



Threats for computers and networks

Security and Privacy
VSE Prag, 9 - 13.6.2008

Institute for Information Processing and
Microprocessor Technology (FIM)
Johannes Kepler University Linz, Austria

E-Mail: sonntag@fim.uni-linz.ac.at
<http://www.fim.uni-linz.ac.at/staff/sonntag.htm>



- What is Malware?
 - Viruses, Trojans, Worms, Rootkits, Backdoors etc
- Crime in the Internet
 - Spam, Phishing, Espionage, Vandalism
- Securing computer
 - Anti-Virus software
- Securing networks
 - Firewalls, IDS
- CVSS



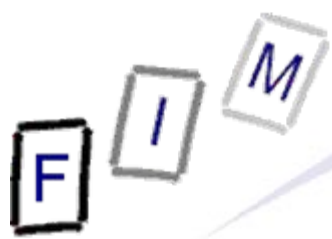
What is "Malware"?

- Malware = Malicious software
 - Any software designed to cause "harm" to ICT equipment
 - Requires this to be done without (informed) consent
 - » Some users really want spyware and install it deliberately ...!
- Reasons for Malware:
 - "Can do": Pranks, proving programming proficiency, ...
 - » No actual damage (or only unintentionally)
 - » In old times, today very rare!
 - Vandalization: Destroy data, leave message
 - » To harm others and gain fame; rather rare today
 - Profiteering: For (in-)direct financial gain
 - » Obtaining valuable information for selling or resources (computing power, bandwidth, non-blacklisted E-Mail sender, ...)

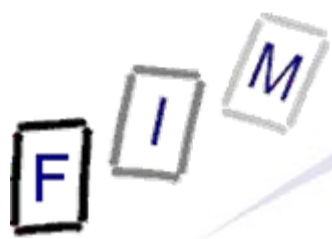


Malware functions

- Malware has typically three functions
 - Not every has all three!
 - » E.g. spyware often lacks infection by being installed deliberately
 - » Vandalism malware has no concealment (or only for a short time)
- ① Infection: Get there without permission
 - Usually requires some kind of user interaction, which is sometimes used as a legal pretext
 - » Often the information is insufficient!
 - Typical examples: Virus, Worm
- ② Concealment: Hide from the user and detection
 - Or from detection/removal programs; to remain on the user for as long as possible
 - Typical examples: Trojan horse, Rootkit, Backdoor
- ③ Profiting: Achieve your aim and making money (payload)
 - Typical examples: Keystroke logger, Botnet, Dialer



- A computer program designed to spread from computer to computer without the user's permission
- Must be transferred to the new destination by a human
 - Inserting an infected diskette, USB stick, forwarding an E-Mail
- Locations of viruses:
 - Boot sectors: Diskettes; rare today
 - Documents: Macros in office documents
 - E-Mails: Some viruses send out copies of themselves
 - » Still require users to execute them, usually by tricking them
 - E.g. pornographic photo, amusing program, important patch, ...
 - Cross-site-scripting: Embedding itself on community sites
 - » Add some code to be shown when visiting this profile/video/...
 - » May send out URLs to bring other users to visit this site



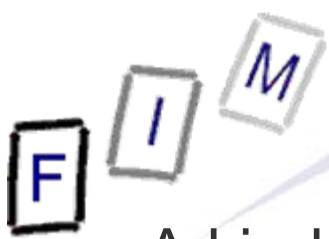
- Methods to avoid detection:
 - Do not infect everything: Small files make it difficult to hide
 - » Infect only sparsely; change techniques below rarely (key, poly.)
 - Hide from OS: Intercept OS calls to modify the result
 - » Reading the infected file returns the perfectly normal data
 - » Countermeasures require directly reading the disk
 - Encryption: Encrypts itself with a random key
 - » Only the decryption part remains unchanged
 - » Dangerous for virus: Self-modifying code is rare and suspicious!
 - Polymorphic code: Similar to encryption, but modifies itself
 - » This includes the decryption part!
 - Replaces small pieces of machine code with different ones with an equivalent functionality



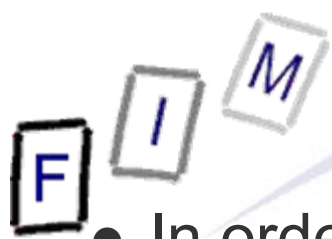
- Similar to a virus, but requires no human interactions
 - Can replicate itself fully automatic!
 - Often used to create botnets
- Examples:
 - Propagation by E-Mail: Sends an E-Mail, which contains an exploit so that mere viewing installs the worm
 - Propagation by Internet: The worm connects to other computers and uses an exploit to install itself there
- Huge problem: Restriction!
 - Worms can easily eat up the whole bandwidth/resources of a network and the computers on them!
 - Difficult for a worm to decide, whether a system has already been infected and should therefore not be attempted again
 - » Some worms install patches for the vulnerability they exploited!



- Software performing a normal function perfectly, but having an additional hidden function as well
 - There is no self-replication, only installation by users!
 - » No hiding behind normal actions, like booting a computer
 - Deliberate installation → gets around security measures
 - » Administrator right? Security warning? → We're just doing a normal installation of a useful program!
- Typical payload of a Trojan: Backdoor (see later)
 - Often also includes a downloading component
 - » So the Trojan remains small; additional code is downloaded over time from a web server or a P2P network and installed
- Extremely simple to program: Ordinary software



- A kind of operating system which virtualizes your normal OS
 - No need for hardware access → needs an exploit (or Trojan)
 - Usually not a goal in itself, but just to hide from detection
- Hides itself even from the operating system
 - Almost no possibility for scanning programs to detect them!
 - Usually grants administrator privileges to the malicious user
- Typically hidden:
 - Files: To keep the one code/data hidden
 - Network connections: For downloading software, accepting commands, passing on the stolen data
 - Registry entries: Which are required for it to run
 - Memory: To avoid detection by inspecting the RAM
- May be purely memory-based to hide even better
 - But will not survive a reboot then!
- "Positive" rootkits exist also: E.g. emulation of CD drives



- In order of difficulty and occurrence in the wild!
- Firmware: Rootkit stored in firmware (system BIOS!)
 - Reprogramming the BIOS; sometimes jumper needed
 - Can also be expansion card BIOSes!
- Virtualized: Modify the boot sequence; boot itself → Then OS
 - Similar to a virtualization software
 - Can intercept all calls to the hardware
 - » Not even reading the hard disk sector by sector can detect it!
- Kernel: Modifying the kernel, typically by a device driver (Win) or a loadable module (Linux)
 - Can modify internal OS structures, e.g. process tables
- Library: Intercept system calls through intended interfaces
 - Can usually be found by comparing the libraries to originals
- Application: Replace system programs by other executables
 - Typically filtering the output from the "real" program



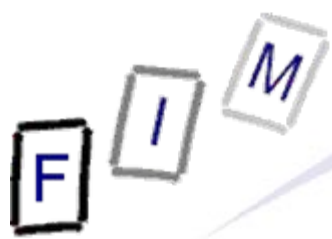
Malware: Rootkit detection

- Comparing the files to originals
 - With numerous OS patches this becomes quite difficult!
 - E.g. "Tripwire": Create hash code and compare against it later
 - Will not work within the system if its file system is subverted!
- Comparing the memory of processes to the code on disk
 - Detects dynamic modifications
 - Compare the sector-wise read DLL to the memory version
- Look for signatures in files, memory, registry, ...
 - Works only for those which are "incomplete" in hiding
- Use statically linked binaries from a read-only memory
 - These can still only detect, what the operating system provides; direct hardware access can still be intercepted
- Generally: Turn off the system & boot from a trusted source
 - Not really usable as continuous precaution for servers!



Malware: The Sony Rootkit

- Sony distributed millions of music CDs containing a rootkit
 - This was installed with the music player to prevent unauthorized copying of the content (DRM)
 - This was not disclosed upon installation
 - » Rootkit was installed before the EULA was shown!
 - There was no possibility to uninstall it
 - » The player could be removed, but not the rootkit part!
- Functionality: Hide any files, registry keys and processes that start with "\$sys\$"
 - Easy to exploit by other malware, which actually **did** happen!
- After discovery, a removal software was provided
 - This would phone home
 - It would only unmask the files, but not remove them
 - Only the second version did actually remove it
- Result: Recall, swap offer, class action (with settlement)



Malware: The Sony Rootkit

4.11.2005: Thomas Hesse
(Sony BMG president of Global Digital Business)

**"Most people don't even know what a rootkit is,
so why should they care about it?"**



- A method to bypass normal authentication
 - Typically a hidden user account or a software stealthily waiting for connections and spawning a root shell
 - Usually an "open" or "symmetric" door: Anyone knowing about it can use it to gain access
 - » Theory: Using asymmetric encryption only the perpetrator could use it, even if it was publicly known!
 - Such approaches exist for specially crafted RSA key generation
- Two main types:
 - Local backdoor: Requires local access; used to become root
 - Remote backdoor: Allows access from anywhere
- Typically an important part of any Malware: Granting the malicious user full control over the machine subverted
 - Dangerous already as such, because of their symmetry!

See <http://www.uscg.iu.edu/hypermail/linux/kernel/0311.0/0666.html> for an example!



Malware: Keystroke loggers

- Recording every single key pressed by the user
 - Related: Taking screenshots in regular intervals
- Used esp. for phishing and generally obtaining passwords
 - Note: From the key log it is not necessarily trivial to isolate the password, but e.g. when the username is known, this becomes easier (example: "root\nsecret\n")
 - Will not help with one-time-passwords and tokens!
 - » Except live monitoring + interception
- Available in hard- and software
 - Especially hardware is also used in legal investigations
- Many countermeasures possible, but most of them are difficult to use or otherwise problematic
 - Web-based keyboards, speech recognition, mouse gestures, Drag&Drop, selecting with the mouse and overwriting, ...



Malware: Hardware keystroke loggers

- Five main types:
 - Inline: Between keyboard cable and computer
 - » Easy to install, but easy to detect
 - Cards: To be mounted within the existing keyboard
 - » Extremely hard to detect
 - Keyboards: Replacement keyboard with logging functionality
 - » Requires an exact duplicate; original may have no marks
 - Dirt, scratches, markings, ... → Probably rarely useful!
 - Sniffers: Listening in on wireless keyboards
 - » Note: Transmission is usually encrypted → Cracking required!
 - Listeners: Each key makes a different sound
 - » Good microphone and training needed!
- Most devices need to be retrieved for obtaining the data
 - If discovered, this is a good moment to apprehend them!

Hardware keystroke loggers



- Image sources:
 - <http://www.keyghost.com/>
 - <http://www.keelog.com/>
 - <http://www.keydevil.com/>
 - <http://spycop.com/>
- Prices: Approx. € 20 - € 120



- Group of automatically working zombie computers
 - A (large) number of home and office computers which have been hacked and are controlled by a single external person
 - » Number can range up to several millions!
- Requires and command & control infrastructure
 - Previously: A hardcoded single server → Vulnerable!
 - » Can be taken down easily, can be used to identify all bots
 - Now: P2P → No central server, updates between bots to recover from control servers being taken down
- Communications is typically over known protocols
 - IRC, DNS (different content), P2P protocols
- Botnets are usually rented to others for use
 - Sending Spam, DoS attack, click fraud, phishing, ...
 - » And, of course, spreading the bot software to obtain new ones!



- Modems connecting to the Internet are redirected to premium rate numbers
 - Also some "legitimate" dialers exist → Provide access to (illegal) content (warez, music, virus code, ...) or porn
 - Based on the premise, that billing is done by a third party (the telecommunication provider) and passed on to the fraudulent person from there → No "individual" billing necessary!
- Today of lessening importance, as dial-up access is continually replaced by ADSL, etc. where no "number" needs to be dialled ("fixed" line termination)
 - Will still work for ISDN/POTS adapters!
 - » No Internet connection through this; but charges will apply!
- Similar approaches: Sending SMS to premium rate services
- Different: Click fraud → The third party is the one defrauded!



- Spyware: Secretly monitoring the user and adapting the behaviour (of some software or the computer)
- Adware: Software with advertising functionality
- These cannot be directly classified as Malware, as many very legitimate examples exist!
 - Spyware: Personalization software
 - Adware: Shareware, demos, freeware with "payment" by ads
- Decision based on:
 - Full disclosure of what is collected and transferred
 - Consent by the user
 - Removability (some Adware removes their "parent" with them!)
- Attention: Many programs exist, which claim to remove spyware → Expensive, but are itself spyware or install more!



Crime in the Internet

- Crime has found the Internet: Because it pays off!
 - Internationality is important here: Committing crimes in foreign countries is an effective barrier to actual prosecution
 - » Little or no information is provided (police cooperation lacking)
 - » It might not even be a crime in these countries!
 - Many "shady" products → Do you want to tell the police your penis enlargement pills didn't show up (or didn't work)?
- Several categories exist
 - Normal crime with evidence in the Internet
 - » Murder with some E-Mails stating the intention
 - Normal crime within the Internet
 - » Defamation, fraud (eBay), paedophilia
 - "Real" Internet crime: Only possible there
 - » Click-fraud, phishing, DoS, viruses, data theft, illegal access

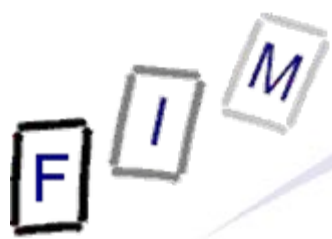
**Internet
crime**

Prosecution is difficult → prevention is more important!



Crime in the Internet: Spam

- In many (but **not all!**) countries sending unrequested advertisement E-Mails is prohibited
- Business areas
 - Obtaining E-Mail addresses (harvesting from websites, hacking computers, generating them, ...)
 - Sending Spam: Renting botnets, (ab-)using open relays
- Often coupled with fraud:
 - Objects sold are not delivered/fake/not working
 - Billing takes place anyway/in advance
- Pays off even for very low response rates
 - 1.000.000 spam mails → ≈ US\$ 100
 - » 0,002 percent response → 20 customers (1 in 50.000!)
- Tool for recruiting victims for other schemes
 - Typical example: Money laundering after phishing



Crime in the Internet: Phishing

- Phishing: Representing as someone else to obtain personal information, which would otherwise have been withheld
 - Usually employed for identity theft
 - A very lucrative business!
 - » Online banking: Average of € 4.500 damage!
- Typical target: Online banking or credit card data
 - User IDs, passwords, PIN, TAN, card number, CVV ...
 - "Victims": Banks, eBay, PayPal
- Phishing technique: Usually two steps
 - Sending out Spam with forged links, requesting users to "confirm" their personal information (or else ...) on the web
 - Forging a website to look like the original
 - » Biggest problem: Making the URL look right
 - » Biggest danger: Cross Site Scripting (→ it **IS** the real website!)



- Exploiting the information gained:
 - Accessing the account and transferring the money to a third person (usually not directly involved)
 - » Credit cards → Buy money, play online casino, buy on eBay, ...
 - Third person retrieves money, pockets 10%, and passes the remaining 90% on by Western Union
 - » Western Union → Practically untraceable!
 - Fourth person picks up money from WU and transfers it on
 - » Requires photo ID; in some countries only number + password!
- Possible precautions:
 - Full content inspection (E-Mails and webpages)
 - Careful reading: Phishing usually misses personal data
 - » "Dear valued customer" instead of "Dear Mr. Sonntag"
 - Browsers alerting (fake URLs, known phishing sites lists, ...)
 - Improved logins (SMS-TAN, recognizing image, iTAN etc.)



- Transaction authentication instead of user authentication
 - Would be a very good help against phishing!
- Basic idea:
 - Generate a password based on destination and amount of the transaction and use this to authenticate it
- Example:
 - Enter destination and amount in a token, which produces a unique password based on this and the token ID
 - Enter the token into the web form
 - The bank verifies it
 - Result: Without the token no transaction is possible, and an existing transaction cannot be "hijacked" and changed

Crime in the Internet: Espionage



- Typically done by secret services/countries/large companies
 - Not much is know; victims usually don't talk or prosecute!
- This is "big business" → There is a lot of money at stake, so the effort is very high
 - Experienced hackers deliberately target specific machines or networks and employ various techniques
 - » May include social engineering, e.g. of management executives
 - Large effort and considerable knowledge
 - » May include bribing "insiders"
 - The Internet is just one attack vector
- Protecting against this is **very** hard!
 - For "normal" companies probably not an important issue
 - Still, some precautions should be taken!



- Typical example: Website defacement
 - Changing the content of the website to show something different, like a political message
 - » Today of little importance any more
 - Modify the content to exploit browser vulnerabilities
 - » Of high importance!
 - » Will target all persons visiting this page
- Prime targets: Large Web 2.0 sites which allow customer-supplied content (or any other site echoing user input)
 - Doesn't require hacking, only lack of verification of input
- Today not only about changing static webpages, but rather changing the database producing the dynamic content
 - SQL injection attacks are very prominent here!



Securing computers

- Every computer today should be secured
 - Not securing it could lead to legal liability!
 - » At least for companies, where the standard is higher
 - » "Normal" measures must be taken, i.e. what everyone else does too and what is seen as the typical minimum investment
- Difficulty of distribution:
 - Keeping scanning software up to date with signatures
 - Making sure it runs everywhere and has not been disabled
 - Ensure that reports are investigated, not just clicked away
 - One possible solution: Back to mainframes!
 - » Not directly, but in the sense of thin-clients: Everything is computed on a central server, and only the UI is remote
 - Remote computers can be secured tightly with read-only software, no interfaces, etc.
 - » Central system can be configured and controlled more easily



- Basic requirement for every computer
 - Looks for viruses (and worms, trojans, etc. depending on manufacturer/product)
- Typical method:
 - Signature scanning: Searching for specific strings (bytes)
 - » The payload, the reproduction area, the decryption part, ...
 - Heuristics: Searching for characteristics in the code
 - » Intended to detect unknown viruses (no signature yet available)
 - » Problem of false positives!
 - » May involve:
 - Executing suspicious code in a virtual machine → What does it do?
 - Decompile and analyze source code
 - Check for "strange" function calls
 - Behavioural blocking: Stopping potentially dangerous behaviour and asking the user for permission
 - » Works very well, but depends largely on the user's attention!



Securing computers: Personal firewall

- Similar to network firewalls, but running only on a single computer and protecting only this one
 - Controls network traffic to and from the computer
 - » Usually not address/port translation
 - Added advantage: Can "inspect" the program also
 - » Which programs running for which user are allowed to connect to which computer/listen on which port
 - E.g.: "explorer.exe" vs. "192.168.2.1" contacting a ssh server
 - Can prompt users whether to allow/deny specific connections
 - » Adapts the protection profile accordingly
 - » Too many warnings → Users will shut it off or just click "yes"
- May itself be the target of an attack (when vulnerable)
 - "Witty" worm: Targeted firewall with buffer overflow
- Typically running as administrator, as normal users are
 - Malware can shut them down/reconfigure them as well!



Securing computers: Content filtering

- Content filtering: Disallowing certain content
 - Typically blocking pornography, advertisements, spam, ...
 - » Companies: Everything unrelated to the work tasks
 - Is only secondarily a security issue!
 - » Many such sites try to infect/harass users (aside from content!)
- Typically based on application gateways
 - Web-proxies, E-Mail servers
- Base functionality:
 - Black-/Whitelists: Explicit lists of what is forbidden/allowed
 - » E.g. attachments of certain types/extensions (.exe, .bat, ...)
 - » May be centralised on a server (and continually updated) or local
 - Content inspection: Looking at the content and deciding, whether this should be passed on or not
 - » Bayesian networks (require training), content anomalies, language, regular expressions, URLs, proximity, ...



Securing networks

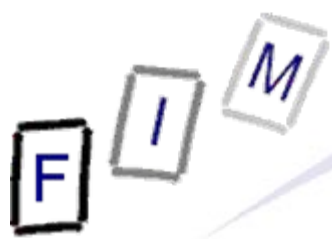
- Networks need to be secured as well as hosts
 - Defence in depth: Don't allow the content in, if it manages to pass, the individual clients try again
 - » Basic premise: Different strategies/programs at each node!
 - If we can't prevent it, at least try to detect it
- Networks should be exactly defined and little connected
 - Only few and clearly defined interfaces between them
 - » These interfaces then perform extensive security measures
 - Inside is connected extensively → barriers are impossible
 - Any wireless network is a huge problem here!
- Thinks especially also of non-computer network components
 - Infrastructure (switches, routers, ...), others (printers)
- Networks are usually the domain of experts, so no "ignorant" user must be dealt with
 - E.g. loosening security for easier working!



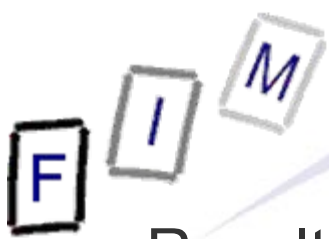
- Firewalls: Blocking in- and outgoing network connections
 - Problem: They see **only** the network traffic; no other information is available → Connection tracking etc.
 - Absolute requirement for each and every network!**
- Contrary to personal firewalls usually at least some incoming connection must be allowed as well
 - Every server application: Web, E-Mail, SSH, VNC, ...
- It must be ensured, that the firewall is on the perimeter
 - I.e., there are no alternative ways into the network
 - » Examples: WLAN, VPN connections
 - Ideally, the internal network is separated into zones as well
 - » DMZ (servers), internal (different departments separately, ...)
- Strictly speaking, the firewall would only inspect packets
 - In most cases content inspection takes place as well



- Rules for firewall rules:
 - Whitelist traffic: Only allow what must be allowed because of necessity; forbid everything else
 - » May require deep packet inspection
 - E.g. FTP: Port number for data connection is passed within
 - Hide information: Do not respond to the outside at all unless necessary (for routing to work; public services)
 - Avoid stateless services: Replies cannot be distinguished from traffic originating from an attacker → Stateful filtering
 - Block known exploits: Enforce passing of "valid" packets only
 - » If a sequence is know to be dangerous, deny it as well
 - Use NAT and private addresses inside
 - » And check the source of all packets
 - Inside address appearing from outside → Drop it!
 - Rate limit dangerous connections if possible
 - » Example: Maximum of one SSH request/second from one IP



- Intrusion Detection Systems should detect unwanted manipulations of computer systems
 - The illegal "being there"
 - In difference to a firewall an IDS doesn't decide/block traffic, it just silently listens and tries to detect (network-based)
 - Host-based IDS are similar to personal firewall + anti-virus SW
 - » Therefore it is located in the inside of the network
 - Special focus therefore on internal → internal attacks as well!
 - » Firewall = Prevention; IDS = Detection (after the fact)
- Can be signature (→ anti virus) or anomaly based
 - Observes the traffic and builds a "baseline" after learning
 - Any deviations from this "normal" behaviour are detected
 - » Usually only significant differences produce alerts, as this method is obviously prone to false positives
 - Rare actions might not be in baseline; some users have a wide spectrum of "normal" behaviour; ...

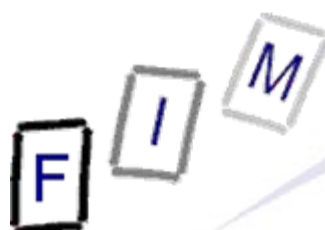


- Result of an alert can be passive or reactive
 - Reactive: Logging of users, blocking switch ports, changing firewall rules, tightening permissions etc.
 - » "Intrusion Prevention System"
 - Passive: Notification of the administrator
- Special IDS variant: Honeypots
 - Systems set up to not be worked with, but attacked
 - » No "normal" change will occur there ever
 - » Contain "interesting" but incorrect/unusable information
 - » Should be "easier" or more attractive to attack
 - » Every modification is therefore a sign of an (successful) attack
 - They are closely monitored and what "happens" to them will then be blocked for the "real" servers!
 - Also used to collect evidence for later prosecution
 - Attention: Should not be usable as stepping stone for further attacks to the network or as **zombies!**

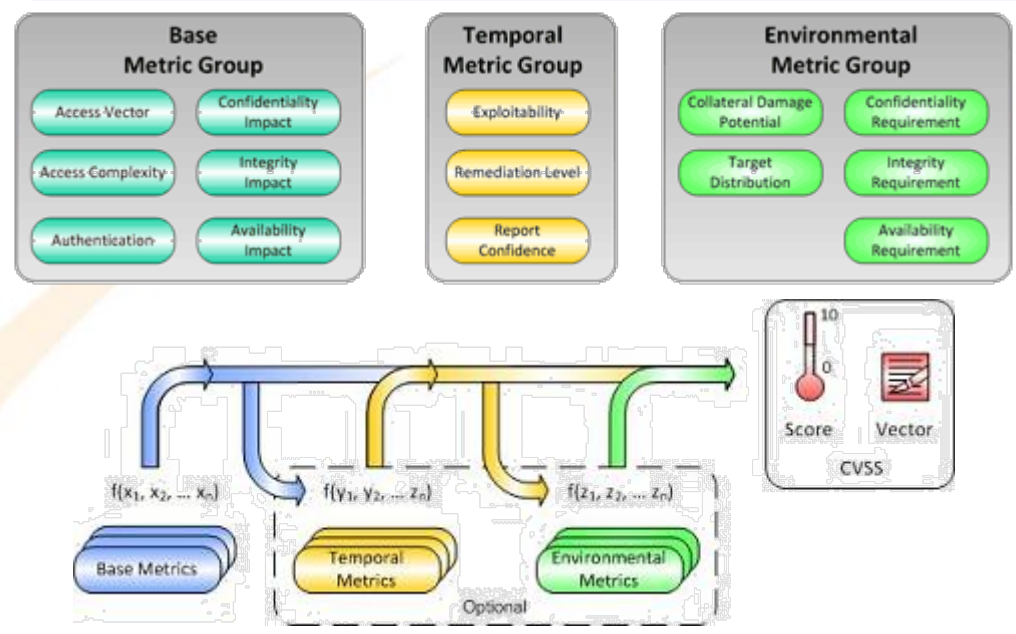
Common Vulnerability Scoring System



- To provide a common way to assess vulnerabilities
 - I.e., how urgent is fixing the problem (ranking)?
- It provides
 - Standardized vulnerability scores: Single score for all hard- & software systems, regardless of vendor (and their scores)
 - » Also immune from their interests/motivations!
 - Open Framework: Clear definition how the score was calc.
 - » Allows comparison between vulnerabilities
 - Prioritized risk: Individualization for each company
 - » Some part of the CVSS is world-applicable, though!
- Note: This is version 2!
- Response (generalized; from CVSSv1 typical responses):
 - 0-3: Wait for service pack; 4-5: Next patch cycle; 6-7: Within 1 week; 7-10: NOW!



How CVSS works

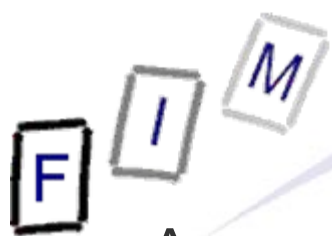


- Three metric groups: Base, temporal and environmental
 - Base: Intrinsic & fundamental characteristics of vulnerability
 - » Constant over time, independent of environment
 - Temporal: What changes over time
 - » Independent of environment
 - Environmental: Properties depending on the actual use of the vulnerable system at the end-users installation



CVSS overview

- Base and temporal metrics specified by security analysts
 - Requires detailed knowledge of the product, the vulnerability and possible/existing exploits
 - Environmental metrics → Calculated by everyone himself!
- Each vulnerability is scored separately
 - Synergistic effects are ignored → Can be added through environmental metrics (security requirements)
- Vulnerabilities are scored according to a typical installation
 - If you have special security precautions, your CVSS could be lower, if security is loosened, higher!
- Worst-case scoring: Better safe than sorry
 - If several attack vectors exist → Easiest one
 - Several products affected, but patch only for one → Unfixed
 - Root level access → Complete loss of C, I, and A!
 - » Plus: Loss of integrity usually affects availability too



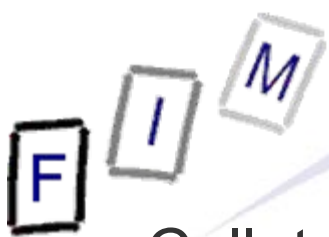
Base metrics

- Access Vector: How the vulnerability can be exploited
 - Local, adjacent network (broadcast/coll. domain), network
- Access Complexity: How difficult is the exploitation after achieving access to the system
 - Low (little skill, manually possible), medium (limit sources, additional data needed, non-default config.), high (special conditions only; race condition, suspicious social eng.)
- Authentication: How often authentication is necessary
 - None, single, multiple (≥ 2 ; even if same credentials)
- Confidentiality Impact: What data can be accessed
 - None, partial (some tables in DB only), complete (everything)
- Integrity Impact: What the attacker could modify
 - None, partial (no control which files), completely (everything)
- Availability Impact: What other users experience
 - None, partial (delays), complete (shutdown, unusable)



Temporal metrics

- **Exploitability:** State of exploit technology available
 - Unproven (theoretical), proof-of-concept (unpractical, needs substantial modifications by skilled attacker), functional (code available and works in most situations), high (exploitable by mobile autonomous code, works every time), not defined
- **Remediation Level:** What is available as a fix
 - Official fix (patch by vendor), temporary fix (official but temporary patch), workaround (unofficial solution), unavailable (no solution, impossible), not defined
- **Report Confidence:** How sure it is the vulnerability exists
 - Unconfirmed (single unconfirmed source, conflicting reports, rumour), uncorroborated (multiple non-official sources), confirmed (acknowledged by vendor, publication, exploit code, exploitation), not defined



Environmental metrics

- Collateral Damage Potential: Potential for loss of life or physical assets; economic/productivity/revenue loss
 - None, low (slight damage), low-medium (moderate loss), medium-high (significant loss), high (catastrophic damage), not defined
 - » Note: What is "slight" or "moderate" is company specific as well!
- Target Distribution: Percentage of systems potent. affected
 - None, low (small scale; 1%-25%), medium (26%-75%), high (76%-100%), not defined
 - » Percentage as affecting the total environment
 - One central single point of failure → 100!
 - » Very high influence on final result!
- Security Requirements: Importance of the asset to the organization; 3 parts (availability, integrity, confidentiality)
 - Low (limited adverse effects), medium (serious effects), high (catastrophic effects), not defined



- Scoring is very complicated and consists of numerous equations and values for each possible value
 - Base score = $((0.6 * \text{Impact}) + (0.4 * \text{Exploitability}) - 1.5) * f(\text{Impact})$
 - » $\text{Impact} = 10.41 * (1 - (1 - \text{ConfImpact}) * (1 - \text{IntegImpact}) * (1 - \text{AvailImpact}))$
 - » $\text{Exploitability} = 20 * \text{AccessVector} * \text{AccessComplexity} * \text{Authentication}$
 - » $f(\text{impact}) = 0$ if $\text{Impact} = 0$, 1.176 otherwise
 - » AccessVector: local=0.395, adjacent=0.646, network=1
 - $\text{TemporalScore} = \text{BaseScore} * \text{Exploitab.} * \text{Remed.Level} * \text{ReportConf.}$
 - $\text{EnvironmentalScore} = (\text{AdjustedTemporal} + (10 - \text{AdjustedTemporal}) * \text{CollateralDamagePotential}) * \text{TargetDistribution}$
 - » AdjustedTemporal = Temporal score recomputed with the base scores impact adjusted by the CAI-requirements
 - » $\text{AdjustedImpact} = \min(10, 10.41 * (1 - (1 - \text{ConfImpact} * \text{ConfReq}) * (1 - \text{IntegImpact} * \text{IntegReq}) * (1 - \text{AvailImpact} * \text{AvailReq})))$
- Complicated calculation! → Use software
 - <http://nvd.nist.gov/cvss.cfm?calculator&adv&version=2>



CVSS example: CVE-2002-0392

Apache Chunked-Encoding Memory Corruption Vulnerability

- Execution of arbitrary code (with webserver privileges) or DoS through incorrect handling of chunked encoding
- Base score = 9
 - Access Vector: Network (possible from remote)
 - Access Complexity: Low (nothing special required)
 - Authentication: None (just send a request)
 - Confidentiality, Integrity Impact = Partial (web content, local user/configuration information; arbitrary code)
 - Availability Impact = Complete (DoS)
- Temporal score = 7.4
 - Exploitability = Functional (exploit code exists)
 - Remediation Level = Official fix (official patch available)
 - Confidence = Confirmed (→ official patch!)
- Environmental: Depending on assessment → 0.0 – 9.2

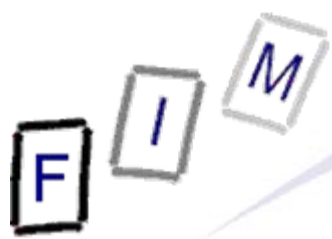


- Many different threats exist → Ignoring them will not help!
- There is no single solution for security
 - Against various attacks various countermeasures are available
- Vulnerabilities must be monitored closely
 - Depending on size, just observe the lists or calculate your own CVSS for all new vulnerabilities yourself
- Minimum configuration for every computer:
 - Anti virus + personal firewall
- Minimum configuration for every network:
 - Firewall
 - Optional: Proxies with content inspection for various protocols
 - » E-Mail: Anti virus, anti spam, WWW: Exploit/virus/URL-filtering
 - » Intrusion detection system for infrastructure hosts and network

F I M

Questions?

Thank you for your attention!



- **Mell, Peter, Scarfone, Karen, Romanosky, Sasha: A Complete Guide to the Common Vulnerability Scoring System Version 2.0**
<http://www.first.org/cvss/cvss-guide.html>