Development Plan for OSS World Challenge 2012

Registration No.	2012-　　　　　　※Registration No. need not be written.
Program title	DDT- Device Driver Tester

1. Program Overview

1) Development goals (background, goals etc.)

We develop DDT, a system for automated testing of closed-source binary device drivers against undesired behaviors, like race conditions, memory errors, resource leaks, etc. One can metaphorically think of it as a pesticide against device driver bugs.

DDT combines virtualization with symbolic execution to thoroughly exercise tested drivers along many execution paths. Symbolic execution allows to reach corner-case execution paths that are hard to reach with traditional testing and which are therefore more likely to cause crashes when the driver is deployed in production.

DDT combines a set of modular dynamic checkers to identify bug conditions and produce detailed, executable traces for every path that leads to a failure. Developers can use these traces to easily reproduce and understand the bugs using existing tools (e.g., WinDbg), thus both proving their existence and helping debug them.

We applied DDT to several closed-source Microsoft-certified Windows device drivers and discovered 14 serious new bugs in various sound and network drivers in only a few hours of testing. These bugs include system crashes ("Blue Screen of Death"), resource leaks and memory corruption caused e.g., by improper user input validation, race conditions, incorrect handling of hardware failure, etc.

DDT is easy to use, as it requires no access to source code and no assistance from users. We therefore envision DDT being useful not only to developers and testers, but also to consumers who want to avoid running buggy drivers in their OS kernels.

2) System configuration

DDT needs a host machine running a 64-bit version of Linux (e.g., Ubuntu) as well as a license of Windows XP Pro SP3 that is installed in the virtual machine and in which the tested drivers run. An 8-cores CPU with 16GB of RAM (or more) is recommended for optimal testing speed.

3) Menus

The program is a command line tool and has a configuration file. The GUI only displays the guest OS and has no menus.

4) Language used for development

DDT uses a combination of C++, C, assembly code, LLVM bitcode, Lua scripts, and dynamically generated x86.

5) Systems used

DDT is composed of the QEMU virtual machine emulator, the KLEE symbolic execution engine, and of the LLVM platform.

6) Plan for each stage of development

DDT works as a proof of concept for Windows network device drivers. We plan to improve it as follows:

- Broaden the supported device classes. While DDT can test any driver out-of-the-box, the quality of the bug reports and the degree of code coverage that can be achieved depends on the specifics of each device class.

- Introduce support for Linux device drivers. This involves the development of Linux-specific analysis plugins (e.g., to catch invocations of the kernel panic handler and report them to the user).

7) Number of personnel input and work assignment

The DDT team consists of 3 members. Vitaly Chipounov and Volodymyr Kuznetsov are the project leaders, George Candea is the project advisor.

2. Long-term prospects of the program developed (No specific form is required.)

We envision DDT to be part of the tool chain of any device driver developer and hope that it will allow to dramatically increase the quality and security of existing and future device drivers.

For this, we need DDT to be as simple to use as possible and integrate seamlessly in existing developers' tool chains and workflows, without requiring developers to change them in any way.

DDT is available on https://s2e.epfl.ch.

A tutorial can be found on
https://s2e.epfl.ch/embedded/s2e/Windows/DriverTutorial.html