Poster: TwinHype: A Novel Approach to Reduce Cloud Downtime

Lei Lu
VMware Inc.
llei@vmware.com

Xing Gao
College of William and Mary
xinggao@cs.wm.edu

Jidong Xiao
Boise State University
jidongxiao@boisestate.edu

***Abstract*—When security vulnerabilities are discovered in hypervisors, cloud providers are facing a dilemma: patching hypervisors requires system rebooting or module reloading, thus incurs service downtime, on the other hand, not patching hypervisors incurs high security risks. To address this problem, we present a novel approach called TwinHype, which runs two hypervisors on the same physical machine. When upgrading, the virtual machines on one hypervisor can be migrated onto the other one to reduce service downtime. After the upgrading, the virtual machines can then be migrated back to the original hypervisor. Compared with traditional migration solutions, our approach has two advantages: first, the service downtime is highly reduced; second, the non-trivial network traffic in traditional migration solutions is eliminated.**

I. Introduction

As the foundation of cloud computing, virtualization technology has drawn plenty of attentions from both academia and industry. The common practice of virtualization is that hypervisor runs multiple virtual machines on top of the same physical machine. However, when hypervisors need to be upgraded, virtual machine users may need to tolerate a certain amount of downtime. It is unpleasant, but unfortunately, even the leading cloud providers, like Amazon, IBM, and Rackspace, have been reported to require their customers to experience non-negligible downtime [2]. In a recent survey, over 18% of IBM Softlayer customers reported a downtime of over one hour [6]. As a result, 29% of surveyed Softlayer customers claimed that they are considering other cloud providers [1], [6].

Thus far, even leading cloud providers have not come up with a reasonable solution to address such problem. One of the major reasons that hypervisors need to be upgraded is that, as hypervisors have become more complex than ever before, security vulnerabilities have become more likely to be introduced. In fact, in recent years, security vulnerabilities in various mainstream hypervisors have been discovered and reported on a regular basis [3], [5], [1]. When these vulnerabilities are reported, cloud providers have no choice other than fixing them by patching hypervisor software.

NDSS '16, 21-24 February 2016, San Diego, CA, USA
Copyright 2016 Internet Society, ISBN 1-891562-41-X
http://dx.doi.org/10.14722/ndss.2016.23xxx

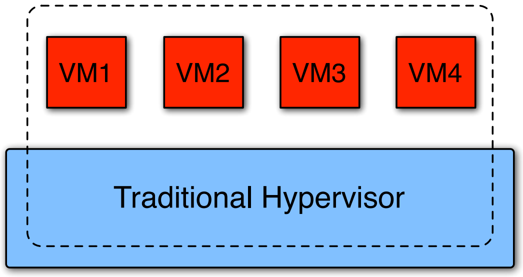

Fig. 1: Architecture of Traditional Hypervisor

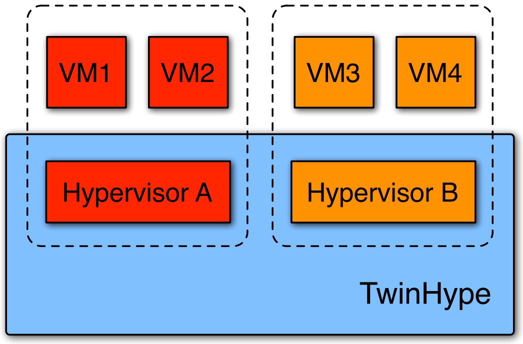

Fig. 2: Architecture of TwinHype

To address the aforementioned problem, we present TwinHype, which enables two hypervisor instances to run on top of the same physical machine. In this way, when patching one hypervisor, the affected virtual machines can be migrated to the second one. The downtime introduced by the upgrade could be reduced or even eliminated since both hypervisors are running on top of the same physical machine. When the upgrade is completed, the virtual machines can be migrated back to the original hypervisor. Compared to existing virtual machine migration solutions, another advantage of our approach is that the non-trivial network traffic incurred by existing virtual machine migration solutions [1], [4] could be avoided, as the result that the migration is occurring on the same physical machine.

II. Architecture and Implementation

Figure 2 shows the system structure of TwinHype. In the traditional virtualization model, as shown in Figure 1, there is only one hypervisor instance running on the physical machine. The hypervisor is responsible for partitioning the physical machine to make it act like multiple real machines. However, in the TwinHype model, there are two hypervisors – Hypervisor A and Hypervior B, and virtual machines can either run on top of Hypervisor A, or on top of Hypervisor B.

Our implementation is based on Linux Kernel-based Virtual Machine (KVM) hypervisor. In a typical KVM model, the hypervisor is implemented as a kernel module of the host Linux system. Each virtual machine is treated as a normal process, and is scheduled by the standard Linux scheduler. To enable two almost-identical KVM modules running on the same physical machine, we modify the original KVM code and extend it to be compatible with the original KVM implementation. Two major changes are:

- Normally, KVM exposes its virtualization capabilities via a character device `/dev/kvm`, and user space tools like QEMU can interact with the KVM module via its APIs to create and run virtual machines. To accommodate a second hypervisor, we modify the current KVM module code to create a separate device file `/dev/kvm2`, and also add a new commandline option for QEMU, so that it can run with the second KVM module and work collaboratively. Consequently, in the TwinHype model, `/dev/kvm` and `/dev/kvm2` serve as two different communication channels, and each of them will be used by its corresponding hypervisor and the user space daemon. In this way, the two hypervisors are isolated, and thus not interfere with each other.
- We ensure that no code dependency exists between the two hypervisors: KVM1 and KVM2. We verify the code to ensure that they can run independently with separate functions and variables, i.e., KVM1 does not make any use of functions and variables defined and exported by KVM2 and vice versa. In certain cases, when variables have to be shared, we define them in the core kernel code instead of exporting them within either KVM1 or KVM2. In this way, we can solve the dependency issue between KVM1 and KVM2, and thus at any time, we can load/unload either KVM1 or KVM2. Another benefit of this design is that, cloud providers can run one hypervisor in most of the time, and only launch the second hypervisor when upgrading. In other words, **even though we enable cloud providers running two hypervisors at the same time, running two hypervisors simultaneously is not required.**

III. Evaluation and Live Demonstration

We perform experiments on a testbed running Ubuntu 15.10 operating system with Linux kernel 4.3.3. Our testbed platform uses Dell T320 Server with Intel Xeon E5-2440, a 2.4GHz processor with 15MB of L3 cache supporting 6-cores plus HyperThreading. The server has a RAM of 24GB and 1TB of hard disk. Our experiments include the following steps: 1. We launch 2 virtual machines respectively on both Hypervisor A and Hypervisor B, and we call them VM1, VM2, VM3, VM4; 2. We try to patch Hypervisor A to fix a security vulnerability. To achieve this and avoid downtime, we migrate virtual machines VM1, VM2 to Hypervisor B, and then conduct the patching task; 3. We migrate VM1, VM2 , VM3, VM4 back to Hypervisor A, and patch Hypervisor B; 4. We migrate VM3, VM4 back to Hyperviosr B. After the experiments, both Hypervisor A and Hypervisor B are upgraded, with the security vulnerability fixed, yet none of the aforementioned virtual machines have experienced any downtime.

We will show a live demonstration of the upgrade procedure, including the patching and the migration procedure.

IV. Conclusion

We present TwinHype, a new solution for cloud providers, not only enabling cloud providers to upgrade hypervisors without incurring significant downtime to their customers, but also helping them avoid the non-trivial network traffic caused by traditional migration schemes. We believe that this new model will benefit cloud customers as well as cloud providers.

References

[1] N. Amit, D. Tsafrir, A. Schuster, A. Ayoub, and E. Shlomo, "Virtual cpu validation," in *Proceedings of the 25th Symposium on Operating Systems Principles (SOSP)*. ACM, 2015, pp. 311–327.

[2] B. Darrow, "Xen security issue prompts amazon, rackspace cloud reboots." https://gigaom.com/2015/02/27/xen-security-issue-prompts-amazon-rackspace-cloud-reboots/, 2015.

[3] N. Elhage, "Virtunoid: Breaking out of kvm," *Black Hat USA*, 2011.

[4] K.-Y. Hou, K. G. Shin, and J.-L. Sung, "Application-assisted live migration of virtual machines with java applications," in *Proceedings of the Tenth European Conference on Computer Systems (EuroSys)*. ACM, 2015, p. 15.

[5] K. Kortchinsky, "Cloudburst: A vmware guest to host escape story," *Black Hat USA*, 2009.

[6] K. Weins, "Xen bug drives cloud reboot: Survey shows users undeterred," http://www.rightscale.com/blog/cloudindustry- insights/xen-bug-drives-cloudreboot-survey-shows-users-undeterred, 2014.