Editor: Michiel van Genuchten
VitalHealth Software
genuchten@ieee.org

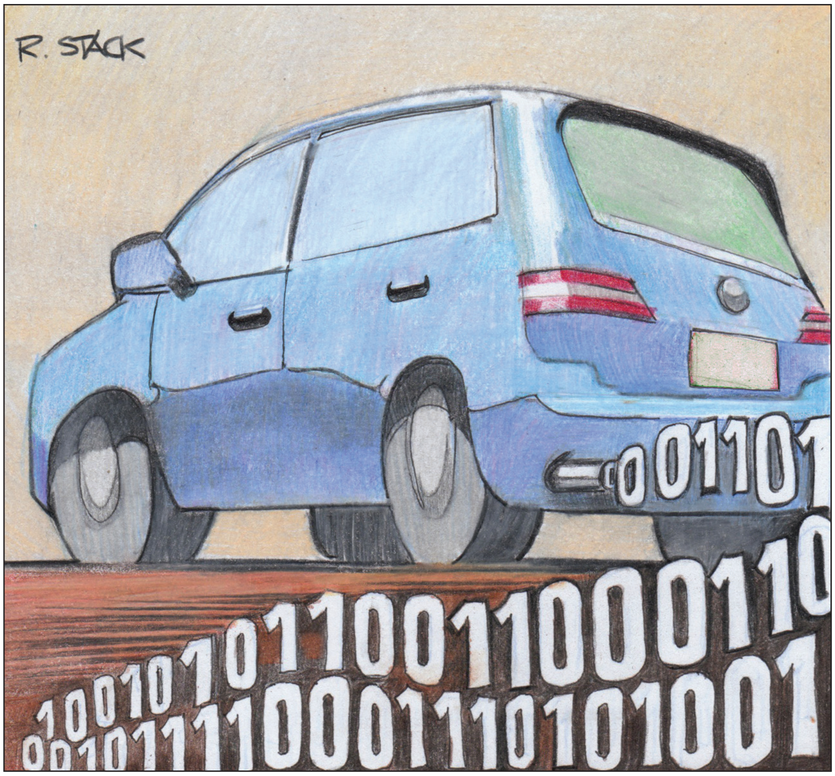
Editor: Les Hatton
Oakwood Computing Associates
lesh@oakcomp.co.uk

When Software Crosses a Line

Les Hatton and Michiel van Genuchten

LIKE MANY PEOPLE, we've been following closely the rapidly unfolding story on the "defeat" software at Germany's giant automotive company, VW. This, we believe, adds a new dimension to our original questions, "Software: What's in it and what's it in?"[1]—particularly, "What's hidden in it, and how many people knew?"

It would appear that aside from the normal and burgeoning functionality in the tens of millions of lines of code embedded in modern automotive systems,[2] in some cases there might be code intended to deceive. The question is, when does a feature cross the line from what lawyers call harmless "advertiser's puff" to outright deceit?



The Defeat Device

In the case of VW, the change appears to have been tiny—just a few lines of code in what's likely to be millions. Allegedly, the software monitored steering movement while the engine ran. On a test harness, the car wheels move but the steering wheel doesn't, unlike normal running, in which both are continually in motion. By tracking this, the software could detect when the car was in test mode and therefore control the degree to which catalytic scrubbing was done on the emissions. Catalytic scrubbers inject a mixture of urea and water into the diesel engine emissions, converting harmful nitrogen oxides into the more benign molecules of nitrogen, oxygen, water, and small amounts of carbon dioxide. The trade-off in a diesel engine is basically one of emission toxicity against car performance. The software, now known as a "defeat device," simply turned up catalytic converters' efficiency when it thought the car was under test. It's believed to have been embedded in approximately 11 million VWs and two million Audis.

Pushing the Boundaries

This development is entirely predictable but nevertheless shocking, although there's a rich history of such hardware and software fragments in devices intended to push the boundaries of what's reasonable. We recall hardware switches in early computers that simply doubled the

0740-7459/16/$33.00 © 2016 IEEE

clock speed for a disproportionately large fee. Similarly, there are possibly apocryphal stories of software wait loops intended to slow software performance that could also be removed for a suitable fee. However, these weren't intended to deceive, although they might well be considered morally suspect.

Going beyond this, one of us had experience in a legal case some years ago in which a specific test case was embedded in a software package such that the software gave favorable results for that test case during demonstrations. However, those results weren't typical for that software. Indeed, such a situation has raised its head again, and there's considerable debate on the Web about the rigging of Android benchmarks.[3] Johann Rost and Robert Glass explored the Jekyll-and-Hyde nature of software under a wider set of categories.[4]

Paying the Consequences

A cynical observer would claim that if someone can get away with something, he or she will, but did the engineers responsible really believe that such a device would never be found? Of course, unless you knew what you were looking for, finding it by inspecting the code would be like finding a needle in a haystack. And, even if you did know, finding your way round a giant software system isn't for the faint-hearted. Nevertheless, you can't defeat the laws of physics or, in this case, chemistry.

The VW defeat device was basically discovered by independent monitoring of exhaust emissions that found glaring differences between what was observed in normal running and what was being claimed. So, it seems naive to think the device wouldn't have been discovered eventually. Ironically, one of software's most important contributions to automotive engineering has been to reduce such noxious emissions by continually retuning the engine as it runs. By overstepping the mark, did the engineers responsible think that people wouldn't mind or that the financial benefit of selling more cars would outweigh any potential downside? If they did, they're likely in for an unpleasant surprise, with VW already setting aside several billion dollars to deal with potential claims.

Software is an ideal medium for this because, unlike other products, its reproduction costs are zero. Only a very small fraction of the millions of diesels sold by VW end up on a test bench, so putting a two-dollar integrated circuit in every car for the deceit would have cost a lot of money. However, copying a few lines of software into every car was a cheap solution.

This is exactly why hackers and spammers can do so much damage. Spammers wouldn't send out a million emails if they had to pay postage. Similarly, if thieves want to cheat somebody at an ATM, they have to go there (to put a fake front on the card reader, observe PIN entry, and so on) and run the risk of exposing themselves. However, if they do the cheating via phishing and emails, they can reach thousands of users without leaving their PC. Software can be turned into a weapon of mass deceit very cheaply, and we might need more explicit governance and legislation to at least discourage companies and individuals from deploying such software.

> Software can be turned into a weapon of mass deceit very cheaply.

As of October 2015, when we wrote this, over one million cars and vans might be affected in the UK, Europe's second-biggest diesel user after Germany, but VW doesn't know. In fact, the company doesn't appear to know whether the software is present or, if so, whether it's activated. VW will also have to consider the possibility of breaking something else in the process of removing the software or even simply deactivating it, owing to the possibility of unintentional side effects. These can occur through, for example, shared global variables or one of a number of mechanisms familiar to professional software engineers. In short, the software's removal could introduce one or more defects.

Perhaps the hardest thing to understand is that this allegedly is due to a very small number of rogue engineers. This is difficult to square with the detailed obligations of revision and specification control for systems that are often safety-related and demand significant oversight. VW's CEO has already lost his job, but we have yet to hear what will happen to the engineers responsible and their respective managers.

An Ethical Conundrum

Speaking of defects, let's raise an interesting question. Is this better or worse than releasing automotive software containing defects that weren't found in testing? One of the more recent examples of this is Toyota's unintended-acceleration bug.[5] Toyota isn't alone; the automotive industry has faced numerous recalls due to software defects that in some cases should have been found before release. Only two months ago, Fiat Chrysler had to recall 1.4 million vehicles fitted with the touchscreen Uconnect radios. A defect let professional hackers remotely take over an unmodified 2014 Jeep Cherokee and perform alarming maneuvers, including turning the engine off while the car was driving by and, later in a parking lot, reversing it into a ditch.[6]

When an automotive manufacturer releases such a defect while advertising how safe its cars are, is it not being similarly misleading? Ignorance is no defense, but does the automotive industry in particular, and many other industries in general, rely too much on end users being generally relaxed about software defects, even if they might be unsafe?

For example, contrast the following two more factually appropriate statements that cover the previous two eventualities:

> *We've adjusted the catalytic converter to behave more efficiently if you drive at a constant speed without moving the steering wheel, so your emissions will be much lower. If you depart from this, as seems very likely, you'll get better performance, but your emissions will be very considerably more noxious.*
>
> *We believe that software innovation is vital in automotive development. However, the systems we release to you are so complicated that they'll quite possibly have defects in them that might sometimes prejudice your safety. However, we hope that most of the time they won't and that the overall experience is beneficial to most drivers.*

Would you still buy the car? You could, of course, argue that these statements arise from different ethical viewpoints. However, any software engineer worth his or her salt will know that the chances of releasing a complicated defect-free software system are effectively negligible.[7] If, by some miracle, that system was defect-free, the engineer would never know it, would never be able to prove it, and would never be able to repeat such a feat systematically.

We await the answers to several obvious questions. Are any other companies doing this, or—if we take a more cynical standpoint—how many are doing this? If they aren't, are they still using software practices almost as dubious? How do we decide what's reasonable, given software's extraordinary ability to give hardware its character?

References

1. M. Genuchten and L. Hatton, "Software: What's In It and What's It In?," *IEEE Software*, vol. 27, no. 1, 2010, pp. 14–16.
2. J. Mössinger, "Software in Automotive Systems," *IEEE Software*, vol. 27, no. 2, 2010, pp. 92–94.
3. A. Lal Shimpi and B. Klug, "They're (Almost) All Dirty: The State of Cheating in Android Benchmarks," 2 Oct. 2013; www.anandtech.com/show/7384/state-of-cheating-in-android-benchmarks.
4. J. Rost and R.L. Glass, *The Dark Side of Software Engineering: Evil on Computing Projects*, John Wiley & Sons, 2011.
5. D. Douglas and M.A. Fletcher, "Toyota Reaches $1.2 Billion Settlement to End Probe of Accelerator Problems," *Washington Post*, 19 Mar. 2014; www.washingtonpost.com/business/economy/toyota-reaches-12-billion-settlement-to-end-criminal-probe/2014/03/19/5738a3c4-af69-11e3-9627-c65021d6d572_story.html.
6. L.P. Vellequette, "Fiat Chrysler Recalls 1.4 Million Vehicles to Install Anti-Hacking Software," *Automotive News*, 24 July 2015; www.autonews.com/article/20150724/OEM11/150729921/fiat-chrysler-recalls-1.4-million-vehicles-to-install-anti-hacking.
7. L. Hatton, "The Chimera of Software Quality," *Computer*, vol. 40, no. 8, 2007, pp. 104–107.

LES HATTON is emeritus professor of forensic software engineering at Kingston University and the managing director of Oakwood Computing Associates. Contact him at lesh@oakcomp.co.uk.

MICHIEL VAN GENUCHTEN is the chief operating officer at VitalHealth. Contact him at genuchten@ieee.org.