Software Defenses (part 2)
Virtualization and Malware

CS 645 Rachel Greenstadt
July 17, 2013
Virtualization and Security

- Background on virtualization
- Virtualization to protect software/platforms from malware
- Virtualization to detect malware
- Virtualized malware
What is virtualization?
Abstraction of computational resources

WITHOUTVIRTUALIZATION

WITHVIRTUALIZATION

VMM= Layer of system software
• Enables multiple OS’s to share hardware
• Allows OS & Apps to run without modifications

New hardware capabilities to facilitate virtualization
Examples of virtualization

- Emulation – translate all instructions
  - QEMU, Bochs
- Paravirtualization – requires changes to OS
  - UML, Xen
- Native virtualization – require same architecture, most instructions executed natively
  - VMWare, virtual box
  - More efficient due to hardware support (Intel-VT)
- OS-Level virtualization – same OS, isolate resources
  - VServer, iOS
Virtualization and Security

- Isolate and protect software from malicious programs (malware)
  - Sandbox programs in own address space
  - Mediate with lean, secure VMM

- Detect malware
  - Use emulation to run malware in a safe environment
  - Stop, move, replay events in virtual machines

- Virtualized malware
  - Rootkits that hide underneath OS
Virtualization for Isolation

- Presenting virtual interfaces to resources can limit and control the access of malicious programs to resources
- A Virtual Machine Monitor (VMM) can do more rigorous access control
Isolation

- Current applications have no isolation
- Solitaire can access network, camera, user documents, passwords etc
- Programs execute in virtual machines (or sandboxes or jails) that can only see its own files/processes
- Programs must request additional permissions from the VMM
One Laptop Per Child

• OLPC project aims to give all children laptops

• Cannot demand 6 year old children secure their machines

• Need way to allow them to get hacked and still be ok

• Answer: virtualization

• Apple takes a similar approach with iOS
Thin Hypervisor

- Existing OS kernels too big and complex to avoid vulnerability
- Small hypervisor does scheduling, isolation
- Parent partition manages permissions
- Keep hypervisor secure
- Push dangerous apps into virtual machines
Malware Detection with Virtualization

- Many properties of virtualization useful for detecting malicious software
  - Tamper resistance
  - Safety/Isolation
  - Instrumentation
  - Rewind/replay properties
What is Malware?

- Malware (malicious software) is any program that works against the interest of the system’s user or owner.
- **Question**: Is a program that spies on the web browsing habits of the employees of a company considered malware?
- What if the CEO authorized the installation of the spying program?
Typical purposes of Malware

• Backdoor access:
  – Attacker gains unlimited access to the machine.

• Denial-of-service (DoS) attacks:
  – Infect a huge number of machines to try simultaneously to connect to a target server in hope of overwhelming it and making it crash.

• Vandalism:
  – E.g., defacing a web site.

• Resource Theft:
  – E.g., stealing other user’s computing and network resources, such as using your neighbors’ Wireless Network.

• Information Theft:
  – E.g., stealing other user’s credit card numbers.
Types of Malware

- Viruses
- Worms
- Trojan Horses
- Backdoors
- Mobile code
- Adware
- Sticky software
Viruses

• Viruses are self-replicating programs that usually have a malicious intent.
• Old fashioned type of malware that has become less popular since the widespread use of the Internet.
• The unique aspect of computer viruses is their ability to self-replicate.
• However, someone (e.g., user) must execute them in order for them to propagate.
Viruses (Cont’d)

• Some viruses are harmful (e.g.,):
  – delete valuable information from a computer’s disk,
  – freeze the computer.

• Other viruses are harmless (e.g.,):
  – display annoying messages to attract user attention,
  – just replicate themselves.
Viruses: Operation

• Viruses typically attach themselves to executable program files
  – e.g., .exe files in MS Windows
• Then the virus slowly duplicates itself into many executable files on the infected system.
• Viruses require human intervention to replicate.
Origin of the term computer virus

• The term computer virus was first used in an academic publication by Fred Cohen in his 1984 paper Experiments with Computer Viruses.

• However, a mid-1970s science fiction novel by David Gerrold, When H.A.R.L.I.E. was One, includes a description of a fictional computer program called VIRUS.

• John Brunner's 1975 novel The Shockwave Rider describes programs known as tapeworms which spread through a network for deleting data.

• The term computer virus also appears in the comic book Uncanny X-Men in 1982.
The first computer viruses

- A program called *Elk Cloner* is credited with being the first computer virus to appear "in the wild". Written in 1982 by Rich Skrenta, it attached itself to the Apple DOS 3.3 operating system and spread by floppy disk.
- The first PC virus was a boot sector virus called (c)Brain, created in 1986 by two brothers, Basit and Amjad Farooq Alvi, operating out of Lahore, Pakistan.
Worms

• Worms are malicious programs that use the Internet to spread.
• Similar to a virus, a worm self-replicates.
• Unlike a virus, a worm does not need human intervention to replicate.
• Worms have the ability to spread uncontrollably in a very brief period of time.
  – Almost every computer system in the world is attached to the same network.
Worms: Operation

• A worm may spread because of a software vulnerability exploit:
  – Takes advantage of the OS or an application program with program vulnerabilities that allow it to hide in a seemingly innocent data packet.

• A worm may also spread via e-mail.
  – Mass mailing worms scan the user’s contact list and mail themselves to every contact on such a list.
  – In most cases the user must open an attachment to trigger the spreading of the worm (more like a virus).
Trojan horses

• A Trojan Horse is a seemingly innocent application that contains malicious code that is hidden somewhere inside it.
• Trojans are often useful programs that have unnoticeable, yet harmful, side effects.
Trojan horses: Operation (1)

- Embed a malicious element inside an otherwise benign program.
- The victim:
  1. receives the infected program,
  2. launches it,
  3. remains oblivious of the fact that the system has been infected.
  
  – The application continues to operate normally to eliminate any suspicion.
Trojan horses: Operation (2)

• Fool users into believing that a file containing a malicious program is really an innocent file such as a video clip or an image.

• This is easy to do on MS Windows because file types are determined by their extension as opposed to examining the file headers.

• E.g.,
  – “A Great Picture.jpg .exe”
  – The .exe might not be visible in the browser.
  – The Trojan author can create a picture icon that is the default icon of MS Windows for .jpg files.
Backdoors

• A backdoor is malware that creates a covert access channel that the attacker can use for:
  – connecting,
  – controlling,
  – spying,
  – or otherwise interacting with the victim’s system.
Backdoors: Operation

• Backdoors can be embedded in actual programs that, when executed, enable the attacker to connect to and to use the system remotely.

• Backdoors may be planted into the source code by rogue software developers before the product is released.
  – This is more difficult to get away with if the program is open source.
Mobile code

- Mobile code is a class of benign programs that are:
  - meant to be mobile,
  - meant to be executed on a large number of systems,
  - not meant to be installed explicitly by end users.

- Most mobile code is designed to create a more active web browsing experience.
  - E.g., Java applets, ActiveX controls.
Mobile code (Cont’d)

• Java scripts are distributed in source code form making them easy to analyze.

• ActiveX components are conventional executables that contain native IA-32 machine code.

• Java applets are in bytecode form, which makes them easy to decompile.
Mobile code: Operation

- Web sites quickly download and launch a program on the end user’s system.
- User might see a message that warns about a program that is about to be installed and launched.
  - Most users click OK to allow the program to run.
  - They may not consider the possibility that malicious code is about to be downloaded and executed on their system.
Adware

• Adware is a program that forces unsolicited advertising on end users.
• Adware is a new category of malicious programs that has become very popular.
• Adware is usually bundled with free software that is funded by the advertisements displayed by the Adware program.
Adware: Operation (1)

- The program gathers statistics about the end user’s browsing and shopping habits.
  - The data might be transferred to a remote server.
- Then the Adware uses the information to display targeted advertisements to the end user.
Adware: Operation (2)

• Adware can be buggy and can limit the performance of the infected machine.
  – E.g., MS IE can freeze for a long time because an Adware DLL is poorly implemented and does not use multithreading properly.
• Ironically, buggy Adware defeats the purpose of the Adware itself.
Sticky software

- Sticky software implements methods that prevent or deter users from uninstalling it manually.
- One simple solution is not to offer an uninstall program.
- Another solution in Windows involves:
  - Installing registry keys that instruct Windows to always launch the malware as soon as the system is booted.
  - The malware monitors changes to the registry and replace the keys of they are deleted by the user.
  - The malware uses two mutually monitoring processes to ensure that the user does not terminate the malware before deleting the keys.
Future Malware

• Today’s malware is just the tip of the iceberg.
• The next generation of malware may take control of the low levels of the computer system (e.g., BIOS, Firmware).
  – The antidote software will be in the control of the malware …
• Also the theft of valuable information can result in holding it for ransom.
Information-stealing worms

- Present-day malware does not take advantage of cryptography much.
- Asymmetric encryption creates new possibilities for the creation of information-stealing worms.
- A worm encrypts valuable data on the infected system using an asymmetric cipher and hold the data as ransom.
Information-stealing worms: Operation

1. The Kleptographic worm embeds a public encryption key in its body.
2. It starts encrypting every bit of valuable data on the host using the public key.
3. Decryption of the data is impossible without the private key.
4. Attacker blackmails the victim demanding ransom.
5. Attacker exchanges the private key for the ransom while maintaining anonymity.
   - Theoretically possible using zero-knowledge proofs
   - Attacker proves that he has the private key without exposing it.
BIOS/Firmware Malware

- Antivirus programs assume that there is always some trusted layer of the system.
- Naïve antivirus programs scan the hard drive for infected files using the high-level file-system service.
- A clever virus can intercept file system calls and present to the virus with fake versions (original/uninfected) of the files on disk.
- Sophisticated antivirus programs reside at a low enough level (in OS kernel) so that malware cannot distort their view of the system.
BIOS/Firmware Malware: Operations (1)

• What is the malware altered an extremely low level layer of the system?
• Most CPUs/hardware devices run very low-level code that implements each assembly language instruction using low level instructions (micro-ops).
• The micro-ops code that runs inside the processor is called firmware.
• Firmware can be updated using a firmware-updating program.
BIOS/Firmware Malware: Operations (2)

• Malicious firmware can (in theory) be included in malware that defeats antivirus programs.

• The hardware will be compromised by the malicious firmware.

• Not easy to do in practice because firmware update files are encrypted (private key inside the processor).
Antivirus programs

• Antivirus programs identify malware by looking for unique signatures in the code of each program (i.e., potential virus) on a computer.
  – A signature is a unique sequence of code found in a part of the malicious program.

• The antivirus program maintains a frequently updated database of virus signatures.
  – The goal is for the database to contain a signature for every known malware program.

• Well known antivirus software includes:
  – Symantec (http://www.symantec.com)
  – McAfee (http://www.mcafee.com)
Polymorphic viruses

• Polymorphism is a technique that thwarts signature-based identification programs.
• Polymorphic viruses randomly encode or encrypt the program code in a semantics-preserving way.
• The idea is to encrypt the code with a random key and decrypt it at runtime.
  – Each copy of the code is different because of the use of a random key.
Polymorphic viruses: 
Decryption technique

• A decryption technique that polymorphic viruses employ involves “XORing” each byte with a randomized key that was saved by the parent virus.

• The use of XOR-operations has the additional advantage that the encryption and decryption routine are the same:
  \[
  \begin{align*}
  &a \text{ xor } b = c \\
  &c \text{ xor } b = a
  \end{align*}
  \]
Polymorphic viruses: Weaknesses

- Many antivirus programs scan for virus signatures in memory.
  - I.e., after the polymorphic virus has been decrypted.
- If the virus code that does the decryption is static, then the decryption code can be used as a signature.
- This limitation can be addressed (somewhat) if the decryption code is scrambled (superficially):
  - randomize the use of registers,
  - add no-ops in the code, …
Metamorphic viruses

• Instead of encrypting the program’s body and making slight alterations in the decryption engine, alter the entire program each time it is replicated.

• This makes it extremely difficult for antivirus writers to use signature-matching techniques to identify malware.

• Metamorphosis requires a powerful code analysis engine that needs to be embedded into the malware.
Metamorphic viruses: Operation

- Metamorphic engine scans the code and generates a different version of it every time the program is duplicated.
- The metamorphic engine performs a wide variety of transformations on the malware and on the engine itself.
  - Instruction and register randomization.
  - Instruction ordering
  - Reversing (negating) conditions
  - Insertion of “garbage” instructions
  - Reordering of the storage location of functions
Emulation/Virtualization in Anti-Virus Tools

- Modern viruses use polymorphism/metamorphism to evade detection by signature
- One way to detect this malware is through emulation in an isolated virtual machine
- Specific emulation techniques can be countered by virus authors (due to limitations on previous slide)
- Result: security through obscurity works here, want to use a different anti-virus tool than the masses
Limitations of Using Virtualization for Malware

- Rice's theorem – nontrivial properties of computer programs cannot be determined automatically
  - Semantic gap – more context inside of a virtual machine than outside of it
  - An operating system sees processes and files versus registers, disk blocks and memory pages outside
  - Instrumentation can help (but possibly corruptible)
Bridging the Semantic Gap

- Xuxian Jiang et al ACM CCS 2007
- Recreate internal state externally in the VMM (bridge the semantic gap)
  - Reconstruct files/directories from raw virtual disk
  - Reconstruct process information from virtual memory
- Compare internal view with reconstructed view to find root kits
- View reconstruction good enough to run out-of-the-box anti-virus software (Symantec, Kapersky, McAfee, F-Secure, etc)
Finding Malicious Websites

- web 2.0 means websites can infect machines using browser vulnerabilities
- Compromised hosting facilities mean not just sketchy websites
- Google used VMs to analyze web-based malware
  - Run IE in a virtual machine
  - Record all HTTP fetches, new processes, changes to registry and file systems
Virtualized Malware

- Not all added security, new opportunities for malware
- Virtualized rootkits
- Side Channel attacks
- Bugs in VMMs
Take the “Blue Pill”

- Developed by Joanna Rutkowska
- VMM as root kit
- Instead of hiding your control of a compromised system in the kernel, insert thin hypervisor below it
- Runs at a higher level of privilege, so can still control the machine
- Hard to detect within the OS
Techniques to Detect Virtualized Malware

- Very difficult to hide that virtualization is happening
- Timing issues
- TLB profiling
- Defeating these requires a big code base
- Hard to distinguish “good” VMM from “bad” VMM
Side Channel Attacks

- How isolated are virtual machines?
- Can they spy on each other?
- VMs share the resources of a single host
- If multiple VMs try to use the same resource they may notice
- Mitigated by only allowing each machine a specified share of the resource
- But not completely, and at great efficiency loss
Conclusions

- Virtualization has important security implications
- “Arms race” between hackers and defenders
- Lot of interesting work in this space