

Xen: A bare metal hypervisor

PIYUSH RAI¹

¹School of Informatics and Computing, Bloomington, IN 47408, U.S.A.

*Corresponding authors: piyurai@iu.edu

+HID - S17-IO-3014

March 9, 2017

Xen is an open-source baremetal hypervisor. This paper explores it's overview, architecture and contains a brief note on its alternatives.

© 2017 <https://creativecommons.org/licenses/>. The authors verify that the text is not plagiarized.

Keywords: Baremetal hypervisor, Xen

<https://github.com/piyurai/sp17-i524/tree/master/paper1/S17-IO-3014/report.pdf>

INTRODUCTION

To provide an isolated environment to different applications in terms of memory, process scheduling and disk access such that one process does not impact the other, used to be a major challenge for system administration [1]. Also, different applications may have different OS requirements. Running an application within its own OS environment should protect it from other malicious applications because of the guaranteed memory and disk allocation to each guest VM (virtual machine).

Xen is a baremetal hypervisor and the only one to be available as open-source [2]. It's based on microkernel design with the advantage of small memory footprint and has a limited guest interface. It's responsibility is to manage CPU and memory, and to handle interrupts. Virtual machines are deployed in the guest domain called DomU which has no access privilege to hardware. A special virtual machine is deployed in the control domain called Domain 0. It contains hardware drivers and the toolstack to control the VMs and is the first VM to be deployed after the system comes up. The hypervisor is the first process to be started after bootloader.

ARCHITECTURAL OVERVIEW AND USE CASES

Xen has a small memory footprint and provides limited interface to guest VMs. It runs at the highest CPU privilege level while the guests are deployed in DomU domain [2]. It's primary responsibility is to manage CPU, memory and interrupts and does not contain the drivers for I/O functions such as networking and storage. The main device driver for a system can be run inside of a virtual machine. It protects the rest of the system from events such as driver crashes as the VM containing the driver can be rebooted independently. The control stack is generally deployed on a Linux VM running in domain 0. It is the first VM to be deployed by the system and can access the hardware directly. It handles system's I/O functions and provides interface to control the system. It's responsible for creating and configuring VMs,

allocating resources and monitoring them and terminating the VMs when they are no longer required. The control stack interface can also be driven through a cloud orchestration stack such as OpenStack. Figure 1 shows the architectural overview of Xen.

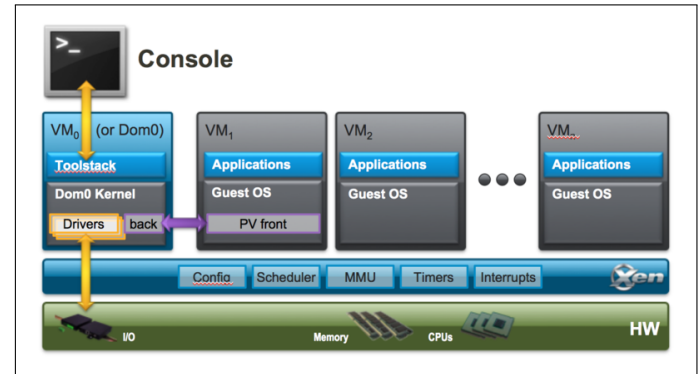


Fig. 1. Xen architecture overview [3].

The guest virtual machines are deployed in domain DomU which has no privilege to access the hardware. The guest VMs can either be deployed in Para virtualization (PV) mode or Hardware-assisted virtualization mode (HVM). It's also possible to use PV drivers inside of a HVM mode to improve its performance. The two types of modes can be deployed simultaneously on a single hypervisor. PV guests work without virtualization support from the hardware, and require PV-enabled kernel and drivers. The application binaries do not need any modification and run as in their native environments. HVM uses virtualization support available on the host CPU such as Intel VT or AMD-V hardware extensions. QEMU is used to emulate the remaining PC hardware. No changes to kernel are required for fully virtualized guests. HVM guests can also use PVHVM drivers i.e. special paravirtual device drivers optimized

for HVM environments to boost up the performance. PVH is a recently developed new virtualization mode where PV guests uses PV drivers for boot and I/O and hardware extensions otherwise. It is expected to simplify the Xen architecture and is the recommended virtualization mode to be used with newer Xen releases [4].

RESOURCES

The Xen Project wiki page presents an overview of the software architecture and host the links to documentation and release notes for the different releases, e.g. the list of feature for release 4.8 can be found at [5]. David Chisnall has also written a book explaining the hypervisor internals and its important features along with their interfaces.

ALTERNATIVES

[6] talks about the limitations during the early days of Xen development like modification to the guest kernels and limited Linux support. The article here [7] states about later years into the development of Xen, when it remained based on the older version of Linux kernel as the developers waited for more recent kernels with support for modern hardware. This allowed KVM to overtake as preferred choice for virtualization. KVM overcame the problems faced by Xen by using the virtualization support in the newly available hardware and reusing the components from the Linux kernel. This gave it the advantage of automatically benefiting from the developments in the mainline kernel. Its incorporation into the Linux version 2.6.20 and contribution from companies like AMD and Intel increased KVM's adaptability among the Linux distributors.

KVM requires hardware assisted virtualization with new releases containing paravirtualization support for certain devices [8]. Xen was originally developed to overcome the problem of limited support for hardware assisted virtualization [4] by using Paravirtualization. Xen is a layer between the kernel and hardware whereas KVM turns the kernel itself into the hypervisor. Xen has its own memory and power management system designed specifically for VMs whereas KVM needs to have support for processes as well.

In his blog [9] Brendan Gregg has listed the observations made during comparisons of different virtualization technologies at Joyent using DTrace. It analyzed the stack trace for different system functionalities such as network and I/O calls stack trace along with the relevant overhead for Zones, Xen and KVM. It mentioned about Zones, with Linux Containers being its equivalent, to be the preferred choice for virtualization for them as it has the lowest overhead for not having to run an entire OS for each application. They stopped using Xen in favor of KVM for provisioning fully virtualized guest machines.

CURRENT STATUS AND THE FUTURE

Xen is actively maintained by Linux Foundation under the trademark "XEN Project". Some of the features included in the latest releases include "Reboot-free Live Patching" (to enable application of security patches without rebooting the system) and KCONFIG support (compilation support to create a lighter version for requirements such as embedded systems) [10]. The overheads in terms of passing data across several layers is being minimized by use of shared memory transports and buffers [9]. Xen Project also strives to minimize the footprint of the change to Linux kernel. The codebase contains less than 150,000 lines

of code [2]. Xen Project hosts a listing of vendors that uses Xen Project software to allow them to advertise their software and services. Xen is used by more than 10 million users [11]. Some of the major cloud hosting service providers using Xen include Amazon and Rackspace. Hypervisors are predicted to become a core aspect of cloud based solutions. Containers are growing in popularity and there's an increasing interest in combination of isolation aspects of hypervisor and container runtime environments [12]. Bellani has written a comprehensive blog [11] about the ongoing work within Xen community towards the development of hypervisor-based containers.

REFERENCES

- [1] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the art of virtualization," in *Proceedings of the nineteenth ACM symposium on Operating systems principles*, ser. SOSP '03. New York, NY, USA: ACM, 2003, pp. 164–177. [Online]. Available: <http://dl.acm.org/citation.cfm?id=945462>
- [2] "Xen Project Software Overview," Web Page, Mar. 2016. [Online]. Available: https://wiki.xenproject.org/wiki/Xen_Project_Software_Overview
- [3] "Xen Architecture Diagram," Web Page, Apr. 2015. [Online]. Available: https://wiki.xenproject.org/wiki/File:Xen_Arch_Diagram.png
- [4] David Chisnall, "Xen PVH," Web Page, Jun. 2014. [Online]. Available: <http://www.informit.com/articles/article.aspx?p=2233978>
- [5] "Xen Project 4.8 Release Note," Web Page, Dec. 2016. [Online]. Available: <https://blog.xenproject.org/2016/12/07/whats-new-with-xen-project-hypervisor-4-8/>
- [6] Amit Shah, "Ten years of KVM," Web Page, Nov. 2016. [Online]. Available: <https://lwn.net/Articles/705160/>
- [7] Thorsten Leemhuis, "Rise of KVM," Web Page, Jun. 2011. [Online]. Available: <http://www.h-online.com/open/features/Xen-lets-KVM-overtake-1262171.html>
- [8] "KVM - Wikipedia," Web Page, Feb. 2017. [Online]. Available: https://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine
- [9] Brendan Gregg, "Virtualization Performance: Zones, KVM, Xen," Web Page, Jan. 2013. [Online]. Available: <http://dtrace.org/blogs/brendan/2013/01/11/virtualization-performance-zones-kvm-xen/>
- [10] "Xen Project 4.7 Feature List," Web Page, Jun. 2016. [Online]. Available: https://wiki.xenproject.org/wiki/Xen_Project_4.7_Feature_List
- [11] Stefano Stabellini, James Bulpin, "Hypervisor in 2017," Web Page, Dec. 2016. [Online]. Available: goo.gl/S3X1oQ
- [12] Stefano Stabellini, "Hypervisor-Based Containers," Web Page, Dec. 2016. [Online]. Available: <https://thenewstack.io/hypervisors-container-era/>