ISOLATION, INTERFACES, AND SANDBOXING

CARSTEN WEINHOLD
APP ISOLATION

Hardware

Operating System

App

App
OS/SERVICE ISOLATION

Hardware

Operating System Kernel

Service Service Service

App App
VM-BASED ISOLATION

- **Virtual Machine**
- **App**
- **App**
- **Service**
- **Service**
- **Service**
- **Operating System Kernel**
- **Virtual Machine Monitor**
- **Hypervisor**
- **Hardware**

**Isolation, Interfaces, and Sandboxing**
VM-BASED ISOLATION

**Virtual Machine**

- App
- Service
- Operating System Kernel
- Virtual Machine Monitor
- Hypervisor

**Hardware**

- Hypervisor
- Virtual Machine Monitor
- Operating System Kernel
- Service
- App
Isolated components interact with each other through interfaces.

Some access control can be enforced at interface (via hardware or by a more privileged component like the operating system kernel).
Interfaces shall limit access to internal sub systems.

Isolation and access control within a component rely on integrity of its control flow and internal (shared) state.

Vulnerabilities in sub systems may expose internal state.
SANDBOXING

- Sandboxes restrict programs such they can only access a (minimal) subset of interfaces or system-level objects
  - **Namespaces**: BSD jails, Linux containers, ...
  - **System-call filters**: SELinux, Seatbelt, ...
  - **Voluntarily**: drop root rights, Linux seccomp, OpenBSD pledge, FreeBSD capsicicum, ...

- Can be combined with program splitting (e.g., render processes in web browsers)
IN-THE-WILD IOS EXPLOIT CHAIN

Discussion of Google Project Zero Blog Post

CARSTEN WEINHOLD
Safari web browser is split into multiple processes, with "render processes" being sandboxed.

The XNU kernel implements part of the graphics driver that the render process is allowed to use.

Other processes run with higher privileges than the web browser, but they are isolated.

The XNU kernel enforces sandbox restrictions and all other isolation.

Nothing can go wrong here ... ?
DISCUSSION OF BLOG POST
LESSONS LEARNED

- Often more than one component has to be attacked
- Multiple bugs may be necessary to gain full access:
  1. Find bug in web browser (or another app) and exploit it
  2. Interact with kernel (of another privileged component) and exploit bug in it to escape from sandbox
- Other exploit chains could require jumping from one sandboxed process to another, before exploiting a privilege escalation bug
- **Sandboxing makes attacks harder, but not impossible**
- If one exploit in the chain does not work (or a component has no bug), it will break the exploit chain
Original source:

Annotated version for this lecture:
Part 1, Part 2