Project Plan

Analyzing and Improving the Performance of SGX

Author: Fung Yuk Leung
UID: 3035277559

Creation Date: 26/9/2018
Last Revised: 29/9/2018
Version: 1.1

TABLE OF CONTENTS

INTRODUCTION

Background

Cloud computing (*e.g.* Google Cloud Platform) is highly popularized and commercialized nowadays because of its advantage of lower cost of computation and storage. In cloud computing, users send their datas to service providers, who process these datas on behalf of users and then return corresponding results. A problem arises in such a model, which is the leakage of datas due to irresponsible or untrusted service providers.

Software Guard Extensions (SGX), is invented to solve this problem. SGX works as a trusted intermediary agent between users and service providers [5]. It measures the safety of execution environments of service providers using *software attestation*. Users can know whether it is secure to send datas to the service provider according to the measurement result. Additionally, SGX uses a processor-hardened container, called *enclave*, to protect datas from illegal accesses [2].

Major Concern

While providing strong security guarantee, SGX leads to a concern about performance. SGX is completely implemented with *microcodes* [5]. Running these *microcodes* requires extra efforts, which are *instruction cycles*. That is, SGX results in an certain overhead and hence lower performance of applications [4]. The trade off between security level and performance is an extremely important consideration in software development. Therefore, some concrete datas about the performance of SGX should be provided to software developers so that they can make a decision of whether to apply SGX in their products, based on these datas as well as their software development requirements.

Scope and Contributions of The Project

This project focuses on the performance of SGX, specifically in applications of *OpenVPN*, *Memcached* and *Lighttpd*.

This project makes two contributions. First, it finds out concrete datas describing the performance of SGX. Specifically, it finds out the number of percentages describing how functions of programs using SGX costs more *instruction cycles* than regular functions.

Second, it points out some suggestions of improvement, especially on the *interface* of SGX, in order to reduce the overhead and hence heighten the performance of SGX.

Outline of The Research Paper

First, the research paper will offer background informations explaining the implementation of SGX, including how *context switching*, *caching*, *memory controller* and *microcodes* are involved. Second, it will show two groups of function codes (one group applies SGX while another does not) that are used in the experiment of measuring the performance of SGX. It will also list out experimental results of running these functions, which are numbers of required *instruction cycles* to call each of them. Third, it will compare the results of these two groups of functions in order to find out and comment the performance of SGX. Last, it will suggest some advices to improve SGX, as well as explaining and proving why these suggestions are effective.

METHOD

This project will use a paper, “Regaining Lost Cycles with HotCalls: A Fast Interface for SGX Secure Enclaves”, as guidance [3], it will focus on performance of three applications, which are *OpenVPN*, *Memcached* and *Lighttpd* (this is because they are representative of cloud services performance). For each application, two groups of functions (each group will have ten functions) will be measured using the method of controlled experiment. Among these two group, only one of them will apply SGX using Software Development Kit (SDK) of version 2.0.1 (the latest version) and hence it will become the experimental group. Another group will use regular programming techniques and hence it will become the control group. Each function will be run 10, 000 times so as to reduce errors or residuals due to *context switch*. For each execution of a function, the *real time stamp counter* instruction (RDTSCP) will be used to measure consumed instruction cycles because this is the most precise way to measure the running time. The performance of SGX will be reflected by the difference of results between the experimental group and control group.

Advices, that are suggested by this project, will be proved to be useful to improve SGX in a similar way. The only difference is that the experimental group will become a group of functions that will apply these advices, and the control group will become a group of functions that will apply original SGX.

All the above functions will be run in an environment of Supermicro server X11SSZ-QF with 64 GB DDR4 RAM@2133 MHz, Intel Core I7-6700k 4GHz with 4 hyper-threaded cores, disabled dynamic frequency and voltage scaling, and an operating system of Ubuntu server 14.04 LTS. The above model is a simulation of actual cloud server with smaller size of RAM and less number of CPUs due to limitation of laboratory. In addition, I7-6700k is selected because only 6th (or later) generation of Intel Core can support SGX [2].

RELATED STUDIES

Papers and websites that are related to the principle and implementation of SGX:

1. Frank McKeen. Intel Labs. Stanford University [Internet]. 2015. Available from: https://web.stanford.edu/class/ee380/Abstracts/150415-slides.pdf
2. Intel [Internet]. USA: Intel. [cited 2018 Sep 26]. Available from: https://software.intel.com/en-us/sgx
3. Victor Costan, Srinivas Devadas. SGX Explained. International Association for Cryptologic Research [Internet]. 2016. Available from: https://eprint.iacr.org/2016/086.pdf

Papers that are related to the performance of SGX:

1. Stefan Brenner et al. SecureKeeper: Confidential ZooKeeper using Intel SGX. Institut für Betriebssysteme und Rechnerverbund [Internet]. 2016. Available from: https://www.ibr.cs.tu-bs.de/users/brenner/papers/2016-middleware-brenner-securekeeper.pdf

Papers that are related to the improvement of SGX:

1. Ofir Weisse, Valeria Bertacco, Todd Austin. Regaining Lost Cycles with HotCalls: A Fast Interface for SGX Secure Enclaves. University of Michigan [Internet]. Available from: http://www.ofirweisse.com/ISCA17_Ofir_Weisse.pdf

Online resources that are related to sample codes using SGX:

1. an simple SGX example. Available from: https://github.com/intel/linux-sgx/tree/master/SampleCode/SampleEnclave

OBJECTIVES

1. Find out the performance of SGX in three applications, which are *OpenVPN*, *Memcached* and *Lighttpd*.
2. Find out any feasible suggestions of improvement on SGX, especially on its interface, so that the performance of SGX can be heightened.

PROJECT SCHEDULE

1. Period: 1/9/2018 - 1/10/2018
   Task: I will read papers and websites that are related to the principle and implementation of SGX
   Milestone: I should finish reading all related studies, except sample codes of SGX, that are mentioned in previous part. I should have a deep understanding about SGX, including how SGX secures the *confidentiality*, *integrity* and *freshness* of private datas as well as how SGX is implemented with *microcodes*.

2. Period: 2/10/2018 - 1/11/2018
   Task: I will read sample codes of functions which applies SGX; I will also write functions that are required to carry out the experiment.
   Milestone: for each application of *OpenVPN*, *Memcached* and *Lighttpd*, ten functions using SGX (the experimental group) and ten functions using regular techniques (the control group) should be written.

3. Period: 2/11/2018 - 1/12/2018
   Task: I will run each function that are written in Period 2 for 10, 000 times; I will also compare the differences of results between the experimental group and the control group.
   Milestone: analysis of the performance of SGX should be completed.

4. Period: 2/12/2018 - 15/12/2018
   Task: I will give suggestions to improve the performance of SGX; I will also write function that applies these suggestions.
   Milestone: for each application of *OpenVPN*, *Memcached* and *Lighttpd*, ten functions applying these suggestions should be written.

5. Period: 16/12/2018 - 1/1/2019

Task: I will run each function that are written in period 5 for 10, 000 times; I will also compare the results with those that are obtained in Period 3.
Milestone: suggestions in Period 4 should be proved whether they can improve SGX or not.

6. Period: 2/1/2019 - the end of FYP period
   Task and milestone: I will finish a complete research paper and give a presentation of it.

REFERENCES

[1] Frank McKeen. Intel Labs. Stanford University [Internet]. 2015. Available from: https://web.stanford.edu/class/ee380/Abstracts/150415-slides.pdf

[2] Intel [Internet]. USA: Intel. [cited 2018 Sep 26]. Available from: https://software.intel.com/en-us/sgx

[3] Ofir Weisse, Valeria Bertacco, Todd Austin. Regaining Lost Cycles with HotCalls: A Fast Interface for SGX Secure Enclaves. University of Michigan [Internet]. Available from: http://www.ofirweisse.com/ISCA17_Ofir_Weisse.pdf

[4] Stefan Brenner et al. SecureKeeper: Confidential ZooKeeper using Intel SGX. Institut für Betriebssysteme und Rechnerverbund [Internet]. 2016. Available from: https://www.ibr.cs.tu-bs.de/users/brenner/papers/2016-middleware-brenner-securekeeper.pdf

[5] Victor Costan, Srinivas Devadas. SGX Explained. International Association for Cryptologic Research [Internet]. 2016. Available from: https://eprint.iacr.org/2016/086.pdf