Linux on zSeries - What's new?

Klaus Bergmann
Linux on zSeries Performance, IBM Lab Boeblingen
klaus_bergmann@de.ibm.com
2003/04/26
The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

Enterprise Storage Server
ESCON*
FICON
FICON Express
HiperSockets
IBM*
IBM logo*
IBM eServer
Netfinity*
S/390*
VM/ESA*
WebSphere*
z/VM
zSeries

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Intel is a trademark of the Intel Corporation in the United States and other countries.
Java and all Java-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries.

Lotus, Notes, and Domino are trademarks or registered trademarks of Lotus Development Corporation.

Linux is a registered trademark of Linus Torvalds.

Microsoft, Windows and Windows NT are registered trademarks of Microsoft Corporation.
Penguin (Tux) compliments of Larry Ewing.

SET and Secure Electronic Transaction are trademarks owned by SET Secure Electronic Transaction LLC.
UNIX is a registered trademark of The Open Group in the United States and other countries.

* All other products may be trademarks or registered trademarks of their respective companies.
Agenda

- Introduction
- Linux inventory
- What's new?
  - SCSI
  - LVM
  - PAV
  - Dynamic device attachment
  - snIPL
  - VIPA
  - IPv6
  - VLAN
  - Useful Linux commands
Linux for S/390 first steps

- Get Linux running on the platform in general
- Get a console
- Get a DASD device
- Get a networking device
- Get SMP running
- Run it under VM
- Compile applications
- Get it stable
- Get it installable

Show it to the world
Linux on zSeries goals

- Become as reliable as z/OS
- Improve performance
- Exploit platform hardware
  - SCSI, iQDIO, IPv6, 3590, FICON
- Improve customer service
- Linux as the first exploiter of new hardware
- Be up to date with Open Source

Make business
SLES8 inventory

- Linux Kernel 2.4.19
- gcc 3.2-31
- glibc 2.2.5-84
- Network Support
  - Fast/Gigabit Eth, Hipersockets, FICON/ESCON CTC, TR, HSTR
- Disk / Tape
  - ECKD DASD, Minidisk, **SCSI**, 3480, 3490, 3590
- 31/64 Bit Support
- Timer Patch
Support for Fibre Channel attached SCSI devices
- tapes, disks, CD-ROM, DVD ...
- Storage Area Networks (SAN) integration
- Requires a z800 or z900 with GA3 + MCL fix
- Fibre Channel switch necessary
- SCSI disk can be much larger than ECKD disk
- Up to 128 SCSI disks per Linux system
- Faster than ECKD I/O
- Boot from SCSI disk currently not possible
- Multiple I/O to device

Exploits new hardware for the platform
SCSI versus ECKD

SCSI
- SCSI disk driver
- SCSI core (scsi_mod.o)
- zfcp
- qdio
- FCP Switch
- ESS
- Disk

ECKD
- DASD driver
- CCW program
- FICON
- ESCON
- ESS
- Disk

Color Coding: Architecture dependent
Architecture independent
Hardware
cat /proc/subchannels
Device sch. Dev Type/Model CU in use PIM PAM POM LPUM CHPIDs
------------------------------------------------------------------------
5901 000F 1732/03 1731/03 yes 80 80 FF 00 2A000000 00000000

cat /proc/scsi/zfcp/map
0x5901 0x00000001:0x5005076300c393cb 0x00000000:0x517e000000000000
0x5901 0x00000003:0x5005076300cc93cb 0x00000000:0x547e000000000000
0x5901 0x00000003:0x5005076300cc93cb 0x00000001:0x547f000000000000
ESSPort -> WWPN -> SCSI target ESS Disk -> FCP LUN -> SCSI LUN

Provided by the ESS Admin

WWPN (World Wide Port Name)
Provided by the adapter itself
"Burned in"

mount /dev/scsi/host0/bus0/target1/lun0/part1 /mnt/my-scsi-disc1
mount /dev/scsi/host0/bus0/target3/lun1/part1 /mnt/my-scsi-disc2
Add multiple physical DASD to one logical volume
Create logical volume larger than 3390 Model 9
- up to 255 GB
Parallel I/O to logical volume possible by striping
- Better Performance
Logical volume size can be changed dynamically if striping is not used
Included in Kernel 2.4
VM/ESA

Guest

Parallel Access Volumes with VM
A lab experiment

Linux cannot enable PAV on the ESS but can use it under VM
IODEVICE ADDRESS=(5680,024),UNITADD=00,CUNUMBR=(5680), STADET=Y,UNIT=3390B
IODEVICE ADDRESS=(5698,040),UNITADD=18,CUNUMBR=(5680), STADET=Y,UNIT=3390A

ATTACH Base and Aliases to the guest

QUERY PAV shows base and alias addresses

cat /proc/dasd/devices
5794(ECKD) at ( 94:  0) is dasda : active at blocksize: 4096, 1803060 blocks, 7043 MB
5593(ECKD) at ( 94:  4) is dasdb : active at blocksize: 4096, 601020 blocks, 2347 MB
5680(ECKD) at ( 94:  8) is dasdc : active at blocksize: 4096, 1803060 blocks, 7043 MB
56bf(ECKD) at ( 94: 12) is dasdd : active at blocksize: 4096, 1803060 blocks, 7043 MB

cat /proc/subchannels | egrep "5680|56BF"
5680 0030 3390/0C 3990/E9 yes FC FC FF C6C7C8CA CBC90000
56BF 0031 3390/0C 3990/E9 yes FC FC FF C6C7C8CA CBC90000

This works only with LVM!
- `vgscan`: create configuration data
- `pvcreate` `/dev/dasdc1`
- `vgcreate` `vg_kb` `/dev/dasdc1`
- `vgdisplay`
vgdisplay -v vg_kb

--- Volume group ---
VG Name               vg_kb
VG Access             read/write
VG Status             available/resizable
VG #                  0
MAX LV                256
Cur LV                0
Open LV               0
MAX LV Size           255.99 GB
Max PV                256
Cur PV                1
Act PV                1
VG Size               6.87 GB
PE Size               4 MB
Total PE              1759
Alloc PE / Size       0 / 0
Free PE / Size        1759 / 6.87 GB
VG UUID               3nwJYn-SxWl-gKym-OvZs-TYIf-CrHP-inO5Yp

--- No logical volumes defined in "vg_kb" ---
More LVM commands

```
lvcreate --name lv_kb --extents 1759 vg_kb

cat /proc/lvm/global
LVM module LVM version 1.0.5(mp-v6)(15/07/2002)
Total:  1 VG  1 PV  1 LV (0 LVs open)
Global: 32300 bytes malloced   IOP version: 10   3:18:35 active

VG:  vg_kb  [1 PV, 1 LV/0 open]  PE Size: 4096 KB
   Usage [KB/PE]: 7204864 /1759 total  7204864 /1759 used  0 /0 free
PV:  [AA] dasdc1                 7204864 /1759     7204864 /1759           0 /0
     +-- dasdd1
LV:  [AWDL ] lv_kb                      7204864 /1759     close
```

```
_lvscan
_lvscan -- ACTIVE            "/dev/vg_kb/lv_kb" [6.87 GB]
_lvscan -- 1 logical volumes with 6.87 GB total in 1 volume group
_lvscan -- 1 active logical volumes
```
pvpath -qa
Physical volume /dev/dasdcl of vg_kb has 2 paths:

<table>
<thead>
<tr>
<th>Device</th>
<th>Weight</th>
<th>Failed</th>
<th>Pending</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>#  0:</td>
<td>94:9</td>
<td>0</td>
<td>0</td>
<td>0 enabled</td>
</tr>
<tr>
<td>#  1:</td>
<td>94:13</td>
<td>0</td>
<td>0</td>
<td>0 disabled</td>
</tr>
</tbody>
</table>

The second path can be enabled:
pvpath -p1 -ey /dev/dasdcl
vg_kb: setting state of path #1 of PV#1 to enabled

pvpath -qa
Physical volume /dev/dasdcl of vg_kb has 2 paths:

<table>
<thead>
<tr>
<th>Device</th>
<th>Weight</th>
<th>Failed</th>
<th>Pending</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>#  0:</td>
<td>94:9</td>
<td>0</td>
<td>0</td>
<td>0 enabled</td>
</tr>
<tr>
<td>#  1:</td>
<td>94:13</td>
<td>0</td>
<td>0</td>
<td>0 enabled</td>
</tr>
</tbody>
</table>

Now LVM is ready to use both paths to the volume
These are preliminary results in a controlled environment. PAV is not yet officially supported with Linux on zSeries!

<table>
<thead>
<tr>
<th>Paths</th>
<th>Write (MB/s)</th>
<th>CPU-load(%)</th>
<th>Read (MB/s)</th>
<th>CPU-load(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.9</td>
<td>6.3</td>
<td>27.0</td>
<td>10.8</td>
</tr>
<tr>
<td>2</td>
<td>18.7</td>
<td>7.7</td>
<td>46.4</td>
<td>19.7</td>
</tr>
<tr>
<td>3</td>
<td>22.4</td>
<td>9.7</td>
<td>65.9</td>
<td>27.0</td>
</tr>
<tr>
<td>4</td>
<td>23.4</td>
<td>11.0</td>
<td>81.4</td>
<td>36.8</td>
</tr>
<tr>
<td>5</td>
<td>23.2</td>
<td>10.5</td>
<td>96.9</td>
<td>39.2</td>
</tr>
<tr>
<td>6</td>
<td>22.6</td>
<td>10.8</td>
<td>106.7</td>
<td>43.8</td>
</tr>
<tr>
<td>7</td>
<td>21.2</td>
<td>11.3</td>
<td>106.7</td>
<td>47.9</td>
</tr>
<tr>
<td>8</td>
<td>21.1</td>
<td>11.3</td>
<td>119.0</td>
<td>50.5</td>
</tr>
</tbody>
</table>
Dynamic device attach

- Possible for DASD and network devices
- Network device example:
  - attach Gigabit card to system which is not included in /etc/chandev.conf for test purposes
  - check which cards are already attached to Linux

```
cat /proc/qeth
```

```
devnos (hex)   CHPID  device    cardtype  port  chksum  prio-q'ing  rtr   fsz  C  cnt
------------   -----  ---------  --------  ----  ------  ----------  ----  ---  --  ---
F100/F101/F102 xf6  eth0    OSD_100  0      no   always q  2    no  64k  128
```

- add Gigabit card with addresses F400,F401,F402
  - first possibility

```
echo qeth1,0xF400,0xF401,0xF402,0,0,0 > /proc/chandev
echo addParms,0x10,0xf400,0xf402,0,0,0 > /proc/chandev
echo reprobe > /proc/chandev
```

```
cat /proc/qeth
```

```
devnos (hex)   CHPID  device    cardtype  port  chksum  prio-q'ing  rtr   fsz  C  cnt
------------   -----  ---------  --------  ----  ------  ----------  ----  ---  --  ---
F400/F401/F402 xo2  eth1    OSD_1000  0      no   always q  2    no  64k  128
F100/F101/F102 xf6  eth0    OSD_100  0      no   always q  2    no  64k  128
```
Dynamic device attach

- add Gigabit card with addresses F400,F401,F402
  - second possibility
    - add next two lines to your /etc/chandev.conf
      
      ```
      qeth1, 0xF400, 0xF401, 0xF402, 0, 0, 0
      add_parms, 0x10, 0xF400, 0xF402, portname: PERF
      ```
      
      ```
      echo read_conf > /proc/chandev
      echo reprobe > /proc/chandev
      ```
      
      `cat /proc/qeth`
      
      ```
      devnos (hex)   CHPID   device     cardtype  port  chksum priq-q'ing  rtr  fsz   C cnt
        --------------- --- ---------- -------------- --     -- ---------- --- --- - ---
      F400/F401/F402 x02       eth1       OSD_1000  0     no  always q 2  no 64k   128
      F100/F101/F102 xF6       eth0        OSD_100  0     no  always q 2  no 64k   128
      ```

- bring up card
  
  ```
  ifconfig eth1
  ```
  
  ```
  eth1 Link encap:Ethernet  HWaddr 00:02:55:9A:12:73
  MULTICAST MTU:1492 Metric:1
  RX packets:0 errors:0 dropped:0 overruns:0 frame:0
  TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
  collisions:0 txqueuelen:100
  Interrupt:34
  ```
**Dynamic device attach**

- **DASD device example**
  - add DASD device which is not included in /boot/zipl/parmfile
  - check which DASD devices already attached to Linux

```
/proc/dasd/devices
5794(ECKD) at ( 94:  0) is dasda : active at blocksize: 4096, 1803060 blocks, 7043 MB
5593(ECKD) at ( 94:  4) is dasdb : active at blocksize: 4096, 601020 blocks, 2347 MB
```

add DASD device 5788 to the system
```
echo "add device 5788" > /proc/dasd/devices
```
```
cat /proc/dasd/devices
5794(ECKD) at ( 94:  0) is dasda : active at blocksize: 4096, 1803060 blocks, 7043 MB
5593(ECKD) at ( 94:  4) is dasdb : active at blocksize: 4096, 601020 blocks, 2347 MB
5788(ECKD) at ( 94:  8) is dasdc : active at blocksize: 4096, 1803060 blocks, 7043 MB
```

- **mount the DASD**
  - `mount /dev/dasdc1 /mnt`
snIPL (simple network IPL)

- Is an interactive tool to remotely control Support Element functions. It allows you to:
  - Boot Linux for zSeries in LPAR mode
  - Send and retrieve operating system messages
  - Deactivate an LPAR
- Runs under Linux (Intel/zSeries)
- snIPL uses network management API which:
  - uses the SNMP protocol to send and retrieve data
- snIPL currently supports only a direct connection to the SE and does not yet support direct connection to the HMC
- SE must be enabled for snIPL access
- Can be found at developer works:
snIPL

# /sbin/snipl <IP Support Element>
snIPL - simple network IPL
available LPARs:

PEL1     PEL2     PEL3     PEL6

Please specify the LPAR's name to operate on (CTRL-D to abort): PEL6

Command (m for help): m
  n   select LPAR image
  i   operating system messages interaction
  l   perform a load
  d   perform a deactivate
  m   print this menu
  x   exit

Command (m for help): l

Please specify the following parameters (CTRL-D uses default value):

Load address (as XXXX in HEX): 5702
Load parameter:
Clear indicator (0/1):
Timeout:
Store status indicator (0/1):

You have specified the following parameters:

Load address: 5702
Load parameter:
Clear indicator: 0
Timeout: 60s
Store status indicator: 0
Perform a LOAD command on partition PEL6 with these parameters? (y/n) y
processing.... acknowledged.
Command (m for help): m
  n   select LPAR image
  i   operating system messages interaction
  l   perform a load
  d   perform a deactivate
  m   print this menu
  x   exit

Command (m for help): i
Starting operating system messages interaction for
partition PEL6 (CTRL-D to abort):

Linux version 2.4.17-0tape-dasd (root@pserver16) (gcc version 2.95.3 20010315 (release)) #1 SMP Wed May 8 16:14:30 CEST 2002
We are running native (31 bit mode)
This machine has an IEEE fpu
On node 0 totalpages: 491520
zone(0): 491520 pages.
zone(1): 0 pages.
zone(2): 0 pages.
Kernel command line: maxcpus=4 dasd=5702-5707,5721-5754,5502-5505 root=/dev/dasd
/5702/part1 ro noinitrd
Highest subchannel number detected (hex) : 4DF6
VIPA (Virtual IP Address)

- Minimize outage due to adapter failure
- Facility for assigning an IP address to a system instead of individual adapters

Linux LPAR, or Linux Guest

- Adapter (OSA card) `eth0`=9.164.188.10
- Adapter (OSA card) `eth1`=9.164.188.11
- Dummy Device VIPA=9.164.188.100

The other adapter will take over the IP traffic if one adapter (eth0 or eth1) fails

Network

Router

All connections initiated from the Network to the VIPA will survive the IP traffic take over. Connections initiated from the Dummy device to the Network will die.
VIPA (Virtual IP Address)

Prereq.: Kernel built with CONFIG_DUMMY switched on

Setup:
- Create a dummy device
  ```
  insmod dummy
  ```
- Assign a virtual IP address (9.164.188.100) to the device
  ```
  ifconfig dummy0 9.164.188.100
  ```
- Enable VIPA on the network devices
  ```
  echo add_vipa4 09A4BC64:eth0 > /proc/qeth_ipa_takeover
  echo add_vipa4 09A4BC64:eth1 > /proc/qeth_ipa_takeover
  ```
- Setup routes to the virtual IP address
  - Static
    ```
    route add -host 9.164.188.100 gw 9.164.188.10
    ```
    or
    ```
    route add -host 9.164.188.100 gw 9.164.188.11
    ```
  - Dynamic by installing a routing daemon like `zebra` or `gated`
IPv6

- Linux for zSeries support for IPv6 applies to Gigabit Ethernet and Fast Ethernet only at the moment.
- Some concepts in IPv6 are different from IPv4, such as neighbor discovery, broadcast, and IPSec.
- From a user point of view the impact of IPv6 is largely limited to the specification of IP addresses
  - 128 Bit that gives 6*E28 addresses per person
  - addresses will be specified in hex format
    3ffe:0400:0280:0:0:0:0:1
  - Leading zeros can be omitted
    3ffe:400:280:0:0:0:0:1
  - First set of concurrent zeros can be omitted
    3ffe:400:280::1
  - IPv4 addresses can be used within IPv6 address range
    139.18.38.71 -------> ::ffff:8b12:2647
IPv4 tools versus IPv6 tools

<table>
<thead>
<tr>
<th>IPv4</th>
<th>IPv6</th>
</tr>
</thead>
<tbody>
<tr>
<td>ping</td>
<td>ping6</td>
</tr>
<tr>
<td>ftp</td>
<td>ncftp</td>
</tr>
<tr>
<td>ssh</td>
<td>ssh -6</td>
</tr>
<tr>
<td>scp</td>
<td>scp -6</td>
</tr>
<tr>
<td>telnet</td>
<td>telnet</td>
</tr>
<tr>
<td>wget</td>
<td>wget -6</td>
</tr>
<tr>
<td>traceroute</td>
<td>traceroute6</td>
</tr>
<tr>
<td>ifconfig &lt;dev&gt;</td>
<td>ifconfig add &lt;dev&gt;</td>
</tr>
</tbody>
</table>
VLAN (Virtual Local Area Network)

- VLAN logically segments the network into different virtual networks
- Organize your network by traffic patterns rather than by physical location
- Members of a VLAN can be part of different physical LANs
Example-create two VLANs (red, green)

Definitions for PC1 (red & green)

```
ifconfig eth1 9.164.188.10 netmask 255.255.224.0
```
Creates two VLAN with id 3 and 5 on physical device eth1

```
vconfig add eth1 3
vconfig add eth1 5
```

Configure VLAN devices

```
ifconfig eth1.3 10.0.0.1 netmask 255.255.255.0 up
ifconfig eth1.5 1.2.3.4 netmask 255.255.0.0 up
```

Definitions for LPAR (red)

```
ifconfig eth0 9.164.188.12 netmask 255.255.224.0
```
Creates one VLAN with id 3 on physical device eth0

```
vconfig add eth0 3
```

Configure VLAN devices

```
ifconfig eth0.3 10.0.0.2 netmask 255.255.255.0 up
```

Check configuration on PC1

```
cat /proc/net/vlan/config
```

<table>
<thead>
<tr>
<th>VLAN Dev name</th>
<th>VLAN ID</th>
<th>Name-Type: VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD</th>
<th>bad_proto_recvd: 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth1.3</td>
<td>3</td>
<td>eth1</td>
<td></td>
</tr>
<tr>
<td>eth1.5</td>
<td>5</td>
<td>eth1</td>
<td></td>
</tr>
</tbody>
</table>
Information about VLAN devices

```
cat /proc/net/vlan/eth1.5
eth1.5  VID: 5   REORDER_HDR: 1  dev->priv_flags: 1
   total frames received:  10914061
   total bytes received:   1291041929
   Broadcast/Multicast Rcvd:  6
   total frames transmitted: 10471684
   total bytes transmitted: 4170258240
   total headroom inc: 0
   total encap on xmit: 10471684
```

Device: eth1
INGRESS priority mappings: 0:0 1:0 2:0 3:0 4:0 5:0 6:0 7:0
EGRESSS priority Mappings:

- 4096 VLAN devices can be created per physical device
- Broad- and multicasts will be sent only to the specific VLAN not to the whole network!
- Less traffic within the LAN
Useful Linux commands

fdasd-allows you to split a DASD into several partitions

/dev/dasdd
reading volume label: VOL1
reading vtoc : ok

Command action
   m  print this menu
   p  print the partition table
   n  add a new partition
   d  delete a partition
   v  change volume serial
   t  change partition type
   r  re-create VTOC and delete all partitions
   u  re-create VTOC re-using existing partition sizes
   s  show mapping (partition number - data set name)
   q  quit without saving changes
   w  write table to disk and exit

Command (m for help): p

Disk /dev/dasdd:
   3339 cylinders,
      15 tracks per cylinder,
      12 blocks per track
   4096 bytes per block
volume label: VOL1, volume identifier: 0X5710
maximum partition number: 3

-----------tracks-----------
Device    start  end    length   Id   System
/dev/dasdd1    2  16001  16000   1 Linux native
Dasdview-delivers information about a given DASD device or displays the contents of a disk dump

dasdview -ixf /dev/dasdd

--- general DASD information -----------------------------------------------------------
device node               : /dev/dasdd
device number             : hex 5710       dec 22288
type                      : ECKD
device type               : hex 3390       dec 13200

--- DASD geometry --------------------------------------------------------------
number of cylinders       : hex d0b        dec 3339
tracks per cylinder       : hex f          dec 15
blocks per track          : hex c          dec 12
blocksize                 : hex 1000       dec 4096

--- extended DASD information -----------------------------------------------------
real device number        : hex 0          dec 0
subchannel identifier     : hex 31         dec 49
CU type (SenseID)         : hex 3990       dec 14736
CU model (SenseID)        : hex e9         dec 233
device type (SenseID)     : hex 3390       dec 13200
device model (SenseID)    : hex a          dec 10
open count                : hex 1          dec 1
req_queue_len             : hex 0          dec 0
chandq_len                : hex 0          dec 0
status                    : hex 6          dec 6
label_block               : hex 2          dec 2
FBA_layout                : hex 0          dec 0
characteristics_size      : hex 40         dec 64
confdata_size             : hex 100        dec 256
Useful Linux commands

cat /proc/sysinfo

cat /proc/sysinfo
Manufacturer: IBM
Type: 2064
Model: 216
Sequence Code: 0000000000051539
Plant: 02

CPUs Total: 17
CPUs Configured: 16
CPUs Standby: 0
CPUs Reserved: 1
Capability: 2928
Adjustment 02-way: 95
Adjustment 03-way: 91
...
Adjustment 17-way: 0

LPAR Number: 3
LPAR Characteristics: Shared
LPAR Name: PEV1
LPAR Adjustment: 750
LPAR CPUs Total: 12
LPAR CPUs Configured: 12
LPAR CPUs Standby: 0
LPAR CPUs Reserved: 0
LPAR CPUs Dedicated: 0
LPAR CPUs Shared: 12

VM00 Name: BERGMANN
VM00 Control Program: z/VM 4.3.0
VM00 Adjustment: 250
VM00 CPUs Total: 3
# Useful Linux commands

## cat /proc/partitions

<table>
<thead>
<tr>
<th>major</th>
<th>minor</th>
<th>#blocks</th>
<th>name</th>
<th>rio</th>
<th>rmerge</th>
<th>rsect</th>
<th>ruse</th>
<th>wio</th>
<th>wmerge</th>
<th>wsect</th>
<th>wuse</th>
<th>running</th>
<th>use</th>
<th>aveq</th>
</tr>
</thead>
<tbody>
<tr>
<td>94</td>
<td>0</td>
<td>721240</td>
<td>dasda</td>
<td>13630</td>
<td>6713</td>
<td>169944</td>
<td>51110</td>
<td>3186</td>
<td>83096</td>
<td>118290</td>
<td>0</td>
<td>52530</td>
<td>169390</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>1</td>
<td>721244</td>
<td>dasda1</td>
<td>13607</td>
<td>7598</td>
<td>169640</td>
<td>51030</td>
<td>3186</td>
<td>83088</td>
<td>118290</td>
<td>0</td>
<td>52450</td>
<td>169310</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>4</td>
<td>2404080</td>
<td>dasdb</td>
<td>21</td>
<td>58</td>
<td>632</td>
<td>60</td>
<td>18</td>
<td>92</td>
<td>1056</td>
<td>100</td>
<td>0</td>
<td>80</td>
<td>120</td>
</tr>
<tr>
<td>94</td>
<td>5</td>
<td>2403984</td>
<td>dasdb1</td>
<td>14</td>
<td>58</td>
<td>576</td>
<td>20</td>
<td>18</td>
<td>92</td>
<td>1056</td>
<td>100</td>
<td>0</td>
<td>80</td>
<td>120</td>
</tr>
<tr>
<td>94</td>
<td>8</td>
<td>721240</td>
<td>dasdc</td>
<td>55</td>
<td>53</td>
<td>864</td>
<td>432</td>
<td>2758</td>
<td>227144</td>
<td>346460</td>
<td>0</td>
<td>9810</td>
<td>347990</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>9</td>
<td>721244</td>
<td>dasdc1</td>
<td>19</td>
<td>23</td>
<td>336</td>
<td>1460</td>
<td>430</td>
<td>27588</td>
<td>227128</td>
<td>346450</td>
<td>0</td>
<td>9730</td>
<td>347910</td>
</tr>
<tr>
<td>94</td>
<td>12</td>
<td>2404080</td>
<td>dasdd</td>
<td>49</td>
<td>30</td>
<td>632</td>
<td>30</td>
<td>80</td>
<td>70</td>
<td>70</td>
<td>10</td>
<td>0</td>
<td>70</td>
<td>70</td>
</tr>
<tr>
<td>94</td>
<td>13</td>
<td>768000</td>
<td>dasdd1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>94</td>
<td>14</td>
<td>768000</td>
<td>dasdd2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>94</td>
<td>15</td>
<td>768000</td>
<td>dasdd3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

## cat /proc/chpids

- C6 online
- C7 online
- C8 online
- C9 online
- CA online
- CB online
- F5 online

...