# Руководство пользователя Лаборатории электроники "Байкал" (ЛэБ)

6 февраля 2018 г.

# Оглавление

# 1 Общая информация о комплексе

Комплекс ЛэБ ("Лаборатория электроники Байкал") предназначен для работы с целевыми платформами на основе процессора Байкал-Т1. Схема аппаратно-программного комплекса ЛэБ представленана рисунке 1. В его состав входят:

- Сервер x86 подключенный к сети Интернет (Сервер ЛэБ). Используется в качестве платформы разработчика программного обеспечения для микросхемы процессора Байкал-Т1.
- Набор целевых платформ на процессоре Байкал-Т1:
  - тонкий клиент minilTX производства АО «Т-Платформы»,
  - 3 оценочных платы БФК 3.x производства АО «Байкал Электроникс»;
- коммутатор Ethernet. Предназначен для подключения целевых платформ к хост-платформе разработчика (серверу ЛэБ).

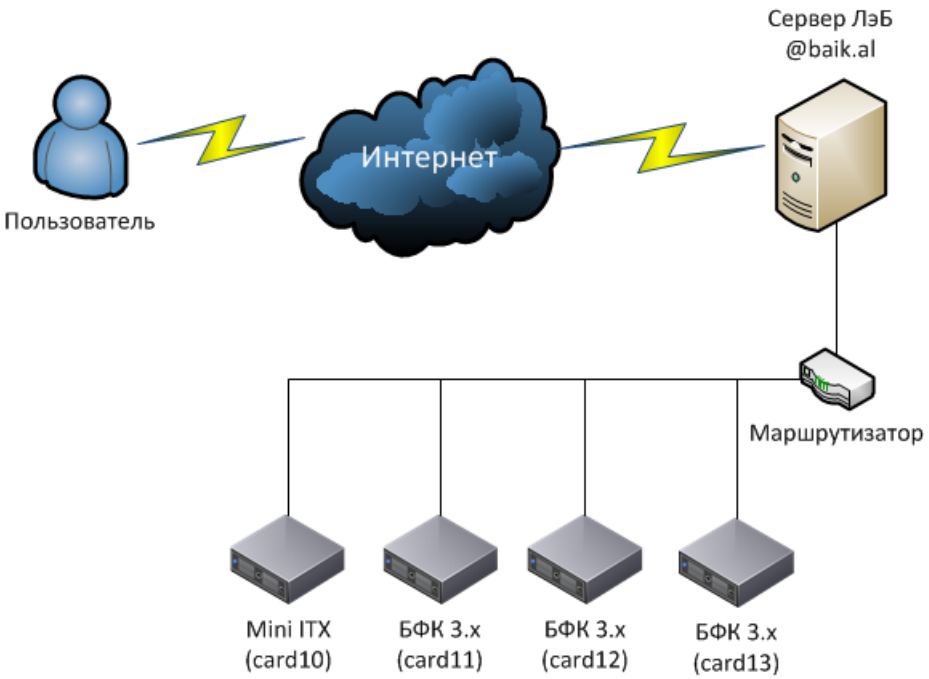


Рис. 1 Схема АПК ЛэБ

Байкал-Т1 – отечественная система на кристалле на базе архитектуры нового поколения MIPS Warrior P-class P5600. Это современный энергоэффективный процессор с большим набором высокоскоростных интерфейсов, предназначенный для широкого диапазона целевых устройств потребительского и промышленного рынков. Более подробная информация представлена на сайте производителя (АО "Байкал Электроникс") в разделе Документация.

Работа с комплексом происходит через сервер ЛэБ, на котором настроена вся необходимая инфраструктура для работы с целевыми платформами. Для работы доступно несколько целевых платформ на основе Байкал-Т1. Более подробная информация об используемой платформе представлена в документе "Руководство по Быстрому Запуску". Система, настроенная на головной машине, позволяет нескольким пользователям одновременно работать с разными целевыми платформами, при этом не мешая друг другу. Для этого создана система резервирования целевых платформ и разграничения доступа к ним по времени.

# 2 Работа с комплексом

## 2.1 Регистрация для работы с ЛэБ

Перед началом работы следует ознакомиться с регламентом работы аппаратно-программного комплекса ЛэБ, доступном на сайте проекта https://baik.al

Для работы с ресурсами ЛэБ необходимо зарегистрироваться на сайте https://baik.al. Для регистрации Вам потребуется сгенерирровать пару SSH-ключей (открытый и закрытый). Детальное описание того, как это сделать для Unix-систем и для Windows приведено в Приложении 1 и Приложении 2.

## 2.2 Информация о целевых платформах

Для определения доступных целевых платформ в системе используется утилита card_info. Она позволяет получить список целевых платформ и информацию о них. Для того получения списка целевых платформ необходимо выполнить команду `card_info -l`.

```
user@baikal-head:~$ card_info -l
List of cards:
1. card10
2. card11
3. card12
4. card13
```

Для получения более подробной информации о целевой платформе, например, `card11`, необходимо выполнить команду `card_info card11`:

```
user@baikal-head:~$ card_info card11
 Info for card number: Card11
 ================================
     Card type                          : BFK3.0
     Console port                       : /dev/ttyCARD11
     BMC management port                : /dev/ttyBMC11
     Current user working with card     : uid 5000
     Ethernet primary port number       : 2
     Vlan number for primary port       : 11
     Primary IP address of the card     : 192.168.191.6
     Primary IP netmask of the card     : 255.255.255.252
     IP address of the server           : 192.168.191.5
     Primary port state                 : up
     Ethernet secondary port number     : 18
     Vlan number for secondary port     : 3
     Secondary IP address of the card   : 192.168.192.2
     Secondary IP netmask of the card   : 255.255.255.0
     IP address of the server           : 192.168.190.1
     Secondary port state               : down
```

В данном выводе указаны:

- Тип целевой платформы: `BFK3.0` В зависимости от типа у платформы может быть разный набор опций по управлению и разный набор портов.
- Консольный порт `/dev/ttyCARD11`, через который производится основное управление целевой платформой. В течение времени, выделенного пользователю, он сможет управлять платформой через этот интерфейс.
- Интерфейс управления питанием и низкоуровневыми функциями целевой платформы: `/dev/ttyBMC11`. Через этот интерфейс осуществляется управление питанием платформы. Для этого предусмотрена специальная программа `БФК`
- Указан uid пользователя, который сейчас имеет права на управление целевой платформой. В данном примере это `uid 5000`.
- Указаны настройки сетевых интерфейсов для данной целевой платформы, порт на коммутаторе, в который воткнута целевая платформа и его настройки, IP, выданный платформе. Аналогичная информация для второго порта целевой платформы при его наличии.

При необходимости диагностики сети целевой платформы можно запросить более детальную информацию командой `card_info card11 detailed`. В этом случае утилита отобразит дополнительную статистическую информацию о портах на коммутаторе, к которым подключена целевая платформа.

## 2.3 Бронирование времени на целевой платформе

Для комплекса реализована система разделения ресурсов по времени. Каждый из пользователей оставляет заявку на использование целефой платформы в определенный промежуток времени.

- График загруженности целевых платформ доступен по адресу https://baik.al/calendar
- На данный момент время бронирования ограничено 24 часами. В случае необходимости более длительного использования целевой платформы следует обратиться к системному администратору комплекса support@baik.al.
- Расписание проверяется раз в 5 минут для всех целевых платформ. В рамках этого промежутка новая бронь может не применяться автоматически. Для того чтобы принудительно установить права доступа к целевой платформе, если наступило время, следует использовать команду `recheck_card_perm`.

Для бронирования времени используется утилита `baikal-scheduler`. Запрос на бронирование временного слота для работы на целевой платформе производится в интерактивном режиме командой `baikal-scheduler reserve`.

В качестве входных параметров эта команда запрашивает у пользователя название целевой платформы, дату и время начала и окончания брони, на который данная платформа будет выделена пользователю. После получения всех данных от пользователя система проверяет, не занято ли данное время, и бронирует его. В случае успешного бронирования выдется ID операции.

```
ivanov@baikal-head:~$ baikal-scheduler reserve
Maximum reserve period is 5 hours.
Enter card to reserve (cardNN): card11
Enter start date (YYYY/MM/DD): 2018/02/05
Enter start time (HH:MM): 9:00
Enter the length (+HH:MM): +3:00
Time for you has been reserved
Your reservation id is: baik.al-card11-20180205T090000
```

В случае, если целевая платформа хотя бы частично занята в требуемый промежуток

времени, то произойдет оповещение о том, что данное время занято и вернет ID брони, которая пересекается с указанным промежутком времени.

```
ivanov@baikal-head:~$ baikal-scheduler reserve
Maximum reserve period is 5 hours.
Enter card to reserve (cardNN): card11
Enter start date (YYYY/MM/DD): 2018/02/05
Enter start time (HH:MM): 10:00
Enter the length (+HH:MM): +3:00
Your reservation overlaps with another reservation: baik.al-card11-20180205
T090000
We will not add your reservation.
```

## 2.4 Отмена бронирования

Для отмены ранее созданной брони используется команда `baikal-scheduler unreserve`. В качестве входных параметров эта команда запрашивает у пользователя название целевой платформы и id брони.

```
ivanov@baikal-head:~$ baikal-scheduler unreserve
Enter card number to unreserve: card11
Enter id of the event to cancel: baik.al-card11-20180205T090000
Removing calendar event with id: baik.al-card11-20180205T090000
```

При попытке удалить бронь, принадлежащую другому пользователю, выдается предупреждение:

```
ivanovr@baikal-head:~$ baikal-scheduler unreserve
Enter card number to unreserve: card11
Enter id of the event to cancel: baik.al-card11-20180205T090000
The event does not belong to you. Not removing.
```

### 2.5 Работа в течение забронированного времени

Система автоматически назначает соответствующие права пользователю для всех устройств и сервисов, необходимых для работы с целевой платформой. На время своего бронирования пользователь может назанчить права принудительно, выполнив команду `recheck_card_perm`. Данная команда сверяется с текущим расписанием для целевой платформы и назначает соответствующие права.

## 3 Работа с целевой платформой

### 3.1 Утилита screen для работы через консоль

Для подключения к платформе используется уитилита `screen`, которая позволяет управлять несколькими сессиями из одной консоли или окна терминала. Команда `screen /dev/ttyCARD11 115200` позволит подключиться к консольному порту целевой платформы CARD11 на скорости 115200 бод. Завершение соединения происходит посредством нажатия последовательности клавиш `<Ctrl+a> <k>` и последующего подтверждения отключения. Для возврата в консоль используется команда `screen -r`.

### 3.2 Утилита SCP

Для работы с файлами можно использовать команду scp. Формат её использования похож на использования команды cp в unix-подобных ОС:
`scp local_file ivanov@baik.al:~/remote_file`, где

- `local_file` – имя файла на текущей системы
- `baik.al` – имя удаленной системы (как задано в конфигурации ssh или реальное)
- `ivanov` – имя пользователя на удаленной системе (не обязательно указывать если оно указано в конфигурации ssh)
- `~/remote_file` – путь к файлу на удаленной системе, полный или относительно домашней директории (в данном случае указано полный путь, с сокращением которое разворачивается в пуь к домашней директории).

Для передачи файла в обратном направлении можно просто поменять две последних части местами: `scp ivanov@baik.al:~/remote_file local_file`
Кроме использования scp многие файловые менеджеры в unix-подобных системах умеют взаимодействовать с удаленными серверами по ssh для передачи файлов.

## 3.3 Утилита WinSCP

Для обмена файлами с Windows-ОС есть возможность использовать программу WinSCP. При осуществлении входа в систему следует, как и в случае с PuTTY, указать путь к файлу с ssh-ключом, как показано на рисунке 2. Данная программа позволяет удобно работать с файлами на удаленной системе.

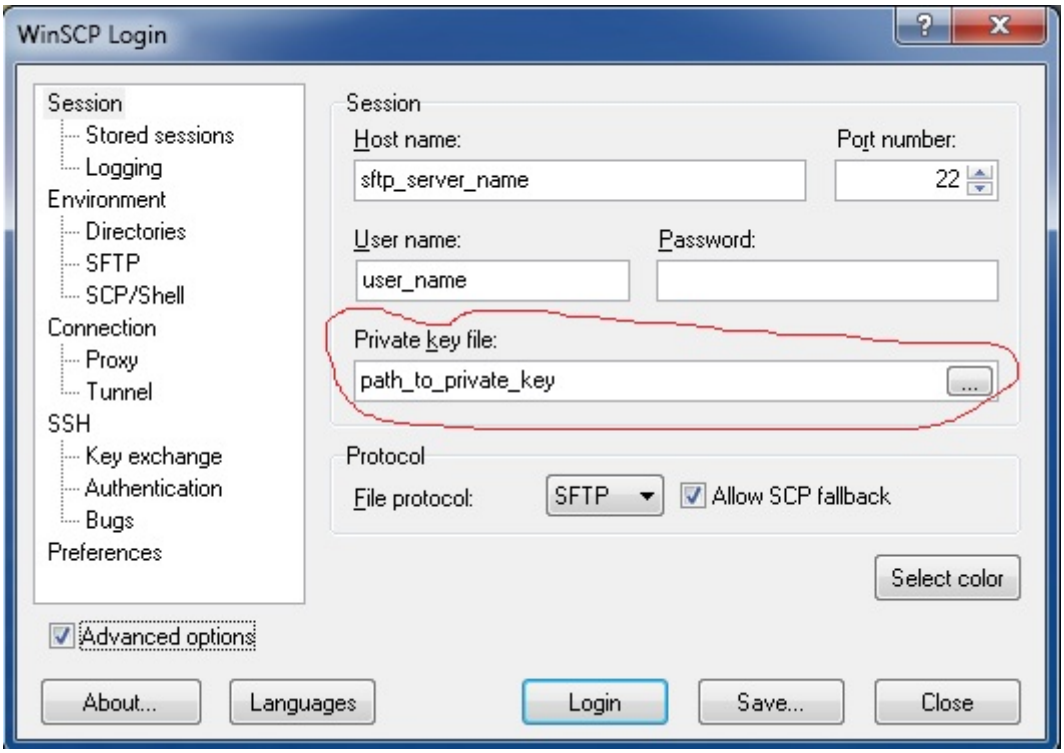


Рис. 2 Указание пути к файлу с ключом

## 3.4 Работа с TFTP

Для возможности загрузки на систему персональных образов на головной систее установлены TFTP и DHCP серверы. Их настройка позволяет на каждую целевую платформу загружать необходимый исполняемый файл. Для каждой целевой платформы в директории `/srv/tftp` созданы поддиректории по имени каждой платформы `/srv/tftp/card10` и т.д. В каждой из этих директорий располагается файл `mainbin.bin`, который необходим для загрузки через DHCP-сервер. Данный каталог доступен пользователю для записи (в строго забронированное время). Для более оперативной работы с целевыми платформами рекомендуется подготовить файлы для загрузки в эти каталоги заранее в домашней директории.

## 3.5 Работа с NFS

На системе выделено пространство для предоставляемых по NFS каталогов пользователей. Для каждого пользователя выделен отдельный каталог. Поскольку в каталоге должны лежать файловые системы для использования на целевых платформах, данный каталог доступен на чтение на самом сервере, но в связи с недостатком прав у пользователей для размещения там системных файлов их использование на запись ограничено.

Для работы с этими каталогами рекомендуетсяи смонтировать данный каталог на целевой платформе командой

`mount -t nfs -o vers=4 192.168.191.1:/srv/nfs/ivanov/dir1 /mnt`, где

- 192.168.191.1 – адрес сервера ЛэБ с nfs (на 1 меньше выданного плате)
- ivanov – логин пользователя
- dir1 – монтируемые подкаталог, если требуется и если он предварительно создан
- /mnt – директория на карте для монтирования

Далее можно использовать каталог и размещать там файлы. В случае если вы планируете использовать несколько целевых платформ одновременно, рекомендуется использовать отдельные подкаталоги для каждой из них.

Для разработки приложений будет выделена отдельная платформа(card10). На ней будет развернута система, на которой будут доступны домашние каталоги пользователей и аутентификация теми же логинами и ключами, что и на головном сервере (это будет реализовано позже).

## 3.6 Работа с питанием

При отладке приложений на целевой платформе (особенно для нативных приложений) могут возникнуть ситуации, когда платформа зависла и не откликается. В этом случае Вам следует ее перезагрузить. Для этого можно использовать утилиту `card_power card11 cycle`. Также можно выключить или включить платформу командами `card_power card11 off` или `card_power card11 on`. Можно получить информацию о текущем состоянии платформы с помощью команды `card_power card11 status`. Данная опция доступна только для оценочных плат БФК 3.x.

# Приложение 1. Генерация SSH-ключа для регистрации

Как следует из названия – это ключ для ssh, т.е. используется программой ssh (или другим ssh-клиентом, в дальнейшем так и будем говорить: **ssh-клиент**) для авторизации на удаленном сервере через демон/сервис sshd (будем называть этот сервер **ssh-сервером**).

Если вы попали на эту страницу, значит в дальнейшем Вы таким способом будете (или от Вас требуют) авторизоваться (т.е. доказать, что это именно Вы) на каком-то unix-подобном сервере.

*Важно то, что для всего этого достаточно **одного** ключа, **не требуется** каждый раз создавать новый ключ.*

**Создание ssh-ключа в unix-подобных системах и использование ssh**

Перед созданием ключа убедитесь, что у Вас еще нет ключа:

- `ls -l ~/.ssh`
- если команда отработала без ошибок и вы получили в выводе файлы `id_rsa/id_dsa`, то у Вас уже есть нужный ключ, создавать не требуется
  - Пример:

```
-rw-rw-r-- 1 user user    89 янв.  24  2013 config
-rw------- 1 user user  1679 янв.  27  2012 id_rsa
-rw-r--r-- 1 user user   393 янв.  27  2012 id_rsa.pub
-rw------- 1 user user 21618 дек.   9 18:10 known_hosts
```

  - если несмотря на наличие ключа Вы хотите создать отдельный ключ для сервера, то используйте **универсальный способ**
- если ssh-ключа у Вас нет, то требуется создать (можно использовать **простой способ**)

**Простой способ создания ssh-ключей**

Для создания ключа введите команду: `ssh-keygen`. В результате эта команда в интерактивном режиме создаст пару ssh-ключей (открытый и закрытый). Местоположение файлов:

- закрытый ssh-ключ (identification): `~/.ssh/id_rsa`
- публичный ssh-ключ (public key): `~/.ssh/id_rsa.pub`.

*Здесь указаны имена файлов в случае использования алгоритма rsa по умолчанию. Но это может зависеть от дистрибутива который вы используете.*

При создании генератор запросит пароль, если введете пустой, то ключ зашифрован не будет (если Вы забудете тот пароль, которым зашифруете ключ, то воспользоваться ключом больше не сможете).

**Универсальный способ создания ssh-ключей**

Будем использовать ту же программу (ssh-keygen), но с дополнительными параметрами:

- **-t тип-ключа (rsa/dsa/...)**: какой тип ключа нужен
- **-b длина ключа**: какой длины (для dsa длина всегда 1024)
- **-f имя_ключа**: в какой файл сохранять ключ, если мы хотим разные ключи для разных целей

Пример:

- вводите: `ssh-keygen -t rsa -b 4096 -f ~/.ssh/id_rsa_baikal`
- в результате:
  - в интерактивном режиме будет создана пара ssh-ключей (открытый и закрытый) типа RSA с длиной ключа 4096 бит (разумеется Вы можете зашифровать закрытую часть ключа паролем).
  - ключи:
    * закрытый ssh-ключ (identification) будет сохранен в `~/.ssh/id_rsa_baikal`
    * открытый ssh-ключ (public key) будет сохранен в `~/.ssh/id_rsa_baikal.pub`
    * собственно о местоположении ключей будет сообщено в процессе создания, в нашем примере это выглядело так:

      ```
      Your identification has been saved in /home/user/.ssh/id_rsa_baikal.
      Your public key has been saved in /home/user/.ssh/id_rsa_baikal.pub.
      ```

В дальнейшем не забудьте прописать в настройках программы ssh для данного сервера путь именно к этому закрытому ssh-ключу (в параметре IdentityFile файла `~/.ssh/config`), о чем — в следующем разделе.

За более подробной информацией обращайтесь к справочным страницам командой `man ssh-keygen` или на сайте OpenBSD и F.A.Q по OpenSSH

**Конфигурирование ssh-программы**

Хотя в данный момент Вы можете уже пользоваться ssh-ключом для доступа на требуемый сервер, рекомендуется для удобства прописать требуемые параметры, чтобы не вводить заново. Особенно это актуально в случае, если для разных серверов требуется разный ssh-ключ.

Программа ssh читает настройки из файла `~/.ssh/config`, для того, чтобы узнать о **всех** доступных параметрах рекомендуем ознакомиться с документацией (`man ssh_config` или на сайте OpenBSD), а здесь мы дадим пример, который каждый может переделать под свои нужды.

Допустим вы хотите входить на вычислительный сервер `baik.al` под пользователем `ivanov` и использовать ключ `~/.ssh/id_rsa` В командной строке Вам придется ввести: `ssh baik.al -l ivanov` где:

- -l: какой логин использовать
- -i: какой ключ использовать (указывается закрытый ssh-ключ (Identity))
- baik.al: на какой сервер заходить.

Довольно длинная команда, но ее можно сильно сократить, для этого нам и нужен `~/.ssh/config` Пример конфигурационного файла `~/.ssh/config`:

```
# настройки для сервера baik.al
Host baikal
    Hostname baikal
    User ivanov
    IdentityFile ~/.ssh/id_rsa
```

Если при наличии этой информации в `~/.ssh/config`, мы введем `ssh baikal`, то мы получим тот же результат, что и в предыдущем случае. Кратко по параметрам:

- #: как и в большинстве конфигурационных файлах unix этот символ используется для комментариев
- `Host`: имя (множество имен), обозначающие секцию.
- `Hostname`: какое на самом деле имя сервера использовать
- `User`: под каким пользователем заходить
- `IdentityFile`: какой ключ авторизует пользователя, указывается его **закрытая часть (т.е. закрытый ssh-ключ)**

# Приложение 2. Создание PuTTY-ключа (Windows и другие системы) и использование PuTTY

Для этого требуется запустить программу PuTTYgen так, как показано на рисунке 3.

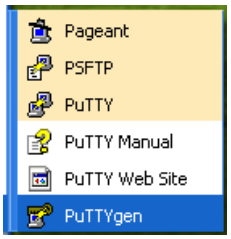


Рис. 3 Запуск программы PuTTYgen

Откроется окно, представленное на рисунке 4:

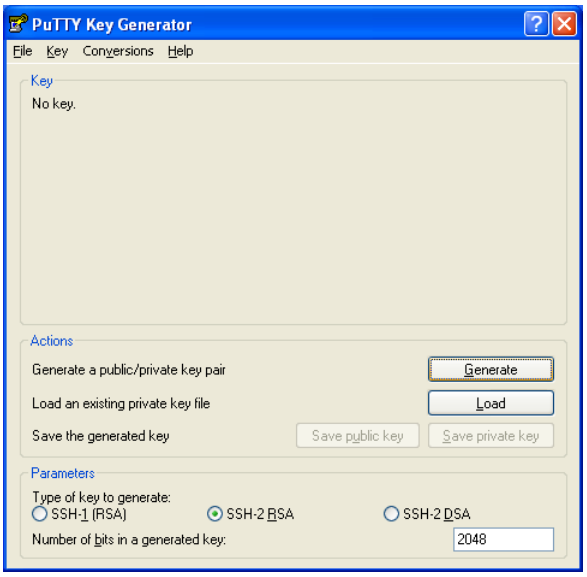


Рис. 4 Параметры генерирования ключа

Установите параметр "SSH-2 RSA"и впишите 2048 в поле ниже и нажмите кнопу **Generate**, после чего запустится процесс создания ключа, показанный на рисунке 5:

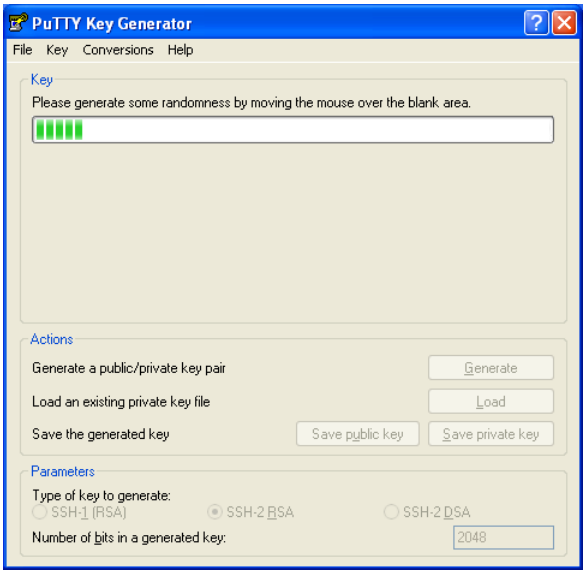


Рис. 5 Генерирование ключа

*Не забудьте подвигать мышкой в районе пустой области под показателями прогресса генерации (программа использует эти движения для создания непредсказуемого ключа). Двигать мышкой надо пока ключ не будет создан.*

После создания окно поменяется на представленное на рисунке 6:

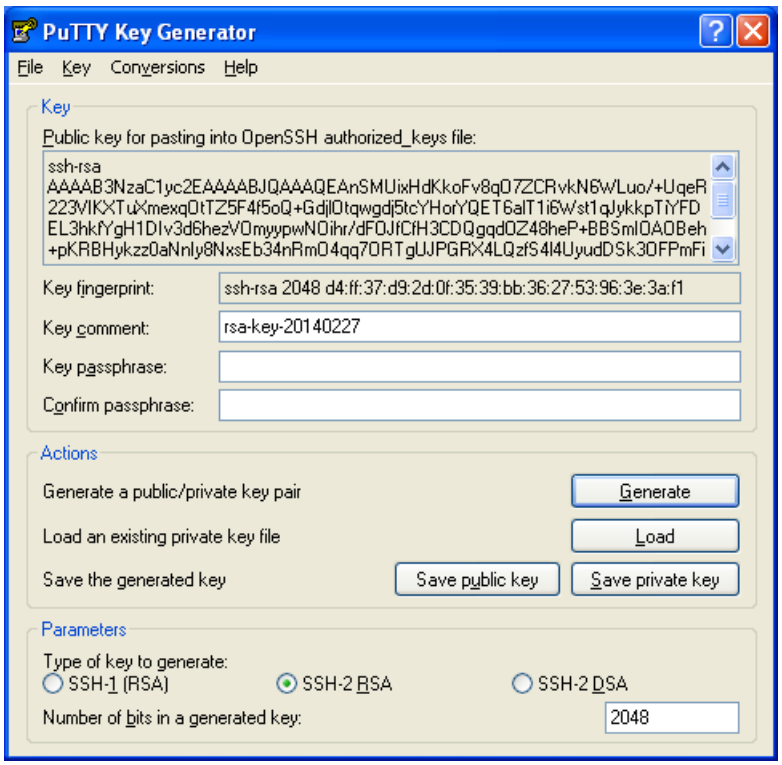


Рис. 6 Окно после создания ключа

*У вас будет другой ключ, поэтому не пугайтесь.*

Не нажимайте кнопку "Save public key вместо этого вам потребуется из поля с названием **Public key for pasting into OpenSSH authorized_keys file** публичный ключ (выделить, скопировать, вставить в простой текстовый файл и сохранить его).

Открытая часть ключа потребуется администратору для того чтобы предоставить вам доступ.

После создания не забудьте сохранить ключ **Save private key**, если пароль (passphrase) для ключа не задан, то возникнет предупреждающее окно - 7:

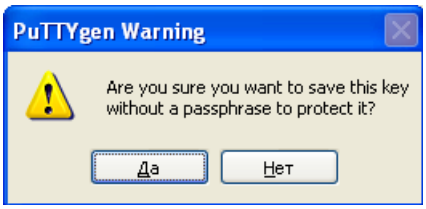


Рис. 7 Предупреждающее окно

Если согласиться, то ключ будет без пароля.

Далее даете имя и сохраняете.

*Данный файл никому давать нельзя, он содержит приватную часть!*

### Конфигурирование программы PuTTY для входа на систему

При запуске программы PuTTY появится окно, представленное на рисунке 8:

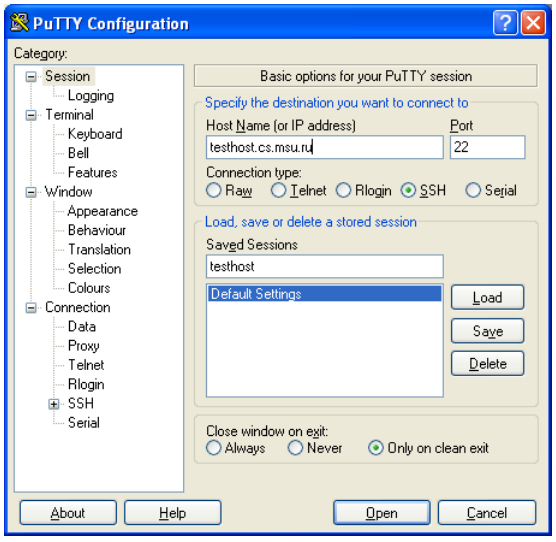


Рис. 8 Конфигурирование PuTTY

Будем считать, что Вы хотите заходить на компьютер testhost.cs.msu.ru с нашим ключом. Параметры планируем сохранять под именем testhost.

Для указания ключа надо перейти на параметры **SSH** (раскрыть) и выбрать **Auth** - рисунок 9

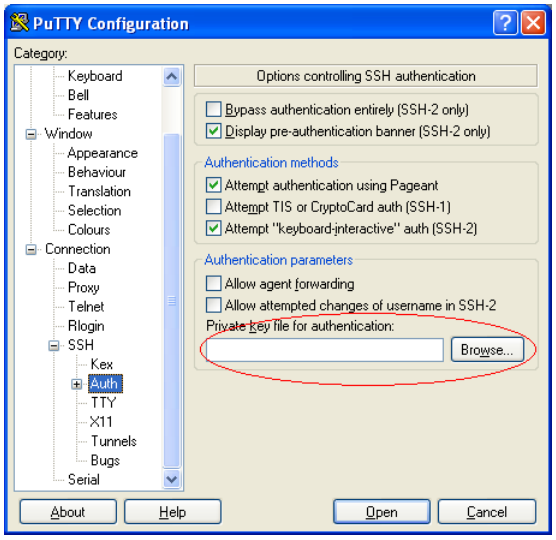


Рис. 9 Указание ключа

Далее надо загрузить файл (**Browse**), найти свой ключ и загрузить/открыть - рисунок 10:

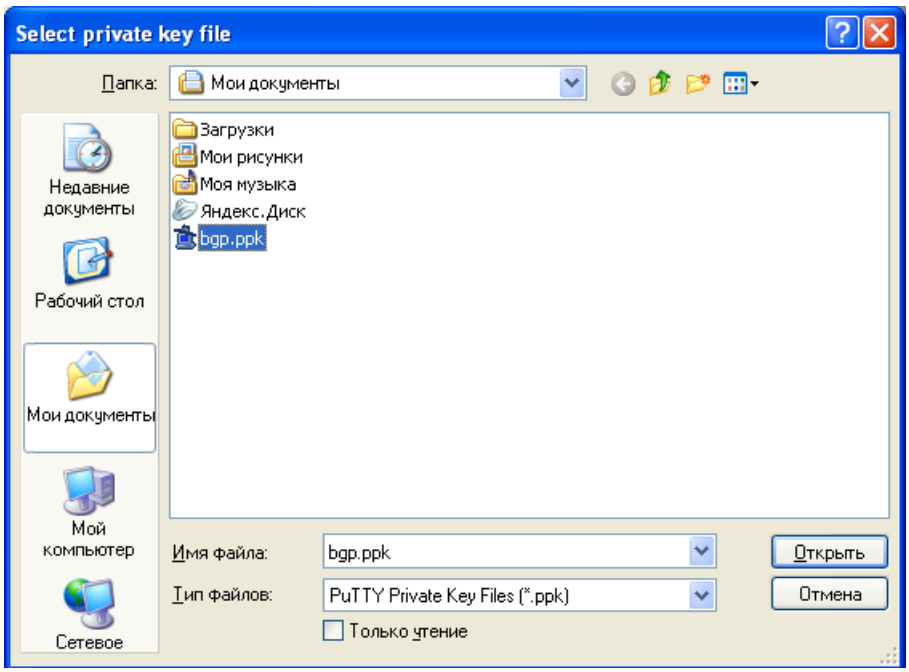


Рис. 10 Выбор файла

После этого надо сохранить параметры (вернуться на **Session** и сохранить) так, как показано на рисунке 11:

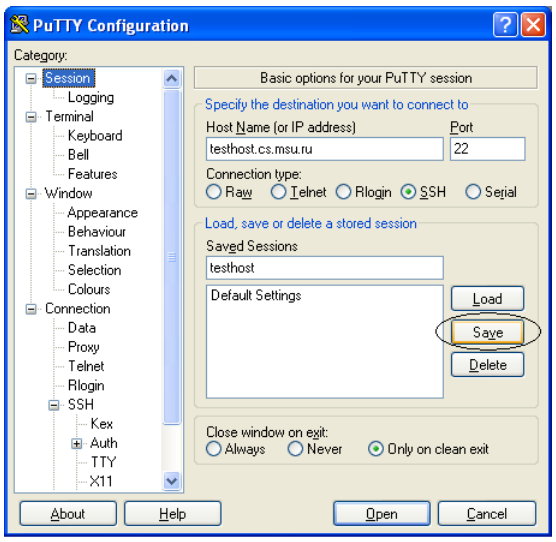


Рис. 11 Сохранение параметров

В результате в следующий раз, когда Вы запустите putty, Вы увидите среди сессий нужную Вам - рисунок 12:

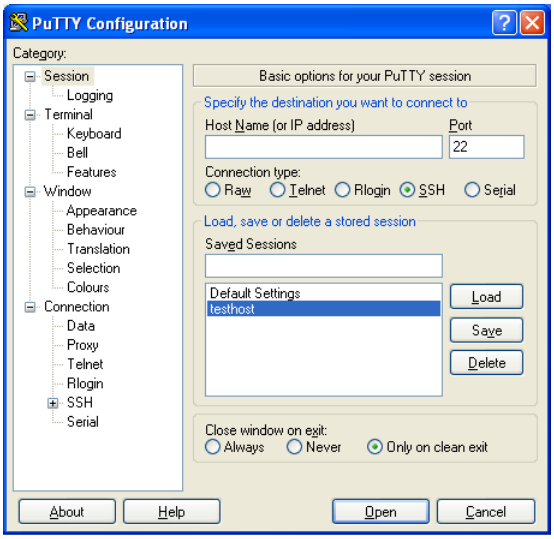


Рис. 12 Выбор сохраненной сессии

Останется только ее запустить/открыть (**Open**):

- выделить и нажать кнопку Open
- либо использовать двойной клик на имени сессии

# Приложение 3. Команды для работы с ЛэБ

В данном разделе представлено краткое описание необходимых для работы с ЛэБ команд.

### baikal_scheduler

*Описание:* Утилита управления планировщиком, позволяет в интерактивном режиме запросить себе временной слот для работы на целевой платформе.

*Использование:*

`baikal_scheduler reserve` – бронирование целевой платформы

`baikal_scheduler unreserve` – отмена созданной ранее брони целевой платформы.

### card_info

*Описание:* Утилита для получения информации о целевых платформах.

*Использование:*

`card_info -l` – получение списка целевых платформ.

`card_info card10` – получение подробной информации о целевой платформе.

`card_info card11 detailed` – получение детальной информации для диагностики сети целевой платформы.

### card_power

*Описание:* Утилита для управления питанием целевой платформы.

*Использование:*

`card_power card11 cycle` – осуществление перезагрузки целевой платформы.

`card_power card11 on` – команда включения целевой платформы.

`card_power card11 off` – команда выключения целевой платформы.

`card_power card11 status` – команда, предоставляющая информацию о текущем состоянии платформы. Данная опция доступна только для оценочных плат БФК 3.0.

### id

*Описание:* Утилита для получения информации об идентификационном номере (ID) бронирования.

*Использование:*

`id -u` – получение своего ID брони.

### mount

*Описание:* Утилита монтирования для работы с каталогами на сервере ЛэБ.

*Использование:*

```
mount -t nfs -o vers=4 192.168.191.5:/srv/nfs/ivanov/dir1 /mnt
```

- `192.168.191.5` – адрес головного сервера с nfs
- `ivanov` – логин пользователя

- `dir1` – монтируемые подкаталог, если требуется и если он предварительно создан
- `/mnt` – директория на платформе для монтирования

### `recheck_card_perm`

*Описание:* Утилита, которую необходимо выполнить при недоступности управления при условии, что время, выделенное для пользователя, наступило.

### `scp`

*Описание:* Утилита для копирования файлов с локального ПК на головной сервер.
*Использование:*
`scp local_file ivanov@baik.al: /remote_file`

- `local_file` – имя файла на текущей системы
- `baikal` – имя удаленной системы (как задано в конфигурации ssh, или реальное)
- `ivanov` – имя пользователя на удаленной системе (не обязательно указывать если оно указано в конфигурации ssh)
- `~/remote_file` – путь к файлу на удаленной системе, полный или относительно домашней директории (в данном случае указано полный путь, с сокращением ~ которое разворачивается в пуь к домашней директории).
- Для передачи файла в обратном направлении можно просто поменять две последних части местами: `scp ivanov@baik.al:~/remote_file local_file`.

### `screen`

*Описание:* Утилита для подключения к целевой платфоме.
*Использование:*
`screen /dev/ttyCARD11 115200` – `/dev/ttyCARD11` – консольный порт платформы, 115200 – скорость передачи данных по консольному порту. Для завершения соединения следует нажать последовательность клавиш `<Ctrl+a> <k>` и подтвердить отключение.