CWE - Common Weakness Enumeration

CAPEC - Common Attack Pattern Enumeration and Classification

➔ Link attack (CAPEC-ID) to vulnerability (CWE-ID)

CVE - Common Vulnerabilities and Exposures

Project started in 2006; goal: easier comparison and data exchange between software security tools

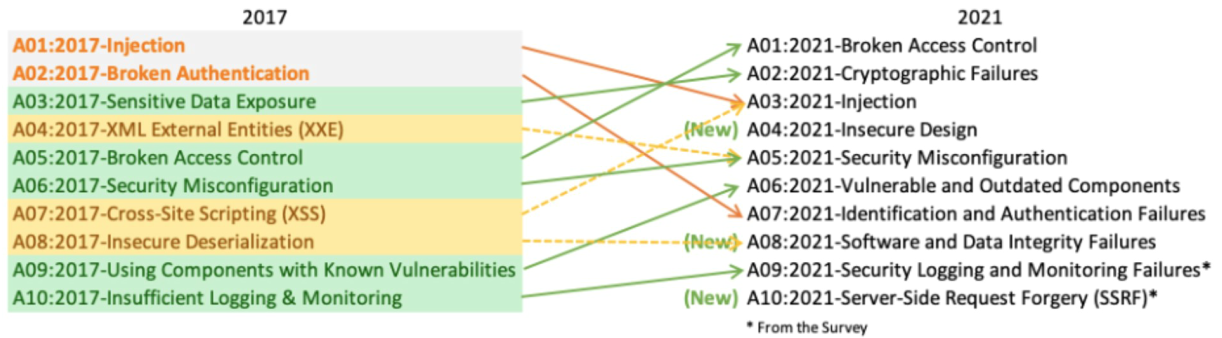
OWASP Open Web Application Security Project

Changes 2017 ➔ 2021

More often:

− Broken Access Control, Security, Misconfiguration, Vulnerable and Outdated Components

− Less often: Injection, Identification and Authentication Failures



A01 - Access control might be present, but may not cover all access paths to an object

Ex: Check rights before displaying Url but not when accessing it

A02 - Many web application do not properly protect sensitive data, such as credit cards, social security numbers, and

authentication credentials, with appropriate encryption or hashing.

A03 - SQL, OS(Shell), XSS

A04 - Missing or ineffective control design

A05 - Security depends on having a secure configuration defined for the application, framework, web server,

application server, and platform. All these settings should be defined, implemented,

and maintained as many are not shipped with secure defaults.

A06 - Outdated libraries, frameworks, and other software modules with known vulnerabilities

A07 - Application functions related to authentication and session management are often not implemented correctly,

allowing attackers to compromise passwords/keys/session tokens to assume other users’ identities.

A08 - Integration of plugins/libraries from untrusted sources, e.g., repositories, remote servers, CDNs

A09 - Cant detect many failed attacks -> cant prepare/improve -> someday attackers find a way in

A10 - Web application fetches remote resource without validating user-supplied URL. Server can access resources

with fewer restrictions than client.

Race Condition - Time of check to time of use (TOCTTOU)

Ex: Test if access to resource is allowed -> change/create a link -> access something where access wasn’t granted

Trusted Path - EX: STRG + ALT + ENTF führt immer zu OS screen (taskmanager, log-of etc), Home Button on mobile

ACL - Access Control List

Software Testing - Fuzzing: Generate Random Input, see if program handles it poor or even crashes

Testing can only confirm presence of vulnerabilities but not absence

INTSEC

CIA - confidentiality, integrity, availability

Accountability - prove that actions happened

Privacy - control of personal data and space

Privacy vs confidentiality

* Prevent some data from being processed/accessed - for the rest keep it confidential

Hexad(Parker) - CIA +

Control - data stolen but not used (yet) -> lost control over it

Authenticity - Signature, proof where information comes from

Utility - encypt data but loose key -> breach of utility (verhältnismäßigkeit)

Data Protection principles:

* Data minimization
* Data Subject should know what has been stored
* Data subject must consent/be asked in advance

EU General Data Protection Regulation(no directive -> does not need to be implemented by member states)

* Explicit vs assumed Consent
* Right to be forgotten
* Easier access and transfer to different provider
* Privacy by design/default
* Notification about data breaches
* High fines(20mio or 4%)

Security by Obscurity vs. Kerckhoffs's principle

Saltzer/Schroeder(1975) 8+2

**1. Economy of mechanism**: Keep the design as simple and small as possible.

**2.** **Fail-safe defaults**: Base access decisions on permission rather than exclusion.

**3.** **Complete mediation**: Every access to every object must be checked for authority.

**4.** **Open design**: The design should not be secret. - Kerckhoffs's principle

**5**. **Separation of privilege**: Where feasible, a protection mechanism that requires two keys to unlock it is more

robust and flexible than one that allows access to the presenter of only a single key.

**6. Least privilege**: Every program and every user of the system should operate using the least set of privileges

necessary to complete the job.

**7. Least common mechanism**: Minimize the amount of mechanism common to more than one user and depended

on by all users.

**8. Psychological acceptability**: It is essential that the human interface be designed for ease of use, so that users

routinely and automatically apply the protection mechanisms correctly.

**9. Work factor:** Compare the cost of circumventing the mechanism with the resources of a potential attacker.

**10. Compromise recording**: In computer systems, mechanisms that reliably record that a compromise has occurred

are used rarely, since it is difficult to guarantee discovery once security is broken.

ISM - Information Security Management System

CC - Common Criteria

TOE - Target of evaluation (product or system that is the subject of the evaluation)

PP - Protection profile (document that identifies security requirements relevant for a particular purpose)

ST - Security target - Specification explaining how security functionality is delivered by the product

EAL - Evaluation Assurance Level

EAL1 - functionally tested („low assurance“)

− Review of functional and interface specifications

− Some independent testing

EAL2 - structurally tested („minimal serious level“)

− Analysis of security functions including high-level design

− Independent testing, review of developer testing

− Penetration testing with „basic“ attack potential

EAL3 - methodically tested and checked

− More testing, some development environment controls

− Site vi sit of development/manufacturing sites

EAL4 - methodically designed, tested, and reviewed

− Source code inspections

− Pentesting „Extended-basic“ attack potential

EAL5 - semiformally designed and tested

Formal model, modular design

Systematic vulnerability search, covert channel analysis

EAL6 - semiformally verified design and tested

− Structured development process

− Pentesting with „high“ attack potential

EAL7 - formally verified design and tested

− Formal presentation of functional specification

− Product or system design must be simple

− Independent confirmation of developer tests

Identification - claiming identity

Authentication - verifying identity

OS Hardening:

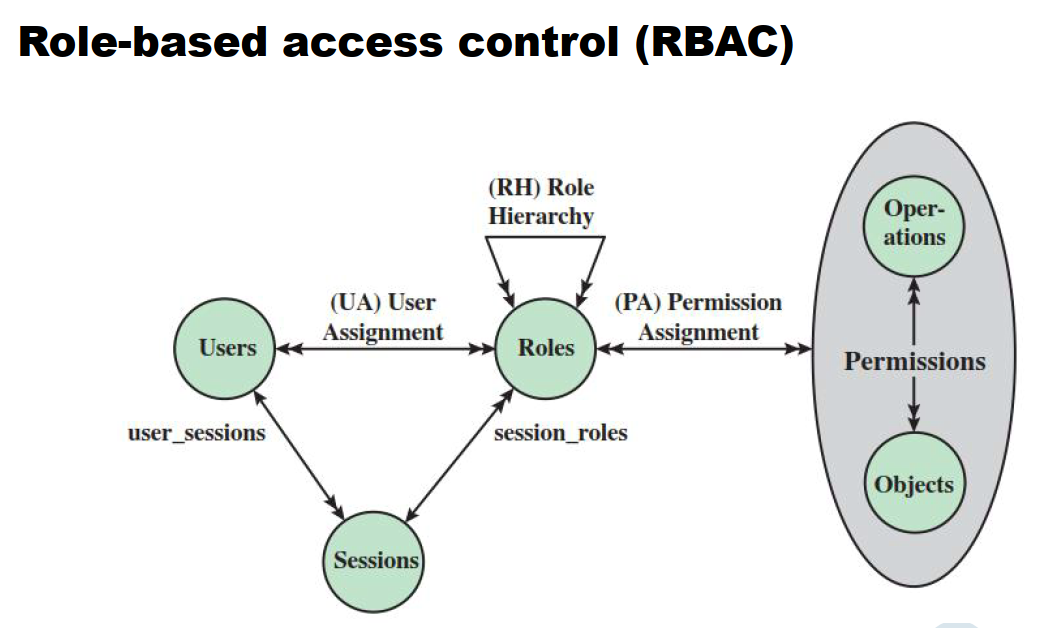
* remove unnecessary services, applications, protocols
* configure users, groups, permissions
* configure resource controls

TPM - Trusted Platform Module (tiny chip on mainboard)

* authenticated boot
* certification
* encryption

ACL/ACE - Access control List/Entry (ACE on object)

Capabilities - Like ACE but on Subject



Umask - rights to be withheld when creating default file

Bsp: umask 017 -> 760 (all for owner, rw for group, none for others)

Unix permissions: Komplexe Regeln (verschiedene Gruppenzugehörigkeiten) nötig, Implementierung schwierig

Aber: einfach zu verstehen, Zugriffsrechte können leicht geprüft werden

Malware detection:

1. Gen: Signature (known malware)

2. Gen: Heuristic rules (code sequenze that is often found in malware)

3. Gen: Active Trap (identify malware by actions)

STRIDE

Spoofing (authentication)

Tampering (integrity)

Repudiation of actions (accountability)

Information disclosure (confidentiality)

Denial of Service (availability)

Elevation of privilege (authorization)

MS SDL

Training-Requirements-Design-Implementation-Verification-Release-Response

ISO 27002

Access control, Communication security, Cryptography, Operations security, supplier relationships, physical/environmental security, HR security

How to read SecurityTarget

TOE overbiew and description

Iobjectives for operational environment

Conformance claims