Malware

Overview

- Defining malware (*malicious logic*)
- Types
  - Trojan horses
  - Computer viruses and worms
- Theory: arbitrary program being a virus undecidable?
- Defenses

Definitions

Malware

- Set of instructions that cause site security policy to be violated

Example

- Shell script on a UNIX system:

```
cp /bin/sh /tmp/.xyzzy

chmod u+s,o+x /tmp/.xyzzy

ls $*

rm ./ls
```

- Place in program called “ls” and trick someone into executing it
- You now have a setuid-to-*them* shell!

Trojan Horse

- Program with an *overt* purpose (known to user) and a *covert* purpose (unknown to user)
  - Often called a Trojan
  - Named by Dan Edwards in Anderson Report
- Example: previous script is Trojan horse
  - Overt purpose: list files in directory
  - Covert purpose: create setuid shell

Example: Gemini

- Designed for Android cell phones
- Placed in several Android apps on Android markets, forums
- When app was run:
  - Gemini installed itself, using several techniques to make it hard to find
  - Then it connected to a remote command and control server, waited for commands
  - Commands it could execute included delete SMS messages; send SMS messages to remote server; dump contact list; dump list of apps

Rootkits

- Trojan horse corrupting system to carry out covert action without detection
- Earliest ones installed back doors so attackers could enter systems, then corrupted system programs to hide entry and actions
  - Program to list directory contents altered to not include certain files
  - Network status program altered to hide connections from specific hosts

Example: Linux Rootkit IV

- Replaced system programs that might reveal its presence
  - *ls*, *find*, *du* for file system; *ps*, *top*, *lsof*, *killall* for processes; *crontab* to hide rootkit jobs
  - *login* and others to allow attacker to log in, acquire superuser privileges (and it suppressed any logging)
  - *netstat*, *ifconfig* to hide presence of attacker
  - *tcpd*, *syslogd* to inhibit logging
- Added back doors so attackers could log in unnoticed
- Also added network sniffers to gather user names, passwords
- Similar rootkits existed for other systems

Defenses

- Use non-standard programs to obtain the same information that standard ones should; then compare
  - *ls* lists contents of directory
  - A locally written program to read directory entries, was non-standard
- Look for specific strings in executables
  - Programs to do this analysis usually not rigged, but easy enough to write your own
- Look for changes using cryptographically strong checksums

Oops …

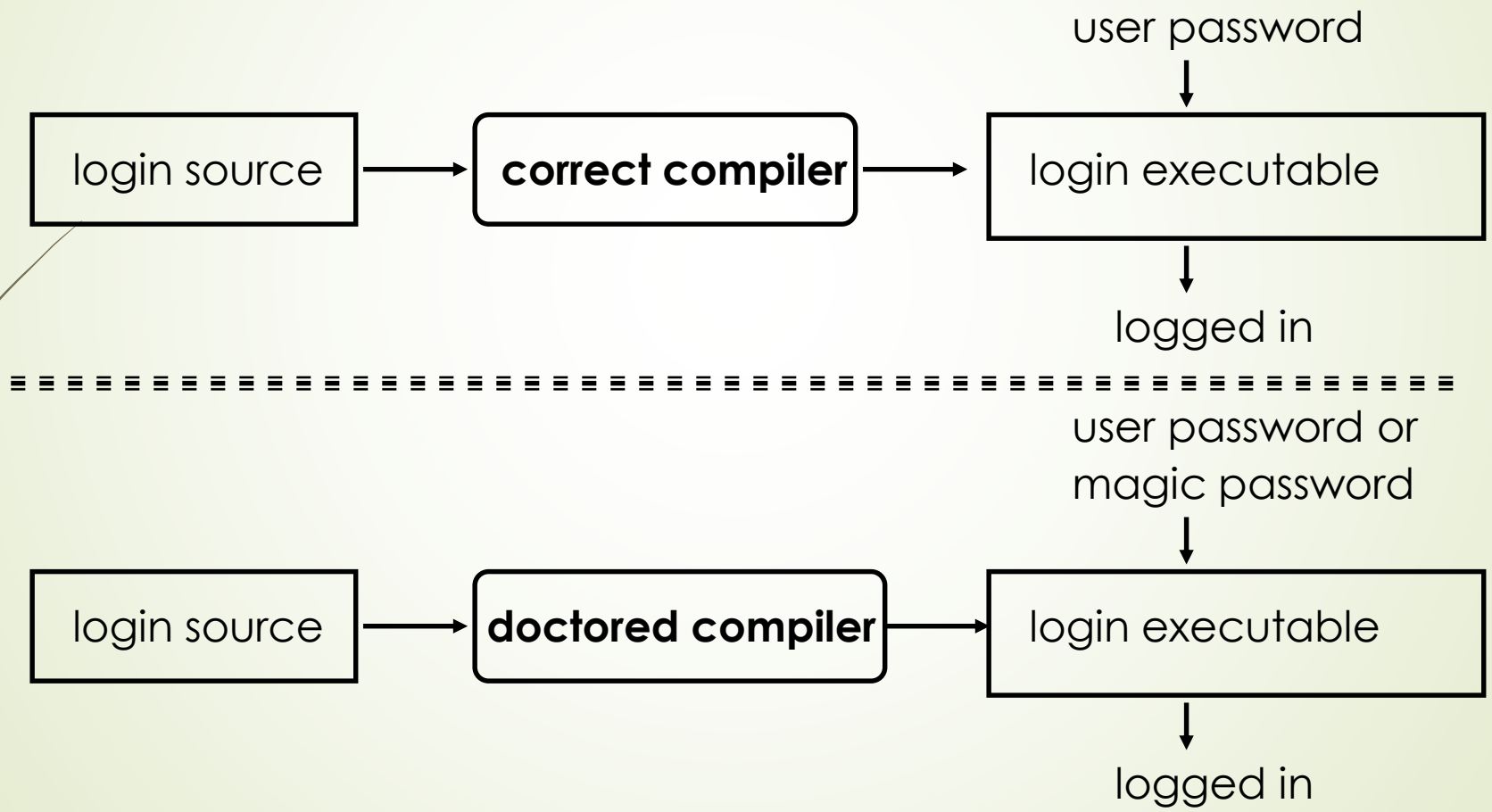
- Sony BMG developed rootkit to implement DRM on a music CDsss
  - Only worked on Windows systems; users had to install a proprietary program to play the music
  - Also installed software that altered functions in Windows OS to prevent playing music using other programs
  - This software concealed itself by altering kernel not to list any files or folders beginning with "sys" and storing its software in such a folder
  - On boot, software contacted Sony to get advertisements to display when music was played
  - Once made public, attackers created Trojan horses with names beginning with "sys (like "sysdrv.exe")
- Result: lawsuits, flood of bad publicity, and recall of all such CDs

Thompson's Compiler

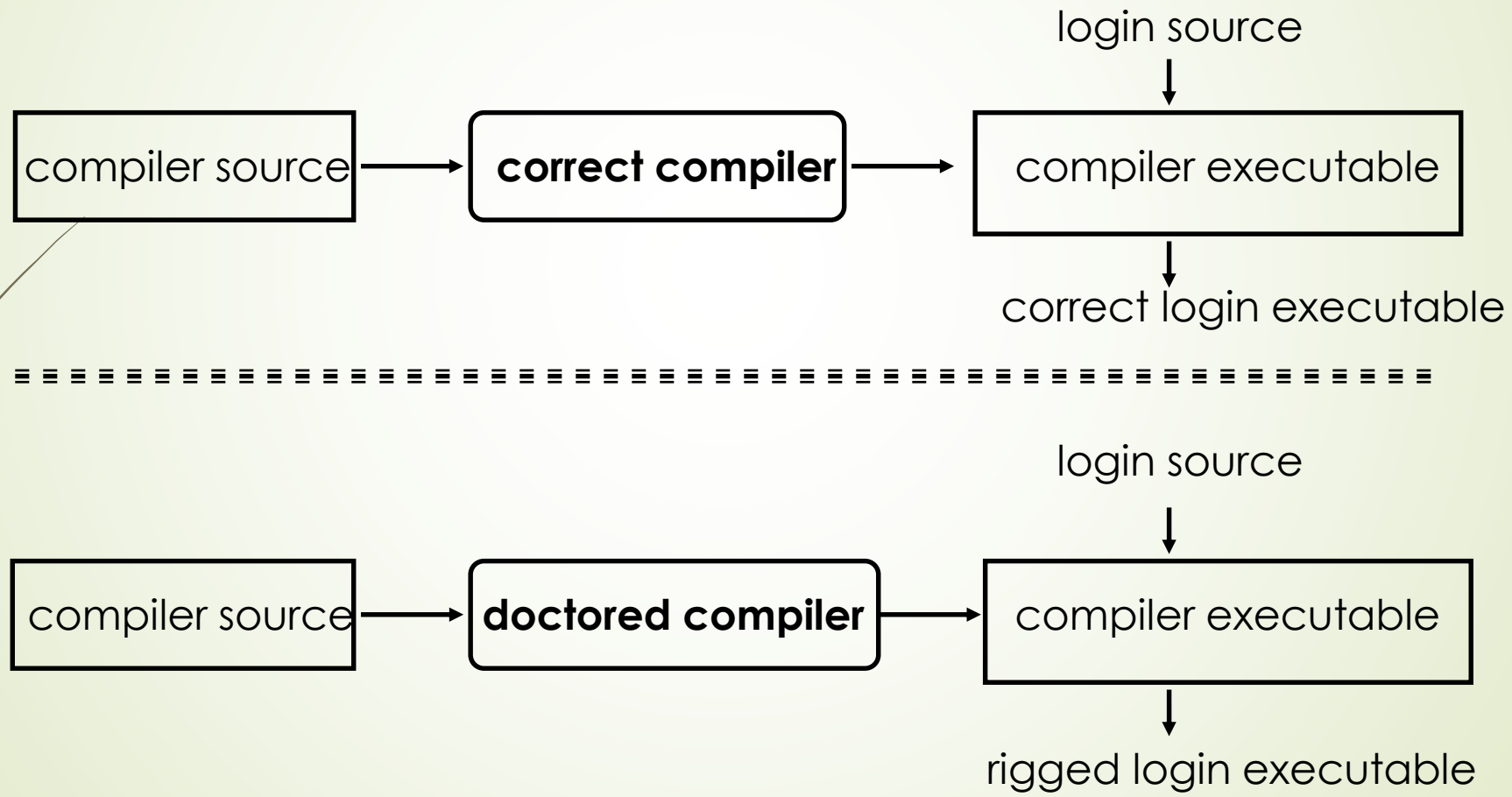
Thompson's Compiler

- Modify the compiler so that when it compiles *login*, *login* accepts the user's correct password or a fixed password (the same one for all users)
- Then modify the compiler again, so when it compiles a new version of the compiler, the extra code to do the first step is automatically inserted
- Recompile the compiler
- Delete the source containing the modification and put the undoctored source back

The *login* Program



The Compiler



Comments

- Great pains taken to ensure second version of compiler never released
  - Finally deleted when a new compiler executable from a different system overwrote the doctored compiler
- The point: *no amount of source-level verification or scrutiny will protect you from using untrusted code*
  - Also: having source code helps, but does not ensure you're safe

Computer Virus

Computer Virus

- Program that inserts itself into one or more files and performs some action
  - *Insertion phase* is inserting itself into file
  - *Execution phase* is performing some (possibly null) action
- Insertion phase *must* be present
  - Need not always be executed

Pseudocode

```
beginvirus:
   if spread-condition then begin
      for some set of target files do begin
         if target is not infected then begin
            determine where to place virus instructions
            copy instructions from beginvirus to endvirus
               into target
            alter target to execute added instructions
         end;
      end;
   end;
   perform some action(s)
   goto beginning of infected program
endvirus:
```

Trojan Horse Or Not?

- Yes
  - Overt action = infected program's actions
  - Covert action = virus' actions (infect, execute)
- No
  - Overt purpose = virus' actions (infect, execute)
  - Covert purpose = none

History

- Programmers for Apple II wrote some
  - Not called viruses; very experimental; 1980
- Fred Cohen
  - 1983, Graduate student who described them
  - Adleman, of RSA fame, named it “computer virus”
  - Tested idea on UNIX systems and UNIVAC 1108 system

First Reports of Viruses in the Wild

- Brain (Pakistani) virus (1986)
  - Written for IBM PCs
  - Alters boot sectors of floppies, spreads to other floppies
- MacMag Peace virus (1987)
  - Written for Macintosh
  - Prints "universal message of peace" on March 2, 1988 and deletes itself
- Duff's experiments (1987)
  - Small virus placed on UNIX system, spread to 46 systems in 8 days
  - Wrote a Bourne shell script virus
- Highland's Lotus 1-2-3 virus (1989)
  - Stored as a set of commands in a spreadsheet
  - Changed a value in a specific row, column and spread to other files

Infection Vectors

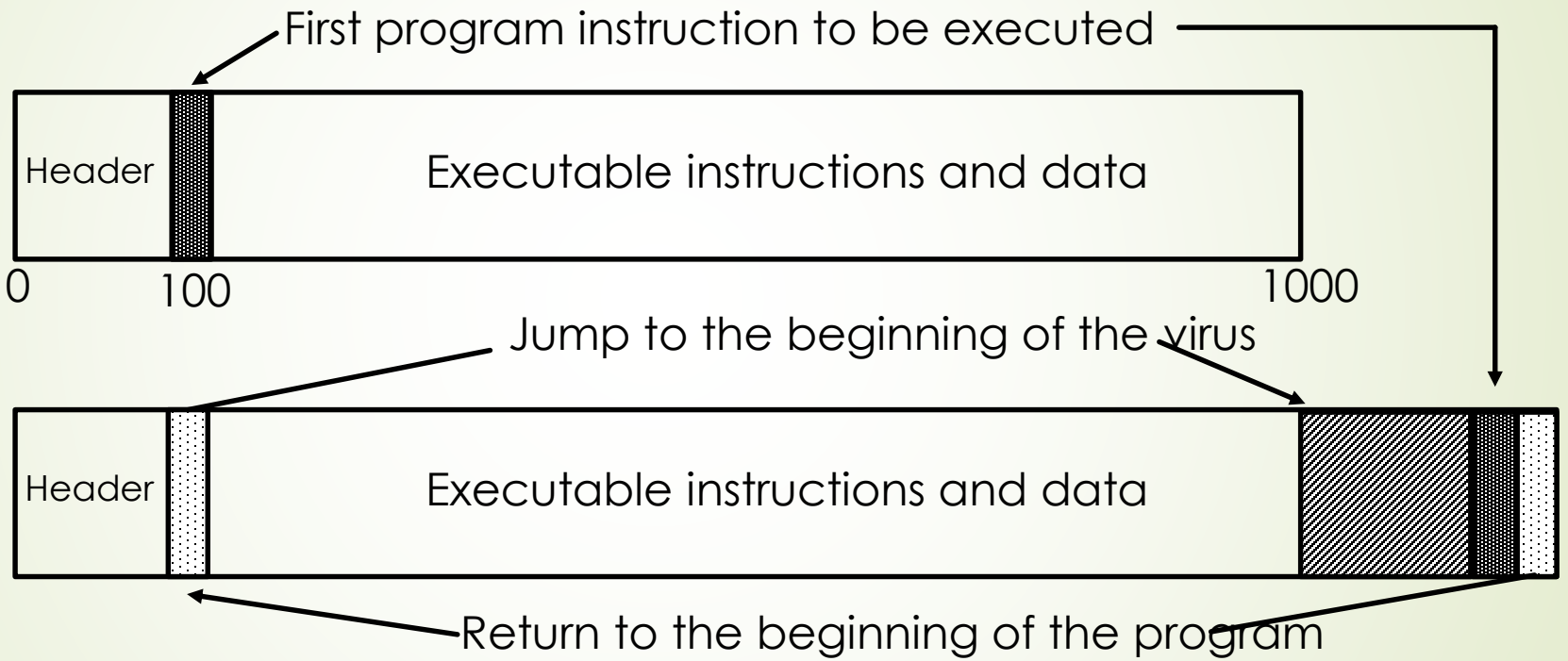
Infection Vectors

- Boot sector infectors
- Executable infectors
- Data infectors
- Some viruses do two or three of these

Boot Sector Infectors

- A virus that inserts itself into the boot sector of a disk
  - Section of disk containing code
  - Executed when system first "sees" the disk
    - Including at boot time …
- Example: Brain virus
  - Moves disk interrupt vector from 13H to 6DH
  - Sets new interrupt vector to invoke Brain virus
  - When new floppy seen, check for 1234H at location 4
    - If not there, copies itself onto disk after saving original boot block
    - if no free space, doesn't infect but if any free space, it infects, possibly overwriting used disk space
    - If there, jumps to vector at 6DH

Executable Infectors



- A virus that infects executable programs
  - Can infect either .EXE or .COM on PCs
  - May append itself (as shown) or put itself anywhere, fixing up binary so it is executed at some point

Executable Infectors (*con't*)

- Jerusalem (Israeli) virus
  - Checks if system infected
    - If not, set up to respond to requests to execute files
  - Checks date
    - If not 1987 or Friday 13th, set up to respond to clock interrupts and then run program
    - Otherwise, set destructive flag; will delete, not infect, files
  - Then: check all calls asking files to be executed
    - Do nothing for COMMAND.COM
    - Otherwise, infect or delete
  - Error: doesn't set signature when .EXE executes
    - So .EXE files continually reinfected

Macro Viruses

- A virus composed of a sequence of instructions that are interpreted rather than executed directly
- Can infect either executables (Duff's shell virus) or data files (Highland's Lotus 1-2-3 spreadsheet virus)
- Independent of machine architecture
  - But their effects may be machine dependent

Example

- Melissa
  - Infected Microsoft Word 97 and Word 98 documents
    - Windows and Macintosh systems
  - Invoked when program opens infected file
  - Installs itself as “open” macro and copies itself into Normal template
  - Invokes mail program, sends itself to everyone in user’s address book
    - Used a mail program that most Macintosh users didn’t use, so this was rare for Macintosh users

Multipartite Viruses

- A virus that can infect either boot sectors or executables
- Typically, two parts
  - One part boot sector infector
  - Other part executable infector

Concealment

Concealment

- Terminate and stay resident (TSR)
- Stealth
- Encryption
- Polymorphism
- Metamorphism

TSR Viruses

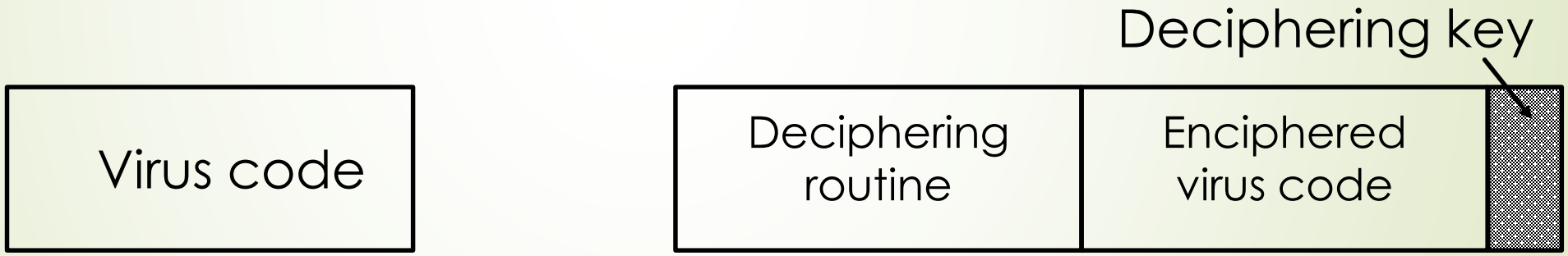
- A virus that stays active in memory after the application (or bootstrapping, or disk mounting) is completed
  - Non-TSR viruses only execute when host application executes
- Examples: Brain, Jerusalem viruses
  - Stay in memory after program or disk mount is completed

Stealth Viruses

- A virus that conceals infection of files
- Example: IDF (also called Stealth or 4096) virus modifies DOS service interrupt handler as follows:
  - Request for file length: return length of *uninfected* file
  - Request to open file: temporarily disinfect file, and reinfect on closing
  - Request to load file for execution: load infected file

Encrypted Viruses

- A virus that is enciphered except for a small deciphering routine
  - Detecting virus by signature now much harder as most of virus is enciphered



Example

```
(* Decryption code of the 1260 virus *)
(* initialize the registers with the keys *)
rA = k1;
rB = k2;
(* initialize rC with the virus; starts at sov, ends at eov *)
rC = sov;
(* the encipherment loop *)
while (rC != eov) do begin
        (* encipher the byte of the message *)
        (*rC) = (*rC) xor rA xor rB;
        (* advance all the counters *)
        rC = rC + 1;
        rA = rA + 1;
end
```

Polymorphic Viruses

- A virus that changes its form each time it inserts itself into another program
- Idea is to prevent signature detection by changing the "signature" or instructions used for deciphering routine
  - At instruction level: substitute instructions
  - At algorithm level: different algorithms to achieve the same purpose
- Toolkits to make these exist (Mutation Engine, Trident Polymorphic Engine)

Example

- These are different instructions (with different bit patterns) but have the same effect:
  - add 0 to register
  - subtract 0 from register
  - xor 0 with register
  - no-op
- Polymorphic virus would pick randomly from among these instructions

Metamorphic

- Like polymorphic, but virus itself is also obscured
  - So two instances of virus would look different when loaded into memory
- When decrypted, virus may have:
  - Two completely different implementations
  - Two completely different algorithms producing same result

Computer Worms

Computer Worms

- A program that copies itself from one computer to another
- Example, Internet Worm of 1988
- Targeted Berkeley, Sun UNIX systems
  - Used virus-like attack to inject instructions into running program and run them
  - To recover, had to disconnect system from Internet and reboot
  - To prevent re-infection, several critical programs had to be patched, recompiled, and reinstalled
- Analysts had to disassemble it to uncover function
- Disabled several thousand systems in 6 or so hours

Stuxnet

- Found in 2010, targeted Siemens centrifuges used in process to enrich uranium
  - Compromised Windows software first, then the PLC in centrifuges
  - Spun them at nonstandard speeds so they tore apart
- Entered system via infected USB stick with a Trojan horse
  - Looked on local network for Windows-based systems to infect
  - if found, infected no more than 3
- On system, checked to see if it was part of a specific industrial control system
  - No: did nothing
  - Ye: acted

Stuxnet (*con't*)

- Tried to download most current version of itself
- Exploited vulnerabilities in infected system's PLC to take control of attached centrifuges
- Believed developed by one or more nation-states due to its complexity, sophistication
- Earlier research showed physical systems vulnerable to attacks from connected computers
- Stuxnet showed these attacks can be launched over the Internet

Bots and Botnets

Bots and Botnets

- *bot*: malware that carries out some action in co-ordination with other bots
- *botnet*: a collection of bots
- *botmaster*: attacker controlling the bots on one or more systems
- *command and control (C&C) server*, *mothership*: system(s) the attacker uses to control the bots
- C&C *channels*: communication paths used to communicate with bots

Life Cycle of a Bot in a Botnet

1. Bot infects system
2. Bot checks for a network connection, looks for either C&C server or another bot it can communicate with
3. Bot gets commands sent by C&C server or other bot
   - These may include adding components to add to what the bot can do
4. Bot executes these commands
   - May send results to somewhere else

Organization of a Botnet

- *Centralized*; each bot communicates directly with C&C server
  - Potential problem: C&C server can become a bottleneck
- *Hierarchical*: C&C server communicates with set of bots, which in turn act as C&C servers for other bots, in a hierarchy
  - Torpig (over 180,000 bots) and Mirai (estimated to have 493,000 bots)
- *Peer-to-peer*: no single C&C server; bots act as peers, and botnet is a peer-to-peer network

IP Flux

- Content delivery networks
  - Netflix and Amazon have many servers
  - Prevent any single server from being overloaded
- *IP flux*
  - Change IP address associated with a particular host name over a very short period of time
- Example: Flame (Fast Flux Botnet)
  - Number of C&C hosts around 100

Other Malware

Logic Bombs

- A program that performs an action that violates the site security policy when some external event occurs
- Example: deletes company's payroll records when one particular record is deleted
  - The "particular record" is the person writing the logic bomb

Adware

- Trojan horse that gathers information for marketing purposes and displays advertisements
- Benign as user had to opt in
- Put it in a banner enticing the user to click on it
- Page may require user to install software to view parts of web site
- If page refreshes automatically, it may direct browser to run an executable
- Some browser plug-ins download, execute files automatically; there may be no indication of this
  - Called *drive-by downloading*
- Example: survey of 900 Android apps
  - 323 had unnecessary permissions

Getting Adware on a System

- Put into software that user downloads
  - Very common with mobile apps
- Problem: app asks for permission to carry out its tasks
  - Some may be unnecessary; often hard for users to minimize permissions set
  - Thus app may have access to camera, microphone, and may be able to make calls without going through dialing interface — and user does not realize this

Spyware

- Trojan horse that records information about the use of a computer for a third party
  - Usually results in compromise of confidential information like keystrokes, passwords, credit card numbers, etc.
  - Information can be stored for retrieval or sent to third party
- Put on a system the way any other malware gets onto system

Example: Pegasus

- Designed for Apple's iPhone
- Sends URL to victim who clicks on it
- First sends HTML file exploiting vulnerability in WebKit
  - Basis for Safari and other browsers
- Loader downloads dynamic load libraries, daemons, other software and installs Pegasus
  - If iPhone has previously been jailbroken, removes all acess to the iPhone provided by the earlier break

Ransomware

- Malware inhibiting use of computer, resources until a ransom is paid
- PC CYBORG (1989) altered AUTOEXEC.BAT
  - Count number of times system was booted
  - On 90th, names of all files on main drive (C:) enciphered and directories hidden
  - User told to send fee to post office box to recover the system
- CryptoLocker (2013) encrypted files and gave victim 100 hours to pay ransom
  - If not, encryption keys destroyed
  - Spread via email as attachments

Theory of Viruses

Theory of Viruses

- Is there a single algorithm that detects computer viruses precisely?
  - Need to define viruses in terms of Turing machines
- It is undecidable whether an arbitrary program contains a computer virus

Defenses

Defenses

- Scanning
- Distinguishing between data, instructions
- Containing
- Behavior analysis
- Limiting sharing
- Statistical analysis

Scanning Defenses

- Malware alters memory contents or disk files
- Compute manipulation detection code (MDC) to generate signature block for data, and save it
- Later, recompute MDC and compare to stored MDC
  - If different, data has changed

Example: *tripwire*

- File system scanner
- Initialization: it computes signature block for each file, saves it
  - Signature consists of file attributes, cryptographic checksums
  - System administrator selects what file attributes go into signature
- Checking file system: run *tripwire*
  - Regenerates file signatures
  - Compares them to stored file signatures and reports any differences
- Files do not contain malicious logic when original signature block generated

Antivirus Programs

- Look for specific "malware signatures"
  - If found, warn user and/or disinfect data
- At first, static sequences of bits, or patterns
- Now also includes patterns of behavior

Behavioral Analysis

- Run suspected malware in a confined area, typically a sandbox, that simulates environment it will execute in
- Monitor it for some period of time
- Look for anything considered “bad”; if it occurs, flag this as malware

Data vs. Instructions

- Malicious logic is both
  - Virus: written to program (data); then executes (instructions)
- Approach: treat "data" and "instructions" as separate types, and require certifying authority to approve conversion
  - Key are assumption that certifying authority will *not* make mistakes and assumption that tools, supporting infrastructure used in certifying process are not corrupt

Containment

- Basis: a user (unknowingly) executes malicious logic, which then executes with all that user's privileges
  - Limiting accessibility of objects should limit spread of malicious logic and effects of its actions
- Approach draws on mechanisms for confinement

Key Points

Key Points

- Malware
- Trajan horse
- Thompson's compiler
- Computer virus
- Computer worms
- Bots and botnets
- Adware, spyware, ransomware
- Theory of viruses
- Defenses