One of the first features merged for the 5.10 kernel development cycle was support for the Arm v8.5 memory tagging extension [PDF]. By adding a "key" value to pointers, this mechanism enables the automated detection of a wide range of memory-safety issues. The result should be safer and more secure code once support for the feature shows up in actual hardware.

As one might expect, the Arm64 architecture uses 64-bit pointers to address memory. There is no need (yet!) for an address space that large, though, so normally only 48 of those bits are actually used by the hardware or 52 bits if a special large-address-space option is enabled. So there are 12-16 bits that can be used for other purposes. Arm systems have long supported a "top byte ignore" feature that allows software to store arbitrary data in the uppermost byte of a virtual address, but the hardware designers have been busy coming up with other uses for those bits as well. The memory tagging extension (MTE) is one of those uses.

Specifically, MTE allows the storage of a four-bit "key" in bits 59-56 of a virtual address?the lower "nibble" of the top byte. It is also possible to associate a specific key value with one or more 16-byte ranges of memory. When a pointer is dereferenced, the key stored in the pointer itself is compared to that associated with the memory the pointer references; if the two do not match, a trap may be raised. Keys can be managed by the application, or they can be randomly generated by the CPU.

Four bits only allow for 16 distinct key values, but that is enough to do some interesting things. If a function like malloc() ensures that allocations that are adjacent in memory have different key values, then an access that overruns any given allocation will be detected by the processor. Use-after-free bugs can be detected by changing the key value immediately when a range of memory is freed. If each stack frame is given its own key, buffer overruns on the
stack will also generate traps. An attempt to dereference a completely wild pointer (or one injected by an attacker) also has a good chance of being detected.

5.10 Merge window, part 1 [LWN.net] [3]

As of this writing, 7,153 non-merge changesets have been pulled into the mainline Git repository for the 5.10 release over a period of four days. This development cycle is clearly off to a strong start. Read on for an overview of the significant changes merged thus far for the 5.10 kernel release.

Intel's Cloud-Hypervisor 0.11 Adds Windows Guest Support [4]

Intel has a shiny new feature release out of their open-source Cloud-Hypervisor that runs atop KVM and leveraging the Rust programming language.

Cloud-Hypervisor 0.11 comes with some prominent improvements for this increasingly used component in the open-source Linux virtualization stack. As mentioned, even Microsoft has been working with Cloud-Hypervisor among other IHVs and ISVs.

Linux Frame-Buffer Console To Drop Accelerated Scrolling Since It's Full Of Bugs - Phoronix [5]

The Linux kernel's frame-buffer console (FBCON) is set to drop accelerated scrolling support since it isn't widely used and now found to be "full of bugs" plaguing the code-base.

Google's Syzbot that continuously fuzzes the Linux kernel using Syzkaller recently began fuzzing the FBCON code within the kernel. As a result of that exposure, the developers are now well aware with "solid proof that it's full of bugs."

The best solution from the developer perspective has been to delete the code / faulty features, such as with the recent deleting of soft scrollback support. Given the use-cases for FBCON and only a few drivers supporting accelerated scrolling, it's the latest feature now slated for removal.

Source URL: http://www.tuxmachines.org/node/143836

Links:
[2] https://lwn.net/Articles/834289/
[3] https://lwn.net/Articles/834157/