Worms & Botnets: Attacks & Defenses

Dawn Song

dawnsong@cs.berkeley.edu

Some slides from John Mitchell [1]

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

Outline

- **Worms**
  - Worm examples and propagation methods
  - Defenses
- **Bots**
  - Structure and use of bots
  - Recognizing bot propagation
  - Recognizing bot operation

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

Statistics: Computer Economics Inc., Carlsbad, California

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

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

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

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

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
- **More Info**
  - 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

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

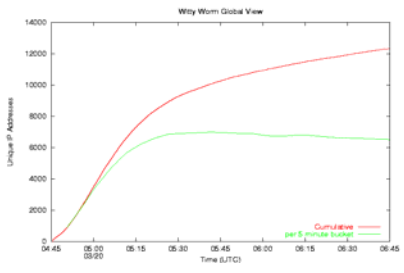
Code Red of July 13 and July 19

- **Initial release of July 13**
  - 1st through 20th month: Spread
    - via random scan of 32-bit IP addr space
  - 20th through end of each month: attack.
    - Flooding attack against 198.137.240.91 (*www.whitehouse.gov)*
  - Failure to seed random number generator ⇒ *linear growth*
- **Revision released July 19, 2001.**
  - White House responds to threat of flooding attack by changing the address of *www.whitehouse.gov*
  - Causes Code Red to die for date ≥ 20th of the month.
  - But: this time random number generator correctly seeded

Slides: Vern Paxson

Witty Worm (I)

- **March 19, 2004, exploiting buffer overflow in firewall (ISS) products**
- **Infected 12,000 machines in 45 mins**



Witty Worm (II)

- **First widely propagated worm w. destructive payload**
  - **Corrupted hard disk**
- **Seeded with more ground-zero hosts**
  - **110 infected machines in first 10 seconds**
- **Shortest interval btw vulnerability disclosure & worm release**
  - **1 day**
- **Demonstrate worms effective for niche too**
- **Security devices can open doors to attacks**
  - **Other examples: Anti-virus software, IDS**

How do worms propagate?

- **Scanning worms**
  - **Worm chooses "random" address**
- **Coordinated scanning**
  - **Different worm instances scan different addresses**
- **Flash worms**
  - **Assemble tree of vulnerable hosts in advance, propagate along tree**
- **Meta-server worm**
  - **Ask server for hosts to infect (e.g., Google for "powered by phpbb")**
- **Topological worm:**
  - **Use information from infected hosts (web server logs, email address books, config files, SSH "known hosts")**
- **Contagion worm**
  - **Propagate parasitically along with normally initiated communication**

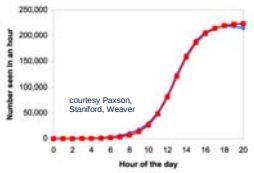
How fast are scanning worms?

- **Model propagation as infectious epidemic**
  - **Simplest version: Homogeneous random contacts**

N: population size
S(t): susceptible hosts at time t
I(t): infected hosts at time t
ß: contact rate
i(t): I(t)/N, s(t): S(t)/N

$$\frac{dI}{dt} = \beta \frac{IS}{N} \qquad \frac{dS}{dt} = -\beta \frac{IS}{N}$$

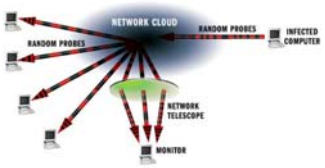
$$\frac{di}{dt} = \beta i(1-i)$$



$$i(t) = \frac{e^{\beta(t-T)}}{1+e^{\beta(t-T)}}$$

How to Measure Worm Scale?

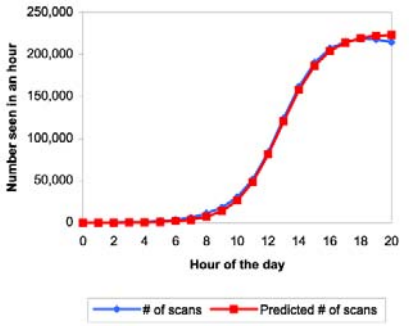
Measuring activity: network telescope



- **Monitor cross-section of Internet address space, measure traffic**
  - **"Backscatter" from DOS floods**
  - **Attackers probing blindly**
  - **Random scanning from worms**
- **LBNL's cross-section: 1/32,768 of Internet**
- **UCSD, UWisc's cross-section: 1/256.**

Code Red I Propagation: Theory Meets Practice

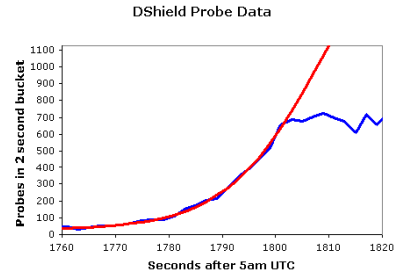
- **Hard to count number of infected hosts**
  - **Count scans by them instead**
- **Theory matches observed**



How to Own the Internet in Your Spare Time in Proceedings of the 11th USENIX Security Symposium (Security '02)

Slammer: The Story Is More Complicated

- **Observed Slammer worm behavior doesn't match theory**
  - **Fast propagating worms encounter links' BW and latency constraints**
  - **Non-universal connectivity**



The Spread of the Sapphire/Slammer Worm, http://www.caida.org/publications/papers/2003/sapphire/sapphire.html

Worm Detection and Defense by Traffic Monitoring

- **Detection via *honeyfarms*: collections of "honeypots" fed by a network telescope.**
  - **Any outbound connection from honeyfarm = worm.**
    (at least, that's the theory)
  - **If telescope covers N addresses, expect detection when worm has infected 1/N of population**
- **Detecting superspreaders**
  - **Hosts that make failed connection attempts to too many other hosts**
  - **Defense: throttling/rate limiting**
    - » **Limiting the number of failed connections by a host**

Challenges for Worm Defense

- **Short interval btw vulnerability disclosure & worm release**
  - **Witty worm: 1 day**
  - **Zero-day exploits**
- **Fast**
  - **Slammer: 10 mins infected 90% vulnerable hosts**
  - **How fast can it be?**
    - » **Flashworm: seconds [Staniford et. al., WORM04]**
- **Large scale**
  - **Slammer: 75,000 machines**
  - **CodeRed: 500,000 machines**

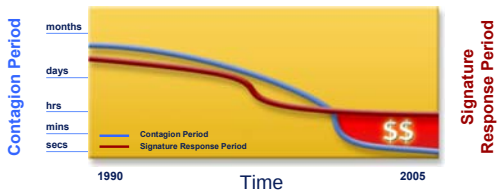
Generic Exploit Blocking

- **Idea**
  - **Write a network IPS signature to generically detect and block all future attacks on a vulnerability**
  - **Different from writing a signature for a specific exploit!**
- **Step #1: Characterize the vulnerability "shape"**
  - **Identify fields, services or protocol states that must be present in attack traffic to exploit the vulnerability**
  - **Identify data footprint size required to exploit the vulnerability**
  - **Identify locality of data footprint; will it be localized or spread across the flow?**
- **Step #2: Write a generic signature that can detect data that "mates" with the vulnerability shape**
- **Similar to Shield research from Microsoft**

Slide: Carey Nachenberg, Symantec

Need for automation

- **Current threats can spread faster than defenses can reaction**
- **Manual capture/analyze/signature/rollout model too slow**



Slide: Carey Nachenberg, Symantec

Generic Exploit Blocking Example #1

Consider MS02-039 Vulnerability (SQL Buffer Overflow):

Field/service/protocol
UDP port 1434
Packet type: 4

Minimum data footprint
Packet size > 60 bytes

Data Localization
Limited to a single packet

```
BEGIN
 DESCRIPTION: MS02-039
 NAME: MS SQL Vuln
 TRANSIT-TYPE: UDP
 TRIGGER: ANY:ANY->ANY:1434
 OFFSET: 0, PACKET
 SIG-BEGIN
   "\x04<getpacketsize(r0)>
   <inrange(r0,61,1000000)>
   <reportid()>"
 SIG-END
END
```

Slide: Carey Nachenberg, Symantec

Generic Exploit Blocking Example #2

Consider MS03-026 Vulnerability (RPC Buffer Overflow):

Field/service/protocol
RPC request on TCP/UDP 135
szName field in CoGetInstanceFromFile func.

Minimum data footprint
Arguments > 62 bytes

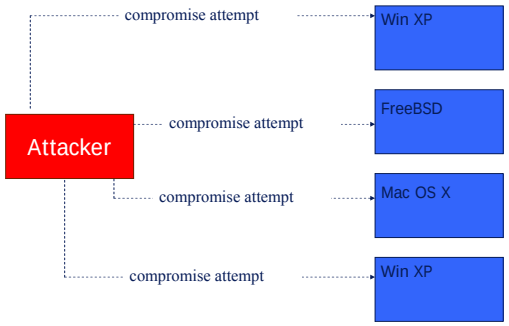
Data Localization
Limited to 256 bytes from start of RPC bind command

```
BEGIN
 DESCRIPTION: MS03-026
 NAME: RPC Vulnerability
 TRANSIT-TYPE: TCP, UDP
 TRIGGER: ANY:ANY->ANY:135
 SIG-BEGIN
   "\x05\x00\x0B\x03\x10\x00\x00
    (about 50 more bytes...)
    \x00\x00.*\x05\x00
    <forward(5)><getbeword(r0)>
    <inrange(r0,63,20000)>
    <reportid()>"
 SIG-END
END
```

Slide: Carey Nachenberg, Symantec

25

Building a Bot Network



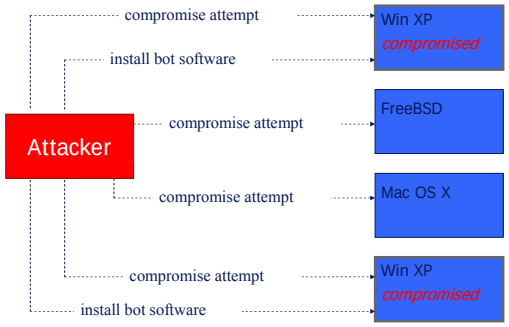
28

Botnet

- **Collection of compromised hosts**
  - **Spread like worms and viruses**
  - **Once installed, respond to remote commands**
- **Platform for many attacks**
  - **Spam forwarding (70% of all spam?)**
  - **Click fraud**
  - **Keystroke logging**
  - **Distributed denial of service attacks**
- **Serious problem**
  - **Top concern of banks, online merchants**
  - **Vint Cerf: ¼ of hosts connected to Internet**

26

Building a Bot Network



29

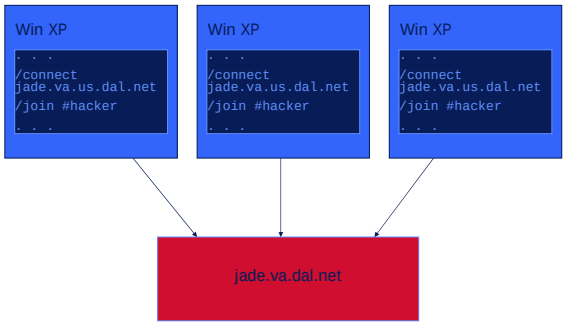
What are botnets used for?

capability	ago	DSNX	evil	G-SyS	sd	Spy
create port redirect	√	√		√	√	√
other proxy	√					
download file from web	√	√		√	√	√
DNS resolution	√			√	√	
UDP/ping floods	√		√	√	√	
other DDoS floods	√			√		√
scan/spread	√	√		√	√	√
spam	√					
visit URL	√			√	√	

Capabilities are exercised via remote *commands.*

27

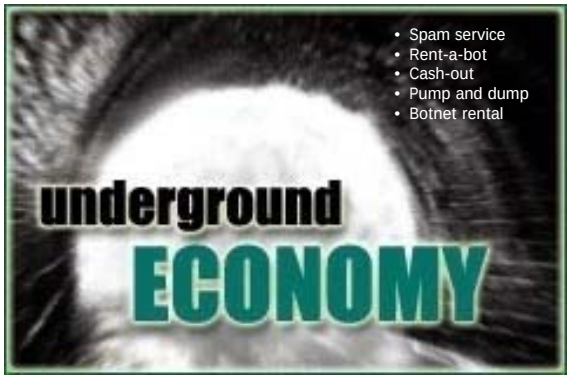
Step 2



30

Step 3

```
(12:59:27pm) -- A9-pcgbdv (A9-pcgbdv@140.134.36.124)
has joined (#owned) Users : 1646

(12:59:27pm) (@PhaTTy) .ddos.synflood 216.209.82.62

(12:59:27pm) -- A6-bpxufrd (A6-bpxufrd@wp95-
81.introweb.nl) has joined (#owned) Users : 1647

(12:59:27pm) -- A9-nzmpah (A9-nzmpah@140.122.200.221)
has left IRC (Connection reset by peer)

(12:59:28pm) (@PhaTTy) .scan.enable DCOM

(12:59:28pm) -- A9-tzrkeasv (A9-tzrkeas@220.89.66.93)
has joined (#owned) Users : 1650
```

Underground commerce

- **Market in access to bots**
  - Botherd: Collects and manages bots
  - Access to proxies ("peas") sold to spammers, often with commercial-looking web interface
- **Sample rates**
  - Non-exclusive access to botnet: 10¢ per machine
  - Exclusive access: 25¢.
  - Payment via compromised account (eg PayPal) or cash to dropbox
- **Identity Theft**
  - Keystroke logging
  - Complete identities available for $25 - $200+
    - » Rates depend on financial situation of compromised person
    - » Include all info from PC files, plus all websites of interest with passwords/account info used by PC owner
    - » At $200+, usually includes full credit report

[Lloyd Taylor, Keynote Systems, *SFBay InfraGard Board*]

Storm Email Worm Case Study

- **Clicking on email attachment/links causes malicious code installed**
  - Fake news story on deadly storm
  - E-cards from family & friends
  - Links to malicious website for drive-by downloads
  - Quick change to stay ahead of AV blocking
    - » Malicious code is modified every 30 minutes, undermining standard signature based AV's ability to block this threat
- **Infected machines form botnet**
  - Largest botnet: 1.7 million bots by end of July
  - P2P architecture instead of centralized
- **Stealth: install rootkits, etc.**
- **Anti-VM: detects VM and won't infect them**
- **For profit:**
  - Botnet sent stock-picking spam, ripping profits for risen stock price

Build Your Own Botnet

- **Pick a vector mechanism**
  - IRC Channels: DCC Filesends, Website Adverts to Exploit Sites
  - Scan & Sploit: MSBlast
  - Trojan: SoBig/BugBear/ActiveX Exploits
- **Choose a Payload**
  - Backdoors
    - » Agobot, SubSeven, DeepThroat
    - » Most include mechanisms for DDoS, Self-spreading, download/exec arbitrary code, password stealers.
- **Do it**
  - Compromise an IRC server, or use your own zombied machines
  - Configure Payload to connect to selected server
  - Load encryption keys and codes
  - Release through appropriate compromised systems
  - Sit back and wait, or start on your next Botnet

[Lloyd Taylor, Keynote Systems, *SFBay InfraGard Board*]

Bot detection methods

- **Signature-based (most AV products)**
- **Rule-based**
  - Monitor outbound network connections (e.g. ZoneAlarm, BINDER)
  - Block certain ports (25, 6667, ...)
- **Hybrid: content-based filtering**
  - Match network packet contents to known command strings (keywords)
  - E.g. Gaobot ddos cmds: .ddos.httpflood
- **Network traffic monitoring**
  - Bot Hunter
    - » Correlate various NIDS alarms to identify "bot infection sequence"
  - Recognize traffic patterns associated with dynamic dns based rallying