Independent Submission
Request for Comments: 9225
Category: Informational
ISSN: 2070-1721

J. Snijders
Fastly
C. Morrow
Google
R. van Mook
Asteroid

1 April 2022

Software Defects Considered Harmful

Abstract

This document discourages the practice of introducing software defects in general and in network protocol implementations specifically. Software defects are one of the largest cost drivers for the networking industry. This document is intended to clarify the best current practice in this regard.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9225.

Table of Contents

1. Introduction

Software defects (informally known as "bugs") have been the cause and effect of innumerable system degradations and failures over the years. Bugs are errors, flaws, or faults in a computer program that cause the program to produce an incorrect or unexpected result.

(Please note: unexpected results caused by bugs are not a valid substitute for high-quality random number generators, though high-quality random number generators are generally not considered to be bugs.)

Endeavoring to reduce the number of degradations in the future, implementers MUST NOT introduce bugs when writing software. This document outlines why bugs are considered harmful and proposes a set of recommendations.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Examples of High-Impact Software Defects

In June 1996, the European Space Agency [ARIANE] launched an unmanned rocket -- costing several billion dollars in development -- only to see it go [KABOOM] 40 seconds after takeoff. A software exception had occurred during the execution of a data conversion from 64-bit floating point to 16-bit signed integer value. The floating point number that was converted had a value greater than what could be represented by a 16-bit signed integer. The vehicle probably would not have disintegrated if the defect had not been written into the software.

As an example of the detrimental effects of bugs in physically hard to reach systems: the [NASA] Deep Impact spacecraft [DEEPIMPACT] was rendered inoperable due to a fault in the fault-protection software, which in turn triggered endless computer reboots. Mission control was unable to recover the system from this error condition because no engineers were available on-site. The commute was deemed infeasible due to a lack of reasonably priced commercial transport options in that region of the solar system.

In 1983, the Soviet Union's Early Warning Satellite System [Serpukhov] announced it had detected a possible missile launch originating in the US; fortunately, a human operator recognized this as a likely system failure. Indeed, a retrospective analysis

suggested the software had misclassified reflections from cloud cover as missile launch blooms. With this bug, the software held the potential to trigger a cascading sequence of events that could've led to the start of a planetary-scale war. Seemingly innocuous software defects can have outsized impact, and sometimes it pays off to simply do nothing and wait.

The US Department of Commerce's National Institute of Standards and Technology [NIST] commissioned a study to develop a deeper understanding of the prevalence of software defects and their cost to society. The study estimated about 0.6 percent of the gross domestic product is squandered due to programming bugs. Each person works approximately one hour a week to compensate for this debt -- an hour that could've been spent in leisure -- in addition to any time spent on the direct consequences of buggy software.

The universal deployment of IP networks on Avian Carriers [RFC1149] is facing a multi-decade delay. After operators discovered that birds are not real (now [confirmed] by the US Government), work began to first understand the many [quirks] of the drones' firmware before proceeding with wider-scale deployment. No clear timelines exist at this point in time.

For more examples, consult the RISKS Digest [RISKS]: it documents a multitude of examples of defects in technological infrastructure and their risk to society. Unsupervised study of the Digest archive may induce a sense of panic.

4. Best Current Practises

1. Authors MUST NOT implement bugs.

2. If bugs are introduced in code, they MUST be clearly documented.

3. When implementing specifications that are broken by design, it is RECOMMENDED to aggregate multiple smaller bugs into one larger bug. This will be easier to document: rather than having a lot of hard-to-track inconsequential bugs, there will be only a few easy-to-recognise significant bugs.

4. The aphorism "It's not a bug, it's a feature" is considered rude.

5. Assume all external input is the result of (a series of) bugs. (Especially in machine-to-machine applications such as implementations of network protocols.)

6. In fact, assume all internal inputs also are the result of bugs.

5. Security Considerations

With the production of fewer bugs, there will necessarily be fewer security impacts. To improve the collective security posture, a thorough review of ALL existing software to find any remaining bugs is RECOMMENDED.

As it is assumed that there is an even distribution of bugs through

all software, it is safe to consider any piece of software to be bug free once a certain number of bugs have been found.

Some philosophers argue in defense of an obviously wrong contrary view that bugs introduce a certain amount of unpredictable variance in behaviour, which in turn could serve to increase security. Such heretics might even go one step further and celebrate the existence of bugs, shielding issues from public scrutiny. However, it [ostensibly] is in society's best interest to fully disclose any and all bugs as soon as they are discovered.

6. IANA Considerations

IANA is assumed to operate flawlessly.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

7.2. Informative References

[ARIANE] Arnold, D. N., "The Explosion of the Ariane 5", August 2000, <https://www-users.cse.umn.edu/~arnold/disasters/ariane.html>.

[confirmed] US Consumer Product Safety Commission (@USCPSC), "Birds are real.", Twitter, 5 January 2022, <https://twitter.com/USCPSC/status/1478794691634155523>.

[DEEPIMPACT] Wallace, M., "Subject: Re: [tz] Deep Impact: wrong time zone?", message to the tz@iana.org mailing list, 23 September 2013, <https://mm.icann.org/pipermail/tz/2013-September/020357.html>.

[incomplete] Raatikainen, P., "Gödel's Incompleteness Theorems", Stanford Encyclopedia of Philosophy, November 2013, <https://plato.stanford.edu/entries/goedel-incompleteness/>.

[IRTF] IRTF, "Internet Research Task Force", <https://www.irtf.org/>.

[KABOOM] Jure, V. A., "Kapow! Zap! Splat! How comics make sound on the page", The Conversation, 10 June 2021,

make-sound-on-the-page-160455>.

[NASA] NASA, "NASA's Deep Space Comet Hunter Mission Comes to an End", September 2013, <https://www.nasa.gov/mission_pages/deepimpact/media/deepimpact20130920.html>.

[NIST] NIST, "Software Errors Cost U.S. Economy $59.5 Billion Annually", Wayback Machine archive, June 2002, <https://web.archive.org/web/20090610052743/http://www.nist.gov/public_affairs/releases/n02-10.htm>.

[ostensibly] Swire, P., "A Model for When Disclosure Helps Security: What Is Different About Computer and Network Security?", 3 Journal on Telecommunications and High Technology Law 163, August 2004, <http://dx.doi.org/10.2139/ssrn.531782>.

[quirks] Stockton, N., "What's Up With That: Birds Bob Their Heads When They Walk", WIRED, January 2015, <https://www.wired.com/2015/01/whats-birds-bob-heads-walk/>.

[RFC1149] Waitzman, D., "Standard for the transmission of IP datagrams on avian carriers", RFC 1149, DOI 10.17487/RFC1149, April 1990, <https://www.rfc-editor.org/info/rfc1149>.

[RISKS] ACM Committee on Computers and Public Policy, "The RISKS Digest", <https://catless.ncl.ac.uk/Risks/>.

[Serpukhov] Long, T., "Sept. 26, 1983: The Man Who Saved the World by Doing ... Nothing", WIRED, September 2007, <https://www.wired.com/2007/09/dayintech-0926-2/>.

Appendix A. Future Research

The existence of this very document of course begs the question: what are software defects, truly? Do bugs happen for a purpose? Is what we perceive as the concept of bugs an indication for a wider issue in the natural world? Do mistakes happen in other domains? Are they evidence of a superior software architect?

An interdisciplinary approach to understand mistakes might be an area of further study for the [IRTF]. It may very well turn out that mistakes are provably detrimental in all domains; however, the authors do not feel qualified to make any statements in this regard. Once made aware of the above thesis, research-oriented interest groups could perhaps take on the task of disproving Goedel's incompleteness theorem [incomplete], and in doing so, put an end to all bugs.

Acknowledgements

The authors wish to thank Bert Hubert, Peter van Dijk, and Saku Ytti for pointing out the many errors Job introduced during the preparation of this document.

Authors' Addresses

Job Snijders
Fastly
Amsterdam
Netherlands
Email: job@fastly.com

Chris Morrow
Google
Reston, Virginia
United States of America
Email: morrowc@ops-netman.net

Remco van Mook
Asteroid
Deventer
Netherlands
Email: remco@asteroidhq.com