On Fault Resilience of File System Checkers

Om Rameshwar Gatla Mai Zheng
Computer Science Department, New Mexico State University

Abstract

File system checkers serve as the last line of defense to recover a corrupted file system back to a consistent state. In this position paper, we study the behavior of file system checkers under emulated faults. We answer two important questions: instead of fixing the original corruption, will an interrupted checker cause more severe damage? If so, can the additional damage be fixed by the existing checker? Our preliminary results show that there are vulnerabilities in popular file system checkers which could lead to unrecoverable data loss under faults.

1 Motivation

Despite of various protection techniques [19, 12, 10, 18, 6], file systems may still become corrupted for various reasons including power outages, system crashes, hardware failures, software bugs, etc [15, 16, 9, 5, 14, 20, 17]. Thus, most file systems come with a checker to serve as the last line of defense to recover the corrupted file system back to a healthy state [15, 2, 11, 1, 3].

Due to such importance, abundant work has been done to improve file system checkers in terms of both performance and reliability [15, 13, 8]. Complementary to the existing efforts, in this paper we study the behavior of file system checkers under faults. This is motivated by a recent accident happened at the High Performance Computing Center (HPCC) in Texas [7, 4], where the Lustre file system suffered severe data loss after experiencing two consecutive power outages: the first one triggered the Lustre checker (i.e., LFSCK [2]) after restarting the cluster, while the second one interrupted LFSCK and led to the final downtime. Since Lustre is built on top of a variant of Ext4 and LFSCK relies on the local file system checker, the overall checking and recovery procedure is complicated. As one step to pinpoint the vulnerabilities and build robust file system checkers, we perform a comprehensive study on the fault resilience of `e2fsck` [1], the default checker for the widely used Ext2/Ext3/Ext4 file systems.

Corruption Types	Percentage
unmountable	0.57%
file content corruption	2.85%
misplacement of files	6.28%
others	0.57%

Table 1: Four types of unrecoverable corruption incurred by an interrupted `e2fsck`; the percentage is defined as the number of occurrences divided by the total number of test images.

2 Methodology

We build a fault-injection framework for `e2fsck` based on a record-and-replay tool [20]. The developers of Ext-series file systems maintain a set of corrupted file system images for regression testing [1], which we utilize as the input to our framework to trigger the file system checker. We record the I/O traffic during the execution of the checker, and replay partial I/O blocks to emulate the effect of an interrupted checker on the file system image. Moreover, after generating the interrupted state, we re-run the checker again without any fault injection. This is to verify that whether the corruption introduced by the interrupted checker can be recovered or not.

3 Preliminary Results

We have evaluated `e2fsck` 1.43.1 using our prototype. 175 test images from `e2fsprogs` [1] were used as inputs. To emulate faults on storage systems with different atomic block sizes, we inject faults in two granularities: after each 512B or 4KB block. Table 1 summarizes the results under 4KB granularity. We classify the new corruptions observed into four types: (1) *unmountable*; (2) *file content corruption*; (3) *misplacement of files*; (4) *others* (e.g., showing "???" after an `ls` command). Note that all these corruptions were observed after running the checker again without interruption, which means they cannot be recovered by the exiting checker. We believe our study can raise the awareness of the reliability vulnerabilities and will serve as a foundation for building fault-resilient checkers for file systems.

4 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant Number 1566554.

References

[1] E2fsprogs: Ext2/3/4 Filesystem Utilities. http://e2fsprogs.sourceforge.net/.

[2] LFSCK: an online file system checker for Lustre. https://github.com/Xyratex/lustre-stable/blob/master/Documentation/lfsck.txt.

[3] XFS File System Utilities. https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Storage_Administration_Guide/xfsothers.html.

[4] Texas Tech University HPCC Power Outage Event. Email Announcement by HPCC at 8:50:17 AM CST on Monday January 11, 2016.

[5] BAIRAVASUNDARAM, L. N., ARPACI-DUSSEAU, A. C., ARPACI-DUSSEAU, R. H., GOODSON, G. R., AND SCHROEDER, B. An analysis of data corruption in the storage stack. *Trans. Storage 4*, 3 (Nov. 2008), 8:1–8:28.

[6] BAIRAVASUNDARAM, L. N., SUNDARARAMAN, S., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. Tolerating file-system mistakes with envyfs. In *Proceedings of the 2009 Conference on USENIX Annual Technical Conference* (Berkeley, CA, USA, 2009), USENIX'09, USENIX Association, pp. 7–7.

[7] CAO, J., WANG, S., DAI, D., ZHENG, M., AND CHEN, Y. A generic framework for testing parallel file systems. In *Proceedings of the 1st Joint International Workshop on Parallel Data Storage & Data Intensive Scalable Computing Systems* (Piscataway, NJ, USA, 2016), PDSW-DISCS '16, IEEE Press, pp. 49–54.

[8] CARREIRA, J. A. C. M., RODRIGUES, R., CANDEA, G., AND MAJUMDAR, R. Scalable testing of file system checkers. In *Proceedings of the 7th ACM European Conference on Computer Systems* (New York, NY, USA, 2012), EuroSys '12, ACM, pp. 239–252.

[9] CHEN, H., ZIEGLER, D., CHAJED, T., CHLIPALA, A., KAASHOEK, M. F., AND ZELDOVICH, N. Using crash hoare logic for certifying the fscq file system. In *Proceedings of the 25th Symposium on Operating Systems Principles* (New York, NY, USA, 2015), SOSP '15, ACM, pp. 18–37.

[10] CHIDAMBARAM, V., SHARMA, T., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. Consistency Without Ordering. In *Proceedings of the 10th Conference on File and Storage Technologies (FAST'12)* (San Jose, California, February 2012).

[11] FARNUM, G. CephFS fsck: distributed file system checking. http://events.linuxfoundation.org/sites/events/files/slides/Vault%20CephFS%20fsck.pdf.

[12] GANGER, G. R., MCKUSICK, M. K., SOULES, C. A., AND PATT, Y. N. Soft updates: a solution to the metadata update problem in file systems. *ACM Transactions on Computer Systems (TOCS) 18*, 2 (2000), 127–153.

[13] GUNAWI, H. S., RAJIMWALE, A., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. Sqck: A declarative file system checker. In *Proceedings of the 8th USENIX Conference on Operating Systems Design and Implementation* (Berkeley, CA, USA, 2008), OSDI'08, USENIX Association, pp. 131–146.

[14] LU, L., ARPACI-DUSSEAU, A. C., ARPACI-DUSSEAU, R. H., AND LU, S. A study of linux file system evolution. In *Presented as part of the 11th USENIX Conference on File and Storage Technologies (FAST 13)* (San Jose, CA, 2013), USENIX, pp. 31–44.

[15] MA, A., DRAGGA, C., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. ffsck: The fast file system checker. In *Presented as part of the 11th USENIX Conference on File and Storage Technologies (FAST 13)* (San Jose, CA, 2013), USENIX, pp. 1–15.

[16] MIN, C., KASHYAP, S., LEE, B., SONG, C., AND KIM, T. Cross-checking semantic correctness: The case of finding file system bugs. In *Proceedings of the 25th Symposium on Operating Systems Principles* (New York, NY, USA, 2015), SOSP '15, ACM, pp. 361–377.

[17] PILLAI, T. S., CHIDAMBARAM, V., ALAGAPPAN, R., AL-KISWANY, S., ARPACI-DUSSEAU, A. C., AND ARPACI-DUSSEAU, R. H. All file systems are not created equal: On the complexity of crafting crash-consistent applications. In *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI '14)* (October 2014).

[18] STEIN, C. A., HOWARD, J. H., AND SELTZER, M. I. Unifying file system protection. In *Proceedings of the General Track: 2001 USENIX Annual Technical Conference* (Berkeley, CA, USA, 2001), USENIX Association, pp. 79–90.

[19] TWEEDIE, S. C. Journaling the linux ext2fs filesystem. In *In LinuxExpo?98: Proceedings of The 4th Annual Linux Expo* (1998).

[20] ZHENG, M., TUCEK, J., HUANG, D., QIN, F., LILLIBRIDGE, M., YANG, E. S., ZHAO, B. W., AND SINGH, S. Torturing databases for fun and profit. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)* (Broomfield, CO, 2014), USENIX Association, pp. 449–464.

FAST'17
FEB 27–MAR 2, 2017
SANTA CLARA, CA
www.usenix.org/fast17

usenix ASSOCIATION

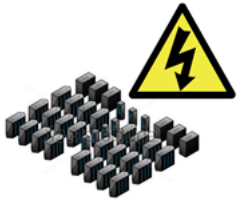
On Fault Resilience of File System Checkers

Om Rameshwar Gatla, Mai Zheng; New Mexico State University

NM STATE UNIVERSITY

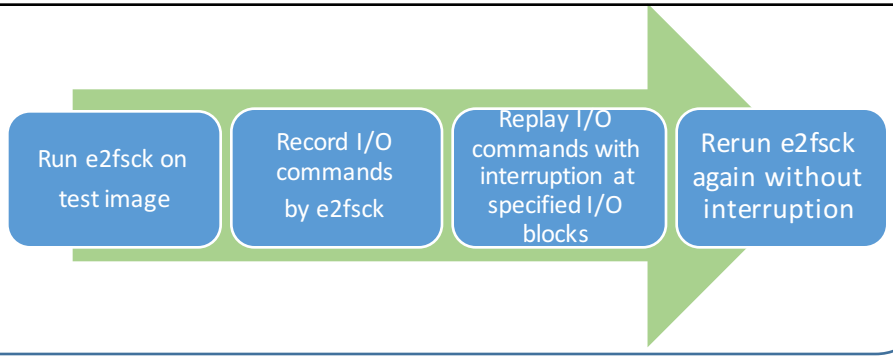
Motivation

- Two consecutive power outages happened at High Performance Computing Center (HPCC) in Texas in 2016
  - One triggered the Lustre file system checker (LFSCK) on restart, another interrupted LFSCK
  - Resulted in severe data loss
- Checking and recovery in Lustre-like systems is complicate
  - LFSCK relies on local checkers
  - need a systematic testing & analyzing methodology



Will an interrupted checker cause more damage to the FS? If so, can the additional damage be fixed by the existing checker?

Methodology



Preliminary Results

- Use 175 test images from *e2fsprogs*
- Two granularities for fault injection: every 4KB or 512B
- 17 images cannot be recovered completely with fault injection size of 4KB
- 34 images cannot be recovered completely with fault injection size of 512B

Corruption Type	Percentage	
	4 KB	512 B
Un-mountable	0.57%	11.42%
File content corruption	2.85%	5.14%
Misplacement of files	6.28%	5.14%
Others	0.57%	0.57%

Table 1. Four types of corruption observed and the corresponding percentages among all test images

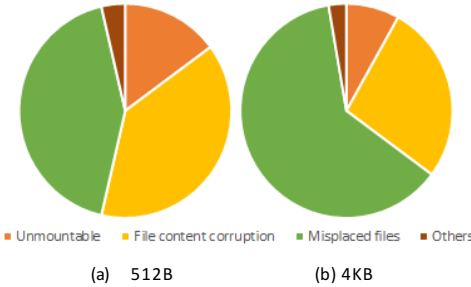


Figure 1. Distribution of symptoms among all faults leading to unrecoverable corruption

Next Step

- Evaluate more file system checkers or repair utilities
- Build a robust file system checker (*rfsck*) which is resilient to faults

On Fault Resilience of File System Checkers

Om Rameshwar Gatla, Mai Zheng

New Mexico State University

Motivation and Background

- Two power outages
- Interrupted the recovery process
- Resulted in severe data loss

TEXAS TECH UNIVERSITY
Information Technology Division

High Performance Computing Center

To All HPCC Customers and Partners,

As we have informed you earlier, the Experimental Sciences Building experienced a major power outage Sunday, Jan. 3 and another set of outages Tuesday, Jan. 5 that occurred while file systems were being recovered from the first outage. As a result, there were major losses of important parts of the file systems for the work, scratch and certain experimental group special Lustre areas.

The HPCC staff have been working continuously since these events on recovery procedures to try to restore as much as possible of the affected file systems. These procedures are extremely time-consuming, taking days to complete in some cases. Although about a third of the affected file systems have been recovered, work continues on this effort and no time estimate is possible at present.

User home areas have been recovered successfully. At present, no user logins are being permitted while recovery efforts proceed on the remaining Lustre areas. Your understanding and patience are appreciated.

Research Questions

- **Question 1:**

  Will an interrupted checker cause more damage to the file system?

- **Question 2:**

  Can the additional damage be fixed by the existing checker?

Methodology

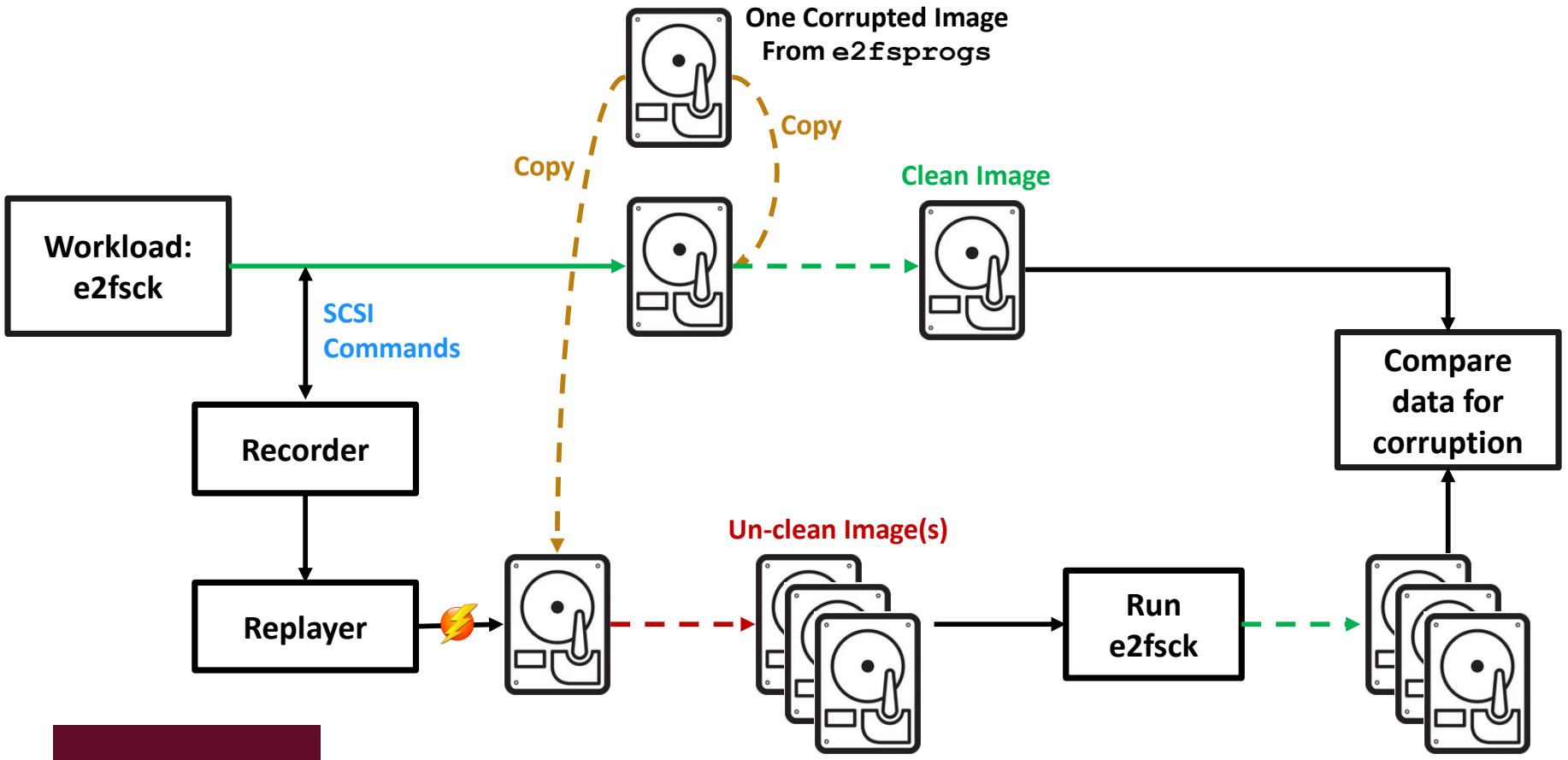
- Large scale file systems rely on local file systems. So we start from local file system checkers
  - **`e2fsck`**, file system checker for ext2, ext3 and ext4

- Use test images from **`e2fsprogs`** to trigger **`e2fsck`**

- Build a fault injection tool based on Linux SCSI target framework [1] to interrupt **`e2fsck`**

[1]: Zheng, Mai, et al. "Torturing Databases for Fun and Profit." OSDI. Vol. 14. 2014.

Methodology



Preliminary Results

- Used 175 test images from **`e2fsprogs`**
  - two fault injection granularities: 512 B and 4 KB
- Four types of corruptions observed:
  - cannot be fixed by **`e2fsck`**

Un-mountable

File Content Corruption

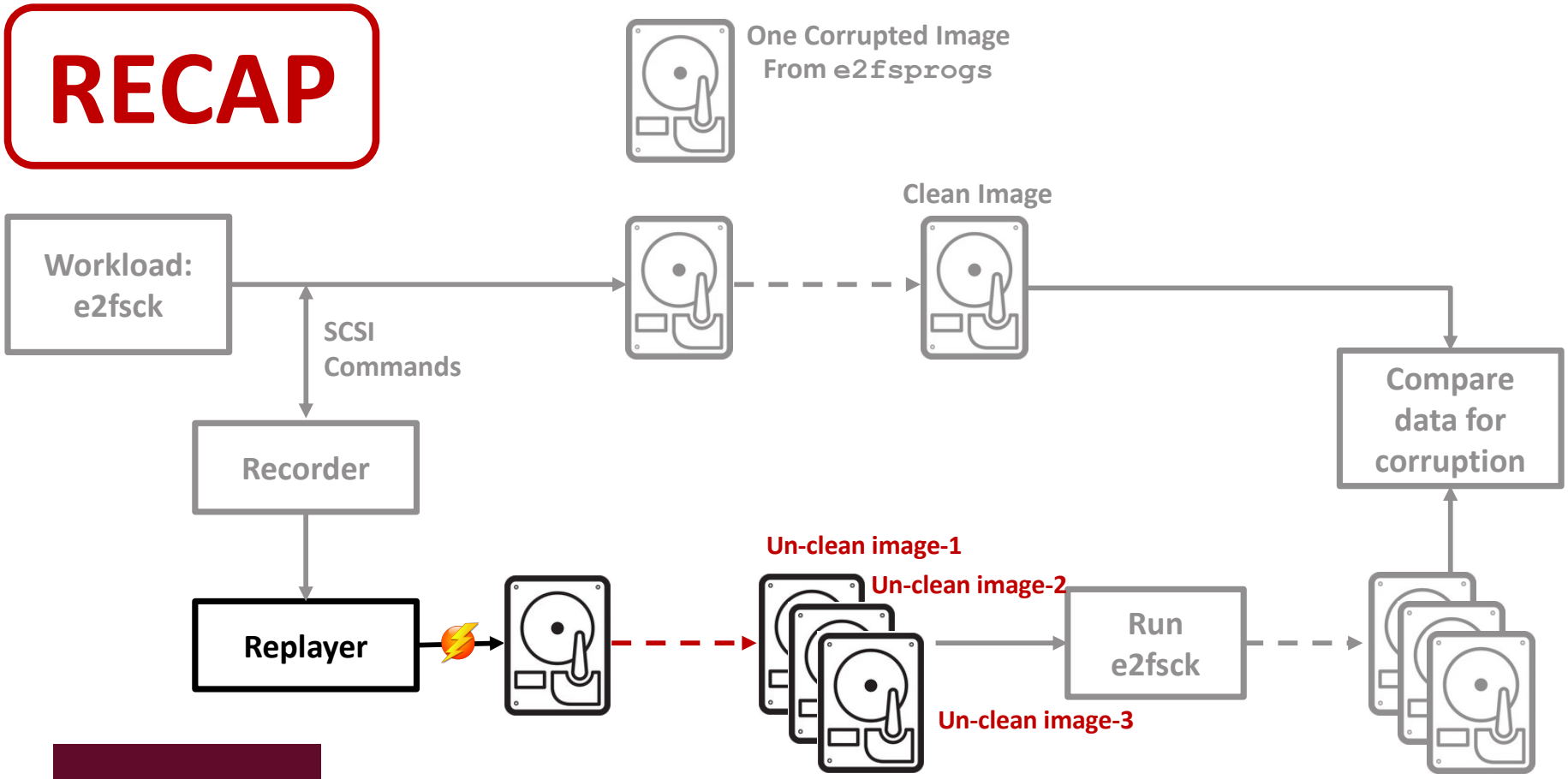
Misplacement of Files

Others

```
-rw-r--r-- 1 root root
-rw-r--r-- 1 root root
-????????? ? ?    ?
-????????? ? ?    ?
-????????? ? ?    ?
-????????? ? ?    ?
-rw-r--r-- 1 root root
```

Fault injection Granularity	# of test images reported corruption
512 B	34
4 KB	17

Preliminary Results



Preliminary Results

- % of un-clean images reporting corruption

Fault Injection Granularity	# of test images reporting corruption	Total no. of un-clean images generated	% of un-clean images reporting corruption
512 B	34	4058	5.91%
4 KB	17	298	12.41%

On going Work

- Evaluate other file system checkers
  - `xfs_repair` for XFS,
  - `btrfs check --repair` for btrfs
- Study the correlation b/w local checkers and checkers of large scale file systems
  - `e2fsck` and LFSCK (Lustre File System Checker)
- Build a resilient file system checker (`r-fsck`)

Conclusion

- Does interrupting checker cause more damage? **YES**
- Can the existing checker fix this damage? **NO**

Raise awareness to make recovery procedure resilient to faults

Conclusion

- Does interrupting checker cause more damage? **YES**
- Can the existing checker fix this damage? **NO**

Raise awareness to make recovery procedure resilient to faults

THANK YOU!