# Programming: LLVM, Glibc, Python, Fortran and More

By *Roy Schestowitz*
Created *10/09/2019 - 5:58pm*
Submitted by Roy Schestowitz on Tuesday 10th of September 2019 05:58:20 PM Filed under Development [1]

- **The New Features Of LLVM 9.0 & Clang 9.0 - Includes Building The Linux x86_64 Kernel [2]**

  The LLVM 9.0 release is running a few weeks behind schedule but should be out in the days ahead along with other LLVM sub-project releases like Clang 9.0. Here's a look at what's on tap for this half-year update to the LLVM compiler infrastructure.

- **A bug found in Glibc limits modern SIMD instructions to only Intel, inhibiting performance of AMD and other CPUs [3]**

  Yesterday, Mingye Wang reported a bug in the Glibc, GNU C Library. According to him, the dl_platform detection performs ?cripple AMD? in the sysdeps in Glibc. The dl_platform check is used for dispatching SIMD (Single instruction, multiple data) libraries.

  Explaining the bug in detail, Wang writes, that in 2017, Glibc got the capability to transparently load libraries for specific CPU families with some SIMD extensions combinations to benefit the x86 users. However, this implementation limits two ?good? sets of modern SIMD instructions to only Intel processors that prevent competitor CPUs with equivalent capabilities to fully perform, something that should not work in any free software package.

- **Find the maximum gap between the successive numbers in its sorted form from a Python list [4]**

  Given a Python list consists of plus or minus numbers, we need to sort that list then find the

maximum gap between the successive numbers in that list regarding of its sign.

- **LEGB? Meet ICPO, Python?s search strategy for attributes [5]**

  When it comes to variables, Python has a well-known search strategy, known by the acronym ?LEGB.? Whenever you mention a variable ? and by ?variable,? I mean a name that could be referencing data, a function, or a class ? Python tries to find it in four different places: The local (function) scope, the enclosing function?s scope, the global scope, and finally in the ?builtins? namespace.

  Variable scoping seems both boring and annoying, but it actually explains a lot about Python?s design. It?s really worth learning about, and won?t take much of your time. Indeed, I have a free e-mail course on the subject; you?re welcome to subscribe.

  But what about attributes? How does Python search for those?

- **Layering security throughout DevOps [6]**

  The DevOps movement has changed how we integrate and publish our work. It has taken us from slow, sometimes yearly, release cycles to daily (or even hourly, in some cases) releases. We are capable of writing code and seeing our changes in production almost instantly. While that can give our customers and us a warm and fuzzy feeling, it can also provide an opening for malicious attackers.

  DevOps was an amazing first step to break down walls and support fast responses to market changes and customer demands, but there is still an important wall we need to break, one important group we need to bring into the fold: security operations (SecOps).

- **Excellent Free Books to Learn Fortran [7]**

  Fortran (Formula translation) is a multi-paradigm programming language invented by John Backus of IBM in the 1950s. It is particularly notable for innovation; it was the first high-level language, using the first compiler.

  The language is designed to be simple to understand, yet retains the efficiency in execution as assembly language ? about 80% as efficient as assembly/machine code. Fortran is machine independent, and a problem oriented language. It is often used in the scientific community, particularly among physicists, and is designed for scientific numerical computing. Fortran allows for high parallelization, it?s easy to optimize, and lends itself particularly well to computationally intensive fields such as finite element analysis, numerical weather prediction,

computational physics, computational chemistry, and computational fluid dynamics.

Fortran has evolved over time, with various standards including Fortran IV, Fortran 77, Fortran 90 and Fortran 95. More recent revisions are Fortran 2003, and Fortran 2008. Since Fortran 9x, it has many structured programming features, dynamic memory, operator overloading, and primitive objects. It is both the language of the past, the current, and the future (high-performance computing is unlikely to cast aside Fortran). Despite its age, Fortran is still very much alive and kicking. Fortran has a vast number of libraries of code.

Development

---

**Source URL:** http://www.tuxmachines.org/node/127976

**Links:**
[1] http://www.tuxmachines.org/taxonomy/term/145
[2] https://www.phoronix.com/scan.php?page=news_item&px=LLVM-9.0-Clang-9.0-Features
[3] https://hub.packtpub.com/a-bug-found-in-glibc-limits-modern-simd-instructions-to-only-intel-inhibiting-performance-of-amd-and-other-cpus/
[4] https://kibiwebgeek.com/find-the-maximum-gap-between-the-successive-numbers-in-its-sorted-form-from-a-python-list/
[5] https://lerner.co.il/2019/09/10/legb-meet-icpo-pythons-search-strategy-for-attributes/
[6] https://opensource.com/article/19/9/layered-security-devops
[7] https://www.linuxlinks.com/excellent-free-books-learn-fortran/