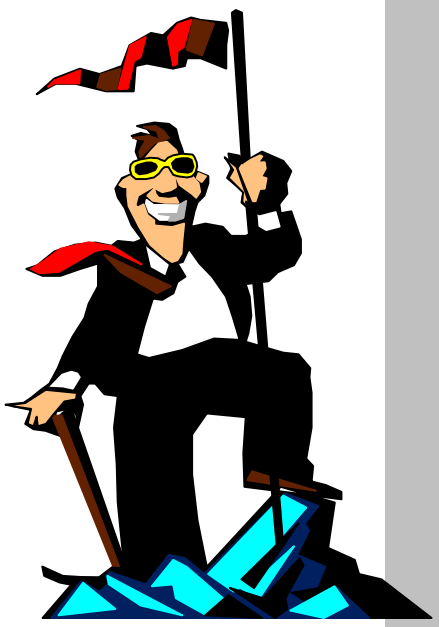
Attacks on Information Systems

http://all.net/

Fred Cohen - Managing Director
Fred Cohen & Associates
fc@all.net - 925-454-0171



A1: errors and omissions

- Erroneous entries or missed entries by designers, implementers, maintainers, administrators, and/or users create vulnerabilities exploited by attackers.
- Examples include:
  - forgetting to eliminate default accounts and passwords when installing a system,
  - incorrectly setting protections on network services, and
  - a wide range of other minor mistakes that can lead to disaster.
- There appear to be an unlimited (finite but unbounded) number of possible errors and omissions in general purpose systems. Special-purpose systems may be more constrained.

A2: power failure

- Failure of electrical power causes computer and peripheral failures leading to loss of availability, sometimes requiring emergency response, and otherwise disrupting normal operations. [Winkelman95] [Agudo96] [NSTAC96] [Dagle96]
- Examples include:
  - The blackout of a small CA city in the late 1980s due to an error in a telephone line assignment,
  - The U.S. power grid cascade failures of the summer of 1996, and
  - Rolling blackouts in the Eastern U.S. in 1984 caused by a harsh winter.
- Power failure is not usually a complex issue to address, although the underlying causes may be.

A3: cable cuts

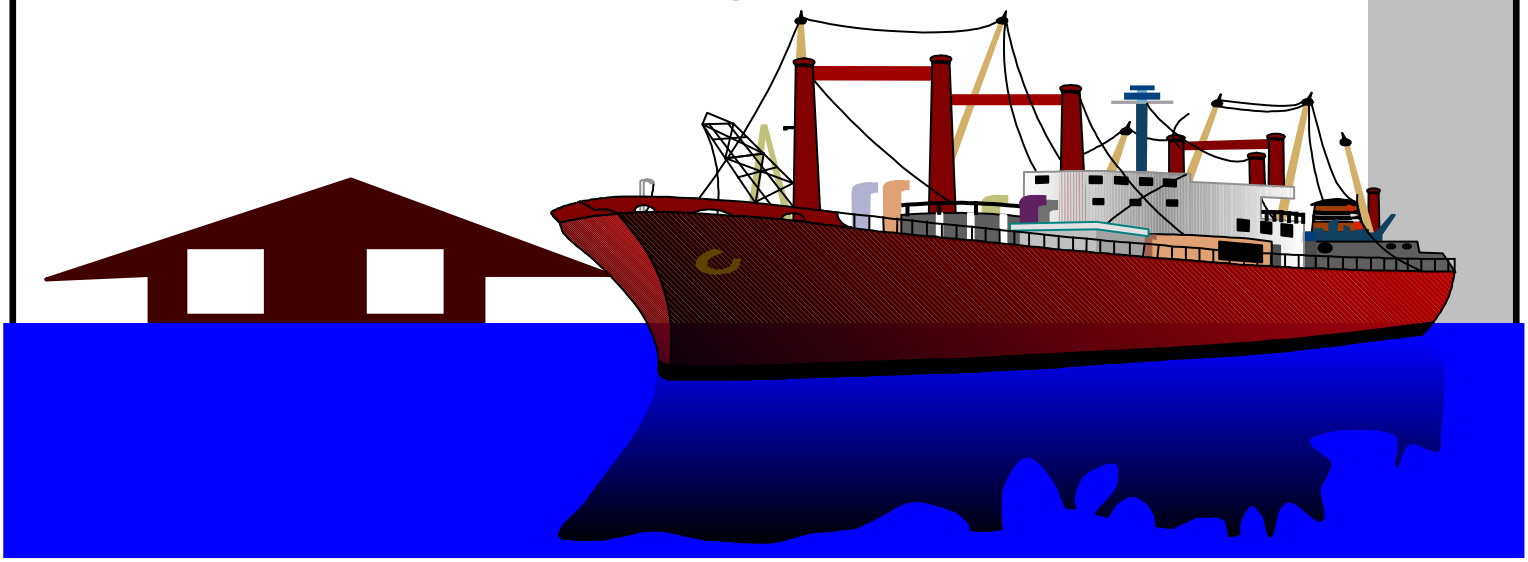
- A cable is cut resulting in disrupted communications, usually requiring emergency response, and otherwise disrupting normal operations.
- Examples include:
  - A cable cut in the late 1980s that disrupted all 18 redundant Internet connections to the Northeastern U.S.
  - A Cable cut in New York City in the 1990s that shut down East Coast air traffic control for more than 8 hours, and
  - Intentionally severed communication lines in the Gulf war.
- The general issue of cable cutting is quite complex and appears to involve solving many large min-cut problems.

A4: fire

- A fire occurs causing physical damage and permanent as well as temporary faults, requiring emergency response, and otherwise disrupting normal operations.
- Examples include:
  - The Chicago telephone substation fire of the late 1980s,
  - Brush fires commonly encountered in the Los Angeles and San Francisco areas, and
  - Fires resulting from military attacks
- The fire issue is not normally a very complex one and statistical models work quite well in most analyses.

A5: flood

- A flood occurs causing physical damage and permanent as well as temporary faults, requiring emergency response, and otherwise disrupting normal operations.
- Examples include:
  - The Chicago flood that wiped out services throughout the downtown area
  - Recent floods in the midwestern United States
  - Floods commonly associated with Sunamis, Hurricanes, etc.
- Although floods are generally considered relatively simple issues to address, their are occasionally somewhat more complex flooding issues than are anticipated.

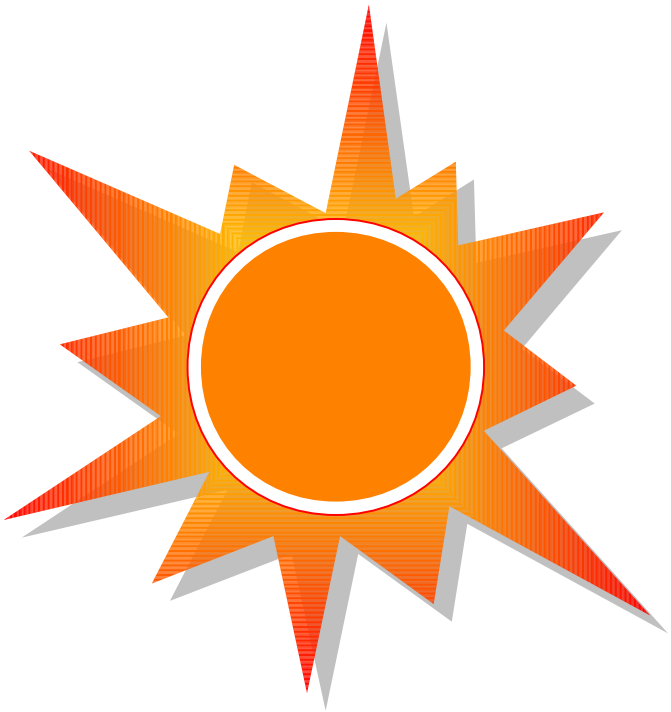


A6: earth movement

- The Earth moves causing physical damage and permanent as well as temporary faults, requiring emergency response, and otherwise disrupting normal operations.
- Statistical techniques and historical data appear to be quite sufficient to analyze Earth movement.

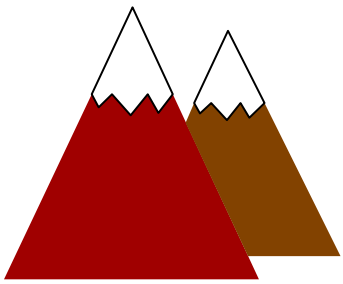
A7: solar flares

- Changes on the surface of the sun cause excessive amounts of radiation to be delivered, typically resulting in noise bursts on radio communications, disrupted communications, and other changed physical conditions.
- Statistical techniques and historical data appear to be quite sufficient to analyze solar flares.



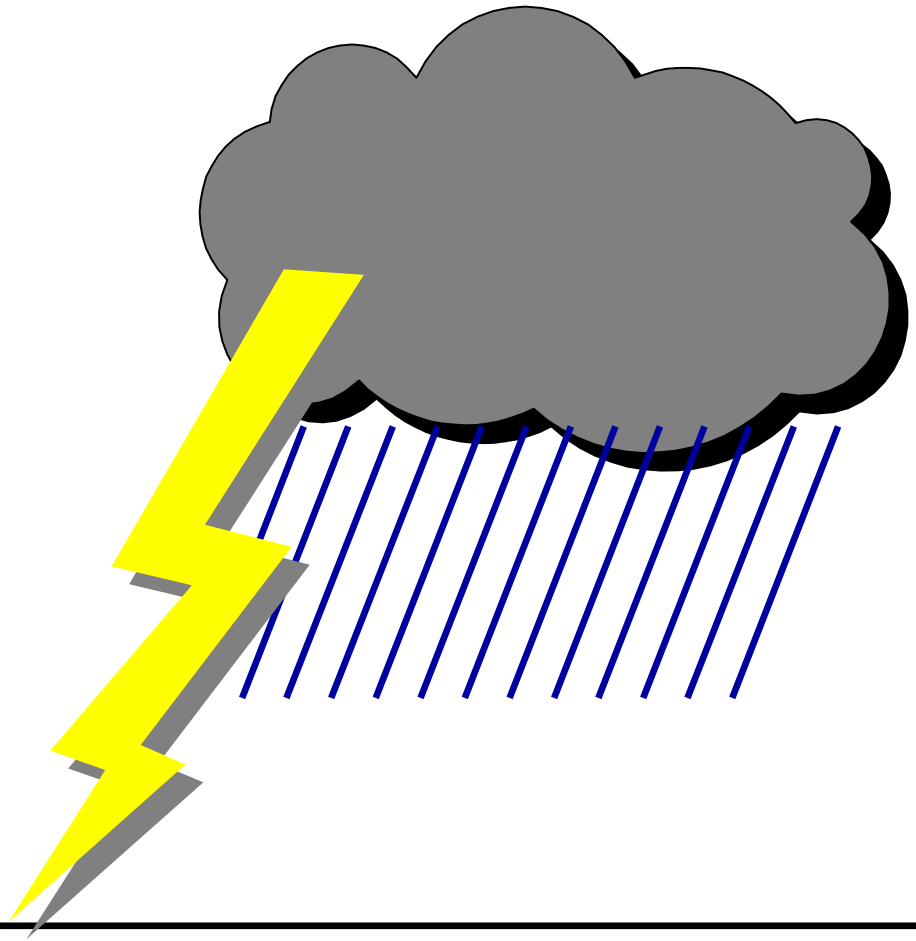
A8: volcanos

- A volcano erupts causing physical damage and permanent as well as temporary faults, requiring emergency response, and otherwise disrupting normal operations.
- Statistical techniques and historical data appear to be quite sufficient to analyze volcanoes.



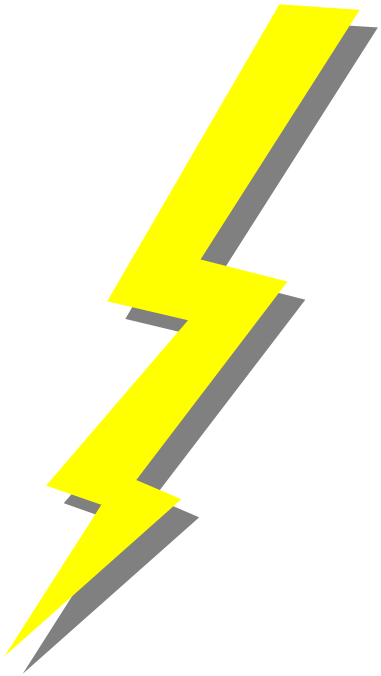
A9: severe weather

- Severe weather conditions (e.g., hurricane, tornado, winter storm) occur causing physical damage and permanent as well as temporary faults, requiring emergency response, and otherwise disrupting normal operations.
- Statistical techniques and historical data appear to be quite sufficient to analyze severe weather.



A10: static

- Static electricity builds up on surfaces and causes transient or permanent failures in components.
- The static issue is not normally a very complex one.

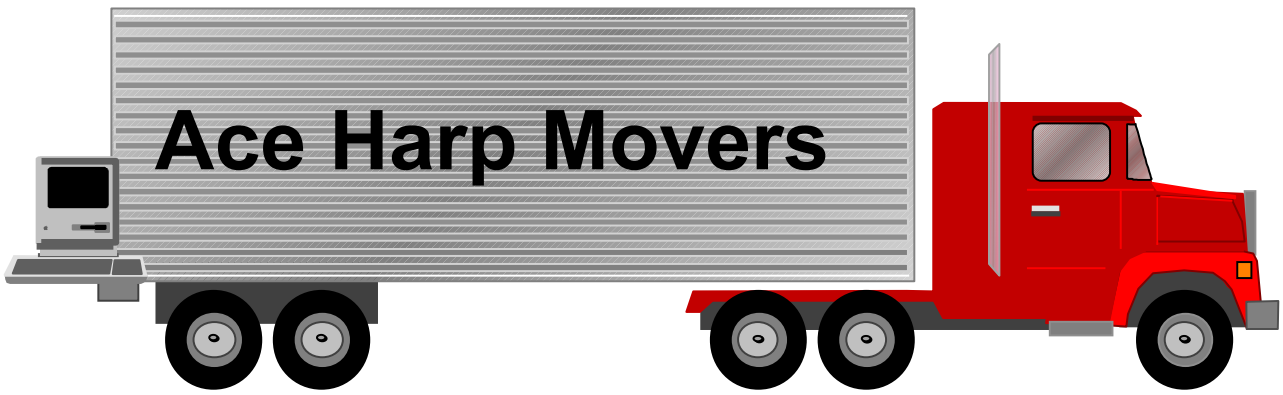


A11: environmental control loss

- Environmental controls required to maintain proper operating conditions for equipment fails causing disruption of services.
- Example causes include:
  - air conditioning failures,
  - heating failures
  - temperature cycling,
  - smoke,
  - dust,
  - vibration,
  - corrosion,
  - gases,
  - fumes,
  - chemicals.
- Statistical techniques and historical data appear to be quite sufficient to analyze environmental control losses in most cases.

A12: relocation

- Relocation of equipment causes physical harm to equipment and different exposures of equipment to physical and environmental vulnerabilities.
- Statistical techniques and historical data appear to be quite sufficient to analyze relocation.



A13: system maintenance

- System maintenance causes period of time when systems operate differently than normal and may result in temporary or permanent inappropriate or unsafe configurations. Maintenance can also be exploited by attackers to create forgeries of sites being maintained, to exploit temporary openings in systems created by the maintenance process, or other similar purposes. Maintenance can accidentally result in the introduction of viruses, by leaving improper settings, and by other similar accidental events.
- Statistical techniques and historical data appear to be quite sufficient to analyze system maintenance.

A14: testing

- Testing stresses systems inducing a period of time when systems operate differently than normal and may result in temporary or permanent inappropriate or unsafe configurations.
- Testing issues are quite complex, and some well-known testing problems are exponential in time and space. Much of the current analysis of protection testing is based on naive assumptions.

A15: inadequate maintenance

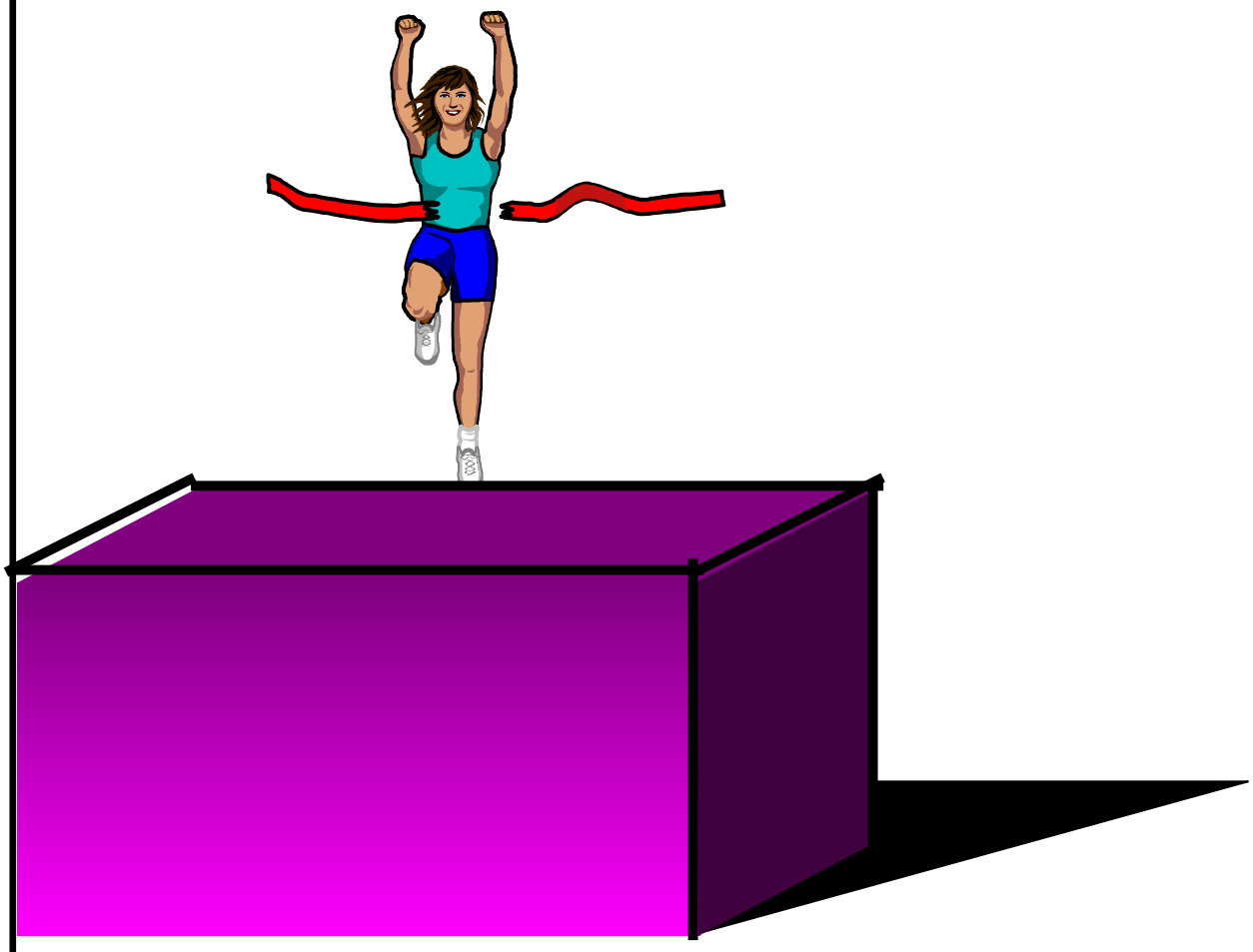
- Inadequate maintenance results in uncovered failures over extended periods of time, possibly inducing a period of time when systems operate differently than normal and may result in temporary or permanent inappropriate or unsafe configurations.
- Statistical techniques and historical data appear to be quite sufficient to analyze maintenance adequacy.

A16: Trojan horses

- Unintended components or operations are placed in hardware, firmware, software, or wetware causing unintended and/or inappropriate behavior.
- Examples include:
  - time bombs, use or condition bombs, additional instructions in memory, operating system modifications, name overloaded programs placed in an execution path, added or modified circuitry, false connectors, false panels, radios placed in components, etc.
- Detecting Trojan horses is almost certainly an undecidable problem (although nobody has apparently proven this it seems clear).

A17: dumpster diving

- Waste product is examined to find information that might be helpful to the attacker.
- Statistical techniques and historical data appear to be quite sufficient to analyze dumpster diving.



A18: fictitious people

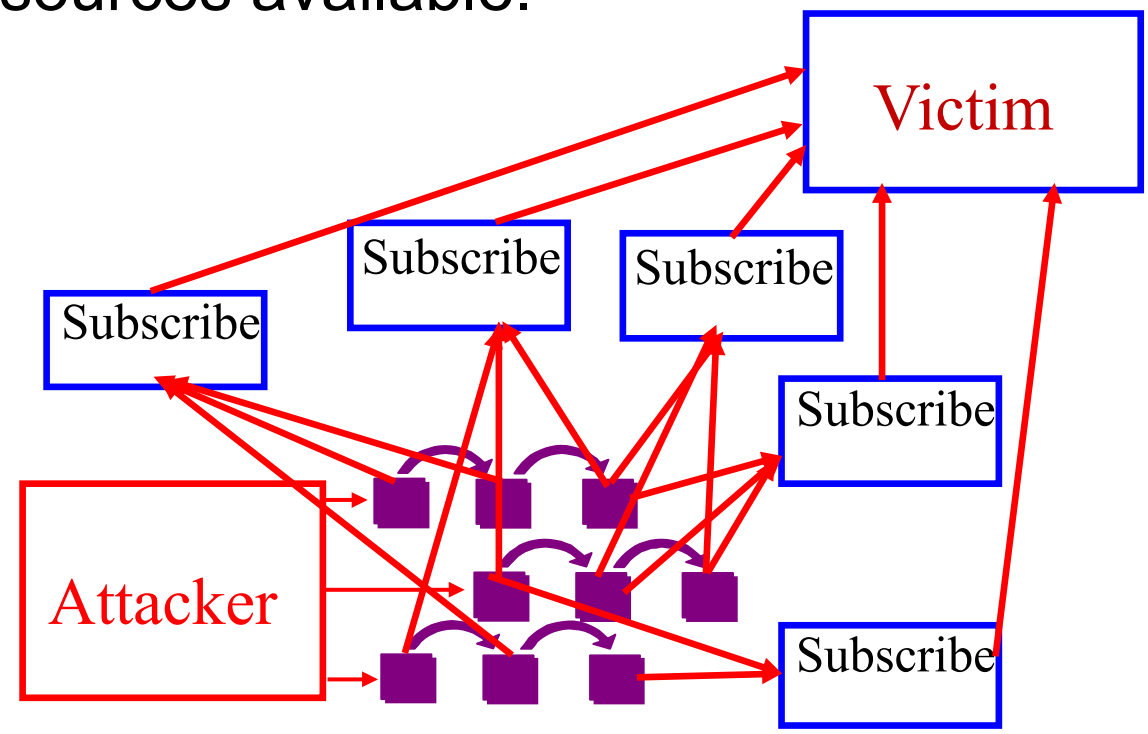
- Impersonations or false identities are used to bypass controls, manage perception, or create conditions amenable to attack.
- Examples include:
  - spies,
  - impersonators,
  - network personae,
  - fictional callers,
  - and many other false and misleading identity-based methods.
- This appears to be a very complex social, political, and analytical issue that is nowhere near being solved.

A19: protection missetting exploitation

- Mis-set protections on files, directories, systems, or components are exploited to examine, modify, delete, or otherwise disrupt normal operation.
- Setting protections properly is not a trivial matter, but there are linear time algorithms for automating settings once there is a decision procedure in place to determine what values to set protection to. Under some conditions, it is impossible to have settings that both provide all appropriate access and deny all inappropriate access. [Cohen91] It is undecidable for a general purpose subject/object system to determine whether a given subject will eventually gain any particular right over any particular object. [Harrison76]

A20: resource availability manipulation

- Resources are manipulated so as to make functions requiring those resources fail.
- Examples include:
  - e-mail overflow used to disrupt system operation, [Cohen93]
  - file handle consumption used to prevent audits from operating, [Cohen91] and
  - overloading unobservable network paths to force communications rerouting.
- Most of the issues with resource availability result from the high cost of making worst- case resources available.



A21: perception management a.k.a. human engineering

- Causing people to believe things that forward the goal.
- Examples include:
  - tricking a person into giving you their password or changing their password to a particular value for a period of time,
  - talking your way into a facility, and
  - causing people to believe in religious doctrine in order to get them to behave as desired.
- This has been a security issue since the beginning of time and appears to be a very complex human, social, political, and legal issue.

A22: spoofing and masquerading

- Creating false or misleading information in order to fool a person or system into granting access.
- Examples include:
  - operator spoofing to trick the operator into giving away a password,
  - location spoofing to trick a person or system into believing a false location,
  - spoofing a login screen to get users to provide identification and authentication information,
  - email spoofing which forges email to generate desired results, and
  - time spoofing which creates false impressions of relative or absolute time.

A23: infrastructure interference

- Interfering with infrastructure so as to disrupt services and/or redirect activities.
- Examples include:
  - creating an accident on a particular road at a particular place and time to cause a shipment to be rerouted,
  - disrupting electrical power in order to deny information services,
  - modifying a domain name server to alter the path of information flow from point to point, and
  - cutting a phone line.
- It appears that analyzing infrastructure interference is quite complex.

A24: infrastructure observation

- Examining the infrastructure in order to gain information.
- Examples include:
  - watching air ticketing information to see when particular people go to particular places as an intelligence indicator,
  - tapping a PBX system in order to record key telephone conversations, and
  - watching for passwords on the Internet.
- Except in cases where cryptography, spread spectrum, or other similar technology is used to defend against an attack, it appears that infrastructure observation is simple to accomplish and expensive to detect.

A25: insertion in transit

- Insertion of information in transit so as to forge desired communications.
- Examples include:
  - adding transactions to a transaction sequence,
  - insertion of routing information packets so as to reroute information flow, and
  - insertion of shipping address information to replace an otherwise defaulted value.
- Although there appears to be a widespread belief that insertion in transit is very difficult, in most cases it is technically straight forward. Complexity only arises when defensive measures are put in place to detect or prevent this sort of attack.

A26: observation in transit

- Examination of information in transit.
- Examples include:
  - telephone tapping,
  - network tapping, and
  - I/O buffer watching.
- Except in cases where cryptography, spread spectrum, or other similar technology is used to defend against such an attack, it appears that observation in transit over physically insecure communications media is simple to accomplish and expensive to detect.

A27: modification in transit

- Modification of information in transit so as to alter communications as desired.
- Examples include:
  - removing end-of-session requests and providing suitable replies, then taking over the session,
  - modification of an amount in an electronic funds transfer request, and
  - rewriting Web pages so as to reroute subsequent traffic through the attacker.
- Modification in transit is roughly equivalent in complexity to the combination of observation in transit and insertion in transit, however, because of the real-time requirements, modification in transit is sometimes difficult in practice.

A28: sympathetic vibration

- Creating or exploiting positive feedback loops or underdamped oscillatory behaviors so as to overload a system.
- Examples include:
  - electrical or acoustic wave enhancement,
  - the creation of packets in the Internet which form infinite loops, and
  - protocol errors causing cascade failures in telephone systems.
- In some underdamped systems, sympathetic vibration is easily induced. It sometimes even happens accidentally. In over-damped systems, sympathetic vibration requires additional energy.

A29: cascade failures

- Design flaws in tightly coupled systems cause error recovery procedures to induce further errors under select conditions.
- Examples include:
  - electrical cascade failures in the U.S. power grid, [WSCC96]
  - telephone system cascade failures causing widespread long distance service outages, [Pekarske90] and
  - inter-system cascades such as power failures bringing down telephone switches required to bring back up power stations.
- Only cursory examination of select cascade failures has been completed. In systems operating at or near capacity, cascade failures are easily induced and must be actively prevented or they occur accidentally. [WSCC96]

A30: bribes and extortion

- Promises or threats that cause trusted parties to violate trust.
- Examples include:
  - bribing a guard to gain entry into a building,
  - kidnapping a key employee's family to gain access to a computer, and
  - using sexually explicit photographs to get insider information.
- This issue is as complex as the general problem of insider attacks.

A31: get a job

- An attacker gets a job in order to gain insider access to a facility.
- Examples include:
  - getting a maintenance job by under-bidding opponents,
  - planting spies in intelligence agencies of competitors.
- This issue is as complex as the general problem of insider attacks.

A32: password guessing

- Sequences of passwords are tried against a system or password repository in order to find a valid authentication sequence.
- Examples include:
  - running the program "crack" on a stolen password file,
  - guessing passwords on network routers and PBX switches, and
  - using well-known maintenance passwords to try to gain entry.
- Password guessing has been analyzed in painstaking detail by many researchers. In general, the problem is as hard as guessing a string from a language chosen by an imperfect random number generator.

A33: invalid values on calls

- Invalid values are used to cause unanticipated behavior.
- Examples include:
  - system calls with pointer values leading to unauthorized memory areas and
  - requests for data from databases using system escape characters to cause interprocess communication to operate improperly, and
  - CGI script exploits wherein input sequences grant unauthorized access (`/bin/sh ...`)
- In most cases, only a few hundred well-considered attempts are required to find a successful attack of this sort against a program.

A34: undocumented or unknown function exploits

- Functions not in the documentation or unknown to the owners are exploited.
- Examples include:
  - back doors placed in systems to facilitate maintenance,
  - undocumented system calls commonly inserted by vendors to enable special functions resulting in economic or other market advantages, and
  - program sequences accessible in unusual ways as a result of improperly terminated conditionals.
- Finding back-doors is, in general, as hard as demonstrating program correctness (at least NP-complete)

A35: inadequate notice exploitation

- Lack of adequate notice is used as an excuse to do things that notice would normally have warned against.
- Examples include:
  - unprosecutable entry via normally unused services,
  - observation of employee keystrokes without notice resulting in law suits,
  - password guessing through an interface not providing notice, and
  - Web server attacks which bypass any notice provided on the home page.
- Notice is trivially demonstrated to be given or not given depending on the method entry.

A36: excess privilege exploits

- A program, device, or person is granted privileges not strictly required in order to perform their function and the excess privilege is exploited to gain further privilege or otherwise attack the system.
- Examples include
  - Unix-based SetUID programs granted root access exploited to grant attackers unlimited access,
  - access to unauthorized need-to-know information by a systems administrator granted too-flexible maintenance access to a network control switch, and
  - user-programmable DMA devices reprogrammed to access normally unauthorized portions of memory.
- Determining whether a privileged program grants excessive capabilities to an attacker appears, in general, to be as hard as proving program correctness.

A37: environment corruption

- The computing environment upon which programs or people depend for proper operation is corrupted.
  - Logical examples include manipulating an environment variable so as to cause command interpretation to operate unusually or manipulation of a paper form so as to change its function without alerting the person filling it out.
  - In the physical domain, this includes the introduction of gasses, dust, or other particles, chemicals, or elements into the physical environment.
  - In the electromagnetic realm, it includes waveforms.
  - In humans, sound, smell, feel, and other sensory input corruption is included.

A38: device access exploitation

- Access to a device is exploited to alter its function or cause its function to be used in unanticipated ways.
- Examples include:
  - removing shielding from a wire so as to leak more electromagnetic emanations,
  - reprogramming a bus device to deny services at a hardware level, and
  - altering microcode so as to associate attacker-defined functions with otherwise unused operation codes.
- Since hardware devices are, in general, at least as complex as software devices, the complexity of detecting such a flaw would appear to be at least NP-complete. Injecting such a flaw, on the other hand, appears to be quite simple - given physical access to a device.

A39: modeling mismatches

- Mismatches between models and the realities they are intended to represent are exploitable by attackers.
- Examples include:
  - use of the Bell- LaPadula model of security [Bell73] as a basis for designing secure operating systems leaves disruption uncovered,
  - modeling attacks and defenses as if they were statistically independent phenomena for risk analysis ignores synergistic effects, and
  - modeling misconfigurations as mis-set protection bits - when the content of configuration files remains uncovered.
- There is some theory about the adequacy of modeling, however, there is no general theory that addresses protection-related issues of modeling flaws. This appears to be a very complex issue.

A40: simultaneous access exploitations

- Two or more simultaneous or split multi-part access attempts are made, resulting in an improper decision or loss of audit information.
- Examples include:
  - large numbers of access attempts over a short period of time so as to cause decision software to act in a previously unanticipated and untested fashion,
  - the execution of sequences of operations required for system takeover by multiple user identities, and
  - the holding of a resource required for some other function to proceed so as to deny completion of that service.
- This problem has been analyzed in a cursory fashion and the number of possible sequences of events appears to be factorial in the combined lengths of the programs coexisting in the environment. [Cohen94-3]

A41: implied trust exploitation

- Programs operating in a shared environment inappropriately trust the information supplied to them by untrustworthy programs.
- Examples include:
  - forged data from Domain Name Servers in the Internet used to reroute information through attackers,
  - forged replies from authentication daemons causing untrusted software to be run by access control software,
  - forged Network Information Service packets causing wrong password entries to be used in authenticating attackers,
  - and network-based administration programs fooled into forwarding incorrect administrative controls.
- In general, analyzing this problem would seem to require analyzing all of the interdependencies of programs.

A42: interrupt sequence mishandling

- Unanticipated or incorrectly handled interrupt sequences cause system operation to be altered.
- Examples include:
  - stack frame errors induced by incorrect interrupt handling,
  - incorrect swapping out of the swapping daemon on unanticipated conditions, and
  - denial of services resulting from improper prioritization of interrupts.
- This problem has been analyzed in a cursory fashion and the number of possible sequences of events appears to be factorial in the combined lengths of the programs coexisting in the environment. [Voas93]

A43: emergency procedure exploitation

- An emergency condition is induced or awaited resulting in behavioral changes that reduce or alter protection.
- Examples include:
  - fires, during which access restrictions are less rigorously enforced,
  - power failures during which automated alarm and control systems fail in a safe mode with respect to some - possibly exploitable - criteria, and
  - incident response during which systems administrators deviate from normal behavioral patterns.
- In most cases, emergency procedures bypass normal controls, making many attacks feasible during an emergency that would be far more difficult during normal operations.

A44: desychronization and time-based attacks

- Systems that depend on synchronization are desynchronized causing them to fail or operate improperly.
- Examples include:
  - Y2K, leap year, and similar issues
  - DCE servers that deny services when desynchronized beyond some threshold,
  - cryptographic systems which depend on synchronized encryption/decryption,
  - software that makes complex decisions based on slight time differences, and
  - time-based locks which may be caused to open or close at the wrong times.
- This problem appears to be similar in complexity to the interrupt sequence mishandling problem. [Voas93] It appears, in general, to be factorial in the number of time-based decisions made in a system.

A45: imperfect daemon exploits

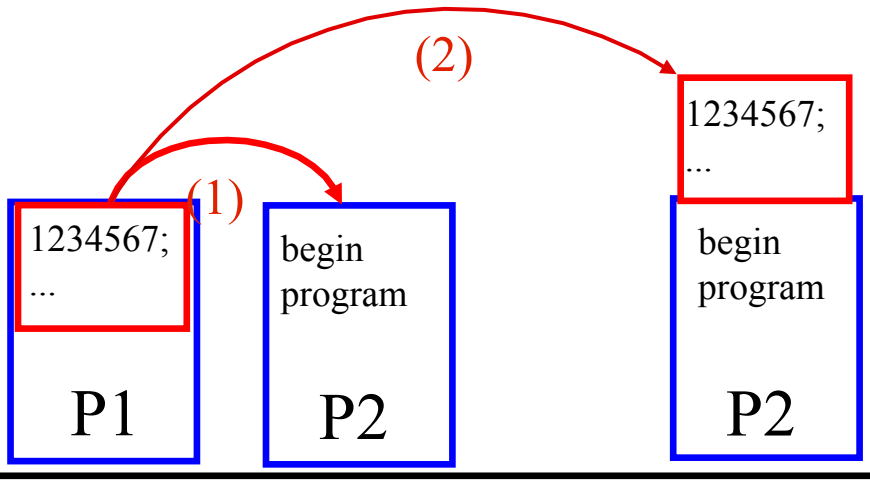
- Daemon programs designed to provide privileged services upon request have imperfections that are exploited to provide privileges to the attacker.
- Examples include:
  - Web, Gopher, Sendmail, FTP, TFTP, and other server daemons exploited to gain access to servers,
  - internal use only daemons such as the Unix cron facility exploited to gain root privileges, and
  - automated backup and recovery daemons exploited to overwrite current versions of programs with previous ones.
- In general, this problem is at least as complex as proving program correctness.

A46: multiple error inducement

- The introduction of multiple errors is used to cause otherwise reliable software to fail in unanticipated ways.
- Examples include:
  - creation of an input syntax error with a previously locked error-log file resulting in inconsistent data state,
  - premature termination of a protocol during an error recovery process - possible causing a cascade failure, and
  - introduction of simultaneous interleaved attack sequences causing normal detection methods to fail.
- Limited work on multiple error effects indicates that even the most well-designed and trusted systems fail unpredictably under multiple error conditions

A47: viruses

- Programs that reproduce and possibly evolve.
- Examples include:
  - the 11,000 or so known viruses,
  - custom file viruses designed to act against specific targets, and
  - process viruses that cause denial of service or thrashing within a system.
- Virus detection has been proven to be undecidable in the general case. [Cohen86] [Cohen84] Viruses are also trivial to write and highly effective against most modern systems.

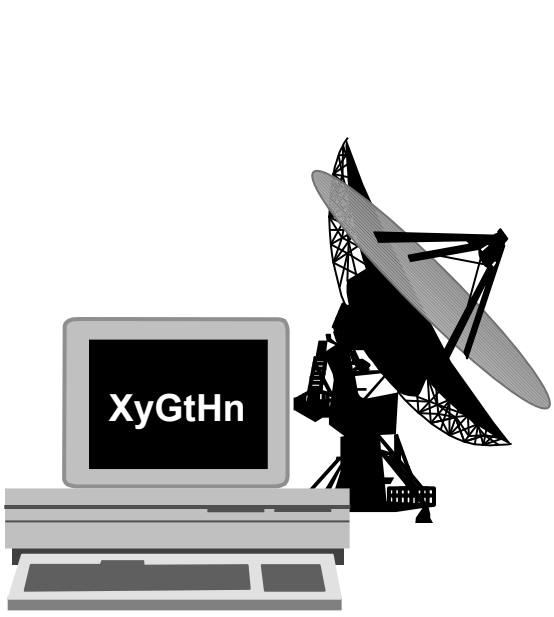


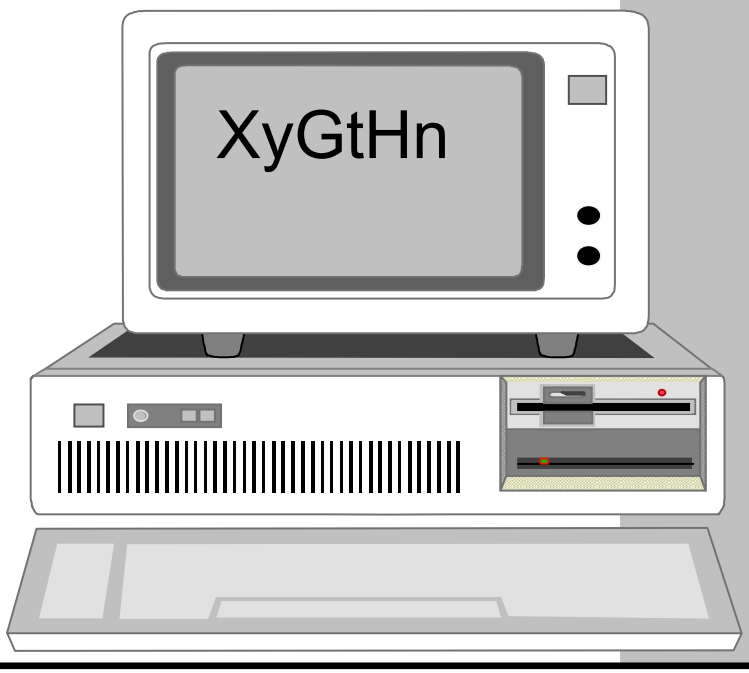
A48: data diddling

- Modification of data through unauthorized means.
- Examples include:
  - non-database-program manipulation of database files,
  - modification of configuration files used to setup further machines, and
  - modification of data residing in temporary files such as intermediate files created during compilation.
- Data diddling is a relatively simple task. If the data is writable, it can be easily diddled.

A49: van Eck bugging

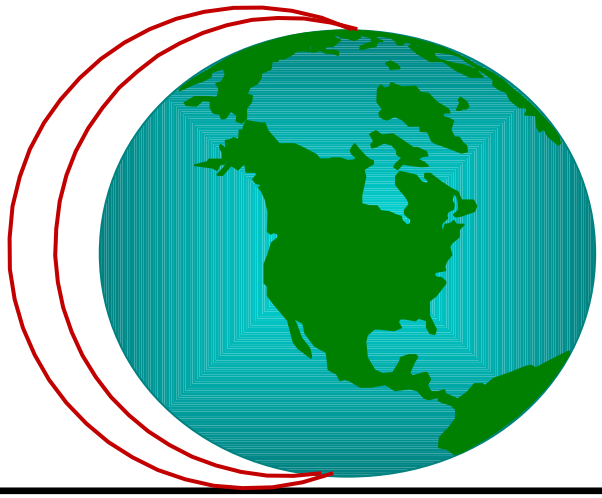
- Electromagnetic emanations are observed from afar.
- Examples include:
  - the tapping of Scotland Yard by a reporter to demonstrate a $100 remote tapping device and observed emanations from financial institutions indicative of pending trades.
- van Eck bugging is relatively easy to do and requires only cursory knowledge of electronics and antennae theory.[vanEck85]





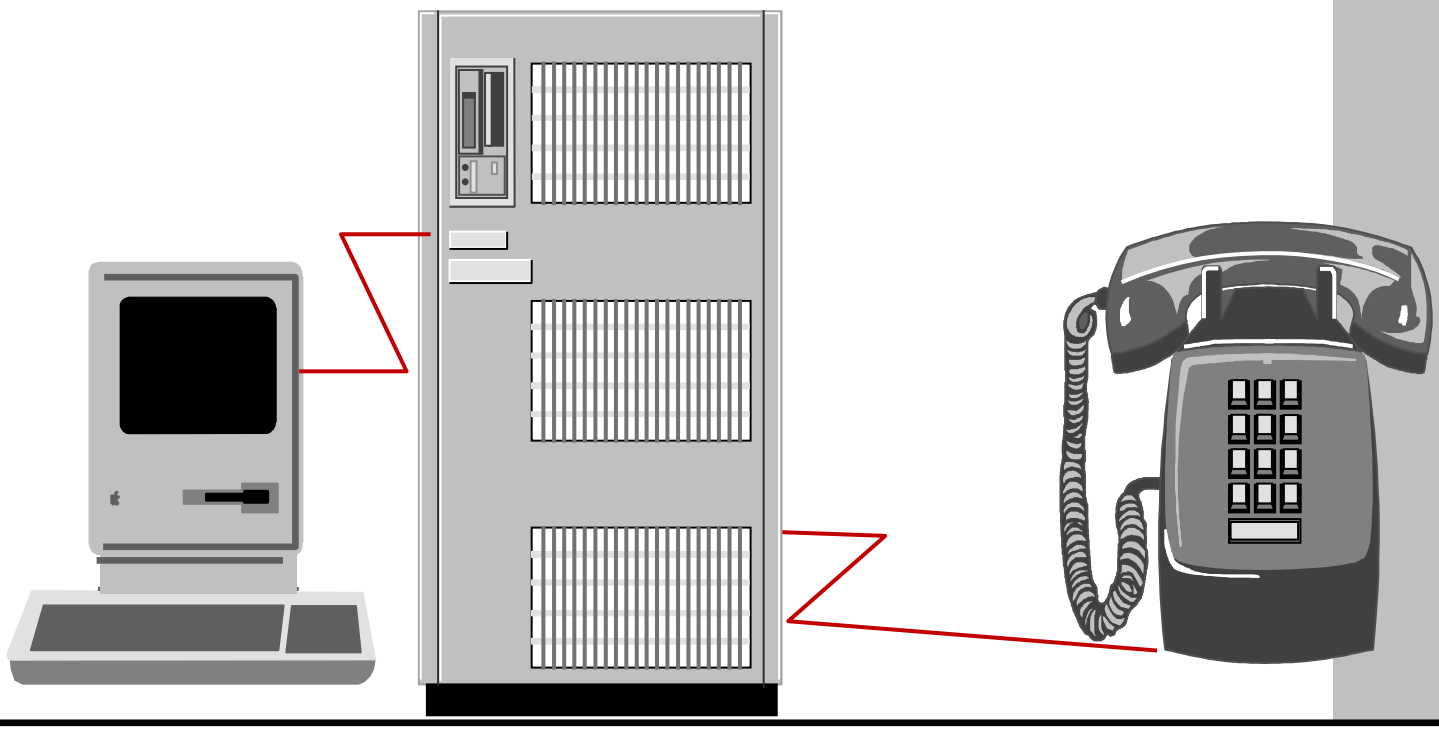
A50: electronic interference

- Jamming signals are introduced to cause failures in electronic communications systems.
- Examples include:
  - the method and apparatus for altering a region in Earth atmosphere, ionosphere, and/or magnetosphere, and
  - common radio jamming techniques.
- Simplistic jamming is straight forward, however, power efficient jamming is necessary in order to have good effect against spread spectrum and similar anti-jamming systems, and this is somewhat more complex to achieve.



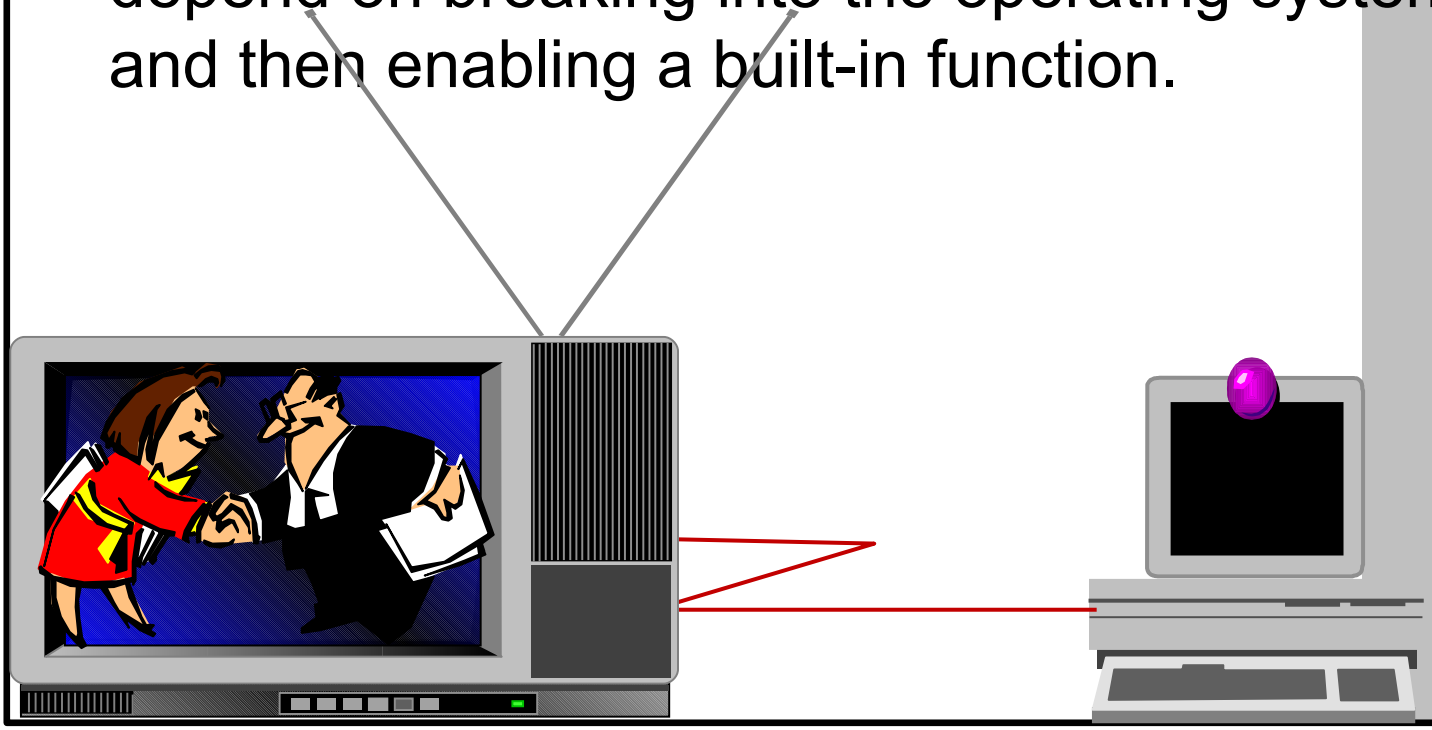
A51: PBX bugging

- Point Branch eXchanges or similar switching centers are attacked to allow connected telephone instruments to be tapped.
- Examples include:
  - on-hook bugging of hand sets,
  - open microphone listening, and
  - silent conference call exploits.
- In cases where functions that support bugging are provided by the PBX, this attack is straight forward.



A52: audio/video viewing

- Audio and video input devices connected to computers for multi-media applications are exploited to allow attackers to look at and listen to events at remote locations.
- Examples include:
  - most versions of video and audio equipment currently connected to multi-media workstations and
  - some video-phone systems.
- Audio and video viewing attacks normally depend on breaking into the operating system and then enabling a built-in function.



A53: repair-replace-remove information

- Repair processes are exploited to extract, modify, or destroy information.
- Examples include:
  - computer repair shops copying information and reselling it, and
  - maintenance people introducing computer viruses.
- This attack requires involvement in the repair process and is normally not directed at a particular victim from its inception but rather directed toward an audience (market segment). There is little complexity involved in carrying out the attack once the position as a repair provider is established.

A54: wire closet attacks

- Break into the wire closet and alter the physical network.
- Examples include:
  - wire tapping,
  - malicious destruction of wiring causing service disruption, and
  - the introduction of video tape players into surveillance channels to hide physical access.
- Wire closet attacks require only technology knowledge and access to the wire closet.

A55: shoulder surfing

- Watching over peoples' shoulders as they use information or information systems.
- Examples include:
  - watching people as they enter their passwords,
  - watching air travelers as they use their computers and review documents, and
  - observing users to understand standard operating procedures.
- This is a trivial attack to carry out.

A56: data aggregation

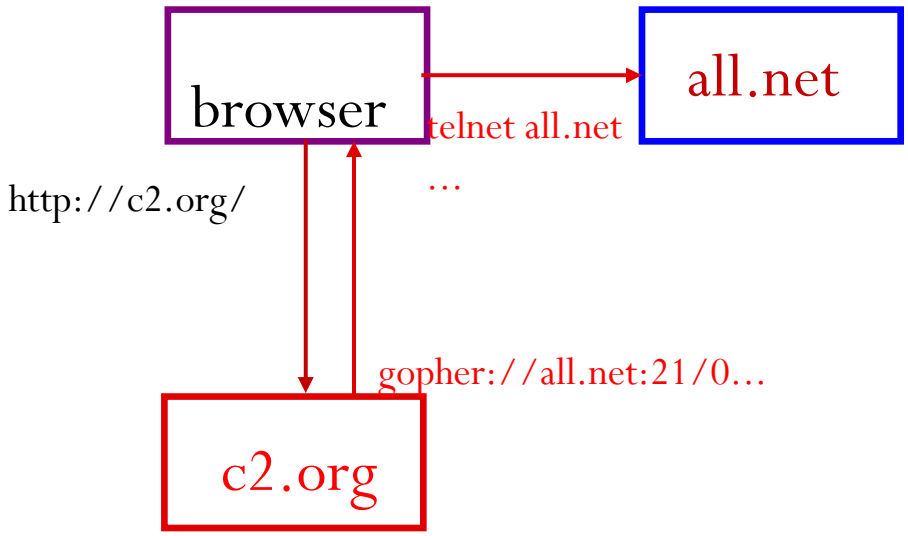
- Legitimately accessible data is aggregated to derive unauthorized information.
- Examples include:
  - getting the total department salary just before and after a new employee is hired to derive the salary of the new hire,
  - attending a wide range of meetings in a particular area in order to gain an overall picture of what work a group is doing, or
  - tracking movements of many people and correlating that information with job titles to derive intelligence indicators.
- Data aggregation can be quite complex both to perform and to protect against. Some work on protecting against these attacks has led to identifying NP-complete problems.

A57: process bypassing

- Bypassing a normal process in order to gain advantage.
- Examples include:
  - retail returns department employees entering false return data in order to generate refund checks,
  - use of computer networks to generate additional checks after legitimate checks have passed integrity checks, and
  - altering pricing records to reflect false inventory levels to cover up thefts.
- This attack is often accomplished by a relatively unsophisticated attacker using only knowledge gained while on the job.

A58: content-based attacks

- The content sent to an interpreter causes the interpreter to act inappropriately.
- Examples include:
  - Web-based URLs that bypass firewalls.
  - malicious macros written in spreadsheet or word processing languages, and
  - compressed archives that contain files with name clashes causing key system files to be overwritten on decompression.
- Many content-based attacks are quite simple or are easily derived from published information. They tend to be quick to operate and simple to program.



A59: backup theft, corruption, or destruction

- Backups protected less comprehensively than on-line copies of information are attacked.
- Examples include:
  - the placement of magnetic devices in backup storage areas in order to erase or corrupt magnetic backups,
  - the infection of backup media by computer viruses, and
  - the theft of backup media being disposed near the end of its lifecycle.
- Except in cases where backup information is encrypted, back-up attacks are straightforward and introduce little complexity.

A60: restoration process corruption or misuse

- The process used to restore information from backup tapes is corrupted or misused to the attackers advantage.
- Examples include:
  - the creation of fake backups containing false information,
  - alteration of tape head alignments so that restoration fails, and
  - use of privileged restoration programs to grant privilege by restoring settings or ownerships to the wrong information.
- Creating fake backups may be complicated by having to reproduce much of what is present on actual backups at a particular site, etc.

A61: hang-up hooking

- Activity termination protocols fail or are interrupted so that termination does not complete properly and the protocol is taken over by the attacker.
- Examples include:
  - modem hang-up failures leaving logged-in terminal sessions open to abuse,
  - interrupted telnet sessions taken over by attackers, preventing proper protocol completion, and
  - refusing to completely disconnect from a call-back modem at the CO, causing the call-back mechanism to fail.
- These classes of attacks are normally simple to carry out.

A62: call forwarding fakery

- Call forwarding capabilities are abused.
- Examples include:
  - the use of computer controlled call forwarding to forward calls from call-back modems to the attacker,
  - forwarding calls to illegitimate locations so as to intercept communications, and
  - the use of programmable call forwarding to cause long distance calls to be billed to the forwarding party's account.
- This class of attacks are relatively simple to carry out but often require a precondition of breaking into a system involved in the forwarding operation.

A63: input overflow

- Excessive input is used to overrun input buffers, thus overwriting program or data storage so as to grant the attacker undesired access.
- Examples include:
  - sendmail overflows resulting in unlimited system access over the Internet,
  - Web server overflows granting Internet attackers unlimited access to servers, and
  - buffer overruns in privileged programs allowing users to gain privilege.
- In the case of denial of service, these attacks are trivial to carry out with a high probability of success. About 20 new examples were published in May of 1997 alone.

A64: illegal value insertion

- Values not permitted by the specification but allowed to pass the implementation are used to cause abnormal results.
- Examples include:
  - negative dates producing negative interest which accrues to the benefit of the attacker,
  - cash withdrawal values which overflow signed integers in balance adjustment causing large withdrawals to appear as large deposits, and
  - pointer values sent to system calls that point to areas outside of authorized address space for the calling party.
- Most such attacks are easily carried out once discovered, but systematically discovering such attacks is, in general, similar to the complexity of gray box testing.

A65: residual data gathering

- Data left as a result of incomplete or inadequate deletion is gathered.
- Examples include:
  - object reuse attacks like the DOS undelete command,
  - electromagnetic analysis of deleted media to regain deleted bits, and
  - electron microscopy techniques used to extract overwritten data.
- Residual data gathering in the case of simple undeletions or allocating large volumes of space and examining their content is straightforward. Looking for residual data on magnetic media using electromagnetic measurements and electron microscopy is somewhat more complex.

A66: privileged program misuse

- Programs with privilege are misused so as to provide unauthorized privileged functions
- Examples include:
  - the use of a backup restoration program by an operator to intentionally restore the wrong information,
  - misuse of an automated script processing facility by forcing it to make illicit copies of legitimate records, and
  - the use of configuration management tools to create vulnerabilities.
- Once a vulnerability has been identified, exploitation is straightforward. Systematically discovering such attacks is, in general, similar to the complexity of gray box testing.

A67: error-induced misoperation

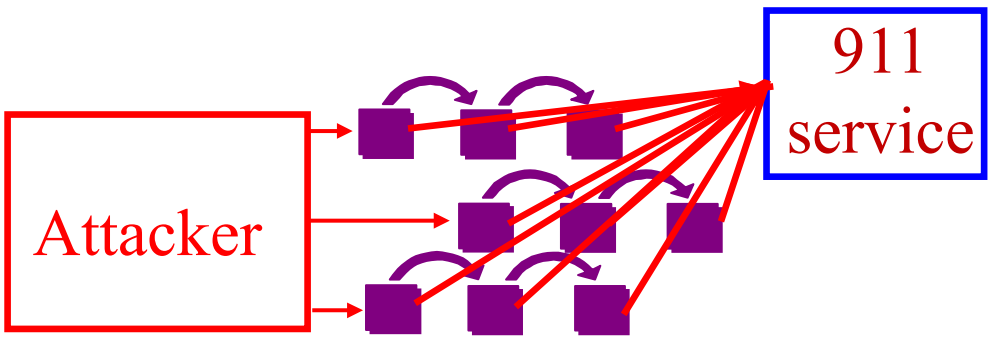
- Errors caused by the attacker induce incorrect operations.
- Examples include:
  - the creation of a faulty network connection to deny network services,
  - the intentional introduction of incorrect data resulting in incorrect output (i.e., garbage in - garbage out), and
  - the use of a scratched and bent diskette in a disk drive to cause the drive to permanently fail.
- Many of these attacks appear to be trivial to accomplish.

A68: audit suppression

- Audit trails are prevented from operating properly.
- Examples include:
  - overloading audit mechanisms with irrelevant data so as to prevent proper recording of malicious behavior,
  - network packet corruption to prevent network-based audit trails from being properly recorded, and
  - consuming some resource critical to the auditing process so as to prevent audit from being generated or kept.
- This class of attacks has not been thoroughly analyzed from a mathematical standpoint, but it appears that in most systems, audit trail suppression is straightforward.

A69: induced stress failures

- Stresses induced on a system cause it to fail.
- Examples include:
  - paging monsters that result in excessive paging and reduced performance,
  - process viruses that consume various system resources, and
  - large numbers of network packets per unit time which tie up systems by forcing excessive high-priority network interrupt processing.
- Although some attacks of this sort appear to be available without substantial effort, in general, understanding the implications of stress on multiprocessing systems is beyond the current theory.



A70: hardware-system failure-flaw exploitation

- Known hardware or system flaws are exploited by the attacker.
- Examples include:
  - a hardware flaw permitting a power-down instruction to be executed by a non-privileged user,
  - causing an operating system to use results of a known calculation error in a microprocessor for a key decision, and
  - sending a packet with a parameter that is improperly handled by a network component.
- Discovering hardware flaws is, in general, similar in complexity to discovering software flaws, which makes this problem at least NP-complete.

A71: false updates

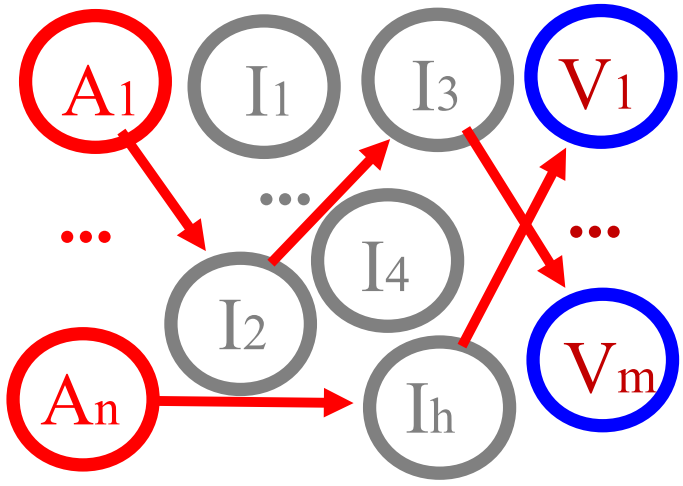
- Causing illegitimate updates to be made.
- Examples include:
  - sending a forged update disk containing attack code to a victim,
  - interrupting the normal distribution channel and introducing an intentionally flawed distribution, and
  - substituting a false update disk for a real one at the vendor or customer site.
- This attack appears to be easily carried out against many installations and examples have shown that even well-trained and adequately briefed employees fail to prevent such an attack.

A72: network service and protocol attacks

- Characteristics of network services are exploited by the attacker.
- Examples include:
  - the creation of infinite protocol loops which result in denial of services (e.g., echo packets under IP),
  - the use of information packets under the Network News Transfer Protocol to map out a remote site, and
  - use of the Source Quench protocol element to reduce traffic rates through select network paths.
- Analyzing protocol specifications to find candidate attacks appears to be straightforward and implementing many of these attacks has proven within the ability of an average programmer.

A73: distributed coordinated attacks

- A set of attackers use a set of intermediary systems to attack a set of victims.
- Examples include:
  - a Web-based attack causing thousands of browsers to attack a single victim site,
  - a set of simultaneous attacks by a coordinated group of attackers, and
- Devising DCAs appears to be simple while tracing a DCA to a source can be quite complex.



A74: man-in-the-middle

- The attacker positions forces between two communicating parties and both intercepts and relays information between the parties so that each believes they are talking directly to the other when, in fact, both are communicating through the attacker.
- Examples include:
  - public key cryptosystem attacks let a man-in-them-middle to fool both parties,
  - an attacker takes over an ongoing telecommunications session when one party decides to terminate it, and
  - WWW gateway machines that rewrite URLs on the fly to cause traffic to always go through their site.
- Man-in-the-middle attacks normally require the implementation of a near-real-time capability, but there are no mathematical impediments to most such attacks.

A75: selected plaintext

- The attacker gets one of the parties to encrypt or sign messages of the attacker's choosing, thus causing information about the victim's system to be revealed.
- Examples include:
  - causing a user of the RSA signature system to reveal their secret key through a series of signatures,
  - the introduction of malicious commands into the data entry stream of a victim who is blindly following directions of a remote person claiming to be assisting them, and
  - inducing a bank to make a series of attacker-specified transactions to cause cryptographic protocols, methods, or keys to be revealed.
- Selected plaintext attacks have differing complexity depending on the system under attack.

A76: replay attacks

- Communicated information is replayed and causes unanticipated side effects.
- Examples include:
  - the replay of encrypted funds transfer transmissions so as to cause multiples of an original sum to be transferred,
  - replay of coded messages causing the repeated movement of troops,
  - replay of transaction sequences that simulate behavior so as to cover up actual behavior, and
  - the delayed replay of events such as races so as to deceive a victim.
- Replay attacks are typically simple to perform and require little or no sophistication.

A77: cryptanalysis

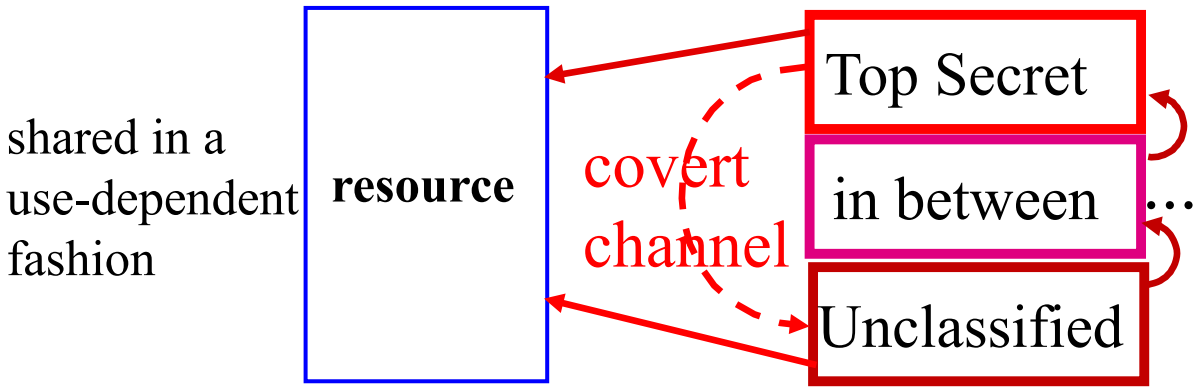
- Cryptographic techniques are analyzed so as to find methods to break codes used to secure information.
- Examples include:
  - frequency analysis for breaking monoalphabetic substitution ciphers,
  - index of coincidence analysis for breaking polyalphabetic substitution ciphers,
  - the breaking of the Enigma cipher in World War II through mathematical and optical techniques combined with knowledge of keys and key usage,
  - exhaustive attacks on the DES encryption standard, code-listeners for breaking many analog speech encoding systems, and
  - improved factoring for breaking cryptosystems based on modular arithmetic.
- Cryptanalysis is a widely studied area.

A78: breaking key management systems

- Keys in cryptographic systems are managed by imperfect management systems that are attacked in order to gain access to keying materials.
- Examples include:
  - attacks based on inadequate randomness in key generation techniques,
  - exploitation of selected plaintext attacks against inadequately implemented automated encryption systems, and
  - breaking into computers housing keying materials.
- Many key management attacks require a substantial amount of computing power, but this is normally on the order of only a few million computations to break a key that could not be broken exhaustively under any feasible scheme.

A79: covert channels

- Channels not normally intended for information flow are used to flow information.
- Examples include:
  - widely known covert channels in secure operating systems,
  - time-based covert channel exploitation in encryption engines, and
  - covert channels created by association of movements of people with activities.
- It has been shown that in any system using shared resources in a non-fixed fashion, covert channels exist. They are typically easy to exploit using Shannon's communications theory.



A80: error insertion and analysis

- Errors are induced into systems to reveal information about or from those systems.
- Examples include:
  - recent demonstrations of methods for inducing errors so as to reveal keys stored in smart-cards and other similar key-transportation devices
  - the introduction of multiple errors into redundant systems so as to cause redundancy to fail.
- The complexity of error insertion is not known, however many researchers have recently claimed to have produced efficient and reliable insertion techniques. The mathematics in this area is quite new and definitive results are still pending.

A81: reflexive control

- Reflexive reactions are exploited by the attacker to induce desired behaviors.
- Examples include:
  - the creation of attacks that appear to come from a friend so as to cause automated response systems to shut down friendly communication,
  - induction of select flaws into the power grid so as to cause SCADA systems to reroute power to the financial advantage of select suppliers, and
  - the use of forged or interrupted signals so as to cause friendly fire incidents.
- The concept of reflexive control is easily understood, and for simplistic automated response systems, finding exploitations appears to be quite simple, but there has been little mathematical work in this are.

A82: dependency analysis and exploitation

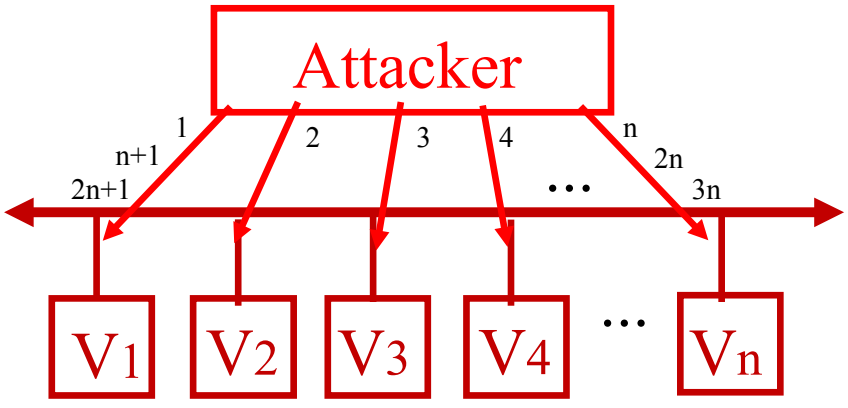
- Interdependencies of systems and components are analyzed so as to determine indirect effects and attack weak points upon which strong points depend.
- Examples include:
  - attacking medical information systems in order to disrupt military deployments,
  - attacking the supply chain in order to corrupt information in an organization,
  - and attacking power grid elements in order to disrupt financial systems.
- The analysis of dependencies appears to require substantial detailed knowledge of an operation or similar operations. Finding common critical dependencies appears to be straightforward, but producing desired and controllable effects may be more complex.

A83: interprocess communication attacks

- Interprocess communications channels are attacked in order to subvert normal functioning.
- Examples include:
  - the introduction of false interprocess signals in a network interprocess communications protocol causing misbehavior of trusted programs,
  - disruption by resource exhaustion so as to prevent proper checking or reduce or eliminate functionality, and
  - observation of data stored in shared temporary data files so as to gain unauthorized access.
- Interprocess communication attacks oriented toward disruption appear to be easily accomplished, but no mathematical analysis of this class of attacks has been published to date.

A84: below-threshold attacks

- Attack detection based on thresholds of activity that differentiate between attacks and similar non-malicious behaviors is exploited by launching attacks that operate below the detection threshold.
- Examples include:
  - breadth-first password guessing attacks,
  - breadth-first port scanning attacks, and
  - low bandwidth covert channel exploits.
- Remaining below detection thresholds is straightforward if the thresholds are known and not possible to guarantee if they are unknown. In most cases, estimates based on comparable policies or widely published standards are adequate to accomplish below-threshold attacks.



A85: peer relationship exploitation

- The transitive trust relationships created by peer-networking are exploited so as to expand privileges to the transitive closure of peer trust.
- Examples include:
  - the activities carried out by the Morris Internet virus in 1988,
  - the exploitation of remote hosts (.rhosts) files in many networks, and
  - the exploitation of remote software distribution channels as a channel for attack.
- Exploiting peer relationships appears to be easily accomplished, requiring only a cursory examination of history for a set of candidate peers and trial and error for exploitation.

A86: inappropriate defaults

- Unchanged default values set into systems at the factory or in a standard distribution process are known to and exploited by attackers to gain unauthorized access.
- Example include:
  - default passwords,
  - default accounts, and
  - default protection settings.
- It may be quite difficult to create a comprehensive lists of appropriate defaults for any nontrivial system because the optimal settings are determined by the application. No substantial mathematics has been done on analyzing the complexity of finding proper settings, but many lists of improper defaults published for select operating systems exist.

A87: piggybacking

- Exploiting a (usually false) association to gain advantage.
- Examples include:
  - walking into a secure facility with a group of other people as one of the crowd,
  - acting like an ex-policeman to gain intelligence about ongoing police activities, and
  - adding a floppy disk to a series of floppy disks delivered as part of a normal update process.
- No published measures of complexity for piggybacking attacks have been made to date, however, certain types of these attacks appear to be trivially carried out.

A88: collaborative misuse

- Collaboration of several parties or identities in order to misuse a system.
- Examples include:
  - creation of a false identity by one party and entry of that identity into a computer database by a second party,
  - provision of attack software by an outsider to an insider participating in an information theft,
  - partitioning of elements of an attack into multiple parts for coordinated execution so as to conceal the fact of or source of an attack, and
  - providing of alibis by one party to another when the collaborated in a crime.
- Collaborative misuse has not been extensively analyzed mathematically.

A89: race conditions

- Interdependent sequences of events are interrupted by other sequences of events that destroy critical dependencies.
- Examples include:
  - checking for the existence of a file before creating it interrupted by the creation of a file of the same name by another owner,
  - the replacement of a mounted file system previously loaded with data in a start-up process, and
  - the mounting of a different tape between an initial read-through and a subsequent restoration.
- Race conditions are not easy to detect. Some automated analysis tools have been implemented to detect certain classes of race conditions in source code and have shown promise.

A90: strategic or tactical deceptions

- Deceptions have been categorized as:
  - concealment, camouflage, false and planted information, ruses, displays, demonstrations, feints, lies, and insight [Dunnigan95]
- Examples include:
  - the creation of a questionnaire asking for detailed security backgrounds under the auspices of a possible contract,
  - the creation of a false front organization such as a garbage collection business to gain access to information in trash, and
  - claiming special capabilities in your upcoming product in order to force other vendors to work in that area even though you never intend to enter into it.
- In general deceptions comprise a complex class of techniques.

A91: combinations and sequences

- Many attacks combine several techniques synergistically in order to affect their goal.
- Examples include:
  - exploiting an emergency response to a flood to gain entry into a terminal room where password guessing gains entry into a system and subsequent data diddling alters billing records,
  - the use of a virus to create protection missettings, exploited by planting a Trojan horse to allow subsequent reentry, and
  - the creation of an attractive Web site designed to attack users who visit it by exploiting browser flaws to set up covert channels through firewalls.
- Combinations and sequences of attacks are at least as complex to create as their individual components, but may be easier to detect.

A92: kiting

- Inherent delays are exploited by creating a ring of events that chase each others' tails, thus creating the dynamic illusion that things are different than the static case would support.
- Examples include:
  - check kiting schemes where delays in processing checks causes the temporary appearance of more funds,
  - techniques for avoiding payments of debts for a long time based on legally imposed delays in and rules regarding the collection of debts by third parties, and
  - the use of revoked keys in key management systems without adequate revocation protocols.
- The complexity of kiting schemes has not been mathematically analyzed in published literature to date.

A93: salami attacks

- Many small transactions are used together for a larger aggregated effect.
- Examples include:
  - taking round-off errors from interest computations and adding them to the thief's account balance,
  - the slow leakage of information through covert channels at rates below normal detection thresholds, and
  - economic intelligence gathering efforts involving the aggregation of small amounts of information from many sources to derive an overall picture of an organization.
- Attacks of this sort are relatively easy to create. Mathematical analysis of the general class of salami attacks has not been done but it seems likely to be similar in complexity to data aggregation effects.

A94: repudiation

- A transaction or other operation is repudiated by the party recorded as initiating it.
- Examples include:
  - repudiating a stock trade,
  - claiming your account was broken into and that you didn't do it, and
  - asserting that an electronic funds transfer was not done.
- Repudiation has been addressed with cryptographic techniques, but for the most part, these techniques are easily broken in the sense that a person wishing to repudiate a future transaction can always act to make repudiation supportable. In stock trades, this problem is addressed by recording all telephone calls and using the recorded message to resolve the issue when a disagreement is identified.

Attack Process / Considerations

- Decide on your goals
  - Motives drive goals
- Gather intelligence required to attain goals
  - Intelligence leaves evidence
- Create a set of plans to achieve goals
  - Planning level implies resources
- Determine risks and benefits of plans
  - Cost, risk taking, etc. imply attacker type
- Do experiments to assure that things will work
  - Experiments require resources, leave trails
- Prepare contingencies for uncertain elements
  - Complexity of plans implies sophistication
- Begin attacks
  - Detections should be run to ground
- Adapt as you go
  - Adaption implies sophistication
- Achieve goals
  - Goals imply motives