Virus Protection and Intrusion Detection

John Mitchell

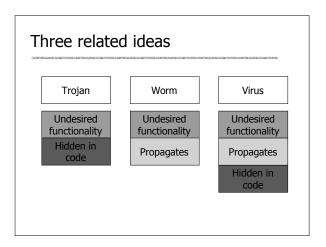
Topics

- ◆ Trojans, worms, and viruses
- ◆ Virus protection
 - Virus scanning methods
- ◆ Detecting system compromise
 - Tripwire
- ◆ Detecting system and network attacks
 - Scanning system call trace
 - Network intrusion detection

What is a Virus?

- ◆ Program embedded in file
- ◆ Spreads and does damage
 - Replicator
 - Portion of virus code that reproduces virus
 - Payload
 - Portion of virus code that does some other function
- ◆ Categories
 - Boot virus (boot sector of disk)
 - Virus in executable file
 - Macro virus (in file executed by application)

Virus scanner is large collection of many techniques



Trojan Horse

... a fake version of PKZIP is being distributed as PKZ300B.ZIP or PKZ300.ZIP. It is not an official version from PKWARE and it will attempt to erase your hard drive if run.

Not a virus since it doesn't replicate

Internet Worm

- ◆ Released November 1988
 - Program spread through Digital, Sun workstations
 - Exploited Unix security vulnerabilities
- **♦** Consequences
 - · No immediate damage from program itself
 - Replication and threat of damage
 - Load on network, systems used in attack
 - Many systems shut down to prevent further attack

Worm description

- ◆ Two parts
 - Program to spread worm
 - look for other machines that might be infected
 - try to find ways of infiltrating these machines
 - Vector program (99 lines of C)
 - compiled and run on the infected machines
 - transferred main program to continue attack
- ◆ Security vulnerabilities
 - fingerd Unix finger daemon
 - sendmail mail distribution program
 - Password cracking
 - Trusted logins

Components of Internet Worm Attack

- ◆ Sendmail
 - Exploit debug option in sendmail to allow shell access
- ◆ Rsh
 - Exploit trusted hosts
- ◆ Fingerd
 - Exploit a buffer overflow in the fgets function

fingerd

- ◆ Written in C and runs continuously
- ◆ Array bounds attack
 - Fingerd expects an input string
 - · Worm uses long string to overwrite memory
- ◆ Attack string
 - Includes machine instructions
 - Overwrites return address
 - Invokes a remote shell
 - Executes privileged commands

sendmail

- ◆ Worm used debug feature
 - Allows a set of commands to be sent to sendmail
 - Send messages to new hosts through the mail system without processing normal mail messages

Password cracking

- ◆ Dictionary attack
 - Read /etc/passwd
 - Used list of ~400 common password strings

Remote shell

- ◆ Unix trust information
 - /etc/host.equiv system wide trusted hosts file
 - /.rhosts and ~/.rhosts users' trusted hosts file
- ◆ Worm exploited trust information
 - Examining files that listed trusted machines
 - · Assume reciprocal trust
 - If X trusts Y, then maybe Y trusts X

Detecting Internet Worm

- ◆ Files
 - Strange files appeared in infected systems
 - Strange log messages for certain programs
- ◆ System load
 - Infection generates a number of processes
 - Systems were reinfected => number of processes grew and systems became overloaded

Thousands of systems were shut down

Stopping the worm

- ◆ System admins busy for several days
 - Devised, distributed, installed modifications
- ◆ Perpetrator
 - Student at Cornell; discovered quickly and charged
 - Sentence: community service, \$10,000 fine
 - Program did not cause deliberate damage
 - Tried (failed) to control # of processes on host machines

♦ Lessons?

- Security vulnerabilities come from system flaws
- Diversity is useful for resisting attack
- "Experiments" can be dangerous

Reference

◆ Eugene H. Spafford; The Internet Worm: Crisis and Aftermath; COMMUNICATIONS OF THE ACM; 32(6), pp. 678-687, Jun 1989

Virus Examples

◆ Jerusalem

- One oldest and most common; many variants
- Will infect both .EXE and .COM files
- Every Friday 13th, deletes programs run that day

◆ Melissa

- Word macro virus spread by email
- Initially distributed in an internet group alt.sex
- Sent in a file called LIST.DOC
- When opened, macro emails to 50 people listed in the address book of the user.

Melissa Email

From: (name of infected user)

Subject: Important Message From (name of infected user) To: (50 names from alias list)

Here is that document you asked for \dots don't show anyone else ;-)

Attachment: LIST.DOC

◆ Recipients likely to open a document from someone they know

Viruses - What's Out There?

- ◆ Wild List http://www.wildlist.org/
 - · Industry standard
 - · Currently 64 participants
 - mostly from security companies
 - keep watch for active viruses
 - About 200 current sightings
 - Viruses not sighted by two independent participants drop off list
- ◆ Virus families
 - Many viruses reuse replicators that have proven effective

Who writes viruses?

- ◆ Limited scientific study
 - Sarah Gordon papers at http://www.research.ibm.com/antivirus/SciPapers.htm
- ◆ Identified four groups by survey
 - Early adolescent, College student, Adult/professiona, Ex-writer of viruses
- ◆ Trends
 - "Those who have continued a normal ethical development have aged out of virus writing"
 - Some are older and more skilled than before
 - Viruses like Zhengxi and Concept point to an advanced knowledge of programming techniques

How hard is it to do?

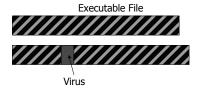
- ◆ Google search: virus construction toolkit
- ◆ First link:
 - Name: OVCT
 - Type: Virus Creation Kit
 - Info:

Overwritting Virus Construction Toolkit is a virus source generator program designed for makeing overwritting virii.

- ◆ Links to ~40 other construction kits at http://www.ebcvg.com/creation_labs.php
 - I do not recommend downloading or running these!!

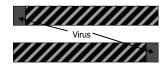
Simple File Infecting Virus

- ◆ Propagate identical copy of itself
- ◆ Identified by "signature"
 - Characteristic bit pattern in virus code
 - Often detects family of viruses with similar replicator



Performance Issues

- ◆ Many files to scan, many signatures
- ◆ Optimizations?



- Many viruses at beginning or end of a file
- Almost all viruses are less than 4KB

More General Limitation

- ◆ Virus must be executed to be effective
 - Most viruses at an entry point or after nonbranching code
- ◆ Antivirus programs check entry points
 - Establish a variable E for the program's entry point
 - Each entry point scans the instruction at that location
 - If it transfers control to another location, set E to that location and go back to step 2
 - Search the bytes at location E for virus signatures

 Reference: Nachenberg article

Virus Encryption

- ◆ Writer may encrypted main portion of virus
 - Decryption code
 - Encrypted Virus code
 - Does not need to be strong encryption
 - Just something to fool fast checker
- ◆ Encrypted code depends on key used
- ◆ Identify virus by decryption routine
 - Decryption routines are often unique
 - Most have at least 10-15 distinct bytes
 - Since small, increase probability of ident error

- Virus Cleaning
- ◆ Virus detection
 - Determine whether there is a virus
- ◆ Virus identification
 - Determine the identity or family of virus
- ◆ Virus cleaning
 - Remove virus from file
 - Requires some knowledge of how virus works
 - How many bytes in replicator,
 - Identify beginning/end of payload,

- .

Identification errors make it harder to clean files

Polymorphic Viruses ◆ Change "shape" as they propagate • Specially designed mutation engines - can generate billions of mutation routines - mutation engine may be more complex than viruses • Combine with encryption - change decryption routine by switching the order of instructions

Polymorphic Virus Detection

- ◆ Sandboxing
 - Run the file on a protected virtual computer
- ◆ Analyze virus body when decrypted
- ◆ Many performance problems
 - How long to run each program?
 - Solve the halting problem

Sophisticated viruses require sophisticated detection

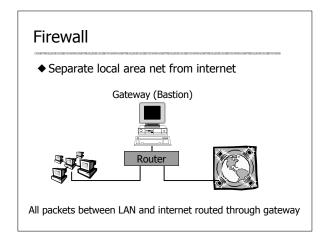
Intrusion detection

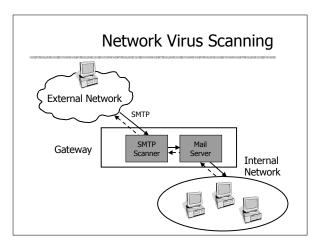
- ◆ An intrusion is
 - "any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource"

[Heady, Luger, et al.]

Outline

- ◆ Intrusion prevention
 - Network firewall
 - restrict flow of packets; cover in another lecture
 - System security
- ◆ Intrusion detection
 - Discover system modifications
 - Tripwire
 - Look for attack in progress
 - Network traffic patterns
 - System calls, other system events





System intrusion detection

- ◆ Many kinds of attacks
 - Network denial of service
 - Attempts to gain user access
 - User attempts to gain root access
- ◆ Premise for intrusion detection
 - "Normal" user behavior is different from behavior of malicious user

General problem

- ◆ How to detect intrusion?
 - Detect damage to system
 - Might be too late...
 - Detect attack as it is happening?
 - How to recognize an attack?

Tripwire

- ◆ Standard attack
 - · Gain user access
 - · Gain root access
 - Replace system binaries to establish backdoor
 - Use backdoor for future activities
- ◆ Tripwire detection point
 - Modification of system binaries

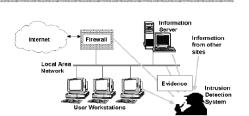
Tripwire System

- ◆ General idea
 - Compute hash of key system binaries
 - Compare current hash to hash stored earlier
 - Report problem if hash is different
 - Store reference hash codes on read-only medium
- ◆ Specialized method
 - Does not catch all intrusions
 - Automates common way of monitoring system

Is Tripwire too late?

- ◆ Typical attack on server
 - Gain access
 - · Install backdoor
 - This can be in memory, not on disk!!
 - Use it
- ◆ Tripwire
 - Is a good idea
 - Wont catch attacks that don't change system files

DARPA intrusion detection program



- Intrusion detection systems monitor network state looking for unauthorized usage, denial of service, and anomalous behavior Such systems have never been formally evaluated... until now

ID Data Sources

- ♦ Host based
 - Audit data from a single host to detect intrusions
- ◆ Multihost based
 - Audit data from multiple hosts to detect intrusions
- ◆ Network based
 - Network traffic data, along with audit data from one or more hosts, is used to detect intrusions

see http://www.cs.purdue.edu/coast/intrusion-detection/

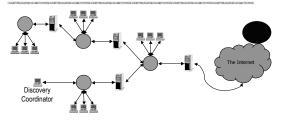
ID Intrusion Models

- ◆ Anomaly detection model
 - Try to figure out what is "normal"
 - Report anomalous behavior
- ◆ Misuse detection model
 - · Maintain data on known attacks
 - Look for activity corresponding to known intrusion techniques (signatures) or system vulnerabilities

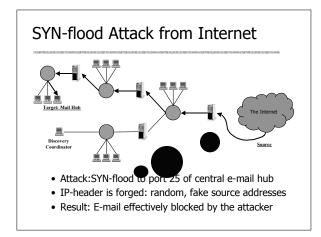
Port sweep attack and detection

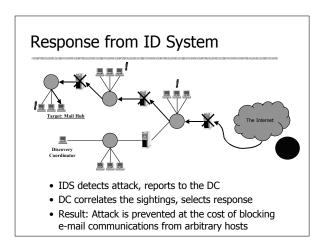
- ◆ Many computers have known vulnerabilities
 - Standard software configuration can be attacked
 - Attack specific to certain OS and version
- ◆ Attackers sweep net to find certain machines
 - Port sweep will try to connect to many ports on many IP addresses
- ◆ This behavior is not "normal"

Example (UCD Computer Security Lab)



- Hosts grouped into neighborhoods
- Nbhd boundary protected by filtering router/firewall
- Centralized Discovery Coordinator directs global activity



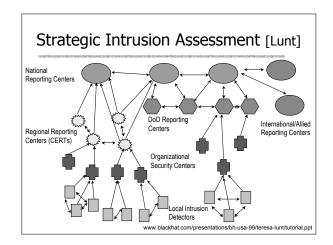


Statistical methods

- ◆ Monitor network traffic, system calls
- ◆ Compute statistical properties
- ◆ Report errors if statistics are outside established range

Difficulties in intrusion detection

- ◆ Lack of training data
 - Lots of "normal" network, system call data
 - Little data containing realistic attacks, anomalies
- ◆ Data drift
 - Statistical methods detect changes in behavior
 - Attacker can attack gradually and incrementally
- ◆ Main characteristics not well understood
 - By many measures, attack may be within bounds of "normal" range of activities
- ◆ False identifications are very costly
 - Sys Admin spend many hours examining evidence



Strategic Intrusion Assessment [Lunt]

- ◆ Test over two-week period
 - AFIWC's intrusion detectors at 100 AFBs alarmed on 2 million sessions
 - After manual review, reduced to 12,000 suspicious events
 - After further manual review, these were reduced to four actual incidents
- ◆ Conclusion
 - · Most alarms are false positives
 - Most true positives are trivial incidents
 - Of the significant incidents, most are isolated attacks to be dealt with locally

Interesting Approach [Wagner, Dean]

- ◆ Problem
 - ID Systems may produce too many false alarms
- ◆ Approach
 - Deduce a model from program source code - Not just by "learning" from observable behavior
 - Use this model to detect unexpected behavior
- ◆ Similar to Ashcraft/Engler work in reverse
 - Examine code to produce FSM model of patterns of system calls

Results so far: lots better than not using code!