

ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ
ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ “СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИИ И ИНФОРМАТИКИ”

Кафедра ВС

Лабораторная работа № 1
«РАСПРЕДЕЛЁННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ»

Выполнил:
ст. гр. МГ-165 Марков В.А.
Проверил:
Фульман В.О.

Новосибирск 2016

Задание на лабораторную работу

1. Подключитесь к ресурсу `ssh://jet.cpct.sibsutis.ru:22`. После первого подключения измените пароль для своей учетной записи.

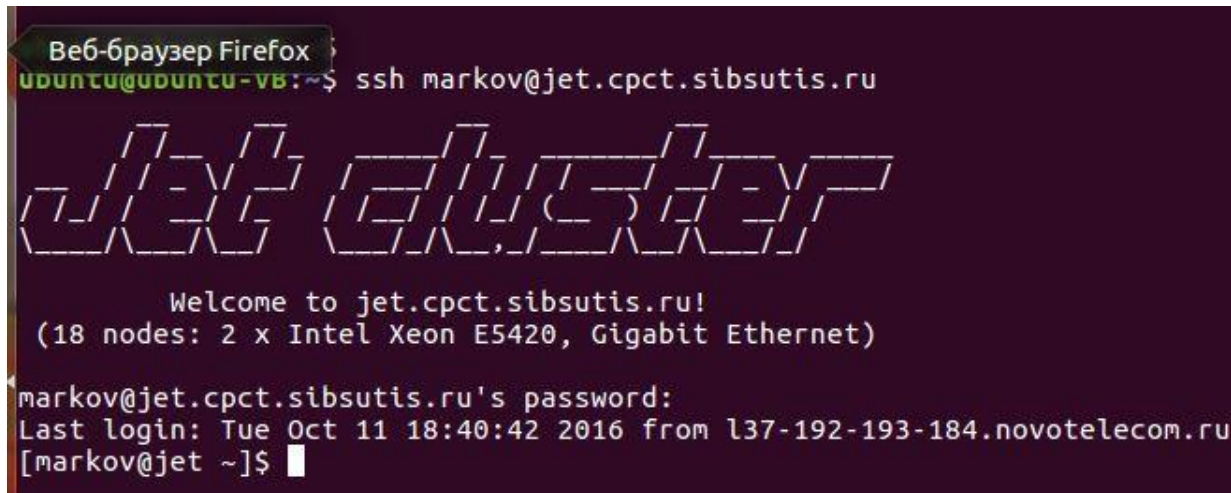


Рисунок 1 - Подключение

2. Подготовьте программное обеспечение, реализующее алгоритм умножения двух прямоугольных матриц целых чисел. Размеры матриц задаются параметрами командной строки. Исходные матрицы генерируются псевдослучайным образом (стандартный генератор). Исходные матрицы и результат их перемножения выводятся в стандартный поток вывода. Язык программирования и средства разработки, с использованием которых будет реализовано программное обеспечение, выбираются из числа доступных в распределённой вычислительной системе Jet.

Листинг 1 - Умножение матриц

```
#include <iostream>
#include <vector>
#include <random>

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

using namespace std;

void generatingMatrix(vector<vector<int>> &a, int row, int col)
{
    std::default_random_engine generator;
    std::uniform_int_distribution<int> distribution(0, 100);

    cout << "Generating matrix:" << endl;

    for (auto i = 0; i < row; ++i) {
        vector<int> v(col);
        a.push_back(v);
    }
}
```

```

    }

    for (auto i = 0; i < row; ++i) {
        for (auto j = 0; j < col; ++j) {
            a[i][j] = distribution(generator);
        }
    }

    for (auto &i : a) {
        for (auto &j : i) {
            cout << j << " ";
        }
        cout << endl;
    }
}

void multiplyMatrix(vector<vector<int>>& a,
vector<vector<int>>& b, int m, int n, int q)
{
    vector<vector<int>> c;

    for (auto i = 0; i < m; ++i) {
        vector<int> v(q);
        c.push_back(v);
    }

    cout << "The new matrix is:" << endl;

    for (auto i = 0; i < m; ++i) {
        for (auto j = 0; j < q; ++j) {
            c[i][j] = 0;

            for (auto k = 0; k < n; ++k) {
                c[i][j] = c[i][j] + (a[i][k]*b[k][j]);
            }
            cout << c[i][j] << " ";
        }
        cout << endl;
    }
}

int main(int argc, char const *argv[])
{
    int m, n; // matrix 1 row, col
    int p, q; // matrix 2 row, col

    vector<vector<int>> matrix1;
    vector<vector<int>> matrix2;

    if (argc < 5) {
        cerr << "usage: m(row1) n(col1) p(row2) q(col2)" <<
endl;
        return -1;
    }

    m = atoi(argv[1]);
    n = atoi(argv[2]);
    p = atoi(argv[3]);
    q = atoi(argv[4]);

    generatingMatrix(matrix1, m, n);
    generatingMatrix(matrix2, p, q);

    multiplyMatrix(matrix1, matrix2, m, n, q);
}

```

```

        return 0;
    }

```

- Опишите задачу, требующую для своего решения один узел и одно ядро на нем. При решении задачи должно запускаться программное обеспечение, реализованное в п. 2. Размеры перемножаемых матриц задаются в виде трех чисел: М (количество строк в 1 таблице), N (количество столбцов в 1 таблице и строк во 2 таблице), Р (количество столбцов 2 таблицы), где М – линейный номер задач в её полном идентификаторе, N – удвоенное значение М, Р – длина строки, содержащей сетевое имя узла, на котором выполняется задача.

Листинг 2 - Конфигурация запуска

```

#PBS -N dcs_lab1
#PBS -l nodes=1:ppn=1
#PBS -j oe

cd $PBS_O_WORKDIR

#string="3434.jet.cluster.local"
string=$PBS_JOBID

col_m1=0

row_m1=$(echo $string | tr -dc '0-9')
let "col_m1 = 2 * row_m1"

foo=${string#$row_m1"."}
fool=${foo%%.*}

col_m2=${#fool}

echo "1 M "$row_m1
echo "1 N "$col_m1
echo "2 M "$col_m1
echo "2 P "$col_m2

cat $PBS_NODEFILE

./task $row_m1 $col_m1 $col_m1 $col_m2

```

- Поставьте задачу по умножению матриц в очередь. Посмотрите состояние очереди. Дождитесь завершения решения задачи и убедитесь в правильности выполнения программного обеспечения.

```
mc [markov@jet]:~/mpi_ft/mpi_fault_tolerance x
[markov@jet mpi_fault_tolerance]$ qsub task.job
6853.jet.cluster.local
[markov@jet mpi_fault_tolerance]$ qstat
Job ID              Name              User              Time Use S Queue
-----
6749.jet            Config            sobol             00:00:04 R debug
6853.jet            mpi_ft            markov            0 R debug
[markov@jet mpi_fault_tolerance]$
```

Рисунок 2 – Мониторинг исполнения задачи

5. Сконфигурируйте учетную запись в системе Jet так, чтобы при подключении по SSH использовалась пара открытый/закрытый ключ. Убедитесь, что теперь при подключении к системе, аутентификация происходит автоматически (без запроса паролей).

Листинг 3 – Конфигурация беспарольного доступа

```
ssh-keygen -t rsa
scp .ssh/id_rsa.pub markov@jet.cpct.sibsutis.ru:
ssh markov@jet.cpct.sibsutis.ru
"cat ./id_rsa.pub >> ./ssh/authorized_keys"
```

6. Используя технологию X11 forwarding запустите графический редактор gedit и создайте в домашнем каталоге на основном узле системы Jet текстовый файл, в котором разместите информацию о себе: номер группы, учетная запись, ФИО.

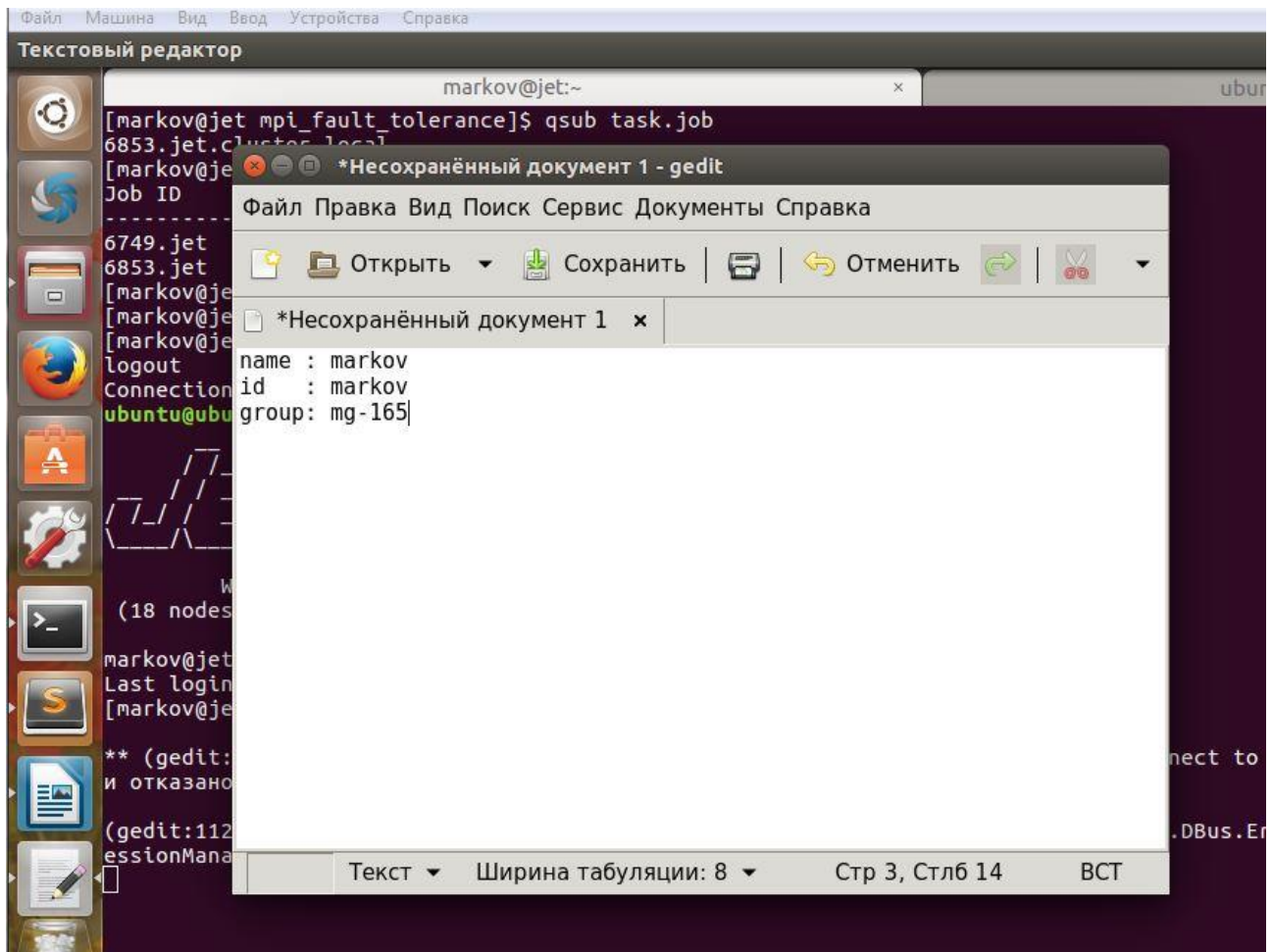


Рисунок 3 – Запуск «графический» приложений по ssh

7. Создайте описание задачи, требующей для своего решения два вычислительных узла и три вычислительных ядра на каждом. Запустите задачу в интерактивном режиме. После получения доступа к командной строке выведите на экран содержимое всех переменных среды окружения, начинающихся с префикса PBS_. Выведите на экран содержимое файла, в котором указаны имена узлов, выделенных для решения задачи. Используя команду mount определите каким образом сконфигурирована файловая система на вычислительном узле.


```
mc [markov@jet]:~/dcs/study/DCS/lab1
/home/students/mg165/markov/dcs/study/DCS/lab1/dcs_lab1.o6854
1 M 6854
1 N 13708
2 M 13708
2 P 3
cn4-comp
cn4-comp
cn3-comp
cn3-comp
Generating matrix:
0 13 76 46 53 22 4 68 68 94 38 52 83 3 5 53 67 0 38 6 42 69 59 93
63 89 27 44 77 48 24 27 36 16 49 90 91 6 91 50 52 32 99 49 26 9 9
30 35 51 59 85 41 84 27 41 54 47 29 18 15 57 81 3 53 50 96 75 56
9 50 14 59 85 59 96 56 14 99 41 14 57 25 49 46 97 12 20 32 63 12
35 45 81 94 65 21 68 91 25 86 47 51 60 82 76 46 96 63 44 83 69 70
72 49 67 68 20 92 87 89 54 14 45 99 21 45 31 51 89 44 47 81 36 21
8 39 96 95 39 27 69 28 78 79 42 28 19 1 19 99 24 82 13 40 60 17 81
5 59 33 71 14 16 49 86 82 56 74 31 13 53 31 59 52 43 26 37 39 45 4
67 34 54 52 83 1 27 64 55 92 26 97 24 85 70 46 82 2 16 71 71 44 58
63 66 71 54 28 82 37 69 31 0 10 52 78 22 5 7 16 73 29 5 48 46 52 4
19 35 62 61 12 55 96 36 84 98 26 25 42 82 90 24 76 70 45 26 67 11
67 68 76 3 23 22 56 65 62 37 22 23 65 16 75 16 4 66 12 95 48 62 9
12 53 16 52 10 84 37 42 33 72 58 88 76 84 70 39 94 61 36 54 45 40
7 42 18 97 13 80 95 17 61 6 35 29 51 83 84 73 33 81 63 4 4 15 77 6
59 83 33 23 54 43 40 99 97 85 87 31 14 100 64 47 94 69 82 54 57 10
```

Рисунок 4 – Результат работы

```
mc [markov@jet]:~/dcs/study/DCS/lab1
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,node=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,nodev,mode=755)
tmpfs on /sys/fs/cgroup type tmpfs (rw,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/lib/systemd/systemd)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpuacct,cpu)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
cgroup on /sys/fs/cgroup/net_cls type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb)
/dev/sda2 on / type ext4 (rw,relatime,data=ordered)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=42,pgrp=1,timeout=300,minproto=5,maxproto=1024)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
mqueue on /dev/mqueue type mqueue (rw,relatime)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime)
configfs on /sys/kernel/config type configfs (rw,relatime)
sunrpc on /proc/fs/nfsd type nfsd (rw,relatime)
/dev/md1 on /home type ext4 (rw,relatime,data=ordered,jqfmt=vfsv0,usrjquota=aquota.user)
/dev/md0 on /opt type ext4 (rw,relatime,data=ordered)
/dev/sda1 on /boot type ext4 (rw,relatime)
/dev/sda5 on /var type ext4 (rw,relatime,data=ordered)
sunrpc on /var/lib/nfs/rpc_pipefs type rpc_pipefs (rw,relatime)
/dev/sda3 on /scratch type ext4 (rw,relatime,data=ordered)
/dev/sda6 on /tmp type ext4 (rw,relatime,data=ordered)
/dev/sda2 on /var/named/chroot/etc/named type ext4 (rw,relatime,data=ordered)
/dev/sda2 on /var/named/chroot/etc/rndc.key type ext4 (rw,relatime,data=ordered)
/dev/sda2 on /var/named/chroot/usr/lib64/bind type ext4 (rw,relatime,data=ordered)
tmpfs on /var/named/chroot/run/named type tmpfs (rw,nosuid,nodev,mode=755)
[markov@jet lab1]$
```

Рисунок 5 – Конфигурация файловой системы

8. Разработайте распределённое программное обеспечение реализующее следующий функционал:
- Модуль отображения состояния распределённой вычислительной системы. Использует информацию, получаемую от первого модуля или загружаемую из локального файла (текущее состояние распределённой системы должно сохраняться в локальный файл). Интерфейс модуля отображения состояния распределённой системы значения не имеет (может быть текстовый, графический и т. п.)
 - Модуль вывода информации о состоянии вычислительных узлов системы Jet. Запуск модуля возможен только на внешнем узле. Состоянием узлов считаем значения полей: state, power_state и jobs.

```
mc [markov@jet]:~/dcs/study/DCS/lab1
[markov@jet lab1]$
[markov@jet lab1]$
[markov@jet lab1]$
[markov@jet lab1]$
[markov@jet lab1]$
[markov@jet lab1]$
[markov@jet lab1]$
[markov@jet lab1]$ ./info_nodes.py
cn1
    state = down
    power_state = Running
cn2
    state = down
    power_state = Running
    jobs = 0/6749.jet.cluster.local
cn3
    state = free
```

Рисунок 6 – Вывод конфигурации кластера

Node	State	Type
cn1	state power_state	Down Running
cn2	state power_state	Down Running
cn3	job	0/6749.jet.cluster.local
cn4	state power_state	Free Running
cn5	state power_state	Free Running
cn6	state power_state	Free Running

Рисунок 7 – Удаленный доступ к выводу конфигурации кластера