European Research Council
Established by the European Commission

Slide of the seminar

NewTurb Cluster

New cluster for Turbulence simulations

NewTurb Cluster

New cluster for Turbulence simulations

Outline

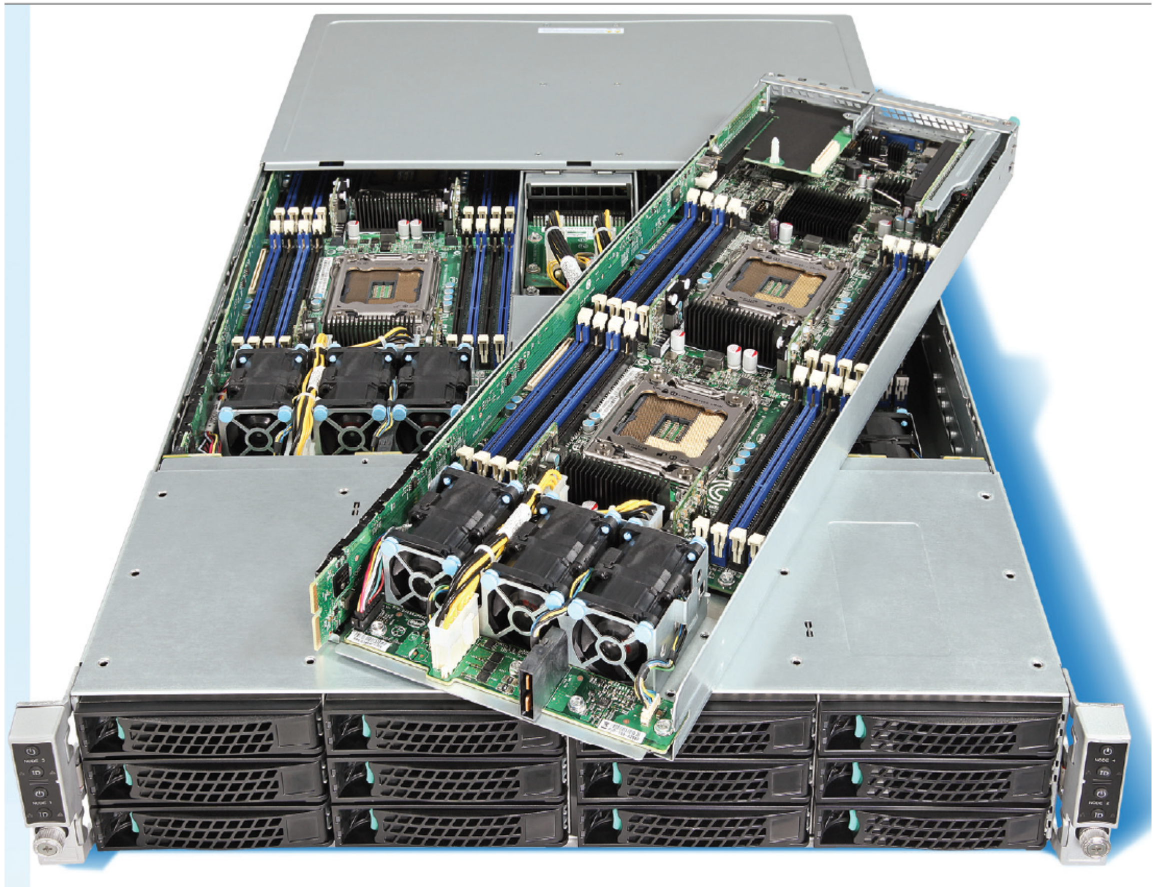
- Hardware presentation
- Installed software
- Batch usage
- Visualization
- Code development
- Next

Hardware

- Cluster is composed by 3 elements:
  - Frontend
    - Two 12-cores 2.4GHz intel Xeon server
    - 256 GB RAM 1.6GHz DDR3 RAM
    - Two 56 Gbit/s QDR Infiniband
    - 1 Nvidia Quadro K4000 GPU
  - 16 computing nodes, each with
    - Two 12-cores 2.4GHz Intel Xeon chip, for a total of 24 cores
    - 256 GB 1.6GHz DDR3 RAM
    - One 56Gbit/s QDR Infiniband
  - Storage server
    - 64 GB RAM
    - 250 TB disk in RAID5 configuration
    - Two 56 Gbit/s QDR Infiniband

4 nodes in 2U

- Highly compact nodes:
  - 4 nodes in a 2U server
  - Energy efficient
- Maintenance
  - Hot swappable node
  - Hot swappable Hds
- We have 4 of these

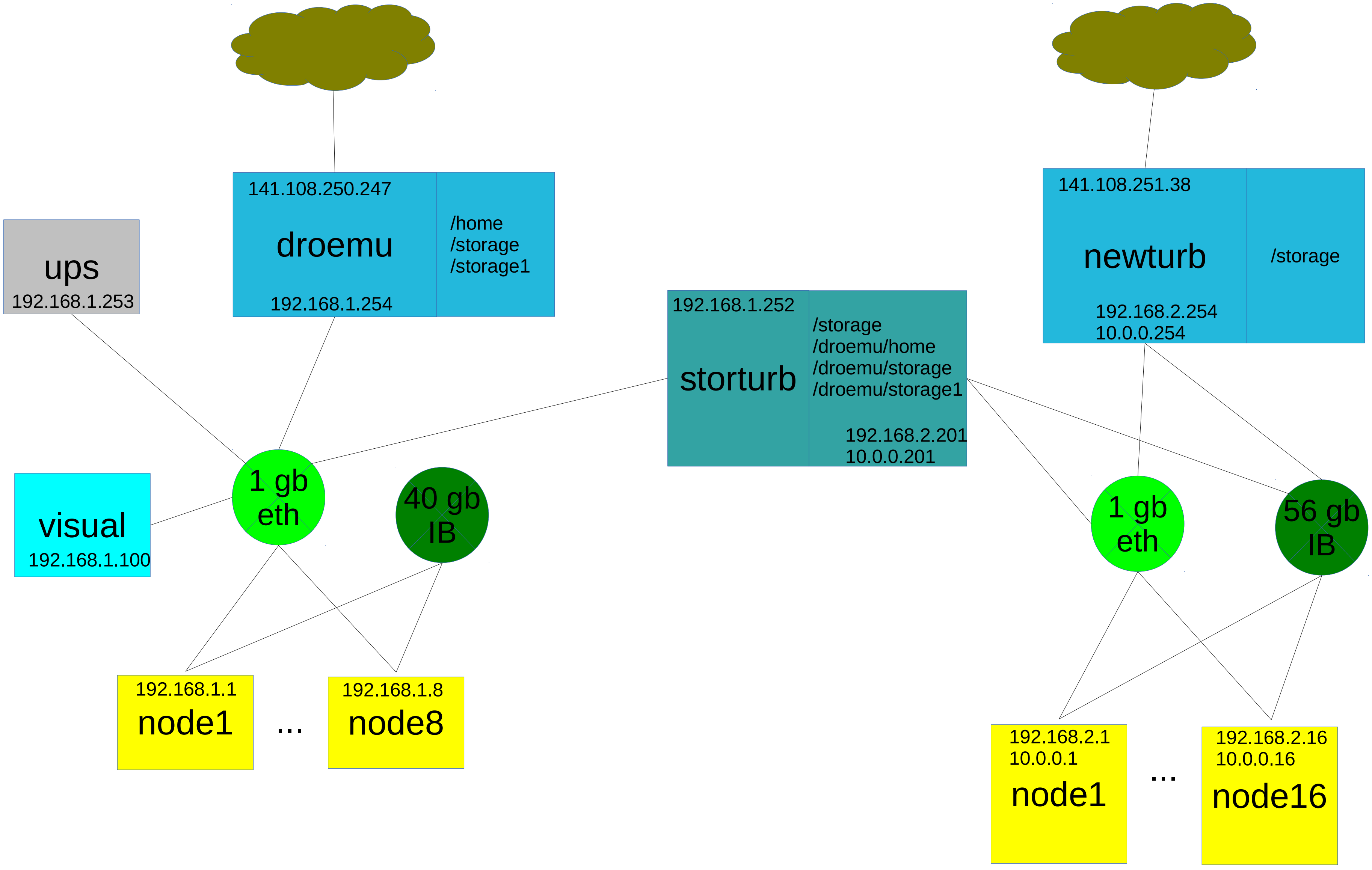


Storage Server

- 72 x 4TB Hds
  - 288 TB raw space
- 64 GB Ram
- Dual Intel XEON 8 cores @ 2.60GHz
- Dual Infiniband FDR 56 Gb/s
- 3 HW Raid controller

Frontend: newturb

- 256 GB Ram
- Same 2x12core Intel CPU
- Two Infiniband FDR 56Gb/s
- Quadro K4000 GPU


ups
192.168.1.253
141.108.250.247
droemu
192.168.1.254
/home
/storage
/storage1
visual
192.168.1.100
1 gb
eth
40 gb
IB
192.168.1.1
node1
...
192.168.1.8
node8
192.168.1.252
storturb
/storage
/droemu/home
/droemu/storage
/droemu/storage1
192.168.2.201
10.0.0.201
141.108.251.38
newturb
192.168.2.254
10.0.0.254
/storage
1 gb
eth
56 gb
IB
192.168.2.1
10.0.0.1
node1
...
192.168.2.16
10.0.0.16
node16

Infiniband benchmark: bandwidth

#bytes	#iterations	BW peak MB/s	BW avg MB/s
2	10	1	1
16	10	12	11
64	10	49	45
256	10	152	152
512	10	347	312
1k	10	883	880
4k	10	3120	3119
16k	10	4637	4636
32k	10	5221	5221
128k	10	5780	5780
1M	10	5967	5967
2M	10	5976	5976
4M	10	5940	5940
8M	10	5992	5992

Infiniband benchmark: latency

#bytes	#iterations	t_min usec	t_max usec	t_typical usec
2	10.0	0.9	9.7	1.2
16	10.0	0.9	2.8	1.0
64	10.0	1.0	2.7	1.0
256	10.0	1.4	8.4	1.4
512	10.0	1.5	4.5	1.6
1k	10.0	1.8	9.1	1.9
4k	10.0	3.1	4.7	3.2
16k	10.0	5.0	9.4	5.3
32k	10.0	7.8	11.8	7.8
128k	10.0	23.5	27.8	23.8
1M	10.0	168.7	174.2	169.4
2M	10.0	334.9	340.2	335.3
4M	10.0	666.1	671.6	667.4
8M	10.0	1328.6	1333.3	1330.8

Software

- Operating system is CentOS v6.5 64bit
  - Community edition of Redhat OS: very well supported
- Libraries
  - OpenMPI v1.8.1
  - HDF v1.8.13
  - fftw v2, v3
  - P3DFFT
  - GSL
- Tools
  - CMake
  - Paraview
  - CUDA v6

Users

- Each user logins on newturb.roma2.infn.it
- His/her home is a LOCAL directory
  - NOT shared with the whole cluster (SPEED hack)
- Place for job preparation:
  - /storage/<<USER>>
  - SHARED with the cluster
    - NFS v4
    - Infiniband link 56Gb/s (hw link speed, not user speed)

Users: filesystems layout

- /storage is the new 250TB area
  - Shared via Infiniband
- /droemu area visible in read-only
  - Gigabit link
- Symlinks for setting up simulations

Code

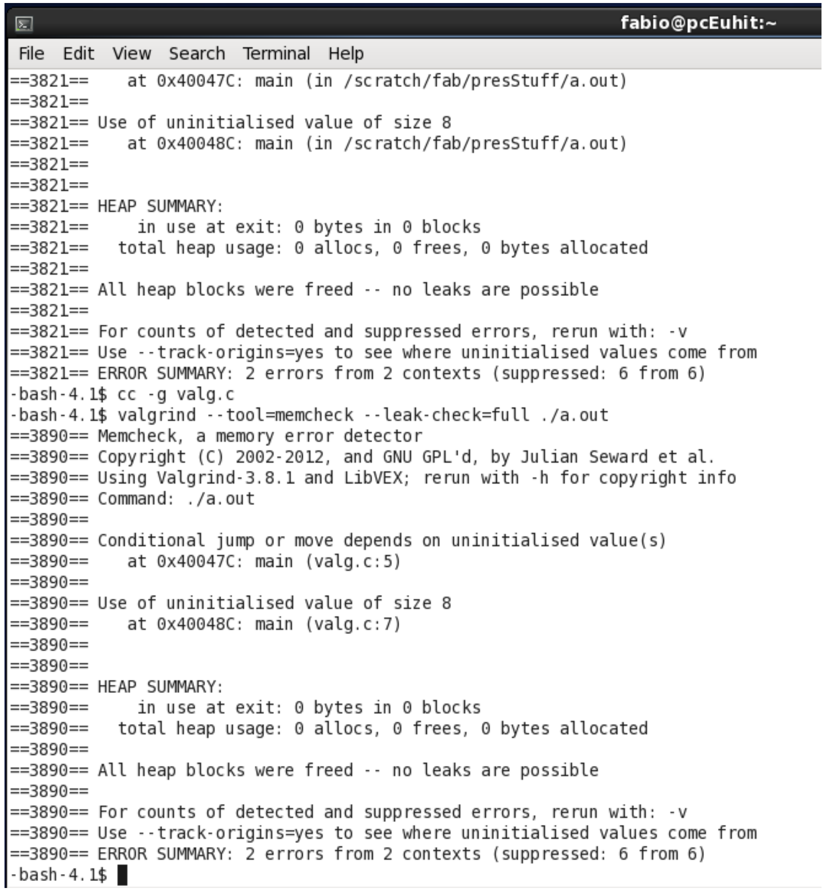
- Compile
- Link
- ldd
- valgrind
- efence
- GDB

Code: Valgrind

- Simulate execution on a virtual PC
- Detects memory errors, memory leaks,...
- Simulate the execution
  - Slow
- Needs some help with system libs
  - Unless you want to contribute to opensource world!

Code: Valgrind

```
1  #include <stdlib.h>
2  int main()
3  {
4        int p,t,b[10];
5        if (p==5)      ERROR
6            t=p+1;
7            b[p]=100;  ERROR
8        return 0;
9  }
```



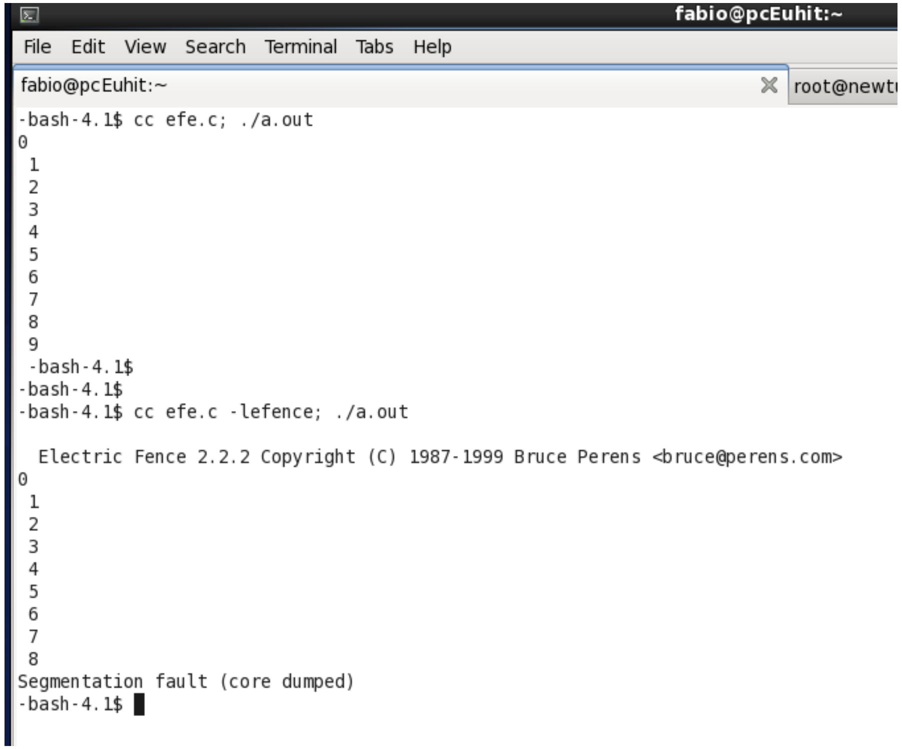
```
==3821==    at 0x40047C: main (in /scratch/fab/presStuff/a.out)
==3821==
==3821== Use of uninitialised value of size 8
==3821==    at 0x40048C: main (in /scratch/fab/presStuff/a.out)
==3821==
==3821==
==3821== HEAP SUMMARY:
==3821==     in use at exit: 0 bytes in 0 blocks
==3821==   total heap usage: 0 allocs, 0 frees, 0 bytes allocated
==3821==
==3821== All heap blocks were freed -- no leaks are possible
==3821==
==3821== For counts of detected and suppressed errors, rerun with: -v
==3821== Use --track-origins=yes to see where uninitialised values come from
==3821== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 6 from 6)
-bash-4.1$ cc -g valg.c
-bash-4.1$ valgrind --tool=memcheck --leak-check=full ./a.out
==3890== Memcheck, a memory error detector
==3890== Copyright (C) 2002-2012, and GNU GPL'd, by Julian Seward et al.
==3890== Using Valgrind-3.8.1 and LibVEX; rerun with -h for copyright info
==3890== Command: ./a.out
==3890==
==3890== Conditional jump or move depends on uninitialised value(s)
==3890==    at 0x40047C: main (valg.c:5)
==3890==
==3890== Use of uninitialised value of size 8
==3890==    at 0x40048C: main (valg.c:7)
==3890==
==3890==
==3890== HEAP SUMMARY:
==3890==     in use at exit: 0 bytes in 0 blocks
==3890==   total heap usage: 0 allocs, 0 frees, 0 bytes allocated
==3890==
==3890== All heap blocks were freed -- no leaks are possible
==3890==
==3890== For counts of detected and suppressed errors, rerun with: -v
==3890== Use --track-origins=yes to see where uninitialised values come from
==3890== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 6 from 6)
-bash-4.1$ 
```

Code: ElectricFence

- Catch memory errors as they occur
  - Run-Time Bound checking
  - HW based: fast but uses RAM
- One guard-area for each allocation
  - 4kb plus your allocation
- Two trips for extra security
  - Band after mem
  - Band before mem

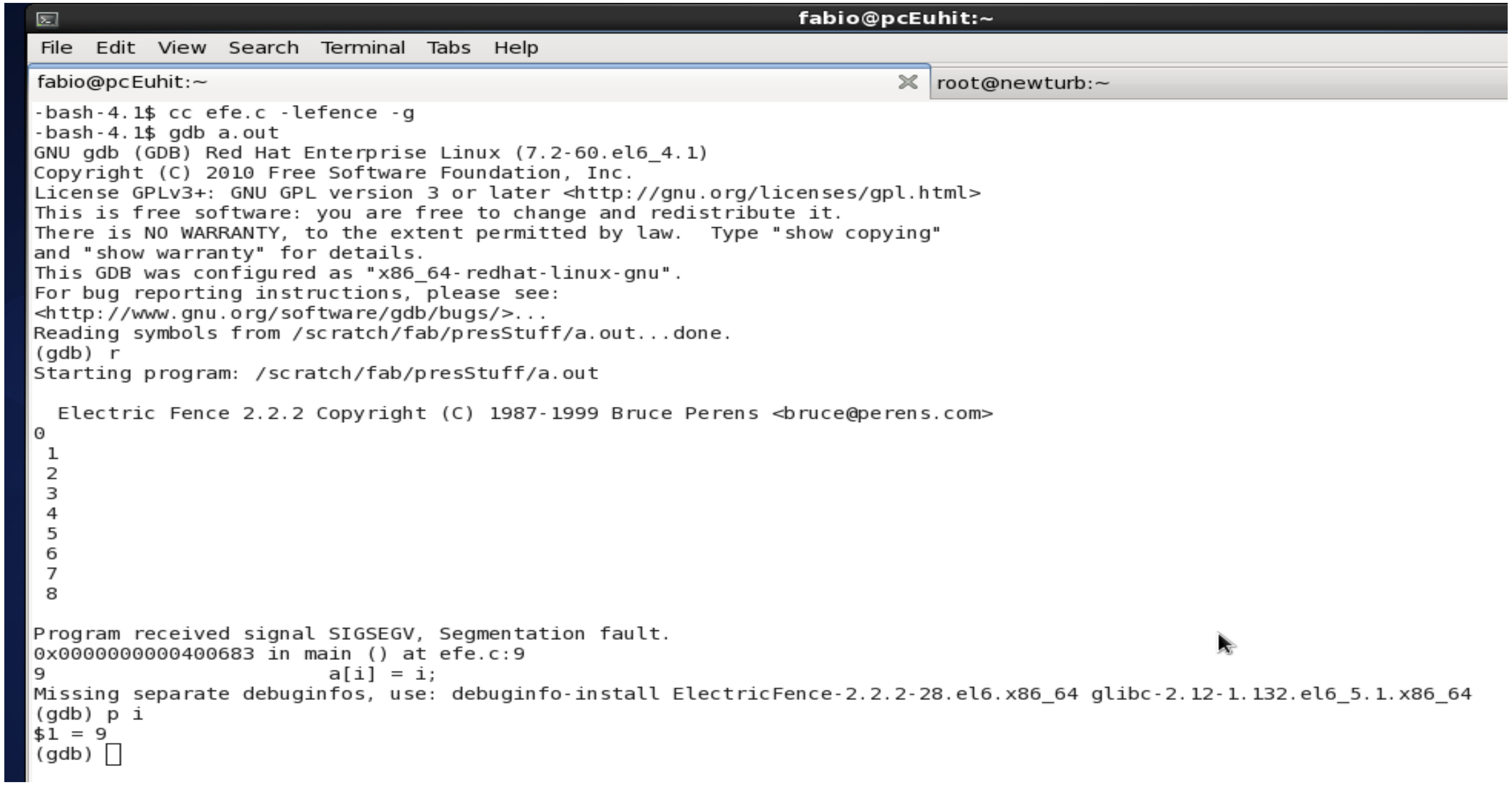
Code: ElectricFence

```c
#include <stdio.h>
#include<stdlib.h>
        int main (void)
        {
            int i;
            int *a = (int*) malloc( 9*sizeof(int));

            for ( i=0; i<=9; ++i){
               a[i] = i;
                printf ("%d\n ", a[i]);
            }

            free(a);
            return 0;
        }
```



```
-bash-4.1$ cc efe.c; ./a.out
0
 1
 2
 3
 4
 5
 6
 7
 8
 9
 -bash-4.1$
-bash-4.1$
-bash-4.1$ cc efe.c -lefence; ./a.out

  Electric Fence 2.2.2 Copyright (C) 1987-1999 Bruce Perens <bruce@perens.com>
0
 1
 2
 3
 4
 5
 6
 7
 8
Segmentation fault (core dumped)
-bash-4.1$ 
```

Code: ElectricFence + GDB



```
-bash-4.1$ cc efe.c -lefence -g
-bash-4.1$ gdb a.out
GNU gdb (GDB) Red Hat Enterprise Linux (7.2-60.el6_4.1)
Copyright (C) 2010 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-redhat-linux-gnu".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /scratch/fab/presStuff/a.out...done.
(gdb) r
Starting program: /scratch/fab/presStuff/a.out

  Electric Fence 2.2.2 Copyright (C) 1987-1999 Bruce Perens <bruce@perens.com>
0
 1
 2
 3
 4
 5
 6
 7
 8

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400683 in main () at efe.c:9
9                       a[i] = i;
Missing separate debuginfos, use: debuginfo-install ElectricFence-2.2.2-28.el6.x86_64 glibc-2.12-1.132.el6_5.1.x86_64
(gdb) p i
$1 = 9
(gdb) 
```

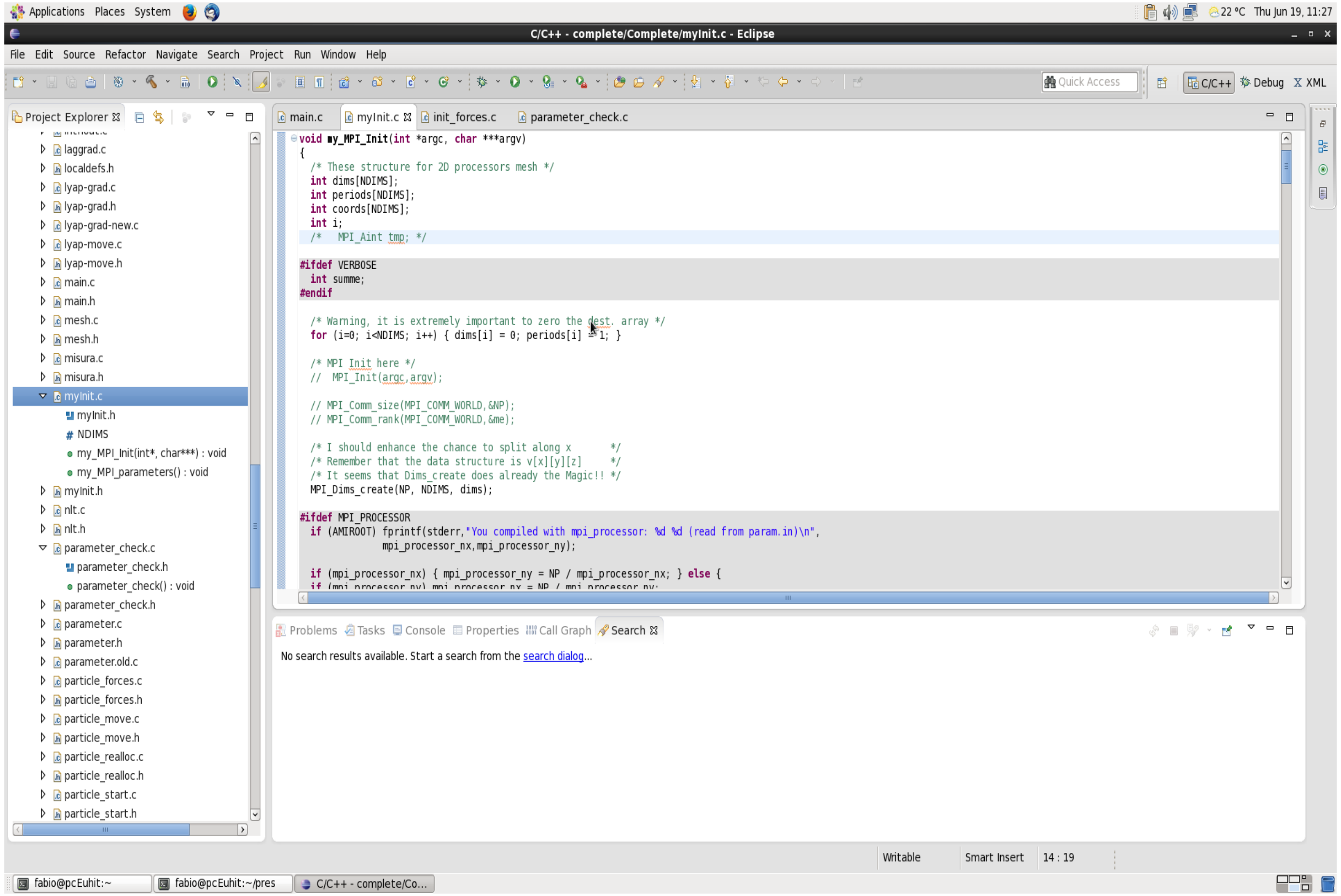
- Debugger helps!

Code: gdb

- Compile with
  - -g -O0
- Run the program
  - command r
- Step
  - command s
- Breakpoint
  - command b
- Pops in case of errors

Code: IDE

- Eclipse
- Integrates with GDB

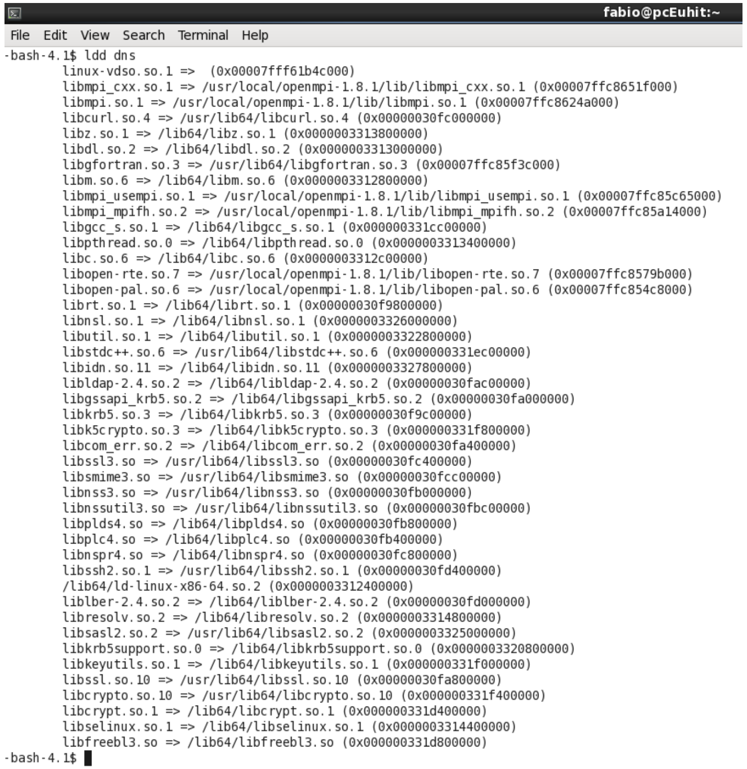


```
void my_MPI_Init(int *argc, char ***argv)
{
  /* These structure for 2D processors mesh */
  int dims[NDIMS];
  int periods[NDIMS];
  int coords[NDIMS];
  int i;
  /*   MPI_Aint tmp; */

#ifdef VERBOSE
  int summe;
#endif

  /* Warning, it is extremely important to zero the dest. array */
  for (i=0; i<NDIMS; i++) { dims[i] = 0; periods[i] = 1; }

  /* MPI Init here */
  //  MPI_Init(argc,argv);

  // MPI_Comm_size(MPI_COMM_WORLD,&NP);
  // MPI_Comm_rank(MPI_COMM_WORLD,&me);

  /* I should enhance the chance to split along x       */
  /* Remember that the data structure is v[x][y][z]     */
  /* It seems that Dims_create does already the Magic!! */
  MPI_Dims_create(NP, NDIMS, dims);

#ifdef MPI_PROCESSOR
  if (AMIROOT) fprintf(stderr,"You compiled with mpi_processor: %d %d (read from param.in)\n",
             mpi_processor_nx,mpi_processor_ny);

  if (mpi_processor_nx) { mpi_processor_ny = NP / mpi_processor_nx; } else {
  if (mpi_processor_ny) mpi_processor_nx = NP / mpi_processor_ny;
```

No search results available. Start a search from the search dialog...

Code: ldd

- Shows the shared libs that will be used



```
-bash-4.1$ ldd dns
        linux-vdso.so.1 =>  (0x00007fff61b4c000)
        libmpi_cxx.so.1 => /usr/local/openmpi-1.8.1/lib/libmpi_cxx.so.1 (0x00007ffc8651f000)
        libmpi.so.1 => /usr/local/openmpi-1.8.1/lib/libmpi.so.1 (0x00007ffc8624a000)
        libcurl.so.4 => /usr/lib64/libcurl.so.4 (0x00000030fc000000)
        libz.so.1 => /lib64/libz.so.1 (0x0000003313800000)
        libdl.so.2 => /lib64/libdl.so.2 (0x0000003313000000)
        libgfortran.so.3 => /usr/lib64/libgfortran.so.3 (0x00007ffc85f3c000)
        libm.so.6 => /lib64/libm.so.6 (0x0000003312800000)
        libmpi_usempi.so.1 => /usr/local/openmpi-1.8.1/lib/libmpi_usempi.so.1 (0x00007ffc85c65000)
        libmpi_mpifh.so.2 => /usr/local/openmpi-1.8.1/lib/libmpi_mpifh.so.2 (0x00007ffc85a14000)
        libgcc_s.so.1 => /lib64/libgcc_s.so.1 (0x000000331cc00000)
        libpthread.so.0 => /lib64/libpthread.so.0 (0x0000003313400000)
        libc.so.6 => /lib64/libc.so.6 (0x000000312c00000)
        libopen-rte.so.7 => /usr/local/openmpi-1.8.1/lib/libopen-rte.so.7 (0x00007ffc8579b000)
        libopen-pal.so.6 => /usr/local/openmpi-1.8.1/lib/libopen-pal.so.6 (0x00007ffc854c8000)
        librt.so.1 => /lib64/librt.so.1 (0x00000030f9800000)
        libnsl.so.1 => /lib64/libnsl.so.1 (0x0000003326000000)
        libutil.so.1 => /lib64/libutil.so.1 (0x0000003322800000)
        libstdc++.so.6 => /usr/lib64/libstdc++.so.6 (0x000000331ec00000)
        libidn.so.11 => /lib64/libidn.so.11 (0x0000003327800000)
        libldap-2.4.so.2 => /lib64/libldap-2.4.so.2 (0x00000030fac00000)
        libgssapi_krb5.so.2 => /lib64/libgssapi_krb5.so.2 (0x00000030fa000000)
        libkrb5.so.3 => /lib64/libkrb5.so.3 (0x00000030f9c00000)
        libk5crypto.so.3 => /lib64/libk5crypto.so.3 (0x000000331f800000)
        libcom_err.so.2 => /lib64/libcom_err.so.2 (0x00000030fa400000)
        libssl3.so => /usr/lib64/libssl3.so (0x00000030fc400000)
        libsmime3.so => /usr/lib64/libsmime3.so (0x00000030fcc00000)
        libnss3.so => /usr/lib64/libnss3.so (0x00000030fb000000)
        libnssutil3.so => /usr/lib64/libnssutil3.so (0x00000030fbc00000)
        libplds4.so => /lib64/libplds4.so (0x00000030fb800000)
        libplc4.so => /lib64/libplc4.so (0x00000030fb400000)
        libnspr4.so => /lib64/libnspr4.so (0x00000030fc800000)
        libssh2.so.1 => /usr/lib64/libssh2.so.1 (0x00000030fd400000)
        /lib64/ld-linux-x86-64.so.2 (0x0000003312400000)
        liblber-2.4.so.2 => /lib64/liblber-2.4.so.2 (0x00000030fd000000)
        libresolv.so.2 => /lib64/libresolv.so.2 (0x0000003314800000)
        libsasl2.so.2 => /usr/lib64/libsasl2.so.2 (0x0000003325000000)
        libkrb5support.so.0 => /lib64/libkrb5support.so.0 (0x0000003320800000)
        libkeyutils.so.1 => /lib64/libkeyutils.so.1 (0x000000331f000000)
        libssl.so.10 => /usr/lib64/libssl.so.10 (0x00000030fa800000)
        libcrypto.so.10 => /usr/lib64/libcrypto.so.10 (0x000000331f400000)
        libcrypt.so.1 => /lib64/libcrypt.so.1 (0x000000331d400000)
        libselinux.so.1 => /lib64/libselinux.so.1 (0x0000003314400000)
        libfreebl3.so => /lib64/libfreebl3.so (0x000000331d800000)
-bash-4.1$ 
```

Batch system

- Torque v4.2.7
  - Defines the queue
  - The server that accepts jobs

- Maui Scheduler v3.3.1
  - Selects job to run
  - The server that shows which job is running

Batch system: 2

- 4 Queues defined:
  - 1 “routing” queue
    - Named route
    - Routes job to the final destination queue based on requested resources
  - 3 “execution” queue
    - Reg_256
      - Highest priority: reserved to jobs of 256 processes
    - Reg_64
      - Middle prio: for jobs of 64 processes
    - Batch
      - Lowest prio: until 64 processes

Batch system: 3

- Job script
  - Text shell script with PBS keywords...
    - #PBS -l nodes=xx:ppn=yy,walltime=zz:zz:zz
    - nodes: number of computing nodes you want
    - ppn: number of processes in each node
    - walltime: maximum time allowed
  - ...and commands
    - mpiexec -np zz -mca btl openib,self -hostfile $PBS_NODEFILE exec-file
      - np: real number of process created
- Submit a job:
  - qsub jobScript.pbs
- Check the queue:
  - showq
  - pbstop

Batch system: 4

- The Job is running...
  - PIC
- StdOut and StdErr will be given back at the END
  - JobScript.oxxx
  - JobScript.exxx
  - Xxx is the jobID queuing number
- Job preparation:
  - Create a directory fo each job under /storage/USER/jobxxx
  - qsub from here
  - Files will not interfere with each other
  - “Easy” debugging

Batch system: 5

- Scheduler policy:
  - 256 cores -> Prio 1256
  - 64 cores ->Prio 164
  - 1-63 cores -> Prio 1-63

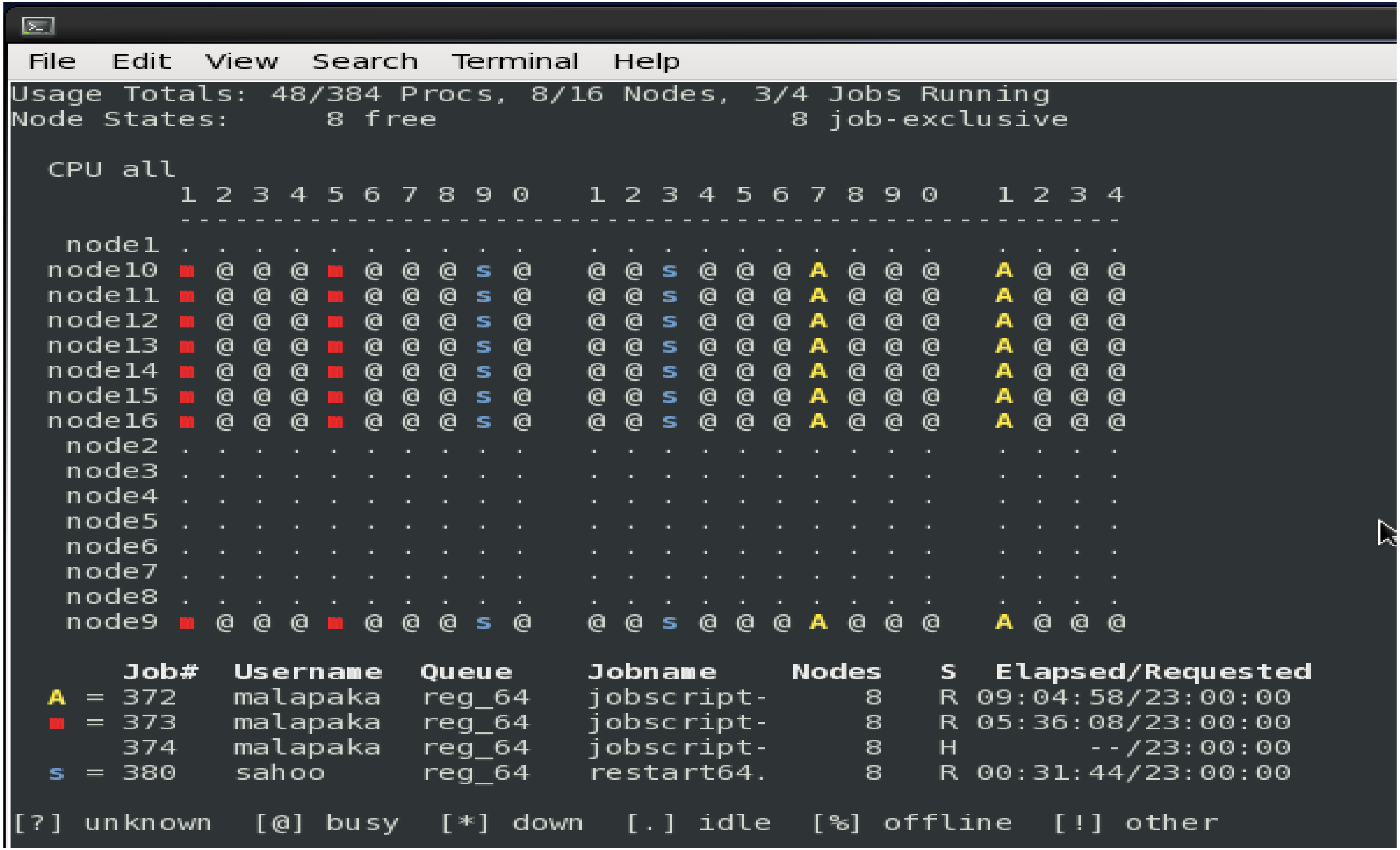
- Backfill active:
  - Small jobs can be scheduled if they don't prevent big ones from running
  - to fill the machine meanwhile

- Queue time into account
- Soon will have a factor for Fair share usage

Queue: showq

ACTIVE	JOBS				
JOBNAME	USERNAME	STATE	PROC	REMAINING	STARTTIME
372	malapaka	Running	64	14:10:00	06:52:26 AM
373	malapaka	Running	64	17:38:50	10:21:16 AM
380	sahoo	Running	64	22:43:14	03:25:40 PM
3 Active Jobs 192 of 384 Processors Active (50.00%)					
8 of 16 Nodes Active (50.00%)					
IDLE	JOBS				
JOBNAME	USERNAME	STATE	PROC	WCLIMIT	QUEUETIME
0	Idle	Jobs			
BLOCKED	JOBS				
JOBNAME	USERNAME	STATE	PROC	WCLIMIT	QUEUETIME
374	malapaka	Hold	64	23:00:00	21:31:34
Total Jobs:4 Active Jobs:3 Idle Jobs:0 Blocked Jobs: 1					

Queue: pbstop



```
Usage Totals: 48/384 Procs, 8/16 Nodes, 3/4 Jobs Running
Node States:      8 free                      8 job-exclusive

   CPU all
            1 2 3 4 5 6 7 8 9 0  1 2 3 4 5 6 7 8 9 0  1 2 3 4
            ---------------------------------------------------
    node1   . . . . . . . . . .  . . . . . . . . . .  . . . .
   node10   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
   node11   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
   node12   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
   node13   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
   node14   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
   node15   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
   node16   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @
    node2   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node3   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node4   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node5   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node6   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node7   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node8   . . . . . . . . . .  . . . . . . . . . .  . . . .
    node9   ■ @ @ @ ■ @ @ @ s @  @ @ s @ @ @ A @ @ @  A @ @ @

       Job#  Username  Queue    Jobname     Nodes  S  Elapsed/Requested
   A = 372   malapaka  reg_64   jobscript-      8  R 09:04:58/23:00:00
   ■ = 373   malapaka  reg_64   jobscript-      8  R 05:36:08/23:00:00
       374   malapaka  reg_64   jobscript-      8  H       --/23:00:00
   s = 380   sahoo     reg_64   restart64.      8  R 00:31:44/23:00:00

[?] unknown  [@] busy  [*] down  [.] idle  [%] offline  [!] other
```

Visualization

- Paraview
  - Can handle VTK, HDF5 files
  - Lots of filters
  - Can be used with a remote machine that has the data
    - No need to move data

Visualization:2

- VTK files are native
  - Serial and parallel
  - ASCII and BINARY

- HDF5 files need a wrapper
  - XDMF, xml based describes underlying data
  - Can handle scalar, 3D, time-varying,....

Visualization: HDF5 files

- Example HDF5 file

```
HDF5 "cb_outP24_000001.h5" {
GROUP "/" {
   GROUP "PS3D" {
      DATASET "b" {
         DATATYPE  H5T_IEEE_F64LE
         DATASPACE  SIMPLE { ( 256, 256, 129 ) / ( 256, 256, 129 ) }
      }
   }
}
}
```

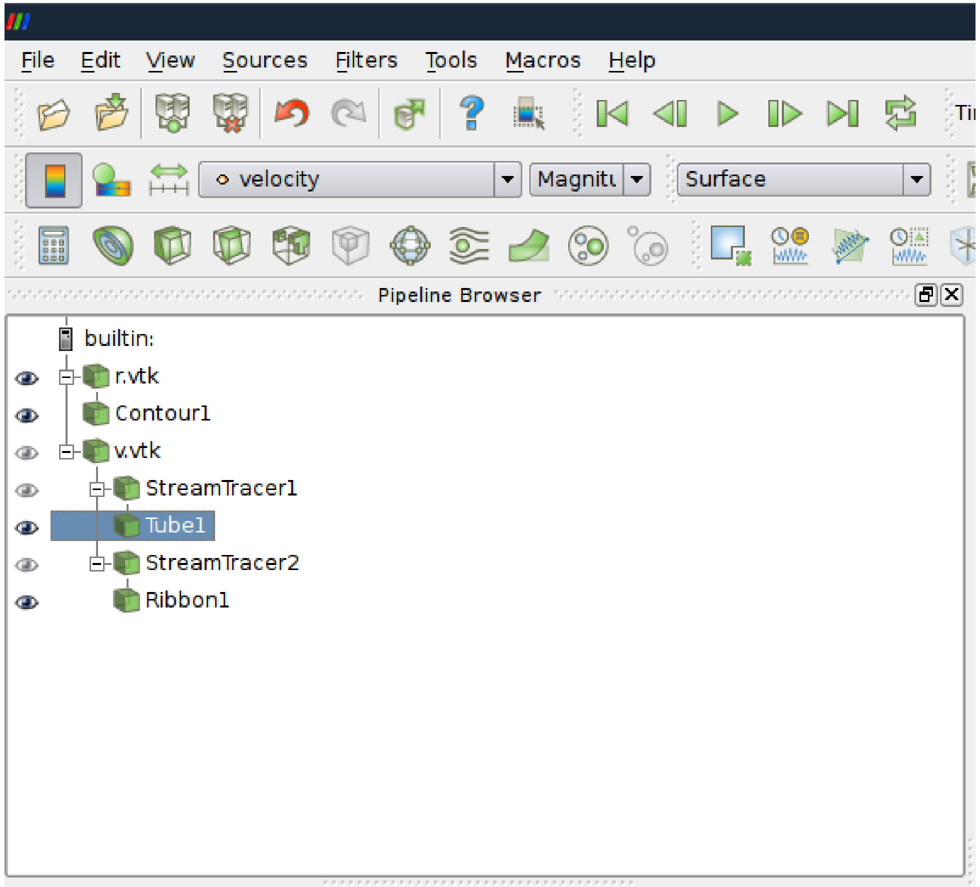
- Wrapper file

```
<?xml version="1.0" ?> <!DOCTYPE Xdmf SYSTEM "Xdmf.dtd" []>
<Xdmf>
<Domain>
<Grid Name="my_Grid" GridType="Uniform">
<Topology TopologyType="3DCoRectMesh" Dimensions="256 256 129" />

<Geometry GeometryType="Origin_DxDyDz">
 <DataItem Dimensions="3" NumberType="Integer" Format="XML">0 0 0</DataItem>
<DataItem Dimensions="3" NumberType="Integer" Format="XML"> 1 1 1</DataItem>
</Geometry>

 <Attribute Name="b" AttributeType="Scalar" Center="Node">
<DataItem Dimensions="256 256 129" NumberType="Double" Precision="8" Format="HDF">
cb_outP24_000001.h5:/PS3D/b
</DataItem>
</Attribute>
    </Grid>
  </Domain>
</Xdmf>
```

Paraview: Filters

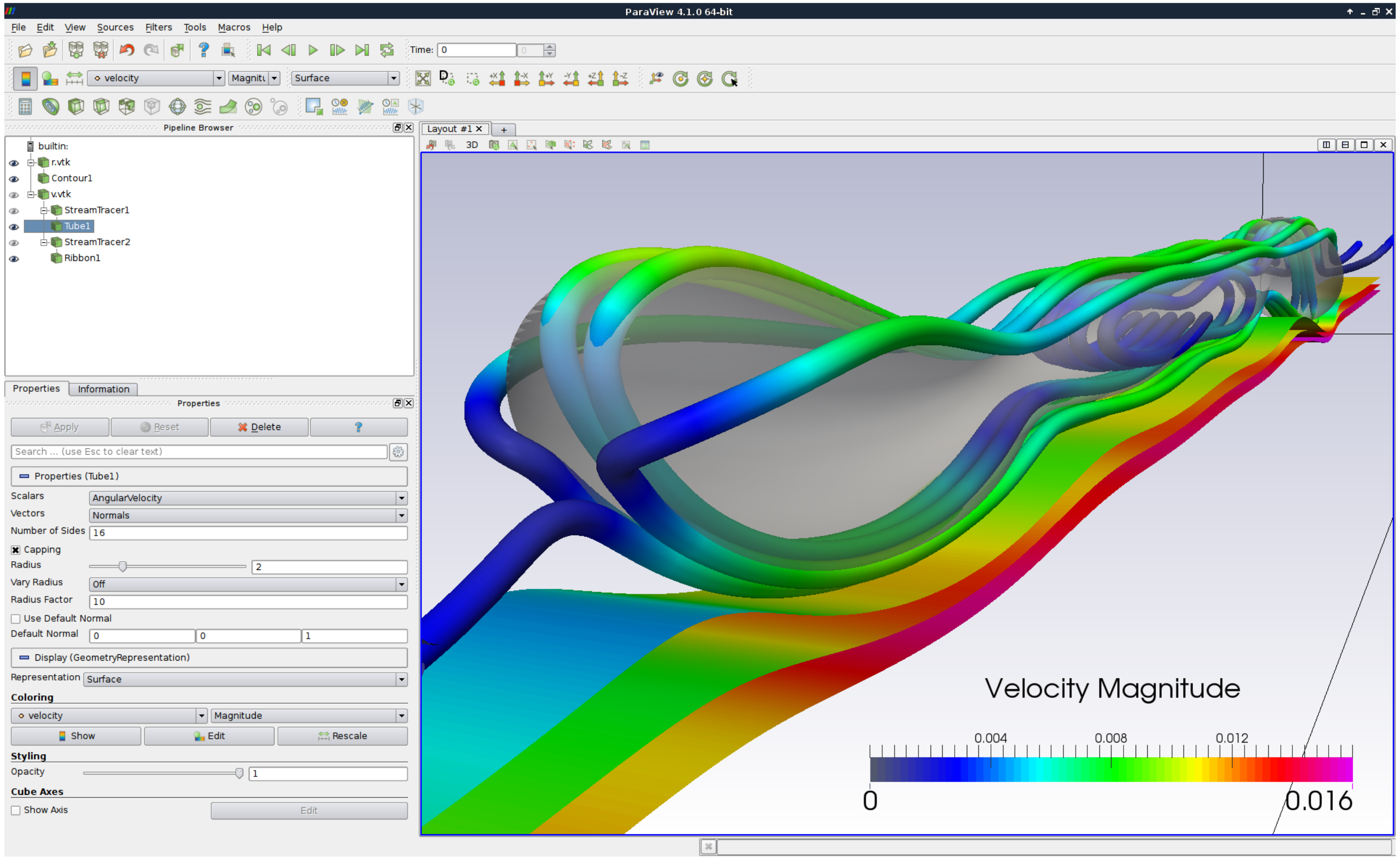
- Load data
- Manipulate with filters
  - Isosurfaces
  - Streamlines
  - Cutting
  - Projecting
- Pipelining metaphore
  - 1st filter output is 2nd filter input

Paraview: demo



builtin:
- r.vtk
  - Contour1
- v.vtk
  - StreamTracer1
    - Tube1
  - StreamTracer2
    - Ribbon1

Paraview:example



Properties (Tube1)

Scalars: AngularVelocity

Vectors: Normals

Number of Sides: 16

☒ Capping

Radius: 2

Vary Radius: Off

Radius Factor: 10

☐ Use Default Normal

Default Normal: 0 0 1

Display (GeometryRepresentation)

Representation: Surface

Coloring: velocity Magnitude

Styling

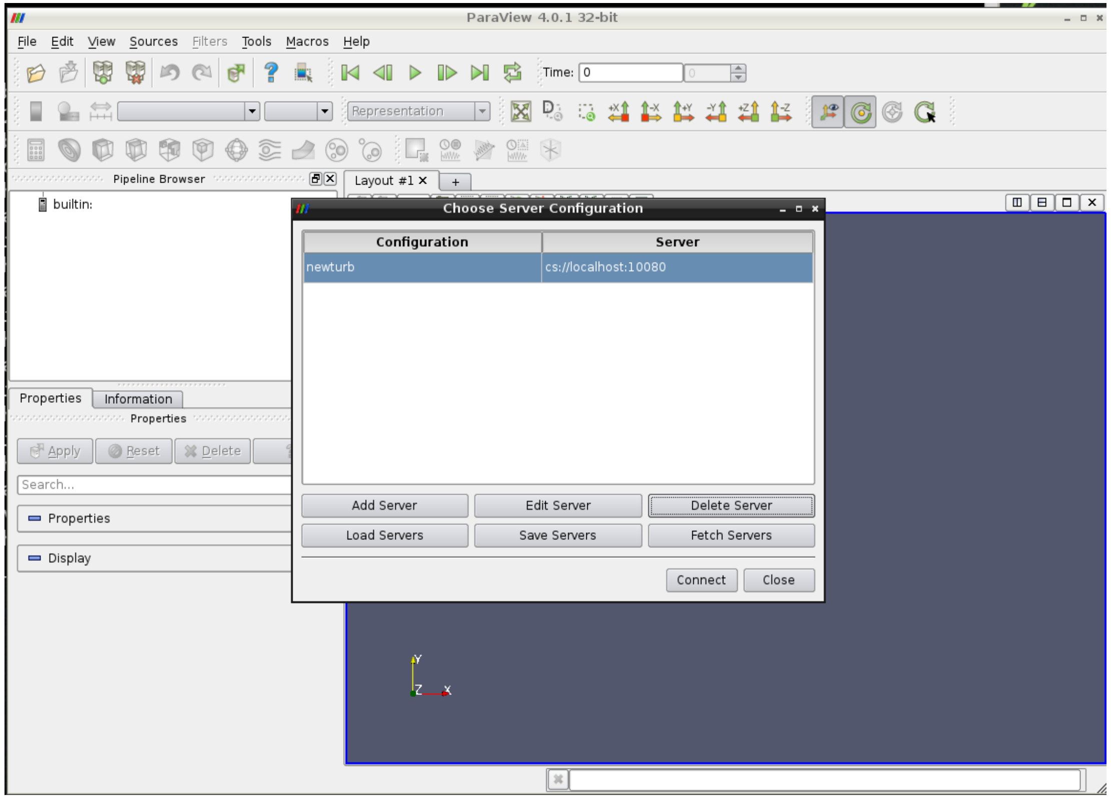
Opacity: 1

Cube Axes

☐ Show Axis

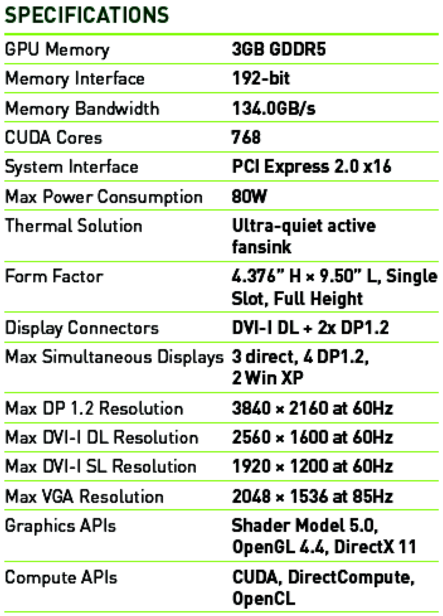
Paraview: client+server

- Version match
  - v4.0.1
- Load Config file
  - First time only
- File → Connect
- You're on newturb!



Configuration	Server
newturb	cs://localhost:10080

Quadro K4000 GPU



SPECIFICATIONS

GPU Memory	3GB GDDR5
Memory Interface	192-bit
Memory Bandwidth	134.0GB/s
CUDA Cores	768
System Interface	PCI Express 2.0 x16
Max Power Consumption	80W
Thermal Solution	Ultra-quiet active fansink
Form Factor	4.376" H × 9.50" L, Single Slot, Full Height
Display Connectors	DVI-I DL + 2x DP1.2
Max Simultaneous Displays	3 direct, 4 DP1.2, 2 Win XP
Max DP 1.2 Resolution	3840 × 2160 at 60Hz
Max DVI-I DL Resolution	2560 × 1600 at 60Hz
Max DVI-I SL Resolution	1920 × 1200 at 60Hz
Max VGA Resolution	2048 × 1536 at 85Hz
Graphics APIs	Shader Model 5.0, OpenGL 4.4, DirectX 11
Compute APIs	CUDA, DirectCompute, OpenCL

- 768 Kepler Cuda cores
- 800 Mhz, 2 FLOP/cyc
- ~1250TFlop/s SP

BUT

- 3 GB RAM
- HOST <-> GPU BW

NEXT

- Other nodes are arriving
  - Cluster will soon have little more then 512 procs
    - 2.5 TFLOP/s DP
  - RAM will be upgraded to 5.5 TB