

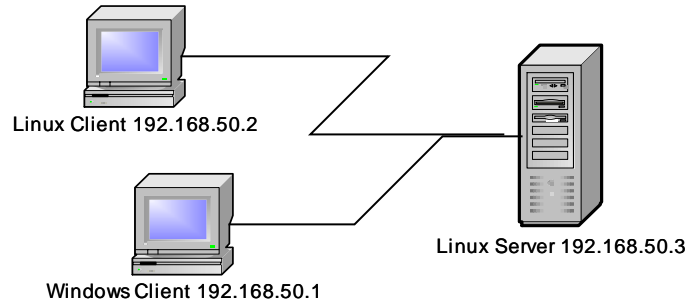
**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ОБРАЗОВАНИЯ  
ДИМИТРОВГРАДСКИЙ ИНСТИТУТ ТЕХНОЛОГИЙ,  
УПРАВЛЕНИЯ И ДИЗАЙНА**

**Лабораторная работа №6  
по курсу: "Методы и средства защиты информации"  
"Использование SSH в ОС Linux"**

**Выполнил студент гр. ВТ-41:  
Потеренко А.Г.  
Проверил преподаватель:  
Петлинский В.П.**

**Димитровград 2006г.**

## 1. Соединение компьютеров



Запустим сетевые сервисы на машине 192.168.50.3 (DBADOMAIN) через скрипт инициализации /etc/init.d/network:

```
[root@DBADOMAIN ~]# /etc/init.d/network start
Устанавливаются параметры сети:           [ OK ]
Активируется интерфейс loopback:           [ OK ]
Активируется интерфейс eth0:                [ OK ]

[root@DBADOMAIN ~]# /etc/init.d/network status
Настроенные устройства:
lo eth0
Активные в настоящее время устройства:
lo eth0
```

## 2. Общие сведения о системах шифрования и ssh

Основное отличие SSH от большинства протоколов удаленной регистрации заключается в том, что SSH обеспечивает шифрование передаваемых данных. Кроме того, данный протокол поддерживает перенаправление, или **туннелирование**, сетевых портов между клиентом и сервером. Это значит, что посредством SSH-соединения могут передаваться пакеты других протоколов.

В настоящее время существуют два SSH протокола, SSH2 и SSH1, первый является усовершенствованием SSH1. SSH2, помимо двойного шифрованного обмена ключами RSA, поддерживает и другие методы.

### 2.1. Протокол SSH версия 1

У каждого узла есть свой RSA-ключ (обычно 1024 бит), который используется для идентификации узла. Этот ключ еще называется **открытым**. Дополнительно, при запуске демона, генерируется еще один RSA-ключ — **ключ сервера** (обычно 768 бит). Этот ключ создается заново каждый час и никогда не сохраняется на диске.

Каждый раз при установке соединения с клиентом демон отправляет ему в ответ свой открытый ключ и ключ сервера. Клиент сравнивает полученный открытый ключ со своей базой данных, чтобы проверить, не изменился ли он. Затем клиент случайным образом генерирует 256-разрядное число и кодирует его, используя одновременно два ключа — открытый ключ и ключ сервера.

Обе стороны используют этот случайный номер как ключ сессии, который используется для кодирования всех передаваемых во время сессии данных. Затем клиент пытается аутентифицировать себя, используя .rhosts-аутентификацию, аутентификацию RSA или же аутентификацию с использованием пароля.

Обычно .rhosts-аутентификация небезопасна и поэтому она отключена.

### 2.2. Протокол SSH версия 2

Версия 2 работает аналогично: каждый узел имеет определенный DSA-ключ, который используется для идентификации узла. Однако при запуске демона ключ сервера не генерируется. Безопасность соединения обеспечивается благодаря соглашению Диффи-Хелмана (Diffie-Hellman key agreement).

Сессия может кодироваться следующими методами: 128-разрядный AES, Blowfish, 3DES, CAST128, Arcfour, 192-разрядный AES или 256-разрядный AES.

## 3. Запуск демона SSH на машине 192.168.50.3

Демон sshd запускается через скрипт инициализации в директории /etc/init.d - /etc/init.d/sshd.

```
[root@DBADOMAIN ~]# /etc/init.d/sshd start
Запускается sshd:                               [ OK ]
[root@DBADOMAIN ~]# /etc/init.d/sshd status
sshd (pid 5012 4474 2096) выполняется...
```

Основные ключи запуска:

```
sshd [-4Ddeigt] [-b bits] [-f config_file] [-g login_grace_time]
      [-h host_key_file] [-k key_gen_time] [-o option] [-p port] [-u len]
```

- D : При использовании этой опции демон не будет переходить в фоновый режим
- 4 : Разрешается использовать IP-адреса только в формате IPv4
- 6 : Разрешается использовать IP-адреса только в формате IPv6
- b : Определяет число битов для ключа сервера (по умолчанию 768). Данную опцию можно использовать, только если вы используете протокол SSH версии 1
- d : Режим отладки (DEBUG). В этом режиме сервер не переходит в фоновый режим и подробно протоколирует свои действия в системном журнале. Использование данной опции особенно полезно при изучении работы сервера
- f : Задаёт альтернативный файл конфигурации. По умолчанию используется /etc/ssh/sshd\_config
- g : Предоставляет клиенту, не прошедшему аутентификацию, дополнительное время, чтобы аутентифицировать себя. По умолчанию время равно 600 секундам. Если за это время клиент не смог аутентифицировать себя, соединение будет прекращено. Значение 0 интерпретируется как бесконечное ожидание
- h : Задаёт альтернативный файл открытого ключа (ключ узла). По умолчанию используется файл /etc/ssh/ssh\_host\_key. Эта опция может понадобиться, чтобы sshd мог выполняться не только от имени суперпользователя root. Кроме этого, частым использованием этой опции является запуск sshd из сценариев, задающих различные настройки в зависимости от времени суток. Например, в дневное (рабочее) время устанавливаются одни опции, а в вечернее (нерабочее) время – другие
- k : Задаёт время, спустя которое ключ сервера будет создан заново. По умолчанию время составляет 3600 секунд (1 час). Данную опцию можно использовать, только если вы используете протокол SSH версии 1
- p : Указывает альтернативный порт, который демон sshd будет прослушивать. По умолчанию используется порт 22
- q : «Тихий режим». В данном режиме протоколирование сессии производиться не будет. Обычно протоколируется начало аутентификации, результат аутентификации и время окончания сессии
- t : Тестовый режим. Данный режим применяется для проверки корректности файла конфигурации
- i : Используется, если нужно запускать sshd через суперсервер xinetd (inetd). Обычно демон sshd не запускается суперсервером xinetd (inetd), а запускается при загрузке системы, потому что демону sshd требуется некоторое время (10 секунд) для генерирования ключа сервера, прежде чем он сможет ответить на запросы клиентов

Основные файлы:

```
[root@DBADOMAIN ~]# ls /etc/ssh
moduli          ssh_host_dsa_key      ssh_host_key.pub
ssh_config      ssh_host_dsa_key.pub  ssh_host_rsa_key
sshd_config     ssh_host_key          ssh_host_rsa_key.pub
```

## • конфигурационный файл сервера /etc/ssh/sshd\_config;

- Опция Port определяет номер порта, на котором сервер sshd ожидает запросов клиентов.
- Опция Protocol определяет порядок версий протоколов, используемых при установке соединения.
- Опция ListenAddress определяет IP-адрес сетевого интерфейса, на котором sshd ожидает запросы на подключение.
- Опции HostKey /etc/ssh/ssh\_host\_key, HostKey /etc/ssh/ssh\_host\_dsa\_key и HostKey /etc/ssh/ssh\_host\_rsa\_key определяют местоположение файлов с ключами.
- Опция ServerKeyBits определяет количество битов, отведенных под ключ сервера, и используется при генерировании ключей.
- Опция LoginGraceTime определяет время в секундах, через которое сервер разрывает соединение в случае неудачной попытки регистрации удаленного пользователя. Эта настройка затрудняет реализацию DoS-атак (Denial of Service), основанных на установлении соединений пользователями, не имеющих доступа к аутентификационной информации.
- Опция KeyRegenerationInterval определяет продолжительность интервала времени в секундах, по окончании которого сервер автоматически генерирует новые ключи. Это настройка повышает безопасность системы за счет затруднения расшифровки третьими лицами трафика, генерируемого соединениями легитимных пользователей.
- Опция PermitRootLogin запрещает регистрацию пользователя root, используя ssh. В случае необходимости выполнения команд на удаленной системе от имени пользователя root безопаснее зарегистрироваться в системе в качестве обычного пользователя и затем повысить свои права доступа с помощью команд su или sudo.
- Опции IgnoreRhosts, IgnoreUserKnownHosts, StrictModes, RhostsAuthentication и RhostsRSAAuthentication повышают безопасность системы за счет запрета использования файлов rhosts или shosts для аутентификации пользователей.
- Опция X11Forwarding запрещает поддержку X-сервера.
- Опция PrintMotd разрешает вывод приветственного сообщения из файла /etc/motd после регистрации пользователя.

- Опция SyslogFacility определяет вывод сообщений sshd.
- Опция LogLevel определяет степень подробности вывода информации sshd.
- Опция RSAAuthentication позволяет использовать RSA аутентификацию. Она используется только протоколом SSH1. Протокол SSH2 использует DSA аутентификацию.
- Опция PasswordAuthentication повышает безопасность системы за счет запрещения использования паролей для аутентификации пользователей, предоставляя возможность для регистрации только тем удаленным пользователям, чьи открытые ключи, созданные с помощью утилиты ssh-keygen, размещены на сервере. Эти ключи так же защищены паролем. Такой предлагаемый вариант значения опции (no) запрещает регистрацию пользователей, которые узнали или подобрали пароль, но не имеют соответствующего ключа на своей системе.
- Опция PermitEmptyPasswords запрещает регистрацию на системе удаленных пользователей, для которых не определены значения паролей.
- Опция AllowUsers определяет список удаленных пользователей, которым разрешена регистрация в системе.
- Опция UsePrivilegeSeparation используется для повышения устойчивости системы к различного рода сбоям.

- конфигурационный **файл клиента** /etc/ssh/ssh\_config;

- Опция Host предназначена для наложения ограничений на имена систем, с которыми разрешено соединение.
- Опции ForwardAgent и ForwardX11 запрещают автоматическое распространение сеансов на вашу систему.
- RhostsAuthentication и RhostsRSAAuthentication запрещают устаревшие и ставшими небезопасными механизмы аутентификации пользователей.
- Опция RSAAuthentication разрешает использовать RSA аутентификацию. Этот тип аутентификации используется только протоколом SSH1. Протокол SSH2 использует DSA аутентификацию.
- Опция PasswordAuthentication повышает безопасность системы за счет запрещения использования паролей для аутентификации пользователей. Более подробно назначение этой опции рассматривалось выше при конфигурировании сервера.
- Опция FallBackToRsh запрещает в случае неудачной попытки соединения с помощью ssh-клиента автоматической попытки установки соединения с помощью rsh-клиента.
- Опция UseRsh запрещает использование небезопасных служб.
- Опция BatchMode требует безусловной проверки имени пользователя и пароля при установке соединения.
- Опция checkHostIP требует обязательной проверки соответствия IP-адреса и имени системы при установке соединения.
- Опция StrictHostKeyChecking запрещает автоматическое добавление ключей удаленных систем в файл known\_hosts.
- Опции IdentityFile ~/.ssh/identity, IdentityFile ~/.ssh/id\_dsa и IdentityFile ~/.ssh/id\_rsa позволяют определить местоположение файлов, используемых при аутентификации.
- Опция Protocol определяет порядок версий протоколов, используемых при установке соединения. В рассматриваемом примере сначала будет предпринята попытка установки соединения по протоколу версии 2 и если она окажется неудачной, будет предпринята попытка установления соединения по протоколу версии 1.
- Опция Cipher определяет алгоритм шифрования информации.
- Опция EscapeChar определяет возможность использования символа <ESC> для приостановки сеанса.

- файл для поддержки модулей PAM /etc/pam.d/sshd;

Для повышения безопасности компиляция SSH осуществлена с поддержкой аутентификации с использованием модулей PAM. Для настройки аутентификации создайте файл /etc/pam.d/sshd, содержащий следующие строки:

```
#%PAM-1.0
auth required /lib/security/pam_stack.so service=system-auth
auth required /lib/security/pam_nologin.so
account required /lib/security/pam_stack.so service=system-auth
account required /lib/security/pam_access.so
account required /lib/security/pam_time.so
password required /lib/security/pam_stack.so service=system-auth
session required /lib/security/pam_stack.so service=system-auth
session required /lib/security/pam_limits.so
```

- файл инициализации /etc/init.d/sshd.

#### Система шифрования SSH:

В SSH используется **смешанная система шифрования**: если соединение по паролю – то это симметричная система шифрования, а если используются открытые и закрытые ключи для идентификации пользователей – то это асимметричная система.

#### 4. Установление SSH соединения по паролю

```
ssh [-1246AaCfGkMNnqsTtVvXxY] [-b bind_address] [-c cipher_spec]
    [-D port] [-e escape_char] [-F configfile] [-i identity_file]
    [-L port:host:hostport] [-l login_name] [-m mac_spec] [-o option]
    [-p port] [-R port:host:hostport] [-S ctl] [user@]hostname [command]
```

-1 : Использовать только первую версию протокола SSH  
-2 : Использовать только вторую версию протокола SSH  
-4 : Разрешается использовать IP-адреса только в формате IPv4  
-6 : Разрешается использовать IP-адреса только в формате IPv6  
-A : Включает перенаправление аутентификации агента соединения  
-a : Отключает перенаправление аутентификации агента соединения  
-b : Определяет сетевой интерфейс для передачи данных на машине с множеством сетевых интерфейсов  
-c : Задаёт список шифров, разделённых запятыми в порядке предпочтения. Только для второй версии протокола SSH. Допускаются значения blowfish, twofish, arcfour, cast, des и 3des  
-D : Локальный порт соединения  
-F : Альтернативный файл конфигурации  
-f : Данная опция переводит ssh в фоновый режим после аутентификации пользователя. Рекомендуется использовать для запуска программы X11.  
-i : Задаёт нестандартный идентификационный файл (для нестандартной RSA/DSA-аутентификации)  
-l : Указывает от имени какого пользователя будет осуществляться регистрация на удалённой машине  
-p : Определяет внешний порт, к которому подключится клиент ssh (по умолчанию используется порт 22)  
-q : «Тихий режим». Будут отображаться только сообщения о фатальных ошибках. Все прочие предупреждающие сообщения в стандартный выходной поток выводиться не будут

Запустив sshd на машине, проверьте правильность настроек, попробовав зайти на неё, используя ssh клиента.

Давайте предположим, что ваша машина имеет адрес 192.168.50.3 и вы входите на неё под именем user1 с соседнего хоста 192.168.50.2. Создадим пользователя user1 на машине 192.168.50.3.



Далее вы работаете удаленно в shell по зашифрованному каналу.

#### 5. Генерирование и управление ssh ключами

Ssh предлагает и другой аутентификационный механизм, основанный на аутентификационных ключах: **метод шифрования с открытым ключом**. Каждый пользователь, желающий использовать ssh с аутентификацией с открытым ключом, должен выполнить команду **ssh-keygen**, чтобы создать ключи. Команда генерирует пары ключей (публичный и частный) и запрашивает passphrase для их защиты. Создаются два файла в каталоге `/root/.ssh/`: **id\_rsa** и **id\_rsa.pub**, пользовательские частный и публичный ключи.

Мы хотим войти с 192.168.50.2 на 192.168.50.3, используя аутентификацию ssh с открытыми ключами. Чтобы сделать это, потребуются четыре шага:

- На 192.168.50.2 генерируем ключевую пару, используя команду `ssh-keygen`, и выбираем passphrase: **qwerty**, чтобы защитить ее.

```

root@DBADOMAIN:~
Файл  Правка  Вид  Терминал  Вкладки  Справка
[root@DBADOMAIN ~]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
c0:f3:03:01:5c:bb:01:8f:17:de:b0:b4:c9:7c:38:04 root@DBADOMAIN
[root@DBADOMAIN ~]# ls /root/.ssh
id_rsa  id_rsa.pub  known_hosts
[root@DBADOMAIN ~]#

```

- Входим на 192.168.50.3, используя парольную идентификацию ssh. Переходим в каталог /root/.ssh, и создаем файл authorized\_keys2. Этот файл используется sshd, чтобы идентифицировать ключевую пару, которую нужно использовать в течение соединения.
- Находясь на 192.168.50.3, получим публичный ключ id\_rsa.pub с 192.168.50.2, и переименуем его (например в host1.pub). Переместим этот файл в /root/.ssh на сервере 192.168.50.3.
- Дописываем в файл authorized\_keys2 содержимое файла host1.pub:

```
cat host1.pub >> authorized_keys2
```

Этот файл содержит все доверительные публичные ssh ключи, помещенные в каталог /root/.ssh. Когда ssh инициировано пользователем, публичный ключ которого соответствует одной из записей файла authorized\_keys2, то стартует схема аутентификации с открытыми ключами.

На машине 192.168.50.2 выполняем команду:

```

root@DBADOMAIN:~
Файл  Правка  Вид  Терминал  Вкладки  Справка
[root@DBADOMAIN ~]# ssh -2 192.168.50.3
Enter passphrase for key '/root/.ssh/id_rsa':
Last login: Fri Nov 17 15:47:18 2006
[root@DBADOMAIN ~]#

```

## 6. Краткий обзор утилит sftp и scp

Давайте сосредоточим внимание на sftp и scp. Первый (Secure File Transfer) - ftp-подобный клиент, который может быть использован для передачи файлов по сети. Он не использует FTP демонов (ftpd или wu-ftpd) для соединения, что позволяет существенно повысить уровень защиты. Анализ систем в Интернет, показывает, что 80% атак было направлено на wu-ftpd. Использование sftp позволяет отключить потенциально опасный wu-ftpd.

Второй (Secure Copy) используется для защищенного копирования файлов по сети. Это - замена небезопасной команды rcp.

Sftp и scp не требуют никакого специально выделенного сервера, так как эти две программы соединяются с ssh сервером. Чтобы использовать sftp и scp в конфигурационный файл /etc/ssh/ssh\_config включается следующая строка:

```
Subsystem sftp /usr/libexec/openssh/sftp-server
```

Учтите, что Вы можете пользоваться sftp и scp только при соединении с хостом, где запущен sshd.

### 6.1. Использование sftp

Sftp использует ssh2 при соединении, это означает, что передаваемые файлы защищены настолько, насколько это возможно.

Есть два основных преимущества при использовании sftp вместо ftp: пароли никогда не передаются открытым текстом, что предотвращает атаки с использованием снифферов (sniffer).

Предположим, что Вам нужно соединиться с помощью `sftp` с хостом `192.168.50.3`, имея на ней аккаунт `root`. Используйте команду:

```
sftp root@192.168.50.3
```

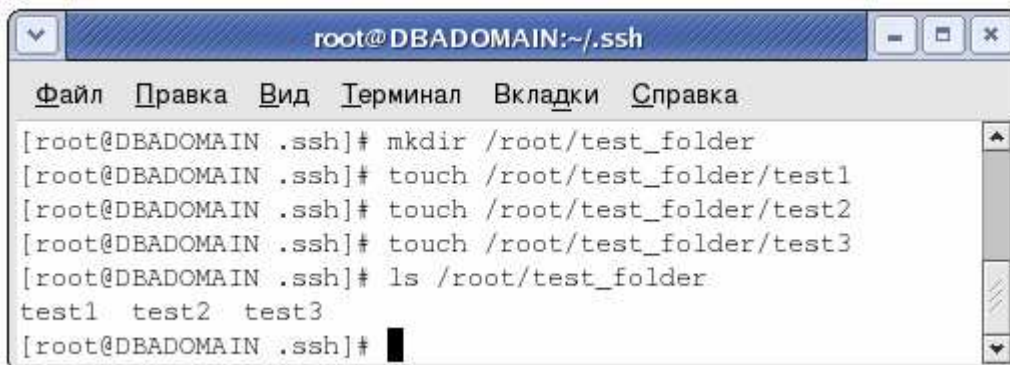
Когда `sftp` будет готов к выполнению команд, высветится приглашение `sftp>`. В справочном руководстве приведен полный перечень возможных команд, некоторые из них приведем здесь:

- `quit`: Выход из приложения.
- `cd directory`: Сменить текущий удаленный каталог.
- `lcd directory`: Сменить текущий локальный каталог.
- `ls [ -R ] [ -l ] [ file ... ]`: Перечень файлов на удаленном сервере, для каталогов - их содержание. Когда задана опция `-R`, дерево каталогов просматривается рекурсивно. (По умолчанию подкаталоги запрошенного каталога не видны). С опцией `-l` показываются также права, владельцы, размер и дата модификации. Если никакие опции не заданы, просматривается содержание `"."`. В настоящее время опции `-R` и `-l` не совместимы.
- `lls [ -R ] [ -l ] [ file ... ]`: То же, что и `ls`, но для локальных файлов.
- `get [ file ... ]`: Передает указанные файлы с удаленной точки на локальную. Каталоги рекурсивно копируются с их содержимым.
- `put [ file ... ]`: Передает указанные файлы с локальной точки на удаленную. Каталоги рекурсивно копируются с их содержимым.
- `mkdir dir (rmdir dir)`: Пытается создать (удалить) каталог `dir`.
- `help` или `?`: Выводит справку по командам `sftp`.

`sftp` поддерживает шаблоны (wildcards) в командах `ls`, `lls`, `get`, и `put`. Формат описан в справочном руководстве `sshregex`. Так как `sftp` использует крипто, это имеет свои минусы: уменьшение скорости соединения, но это не так важно по сравнению с проблемами защиты сети.

Предположим, что мы хотим получить папку с файлами `192.168.50.3:/root/test_folder` на хост `192.168.50.2` в каталог `/root`. Операции:

Хост `192.168.50.3`:

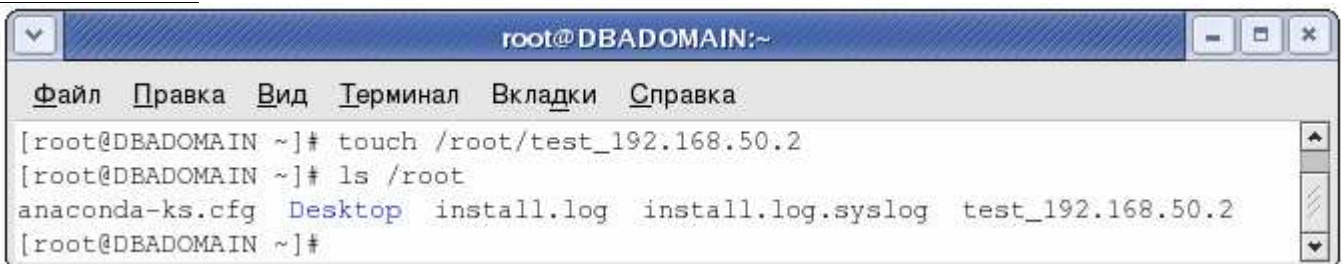


```

root@DBADOMAIN: ~/.ssh
[root@DBADOMAIN .ssh]# mkdir /root/test_folder
[root@DBADOMAIN .ssh]# touch /root/test_folder/test1
[root@DBADOMAIN .ssh]# touch /root/test_folder/test2
[root@DBADOMAIN .ssh]# touch /root/test_folder/test3
[root@DBADOMAIN .ssh]# ls /root/test_folder
test1 test2 test3
[root@DBADOMAIN .ssh]#

```

Хост `192.168.50.2`:



```

root@DBADOMAIN: ~
[root@DBADOMAIN ~]# touch /root/test_192.168.50.2
[root@DBADOMAIN ~]# ls /root
anaconda-ks.cfg Desktop install.log install.log.syslog test_192.168.50.2
[root@DBADOMAIN ~]#

```



```

root@DBADOMAIN:~
Файл  Правка  Вид  Терминал  Вкладки  Справка

[root@DBADOMAIN ~]# sftp root@192.168.50.3
Connecting to 192.168.50.3...
Enter passphrase for key '/root/.ssh/id_rsa':
sftp> ls
Desktop
a local "dynamic" application-level port forwarding.
anaconda-ks.cfg
install.log
install.log.syslog
test_folder
works by allocating a socket to listen to port on the local
sftp> lls
anaconda-ks.cfg  Desktop  install.log  install.log.syslog  test_192.168.50.2

sftp> put test_192.168.50.2
Uploading test_192.168.50.2 to /root/test_192.168.50.2
test_192.168.50.2          100%   0    0.0KB/s   00:00
sftp> ls
Desktop
a local "dynamic" application-level port forwarding.
anaconda-ks.cfg
install.log
install.log.syslog
test_192.168.50.2
test_folder
works by allocating a socket to listen to port on the local
sftp>

sftp> get test_folder/test1
Fetching /root/test_folder/test1 to test1
sftp> get test_folder/test2
Fetching /root/test_folder/test2 to test2
sftp> lls
Desktop  install.log.syslog  test1  test_192.168.50.2  test2
sftp>

```

## 6.2. Использование Scp

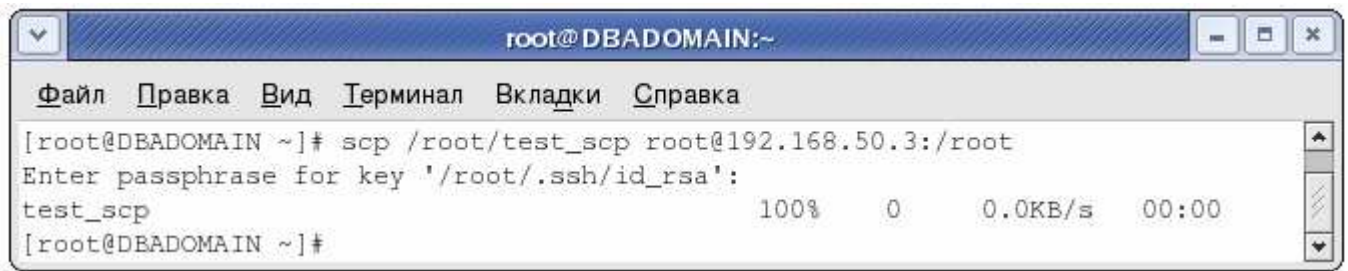
Scp (Secure Copy) применяется для надежного копирования файлов по сети. Возможно, это самый простой способ передачи файлов на удаленную машину. Предположим, что Вы хотите скопировать файл `test_scp` из каталога `/root` на Ваш аккаунт `root` в каталог `/root` на машине `192.168.50.3`.

Для применения `scp` введите в командной строке:

```
scp /root/test_scp root@192.168.50.3:/root
```

таким образом, файл `test_scp` скопирован с тем же именем.





```

root@DBADOMAIN:~
Файл  Правка  Вид  Терминал  Вкладки  Справка
[root@DBADOMAIN ~]# scp /root/test_scp root@192.168.50.3:/root
Enter passphrase for key '/root/.ssh/id_rsa':
test_scp                                100%   0    0.0KB/s   00:00
[root@DBADOMAIN ~]#

```

Команда:

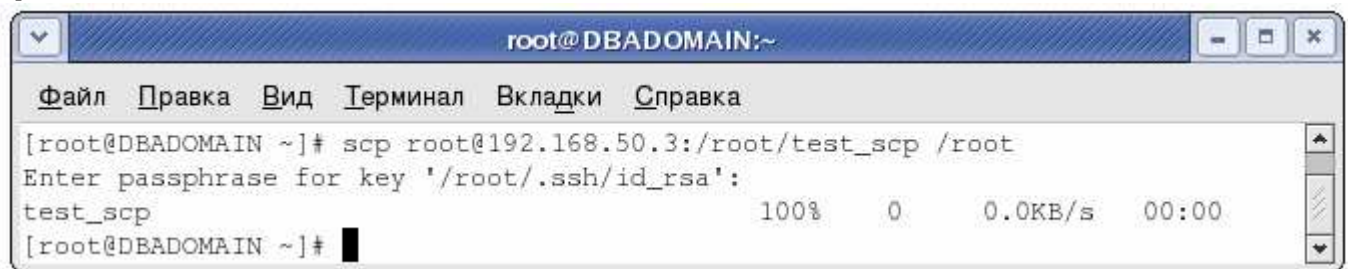
```
scp /root/* root@192.168.50.3:/root
```

копирует все файлы из каталога /root на хосте 192.168.50.2 в каталог /root на хосте 192.168.50.3.

Команда:

```
scp root@192.168.50.3:/root/test_scp /root
```

копирует файл test\_scp из каталога /root машины 192.168.50.3 на локальную машину в каталог /root.



```

root@DBADOMAIN:~
Файл  Правка  Вид  Терминал  Вкладки  Справка
[root@DBADOMAIN ~]# scp root@192.168.50.3:/root/test_scp /root
Enter passphrase for key '/root/.ssh/id_rsa':
test_scp                                100%   0    0.0KB/s   00:00
[root@DBADOMAIN ~]# █

```

Scp поддерживает много параметров и позволяет передавать файлы с одной удаленной машины на другую, как показано в следующем примере:

```
scp myname@host1:remote_dir/filename myname@host2:another_dir
```

Очевидно, используя scp, Вы должны знать структуру каталогов на удаленной машине, поэтому на практике предпочитают sftp.

## 7. SSH клиенты putty и SSH Secure Shell для Windows

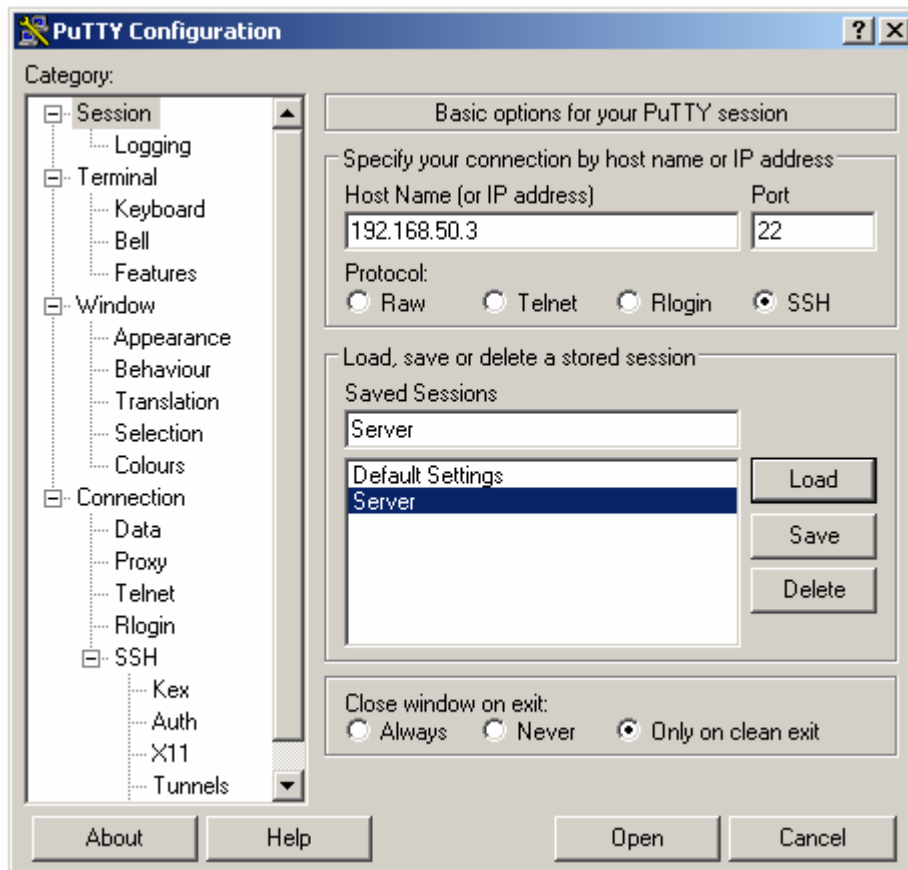
Для работы по ssh протоколу в среде Windows можно использовать программы-аналоги клиента ssh для Linux. Это пакеты утилит putty и SSH Secure Shell. Основная разница между этими программами заключается в том, что первая не требует административных прав для установки, а во второй реализован более удобный графический интерфейс.

Выполнить следующие задания:

- Установить комплекс утилит из набора putty.
- Изучить назначение и возможности основных утилит набора - **putty.exe**, **psftp.exe**, **pscp.exe**, **puttygen.exe**, **pageant.exe**, используя файл справки - putty.hlp.
- Практически освоить работу с putty.exe, используя парольную аутентификацию.
- Аналогично для программы psftp.exe.
- Аналогично но для программы pscp.exe.
- Аналогично для SSH Secure Shell, после её установки администратором.

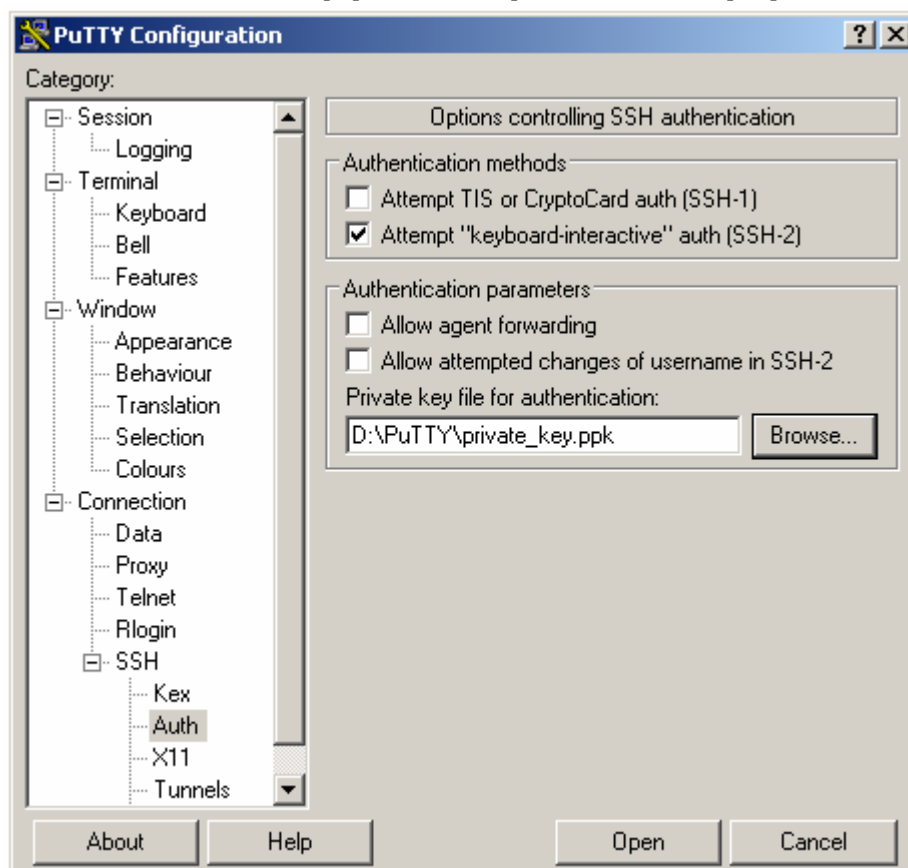
Работа putty.exe

PuTTY.exe запускается на машине 192.168.50.1.



Как видно на рисунке мы загружаем предварительно сохраненную конфигурацию Server. Сервер – 192.168.50.3. Клиент подключается на внешний порт 22. Также необходимо выбрать Window→Translation→Character Set: UTF-8 для правильной кодировки русского шрифта.

Если мы используем парольную аутентификация, то мы жмем Open и вводим пароль и логин. При использовании ключей необходимо их сгенерировать, отправить OK на сервер и указать ЗК в SSH→Auth:



### Работа pscp.exe

PuTTY безопасный клиент копирования файлов

Релиз 0.52

Использование: pscp [options] [user@]host:source target  
 pscp [options] source [source...] [user@]host:target  
 pscp [options] -ls user@host:filespec

Options:

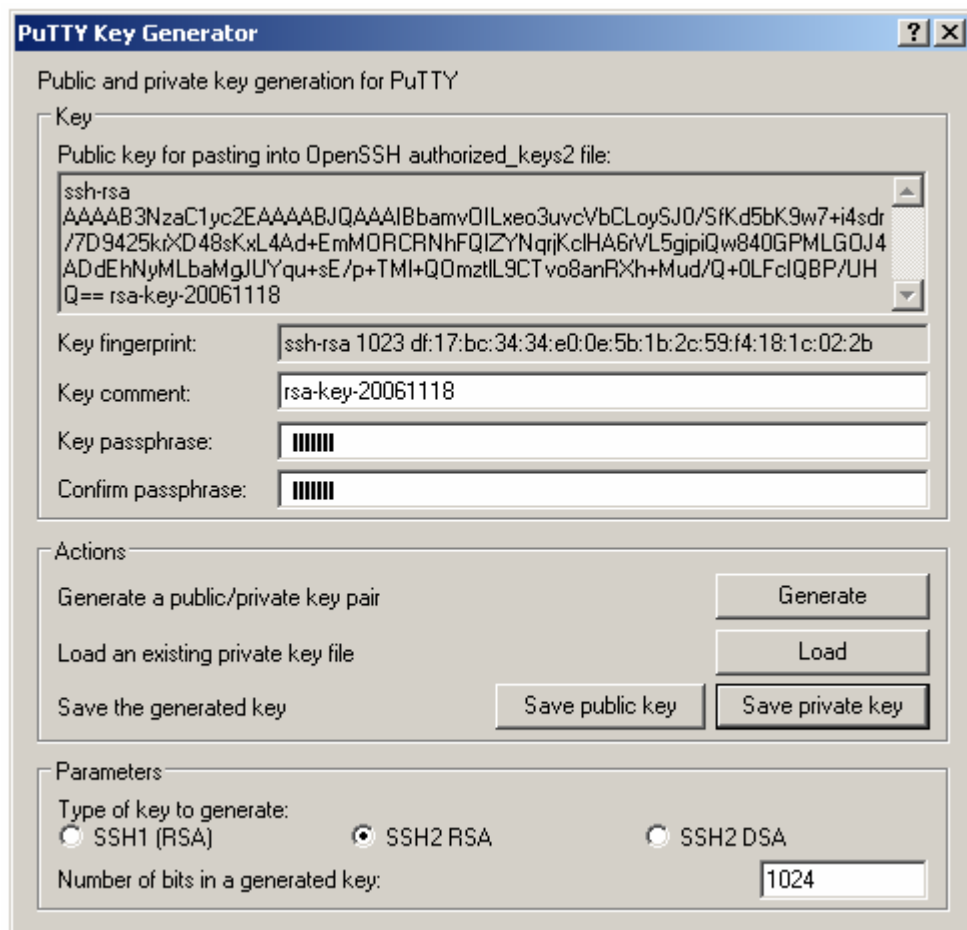
-p сохранить атрибуты файлов  
 -q без показа статистики  
 -r копировать директории рекурсивно  
 -v показывать сообщения более подробно  
 -P port внешний порт  
 -pw passw логин с указанным паролем  
 -unsafe позволить групповые символы от сервера (Опасно)

Пусть на машине 192.168.50.1 в каталоге d:\PUTTY находится файл public\_key.pk. Его необходимо переместить на машину 192.168.50.3 в каталог /root/.ssh:

```
C:\Documents and Settings\ADMIN>d:\PUTTY\pscp.exe d:\PUTTY\public_key.pk root@192.168.50.3:/root/.ssh
root@192.168.50.3's password:
public_key.pk | 0 kB | 0.3 kB/s | ETA: 00:00:00 | 100%
```

Как видно здесь парольная аутентификация пользователя root. Сначала вводим пароль, затем происходит перемещение файла.

### Работа puttygen.exe



Служит для генерирования пары ключей. При генерировании ключей необходимо двигать мышью на форме. Затем появится окно выше, в котором можно будет сохранить ключи в файлах на диск.

### Работа psftp.exe

PuTTY безопасный клиент протокола SFTP  
Релиз 0.52  
Использование: `psftp [options] user@host`

Пусть на машине 192.168.50.1 в каталоге `d:\PUTTY` находится файл `public_key.pk`. Его необходимо переместить на машину 192.168.50.3 в каталог `/root`:

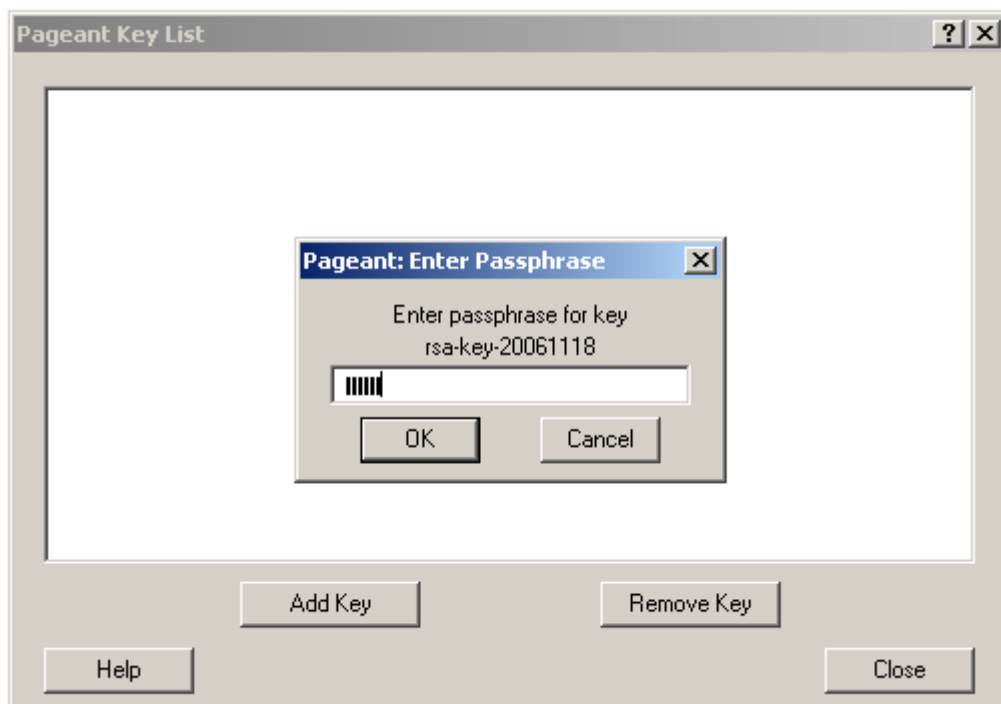
```
C:\Documents and Settings\ADMIN>d:\PUTTY\psftp.exe root@192.168.50.3
Using username "root".
root@192.168.50.3's password:
Remote working directory is /root
psftp> lcd d:\putty
New local directory is d:\putty
psftp> put public_key.pk
local:public_key.pk => remote:/root/public_key.pk
psftp>
```

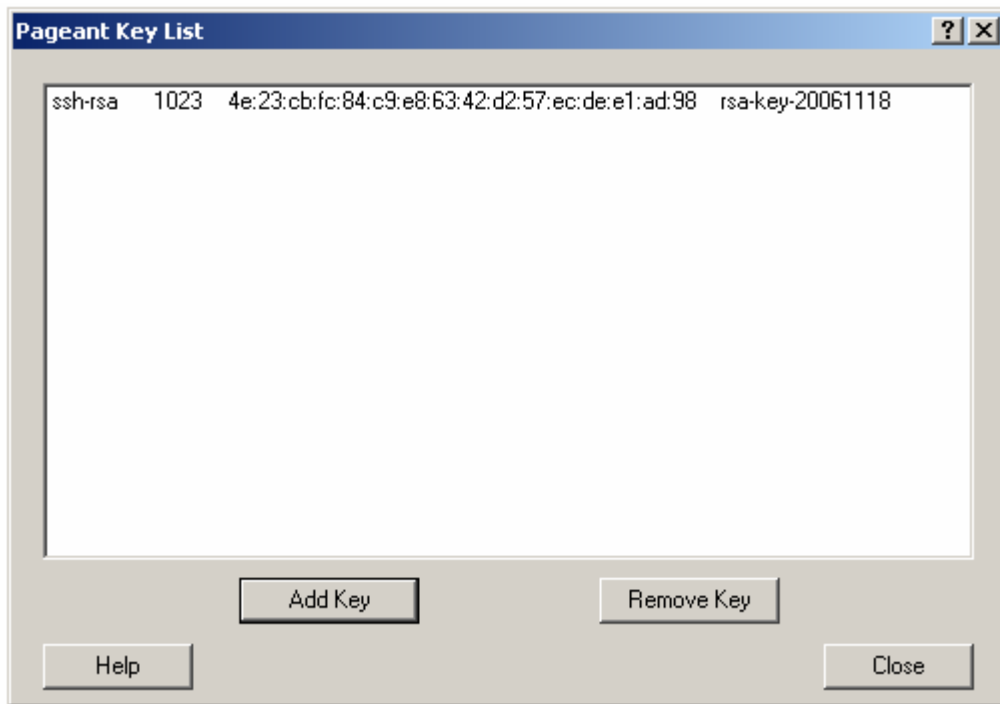
Пусть на машине 192.168.50.3 в каталоге `/root` находится файл `public_key.pk`. Его необходимо переместить на машину 192.168.50.1 в каталог `d:\PUTTY`:

```
psftp> get public_key.pk
remote:/root/public_key.pk => local:public_key.pk
psftp>
```

### Работа pageant.exe

Используется для того, чтобы не вводить `passphrase` при новом соединении в программе `putty.exe`. Ключи декодируются в первый раз и хранятся в памяти в процессе работы данной программы. Она как бы перехватывает запросы приватного ключа от клиента к серверу.





### SSH Secure Shell

