Bevezetés a Kommondor használatába

Előadó: dr. Környei László

Egyetemi docens (SZE, Matematika és Számítástudomány Tanszék)

HPC Szakértő (KIFÜ)

Online

2023. November 15.

SZÉCHENYI 2020

MAGYARORSZÁG KORMÁNYA

Európai Unió
Európai Regionális Fejlesztési Alap

BEFEKTETÉS A JÖVŐBE

Tervezett Napirend

- *08:45* *Online Csatlakozás*
- **09:00** **Köszöntő**
- 09:05 Hardver környezet. Hozzáférés. Adatátvitel. Modul környezet, telepített fordítók és fordítás.
- *10:00* *Kávészünet*
- 10:10 Az ütemező használata. Jobscript készítése, futtatása. Többszálas programok fordítása, futtatása jobscript segítségével.
- 11:00 Kérdések és válaszok
- ***11:30*** ***Ebédszünet***
- 12:30 Többfolyamatos programok fordítása és futtatása jobscript segítségével. Közösen használt számítógépek használatának etikettje. Futásidőbecslés.
- *13:30* *Kávészünet*
- 13:40 GPU-t használó programok fordítása és futtatása jobscript segítségével.
- 14:30 Kérdések és válaszok
- **16:00** **Zárás**

Köszönet

- Szervezés:
  - Tóth-Bíró Ágnes, Horváth Erzsébet (KIFÜ)
- Technikai segítség:
  - Tamás István, Debreczeni Attila (KIFÜ)
- Tréning és diák:
  - https://docs.hpc.kifu.hu/index.html (KIFÜ)
  - Felhasználói kézikönyv: https://portal.hpc.kifu.hu/files/Felhasznaloi_kezikonyv_1109.pdf

- Ha kérdés van:
  - Elakadásnál kapcsoljuk be a mikrofont s kérdezzünk bele nyugodtan!
  - Kérdéseket írjuk a cset ablakba, ezeket a kérdések szekcióban megválaszoljuk!
- Cél:
  - Gyors betekintést nyerjünk a Komondor használatába
  - Programok fordításáról, futtatásáról gyakorlati áttekintést kapjunk
- Forma:
  - Rövid elméleti háttér és gyakorlati bemutató
  - Közös feladatmegoldás és megoldások átbeszélése
  - Foglalkozzunk pár percet a feladattal, mielőtt kérdezünk!
- Visszajelzés
  - Kérek mindenkit, töltsük ki a visszajelzést
  - https://konferencia.kifu.hu/event/38/surveys/23

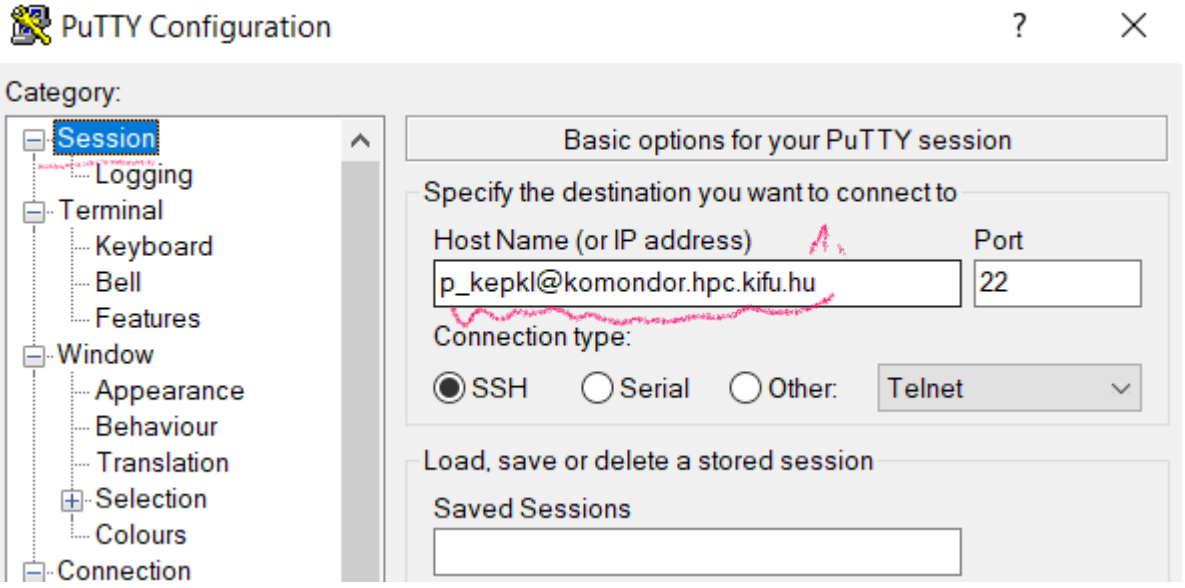
Bejelentkezés

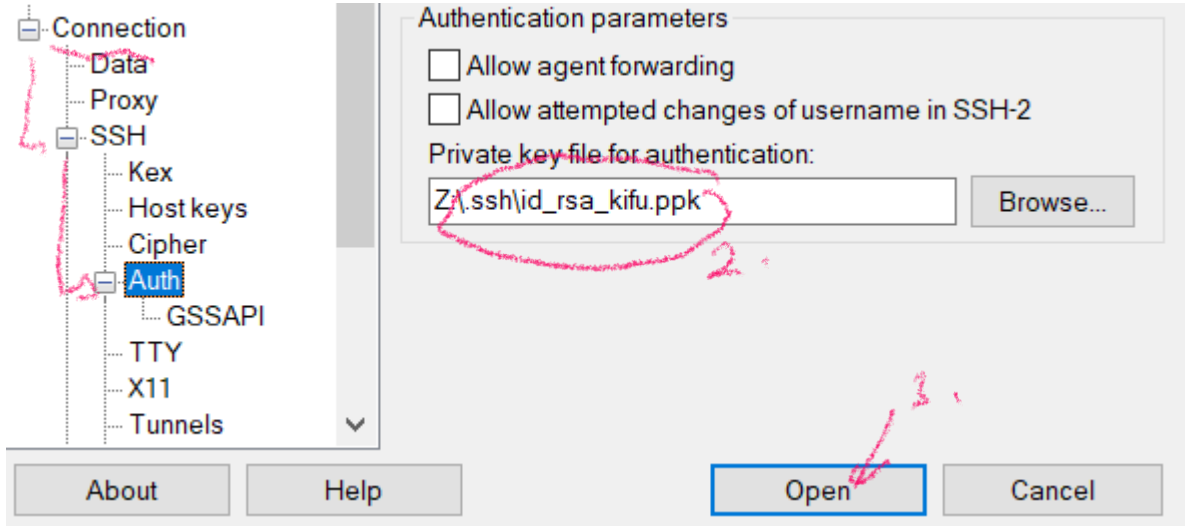
A Komondor elérése - Jelentkezzünk be!

- Linux

```
leslie@ tifa   ~ $ ssh -i ~/.ssh/id_rsa p_kepkl@komondor.hpc.kifu.hu
(p_kepkl@komondor.hpc.kifu.hu)
Komondor requires two factor authentication. Please login with eduID on this URL in your browser:
https://ack.hpc.kifu.hu/VYXJ0TRi
then press ENTER!
```

- Windows






```
| Komondor requires two factor authentication.
> URL in your browser:
| https://ack.hpc.kifu.hu/LEaUTWtr
| then press ENTER!
```

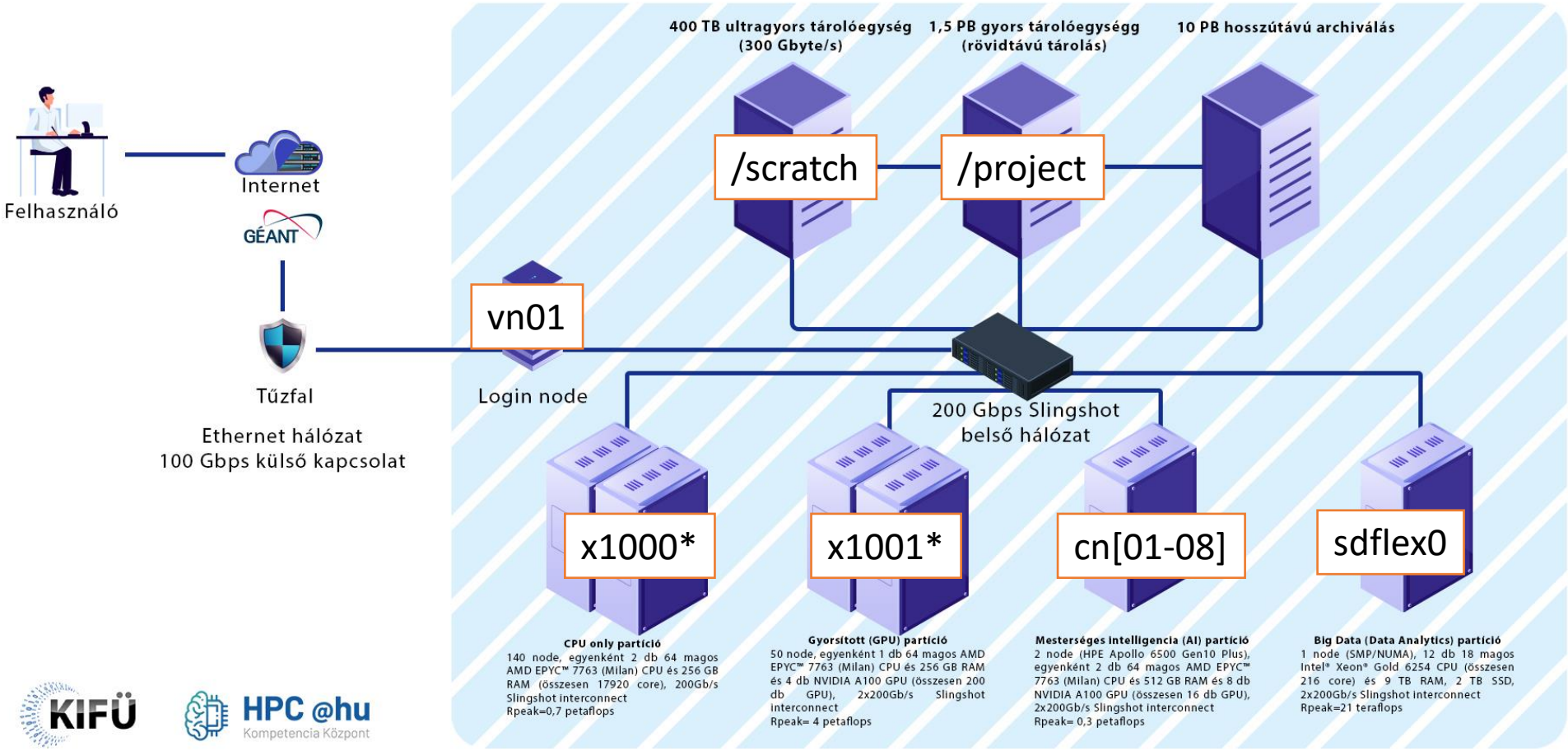
Hardver

A Komondor

- 2021-ben kezdődött a kialakítása
- 2023-ban adták át
- Debrecenben található
- A világ 199-edik leggyorsabb szuperszámítógépe
- 6 PetaFlops a teljesítménye
- https://www.top500.org/system/180079/

Komondor partíciók

A KOMONDOR SZUPERSZÁMÍTÓGÉP FELÉPÍTÉSE



KIFÜ HPC @hu Kompetencia Központ

A Komondor tárhelyei

Név	*scratch*	*project*	*tape*	*home*
Méret	400TB	1,5PB	10PB	
Megnevezés	Ultragyors SSD alapú tárhely	Gyors HDD alapú tárhely	Hosszútávú archiváló	
Típus	Lustre	Lustre	DMF + Spectra Tfinity	
Elérés	/scratch/<project_id>	/project/<project_id>		/home/<user_id>
Területi kvóta	1TB	4TB		20GB
Inode kvóta	300k	1M		100k
Bővíthető	eseti elbírálás	eseti elbírálás		nem

- Tárhely [df -h]:
  - project (2.7P)
  - scratch (400T)

Kvóta a Komondoron [squota]

account	Filesystem	used	soft limit	hard limit	grace	files	soft limit	hard limit	grace
urbaikl	/project	1.9 GB	20.0 GB	0.0 KB	-	27874	100000	0	-
urbairpo	/project	56.8 GB	4.0 TB	4.4 TB	-	235245	1000000	1100000	-
urbairpo	/scratch	146.1 GB	1.0 TB	1.1 TB	-	32530	300000	330000	-
hpcteszt	/project	1.5 TB	0.0 KB	0.0 KB	-	170706	0	0	-
hpcteszt	/scratch	156.0 KB	0.0 KB	0.0 KB	-	7	0	0	-

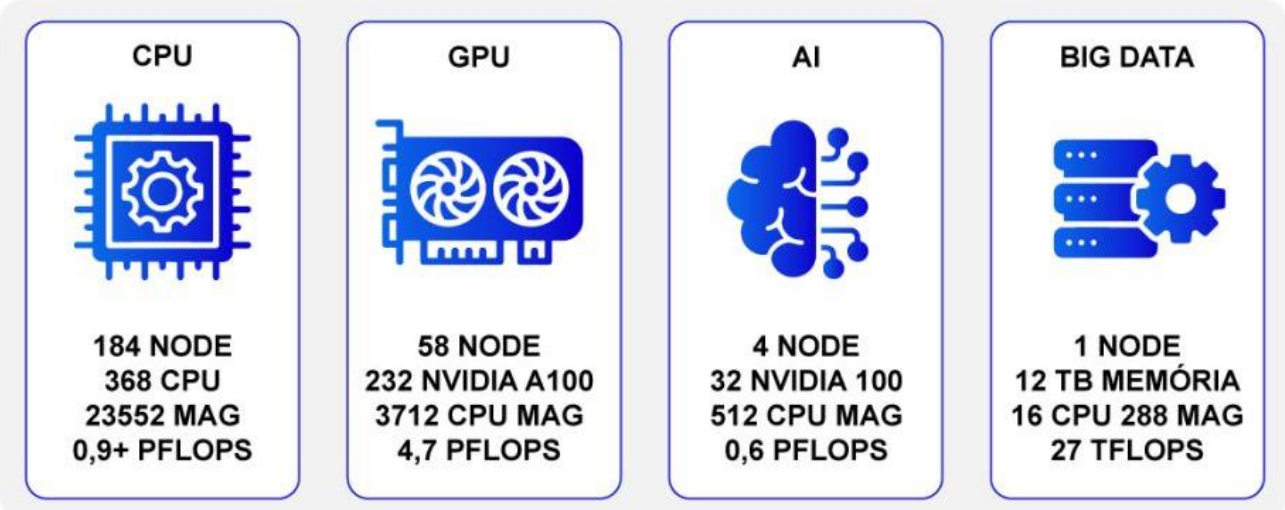
- account: felhasználó/projekt
- Filesystem: fájlrendszer
- used: felhasznált tárhely
- files: felhasznált fájlok száma
- soft limit: átléphető, időlimittel [grace]
- hard limit: nem léphető át

Közösen használt tárhely

- Scratch meghajtón könyvtár létrehozás:
```
mkdir /scratch/p_kkv/dir
```
- Jogosultságok beállítása (egyszer, unix)
```
chgrp p_kepzes /scratch/p_kepzes/dir
chgrp g+s /scratch/p_kepzes/dir
```
- Jogosultságok beállítása (egyszer, access control list - ACL)
```
setfacl -Rdm u::rwx,g:p_kepzes:rwx,o::--- /scratch/p_kepzes/dir
setfacl -Rm u::rwx,g:p_kepzes:rwx,o::--- /scratch/p_kepzes/dir
```

- Próbáljunk létrehozni a közös könyvtárban fájlokat!
- Hasonlóan /project-ben (root jog szükséges)
- Másoljuk át a gyakorlat anyagát a saját könyvtárunkba
```
cp /scratch/p_kepzes/gyakorlat.tar.gz ~
cd
tar zxf gyakorlat.tar.gz
```

A Komondor hardvere



- Processzorok:
  - `lscpu`
  - `srun -p cpu lscpu`
  - `srun -p gpu --gres=gpu:0 lscpu`
- Memória:
  - `free -h`
  - `srun -p cpu free -h`
- Partíciók:
  - `sinfo`
  - `sinfo -s`
- Processzoridők:
  - `sbalance`

sbalance

```
[p_kepkl@vn01 ~]$ sbalance
| account    |   CPU usage |   CPU limit |   GPU usage |   GPU limit |
|:-----------|------------:|------------:|------------:|------------:|
| p_kepzes   |          11 |           0 |           0 |           0 |
[p kepkl@vn01 ~]$
```

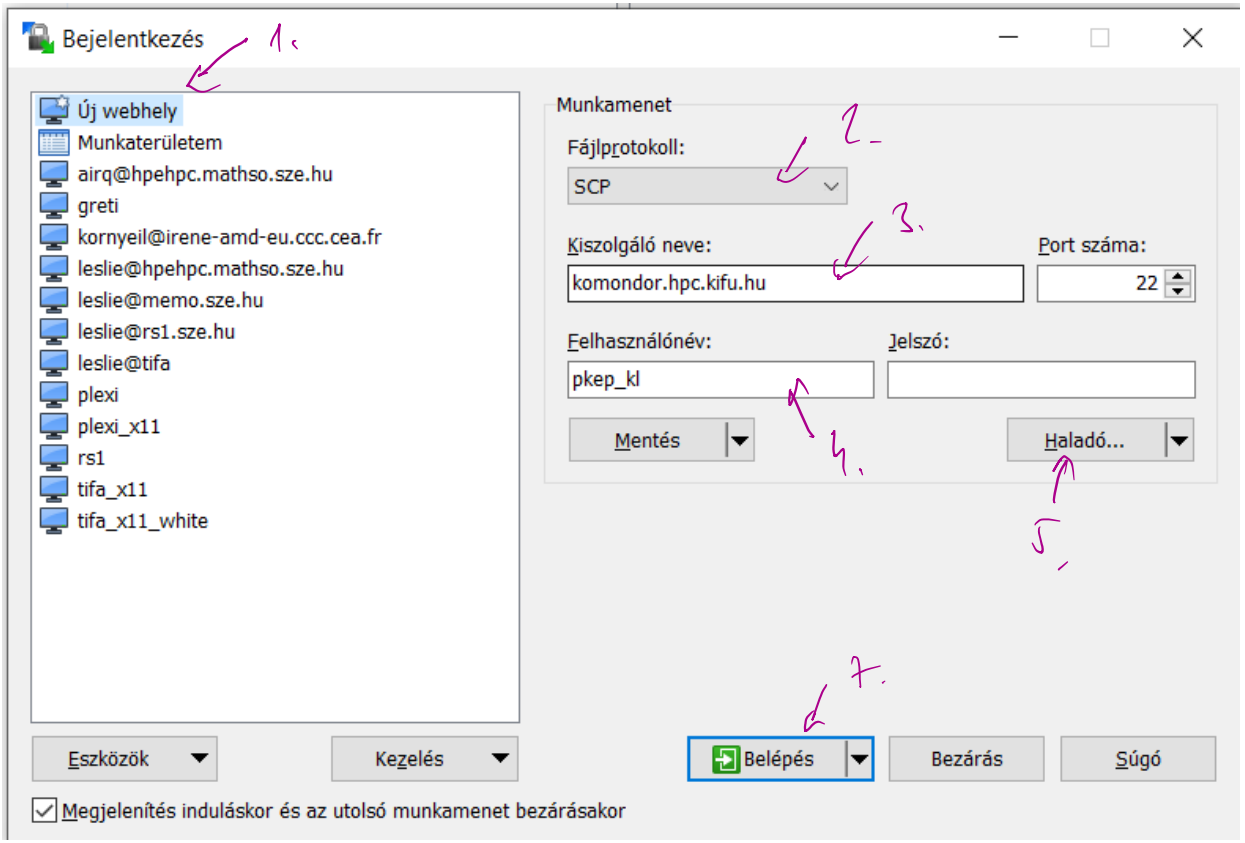
- account: projekt név
- CPU usage: elhasznált CPU órák száma
- CPU limit: használható CPU órák száma
- GPU usage: elhasznált GPU órák száma
- GPU limit: használható GPU órák száma

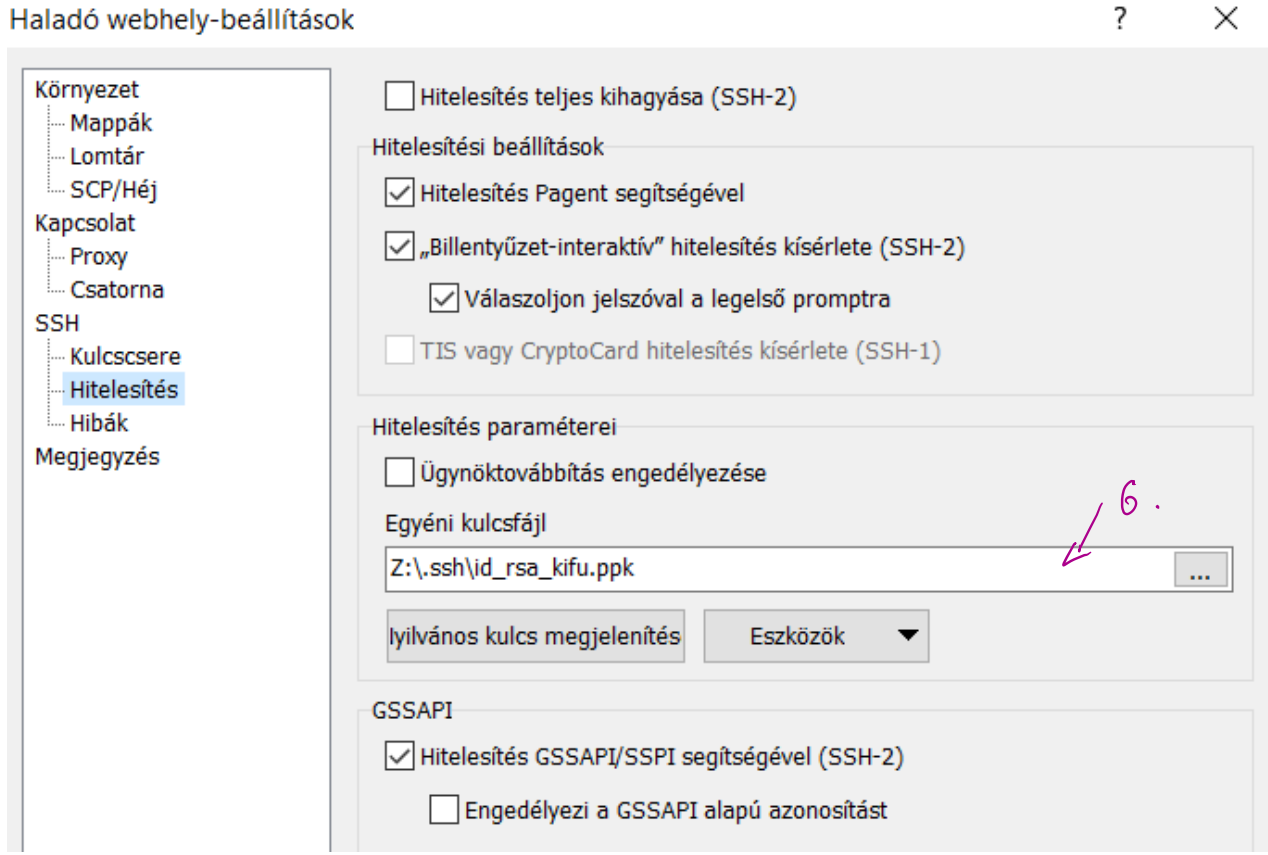
Adatátvitel - Linux SCP

```
leslie@ tifa   ~ $ ls -l image01.png
-rw-r--r-- 1 leslie leslie 826874 Feb 21  2022 image01.png
leslie@ tifa   ~ $ scp -i ~/.ssh/id_rsa image01.png p_kepkl@komondor.hpc.kifu.hu:
(p_kepkl@komondor.hpc.kifu.hu)
Komondor requires two factor authentication. Please login with eduID on this URL in yo
ur browser:
https://ack.hpc.kifu.hu/PGWiEUTo
then press ENTER!

image01.png                                   100%  807KB  14.0MB/s   00:00
leslie@ tifa   ~ $
```

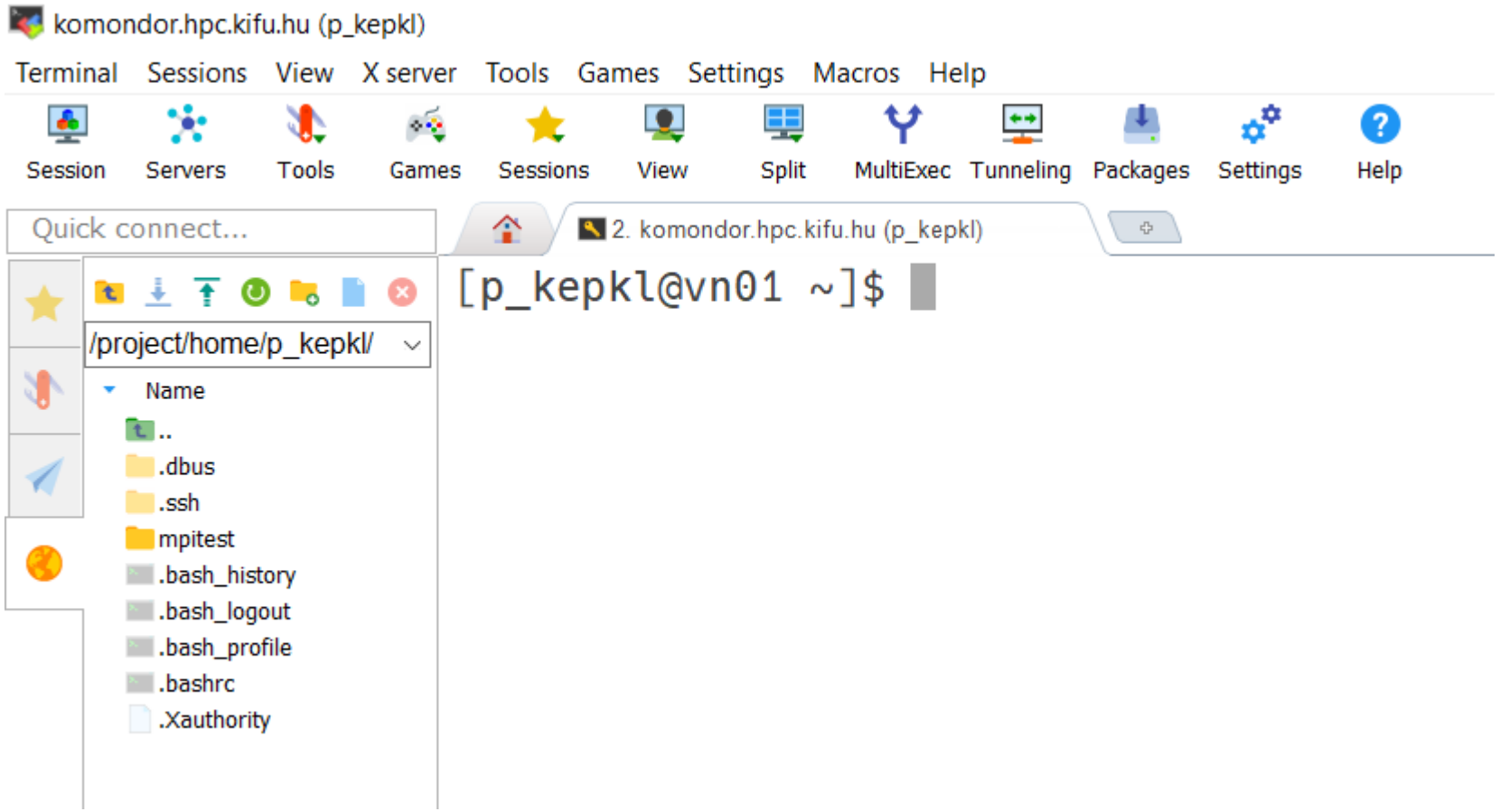
```
leslie@ tifa   ~ $ scp -i ~/.ssh/id_rsa p_kepkl@komondor.hpc.kifu.hu:image01.png .
(p_kepkl@komondor.hpc.kifu.hu)
Komondor requires two factor authentication. Please login with eduID on this URL in yo
ur browser:
https://ack.hpc.kifu.hu/uFXFGkve
then press ENTER!

image01.png                                   100%  807KB  10.9MB/s   00:00
leslie@ tifa   ~ $
```

Adatátvitel - Windows - WinSCP





Adatátvitel - Windows - MobaXterm



feltöltés

letöltés

Adatátvitel - Feladat

- Töltsük le az alábbi fájlokat a gépünkre, majd töltsük fel a Komondorra:
  https://ftp.gromacs.org/gromacs/gromacs-2023.2.tar.gz
  https://sourceforge.net/projects/joe-editor/files/JOE%20sources/joe-4.6/joe-4.6.tar.gz/download

Modul környezet, fordítók, fordítás.

Module környezet

- Cél: megfelelő szoftverkörnyezet beállítása
- Parancs: module
  - avail telepített modulok listája
  - load <m> modul betöltése
  - list betöltött modulok listázása
  - remove <m> betöltött modul törlése
  - purge összes betöltött modul törlése
  - switch <m> <m> váltás modulok között
- Programozási környezetek
  - `module avail PrgEnv`

C fordítók és fordítás

	modul	verzió	c fordító	c++ fordító	verzió	fordítás
GNU	PrgEnv-gnu	8.4.0	cc	CC	gcc 12.2.0	cc -o valami valami.c
Cray	PrgEnv-cray	8.4.0	cc	CC	clang 15.0.1	cc -o valami valami.c
AMD	(jelenleg nem működik)					
Intel	(jelenleg nem működik)					

- Az `1_hello` könyvtárban fordítsuk le és futtassuk a `hello.c` programot!
  - `module load PrgEnv-gnu craype-x86-milan`
  - `cc -Wall -o hello.gnu hello.c`
  - `./hello.gnu`
  - `srun ./hello.gnu`
  - `srun -p cpu ./hello.gnu`
  - `module load PrgEnv-cray craype-x86-milan`

Program fordítása forrásból

Program fordítása forrásból - joe

- Lépjünk a scratch meghajtóra
  ```
  cd /scratch/p_kkv??
  ```
- Csomagoljuk ki a forráskódot
  ```
  tar zxf ~/joe-4.6.tar.gz
  ```
- Konfiguráljuk a forrást
  ```
  cd joe-4.6
  ./configure --prefix=$HOME
  ```
- Fordítsuk le
  ```
  make -j 4
  ```
- Installáljuk
  ```
  make install
  ```

Program fordítása forrásból - gromacs

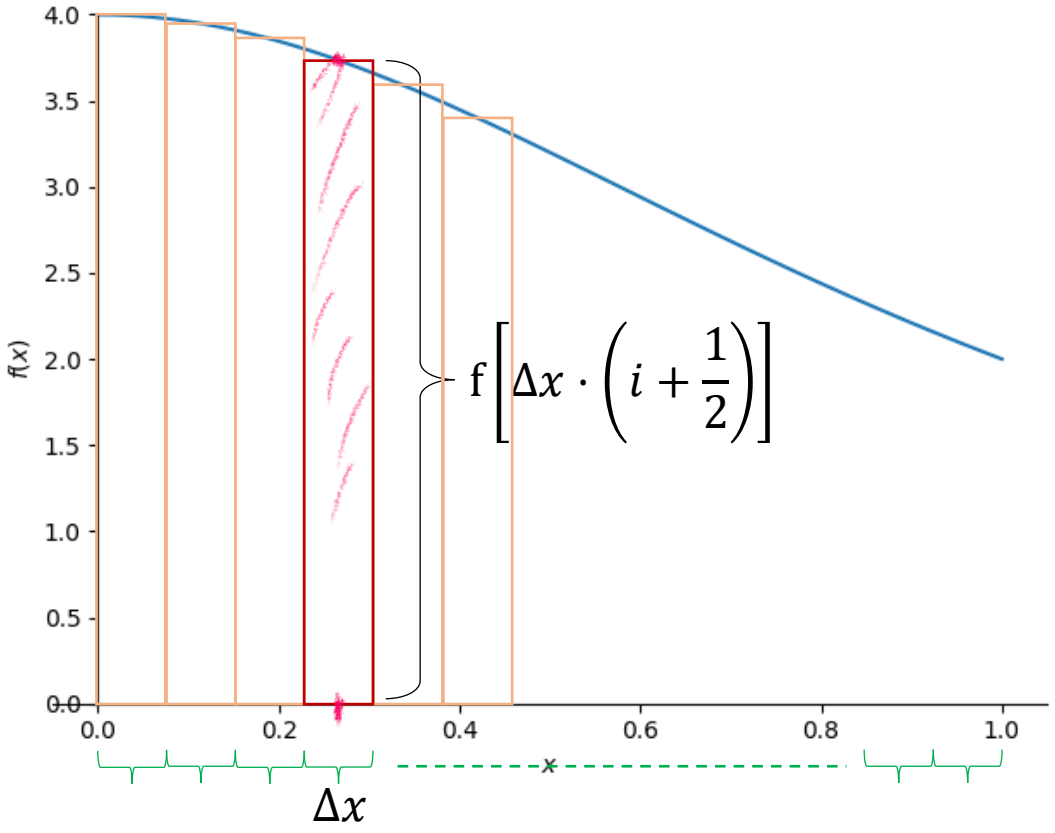
- Lépjünk be egy node-ra interaktív módban
  `srun --mem=32G -p cpu -c 32 --pty /bin/bash`
- Lépjünk a /scratch könyvtárba és csomagoljuk ki a forrást
  `cd /scratch/p_kep??`
  `tar zxf ~/gromacs-2023.2.tar.gz`
- Használjuk az előre telepített fftw és cmake modulokat
  `module load PrgEnv-gnu craype-x86-milan fftw cmake`
- Kövessük a program leírásának utasításait a fordításhoz: https://manual.gromacs.org/documentation/current/install-guide/index.html
- Fordítsunk MPI nélkül!
- Fordításnál használjunk 32 processzt.

Program fordítása forrásból - gromacs mpi

- Önálló feladat:
  Fordítsuk le és installáljuk a gromacs csomagot MPI támogatással a leírás alapján!
  https://manual.gromacs.org/documentation/current/install-guide/index.html
- Installálás után töröljük le a kicsomagolt forráskódot!
- Lépjünk ki a node-ról, ha már nem használjuk többet!

Példaprogram: Numerikus integrálás

Integrálás érintő formulával



Analitikusan:

$$\int_0^1 \frac{4}{1+x^2}\mathrm{dx} = 4 \cdot \mathrm{arctg}1 = \pi$$

Numerikusan:

$$\int_0^1 \frac{4}{1+x^2}\mathrm{dx} \simeq \sum_{i=0}^{n} \Delta x \cdot f\left[(i+\frac{1}{2}) \cdot \Delta x\right]$$

Implementáció

```
    I A  numint_i.c (c)
#include <stdio.h>
#include <stdlib.h>
#include <math.h>

long double f( long double x )
{
    // ezt integráljuk
    return 4.0/(1.0 + x*x);
}

int main( int argc, char ** argv )
{
    // egy argumentum: reszintervallumok szama
    long long n = atol( argv[1] );

    long double dx = 1.0 / n;
    long double dxhalf = 0.5*dx;
    long double sum = 0.0;
    long long i;

    for( i = 0; i < n; i++ )
        sum += f( dxhalf + dx * i );
    sum *= dx;

    printf("Integralosszeg = %16.14Le\n", sum);
    printf("Hiba = %16.14Le\n", sum - M_PI);
    return 0;
}
```

Analitikusan:

$$\int_0^1 \frac{4}{1+x^2}\mathrm{dx} = 4 \cdot \mathrm{arctg}1 = \pi$$

Numerikusan:

$$\int_0^1 \frac{4}{1+x^2}\mathrm{dx} \simeq \sum_{i=0}^{n} \Delta x \cdot f\left[(i+\frac{1}{2}) \cdot \Delta x\right]$$

Futtatás

- Önálló feladat:
  A 2_numint könyvtárban fordítsuk le és futtassuk a numint.c programot cray és gnu fordítóval majd futtassuk 1 000 000 000 részintervallummal!
- Futtassuk a cpu partíción az srun parancs segítségével!

A SLURM ütemező használata. Jobscript készítése, futtatása.

SLURM - futtatás feladatkezelővel

- Klaszter menedzsment és feladat ütemező és erőforrás kezelő
- Feladatok indítása, futtatása, monitorozása
- Parancsok:
  - sinfo partíciók, gépek állapotának összesítése
  - srun feladat sorbaállítása közvezlen parancssorból
  - sbatch feladat sorbaállítása feladatszkript segítségével
  - squeue jelenleg kezelt feladatok
  - scancel feladat megszakítása
  - sacct információk futtatott vagy futó feladatokról

Feladatszkript formátuma

```
#!/bin/bash
# fajlnev: hello.job
#SBATCH --job-name=feladat_neve
#SBATCH --account=p_kepzes
#SBATCH --nodes=1
#SBATCH --partition cpu
# #SBATCH --mem=128G
# #SBATCH --exclusive
#SBATCH --time=0-00:05:00
#SBATCH --error=slurm-%x-%j.err
#SBATCH --output=slurm-%x-%j.out

module load PrgEnv-gnu craype-x86-milan
./hello.gnu

# elküldés: sbatch hello.job
```

fájlnévben cserélt szimbólumok:

%j	feladatazonosító
%N	rövid gépnév
%n	gép azonosító (szám 0-tól)
%t	folyamatazonosító (szám 0-tól)
%u	felhasználónév
%x	feladat neve

Feladatszkript formátuma - rövid paraméterek

```
#!/bin/bash
# fajlnev: hello.job
#SBATCH -J feladat_neve
#SBATCH -A=p_kepzes
#SBATCH -N 1
#SBATCH -p cpu
# #SBATCH --mem=128G
# #SBATCH --exclusive
#SBATCH -t 0-00:05:00
#SBATCH -e slurm-%x-%j.err
#SBATCH -o slurm-%x-%j.out

module load PrgEnv-gnu craype-x86-milan
./hello.gnu

# elküldés: sbatch hello.job
```

fájlnévben cserélt szimbólumok:

%j	feladatazonosító
%N	rövid gépnév
%n	gép azonosító (szám 0-tól)
%t	folyamatazonosító (szám 0-tól)
%u	felhasználónév
%x	feladat neve

Feladatszkript készítése

- A 2_numint könyvtárban készítsünk feladatszkriptet a numint.gnu programhoz, majd 1 000 000 000 részintervallummal küldjük be a feladatot.
- Becsüljük meg a futásidőt! Mi történik, ha túl rövid időkorlátot adunk meg? Küldjük be a feladatot újra!
- Vizsgáljuk meg a feladatok által használt erőforrásokat az alábbi paranccsal:

```
sacct --format=account,jobname,jobid,CPUTimeRAW,cputime,NCPUS
```

- Az sacct parancs segítségével írjuk ki a sikeresen lefutott feladat maximális memóriafoglalását és futásidejét!

Közös használat etikettje

Szuperszámítógép használat szabályai

- Mire használható: https://docs.hpc.kifu.hu/first-steps/rules.html#mire-hasznalhato-a-szuperszamitogep-infrastruktura
  - kutatási és oktatási célra
  - nagy számítási igényű vagy nagy memóriaigényű feladatokra
- Mire nem használható:
  - profitorientált tevékenységhez - kivéve KIFÜ által engedélyezett partnerek
  - nem a hardvernek megfelelő feladatra - például webes szolgáltatások, fájlmegosztás
- Felelősség:
  - Az oktató felel a diákjaiért.
  - A saját azonosítóért mindenki saját maga felel.

A head node [vn01] használatának szabályai

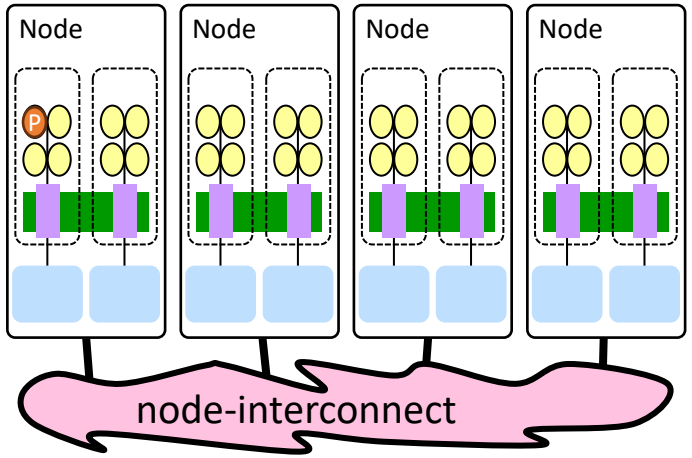
- Mire használható:
  - Fájlműveletek: másolás, mozgatás, becsomagolás (egy szálon), kicsomagolás
  - Egyszerűbb elő- és utófeldolgozás, melyeknek nincs nagyobb erőforrásigényük
  - Egyszerűbb, gyorsabb fordítások és tesztszimulációk
- Figyeljük a jelenleg foglalt erőforrásokat: w, free
- Mire nem használható:
  - Hosszabb, erőforrásigényes szimulációk, fordítások
  - Komolyabb elő- vagy utófeldolgozás
  - Párhuzamos csomagolás (man xz)

KIFÜ

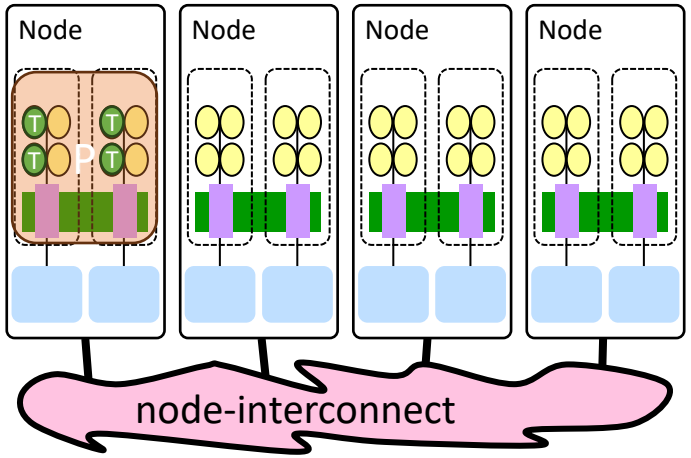
OpenMP, MPI és hibrid programok fordítása és futtatása

Soros, OpenMP, MPI és Hibrid feladatok

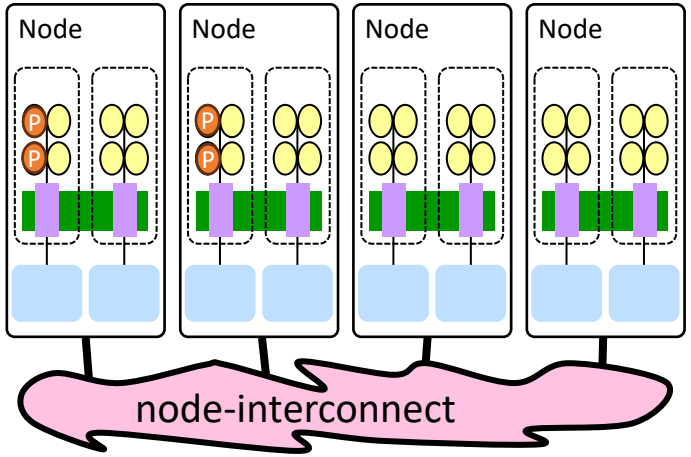
Soros



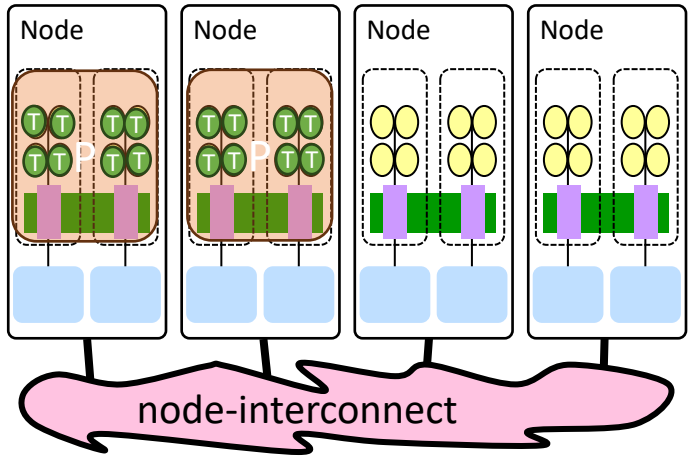
OpenMP



MPI



MPI+OpenMP



P processz

T szál

OpenMP program futtatás

```
#!/bin/bash
# fajlnev: hello_omp.job
#SBATCH --job-name=hello_omp
#SBATCH --nodes=1
#SBATCH --cpus-per-task=8          #vagy -c
#SBATCH --partition cpu
# #SBATCH --mem=128G
# #SBATCH --exclusive
#SBATCH --time=0-00:05:00
#SBATCH --error=slurm-%x-%j.err
#SBATCH --output=slurm-%x-%j.out

module load PrgEnv-gnu craype-x86-milan
export OMP_NUM_THREADS=8
./hello_omp.gnu

# elküldés: sbatch hello_omp.job
```

fordítás:
cc -fopenmp -o hello_omp.gnu hello_omp.c

futtatás:
OMP_NUM_THREADS=8 srun -c 8 ./hello_omp.gnu

MPI program futtatás

```
#!/bin/bash
# fajlnev: hello_mpi.job
#SBATCH --job-name=hello_mpi
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=4
#SBATCH --partition cpu
# #SBATCH --mem=128G
# #SBATCH --exclusive
#SBATCH --time=0-00:05:00
#SBATCH --error=slurm-%x-%j.err
#SBATCH --output=slurm-%x-%j.out

module load PrgEnv-gnu craype-x86-milan
srun ./hello_mpi.gnu

# elküldés: sbatch hello_mpi.job
```

fordítás:
cc -o hello_mpi.gnu hello_mpi.c

futtatás?:
`srun -n 8 -N 2 ./hello_mpi.gnu`

Hibrid program futtatás

```
#!/bin/bash
# fajlnev: hello_hybrid.job
#SBATCH --job-name=hello_hybrid
#SBATCH --nodes=1
#SBATCH --nodes=2
#SBATCH --ntasks-per-node=3
#SBATCH --cpus-per-task=4          #vagy -c
#SBATCH --partition cpu
# #SBATCH --mem=128G
# #SBATCH --exclusive
#SBATCH --time=0-00:05:00
#SBATCH --error=slurm-%x-%j.err
#SBATCH --output=slurm-%x-%j.out

module load PrgEnv-gnu craype-x86-milan
export OMP_NUM_THREADS=4
srun ./hello_hybrid.gnu

# elküldés: sbatch hello_hybrid.job
```

fordítás:

```
cc -fopenmp -o hello_hybrid.gnu hello_hybrid.c
```

futtatás:

```
OMP_NUM_THREADS=4 srun -n 2 -c 4\
    --ntasks-per-node=3 ./hello_hybrid.gnu
```

KIFÜ

Futásidőbecslés

Futásidő becslés célja

- Célja:
  - Megfelelő mennyiségű erőforrást foglaljunk a szimulációhoz
  - Több erőforrást foglalva nem kapunk lényegesen hamarabb eredményt
  - A queue-ban töltött várakozási idő megfelelően rövid
  - Nem pazaroljuk a processzoridőnket
- Módja:
  - A szimuláció futásidejének viselkedését tanulmányozzuk
  - Teszteseteket hozunk létre:
    - Kis méretű szimuláció a workflow működésének ellenőrzésére
    - Közepes méretű szimuláció a futásidő viselkedésének megismerésére
    - Produkciós méretű szimuláció a tényleges futásidő megismerésére

Futásidő modell

- Futásidő(N, t, p) = inicializálás(N, p)
  + számítás(N, t, p)
  + kommunikáció(N, t, p)
  + sorbanállás(p, random)

  ahol N: szimulált tartomány mérete (pl. részecskeszám)
  t: szimulált idő vagy iterációk száma
  p: processzek száma
- t általában lineáris, nem befolyásolja a memóriahasználatot
- N általában nem lineáris futásidőben, lineáris memóriahasználatban

Párhuzamosítás jóságának jellemzése

- Futásidő: $T(N, p, t)$
- Gyorsulás: $S(N, p, t) = T(N, 1, t) / T(N, p, t)$
  gyorsulás = soros futásidő / párhuzamos futásidő
- Párhuzamos hatékonyság:
  $E(N, p, t) = S(N, p, t) / p$
- Futtatás költsége:
  $K(N, p, t) = p * T(N, p, t)$

Futásidőbecslés lépései

- Kis méretű szimuláció futtatása:
  - t és N elég kicsi ahhoz, hogy 1 processzort használva is belátható időn belül lefusson a szimuláció
- Közepes méretű szimuláció futtatása:
  - N-et és p-t növelem úgy, hogy a futásidő a „jól skálázódó tartományba” kerüljön, azaz nagyjából p-szeres gyorsulást érjünk el.
  - p-t növelem, amíg a „jó skálázódás” megmarad (általában duplázom).
  - N-et növelem, amíg a produkciós méretet el nem érem (általában duplázom).
- Produkciós méretű szimuláció futtatása:
  - t-t produkciós mértékre növelem, s az előző pontban megkapott p processzorszámmal futtatom.

CUDA programok fordítása és futtatása

GPU használat - fordítás, eszközök listázása

- Környezet:
  ```
  module load PrgEnv-gnu craype-x86-milan cuda
  module switch gcc/12.2.0 gcc/11.2.0
  ```
- Fordítás:
  ```
  nvcc -o devid devid.cu
  ```
- Futtatás:
  ```
  srun -p gpu --gres=gpu:2 devid
  ```
- Feladat:
  Készítsünk feladatszkriptet!

GPU használat - feladatszkript

```
#!/bin/bash
# fajlnev: devid_cuda.job
#SBATCH --job-name=devid_cuda
#SBATCH --nodes=1
#SBATCH --partition gpu
# #SBATCH --mem=128G
# #SBATCH --exclusive
#SBATCH --time=0-00:05:00
#SBATCH --error=slurm-%x-%j.err
#SBATCH --output=slurm-%x-%j.out
#SBATCH --gres=gpu:2

module load PrgEnv-gnu craype-x86-milan cuda
./devid

# elküldés: sbatch devid_cuda.job
```

- Feladat:
  - Fordítsuk le, majd futtassuk a hello.cu fájlt!
  - Készítsünk hozzá feladatszkriptet!

(Forrás: https://www.computer-graphics.se/hello-world-for-cuda.html)

GPU használat - PyTorch installálás

- Nézzük meg az installációs dokumentációt: https://pytorch.org/get-started/locally/#linux-pip

- Válasszuk ki: stable, linux, pip, cuda 11.8
  (module avail cuda)

- pip3.9-et használjunk!

GPU használat - PyTorch példa

- Feladat:
  Harmadrendű polinom illesztése gradiens módszerrel
- Forrás:
  https://pytorch.org/tutorials/beginner/pytorch_with_examples.html
- Futtatás:

```
python3.9 polyfit_numpy.py
srun -p cpu python3.9 polyfit_numpy.py
srun -p cpu python3.9 polyfit_torch.py
srun -p gpu --gres=gpu:1 python3.9 polyfit_torch.py
```

KIFÜ

Köszönöm megtisztelő figyelmüket!

dr. Környei László
laszlo.kornyei@math.sze.hu

Kérek mindenkit, töltsük ki a visszajelzést!

https://konferencia.kifu.hu/event/32/surveys/20

hpc.kifu.hu
@HPC.CC.hu
@HPC_hu

tartalék

Portál: projektigénylés, ssh kulcs generálás

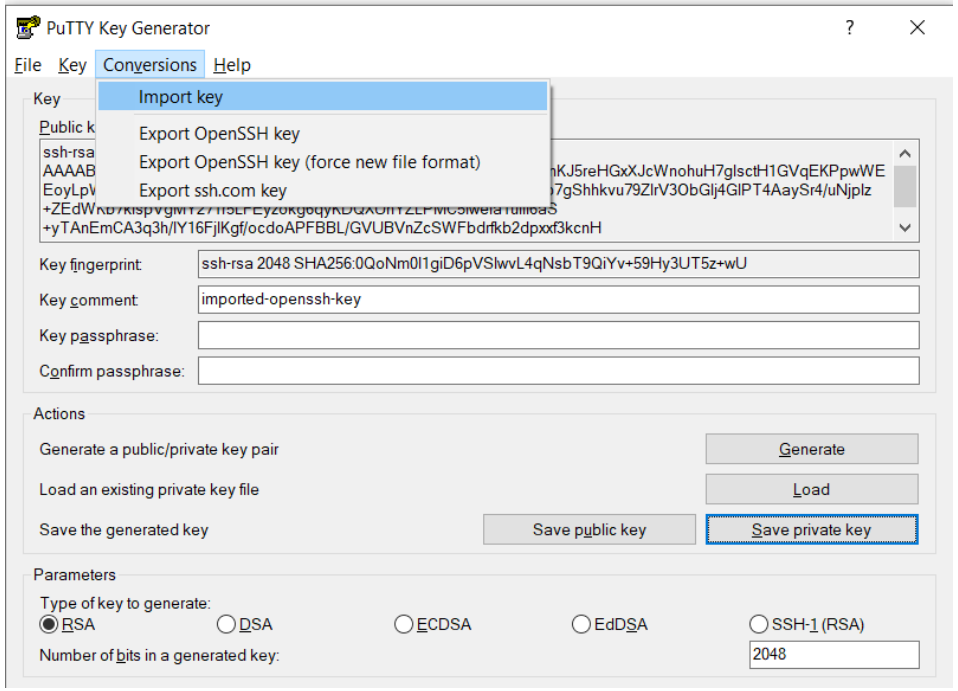
Projektigénylés

- A portál elérése: https://portal.hpc.kifu.hu/v2/
- Új projekt igénylése (akadémiai projekt): https://portal.hpc.kifu.hu/v2/project/user
  - Szükséges adatok:
    - Munkahelyi vezető (pl. tanszékvezető) adatai
    - Projekt rövid megnevezés, azonosító
    - Erőforráscsomagok: különféle tárhely/CPU idő/GPU idő
    - Használni kívánt szoftverek
    - A kutatás bemutatása: rövid leírás, tudományterület, előzményei, leírása, várható eredményei
  - Beadás után munkahelyi vezetővel aláíratni, aláírt példányt feltölteni

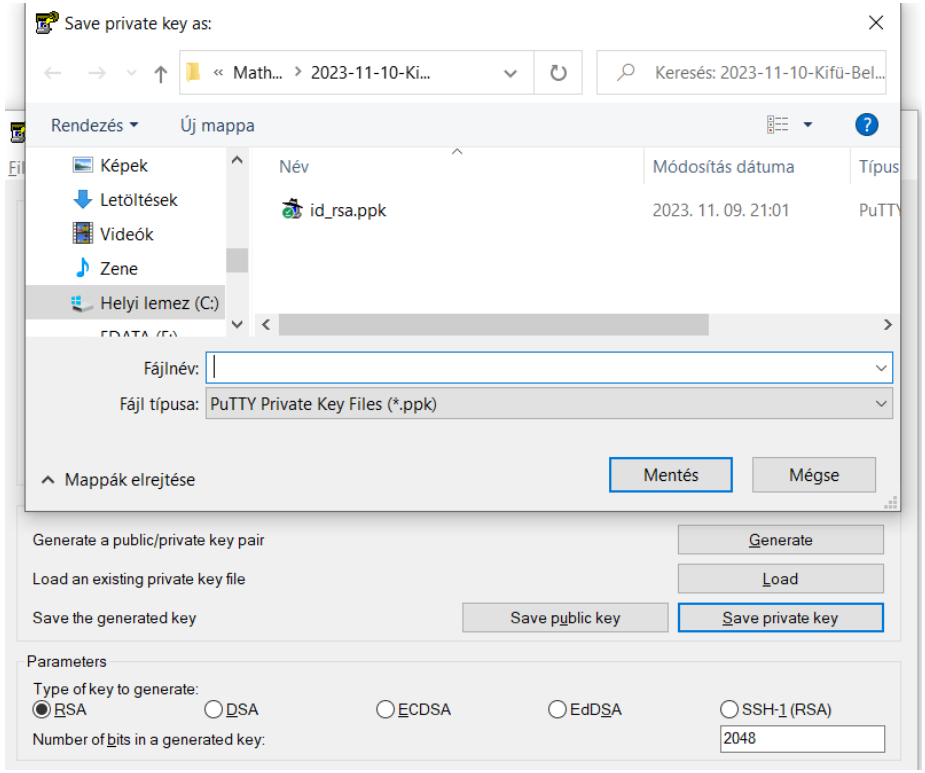
SSH kulcs generálás

- Ha **még nincs kulcs**: generálhatok magamnak, és a portálban is. A legegyszerűbb a portálban generálni!
  - Linux: privát kulcsot el kell menteni (pl) ide: `~/.ssh/id_rsa_kifu`
    Belépés a Komondorra:
    `ssh -i ~/.ssh/id_rsa_kifu user@komondor.hpc.kifu.hu`
  - Windows: portálban generált privát kulcsot el kell menteni és konvertálni ppk-ra puttygen segítségével!
  - Windows: puttygen segítségével generálom a kulcsot, a publikus részt az ablakból portálba másolom. A ppk-t a gépemre mentem
- Ha **már van kulcs**:
  - Linux: a publikus kulcs közvetlen feltölthető
  - Windows: puttygen-be betöltöm a ppk-t. által generált publikus kulcsok nem feltétlen kompatibilisek a portállal!

SSH kulcs import portálból – Puttygen, Windows

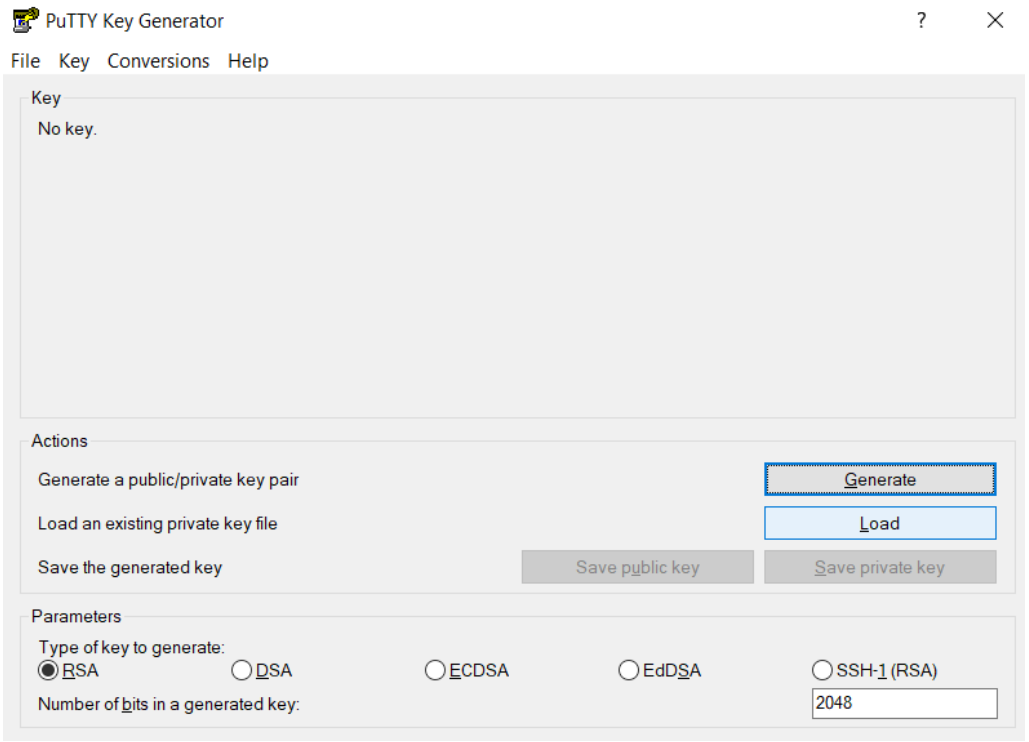


1. portálon generált kulcs importálása

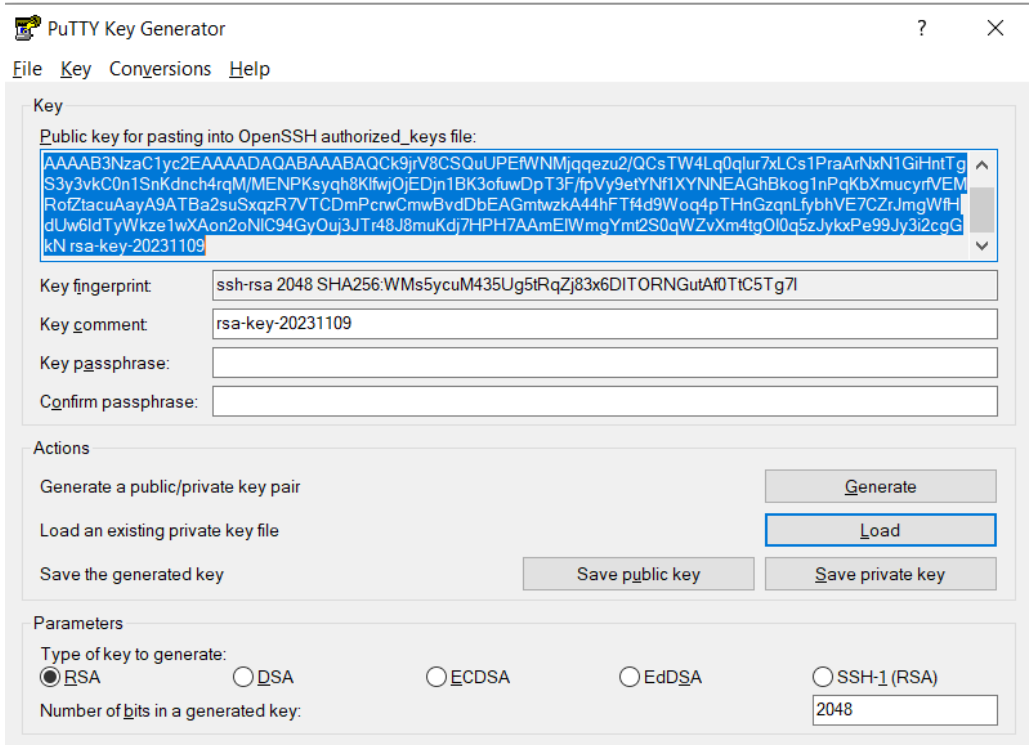


2. portálon generált kulcs exportálása ppk-ba

SSH kulcs export portálba– Puttygen, Windows



1. kulcs betöltés vagy generálás



2. publikus kulcs kilelölése és másolás ctrl-c-ctrl-v-vel
3. generálás esetén ppk mentése