Advanced Detection of Distributed Concurrency Bugs in Cloud RAID through Log Mining and Enhancement Strategies

M.Anitha[1],E.Nagaraju[2],B.Siva Shankar[3]
#1 Assistant Professor & Head of Department of MCA, SRK Institute of Technology, Vijayawada.

#2 Assistant Professor in the Department of MCA,SRK Institute of Technology, Vijayawada.

#3 Student in the Department of MCA, SRK Institute of Technology, Vijayawada

ABSTRACT_ Cloud systems suffer from distributed concurrency bugs, which often lead to data loss and service outage. This paper presents CLOUDRAID, a new automatical tool for finding distributed concurrency bugs efficiently and effectively. Distributed concurrency bugs are notoriously difficult to find as they are triggered by untimely interaction among nodes, i.e., unexpected message orderings. To identify simultaneousness bugs in cloud frameworks proficiently and successfully, CLOUDRAID breaks down and tests consequently just the message orderings that are probably going to uncover blunders. In particular, CLOUDRAID mines the logs from past executions to reveal the message orderings that are doable however deficiently tried. Likewise, we likewise propose a log upgrading procedure to present new logs consequently in the framework being tried. These additional logs added work on additional the adequacy of CLOUDRAID without presenting any recognizable exhibition above. Our log- based approach makes it appropriate for live frameworks. We have applied CLOUDRAID to break down six delegate disseminated frameworks: Hadoop2/Yarn, HBase, HDFS, Cassandra, Animal specialist, and Flink. CLOUDRAID has prevailed with regards to testing 60 distinct variants of these six frameworks (10 renditions for each framework) in 35 hours, uncovering 31 simultaneousness bugs, including nine new bugs that have never been accounted for. For these nine new bugs identified, which have all been affirmed by their unique engineers, three are basic and have previously been fixed.

1.INTRODUCTION

Distributed systems, such as scale-out computing frameworks distributed key-value stores scalable file systems and cluster management servicesare the fundamental building blocks of moderncloud applications. As cloud applications provide 24/7online services to users, high reliability of their underlyingdistributed systems becomes crucial. However, distributedsystems are notoriously difficult to get right. There are widelyexisting software bugs in real-world distributed systems,which often cause data loss and cloud outage, costing serviceproviders millions of dollars per outrage.

Among all types of bugs in distributed systems, distributedconcurrency bugs are among the most troublesome. These bugs are triggered by complex interleavingsof messages, i.e., unexpected orderings of communicationevents. It is difficult for programmers to correctlyreason about and handle concurrent executions on multiplemachines. This fact has motivated a large body of research ondistributed system model checkers whichdetect hard-to-find bugs by exercising all possible messageorderings systematically. Theoretically, these model checkerscan guarantee reliability when running the same workloadverified earlier. However, distributed system model checkers face the state-space explosion problem. Despite recentadvancesit is still difficult to scale them to many largereal-world applications. For example, in our experimentsfor running the WordCount workload on Hadoop2/Yarn,5,495 messages are involved. Even in such a simple case, itbecomes impractical to test exhaustively all possible messageorderings in a timely manner.

2.LITERATURE SURVEY

The most crucial step in the software development process is conducting a literature survey. The time factor, economy, and company traffic redundancy elimination all need to be determined before the tool can be developed. Once these requirements are met, the operating system and programming language that can be used to develop the tool are the next steps. When the developers begin fabricating the instrument the software engineers need parcel of outside help. Senior programmers, books, and websites are all good places to look for this assistance. We must be familiar with the following ideas for developing the proposed system before building it..

1.A new general framework for secure public key encryption with keyword search

Boneh et al. introduced Public Key Encryption with Keyword Search (PEKS). enables users to search encrypted documents on an untrusted server without disclosing any information in Eurocrypt'04 The cryptographic research community has paid a lot of attention to this idea because it is very useful in many applications. However, the Keyword Guessing Attack (KGA) that is launched by a malicious server is a limitation of all existing PEKS schemes. Dual-Server Public Key Encryption with Keyword Search (DS-PEKS) is the name of the new PEKS framework we propose in this paper. This new system can endure every one of the assaults, including the KGA from the two untrusted servers, as long as they don't intrigue.

Following that, we present a generic DS-PEKS construction employing a brand- new SPHFs variant, which is of independent interest..

2.Searchable symmetric encryption: Improved definitions and efficient constructions

A party can outsource the private storage of his data to another party using searchable symmetric encryption (SSE), while still maintaining the ability to selectively search over it. This issue has been the focal point of dynamic exploration and a few security definitions and developments have been proposed.In this paper we start by evaluating existing documentations of safety and propose new definitions. Interestingly, our constructions are more efficient than any of the previous ones, in addition to meeting stronger security guarantees.

Further, earlier work on SSE just viewed as the setting where just the proprietor of the information is fit for submitting search questions. We think about the natural extension where anyone, not just the owner, can submit search queries. In this multi-user setting, we present an effective construction and formally define SSE..

3.Public Key Encryption with Keyword Search based on K-Resilient IBE

Abstract. Alice receives an encrypted email from Bob. For some reason (such as routing), a gateway wants to see if an email contains a particular keyword. However, Alice does not want anyone but herself to decrypt the email, including the gateway itself. Public key encryption with keyword search (PEKS) is required in this situation. K-Resilient Public Key Encryption with Keyword Search, or KR-PEKS, is the brand-new method we develop in this paper.
Without the random oracle, the new plan withstands a targeted keyword attack with confidence. The KR-PEKS was constructed using the ability to construct a Public Key Encryption with Keyword Search from an Identity Based Encryption. By demonstrating that the utilized IBE has a concept of key privacy, the new scheme's security was demonstrated. The plan was then adjusted in two unique ways to satisfy every one of the accompanying: The first change was made to make it possible to search for multiple keywords, and the second change was made to get rid of the need for secure channels..

4.Generic constructions of secure-channel free searchable encryption with adaptive security

For looking through catchphrases against scrambled information, public key encryption conspire with watchword search (PEKS), and its augmentation secure-channel free PEKS

(SCF-PEKS), has been proposed. In this paper, we broaden the security of SCF-PEKS, calling it versatile SCF-PEKS, wherein a foe (displayed as a "malevolent however genuine" recipient) is permitted to give test questions adaptively. We show that versatile SCF-PEKS can be conventionally developed by unknown character based encryption as it were.

3.PROPOSED SYSTEM

We propose a new approach, CLOUDRAID, for detecting concurrency bugs in distributed systems efficiently and effectively. CLOUDRAID leverages the run-time logs of live systems and avoids unnecessary repetitive tests, thereby drastically improving the efficiency and effectiveness of our approach. We describe a new log enhancing technique for improving log quality automatically. This enables us to log key communication events in a system automatically without introducing any noticeable performance penalty. The enhanced logs can further improve the overall effectiveness of our approach.

3.1IMPLEMENTATION

3.1.1 Admin

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as View All Users and Authorize, View All Datasets,View All Bug Report Datasets By Chain,View All Severity Category Results,View All Bug Fixed Results,View All Bug Resolved Results.

3.1.2 View and Authorize Users

Inthismodule,facultyregisterandlogintothesystem.Heallowsuploadingmaterials, events, attendance, marks in the system. He can view theirstudent'sattendancedetails, marksdetails, and updatehisprofile.

3.1.3 End User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like MyProfile,UploadDatasets,View All Datasets,Find Bug Severity Category,Find Severity Category Results By Hashcode.

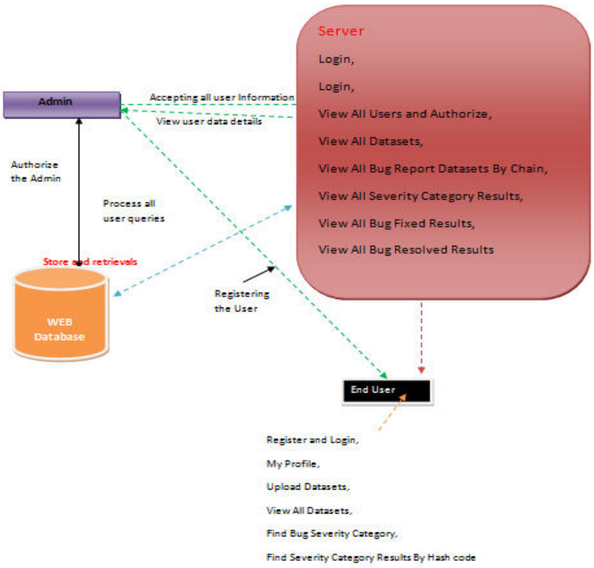

Fig 1:SYSTEM ARCHITECTURE

4.RESULTS AND DISCUSSION

FIG-1: The figure above shows the user's homepage. A user can perform operations such as viewing all users and authorizing them, viewing all datasets, viewing all bug report datasets by chain, viewing all severity category results, viewing all bug fixed results, and viewing all bug resolved results

FIG-2: the above figure shows the interface of the uploading the dataset.

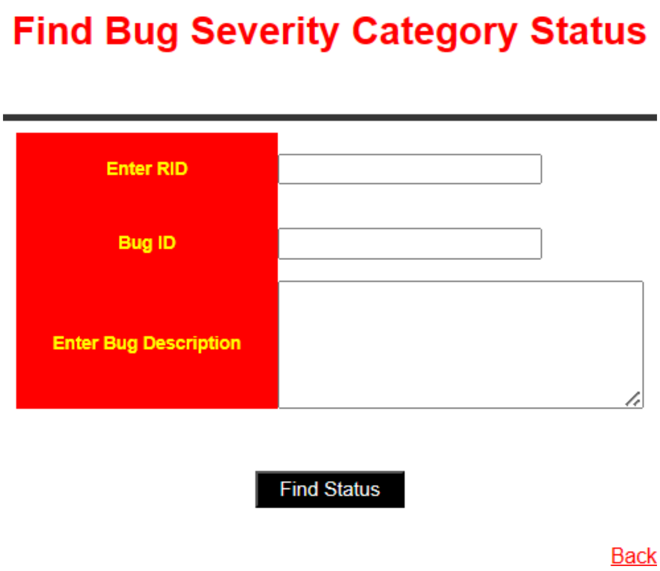

FIG-3: The figure above shows the findings of bug severity category status. You need to provide the RID, bug ID, and enter a description. After clicking on 'Find Status,' it will display the type of bug (blocker, normal, major, minor, trivial)

FIG-4: The figure above shows the findings of the severity of bug status. It displays the types of bugs such as normal, blocker, major, and minor.

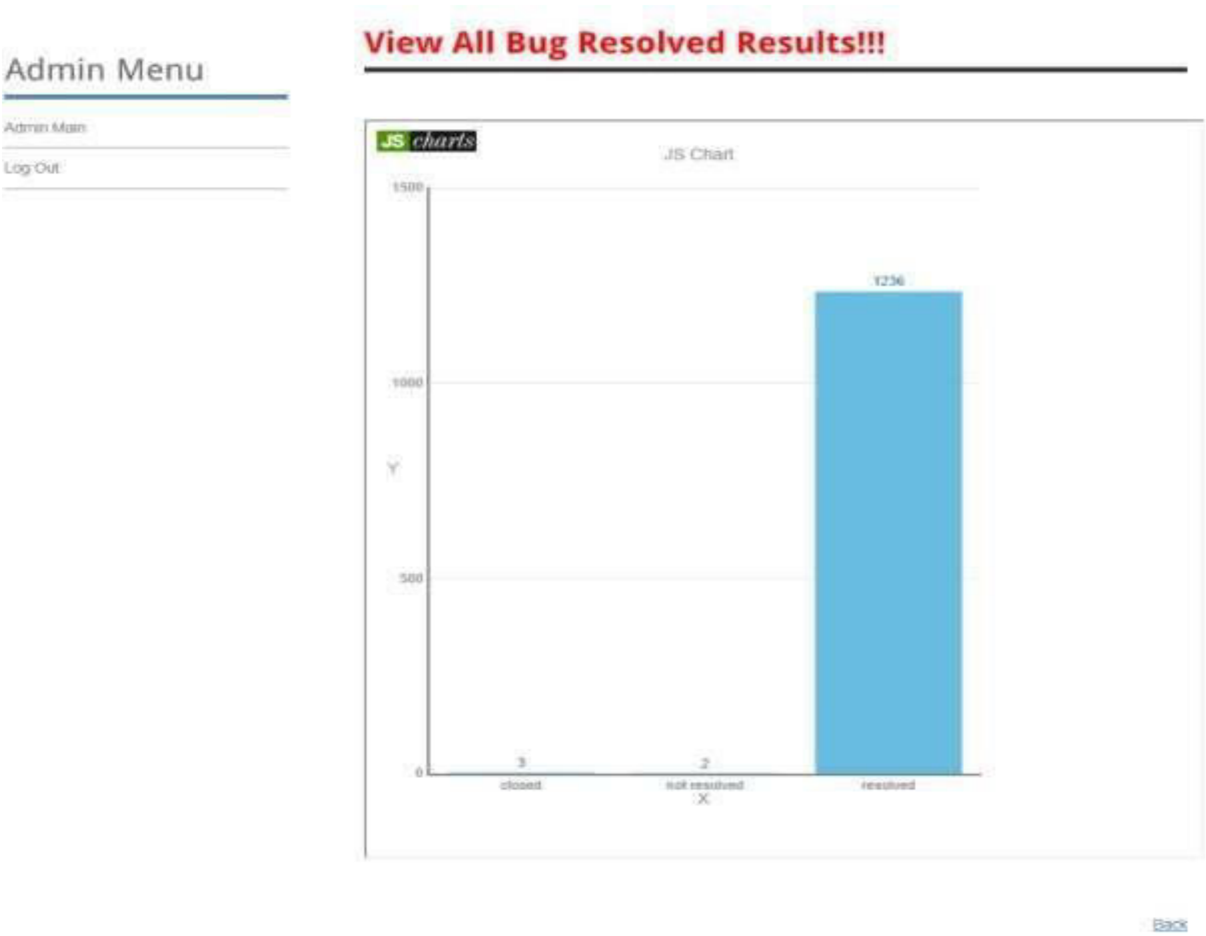

FIG-5 The figure above shows the bar graph representation of all bugs resolved. Out of 1,241 bugs, 1,239 were resolved, 3 were closed, and 2 were not resolved

5.CONCLUSION

We present CLOUDRAID, a simple yet effective tool for detectingdistributed concurrency bugs. CLOUDRAID achieves itsefficiency and effectiveness by analyzing message orderingsthat are likely to expose errors from existing logs. Ourevaluation shows that CLOUDRAID is simple to deploy andeffective in detecting bugs. In particular, CLOUDRAID cantest 60 versions of six representative systems in 35 hours,finding successfully 31 bugs, including 9 new bugs that have never been reported before.

REFERENCES

[[1] J. Dean and S. Ghemawat, "Mapreduce: Simplified data processing on large clusters," Commun. ACM, vol. 51, no. 1, pp. 107–113, Jan. 2008. [Online]. Available: http: //doi.acm.org/10.1145/1327452.1327492

[2] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, and E. Baldeschwieler, "Apache hadoop yarn: Yet another resource negotiator," in Proceedings of the 4th Annual Symposium on Cloud Computing, ser. SOCC '13. New York, NY, USA: ACM, 2013, pp. 5:1–5:16. [Online]. Available: http://doi.acm.org/10.1145/2523616.2523633

[3] L. George, HBase: the definitive guide: random access to your planet-size data. " O'Reilly Media, Inc.", 2011.

[4] A. Lakshman and P. Malik, "Cassandra: a decentralized structured storage system," ACM SIGOPS Operating Systems Review, vol. 44, no. 2, pp. 35–40, 2010.

[5] Z. Guo, S. McDirmid, M. Yang, L. Zhuang, P. Zhang, Y. Luo, T. Bergan, P. Bodik, M. Musuvathi, Z. Zhang, and L. Zhou, "Failure recovery: When the cure is worse than the disease," in Proceedings of the 14th USENIX Conference on Hot Topics in Operating Systems, ser. HotOS'13. Berkeley, CA, USA: USENIX Association, 2013, pp. 8–8. [Online]. Available: http://dl.acm.org/citation.cfm?id=2490483.2490491

[6] D. Yuan, Y. Luo, X. Zhuang, G. R. Rodrigues, X. Zhao, Y. Zhang, P. U. Jain, and M. Stumm, "Simple testing can prevent most critical failures: An analysis of production failures in distributed dataintensive systems," in Proceedings of the 11th USENIX Conference on

Operating Systems Design and Implementation, ser. OSDI'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 249–265. [Online]. Available: http://dl.acm.org/citation.cfm?id=2685048.2685068

[7] H. S. Gunawi, M. Hao, T. Leesatapornwongsa, T. Patana-anake, T. Do, J. Adityatama, K. J. Eliazar, A. Laksono, J. F. Lukman, V. Martin, and A. D. Satria, "What bugs live in the cloud? a study of 3000+ issues in cloud systems," in Proceedings of the ACM Symposium on Cloud Computing, ser. SOCC '14. New York, NY, USA: ACM, 2014, pp. 7:1–7:14. [Online]. Available: http://doi.acm.org/10.1145/2670979.2670986

[8] T. Leesatapornwongsa, J. F. Lukman, S. Lu, and H. S. Gunawi, "Taxdc: A taxonomy of non-deterministic concurrency bugs in datacenter distributed systems," in Proceedings of the Twenty-First International Conference on Architectural Support for Programming Languages and Operating Systems, ser. ASPLOS '16. New York, NY, USA: ACM, 2016, pp. 517–530. [Online]. Available: http://doi.acm.org/10.1145/2872362.2872374

[9] T. Leesatapornwongsa, M. Hao, P. Joshi, J. F. Lukman, and H. S. Gunawi, "Samc: Semantic-aware model checking for fast discovery of deep bugs in cloud systems." in OSDI, 2014, pp. 399–414.

[10] H. Lin, M. Yang, F. Long, L. Zhang, and L. Zhou, "Modist: Transparent model checking of unmodified distributed systems," in 6th USENIX Symposium on Networked Systems Design & Implementation (NSDI), 2009.

[11] J. Simsa, R. E. Bryant, and G. Gibson, "dbug: systematic evaluation of distributed systems." USENIX, 2010.

[12] H. Guo, M. Wu, L. Zhou, G. Hu, J. Yang, and L. Zhang, "Practical software model checking via dynamic interface reduction," in Proceedings of the Twenty-Third ACM Symposium on Operating Systems Principles. ACM, 2011, pp. 265–278.

[13] D. Borthakur et al., "Hdfs architecture guide," Hadoop Apache Project, vol. 53, 2008.

[14] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed, "Zookeeper: Waitfree coordination for internet-scale systems." in USENIX annual technical conference, vol. 8, no. 9, 2010.

[15] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and K. Tzoumas, "Apache flink: Stream and batch processing in a single engine," Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, vol. 36, no. 4, 2015.

[16] (2018) Wala home page. [Online]. Available: http://wala. sourceforge.net/wiki/index.php/Main_Page /.

[17] W. Xu, L. Huang, A. Fox, D. Patterson, and M. I. Jordan, "Detecting large-scale system problems by mining console logs," in Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles. ACM, 2009, pp. 117–132.

AUTHOR'S PROFILE

Ms.M.Anitha Working as Assistant Professor & Head of Department of MCA ,in SRK Institute of technology in Vijayawada. She done with B .tech, MCA ,M. Tech in Computer Science .She has 14 years of Teaching experience in SRK Institute of technology, Enikepadu, Vijayawada, NTR District. Her area of interest includes Machine Learning with Python and DBMS.

Mr.E.Nagaraju completed his Masters of Computer Applications. He has published A Paper Published on ICT Tools for Hybrid Inquisitive Experiential Learning in Online Teaching-a case study Journal of Engineering Education Transformations, Month 2021, ISSN 2349- 2473, eISSN 2394-1707. Currently working has an Assistant professor in the department of MCA at SRK Institute of Technology, Enikepadu, NTR (DT). His areas of interest include Artificial Intelligence and Machine Learning.

Mr. B. Siva Shankar is an MCA Student in the Department of Computer Application at SRK Institute Of Technology, Enikepadu, Vijayawada, NTR District. He has Completed Degree in B.Sc(MPC) from Government Autonomous College, Central Jail Road, Rajahmundry,East Godavari District. His area of interest are Java and Python..