**Introduction**

Virtual Machines simulate a machine (abstract or real) that is typically different from the target machine it is being simulated on. Virtual machines can emulate the [computer architecture](http://en.wikipedia.org/wiki/Computer_architecture) and functions of a real world computer without extra hardware. This power and convenience has caused virtual machine applications to become increasingly more popular. This virtualization, however, may bring some new security risks to the table. Rootkits are a type of malicious software that is activated before the operating system each time a system is booted up. With the rise of virtual machine usage, virtual machine based rootkits have also come into play. Virtual machine rootkits work by dropping a virtual machine monitor (VMM) underneath an OS installation. Virtual machine rootkits, like regular rootkits, may install hidden files, processes, user accounts, or other malicious data on a system. However, it does this on a separate OS, making it virtually invisible to the target OS and its anti-malware applications. This paper will evaluate some existing virtual machine rootkits and proposed defenses against them. It will also provide new insight on tactics for virtual machine rootkit defense.

**Installation**

1. Obtain root privileges/access to kernel mode
2. Install virtual machine rootkit on disk
3. Modify system boot records to ensure rootkit runs before the target OS
   1. This can be done during final stages of shutdown (when modifications to hard disk’s boot blocks won’t be detected by anti-malware applications)
   2. LastChanceShutdown Notification on Windows XP
   3. Modify shut-down scripts so installation runs after all other processes are killed
4. Target system’s disk space now contained in virtual disk, VMM translates address to equivalent location on physical disk 🡪 Target System no longer has access to physical disk and rootkit has control of the system

**Forms of Attack**

1. No interaction with target system
   1. Ex. Spam relays, denial of service zombies, phishing web servers
2. Target system observers
   1. Logging of hardware-level data via modification of VMM’s device emulation software
   2. Virtual machine introspection
3. Target system modifiers
   1. Ex. Modifying network communication, e-mails, or target applications
   2. Virtual machine introspection

The virtual machine rootkit is vulnerable during a short period of time in-between powering up the system and when the virtual machine is started. The rootkit can minimize this vulnerability by emulating a system shutdown while it remains in a running state.

**Forms of Defense**

1. Run detection below the virtual machine rootkit, out of its control
   1. Secure hardware (Intel’s LaGrande, AMD’s platform for trustworthy computing, Copilot) allow for development of low-layer security software
   2. Boot from a safe medium, i.e. CD\_ROM, USB drive, or network boot server
      1. Note: device must be completely powered off first to avoid rootkit shutdown emulation
   3. Use secure VMM, one which gains control of system before operating system boots. This could perform a check to stop a rootkit from modifying the systems boot sequence.
      1. “Using a secure VMM, we implemented an enhanced version of secure boot which can prevent VMBR installations. The goal of our secure boot system is to provide attestation for existing boot components, such as the disk’s master boot record, the file system’s boot sector, and the OS’s boot loader and also to allow legitimate updates of these components. All attempted updates of these components are verified (by checking the cryptographic signature) before they are allowed to complete. The verification code resides in a separate virtual machine, so it is protected from malicious code running within the guest. We implement this secure boot system using a Virtual PC VMM and a Windows XP guest operating system.” [king06.pdf]

\* The methods described above are often unrealistic and not practiced frequently enough

1. Run detection within virtual machine on target system (if possible)
   1. Look for excess CPU overhead using clock that can’t be manipulated by rootkit
   2. Look for excess memory usage (although this can be masked by the rootkit)