



CENS-i arvutusklastar

Pearu Peterson, Marko Vendelin, Ardo Illaste

- CENS-i arvutusklastri ajalugu
- CENS-i arvutivõrgu struktuur
- Arvutusklastrite terminoloogiast
- SGE - arvutusklastri haldamise süsteem
- Arvutusklastri kasutamine, näited

CENS-i klaster

Algus



Slide 1 - 13

Tagasi

Täisekraan

Kinni

Lõpeta



1. CENS-i arvutusklastri ajalugu

Olav Kongas, Pearu Peterson, Marko Vendelin

1997 a. HP server ja 3 terminali

- Alpha lauaarvutite (>3) klaster
- Linuxi lauaarvutite klaster, Mosix
- AMD protsessoritel põhinev lauaarvutite klaster, Mosix, 32-bit, ca 25 arvutit
- AMD ja Opteroni protsessoritel põhinev lauaarvutite klaster, Mosix — hiljem SGE, 64-bit, ca 15 arvutit

Tüüpülesanne: parameetrite ruumi skanneerimine

CENS-i klaster

Algus



Slide 2 - 13

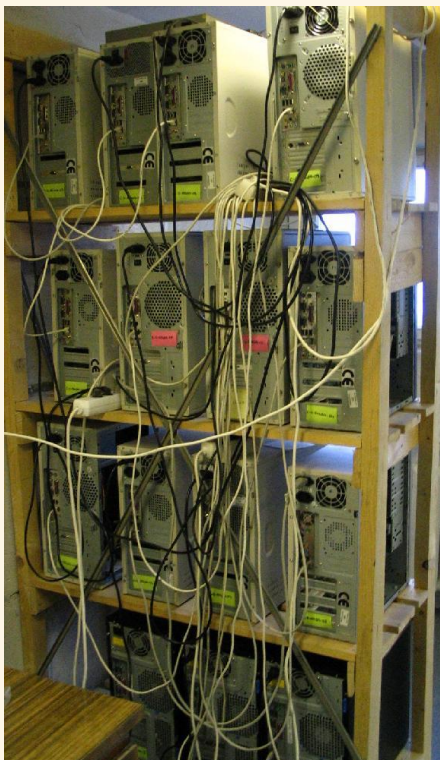
Tagasi

Täisekraan

Kinni

Lõpeta

2. Vana klaster



CPU: Athlon 64 (12tk), Opteron 244 (3tk)
RAM: 1 kuni 2 GB



CENS-i klaster

Algus



Slide 3 - 13

Tagasi

Täisekraan

Kinni

Lõpeta

3. Uus klaster



arvutusvõimuse
kasv: 18 x!

CPU: Opteron 2216 (47tk), Opteron 2210 (2tk), iga sõlm
kahe dualse protsessoriga
RAM: 4 GB, ühel sõlmel 16 GB



CENS-i klaster

Algus



Slide 4 - 13

Tagasi

Täisekraan

Kinni

Lõpeta



4. CENS-i arvutivõrgu struktuur

kev.ioc.ee - "kõigi eestlaste vanem" — kasutajate faili server, arvutusklastri server, X-terminalide server

cens.ioc.ee — E-maili ja veebi server — Xen-i virtuaalmasin kev'is

cowboy, fanatt, pc151, pc185, mech031, . . . — kasutajate X-terminalid

c-a-opt2216-0, c-o-ath64-00, c-o-opt244-0, . . .
— arvutusklastri sõlmed

kev-backup — backup server

CENS-i klaster

Algus



Slide 5 - 13

Tagasi

Täisekraan

Kinni

Lõpeta

5. Terminoloogia

- **Arvutusklast***er* e *grid* — kogum arvutusressurssidest, mis täidavad ülesandeid, kasutaja jaoks üks arvutusressurss.
- **Arvutusressurssid** — protsessorid, mälu, andmekandjad, I/O seadmed, tarkvara, nende litsentsid, jne.
- **Arvutussõlm** e *node* — arvutusressursside riistvaralise ühikud, sisaldab 1 või rohkem arvutusühikut
- **Arvutusühik** e *slot* — üks ülesannete täitmiseks ettenähtud protsessor
- **Ülesanne** e *job* — kasutaja programm, mida ta saadab arvutusklastrisse täitmiseks; ülesannetele saab määrata ressursside vajadused
- **Järjekord** e *queue* — kogum ülesannetest, mida täidetakse samaaegselt ülesannete vajadustele vastavate ressurssidega arvutussõlmedes; järjekorral on defineeritud poliitika

CENS-i arvutusklast^{er}: 62 arvutussõlme, 208 arvutusühikut



CENS-i klaster

Algas

◀ ▶

◀ ▶

Slide 6 - 13

Tagasi

Täisekraan

Kinni

Lõpeta

6. SGE e Sun Grid Engine

- Kasutaja annab SGE süsteemile ülesande
 - kasutaja deklareerib ressursside vajadused
 - ülesande algolek on *ootel* e *pending*
 - määratakse kasutaja *prioriteet*, fikseeritakse ülesande andmise aeg
- Järjekorras vabaneb ressurss
 - SGE leiab ülesande, millel on kõrgeim prioriteet ja/või pikim ooteaeg ning mille vajadused oleks rahuldatud vabanenud ressursside arvelt
 - ülesanne lisatakse sobivasse järjekorda ehk käivitatakse; ülesanne on seotud järjekorraga
 - ülesande lõppemisel see eemaldatakse järjekorrast, ehk vabastatakse ressurss järgmise ülesande jaoks

CENS-i arvutusklasteri:

- saab käivitada täpselt üks ülesanne ühel arvutusühikul
- on defineeritud üks üldine järjekord: `a11.q`



CENS-i klaster

Algas



Slide 7 - 13

Tagasi

Täisekraan

Kinni

Lõpeta



7. SGE kasutaja liides

X-i kasutajaliidese programm —

qmon

Käsurea kasutajaliidese programmid —

qacct, qalter, qconf, qdel, qhold, **qhost**,
qlogin, qmake, qmod, qresub, qrls, **qrsh**, qs-
elect, qsh, **qstat**, **qsub**, qtcsh

Veebi info

- **qstat** —

<http://cens.ioc.ee/qstat/>

- HOWTO dokumendid —

<http://cens.ioc.ee/cens/local/howtos/using-cens-cluster>

CENS-i klaster

Algus



Slide 8 - 13

Tagasi

Täisekraan

Kinni

Lõpeta

8. Näide 1: shelli skripti käivitamine

```
#!/bin/sh
#
# Fail: simple.sh
#

date          # N"aita kuup"aeva ja kellaaega
echo "Magama 2 sekundit"
sleep 2
echo "Magama veel 5 sekundit"
sleep 5
echo "Valmis!"
date          # N"aita kuup"aeva ja kellaaega

# Skript faili lqpp
kev:$ qsub -cwd simple.sh

pearu@kev:~/demo$ qsub -cwd simple.sh
Your job 1319 ("simple.sh") has been submitted
pearu@kev:~/demo$ tail -f simple.sh.o1319
Wed Feb 21 13:10:09 EET 2007
Magama 2 sekundit
Magama veel 5 sekundit
Valmis!
Wed Feb 21 13:10:16 EET 2007
```



CENS-i klaster

Algus



Slide 9 - 13

Tagasi

Täisekraan

Kinni

Lõpeta

9. Näide 2: Pythoni skripti käivitamine

```
#!/usr/bin/env python
#
# Fail: simple.py
#
import time

print time.asctime () # N"aita kuup"aeva ja kellaaega
print "Magama 2 sekundit"
time.sleep(2)
print "Magama veel 5 sekundit"
time.sleep(5)
print "Valmis!"
print time.asctime () # N"aita kuup"aeva ja kellaaega

# Skript faili lqpp

kev:$ qsub -cwd simple.py

pearu@kev:~/demo$ qsub -cwd simple.py
Your job 1321 ("simple.py") has been submitted
pearu@kev:~/demo$ cat simple.py.e1321
import: unable to open X server ``.
...
```



CENS-i klaster

Algus



Slide 10 - 13

Tagasi

Täisekraan

Kinni

Lõpeta



10. Näide 2': Pythoni skripti käivitamine

```
#!/bin/sh
#
# Fail: python.sh
#
/usr/bin/env python "$*"
# Skript faili lõpp
```

```
kev:$ qsub -cwd `which python.sh` simple.py
```

```
pearu@kev:~/demo$ qsub -cwd `which python.sh` simple.py
Your job 1324 ("python.sh simple.py") has been submitted
pearu@kev:~/demo$ tail -f python.sh.o1324
Wed Feb 21 13:35:08 2007
Magama 2 sekundit
Magama veel 5 sekundit
Valmis!
Wed Feb 21 13:35:15 2007
```

CENS-i klaster

Algus



Slide 11 - 13

Tagasi

Täisekraan

Kinni

Lõpeta



11. Näide 3: Interaktiivsed programmid

Ühendus vaba arvutussõlmega luuakse SSH kaudu.

XTerm —

```
kev:$ qrsh -cwd -now no xterm
```

XMaple —

```
kev:$ qrsh -cwd -now no xmaple
```

CENS-i klaster

Algus



Slide 12 - 13

Tagasi

Täisekraan

Kinni

Lõpeta

12. Näide 4: Parameetri ruumi skaneerimine

```
qsub -cwd -t 1-100:1 python.sh scan.py

#!/usr/bin/env python
# File: scan.py
import os, sys, math, time
from random import uniform

# Get task ID
task_id = int(os.environ.get('SGE_TASK_ID', '0'))

# Helper functions.
def lock(): # cache file
def release(): # cache file
def get_cache(): # from file
def put_cache(cache): # to file

if task_id==0: # show results
    ...
else:
    # calculate
    x = task_id*math.pi/100.0
    y = math.sin(x)
    time.sleep(uniform(5,25))
    # Save results
    lock(); cache = get_cache()
    cache[task_id] = (x,y)
    put_cache(cache); release()

#EOF
```



CENS-i klaster

Algus



Slide 13 - 13

Tagasi

Täisekraan

Kinni

Lõpeta