Hard Questions Raised When A Software 'Glitch' Takes Down An Airliner

Taylor Armerding Contributor
Cybersecurity
I cover software security and privacy.

The recent crash of a Lion Air jetliner made it clear once again that software integrity can be a matter of life and death. Credit: Getty GETTY

It doesn't take a failure of anything big to cause big trouble – as in massive,

catastrophic and lethal damage to a sophisticated transportation system.

The U.S. space shuttle Challenger exploded 73 seconds after liftoff in 1986 due to a failure of O-ring gaskets. A rocket booster came loose, which then ruptured an external fuel tank.

Some of the worst airline disasters in history were caused by instrument malfunctions: In 1997, Korean Air Flight 801 crashed three miles short of the runway in Guam in 2009, due to a fault with the Ground Proximity Warning System, killing 228 people.

Defects in software code – strings of numbers and letters – can do it too. Late last month, a Lion Air Boeing 737 Max 8 jetliner crashed into the Java Sea off Indonesia, killing all 189 passengers and crew, due to what investigators described as a "glitch" in the plane's flight-control software.

Some glitch. In most cases, that word implies a minor, temporary malfunction that can be easily fixed and doesn't cause a major problem.

Not this time. Following the crash, the Federal Aviation Administration issued an emergency notice to operators of Boeing 737 Max 8 and 9 planes, warning that faulty "Angle of Attack" sensor readings "could cause the flight crew to have difficulty controlling the airplane." This, it said, in a euphemistic phrase for a deadly crash, could lead to "possible impact with terrain." Or in this case, the ocean.

Which raises questions yet again about the reality that modern society is increasingly dependent on the security and integrity of software, not just for the magical conveniences that computers, smartphones, apps and smart devices provide, but also for the life and safety of people when they travel. Clearly, that software is not always perfect.

Yes, by any statistical measure, commercial air travel is much safer than driving.

In recent years there have been somewhere between 30,000 to 35,000 fatal auto accidents annually. Airline crashes are counted in the dozens. Some years there have been zero fatalities. Your odds of dying in a car accident are 1 in 114. Your odds of dying in a plane crash are 1 in 9,821.

And yet ... and yet, as we all know, personal vehicle fatalities per accident are generally in the single digits. Deaths from airline crashes are in the multiple dozens to hundreds.

And there is essentially no way to prevent all of them. Martin Fisher, an information security manager for an Atlanta-based hospital system who also has an interest in commercial aviation, notes the obvious: “Any complicated system is going to have issues at times and it's possible that there will be issues that will generate failures.”

So, given that systems that rely on software get more complicated all the time, is there a danger that without more rigorous software testing, public trust in commercial aviation will begin to erode?

Fisher said it comes down to how many, or how few, incidents are considered acceptable. But he said he was quite certain that the software in question was rigorously tested.

“As far as I know, the flight system isn't brand new. It’s a continuation of the 737 flight systems already developed,” he said. “The 737-MAX went through extensive flight testing so I would expect that any problems that could be found were

found."

But "could be found" could be part of the problem, according to Sammy Migues, principal scientist at Synopsys. Prior to automation, "we had a pretty short list of things that could go wrong – metallurgy, engine, construction, etc.," he said.

"Now there are a million things that could go wrong, from software, software integration, software errors, software interfaces, unexpected conditions that software has to deal with, and so on. There are way more situations that can adversely impact passenger safety.

"That means the software cannot be exhaustively tested even for things we know about, and can't possibly be tested for known unknowns and unknowns," he said, adding, "Who knows if we can even re-create in simulators all the conditions that might be found in the air?"

Does that mean it would be better if there were more manual operation of aircraft? Not necessarily, said Larry Trowell, associate principal consultant at Synopsys, given that manual operation still depends on instrument readings.

"It looked from the reports like it was a sensor problem, that was relaying information to the automated system as well as to the pilots," he said. "If you can't trust the data going into the system, going back to manual won't really help."

Trowell also said more lines of code "doesn't mean things are too complicated to get right. Testing of these components is getting better, and if one chip is shown to be malfunctioning, the entire subsystem is normally replaced, since it's easier to swap out a board than to debug an issue."

Of course, every catastrophic event attributed to a problem with software raises the question of whether it was caused by a cyber attack. There have been no

suggestions of that in the Lion Air case.

But Migues said that doesn't mean it's impossible. "Can hackers do that? Yes. Period," he said. "The attack surface for software – any software in any industry – goes back to the employees, the contractors, the builders, the integrators, the OEMs, different countries, different companies, and so on. No one can do sufficient supply chain security to catch all the potential back doors.

"Then there's the potential remote attack, the local attacks in the hangar or airport, the local attacks in the plane, and so on. Can it be planned to happen to lots of planes with the same software? The answer has to be yes. Would we be able to catch it before it happened? The answer has to be maybe."

Fisher said he thought it unlikely that software was solely to blame for the Lion Air crash. "If – and it's a big if – the software is implicated, I would assume it was some sort of sensor failure combined with unexpected software results combined with potential pilot error," he said.

Migues isn't entirely sure that the root cause will be made public. In the case of physical technology, "every incident, as far as we know, is reviewed over the course of weeks/months/years until the exact root cause is made public and everyone learns from it."

"With software, I don't know," he said. "I'm willing to believe that every incident is reviewed, but I don't know that there's always been a resolution that has been made public."

I write about software security - and insecurity - personal privacy and Big Data. I have written for CSO Online, the Sophos Naked Security blog and now for Synopsys.