On Forensic Virtual Machines

Mohammad-Reza Memarian

21 Oct 2014

1 Introduction

Forensic Virtual Machine (FVM) is introduced in [1]. In the mentioned paper authors tried to define a new way for finding malicious images by searching for malicious symptoms instead of malicious behavior from an external source which is another VM in memory space of monitored VM using Virtual Machine Introspection. Each FVM is specialized for searching of one symptom. FVMs are required to be lightweight so numerous FVM can operate each one searching for one symptom. In [2], researchers used mini-OS to implement FVM. FVMs have read access to all other guest VMs. They communicate through a secure channel with each other. FVMs also communicate to a command and control center through Dom0. Command and control center compiles all the received information and takes the necessary remedial actions through Dom0 and hypervisour, such as freezing infected VM's memory or not allocating any CPU cycle to the infected VM. Each FVM spend a predefined time on each VM that is specified by a TTL. When TTL is expired, FVM moves to other VM. Each FVM contains a copy of a mobility algorithm which defines the next VM to be monitored. The design proposed in [1] and [2] is depicted in figure 1.

2 Problem Statement

There are three problems with approaches in [1] and [2]. These problems can be found in the following:

1. As number of symptoms that must be checked increases, number of FVMs should increase too. Apart from matter of resource consumption and scheduling, the TCB of each VM increases. This means that when number of FVMs increase, each guest VM must trust more FVMs.

2. Each FVM checks the Memory of each guest VM and when it's TTL is expired, moves to next VM. There is probability that while FVM X is looking for symptom X in VM A, that symptom does not appear in that time. There is probability that until the time FVM X goes back to VM A to look for the same symptom, that symptom is not anymore available. So detection of malicious

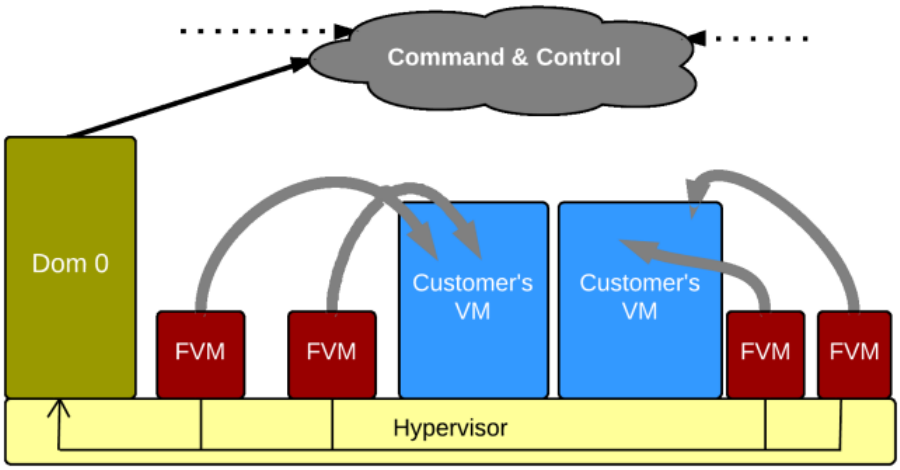

Figure 1: Design of solution is related works

symptoms can not take place as soon as they appear.

3. FVMs exchange messages via secure multi-cast channels to share information about the discovery of symptoms within the VMs. Although this communication is done through a secure channel, but it can expand attack surface.

3 Probable solution

The probable proposed solutions is depicted in Figure 2. As 3 problems mentioned in the previous section, the probable solutions can be as following:

1. Instead of having one FVM per symptom, we can have one FVM per VM. In this case no matter how many symptoms we are going to look for, number of FVMs has direct relationship with number of VMs. As an example, if we have 200 symptoms, with 40 VMs, we do not need to have 200 FVMs but we can have 40 FVM and each FVM must look for all symptoms in only one VM. In this way TCB of each VM reduces dramatically as each VM must only trust one FVM instead of 200 FVM.

2. As each FVM looks into only one VM and constantly, the second problem can be tackled as FVMs do not move to other VMs. Also Mobility algorithms can be removed from FVMs which makes them lighter.

3. Each FVM decides locally what to do in case of symptom detection and

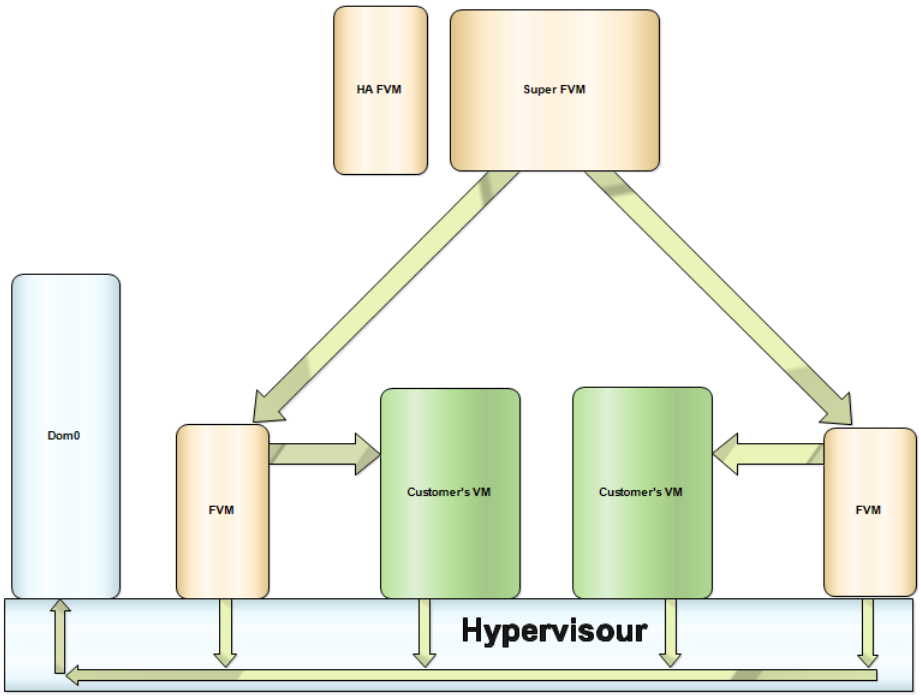

Figure 2: Design of solution is related works

there is no need for communication with other FVM. This way the risk of Information leakage in reduced. In this way Command and Control center proposed in [2] can be deleted from the design.

Also in our design we added a component called super FVM and HA-FVM. Super FVM checks memory of other FVMs only. In case that either malicious behavior or any malicious symptoms is detected that shows any FVM is compromised or by any mean that FVM becomes unavailable, The compromised FVM goes out of the cycle and HA-FVM is assigned to the monitored guest VM instead of the previous FVM.

References

[1] S. T. A. C. I. A. N. Keith Harrison, Behzad Bordbar, "A framework for detecting malware in cloud by identifying symptoms," in *2012 IEEE 16th international Enterprise Distributed object computing conference*.

[2] J. S. K. H. C. I. D. Adrian L. Shaw, Behzad Bordbar, "Forensic virtual

machine: Dynamic defence in the cloud via introspection," in *2014 IEEE International Conference on Cloud Engineering*.