What is a secure programming language?

(Personal reflection)

1. **What factors determine a secure programming language?**

   A secure programming language can provide type safety, static data types, and *memory management* (Croft et al., 2021). However, it is essential also to consider the *usage domain* and *programming paradigms.* These additional considerations determine the functionalities supported by a given language.

   - *Type safety.* This language support prevents mismatched operations on data types. These languages generally prevent buffer overflow errors since they handle typing and memory management tasks (Jim et al., 2002)
   - *Data Type/Type Checking.* Data assignment errors are raised either at compile time or runtime.
   - *Memory management.* Is whether or not a language provides memory safety, control and management and whether or not the responsibility for memory lies with the user.
   - *Usage domain.* Command-line (shell) languages' security requirements differ from a language whose domain is Mobile or even Cloud.
   - *Paradigm supported.* A paradigm determines how a programming language supports various programming concepts (Van Roy, 2009). Classifying languages into different paradigms is difficult—27 paradigms currently exist. But, four stand out: procedural (or imperative), object-oriented, functional programming and relational/logic programming.

2. **Is Python a secure programming language?**

   Yes and No.

   - Yes. Because it does exhibit some level of *memory management* and *type safety* that enables developers to write reliable code.
   - Yes. Python helps to prevent buffer overflow errors through the use of "long integer object".

- No. Because it defaults to the use of the `eval()` function to evaluate string inputs. It is impossible to control the evaluation operation and allows malicious code to execute without limits.
- No. Because Python has no control over accessing closed file descriptors.
- No. Because the `input()` function evaluates the input as an expression with no control or limits to execution.
- No. A large number of vulnerabilities in Python libraries (Branca, 2014).

3. **Is Python best suited to develop operating systems versus C?**

No.

- *Security knowledge*. According to the excellent work by Croft et al. (2021), C/C++ developers (compared with Python developers) have a greater appreciation for security-related concerns based on StackOverflow data. Still, a smaller margin exists when compared to GitHub data. Developing an operating system requires access to the host machine's underlying hardware. Such direct access inevitably must deal with memory and data as well as device instructions.
- *Performance*. CPython is the default Python implementation implemented in the C language. CPython compiles Python source code into bytecode and then passes this bytecode to an interpreter that interprets it step-by-step. It is, however, possible to compile Python code directly to C code using the Cython project (Cython, ND). Still, this project is aimed mainly at building C extensions into the Python interpreter.
- *Lack of true multithreading*. CPython does not support proper multithreading. Due to the use of a Global Interpreter Lock (GIL), only one thread can process Python bytecode at any time (Beazley, 2010). Therefore, Python is not suited for CPU-intensive tasks. However, utilising external libraries may execute concurrent code as they are not subject to the GIL. Work done by Antoine Pitrou improves the GIL in Python version 3.2.

However, Insights (2020) and IEEE (2021) note that Python is well-suited for embedded devices—C/C++ are one or two points below—despite the above points.

References

Beazley, D. (2010). Understanding the python gil. *PyCON Python Conference*. Atlanta, Georgia.

Croft, R., Xie, Y., Zahedi, M., Babar, M.A. & Treude, C. (2021). An Empirical Study of Developers' Discussions about Security Challenges of Different Programming Languages. arXiv:2107.13723.

Cython (ND) Cython C-Extensions for Python. Available from https://cython.org/ [Accessed 02 Sep. 2021]

Branca, E. (2014) Secure Coding with Python. Available from https://owasp.org/www-pdf-archive/OWASP_Romania_Branca.pdf [Accessed 02 Sep. 2021]

IEEE (2021) Top Programming Languages 2021: Python dominates as the de facto platform for new technologies. Available from https://spectrum.ieee.org/top-programming-languages-2021 [Accessed 2 Sep. 2021]

Insights (202) Top 17 Programing Languages for Embedded Systems Work. Available from https://insights.dice.com/2020/08/21/top-17-programming-languages-embedded-systems-work/ [Accessed 02 Sep. 2021]

Van Roy, P. (2009) Programming paradigms for dummies: What every programmer should know. *New computational paradigms for computer music*, 104:616-621.

Bibliography

Jim, T., Morrisett, J.G., Grossman, D., Hicks, M.W., Cheney, J. & Wang, Y. (2002). Cyclone: a safe dialect of C. *USENIX Annual Technical Conference, General Track*:275-288.