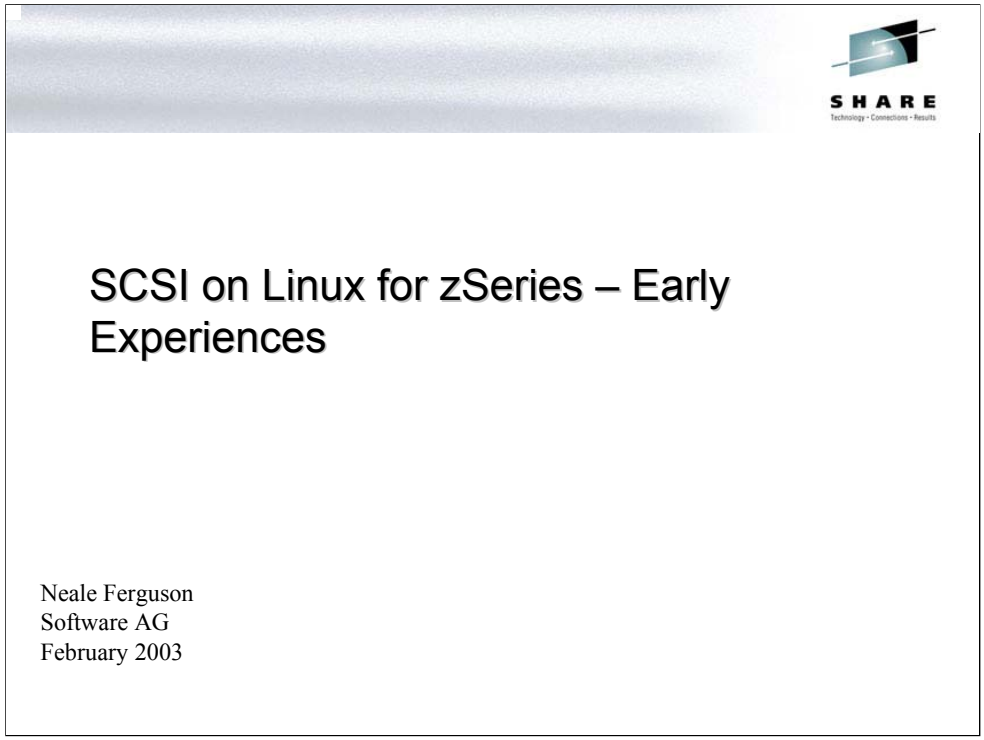


Since its debut in December 1999, Linux on S/390 and zSeries has evolved with great speed. Device support has been a major area of development. A major milestone for device support was reached with the addition of SCSI over fibre support.

This support was first demonstrated to the public with the almost bizarre exhibition at LinuxWorld 2002 in New York of a z900 burning CDs. Such an exhibition was good theatre as it generated great interest from the mainstream.

Software AG was approached to take part in an early field test of this support in July 2002. For this test, however, nothing as esoteric as CD burning would be done, just regular disk operations using our IBM Shark and EMC Symmetrix subsystems.

This report documents the experiences of installing, configuring and using this support. Before getting into the "guts" of the matter I'll get ahead of myself and state that it is a testament to the underling SCSI code and that developed by IBM that we were up and running on the first day.

Thanks

- Software AG began participation in the ESP in July, 2002
- IBM have authorized Software AG to present this material
- Thanks for their assistance during the ESP goes to:
  - Ingolf Salm – IBM Germany
  - Pam Hares – IBM UK
  - Christoph Arenz – IBM Germany (& US)

Acknowledgements

- Uli Kuhna – for SAN and Shark configuration
- Wolfgang Buettner – for z/VM & IOCDS configuration
- Dr. Gerhard Banzhaf (IBM) – material from his presentation “FCP Channel for z800 and z900”
- Material adapted from “Device Drivers and Installation Commands”.
- Material sourced from “Getting Started with zSeries Fibre Channel Protocol”.

Agenda

- Background
- Environment
- Configuration
- 31-bit Experiences
- 64-bit Experiences
- Further work

Background

- Allows Linux for S/390 (31-bit mode) and Linux for zSeries (64 bit mode) to access:
  - Distributed storage devices
    - With FCP interfaces (via switches)
    - With parallel SCSI interfaces (via additional bridges)
  - With Linux for zSeries (S/390) running
    - In a partition
    - Under z/VM (requires z/VM 4.3 RSU001)
- Provides access to distributed (open) storage and SAN world
- Based on existing FICON and FICON Express hardware

Background

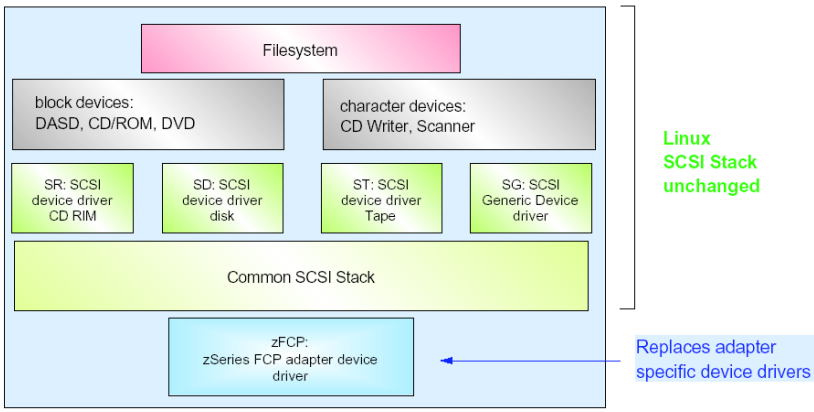
- New CHPID type: FCP
- Uses 2-port Fibre Channel cards FICON and FICON Express
  - Optical only
  - Short wave and long wave
  - 1 Gbit/s today: 2 Gbit/s has been announced
  - Currently 232KB buffer in card: 2MB proposed
- Different Firmware Load
  - Selected via definition of CHPID type in IOCP (HCD)
- QDIO protocol for communication between Processor/Memory and Channel
  - Based on scheme introduced with OSA Express
  - Continuously running channel programs
    - Reduces I/O path lengths
    - Reduces number of interrupts

Background

- The zfcp driver is a low-level or host-bus adapter driver supplementing the Linux SCSI I/O subsystem (SCSI stack).
- zfcp driver is open source.
- Linux for zSeries and S/390 can make use of all SCSI device types currently supported by Linux on other platforms including
  - SCSI disks,
  - Tapes,
  - CD-ROMs, and ,
  - DVDs.

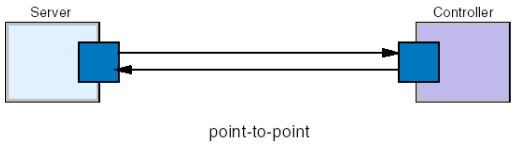
The `zfcp` driver is a low-level or host-bus adapter driver supplementing the Linux SCSI I/O subsystem (SCSI stack). Thus, Linux for zSeries and S/390 can make use of all SCSI device types currently supported by Linux on other platforms including SCSI disks, tapes, CD-ROMs, and DVDs. Both ESA (31 bit Linux) and ESAME (64 bit Linux) are supported.

Background

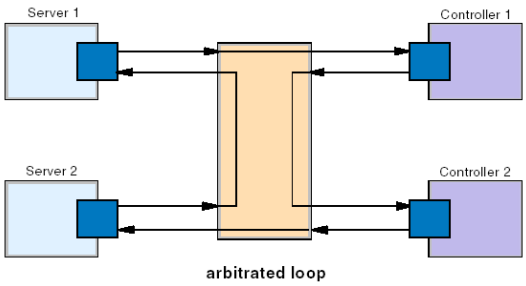

Filesystem
block devices:
DASD, CD/ROM, DVD
character devices:
CD Writer, Scanner
SR: SCSI
device driver
CD RIM
SD: SCSI
device driver
disk
ST: SCSI
device driver
Tape
SG: SCSI
Generic Device
driver
Common SCSI Stack
zFCP:
zSeries FCP adapter device
driver
Linux
SCSI Stack
unchanged
Replaces adapter
specific device drivers

SAN Topologies

- Point-to-point



- Arbitrated Loop



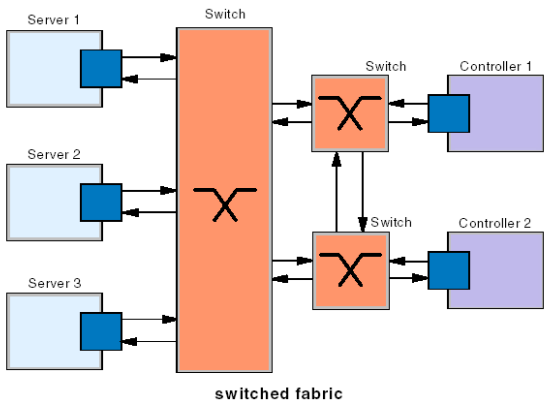
Point-to-point: This is the simplest topology to configure. A point-to-point configuration is a direct connection between two endpoints. Typically, it consists of a host, a device (such as a disk controller), and a dedicated fibre link.

Arbitrated Loop: This is a ring topology that shares the fibre channel bandwidth among multiple endpoints. The loop is implemented within a hub that interconnects the endpoints.

An arbitrated scheme is used to determine which endpoint gets control of the loop. The maximum number of ports is 127.

SAN Topologies

- Switched Fabric



This topology provides the most flexibility and makes the best use of the aggregated bandwidth by the use of switched connections between endpoints. One or more switches are interconnected to create a fabric, to which the endpoints are connected.

Environment

- Hardware

Component	Model	Microcode Level
IBM z900	2064-103	Level 8 (ML8)
McData SAN Switch	Connectrix ED-64M	01.04.002
Shark ESS	2105-F20	1.5.0.107
EMC	8730	
IBM Magstar	3590-E	

Environment

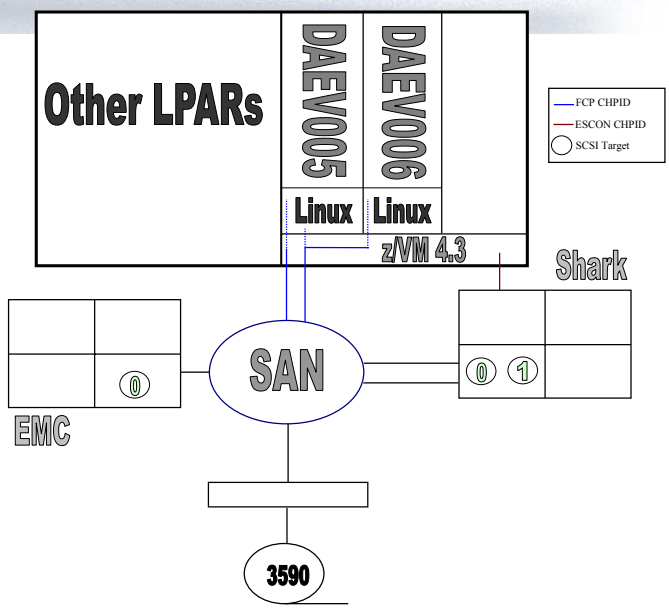
- Software

Component	Function	Level
z/VM	Hypervisor	4.3 RSU 002
Linux for S/390	31-bit kernel	2.4.7+
Linux for zSeries	64-bit kernel	2.4.17+
ESM Manager	SAN Manager	2.0

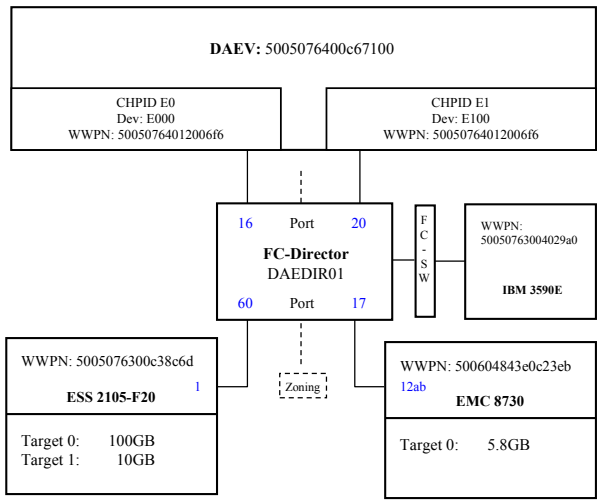
Environment

- Linux details
  - CONFIG_MSDOS_PARTITION=y
  - CONFIG_ZFCP=m
  - New utils-linux – `fdisk` required
  - Will (eventually) need new devs.rpm but not mandatory
  - Manuals geared towards devfs but we were using SLES7

Environment



SAN Configuration



Configuration

- z/VM User Directory

```
**** PROFILE LNXGST for LINUX Guests
PROFILE LNXGST
 MACHINE ESA
 OPTION QUICKDSP
 SHARE REL 1000
 IPL CMS PARM AUTOCR
 CONSOLE 009 3215 T OPERATOR
 SPOOL 00C 2540 READER *
 SPOOL 00D 2540 PUNCH  A
 SPOOL 00E 1403 A
 LINK MAINT 190 190 RR
 LINK MAINT 19D 19D RR
 LINK MAINT 19E 19E RR
*
USER DAEV005 ******** 256M 512M G
   INCLUDE LNXGST
   ACCOUNT LINUX R.WAITE
   MDISK 0191 3390 2651 0025 VVSYS1 MR
   MDISK 0192 3390 0001 2838 VVTL99 MR
   MDISK 0193 3390 2839 0500 VVTL99 MR
   SPECIAL E00A CTCA TCPIP
   SPECIAL E00B CTCA TCPIP
* HyperSockets with MTU size of 8k
   DEDICATE D10 D10
   DEDICATE D11 D11
   DEDICATE D12 D12
```

```
USER DAEV006 ******** 256M 512M G
   INCLUDE LNXGST
   ACCOUNT LINUX R.WAITE
   MDISK 0191 3390 2676 0025 VVSYS1 MR
   MDISK 0192 3390 3839 2838 VVTL99 MR
   MDISK 0193 3390 3339 0500 VVTL99 MR
   SPECIAL E00C CTCA TCPIP
   SPECIAL E00D CTCA TCPIP
* HyperSockets with MTU size of 8k
   DEDICATE D04 D04
   DEDICATE D05 D05
   DEDICATE D06 D06
```

Note: The SCSI devices ‘E000/E100’ were ATTACHed to each user as required:

```
FCP  E000 ATTACHED TO DAEV006  E000 CHPID E0
```

Configuration

- IOCDS

```
CHPID PATH=(E0),SHARED,
          PARTITION=((DAEV,DAEX,DALI),(DAEV,DAEX,DALI)),TYPE=FCP
CNTLUNIT CUNUMBR=E0FC,PATH=(E0),UNIT=FCP
IODEVICE ADDRESS=(E000,016),CUNUMBR=(E0FC),UNIT=FCP
CHPID PATH=(E1),SHARED,
PARTITION=((DAEV,DAEX,DALI),(DAEV,DAEX,DALI)),TYPE=FCP
CNTLUNIT CUNUMBR=E1FC,PATH=(E1),UNIT=FCP
IODEVICE ADDRESS=(E100,016),CUNUMBR=(E1FC),UNIT=FCP
```

Configuration

- CHPID/Device/SCSI Device
  - S390 device != SCSI device
  - S390 device is conduit to SCSI
  - May be 1,…,*n* SCSI devices at end of conduit
  - Each SCSI device may be partitioned to produce multiple targets

Configuration

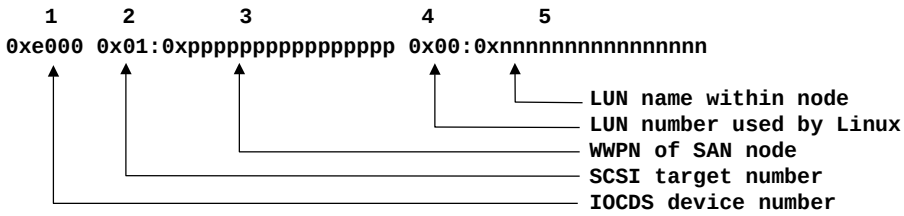
- A new addressing scheme was developed for Fibre Channel Protocol (FCP) usage, built around World Wide Names (WWN) that are eight bytes long.
- Part of the name represents an address type, part is a number that identifies the manufacturer, and part is a unique number assigned (by the manufacturer) for each port or node.
- A node is typically a box that contains information. Nodes have one or more ports (and only FC ports are relevant here). A given box may have several addresses; one for the node itself, and one for each FC port contained in the node. Abbreviations are:
  - WWNN is a World Wide Node Name
  - WWPN is a World Wide Port Name
  - WWN is any World Wide Name (WWNN or WWNP)

Configuration

- Linux provides extensive support for SCSI devices, based on the more advanced forms of traditional SCSI addressing.
- A SCSI device is expected to have a number (target address, typically a single digit) and this target may have multiple LUNs.
- The full SCSI addressing scheme can be used:
  - Device number (from your IOCDS) (coded as a “host number”)
  - A bus number on this adapter (always zero in current zSeries implementations)
  - A SCSI target number on this bus
  - A LUN number on this SCSI target
  - A partition or device within this LUN

Configuration

- FCP Mapping



Configuration

1. Device number
   - This device number must be defined in the IOCDS and be assigned to the FCP channel attached to the FCP switch.
     - You can use the same device number for all your FCP connections (by a given Linux image), or,
     - You can elect to use multiple device numbers (all assigned to the FCP channel, of course).

Configuration

2. Target number
   - The standard Linux SCSI support understands traditional SCSI addressing, with adapter numbers, bus numbers, target (“SCSI address”) numbers, and LUNs.
   - You assign this number.
   - Usable target addresses range from 1 to any positive 31-bit number.
   - Traditional SCSI target addresses ranged from 0-7 or 0-15. More recent SCSI architecture allows a much larger number.

Configuration

3. WWPN
   - The World Wide Port Name of the device containing the LUN.
   - The WWPN *as seen by the FC switche*s. WWPNs reported by devices may be slightly different.
   - You do not assign this number; it is built into the FCP devices and you must determine the proper number by querying your FCP elements.

Configuration

4. Linux LUN
   - The number *to be used by* Linux.
   - You assign this number.
   - Normal usage is to start with 0 and to increment by one for each LUN.
5. SAN Device LUN
   - The number used by the remote device.
   - You do not assign this number.
   - You must determine the number assigned by the node controller.

Configuration

- There are different parameters that can or must be supplied by the user to allow for proper zfcp operation:
  - Address mappings between Linux SCSI and FCP schemes (optional for each SCSI target)
  - Logging level to determine the verbosity of the zfcp device driver (optional, default value is used if not supplied)

Configuration

- The zfcp driver provides different means of configuration:
  - Kernel parameters
  - Module parameters (such as for use in `modules.conf`)
  - Various proc file system entries in `/proc/scsi/zfcp`

There are different parameters that can or must be supplied by the user to allow for proper `zfcp` operation:

•Address mappings between Linux SCSI and FCP schemes (optional for each SCSI target)

•Logging level to determine the verbosity of the `zfcp` device driver (optional, default value is used if not supplied)

For these purposes, the `zfcp` driver provides different means of configuration:

•Kernel parameters

•Module parameters (such as for use in `modules.conf`)

•Various proc file system entries in `/proc/scsi/zfcp`

Configuration

- Module Parameters and proc File System entries

Function	Module Parameter	Kernel Parameter	proc-fs entry
Set logging level	loglevel zfcp_	loglevel	/proc/scsi/zfcp/mod_parm
Get logging level (and other global module information)	N/A	N/A	/proc/scsi/zfcp/mod_parm
Add address mapping(s)	map	zfcp_map	/proc/scsi/zfcp/add_map
Get all existing address mappings	N/A	N/A	/proc/scsi/zfcp/map

Configuration

- Additional device nodes (not required in SLES8):

```
mknod /dev/sda  b 8 0
mknod /dev/sda1 b 8 1
mknod /dev/sda2 b 8 2
mknod /dev/sda3 b 8 3
mknod /dev/sda4 b 8 4
mknod /dev/sda5 b 8 5
mknod /dev/sda6 b 8 6
mknod /dev/sda7 b 8 7
mknod /dev/sda8 b 8 8
mknod /dev/sdb  b 8 16
mknod /dev/sdb1 b 8 17
mknod /dev/sdb2 b 8 18
mknod /dev/sdb3 b 8 19
mknod /dev/sdb4 b 8 20
```

The SLES7 Linux system we were using did not use the `devfs` file system nor did it have definitions in the `/dev` directory. To rectify this we used the commands found above. SLES8 has the nodes defined and does not require these commands to be issued.

Configuration

- ESS Devices

Target	Logical Unit	Size	Device Name
0	0	25GB	/dev/sda1
	1	50GB	/dev/sda2
	2	25GB	/dev/sda3
	3	25GB	/dev/sda4
1	0	5GB	/dev/sdb1
	1	5GB	/dev/sdb2

Configuration

- Driver load script
  - **ln –s /etc/init.d/scsi_load /etc/init.d/rc3.d/S12scsild**

```
#!/bin/sh
rmmod zfcp
modprobe qdio
modprobe scsi_mod
insmod zfcp loglevel=0 map="\
0xe000 0x01:0x5005076300c38c6d 0x00:0x5200000000000000;\
0xe000 0x01:0x5005076300c38c6d 0x01:0x5201000000000000"
modprobe sd_mod
modprobe st
```

Based on the example provided in the Redpaper “Getting Started with zSeries Fibre Channel Protocol” we created the above script to load the appropriate device drivers and provide the correct parameters for the ESS devices.

Configuration

- During scsi_load:

```
scsi0 : zfcp
  Vendor: IBM       Model: 2105F20           Rev: .107
  Type:   Direct-Access                      ANSI SCSI revision: 03
Attached scsi disk sda at scsi0, channel 0, id 1, lun 0
zfcp: FSF: zfcp_fsf_send_fcp_command_task_handler: status for SCSI Command:
00000000 0000
zfcp: FSF: zfcp_fsf_send_fcp_command_task_handler: SCSI status code 0x2
00000000 00000000 00000202 00000000 00000020 00000000
70000600 00000018 00000000 29000000 00000000 00000000 00000000 00000000

SCSI device sda: 195312512 512-byte hdwr sectors (100000 MB)
 dasdbm:(nonl)/        : dasdbm1
```

Configuration

- Following scsi_load:

```
Module                  Size  Used by
st                     27696   0  (unused)
sd_mod                 12048   0  (unused)
zfcp                  345312   0
scsi_mod               61344   3  [st sd_mod zfcp]
nfsd                   69648   4  (autoclean)
qeth                  153072   1  (autoclean)
qdio                   33968   2  (autoclean) [zfcp qeth]
ipv6                  247472  -1  (autoclean) [qeth]
8021q                  12928   0  (autoclean) [qeth]
ctc                    49840   1  (autoclean)
fsm                     1920   0  (autoclean) [ctc]
```

Configuration

- Partitioning using `fdisk`:

```
Disk /dev/sda: 255 heads, 63 sectors, 12157 cylinders
Units = cylinders of 16065 * 512 bytes

   Device Boot    Start       End    Blocks   Id  System
/dev/sda1             1      3188  25607578+  83  Linux
/dev/sda2          3189      6376  25607610   83  Linux
/dev/sda3          6377     12157  46435882+  83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
```

Our SLES7 system did not have the `fdisk` utility. However, thanks to our excellent working relationship with SuSE (and Bernd Kaindl in particular), we were able to acquire a newly built `utils-linux` package containing the necessary command. We used this command to divide the area of the SCSI device into several partitions as shown above.

Configuration

- Mounting devices
  - **ln –s /etc/init.d/scsi_mount /etc/init.d/rc3.d/S12scsimn**

```
#!/bin/sh
mount /dev/sda1 /FS/scsi01
mount /dev/sda2 /FS/scsi02
mount /dev/sda3 /FS/scsi03
mount /dev/sdb1 /FS/scsi04
mount /dev/sdb2 /FS/scsi05
```

Configuration

- Unmounting devices
  - **ln –s scsi_unmount rc3.d/K13scsium**

```
#!/bin/sh
umount /FS/scsi01
umount /FS/scsi02
umount /FS/scsi03
umount /FS/scsi04
umount /FS/scsi05
```

Configuration

- Contents of `/proc/partitions`:

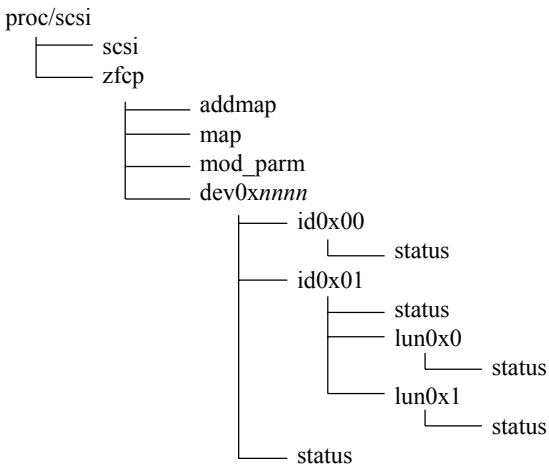
```
major minor  #blocks  name

   8     0   97656256 dasdbm
   8     1   25607578 dasdbm1
   8     2   25607610 dasdbm2
   8     3   46435882 dasdbm3
   8    16    9765632 dasdbn
   8    17    5121008 dasdbn1
   8    18    4643840 dasdbn2
  94     0     360000 dasda
  94     1     359988 dasda1
  94     4    2043360 dasdb
  94     5    2043348 dasdb1
```

➢ Suspect device naming due to IBM Partitioning being enabled

Configuration

- proc Filesystem has lots of SCSI information:



Linux provides a lot of data within the proc file system that allows you to inspect, monitor, and change the configuration and operation of the SCSI subsystem. Graphically, the sub-tree available looks like the above.

Configuration

- Contents of `/proc/scsi/scsi`:

```
Attached devices:
Host: scsi0 Channel: 00 Id: 01 Lun: 00
  Vendor: IBM      Model: 2105F20          Rev: .107
  Type:   Direct-Access                    ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 01 Lun: 01
  Vendor: IBM      Model: 2105F20          Rev: .107
  Type:   Direct-Access                    ANSI SCSI revision: 03

Host: scsi0 Channel: 00 Id: 01 Lun: 00
  Vendor: EMC      Model: SYMMETRIX        Rev: 5567
  Type:   Direct-Access                    ANSI SCSI revision: 02
```

Configuration

- Contents of `/proc/scsi/zfcp/dev0xe000/status`:

```
FCP adapter

FCP driver $Revision: 3.60.4.1 $ (or for cryptography's sake 0x0003003c)

device number:            0xe000     registered on irq:             0x0013
WWNN:          0x5005076400c67100
WWPN:          0x5005076401206f6      S_ID:                        0x611413
HW version:               0x0002     LIC version:                0x0000001a
FC link speed:            1 Gb/s     FC service class:                    3
FC topology:   fabric
SCSI host number:     0x00000000

Attached ports:                2     QTCB size (bytes):               1696
Max SCSI ID of ports: 0x00000001     Max SCSI LUN of ports:     0x00000001
FSF req seq. no:      0x003d03c8     FSF reqs active:                   16
Scatter-gather table-size:    57     Max no of queued commands:       4096
Uses clustering:               1     Uses New Error-Handling Code:       1
ERP counter:          0x00000000     Adapter Status:            0x5400006f

Adapter Structure information:
Common Magic:         0xfcfcfcfc     Specific Magic:            0xaaaaaaaa
Adapter struct at:    0x0de25000     List head at:              0x0de25008
Next list head:       0x109d82b0     Previous list head:        0x109d82b0

Scsi_Host struct at:  0x0d868400
Port list head at:    0x0de250a0
Next list head:       0x0d869808     Previous list head:        0x0d868808
List lock:            0x00000000     List lock owner PC:        0x00000000
```

Configuration

- `/proc/scsi/zfcp/dev0xe000/status` (cont.):

```
O-FCP req list head:  0x0de250d0
Next list head:       0x0d89c608     Previous list head:       0x0d87a608
List lock:            0x00000000     List lock owner PC:       0x00000000

Request queue at:     0x0de250f8
Free index:                002     Free count:                    128
List lock:            0x00000000     List lock owner PC:       0x00000000
current TOD:          1326383201267704O160
time lock held:       35007809781

Response queue at:    0x0de25550
Free index:                072     Free count:                    000
List lock:            0x00000000     List lock owner PC:       0x00000000

DEVICE INFORMATION (devinfo):
Status: "OK"
Control Unit Type:        0x1731     Control Unit Model:            0x03
Device Type:              0x1732     Device Model:                  0x03
CIWs:       0x40720080 0x41830004 0x42820040 0x431b1000
            0x441f0000 0x00000000 0x00000000 0x00000000

DEVICE INFORMATION (devstat):
Interrupt Parameter:  0x00000002     Last path used mask:           0x00
Channel Status:             0x80     Device Status:                 0x00
Flag:                 0x00000204     CCW address (from irb):   0x0d863a40
Response count:       0x00000000     Sense Count:              0x00000000
IRB:        0x00c0c0c9 0x0d863a40 0x00800000 0x00000000
Sense Data: 0x00000000 0x00000000 0x00000000 0x00000000
```

Configuration

- Contents of …`/dev0xe000/id0x01/status`:

```
Port Information:

WWNN:          0x0000000000000000     WWPN:              0x5005076300c38c6d
SCSI-ID:               0x00000001     Max SCSI lun:              0x00000001
D-ID:                  0x00614013
Handle:                0x00000025

Attached units:                2
ERP counter:           0x00000000
Port Status:           0x54000003

Port Structure information:
Common Magic:          0xfcfcfcfc     Specific Magic:            0xbbbbbbbb
Port struct at:        0x0d869800     List head at:              0x0d869808
Next list head:        0x0d868808     Previous list head:        0x0de250a0

Unit list head at:     0x0d869820
Next list head:        0x0d869508     Previous list head:        0x0d869208
List lock:             0x00000000     List lock owner PC:        0x00000000

Parent adapter at:     0x0de25000
```

Configuration

- Contents of …`/id0x01/lun0x0/status`:

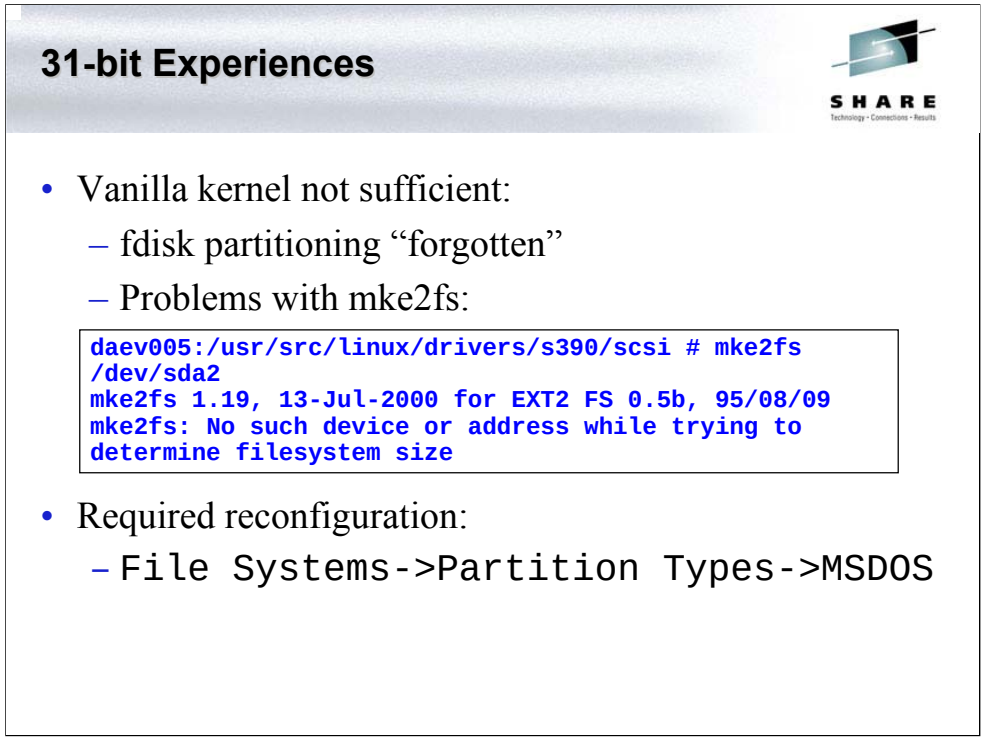
```
Unit Information:

SCSI lun:              0x00000000     FCP lun:           0x5200000000000000
Handle:                0x0000001d

ERP counter:           0x00000000
Unit Status:           0x54000000

Unit Structure information:
Common Magic:          0xfcfcfcfc     Specific Magic:            0xcccccccc
Unit struct at:        0x0d869500     List head at:              0x0d869508
Next list head:        0x0d869208     Previous list head:        0x0d869820

Parent port at:        0x0d869800     SCSI dev struct at:        0x0e9f9e00
```



With the apparently successful partitioning of the device we attempted to use the `mke2fs` command to create an ext2fs on the various partitions. However, it was at this point we began running into trouble. The first partition appeared to work but the file system appeared to occupy the entire device not just the first partition (24414063 is the device size not the partition size):.

An attempt to make a file system on the second partition did not work at all:

```
daev005:/usr/src/linux/drivers/s390/scsi # mke2fs
/dev/sda2
mke2fs 1.19, 13-Jul-2000 for EXT2 FS 0.5b, 95/08/09
mke2fs: No such device or address while trying to
determine filesystem size
```

We then specified `loglevel=3` on `insmod zfcp` command to log debugging information. The log shows no activity when references to `sda2`/`sda3` are made but lots of activity for `sda1`.

In order to determine if the error was peculiar to the 100GB device, we arranged with Software AG IT services to define another small SCSI device. We again used `fdisk` to split this 10GB device into two 5GB partitions. However, the same errors were encountered. Indeed, we experienced errors during load of the device drivers.

Following advice from Stefan Bader of IBM, we updated the Linux kernel configuration to enable support for: `File Systems->Partition Types->MSDOS`. After rebooting to pick up this new kernel, all our partitions became accessible.

31-bit Experiences

- EMC Configuration

```
zfcp: FSF: zfcp_fsf_send_fcp_command_task_handler: status for SCSI Command:
2a000000 00000000 0200
zfcp: FSF: zfcp_fsf_send_fcp_command_task_handler: SCSI status code 0x2
00000000 00000000 00000a02 00000400 00000012 00000000
70000700 0000000a 00000000 27000000 0000
SCSI disk error : host 0 channel 0 id 1 lun 0 return code = 8000002
Current sd08:00: sns = 70  7
ASC=27 ASCQ= 0
Raw sense data:0x70 0x00 0x07 0x00 0x00 0x00 0x00 0x0a 0x00 0x00 0x00 0x00 0x27 0x00
0x00 0x00 0x00 0x00
 I/O error: dev 08:00, sector 0
SCSI device sda: 15360 512-byte hdwr sectors (8 MB)
 dasdbm:(nonl)/        : dasdbm1
```

Initially my insmod parameter was as follows:

`0xe100 0x01:0x500604843e0c23eb 0x00:0x0000000000000001`

After (apparently) successfully accessing the disk I attempted to partition it using `fdisk`. The utility reported success, but `syslog` messages were generated that indicated this was not the case.

31-bit Experiences

- Product testing:
  - Adabas databases
    - 100MB data files
  - AQA tables (SQL access to Adabas)
  - All worked as expected

I created a development environment for our Adabas and Adabas SQL Access (AQA) products, and proceeded to build them. The build involved significant amounts of compilation and linking that would allow me to exercise the disks. No problems were encountered during the build processes.

I created 2 Adabas databases on the Linux guest DAEV005 using the SCSI disks. One was for the normal installation verification databases and one for use by AQA and has sizes of:

- ASSO - 100MB
- DATA - 100MB
- WORK - 60MB

I installed AQA on DAEV005, built our internal test suite, and ran the various test scripts. In addition I rebuilt the AQA package using the SCSI area as the build area. No problems experienced.

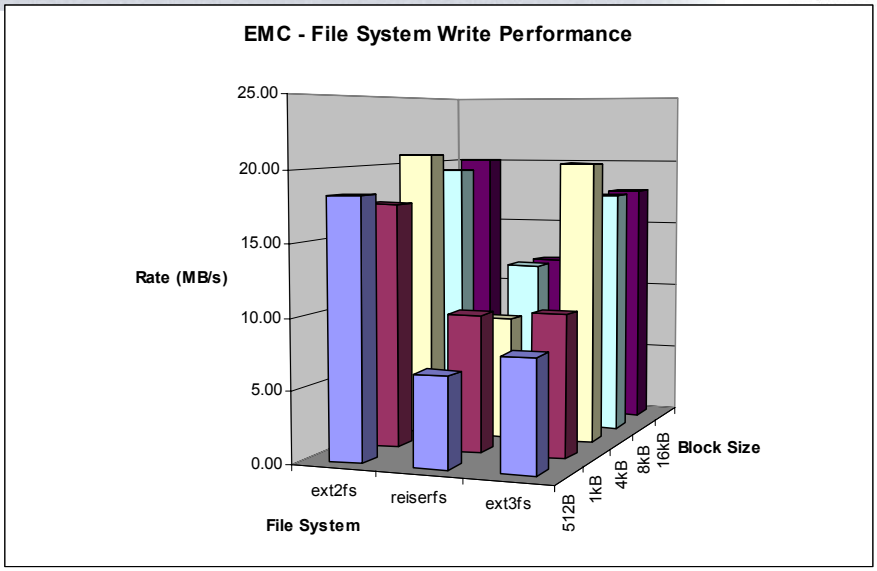
31-bit Experiences

- Simplistic benchmarking
  - Used dd to exercise disk writes
  - Changed the underlying file system
    - ext2
    - ext3
    - reiser
  - Uncontrolled environment & unscientific methodology
    - CPUs shared with other LPARs
    - LPAR has low weighting
  - Results are indicative only

I performed simple experiments on the disks using the script found below to create a file half the size of the disk area. On DAEV005 I used the EMC devices, on DAEV006 I used the ESS devices. The testing was done in an uncontrolled environment: running on a lowly weighted LPAR sharing 1 of 3 processors with several other LPAR-based z/OS and VSE systems.

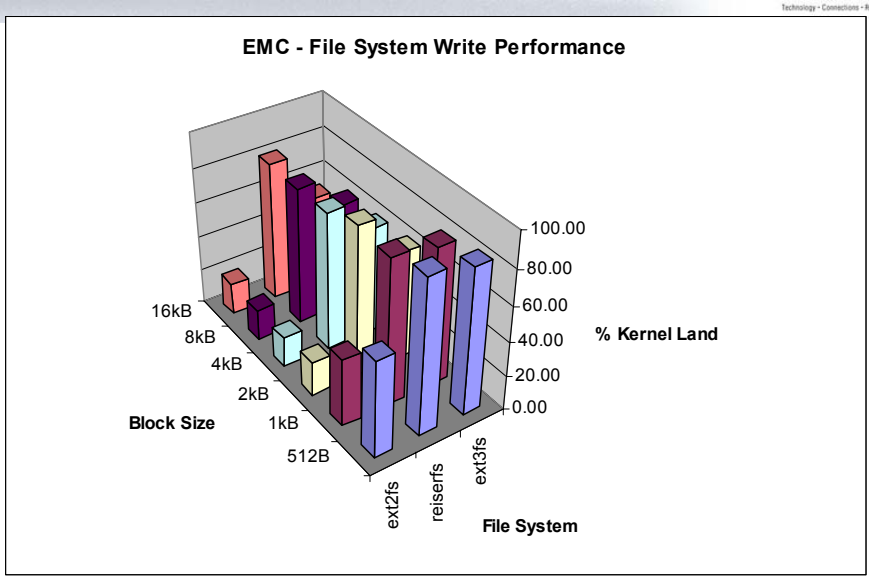
```
#!/usr/bin/rexx
/* */
trace o
fs.1 = 'mke2fs'
fs.2 = 'mkreiserfs'
fs.3 = 'mke2fs -j'
fs.0 = 3
id.1 = 'ext2fs'
id.2 = 'reiserfs'
id.3 = 'ext3fs'
op.1 = ''
op.2 = '<tstfs.in'
op.3 = ''
bs.1 = 512; bs.2 = 1024; bs.3 = 4096; bs.4 = 8192; bs.5 = 16384; bs.0  = 5
Target = '/FS/scsi04' /* '/FS/tempfs' on DAEV005 */
FCPdev = '/dev/sdb1'  /* '/dev/sda2' on DAEV005 */
'echo y >tstfs.in'
Capacity = 'df'(Target)
parse var Capacity 'Mounted on' . Capacity .
Capacity = Capacity % 2
do I_fs = 1 to fs.0
        'umount' Target
        fs.I_fs FCPdev '>/dev/null 2>&1' op.I_fs
        'mount' FCPdev Target
        say COPIES('-',60)
        do I_bs = 1 to bs.0
                say '=>' bs.I_bs
                do I_Task = 1 to 4
                        count = (Capacity * (1024 / bs.I_bs)) % 1
                        say id.I_fs '('count')'
                        'time dd if=/dev/zero of='Target'/dummy.file',
                                'bs='bs.I_bs 'count='count
                        'sleep 2s'
                        'time rm' Target'/dummy.file'
                        'sleep 2s'
                end
        end
end
exit
```

31-bit Experiences



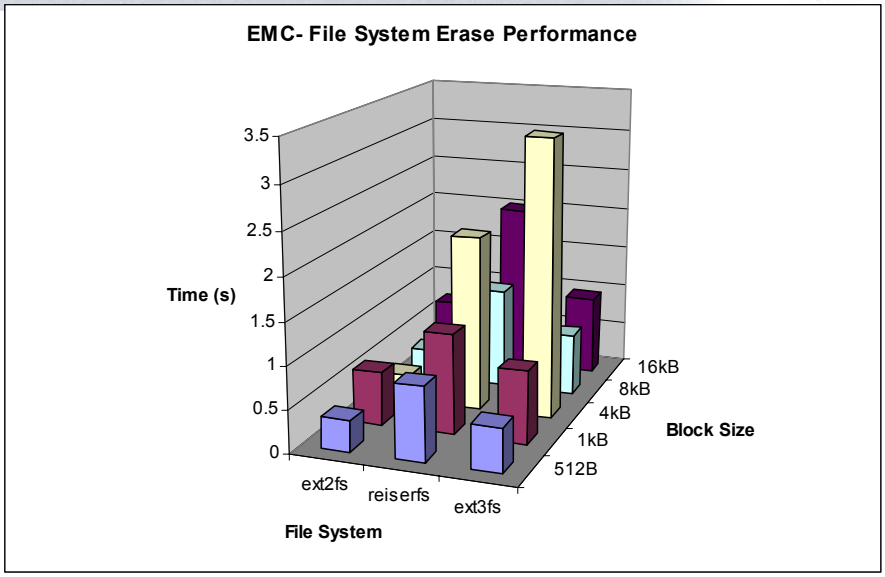
**`dd if=/dev/zero of=/FS/tempfs/dummy.file bs=`*`n`*` count=`*`nnnnnnn`*

31-bit Experiences



`dd if=/dev/zero of=/FS/tempfs/dummy.file bs=`*`n`* `count=`*`nnnnnnn`*

31-bit Experiences




```
rm /FS/tempfs/dummy.file
```

64-bit Experiences

- Product Testing
  - Adabas & AQA as per 31-bit
  - Included Natural
  - All worked as expected
- Similar transfer rate found when doing simplistic "dd" test

The same process was carried out for our 64-bit Linux system. The kernel was rebuilt, device nodes created, and device driver loaded. The startup and shutdown scripts created for the 31-bit system were reused in the 64-bit system. The devices were successfully sensed, drivers loaded and file systems mounted.

I created 2 Adabas databases on the Linux guest DAEV006 using the SCSI disks. One was for the normal vehicles etc. db and one for use by AQA and has sizes of:

- ASSO - 100MB
- DATA - 100MB
- WORK - 60MB

I installed AQA on DAEV006, built the internal test suite, and ran the test scripts. In addition I rebuilt the AQA package using the SCSI area as the build area. No problems experienced.

Activities included:

§ Generation of large databases;

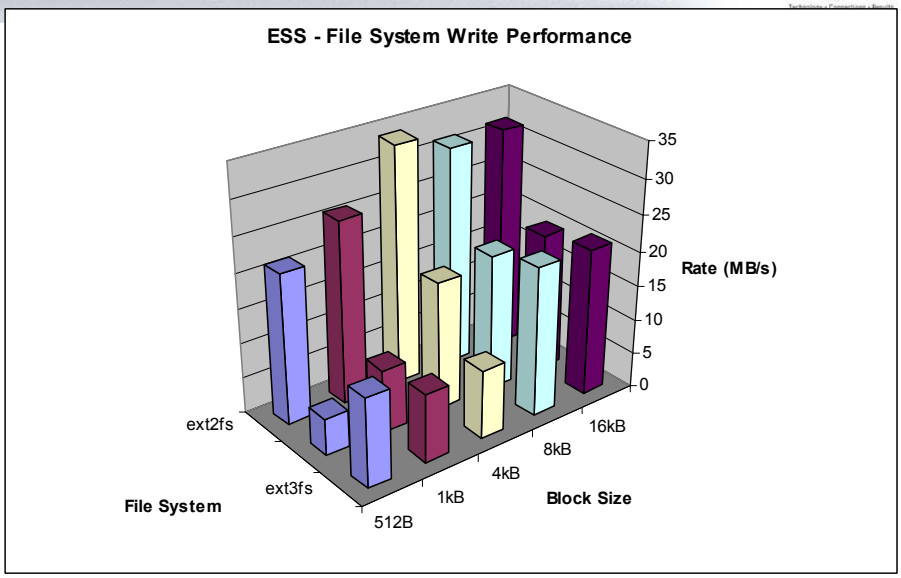
§ Large tables and indexes were created, filled, interrogated and dropped;

§ Full logging was enabled to cause the production of large trace files (~2GB).

§ Natural was used to extract data from the Adabas database.

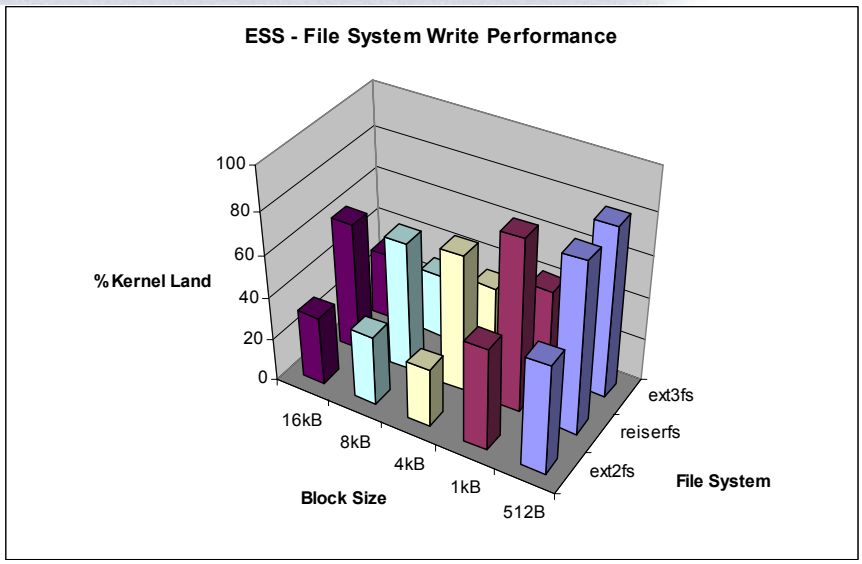
No problems were encountered.

64-bit Experiences



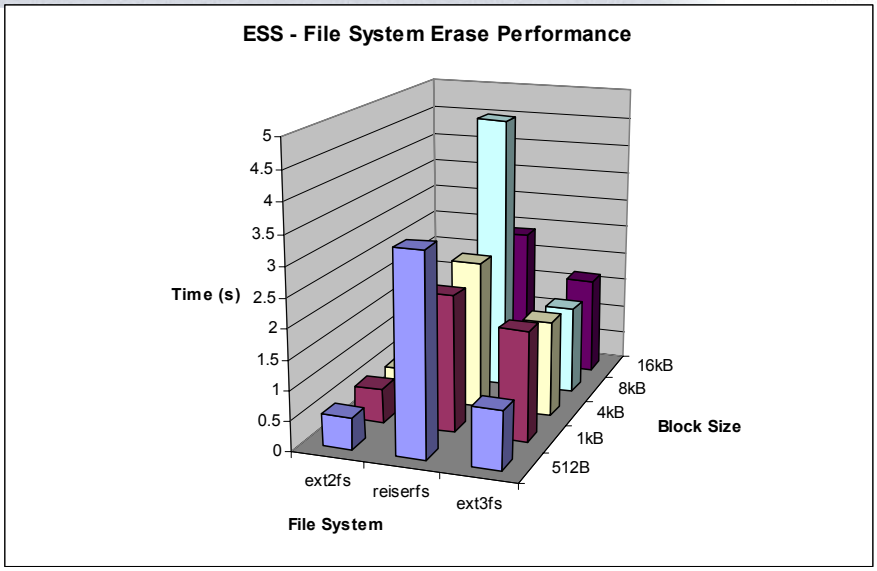

```
dd if=/dev/zero of=/FS/scsi03/dummy.file bs=n count=nnnnnnn
```

64-bit Experiences



`dd if=/dev/zero of=/FS/scsi03/dummy.file bs=`*`n`* `count=`*`nnnnnnn`*

64-bit Experiences



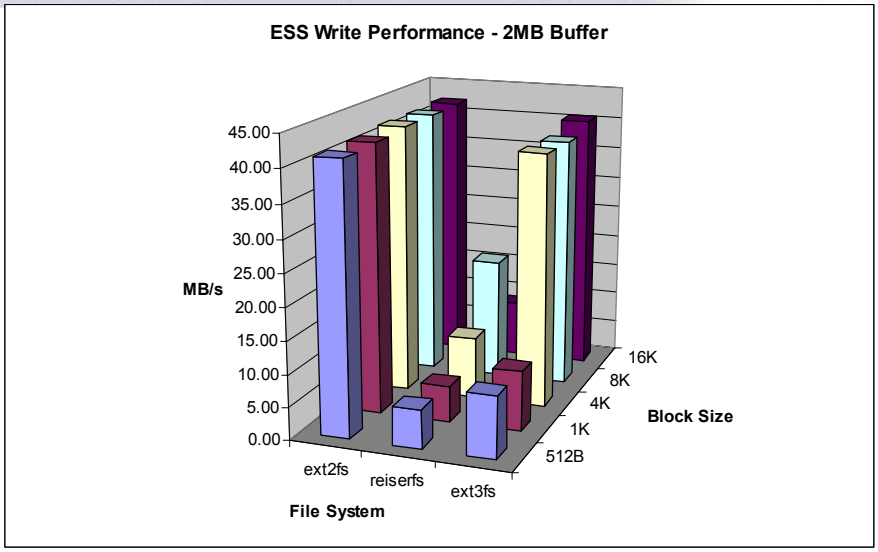

```
rm /FS/scsi03/dummy.file
```

Microcode Upgrade

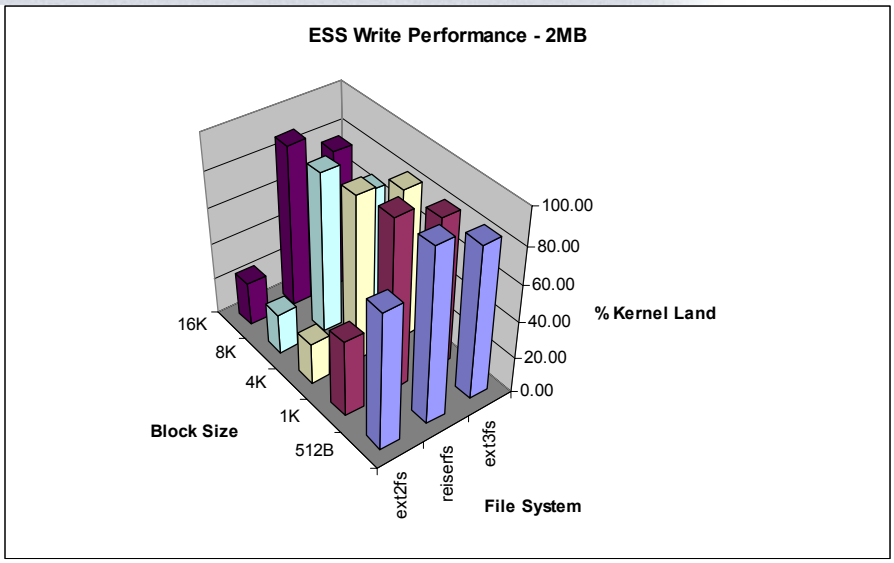
- Applied September 2002
- Increases buffer from 234K to 2MB
- Necessary for streaming tape
- Should improve big transfers in general

The previous tests were repeated after a microcode fix was applied to the 2064 to support 2MB buffers within the FCP card. Note the greatly improved data rate for Ext2 and Ext3.

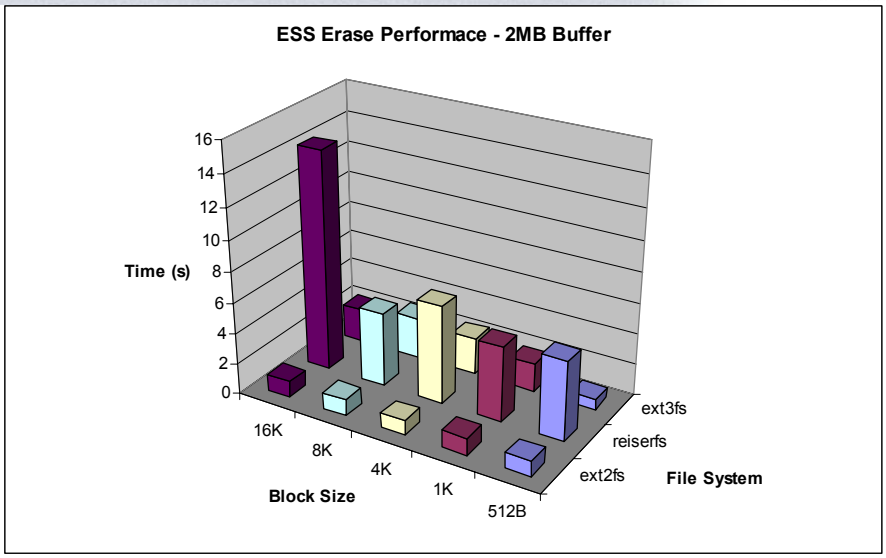
ESS – Write Performance 2MB Buffer



ESS – Write Performance 2MB Buffer



ESS - Erase Performance 2MB Buffer



Tape – 3590E

- Loaned device
- Two new RPMs:
  - IBMTapeutil

    A tape utility program that exercises or tests the functions of the Linux device driver, IBMtape. It performs tape and medium changer operations.
  - IBMTapeconfig

    A script which creates and destroys IBMtape device special files according to the information logged in /proc/scsi/IBMtape and /proc/scsi/IBMchanger files.

IBM Tape Utility


```
------------------------- General Commands: -----------------------------
    1. Open a Device                  7. Request Sense
    2. Close a Device                 8. Log Sense Page
    3. Inquiry                        9. Mode Sense Page
    4. Test Unit Ready               10. Switch Tape/Changer Device
    5. Reserve Device                11. Create Special Files
    6. Release Device                12. Query Driver Version
    Q. Quit IBMtapeutil
-------------------------- Tape Commands: -------------------------------
   20. Rewind                        33. Set Block Size
   21. Forward Space Filemarks       34. Retension Tape
   22. Backward Space Filemarks      35. Query/Set Tape Position
   23. Forward Space Records         36. Query Tape Status
   24. Backward Space Records        37. Load Tape
   25. FSFM                          38. Unload Tape
   26. BSFM                          39. Lock Tape Drive Door
   27. Space to End of Data          40. Unlock Tape Drive Door
   28. Read and Write Tests          41. Take Tape Offline
   29. Write Filemarks               42. Enable/Disable Compression
   30. Read or Write Files           43. Flush Driver's Buffer
   31. Erase                         44. Self Test
   32. Reset Drive                   45. Display Message
------------------------- IBMtape Commands: -----------------------------
   46. Query Sense                   52. Locate Tape Position
   47. Query Inquiry                 53. Read Tape Position
   48. Query/Set Tape Parameters     54. Query Mtdevice Number
   49. Query/Set Tape Position       55. Synchronize Buffers
   50. Query/Set MT/ST Mode          56. List Tape Filemarks
   51. Report Density Support
------------------------- Service Aid Commands: -------------------------
   70. Dump Device                   72. Load Ucode
   71. Force Dump                    73. Reset Drive
-------------------------------------------------------------------------
   99. Back To Main Menu
```

IBM Tape Utility


```
Inquiry Data:
Peripheral Qualifer------------------0x00
Peripheral Device Type---------------0x01
Removal Medium Bit-------------------1
Device Type Modifier-----------------0x00
ISO version--------------------------0x00
ECMA version-------------------------0x00
ANSI version-------------------------0x03
Asynchronous Event Notification Bit---0
Terminate I/O Process Message Bit-----0
Response Data Format-----------------0x02
Additional Length--------------------0x33
Medium Changer Mode------------------0x00
Relative Addressing Bit--------------0
32 Bit Wide Data Transfers Bit--------0
16 Bit Wide Data Transfers Bit--------0
Synchronous Data Transfers Bit--------0
Linked Commands Bit------------------0
Command Queueing Bit-----------------0
Soft Reset Bit-----------------------0
Vendor ID----------------------------IBM
Product ID---------------------------03590E11
Product Revision Level---------------E350
```

FCP Attached SCSI Tape