Seer: A Lightweight Online Failure Prediction Approach

Burcu Ozcelik  
Barcelona Supercomputing Center  
08034 Barcelona, Spain  
Email: burcu.mutta@bsc.es

Cemal Yilmaz  
Faculty of Engineering and Natural Sciences  
Sabanci University  
34956 Istanbul, Turkey  
Email: cyilmaz@sabanciuniv.edu

Abstract—In [1], we present a lightweight online failure prediction approach, called Seer, to predict the manifestation of failures at runtime, i.e., while the system is running and before the failures occur, so that preventive and/or protective measures can proactively be taken to improve software reliability. One way Seer differs from the other related approaches is that it collects information from inside program executions, from which the existing approaches generally refrain themselves due to the typically excessive runtime overheads incurred. Seer overcomes this issue by pushing the substantial parts of the data collection task onto the hardware with the help of hardware performance counters (HPCs) – CPU resident counters that record various low level events occurring on a CPU, such as the number of instructions executed and the number of branches taken. At a very high level, Seer operates as follows: functions, called seer functions, that can reliably distinguish failing executions from passing executions are determined; these functions are then instrumented in such a way that after every invocation of a seer function, a binary prediction (i.e., passing or failing) about the future of the execution is made; the instrumented system is deployed and the sequence of predictions made by the seer functions are analyzed at runtime using fixed-length sliding windows to predict the manifestation of failures. We have evaluated Seer by conducting a series of experiments on three software systems in the presence of both single and multiple defects. At the lowest level of runtime overheads, Seer predicted the failures about 54% way through the executions (when the duration of an execution is measured as the number of function calls made in the execution) with an F-measure of 0.77 (computed by giving equal importance to precision and recall) and a runtime overhead of 1.98%, on average. At the highest level of prediction accuracies, Seer predicted the failures about 56% way through the executions with an F-measure of 0.88 and a runtime overhead of 2.67%, on average. Furthermore, Seer performed significantly better than the other online failure prediction approaches used in the empirical studies.

One way we have been extending this line of work is by combining the low-level internal execution data collected by HPCs with the high-level external data, which is collected directly from outside executions, such as the number of processes and the CPU, memory, and network utilization, to further improve the quality of predictions. Another avenue we have been extensively investigating is using HPC-collected data in a related domain to detect the presence of ongoing side-channel attacks [2], [3], [4], [5] against software implementations of cryptographic applications at runtime. One type of attack we are currently interested in, is the cache-based attacks where a spy process discovers a secret key processed by a cryptographic application via creating intentional contentions in a cache memory with the victim [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21]. One approach that we have had great success with, monitors the contentions in shared resources by using HPCs and issues warnings whenever the extent to which the victim process suffers from these contentions reaches a suspicious level.

Acknowledgments

This research was supported by a Marie Curie International Reintegration Grant within the 7th European Community Framework Programme (FP7-PEOPLE-IRG-2008), and by the Scientific and Technological Research Council of Turkey (109E182).

References


