Survey on Fault Tolerance and Residual Software Fault of the System by Using Fault Injection

Harunya B, Deepa N R, Dr.Gunasekaran S

Abstract— Fault injection is used to characterize the failure to validate and compare the fault tolerant mechanisms. A systematic and quantitative approach is using fault injection to guide the design and implementation of fault tolerance systems. The two experimental approaches were used to analyse the software fault by fault injection. In the first experiment, a set of faults is injected by SWIFI tool to evaluate the accuracy of injected faults. The second experiment, the fault triggers and fault types are used to cause the impact of fault in target system and program results. The injected fault representatives are used to observe the field data in software faults without using any excess time is spent by the fault. The general pin-hole fault injection tool can be used to demonstrate the application of MESSALINE. This application can be used to validate the inter locking systems for railway control application and dependable communication systems. The NFTAPE tool can be used to conduct the automated faults to be injected and available for lightweight fault injectors. The problem is solved by triggers, monitors and other components of fault injectors. The G-SWIFT approach can be used for removing the fault and filtering the result. This approach can be used to improving the fault representative of the system.

Index Terms— Accuracy, Fault injection, Fault tolerance, G-SWIFT, MESSALINE, NFTAPE, Software faults, SWIFI tool.

1 Introduction

SOFTWARE testing is the process of evaluating a system or its component with the intent to find that whether it satisfies the specified requirements or not. Testing is executing a system in order to identify any gaps, errors or missing requirements in contrary to the actual desire or requirements. Software Testing can be defined as a process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item. It is a process of verifying and validating that a software application or program and it meets the business and technical requirements that guide the design and development works is expected. To improve quality the coders can use the information while thinking about their designs, during coding and debugging.

2 Literature Survey

Wee teck NG and Peter M. Chen etal, 1999, [1] proposed that the safe sync, techniques can be used to write dirty file cache data is reliability in disk with last stage crash. Safe sync is not require the hardware support and used in several platform. The Systematic and quantitative approach can be used to design and implementation of fault tolerance system is used to guide the fault injection. In the iterative approach a can be used to increase the robustness of system errors. The fault injection test on BIOS safe sync of the system is used to remove the dependencies of virtual memory system and device drivers. To remove the dependencies on the kernel device drivers by using BIOS interface to the disk. The RIO cache achieves the higher reliability to write through file cache because the virtual memory protection can protect file cache from disk. It can be used to protect the file cache from the disk. Finally, BIOS can be used to no hardware features and modify the PC firmware. RIO and a write through file typically lose no data on a crash.

Henrique Madeira and Diamantino Costa etal, 2000, [2] proposed the fault injection has been used to evaluate the fault tolerance mechanisms and to assess the impact of fault in the system. The SWIFI tool used to inject the fault at machine level code that is fault classification and fault emulation of different classes in the system. To check the defect in proposed ODC, the ODC can be used to trigger the fault and to characterise the fault types. This fault types has consist of two fault injection parameters what and where, the ODC fault triggers are not define the SWIFI fault triggers. The Xeception is used to debugging and performance to monitoring the process of injected faults by software, to monitor the faults and impact of targeted systems. It affects the processing of running targeted system and inject the faults in application are not in source code. The large number of SOR programs to get the higher percentage to correct the results and the parallel way to observe the large percentage of crashes. Finally the problem to understand the faults triggers required for the emulation of software faults that comprehensive test procedure.

Christmansson.J and Chillararege. R etal, 1996, [3] proposed to estimate coverage of fault by using an analytical dependability model. The problem is to rectify the field data is limited and

- *Harunya B is currently pursuing masters degree program in Software Engineering in Coimbatore Institute of Engineering and Technology, India, PH-9994213335. E-mail: bharunya@gmail.com.*
- *Deepa N R is currently working in Computer Science and engineering in CIET , PH-97919287667. E-mail: deepu.me28@gmail.com*

difficult to define the realistic fault to sets of fault injectors. The experiment is conducted without any excessive time spent waiting for consequence of fault and to accelerate the failure process without affecting the probabilities of system. To analyse the result by using the IBM product to generate the set of errors, that maintain the distribution of set of faults and mapping the representative of software faults and injected errors. It assured to emulate the software faults and errors that are not injected by hardware faults. Finally, the faults are uniformly distributed over the affected modules cannot be rejected and to select 50% of error types such that 79.4% defects to be observed. It assures the 88.9% of timing defects is covered to cause high severity failures

Jean Arlat and Martine aguera etal, 1993, [4] proposed the problem of dependability validation of fault tolerant computing system and to validate fault tolerant mechanisms. In this proposed system used to check the physical level hardware or software prototype based on fault injection is directly to validate the design process and to clearly identify the fault injection. The general pinhole fault injection tool can be used to demonstrate the application of MESSALINE and this application is used to validate the two systems that are computer inter locking system for railway control application and dependable application system. The first targeted system is to validate PAI validation and to characterize the self testing of processing units and to identify the process to lack of diagnosis for 46.39%. The characterized deficiency of 10.90% is tested in software diagnosis to observe the 35.49% of deficiency. The second target system is use to multicast communication system (MCS) to implement the token ring of local area network. Finally, the 90% of the fault to be observe in the system and further the fault injection is validate use of network attachment controllers to be improved by process of self checking and MESSALINE is used in this system in future the rapid prototyping and simulation approaches are directly used in this system.

David T.Stott and Benjamin Floering etal, 2000, [5] proposed the several fault injection tools that are available for dependability assessment. The problem to be rectified of an injection tools is good for developing the design and no single tool to inject all fault models in this systems. To proposed a new tool NFTAPE to be developed to find the automated faults injection tools to support the set of features to evaluate the computer systems. The first, it can be used for hardware fault injectors Myrinet LAN and it uses the SWIFI fault to target the space imaging application. The component of fault injection tool are replaced by lightweight fault injectors is used to triggers the process and remaining function are handled by NFTAPE and it is flexibility to conduct the fault injection. Finally, the hardware fault injection which injects the error of data and control the signals over a Myrinet LAN is used to measure the total propagation delay as 3ns with roughly calculated by Myrinet is 1.28 gbps and allowed the 1.5ns to be required specification.

Roberto Natella and Joao A.Duraes etal, [6] proposed to improve the representativeness of the fault by using the classification algorithm and software metrics. The supervised and unsupervised algorithm can be used to improve the fault loads. This approach is used to reduce the cost and time. The fault load operators of G-SWIFT tool, the several faults can be injected and 3.8 million experiments were performed.

3 Experimental Results

The results analysis describes the several experiments that are BIOS safe sync, SWIFI Tool (Xeception) and representativeness of fault triggers and fault types, analytical dependability model, MESSALINE tool and NFTAPE tool.

In this BIOS Safe sync is used to measure the reliability of disk and safe sync are not require the hardware support and used in several platforms. BIOS safe sync is 40% more reliable than the write through file cache. SWIFI tool to evaluate the accuracy of injected faults and this result help to shows the fault emulation such as code size, complexity and recursive of execution results. The emulation software fault observed by field data and to map the representative of software faults and injected errors. The MESSALINE tool has been used to validate the systems by using the PAI validation methodology for self checking and multicast communication system has been validating the network controller of the system. NFTAPE tool is used to test the automated fault injection and is used to solve the problem for triggers, monitors and other components of faults. In G-SWIFT tool the faults can be injected and 3.8 million experiments were performed.

Table.1 Comparison of various methods and techniques used

Methods	Techniques Used	Results
Safe Sync	BIOS safe sync	Corruption rate - 1.9%
SWIFI	SOR	Higher % to correct the results
MESSALINE	MCS (Multicast Communication systems)	To observe 90% of faults
NFTAPE	Myrinet LAN	Measures 1.5ns

4 Conclusion

In this paper, the injected faults is used to evaluate the fault tolerance systems by using the various approaches like safe sync, SWIFI tool, emulates the software faults based on field

data, MESSALINE and NFTAPE. The safe sync experimental methods used the BIOS safe sync is more reliable and measure the corruption rate of 1.9% which is 40% more reliable to write through file cache. The SWIFI tool is used to evaluate the accuracy of injected faults, the fault triggers and fault types are used to cause the impact of fault in targeted systems and program results. The SOR programs to correct the higher percentage of results. To emulate the field data is conducted by without any excess time to spent waiting time of fault. The MESSALINE tool used to validate the target system of multicast communication system with 90% fault to be observed in the system. The NFTAPE tool is used to control the signals over the Myrinet LAN performed to measures the 1.5ns to be allow the required specification. In this future work, the similarities between these open source systems and industrial ones, the results can potentially be extended to other systems.

REFERENCES

[1] Ng W.T and P.M. Chen, June 1999 "Systematic Improvement of Fault Tolerance in the RIO File Cache," Proc. IEEE Fault Tolerant Computing Symp. (FTCS).

[2] J.Christmansson and R.Chillareege,"Generation of an Error Set that Emulates Software Faults based on Field Data,"Proc.26th IEEE Fault Tolerant Computing Symp.,pp.304-313,1996.

[3] D. Stott, B.Floering, "NFTAPE: A Framework for Assessing Dependability in Dis-tributed Systems with Lightweight Fault Injection,"Proc. Int'l Computer Performance and Dependability Symp., 2000

[4] J. Arlat et al, "Fault Injection and Dependability Evaluation of Fault Tolerant systems ", IEEE Transactions on Computers, vol. 42, no. 8, pp. 919-923, Aug. 1993.

[5] H. Madeira, D. Costa, and M. Vieira, "On the Emulation of Software Faults by Software Fault Injection," *Proc. IEEE Int'l Conf. Dependable Systems and Networks*, pp. 417-426, 2000

[6] M. Kalyanakrishnam, Z. Kalbarczyk, and R. Iyer, "Failure Data Analysis of a LAN of Windows NT Based Computers," Proc. Symp. Reliable Distributed Database Systems (SRDS-18), pp. 178-187, 1999.

[7] R. Chillarege, "Orthogonal Defect Classification," Handbook of Software Reliability Eng., chapter 9, IEEE CS Press, McGraw-Hill, 1995.

[8] K. Kanoun, J. Arlat, D. Costa, M.D. Cin, P. Gil, J.C. Laprie, H. Madeira, and N. Suri, "DBench: Dependability Benchmarking," Proc. Supplement of the IEEE/IFIP Int'l Conf. Dependable Systems and Networks (DSN '01), 2001.

[9] J. Voas, F. Charron, G. McGraw, K. Miller, and M. Friedman, "Predicting How Badly ―Good‖ Software can Behave," *IEEE Software*, vol. 14, no. 4, 1997

[10] J. Carreira, H. Madeira, and J.G. Silva, "Xception: A Technique for the Experimental Evaluation of Dependability in Modern Computers," *IEEE Trans. Software Eng.*, vol. 24, no. 2, pp.125-136, Feb. 1998

[11] J Voas and K Miller Dynamic testability analysis for assessing fault tolerance High Integrity Systems Journal 1(2):171-178, 1994.

[12] M. Daran and P. Thévenod-Fosse, "Software Error Analysis: A Real Case Study Involving Real Faults and Mutations", Proc. of 3rd Symp. on Software Testing and Analysis, ISSTA-3, San Diego, USA, pp. 158-171, Jan. 1996.

[13] R. K. Iyer and D. Tang, "Fault-tolerant computer system design," in Experimental Analysis of Computer System Dependability" Prentice Hall, 1996.

[14] J.A. Duraes and H. Madeira, "Emulation of Software Faults: A Field Data Study and a Practical Approach," IEEE Trans. Software Eng., vol.32, no.11, pp.849-867, Nov. 2006

[15] D. Powell, E. Martins, and J. Arlat, "Estimators for fault tolerance, coverage evaluation," IEEE Transactions on Computers, vol. 44, pp.261–274, 1995.