TOSHIBA
Leading Innovation

Evaluation of Data Reliability on Linux File Systems

Yoshitake Kobayashi
Advanced Software Technology Group
Corporate Software Engineering Center
TOSHIBA CORPORATION

Apr. 12, 2010
CELF Embedded Linux Conference

Outline

- Motivation
- Evaluation
- Conclusion

Motivation

We want

- NO data corruption
- data consistency
- GOOD performance

We do NOT want

- frequent data corruption
- data inconsistency
- BAD performance

Ext3 *Ext4* *XFS* *JFS* *ReiserFS* *Btrfs* *Nilfs2* *……*

enough evaluation?

NO!

Reliable file system requirement

For data consistency

- journaling
- SYNC vs. ASYNC
  - SYNC is better

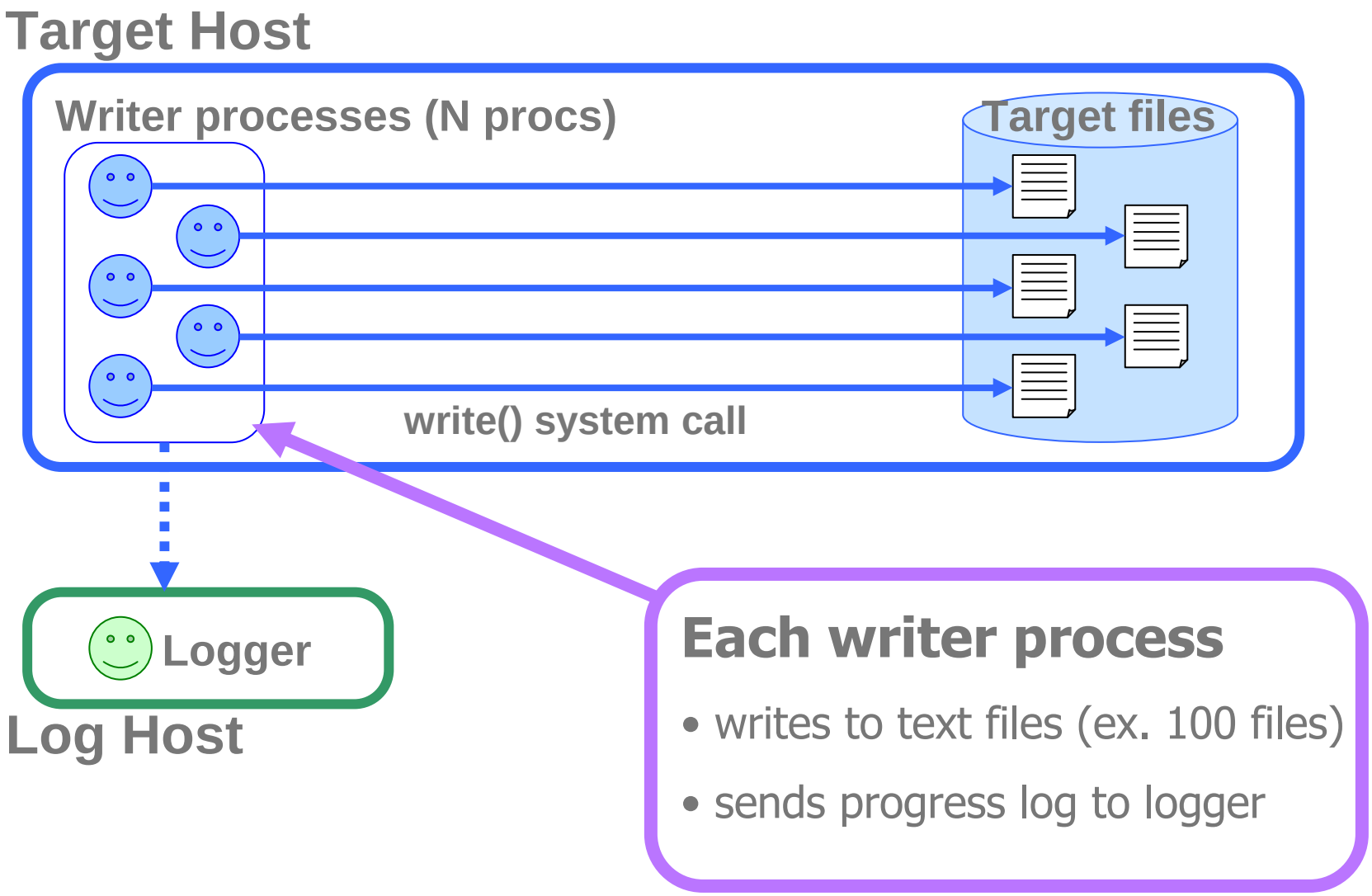
Focus

- available file systems on Linux
- data writing
- data consistency

Metrics

- logged progress = file size
- estimated file contents = actual file contents

Evaluation: Overview



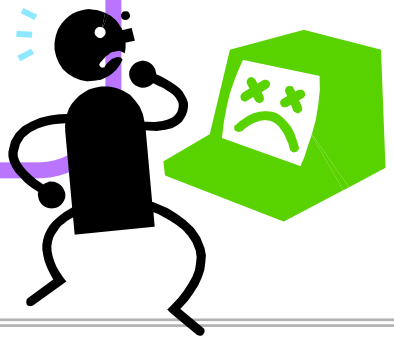
Target Host

Writer process

- writes to text files
- sends progress log to logger

How to crash

- modified reboot system call
  - forced to reboot
  - 10 seconds to reboot



Target Host

Writer process

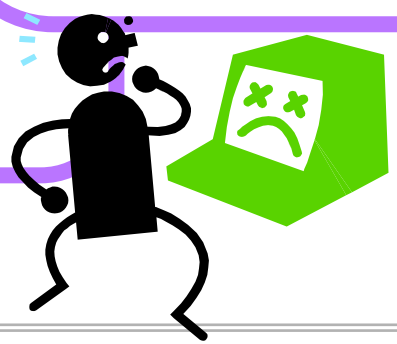
- writes to text files
- sends progress log to lo

How to crash

- modified reboot system
  - forced to reboot
  - 10 seconds to reboot

Test cases

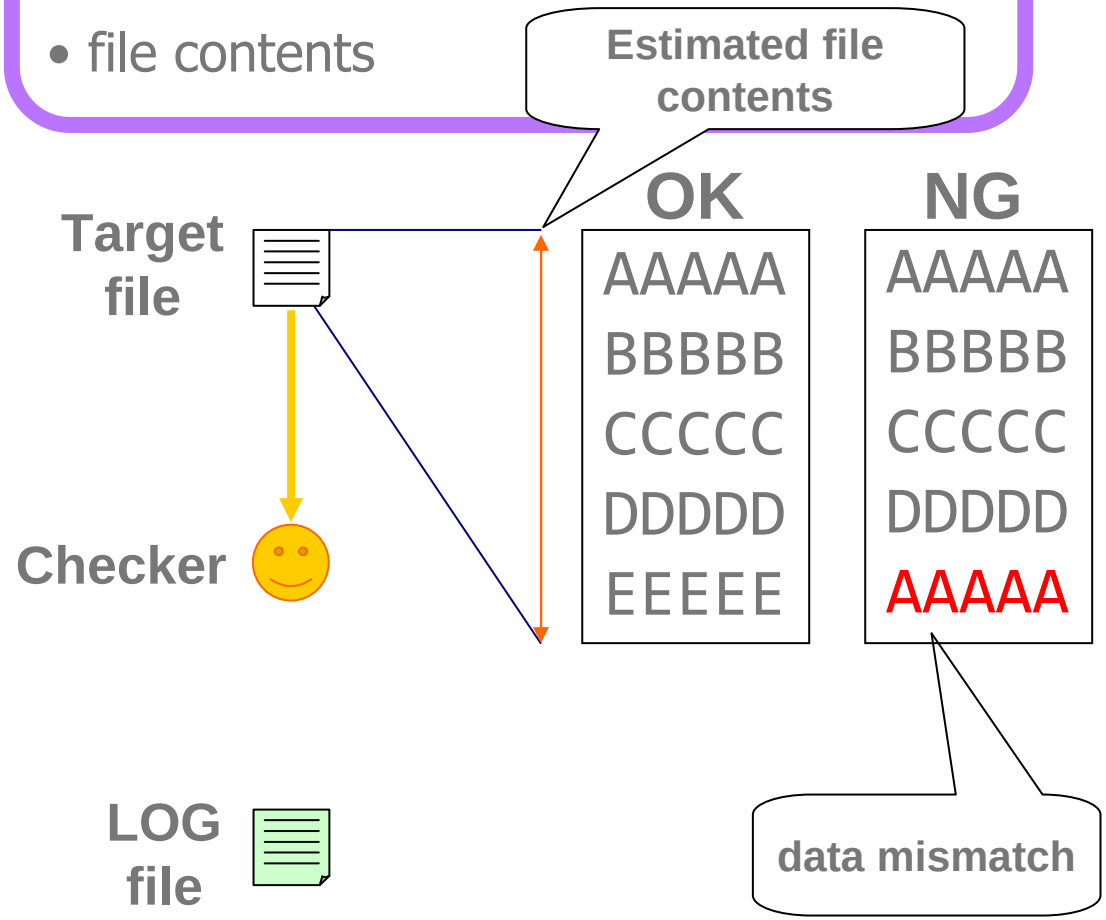
1. create: open with O_CREATE
2. append: open with O_APPEND
3. overwrite: open with O_RDWR
4. write->close: open with O_APPEND and call close() on each write()



Verification

Verify the following metrics

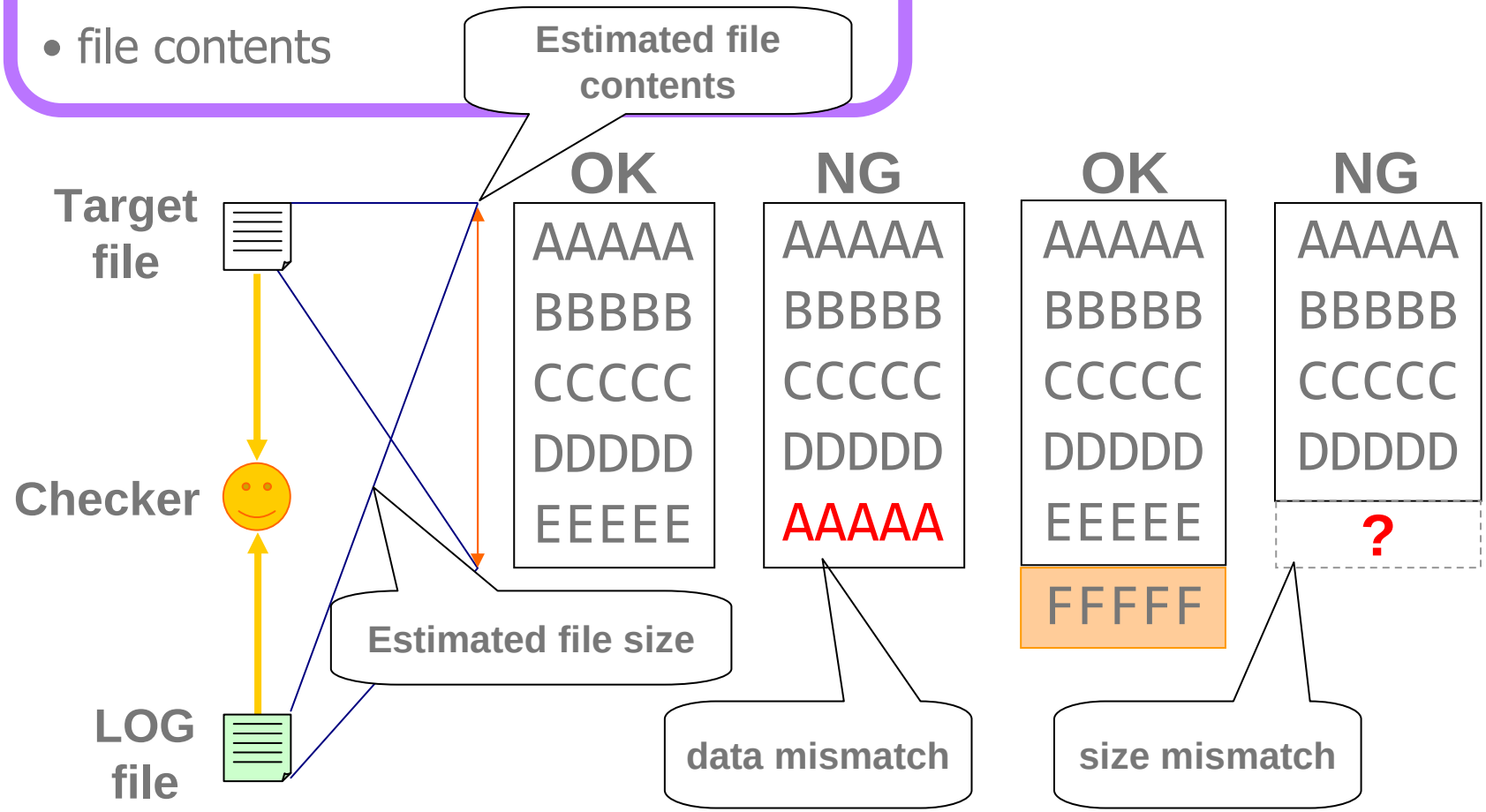
- file size
- file contents



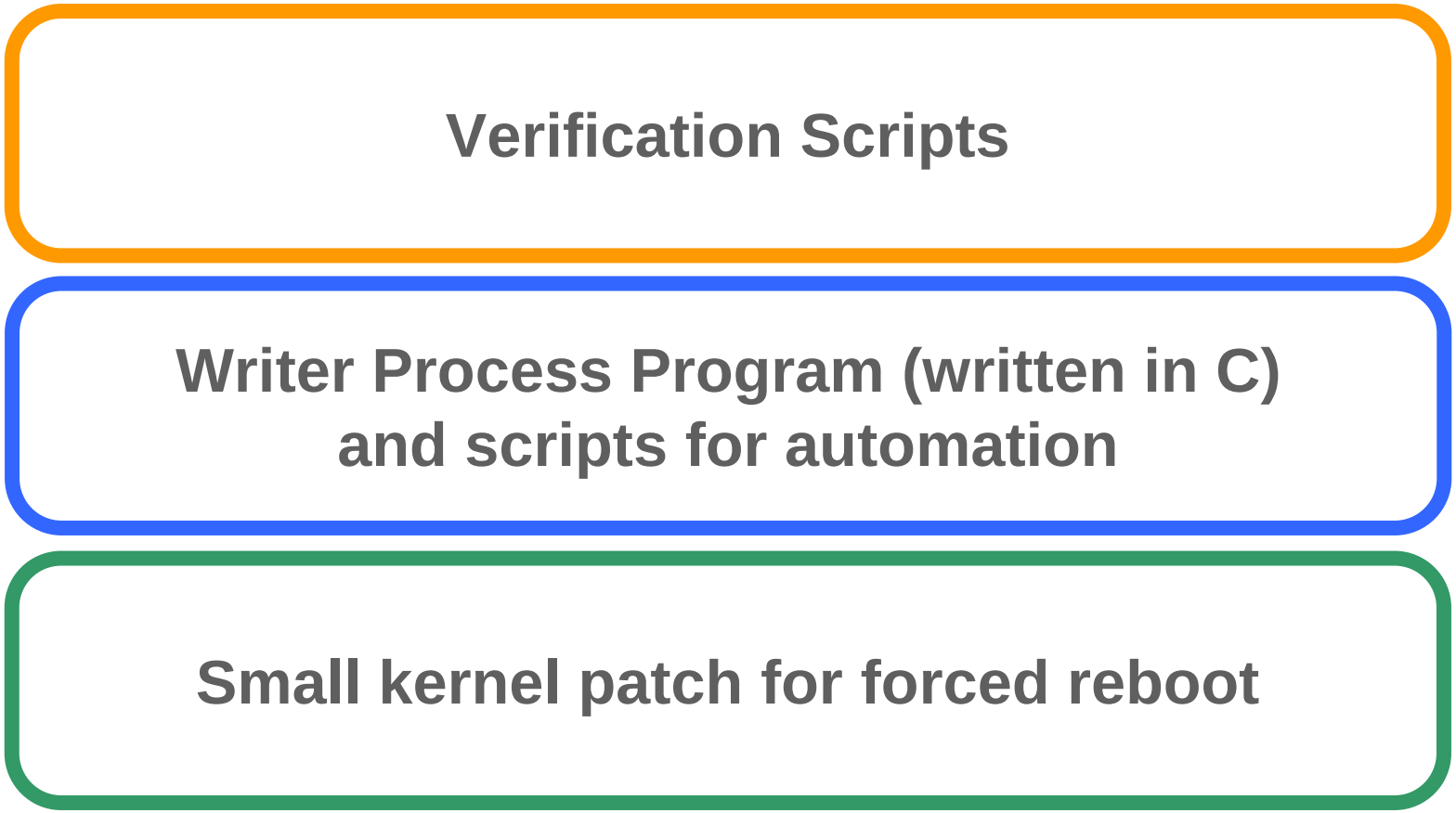
Verification

Verify the following metrics

- file size
- file contents



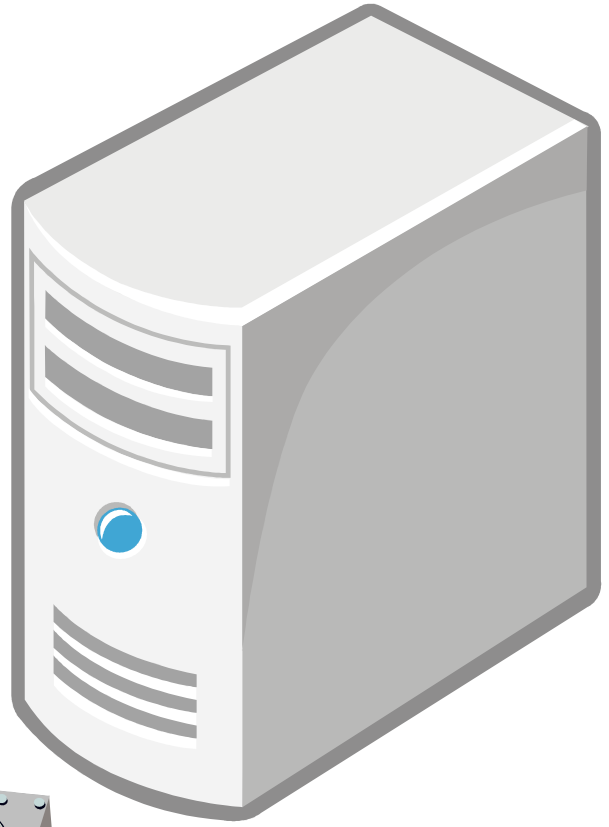
Simple software stack

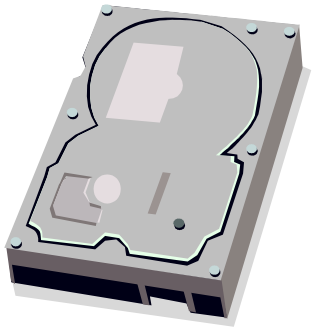


Environment

Hardware

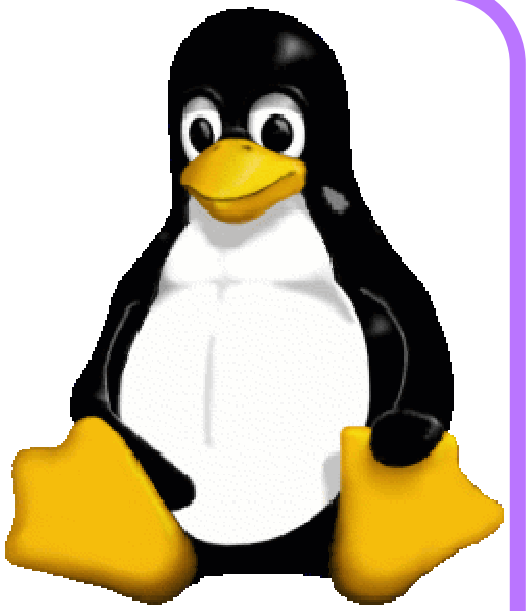
- Host1
  - CPU: Celeron 2.2GHz, Mem 1GB
  - HDD: IDE 80GB (2MB cache)
- Host2
  - CPU: Pentium4 2.8GHz, Mem 2GB
  - HDD: SATA 500GB (16MB cache)





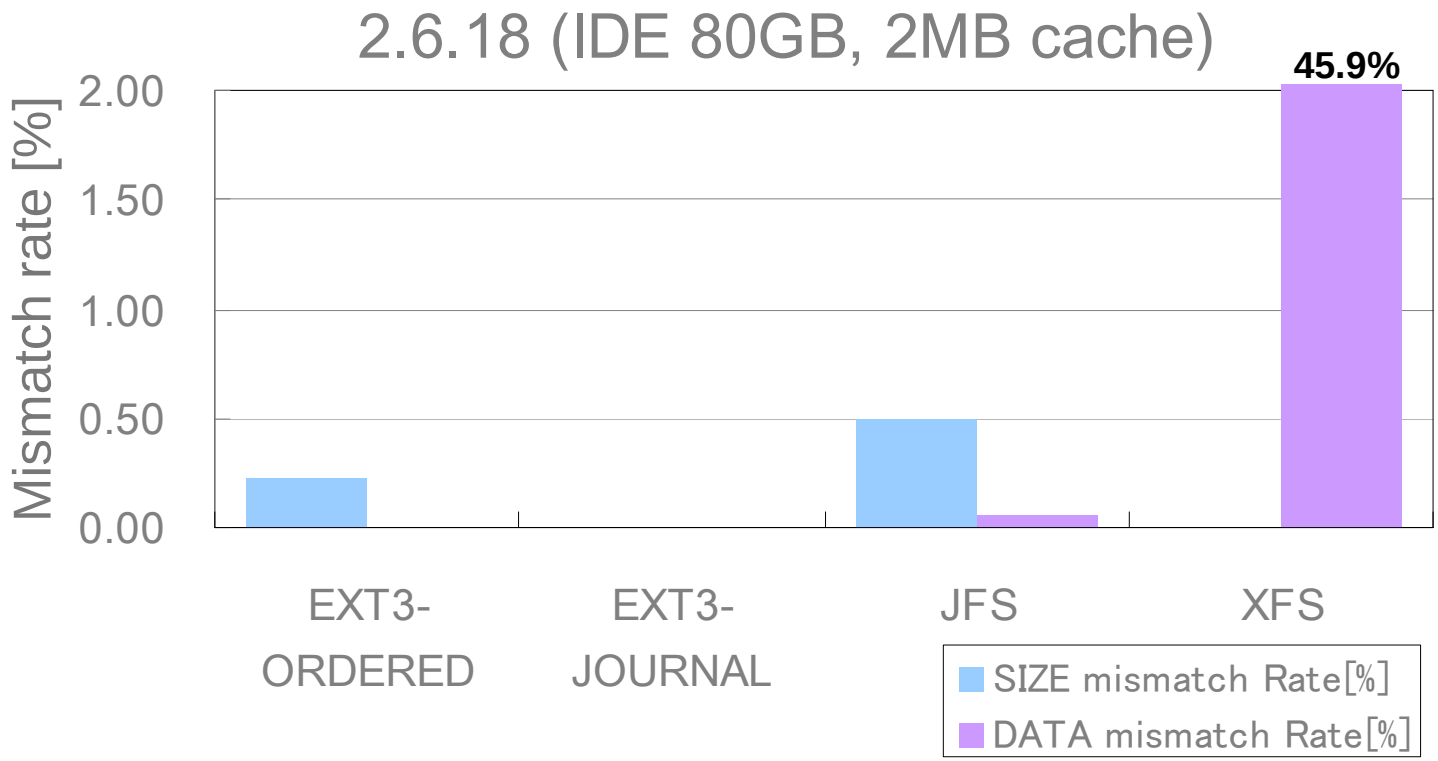
Environment

Software



- Kernel version
  - 2.6.18 (Host1 only)
  - 2.6.31.5 (Host1 and Host2)
  - 2.6.33 (Host2 only)
- File system
  - ext3 (data=ordered or data=journal)
  - xfs (osyncisosync)
  - jfs
  - ext4 (data=ordered or data=journal)
- I/O scheduler
  - kernel 2.6.18 tested with noop scheduler only
  - kernel 2.6.31.5 and 2.6.33 are tested with all I/O schedulers
    - noop, cfq, deadline, anticipatory(2.6.31.5 only)

Summary: kernel-2.6.18 (IDE 80GB, 2MB cache)



- Number of samples: 1800
- Rate = F / (W * T)
  - Total number of mismatch: F
  - Number of writer procs: W
  - Number of trials: T

File System	SIZE mismatch		DATA mismatch	
	Count	Rate[%]	Count	Rate[%]
EXT3-ORDERED	4	0.22	0	0.00
EXT3-JOURNAL	0	0.00	0	0.00
JFS	9	0.50	1	0.06
XFS	0	0.00	827	45.94

Perspectives

The test results summarized in three different perspectives

- test cases
  - create, append, overwrite, open->write->close
- I/O schedulers
  - noop, deadline, cfq, anticipatory
- write size to disk
  - 128, 256, 4096, 8192, 16384

Focused on Test case: kernel-2.6.18 (IDE 80GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	create	0	0
	append	0	0
	overwrite	0.89	0
	write->close	0	0
ext3(journal)	create	0	0
	append	0	0
	overwrite	0	0
	write->close	0	0
JFS	create	2.00	0
	append	0	0
	overwrite	0	0.22
	write->close	0	0
XFS	create	0	69.33
	append	0	58.22
	overwrite	0	0
	write->close	0	56.22

- #samples: 450

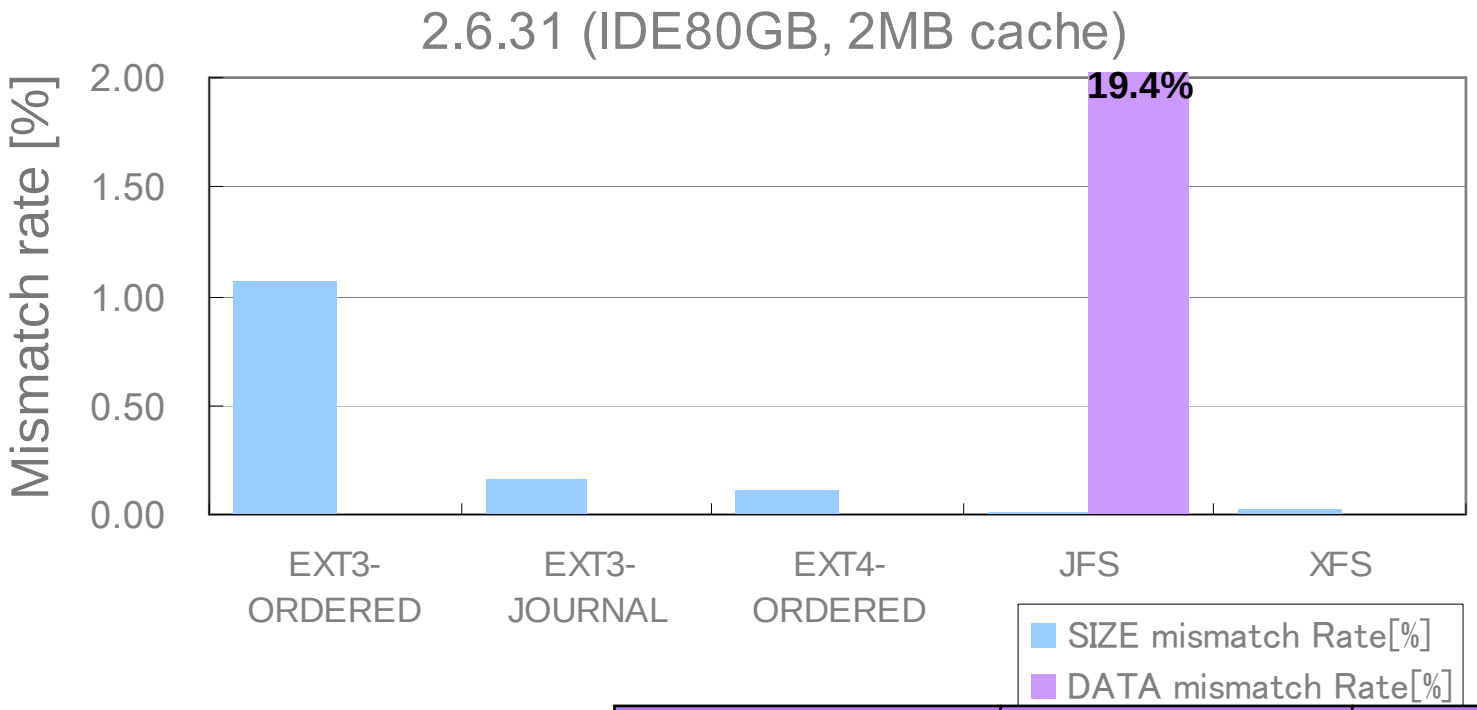
Focused on write size: kernel-2.6.18 (IDE 80GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	256	0	0
	4096	0	0
	8192	0.67	0
ext3(journal)	256	0	0
	4096	0	0
	8192	0	0
XFS	128	0	25.50
	4096	0	58.83
	8192	0	53.5
JFS	128	0	0
	4096	0	0.17
	8192	1.5	0

- #samples: 600

The bigger write size , the more size mismatch ??

Summary: kernel-2.6.31.5 (IDE80GB, 2MB cache)



- Number of samples: 16000

File System	SIZE mismatch		DATA mismatch	
	Count	Rate[%]	Count	Rate[%]
EXT3-ORDERED	171	1.07	0	0
EXT3-JOURNAL	25	0.16	0	0
EXT4-ORDERED	17	0.11	0	0
JFS	2	0.01	3104	19.40
XFS	3	0.02	0	0

Focused on test case: kernel-2.6.31.5 (IDE 80GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	create	1.20	0
	append	0.70	0
	overwrite	1.13	0
	write->close	1.25	0
ext3(journal)	create	0.45	0
	append	0	0
	overwrite	0	0
	write->close	0.18	0
ext4(ordered)	create	0	0
	append	0	0
	overwrite	0.43	0
	write->close	0	0
XFS	create	0	0
	append	0	0
	overwrite	0.08	0
	write->close	0	0
JFS	create	0	26.08
	append	0	25.58
	overwrite	0.05	0
	write->close	0	25.95

- #samples: 4000

Focused on I/O sched: kernel-2.6.31.5 (IDE 80GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	noop	0.45	0
	deadline	0.33	0
	cfq	2.00	0
	anticipatory	1.50	0
ext3(journal)	noop	0	0
	deadline	0	0
	cfq	0.40	0
	anticipatory	0.23	0
ext4(ordered)	noop	0	0
	deadline	0	0
	cfq	0	0
	anticipatory	0.43	0
XFS	noop	0.03	0
	deadline	0	0
	cfq	0.03	0
	anticipatory	0.03	0
JFS	noop	0.05	0
	deadline	0	0.98
	cfq	0	52.78
	anticipatory	0	23.85

- #samples: 4000

Focused on write size: kernel-2.6.31.5 (IDE 80GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	128	0	0
	256	0	0
	4096	0	0
	8192	3.13	0
	16384	2.22	0
ext3(journal)	128	0	0
	256	0	0
	4096	0	0
	8192	0.16	0
	16384	0.63	0
ext4(ordered)	128	0	0
	256	0	0
	4096	0	0
	8192	0.25	0
	16384	0.28	0
XFS	128	0	0
	256	0	0
	4096	0	0
	8192	0	0
	16384	0.09	0
JFS	128	0	20.06
	256	0	22.94
	4096	0.06	18.22
	8192	0	17.63
	16384	0	18.16

- #samples: 3200

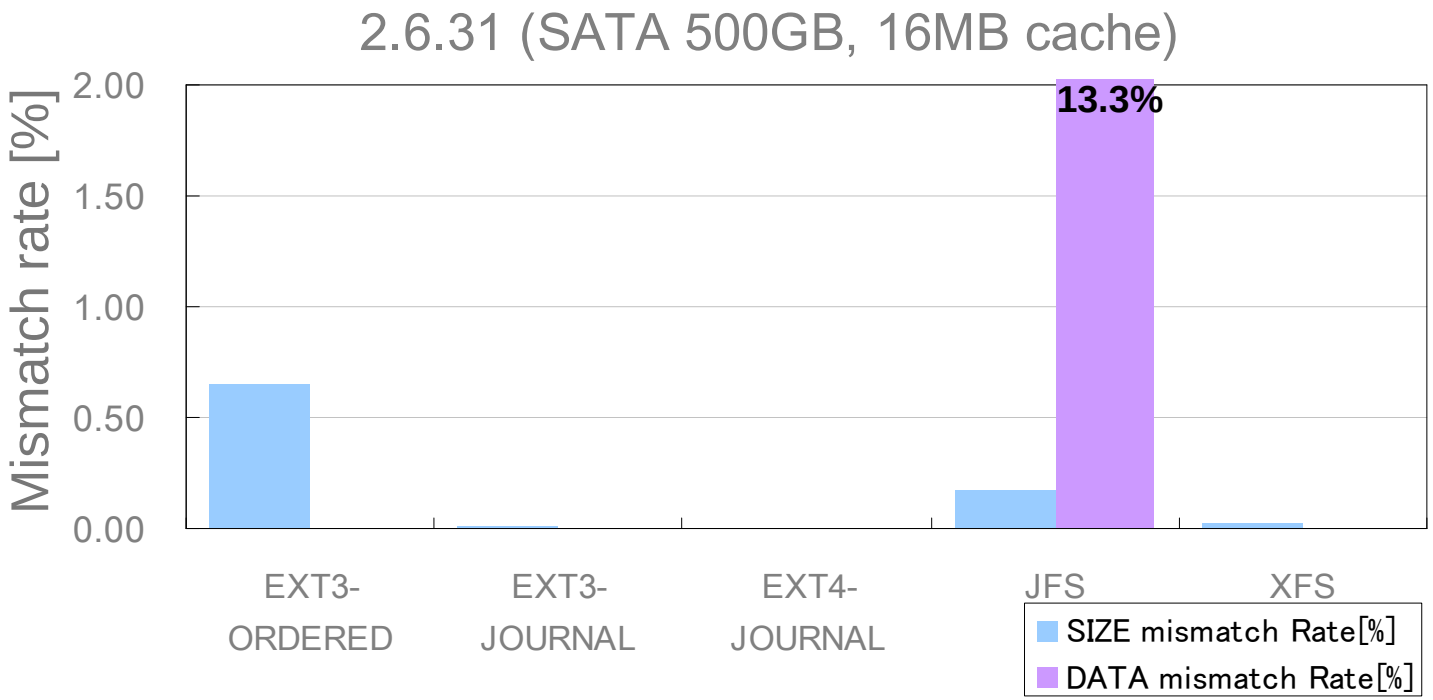
Focused on write size: kernel-2.6.31.5 (IDE 80GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	128	0	0
	256	0	0
	4096	0	0
	8192	3.13	0
	16384	2.22	0
ext3(journal)	128	0	0
	256	0	0
	4096	0	0
	8192	0.16	0
	16384	0.63	0
ext4(ordered)	128	0	0
	256	0	
	4096	0	
	8192	0.25	
	16384	0.28	
XFS	128	0	
	256	0	
	4096	0	0
	8192	0	0
	16384	0.09	0
JFS	128	0	20.06
	256	0	22.94
	4096	0.06	18.22
	8192	0	17.63
	16384	0	18.16

- #samples: 3200

The bigger write size, the more size mismatch ?

Summary: kernel-2.6.31 (SATA500GB, 16MB cache)



- Number of samples: 16000

File System	SIZE mismatch		DATA mismatch	
	Count	Rate[%]	Count	Rate[%]
EXT3-ORDERED	104	0.650	0	0.000
EXT3-JOURNAL	1	0.006	0	0.000
EXT4-JOURNAL	0	0.000	0	0.000
JFS	28	0.175	2129	13.306
XFS	3	0.019	0	0.000

Focused on test case: kernel-2.6.31.5 (SATA 500GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	create	0.85	0
	append	0.10	0
	overwrite	0.23	0
	write->close	1.43	0
ext3(journal)	create	0	0
	append	0	0
	overwrite	0	0
	write->close	0.03	0
ext4(journal)	create	0	0
	append	0	0
	overwrite	0	0
	write->close	0	0
XFS	create	0	0
	append	0	0
	overwrite	0.08	0
	write->close	0	0
JFS	create	0.23	17.9
	append	0.33	22.23
	overwrite	0.15	0
	write->close	0	13.10

- #samples: 4000

Focused on I/O sched: kernel-2.6.31.5 (SATA 500GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	noop	0.63	0
	deadline	0.90	0
	cfq	0.88	0
	anticipatory	0.20	0
ext3(journal)	noop	0	0
	deadline	0	0
	cfq	0	0
	anticipatory	0.03	0
ext4(journal)	noop	0	0
	deadline	0	0
	cfq	0	0
	anticipatory	0	0
XFS	noop	0.03	0
	deadline	0.03	0
	cfq	0.03	0
	anticipatory	0	0
JFS	noop	0.40	0.03
	deadline	0.28	0.38
	cfq	0	25.63
	anticipatory	0.03	27.20

- #samples: 4000

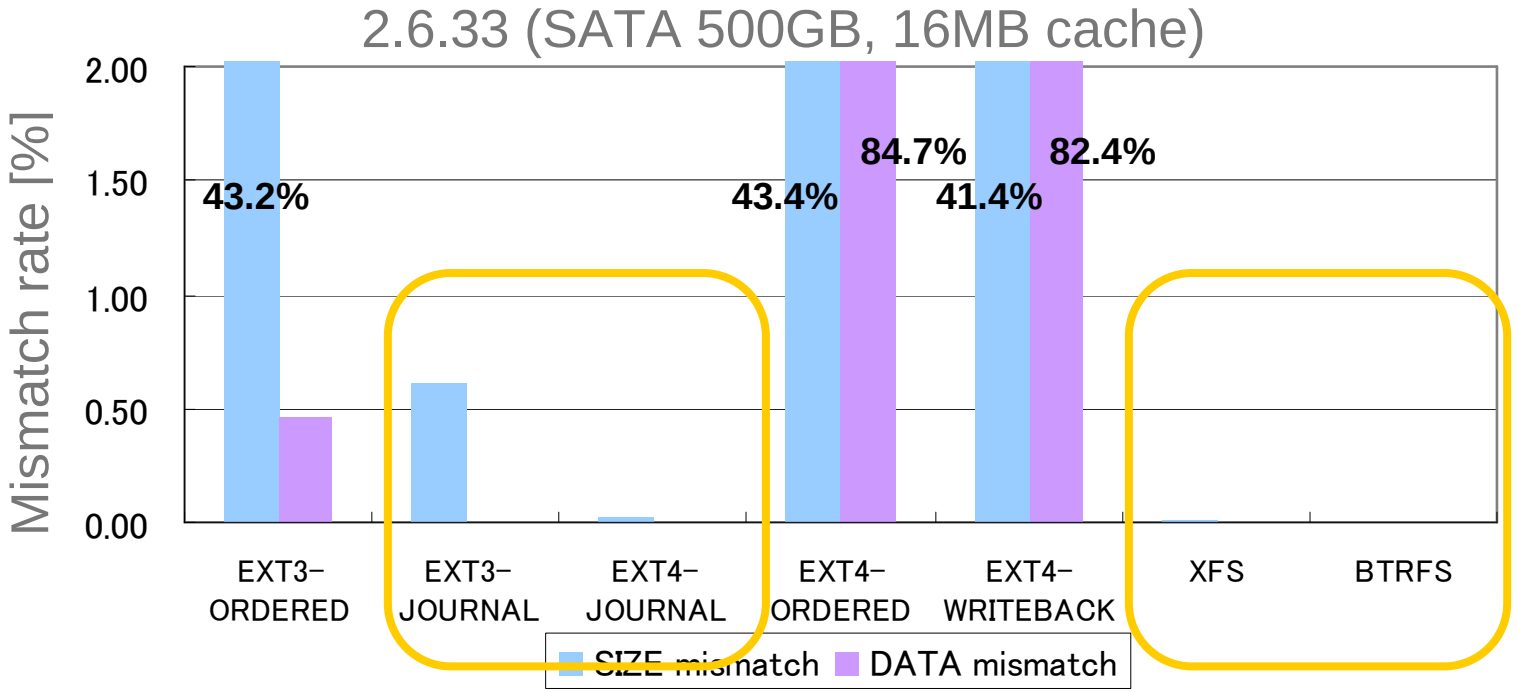
Focused on write size: kernel-2.6.31.5 (SATA 500GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(ordered)	128	0	0
	256	0	0
	4096	0	0
	8192	1.69	0
	16384	1.56	0
ext3(journal)	128	0	0
	256	0	0
	4096	0	0
	8192	0	0
	16384	0.03	0
ext4(journal)	128	0	0
	256	0	
	4096	0	
	8192	0	
	16384	0	
XFS	128	0	
	256	0	
	4096	0	0
	8192	0	0
	16384	0.09	0
JFS	128	0.66	13.44
	256	0	15.03
	4096	0	18.48
	8192	0	9.38
	16384	0.22	10.25

- #samples: 3200

The bigger write size, the more size mismatch

Summary: kernel-2.6.33 (SATA500GB, 16MB cache)



- Number of samples: 12000

File System	SIZE mismatch		DATA mismatch	
	Count	Rate[%]	Count	Rate[%]
EXT3-ORDERED	5179	43.16	55	0.46
EXT3-JOURNAL	74	0.62	0	0.00
EXT4-JOURNAL	3	0.03	0	0.00
EXT4-ORDERED	5205	43.38	10161	84.68
EXT4-WB	4965	41.38	9893	82.44
XFS	2	0.02	0	0.00
BTRFS	0	0.00	0	0.00

Focused on test case: kernel-2.6.33 (SATA 500GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(journal)	create	0.63	0
	append	0.73	0
	overwrite	0	0
	write->close	0.50	0
ext4(journal)	create	0.03	0
	append	0	0
	overwrite	0.05	0
	write->close	0	0
xfs	create	0	0
	append	0	0
	overwrite	0.05	0
	write->close	0	0
btrfs	create	0	0
	append	0	0
	overwrite	0	0
	write->close	0	0

- #samples: 4000

Focused on I/O sched: kernel-2.6.33 (SATA 500GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(journal)	noop	0.65	0
	deadline	0.53	0
	cfq	0.68	0
ext4(journal)	noop	0	0
	deadline	0.05	0
	cfq	0.03	0
xfs	noop	0	0
	deadline	0.03	0
	cfq	0.03	0
btrfs	noop	0	0
	deadline	0	0
	cfq	0	0

- #samples: 4000

Focused on write size: kernel-2.6.33 (SATA 500GB)

File System	Test case	Size mismatch [%]	Data mismatch [%]
ext3(journal)	128	0	0
	256	0	0
	4096	0	0
	8192	1.13	0
	16384	1.96	0
ext4(journal)	128	0	0
	256	0	0
	4096	0	0
	8192	0.08	0
	16384	0.42	0
XFS	128	0	
	256	0	
	4096	0	
	8192	0	
	16384	0.08	
btrfs	128	0	0
	256	0	0
	4096	0	0
	8192	0	0
	16384	0	0

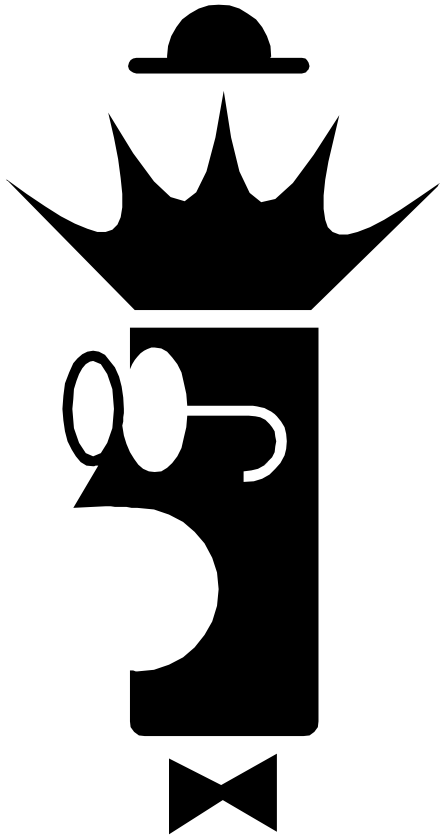
- #samples: 2400

The bigger write size, the more size mismatch

Try to evaluate experimental file systems…

Evaluation failed on….

- nilfs2
  - caused file system full
  - nilfs_cleanerd not fast enough
- btrfs
  - caused kernel crash
  - couldn’t recovery anymore

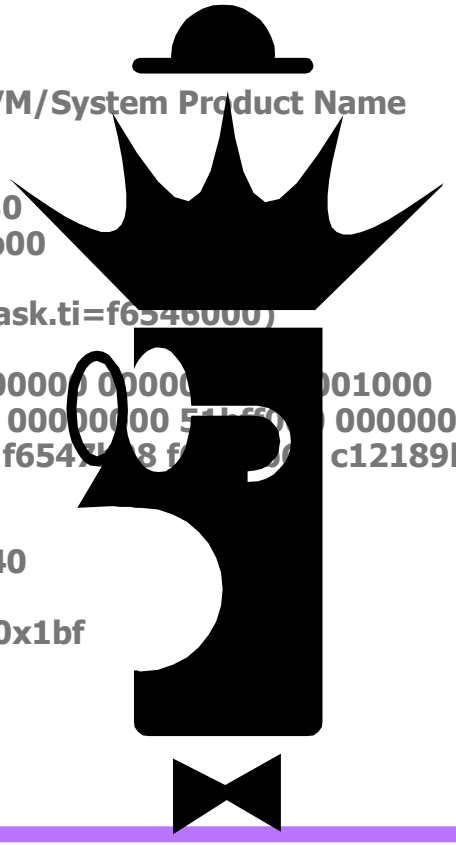


Btrfs error log

Error Log

```
[    9.610419] ------------[ cut here ]------------
[    9.610508] kernel BUG at fs/btrfs/free-space-cache.c:446!
[    9.610588] invalid opcode: 0000 [#1] SMP
[    9.610715] last sysfs file: /sys/devices/virtual/net/lo/operstate
[    9.610794] Modules linked in:
[    9.610893]
[    9.610966] Pid: 1716, comm: mount Not tainted 2.6.33 #1 P5S800-VM/System Product Name
[    9.611090] EIP: 0060:[<c124ff76>] EFLAGS: 00010286 CPU: 1
[    9.611180] EIP is at remove_from_bitmap+0x6f/0x265
[    9.611252] EAX: ffffffff EBX: f6b7b240 ECX: 00008001 EDX: f6547b30
[    9.611252] ESI: f6547b98 EDI: f6547b7c EBP: f6547b4c ESP: f6547b00
[    9.611252]  DS: 007b ES: 007b FS: 00d8 GS: 0033 SS: 0068
[    9.611252] Process mount (pid: 1716, ti=f6546000 task=f7158f30 task.ti=f6546000)
[    9.611252] Stack:
[    9.611252]  08000000 00000000 f6547b34 f6547b2c c129ba78 49c00000 00000[illegible]001000
[    9.611252] <0> 00000000 00000000 f6a40000 f6a40000 00002000 00000[illegible]00 [illegible] 00000000
[    9.611252] <0> 00000000 00000000 f6b7b240 f6547b90 c1250c0d f654[illegible]8 f[illegible] c12189bd
[    9.611252] Call Trace:
[    9.611252]  [<c129ba78>] ? div64_u64+0x4a/0x52
[    9.611252]  [<c1250c0d>] ? btrfs_remove_free_space+0x315/0x340
[    9.611252]  [<c12189bd>] ? spin_lock+0x8/0xa
[    9.611252]  [<c121b605>] ? btrfs_alloc_logged_file_extent+0x80/0x1bf
[    9.611252]  [<c12188da>] ? btrfs_lookup_extent+0x5c/0x65
[    9.611252]  [<c124d333>] ? replay_one_extent+0x38f/0x518
```

Cont....



Conclusion

Evaluation result shows:

- XFS and JFS data/size mismatch rate depends on kernel version
- SYNC write mode is not safe enough in most cases
- Large write size caused more data inconsistency than small size
- BEST result in EXT4-Journal on 2.6.31
  - effects of write barriers?
- GOOD results on XFS(for 2.6.31 and 33) and Ext3-journal
  - NOTE: Ext3 performance is much better than XFS in random write

Future work

- evaluate other file systems


TOSHIBA
Leading Innovation