

Lecture Agenda and Notes Draft:**Unix on Local Machine**

| | |
|---|---|
| Course Overview [20 min] | 2 |
| What is Unix [20 min] | 2 |
| Unix Shells & Command-Line Interface (CLI) [20 min] | 3 |
| Unix Filesystem [20 min] | 4 |
| Piping 1 [20 min] | 6 |
| Reading | 7 |

Problemsheet:**Unix on Local Machine****8**

Lecture Agenda and Notes Draft:

Unix on Local Machine

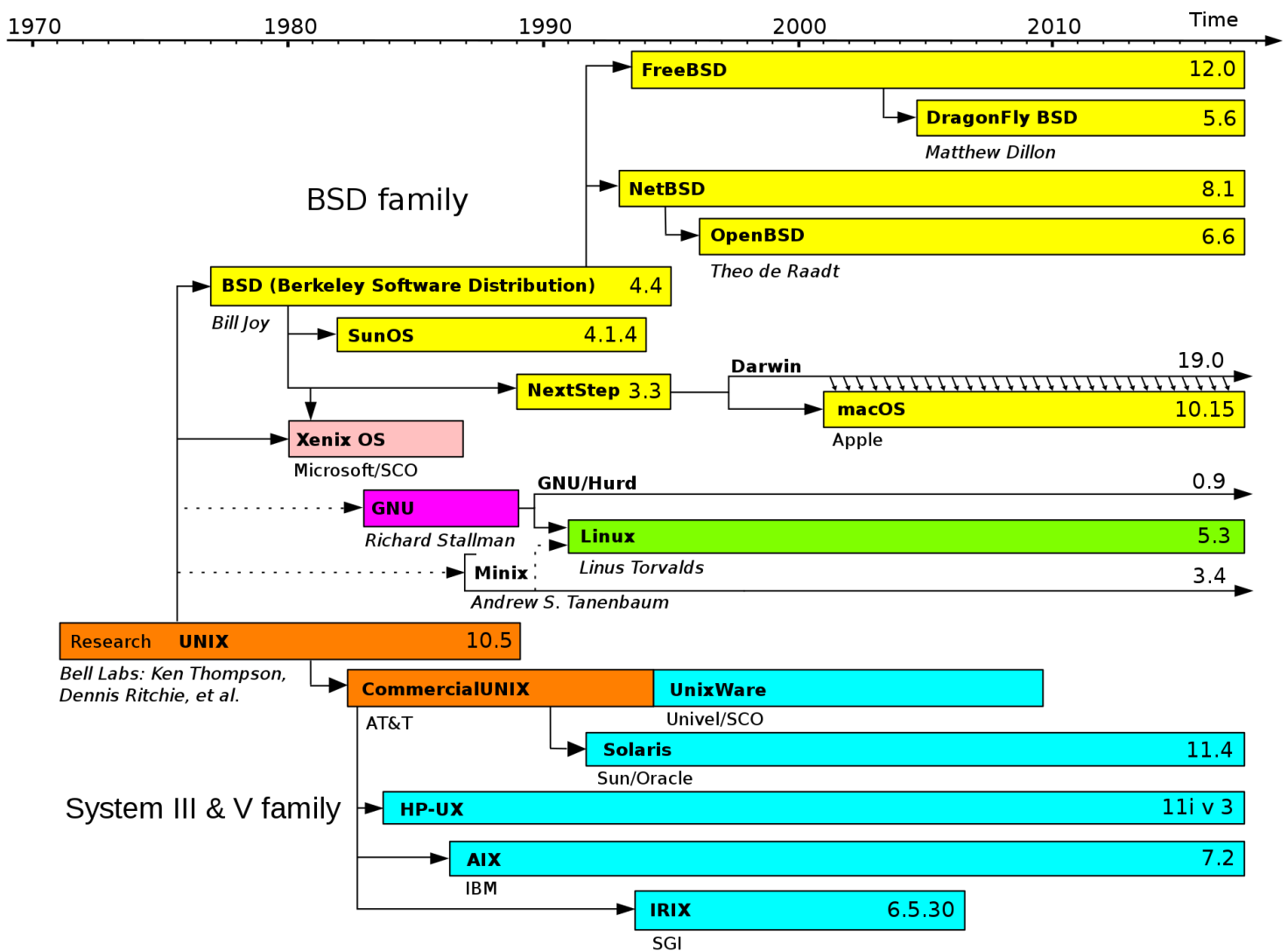
The following represents a very brief overview of the topics covered during the demonstration (lecture) part of the class. This material is publicly released to the students.

Course Overview [20 min]

This material is in the external slides.

What is Unix [20 min]

1. [What is Linux? - Linux Explained](#) is a great introduction and overview of the Linux, which is among the most popular Unix distributions.
2. [Difference between UNIX and Windows Operating System](#)
3. [Unix vs. Windows](#)
4. Below is a figure illustrating the evolution of Unix systems from the [Darwin wiki page](#).



5. A [Linux distribution](#) contains a particular version of the Linux kernel and a set of open-source and proprietary applications. There are quite a few [Linux distros](#) out there.
 - a. Open-source software (OSS) is computer software that is released under a license in which the copyright holder grants users the rights to use, study, change, and distribute the software and its

source code to anyone and for any purpose.[1][2] Open-source software may be developed in a collaborative public manner. [Read more](#)

- b. Proprietary software, also known as non-free software or closed-source software, is computer software for which the software's publisher or another person reserves some rights from licenses to use, modify, share modifications, or share the software. [Read more](#)
- c. [Comparison of open-source and closed-source software - Wikipedia](#)
6. Popular Linux distro families include [Debian](#) (e.g., Ubuntu), [Fedora](#) (RHEL, CentOS, Fedora), [openSUSE](#), etc.
 - a. Differences between Debian and RedHat include using different package managers (dpkg, yum).
 - b. Various licensing options are there (e.g. RHEL is paid, while Fedora Core is not)
 - c. Updates are released on a different time schedule, which leads to varying security .
7. Each distro has its own set of pre-installed applications and [package managers](#).
 - a. Examples for the Debian (Ubuntu, Mint): dpkg, apt . For Redhat (CentOS, Fedora) - это rpm, yum, dnf . The [comparison of commands](#) can be useful for your system.

```
apt-cache search package_name.
```

```
apt-get install
```

```
apt-get remove
```
8. Getting info on your OS :

```
uname -s — OS name
uname -r — kernel version
cat /etc/os-release — Distribution name
```

Unix Shells & Command-Line Interface (CLI) [20 min]

1. A Unix shell is a command-line interpreter or shell that provides a command line user interface for Unix-like operating systems. The shell is both an interactive command language and a scripting language, and is used by the operating system to control the execution of the system using shell scripts. [Read more](#)
2. Unix shells provide filename wildcarding (this class), piping, here documents, command substitution, variables and control structures for condition-testing and iteration (next class).
3. The most popular interpreters (shells) are: sh, bash, csh, zsh.

```
username@hostname~$
~ = current_directory
$ = shell_type (for common user $, root #)
```

4. Structure of the typical command:

```
command [option(s)/parameter(s)...] [argument(s)...]
ls --all --format=long
ls -al
symbols: $, ?
```

These commands above are equivalent.

5. To get help on the command, use the man program present in most Unix distros out of the box, e.g.

```
man ls
```
6. Two types of commands exist:
 - internal/built-in - cd, set, export, ls
 - external - stored in a binary file located in a certain list of default locations, e.g. /bin, /usr/bin, /usr/local/bin.

To invoke a command, the shell uses an environment variable named PATH, to search for binary files referenced by the external commands.

To check if the command is a built-in, internal, or an alias, use `which`:

```
which cd
which bash
```

Unix Filesystem [20 min]

1. Folders and file listing:

`.dotfiles` - hidden and shell session customization;
`cd`, `pwd`, `ls`, `tree` (should be installed).

You can print a long list of the contents of the current directory using `ls -l`

```
$ ls -l
-rw-r--r-- 1 user staff      3606 Jan 13  2017 report2018.txt
```

`-rw-r--r--` - This shows access permissions to a file or directory (first `d` or `-`), for owner, (`-rw-`), owner's group (`r--`) and the user (`r--`).

`1` - number of links to this file (soft and hard links exist, we'll get to that later today).
`user` - owner name;
`staff` - owner group;
`3606` - filesize
`Jan 13 2017` - date of last change;
`report2018.txt` - filename.

2. Creating and removing files and directories:

```
touch,
mkdir,
rm,
rm -r,
wc,
find --delete - differs from rm -r. What is better for removing large number of files?
Что лучше использовать при удалении большого кол-ва файлов? (ответ: find --delete - будет
быстрее, он батчами удаляет)
brace extention: touch text1 text2 text5
ls text{1..2}
```

3. The inode (index node) is a data structure in a Unix-style file system that describes a file-system object such as a file or a directory. Each inode stores the attributes and disk block locations of the object's data.

[Read more](#)

Each file is assigned an inode ID. To view the inode information, use the `stat` command:

Индексный узел `2678726` каждый inode ссылается на 1 файл раздела дискового пространства, и имеет информацию о файле. Inodes для директорий ссылается на структуру директорий.

```
2678726 -rw-r--r-- 1 user staff      3606 Jan 13  2017 report2018.txt
```

`ln test.sh test_link.sh` - Creates a [Hard link](#). A hard link lets you have multiple "file names" (for lack of a better description) that point to the same inode. This only works if those hard links are on the [same file system](#).

`ln -s test.sh test_slink.sh` - Create a [symlink](#) with the new inodes.

Changing a name or the permissions on the symlink does not affect the original file.

4. **umask** - user mask. When creating a file or a folder, the OS assigns default access permissions. By default, directory access permissions are 0777 (rwx r-x r-x), and for a file 0666 (rw-rw-rw).

User permissions = 0777 - umask

umask, for a normal user,

```
umask 0002
touch test_root.sh = 664
-rw-rw-r--
```

For a root user (a sudoer)

```
umask 0022
touch test_regular.sh = 644
-rw-r--r--
```

5. Changing permissions is possible with **chmod**:

```
chmod a+rw test.sh - give write execute read permission to file;
chmod a-rw test.sh - permissions to noone;
chmod 0777 - apply umask;
```

for dir apply permission umask for all files in dir:

```
mkdir -p test & touch ./test/test1.sh ./test/test2.sh
chmod -R 0557 ./test
```

6. Special permissions:

- a. **Sticky bit** - forbidden to remove the directory except by the owner (бит t):

```
mkdir sticky_dir
chmod 1755 sticky_dir drwxr-xr-t
```

- b. **SUID**

e.g. a directory

```
-rwsr-xr-x 1 root root 54256 Mar 26 2019 /usr/bin/passwd,
```

allows to change passwords not only for a root user

SUID allows users to run an executable with the file system permissions of the executable's owner or group respectively and to change behaviour in directories.

```
mkdir test.sh
chmod 6755 test.sh
```

- c. The **SGID** bit is analogous to **SUID**, but not the user but an owner group is set. Used with **+s**

```
chmod g+s /test.sh
```

7. **File descriptors**. A file descriptor is a number that uniquely identifies an open file in a computer's operating system. It describes a data resource, and how that resource may be accessed.

In Linux, libc opens for each launched application 3 unique file descriptors by default, numbering them as 0,1,2. **man stdio** / **man stdout**:

| Name | File descriptor | Description | Abbreviation |
|-----------------|-----------------|---|---------------|
| Standard input | 0 | The default data stream for input, for example in a command pipeline. In the terminal, this defaults to keyboard input from the user. | stdin |
| Standard output | 1 | The default data stream for output, for example when a command prints text. In the terminal, this defaults to the user's screen. | stdout |
| Standard error | 2 | The default data stream for output that relates to an error occurring. In the terminal, this defaults to the user's screen. | stderr |

```
ls -lah /proc/$$/fd/ - prints the file descriptors
```

Piping 1 [20 min]

1. I/O redirection:

redirect to file >

redirect and append >>

```
echo "Hello!" > text
```

```
cat /proc/cpu_info 2>>/tmp/error.txt
```

a. Redirect standard output:

```
find /usr games 2> text-error
```

b. Redirect standard input:

```
uniq -c </tmp/error.txt
```

< - input of the file contents into the stdin of the process

c. Combination of output (channel 1) and error (channel 2) with &> (&>>):

```
find /usr admin &> newfile
```

```
(output input and error find: `admin': No such file or
directory
```

d. Commands:

```
less,
sort,
head,
tail,
grep;
tr,
```

tar,

2. Pipes:

- i. pv package - pipe viewer
- ii. cat /etc/passwd | grep root

The first command's output automatically becomes the second command's input

- iii. xargs works as an iterator over its input feeding through pipe. It is useful when you need to modify output of the pipe line-wise. E.g.

```
cat /etc/passwd | cut -d: -f1 | xargs -I'{}' echo
'{}@skoltech.ru'
```

3. standard filesystem contents & conventions.

- a. /bin - executable binaries (cat, kill), shipped with the system.

```
ls /bin | grep cat
```

- b. /var - journal and log files /var/log; /var/log/messages

- c. /etc - editable configuration files;

- d. /usr - system resources,

- i. /usr/bin - user executable binaries that can be invoked by all users (sed, awk, curl)

- ii. /usr/local/bin - external compiled commands

- e. /dev - directory with device files for reading and writing

- i. /dev/sd* hard drives

- ii. /dev/null - an empty device where you can redirect output of stdout or any other stream

- f. /dev/urandom - random num generator

- g. /proc - virtual filesystem

```
ls -llt /proc - each folder corresponds to a process identifier (PID) of each process
currently running on a system.
```

with /proc one can view the filesystem and file descriptors of various processes /proc:

```
echo $$ - which PID called echo
```

```
ls -lah /proc/$$/fd/ - shows file descriptors for stdin stdout stderr of the echo
process
```

```
echo "hello world" > /proc/$$/fd/0 - shows hello world, which redirects to stdin
(does not work on Darwin)
```

Reading

- Carinhas, Philip. "Linux Fundamentals." (2000). [PDF version](#)
- Ceruzzi, Paul E., E. Paul, and William Aspray. A history of modern computing. MIT press, 2003. (<https://mitpress.mit.edu/books/history-modern-computing>)
- https://www.softcover.io/read/fc6c09de/unix_commands