CLC _____ Number _____

UDC _____ Available for reference ☐Yes ☐No

SUSTech Southern University of Science and Technology

Undergraduate Thesis

Thesis Title: Real-time capturing of system calls on ARM

中期报告

Student Name: Haonan Li

Student ID: 11712510

Department: Department of Computer Science and Engineering

Program: Computer Science and Technology

Thesis Advisor: Fengwei Zhang

Date: March 25, 2021

COMMITMENT OF HONESTY

1. I solemnly promise that the paper presented comes from my independent research work under my supervisor's supervision. All statistics and images are real and reliable.

2. Except for the annotated reference, the paper contents no other published work or achievement by person or group. All people making important contributions to the study of the paper have been indicated clearly in the paper.

3. I promise that I did not plagiarize other people's research achievement or forge related data in the process of designing topic and research content.

4. If there is violation of any intellectual property right, I will take legal responsibility myself.

Signature:

Date:

Real-time capturing of system calls on ARM

Haonan Li

(Department of Computer Science and Engineering Advisor: Fengwei Zhang)

[**ABSTRACT**]: Reproducing a program is difficult. In the filed of application development, engineers occasionally reproduce bugs only relay on bug reports uploaded by users and attempts to emulate the failure. Unfortunately, bugs are usually not reproduced such faithfully. This is mainly because the execution of a program is always accompanied by many non-deterministic events. The recording of these non-deterministic events is an effective way to address this issue. It is noticeable that system calls are the primary source of non-deterministic events, hence we need to capture these system calls.

In this thesis, I develop a system call capturing tool. This tool utilizes Linux Tracepoint to record system calls across the entire system, with low overhead and transparent trapping. I evaluate it with real-world bugs and show that the tool works well in practice in combination with a replay system.

[**Keywords**]: Syscall, Record, Linux

Contents

1. Intoduction **2**
1.1 Problem Description 2
1.2 Current Solutions 2
1.3 General Idea 2

2. Background **2**

3. Related Work **2**

4. Design **2**

Bibliography **3**

1. Intoduction

1.1 Problem Description

The program would often fail. To sufficiently understand and prevent failures, developers requires firstly reproduce these bugs, which ensures the same output and bugs no matter how many times it is re-executed. However, directly re-exection is not suitable for non-deterministic failures, as they may not appear in a re-rection procedure. Non-deterministic failures are the consequence of non-deterministic instructions.

Instructions for running a program can be divided into two categories. One is deterministic, i.e., the behavior of the program is determined in each execution. The other type is non-deterministic, meaning that execution in different situations will have different results. Although most of the CPU execution is deterministic, non-deterministic instructions are also pervasive. This is mainly because of the fact that the execution of a program is not in an isolated system. In fact, the operating system plays a critical role in program initialization, system calls, and scheduling throughout the program lifecycle. Typical sources of nondeterminism include system calls, interrupts, signals, and data races for concurrency programs.

All these non-deterministic events can be futher classified into two types: inconstancy of the data flow - for example, certain system calls such as `getrandom()` and `getpid()`, and inconstancy of the control flow - for example, concurrency bug due to memory access in inconsistent order.

1.2 Current Solutions

Record-and-replay is a type of approaches that addresses this challenge. Most Record-and-replay systems work by first recording non-deterministic events during the original run of a program and then substituting these records during subsequent re-execution. Record-and-replay system could ultimately guarante that each replay will be identical with the initial version. The fact that a number of replay systems have been built and put into use in recent years illustrates the value of record-and-replay systems in practice.[4]

There are several ways to capture calls inline at raw runtime:…

1.3 General Idea

In this thesis, I propose a novel calls capturing tools.

2. Background

3. Related Work

4. Design

Bibliography

[1] Sanjay Bhansali, Wen-Ke Chen, Stuart de Jong, Andrew Edwards, Ron Murray, Milenko Drinić, Darek Mihočka, and Joe Chau. 2006. Framework for instruction-level tracing and analysis of program executions. In *Proceedings of the 2nd international conference on Virtual execution environments* (VEE '06). Association for Computing Machinery, New York, NY, USA, (June 2006), 154–163. ISBN: 978-1-59593-332-4.

[2] George W. Dunlap, Samuel T. King, Sukru Cinar, Murtaza A. Basrai, and Peter M. Chen. 2003. ReVirt: enabling intrusion analysis through virtual-machine logging and replay. *ACM SIGOPS Operating Systems Review*, 36, SI, (December 2003), 211–224. ISSN: 0163-5980.

[3] Baris Kasikci, Weidong Cui, Xinyang Ge, and Ben Niu. 2017. Lazy Diagnosis of In-Production Concurrency Bugs. en. In *Proceedings of the 26th Symposium on Operating Systems Principles*. ACM, Shanghai China, (October 2017), 582–598. ISBN: 978-1-4503-5085-3.

[4] Robert O' Callahan, Chris Jones, Nathan Froyd, Kyle Huey, Albert Noll, and Nimrod Partush. 2017. Engineering record and replay for deployability. In *2017 USENIX Annual Technical Conference (USENIX ATC 17)*. USENIX Association, Santa Clara, CA, (July 2017), 377–389. ISBN: 978-1-931971-38-6.

[5] Joseph Tucek, Shan Lu, Chengdu Huang, Spiros Xanthos, and Yuanyuan Zhou. 2007. Triage: diagnosing production run failures at the user's site. *ACM SIGOPS Operating Systems Review*, 41, 6, (October 2007), 131–144. ISSN: 0163-5980.

[6] Kaushik Veeraraghavan, Dongyoon Lee, Benjamin Wester, Jessica Ouyang, Peter M. Chen, Jason Flinn, and Satish Narayanasamy. 2012. DoublePlay: Parallelizing Sequential Logging and Replay. *ACM Transactions on Computer Systems*, 30, 1, (February 2012), 3:1–3:24. ISSN: 0734-2071.