Contribution ID: **161** Type: **not specified**

Security improvements in GCC

Wednesday, 22 September 2021 08:30 (30 minutes)

There are multiple security features that have been requested for the Linux Kernel for a long time (https://outflux.net/slides/2020/lpc/gcc-and-clang-security-feature-parity.pdf). This wishlist includes wipe call-used registers on return, auto-initialization of stack variables, unsigned overflow detection, etc ⋯

Some of these security features have been available in CLANG, or other compilers for some time. The lack of these features in GCC makes it less competitive than other compilers regarding security.

For over a year, we have been working hard in order to make GCC comparable with, or even better than other compilers in this area.

The focus of this talk is on two security features that we have recently implemented in GCC11, or that we are currently working on for GCC12.

The first feature is called “wipe call-used registers on return”. This is a technique to mitigate ROP (Return-Oriented Programming) and addresses the register erasure problem as mentioned in the “SECURE project and GCC”talk at Cauldron 2018 (https://gcc.gnu.org/wiki/cauldron2018#secure).

This project has been completed and the corresponding patch has been committed to GCC11. In this patch, we have added the new “-fzero-call-used-regs”option, plus the new function attribute “zero_call_used_regs”, to GCC.

To improve kernel security, this new feature is now used in the Linux Kernel. See https://patchwork.kernel.org/project/linux-kbuild/patch/20210505191804.4015873-1 keescook@chromium.org for more details.

The second feature is called “stack variables auto-initialization”. It is a technique to provide automatic initialization of automatic variables. LLVM has supported the -ftrivial-auto-var-init=pattern/zero option to provide this functionality. This is currently implemented as a plugin in the Linux kernel source tree, but ideally the compiler supports it natively, without the need for an external plugin.

This project is ongoing. The 7th version of the patch has been submitted to GCC upstream for review and discussion (https://gcc.gnu.org/pipermail/gcc-patches/2021-July/576341.html).

In the talk we provide a high-level overview of these two features. This includes a description of the issues, the motivation, major considerations, some interesting implement

I agree to abide by the anti-harassment policy

I agree

Primary author: ZHAO, Qing

Presenter: ZHAO, Qing

Session Classification: GNU Tools Track

Track Classification: GNU Tools Track