Confinement

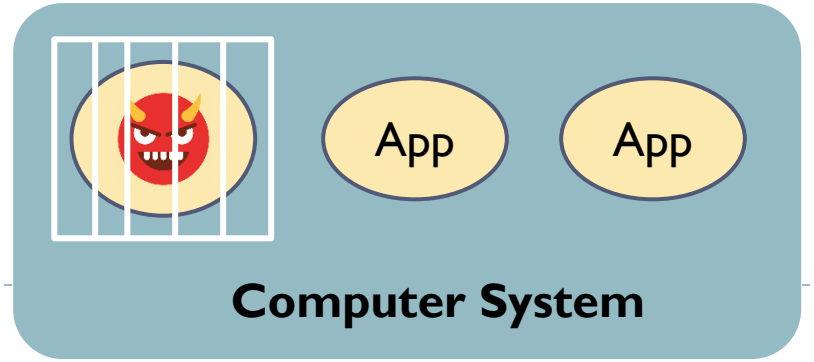
An important security strategy in OS protection

- When some component (e.g., application) in the system is compromised or malicious, we need to prevent it from harming the rest of system.
- Confinement: restricts the impact of each component on others.
- Follow the principle of least of privilege

Application scenario

- Cut off the propagation chain.
- Malware testing and analysis

Can be implemented at different levels



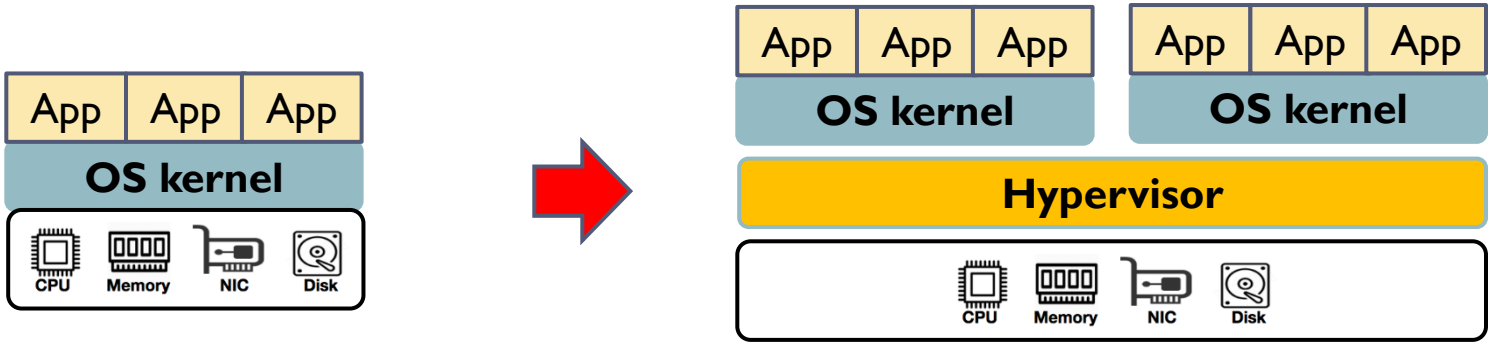
OS Level Confinement: Virtual Machine

Virtualization: the fundamental technology for cloud computing

- Different operating systems (virtual machines) run on the same machine
- Each virtual machine has an independent OS, logically isolated from others

Technical support

- Software layer: hypervisor or virtual machine monitor (VMM) for virtualizing and managing the underlying resources, and enforcing the isolation
- Hardware layer: hardware virtualization extensions (Intel VT-x, AMD-V) for accelerating virtualization and improving performance



Virtual Machine for Malware Analysis

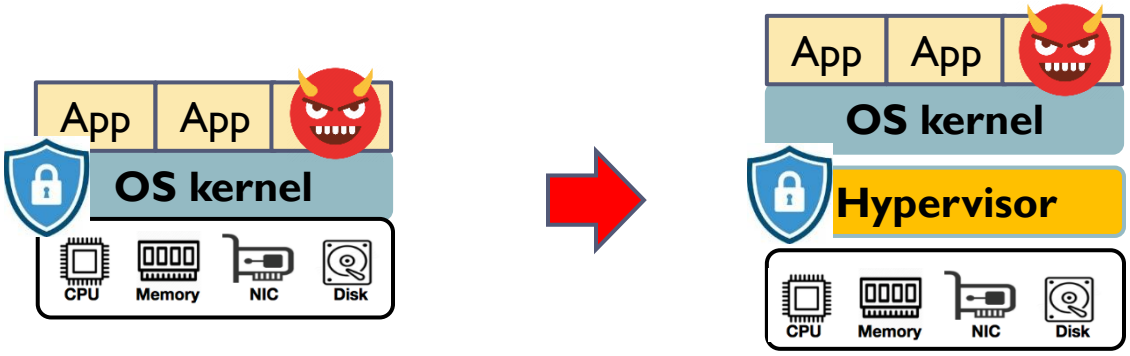
Malware analysis: deploy the malware and observe its behaviors.

Deploying the malware in the native OS

- The malware could compromise the entire OS (e.g., rootkit)
- The observation results are not reliable and could be manipulated.

Virtual machine: an ideal environment for testing malware

- The malware cannot cause damages outside of the VM
- The malware's behavior can be observed from the hypervisor/host OS



Limitations of Virtualization

The introduction of hypervisor can incur large attack surface

- The hypervisor has big code base, and inevitably brings more software bugs
- The hypervisor has higher privilege than the OS kernel. If it is compromised, then the attacker can take control of the entire system more easily.

The performance of a VM could be affected by other VMs due to the sharing of hardware resources.

Challenges of malware analysis with virtualization

- Although hypervisor has a complete view of VMs, there exists semantic gaps between high-level activities inside VMs and observed low-level behaviors
- This solution is not compatible with Trusted Execution Environment (TEE)
- A smart malware can detect that it is running inside a VM, not the actual environment, e.g., larger memory latency variance, reduced TLB size, etc. Then it behaves like normal applications,

Process Level Confinement: Container

A standard unit of software

- A container is a lightweight, standalone, executable software package that packages everything needed to run the application
  - Code, system tools and libraries, configurations.
- A Container Engine (e.g., Docker) is introduced to manage containers

Advantages of containers

- Portability: containers can run consistently across different environments, from development to production, reducing compatibility issues.
- Efficiency: sharing OS reduces overhead, with high resource utilization.
- Isolation: Applications operate in their own environment, minimizing conflicts and enhancing security.