Лабораторная работа № 2.1
«Создание и управление криптографическими ключами»

Цель: получить навыки работы с утилитами по созданию криптографических ключей. Получить представление о принципе работы авторизации по ключам.

1. Теоретическая часть

SSH или Secure Shell — это зашифрованный протокол, который часто используется для взаимодействия и удаленного управления серверами. Если вы захотите что-либо сделать на удаленном сервере, скорее всего, вам придется воспользоваться SSH и работать через терминал.

В SSH существует несколько способов авторизации:

- по ip адресу клиента
- по публичному ключу
- стандартный парольный метод

Вы можете каждый раз вводить пароль пользователя или использовать более безопасный и надежный способ — ключи SSH. Как работает ssh версии 2:

При запросе клиента сервер сообщает ему, какие методы аутентификации он поддерживает (это определяется в опции PreferredAuthentications sshd.conf) и клиент по очереди пытается проверить их. По умолчанию клиент вначале пытается аутентифицироваться своим адресом, затем публичным ключом и, если ничего не сработало, передаёт пароль, введённый с клавиатуры (при этом пароль шифруется асимметрическим шифрованием). После прохождения аутентификации одним из методов из имеющихся у клиента и сервера пар ключей генерируется ключ симметрического шифрования, который,

генерируется на основании своего секретного и удалённого публичного ключей. После чего все последующие данные, передаваемые через ssh, шифруются данным ключом (обычно используется алгоритм aes с длиной ключа 128 бит). Протокол ssh версии 1 имел некоторые баги в шифровании передаваемого трафика и являлся по сути методом безопасной аутентификации, поэтому по современным меркам данный протокол считается небезопасным. Протокол версии 2 поддерживает более современные методы шифрования трафика, также вместе с данными посылаются контрольные суммы формата sha, что исключает подмену или иную модификацию передаваемого трафика(чего не было у ssh версии 1).

1.1 Идентификация по адресу клиента

При данном способе аутентификации происходит следующее:
каждый клиент и сервер имеют свои пары ключей RSA, которые называются ключи хоста. При этом существует несколько методов проверки адреса клиента.
Сервер смотрит файлы $HOME/.rhosts, $HOME/.shosts, /etc/hosts.equiv или /etc/ssh/shosts.equiv, если же сервер настроен на проверку ключей клиентов(а это нужно в соображениях безопасности, т.к. иначе злоумышленник может подменить ip клиента на свой), то он дополнительно проверяет /etc/ssh/ssh_known_hosts и $HOME/.ssh/known_hosts. Естественно, что файлы, расположенные в домашних каталогах сервера, действуют на пользователя, в чьём каталоге они размещены, а файлы, расположенные в /etc имеют глобальный эффект. Синтаксис вышеперечисленных файлов:

.rhosts - определяет адрес машины и имя пользователя, с которой данному пользователю открыт доступ (файл расположен в домашнем каталоге пользователя)

.shosts - аналогичен .rhosts, но предназначен исключительно для ssh, поэтому использовать лучше именно данный файл. Пример .shhosts:

```
user1.test.ru user1
userstend.test.ru user1
null.test.ru user1
```

/etc/hosts.equiv - также содержит пары имя машины/имя пользователя, но имеет эффект на всех пользователей

/etc/shosts.equiv - аналог hosts.equiv, но применяется только ssh, что также более предпочтительно. Пример файла /etc/shhosts.equiv

```
+ user1.test.ru user1
- server.test.ru xakep
```

Знак + означает разрешение пользователю работать с сервером с данного адреса, знак - запрещает подобное действие.

/etc/ssh/ssh_known_hosts и $HOME/.ssh/known_hosts - данные файлы содержат список адресов и соответствующих им публичных ключей. При запросе клиента сервер генерирует рандомную строку и шифрует её публичным ключом удалённого хоста. Клиент, получив данную строку, расшифровывает её своим секретным ключом (который имеется только у него) и зашифровывает полученную строку ключом сервера. Сервер получает зашифрованное сообщение, расшифровывает своим секретным ключом и сравнивает с исходной. Если строки совпали, то клиент имеет валидный секретный ключ, что даёт ему право захода на данный сервер. Но для начала клиент должен иметь правильный адрес, которому соответствует публичный

ключ на сервере в файле ssh_known_hosts. Файл состоит из 3-х полей: адрес(или адреса, разделённые запятой), публичный ключ для него одной(!) строкой и дополнительное поле комментариев(необязательно). Пример файла known_hosts:

```
user1.test.ru {SOME_VERY_LONG_PUBLIC_KEY}
```

Адрес клиента должен быть в полном формате(name.domain), иначе могут быть проблемы. Кроме этого, в адресе можно использовать шаблоны * и ?. Публичные ключи вставляются в данный файл самим администратором из генерированных клиентом ssh(identity.pub) публичных ключей. Вообще создание ssh_known_hosts - это прерогатива администратора(root).

При аутентификации по хосту лучше использовать ssh_known_hosts, т.к. этот метод достаточно безопасен, если публичные ключи клиентов были получены из доверенного источника. Другие методы аутентификации не исключают подмену адреса, и потому считаются небезопасными.

1.2 Аутентификация пользователя по его публичному ключу.

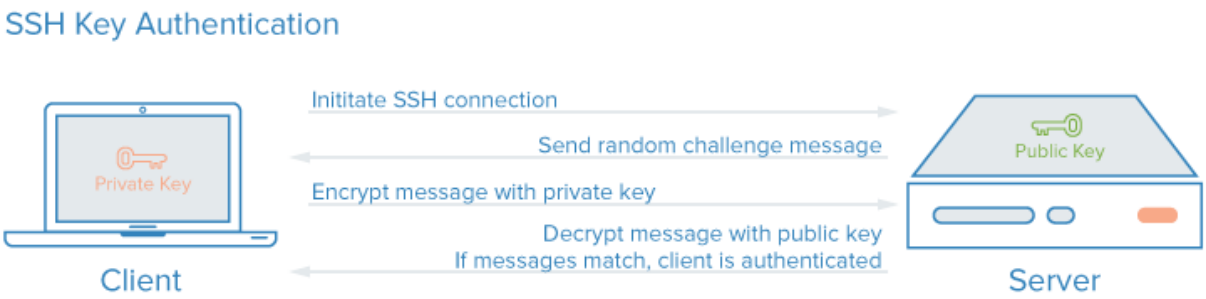
Аутентификация удалённого пользователя по ключу идентична проверке ключа хоста (с посылкой рандомной строки) за тем исключением, что проверяется не адрес клиентской машины, а ключ клиента и имя пользователя. Данному пользователю на сервере может соответствовать его публичный ключ, тогда клиент, имея секретный ключ сможет заходить на сервер без пароля.

Каждая пара ключей состоит из открытого и закрытого ключа. Секретный ключ сохраняется на стороне клиента и не должен быть доступен кому-либо

еще. Утечка ключа позволит злоумышленнику войти на сервер, если не была настроена дополнительная аутентификация по паролю.

Открытый ключ используется для шифрования сообщений, которые можно расшифровать только закрытым ключом. Это свойство и используется для аутентификации с помощью пары ключей. Открытый ключ загружается на удаленный сервер, к которому необходимо получить доступ. Его нужно добавить в специальный файл ~/.ssh/authorized_keys или в директорию ~/.ssh/<имя_ключа>

Когда клиент попытается выполнить проверку подлинности через этот ключ, сервер отправит сообщение, зашифрованное с помощью открытого ключа, если клиент сможет его расшифровать и вернуть правильный ответ — аутентификация пройдена.



1.3 Обычная парольная аутентификация.

Пароль - это секретная информация (или просто секрет), разделенная между пользователем и удаленным сервером. Пользователь помнит этот секрет, а сервер хранит либо копию секрета, либо значение, вычисленное на основе секрета. Во время аутентификации происходит сопоставление пароля, введенного пользователем, и значения, хранимого сервером. Аутентификация при помощи паролей - наиболее распространенный вид аутентификации. Но, если злоумышленник знает чужой пароль, то имеет возможность выдавать

себя за другого субъекта, и сервер не может отличить его от настоящего пользователя.

Пользователь А передает по сети на сервер свое имя и пароль. Некто, наблюдающий за средой передачи, например, пользователь С, может похитить пароль пользователя А. Как только это происходит, пользователь С может выдавать себя за пользователя А до тех пор, пока пароль не будет изменен, а это может продолжаться достаточно долгое время. Поэтому для безопасности вычислительной среды требуется регулярно менять пароли.

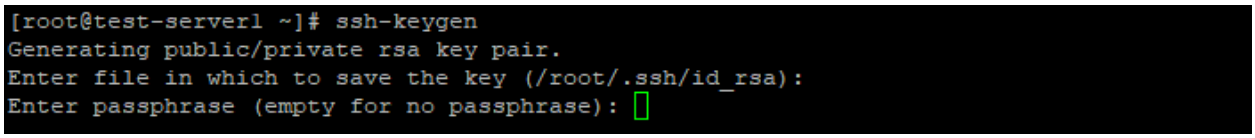
Существует несколько способов получения секретного пароля в сети. Пользователь С может использовать программу-анализатор, или сниффер. Программы-анализаторы легко доступны в Интернете, они позволяют перехватывать сетевой трафик между компьютерами одной локальной сети. Для перехвата пароля пользователю С можно даже не находиться в одном помещении с пользователем А и не иметь доступ к его компьютеру - ему достаточно лишь сетевого подключения к той же самой локальной сети. После смены пароль остается неизвестен пользователю С только до очередного запуска сниффера.

2. Практическая часть.

2.1 Создание ключей в Linux

На клиентской стороне должен быть установлен пакет ssh (openssh). На клиентском компьютере в командной строке выполните команду генерации ключей:

```
ssh-keygen
```



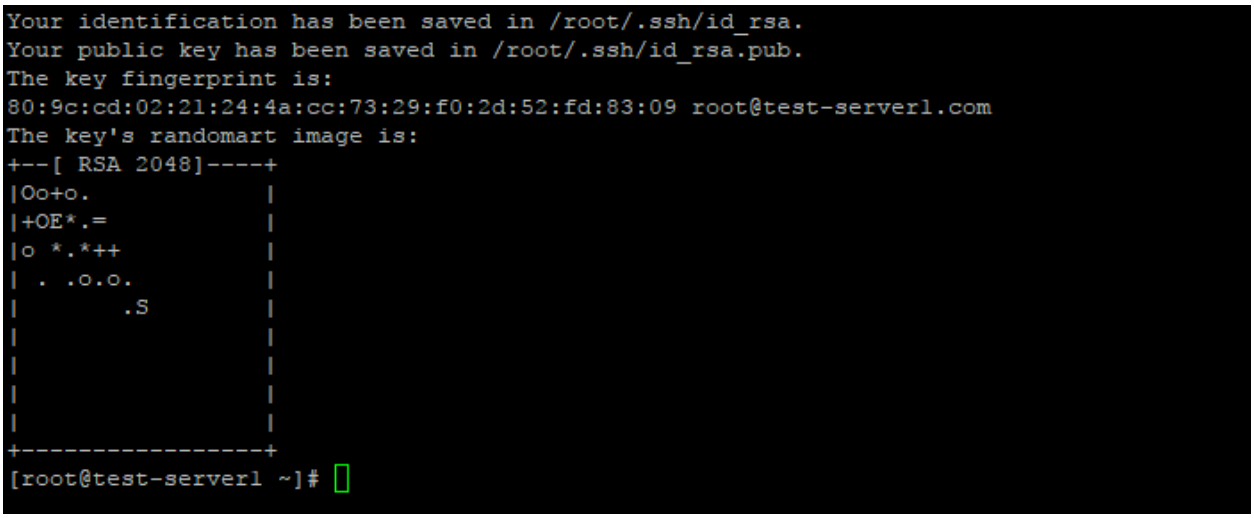
Введите путь файла, в который будут помещены ключи. Каталог по умолчанию указана в скобках, в примере /домашний_каталог/.ssh/id_rsa. Если хотите оставить расположение по умолчанию, нажмите Enter.

Пароль (passphrase) используется для ограничения доступа к закрытому ключу. Его можно не указывать, но использование дополнительного шифрования имеет только один минус — необходимость вводить пароль, и несколько преимуществ:

- Пароль никогда не попадет в сеть, он используется только на локальной машине для расшифровки ключа. Это значит, что перебор по паролю больше невозможен.
- Секретный ключ хранится в закрытом каталоге и у клиента ssh нет к нему доступа пока вы не введете пароль;
- Если злоумышленник хочет взломать аутентификацию по ключу SSH, ему понадобится доступ к вашей системе. И даже тогда ключевая фраза может стать серьезной помехой на его пути.

Если не хотите использовать секретную фразу, нажмите Enter без заполнения строки.

Успешно сгенерировав пару ключей вы увидите уведомление:



Структура ключа:

- id_rsa.pub — открытый ключ. Его копируют на сервера, куда нужно получить доступ.
- id_rsa — закрытый ключ. Его нельзя никому показывать.

Открытый ключ хранится в файле /домашний_каталог/.ssh/id_rsa.pub, закрытый — /домашний_каталог/.ssh/id_rsa

Теперь необходимо перенести публичный ключ на сервер, к которому будет производиться подключение (для данной лабораторной работы это компьютер соседней бригады), для этого:

```
echo ssh-rsa строка-публичного-ключа >> ~/.ssh/authorized_keys
```

или

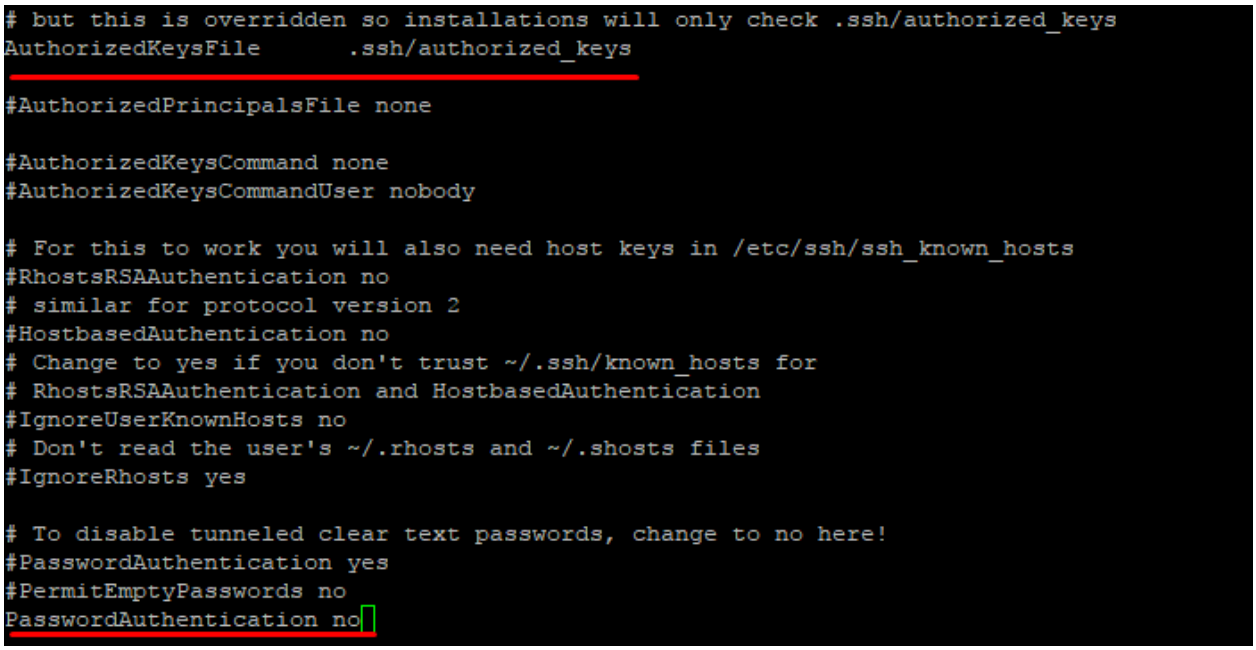
```
cat <файл публичного ключа> > ~/.ssh/ authorized_keys
```

После обозначенных команд попробуйте подключиться к серверу без использования пароля.

При успешном подключении можно отключить идентификацию по паролю. Подключитесь к серверу по SSH, используя пароль, и откройте файл sshd_config для редактирования

```
vi /etc/ssh/sshd_config
```

Убедитесь, что указан правильный путь к открытым ключам SSH, поставьте значение параметра «PasswordAuthentication no»



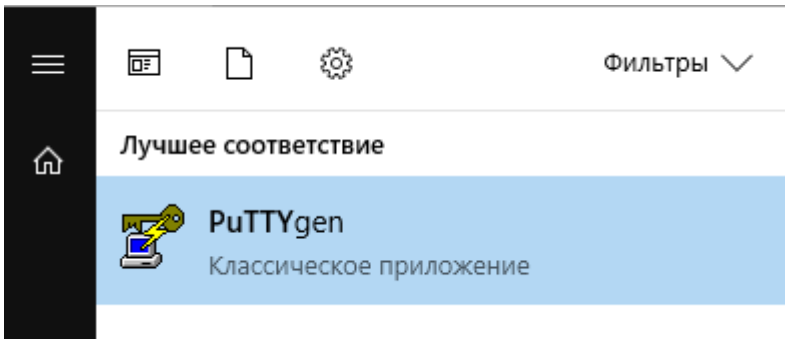
Перезапустите службу sshd

```
service sshd restart
```

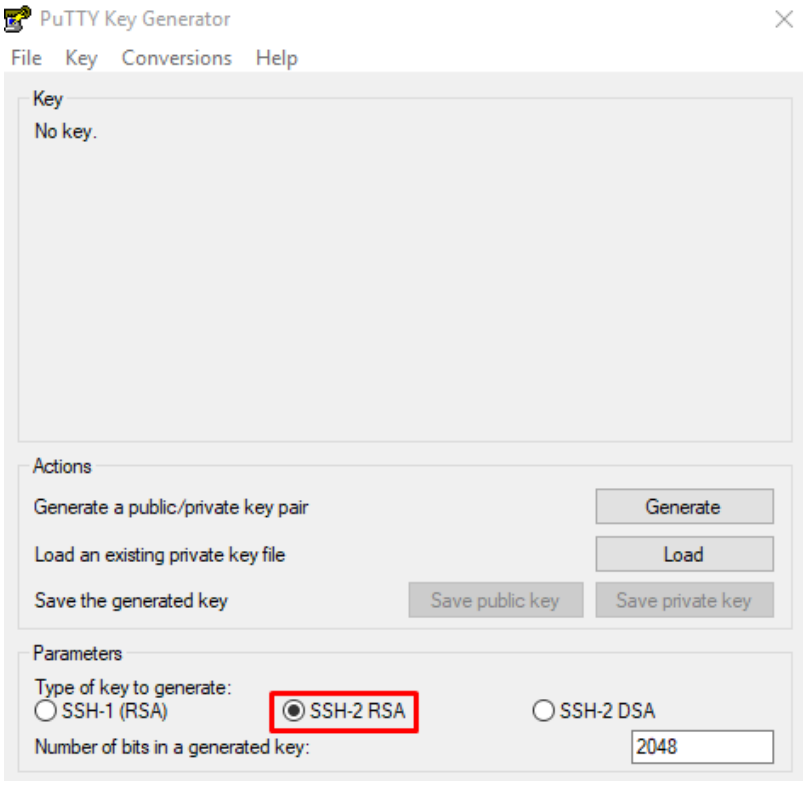
2.2 Создание ключей в Windows

Подключиться по SSH с ОС Windows на сервер можно через PuTTY или OpenSSH. Генерация ключей в этом случае выполняется также при помощи этих программ. В примере мы используем клиент PuTTY.

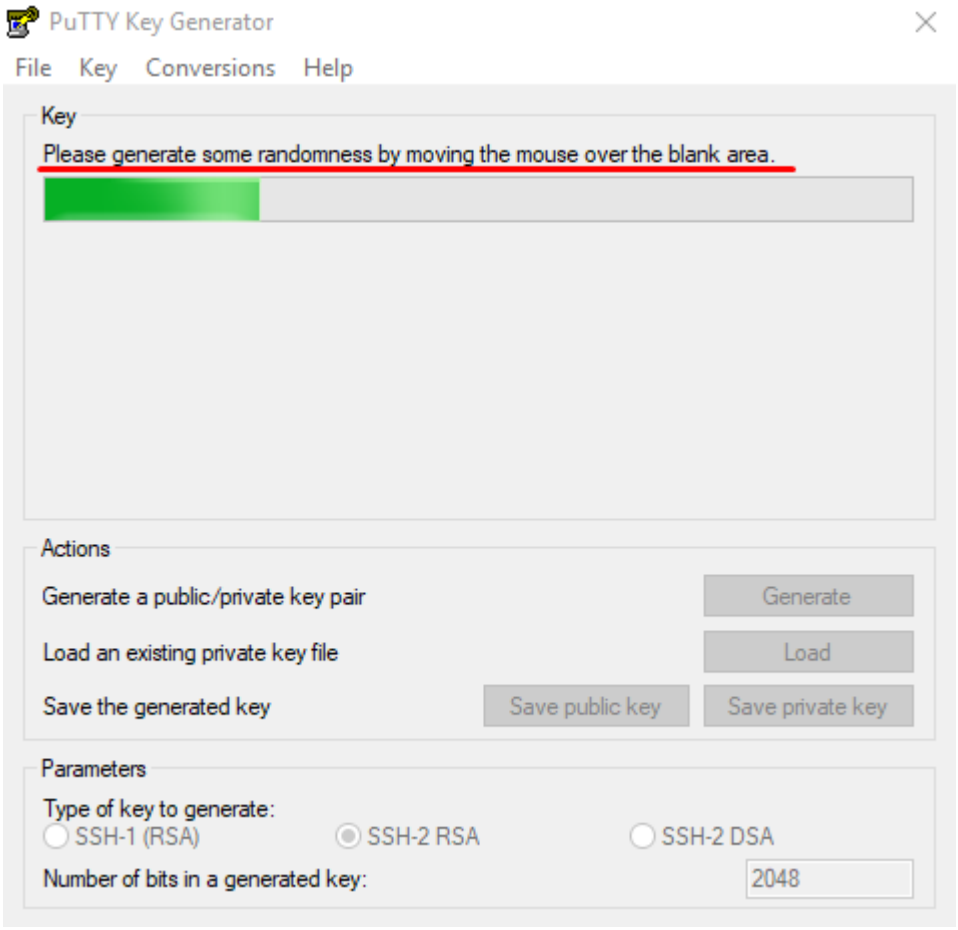
Запустите приложение PuTTYgen, которое устанавливается вместе с PuTTY.



Выберите тип ключа SSH2-RSA и нажмите Generate.



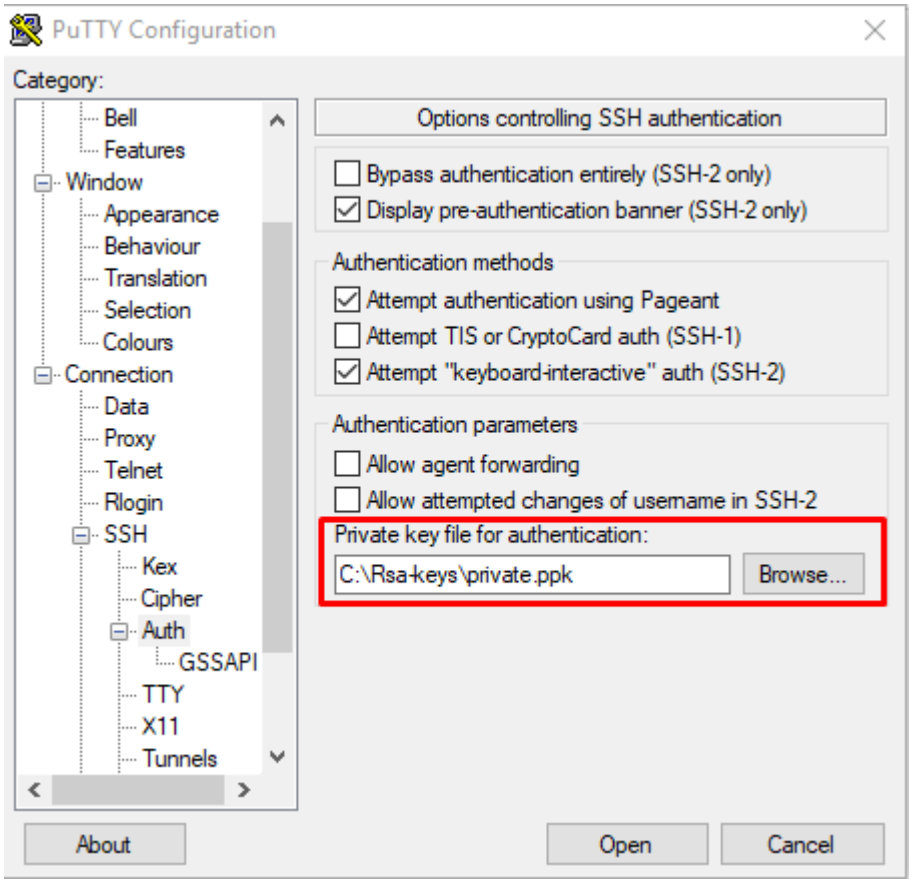
В процессе генерации ключей несколько раз произвольно проведите мышкой по экрану приложения для создания случайных величин, используемых для ключей.



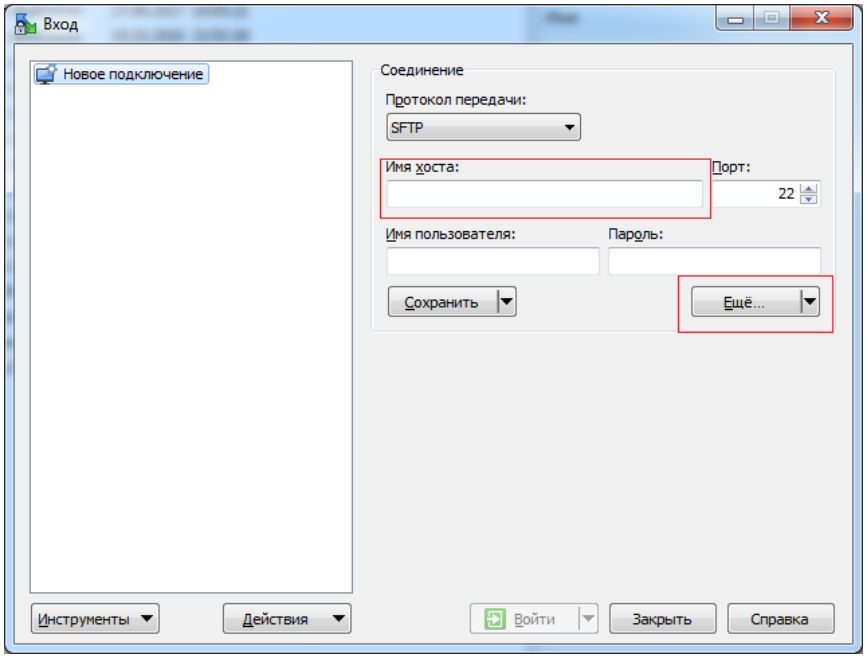
После завершения создания ключей, открытый ключ выводится на экран, закрытый хранится в памяти приложения. Чтобы сохранить эти ключи нажмите «Save public key» и «Save private key». Укажите расположение файлов с ключами. При сохранении закрытого ключа, если не заполнено поле «Key passphrase», появится запрос подтверждения сохранения без секретной фразы.

Теперь открытый ключ необходимо скопировать на сервер в файл «authorized_keys» (аналогично с предыдущим пунктом).

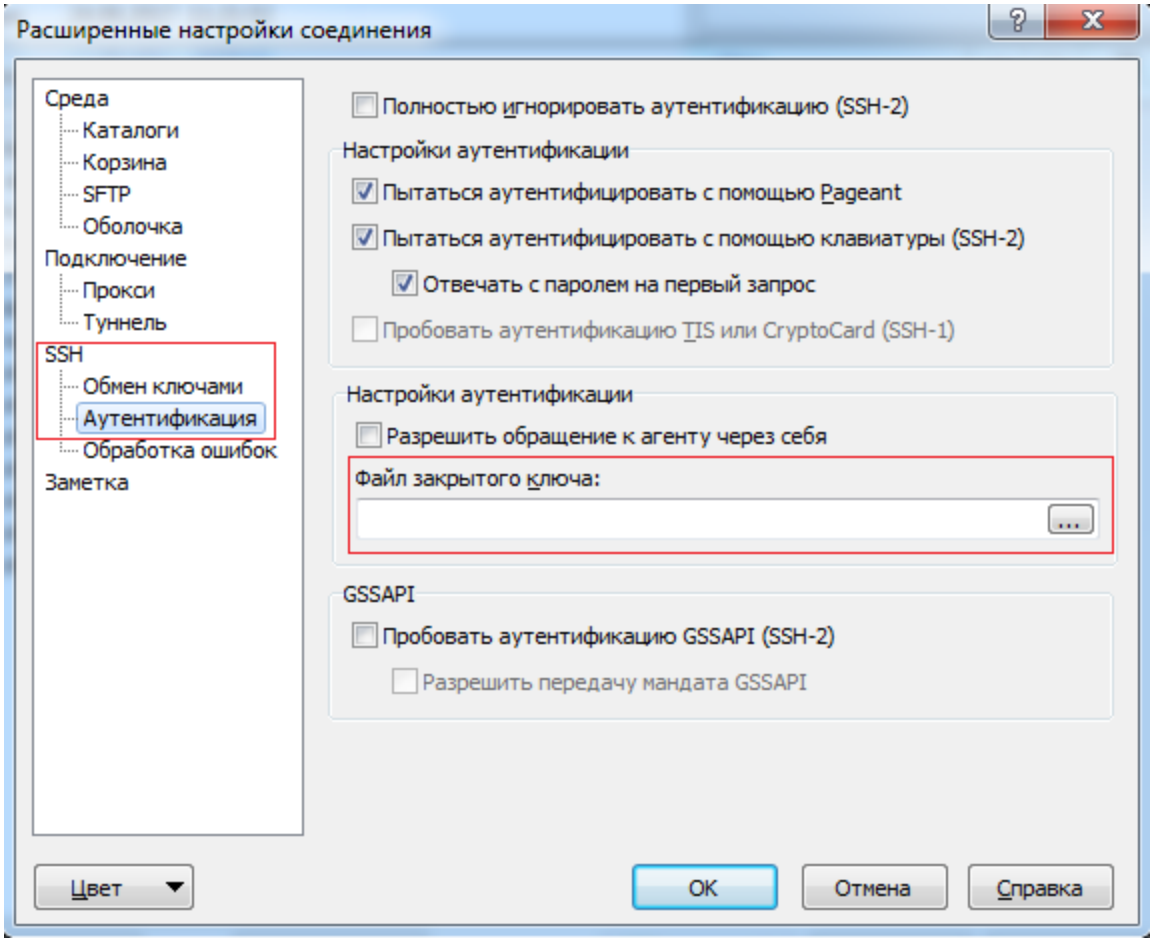
При запуске PuTTY укажите путь к закрытому ключу на локальном компьютере. Для этого во вкладке Connections → Auth выберите необходимый путь.



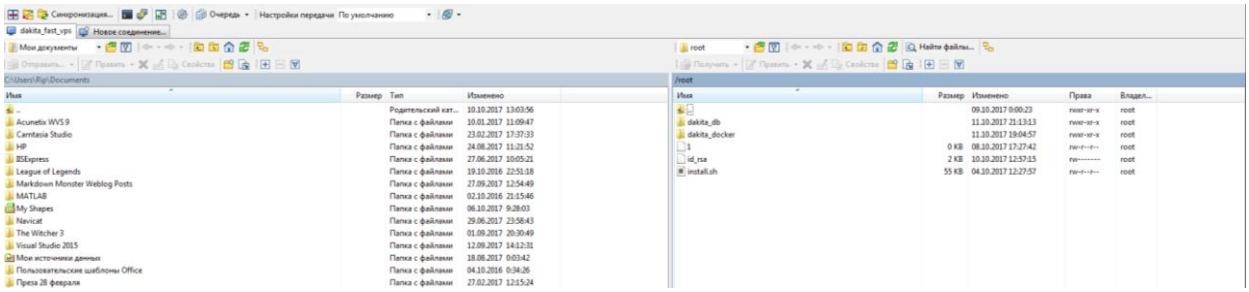
Попробуйте использовать WinSCP для подключения к удаленному серверу с использованием ключа. Для этого скачайте WinSCP Portable и откройте его.



В поле имя хоста укажите адрес сервера к которому необходимо подключиться. В поле имя пользователя введите имя пользователя, с каким необходимо подключиться. Для указания private ключа – нажмите на кнопку «Ещё..», далее «SSH» -> «Аутентификация». В строке «Файл закрытого ключа», выберите файл закрытого ключа, сгенерированный ранее.



Нажмите ОК, затем «Войти». При успешном подключении вы должны увидеть файловую систему Linux сервера.



3. Задание

Для сдачи лабораторной работы необходимо сгенерировать ключи доступа и настроить соединение для входа без пароля:

1) с ПК студента на сервер к своему логину (на индивидуальный порт);
2) с своего логина на сервере к логину другого студента (нужно договориться с ним).

Оба соединения отмечены широкими серыми стрелками на рисунке ниже: