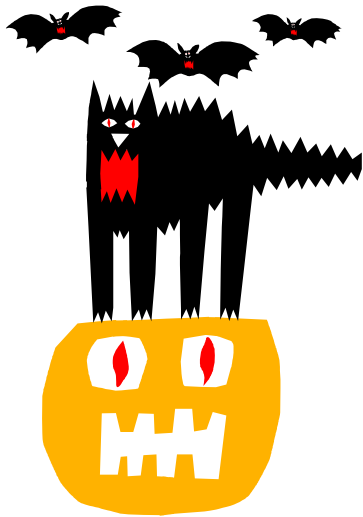


# Firewall Fairytales

(and other scary stories)



**Dr. Steve G. Belovich**



## Note:

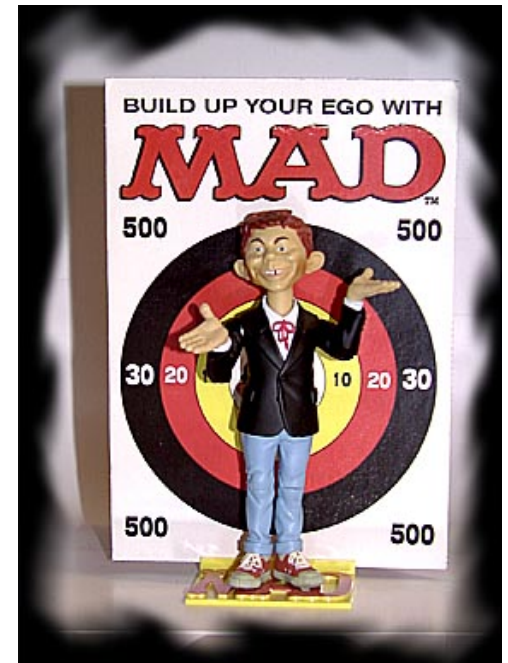
Dr. Steve G. Belovich is CEO of IQware. The conflict of interest was resolved by peer review of slide content.

# Cyber Attacks & Viruses

What, Me Worry?

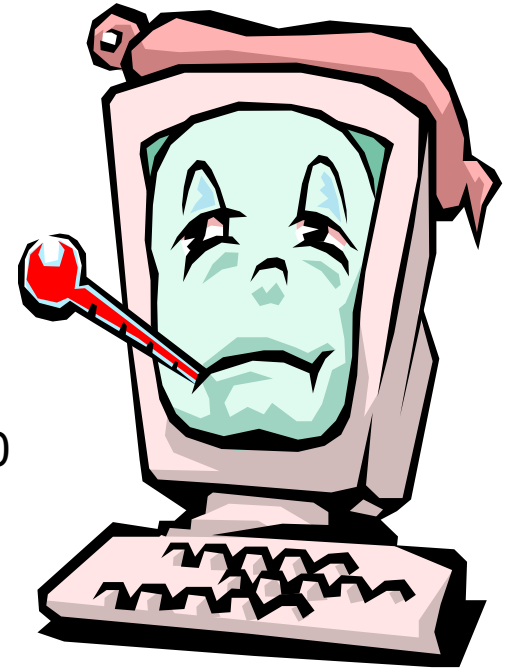
## We Care Because:

- They cause damage.
- Attacks are increasing in frequency & severity.
- Cleanup is expensive.
- Cost of business disruption is a lot higher.
- Cost of data theft is higher still.
- Cost of a “time bomb” is incalculable.



# Diary of a Cyber Attack

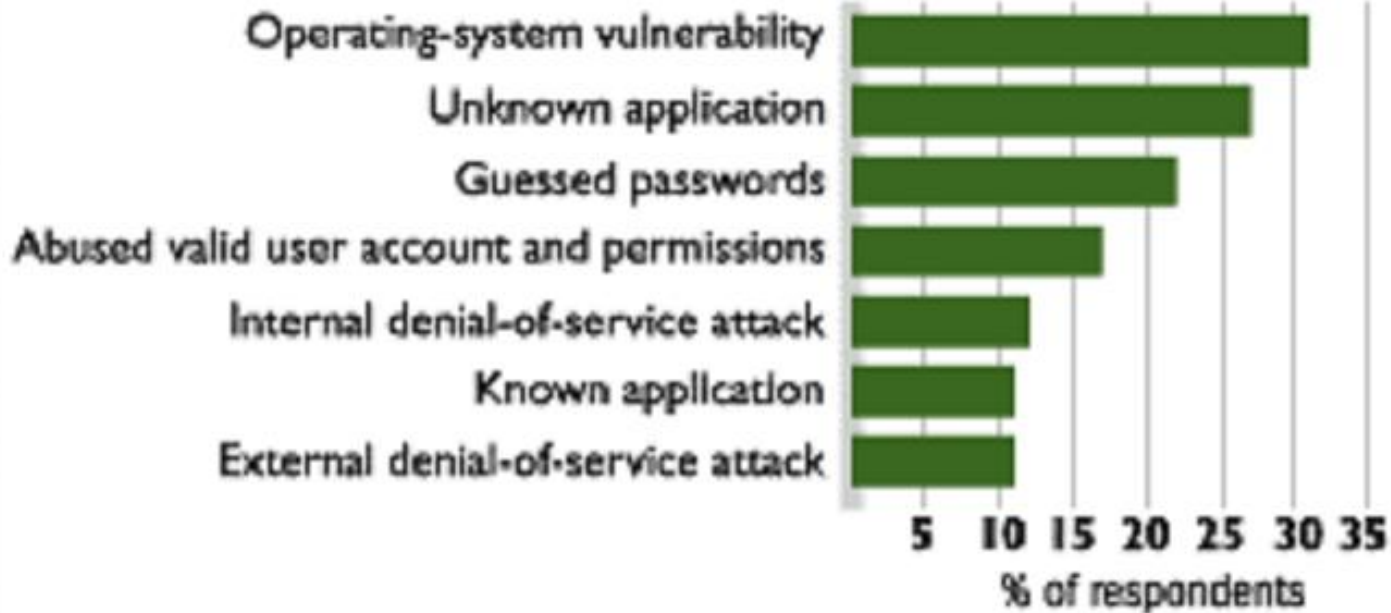
- “SQL Slammer” hit at 12:30am on 1/25/03.
- Exploited known flaws in desktop O/S, SQL Server and MDSE 2000 apps.
- Infected over 67,000 machines within 10 minutes.
- Number of copies doubled every 8.5 seconds.
- Looked for vulnerable machines at rate of 55,000,000 per second within three minutes of its appearance.
- Lack of Internet bandwidth limited replication rate.
- Bank of America lost communication with its ATMs.
- Microsoft’s own network was affected because they failed to install their own patch.
- In October 2002, Microsoft released a patch for another SQL Server problem that, if installed, would have made security-patched systems vulnerable again.



# Cyber Attack Facts

## Attack Methods

What were the primary methods of attack used by intruders?

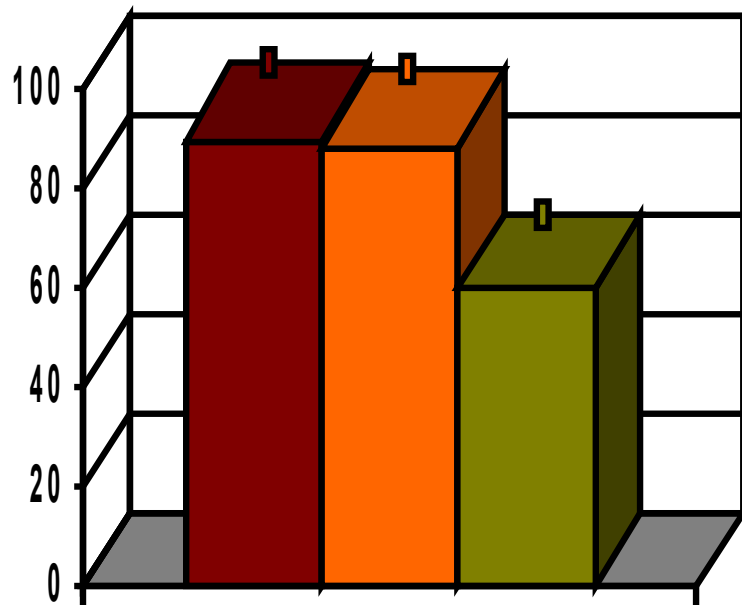


Note: Multiple responses allowed.

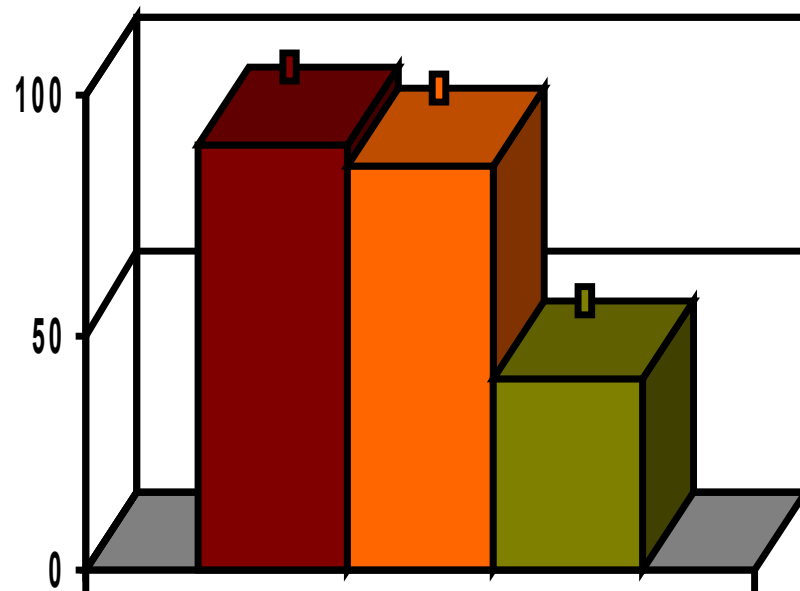
DATA: INFORMATIONWEEK RESEARCH GLOBAL INFORMATION SECURITY SURVEY OF 4,500 SECURITY PROFESSIONALS, 2001

# Cyber Attack Survey

## Defense Methods



## Defense Effectiveness



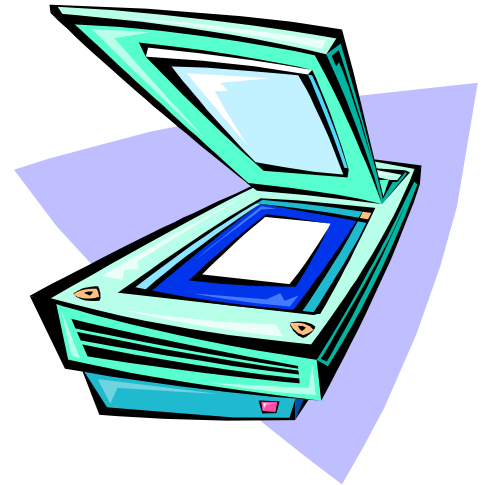
# Hacker Methods: Footprinting

- Footprinting - discovering an organization's network information:
  - Internet - domain names, network blocks, TCP services, system architecture, IDS, ACLs, banners, routing tables, SNMP, etc.
  - Intranet - protocols in use (e.g., IP, IPX, NetBUI, etc.), network blocks, IP addresses of reachable systems, etc.
  - Remote access - VPNs & related protocols, phone numbers
  - Extranet - Access control mechanism, connection origination/destination.



# Hacker Methods: Scanning

- Scanning - discovering which systems are alive and what they are running.
  - Ping Sweeps - sending ICMP ECHO(type 8) packets to target systems in a range of IP addresses to get ICMP ECHO\_REPLY.
  - Port Scans - connecting to TCP (or UDP) ports on target system to identify services that are running.
  - Active Stack Fingerprinting - sending packets to target system and examining IP stack to detect an O/S specific implementation (e.g.,FIN packet probe, TCP initial window size, ACK value, ICMP message quoting, etc.).
  - Passive Stack Fingerprinting - passively monitoring network traffic for same purpose as above.



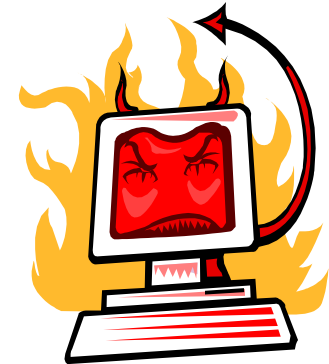
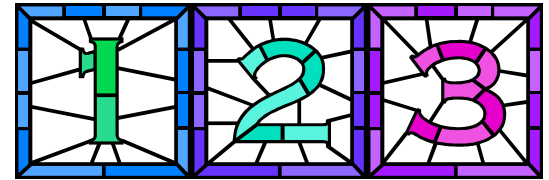
## Automated tools

- 1) [Superscan - www.foundstone.com/rdlabs/termsfuse.php?filename=superscan.exe](http://www.foundstone.com/rdlabs/termsfuse.php?filename=superscan.exe)
- 2) [NetScanTools Pro 2000 - www.nwpsw.com](http://www.nwpsw.com)

ICMP: Internet Control Messaging Protocol  
TCP: Transmission Control Protocol  
UDP: User Datagram Protocol

# Hacker Methods: Enumeration

- Enumeration - identifying valid information about the following areas:
  - Network Resources and Shares
  - Users and Groups
  - Applications and Banners
- Enumeration techniques are O/S specific, including:
  - Password guessing
  - Eavesdropping on network password exchange
  - Denial-of-Service (DOS)
  - Buffer Overflows



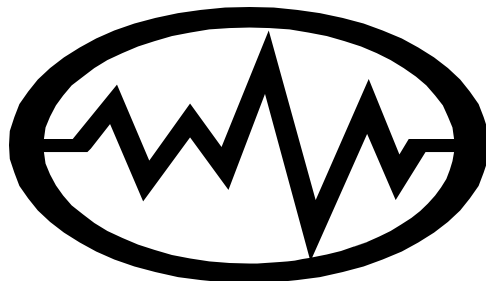
## Buffer Overflow Exploiting

- 1) Phrack 49/14 & 55/15, [www.phrack.org](http://www.phrack.org)
- 2) [www.cultdeadcow.com](http://www.cultdeadcow.com) (general hacking stuff)

# Network Hacking



- Goal:
  - Own the network by listening to sensitive traffic & redirecting traffic to unauthorized systems.
- Methods:
  - Discovery - tracerouting, port scanning, O/S identification, Cisco (et al) Packet Leakage, banner grabbing, SNMP protocol insecurities, various router/gateway-specific weaknesses, etc.
  - Back Doors - default accounts, device specific weaknesses
  - Shared vs. Switched - detecting the network media/protocol, ARP Redirecting, RIP spoofing.
  - Wireless Network Hacking - War Driving, WEP attacks (exploiting implementation of RC4 stream cipher and the 50% chance of repeat initialization vector every 4,823 packets), WAP/WTLS attacks, etc.

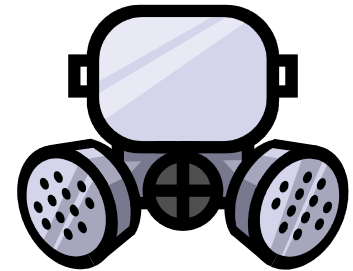


**ARP:** Address Resolution Protocol - maps a 32-bit IP to a 48-bit physical hardware address.  
**RIP:** Routing Information Protocol  
**SNMP:** Simple Network Management Protocol.  
**WAP:** Wireless Application Protocol (cellular phone).  
**WTLS:** Wireless Transport Layer Security, protects data transferred between cellular phones and WAP gateway.

# Firewalls - Overview



- Described by Cheswick and Bellovin in their book.
- Two Main Types
  - Application proxies - more secure, very restrictive on performance, used mostly for out-bound traffic.
  - Packet Filtering gateways - less secure, allows higher network throughput, used mostly for in-bound traffic.
- Most have built-in security flaws due to design errors
- Many breaches due to misconfiguration and/or misadministration.



# Firewall Hacking



## Identification

- Port scanning - done randomly to avoid IDS response.
- Banner grabbing - helps determine type and version
- Route tracing - See the responses and deduce location of firewall.

## Scanning Through Firewalls

- Raw Packet Transmission - “Hping” tool (see [www.kyuzz.org/antirez/hping.html](http://www.kyuzz.org/antirez/hping.html)) sends TCP packets to ports and analyzes responses.
- Firewalking - a popular tool at [www.packetfactory.net/projects/firewalk](http://www.packetfactory.net/projects/firewalk) that builds packets with IP TTL that expires one hop past firewall. If firewall allows it, it will pass, expire and generate a “ICMP TTL expired in transit” error message. Otherwise, packet will be dropped and generate a null response or ICMP type 13 admin prohibited filter packet.

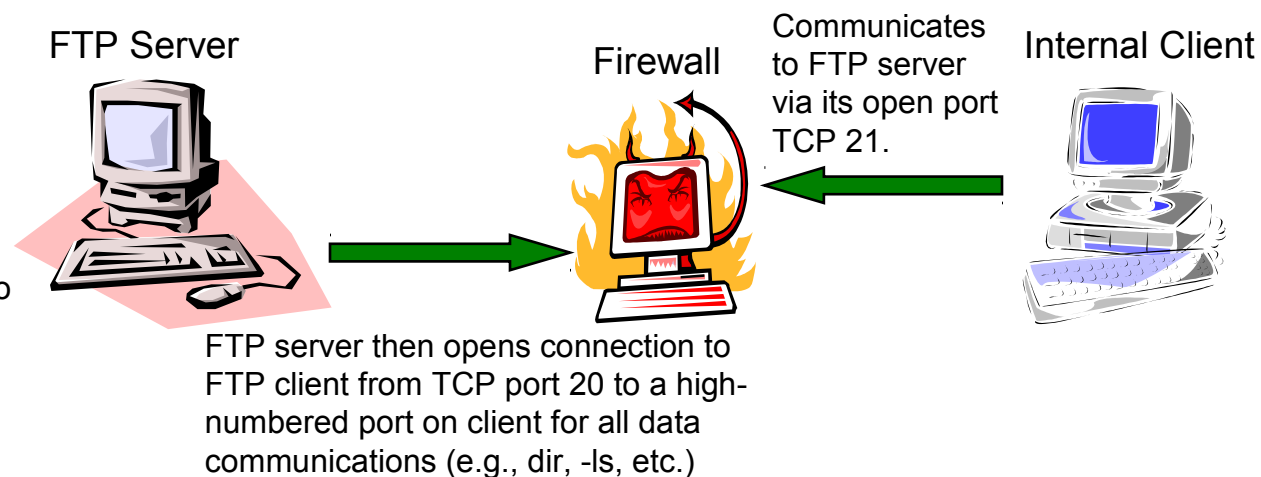


# Firewall Hacking - 2

- Source Port Scanning - works on firewalls that do not keep state information.

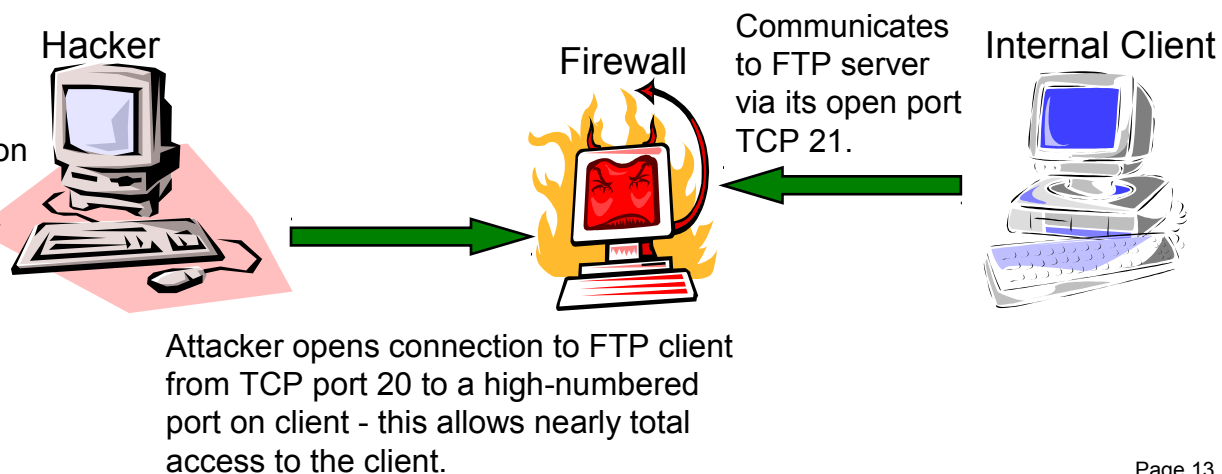
## Normal Operation

Packet filtering firewall must keep open all connections from source port TCP 20 to high-numbered ports on its internal network to allow FTP data channel to pass through firewall (TCP 53 zone transfers are also usable for this attack)



## Attack Scenario

Packet filtering firewall does not maintain state and cannot track one TCP connection with another. So, **all connections from source port 20 to high numbered ports on its internal network are allowed** and effectively pass through the firewall unhindered.



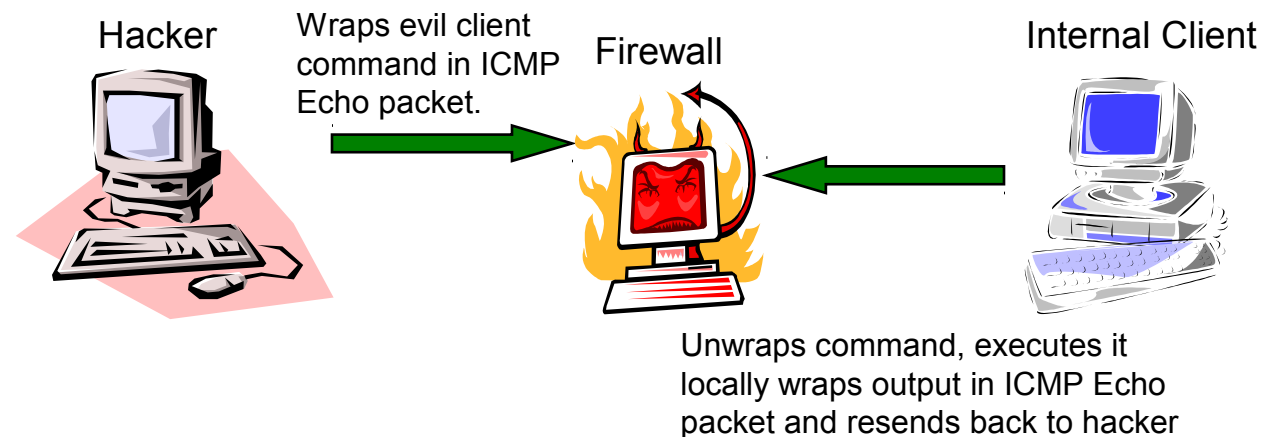
# Firewall Hacking - 3

## □ Packet Filtering

- Weak ACLs - broad ACLs allow unintended traffic creating vulnerabilities.
- Checkpoint Firewall Weaknesses - versions 3.0 & 4.0 open ports by default. UDP 520 (RIP), UDP 53 (DNS lookups), & TCP 53 (zone xfers) are allowed from any host to any host. Creates potential weakness once hacker has already compromised a system beyond the firewall (or used a trojan).
- ICMP and UDP tunneling - wrapping real data in a packet header. Firewalls & routers that let ICMP Echo, ICMP Echo Reply & UDP packets through are vulnerable to this attack. Relies on an already compromised client box beyond the firewall.

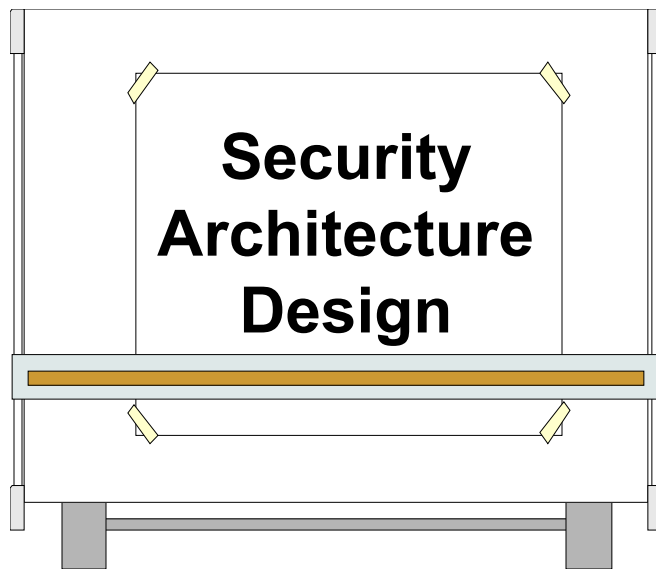
### **Tunneling Example**

Depends upon an already compromised client box - very easy to do via a trojan.



## Why is There No Simple Security Solution?

# Software Structure & Security



- Most software “tools” are really “building blocks” - application inherits all security errors and omissions.
- Application also inherits all security errors and omissions of all lower layers (O/S, utilities, etc.)
- Security must be designed into the software from the ground up - it can’t be “bolted on” ex post facto.
- Software **architecture** is critical to security and reliability - the language or development environment is not.

## Why is There No Simple Security Solution?

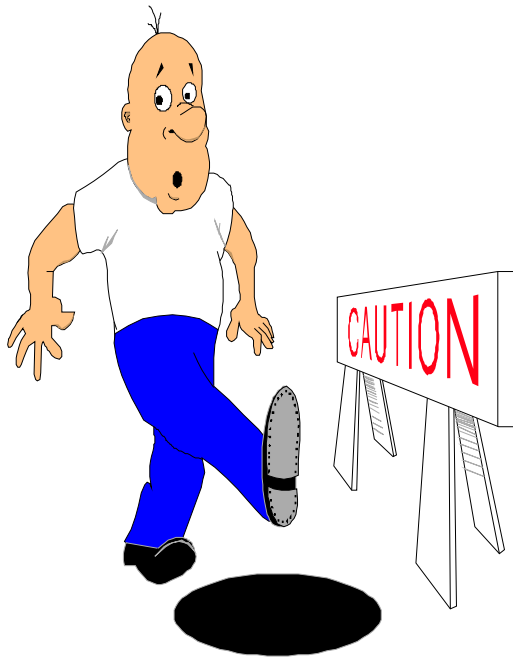
# Software Development & Security



- Software creation is largely a manual process - time consuming and fraught with errors.
- Compilers cannot check for security errors, logic errors and plain stupidity.
- O/S changes to handle new features may cause new security “leaks”.
- O/S change force apps to change because they “see” O/S only through the API (Application Programmer Interface).
- Proper security-oriented changes to desktop O/S would force significant changes to the **entire installed application software base.**

# Why is There No Simple Security Solution?

# Software Security Quality Assurance



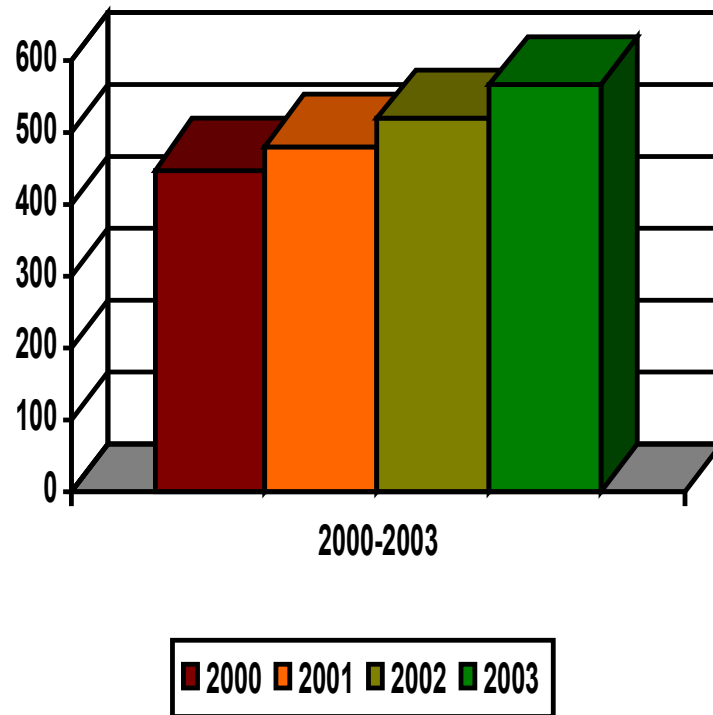
- No uniform quality standards.
- No uniform security standards.
- No uniform testing standards.
- No uniform production standards.
- Not enough research is done in this area.
- Not taught in universities.
- Not enough knowledge available.

## Why is There No Simple Security Solution?

# Economic & Social Forces

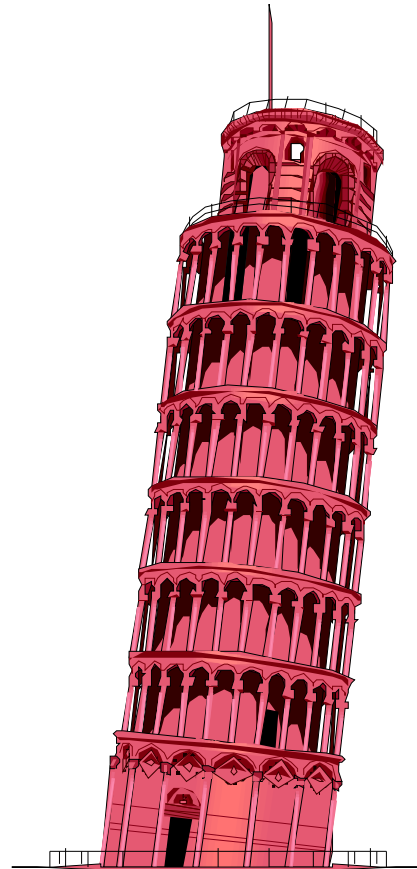
- Demand for programming has far exceeded competent supply.
- Growth rate of IT market is 9%
- Top 10 vendors have 55% of market share & 67% of workforce in packaged software.
- Software industry is not well-understood.
- Market surveys verify that consumers want new features (whether they work well or not) - bug fixes are not important!
- Time-to-market controls what's in the final O/S and app releases.
- Until 9/11/01, security has had a high cost and low perceived market value.
- No significant consequences for bad design or bad performance.

IT Spending 2000-2003  
(Billions of Dollars)



# Software's Vertical Structure

- **Business Applications**
- **Databases**
- **Network communication software, Internet**
- **Languages, Compilers & Tools**
- **Utilities & Libraries**
- **Operating system**
- **ISA (Hardware Layer)**

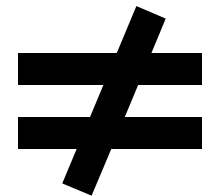
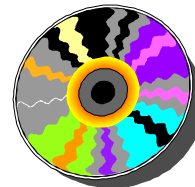
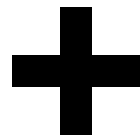
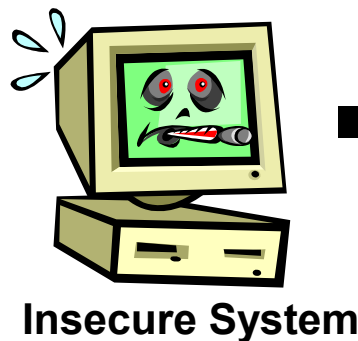


- **Libraries**
- **Books**
- **Chapters**
- **Paragraphs**
- **Sentences**
- **Words**
- **Alphabet**

- 1) Layers made by different & competing vendors
- 2) Minimal universal standards between layers
- 3) inter-layer interfaces changing often and without warning
- 4) Interface features come and go from release-to-release

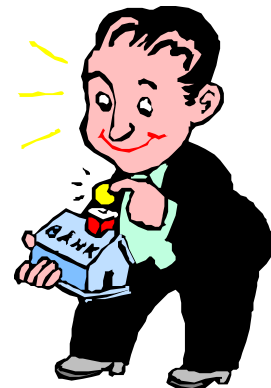
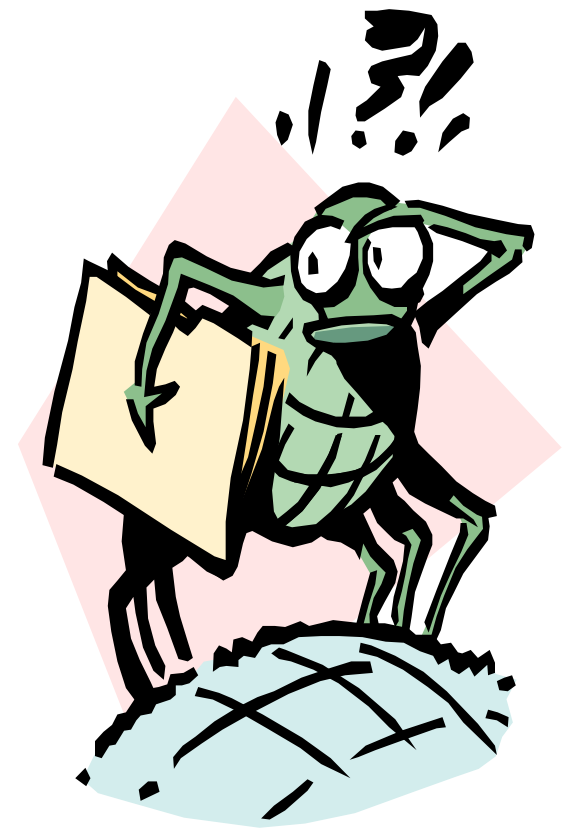
# Security Conclusions

- Security cannot be just “added-on”.
- A mythical desktop security “add-on” can’t be compatible with existing IT systems
- A “magic-CD” that will 100% protect your desktop without any other changes is not possible now.
- Going over each line of code - a famous Microsoft quote - won’t fix the security problem.
- Must redesign the O/S, and application and network software architecture.
- Security is more of an economic issue than a technical one.



# Common Security Questions

- How can I protect my PC?
- How can I prevent viruses from damaging my IT systems?
- How can I clean up the damage from viruses?
- How can I prevent malicious attacks from stealing data?
- How can I prevent malicious attacks from planting a time bomb?
- How can I take advantage of new technology?
- How can I do all this and still save money?

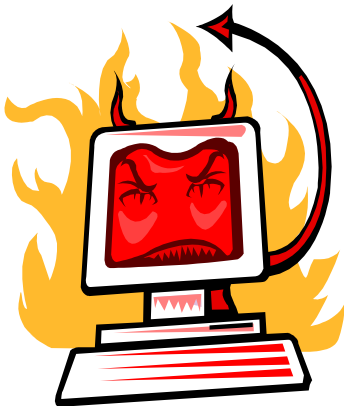


# Very Useful, Achievable Objectives

- Protect data assets
- Provide continuity of operations
- Make IT systems tamper-proof



# Unachievable Objectives (of limited value)

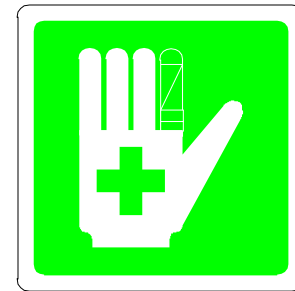


- Prevent cyber attacks
- 100% “Cyber-shield” each desktop
- Prevent all human errors
- Prevent “insider attacks”

# Firewalls & Anti-Virus Software

## What They Can Do

- Firewalls help protect desktops against network transmitted viruses.
- Anti-virus software scans and finds instances of known viruses.



## What They Cannot Do

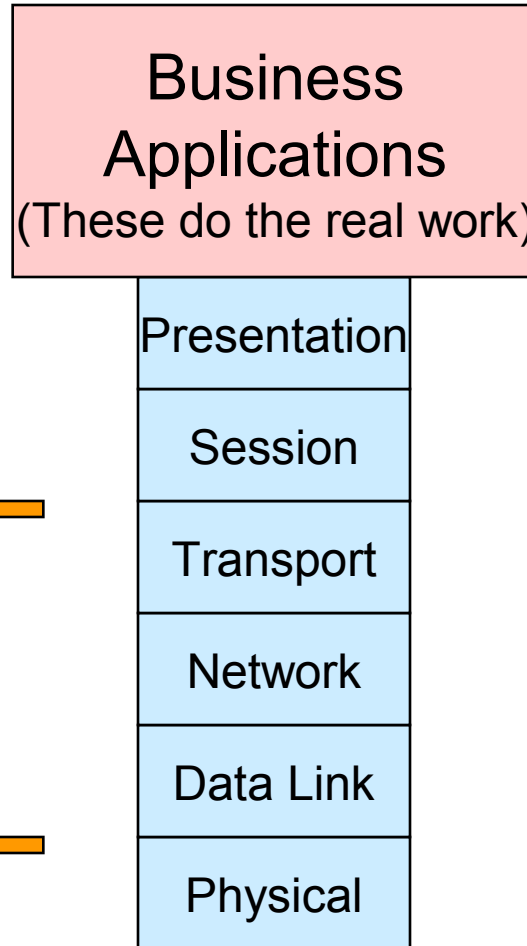
- Prevent damage from new viral strains.
- Clean up damage from cyber attacks.
- Prevent critical data loss from cyber-thieves.
- Prevent damage from operator error(s)
- Ensure proper operation of application software
- Make the IT system tamper-proof.
- Provide 100% security.



# The OSI 7-Layer Model

Of network-oriented software

## Protection Methods



□ The application does the useful work - it is the “payload” of your IT system.

No mechanisms for guaranteeing correct program operation.

□ The 6 underlying logical layers are there to support and enable the application.

---

Encryption, firewalls, anti-virus software, intrusion detection systems, etc.

□ All seven layers must be protected to have 100% security.

---

Cardkeys, Biometrics, etc.

□ Protecting just the lower layers via firewalls and anti-virus software is not enough - as experience has proven.

# Security Research Results

- Security has been important (to “techies”) for about 40 years.
- Research on security which started in the 1960s (and through the 1980s) led to three important conclusions:
  - Discovering security flaws then fixing them one-by-one will NOT work.
  - Effective security can only be achieved by **architecting** a system in accordance with a secure system model.
  - Effective security requires proper architecture, coding and deployment of the O/S, the application and all layers in between.



***Putting out fires  
doesn't work!***

# System Security Objectives



- ❑ Secure systems control access to information.
- ❑ Only properly authorized people are allowed to read, write, create or delete information.
- ❑ Only properly authorized processes are allowed to read, write, create or delete information.
- ❑ System must be properly architected, coded and deployed to be secure.

# What Secure Systems Must Do

## □ Policy

- **Security Policy** - System must enforce a well-defined security policy.
- **Marking** - System must associate all objects with access control labels (sensitivity & access modes).

## □ Accountability

- **Identification** - System must identify individuals and their various authorizations in a secure manner.
- **Audit Trail** - System must keep & protect audit trail so actions may be traced to responsible party.

## □ Assurance

- **Evaluation** - System must have hardware/software mechanisms that can be independently evaluated to assure that policy & accountability are enforced.
- **Continuous Protection** - System must continuously protect trusted mechanisms that enforce policy & accountability from tampering.





# Secure System Classification

## (DOD)

- **D - Minimal Protection**
- **C - Discretionary Protection**
  - **C1 - Discretionary security protection** - separates users & data, uses credible controls to enforce access limitations on an individual basis.
  - **C2 - Controlled Access Protection** - users individually accountable for their actions, security audit trail, resource isolation.
- **B - Mandatory Protection**
  - **B1 - Labeled Security Protection** - security policy model, keeps integrity of sensitivity labels, sensitivity labels must be held in all major system data structures, demonstration of reference monitor implementation.
  - **B2 - Structured Protection** - formal security policy model, discretionary & mandatory access control enforcement extended to all subjects & objects, separation of critical & non-critical system elements, stringent configuration management controls, covert channels are addressed, relatively resistant to penetration.
  - **B3 - Security Domains** - Reference monitor mediates all accesses of subjects to objects, be 100% tamperproof, TCB (trusted Computing Base) only contains security-relevant code & data structures, system engineered for minimal complexity, security-relevant events are signaled, system recovery is required, highly resistant to penetration.
- **A - Verified Protection**
  - **A1 - Verified Design** - Functionally same as B3, full mathematical verification of design.



# Secure System Classification

(NIST - National Institute of Standards & Technology)

(NIAP - National Information Assurance Partnership)

ISO 15408

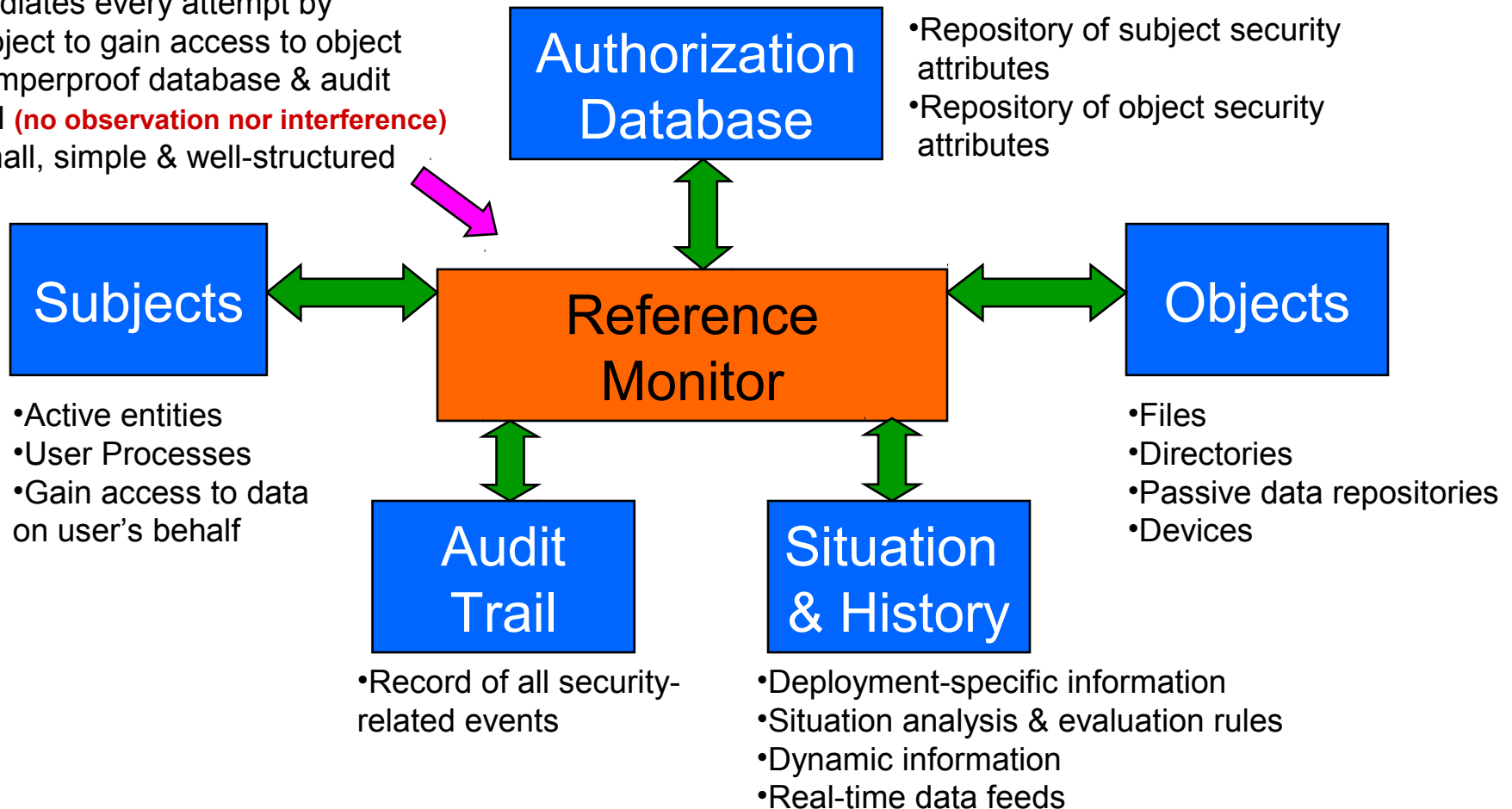
- **EAL-1 - Functionally Tested** - independent testing of selected features.
- **EAL-2 - Structurally Tested** - independent testing of selected features using limited developer design data.
- **EAL-3 - Methodically Tested & Checked** - independent testing using limited developer design data, selective developer result confirmation, evidence of develop search for obvious vulnerabilities.
- **EAL-4 - Methodically Designed, Tested & Reviewed** - independent testing using low-level vendor design data, search for vulnerabilities, development controls, automated configuration management.
- **EAL-5 - Semiformally Designed & Tested** - independent testing of all of the implementation (TOE), formal model, semiformal conformance to design specs, vulnerability assessment for attackers with moderate potential.
- **EAL-6 - Semiformally Verified Design & Tested** - independent testing of 100% of TOE, modular & layered approach to design, structured presentation, vulnerability assessment for attackers with high potential, systematic search for covert channels.
- **EAL-7 - Formally Verified Design & Tested** - same as above, but all models, specs & presentations are formal, TOE is tightly focused on security functionality, amenable to formal analysis, design complexity must be minimized.

# The Reference Monitor

## (A Secure System Architecture)

- Enforces security policy
- Mediates every attempt by subject to gain access to object
- Tamperproof database & audit trail (**no observation nor interference**)
- Small, simple & well-structured

- Repository of subject security attributes
- Repository of object security attributes

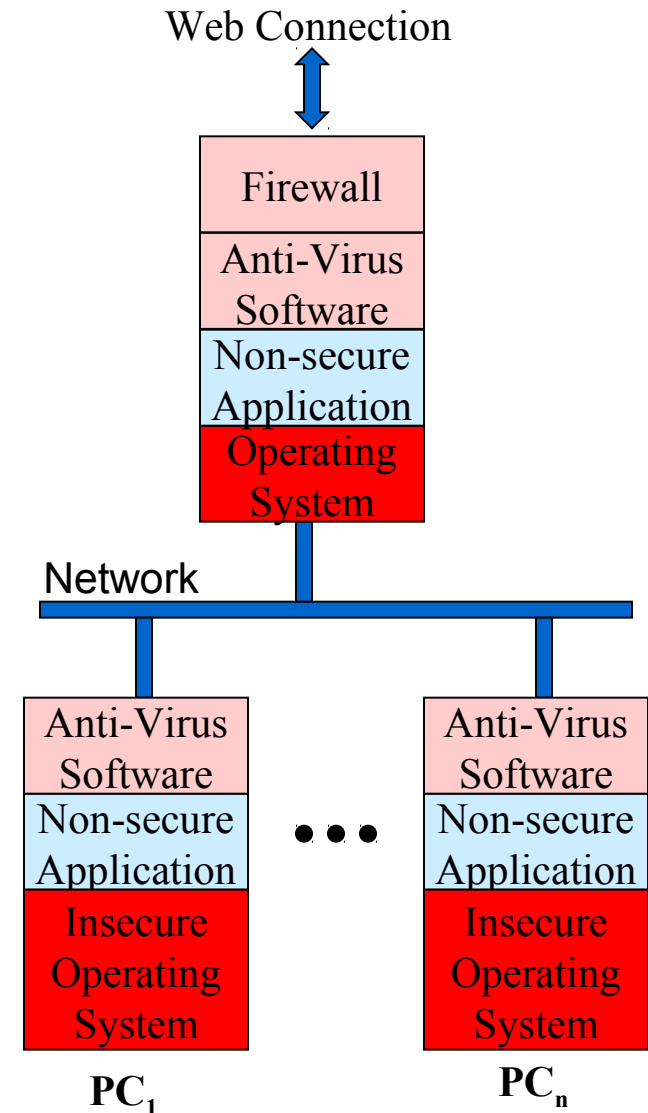


# So, How Do We Fix This?

(Things to Avoid)

- Avoid the traditional IT architecture - the risk exposure is too great.
- Protecting the desktop does not equate to protecting your business operations 24 x 7.
- Avoid deploying critical software on non-secure platforms. This minimizes risk exposure.
- Avoid deploying critical functionality on non-secure platforms. Critical functionality can be separate from the software that supports and/or delivers it.

## Traditional, Non-secure IT Architecture



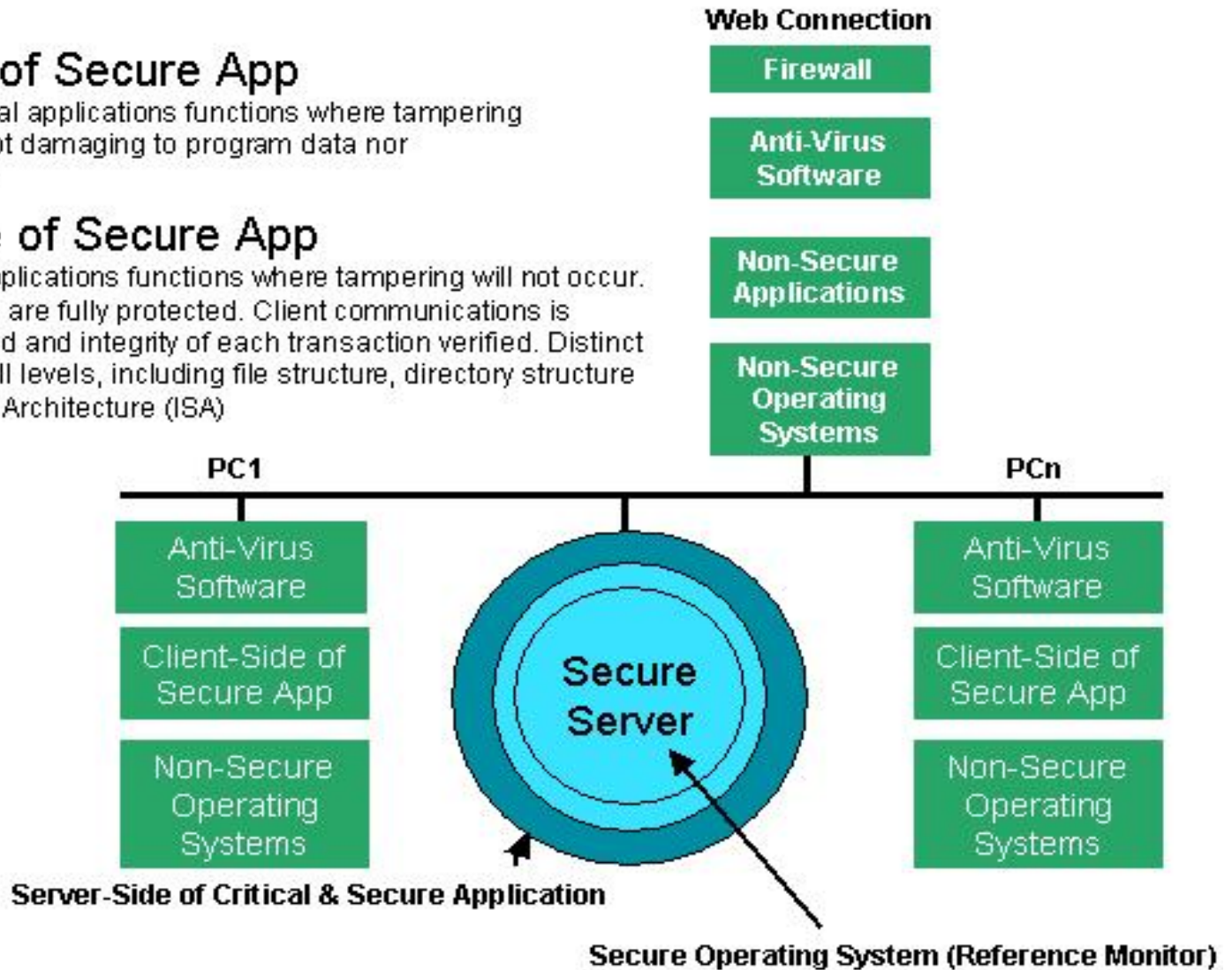
# Secure Architecture

## Client Side of Secure App

Performs non-critical applications functions where tampering may occur but is not damaging to program data nor program operation

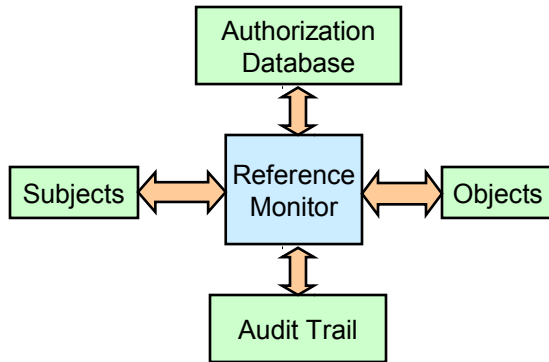
## Server Side of Secure App

Performs critical applications functions where tampering will not occur. Data and programs are fully protected. Client communications is encrypted, regulated and integrity of each transaction verified. Distinct from client O/S at all levels, including file structure, directory structure and Instruction Set Architecture (ISA)





# Final Thoughts



Application
Presentation
Session
Transport
Network
Data Link
Physical

- Software has a vertical structure so ensure that your apps are on a solid base.
- Use the right tool for the job:
  - Use secure system for critical information.
  - Use secure system for critical applications.
- Architect critical apps consistent with Reference Monitor structure.
- Use secure coding techniques
- Deploy apps carefully, using all available security controls.