмите клавишу <Enter>.

# Содержимое каталогов, информация о файлах

- \* `$ ls` //содержимое текущего каталога
- \* `$ ls /home/stud/M13...` //содержимое каталога /home/stud/M13...
- \* `$ ls -l` //подробная информация о файлах и подкаталогах
- \* `$ ls -a` //вывод информации о скрытых файлах и подкаталогах
- \* `$ ls -l -a`  $\Leftrightarrow$  `$ ls -la` // комбинация опций

# Вывод имени текущего каталога

- \* `$ pwd` // сейчас вы находитесь в том каталоге, который увидели в выводе данной команды. Выводится полный путь от корня (/).

# Смена текущего каталога

- \* `$ cd` // смена текущего каталога
- \* `$ cd ..` // перейти на один уровень вверх
- \* `$ cd Ivanov` // перейти в подкаталог «*Ivanov*» текущего каталога
- \* `$ cd /tmp` // перейти в каталог `/tmp`
- \* `$ cd /home/stud/M13.../Ivanov`
- \* `$ cd ../../B1/B2`

# Скрытые файлы и директории

\* `$ls -a ~` // вывод содержимого домашней директории.

Файлы, начинающиеся на «.» - скрытые (.bash\_profile, .bashrc, .bash\_history и др.)

Обычно это либо файлы настроек, либо файлы, в которые производит запись сама операционная система.

# Типы файлов

В Linux **файл** - просто **поток байтов**, поэтому

Типы файлов:

- \* обычные файлы;
- \* каталоги;
- \* файлы физических устройств (жесткие и съемные диски, терминал, принтер и т. д.)
- \* именованные каналы (named pipes);
- \* сокеты или «гнезда» (sockets);
- \* символические ссылки (symlinks).

# Физические устройства

Соответствующие файлы расположены в каталоге **/dev**

Типы устройств:

- \* Символьные (байт-ориентированные). Пример: терминалы.
- \* Блочные (блок-ориентированные). Пример: жесткие диски.

# Каналы и сокеты

- \* Логические абстракции, предназначенные для передачи информации между различными программами (процессами), работающими как на одном компьютере, так и на разных.
- \* **Именованные каналы** – используются при взаимодействии процессов, располагающихся на одном компьютере или на разных.
- \* **Неименованные (анонимные) каналы** – только в пределах одной операционной системы.
- \* **Сокеты** предназначены в основном для передачи данных по сети между разными компьютерами.

# Ссылки

Типы ссылок:

- \* **Жесткая ссылка** – другое имя того же файла.
- \* **Символическая ссылка** (аналог ярлыка в Windows).

Редактировать файл можно, обратившись к нему по оригинальному имени, жесткой или символической ссылке.

`$ ln имя_файла_или_каталога имя_ссылки //создание жесткой ссылки`

`$ ln -s имя_файла_или_каталога имя_ссылки // создание символической ссылки`

# Обозначения типов файлов

\* `$ ls -la ~`

\* `$ ls -la /dev`

Первый символ в каждой строке:

\* `-` = обычный файл (текстовый файл, программа, ... );

\* `d` = каталог - directory;

\* `b` = файл блочного устройства;

\* `c` = файл символьного устройства;

\* `s` = сокет (гнездо) - socket;

\* `p` = именованный канал - pipe;

\* `l` = символическая ссылка - link.

# Удаление файлов

- \* `$ rm [-f] [-i] имя_файла ... [имя_файла ]` //удаление файла(-ов)
- \* `$ rm -r [-f] [-i] имя_каталога ... [имя_файла ...]` //удаление каталога(-ов) и файла(-ов)
- \* `«-f» ⇔ «--force»` //удаление без вопросов и уведомлений о несуществующих файлах
- \* `«-i» ⇔ «--interactive»` //выводить запрос перед удалением каждого файла
- \* `«-r» ⇔ «--recursive»` //удаление каталога и всех вложенных подкаталогов

Никакой мусорной корзины нет!!! Удаляете навсегда!!!

# Удаление пустых каталогов

- \* `$ rmdir [-p] каталог` // удаление пустого каталога
- \* «-p» ⇔ «--parents» // удаление каталога и его пустых надкаталогов

При помощи **rmdir** удаляются только пустые каталоги! Для удаления непустых используется «rm -r».

# Работа со ссылками

- Задание: 1) создать жесткую и символическую ссылки на Ваш файл *Ivanov.txt*;
- 2) удалить исходный файл;
  - 3) `ls -la`;
  - 4) удалить жесткую ссылку;
  - 5) `ls -la`

# Права доступа к файлам

\* `$ ls -la ~/test.out`

\* `-rwxrwxr-x 1`

`stud`  
Пользователь-владелец

`stud`  
Группа-владелец

## 3 тройки бит:

`rwx`

Права

пользователя-владельца

`rwx`

Права

группы-владельца

`r-x`

Права

остальных пользователей

**r** – право на чтение

**w** – право на запись

**x** – право на выполнение

# Цифровое представление прав

- \* Каждая из 3-х троек бит – число в двоичной системе счисления.
- \* Если право есть, то соответствующий разряд – 1, права нет - 0.

**r – x**

$$1\ 0\ 1 = 1*2^0 + 0*2^1 + 1*2^2 = 1+0+4=5$$

**r w x**

$$1\ 1\ 1 = 1*2^0 + 1*2^1 + 1*2^2 = 1+2+4 = 7$$

Итого:

**rwX rwX r-x ↔**

**775**

# Изменение прав доступа

## \* 1-ый вариант:

```
$ chmod [-v] [-f] [-R] MODE файл_или_директория
```

**MODE** – тройка цифр

«**-v**» ⇔ «**--verbose**» // «болтливый» режим

«**-f**» ⇔ «**--silent**» // «тихий» режим – без уведомлений

«**-R**» ⇔ «**--recursive**» // смена разрешений для всех файлов и поддиректорий данного каталога

Пример:

```
$ chmod 640 Ivanov.txt
```

# Изменение прав доступа

\* 2-ой вариант:

**\$ chmod *wXr* имя\_файла**

где вместо символа *w* подставляется:

- \* либо символ «**u**» (т.е. пользователь, являющийся владельцем);
- \* либо «**g**» (группа);
- \* либо «**o**» (все пользователи, не входящие в группу-владелец);
- \* либо «**a**» (все пользователи системы - и владелец, и группа, и все прочие).

Вместо *X* ставится:

- \* либо «**+**» (предоставляем право);
- \* либо «**-**» (лишаем соответствующего права);
- \* либо «**=**» (установить указанные права вместо имеющихся),

Вместо *r* — символ, обозначающий соответствующее право: **r**, **w** или **x**.

Пример: **\$ chmod g+x file.sh**

# Смена владельца

`$ chown [-v] [-f] [-R] [OWNER][:GROUP] файл(каталог)`  
*//смена пользователя-владельца и группы-владельца*

`$ chgrp [-v] [-f] [-R] [GROUP] файл(каталог)`

«-v» ⇔ «--verbose» // «болтливый» режим

«-f» ⇔ «--silent» // «тихий» режим – без уведомлений

«-R» ⇔ «--recursive» // смена разрешений для всех файлов и поддиректорий данного каталога

## Примеры:

`$ chown root:staff /u`

`$ chgrp -R staff /A/B`

# Копирование файлов

**\$ cp [-i] [-f] [-u] [-R] SOURCE DEST**

**\$ cp [-i] [-f] [-u] [-R] SOURCE DIRECTORY**

**SOURCE** - файл (каталог), который копируем

**DEST** – имя файла, куда копируем (целевой файл)

**DIRECTORY** – каталог, куда копируем

«-i» ↔ «--interactive» //запрашивать перед перезаписью

«-f» ↔ «--force» //перезаписывание без уведомлений

«-R» ↔ «--recursive» //рекурсивно копирует каталоги

«-u» ↔ «--update» //копирует, когда целевой файл  
старше

Пример: \$ cp abc.txt /tmp/def\_xxx /home/stud/some\_dir

# Перемещение файлов

**\$ mv [-i] [-f] [-u] SOURCE DEST**

**\$ mv [-i] [-f] [-u] SOURCE DIRECTORY**

**SOURCE** - файл (каталог), который перемещаем

**DEST** – имя файла, в который перемещаем (целевой файл)

**DIRECTORY** – каталог, куда перемещаем

«-i» ⇔ «--interactive» //запрашивать перед перезаписью

«-f» ⇔ «--force» //перезаписывание без уведомлений

«-u» ⇔ «--update» //перемещает, когда целевой файл старше

# Создание каталогов

**\$ mkdir [-p] [-m MODE] *каталог***

**«-p»** ⇔ «--parents» //создание каталога и всех надкаталогов

**«-m»** ⇔ «--mode» //режим доступа (задается как в **chmod**)

Примеры:

```
$ mkdir -p /A/B/C
```

```
$ mkdir -m 770 newdir
```

# Поиск файлов и каталогов

**\$ find [список\_каталогов] критерий\_поиска**

[список\_каталогов] – при отсутствии – текущий каталог.

Критерии:

-name – по имени файла

-path – в полном пути

-group – файлы, принадлежащие группе

Примеры:

```
$ find /usr/share/doc /usr/doc /usr/locale/doc -name instr.txt
```

```
$ find . -path './sr*sc' – найдёт, например, './src/misc'
```

# Просмотр файлов

**\$ cat [-n] *имя\_файла* //вывод всего файла сразу**  
**«-n» ⇔ «--number» //вывод номеров строк**

**\$ more [OPTIONS] *имя\_файла* //вывод постранично**  
**“q” - выход из режима просмотра**

**\$ less *имя\_файла* //более гибкие возможности вывода**  
***/pattern* – поиск в тексте по шаблону**  
**n – повтор поиска**

# Утилита sed

**sed – Stream EDitor**

**\$ sed [-n] [ адрес [ , адрес ] ] команда [ аргументы ]**

«-n» - подавление вывода

Адреса это либо номера строк, либо специальные символы, либо регулярное выражение.

**\$** — последняя строка

**начало~N** — Каждая N-я строка, начиная с номера начало

**/регулярное\_выражение/** — строки, попадающие под регулярное\_выражение

**Примеры:**

**1~2** — Каждая вторая строка

**/REGEXP/** — все строки, в которых встречается **/REGEXP/**

**10,20** — строки с 10-й по 20-ю

# Утилита sed

- \* Основные команды:

- \* **[адрес] а текст** — добавить новую строку с текстом после указанной строки

Пример:

- \* `$ cat sed_test`

```
sed_test_1 11111
```

```
sed_test_2 22222
```

```
sed_test_3 33333
```

- \* `$ sed '2 a new_line' sed_test`

```
sed_test_1 11111
```

```
sed_test_2 22222
```

```
new_line
```

```
sed_test_3 33333
```

# Утилита sed

- \* Основные команды:

- \* `[адрес [, адрес]] с текст` — удаляет выбранные строки и заменяет их на `текст`

Пример:

- \* `$ cat sed_test`

```
sed_test_1 11111
```

```
sed_test_2 22222
```

```
sed_test_3 33333
```

- \* `$ sed '2 c new_line' sed_test`

```
sed_test_1 11111
```

```
new_line
```

```
sed_test_3 33333
```

# Утилита sed

## \* Основные команды:

\* [адрес [, адрес]] s/регулярное\_выражение/замена/флаги — заменяет регулярное\_выражение на замена с учётом флагов:

- \* g — во всей строке
- \* i — без учёта регистра
- \* p — выводить результат замены

## Пример1:

```
* $ sed -ne 's/t/T/g' sed_test
```

```
sed_TesT_1 11111
```

```
sed_TesT_2 22222
```

```
sed_TesT_3 33333
```

## Пример2:

```
* $ sed 's/Nick|nick/John/g' report.txt > report_new.txt //замена
```

```
Nick и nickname John
```

# Утилита sed

**\$ sed 's/word\_to\_change/changing\_word/g' file //замена слова «word\_to\_change» на слово «changing\_word» в file**

Примеры:

**\$ sed 's/Nick/John/g' report.txt > report\_new.txt**

**\$ sed 's/Nick|nick/John/g' report.txt > report\_new.txt //замена Nick или nick на John**