HW #3 Verifying & Validating Safety-Critical Systems

Com S 4150/5150, Spring, 2025
Due April 3; turn in on Gradescope as a pdf
Participants: Benjamin Smith, Jonathan Tan, Gurumanie Singh Dhiman, Jaret Van Zee

Re-read the Homework Policy provided in Homework 1, and make sure that you understand it before beginning this homework.

1. *Static analysis of safety-critical software.* (30 pts.) Gerard Holzmann recommended that Static Code Analysis tools be used in the development of safety-critical code ("The Power of 10, Rules for Developing Safety-Critical Code"). In an article in CACM, "Mars Code," Holzmann describes the intertwined roles that (1) static code analysis, (2) code review, and (3) model checking had in making the software on the Mars Science Laboratory spacecraft & its rover, Curiosity, highly reliable and in reducing project risk.
document

a) How were the results from static code analysis tools used to assist with code peer review?

- Their redefined code review process allowed them to thoroughly scrub large amounts of code efficiently.
- Usually, code peer review is done with an expert, a few hundred lines of code at a time, to avoid burnout and reviewer/system strain. Static analyzers don't tire and patiently report all violations.
- They used four analyzers at once.
- The output of the four tools was uniformly reformatted so that all the reports could be seen in the *scrub*.
- *Scrub* also took human-generated peer code review comments as input.
- Then, the module owner looks over the human and static-generated reports and decides what to do.
- Overall, it streamlined the process considerably.

b) What parts of the code were model checked with SPIN? Why were these parts selected for model checking?

- Key parts of the control software used for verification.
- Target verification of distributed systems software with asynchronous threads of execution.
- Spin tries to find system executions that violate user-defined requirements.
- Used on critical software components including a dual-CPU boot control algo, nonvolatile flash file system, and the data management subsystem.
- These parts were selected because they were hard to test otherwise because it was unreasonably difficult to prove concurrent algorithms to be correct under all execution scenarios, and to reduce the risk of inflight surprises.

c) How was software redundancy used to reduce missions' risk (pp. 9-10)?

- Assertions in the code used as a protection mechanism were based on redundancy.
- Assertion validation like this is almost always redundant but sometimes does help with near-impossible events.

- Redundancy was also used to protect the critical landing sequence. The main CPU and the backup were used simultaneously, running not the same thing but two different versions of the entry-descent-and-landing code.

2. *Testing "untestable" safety-critical software*. (30 pts.) John Knight et al. describe testing a safety-critical software system for brain surgery (developed in collaboration with Iowa researchers) that had been described as untestable. Knight et al.'s short paper is posted on Canvas.
Document

(a) What were the issues that challenged the testing of the system?

1. The system uses a distributed architecture and depends on custom software.
2. It is not feasible to power down the system on a routine or extended basis for testing.
3. The system requires input from a physician.
4. Correct operation of the system is very difficult to determine because complex calculations are required to perform the necessary control.
5. Vast number of possible test cases possible when testing the system.

(b) Describe how the authors' testing approach addressed each of these issues.

1. Reversal Checks, testing starting at the output and working backward to the input, were used to test existing software already in the system.
2. The team developed several synthetic devices to emulate the actual devices (e.g., X-ray and other high-energy electrical devices) to address any accessibility issues to the actual devices. A test harness was used to produce results and images from the X-ray.
3. A pseudo-user was created as a physician to give the necessary inputs into the software interface when information/commands are sent from the test harness.
4. The team implemented reversal checks to address issues of determining correctness in the outputs. *Reversal Checks* are performed by working backward from the output and computing what the theoretical input should have been. Then, check to see if the theoretical input matches the actual input.
5. To limit test cases, the research team employed a specification limitation technique in the overall test plan. They did this by purposefully limiting the specifications in several areas of the overall system.

3. *Beyond safety: application to other high-dependability systems.* (40 pts.) The Risks digest logs incidents that are risks to the public in computers and related systems. Note that the digest has search features. Within the incidents reported in 2023 and 2024, find two incidents, both of which you find interesting and both of which **involve accidents or near-accidents** that might have been avoided by the use of Leveson's approach to building safety-critical systems (i.e., anything we've covered in lectures). The incidents can involve safety or other dependability properties, including security, privacy, or reliability.

(a) Provide a description of each of the two incidents.

(b) For each, discuss how using Leveson's approach might have prevented the incident or its recurrence or might have mitigated its effects. This question aims to show how specific design-for- safety mechanisms can reduce risk in real-world applications.

Incident 1: Woman Died Trapped in Burning SUV After Vehicle Malfunction (2023). On December 9, 2023, 73-year-old Mary L. Frahm died inside her 2009 Dodge Journey after it caught fire on the side of a Wisconsin road. She was trapped inside the vehicle, unable to unlock the doors or open the windows. Just before the fire, she had called her fiancé and said her car was "acting up" and she couldn't get out. Various systems malfunctioned: the speedometer was erratic, and windows and doors were unresponsive. Investigators found a failure in the Totally Integrated Power Module (TIPM), a central unit responsible for controlling most of the vehicle's electrical systems. The TIPM in her model had a known history of electrical issues, including short circuits and spontaneous system failures. [1, 2]

Leveson discussion: The TIPM design created a single point of failure that could simultaneously turn off multiple safety systems, violating Leveson's framework. A single failure in the electrical system cascaded across numerous other systems, like the door and window locks. Leading to the woman's death. Moreover, the lack of mechanical overwrite to the system violates Leveson's safety requirements. It shows that the design did not prioritize fail-safe behavior, violating fundamental system safety principles.

Incident 2: In August 2023, Microsoft faced criticism for reportedly violating its security policies (by not storing its security keys within Hardware Security Modules) in a way that allowed adversaries, believed to be the Chinese government, to use an expired key to gain access to email accounts of U.S. government agencies. This incident raised severe concerns about the security practices of Microsoft because, in addition to the inappropriateness of the expired key, Microsoft told its customers that it had inappropriately allowed the expired vital to remain in the consumer's account for months, exposing concerns about Microsoft's security protocols regarding sensitive government email accounts. [1, 2]

Leveson's Discussion: Leveson's approach to software system safety mentions, "all inputs used in specifying output events must be properly limited in the time they can be used (data age)''. Output commands that may not be able to be executed immediately must be limited in the time they are valid." This indicates that in the incident above, if Microsoft had kept their keys in a Hardware Security Module, or better yet, if they had ensured that their expired keys were revoked access from the system, all of this could have been prevented. This situation demonstrates that Leveson and her Data Age Criterion were violated and should have been obsolete and discarded appropriately.