svnUPDATE

by Steven Kreuzer

It has been a topic richly debated in the community for years, but Address Space Layout Randomization has landed in FreeBSD 13. Depending on who you ask, this state-of-the-art mitigation technique will either keep your machine safer than encasing it in concrete or it's a technology that has lived past its prime and only provides a false sense of security. As for my thoughts on the topic, as with politics, I don't like to discuss security at the dinner table, but I will mention that I recently added WITH_PIE=yes into /etc/make.conf.

Implement Address Space Layout Randomization (ASLR)— https://svnweb.freebsd.org/changeset/base/343964

With this change, randomization can be enabled for all non-fixed mappings. It means that the base address for the mapping is selected with a guaranteed amount of entropy (bits). If the mapping was requested to be superpage aligned, the randomization honors the superpage attributes.

Although the value of ASLR is diminishing over time as exploit authors work out simple ASLR bypass techniques, it eliminates the trivial exploitation of certain vulnerabilities, at least in theory. This implementation is relatively small and happens at the correct architectural level. Also, it is not expected to introduce regressions in existing cases when turned off (default for now) or cause any significant maintenance burden.

The randomization is done on a best-effort basis—that is, the allocator falls back to a first fit strategy if fragmentation prevents entropy injection. It is trivial to implement a strong mode where failure to guarantee the requested amount of entropy results in mapping request failure, but I do not consider that to be usable.

Add WITH_PIE knob to build Position Independent Executables— https://svnweb.freebsd.org/changeset/base/344179

Building binaries as PIE allows the executable itself to be loaded at a random address when ASLR is enabled (not just its shared libraries). With this change, PIE objects have a .pieo extension and INTERNALLIB libraries libXXX_pie.a.

MK_PIE is disabled for some kerberos5 tools, Clang, and Subversion, as they explicitly reference .a libraries in their Makefiles. These can be addressed on an individual basis later. MK_PIE is also disabled for rtld-elf because it is already position-independent using bespoke Makefile rules. Currently only dynamically linked binaries will be built as PIE.

Take steps towards multicore bhyve AMD support— https://svnweb.freebsd.org/changeset/base/343075

Vmm's CPUID emulation presented Intel topology information to the guest, but disabled AMD topology information and in some cases passed

through garbage. CPUID leaves 0x8000_001[de] were passed through to the guest, but guest CPUs can migrate between host threads, so the information presented was not consistent. This could easily be observed with 'cpucontrol -i 0xfoo /dev/cpuctl0'.

Slightly improve this situation by enabling the AMD topology feature flag and presenting at least the CPUID fields used by FreeBSD itself to probe topology on more modern AMD64 hardware (Family 15h+). Older stuff is probably less interesting. I have not been able to empirically confirm it is sufficient, but it should not regress anything either.

Mask Spectre feature bits on AMD hosts— https://svnweb.freebsd.org/changeset/base/343166

For parity with Intel hosts that already mask out the CPUID feature bits that indicate the presence of the SPEC_CTRL MSR, do the same on AMD. Eventually we may want to have a better support story for guests, but for now, limit the damage of incorrectly indicating an MSR we do not yet support. Eventually, we may want a generic CPUID override system for administrators, or for minimum supported feature sets in heterogeneous environments with failover. That is a much larger scope effort than this bug fix.

Add definitions for AMD Spectre/Meltdown CPUID information— https://svnweb.freebsd.org/changeset/base/343120

No functional change, aside from printing recognized bits in CPU identification. The bits are documented in 111006-B "Indirect Branch Control Extension"[1] and 124441 "Speculative Store Bypass Disable."[2]

Notably missing— (left as future work):

- Integration with hw.spec_store_bypass_disable and hw_ssb_active flag, which are currently Intel-specific
- Integration with hw_ibrs_active global flag, which are currently Intel-specific
- SSB_NO integration in hw_ssb_recalculate()
- Bhyve integration (PR 235010)

Implement per-CPU pmap activation tracking for RISC-V— https://svnweb.freebsd.org/changeset/base/344108

This reduces the overhead of TLB invalidations by ensuring that we only interrupt CPUs that are using the given pmap. Tracking is performed in pmap_activate(), which gets called during context switches: from cpu_throw(), if a thread is exiting or an AP is starting, or cpu_switch() for a regular context switch. For now, pmap_sync_icache() still must interrupt all CPUs.

Implement transparent 2MB superpage promotion for RISC-V— https://svnweb.freebsd.org/changeset/base/344106

The changes are largely modelled after amd64. arm64 has more stringent requirements around superpage creation to avoid the possibility of TLB conflict aborts, and these requirements do not apply to RISC-V, which like amd64, permits simultaneous caching of 4KB and 2MB translations for a given page. RISC-

V's PTE format includes only two software bits, and as these are already consumed, we do not have an analogue for amd64's PG_PROMOTED. Instead, pmap_remove_l2() always invalidates the entire 2MB address range. pmap_ts_referenced() is modified to clear PTE_A, now that we support both hardware- and software-managed reference and dirty bits. Also fix pmap_fault_fixup() so that it does not set PTE_A or PTE_D on kernel mappings.

Provide userspace versions of do_cpuid() and cpuid_count() on i386— https://svnweb.freebsd.org/changeset/base/344118

When generating PIC code, some older compilers cannot handle inline asm that clobbers %ebx (because %ebx is used as the GOT offset register). Userspace versions avoid clobbering %ebx by saving it to stack before executing the CPUID instruction.

STEVEN KREUZER is a FreeBSD Developer and Unix Systems Administrator with an interest with retro-computing and air-cooled Volkswagens. He lives in Queens, New York, with his wife, daughter, and dog.