End to End Defense against Rootkits in Cloud Environment

Design- Part 1

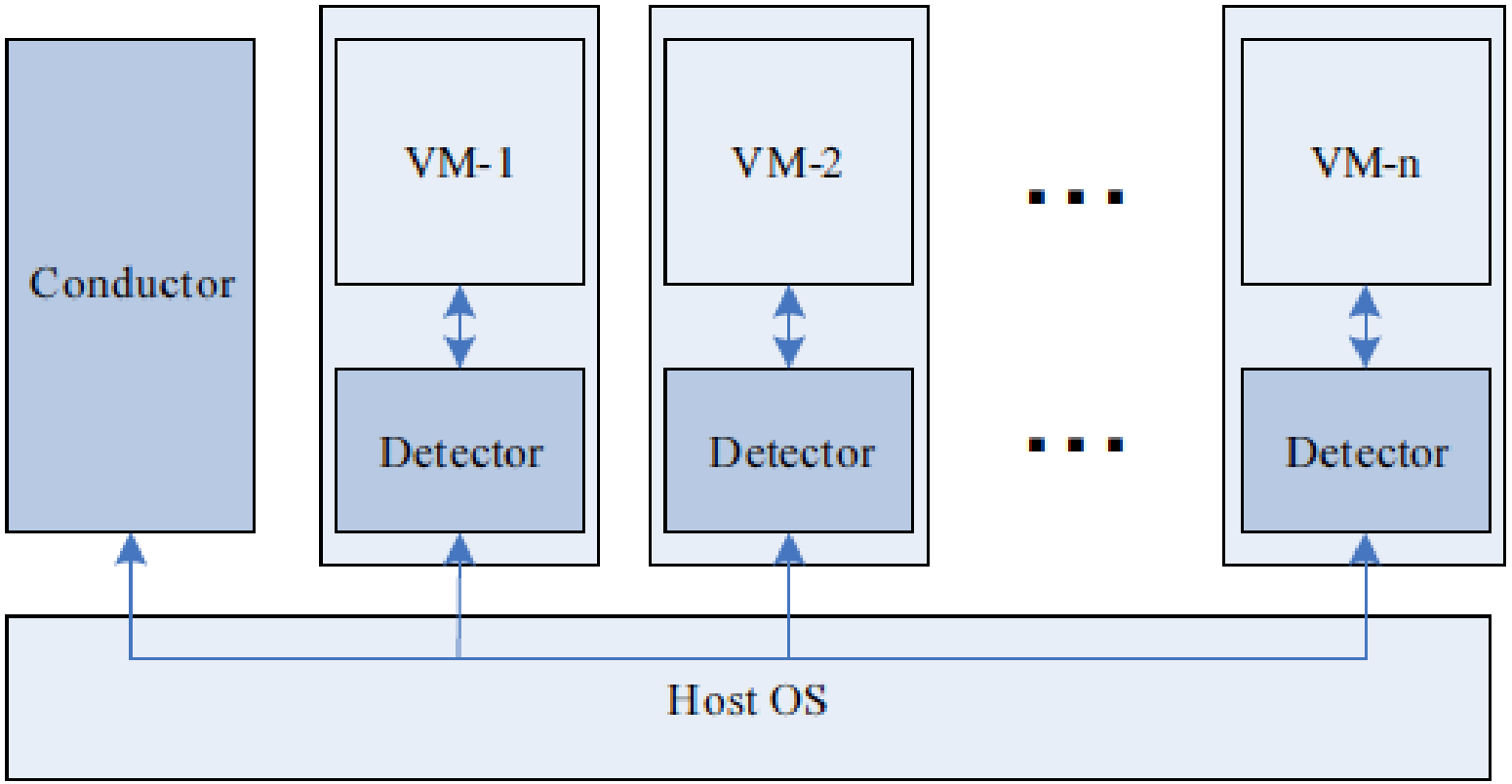
Sachin Shetty

Associate Professor
Electrical and Computer Engineering
Director, Cybersecurity Laboratory
Tennessee State University

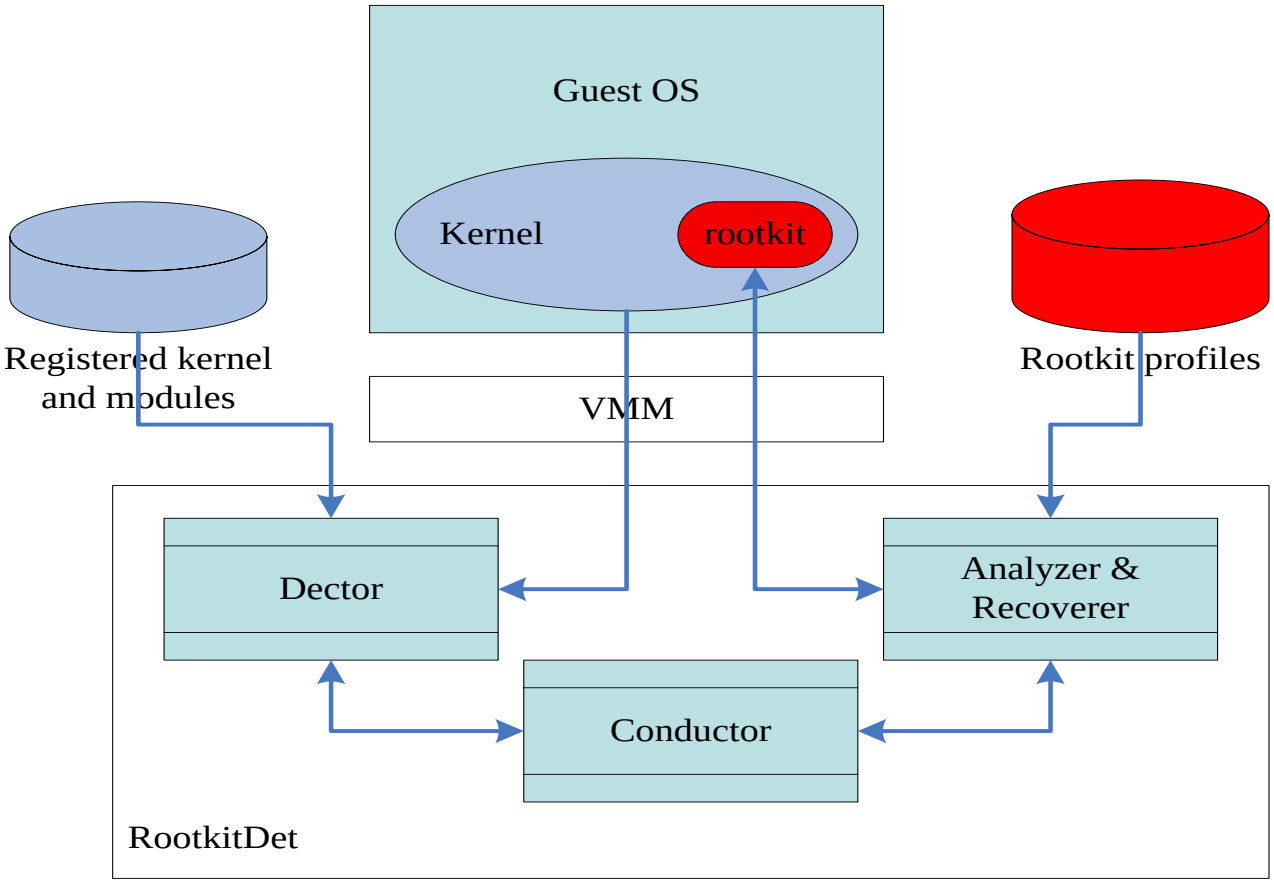
RootkitDet Overview

- Detection
  - Registration procedures indicate legitimate code of guest OSes
  - Detection procedures find out suspicious code in the kernel space
- Diagnosis
  - Perform static analysis on the code of rootkits to collect characteristic information
  - Categorize by matching to profiles of known rootkits
- Recovery

RootKitDet Overview



RootkitDet: Basic Architecture



RootKitDet- Detection

- First step in the RootkitDet system is to detect the kernel-level rootkits installed into the guest OSes.
- RootkitDet system identifies suspicious code, which is taken as the code of rootkits, in the kernel space of guest OSes.
- By ”suspicious”, we mean a memory region that is not supposed to hold any code or a region that holds illegitimate code.
- To separate the code of rootkits from legitimate code, we introduce a simple, practical and effective registration procedure .

RootKitDet- Registration

- Registration procedure allows administrator of a guest OS to register the kernel and potential LKMs of the guestOS in advance.
- Registration of the kernel provides enough information to bridge semantic gap in our system.
- Registration information includes source code configuration file, system. map as well as the binary file of the kernel.
- The kernel of a guest OS should be registered prior to the execution of the virtual machine which the guest OS runs on.

RootKitDet: Registration

- Registration of LKMs is critical to separating legitimate code and the rootkits.
- To register a LKM that is probable to be loaded during the lifetime of the guest OS, the administrator should provide the module's name and object file.
- A module should be registered before it is loaded into the kernel, even if the guest OS is running.
- We suppose that registration procedure is performed through a secure channel, which is unknown to the attacker.

RootKitDet: Detection

- To detect suspicious code in the kernel space of a guest OS, RootkitDet system reconstructs the page directory of the kernel space of the guest OS, identifies all executable regions and compares them with expected executable regions which hold legitimate code.
- Detects whether extra executable regions exist in the kernel space.
  - Extra regions are different from that holds legitimate code
  - Detects whether some code resides in unused space of modules.
  - Detects malicious modifications to the legitimate code by computing SHA-1 checksums of the legitimate code
  - Any mismatch means that legitimate code is modified by the rootkits.

RootKitDet: Diagnosis

- Diagnosis involves categorizing the detected rootkits and precisely identify the objects and data structures that are modified by the rootkit.
- Generate profiles of known typical rootkits in advance.
- RootkitDet system performs static analysis on the code of the detected rootkit to collect characteristic information, which is used to categorize the rootkit by matching with the profiles of known typical rootkits.

RootKitDet: Profile

- Tactic adopted by the rootkit to achieve its intention.
- We describe the tactic by a set of semantic actions, including external function calls, access to global variables and dynamic allocated data structures.
- The data structures that we should recover according to its tactic.
- In general, these data structures are dynamically allocated but we can find its location tracking down from a global variable with fixed location.

RootKitDet: Recovery

- Recover the objects and data structures that were modified by the rootkit.
- Rootkit may make modifications to control data and non-control data.
  - Control data are usually function pointers
  - Expected values of control data are already known
  - Modifications to non-control data are various and usually there are no expected values for them.
- Some modifications to non-control data break the links to other objects or violate some invariant that keeps in uninfected kernel.
- We can figure out how to recover such modifications in the kernel's context.

RootKitDet Components

- RootkitDet comprises several components:
  - Registration
  - Conductor
  - Detector
  - Analyzer
  - Inspector.
- All components except inspector are independent of the hypervisor, and run in a different OS running on a virtual machine or a physical machine

RootKitDet: Inspector

- Inspector is integrated into the hypervisor to provide a reliable interface to access the kernel space memory and CPU registers of guest OS.
  - Used by detector and analyzer.
- Reading or writing the memory of the guest OSes does not require stopping the OS because our system accesses unusually changed memory during detection and recovery procedures in most of the time.
- Inspector is easily developed in most cloud platforms due to its simplicity

RootKitDet: Detector

- Detector performs three detection procedures to find out whether kernel-level rootkits exists in guest OS according to the commands coming from
- the conductor.
- In detection procedure 1, detector reconstructs the list of loaded modules and generates the list of executable regions in the kernel space, then compares them to find out whether extra executable regions exist besides the
- regions of the kernel code and registered modules.

RootKitDet: Detector

- In detection procedure 2, detector checks whether some code resides in the unused space of each module.
- In detection procedure 3, detector calculates checksums for the code of the kernel and modules, and compares them with original ones, which are provided by the conductor, to check integrity of the legitimate code in the kernel space.

RootKitDet: Detector

- Detection procedure 1 and 2 might be bypassed because detector depends on the memory of guest OSes, which might be under the control of rootkits.
- For instance, a rootkit may tamper with the information of a module and change the module's code size to a bigger value, and put its code right behind the module's code, pretending itself as part of the module to escape from detection.
- We leave this problem to the conductor and the conductor resolves it when generating the original hash values for all of the modules.

RootKitDet: Conductor

- Conductor is the heart of our system.
- It periodically sends commands to detector to start detection procedures when the guest OS is running.
- Once rootkits are detected, it receives the detection report from detector, then raises an alert to the administrator and activates analyzer.
- Conductor also helps detector during detection procedure 3 by generating original checksums of the loaded modules of the guest OS as well as descriptions of each module, which are used to detect smart rootkits that escape from procedure 1 and 2.

RootKitDet: Registration

- Registration component stores information of the guest Oses provided by the administrator in registration procedure.
- It provides information of the kernel to bridge semantic gap in the three steps of RootkitDet system.
- Besides, it provides the necessary information of the kernel and legitimate modules to help RootkitDet system separate rootkits.

RootKitDet: Analyzer

- Analyzer diagnoses the code of the detected rootkit by performing static analysis to collect related characteristic information and attempts to categorize the detected rootkit heuristically.
- If the analyzer succeeds in categorizing the rootkit by matching the characteristic information with the profiles of known rootkits, it can finally perform recovery of the guest OSes.
- The analyzer performs static analysis instead of dynamic analysis due to the following reasons.

RootKitDet: Analyzer

- First, dynamic analysis is not applicable in practice due to its heavy overhead to guest OSes.
- Second, dynamic analysis requires the execution of the code of rootkits to analyze its behavior while static analysis does not.
- Finally, the characteristic information collected through static analysis is enough in most cases although it is sketchy and rough.