

# Кластер САФУ

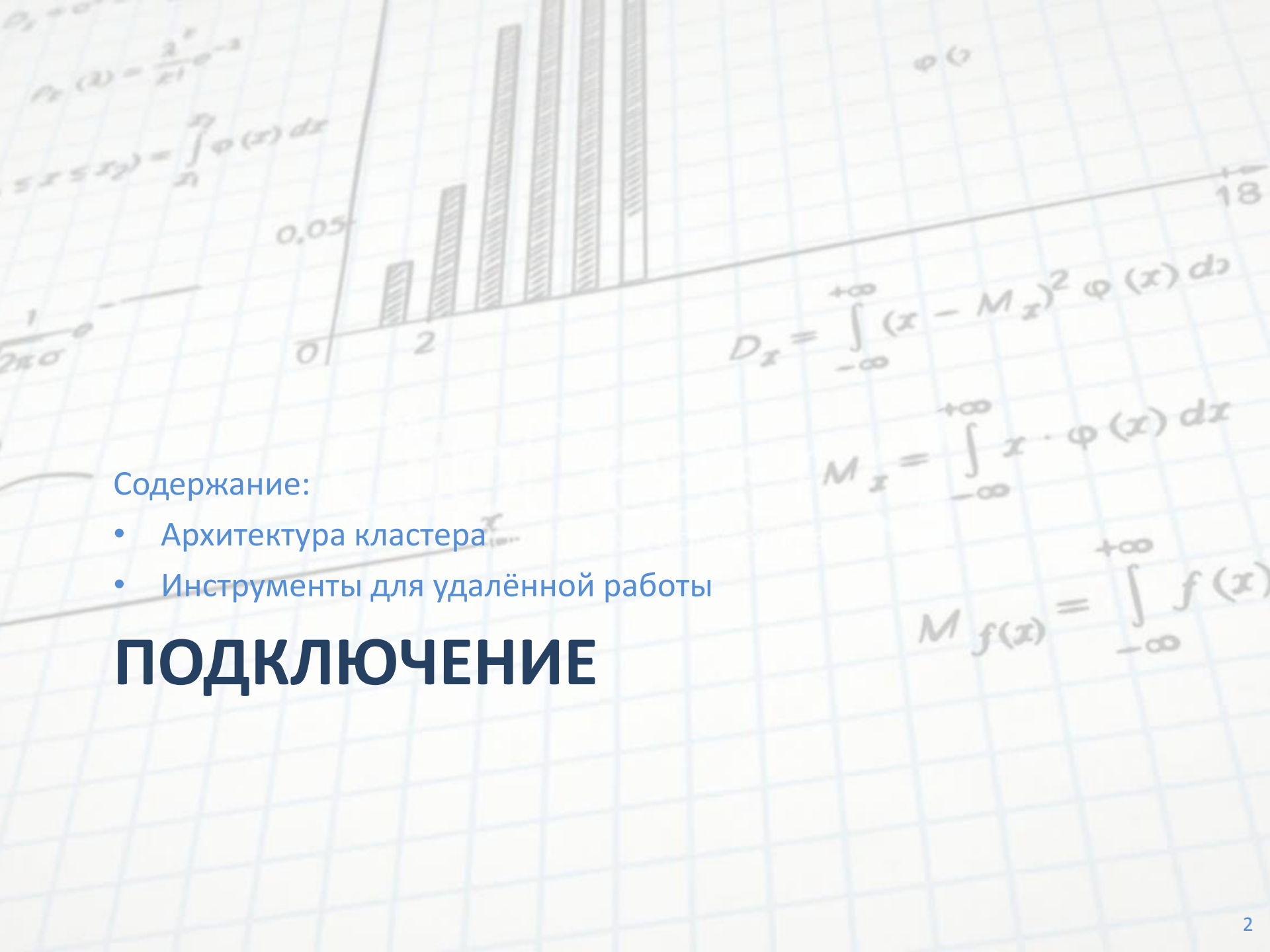
Подключение и работа с МРІ



Содержание:

- Архитектура кластера
- Инструменты для удалённой работы

# ПОДКЛЮЧЕНИЕ



# Архитектура



Сеть университета

ИМИКТ

fujitsu-hpc-01

Кластер

12 выч. Узлов  
2 x Intel Xeon (10 Core), 64 GB

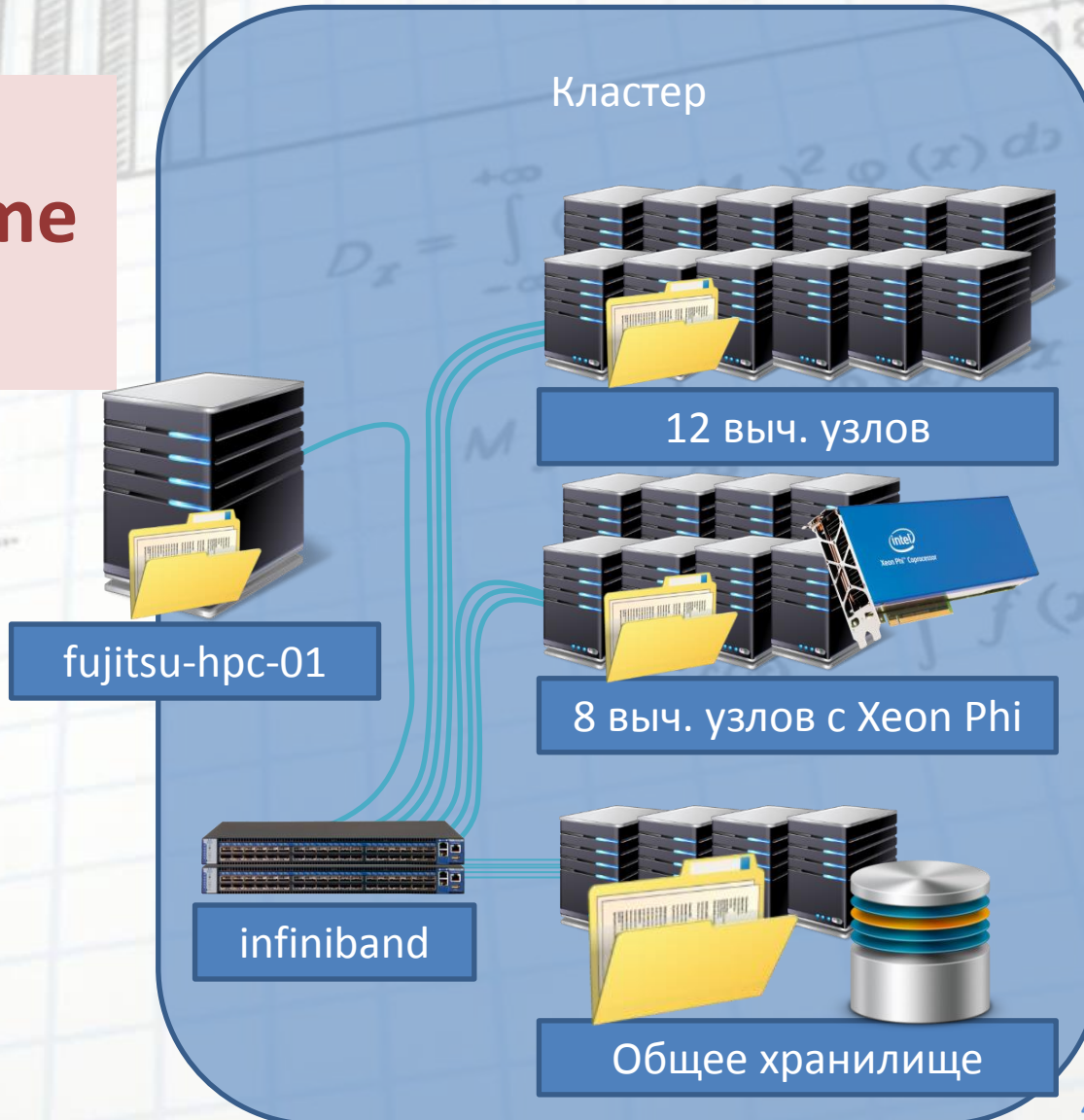
8 выч. узлов с Xeon Phi

infiniband

Общее хранилище

# Файловая система

- На всех узлах директория /home является общей!



# Подключение

- Центральный узел:

- SSH: [fujitsu-hpc-01.narfu.ru](https://fujitsu-hpc-01.narfu.ru)

- Инструменты:

- PuTTY – SSH-клиент

- командная строка

- WinSCP – SCP-клиент

- работа с файлами

- Xming – X-сервер

- работа с графическими приложениями (например, некоторые компоненты ANSYS)

# PuTTY

## Создайте новый профиль:

- Host: `fujitsu-hpc-01.narfu.ru`
- Port: `22`
- Connection type: `SSH`

## Установите поддержку Unicode:

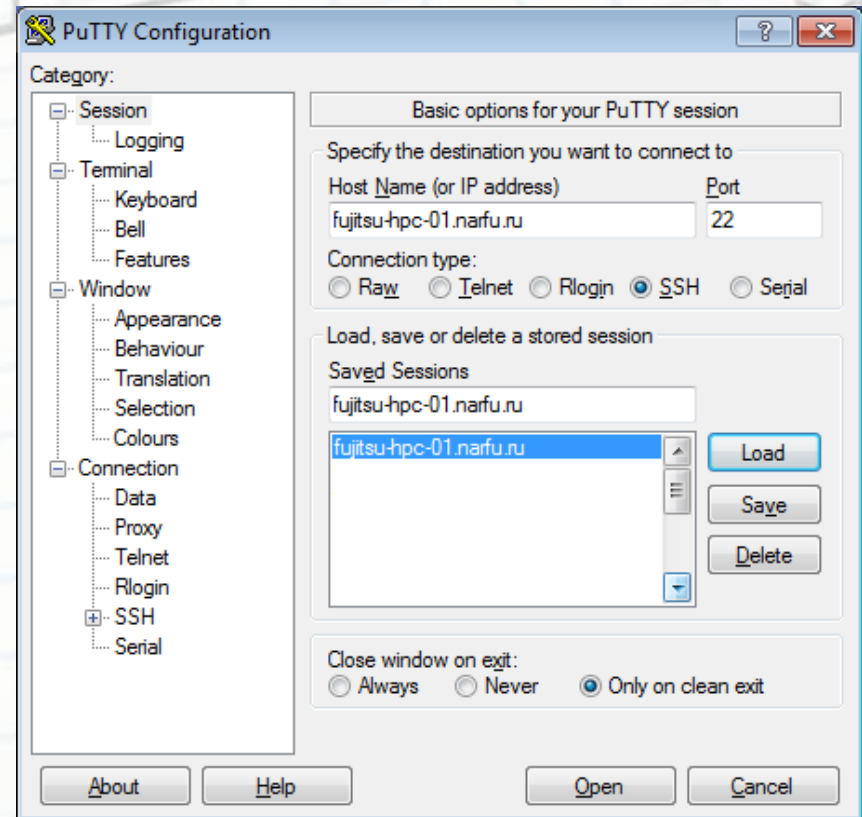
### Window\Translation:

- Remote character set: `UTF-8`

## Если нужна поддержка приложений с графическим интерфейсом, установите:

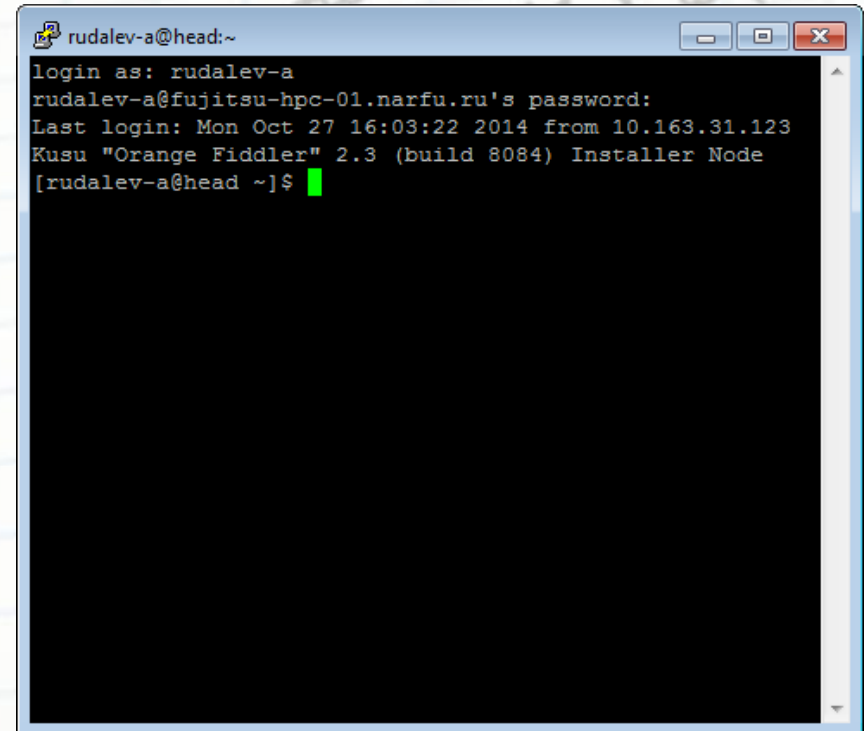
### Connection\SSH\X11:

- Enable X11 forwarding



# РuTTY: вход в систему

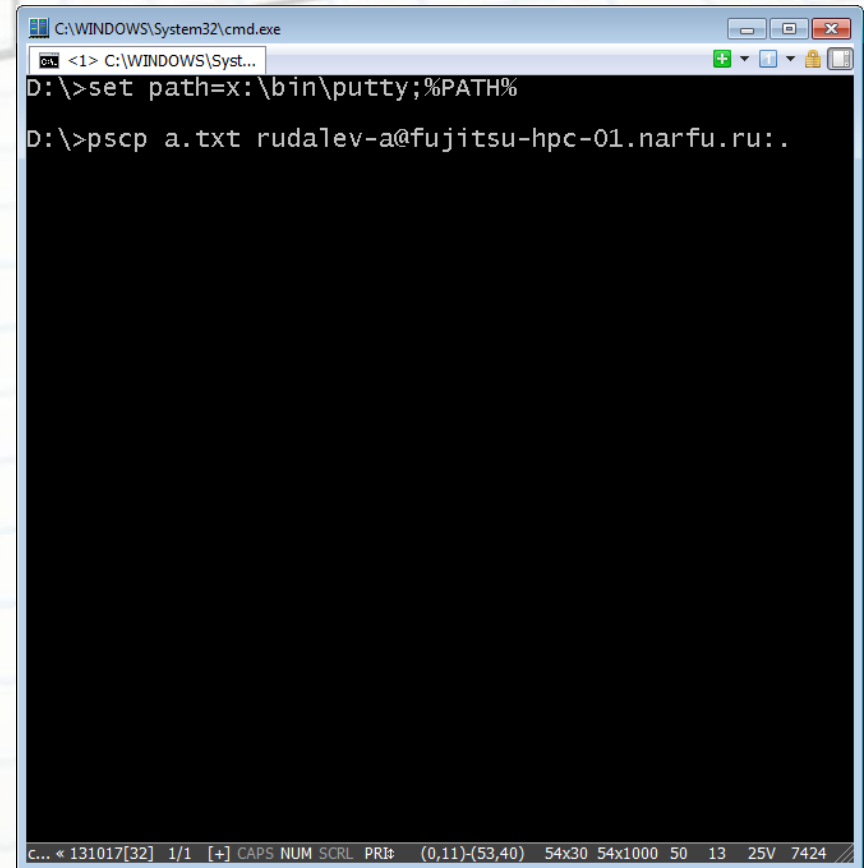
- Введите имя пользователя
- Введите пароль  
Внимание: при вводе пароля символы не печатаются на экране!!!
- В приглашении «командной строки» указывается имя пользователя и имя узла.
- Внутреннее имя головного узла: **head**



```
rudalev-a@head:~  
login as: rudalev-a  
rudalev-a@fujitsu-hpc-01.narfu.ru's password:  
Last login: Mon Oct 27 16:03:22 2014 from 10.163.31.123  
Kusu "Orange Fiddler" 2.3 (build 8084) Installer Node  
[rudalev-a@head ~]$
```

# Копирование файлов: scp

- **scp** – стандартная UNIX-программа для удалённого копирования файлов по протоколу SSH
- **pscp** – реализация scp для MS Windows в пакете PuTTY
- **Рекомендация:** используйте программу **ConEmu** при работе в командной строке MS Windows



```
C:\WINDOWS\System32\cmd.exe
C:\WINDOWS\System32\cmd.exe
D:\>set path=x:\bin\putty;%PATH%
D:\>pscp a.txt rudalev-a@fujitsu-hpc-01.narfu.ru:.
```

The screenshot shows a Windows command prompt window titled "C:\WINDOWS\System32\cmd.exe". The prompt is at "D:\>". The user has entered the command "set path=x:\bin\putty;%PATH%" and then "pscp a.txt rudalev-a@fujitsu-hpc-01.narfu.ru:". The status bar at the bottom shows "C:\WINDOWS\System32\cmd.exe" and "131017[32] 1/1 [+ CAPS NUM SCRL PRI: (0,11)-(53,40) 54x30 54x1000 50 13 25V 7424".



# Копирование файлов: WinSCP

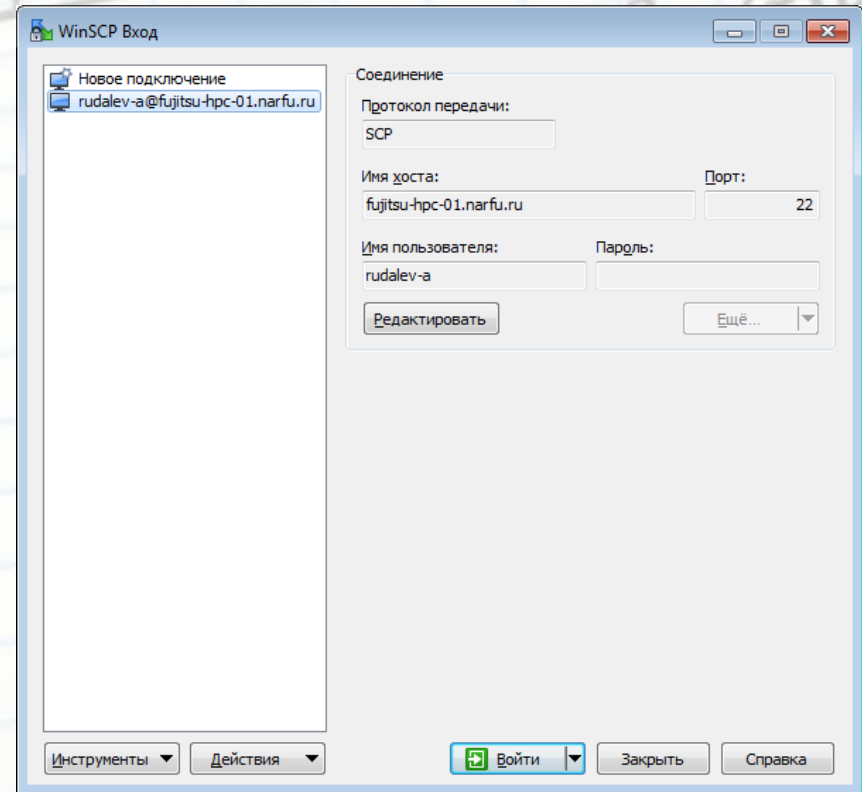
- **WinSCP** – реализация SFTP/FTP/SCP графического клиента для MS Windows

## Необходимо указать:

- Протокол передачи: **SCP**
- Имя хоста:  
**fujitsu-hpc-01.narfu.ru**
- Имя пользователя: **Ваш  
ЛОГИН**

## Не указывайте пароль:

- тогда он будет запрашиваться при подключении



# Копирование файлов: WinSCP

The screenshot displays the WinSCP interface with two panels. The left panel, titled 'D:\cluster', shows a local file system with the following table:

Имя	Расширение	Размер	Тип	Изменено
..			Родительская па...	29.10.2014 10:25:22
a.txt		0 B	Файл "TXT"	29.10.2014 10:26:04
It is My computer.txt		0 B	Файл "TXT"	29.10.2014 10:26:04

The right panel, titled '/home/rudalev-a/cluster', shows a remote file system with the following table:

Имя	Расширение	Размер	Изменено	Права	Владел...
..			29.10.2014 11:25:40	rw-r-----	rudalev...
a.txt		0 B	29.10.2014 11:26:04	rw-r--r--	rudalev...
It is cluster.txt		0 B	29.10.2014 11:26:04	rw-r--r--	rudalev...

At the bottom of the interface, the status bar shows '0 B из 0 B в 0 из 2' for both panels, a lock icon, 'SCP', and a timer '0:05:03'.

Содержание:

- Утилиты и инструменты
- Настройка окружения
- Система управления заданиями

# РАБОТА НА КЛАСТЕРЕ

# Редактирование файлов: nano

- **nano** – консольный текстовый редактор
- Запуск: **> nano имя\_файла**
- Управление через сочетания клавиш **Ctrl+....** С.м. подсказку в низу
  - **Ctrl+O** – записать изменения, нужно подтвердить имя файла нажав **Enter**
  - **Ctrl+X** – выйти из программы
- Поддерживает подсветку синтаксиса, если она включена в файле **~/.nanorc**
  - например, для C/C++:  
**include /usr/share/nano/c.nanorc**
- Так же есть и классические **vi** и **emacs**

```
rudalev-a@head:~/cluster
GNU nano 2.0.9 File: a.txt Modified
nano – консольный текстовый редактор для Unix и Unix-подобных операционных систем,
основанный на библиотеке curses и распространяемый под лицензией GNU GPL.
В настоящее время включен в большинство дистрибутивов Linux по умолчанию
и в установке не нуждается.

Чтобы запустить nano, следует открыть терминал и выполнить:

> nano имя файла

^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell

rudalev-a@head:~/cluster
GNU nano 2.0.9 File: hello.cpp Modified

#include <iostream>

int main() {
    std::cout << "Hello World!" << std::endl;
    return 0;
}
```

# Файловый менеджер: mc

- **mc** – классический консольный двухпанельник для UNIX
- Запуск:  
> **mc**
- Просмотреть файл: **F3**
- Изменить файл: **F4**
- Скопировать: **F5**
- Переместить: **F5**
- Переименовать: **Shift+F6**
- Удалить: **F8**
- Выход: **F10**

```
mc [rudalev-a@head]:~/mpi-pi
Left File Command Options Right
<- ~/cluster .[^]> <- ~/mpi-pi .[^]>
'n Name Size Modify time 'n Name Size Modify time
/.. UP--DIR Oct 29 15:27 /.. UP--DIR Oct 29 15:27
It is cluster.txt 5 Oct 29 12:08 *build.sh 560 Mar 18 2014
a.txt 581 Oct 29 12:16 main.c 3116 Feb 11 2014
hello.c 37 Oct 29 12:17 mpi_pi.c 2825 Mar 19 2014
hello.cpp 92 Oct 29 12:29 *pi_intelmpi 9537 Mar 18 2014
 pi_intelmpi.pbs 168 Oct 22 16:20
*pi_openmpi 9946 Mar 19 2014
 pi_openmpi.pbs 185 Mar 18 2014
run_openmpi.pbs 162 Mar 19 2014

UP--DIR 1899G/1275G (148%) mpi_pi.c 1899G/1275G (148%)
Hint: You may specify the external viewer with the shell vars VIEWER or PAGER.
[rudalev-a@head mpi-pi]$
1Help 2Menu 3View 4Edit 5Copy 6RenMov 7Mkdir 8Delete 9PullDn10Quit
```

# Environment modules

- Средство управления переменными окружения
- Список активированных модулей:
  - > `module list`
- Список доступных модулей:
  - > `module avail`
- Активация модуля:
  - > `module load ИМЯ_МОДУЛЯ`
  - Используйте автодополнение по клавише Tab
- Замена модуля
  - > `module switch ЧТО НА_ЧТО`

# PBS

- Система управления заданиями
- Файл задания
- Просмотр очереди:
  - > `qstat`
- Помещение задания в очередь:
  - > `qsub` файл.pbs
- Удаление из очереди:
  - > `qdel` ID\_задания

# PBS: Пример задания

```
[rudalev-a@head ~]$ cat Hello.pbs <- Вывод содержимого  
файла
```

```
#!/bin/bash <- Кто исполнит скрипт  
#PBS -N HelloWorld <- Краткое название задачи  
echo Hello world! <- Команда
```

```
[rudalev-a@head ~]$ qsub Hello.pbs <- Запуск скрипта в PBS
```

```
809.Head <- ID задачи
```

```
[rudalev-a@head ~]$ cat HelloWorld.o809
```

```
Hello world! <- Вывод STDOUT
```

```
[rudalev-a@head ~]$ cat HelloWorld.e809
```

```
<- Вывод STDERR
```



# PBS: Пример задания

```
#!/bin/bash
#PBS -N HelloWorld
#PBS -l select=5          <- запросить 5
ядер
#PBS -l walltime=00:01:00 <- макс. время
работы

echo Hello world! I am main process
date +"Date: %Y-%m-%d"
echo Time: `date +%H:%M:%S.%N`
echo Main host: `/bin/hostname`
echo Can run on: `cat ${PBS_NODEFILE}`
echo - PBS Environment -----
export | grep PBS
echo -----
```

# PBS: Пример задания

```
[rudalev-a@head ~]$ qsub Hello.pbs
810.head

[rudalev-a@head ~]$ cat HelloWorld.o810

Hello world! I am main process
Date: 2014-10-29
Time: 14:25:50.516809978
Main host: cn20
Can run on: cn20.fujitsu-hpc cn20.fujitsu-hpc cn20.fujitsu-hpc cn20.fujitsu-hpc cn20.fujitsu-hpc
- PBS Environment -----
declare -x PBS_ENVIRONMENT="PBS_BATCH"
declare -x PBS_JOBCOOKIE="0000000024CC5E53000000001CB8383E"
declare -x PBS_JOBDIR="/home/rudalev-a"
declare -x PBS_JOBID="810.head"
declare -x PBS_JOBNAME="HelloWorld"
declare -x PBS_MOMPRT="15003"
declare -x PBS_NODEFILE="/var/spool/PBS/aux/810.head"
declare -x PBS_NODENUM="0"
declare -x PBS_O_HOME="/home/rudalev-a"
declare -x PBS_O_HOST="head.fujitsu-hpc"
declare -x PBS_O_LANG="en_US.UTF-8"
declare -x PBS_O_LOGNAME="rudalev-a"
declare -x PBS_O_MAIL="/var/spool/mail/rudalev-a"
declare -x PBS_O_QUEUE="workq"
declare -x PBS_O_SHELL="/bin/bash"
declare -x PBS_O_SYSTEM="Linux"
declare -x PBS_O_WORKDIR="/home/rudalev-a"
declare -x PBS_QUEUE="workq"
declare -x PBS_TASKNUM="1"
-----
```

# Ключи PBS

- **#PBS -l select=5**
  - Запросить 5 ядер для 5 процессов
  - Для большинства **MPI-приложений** его будет достаточно
  - PBS\_NODEFILE: `cn20.fujitsu-hpc cn20.fujitsu-hpc cn20.fujitsu-hpc cn20.fujitsu-hpc cn20.fujitsu-hpc`
- **#PBS -l select=5:ncpus=10**
  - Запросить по 10 ядер для 5 процессов (всего 50), каждый процесс будет использовать 10 потоков
  - Для **OpenMP-приложений** можно использовать: `-l select=1:ncpus=10`
  - PBS\_NODEFILE: `cn20.fujitsu-hpc cn20.fujitsu-hpc cn16.fujitsu-hpc cn16.fujitsu-hpc cn14.fujitsu-hpc`
- **#PBS -l nodes=5:ppn=2:ncpus=20**
  - Запросить по 2 ядра на 5 разных вычислительных узлах, на каждом узле будет использовано 20 ядер
  - Этот вариант используется когда необходимо **целиком зарезервировать узлы**, запретив запускать на них другие задачи.
  - PBS\_NODEFILE: `cn20.fujitsu-hpc cn20.fujitsu-hpc cn16.fujitsu-hpc cn16.fujitsu-hpc cn14.fujitsu-hpc cn14.fujitsu-hpc cn17.fujitsu-hpc cn17.fujitsu-hpc cn15.fujitsu-hpc cn15.fujitsu-hpc`

Содержание:

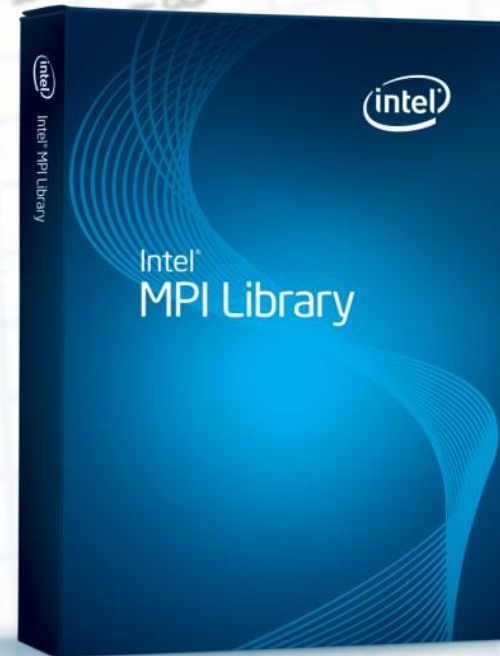
- Реализации MPI
- Настройка окружения
- Компиляция
- Запуск

# КОМПИЛЯЦИЯ И ЗАПУСК MPI-ПРИЛОЖЕНИЙ

# Реализации MPI

OpenMPI

Intel MPI



# Компиляция

- Настройка окружения:

- OpenMPI: > module load openmpi-pbs

- Intel MPI: > module load intel/mpi

- Компиляция

- Си: > mpicc файл.c -o исп\_файл

- C++: > mpicxx файл.cxx -o исп\_файл

- F77: > mpif77 файл.f77 -o исп\_файл

- F90: > mpif77 файл.f90 -o исп\_файл

# Требования к MPI-программе

- Запуск в пакетном режиме через систему PBS
- Запуск MPI-приложений разрешён только на вычислительных узлах
- Для `mpirun` не надо указывать ключи определяющие количество потоков. Все данные будут взяты из переданных переменных окружения
- Не забывайте в скрипте запуска переходить в рабочую директорию и заново определять переменные окружения, например, через `modules load`

# Hello World: пример на Си

```
/* C Example
   file: mpi_hello.c
*/
#include <stdio.h>
#include <mpi.h>

int main (argc, argv)
    int argc;
    char *argv[];
{
    int rank, size;

    MPI_Init(&argc, &argv); // starts MPI
    MPI_Comm_rank(MPI_COMM_WORLD, &rank); // get current process id
    MPI_Comm_size(MPI_COMM_WORLD, &size); // get number of processes
    printf( "Hello world from process %d of %d\n", rank, size );
    MPI_Finalize();
    return 0;
}
```



# Hello World: PBS-скрипт

```
#!/bin/bash
#PBS -N mpi_hello
#PBS -l select=10
#PBS -l walltime=00:05:00

cd ${PBS_O_WORKDIR}
module load openmpi-pbs

mpirun mpi_hello
```

# Hello World: Компиляция и запуск

- Подключите OpenMPI
  - > `module load openmpi-pbs`
- Создайте файлы `mpi_hello.c` и `mpi_hello.pbs`:
  - > `nano mpi_hello.c`
  - > `nano mpi_hello.pbs`
- Скомпилируйте:
  - > `mpicc mpi_hello.c -o mpi_hello`
- Запустите:
  - > `qsub mpi_hello.pbs`
  - 888.head
- Посмотрите результат:
  - > `cat mpi_hello.o888`

Содержание:

- Запуск offload-приложения на Intel Xeon Phi

# КОМПИЛЯЦИЯ И ЗАПУСК ПРИЛОЖЕНИЙ ДЛЯ XEON PHI

# Xeon Phi: offload example ¼ main.cpp

```
#include <stdio.h>
#include <omp.h>

#pragma offload_attribute(push, target(mic))
int getThreadCount()
{
    int thread_count;

    #pragma omp parallel
    {
        #pragma omp single
            thread_count = omp_get_num_threads();
    }
    return thread_count;
}
#pragma offload_attribute(pop)
...
```

# Xeon Phi: offload example 2/4

## main.cpp

```
int main()
{
    printf("Intel CPU:\n");
    printf("Number of threads on CPU: %d\n", getThreadCount());

    int number_of_coprocessors =
        _Offload_number_of_devices();

    printf("Intel Xeon Phi:\n");
    printf("Number of coprocessors: %d\n",
        number_of_coprocessors);

    for (int i = 0; i < number_of_coprocessors; ++i)
    {
        int thread_count;
        #pragma offload target(mic:i)
        {
            thread_count = getThreadCount();
        }
        printf("Number of threads on MIC%d: %d\n", i, thread_count);
    }

    return 0;
}
```

# Xeon Phi: offload example 3/4

```
#!/bin/bash
#PBS -N lab1-mics
#PBS -l select=1:ncpus=20:nmics=1
#PBS -l walltime=00:05:00

cd ${PBS_O_WORKDIR}
echo Main host: ` /bin/hostname `
export OMP_NUM_THREADS=
module load intel/composer_xe
module load intel/mpi

./lab1_thread_test
```

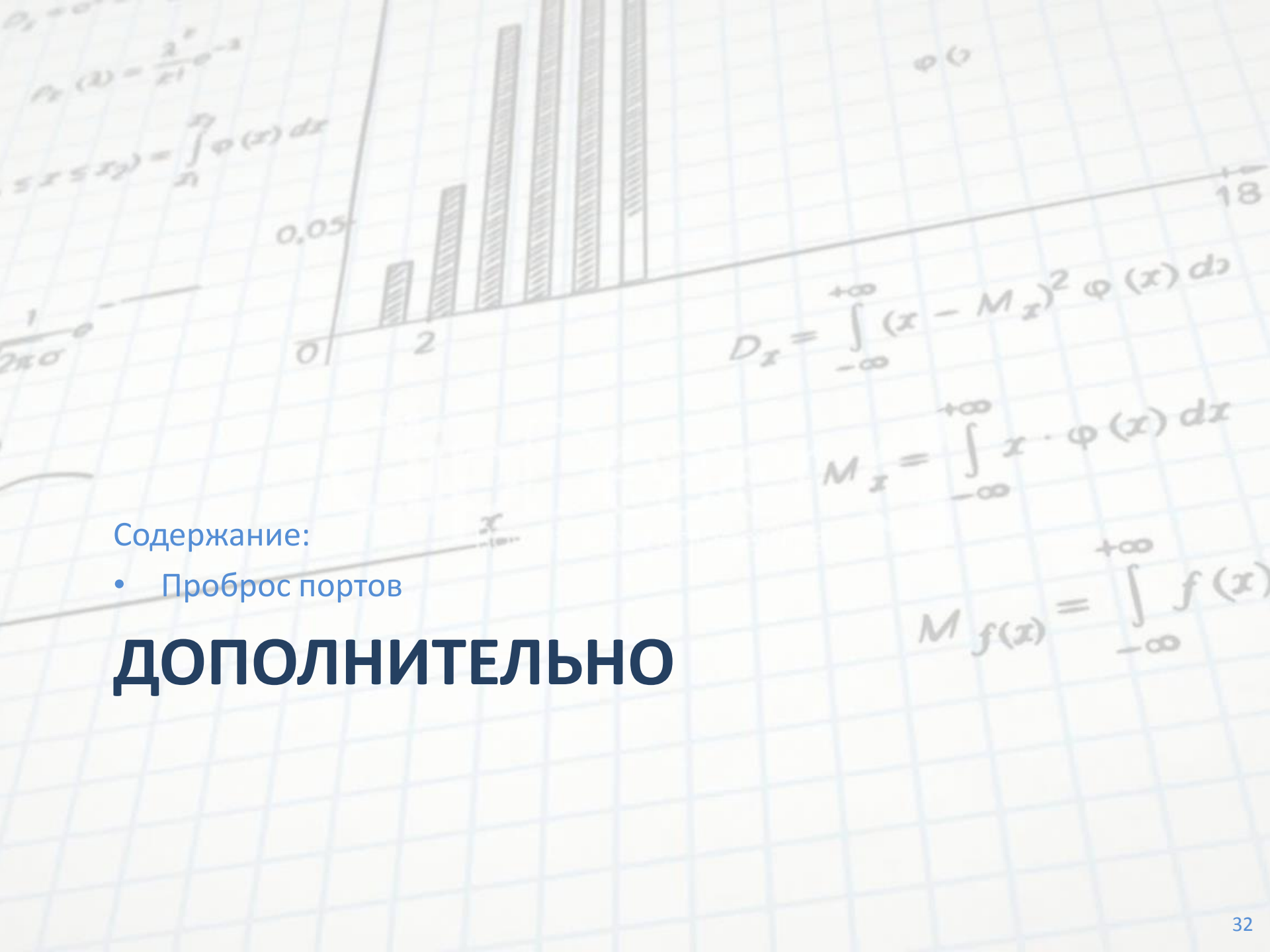
## Xeon Phi: offload example 4/4

```
$ module load intel/composer_xe
$ module load intel/mpi
$ icc -O2 -openmp main.cpp -o
lab1_thread_test
```

Содержание:

- Проброс портов

# ДОПОЛНИТЕЛЬНО



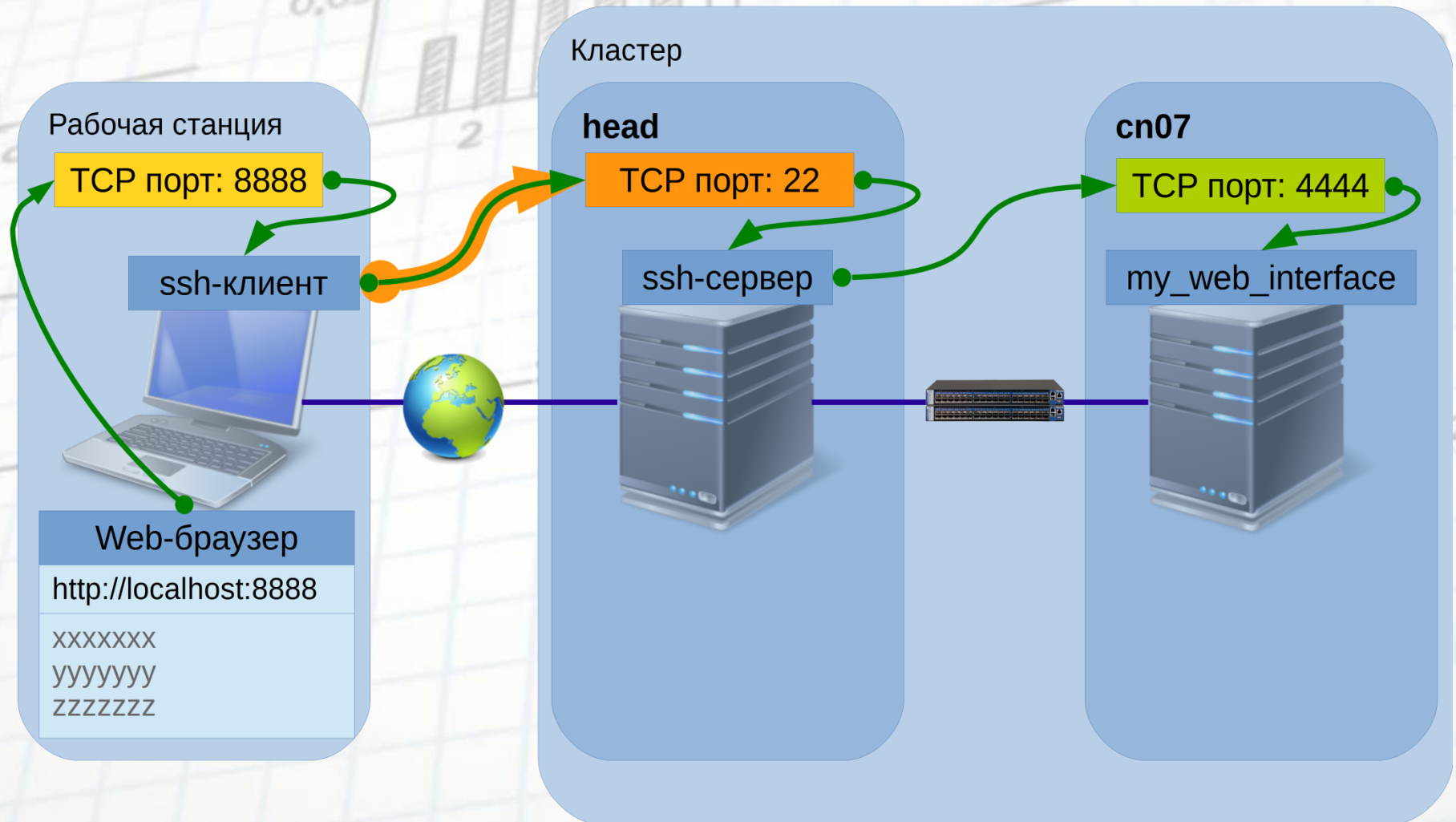


# Проброс портов: Задача

- **Пример:** На кластере запущена задача, которая имеет Web-интерфейс для мониторинга результатов. Он предоставляется процессом `my_web_interface`, работающим на 4444 tcp-порте вычислительного узла `cn07`.
- **Необходимо:** С рабочего компьютера пользователя подключится к этому web-интерфейсу.
- **Проблема:** Если в web-браузере на компьютере пользователя набрать адрес <http://cn07:4444/> - то подключения не будет (вычислительный узел спрятан за головным узлом кластера).
- **Решение:** Запустить на рабочем компьютере посредника, который откроет локальный tcp-порт (например, 8888) и будет через ssh-соединение к головному узлу кластера «соединит» его с портом 4444 вычислительного узла `cn07`. Тогда в адресной строке Web-браузера на рабочем компьютере можно ввести <http://localhost:8888/>

# Проброс портов

```
*nix: ssh -N -L 8888:cn07:4444 user@head
win: plink -N -L 8888:cn07:4444 user@head
```





Контактное лицо:

Рудалёв Александр Васильевич <[a.rudalev@narfu.ru](mailto:a.rudalev@narfu.ru)>

**В ДОБРЫЙ ПУТЬ!**