

Вычислительные системы. Лабораторная работа №3.  
8 факультет, 1 курс, 1 семестр 2018/19 учебного года  
Сети и телекоммуникации в ОС UNIX

«Компьютер без сети подобен телефону,  
не включенному в розетку!»

Козьма Прутков?

## 0. Сценарий работы

- 0.1. Изучить справочный материал и дополнительную литературу. Прослушать лекцию, посетить консультацию.
- 0.2. Собственноручно проделать основные примеры и продемонстрировать владение сетевыми средствами преподавателю. Обязательно выполнить не менее двух заданий связи с внешней средой (пп. 6, 7, 9–10).
- 0.3. Запротоколировать содержательное подмножество сеанса, за исключением команд, дающих нетекстовый вывод, и непротоколируемых пунктов 6, 7, 9–10 с учетом замечаний преподавателя. Оформить отчет на бланке.

## 1. Удаленные команды UNIX

Во всех примерах предполагается, что пользователь с именем 118001 работает на машине axp4 в своей домашней директории (/stud/118001) и что в ней существуют файлы file1, file2 и директории dir1, dir2. В промптере интерпретатора команд системы отображаются: имя машины (axp4), пользовательское имя (118001) и текущая директория (~ / dir1, ~ обозначает домашнюю директорию пользователя):

```
118001@axp4:~/dir1$
```

Убедиться в доступности сервера в сети TCP/IP можно с помощью команды ping:

```
ping 192.168.2.202 ping alpha ping ejudge
```

### 1.1 Переход на удаленную машину

Исторически осуществлялся с помощью различных команд UNIX берклеевского и TCP/IP-шного происхождения:

**rlogin <имя\_узла>** (запускается процесс **login** того же пользователя на псевдотерминале указанной удаленной машины, как правило без запроса пароля, когда между серверами установлены доверительные отношения);

**telnet <имя\_узла>** (запускается процесс инициализации псевдотерминала **getty** на удаленной машине). При использовании **telnet** как обычно при входе в систему, необходимо ввести имя пользователя и пароль.

Интерфейс **telnet** часто используется для настройки функциональных программ различного оборудования, реализованных на базе ядра UNIX (роутеры, телевизоры типа Smart TV и др.).

В настоящее время используется *современная, защищенная версия telnet* с шифрованием и компрессией данных **ssh <имя\_узла>**

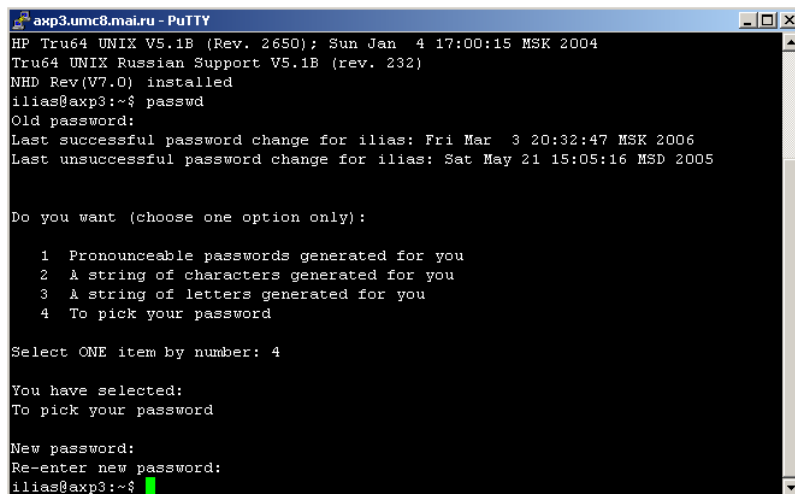
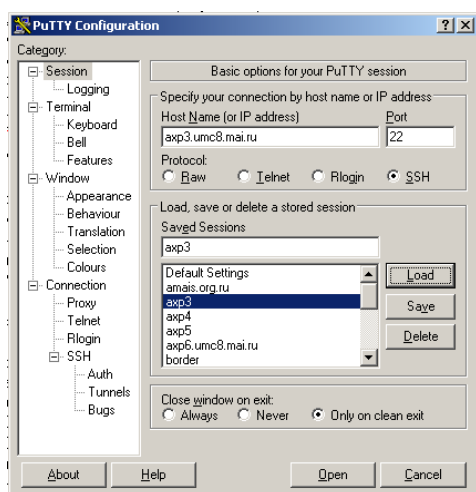
**1.2** Выполнение одиночных команд пользователя на удаленной машине без логина осуществляется командой **rsh** или безопасным образом с помощью **ssh**:

```
ssh [-l <пользователь>] <имя_узла> <команда> или
```

```
ssh [<пользователь>@]<имя_узла> <команда>
```

**1.3** Для эмуляции сеанса терминала удаленной ЭВМ в среде MS Windows удобно использовать свободно распространяемый клиент PuTTY (<http://www.chiark.greenend.org.uk/~sgtatham/putty/>).

Среди предлагаемых утилитой протоколов – берклеевский **rlogin**. При вызове PuTTY появляются окно PuTTY Configuration, где можно настроить параметры терминального соединения. После установления связи с сервером появится окно-эмулятор терминала. Ввиду замечательных особенностей протокола TCP/IP эмуляция терминала с помощью PuTTY возможна через глобальную сеть (интернет) для любого компьютера сети, доступной для оказания соответствующих услуг.



**1.4.** Для установления доверительного доступа к удалённому компьютеру, предоставляющему услуги доступа по протоколу **ssh**, надо сгенерировать пару асимметричных ключей (публичный и приватный):

```
ssh-keygen -t dsa
```

Далее публичный ключ копируется на удалённую машину в специальный подкаталог `.ssh`

```
ssh cameron 'cat >>.ssh/authorized_keys' <./ssh/id_dsa.pub
```

## 2. Удалённое копирование файлов и директорий (на примерах)

Производятся т. н. «берклеевскими» (разработанными в Калифорнийском университете в Berkeley и ставшими одним из стандартов *de facto*) удаленными (**remote**) командами: **rcp**, **rsh** и **rlogin**. В настоящее время в лабораторном классе используются только защищенные (**secure**) версии команд копирования и эмуляции терминала: **scp** и **ssh**, отличающиеся от их первоначальных версий **rcp** и **rsh** шифрованием сетевого трафика и аутентификацией сторон. Кстати, **PutTY** тоже может быть использован для защищенного копирования файлов (дополнительная команда **pscp**).

**2.1.** Три способа копирования файла того же пользователя с текущей машины **axp4** на удаленную машину **axp5**:

```
scp file1 axp5:/stud/118001
```

```
scp file1 axp5:~/file1
```

```
scp file1 axp5:`pwd`
```

(апострофы обратные, означают подстановку результата выполнения команды **pwd** на текущей машине в аргумент команды **scp**; здесь текст становится командой, а в результате её выполнения генерируется имя целевой директории).

Таким образом эта команда означает копирование файла **file1** с текущей ЭВМ в директорию, одноименную текущей, на удаленной ЭВМ, т. е. подразумевается идентичность файловых структур обеих ЭВМ. В случае различных иерархических структур файловых систем не следует использовать конструкцию с **`pwd`**.

**2.2.** Копирование файла с удаленной машины **axp6** на текущую (**axp4**).

```
scp axp6:/stud/118001/file2 dir1/file3
```

```
scp axp6:~/file2 .
```

(точка означает текущую директорию)

```
scp axp6:`pwd`/file2 file3
```

(апострофы обратные, в команде два пробела, после **scp** и перед **file3**!)

В первом случае файл копируется в директорию **dir1** под именем **file3**, 3-й случай предполагает *идентичность* структур файловых систем исходной и целевой машин.

**2.3.** «Тройной тулуп». Копирование файла с одной удаленной машины на другую с переименованием; команда выдается с *третьей* машины (например, это может быть **axp3**):

```
scp cameron:`pwd`/file1 alpha:`pwd`/file3
```

Вместо **`pwd`** подставляется путь к текущей директории на текущей ЭВМ, т. е. данная команда также предполагает *идентичность* файловых иерархий на всех вовлеченных в указанную операцию машина; работает только в случае наличия открытого ключа пользователя машины **axp6** на узле **axp3**. Копирование происходит, минуя машину, на которой выполняется команда.

**2.4.** Копирование директории вместе со всем ее содержимым на удаленную машину в домашний каталог пользователя (предполагается, что на удаленной машине **axp6** директории с именем **dir1** не существует):

```
scp -r dir1 axp6:
```

Для синхронизации каталогов на разных ЭВМ удобно использовать команду **rsync**:

```
rsync -a dir axp6:
```

**2.5.** Копирование директории с удаленной машины вместе со всем ее содержимым в текущую директорию пользователя под тем же именем:

```
scp -r axp6:dir2 .
```

обратите внимание: точка.

Здесь также предполагается, что директория **dir2** на текущей машине отсутствует, а на удаленной машине находится в домашнем каталоге пользователя. В противном случае в соответствии с ключом **-r** часть файлов будет заменена одноименными вновь пришедшими.

```
tar -cf - dir1 | ssh axp6 'tar -C /stud/118001 -xf -'
```

эквивалентно **2.4** (апострофы обычные!), но копирование осуществляется с сохранением времён последних модификаций и режимов (**modes**) доступа для всех файлов внутри **dir1**. Этот вариант гораздо быстрее, поскольку при большом количестве копируемых файлов большая часть времени тратится на установление соединений. Это особенно актуально при использовании низкорезактивных беспроводных сетей (**Yota** и, особенно, спутниковый канал). Программа архивации **tar** будет рассмотрена далее. Там же вы узнаете, что в этом примере лучше использовать **pax**:

```
pax -w dir | ssh axp6 'pax -r'
```

Для проверки и **протоколирования** сеансов с **scp**, **ssh** применяются комбинации команд **ls**, **cat** до и после проверяемой (протоколируемой) операции для текстовых файлов небольшого размера.

Фрагмент протокола копирования файла в специально созданный каталог на удаленной машине:

```
118001@axp5:~$ ls -l
total 2
-rw-r--r--  1 118001  stud      70 Sep 10 15:10 f1.txt
-rw-r--r--  1 118001  stud      46 Sep 10 15:10 f2.txt
118001@axp5:~$ ssh axp6 "mkdir home" # здесь кавычки можно опустить!
118001@axp5:~$ scp f2.txt axp6:~/home/f1.txt
118001@axp5:~$ ssh axp6 "ls -l home"
6total 1
-rw-r--r--  1 118001  stud      46 Aug  6 19:07 f1.txt
118001@axp5:~$ cat f2.txt
file1
Hello, World!
Hello, Ken!!!
118001@axp5:~$ ssh axp6 "cd home; cat f1.txt"
file1
Hello, World!
```

Hello, Ken!!!

## 2.6. Сетевая файловая система NFS

NFS предложена фирмой Sun более 25 лет назад (в 1984 г.) и реализована во многих версиях ОС UNIX. В MS Windows также имеются клиентский сервис NFS (службы NFS для Windows). Другими известными сетевыми новациями Sun того времени являются удаленный вызов процедур RPC и соответствующий формат представления данных XDR (External Data Representation).

NFS обеспечивает прозрачный доступ пользователя к файловым системам удаленных ЭВМ как к локальным поддиректориям без какого-либо изменения программного обеспечения. Так, находясь на ПЭВМ, можно с помощью, например, редактора jEdit редактировать файлы, находящиеся на сервере SUN UltraSparc, если, конечно, это дозволено правами доступа. Данная концепция прозрачного доступа к удаленным файловым системам является основной для локальных сетей в среде MS Windows.

NFS реализуется с помощью резидентных процессов-демонов, и даёт дополнительную нагрузку на сеть и локальные ресурсы ЭВМ.

С помощью NFS также можно осуществлять межмашинное копирование и перемещение файлов.

В настоящий момент каталоги пользователей хранятся на файловом сервере лаборатории и экспортируются по NFS: на доступных студентам машинах в директориях `/stud` и `/home` смонтированы одноимённые файловые системы файлового сервера **cameron**. Поскольку студенческие машины реализованы как бездисковые рабочие станции, то и вся корневая директория `/` также смонтирована на файловом сервере с помощью NFS.

Кроме того, на сервере тестирования alpha каталоги удалённых машин монтируются автоматически под именами соответствующих машин при первом обращении в директории `/net`. Текущая машина доступна, кроме того, под названием `localhost`. Таким образом, при наличии NFS полный путь к файлам на каждом компьютере сети может начинаться с `/net/localhost` при условии, что соответствующая файловая система экспортирована.

Смонтированные удаленные файловые системы используются точно так же, как и локальные (т.е. межмашинное копирование/перемещение между директориями конкретного пользователя можно осуществлять *обычными* методами: `cp`, `mv` (!) и т.д.). Кроме того, работая на одной машине, можно по NFS перейти при помощи `cd` в свою директорию на другой машине и работать там. Однако, не надо забывать, что постоянная интенсивная работа по NFS перегружает локальную сеть теми обменами, которые на локальной файловой системе выполняются дисковым контроллером.

### Пример:

```
cp file1 /net/axp5/stud/118001      эквивалентен примеру в пункте 2.1
```

```
scp file1 axp5:/stud/118001
```

Таким образом NFS позволяет выполнять работы на дисках сетевых ЭВМ также просто, как на локальных.

Для работы NFS необходимо присутствие в памяти сетевых ЭВМ соответствующих фоновых процессов-демонов `nfsd`, в чем можно убедиться командой `ps auxw | grep nfsd`.

Проверка и протоколирование копирования файлов по `nfs` осуществляется аналогично берклеевским удалённым командам.

## 2.7. Кластеры UNIX.

Операционная система UNIX часто бывает оснащена средствами кластеризации. Установленное на группе серверов локальной сети программное обеспечение кластеризации позволяет автоматически перераспределять задания между серверами, обеспечивая их равномерную загрузку. При этом никаких изменений выполняемых заданий не требуется.

## 2.8. Виртуальные приватные сети

В локальной сети учебной лаборатории такие сети используются для соединений по небезопасным протоколам.

## 3. Передача файлов при помощи протокола FTP

Концепция сетевого сервиса `ftp` (File Transfer Protocol) лучше соответствует глобальным сетям, нежели NFS или берклеевские сетевые утилиты. В частности, каждый сеанс соединения по `ftp` не шифруется и осуществляется по явной парольной авторизации, в том числе допускающей анонимный доступ. Для еще большей безопасности существуют защищённые версии протокола `ftp`: `sftp` и `sftp2` — с шифрованием и компрессией трафика. При запуске `sftp` перед именем узла сети надо через `@` указать имя пользователя. Например: `sftp 118001@axp4`.

Передача файлов UNIX–UNIX с помощью `sftp` осуществляется *абонентами* `sftp` (пользователями, которым разрешен этот сервис) в следующем порядке:

**3.1.** Запустить утилиту-клиент `sftp` для подсоединения к узлу сети, с которым надо осуществить обмен файлами по протоколу `ftp`:

```
sftp axp4
```

**3.2.** Указать имя пользователя и, после запроса, пароль. В случае успеха появится сообщение `User 118001 logged in` и промптер `sftp>` – приглашение ко вводу команд FTP. Далее выдаются внутренние команды `sftp` (в том числе такие как, аналогичные UNIX и MS DOS (!) `cd`, `pwd`, `ls`, `dir`).

**3.3.** Проверить с помощью команды `sftp pwd` имя целевой директории, в которую попадает клиент `sftp` по умолчанию.

```
sftp> pwd
Remote directory: /stud/118001
```

**3.4.** В случае несоответствия перейти в свою домашнюю директорию на удаленной машине:

```
cd /stud/118001
```

3.5. Просмотреть содержимое текущей директории на удаленной машине командой **ls -l** (или **dir** - для получения более подробной информации), а на локальной машине **!ls**. Для смены текущей директории на локальной машине полезна команда **lcd**.

3.6. Получить файл **file1** с удаленной машины:

```
get file1
```

3.7. Передать файл **file2** на удаленную машину:

```
put file2
```

Предполагается, что передаваемые и получаемые файлы существуют.

3.8. Завершение **sftp**: **Ctrl-D**, **bye** или **quit**.

**ЗАМЕЧАНИЕ.** Передача нетекстовых файлов (например, **tar**-архивов), осуществляется по умолчанию в режиме **binary**. Режим передачи текстовых файлов **sftp** устанавливается внутренней командой **sftp ascii**.

3.9. Существуют аналоги команд **get** и **put** для групповой передачи файлов по маске (имена порождаются регулярными выражениями).

```
mput *.tu
```

```
mget kurs?.c
```

Возможно сокращение команд **sftp** до двух-трех букв (так, чтобы сокращение было бы однозначно), например команде **binary** соответствует **bin** или **b** (в разных **sftp**-клиентах количество команд различно, но основной набор неизменен).

Для *протоколирования* сеансов **sftp** надо выдавать подтверждающие команды **ls**, **cat** и **hostname**, иногда предварённые **ssh**. Поскольку данный клиент **ftp** имеет строчный интерфейс, протоколирование его работы ведётся также, как и фиксация обычного сеанса UNIX.

Подробности об утилитах семейства **ftp** можно найти во многих книгах и документах, посвященных UNIX, глобальным сетям и интернет.

В командной строке **ftp** можно инициировать выполнение команды UNIX, разрешённой пользователю на *локальной* машине:

```
ftp> !ls -l
```

## 4. Архивация и упаковка

В ОС UNIX существуют специальные программы для упаковки и архивации данных, известные далеко за пределами этой системы. К ним относятся такие программы, как **tar**, **gzip** и **pax**. Большинство архивных форматов Unix (**tar**, **gzip**, **bzip**, **bzip2**) понимают Windows-архиваторы WinRAR и 7Zip, а также распространенные файловые менеджеры Total Commander и FAR (с подключаемыми модулями). Существуют UNIX-утилиты **unrar** и **unzip** для распаковки архивов **rar** и **zip**.

Сжатие и упаковку целесообразно применять для транспортировки данных. Команда **tar** применяется для создания архива множества файлов с сохранением иерархической структуры каталогов, прав доступа и владельцев, а **gzip** — для сжатия одного файла, например, того же архива, созданного **tar**, с целью экономии места для хранения и сокращения времени его передачи по каналам связи.

Таким образом, архиватор **tar** позволяет заменить несколько файлов и директорий одним архивным файлом, а упаковщик **gzip** используется для упаковки/распаковки передаваемых файлов/архивов.

В последние годы архиватор **pax** считается более удобным и совершенным, чем стандартная связка **tar** и **gzip**.

4.1. Архивация в режиме командной строки UNIX и MS Windows осуществляется одинаково:

```
tar -cf имя_архива.tar имя_директории либо
```

```
pax -w -x tar имя_директории > имя-архива.tar либо
```

```
pax -w имя_директории > имя-архива.pax
```

— создает архив и помещает в него все файлы и поддиректории указанной директории.

4.2. Разархивация

```
tar -xf имя_архива.tar либо
```

```
pax -r < имя-архива.pax
```

4.3. Просмотр оглавления архива

```
tar -tvf имя_архива.tar либо
```

```
pax < имя-архива.pax
```

4.4. Упаковка (может не применяться при передаче файлов по каналам с аппаратной поддержкой протоколов сжатия данных):

```
gzip -9 -c имя_архива.tar > имя_архива.tgz
```

```
pax -w -x cpio имя_директории > имя-архива.pax
```

```
tar -zcf имя_архива.tar.gz имена_сжимаемых_файлов
```

```
tar -jcf имя_архива.tar.bz2 имена_сжимаемых_файлов
```

4.5. Распаковка

```
gzip -d -c имя_архива.tgz > имя_архива.tar
```

4.6. Вариант распаковки с одновременной разархивацией (конвейер команд UNIX):

```
gzip -c -d a.tgz | tar -xf -
```

во многих утилитах UNIX под файлом **-** подразумевается стандартный ввод.

4.7. В настоящее время вместо **gzip** используется утилита **bzip2**, которая во многих случаях сжимает в полтора раза лучше.

4.8. Интересно, что имеется аппаратная реализация архиватора **gzip**  
<http://www.ixbt.com/news/hard/index.shtml?09/55/74>

## 5. Замечания по кодировке

При обменах данных UNIX↔WINDOWS возникает необходимость перекодировки, т. к. кодировка знаков кириллицы (и других со старшим битом кодового байта, равным единице: национальные алфавиты размещаются в старшей части кодовой таблицы), а также коды конца файла и конца строки в кодировках КОИ-8 и других, принятых в UNIX, и в кодировке в MS Windows отличаются. Поэтому переносимые из одной среды в другую файлы должны быть *перекодированы* не только как данные: необходимо также выполнить преобразование служебных элементов файла. Следует отметить что команда **dd** оригинального UNIX, допускающая перекодировку, тем не менее не может быть использована для перекодирования файлов с кириллицей из кодировки Windows CP-1251 в UTF-8. Вопросы перекодировки рекомендуется проработать **заранее** на домашних компьютерах.

Во всех современных версиях UNIX имеется удобный стандартный перекодировщик **iconv**. (Раздел XCU5 стандарта XOpen CAE Specification).

Вызов перекодировщика из командной строки UNIX для преобразования файла, закодированного КОИ-8, в кодировку Windows CP1251 осуществляется так:

```
iconv -c -f koi8-r -t cp1251 <файл>
```

Например, электронный словарь **dict** обычно выдает перевод в кодировке UTF8. Чтобы прочитать его на терминале со знакогенератором, прошитым в КОИ-8, можно воспользоваться следующим конвейером:

```
dict elephant | iconv -f utf-8 -t koi8-r
```

При переводе из одной кодировки в другую в целевой кодировке может не оказаться знака, соответствующего данному (например, при перекодировании текста с иероглифами из кодировки unicode в CP-1251, где есть только русские и латинские буквы, иероглифы не могут быть представлены и перекодировка прекращается). Для игнорирования таких ситуаций и предназначен ключ **-c**. Но при перекодировке, например, из упрощённого китайского в utf-8 (8-битный обратный совместимый с ASCII способ кодирования знаков стандарта ISO10646) таких исключительных ситуаций не возникнет, поскольку выходной алфавит богаче входного, и ключ **-c** не потребуется:

```
iconv -f gb2312 -t utf-8 <файл>
```

Результат **iconv** всегда помещается на стандартный вывод и может быть перенаправлен в файл. Поддерживается перекодировка групп файлов, заданных метасимволами имён.

При передаче по сети файлов из MS Windows в UNIX необходимо учитывать различие форматов текстовых файлов:

5.1. В MS Windows строка текстового файла заканчивается парой символов Carriage Return (CR) — возврат каретки (код 13, **\r**), и Line Feed (LF) — перевод строки (код 10, **\n**). В системе UNIX строка заканчивается *одним* символом LF. Перекодировщик **iconv** не удаляет из текста лишние символы CR и файл после передачи в UNIX становится «нетекстовым» и надо удалять знаки CR вручную в текстовом редакторе. Впрочем, Notepad++ и многие другие редакторы умеют преобразовывать текстовые файлы в необходимые форматы с правильными концами строк.

5.2. Для конвертации текстового файла в Windows-формат из UNIX-формата и обратно существуют утилиты **unix2dos** и **dos2unix**. Конвертация осуществляется на месте путем замены оригинального файла:

```
unix2dos <unix-file>  
dos2unix <dos-file>
```

5.3. Из текста не удаляются *символы* конца файла, проставляемые некоторыми редакторами MS Windows типа Norton Editor, Lexicon и пр. Эти символы в MS Windows обозначаются значком ← (код 26, Ctrl+Z).

5.4. В текстовых редакторах jEdit, Notepad++, встроенном редакторе файловой оболочки FAR также существует возможность записать текстовый файл сразу в кодировке koi8-r и в Unix-формате.

5.5. Форматы файлов MS Word и Excel являются нетекстовыми по определению и не конвертируются средствами Unix. Некоторые программы допускают сохранение файла с конвертацией в ASCII Text — в этом случае можно сохранить файл в текстовом формате, а потом конвертировать в КОИ-8 обычным образом. Переносимые в UNIX нетекстовые данные в перекодировании не нуждаются, нужно только правильно выбрать режим передачи таких файлов (**bin**). (*Кстати, Postscript- и XML-документы всегда текстовые!*)


5.6. Для операционной системы Windows стандартной кодировкой русских букв является cp1251 или Unicode (UTF16). Для конвертации файлов, содержащих русские буквы в этих кодировках, можно использовать другие средства, например внутренний редактор FAR или тот же **iconv**. Удобно делать это с помощью современных текстовых редакторов jEdit (Buffer Options) и Notepad++ (Plugin→ Convert→ ...).


5.7. Владение средствами перекодировки особенно важно при работе на различных программно-аппаратных платформах с разными алфавитами.

## 6. Обмен файлами между MS Windows и UNIX. Floppynet

Для обмена данными между флешками и ОС UNIX предусмотрен интерфейс USB на лицевой панели ПЭВМ или разъём на удлинителе, подключенном к задней панели.

6.1. Для ПЭВМ, функционирующей под управлением MS Windows, flash-накопитель необходимо воткнуть в гнездо USB, войти в Windows под гостевым аккаунтом и убедиться, что флешка обнаружилась Windows (должен

появиться новый накопитель среди аналогичных устройств; о появлении нового устройства сигнализирует иконка  в системном трее).

После окончания передачи данных с flash-накопителя его необходимо корректно отмонтировать: закрыть или увести с накопителя все использующие его программы, кликнуть левой кнопкой по иконке  и выбрать свою флешку из списка USB-устройств, предлагаемых к отсоединению от системы. Это запустит процедуру размонтирования, после чего можно вынуть накопитель из разъема.

6.2. Передача данных с USB-устройств на UNIX-сервера через USB-порты рабочих станций UNIX на платформе Intel, находящихся в терминальном классе, осуществляется аналогично. В современных версиях UNIX монтирование/размонтирование съёмных накопителей осуществляется автоматически. Файловая система правильно воткнутой и смонтированной флешки как правило располагается в директории /media/метка-тома-флешки. Перед извлечением флешки из USB-разъёма её необходимо размонтировать. Легче всего это сделать средствами среды Gnome, кликнув правой кнопкой мыши по имени раздела флешки в списке устройств в файловом навигаторе Nautilus. В появившемся контекстном меню выбрать пункт размонтирования.

Запрещается подключение энергоёмких USB-устройств (винчестеров, CD-ROM и т. п.).

## 7. Удаленный доступ к лабораторной системе UNIX

Протоколы ftp и telnet являются старейшими услугами удаленного доступа к ОС UNIX, широко применяемыми в сети интернет. Практически с любого компьютера, подключенного к сети Интернет, можно получить доступ к любому другому (или даже к своему) узлу, обеспечивающему серверные услуги ftp и telnet.

Например, при выставлении серверов лаборатории в интернет можно получить удаленный доступ к серверам УМЦ-8 из интернет базовыми утилитами (в DOS-окне Windows, из командной строки Linux и т. д.):

```
sftp axp3.umc8.ru
```

```
ssh axp4.umc8.ru
```

Для обращения к этим серверам из MS Windows можно также использовать любой FTP-клиент (FAR, CuteFTP, Total Commander), либо веб-браузер (Mozilla Firefox, Opera, Internet Explorer, Lynx, Links) — в этом случае надо использовать универсальный локатор ресурса URL вида:

```
sftp://axp3.umc8.ru
```

## 8. Электронная почта

Локальные средства электронной почты между пользователями UNIX были с легкостью распространены на глобальные сети, обгоняя другие услуги интернета.

Передача и маршрутизация почты как между ЭВМ, так и локально, осуществляется системным демоном UNIX, например, **sendmail** или **postfix**. Полученные сообщения для конечных пользователей сохраняются в специальной директории на локальном компьютере для дальнейшего прочтения.

Чтение и посылка почты производится почтовыми клиентами, например, командой **mail** или **mutt**. В письме указывается адрес получателя (пользовательское имя для локального письма, либо полное сетевое имя для сетевой пересылки), предмет письма (Subject) и собственно текст письма, который может быть взят из файла или стандартного ввода.

**Примеры** посылки программы и тестовых примеров на проверку преподавателю:

```
mail alk < program1.c  
a.out | mail alk
```

Широкое распространение в UNIX получил более мощный клиент **mutt**, поддерживающий MIME, шифрование и цифровую подпись PGP.

**Замечание 1:** Вследствие нерусифицированности некоторых почтовых серверов (что, впрочем, не противоречит стандартам RFC) кириллическая почта нередко латинизируется установкой старшего бита каждого кодового байта в ноль. Одно из решений проблемы передачи кириллицы по 7-битному каналу — кодировка типа VOLAPUK, в которой русские буквы кодируются латинскими фонемами. Другой способ — архивация 8-битного текста и передача архива в двоичном режиме ftp или использование текстового кодирования бинарных файлов (MIME).

**Замечание 2:** Для передачи писем между компьютерами сети УМЦ-8 надо указать имя пользователя на целевой машине в следующем формате: user@host, например, 118001@axp3.umc8.ru или в кратком варианте 118001@axp3. При помощи служб SMTP и DNS письмо дойдёт до адресата, находящегося на любой ЭВМ локальной сети.

## 9. Беспроводной доступ к сети

Для беспроводного доступа к сети в учебном классе УМЦ-8 установлена точка доступа WiFi. Точка доступа представляет собой одноплатную бездисковую сетевую (Ethernet+Wireless) микро-ЭВМ, работающую под управлением функциональной программы (т.н. прошивки) — сетевой операционной системы, которая, как правило, является версией Linux. Эта программа при включении питания автоматически загружается в оперативную память из флеш-памяти и может обновляться по сети. Точка доступа ZyXEL G-560 реализует модифицированный протокол IEEE 802.11g+ (от 56 Мбит/с до 125 Мбит/с). Её описание находится на сайте компании ZyXEL по адресу <http://zyxel.ru/content/catalogue/48/6/20/256> Точка доступа ZyXEL любезно предоставлена сотрудником Российского представительства, выпускником факультета № 8 Алферьевым В. Н.

Чтобы осуществить беспроводной доступ, студент должен иметь соответствующее устройство со встроенным или внешним WiFi: ноутбук, КПК, планшет или телефон. При включенной функции WiFi устройство

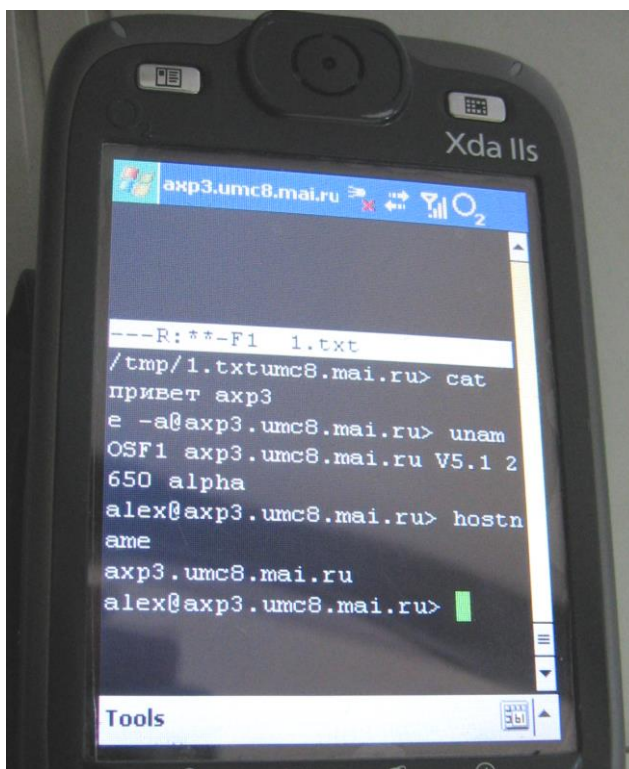
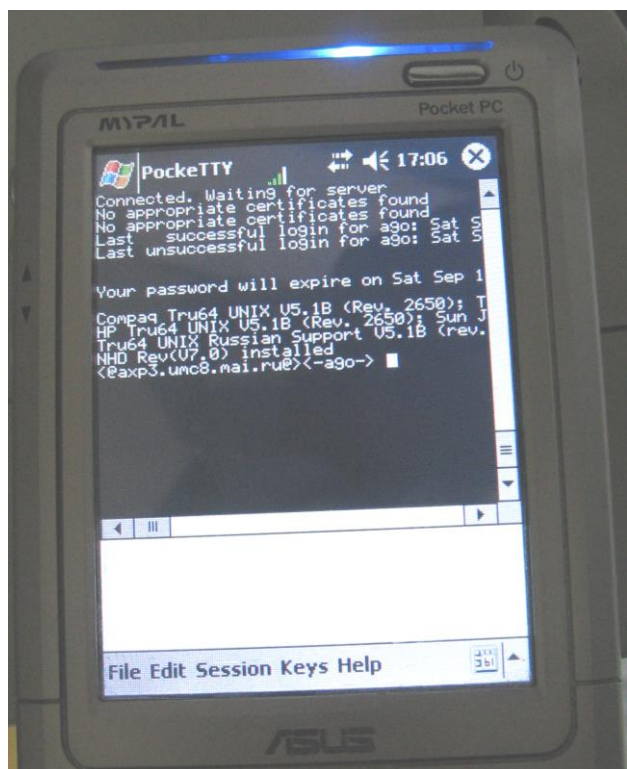
автоматически ищет и отображает доступные беспроводные сети. Беспроводной доступ к сети UNIX-класса 438<sup>B</sup> осуществляется любым терминальным клиентом (тем же PuTTY) через беспроводную сеть ZyXEL. Поскольку DNS-сервер при этом назначается автоматически, доступ к узлам сети возможен по именам, а не по IP-адресам.

Аналогично осуществляется доступ с устройств протокола Bluetooth. Точка доступа — D-Link DBT-900AP — реализует протокол Bluetooth v1.1 со встроенным 128-битным шифрованием, обеспечивая доступ к сети Ethernet для не более чем 7 пользователей на расстоянии до 20 м с помощью релевантного терминального клиента в установленном на телефоне или КПК программном обеспечении. Например, на КПК Fujitsu Siemens Loox 720 под управлением MS Windows Mobile 5.0 удалось не только установить связь с точкой доступа, но и с помощью ftp-клиента SoftX FTP произвести передачу файлов:

Диспетчер BlueTooth → Новое → Соединить с сетью → \*в списке доступных устройств появляется DBT-900AP\* → Ярлык → **указать пароль доступа (root)** →

Далее был запущен ftp-клиент и передача файлов осуществлялась обычным образом.

С различных КПК удалось осуществить соединение по обоим беспроводным интерфейсам. Для обмена использовались Pocket-версии PuTTY: **PockeTTY** и **Pocket PuTTY**



*«Пиво отпускается только членам профсоюза»*

*И. Ильф, Е. Петров*

**10.** Доступ в интернет во время лабораторных работ в случае необходимости для выполнения конкретных заданий осуществляется обычным для UNIX браузером, например Mozilla Firefox. Некоторый интерес представляет использование текстовых браузеров на терминале или в текстовом окне. Опробовать их можно и в локальной сети TCP/IP.

## Литература

1. Цикритзис Д., Бернстайн Ф. *Операционные системы*. - М.: Мир, 1977.
2. Танненбаум Э. *Многоуровневая организация ЭВМ*. - М.: Мир, 1979
3. *UNIX on-line manuals (rcp, rlogin, rwho, telnet, rsh, ruptime, hostname, ftp)*.
4. Дейтел Г. *Операционные системы*. - М.: Мир, 1987. Том 2, пп. 16.1,2,3,10,11,12, гл. 18,19.
5. Кристиан К. *Введение в операционную систему UNIX*. - М.: Финансы и статистика, 1985.
6. Баурн С. *Операционная система UNIX*. - М.: Мир, 1986.
7. *Официальный сайт проекта GNU: <http://www.gnu.org>*
8. *Архив программного обеспечения Simtel: <http://www.simtel.net/>*
9. Готье, *Руководство по ОС UNIX*. - М.: Финансы и статистика, 1985.
10. МакМаллен Дж., *UNIX (Complete Idiot's Guide) // Пер. с англ.* - М.: ЮНИТИ, 1996.
11. Браун П., *Введение в операционную систему UNIX*. - М.: Мир, 1987.
12. *UNIX. Руководство системного администратора*. - К.: BHV, 1997.
13. Банахан М., Раттер Э. *Введение в операционную систему UNIX*. - М.: Радио и связь, 1986.

14. Керниган Б.В., Пайк Р. UNIX – универсальная среда программирования. – М.: Финансы и статистика, 1992.
15. Храмов П., Лабиринты Интернет. - М., 1995.
16. Топхем Д., Хай Ван Чыонг. Юникс и Ксеникс: Пер. с англ. – М.: Мир, 1988.
17. Тихомиров В.П., Давидов М.И. Операционная система ДЕМОС: инструментальные средства программирования. – М.: Статистика, 1988. (C Shell, make, lex, yacc).

#### Вопросы к зачёту. ОС UNIX. Сети.

1. Основы построения компьютерных сетей. Понятие протокола. Протоколы различных уровней.
2. Семейство протоколов TCP/IP как основа построения локальных и глобальных сетей на базе ОС UNIX.
3. Доменная система имен. Сеть интернет.
4. Основные сервисы в рамках интернет: ftp, telnet, email и т.д.
5. Удаленные команды login, who, sh и uptime.
6. Команды идентификации узла сети, системы, пользователя, сеанса и терминала.
7. Эмуляция терминала удалённой ЭВМ (telnet, PuTTY).
8. Удаленное копирование и передача файлов и директорий (rcp, ftp, ...).
9. Безопасность удаленных команд (scp, sftp, ssh, ...).
10. Доступ к файловым системам сети с помощью NFS.
11. Кластеризация ЭВМ в сети ОС UNIX.
12. Управление очередью печати.
13. Команды dd, tr и iconv и их использование для перекодировки файлов.
14. Формат текстового файла в различных ОС. Способы преобразования к формату UNIX.
15. Электронная почта в ОС UNIX. Адресация абонентов. Приёмы использования стандартного почтового клиента.
16. Архивация файлов и директорий с помощью tar и рах. Обслуживание архива tar или рах: просмотр оглавления, выборка, разархивация.
17. Сжатие/распаковка файлов с помощью gzip и bzip2.

Составители: проф. Зайцев В. Е., ст. преп. Сеницкий П. А., Дзюба Д. В., Лебедев А. В., Перетягин И. А., Овечкис А. Г., Филимонов Н.С., доц. Сошников Д. В., асс. Измайлов А. А. и прогр. Артамонов В.Н.