Matt Renzelmann

From:	Michael Swift <swift@cs.wisc.edu>
Sent:	Tuesday, March 27, 2012 6:29 AM
To:	Matt Renzelmann
Subject:	Fwd: symdrive paper

- Mike

Begin forwarded message:

> **From:** Gilles Muller <Gilles.Muller@lip6.fr>
> **Date:** March 27, 2012 3:56:22 AM CDT
> **To:** Michael Swift <swift@cs.wisc.edu>
> **Subject: symdrive paper**
>
> Mike,
>
> Some comments about the introduction. I tried playing the bad guy to see what could be not so convincing.
>
> - The problem does not seem that common or general. The first paragraph need numbers about often "compile tested only," appears. The second paragraph about the E1000 says that only one of the hardware configurations is not tested out of 18. Could you come up with numbers that demonstrate that the problem is more common?
>
> - The third paragraph claims that driver testing is easier thanks to symdrive but does not quantify the improvement. Does the user gets a confidence that all paths are thoroughly tested?
> Also, the sentence:
> "broader testing of drivers, because anybody can test almost any driver"
> raises the problem of who is going to use symdrive. I am not sure that anyone is going to test a driver. Maybe you should better target the needs of the driver developer.
>
> We built SymDrive on S2 E [12], a system that pro-
> vides system-level symbolic execution. SymDrive uses
> the symbolic execution capability of S2 E to simulate all
> possible hardware inputs to a device driver, thereby elim-
> inating the need for the device.
> ->
> This sentence is confusing (maybe killing the paper), it looks like SymDrive is only a engineering delta on S2E, and that all hard problems are solved by S2E.
> The rest of this paragraph contains the novelty, you should focus on it. Why is it complex to develop a testing environment for drivers, why this is not "just an improved symbolic execution environment".
>
> SymDrive consists of three components: (i) a modi-
> fied version of the S2 E symbolic-execution engine; (ii)
> SymGen, a static-analysis and code transformation tool

that analyzes and prepares drivers for testing; and (iii)
a test framework that invokes checkers for verifying and
validating driver behavior.
->
The 2 first items again make it looks like an improvement over existing tools. Would it be possible to focus on the 3rd item?

SymDrive extends S2 E's lim-
ited support for driver testing to include more forms of
I/O, including memory-mapped I/O and DMA, and more
classes of drivers.
->
is this an engineering limitation of S2E or have you solved a more fundamental problem?

While prior systems enabled symbolic execution of
drivers, they did not provide the ability to do large
numbers of drivers or a wide variety of drivers, as ev-
idenced by their testing of only a few drivers in two
classes [23, 12].
->
This should be said before introducing symdrive. If I understand correctly, you are making the case that existing symbolic execution tools are promising, but they are not yet usable by the ordinary driver developer. I think this should come first before introducing symdrive. You must then point out what are the (fundamental) limitations and then show how symdrive solve them.

I also think that you should provide a metric for evaluating Symdrive. Just saying that you found x bug is not sufficient, because the reader does not know if it's good or not. Maybe you should correlate this with the usability of the tool by the "real" people.

Hope this helps,

Gilles

--
http://lip6.fr/Gilles.Muller