Администрирование Linux. Часть3.

Пособие для дистанционных курсов.
http://linuxnavigator.ru.

03.07.2009

Оглавление

Первоначальная настройка сервера. 3
Управление службами. 3
Запуск службы при старте компьютера. 3
Запуск и остановка служб вручную. 4
Настройка сети. 5
Интерфейсы. 5
Маршрут по умолчанию. 6
Отключение zeroconf. 6
Имена интерфейсов. 6
Клиент DNS. 7
Обновление системы. 7
Yum. 8
Репрозиторий dag. 8
Настройка ssh. 9
Краткое описание. 9
Запрет входа суперпользователя. 9
Использование ключей. 10

ПЕРВОНАЧАЛЬНАЯ НАСТРОЙКА СЕРВЕРА.

После установки сервера необходимо настроить сеть и обновление программного обеспечения. Вообще то, сеть можно было настроить сразу при установке, но я подумал, что научить вас изменять параметры сети на уже установленном сервере — это очень позитивно. Поэтому сеть будем настраивать руками.

Обновление программного обеспечения в CentOS производится при помощи программы *yum*. Это надстройка над стандартным менеджером пакетов *rpm*, облегчающая задачи установки, обновления и удаления пакетов. По большому счету она уже настроена, нам только потребуется указать дополнительный источник пакетов (репозитарий), в котором находятся полезные программы, не вошедшие в дистрибутив.

УПРАВЛЕНИЕ СЛУЖБАМИ.

В данном курсе не стоит задачи изучения системы инициализации дистрибутива, но вам придется управлять различными службами: запуск, остановка, запуск при старте компьютера и прочее. Мы рассмотрим, как это эти задачи выполняются в дистрибутивах с системой инициализации SystemV, таких как: RedHat, CentOS, SuSE и их производных.

ЗАПУСК СЛУЖБЫ ПРИ СТАРТЕ КОМПЬЮТЕРА.

Для управления запуском службы при старте компьютера используется программа *chkconfig*. Она позволяет:

- Посмотреть какие сервисы, на каких уровнях выполнения запускаются.
- Включить запуск сервиса.
- Выключить запуск сервиса.

Параметр *--list*, заставляет программу выводить на стандартный вывод список сервисов, а так же номера уровней выполнения.

```
# chkconfig --list | head
NetworkManager  0:выкл  1:выкл  2:выкл  3:выкл  4:выкл  5:выкл  6:выкл
NetworkManagerDispatcher        0:выкл  1:выкл  2:выкл  3:выкл  4:выкл  5:выкл 6:выкл
acpid           0:выкл  1:выкл  2:выкл  3:вкл   4:вкл   5:вкл   6:выкл
anacron         0:выкл  1:выкл  2:вкл   3:вкл   4:вкл   5:вкл   6:выкл
apmd            0:выкл  1:выкл  2:вкл   3:вкл   4:вкл   5:вкл   6:выкл
atd             0:выкл  1:выкл  2:выкл  3:вкл   4:вкл   5:вкл   6:выкл
auditd          0:выкл  1:выкл  2:вкл   3:вкл   4:вкл   5:вкл   6:выкл
autofs          0:выкл  1:выкл  2:выкл  3:вкл   4:вкл   5:вкл   6:выкл
avahi-daemon    0:выкл  1:выкл  2:выкл  3:вкл   4:вкл   5:вкл   6:выкл
avahi-dnsconfd  0:выкл  1:выкл  2:выкл  3:выкл  4:выкл  5:выкл  6:выкл
#
```

В этом примере список ограничен десятью строками, на самом деле у вас установлено гораздо больше служб.

Если быть более точным, то программа выводит список стартовых скриптов, расположенных в директории */ets/rc.d/init.d*.

Если вам необходимо получить информацию о конкретной службе, просто допишите имя её стартового скрипта в конце командной строки.

```
# chkconfig --list sshd
sshd            0:выкл  1:выкл  2:вкл   3:вкл   4:вкл   5:вкл   6:выкл
#
```

В примере показано, на каком уровне выполнения будет запускаться служба ssh. С точки зрения сервера, нас интересует 3-й уровень выполнения. Как вы видите, на третьем уровне служба будет включаться.

Что бы отключить запуск службы при старте компьютера следует выполнить следующую команду:

```
# chkconfig sshd off
# chkconfig --list sshd
sshd            0:выкл  1:выкл  2:выкл  3:выкл  4:выкл  5:выкл  6:выкл
#
```

После выполнения команды, программа *chkconfig* покажет, что данная служба при старте компьютера запускаться не будет.

Обратите внимание, что эта команда не выключает, работающий в данный момент демон *sshd*. Для выключения нужно использовать другие способы.

Для включения службы при старте компьютера, программе *chkconfig* надо давать параметр *on*.

```
# chkconfig sshd on
# chkconfig --list sshd
sshd            0:выкл  1:выкл  2:вкл   3:вкл   4:вкл   5:вкл   6:выкл
#
```

Chkconfig не включает службу, он только делает так, что бы служба запускалась при старте компьютера.

ЗАПУСК И ОСТАНОВКА СЛУЖБ ВРУЧНУЮ.

Большинство программа можно запускать прямо в командной строке, указав имя программы и параметры. Управлять программой и останавливать ее можно посылая соответствующие сигналы. Но сейчас рекомендуется пользоваться стартовыми скриптами соответствующих служб.

Стартовые скрипты располагаются в директории */etc/rc.d/init.d*. К ним так же можно добраться по символьной ссылке */etc/init.d*.

Обычно стартовые скрипты поддерживают несколько параметров:

- start — запуск службы.
- stop — остановка службы.
- restart — перезапуск службы.
- status — получение информации о службе.

Кроме перечисленных, могут использоваться и другие параметры.

Давайте рассмотрим примеры.

Проверка запущен или нет сервис xinetd, при помощи стартового скрипта сервиса.

```
# /etc/init.d/xinetd status
xinetd (pid 2099) выполняется...
#
```

Остановка сервиса xinetd.

```
# /etc/init.d/xinetd stop
Останавливается xinetd:                                    [  OK  ]
#
```

Запуск сервиса xinetd.

```
# /etc/init.d/xinetd start
Запускается xinetd:                                        [  OK  ]
#
```

В семействе дистрибутивов RedHat можно использовать специальную программу *service*, которой в качестве аргумента командной строки передаётся имя стартового скрипта и параметры скрипта.

Все тоже самое, но при помощи программы *service*.

```
# service xinetd status
xinetd (pid 6986) выполняется...
# service xinetd stop
Останавливается xinetd:                                    [  OK  ]
# service xinetd start
Запускается xinetd:                                        [  OK  ]
#
```

НАСТРОЙКА СЕТИ.

ИНТЕРФЕЙСЫ.

В дистрибутиве CentOS используется система инициализации System V, поэтому различные настройки необходимо делать при помощи конфигурационных файлов стартовых скриптов, расположенных в директории */etc/sysconfig/network-scripts*.

Сначала настроим сетевые интерфейсы. Напомню, что на сервере у нас два интерфейса:

- *eth0* — подключен к внешнему серверу и через него мы будем выходить в интернет.
- *eth1* — к нему подключена рабочая станция ALT Linux.

Переходим в директорию */etc/sysconfig/network-scripts*. Именно в ней находятся файлы, при помощи которых мы настроим сетевые интерфейсы. Имена интересующих нас файлов начинаются с *igcfg-*, а заканчиваются именем интерфейса. В нашем случае, после установки системы там буду два файла: *ifcfg-eth0* и *ifcfg-eth1*.

Как и в любом другом конфигурационном файле стартовых скриптов, в нем содержаться переменные, значения которых мы должны поменять.

- *DEVICE* — имя сетевого интерфейса.
- *BOOTPROTO* — каким образом будет получен IP адрес при старте компьютера. *Static* — будем определять явно, *dhcp* — получим от dhcp сервера.
- *HWADDR* — MAC адрес интерфейса.
- *IPADDR* — IP адрес интерфейса. В случае *dhcp* эту переменную определять не надо.
- *NETMASK* — маска подсети. В случае *dhcp* эту переменную определять не надо.
- *BROADCAST* — широковещательный адрес. В случае *dhcp* эту переменную определять не надо.
- *NETWORK* — адрес сети.
- *ONBOOT* — *yes* — включать интерфейс при старте компьютера, *no* — не включать при старте.

Отредактируем первый файл, *ifcfg-eth0*.

```
# vi ifcfg-eth0
```

Содержимое файла должно выглядеть следующим образом:

```
DEVICE=eth0
BOOTPROTO=static
HWADDR=00:0C:29:15:FA:D8
IPADDR=192.168.56.5
NETMASK=255.255.255.0
BROADCAST=192.168.56.255
```

```
NETWORK=192.168.56.0
ONBOOT=yes
```

Само собой разумеется, что вы подставите свои атрибуты, которые вам даст преподаватель.

Теперь изменим содержимое файла *ifcfg-eth1*. Его содержимое приведено ниже.

```
DEVICE=eth1
BOOTPROTO=static
HWADDR=00:0C:29:15:FA:E2
IPADDR=172.16.14.5
NETMASK=255.255.255.0
BROADCAST=172.16.14.255
NETWORK=172.16.14.0
ONBOOT=yes
```

МАРШРУТ ПО УМОЛЧАНИЮ.

После определения параметров сетевых интерфейсов следует позаботиться о маршруте по умолчанию. *Его значение вы должны получить у преподавателя*. Перейдем в директорию */etc/sysconfig* и отредактируем файл *network*. В конце файла добавим только одну строку:

```
GATEWAY=192.168.56.1
```

Параметр *GATEWAY* должен быть только один! Если у вас несколько таких параметров, обязательно удалите лишние.

ОТКЛЮЧЕНИЕ ZEROCONF.

Поскольку мы не пользуемся автоматической настройкой сетевых интерфейсов при помощи DHCP. И не собираемся использовать конфигурацию а ля Microsoft, в случае отсутствия DHCP сервера. Мы отключим zeroconf.

В конце файла */etc/sysconfig/network* допишем ещё одну строку:

```
NOZEROCONF=yes
```

Теперь при просмотре списка сетей у нас не будет вылезать странная сеть *169.254.0.0*.

ИМЕНА ИНТЕРФЕЙСОВ.

Очень важно правильно описать преобразование имени вашего компьютера в локальном файле */etc/hosts*. Программа установки ничего в нем не написала, обязательно допишите интерфейсы вашей машины в этот файл.

Формат файла очень простой, в одной строке мы описываем один сетевой интерфейс. Сначала записываем IP адрес интерфейса, потом полные и краткие имена машины. Обязательно добавляйте краткие имена машины, без них некоторые сервисы не смогут нормально запускаться и потребуют дополнительной конфигурации.

Отредактируем файл */etc/hosts*. Добавим в него две строки.

```
192.168.56.5 server.st1.kryukov.biz server
172.16.14.5 server.st1.kryukov.biz server
```

Перезапустим сеть.

```
# service network restart
```

Теперь проверим, как настроились наши интерфейсы.

```
# ifconfig
```

Для получения списка можно воспользоваться программой *ip*.

```
# ip addr show
```

В списке должны присутствовать интерфейсы: *lo*, *eth0* и *eth1*.

Посмотрим таблицу маршрутизации.

```
# route -n
```

Или так:

```
# ip route show
```

В этом списке должны быть маршруты к сетям: *172.16.2.0*, *172.16.184.0* и маршрут по умолчанию *0.0.0.0*. У вас, естественно, должны показываться сети и IP адреса, которые вам дал преподаватель.

КЛИЕНТ DNS.

Последней настройкой связанной с сетью, является определение DNS серверов, услугами которых мы будем пользоваться. Конфигурационный файл клиента DNS */etc/resolv.conf*. При помощи параметра *nameserver* мы определяем IP адрес DNS сервера. Если у нас есть несколько DNS серверов, нам потребуется определить несколько параметров *nameserver*, но не более 3-х.

Настроим клиент DNS. Отредактируем файл */etc/resolv.conf*.

```
# vi /etc/resolv.conf
```

Содержимое файла приведено ниже.

```
search st1.kryukov.biz
nameserver 192.168.56.1
```

Параметр *search* — это не обязательный параметр и его можно не писать.

Теперь проверим работоспособность сети.

```
# ping -c4 mail.ru
```

Если машина пингуется, значит сеть работает. Внутренняя сеть, к которой подключен клиент, тоже должна работать. Пропингуйте клиентскую машину по её IP адресу.

Если машина mail.ru не пингуется, значит вы ошиблись при настройке, либо преподаватель забыл дать вам доступ в Интернет.

ОБНОВЛЕНИЕ СИСТЕМЫ.

Для обновления системы в CentOS используется программа *yum*. Конечно, вы можете сами качать пакеты обновлений и устанавливать их руками при помощи *rpm*, но это жутко неудобно. Поэтому будем пользоваться *yum* для установки, удаления пакетов и для обновления системы. Программа сама будет разбираться с зависимостями между пакетами и разрешать их.

К сожалению, в стандартной поставке дистрибутива идет достаточно небольшое количество пакетов (программ). По этому, нам придется пользоваться сторонними репрозитариями (наборами пакетов), благо они очень легко подключаются к *yum*.

YUM.

Основной конфигурационный файл программы */etc/yum.conf* мы трогать не будем, как и остальные его конфигурационные файлы. Просто посмотрим, какие файлы и директории могут использоваться в работе программой.

- */etc/yum.conf* — основной конфигурационный файл.
- */etc/yum/yum-updatesd.conf* — конфигурационный файл демона автоматического обновления *yum-updatesd*. Демон, прежде всего, необходим для уведомления об обновлениях. Сообщения передаются программам, работающим в графической оболочке X Window и для сервера особой необходимости в этой программе нет. Тем более что в конфигурационном файле запрещено автоматическое обновление системы.
- */etc/yum.repos.d* — директория, в которой находятся файлы описывающие доступные репрозитарии. Добавление новых репрозитариев сводится к добавлению новых файлов в этой директории.

Основные команды программы *yum* можно узнать запустив её в командной строке без каких либо параметров.

Расскажу только про самые используемые команды программы.

- *install* — позволяет установить пакет с определенным именем. В имени пакета можно использовать шаблоны.
- *remove* — позволяет удалить пакет с указанным именем.
- *list* — позволяет получить список всех пакетов установленных репрозиториев. В списке будет указано: установлен или нет пакет, в каком репрозитарии он находится. После подключения всех дополнительных репрозиториев я бы рекомендовал сохранить список файлов в отдельном файле (*yum list > list_of_files*), что бы потом каждый раз не ждать пока сработает программа.
- *check-update* — проверяет наличие обновлений.
- *update* — обновляет систему.
- *clean* — очищает различные локальные хранилища программы. Например, очень полезно использовать *yum clean packages*, для удаления скачанных пакетов (*не путайте с удалением пакета из системы*). По умолчанию, после установки пакета, файл самого пакета не удаляется.

РЕПРОЗИТОРИЙ DAG.

Dag (http://dag.wieers.com) — это пожалуй самый известный репрозиторий, использующийся различными дистрибутивами, производными от RedHat Linux. Подключить его очень просто. На сайте проекта найдите файл с пакетом для своего дистрибутива и установите его. В пакете находится файл, который будет описывать этот репрозитарий и *yum* сможет им пользоваться.

Для CentOS нет специального файла, но мы то с вами знаем, чей это клон :). Поэтому будем пользоваться файлом, предназначенным для RedHat Enterprise Linux 5. Список поддерживаемых OS можно посмотреть тут: http://dag.wieers.com/rpm/FAQ.php#B .

Установим пакет, который содержит конфигурацию для *yum* и ключ этого репозитория.

```
# rpm -Uhv  http://apt.sw.be/redhat/el5/en/i386/rpmforge/RPMS/rpmforge-release-0.3.6-1.el5.rf.i386.rpm
```

Путь к устанавливаемому пакету пишется одной строкой.

Посмотрите содержимое директории */etc/yum.repos.d*, в ней вы увидите два файла, относящихся к dag:

- *mirrors-rpmforge* — список зеркал репозитория.

- *rpmforge.repo* — описание самого репрозитория.

После установки обновим систему до текущего состояния.

```
# yum update
```

На предложение обновления системы, отвечайте *y*.

Операция по обновлению системы занимает много времени.

Перегрузите систему.

Вы должны понимать, что dag — это не стандартный репрозиторий. Создатели дистрибутива не несут ответственности за его пакеты. Иногда бывает, что пакеты из dag не корректно собраны и могут возникнуть проблемы при обновлении. Создатель репрозитория старается быстро исправлять возникшие ошибки и через некоторое время проблемы устраняются.

НАСТРОЙКА SSH.

Сначала может показаться, что настраивать ssh не надо, все и так работает хорошо. Вообще-то это действительно так, но есть маленькие нюансы, которые следует исправить.

КРАТКОЕ ОПИСАНИЕ.

Для начала, что такое ssh? Это набор программ, которые позволяют безопасно подключаться к удаленному компьютеру, копировать файлы, создавать защищенные туннели.

Основной проблемой программ telnet, rsh и rcp, которые использовали раньше, заключается в том, что они ничего не шифровала при передаче данных. Злоумышленники могли легко получить логины и пароли пользователей.

Ssh, в отличии от telnet, сначала создает зашифрованный канал и только потом начинает передавать данные. В программе можно применять различные алгоритмы шифрования и способы аутентификации.

В Linux обычно используется бесплатная версия программы, реализующей ssh — openssh.

ЗАПРЕТ ВХОДА СУПЕРПОЛЬЗОВАТЕЛЯ.

В первую очередь необходимо запретить удаленное хождение суперпользователя. Даже если вы используете очень сложный пароль, китайцев в Интернет все равно больше. Существует вероятность, что одному из них повезет, и он совершенно случайно наберет абракадабру, соответствующую вашему паролю.

Для начала заведем пользователя, с правами которого мы будем ходить на сервер.

```
# useradd student
# passwd student
```

При помощи программы putty (я подозреваю, что вы пока еще работаете из Windows), подключитесь к вашей виртуальной машине по сети. *Вспомните структуру нашей сети. Вы подключаетесь на 22 порт роутера, который перекидывает все пакеты на ваш сервер.*

На приглашение *login as* введите имя пользователя *student*. На приглашение *password*, пароль пользователя *student*. Пароль при вводе никак не отображается. Если все правильно, вы получаете приглашение командной строки.

login as: student

student@84.19.178.61's password:

[student@server ~]$

Воспользуемся программой *su* для смены пользователя. Получим права пользователя root.

```
[student@server ~]$ su -
Пароль:
[root@server ~]# vim /etc/ssh/sshd_config
```

Если у вас разрешено использование первого протокола ssh, это необходимо запретить. Первая версия протокола была уже успешно взломана. Где то в начале файла ищите параметр *Protocol*. Если его значение равно *2,1* , удалите *1*. То есть, этот параметр должен выглядеть следующим образом:

```
Protocol 2
```

Так же запрещаем суперпользователю вход в систему по ssh, находим параметр *PermitRootLogin* и определяем его явным образом:

```
PermitRootLogin no
```

Я так же рекомендовал бы явно указать список пользователей, которым можно входить в систему по ssh. Это можно сделать при помощи параметра *AllowUsers*. Например, вот так:

```
AllowUsers student admin* web*
```

В данном примере вход разрешен пользователю student, и всем пользователям, чьи логины начинаются с admin и web.

Сохраним файл и перезапустим демон ssh.

```
# service sshd restart
```

Пока не закрывайте текущий сеанс, если вы ошибётесь, у вас будет возможность исправить ошибки. Попытайтесь снова войти в систему как пользователь root.

```
login as: root
root@84.19.179.61's password:
Access denied
```

Не получилось. А теперь пользователь student:

```
login as: student
student@84.19.179.61's password:
Last login: Thu Mar 13 18:39:21 2008 from 172.16.184.1
[student@server ~]$
```

Получилось то, чего мы добивались. Суперпользователь не может зайти в систему, а пользователь student может.

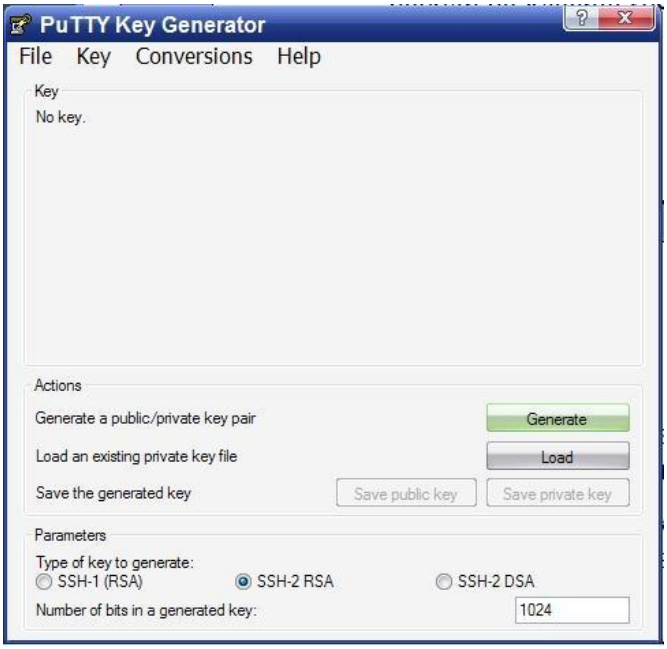
ИСПОЛЬЗОВАНИЕ КЛЮЧЕЙ.

Теперь займемся аутентификацией по ключам. Почему предпочтительно использовать ключи? Это очень просто объяснить — в случае использования ключей по сети не передаются пароли. Конечно, ssh сначала шифрует сеанс связи и только потом... Но, вы набираете пароли на клавиатуре, не дай бог на ваш Windows компьютер будет поставлен kyelogger и злоумышленники получают пароль к вашему серверу. При использовании ключей, максимум что получит злоумышленник — это пароль от ключа. Конечно, можно своровать и ключ, но это уже два действия: узнать пароль и своровать ключ.

Аутентификация по ключам делается очень просто. На клиентской машине необходимо сгенерировать приватный и публичный ключи. Публичный ключ передается на сервер и заносится в специальный файл.

Для подключения к серверу с Windows машин, я бы рекомендовал установить две программы: *putty* и *winscp*. Со второй программой поставляется генератор ключей *PuTTYgen* и программа агент *pageant*.

Запустим *PuTTYgen*:



Используйте ключи RSA (*SSH-2 RSA*) размером не менее 1024 бита. Нажимаете кнопку *Generate* и генерируете пару ключей для вашего компьютера. При сохранении приватного ключа не забудьте задать пароль на него. Программу пока не закрывайте.

На сервере, в домашней директории пользователя создаем директорию *.ssh*, в которой создаем файл *authorized_keys*.

```
$ mkdir ~/.ssh
$ touch ~/.ssh/authorized_keys
```

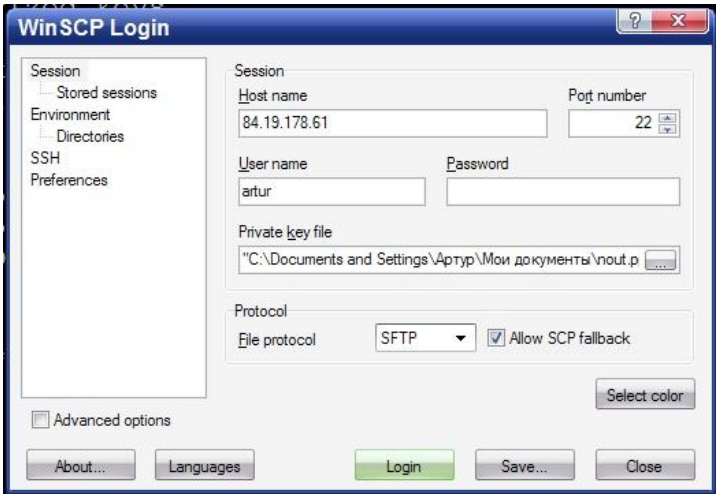
Открываем этот файл на редактирование в любимом редакторе и копируем через клипбоард (*в putty данные в окно программы копируются при нажатии на правую кнопку мыши*) ключик из окна программы *PuTTYgen*. Ключ должен получится в виде одной строки. Например, такой:

```
ssh-rsa AABA.........JFk1CO== rsa-key-20080113
```

В приведенном примере я очень сильно сократил ключик, заменив его большую часть точками.

Устанавливаем правильные права доступа на директорию и файл с ключами:

```
chmod go-w $HOME $HOME/.ssh $HOME/.ssh/authorized_keys
```



Теперь попробуйте подключится к серверу программой *winscp* при помощи ключа. Программе при запуске вместо пароля укажите файл приватного ключа.

Программе *winscp* можно явно указывать файл ключа. Но для того, что бы этот файл мог использовать *putty*, придется запустить еще одну программу — *pageant*. После запуска, она появляется в системном трее. Нажимаем на иконке правую кнопку мышки, в меню выбираем пункт *AddKey* и указываем файл с приватным ключом. Теперь достаточно запустить программу *putty* и она сама возьмет этот ключик у программы. Причем *putty* не будет спрашивать пароль для ключа. Т.е. пароль на ключ мы вводим только один раз, при запуске *pageant*. Очень удобно.

```
login as: student
Authenticating with public key "rsa-key-20080223" from agent
Last login: Thu Mar 13 18:50:37 2008 from 88.151.128.14
[student@server ~]$
```

Кстати, если вы используете *pageant*, программа *winscp* тоже автоматически обращается к нему и ей не надо явно указывать файл ключа и вводить пароль.

Если вы хотите заходить с другого компьютера, достаточно скопировать на него приватный ключ.

В случае Linux клиента, ключ генерируется при помощи программы *ssh-kyegen*. Файлы публичного и приватного ключей будут находиться в директории *~/.ssh*. Дальше, все как и в случае Windows клиента. Скопируйте публичный ключ на сервер и поместите его в файл *authorized_keys*.

Вместо программы *pageant* используйте *ssh-add*, которая спросит вас пароль на ключ. И в дельнейшем вам не придется каждый раз вводить пароль при использовании программы *scp* и *ssh*.

В следующей версии этого документа я постараюсь подробно описать использование программ *ssh* и *scp*.