TOSHIBA
Leading Innovation

Evaluation of Data Reliability on Linux File Systems

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

Dec. 18, 2009

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

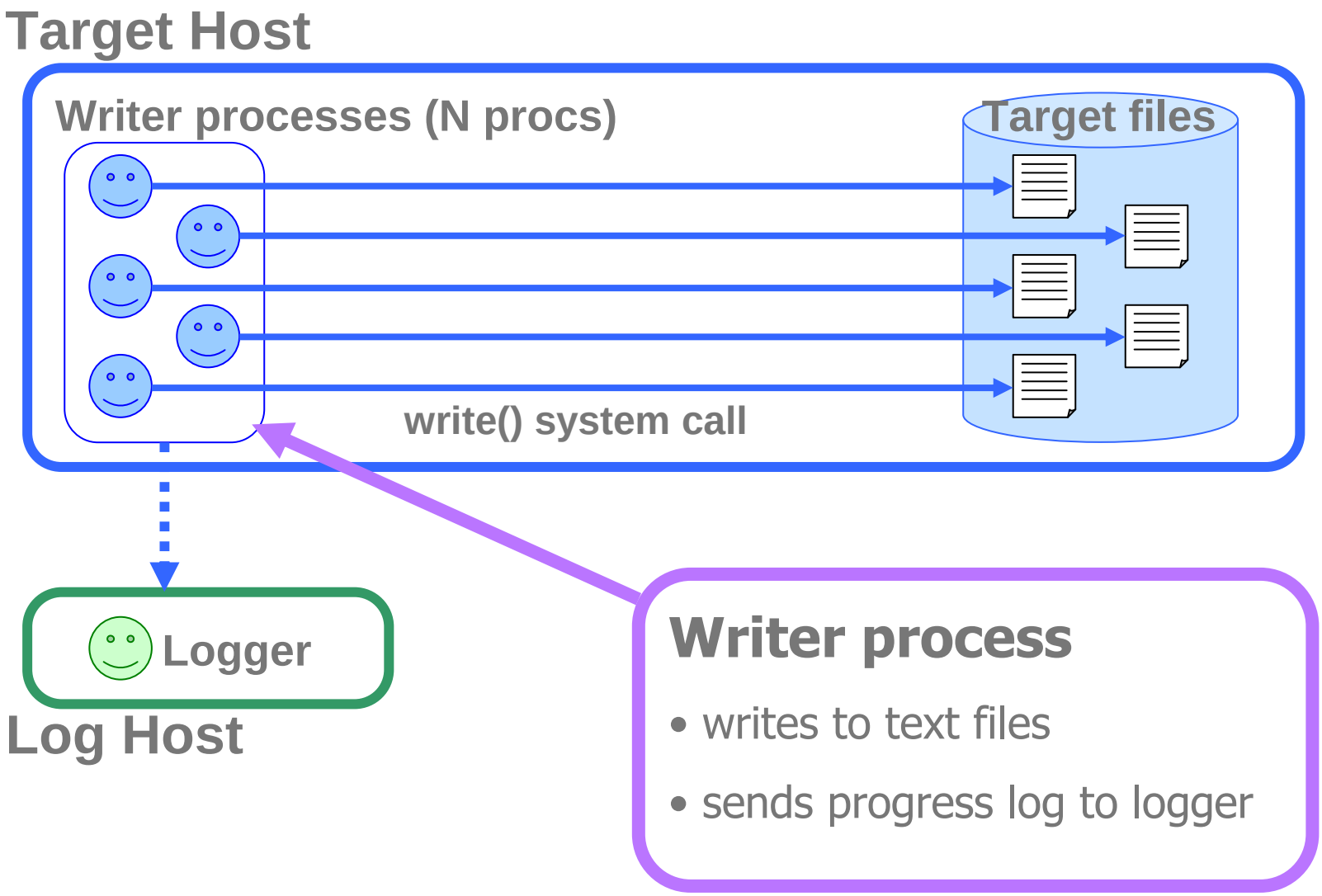
Focus

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

Metrics

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

Evaluation: Overview



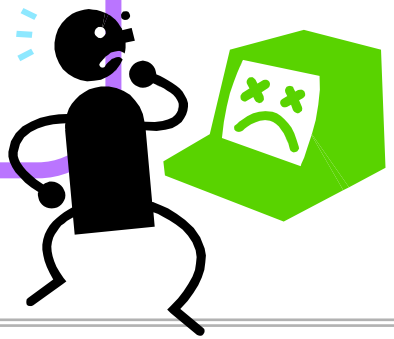
Target Host

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

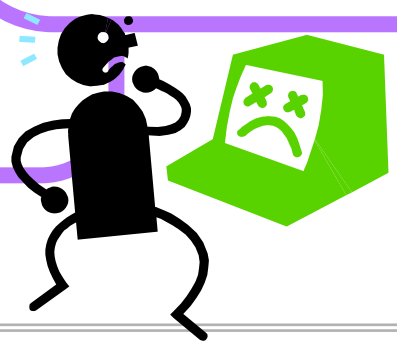
- writes to text files
- sends progress log to lo

How to crash

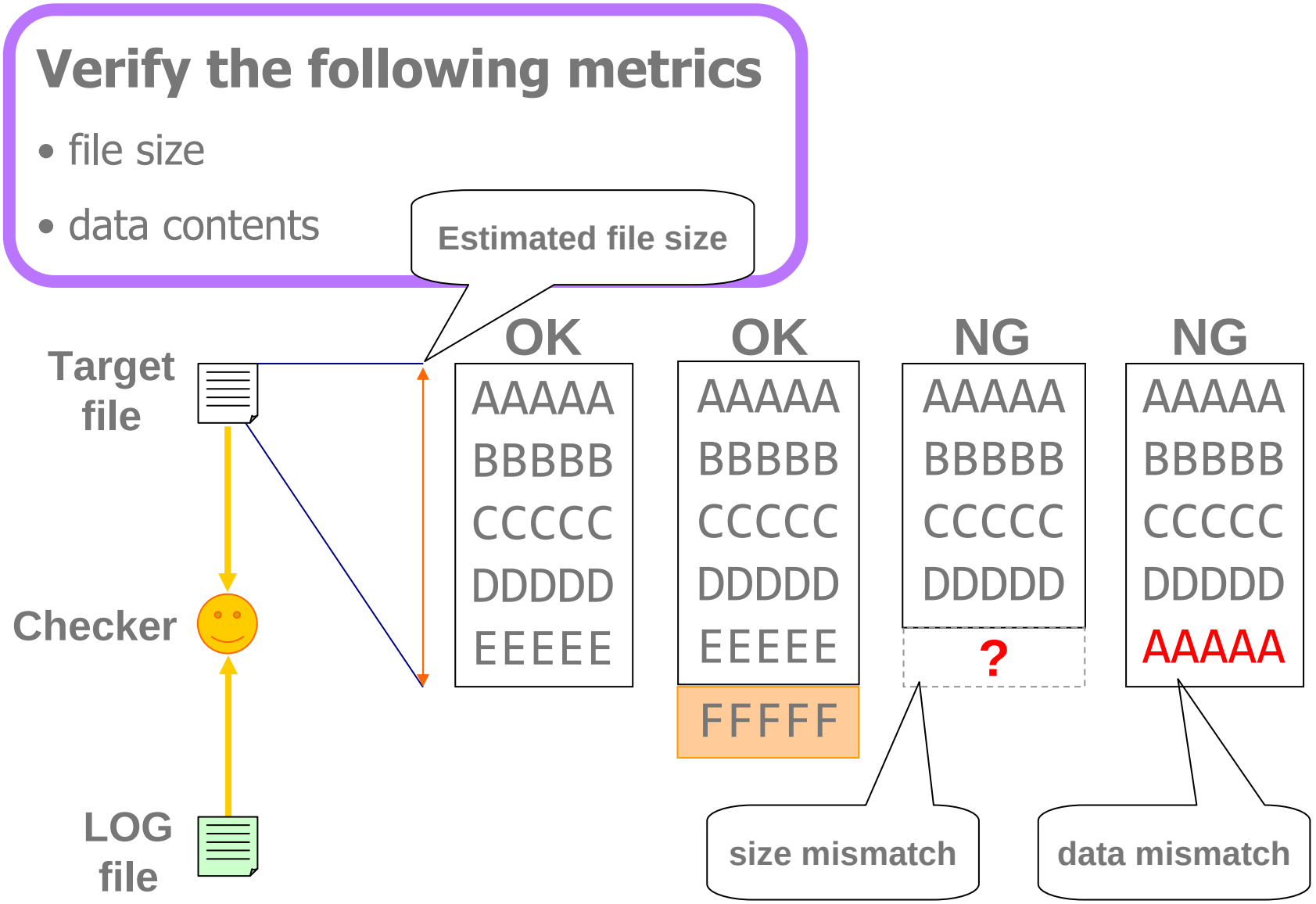
- modified reboot system
  - forced to reboot
  - 10 seconds to reboot

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()



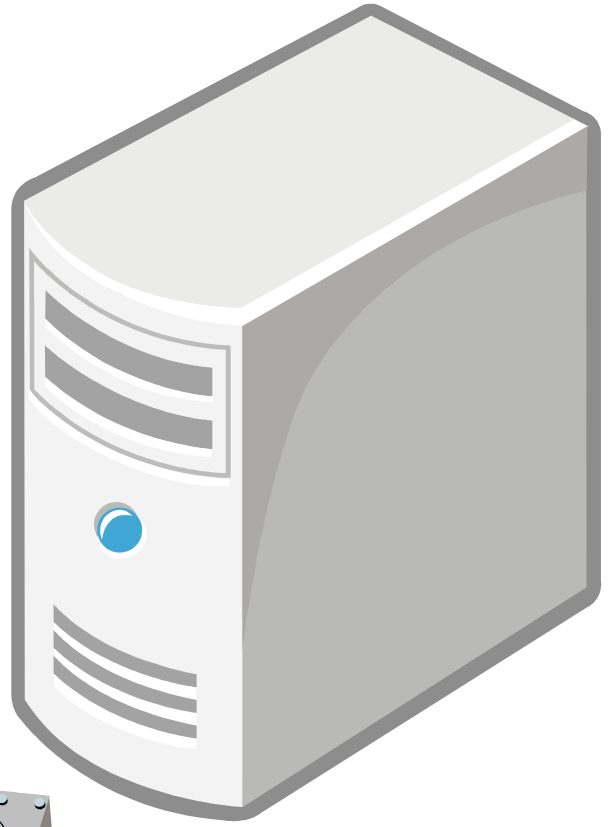
Verification

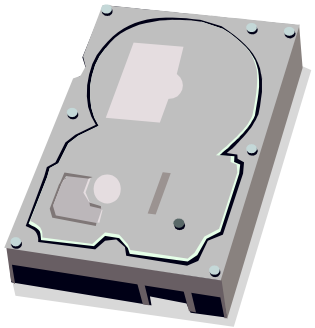


Environment

Hardware

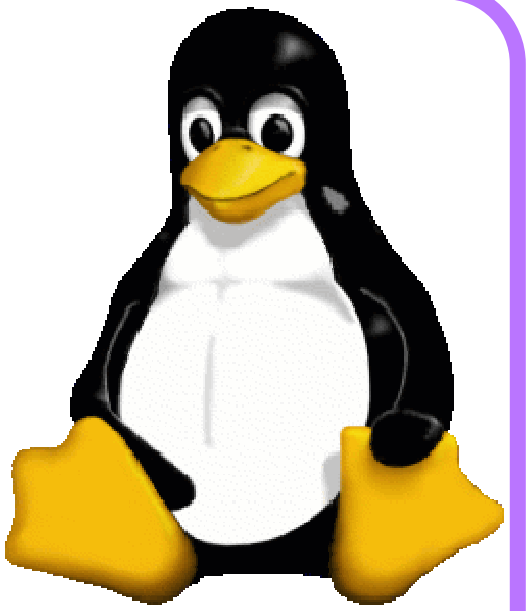
- Host1
  - CPU: Celeron 2.2GHz, Mem 1GB
  - HDD: IDE 80GB (2MB cache)
- Host2
  - CPU: Pentium4 2.8GHz, Mem 2GB
  - HDD: SATA 500GB (16MB cache)





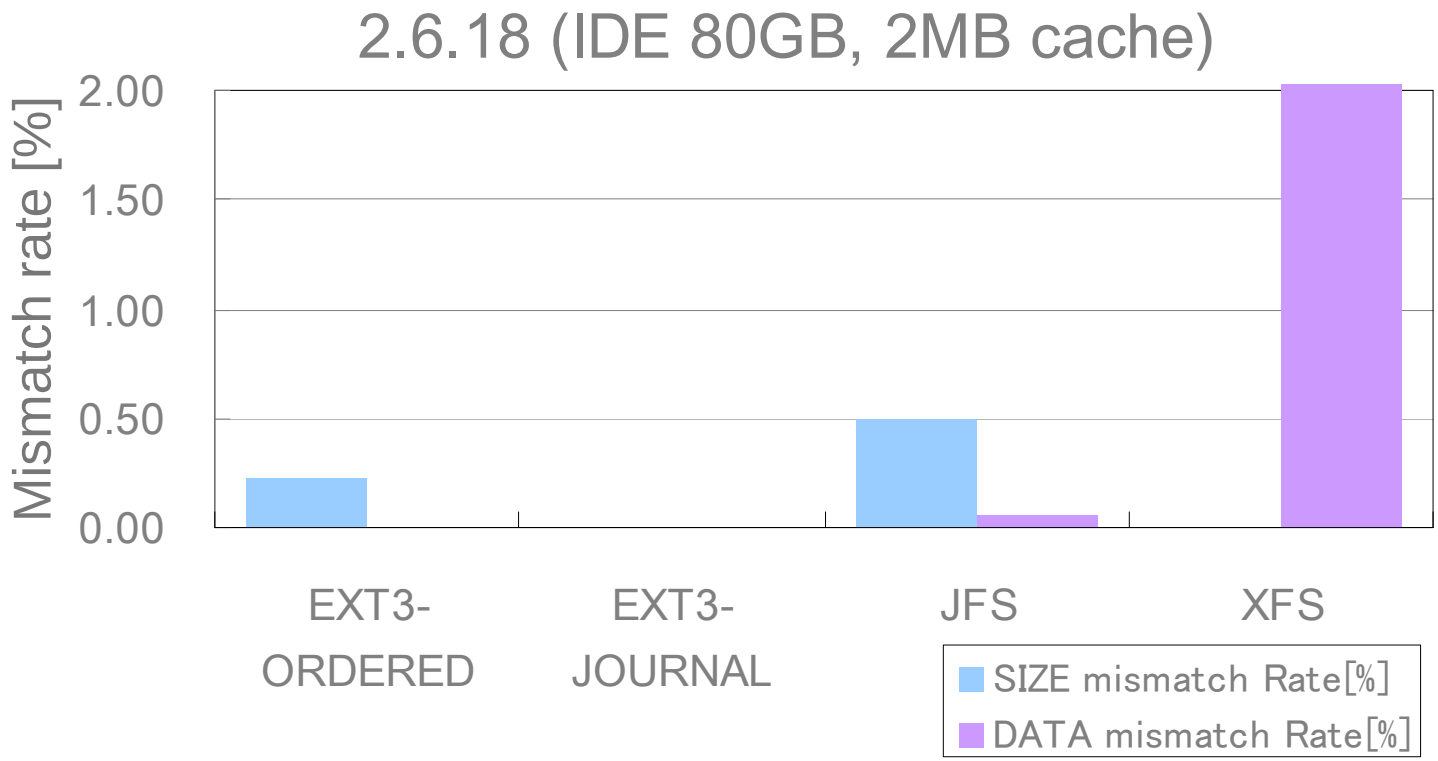
Environment

Software



- Kernel version
  - 2.6.18 (Host1 only)
  - 2.6.31.5
- File system
  - ext3 (data=ordered or data=journal)
  - xfs (osyncisosync)
  - jfs
  - ext4 (data=ordered used on Host 1, data=journal used on Host2)
- I/O scheduler
  - kernel 2.6.18 tested with noop scheduler only
  - kernel 2.6.31.5 tested with all I/O schedulers
    - noop, cfq, deadline, anticipatory

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

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

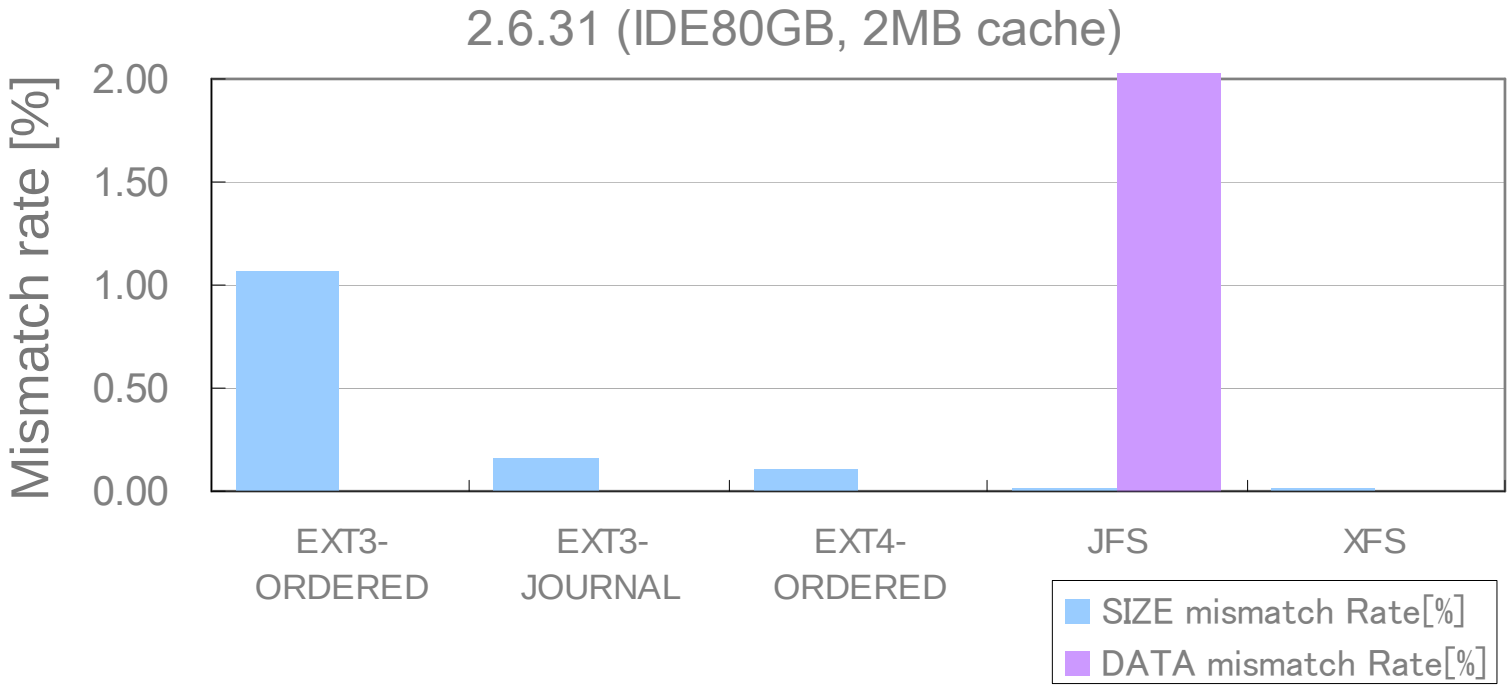
Focused on write size: kernel-2.6.18 (IDE 80GB)

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

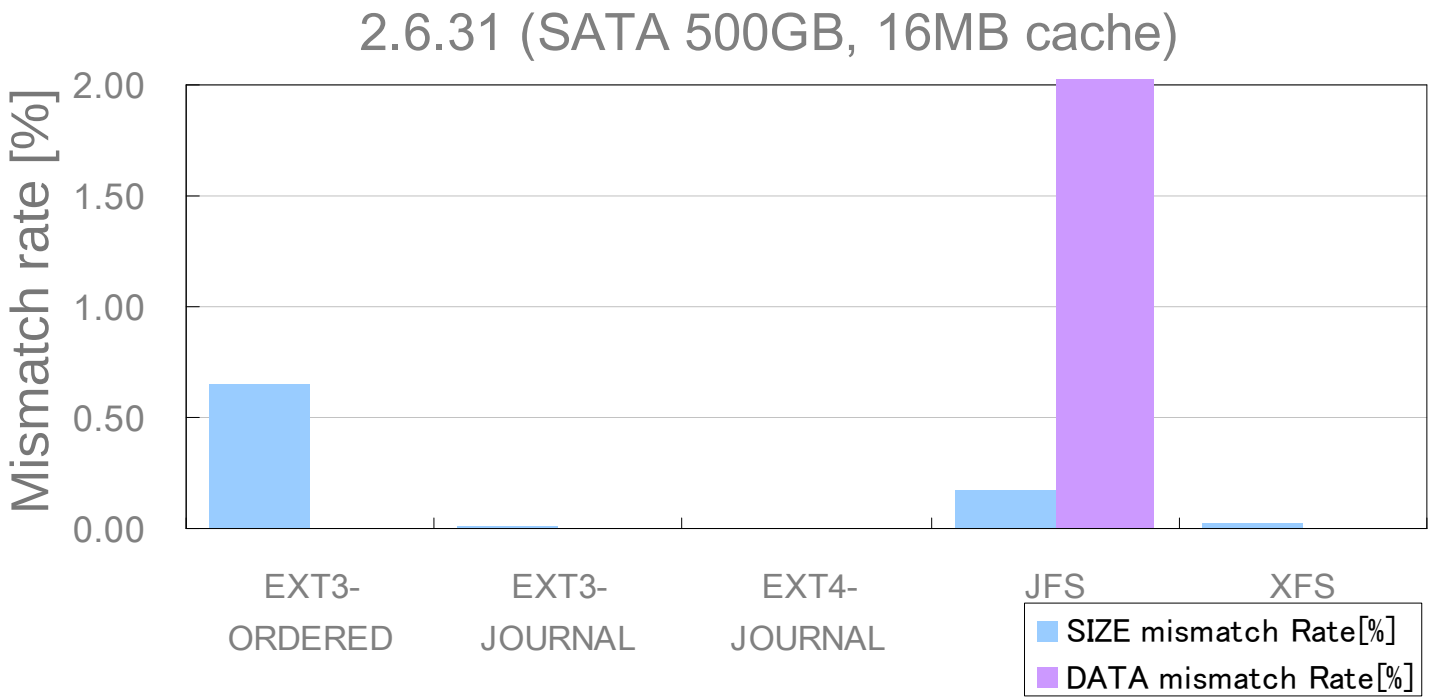
Focused on write size: kernel-2.6.31.5 (IDE 80GB)

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

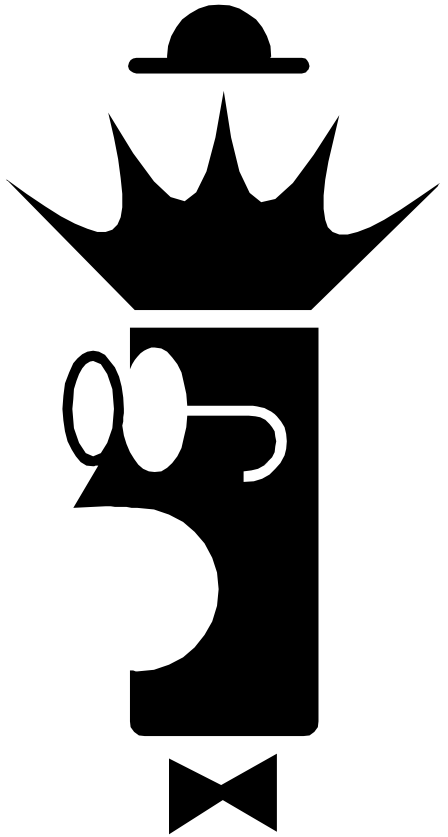
- #samples: 3200

The bigger write size, the more size mismatch

Try to evaluate other file systems…

Evaluation failed

- nilfs2
  - caused file system full
  - nilfs_cleanerd not fast enough
- btrfs
  - caused kernel crash
  - recovery failure



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
- BEST result on EXT4 with journal mode
  - effects of write barriers?
- GOOD results on XFS(only 2.6.31.5) and Ext3-journal
  - NOTE: Ext3 performance is much better than XFS (random write)

Future work

- evaluate other file systems


TOSHIBA
Leading Innovation