Understanding Node Change Bugs for Distributed Systems

Jie Lu, Chen Liu, Lian li, XiaoBing Feng

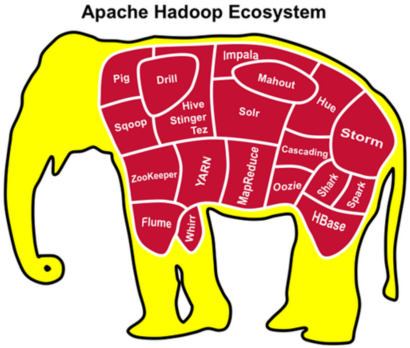
State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences

University of Chinese Academy of Sciences

August 13, 2019

Distributed system

- Open Source



- Industry

Google Cloud Platform

amazon web services

Microsoft Azure

IBM Bluemix™

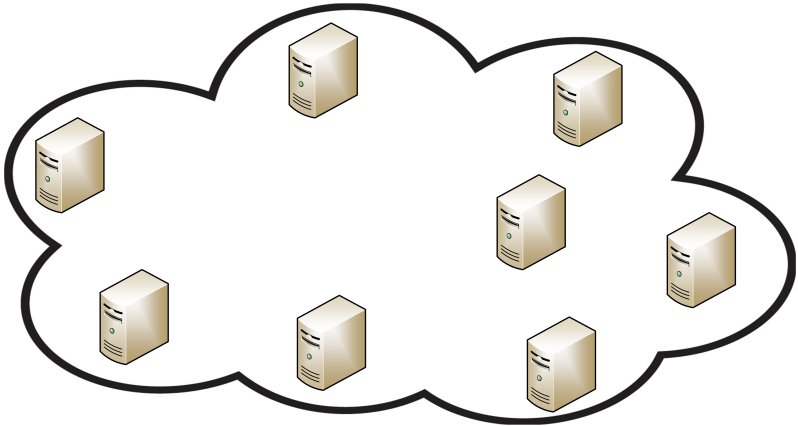
DigitalOcean

OPENSHIFT by Red Hat

Alibaba Cloud aliyun.com

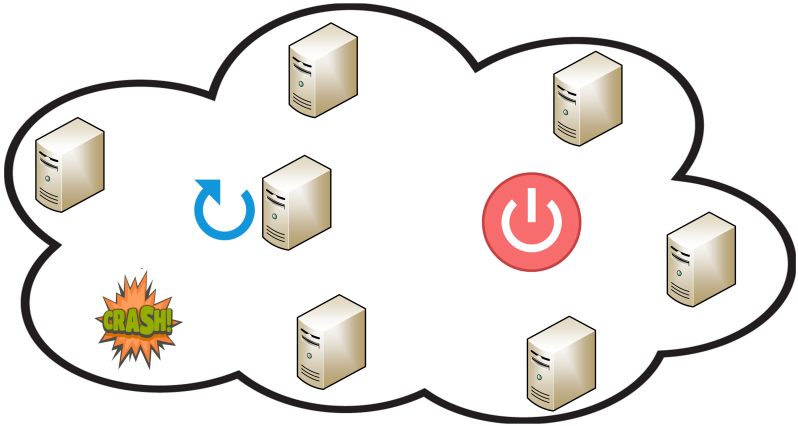
Distributed system

- Running on cluster of nodes



Distributed system

- Node can leave or join the cluster at any time.

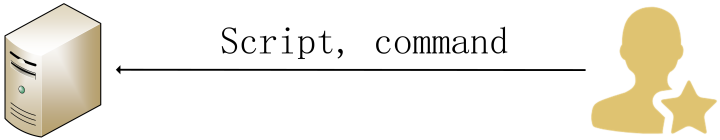


Node operations

- Graceful shutdown.
- Crash
- Reboot
- Adding the fresh node.

Node operation

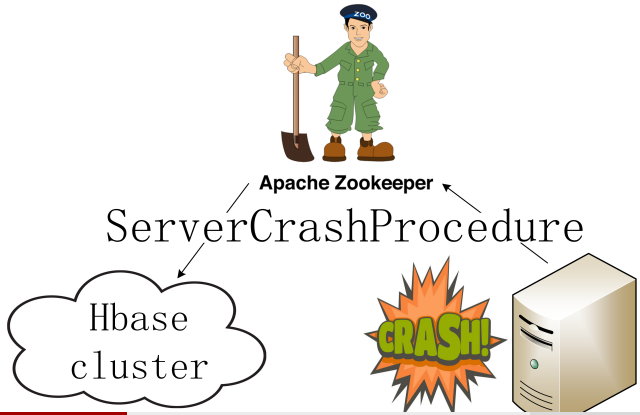
- Graceful shutdown.
  - Leaving node can perform local aftermath: clean tmp file...
  - Protectively tell the cluster that it is leaving.




```
1  Runtime.getRuntime().addShutdownHook(new Thread() {
2      public void run() {
3          cleanTmpFile();
4          sysnch();
5          System.out.println("Application Terminating ...");
6      }
7  });
8
```

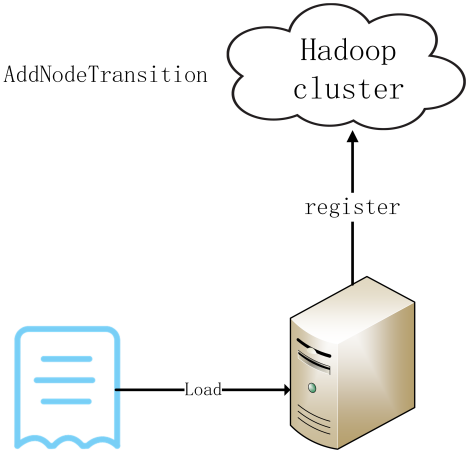
Node operation

- Crash.
  - No local aftermath
  - Cluster checks the node periodically.



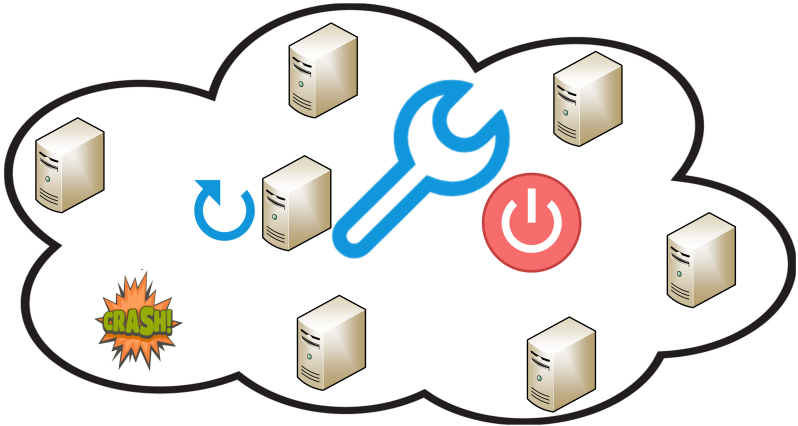
Node operation

- Reboot.
  - Dead node rejoins the cluster
- Adding fresh node.



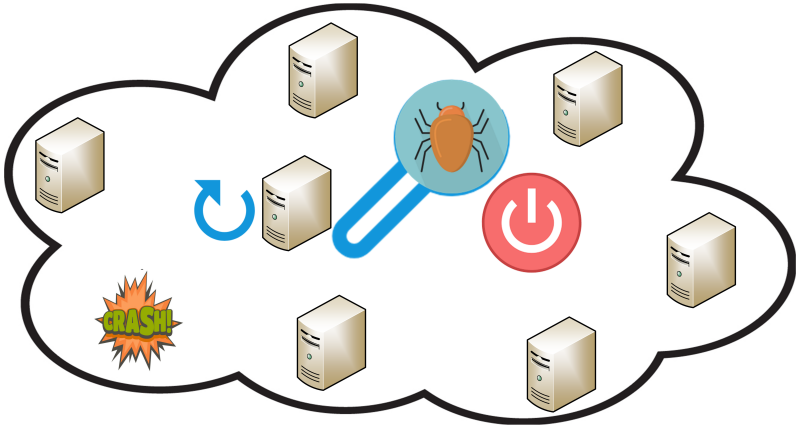
Distributed system

- Resilient



Distributed system

- Coroner condition : **N**ode **C**hange bugs(NCBugs)!



Motivation

- Lack of understanding NCBugs.
  - TaxDC[1]:Only distributed concurrency bugs.
  - CREB[2]:Only crash bugs!
  - CbsDB[3]:A comprehensive study on all bugs!
- Current research is black box testing.
  - Fault injection[4]:Randomly injecting or at IO point.
  - Model checking[5]:Injecting at each different state.

[1] Tanakorn Leesatapornwongsa et al. "TaxDC: A taxonomy of non-deterministic concurrency bugs in datacenter distributed systems". In: *ACM SIGPLAN Notices*. Vol. 51. 4. ACM. 2016, pp. 517–530.

[2] Yu Gao et al. "An empirical study on crash recovery bugs in large-scale distributed systems". In: *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ACM. 2018, pp. 539–550.

[3] Haryadi S Gunawi et al. "What bugs live in the cloud? a study of 3000+ issues in cloud systems". In: *Proceedings of the ACM Symposium on Cloud Computing*. ACM. 2014, pp. 1–14.

[4] Gun11.

[5] Tanakorn Leesatapornwongsa et al. "SAMC: Semantic-Aware Model Checking for Fast Discovery of Deep Bugs in Cloud Systems.". In: *OSDI*. 2014, pp. 399–414.

Methodology

- Node change bugs : Bugs that happen after perform node operation

Methodology

- Node change bugs : Bugs that happen after perform node operation
- KEYWORD search in bug tracker system of five systems: YARN, HDFS, HBase, Zookeeper, Cassandra

Methodology

- Node change bugs : Bugs that happen after perform node operation
- KEYWORD search in bug tracker system of five systems: YARN, HDFS, HBase, Zookeeper, Cassandra
- 6660 issues, manually identify 620 real NCBugs.

Methodology

- Most bugs happen after reboot and crash.
  - Many researches focus on them. e.g. fault injection, module checking.
- Graceful shutdown can also led bugs.
  - Little researches pay attention to them.
- Little bugs are caused by fresh boot.

Table: Distribution of collected bugs.

type	Cassandra	HBase	Hdfs	YARN	Zookeeper	total
shutdown	2	31	14	18	7	72
crash	10	65	22	15	4	116
reboot	109	79	111	97	26	422
fresh boot	9	0	0	0	0	9
Total	130	175	147	133	37	619

Methodology

- Randomly choose 120 bugs for deeply understanding.
- Study their patches, source code, comments in bug report.
- Deeper and deeper: Randomly choose 30 of them to reproduce!
- No tool can help user reproduce bugs.

NCTrigger

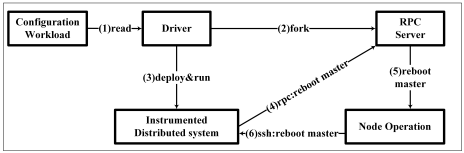
- Users write configuration, NCTrigger can automatically deploy the distributed system and reproduce the bug.
- NCTrigger can help user reduce at least half time:deploy and debug time.

Configuration

```
bugid = HDFS_4596
system_home =$HOME/hadoop-2.0.4-alpha
master = hadoop1
slaves = hadoop1,hadoop2
workload = checkpoint
injection = saveRenameCheckpointImage 3 reboot
```

workload

```
hadoop dfs -mkdir /input
hadoop dfs -put File /input
hdfs SNN -checkpoint
```

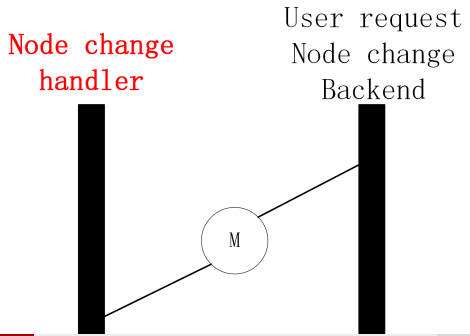


Research questions

- RQ1: What are common bug patterns and root causes?
- RQ2: Do NCbugs have severe impacts?
- RQ3: what are NCbugs trigger condition?
- RQ4: Are NCBugs hard to fix?

RQ1:Common pattern

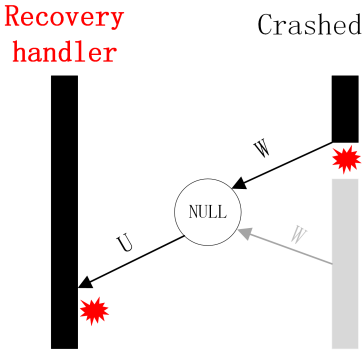
- 33.3% NCBugs are Concurrency bugs
  - The node change handler thread have conflict access of shared resource with another thread.
  - These threads can be on different node
  - The shared resource can on different node.
- Extending traditional concurrency detection technology to detect them.



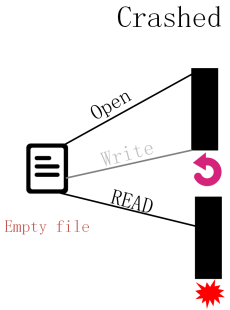
RQ1:Common pattern

- 33.8% NCBugs are caused by the improper recovery handler.
  - Missing safety check.

HB-3023



C-11995



RQ1:Common pattern

- 33.8% NCBugs are caused by the improper recovery handler.
  - Missing synchronization
  - We can use static analysis (Like DScope[6]) to find this pattern:**obtain the node address out of loop, but connect the node in the loop.**

```
1    protected void fetchBlockByteRange(LocatedBlock blockID){
2-    address = read(blockID);
3     while (true) {
4+       address = read(blockID);//miss
5        try{
6           connect(address);
7         }catch(Exception e){//ignore exception and retry};
8     }
9  }
```

[6] Ting Dai et al. "DScope: Detecting Real-World Data Corruption Hang Bugs in Cloud Server Systems". In: *Proceedings of the ACM Symposium on Cloud Computing*. ACM. 2018, pp. 313–325.

RQ1:Common pattern

- 33.8% NCBugs are caused by the improper recovery handler.
  - Excessive retry:Hang
  - Example:HMaster retry to connect to RS0, this will hang the shutdown of HMaster

```
1    while (true) {
2      try{
3-        hris = getServer(serverName); //serverName is RS0
4+         // Skip getting user regions if HMaster is stopped.
5+        if (!this.server.isStopped()) {
6+           hris = getServer(serverName);
7+        }
8       } catch (Exception e)( //retry)
9      break;
10   }
```

RQ1:Common pattern

- 18.3% NCBugs can cause null pointer exception.
- Some of them are simple: Callee returns a null value directly, but caller doesn't check it.

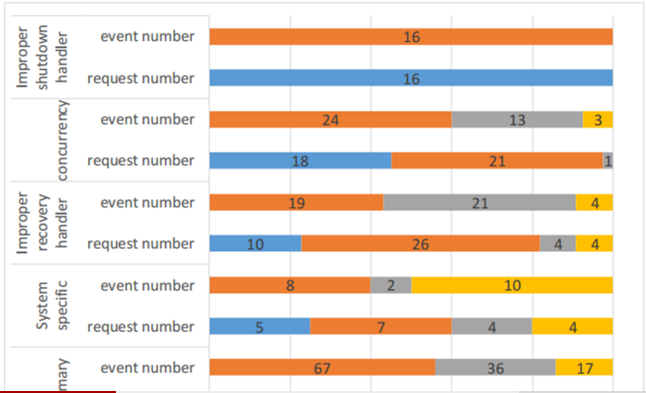
```
1 ret = null;
2 while (shouldRun){//can be reset by shutdown
3   ret = retry();
4 }
5 return ret;
```

RQ2:Impact

- Allmost all(97.5%) NCBugs have serious impacts to the system, which cause node down, data loss, operation failure, or performance degradation.
- We need hunt them.

RQ3:Trigger condition

- More than 40% NCBugs do not need any request to trigger, and 85% NCBugs need no more than two requests and events.
- Model checking and fault injection still have chance to improve.



RQ4:Fixing

- 25%NCBugs are fixed at root cuase point, left are fixed by preventing the error propagation.
- Randomly choose 120 non-NCBugs:Comparing their number and length of patches, comments number and time cost.
- It is no more complex to fix a NCBug than other types of bugs. Shutdown bugs are easier to fix.
- Tolerating node change is implemented as normal software components of distributes system

Table: Complexity to apply a patch.

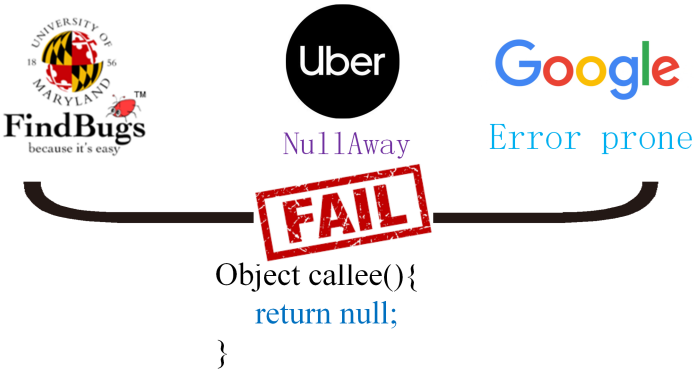
Operation	LOC of a patch	# patches	# comments	times (day)
shutdown	45.825	2.98	16.13	83.26
crash	55.98	3.00	20.93	74.49
reboot	50.28	3.50	23.63	87.86
non-NC bugs	61.62	2.81	17.48	82.20

NPEDetector

- Simple:callee returns null directly, but caller doesn't check it.

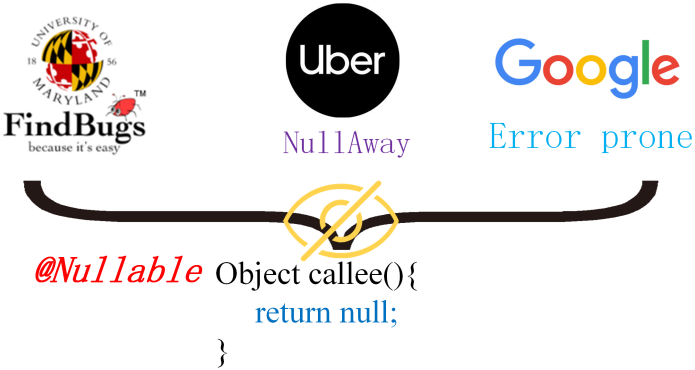
NPEDetector

- Simple:callee returns null directly, but caller doesn't check it.
- Current compile-time static checkers can't find them!



NPEDetector

- Simple:callee returns null directly, but caller doesn't check it.
- Current compile-time static checkers can't find them!
  - Need @Nullable before method that can really return null!



NPEDetector

- Simple:callee returns null directly, but caller doesn't check it.
- Current compile-time static checkers can't find them!
  - Need @Nullable before method that can really return null!

NPEDetector

- Goal: Helping developer find the nullable methods!

NPEDetector

- Goal: Helping developer find the nullable methods!
- Simple way

NPEDetector

- Goal: Helping developer find the nullable methods!
- Simple way
  - Finding all methods that return null directly!

NPEDetector

- Goal: Helping developer find the nullable methods!
- Simple way
  - Finding all methods that return null directly!
  - Measuring the null chance of each method!

$$Score = CheckedCallersN - UnCheckedCallersN + \#Exception * Weigh$$

NPEDetector

- Goal: Helping developer find the nullable methods!
- Simple way
  - Finding all methods that return null directly!
  - Measuring the null chance of each method!

$$Score = CheckedCallersN - UnCheckedCallersN + \#Exception * Weigh$$

@Nullable

NPEDetector

- Goal: Helping developer find the nullable methods!
- Simple way
  - Finding all methods that return null directly!
  - Measuring the null chance of each method!

$$Score = CheckedCallersN - UnCheckedCallersN + \#Exception * Weigh$$

@Nullable

NPEDetector

- 8 distributed system, including 3 that we never study.

NPEDetector

- 8 distributed system, including 3 that we never study.


NPEDetector

- 8 distributed system, including 3 that we never study.


- Top 100, 60 are true bugs, 23 are fixed, 23 are confirmed.
- Including 7 new NCBugs.

Table: Results of applying NPEDetector to 8 distributed systems.

System	YARN	HDFS	HBase	Cassandra	Zookeeper	CloudStack	Storm	Helix	total
submitted bugs	4	9	11	3	12	12	9	5	65
confirmed bugs	0	7	8	0	1	0	5	2	23
false positive	0	0	1	0	0	1	0	3	5
uncertainty	0	2	0	3	8	0	1	0	14
fixed bugs	4	0	2	0	3	11	3	0	23

Example

- HBASE-20419

```
1 //callee
2  public List listChildren(){
3    try{
4      return zkw.getChildren();
5    }
6    catch(NoNodeException ke){
7      return null;
8    }
9  }
```

Example

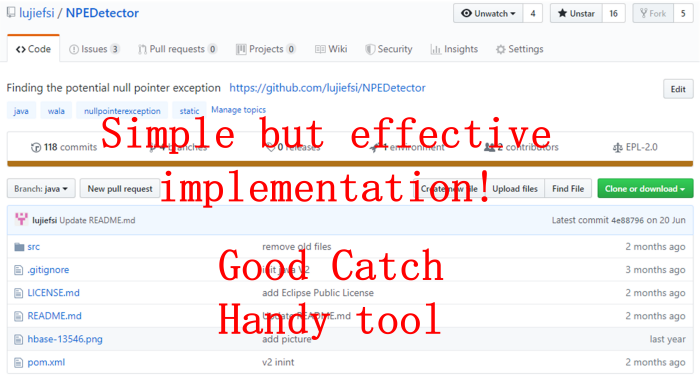
- HBASE-20419

Example

- HBASE-20419

Conclusion

- We perform a comprehensive study on 620 CR bugs in 5 distributed systems, and obtain many interesting findings
- Develop NCTrigger to help user reproduce NCBugs.
- Develop NPEDetector to help developer avoid NPE.



https://github.com/lujiefsi/NPEDetector

Thank You

Q & A