Robustness principle

In computing, the **robustness principle** is a design guideline for software that states: "be conservative in what you do, be liberal in what you accept from others". It is often reworded as: "be conservative in what you send, be liberal in what you accept". The principle is also known as **Postel's law**, after Jon Postel, who used the wording in an early specification of TCP.[1]

In other words, programs that send messages to other machines (or to other programs on the same machine) should conform completely to the specifications, but programs that receive messages should accept non-conformant input as long as the meaning is clear.

Among programmers, to produce compatible functions, the principle is also known in the form be contravariant in the input type and covariant in the output type.

Contents

Interpretation

RFC 1122 (1989) expanded on Postel's principle by recommending that programmers "assume that the network is filled with malevolent entities that will send in packets designed to have the worst possible effect".[2] Protocols should allow for the addition of new codes for existing fields in future versions of protocols by accepting messages with unknown codes (possibly logging them). Programmers should avoid sending messages with "legal but obscure protocol features" that might expose deficiencies in receivers, and design their code "not just to survive other misbehaving hosts, but also to cooperate to limit the amount of disruption such hosts can cause to the shared communication facility".[3]

Criticism

In 2001, Marshall Rose characterized several deployment problems when applying Postel's principle in the design of a new application protocol.[4] For example, a defective implementation that sends non-conforming messages might be used only with implementations that tolerate those deviations from the specification until, possibly several years later, it is connected with a less tolerant application that rejects its messages. In such a situation, identifying the problem is often difficult, and deploying a solution can be costly. Rose therefore recommended "explicit consistency checks in a protocol ... even if they impose implementation overhead".

From 2015 to 2018, in a series of Internet-Drafts, Martin Thomson argues that Postel's robustness principle actually leads to a *lack* of robustness, including security:[5]

> A flaw can become entrenched as a de facto standard. Any implementation of the protocol is required to replicate the aberrant behavior, or it is not interoperable. This is both a consequence of applying the robustness principle, and a product of a natural reluctance to avoid fatal error conditions. Ensuring interoperability in this environment is often referred to as aiming to be "bug for bug compatible".

In 2018, a paper on privacy-enhancing technologies by Florentin Rochet and Olivier Pereira showed how to exploit Postel's robustness principle inside the Tor routing protocol to compromise the anonymity of onion services and Tor clients.[6]

See also

- Unix philosophy
- Static discipline

References

1. Postel, Jon, ed. (January 1980). *Transmission Control Protocol* (https://tools.ietf.org/html/rfc761). IETF. doi:10.17487/RFC0761 (https://doi.org/10.17487%2FRFC0761). RFC 761 (https://tools.ietf.org/html/rfc761). Retrieved June 9, 2014.
2. Braden, R., ed. (October 1989). *Requirements for Internet Hosts: Communication Layers* (https://tools.ietf.org/html/rfc1122). IETF. doi:10.17487/RFC1122 (https://doi.org/10.17487%2FRFC1122). RFC 1122 (https://tools.ietf.org/html/rfc1122). Retrieved June 9, 2014.
3. Wilde, Erik (2012) [1999]. *Wilde's WWW: Technical Foundations of the World Wide Web* (https://archive.org/details/springer_10.1007-978-3-642-95855-7). Springer-Verlag. p. 26 (https://archive.org/details/springer_10.1007-978-3-642-95855-7/page/n48). doi:10.1007/978-3-642-95855-7 (https://doi.org/10.1007%2F978-3-642-95855-7). ISBN 978-3-642-95855-7.
4. Rose, M. (November 2001). *On the Design of Application Protocols* (https://tools.ietf.org/html/rfc3117). IETF. doi:10.17487/RFC3117 (https://doi.org/10.17487%2FRFC3117). RFC 3117 (https://tools.ietf.org/html/rfc3117). Retrieved June 9, 2014.
5. Thomson, Martin (May 2019). *The Harmful Consequences of the Robustness Principle* (https://tools.ietf.org/html/draft-iab-protocol-maintenance). IETF. Retrieved October 4, 2019.
6. Rochet, Florentin; Pereira, Olivier (2018). "Dropping on the Edge: Flexibility and Traffic Confirmation in Onion Routing Protocols" (https://petsymposium.org/2018/files/papers/issue2/popets-2018-0011.pdf) (PDF). *Proceedings of the Privacy Enhancing Technologies Symposium*. De Gruyter Open (2): 27–46. ISSN 2299-0984 (https://www.worldcat.org/issn/2299-0984).

External links

- History of the (Internet) Robustness Principle (https://ironick.typepad.com/ironick/2005/05/my_history_of_t.html) (Nick Gall, May 2005)
- Internet Protocol (https://www.postel.org/ien/txt/ien111.txt), page 22; J. Postel, IEN 111, August 1979.
