GraphIT! User Guide

Nguyen Minh Duc

Содержание

- Требования
- ГрафИТ!
- Удаленный доступ
- Передача данных
- Редакторы
- Компиляция
- Запуск

Требования

Требования

- SSH-логин на GraphIT!
- Авторизация по SSH2 ключу
- Unix
  - Shell: sh, csh, bash…
  - OpenSSH client (с поддержкой SSH-2 protocol)
  - Optional X-server
- Windows
  - SSH client: PuTTY, WinSCP, Cygwin, SSH Secure Shell Client и Teraterm
  - Optional X-server: Xming

ГрафИТ!

Общая информация

- Адрес: graphit.parallel.ru
- SSH порт: 22
- Логин выдается после регистрации на кластере
- Авторизация **только по SSH ключу!**
- http://parallel.ru/gpu/graphit-howto.html

Установленное ПО

- ОС RedHat Enterprise Linux, версия 5, апдейт 5, сборка для архитектуры x86_64
- Библиотека обмена сообщениями OpenMPI 1.4.2
- Система управления заданиями Cleo
- CUDA Toolkit и CUDA SDK версии 4.1 для Linux, включая компилятор nvcc (поддерживает технологии CUDA и OpenCL)
- Набор утилит GNU (включая make) и компилятор gcc
- Текстовые редакторы vim и emacs
- Кросс-платформенная среда исполнения mono, версия 2.0.1 (совместима с .NET)
- Компилятор ncc языка Nemerle и система расширений NUDA (=Nemerle Unified Device Architecture) для программирования ГПУ

Генерация ключей

Unix – ssh-keygen

- **ssh-keygen -** authentication key generation, management and conversion
- **ssh-keygen [–b bits] [-t type] [-f filename]**
  - **-b bits** - Specifies the number of bits in the key to create. For RSA keys, the minimum size is 768 bits and the default is 2048 bits. Generally, 2048 bits is considered sufficient. DSA keys must be exactly 1024 bits as specified by FIPS 186-2.
  - **-t type** - Specifies the type of key to create. The possible values are ``rsa1'' for protocol version 1 and ``rsa'' or ``dsa'' for protocol version 2.
  - **-f filename** - Specifies the filename of the key file.

Unix – Generating DSA keys

```
me:~ conqueror$ ssh-keygen -t dsa -f graphit_dsa_key
Generating public/private dsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in graphit_dsa_key.
Your public key has been saved in graphit_dsa_key.pub.
The key fingerprint is:
6f:e1:ca:5a:98:2b:26:14:18:38:28:f3:a3:62:77:db conqueror@me
The key's randomart image is:
+--[ DSA 1024]----+
|o                |
|B                |
|.*               |
|. +              |
| . o    S .      |
|o... . o o .     |
|o.. . = . +      |
|  . o. E o       |
|   o .o.o        |
+-----------------+
me:~ conqueror$
```

Unix – Generating RSA keys

```
me:~ conqueror$ ssh-keygen -t rsa -f graphit_rsa_key
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in graphit_rsa_key.
Your public key has been saved in graphit_rsa_key.pub.
The key fingerprint is:
7e:f9:6c:22:f8:62:f0:de:d5:05:48:ab:7b:98:ef:74 conqueror@me
The key's randomart image is:
+--[ RSA 2048]----+
|           .     |
|          . o    |
|           o .   |
|          .   .  |
|         S     . |
|     .  . + o .  |
|      o .= * E   |
|       =..*.+.   |
|      o.oooooo   |
+-----------------+
me:~ conqueror$
```

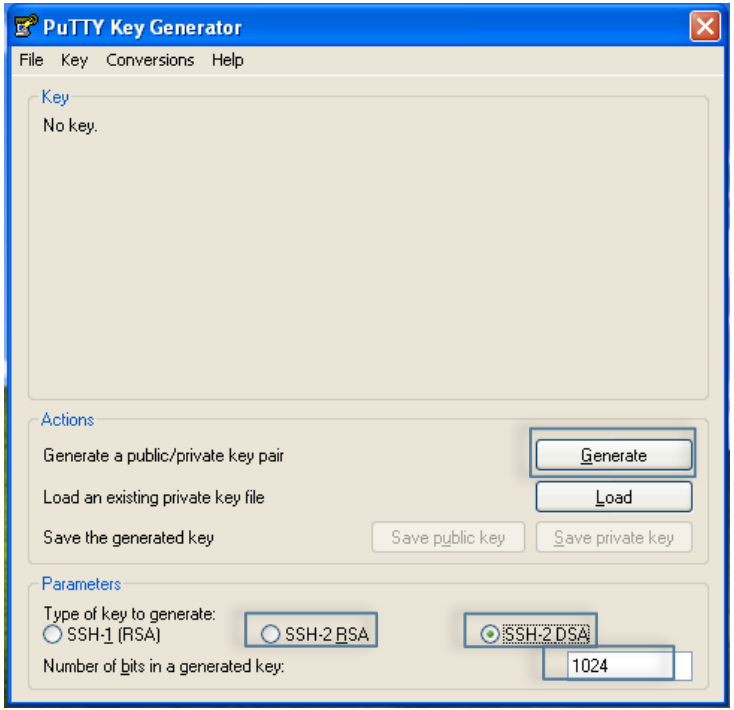
Windows – PuTTYgen

- PuTTYgen is a key generator. It generates pairs of public and private keys to be used with PuTTY, PSCP, and Plink, as well as the PuTTY authentication agent. PuTTYgen generates RSA and DSA keys.

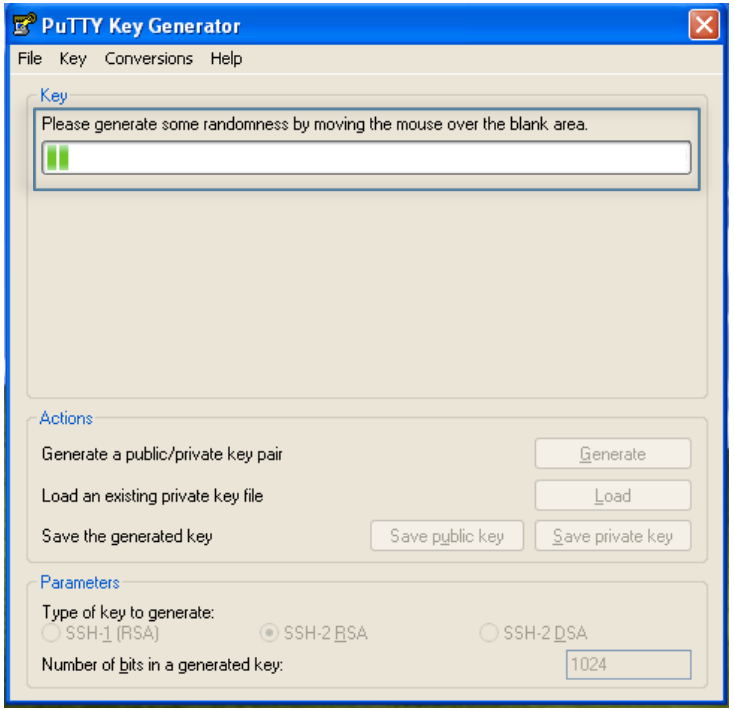
- Download link:

http://the.earth.li/~sgtatham/putty/latest/x86/puttygen.exe

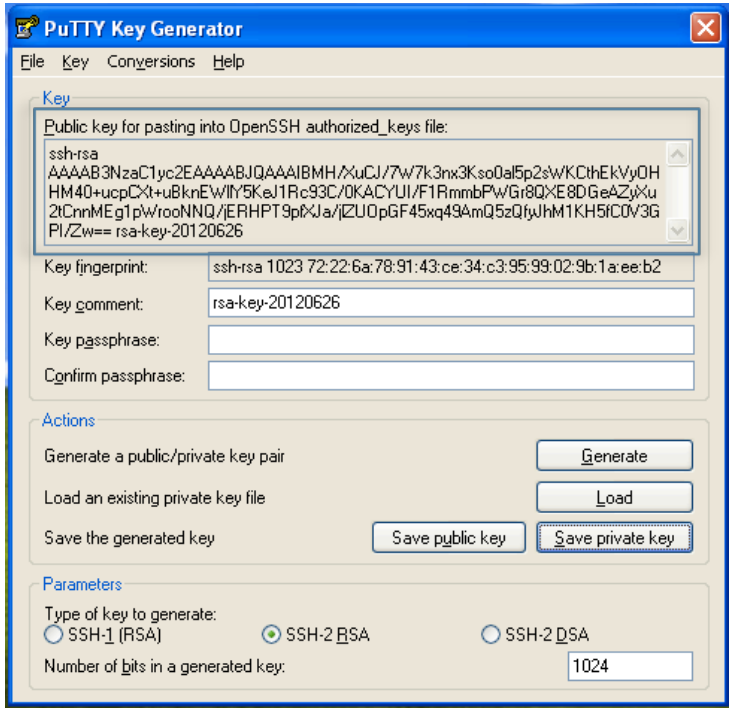
Windows – Generating DSA and RSA keys



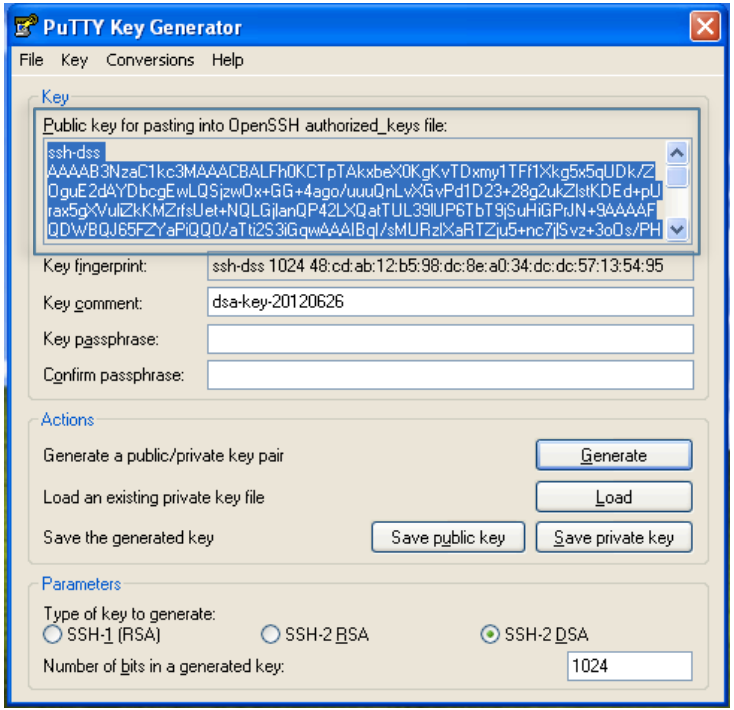
Windows – Generating DSA and RSA keys



Windows – Generating DSA and RSA keys



Windows – Generating DSA and RSA keys



Удаленный доступ

Connecting from Unix

- **ssh** - OpenSSH SSH client (remote login program)

  ssh connects and logs into the specified **hostname** (with optional **username**)

ssh [–X] [-i identity_file] [–p port] [username@]hostname

- **-X** - Enables X11 forwarding
- **-p port** - Port to connect to on the remote host
- **-i identity_file** - Selects a file from which the identity (private key) for RSA or DSA authentication is read

Unix - ssh

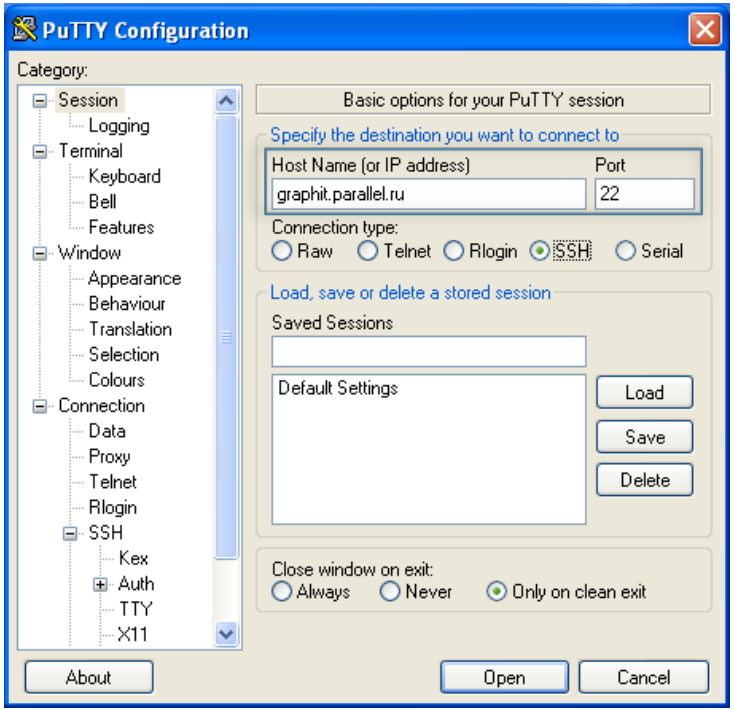
```
me:~ conqueror$ ssh -I .ssh/graphit_dsa_key.private edu-acad12-prep03@graphit.parallel.ru
no support for PKCS#11.
Last login: Tue Jun 26 09:57:10 2012 from 89.249.162.36
-bash: /opt/hmpp-workbench/bin/hmpp-env.sh: Нет такого файла или каталога
[edu-acad12-prep03@graphit ~]$
```

Connecting from Windows

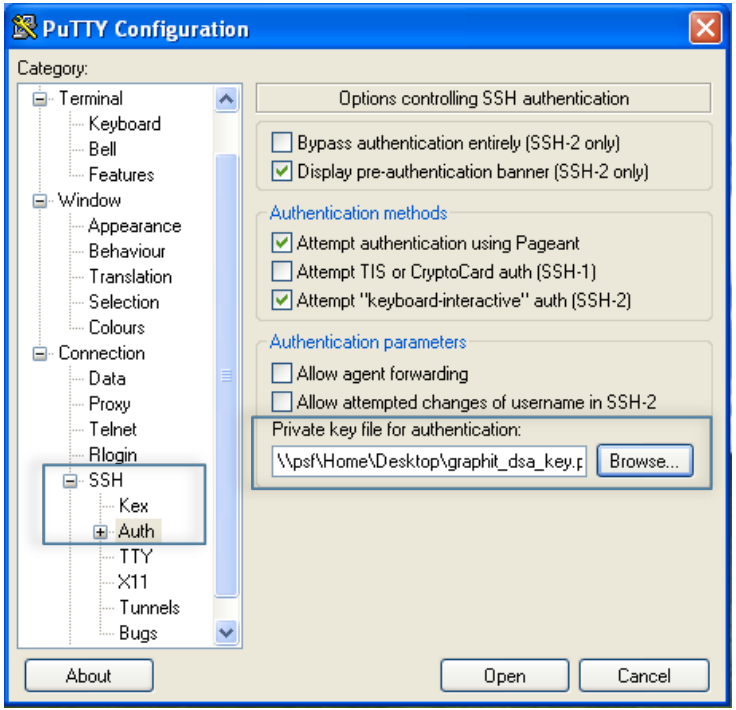
- **PuTTY** is a free (MIT-licensed) Win32 Telnet and SSH client
- Download link:

http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe

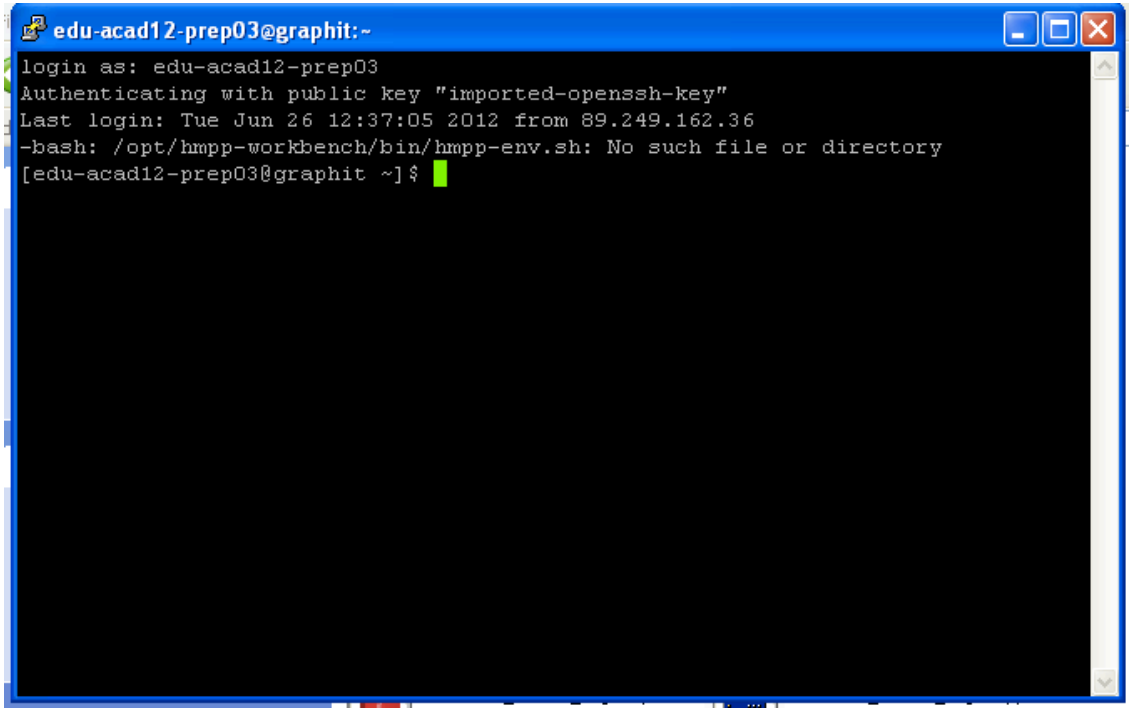
Windows - PuTTY



Windows - PuTTY



Windows - PuTYY



Передача данных

- **scp** – secure copy (remote file copy program)

  scp copies files between hosts on a network. It uses ssh for data transfer, and uses the same authentication and provides the same security as ssh.

scp **[-r]** [-i identitty_file] [-P port] [[user@]host1]:**file1** [[user@]host2]:**file2**

- -r - Recursively copy entire directories
- file1 and file2 can be **absolute paths or relative paths with filenames or directories' names**

Unix - scp

- **Copying a file from localhost to GraphIT!**

```
me:~ conqueror$ scp -i graphit_dsa_key Downloads/test.txt edu-acad12-prep03@graphit.parallel.ru:~/
```

- **Copying a file from GraphitIT to localhost**

```
me:~ conqueror$ scp -i graphit_dsa_key edu-acad12-prep03@graphit.parallel.ru test.cpp
```

- **Copying a directory to GraphIT**

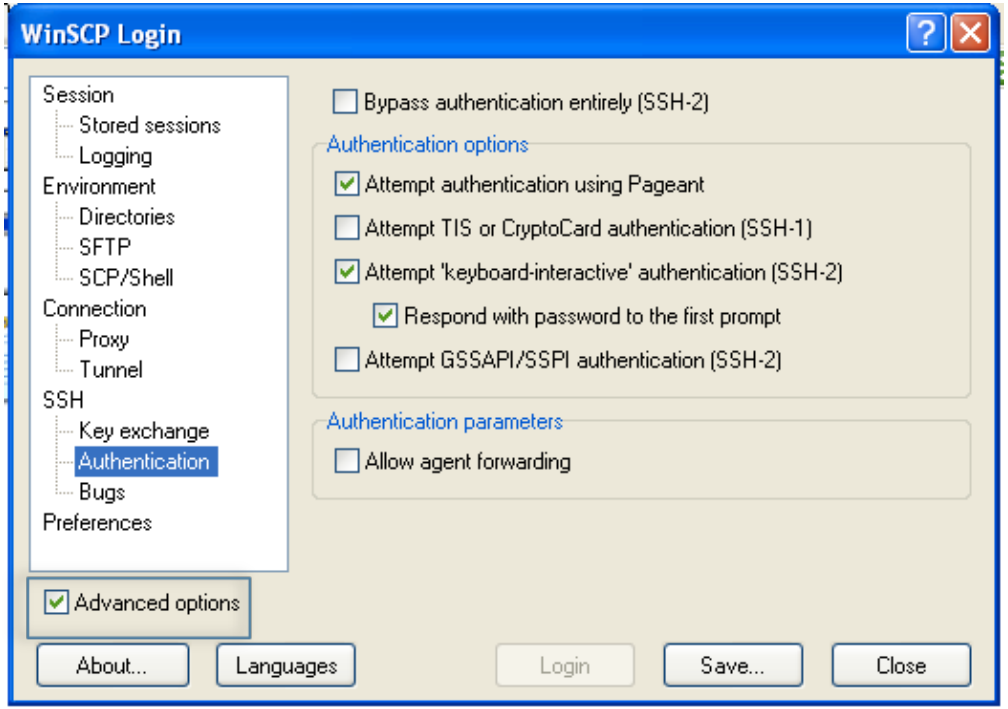
```
me:~ conqueror$ scp -r -i graphit_dsa_key Downloads/test.txt edu-acad12-prep03@graphit.parallel.ru:~/
```

Windows

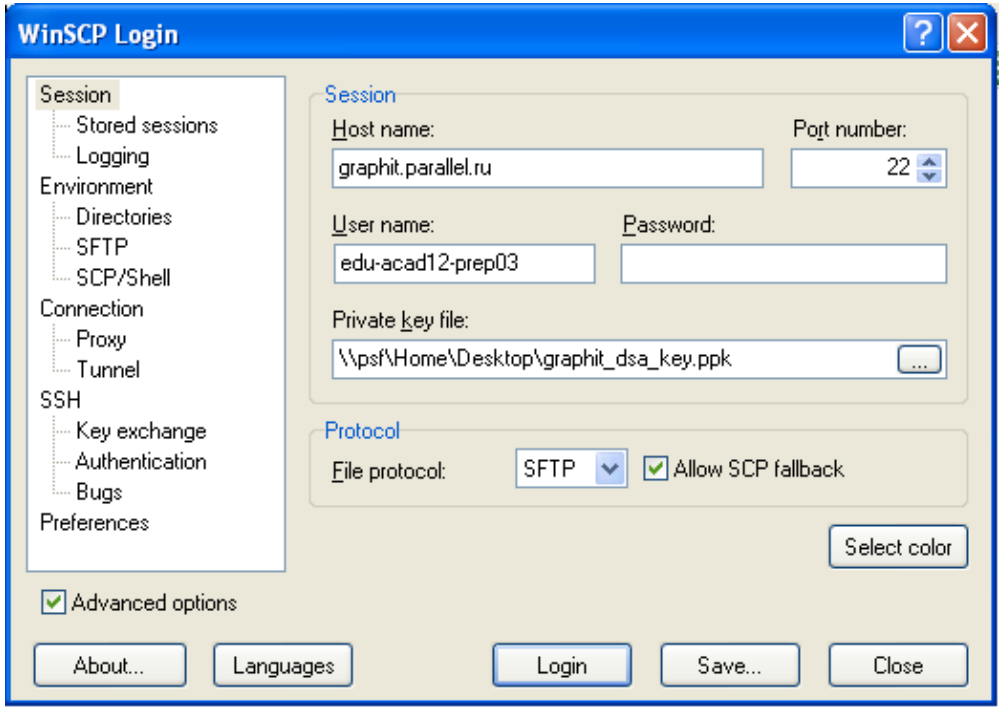
- **WinSCP** is an open source free SFTP client, SCP client, FTPS client and FTP client for Windows.
- Download link:

http://winscp.net/download/winscp438.zip

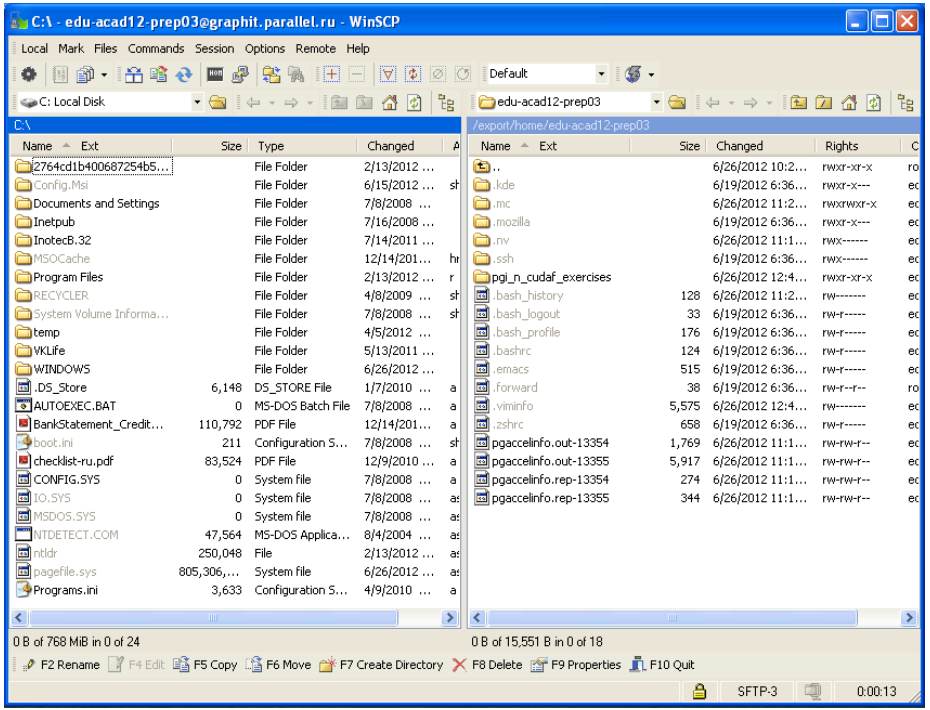
Windows - WinSCP



Windows - WinSCP



Windows - WinSCP



Редактеры

Vim – Vi IMproved

- **vim - Vi IMproved**, a programmers text editor
- **Enhancements:**
  - multi level undo
  - multi windows and buffers
  - syntax highlighting
  - command line editing
  - filename completion
  - on-line help
  - visual selection
- **Vim on-line help system** can be started
  - by typing "**:help**" command,
  - or by pressing **<HELP>** key or **<F1>** key instead.

Vim – General Modes

- **Normal** – For navigation and manipulation of text.
  - This is the default mode.
  - **To switch to Normal mode** – type **<ESC>**
- **Insert** - For inserting new text.

  From the Normal mode
  - **To append after cursor** – type **<a>**
  - **To insert before cursor** – type **<i>**
  - **To append at end of line** – type **<A>**
  - **To insert at beginning of line** – type **<I>**
- **Visual** - For navigation and manipulation of text selections, this mode allows you to perform most normal commands, and a few extra commands, on selected text.
  - **To switch to Visual mode** - from Normal mode type **<v>**
- **Command-line** - For entering editor commands.
  - **To switch to Command-line mode** - from Normal mode type **<:>**
  - **To exit saving changes** – type **<wq!> <Enter>**
  - **To exit without saving changes** - type **<q!> <Enter>**

Vim – Basic Commands

- **Moving around**:
  - **<h>** moves the cursor one character left
  - **<j>** moves the cursor one character down
  - **<k>** moves the cursor one character up
  - **<l>** moves the cursor one character right
- **Deleting (Normal mode)**:
  - **<x>** deletes one character
  - **<dw>** deletes one word
  - **<dd>** deletes one line
- **Selecting, Cutting, Copying and Pasting (Visual mode):**
  - **To select a text** - enter Normal mode, move the cursor to the first position of the text, enter Visual mode, and move the cursor to the last position of the text
  - **To cut selected text** – type **<x>**
  - **To copy selected text** – type **<c>**
  - **To paste selected text** after the position of the cursor – type **<p>**
- **Undo and Redo (Normal mode)**:
  - **To undo last change** – type **<u>**
  - **To redo last undone change** – type **<.>**

MC – Midnight Commander

- If Vim is too complicated, use **MC** instead!
- **The Midnight Commander (MC)** is a directory browser/ file manager for Unix-like operating systems with built-in text editor.

Basic functional keys:

- <F1> - help
- <F2> - user menu
- <F3> - view
- <F4> - edit
- <F5> - copy
- <F6> - move
- <F7> - new dir
- <F8> - delete
- <F9> - MC menu
- <F10> - quit

Компиляция

Компиляция CUDA-программ

- Компиляция программ, использующих только технологию CUDA и написанных без MPI, осуществляется при помощи компилятора nvcc.

  nvcc -o myprog myprog.cu

- Компиляция программ, использующих CUDA+MPI, используется при помощи утилиты mpicc (для C-программ) и mpicxx (для C++-программ). В качестве компилятора по умолчанию настроен nvcc, который поддерживает файлы на языках C, C++ и CUDA C.

  mpicc -o myprog myprog.c myprog.cu

- В данный момент на кластере установлен GPU Computing SDK 4.1. Он доступен в /export/opt/gpusdk41

Компиляция OpenCL-программ

- OpenCL-программы компилируются точно так же, как и любые другие.
- Текст собственно OpenCL-ядра передаётся во время исполнения как строка, конкретный механизм передачи определяется программистом.
- Поскольку в качестве компилятора по умолчанию mpicc/mpicxx указан nvcc, требуется явно указать gcc/g++:

```
OMPI_CC=gcc mpicc -lOpenCL -o myprog myprog.c
OMPI_CXX=g++ mpicxx -lOpenCL -o myprog myprog.cpp
```

Компиляция OpenACC-программ

- C программы:

  pgcc -acc -Mcuda -Minfo=all -ta=nvidia –o myprog myprog.c

- Fortran программы:

  pgfortran -acc -Mcuda -Minfo=all -ta=nvidia –o myprog myprog.f90

Запуск

Запуск программ

- По умолчанию программа запускается из расчёта 1 процесс на 1 ГПУ. Для запуска программы необходимо выполнить следующую команду:

  mpirun -np <число-ГПУ> <имя-программы> <аргументы>

  либо

  cleo-submit -np <число-ГПУ> <имя-программы> <аргументы>

Запуск программ

- В настоящее время отсутствует возможность прикреплять конкретные процессы к конкретным ГПУ, поэтому в начале работы процессу нужно будет указать правильный номер ГПУ.
- Гарантируется, что на каждый узел назначается по 3 процесса с номерами (MPI_commRank) вида 3*i, 3*i+1 и 3*i+2, где i - целое число. Таким образом номер ГПУ (numGpu) можно получить как остаток от деления ранга на 3.
- ГПУ на узлах сконфигурированы на использование в монопольном режиме, так что попытка использовать ГПУ, который уже используется в другом процессе, закончится ошибкой.
- Способ установки правильного номера ГПУ зависит от используемой технологии программирования:

  В CUDA-программах: cudaSetDevice(numGpu)

  В OpenCL-программах: выбрать устройство с номером numGpu из тех, что возвращаются функцией clGetDeviceIDs()

  В NUDA-программах: Nuda.Device.currentIndex = numGpu;

Запуск программ

- Возможен режим, в котором на узел (12 ядер, 3 ГПУ) запускается 1 MPI-процесс, в распоряжение которого отдаются все ресурсы узла, и явно устанавливать номер ГПУ нет необходимости. Для этого надо выполнить следующую команду:

  mpirun -as single -np <число-узлов> <имя-программы> <аргументы>

- В частном случае, когда приложению для работы достаточно одного узла:

  mpirun -as single -np 1 <имя-программы> <аргументы>

http://www.parallel-computing.pro

e-mail: dn@parallel-computing.pro

Thank you! ☺