CS 155 Spring 2006

Network Worms: Attacks and Defenses

John Mitchell

with slides borrowed from various (noted) sources

Outline

- Worm propagation
  - Worm examples
  - Propagation models
- Detection methods
  - Traffic patterns: EarlyBird
  - Watch attack: TaintCheck and Sting
  - Look at vulnerabilities: Generic Exploit Blocking
- Disable
  - Generate worm signatures and use in network or host-based filters

2

Worm

- A worm is self-replicating software designed to spread through the network
  - Typically exploit security flaws in widely used services
  - Can cause enormous damage
    - Launch DDOS attacks, install bot networks
    - Access sensitive information
    - Cause confusion by corrupting the sensitive information
- Worm vs Virus vs Trojan horse
  - A virus is code embedded in a file or program
  - Viruses and Trojan horses rely on human intervention
  - Worms are self-contained and may spread autonomously

3

Cost of worm attacks

- Morris worm, 1988
  - Infected approximately 6,000 machines
    - 10% of computers connected to the Internet
  - cost ~ $10 million in downtime and cleanup
- Code Red worm, July 16 2001
  - Direct descendant of Morris' worm
  - Infected more than 500,000 servers
    - Programmed to go into infinite sleep mode July 28
  - Caused ~ $2.6 Billion in damages,
- Love Bug worm: $8.75 billion

4 Statistics: Computer Economics Inc., Carlsbad, California

Aggregate statistics

Financial Impact of Virus Attacks 1995—2005

Worldwide Impact (US $)	
2005	$14.2 Billion
2004	17.8 Billion
2003	13.0 Billion
2002	11.1 Billion
2001	13.2 Billion
2000	17.1 Billion
1999	12.0 Billion
1998	6.1 Billion
1997	3.3 Billion
1996	1.8 Billion
1995	500 Million

Source: Computer Economics, 2006 Figure 1

5

Internet Worm (First major attack)

- Released November 1988
  - Program spread through Digital, Sun workstations
  - Exploited Unix security vulnerabilities
    - VAX computers and SUN-3 workstations running versions 4.2 and 4.3 Berkeley UNIX code
- 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

6

Internet Worm Description

- Two parts
  - Program to spread worm
    - look for other machines that could 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
  - Trusted logins (.rhosts)
  - Weak passwords

7

Three ways the worm spread

- Sendmail
  - Exploit debug option in sendmail to allow shell access
- Fingerd
  - Exploit a buffer overflow in the fgets function
  - Apparently, this was the most successful attack
- Rsh
  - Exploit trusted hosts
  - Password cracking

8

sendmail

- Worm used debug feature
  - Opens TCP connection to machine's SMTP port
  - Invokes debug mode
  - Sends a RCPT TO that pipes data through shell
  - Shell script retrieves worm main program
    - places 40-line C program in temporary file called x$$,l1.c where $$ is current process ID
    - Compiles and executes this program
    - Opens socket to machine that sent script
    - Retrieves worm main program, compiles it and runs

9

fingerd

- Written in C and runs continuously
- Array bounds attack
  - Fingerd expects an input string
  - Worm writes long string to internal 512-byte buffer
- Attack string
  - Includes machine instructions
  - Overwrites return address
  - Invokes a remote shell
  - Executes privileged commands

10

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
- Password cracking
  - Worm was running as daemon (not root) so needed to break into accounts to use .rhosts feature
  - Dictionary attack
  - Read /etc/passwd, used ~400 common password strings

11

The worm itself

- Program is called 'sh'
  - Clobbers argv array so a 'ps' will not show its name
  - Opens its files, then unlinks (deletes) them so can't be found
    - Since files are open, worm can still access their contents
- Tries to infect as many other hosts as possible
  - When worm successfully connects, forks a child to continue the infection while the parent keeps trying new hosts
- Worm did not:
  - Delete system's files, modify existing files, install trojan horses, record or transmit decrypted passwords, capture superuser privileges, propagate over UUCP, X.25, DECNET, or BITNET

12

Detecting Morris 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
    - Apparently not intended by worm's creator

Thousands of systems were shut down

13

Stopping the worm

- System admins busy for several days
  - Devised, distributed, installed modifications
- Perpetrator
  - Student at Cornell; discovered quickly and charged
  - Sentence: community service and $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

14

Sources for more information

- Eugene H. Spafford, The Internet Worm: Crisis and Aftermath, CACM 32(6) 678-687, June 1989
- Page, Bob, "A Report on the Internet Worm", http://www.ee.ryerson.ca:8080/~elf/hack/iworm.html

15

Some historical worms of note

Worm	Date	Distinction
Morris	11/88	Used multiple vulnerabilities, propagate to "nearby" sys
ADM	5/98	Random scanning of IP address space
Ramen	1/01	Exploited three vulnerabilities
Lion	3/01	Stealthy, rootkit worm
Cheese	6/01	Vigilante worm that secured vulnerable systems
Code Red	7/01	First sig Windows worm; Completely memory resident
Walk	8/01	Recompiled source code locally
Nimda	9/01	Windows worm: client-to-server, c-to-c, s-to-s, ...
Scalper	6/02	11 days after announcement of vulnerability; peer-to-peer network of compromised systems
Slammer	1/03	Used a single UDP packet for explosive growth

Kienzle and Elder

16

Increasing propagation speed

- Code Red, July 2001
  - Affects Microsoft Index Server 2.0,
    - Windows 2000 Indexing service on Windows NT 4.0,
    - Windows 2000 that run IIS 4.0 and 5.0 Web servers
  - Exploits known buffer overflow in idq.dll
  - Vulnerable population (360,000 servers) infected in 14 hours
- SQL Slammer, January 2003
  - Affects in Microsoft SQL 2000
  - Exploits known buffer overflow vulnerability
    - Server Resolution service vulnerability reported June 2002
    - Patched released in July 2002 Bulletin MS02-39
  - Vulnerable population infected in less than 10 minutes

17

Code Red

- Initial version released July 13, 2001
  - Sends its code as an HTTP request
  - HTTP request exploits buffer overflow
  - Malicious code is not stored in a file
    - Placed in memory and then run
- When executed,
  - Worm checks for the file C:\Notworm
    - If file exists, the worm thread goes into infinite sleep state
  - Creates new threads
    - If the date is before the 20th of the month, the next 99 threads attempt to exploit more computers by targeting random IP addresses

18