CrashTuner: Detecting Crash Recovery Bugs in Cloud Systems via Meta-info Analysis

Jie Lu, Chen Liu, Lian Li, Xiaobing Feng

Alibaba Group
阿里巴巴集团

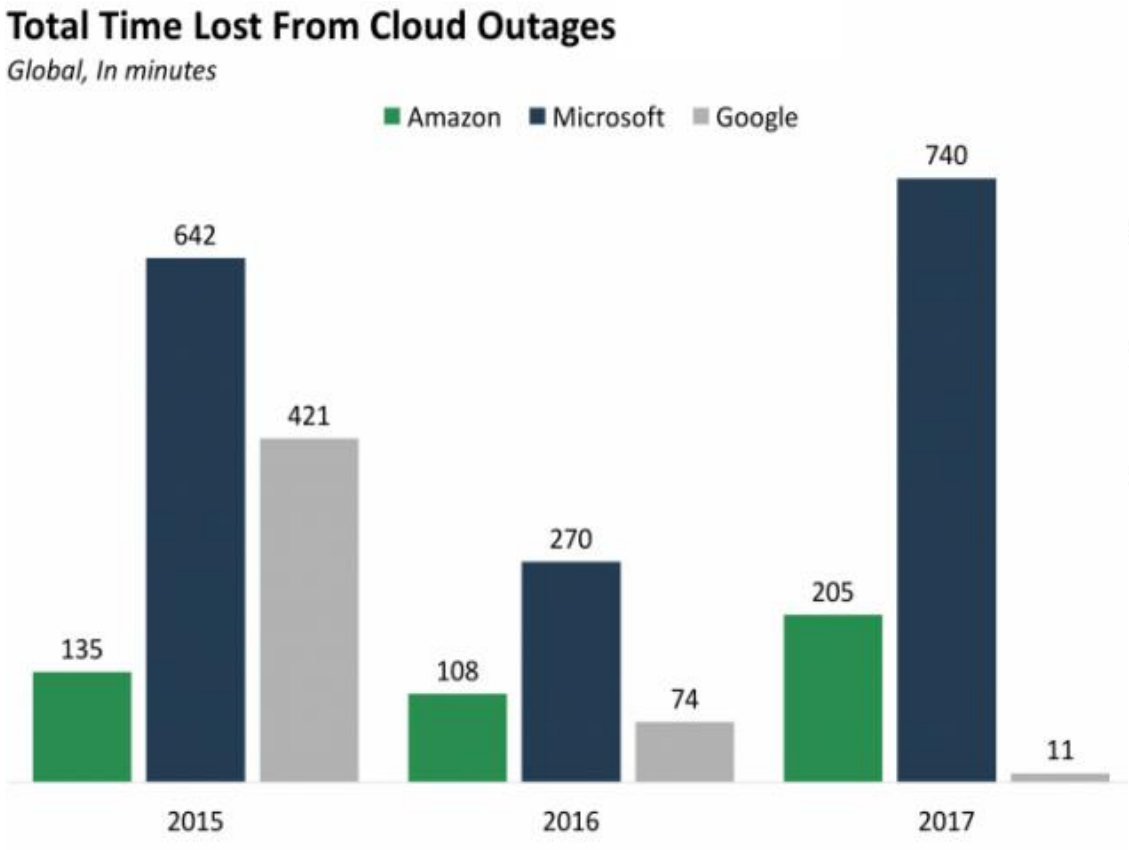
Institute of Computing Technology
of the Chinese Academy of Sciences

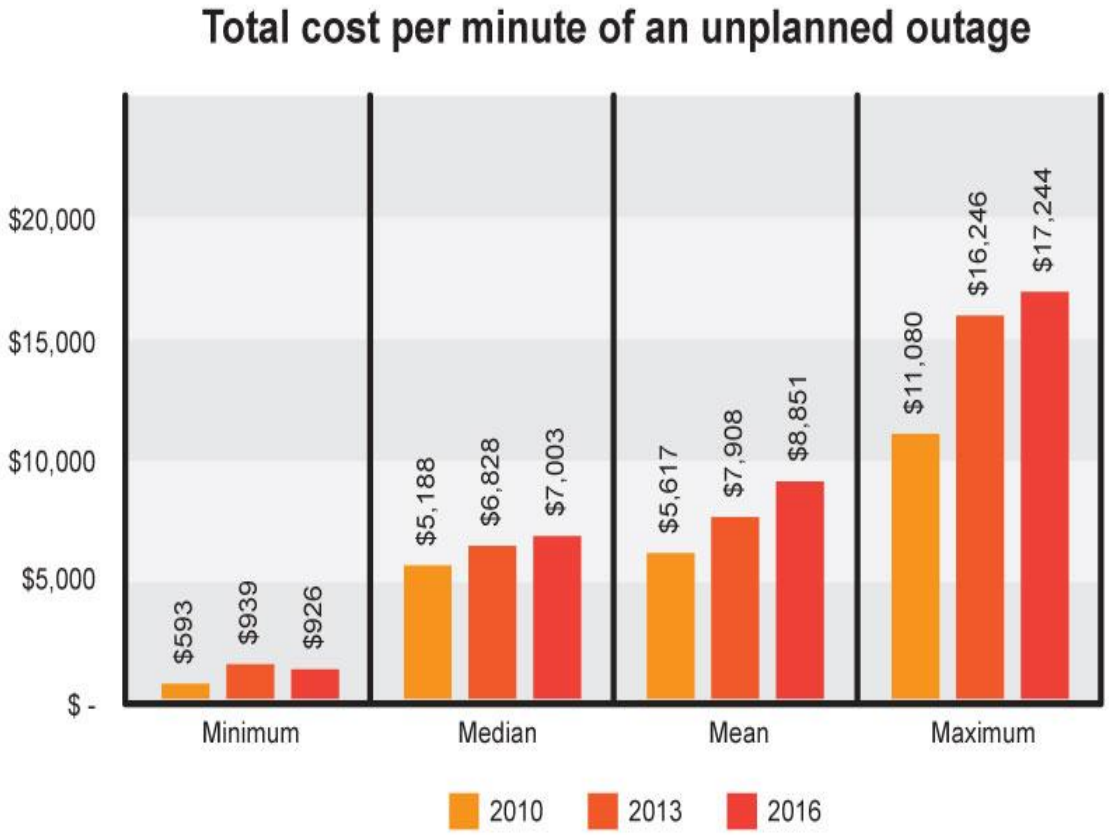
University of Chinese
Academy of Sciences

Reliability of Distributed system is important

- Downtime

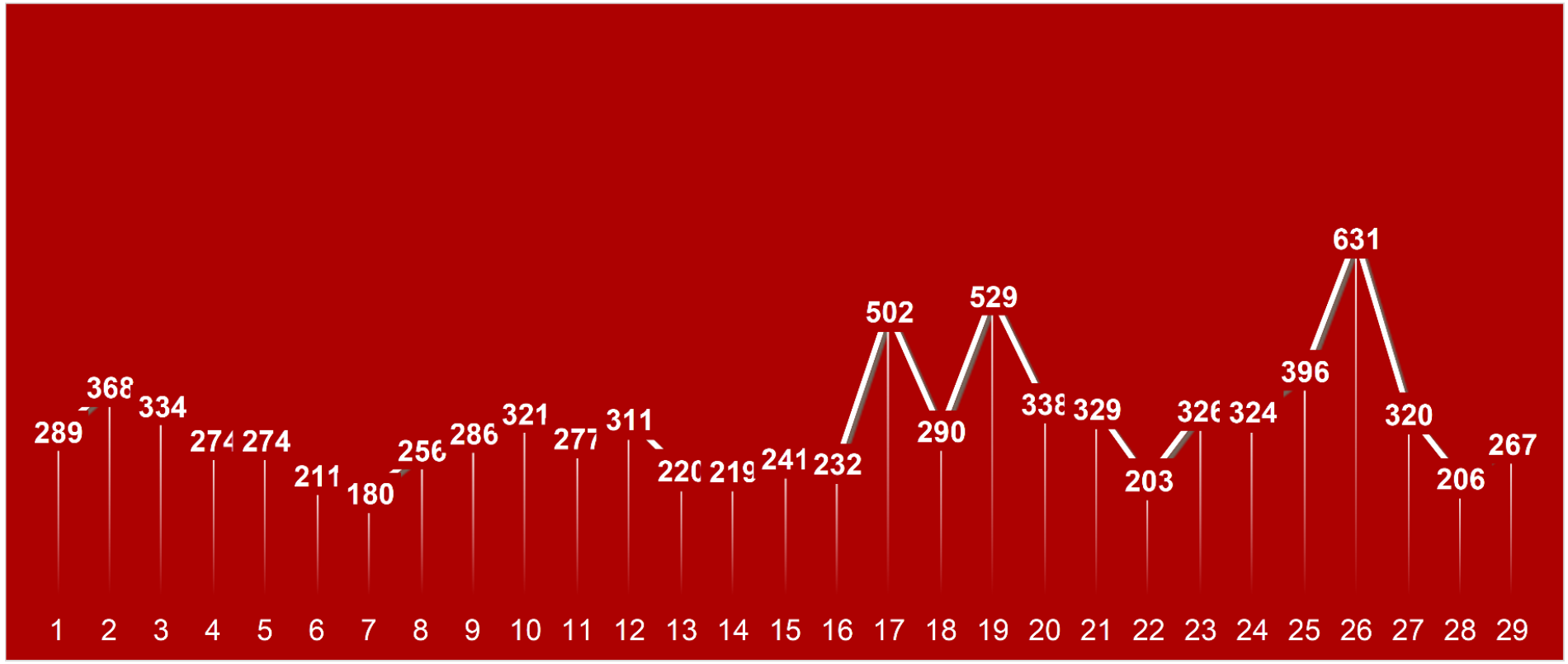
Economic loss: $20,000 per minute





Distributed systems

- At least 180 nodes will crash each day in google cluster



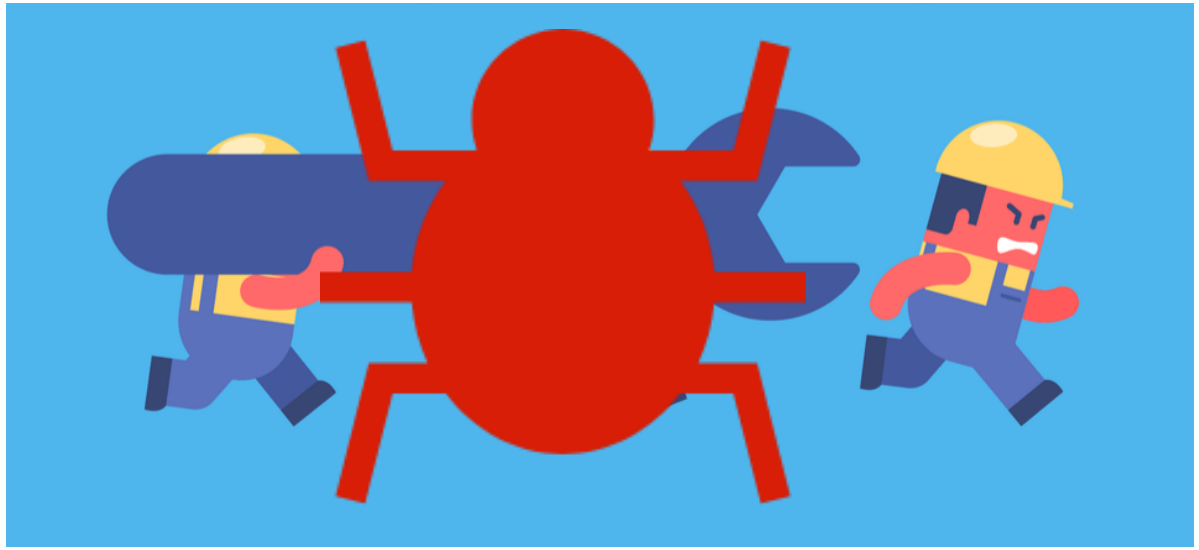
Crash Recovery

Crash Recovery must be a first-class operation of distributed systems

[YCSB SoCC2010]

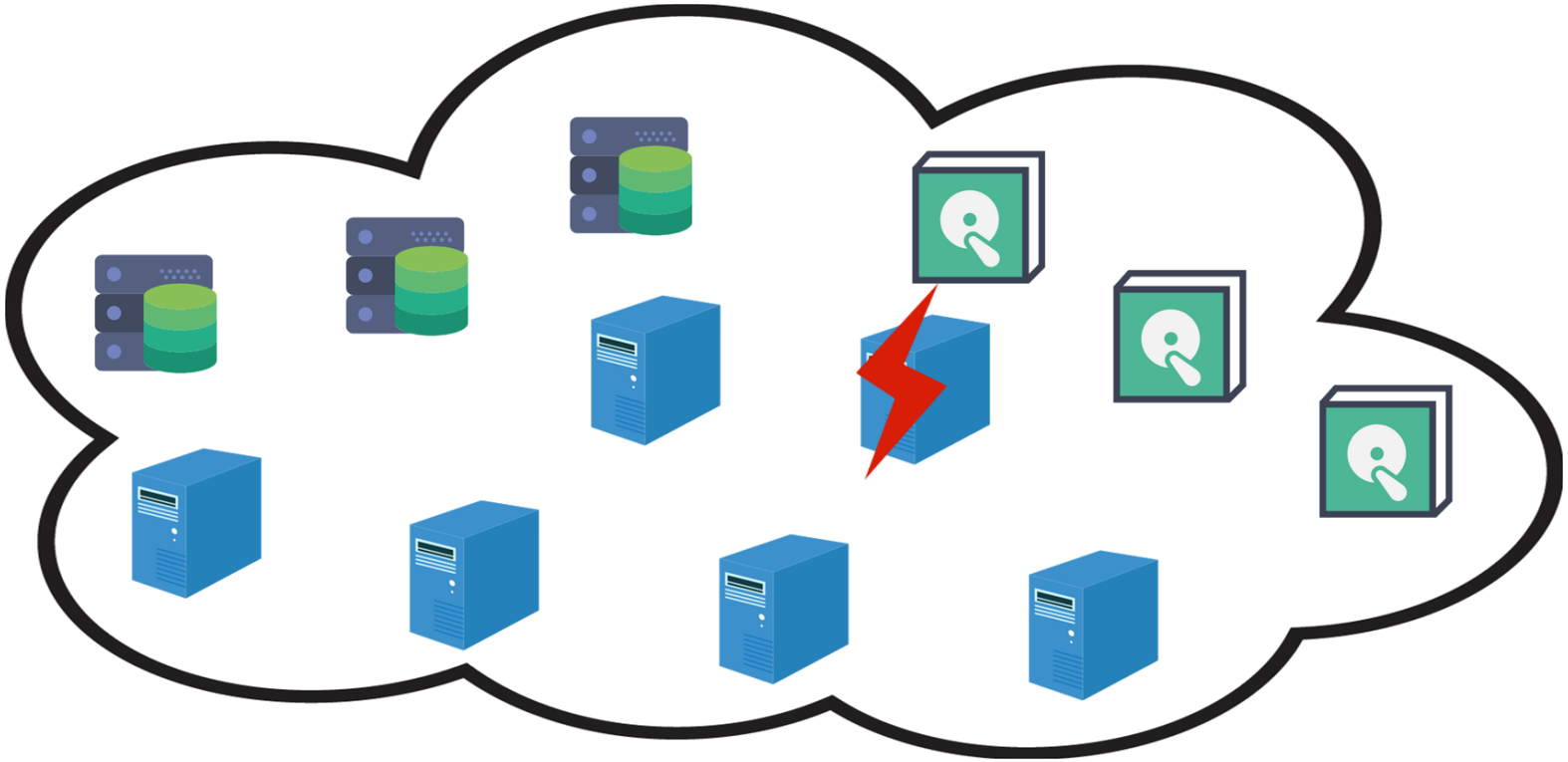
Crash Recovery Bugs

Crash Recovery bugs cause catastrophic failures[Hotos 2013]



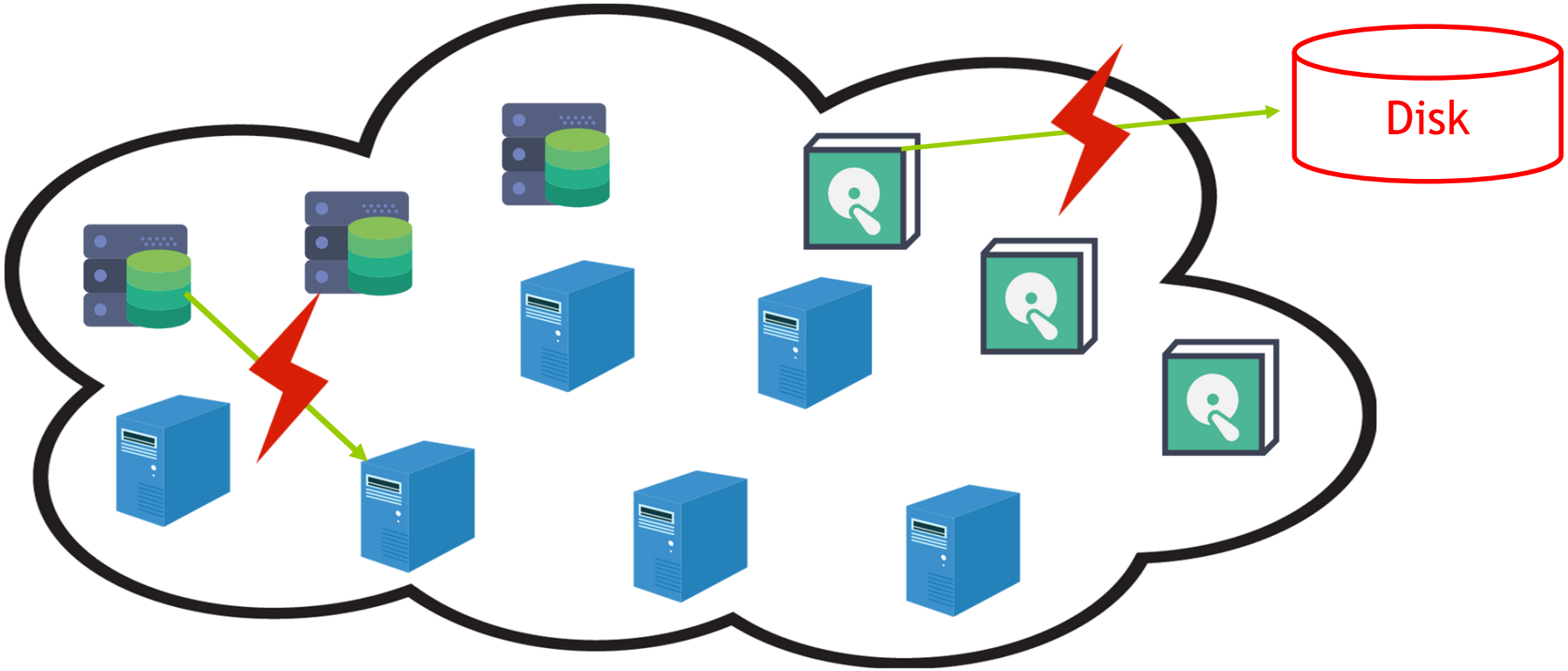
Existing detection technologies

- Random fault injection: low coverage



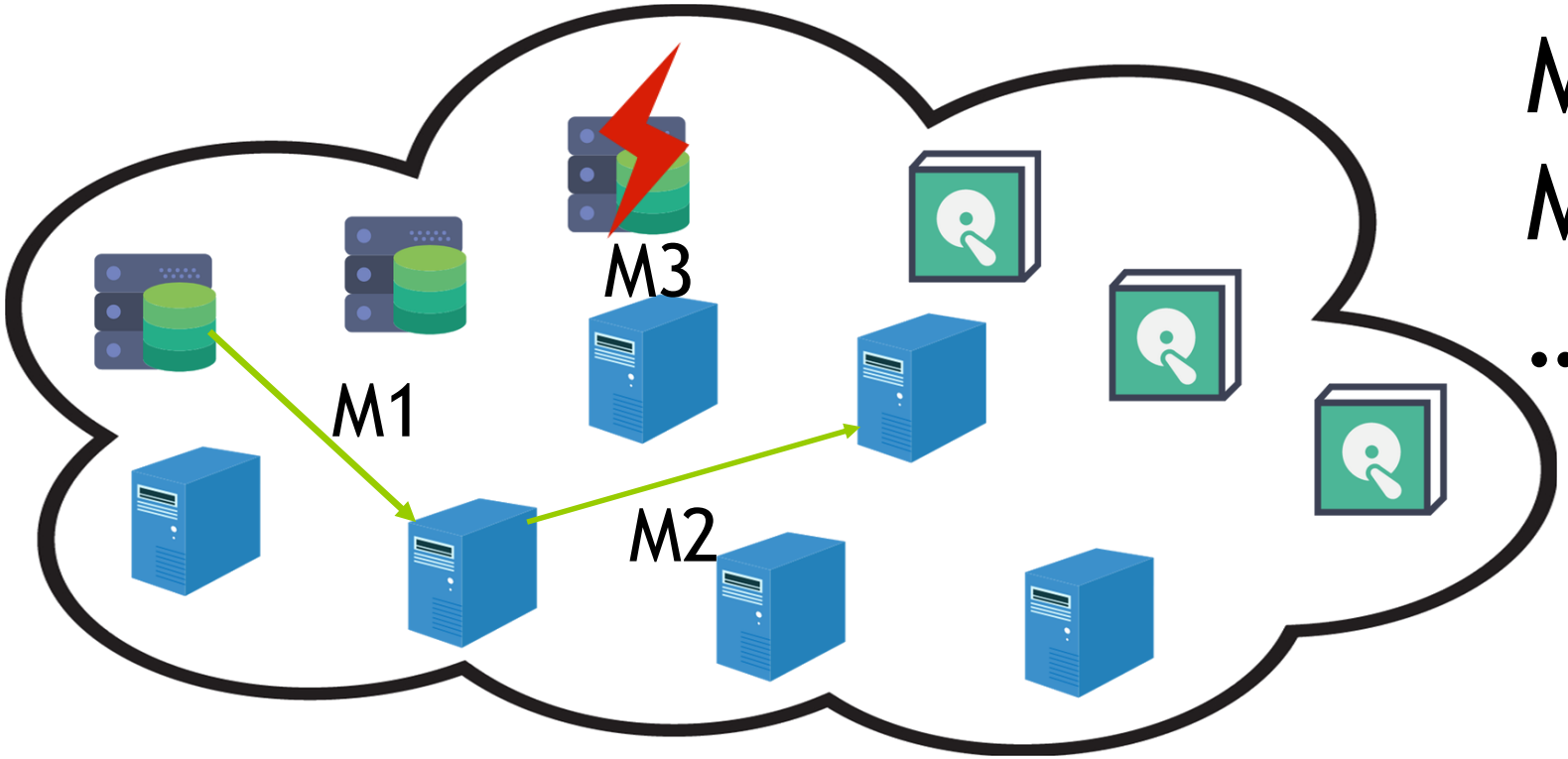
Existing detection technologies

- IO around fault injection: incomplete test



Existing detection technologies

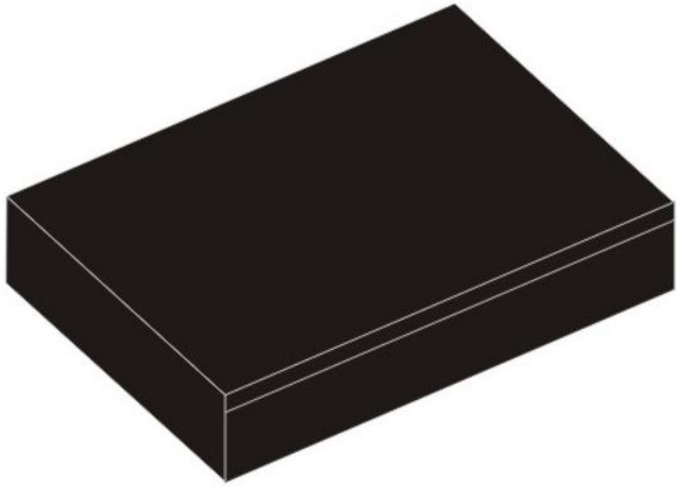
- Model checking: too large search space.



M1,M2,M3
M1,M3,M2
.....

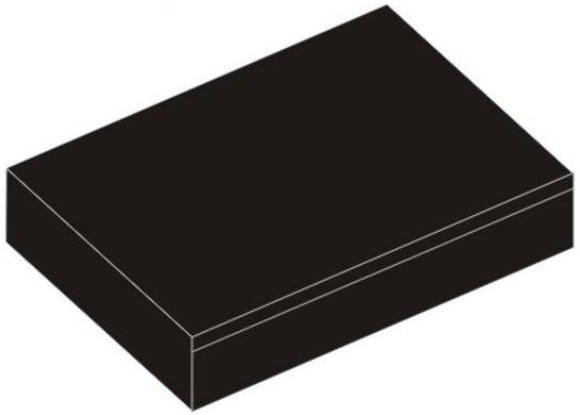
Motivation

- Existing detection technologies are insufficient
  - Random fault injection: low coverage.
  - IO around fault injection: incomplete test
  - Model checking: too large search space
- They don’t dig into the details of crash recovery bugs.



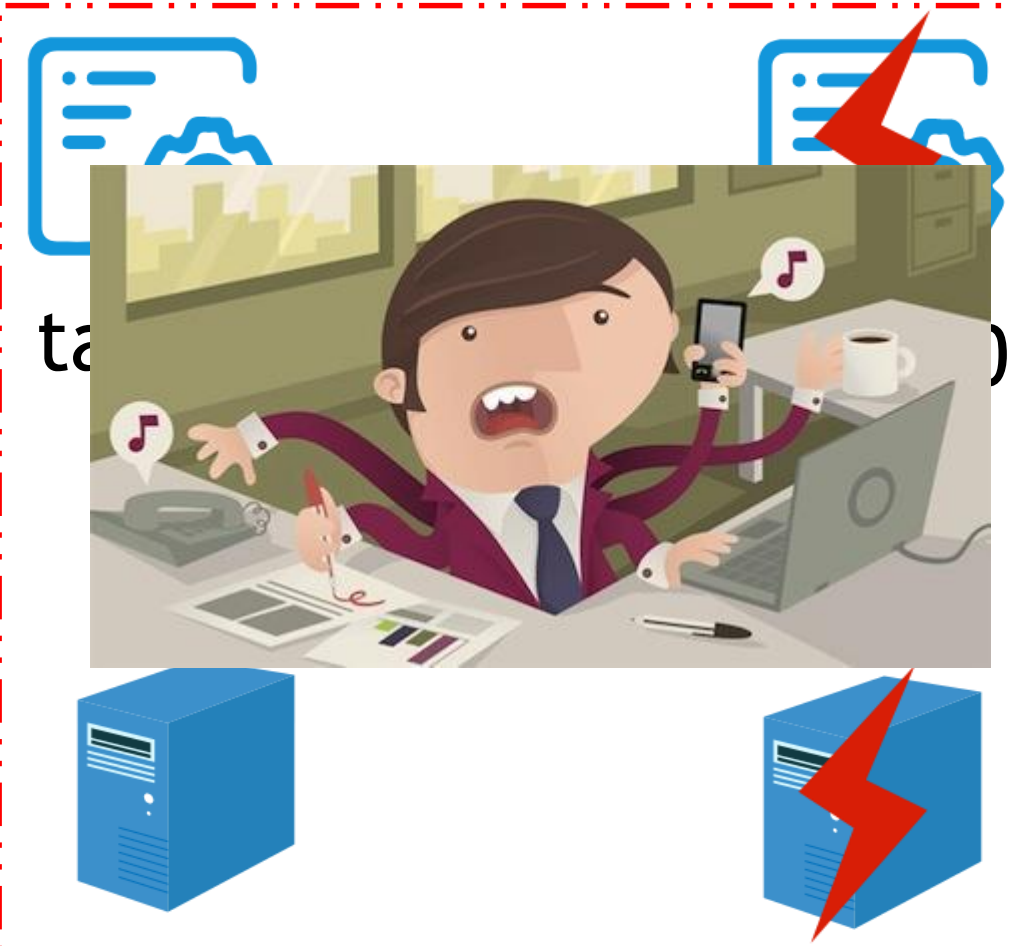
Key Point

What are affected by Crash and Recovery?



Two critical operations of Crash Recovery

Fault-tolerant

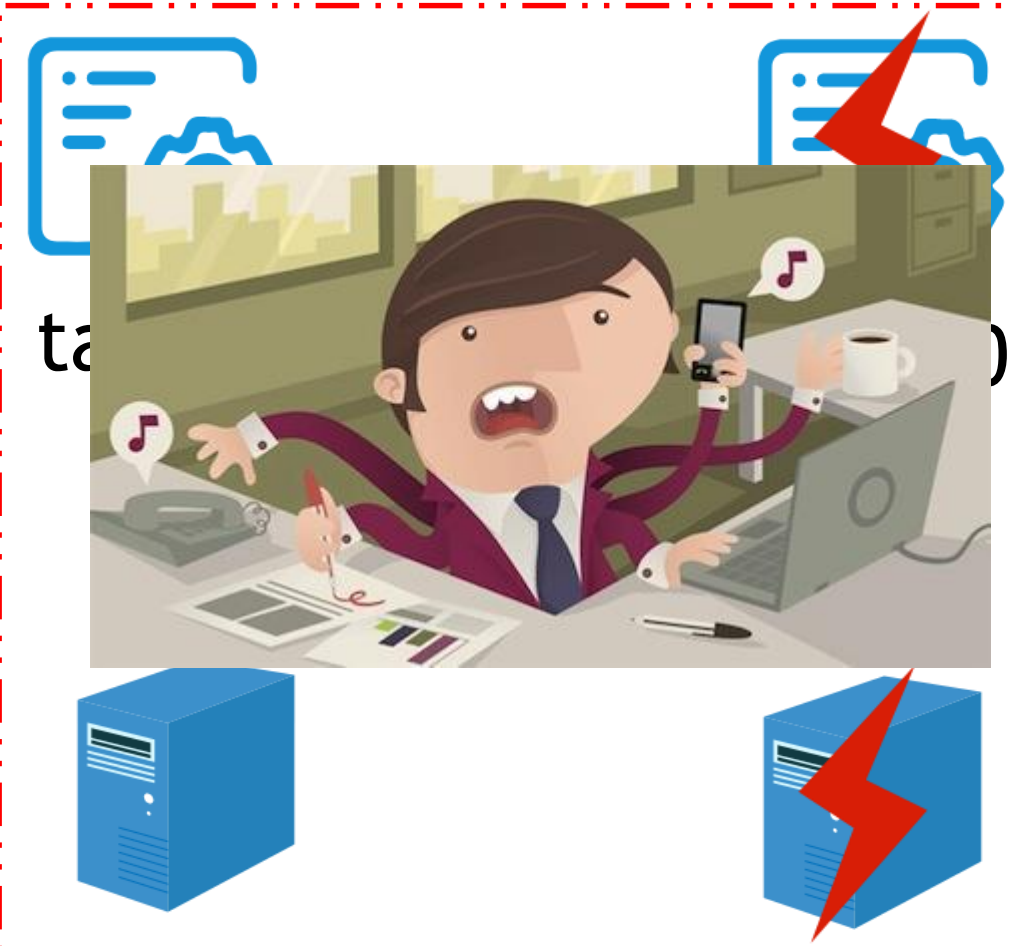


Meta-info maintenance

- file
- Job
- taskId
- Attempt
- JobId
- blockId

Two critical operations of Crash Recovery

Fault-tolerant



Meta-info maintenance

Reasons

- Meta-info may hide in the deep call chain.

- Developer can’t be aware of crash continuously.
  - Especially for new developer.

Crash Recovery bugs

Master

```
NodeId nodeId = "node2";
livingNodes.add(nodeId);
```

Meta -info variable

Crash Recovery bugs

Master

```
NodeId nodeId = “node2”;
livingNodes.add(nodeId);

// …..

while(livingNodes.has(“node2”))
    retryConnect(“node2”);

// …..
```

Meta -info variable

Recovery Handler

```
livingNodes.remove(nodeId);
```

Crash Recovery bugs

Bug pattern1: Post-Write

Master | Recovery Handler

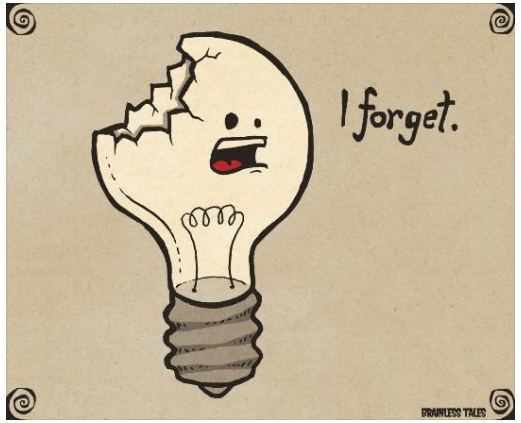
```
NodeId nodeId = "node2";
livingNodes.add(nodeId);

// .....

while(livingNodes.has("node2"))
    retryConnect("node2");
```

Meta -info variable

```
livingNodes.remove(nodeId);
```



Crash Recovery bugs

master

```
int index = getIndex("node2");
//....

node = livingNodes.get(index);
```

Crash Recovery bugs

Bug pattern2: Pre-Read

master Recovery Handler

```
int index = getIndex("node2");
//....

node = livingNodes.get(index);
```

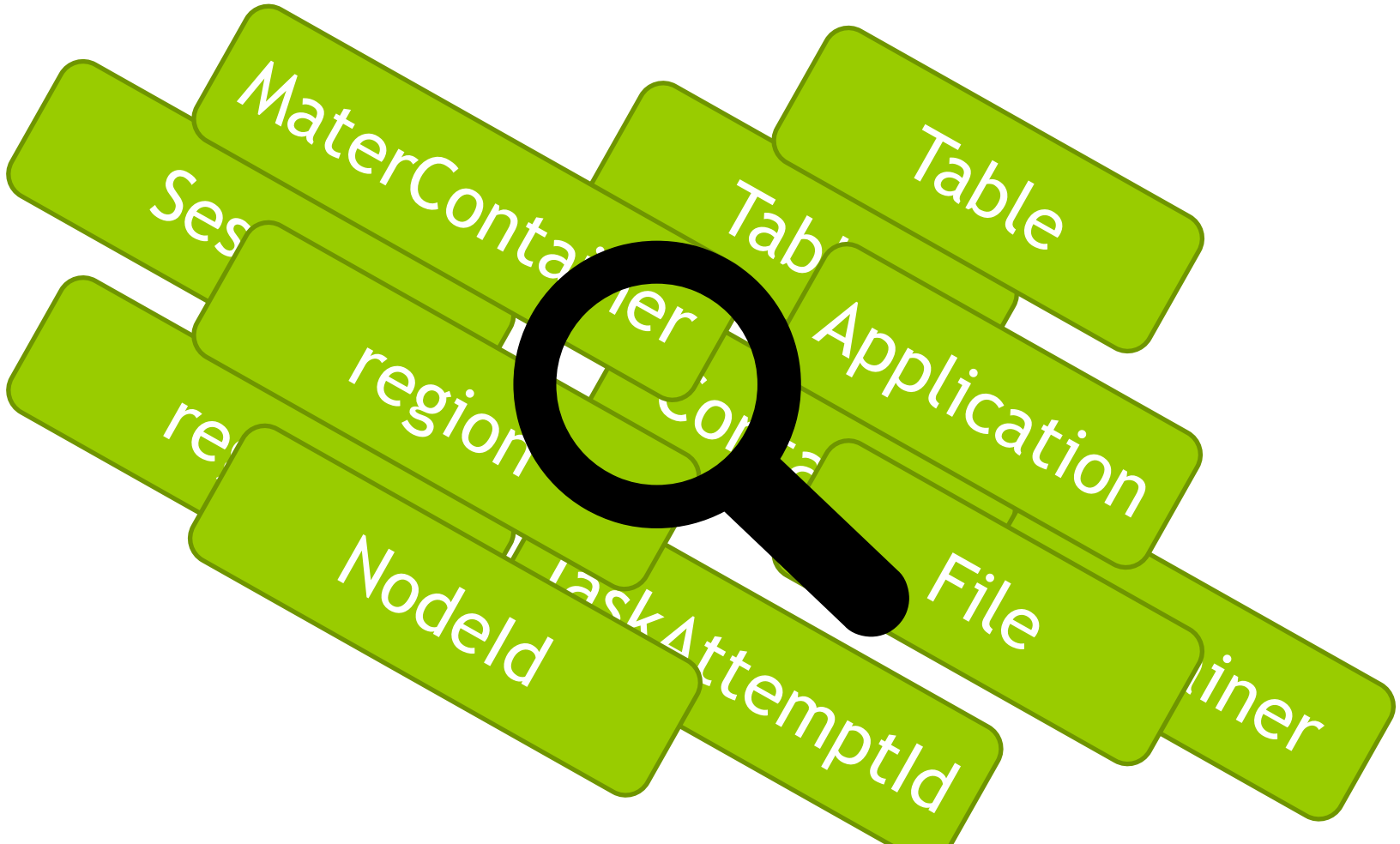
```
livingNodes.remove(nodeId)
```

CrashTuner: Detecting Crash Recovery Bugs

- 1. Meta-info variables Identification(Challenge 1)
- 2. Crash Points discovery:
  - Crash Points : Post-write or pre-read of meta-info variables
- 3. Crash Injection(Challenge 2)
- 4. Bug report

1. Meta-info variables Identification

- **Challenge1 : How to find all meta-info variables automatically**



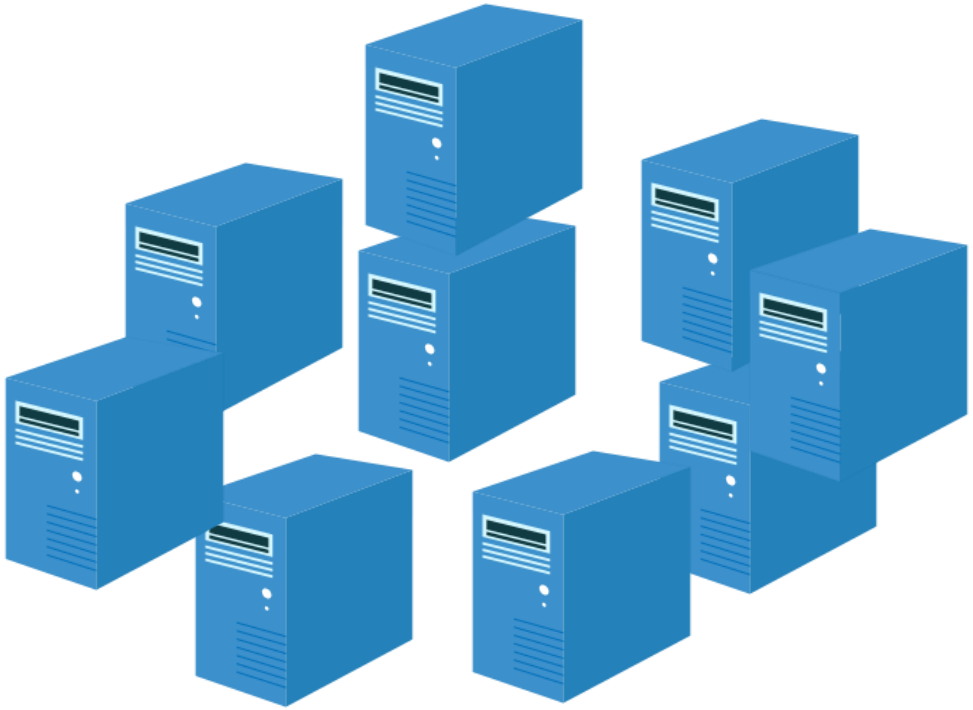
3. Crash Injection

- **Challenge2 : How to find the target node of Crash injection?**

```
livingNodes.add(nodeId);

// Crash Point
```

?



1. Meta-info variables Identification

- Rule 1 (Basic):Node variable is basic meta-info variable
  - Method: Log analysis

```
1 public void registerNode(NodeId nodeId) {
2     LOG.info("registering node " + nodeId);
3 }
```

10.5.1.11
IP address

1. Meta-info variables Identification

- Rule 1 (Basic):Node variable is basic meta-info variable
- Rule 2 (Variable Transitivity): Variables related to meta-info variable are also meta-info variables.

```
1 public void assignTask(NodeId nodeId,
2     ContainerId containerId) {
3     LOG.info("Using container " + containerId +
4                                 "on node" + nodeId);
5 }
```

1.Rule of Identify the meta-info variables

- Rule 1 (Basic):Node variable is basic meta-info variable
- Rule 2 (Variable Transitivity): Variables related to meta-info variable are also meta-info variables.

Many meta-info variables are not logged

```
1 public void registerNode(NodeId nodeId) {
2     Log.info("registering node" + nodeId);
3     RPCfun(nodeId);
4 }
5 public void RPCfun(NodeId nodeId) {
6     livingNodes.add(nodeId);
7 }
```

Alias analysis

1.Rule of Identify the meta-info variables

- Rule 1 (Basic):Node variable is basic meta-info variable
- Rule 2 (Variable Transitivity): Variables related to meta-info variable are also meta-info variables.
- Rule 3 (Type Transitivity): Variables with the same type as meta-info variables are also meta-info variables.

```
public void registerNode(NodeId nodeId) {
    Log.info("registering node" + nodeId);
    RPCfun(nodeId);
}
```

Static type analysis

```
List<NodeId> livingNodes = null;
```

2. Crash Points discovery

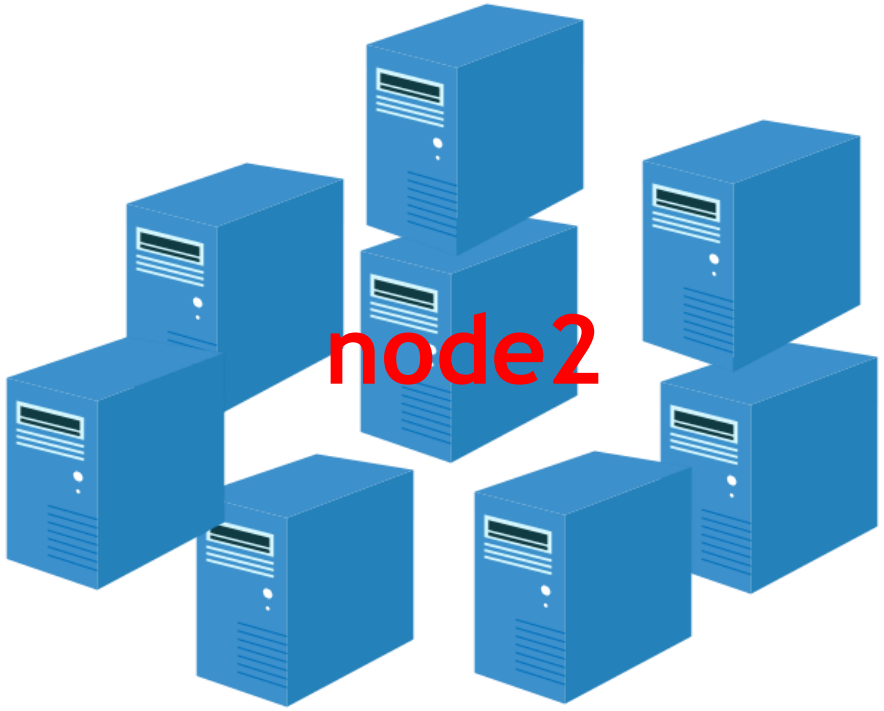
- Pre-read and Post-write of meta-info variables
- Refer to the paper for more details

3. Crash Injection

- Case 1: Direct inferring

liveNodes.add("node2")

Crash point



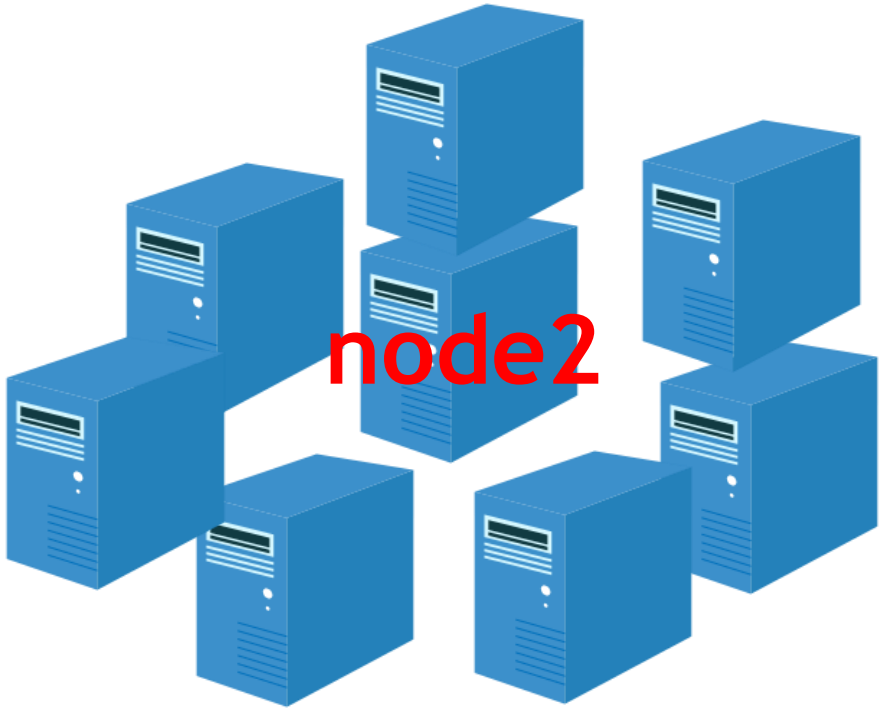
3. Crash Injection

- Case 2: Indirect inferring

containers. add("container1")

LOG: Assigned container1 to node2

container1 ⟺ node2



4. Bug report

- One bug happens when
  - Memory exception: NPE , Array Index Out of Bounders
  - Uncommon exception:e.g InvalidStateTransaction
  - Job hangs, innocent node crash

Evaluation

- Five distributed Systems under testing

System	Configure Change	Workload
Hadoop2/Yarn	enable opportunistic	Wordcount
HDFS	—	TestDFSIO,curl
HBase	—	PE,curl
Zookeeper	—	Smoketest
Cassandra	—	Stress

New bugs found by CrashTuner

- 21 new bugs, 16 are fixed, 10 critical

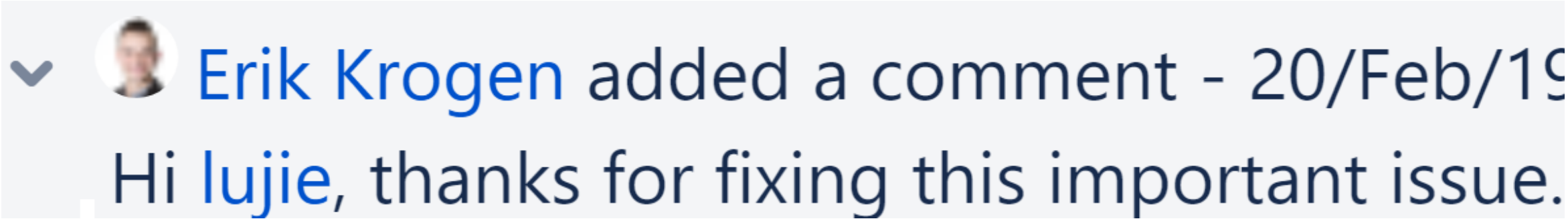
Bug ID	Priority	Scenario	Status	Symptom	Meta-info
YARN-9238	Critical	pre-read	Fixed	Allocating containers to removed ApplicationAttempt	ApplicationAttemptId
YARN-9165	Critical	pre-read	Fixed	Scheduling the removed container	ContainerId
YARN-9193	Critical	pre-read	Fixed	Allocating container to removed node	NodeId
YARN-9164(2)	Critical	pre-read	Fixed	Cluster down due to using the removed node	NodeId
YARN-9201	Major	pre-read	Fixed	Invalid event for current state of ApplicationAttempt	ContainerId
HDFS-14216(2)	Major	pre-read	Fixed	Request fails due to removed node	DataNodeInfo
YARN-9194	Critical	pre-read	Fixed	Invalid event for current state of ApplicationAttempt	ApplicationId
HBASE-22041	Critical	post-write	Unresolved	Master startup node hang	ServerName
HBASE-22017	Critical	pre-read	Fixed	Master fails to become active due to removed node	ServerName
YARN-8650(2)	Major	pre-read	Fixed	Invalid event for current state of Container	ContainerId
YARN-9248	Major	pre-read	Fixed	Invalid event for current state of Container	ApplicationAttemptId
YARN-8649	Major	pre-read	Fixed	Resource Leak due to removed container	ApplicationId
HBASE-21740	Major	post-write	Fixed	Shutdown during initialization causing abort	MetricsRegionServer
HBASE-22050	Major	pre-read	Unresolved	Atomic violation causing shutdown aborts	RegionInfo
HDFS-14372	Major	pre-read	fixed	Shutdown before register causing abort	BPOfferService
MR-7178	Major	post-write	Unresolved	Shutdown during initialization causing abort	TaskAttemptId
HBASE-22023	Trivial	post-write	Unresolved	Shutdown during initialization causing abort	MetricsRegionServer
CA-15131	Normal	pre-read	Unresolved	Request fails due to using removed node	InetAddressAndPort

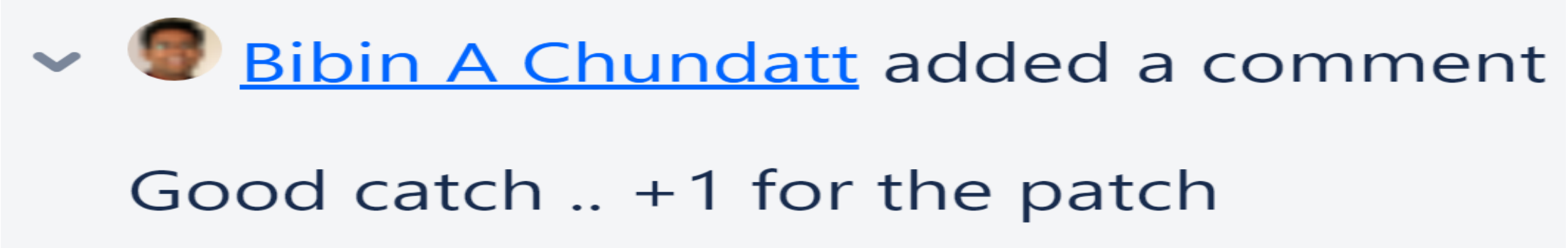
Complexity of New Bugs

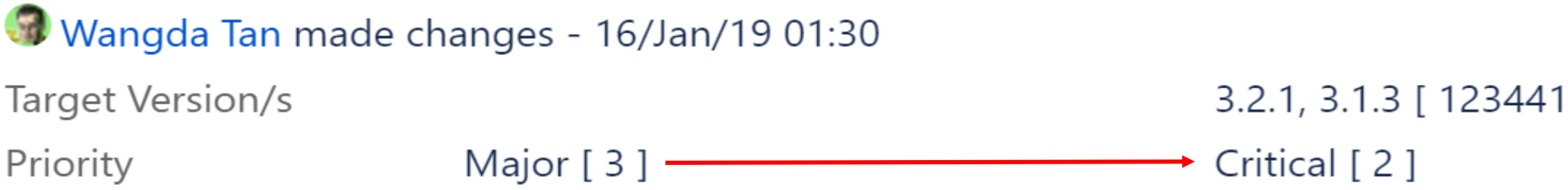
- Easy to find Complex bugs
- Confirmed quickly
  - We can easily give a unit test to reproduce each bug.

	LOC of patch	# patches	# days to fix	# comments
CREB bugs	117	4	92	26
New bugs	114.8	3.8	16.8	8.6

Feedback from developer







Comparison

Methods	Effectiveness	Efficiency
Random fault injection	7x	53X
IO around crash injection	21X	92X

Limitations and Future Work

- CrashTuner maybe not good enough to test system with Bad Log Quality
  - Developer can annotate the meta-info type.
- CrashTuner only injects one crash.
  - We can extend CrashTuner to test two or more crash events .
- CrashTuner only test Java based system
  - Our study on k8s (implemented with Golang) shows that it also have meta-info related crash-recovery bugs.
  - We are extending CrashTuner to work with System written by Golang and C++.

Relate Works

- Crash Recovery bug study
  - CBSDB[Socc2014],TaxDC[ASPLOS2016], CREB[FSE2018]
- Crash Recovery bug detection
  - Fault injection: Fate[NSDI2011], Fcatch[ASPLOS2018]
  - Mode checking:FlyMC[EuroSys2019], SAMC[OSDI2014]
- Log analysis for distributed system
  - Stitch[OSDI2016], lprof[OSDI2014]

Conclusion

- *Abstraction is so fundamental that sometimes we forget its importance!* [Operating Systems: Three Easy Pieces]
- Meta-info is a well-suited abstraction for distributed systems!
- Crash recovery bugs can happen while updating meta-info.
- CrashTuner:21 new bugs, 16 are fixed.

Questions?

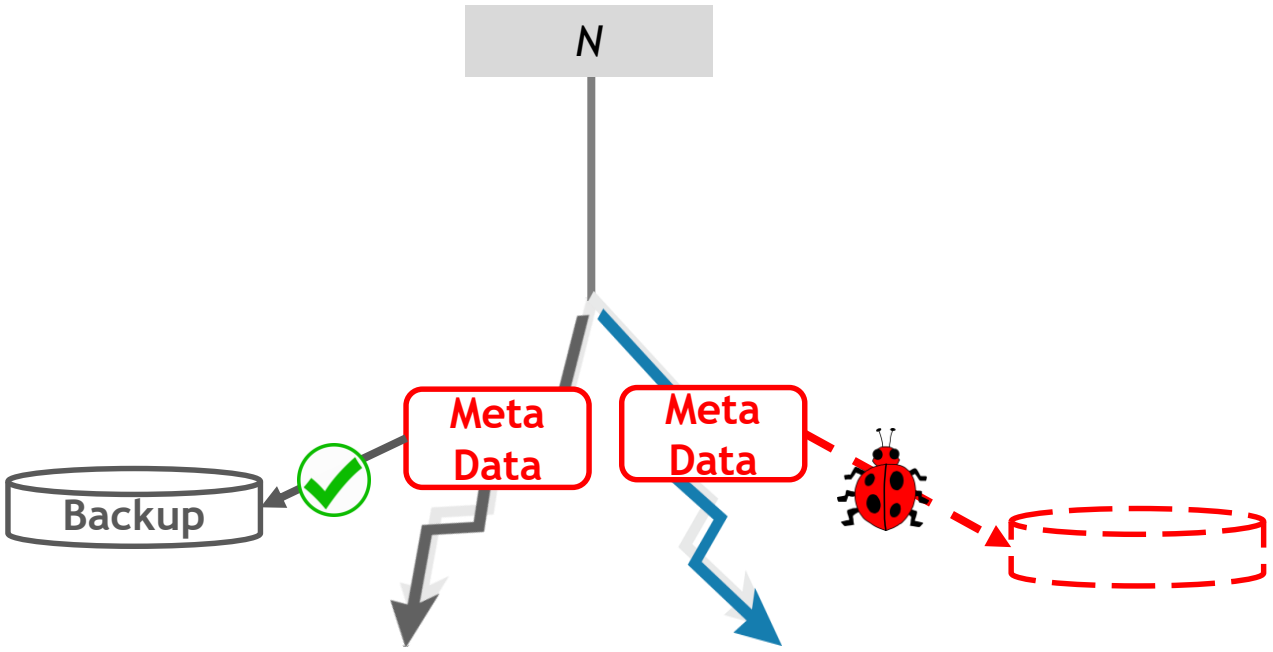
Thank you!

Two questions

- How many errors are still not identified?
- What other factors would incur the errors?

Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Meta-data Backup is not performed correctly (16.5%,CREB)



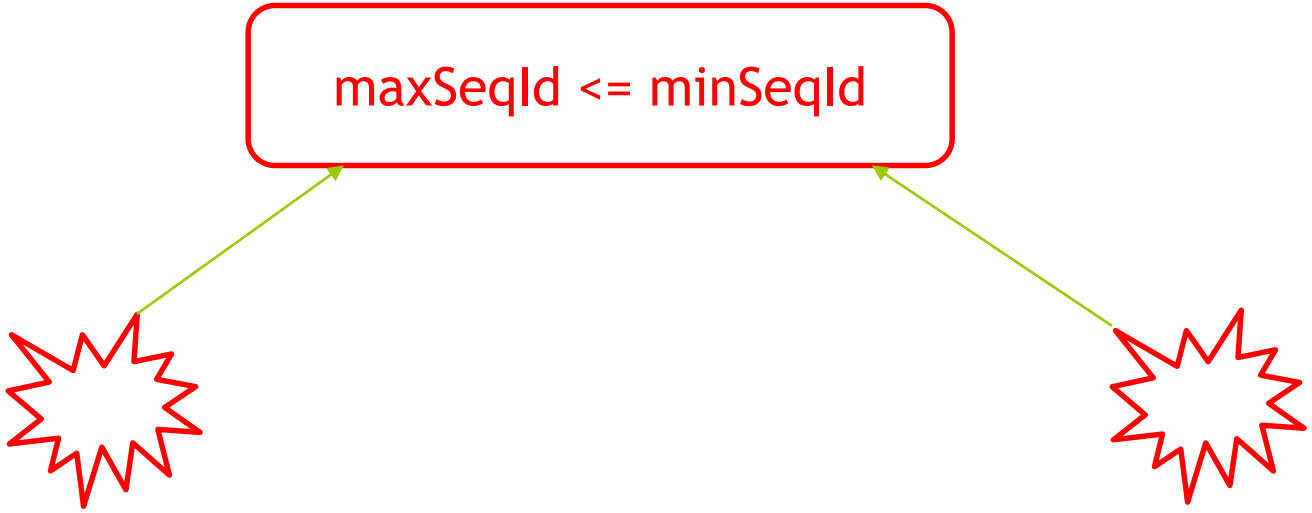
Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Multi-Crash(30%)

Key Point
Relation between crashes

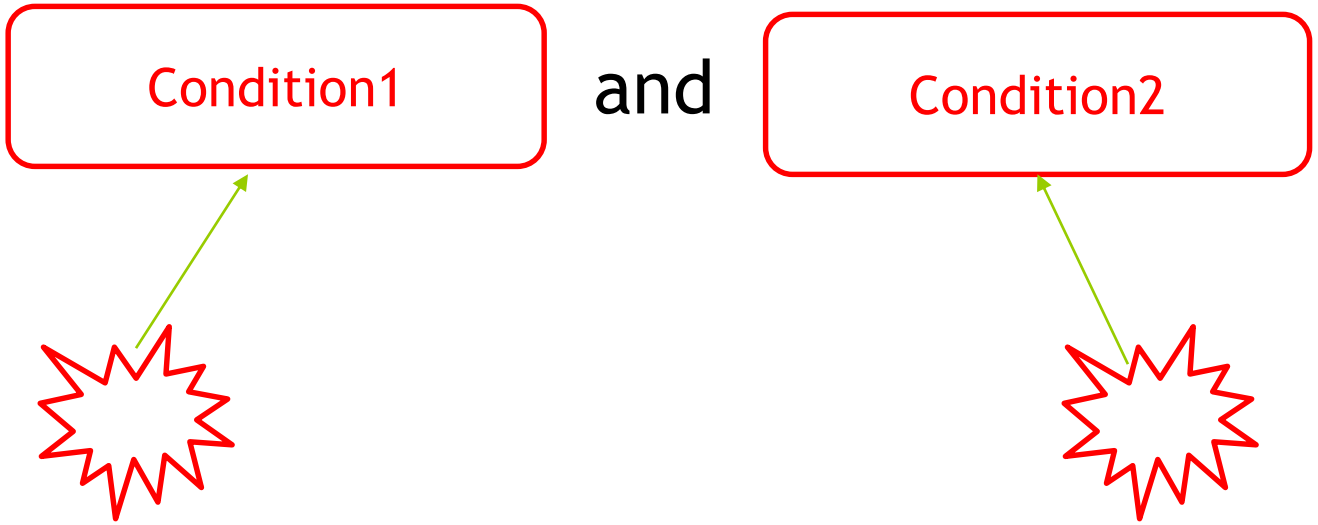
Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Multi-Crash(30%)



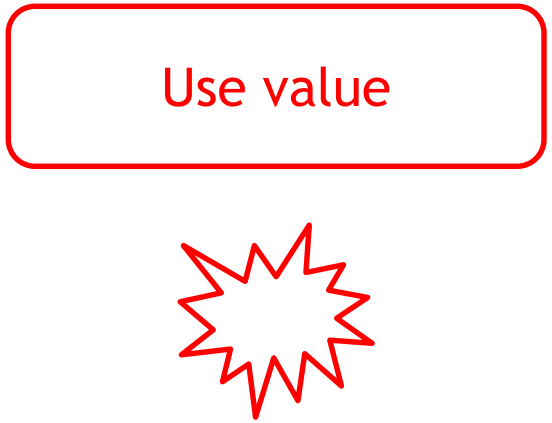
Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Multi-Crash(30%)



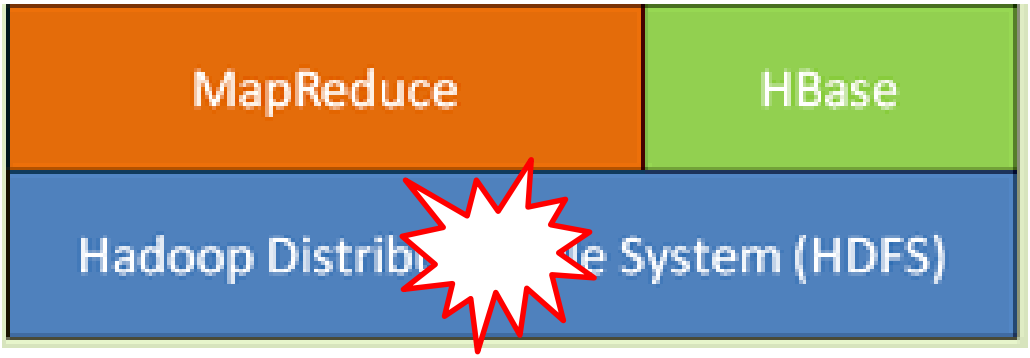
Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Multi-Crash(30%)



Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Crash at the lower layer(4%)



Cross-layer bugs

Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Special configuration(10%)

Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Timing

HDFS owns 201 Unit Test about one crash

Two questions

- How many errors are still not identified?
- What other factors would incur the errors?
- Input (workload)

Two questions

- How many errors are still not identified?
- What other factors would incur the errors?

input1

input2

conf1

message2

message1

conf2

event1,2

input2

message1

crash1

conf1

input1

conf2

message2

More systematically

Model checking