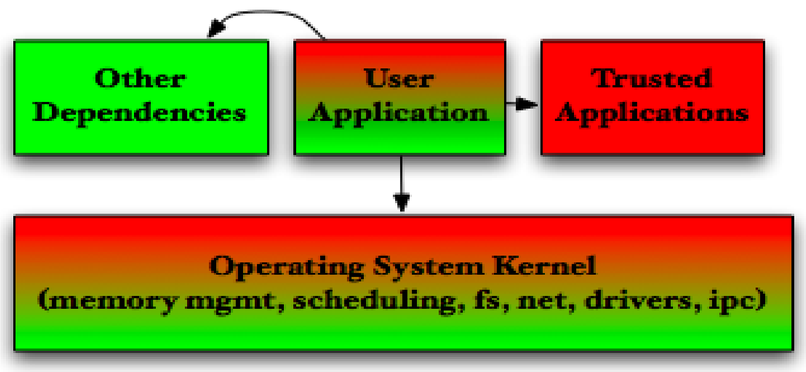
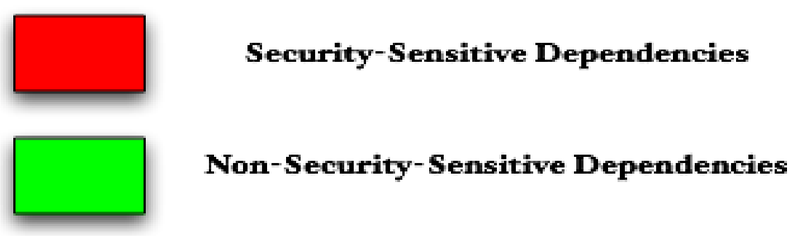
Exploring the TCB of Applications

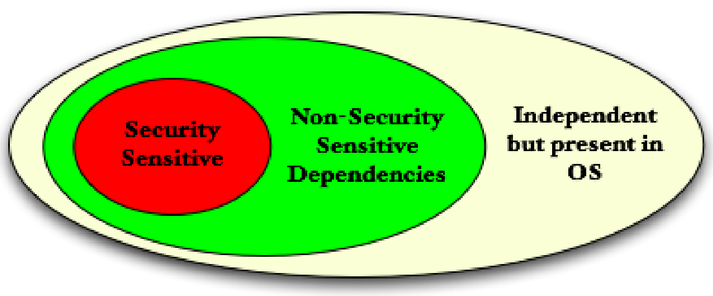
Hayawardh Vijayakumar, Trent Jaeger

- A Trusted Computing Base (TCB) of an operating system (OS) is the minimal amount of software necessary to enforce the security of the system. The TCB runs at a high privilege level, so a compromise of any TCB component compromises the whole system.
- The TCB has become bloated because it is the common factor among all applications running in a typical system.
- From the point of view of a single application, however, not all of the TCB is used. The TCB is determined by the security-sensitive operations performed by the application.
- How much code that is security-sensitive is needed by a single application taken in isolation?

The Problem. Given an application and its characteristics, we aim to precisely define and locate its security-sensitive dependencies (i.e.,) its TCB, both in the kernel and user-space.

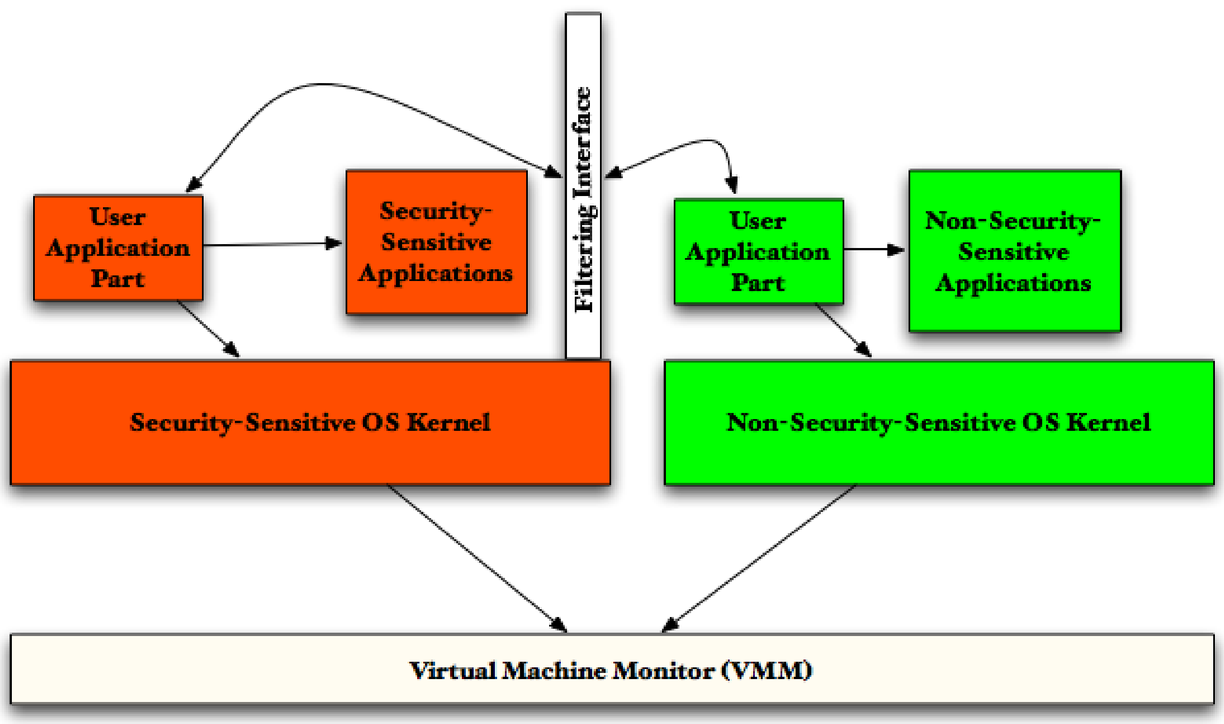






A Single Application's Dependencies

Partitioning Applications



- Our aim is to automate identification of the security-sensitive code of an application, with limited annotations from a user.
- Once this is done, we could partition an application and its dependencies into security-sensitive and non-security-sensitive parts.
- To isolate the security-sensitive part from the other part, we can have each part running in its own virtual machine (VM). Any non-security-sensitive operation can be forwarded to the other VM.
- Hence, compromise of the non-security-sensitive VM will not automatically lead to compromise of the security-sensitive VM.

Initial Results

- Runtime coverage analysis of the kernel by particular applications was studied. We call this the "slice of the kernel w.r.t an application".
- The program dhcpc (DHCP client) used only 4.9% of the kernel when run over 2 days on an active computer.
- *Coverage results for all the programs in the TCB of a typical Linux system came to around only 22.5% when run over 3 days.*
- These results suggest that the security-sensitive part of the kernel w.r.t an application will be less.

Next Steps

- Find ways of locating security-sensitive operations in a system given the characteristics of the application
- Automatically partitioning the application into a VM with a minimized kernel once its security-sensitive dependencies have been found
- Proving formal correctness of operations in the minimized system

Summary

- We aim to identify the security-sensitive parts of an application and methods to automatically partition an application and its dependencies into security-sensitive and non-security-sensitive parts
- Initial results suggest this might yield considerable benefit and the security-sensitive part may be small.

Work Sponsored by NSF