Contribution ID: **241** Type: **not specified**

Closing the script execution control gap

Thursday 19 September 2024 10:30 (25 minutes)

Secure systems need to control code execution, to either deny untrusted (and potentially malicious) code, or to run it in a confined environment (i.e. a sandbox restricting access to resources). Linux provides a wide range of access control systems for different use cases but one remaining major gap is script execution control. Indeed, the kernel can only mediate access to resources it manages, but scripts are executed by interpreters that are not aware of the system security policy. In a nutshell: ./script.sh vs. sh script.sh

We are proposing to close this gap with a set of new kernel features (previously known as O_MAYEXEC). This is the first step to be able to have full control over code executed on a system. The next steps include script interpreters and dynamic linkers enlightenment, but also configuration of the execution policy by system components.

We'll first give an update on the ongoing kernel side implementation, and we'll explain the reasons leading to these interfaces, including perquisites and limitations.

We'd then like to discuss and answer questions about code execution control, the current status in user space changes (e.g. Python), and especially the required changes to system components in charge of launching services and applications to control the execution policy (e.g. with systemd's unit).

Primary author: SALAÜN, Mickaël (Microsoft)

Presenter: SALAÜN, Mickaël (Microsoft)

Session Classification: Kernel <-> Userspace/Init/System Management boundaries and APIs MC

Track Classification: Kernel <-> Userspace/Init/System Management boundaries and APIs MC