Spyware & Rootkits

References

- [1] Copilot –A coprocessor based kernel runtime integrity Monitor by Petroni et al, Usenix Security 2004
- [2]Fast User-mode rootkit scanner for the Enterprise by Wang & Beck, Usenix LISA 2005

Some Definitions --Spyware

- Spyware: software that either spies on the user activities or benefits a third party
  - Unsolicited pop-up advertisements
  - Tracking user behavior for marketing purposes
  - Theft of personal information
  - Routing http traffic to advertisement sites

Malware

- More general –any malicious software
  - Trojans
  - Worms
  - Viruses
  - Spyware
  - Rootkits

Rootkit

- Stealth malware
  - with root permission
  - modifies the kernel to keep its activities secret or hidden
  - Typically hides files, processes, logins, network connections

Kernel level Rootkits

- Run inside the kernel
- Have access to kernel data structures
- Loadable Kernel Modules or Device Drivers
- Patch, hook or replace system calls

User level Rootkits

- Run as a user program
- Modify kernel utilities or APIs
  - Typically, resource enumeration APIs
  - Ps, ls, netstat in Linux/Unix
  - Registry entries (RegEnumValue) and process enumeration (QuerySystemInformation) APIs in Windows

User level rootkit detection

- Run the existing utilities
- Gives the view from modified/compromised tools
- Run the same utilities in a safe mode
  - Either from a CD or a read-only version of the tools
- Compare the two versions

User level rootkit detection

- The “delta” identifies the rootkits
- Turn the “stealth” of rootkits against them
- Multiple views of resources from different points
  - But, at the same point in time
- Could also generate multiple views over time through snapshots

Rootkit Detection

- Time-based diffs. Used in
  - Tripwire (1994), Strider (2003)
- More general
  - Detect hiding and non-hiding changes
- Cross-time differences can generate false positives
  - Legitimate modifications to O/S

Rootkit Detection

- Detect API interceptions
  - Employed in a number of tools
  - Can only detect changes to monitored APIs
  - Software-patching, security wrappers, fault-tolerant wrappers trigger false positives

Linux rootkits[1]

rootkit name:	loads via:	overwrites syscall jump	adds new syscall jump	modifies kernel text	adds hook to /proc	adds inet protocol
Complete rootkits:						
adore 0.42	LKM	x				
knark 2.4.3	LKM	x			x	x
rial	LKM	x				
rkit 1.01	LKM	x				
SucKIT 1.3b	kmem	x	x			
synapsys 0.4	LKM	x				
Demonstrates module or process hiding only:						
modhide1	LKM	x				
phantasmagoria	LKM			x		
phide	LKM	x				
Demonstrates privilege escalation backdoor only:						
kbd 3.0	LKM	x				
taskigt	LKM				x	
Demonstrates key logging only:						
Linspy v2beta2	LKM	x				

Table 1: Features of example Linux kernel-modifying rootkits

Rootkit mechanisms

- Use LKM or /dev/kmem interfaces
- Modify the system call table addresses to point to compromised or wrapped system calls
  - System call interposition
- Add new system calls to systemcall table
  - SucKIT rootkit

Rootkit mechanisms

- Add additional instructions to the system call routines
  - Phantasmagoria rootkit
- Add hooks to /proc file system
  - Knark, taskigt
- Register new inet protocol handlers
  - Knark –allows kernel level access when certain packets received

Rootkit Detectors

rootkit detector:	Kernel memory access /dev/kmem	Kernel memory access detector LKM	synchronous detection	user-space symptom detection
KSTAT	x	x		x
St. Michael		x	x	
Carbonite		x		
Samhain	x			x
chkrootkit				x
checkps				x
Rkscan				x
RootCheck				x
Rootkit Hunter				x

Table 2: kernel-modifying rootkit detector mechanisms

Rootkit Detectors

- Signature based
- Look for specific files, processes and other modifications of known rootkits
  - Chkrootkit
- Works well for known problems

Rootkit Detectors

- Check /proc file system entries
- Check /proc/ksyms for symbols exported by rootkits
- Employ multiple view differencing approach
- These work for some rootkits

Coprocessor based Detectors

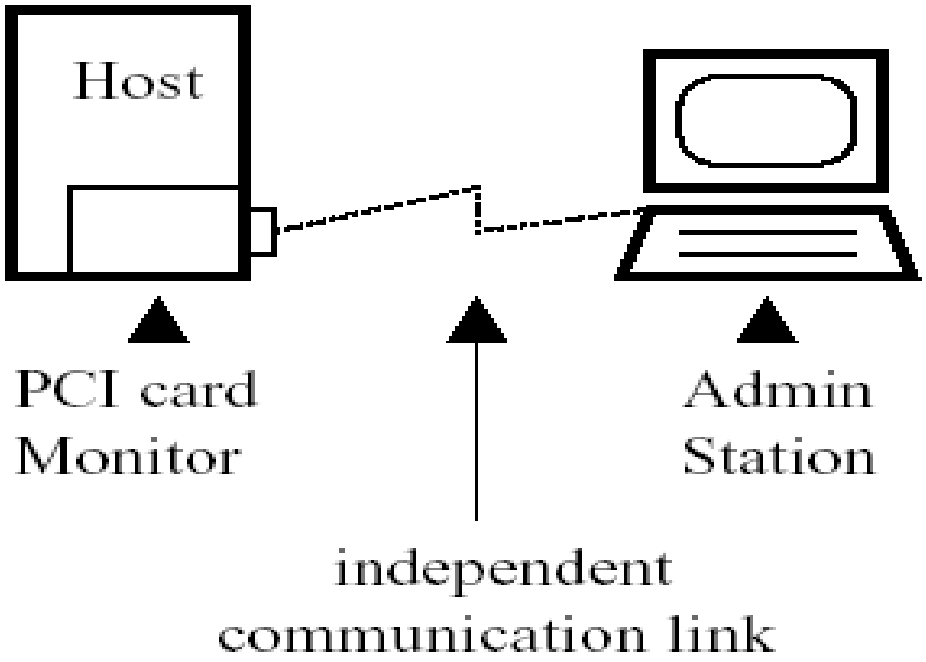

Figure 1: Copilot monitor prototype testbed architecture

Coprocessor Requirements

- Unrestricted memory access
  - Access full range of physical memory
- Transparency
  - Should not impact host processor
- Independence
  - Should not depend on host processor for accessing resources

Coprocessor Requirements

- Sufficient processing power
  - Employ hashing and encryption checks
- Sufficient memory resources
  - Keep baseline state for comparison
- Out-of-band communication
  - Need to report intrusions to admins

Coprocessor

- PCI Bus master card
- DMA access
- Virtual memory address translation

Monitored symbols/areas

symbol	use
_text	beginning of kernel text
_etext	end of kernel text
sys_call_table	kernel's system call table
swapper_pg_dir	kernel's Page Global Directory
idt_table	kernel's Interrupt Descriptor Table
modules	head of kernel's LKM list

Table 4: Symbols taken from System.map

Virtual memory translation

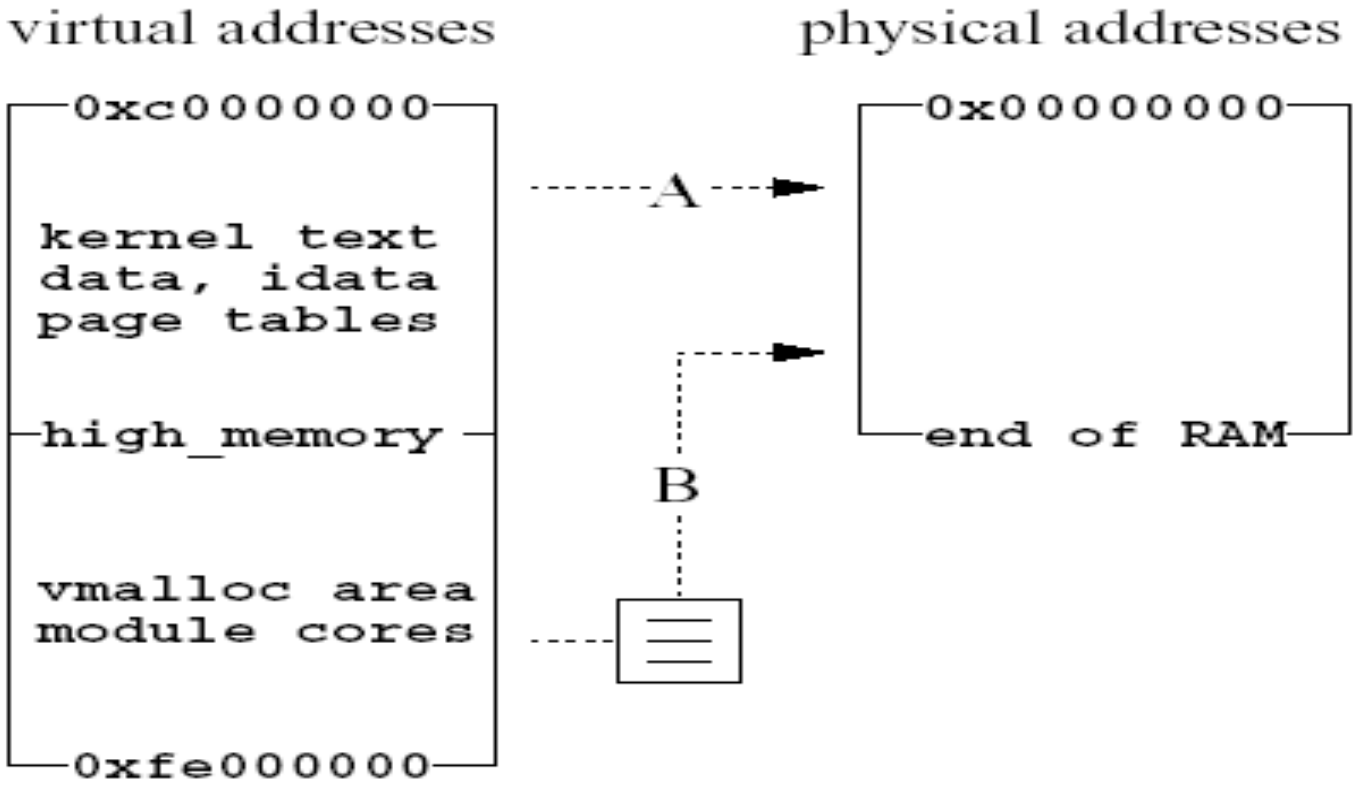

Figure 2: Virtual address translation

Virtual Memory Translation

- Linux uses linear mapping of virtual addresses oxC0000000 to oxC0000000+size of physical RAM
- Page table and other structures within these virtual addresses
  - Can locate them in physical memory easily
  - Then use page tables to do translation

Summary of Today's class

- Rootkits are stealth malware
  - Try to stay hidden
  - Could potentially be trojans
- Employ system call interposition, system call modifications, new system calls and other mechanisms

Summary of Today's class

- User level rootkits are easier to detect –employ multiple views in time or from different points
- Root level rootkits are harder to remove
  - Require many advanced kernel level mechanisms