Flexible Access Control for the Xen Hypervisor

Sandra Rueda, Joshua Schiffman, Hayawardh Vijayakumar, Trent Jaeger

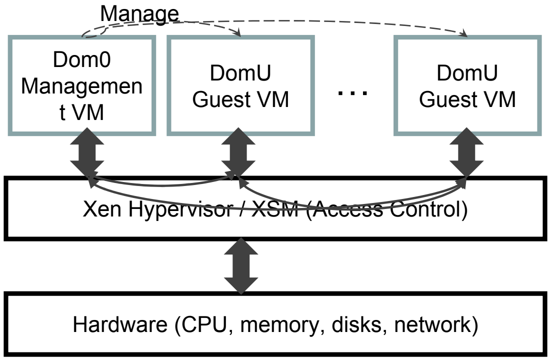
Virtualization, running many virtual machines (VMs) on a single physical machine, brings the challenge of effective isolation. Virtual machines need to be separated from each other, and their execution should be as close as possible to running singly on a physical machine. A mandatory access control (MAC) system is suitable for enforcing this separation. In a MAC system, the overall system policy is defined by the administrator, as contrasted with a discretionary access control (DAC) system in which users define policy over what they own. The Xen hypervisor has emerged as a powerful open source industry standard for virtualization. There has been recent development on a MAC system for Xen. We aim to aid the development of the system, to make it flexible, and demonstrate its applications.

Xen Security Modules

The Xen Security Modules (XSM) / Flask is a MAC system being built into Xen, similar to the Linux Security Modules (LSM) / SELinux for Linux.

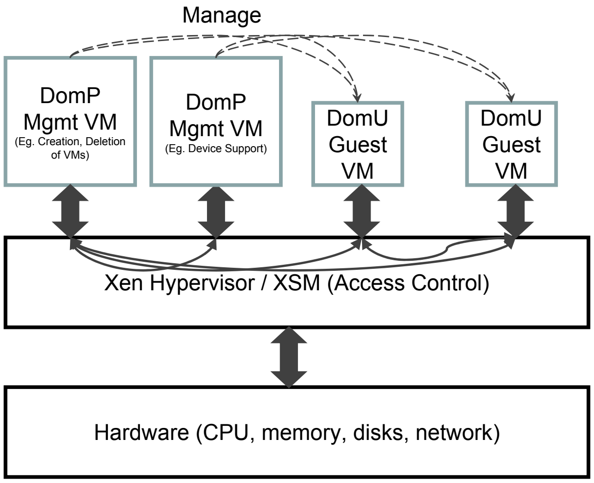
Just as SELinux allows us to specify what actions "subjects" (e.g. processes) can perform on "objects" (e.g. files), Flask allows us to specify what actions one VM can perform on another VM.

This allows an administrator to specify a set of allowable operations for each VM, in a policy. We look into both the XSM framework and analysis of the policy, and how to modify the policy for various applications.


Manage
Dom0 Managemen t VM
DomU Guest VM
. . .
DomU Guest VM
Xen Hypervisor / XSM (Access Control)
Hardware (CPU, memory, disks, network)

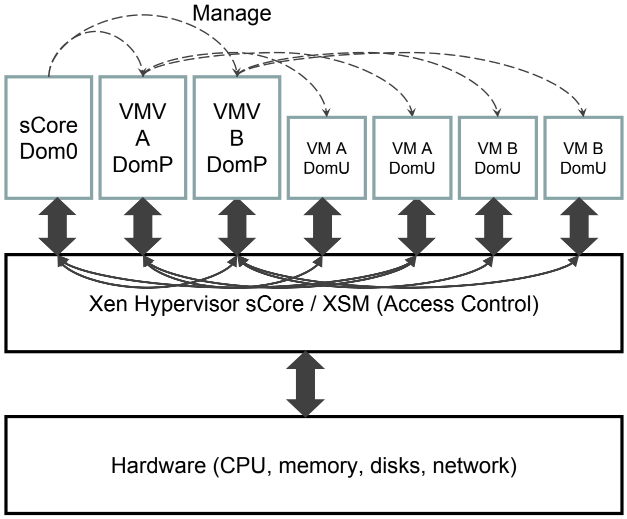
Extending Flexibility

In the current Xen hypervisor, a single domain (dom0) is all-powerful and responsible for management of other unprivileged domains (domUs). We aim to disaggregate the various functionalities of dom0 into separate VMs. To this end, we extend Xen to support privileged domains (domPs) , into which privileged functionality can be split. The policy can then be used to determine the functionality of various domPs. This gives a very flexible technique to define the functions of each domain.


Manage
DomP Mgmt VM
(Eg. Creation, Deletion of VMs)
DomP Mgmt VM
(Eg. Device Support)
DomU Guest VM
DomU Guest VM
Xen Hypervisor / XSM (Access Control)
Hardware (CPU, memory, disks, network)

Application

As an initial demonstration of flexibility of the system, we apply it to a framework for integrity measurement. A privileged domain, called a Virtual Machine Verifier (VMV), is used to attest the integrity of the VMs it manages. This system is run on a modification of Xen, sCore, that allows a root of trust installation. The policy is setup to allow VMVs to create and manage its set of VMs.


Manage
sCore Dom0
VMV A DomP
VMV B DomP
VM A DomU
VM A DomU
VM B DomU
VM B DomU
Xen Hypervisor sCore / XSM (Access Control)
Hardware (CPU, memory, disks, network)

Publication

Joshua Schiffman, Thomas Moyer, Hayawardh Vijayakumar, Patrick McDaniel and Trent Jaeger , "**Verifying Virtual Machine Integrity by Proxy"**, submitted for publication.

Work sponsored by Air Force Research Lab