Contribution ID: **210** Type: **not specified**

Avoiding Security Flaws

Tuesday, August 25, 2020 10:30 AM (30 minutes)

At the end of the day, "security flaws" are just a special case of "regular" bugs, so anything that helps avoid bugs will also help with reducing the incidence of security flaws. This explores the approaches taken to avoiding bugs generally and security flaws in particular.

Find and fix bugs before they are released. This is fundamentally a matter of testing. Whether that's done via unit testing, functional testing, regression testing, or fuzzing, there are a few basic dependencies:
- code coverage (how do you know which code got tested?)
- deterministic failure (hard to fix a bug if it can't be reproduced)
- disable randomization during debugging
- always initialize memory allocation contents

Limit userspace behaviors to avoid hitting bugs (if you can't reach a bug, you can't trip over it), mainly via attack surface reduction:
- DAC (everyone understands uids, and file permissions)
- MAC (LSMs: SELinux, AppArmor, etc)
- seccomp (syscall limitations)
- Yama (ptrace limitations)

And most importantly, generalize any work done to fix bugs. Instead of fixing the same kind of bug over and over, focus on removing entire classes of bugs.
- redesign APIs that were easy to misuse (avoid shooting yourself in the foot)
- remove features that only causes problems (e.g. %n in format strings)
- create detection systems that catch a bug before it happens (e.g. saturate reference counters)

I agree to abide by the anti-harassment policy

I agree

Primary author: COOK, Kees (Google)

Presenter: COOK, Kees (Google)

Session Classification: Kernel Dependability & Assurance MC

Track Classification: Kernel Dependability & Assurance MC