Confidentiality-- assets should be accessible only by authorized (no inapropriate disclosure) Secrecy, anonymity, privacy; **Integrity**-- assets can only be modified by authorized parties/ways (no mod) non-repudiation, authenticity, accountability; **Availability**-- authorized shouldn't be prevented from access (protection against denial-of-service) reliability
Vulnerability: a weakness in the system; A vulnerability can be exploited to cause loss or harm ie: unpatched system; **Threat:** A set of circumstances that has the potential to cause loss or harm; Any potential danger that is associated with the exploitation of a vulnerability; E.g. threat agent – intruder accessing network, a process accessing confidential data, worker copying confidential info; **Risk:** The likelihood of a threat source exploiting a vulnerability and the corresponding business impact ; **Control:** Action/device/procedure/ technique that eliminates/reduces a vulnerability and/or counter threats. Mitigate/reduce potential risk///**Identification & Authentication:**asserting who person is,proving identity by something you know,have,are; **Access Control:** Ensures only authorized users can gain access to protected resources; **Communication Security:** Conceals data against unauthorized access; **Physical Controls:** Detective measures – alarm, metal or movement detectors. Preventive measures – locks on doors, guards, fencing; **Administrative Controls** – "soft controls", screening and management of personnel, password management, security documentation///**Preventative**—avoid incident by blocking attack or removing vulnerability;**Detective** – identifies an incident's activities (either as it happens or some time after the fact); **Corrective** – fixes systems after an incident has occurred; **Deterrent** – to discourage a potential attacker by making the attack harder but not impossible; **Recovery** – to bring the environment back to regular operations
1. Economy of Mechanism (Simplicity) –The security mechanisms should be as simple as possible.Implementation of unnecessary security mechanisms should be avoided **2. Fail-safe Defaults –**When a system fails, it should do so securely,default should be no access; explicit grant access. e.g. if a firewall fails, no packets will be forwarded **3. Complete Mediation -** All requests/access to objects should be checked to ensure that they are allowed.e.g. computer memory enforces checks on every memory access requests **4. Open Design -** the security of a system should not depend on the secrecy of its protection mechanisms, i.e. don't rely on security through obscurity. e.g. cryptography should still be secure if everything, except for the keys, is not kept secret **5. Separation of Privilege –** no complete power – security should not reply only on a single mechanism, two or more conditions must be met before access should be permitted.e.g. the use of token ID in web requests instead of relying only on cookies **6. Least Privilege –** Any component and user of a system should operate using the least set of privileges to complete its job, "N2K" **7. Least common mechanism –**avoid having multiple subjects sharing mechanisms to grant access to a resource, mechanisms used to access resources should not be shared.e.g. sharing of the network with an attacker allows him to eavesdrop packets **8. Psychological acceptability –** easy design interface so user does protection method correctly. if users protection goals = mechanisms to use, mistakes will be minimized, else errors will be made.
ATTACK: Phishing- Social Engineering; **Delivery**: mail w infected file you click on, link to site asking for info or download/install malware (drive-by/with consent)
THREAT: Insider Threat- access rights remain to data when person changes positions, downloading/storing company info **technical controls:** encryption, access control, minimum privilege, log and monitor activities, auditing and reporting **non-technical controls**: enforcing security policy(prohibit access to non-work sites), perform background checks
HTTP Requests: GET= requests data URL; **POST =** submits data POST request; **Cookies:** server issues cookie,browser adds following header to requests back to server
Same Origin Policy: keeps content that came from different origins from each other; browser permits scripts contained in a web page to access data in the other web page only if both web pages have the same origin, script run on one page can modify stuff on another page of the same origin
ATTACK: Broken Authentication - attacking authentication or session management(get control of sesh w cookies & not login) **Design flaws:** don't want the design to be a certain way ie: no/minimal pw quality control **Implementation flaws**: problems with the way things function ie: failed-open login (giving in wrong un/pw still gets you in), multistage logins (bypass some stages, interfere w login)//**ECB Cipher:** tokens can be decrypted **CBC Cipher:** super ciphered,plaintext is put against preceding block of ciphertext//**Authentication Controls:** use strong credentials, handle credentials secretly, prevent misuse of account recovery, Log,monitor, notify **Session Management Controls:**Generate Strong Tokens, RANDOM, protect tokens(only over HTTPS, invalidate old tokens) LOG, MONITOR, ALERT (requests w invalid tokens=alerts)
ATTACK: Injection- attack vector: almost any source of data can be an injection vector, security weakness: very prevalent impact: data loss, corruption, disclosure to attackers, loss of accountability, denial of access;OS Command Injection, Injecting via ASP, SQL Injection
ATTACK: CROSS-SITE SCRIPTING -goal is to inject script on victims page, your page can run my stuff, same origin policy is broken; **Stored XXS :** Persistant, Samy's Myspace Worm **Reflected XXS :** non-persistant, "help" message// **Risk:** virtual defacement(inject malicious data to feed misleading data to users), inject trojan functionality, mine cryptocurrencies!!!! and escalating the client-side attack (capture browsing history, key strokes, whatever)//**Control:** input validation (check/sanitize),
ATTACK: Cross-Site Request Forgery: exploits a websites assumption that all requests that originate from a users browser is the user, you click on one site, click on an attacker site and they send a request to the site you visited through their browser// **Risk:** transfering money to another bank account, change users pw, add/delete content from website, targets state changing request//**Control:** check origin, open new browser, no "remember me", unpredictable tokens w timestamp, automatic logout, confirmation page
ATTACK: XXE = XML External Entities: Attack vector: attackers can exploit vulnerable XML processors if they can upload XML or put hostile content in XML doc; Security Weakness: older XML processors allow specification of external entity; Impact: data extraction, execute remote request from server, scan internal systems, denial-of-service
Requirements & Analysis>> Functional Requirements (system allows users to view books info by category, will allow users to add new info)Non-functional Requirements(UI will be suited for screens w certain dimension, will be available 99.99% of time for 24 hr pds)/Security Requirements & Threat Modeling/Security Risk Analysis; **Design** >> ER diagrams; Threat Modeling/Security Risk Analysis; **Implementation** >> code; Security Code Reviews; **Testing & Assurance >>** Security/Penetration Testing/Configuration
Data Flow Diagram (DFD): Data flow between different entities (external agent >> outside person/unit/system/organization that interacts w system; data store >> data at rest, persons, places, objects, events, file or db; process >> work performed by system in response to incoming data flow or condition) Req&Analysis
Risk Analysis Process Framework: Assets(cost/importance/impacts) + threats (likelihood, severity, impacts) + vulnerabilities(likelihood) -> risks -> security measures //**Risk Analysis Approach:** Quantitative RA >> monetary/numeric value to asset val,threat freq,severity of vulnerability; Single Loss Expectancy formula = Asset Value x Exposure Factor; Annual Loss Expectancy = SLE x Annualized Rate of Occurence(freq of threat taking place in 12 month timeframe); Qualitative RA >> opinion & scenario based rating scale
Residual Risk = (threats x vulns x asset value) x controls gap OR total risk - countermeasures//**Handling Risk:** Risk reduction/mitigation- implement a countermeasure; Risk transference- transfer portion/all of potential cost of loss to 3rd party(insurance): Risk acceptance- do nothing, deal w potential loss&cost if risk occurs; Risk avoidance- discontinue activity//**Threat Modeling:** uses models to find security problems, use abstractions to think about risks, find issues in things you havent built yet, process to understand security threat to a system, KEY TO A FOCUSED DEFENSE//Threat Modeling Process: to understand security reqs, find problems while there is time to fix them, build mitigation into design
1. What are you building? characterize system: Diagrams; Most threats occur across boundaries; identify who controls what in system **2. What can go wrong?** find threats; Don't think like an adversary!! Too unstructured; More structured way is to identify your assets, then identify the threats on those assets; Threat modeling approaches: focus on assets, focus on attacks, focus on software **Attack Trees:** goal is root node, different ways of achieving goal as leaf nodes; **STRIDE (focus on software):** you can apply STRIDE to each element (external, process, data store, data flow) Spoofing: pretending to be someone/something your not (Spoof w web browser), Tampering: modifying something youre not supposed to (Tampering w web browser: request forgery), Repudiation: claiming you didnt do something, Information Disclosure: exposing info to unauthorized peep(confidentiality issue); Denial of Service: reducing ability of valid users to access resource; Elevation of privilege: when unprivileged gains privileged status **3. What should you do about what can go wrong?** Address threats, Mitigate threats- reduce risk by making it harder for an attacker to take advantage of a threat with control .ie: Spoofing person >> strategy: identification & authorization, Tampering w file >> strategy: OS, Logs come under attack(repudiation) >> strategy: protect logs, network monitoring(info disclosure) >> encryption, Eliminating threats – removing the function/feature associated with the risk; Transferring threats – letting someone or something else handle the risk; Accepting the risk – if not worth the expense or cost **4. Did you analyze well?** check model: what you built, check threats: did you do right response/ find all threats, check your tests: make sure you built a good test to detect problem
Functional requirement (behavioral) - something the system must do, function to be provided by the system in terms of an operation that can be used by an agent ie: system shall allow user to edit personal data, technical(time), soft/hardware; **Nonfunctional requirement (NFR)** - property or quality the system must have ie: security,performance,availability, confidentiality; Quality Requirements ie: temp control cycle must execute in 100 milisec; Constraints requirements - Imposed by the client or the environment in which the system operates. ie: system shall provide personnel info only to HR, cost, operation(tech support/storage space) **Specification Properties:** Requirement validation involves checking that the specification is complete,, consistent, unambiguous, and correct **Security Goals:** something that any stakeholder wishes to achieve or avoid, general statement about the security of an asset,broad behavior of system, apply to entire system, not testable e.g. The system shall prevent/detect action on/to/with asset. **Determining Security Goals** Applying the management principles (least privilege, separation of duties, audit trails etc.) to the assets and business goals of the system, based on threat identification >>Asset Identification: Personal Information>>Harm Identification: possible hard to personal information>>Confidentiality, Integrity, Availability **Defining Security Requirements:** Requirements are specific, apply to functional reqs & should be testable, and are for stakeholders. Internal Sources (systems specification, functional reqs, threat modeling); External Sources (Regulations and Compliances, national privacy laws) **Analyzing Security Requirements:** Verification -analysis of an artifact on its own, ensuring its completeness and internal consistency; Validation- relationship of the artifact to the artifacts from which it is derived **Security requirements satisfaction arguments are in 2 parts:** The 'formal' outer argument – constructed based on the behavior specification for the system-Uses claims about system behavior to demonstrate that the security requirement is satisfied in a system that confirms to the behavior specification-Using a chosen logic, perform proof that the security requirement is satisfied.The 'informal' inner argument – constructed to support the outer argument.-The inner argument is a set of informal arguments that support the claims used in the outer argument.