CloudPhylactor: Harnessing Mandatory Access Control for Virtual Machine Introspection in Cloud Environments

Benjamin Taubmann, Noelle Rakotondravony, Hans P Reiser
University of Passau
{bt,nr,hr}@sec.uni-passau.de

December 16, 2015

Virtual machine introspection (VMI) can be used for several security relevant objectives such as intrusion detection, malware analysis or forensic evidence collection on virtual machines (VMs) [2]. However, current solutions that attempt to bring this technique to cloud environments have several drawbacks. CloudVMI introduces a significant overhead to the monitoring process and the monitored VM. LiveCloudInspector provides only a limited set of pre-defined commands to a cloud customer [1, 3]. Additionally, both approaches introduce a new threat to the cloud infrastructure as the monitoring tool is running in the most privileged VM. Thus, if an attacker is able to exploit flaws in the implementation of the monitoring framework, he can gain access to other VMs that run on the same cloud node.

The objective of this research is to use the mandatory access control system of Xen to grant dedicated monitoring virtual machines (MVMs) the permissions to access the main memory of other ones, so that they are able to perform VMI based analysis. Furthermore, we have to limit the monitoring access of the users so that for each customer, the instantiated MVMs can only inspect his own machines and not the VMs of other cloud users. As the policies cannot be modified during runtime, we also have to provide a way so that allows cloud customers to restrain the access of their own MVMs. For instance, a customer may choose that his production VMs be monitored by separated MVMs; and in case the MVM was subeverted, it would not be able to access other machines with more confidential information.

Our prototype implementation – CloudPhylactor – is based on Xen and OpenNebula and demonstrates that our approach does not introduce performance issues while giving cloud costumers means to do introspection on their VMs. This is mainly caused by the fact that a monitoring framework can directly access the Xen interface without any additional layer. We also show that this approach minimizes the impact of successful attacks that target VMI based monitoring frameworks. As the monitoring is running in a VM with limited permissions an attacker with access to a MVM is only able to access a small subset of VMs of one costumer that are running on the cloud node instead of all running VMs.

To sum up we can state that the approach of this work is a feasible way to provide VMI-as-a-Service in cloud environments so that customers are for instance able to use VMI based intrusion detection systems to secure their VMs.

References

[1] H. w. Baek, A. Srivastava, and J. V. d. Merwe. CloudVMI: Virtual Machine Introspection As a Cloud Service. In *Proc. of the 2014 IEEE Int. Conf. on Cloud Engineering*, 2014.

[2] T. Garfinkel and M. Rosenblum. A Virtual Machine Introspection Based Architecture for Intrusion Detection. In *Proceedings Network and Distributed Systems Security Symposium*, 2003.

[3] J. Zach and H. P. Reiser. LiveCloudInspector: Towards Integrated IaaS Forensics in the Cloud. In *Distributed Applications and Interoperable Systems*. 2015.