Cristina Nita-Rotaru

CS526: Information security

Malware

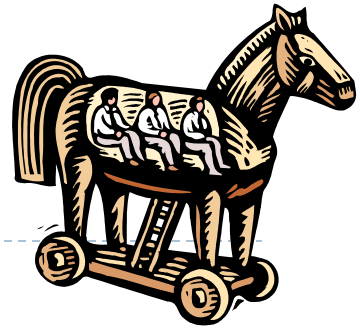
Readings for This Lecture

- Wikipedia
  - Malware
  - Computer Virus
  - Botnet
  - Rootkit
  - Morris Worm

Malware Types

- Concealment:
  - Trojan horses, backdoors (trapdoors), logic bombs, rootkits
- Infectious:
  - Viruses, worms
- Malware for stealing information:
  - Spyware, keyloggers, screen scrapers
- Malware for profit:
  - Dialers, scarewares, ransomware
- Botnets
- Many malwares have characterstics of multiple types

Trojan Horse



- Software that appears to perform a desirable function for the user prior to run or install, but (perhaps in addition to the expected function) steals information or harms the system.
- User tricked into executing Trojan horse
  - Expects (and sees) overt and expected behavior
  - Covertly perform malicious acts with user's authorization

Example: Attacker:

Place the following file

cp /bin/sh /tmp/.xxsh

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

rm ./ls

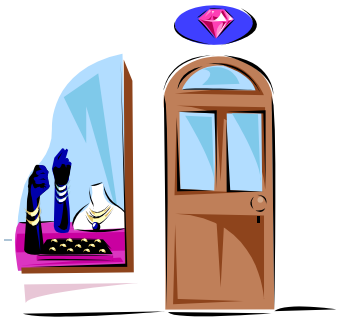
ls $*

as /homes/victim/ls

- *Victim*

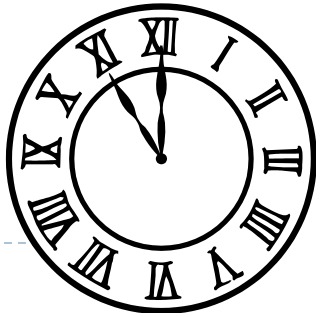
ls

Trapdoor or Backdoor



- Secret entry point into a system
  - Specific user identifier or password that circumvents normal security procedures
- Commonly used by developers
  - Could be included in a compiler

Logic Bomb



- Embedded in legitimate programs
- Activated when specified conditions met
  - E.g., presence/absence of some file; Particular date/time or particular user
- When triggered, typically damages system
  - Modify/delete files/disks

Example of Logic Bomb

- In 1982, the Trans-Siberian Pipeline incident occurred. A KGB operative was to steal the plans for a sophisticated control system and its software from a Canadian firm, for use on their Siberian pipeline. The CIA was tipped off by documents in the Farewell Dossier and had the company insert a logic bomb in the program for sabotage purposes. This eventually resulted in "the most monumental non-nuclear explosion and fire ever seen from space".

Spyware

- Malware that collects little bits of information at a time about users without their knowledge
  - Keyloggers: stealthly tracking and logging key strokes
  - Screen scrapers: stealthly reading data from a computer display
  - May also tracking browsing habit
  - May also re-direct browsing and display ads

Scareware

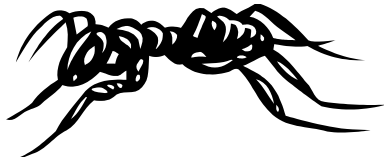
- Software
  - With malicious payloads, or of limited or no benefit
  - Sold by social engineering to cause shock, anxiety, or the perception of a threat
- Rapidly increasing
  - Anti-Phishing Working Group: # of scareware packages rose from 2,850 to 9,287 in 2nd half of 2008
  - In 1st half of 2009, the APWG identified a 583% increase in scareware programs
  - A 2010 study by Google found 11,000 domains hosting fake anti-virus software, accounting for 50% of malware delivered via Internet advertising


SECURITY WARNING!
serious security threat detected
Your computer is infected with Spyware.
Your Security and Privacy are in DANGER.
Spyware programs can steal your credit card numbers and bank information details. The computer can be used for sending spam and you may get popups with adult or any other unwanted content.
If
- You have visited adult or warez websites during past 3 days.
- Your homepage has changed and does not change back.
- Your computer performance has dropped down dramatically.
- You are suspecting someone is watching you.
Then your computer is most likely
INFECTED WITH SPYWARE.
We are sorry, but the trial version is unable to remove these threats.
We strongly recommend you to purchase Full version.
You will get 24x7 friendly support and unlimited protection.
Continue Unprotected
Get Full version of SpySheriff Now!

Ransomware

- Holds a computer system, or the data it contains, hostage against its user by demanding a ransom
  - Disable an essential system service or lock the display at system startup
  - Encrypt some of the user's personal files, originally referred to as cryptoviruses, cryptotrojans or cryptoworms
- Victim user has to
  - Enter a code obtainable only after wiring payment to the attacker or sending an SMS message
  - Buy a decryption or removal tool

Virus



- Attach itself to a host (often a program) and replicate itself
- Self-replicating code
  - Self-replicating Trojan horses
  - Alters normal code with "infected" version
- Operates when infected code executed
  - If spread condition then
    - For target files
      - if not infected then alter to include virus
  - Perform malicious action
  - Execute normal program

Worm

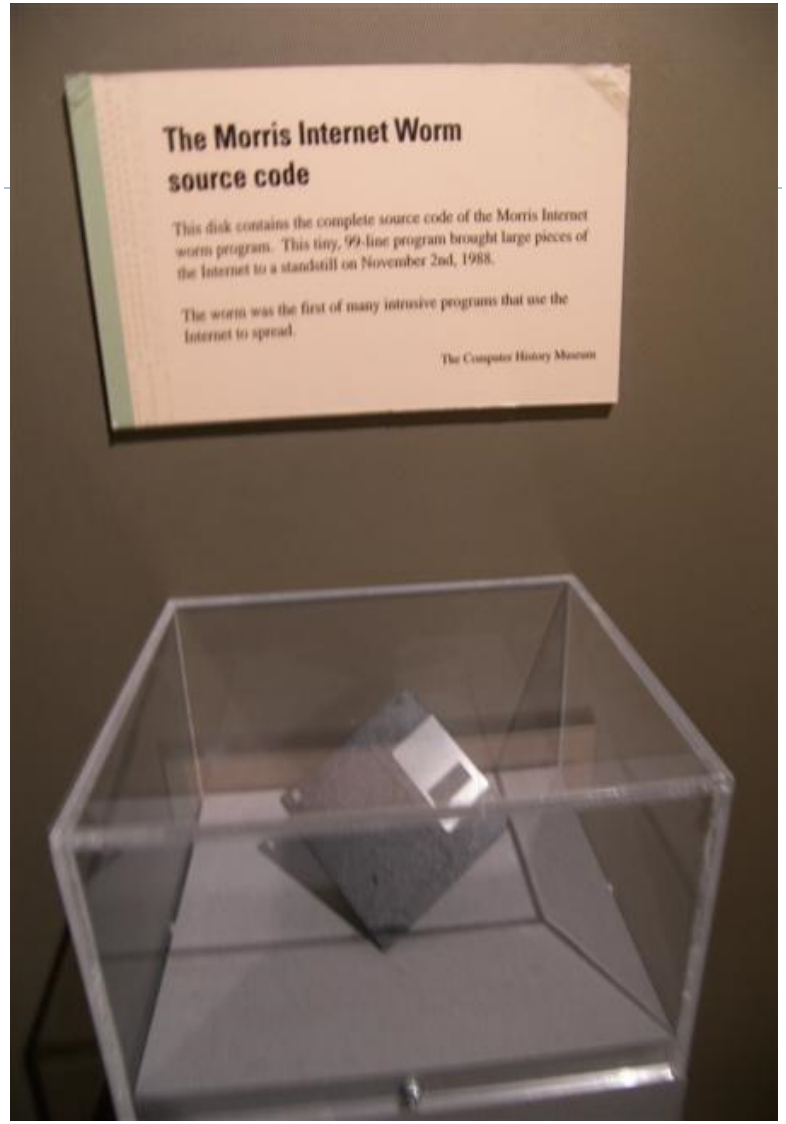
- Self-replicating malware that does not require a host program
- Propagates a fully working version of itself to other machines
- Carries a payload performing hidden tasks
  - Backdoors, spam relays, DDoS agents; …
- Phases
  - Probing → Exploitation → Replication → Payload

General Worm Trends

- Speed of spreading
  - Slow to fast to stealthy
- Vector of infection
  - Single to varied
  - Exploiting software vulnerabilities to exploiting human vulnerabilities
- Payloads
  - From "no malicious payloads beyond spreading" to botnets, spywares, and physical systems

Morris Worm (November 1988)

- First major worm
- Written by Robert Morris
  - Son of former chief scientist of NSA's National Computer Security Center
  - Currently a Professor at MIT



Morris Worm Description

- Two parts
  - Main 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

Vector 1: Debug feature of sendmail

- Sendmail
  - Listens on port 25 (SMTP port)
  - Some systems back then compiled it with DEBUG option on
- Debug feature gives
  - The ability to send a shell script and execute on the host

Vector 2: Exploiting fingerd

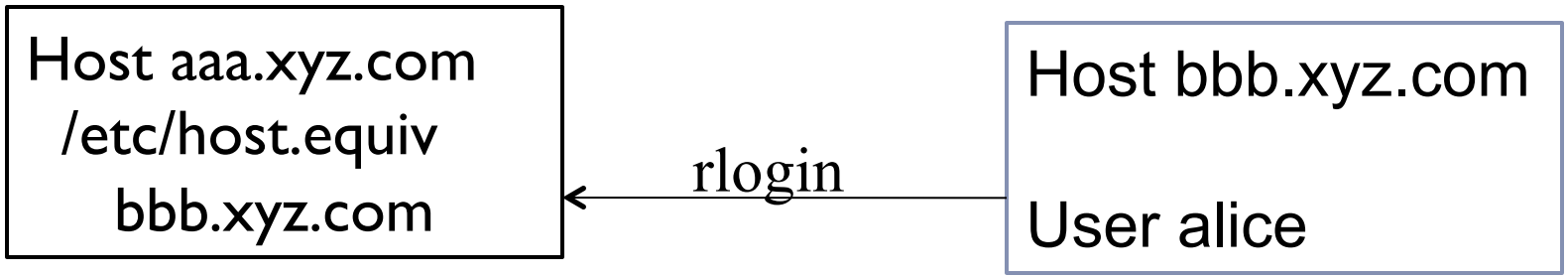
- What does finger do?
- Finger output
  - arthur.cs.purdue.edu% finger ninghui
  - Login name: ninghui In real life: Ninghui Li
  - Directory: /homes/ninghui Shell: /bin/csh
  - Since Sep 28 14:36:12 on pts/15 from csdhcp-120-173 (9 seconds idle)
  - New mail received Tue Sep 28 14:36:04 2010;
  - unread since Tue Sep 28 14:36:05 2010
  - No Plan.

Vector 2: Exploiting fingerd

- Fingerd
  - Listen on port 79
- It uses the function gets
  - Fingerd expects an input string
  - Worm writes long string to internal 512-byte buffer
- Overrides return address to jump to shell code

Vector 3: Exploiting Trust in Remote Login

- Remote login on UNIX
  - rlogin, rsh
- Trusting mechanism
  - Trusted machines have the same user accounts
  - Users from trusted machines
  - /etc/host.equiv – system wide trusted hosts file
  - /.rhosts and ~/.rhosts – users' trusted hosts file



Vector 3: Exploiting Trust in Remote Login

- Worm exploited trust information
  - Examining trusted hosts files
  - Assume reciprocal trust
    - If X trusts Y, then maybe Y trusts X
- Password cracking
  - Worm coming in through fingerd was running as daemon (not root) so needed to break into accounts to use .rhosts feature
  - Read /etc/passwd, used ~400 common password strings & local dictionary to do a dictionary attack

Other Features of The Worm

- Self-hiding
  - Program is shown as 'sh' when ps
  - Files didn't show up in ls
  - Find targets using several mechanisms:
    - 'netstat -r -n ', /etc/hosts, …
- Compromise multiple hosts in parallel
  - When worm successfully connects, forks a child to continue the infection while the parent keeps trying new hosts
- Worm has no malicious payload
- Where does the damage come from?

Damage

- One host may be repeatedly compromised
- Supposedly designed to gauge the size of the Internet
- The following bug made it more damaging
  - Asks a host whether it is compromised; however, even if it answers yes, still compromise it with probability 1/8

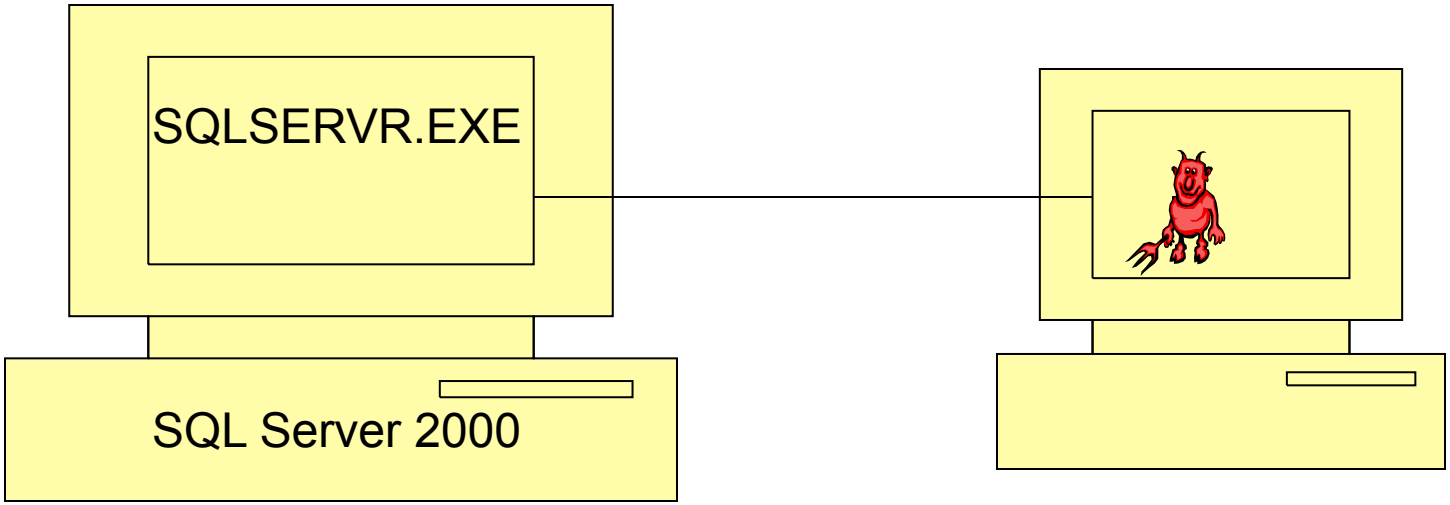
Increasing propagation speed

- Code Red, July 2001
  - Affects Microsoft Index Server 2.0,
  - Exploits known buffer overflow in Idq.dll
  - Vulnerable population (360,000 servers) infected in 14 hours

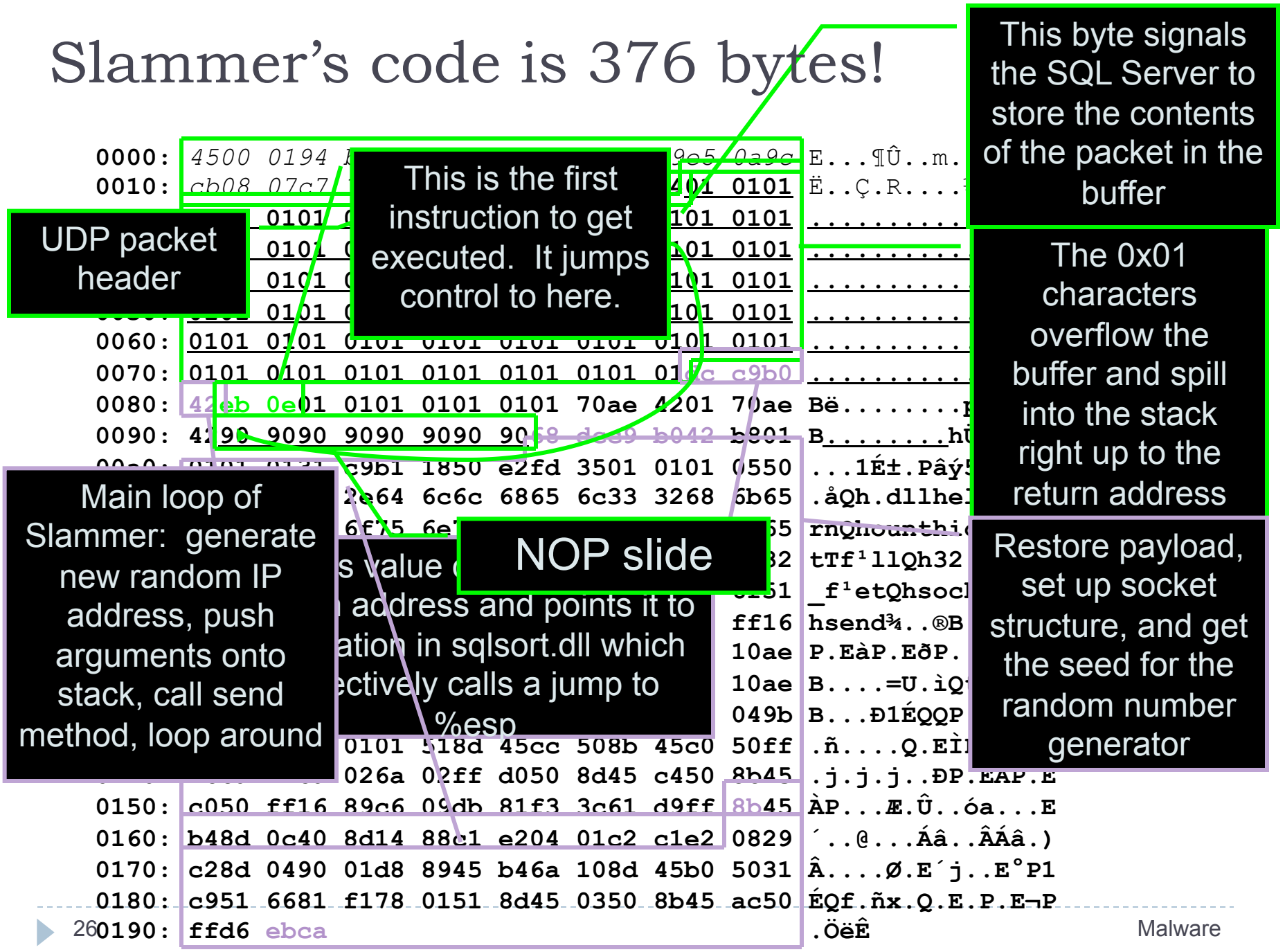
- SQL Slammer, January 2003
  - Affects in Microsoft SQL 2000
  - Exploits known months ahead of worm outbreak
    - Buffer overflow vulnerability reported in June 2002
    - Patched released in July 2002 (Bulletin MS02-39)
  - Vulnerable population infected in less than 10 minutes

Slammer Worm, Jan 2003

- MS SQL Server 2000 receives a request of the worm
  - SQLSERVR.EXE process listens on UDP Port 1434



Slammer's code is 376 bytes!



Research Worms

- Warhol Worms
  - infect all vulnerable hosts in 15 minutes – 1 hour
  - optimized scanning
    - initial hit list of potentially vulnerable hosts
    - local subnet scanning
    - permutation scanning for complete, self-coordinated coverage
- Flash Worms
  - infect all vulnerable hosts in 30 seconds
  - determine complete hit list of servers with relevant service open and include it with the worm

Email Worms: Spreading as Email Attachments

- Love Bug worm (ILOVEYOU worm) (2000):
  - May 3, 2000: 5.5 to 10 billion dollars in damage
- MyDoom worm (2004)
  - First identified in 26 January 2004:
  - On 1 February 2004, about 1 million computers infected with Mydoom begin a massive DDoS attack against the SCO group

Nimda Worm (September 18, 2001)

- Key Vulnerability to Exploit
  - Microsoft Security Bulletin (MS01-020): March 29, 2001
  - A logic bug in IE's rendering of HTML
  - Specially crafted HTML email can cause the launching of an embedded email
- Vector 1: e-mails itself as an attachment (every 10 days)
  - runs once viewed in preview plane
- Vector 2: copies itself to shared disk drives on networked PCs
  - Why this may lead to propagating to other hosts?

Nimda Worm

- Vector 3: Exploits various IIS directory traversal vulnerabilities
  - Use crafted URL to cause a command executing at
  - Example of a directory traversal attack:
    - http://address.of.iis5.system/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir+c:\
- Vector 4: Exploit backdoors left by earlier worms
- Vector 5: Appends JavaScript code to Web pages

```
<script language="JavaScript">
window.open("readme.eml", null, "resizable=no,top=6000,left=6000")
</script>
```

Nimda Worm

- 'Nimda fix' Trojan disguised as security bulletin
  - claims to be from SecurityFocus and TrendMicro
  - comes in file named FIX_NIMDA.exe
    - TrendMicro calls their free Nimda removal tool FIX_NIMDA.com

Storm botnet

- First detected in Jan 2007
- Vectors (primarily social engineering):
  - Email attachments
  - Download program to show a video
  - Drive-by exploits
- DDoS spam fighting sites, and whichever host discovered to investigate the botnet
- Peer-to-peer communications among bots
  - for asking for C&C server

Zombie & Botnet

- Secretly takes over another networked computer by exploiting software flows
- Builds the compromised computers into a zombie network or botnet
  - a collection of compromised machines running programs, usually referred to as worms, Trojan horses, or backdoors, under a common command and control infrastructure.
- Uses it to indirectly launch attacks
  - E.g., DDoS, phishing, spamming, cracking

Rootkit

- A rootkit is software that enables continued privileged access to a computer while actively hiding its presence from administrators by subverting standard operating system functionality or other applications.
- Emphasis is on hiding information from administrators' view, so that malware is not detected
  - E.g., hiding processes, files, opened network connections, etc
- Example: Sony BMG copy protection rootkit scandal
  - In 2005, Sony BMG included Extended Copy Protection on music CDs, which are automatically installed on Windows on CDs are played.

Types of Rootkits

- User-level rootkits
  - Replace utilities such as ps, ls, ifconfig, etc
  - Replace key libraries
  - Detectable by utilities like tripwire
- Kernel-level rootkits
  - Replace or hook key kernel functions
  - Through, e.g., loadable kernel modules or direct kernel memory access
  - A common detection strategy: compare the view obtained by enumerating kernel data structures with that obtained by the API interface
  - Can be defended by kernel-driver signing (required by 64-bit windows)

More Rootkits

- Bootkit (variant of kernel-level rootkit)
  - Replace the boot loader (master boot record)
  - Used to attack full disk encryption key
  - Malicious boot loader can intercept encryption keys or disable requirement for kernel-driver signing
- Hypervisor-level rootkits
- Hardware/firmware rootkits
- Whoever gets to the lower level has the upper hand

How does a computer get infected with malware or being intruded?

- Executes malicious code via user actions (email attachment, download and execute trojan horses, or inserting USB drives)
- Buggy programs accept malicious input
  - daemon programs that receive network traffic
  - client programs (e.g., web browser, mail client) that receive input data from network
  - programs read malicious files with buggy file reader program
- Configuration errors (e.g., weak passwords, guest accounts, DEBUG options, etc)
- Physical access to computer

Spyware

- Spyware: software designed to intercept or take partial control over the user's interaction with the computer, without the user's informed consent
  - secretly monitors the user's behavior
  - collect various types of personal information

Spyware Techniques

- Log keystrokes
- Collect web history
- Scan documents on hard disk

Types of Spyware

- Spyware-infected executables
  - Content-type header
  - URL extension

- Drive-by downloads (DB-DL):
  - Malicious web content (Javascript embedded in HTML)
  - Produce event triggers

Spyware Functions

- Spyware-infected executables
  - Contain various spyware functions
  - Executables may have multiple functions

spyware function	May 2005	October 2005
keylogging	0.04%	0.15%
dialer	0.14%	0.9%
Trojan downloader	9.1%	13%
browser hijacker	60%	85%
adware	91%	75%

Event Triggers for DB-DLs

- Event occurs that matches a trigger
- Trigger Conditions
  - Process creation
  - File activity (creation)
  - Suspicious process (file modification)
  - Registry file modified
  - Browser/OS crash

2: Malware defenses

Anti-Virus Software

- Signature-based detection
  - Uses pattern matching
  - Searches for known patterns of data belonging to malwares in executable programs or other types of files
  - Maintains and updates a blacklist of signatures
- Problems
  - Cannot detect new malwares, variants of malwares, etc.
  - Hard to keep up with new malware
    - More malwares are created each day than benign programs

Polymorphic Malwares

- Uses a polymorphic engine (a mutation engine or mutating engine) to generate multiple copies of the same malware that look different
  - E.g., serve a different version to each computer subject to a drive-by download attack
- Typically encrypts the majority of the code, each time with a different key is used
- Weakness: decryption code often remains the same

Metamorphic Malware

- A malware automatically changes itself each time it propagates
- Each new version has different code, though the same functionality
- Uses techniques that include
  - Adding varying lengths of NOP instructions, permuting use of registers, add useless instructions, use functional equivalent instructions, reorder functions, reorder data structures, etc.

Semantic, or Heuristics Based Malware Detection

- Uses patterns that looks for specific code behavior instead of specific strings
- Execute the program to identify potentially malicious behavior
- Main limitations
  - Performance overhead
  - Potential of high false positives

Application Whitelisting

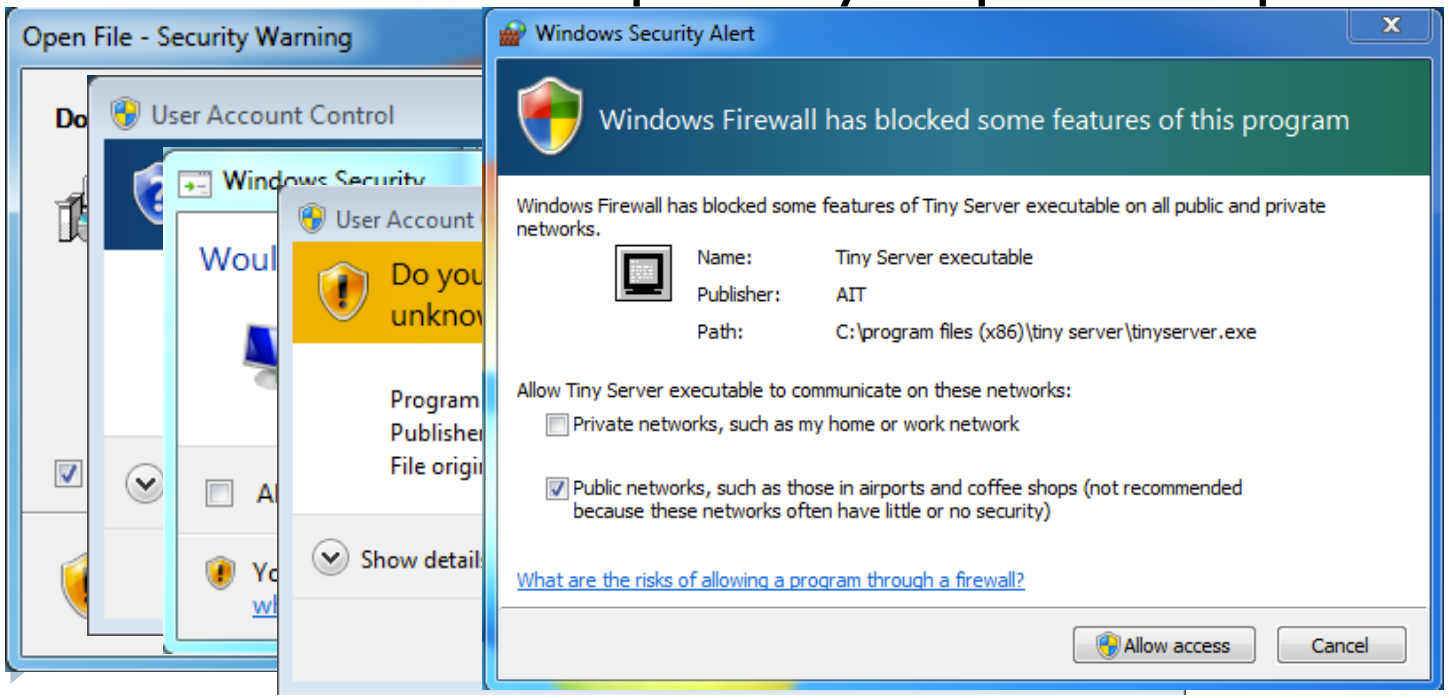
- Instead of finding malwares and stop then, list all known good/allowed programs and only run them.
- Typically deployed by enterprise, who can afford to maintain a list of allowed programs

CodeShield: Personalized Application Whitelisting

- Practical Application Whitelisting on Windows desktops
  - Give the user flexibility
    - Allow the user to add software to the whitelist
  - Maintain the security advantage of whitelisting
    - New software isn't automatically allowed onto whitelist
    - Protect against certain types of Social Engineering attacks
- Not designed to stop all infection
  - Make persistence harder
  - Prevent most current attacks
- Focus on usability
  - A key challenge of many security mechanisms is the ability for a typical user to understand and use it

Analysis of Existing Security Interface

- Users are asked questions they do not know how to answer and presented with info that is difficult to understand
- Users are asked to make a decision too often
- Users are made to passively respond and provided an



Design Principles

- Reduce – decrease the number of times users are asked to make a decisions
- Simplify – ask questions that a user can understand
- Safe – do not provide an easy and insecure way out.
- Active – avoid passively respond to security prompts

Design of Personalized Whitelisting

- Normal Mode
- Installation Mode

Only execute known software	Execute all software
Trusted Signatures = add to whitelist	Executed = added to whitelist
Trusted Installers = add to whitelist	Written = added to whitelist
All else blocked	Try to exit installation mode quickly

- "Stopping" vs "Warning" approach
- The decision a user needs to make
  - "Do I want to install new software now"

Design Principles in Practice

- Reduce – there is a single security decision to make for installing any application
- Simplify – this paradigm more closely matches how typical users understand their actions. "I' m adding something new"
- Safe – Not allowing new code is the easiest action
- Active – In order to add new software, the user needs to actively participate and initiate the action

Installation Mode vs Normal Mode

- This dual mode can more closely match the mental model of a typical user
  - Users may not understand "Do you want to allow this program to make changes"
  - But most can be educated about "Do you want to add something new to your computer right now"

- Furthermore, users can be educated about when not to enter installation mode

The Burden Benefit of Installation Mode

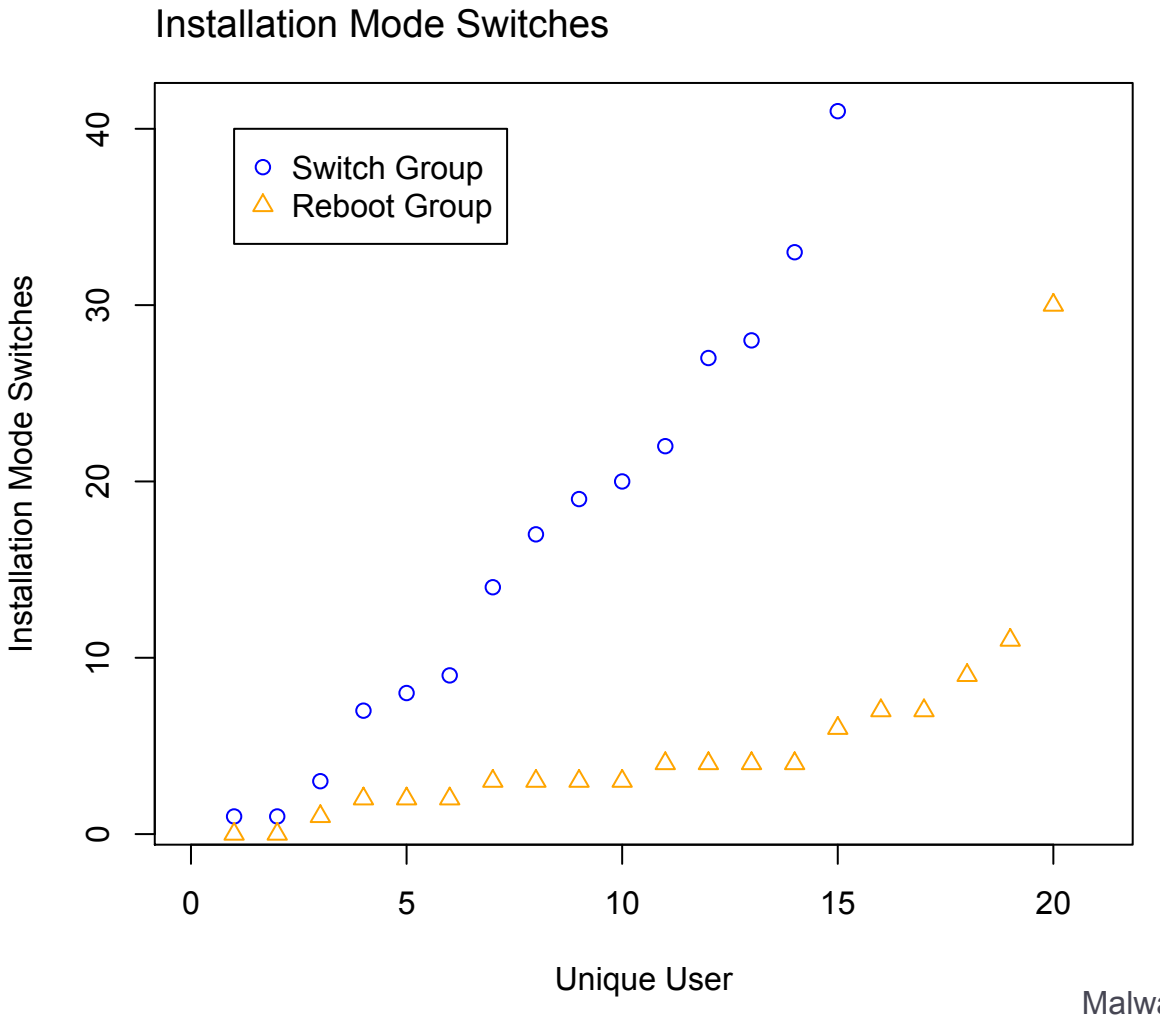
- Simple switch to installation mode
  - Advantage – it's easy
  - Disadvantage – user may enter installation mode often
- High overhead switch to installation mode (ex. reboot)
  - Advantage – it makes a user less likely to switch unless needed
  - Disadvantage – high overhead may lead to annoyance
- Advantage of reboot
  - Clear out memory, malware in memory can't take advantage of installation mode
  - Minimal number of applications active just after reboot

User Study

- 35 person user study running CodeShield for 6 weeks
- Longest use of CodeShield is 203 days (8 switches, 25 days/switch), next is 168 days (13 switches, 13 days/switch).
- Participants sat through a 30 minute training session
- Then installed CodeShield (standalone installer)
- Take a survey, Run for 6 weeks, Take a survey
- Uninstall if they want to
- 7 of 38 participants continued to use CodeShield at least 3 months after study ended.
  - 5 were using reboot only client
  - 2 using switch or reboot

Switches to Installation Mode

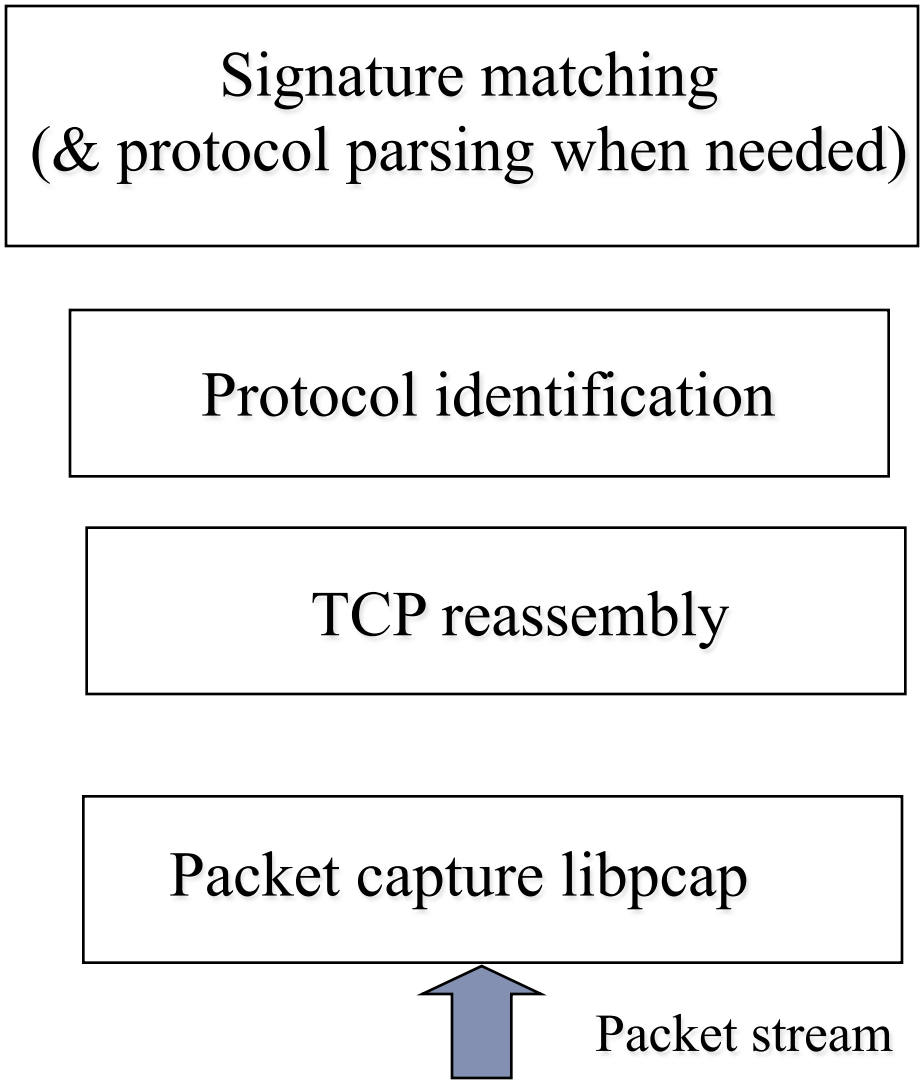
- Switch
  - Median - 17
  - Useful - 13
- Reboot
  - Median - 3.5
  - Useful - 3.5



Network IDSs

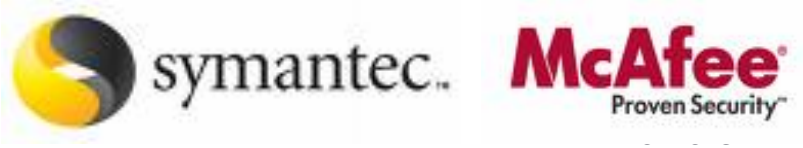
- Deploying sensors at strategic locations
  - E.G., Packet sniffing via tcpdump at routers
- Inspecting network traffic
  - Watch for violations of protocols and unusual connection patterns
- Monitoring user activities
  - Look into the data portions of the packets for malicious code
- May be easily defeated by encryption
  - Data portions and some header information can be encrypted
  - The decryption engine may still be there, especially for exploit

Architecture of Network IDS

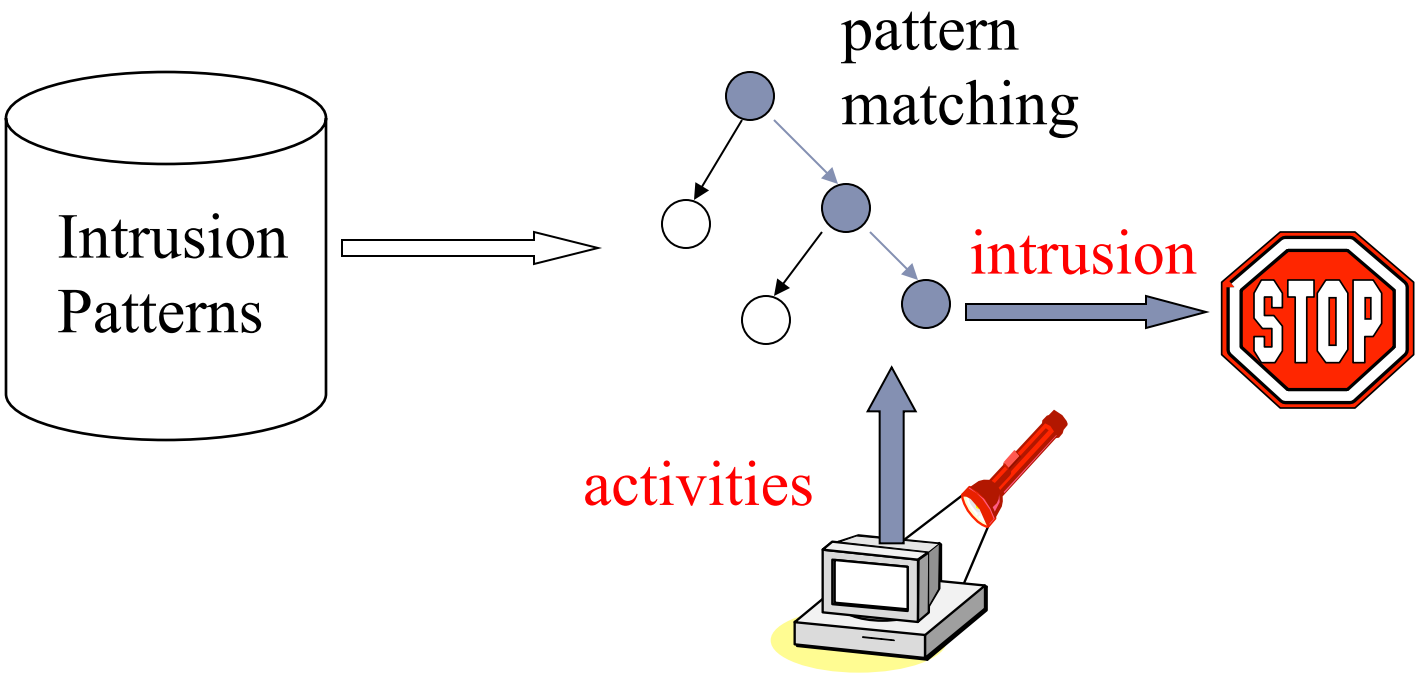


Host-Based IDSs

- Running on a single host
- Monitoring
  - Shell commands
  - System call sequences
  - Etc.



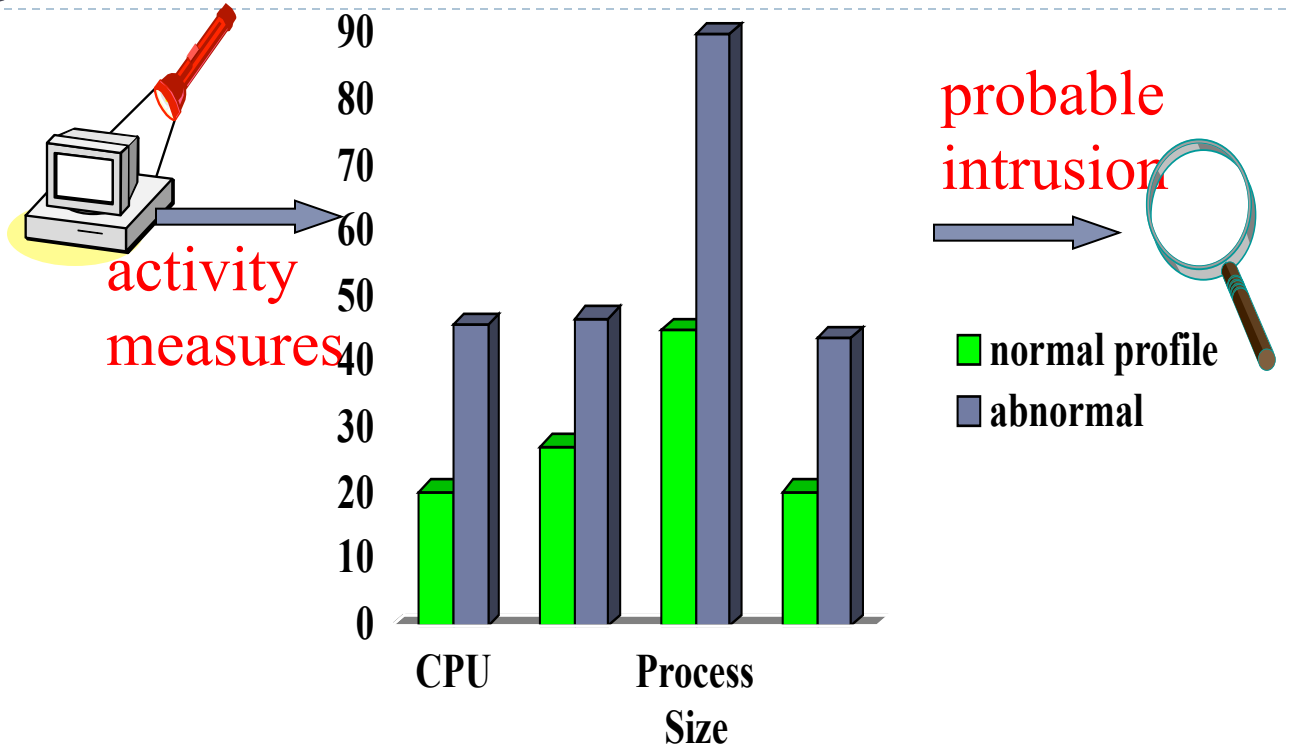
Misuse Detection (aka Signature detection)



Example: ***if*** (src_ip == dst_ip) ***then*** "land attack"

Can't detect new attacks

Anomaly Detection



Problem: Relatively high false positive rate

- Anomalies can just be new normal activities.
- Anomalies caused by other element faults
  - E.g., router failure or misconfiguration, P2P misconfiguration

Problems with Current IDSs

- Inaccuracy for exploit based signatures
- Cannot recognize unknown anomalies/intrusions
- Cannot provide quality info for forensics or situational-aware analysis
  - Hard to differentiate malicious events with unintentional anomalies
    - Anomalies can be caused by network element faults, e.g., router misconfiguration, link failures, etc., or application (such as P2P) misconfiguration
  - Cannot tell the situational-aware info: attack scope/target/strategy, attacker (botnet) size, etc.

Key Metrics of IDS/IPS

- Algorithm
  - Alarm: A;
  - Intrusion: I
  - Detection (true alarm) rate: P(A|I)
    - False negative rate P(¬A|I)
  - False alarm (aka, false positive) rate: P(A|¬I)
    - True negative rate P(¬A|¬I)