# Allgemeines

## Risiko

Das Grenzrisiko ist das größte noch vertretbare Risiko eines bestimmten technischen Vorganges oder Zustandes.

Risiko = Schaden * Eintrittswahrscheinlichkeit

## Sicherheitsziele

- Vertraulichkeit
- Integrität
- Verfügbarkeit

-> CIA Triad“ (Confidentiality, Integrity, Availability)

Weitere Sicherheitsziele sind z.B.

- Authentizität
- Nichtabstreitbarkeit

## Schutzbedarf

Kategorisiert in z.B

- normal
- hoch
- sehr hoch

- gering
- mittel
- hoch
- sehr hoch

## Kategorien von Angriffen

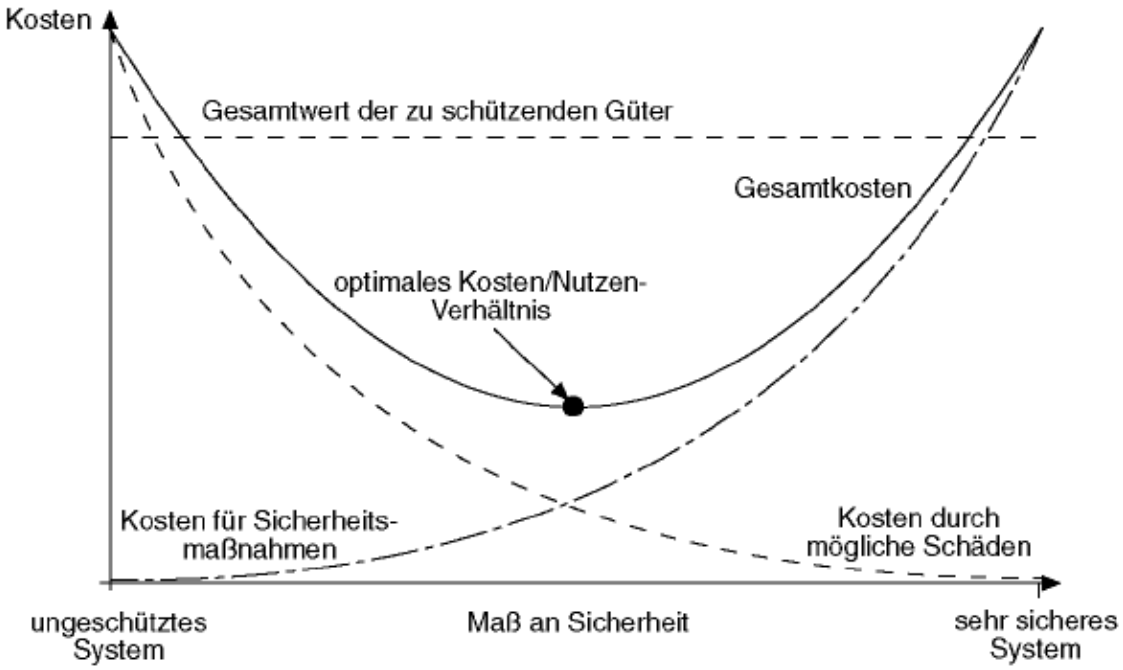
- Unberechtigter Zugriff auf Daten
  - Sniffing
  - Man in the Middle (MitM)
- Täuschung/Akzeptanz von falschen Daten
  - Spoofing
  - MitM
- Unterbrechung der Funktionalität
  - Denial of Service (DoS), Distributed Denial of Service (DDoS)
- Widerrechtliche Verwendung
  - Command Injection

## Phasen eines Angriffs

1. Sammeln von Daten über das Angriffsziel
   z.B. Verwendete Systeme, User-Kennungen,. . .
2. Ausnutzen von gefundenen Sicherheitsproblemen (Zugriff, Ausweiten der Rechte)
   z.B. SQL-Injection auf eine Datenbank

3. Aufrechterhaltung des Zugriffs
   z.B. Anlegen eines eigenen Benutzer-Accounts
4. Verwischen von Spuren
   z.B. Löschen von Logfiles

## Kosten/Nutzen von Sicherheit



# Kryptographie

Wissenschaft von der Geheimhaltung von Nachrichten.

Unvershlüsselter Text (Klartext, Plaintext) wird mittels einer Funktion, die durch einen Schlüssel (Key) parametrisiert wird, in den verschlüsselten Text (Ciphertext) umgewandelt.

Komplexes, mathematisches Problem als Basis
Einfache Berechnung eines Ergebnisses, allerdings schwierig diese Berechnung umzukehren.

## Schutzziele

- Vertraulichkeit -> Verschlüsselung
- Integrität -> Prüfsumme/Hash
- Authentizität -> Signatur

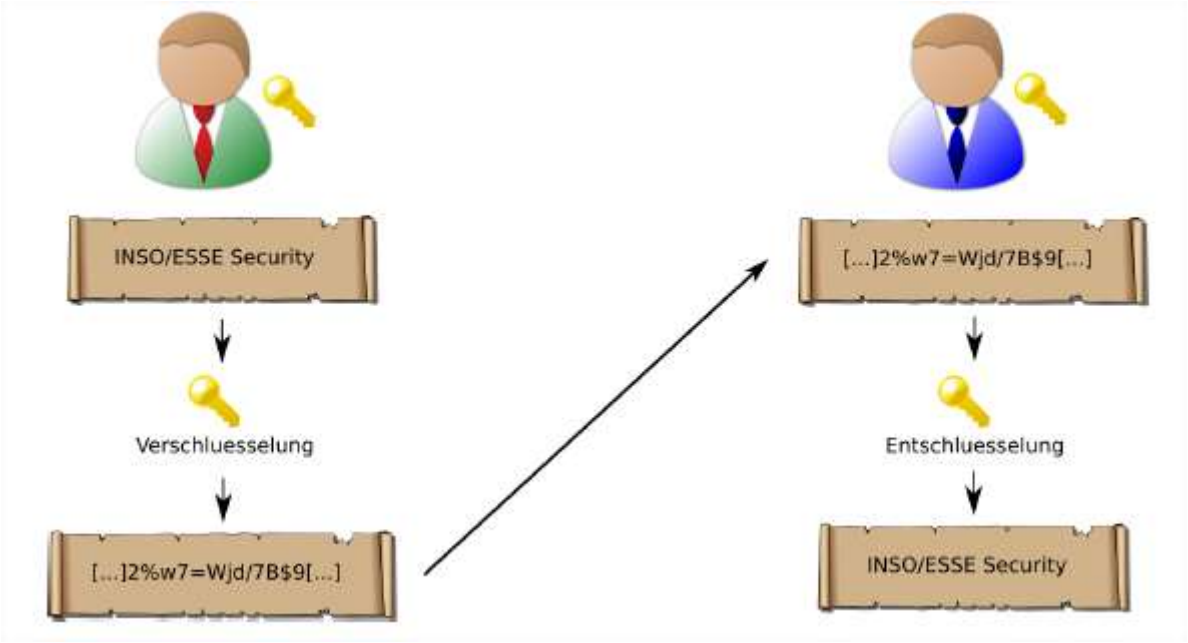
## Kerckhoffs' Prinzip

Sicherheit darf nur von Geheimhaltung des Schlüssels abhängen. Nicht von der Geheimhaltung des Algorithmus!

## Hash-Verfahren

- Hash ist ein eindeutiger Fingerprint eines Dokuments
- Einwegfunktionen

## Symmetrische Kryptographie

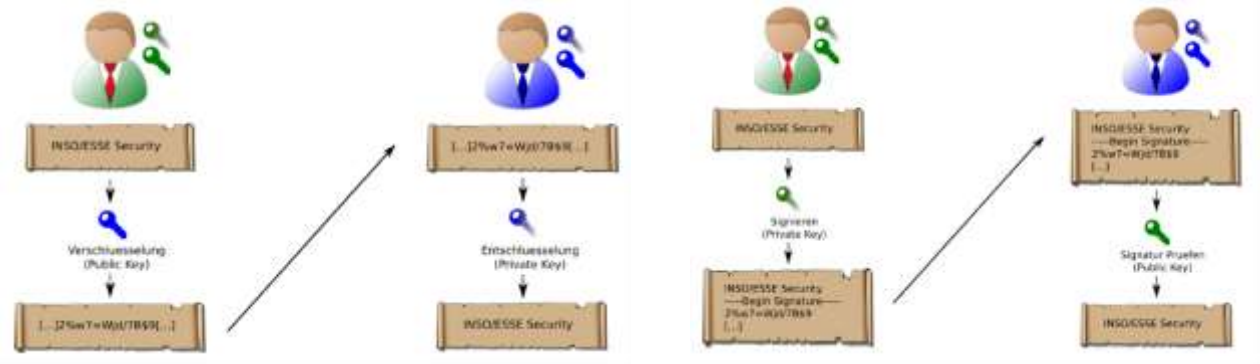



Beispiele: DES, 3DES, AES

Problem ist Schlüsseltausch/Entzug bei großer Anzahl an TeilnehmerInnen

## Asymmetrische Kryptographie

Verschlüsselung: Signatur:



Beispiele: RSA

## Hybridverschlüsselung

Kombination aus symmetrischer und asymmetrischer Verschlüsselung.

1. Verschlüsselung der zu verschlüsselnden **Daten** mit einem neu erstellten **symmetrischen** Schlüssel
2. Verschlüsselung des soeben erstellten symmetrischen **Schlüssels** mittels des **öffentlichen Schlüssels**
3. Asymmetrisch verschlüsselten symmetrischen Schlüssel an das symmetrisch verschlüsselte Dokument anhängen -> **Signatur**

## Probleme bei privaten/öffentlichen Schlüsseln

- Integritätsschutz für Schlüssel
- Schlüssel enthält keine Informationen zum Besitzer/zur Besitzerin
- Übergreifende Festlegungen für die Verwendung der PKI erforderlich

## Chipkarten

- Speicherkarten:
  - Keine Sicherheitsmerkmale für eine Zugriffskontrolle vorhanden
- Prozessorkarten:
  - Betriebssystem der Karte regelt Zugriff auf Daten
  - Möglichkeit Zugriff über Personal Identification Number (PIN) zu regeln
  - Anwendung z.B. mit kryptographischen Schlüsseln
  - Gut: Privater Schlüssel verlässt die Chipkarte nicht

Kontaktbehaftet vs. Kontaktlos

**Angriffe:**

- Side Channel Attacken (Stromverbrauch, Rechenzeit,. . . )
- Physikalisch (Temperatur, Spannung, . . . )
- Man in the Middle (z.B. über Kartenterminal)
- Social Engineering (z.B. Diebstahl, Ausspionieren der PIN)

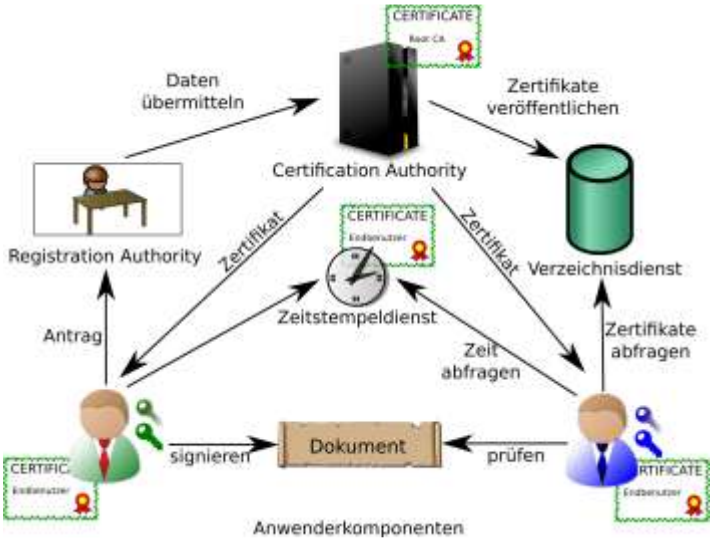
**Sicherheitsklassen von Kartenleser:**

- Sicherheitsklasse 1: Keine besonderen Sicherheitsmerkmale
- Sicherheitsklasse 2: Gerät enthält PIN-Pad
- Sicherheitsklasse 3: PIN-Pad und Display

- Sicherheitsklasse 4: PIN-Pad, Display und eigenes Sicherheitsmodul zur Authentisierung des Geräts

## Public Key Infrastructure (PKI)

### Bestandteile




- Registration Authority (RA)
- Certification Authority (CA)
- Verzeichnisdienst (DIR)
- Time Stamping Authority (TSA)
- Sperrinformationen (CRL/OCSP)
- Anwenderkomponenten
- Zertifikate
- Organisatorische Festlegungen

### Registration Authority (RA)

Registrierung neuer AnwenderInnen

- Überprüfung der Identität
- Weiterleitung an CA zur Ausstellung von Zertiffkaten für die PKI

Sperre von Zertiffkaten/BenutzerInnen

### Certiffication Authority (CA)

Aufgaben:

- Ausstellung und Verwaltung der Zertifikate
- Schlüsselerzeugung für EndbenutzerInnen
- Personalisierung von Chipkarten
- Versand der Informationen an EndbenutzerInnen (PIN-Brief, Zustellung Chipkarte,. . . )

EndbenutzerInnen-Zertikate werden durch CA-Zertifikat signiert

Erstellung/Signatur von Sperrinformationen

Besondere Anforderungen für sichere Betriebsumgebung

Speicherung und Zugriff auf private Schlüssel!

Kompromittierung des privaten Schlüssels einer CA!
-> alle ausgestellten Zertifikate und damit signierten Dokumente unsicher

### Verzeichnisdienst (DIR)

Zugriff auf öffentliche Zertifikate

Beliebige Schnittstellen möglich:

- Lightweight Directory Access Protocol (LDAP)
- Online Certificate Status Protocol (OCSP)

Zugriff auf Zertifikat muss eindeutig sein

### Sperrinformationen

Vertraulichkeit des privaten Schlüssels erforderlich

Dem Schlüssel selbst ist nicht anzusehen, ob er gesperrt ist

Anwendung muss Sperrstatus eines empfangenen Zertikats prüfen

Protokolle zum Verteilen von Sperrinformationen:

- Certificate Revocation List (CRL)
- Online Certificate Status Protocol (OCSP)

### Zeitstempeldienste (TSA)

Service stellt signierte Zeitstempel aus

Integrität der zeitlichen Information kann durch Signatur des TSA sichergestellt werden

## Zertifikate

Zertifikate als Entität für die Vertrauensstellung

Zuordenbarkeit eines PublicKeys zu einer Identität

Bestätigung wird durch Zertifizierungsstelle (CA) durchgeführt

Zertifikat ist im Rahmen der PKI öffentlich

Zusätzliche Angaben vorhanden

- Zeitliche Gültigkeit
- Informationen zum Aussteller (CA)
- Vorgesehener Einsatzzweck (KeyUsage und ExtendedKeyUsage)

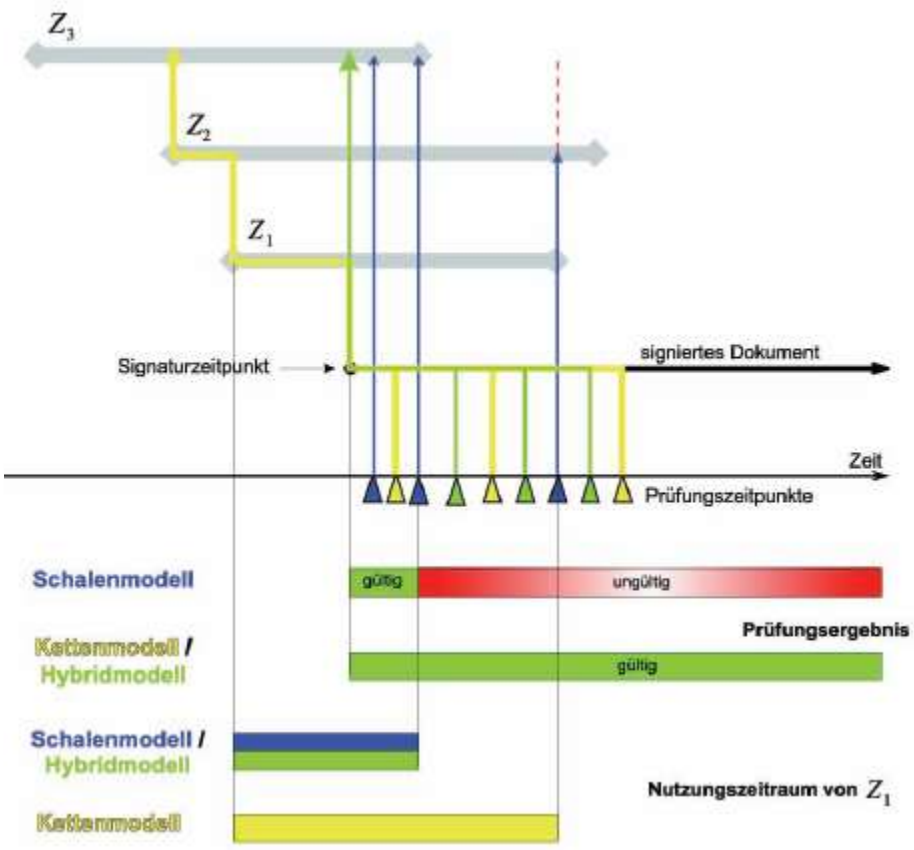
## Gültigkeit von Signaturen

Festlegung der Regeln für die Überprüfung einer Signatur

Integration des gesamten Zertifikatspfads

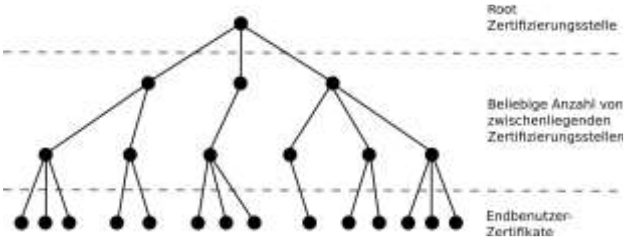
Unterschiedliche Modelle vorhanden

- Schalenmodell: Zum Zeitpunkt der Prüfung alle Zertifikate im Pfad gültig
- Kettenmodell: Zum Zeitpunkt der Erstellung der Signatur war jeweils das signierende Zertifikat gültig
- Hybridmodell: Zum Zeitpunkt der Erstellung der zur prüfenden Signatur waren alle Zertifikate im Pfad gültig




## Vertrauensmodelle

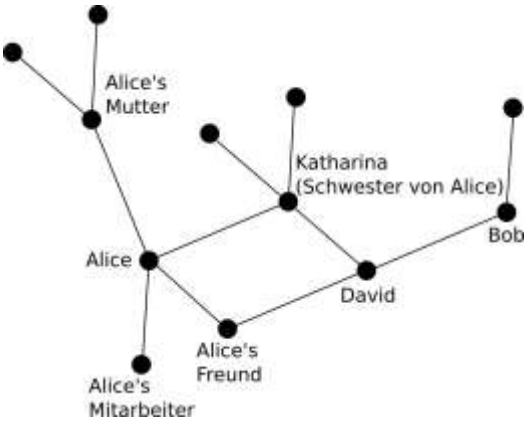
**Hierarchische Struktur**




- Hierarchischer Aufbau: Vertrauensstellung über eine Root-CA
- Zwischenliegenden Zertifizierungsstellen wird indirekt vertraut
- Verwaltung großer Infrastrukturen möglich

**Web-of-Trust**

Direkte Vertrauensstellung, Beispiel: Pretty Good Privacy (PGP)




KommunikationsteilnehmerIn muss Zugehörigkeit des Schlüssels zur Identität des Kommunikationspartners/der Kommunikationspartnerin überprüfen

## PGP / GPG

PGP … Pretty Good Privacy

GPG … GNU Privacy Guard

Inkludieren von Informationen zu BenutzerInnen

## Transport Layer Security (TLS)

TLS ist der Nachfolger von SSL (Secure Sockets Layer)

Im Netzwerkstack oberhalb von TCP

HTTPS = HTTP over TLS

Verwendung für

- Authentifizierung (einseitig oder beidseitig)
- Verschlüsselung der Kommunikation

Verschlüsselung nur zwischen zwei Komponenten auf Layer 4 möglich

Vertraulichkeit für Ende zu Ende über mehrere Stationen muss auf höheren Ebenen sichergestellt werden

## Cipher Suiten

legen kryptographische Methoden für die Kommunikation fest

Teile:

- Schlüsselaustausch (RSA, DHE, . . . )
- Authentifizierung (RSA, DSS, . . . )
- Verschlüsselung (3DES, AES, . . . )
- Hashfunktion (MD5, SHA)

## Kommunikationsablauf

# Netzwerksicherheit

## Beispiele für Attacken / Bedrohungen

- Vertraulichkeit
  - Google Hacking
  - Covert Channels
  - Sniffing
- Integrität, Authentizität, Nichtabstreitbarkeit
  - Spoofing
- Verfügbarkeit
  - Session Hijacking
  - Denial of Service (DoS)
  - Distributed Denial of Service (DDoS)

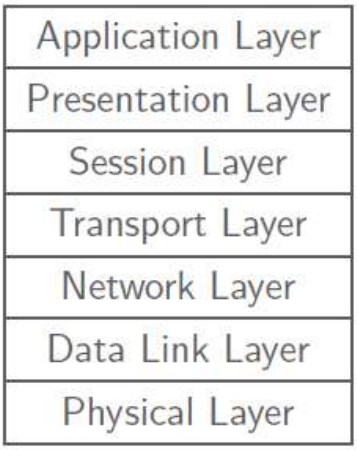
## ISO/OSI-Modell

ISO – International Organization for Standardization

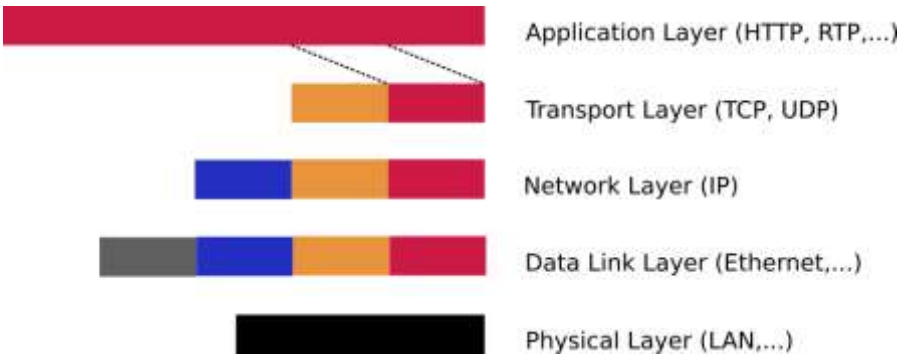
OSI – Open Systems Interconnection

**Schichtenmodell**

- Reduzierung der Komplexität der Abhängigkeiten
- Trennung der Aufgaben in einzelnen Schichten
- Schichten weitgehend unabhängig voneinander
- Genau definierte Schnittstellen zwischen den Schichten
- Höhere Schichten greifen auf Funktionen niedrigerer Schichten zu




## TCP/IP – Schichtenmodell




## ARP – Address Resolution Protocol

Basis RFC 826

Herausforderung logische Adressen/Hardware Adressen

Umwandlung von IP-Adressen in Hardware Adressen (z.B. Ethernet MAC-Adressen)

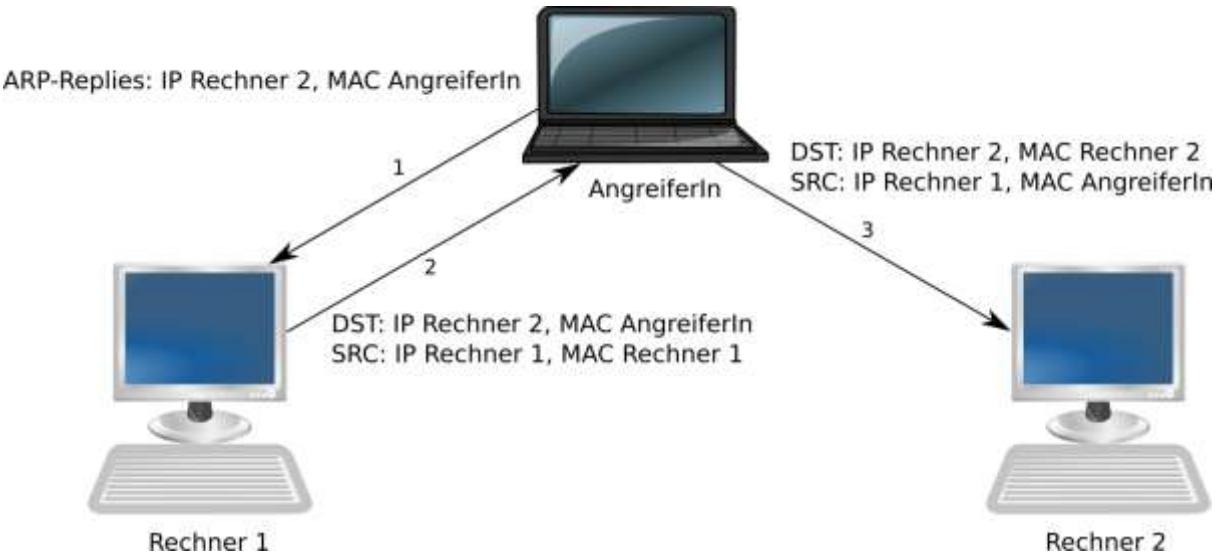
ARP Cache

Speicherung von eigenen und fremden Anfragen

ARP Poisoning

- Black Hole
- Man in the Middle

Spoofing von MAC-Adressen leicht möglich!




## ICMP – Internet Control Message Protocol

Basis RFC 792

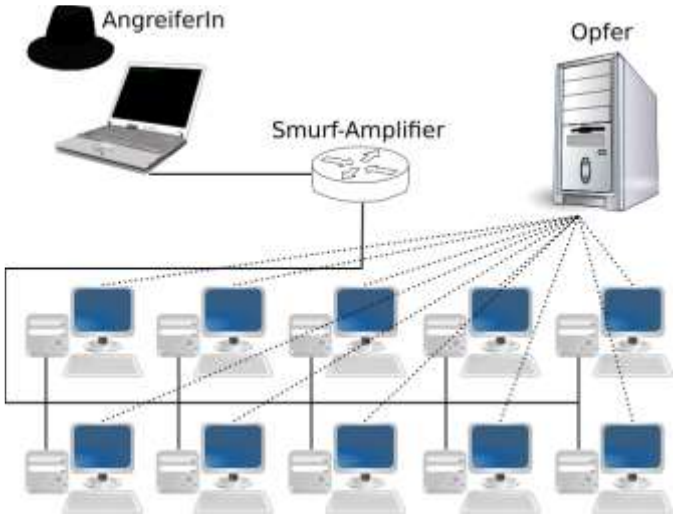
Fehlermeldungen

- Port Unreachable
- Host Unreachable
- Time Exceeded

Diverse andere Informationen

- Uhrzeit
- Echo Request/Echo Reply

## ICMP – Smurf-Attacke




## TCP – Transmission Control Protocol

Basis RFC 793

verbindungsorientiert (Three-Way-Handshake), verlässlich (Timeout, Retransmission)

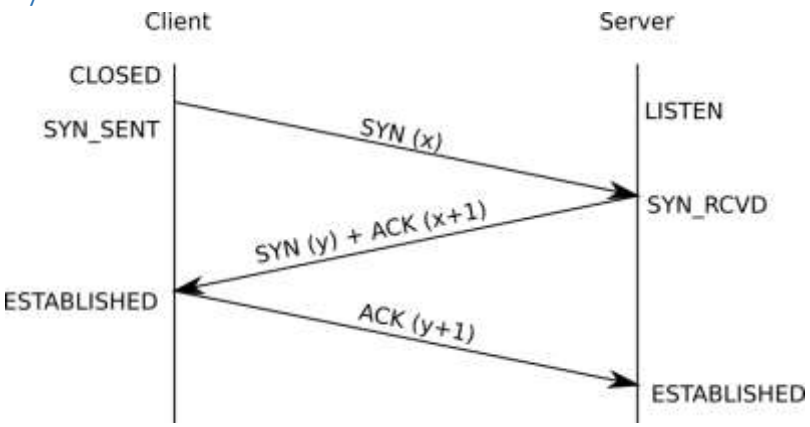
Flow Control

Exemplarische Zustände einer TCP Verbindung: LISTEN, ESTABLISHED, CLOSED

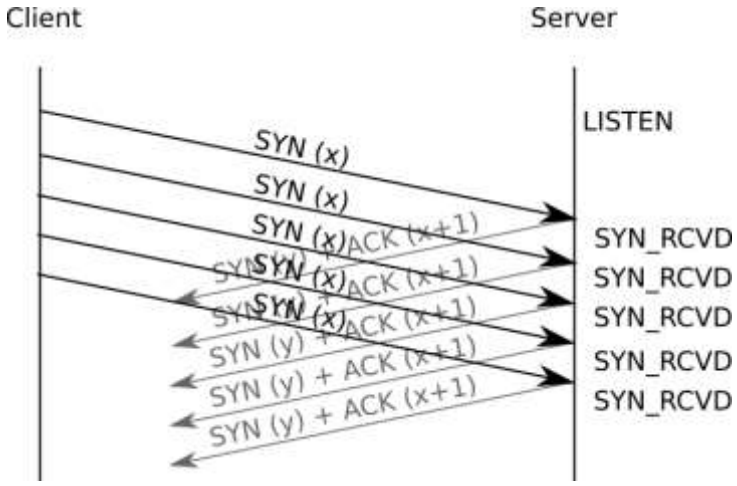
keine Broadcast Empfänger möglich – Gegensatz zu UDP

**Land-Attacke** (Source Address == Destination Address, Source Port == Destination Port)

## TCP – Three-Way-Handshake



## TCP – SYN Flooding




## Vorbereitungen für Angriffe auf Computersysteme

Nicht für jeden Angriff gleich!

Für Angriffe auf konkrete Systeme oft

- Host Scan
- Port Scan
- OS Detection
- Vulnerability Scan

Scanning

- oft auffällig
- daher tw. „Slow Scan“, „Stealth Scan“
- Ablenkung

## WLAN

Potenziell unsichere Netze

AngreiferInnen haben Zugriff auf das Übertragunsmedium

Hotels, Flughäafen, Cafes, Nachbarn, Firmen,. . .

Angriffe sind z.B.

- Sniffing
- Angreifer spooft einen AP, Client
- Denial of Service

## Lösungsansätze

Absicherung der Clients/Server

Separation von Netzwerksegmenten (physisch, logisch)

Sicherung des übertragungswegs (Internet, WLAN,. . . ) z.B via Virtual Private Network (VPN), SSL/TLS,. . .

Sicherung des Zugangs zum Netzwerk (z.B. Firewalls)

Sicherung der Nutzdaten (Verschl¨usselung von e-mails,. . . )

Zusätzliche Maßnahmen wie

- Intrusion Detection Systems
- Intrusion Prevention Systems
- Honeypots, -nets, -clients

## Firewalls

Ziel: Unterbindung von unerlaubten Zugriffen

Unterschiedliche Arten von Firewalls

- Paketfilter
- Stateful Inspection
- Proxy Firewall

Positionierung von Firewalls i.A. an der Grenze zwischen zwei Netzwerkzonen (”Zonenmodell“, DMZ)

## Intrusion (Detection/Prevention) Systeme

### Intrusion Detection System (IDS)

Erkennung von Angriffen

- Signaturen
- Anomalie-Erkennung

Senden eines Alarms bei erkanntem Angriff

### Intrusion Prevention System (IPS)

Zusätzlich zur Erkennung: Automatische Ergreifung von Maßnahmen
z.B. Aktivierung einer Firewall-Regel

## Honeypot und Honeynet

Honeypots sind eine ergänzende Sicherheitsmaßnahme

”A honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource.“ (Spitzner)

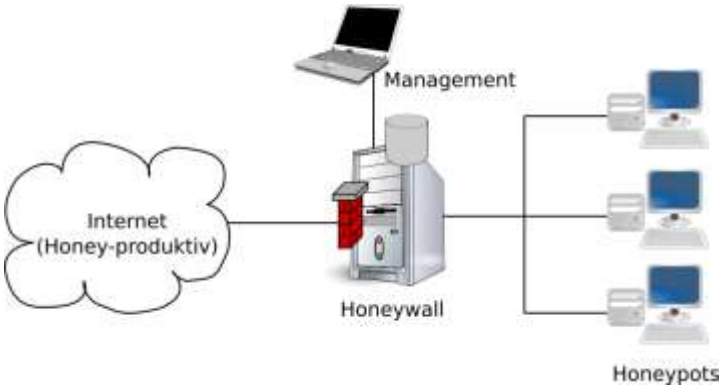
Kein Produktionssystem

Kein legitimer Zugriff

Leichtere Analyse von auftretendem Traffic

Honeynet ist ein Netzwerk von Honeypots

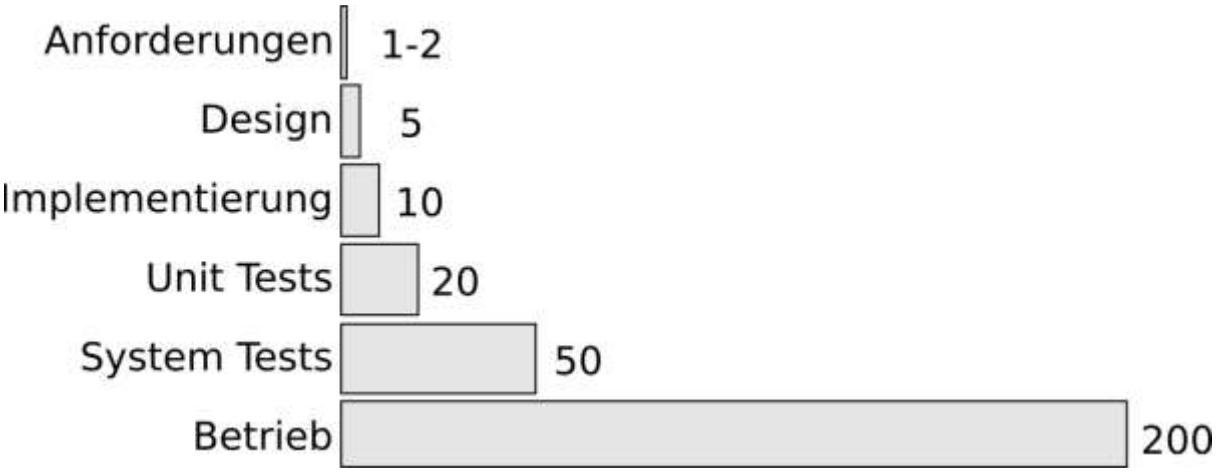
So etwas wie eine Firewall bzw. ein Gateway (Honeywall)




Verbund von Honeypots mit diversen Diensten

# Sicherheit in der Softwareentwicklung

## Grundlagen




Vorgehensmodell mit Berücksichtigung von Sicherheit

Definition von Aktivitäten und Verantwortlichkeiten (z.B. Bedrohungsanalysen, Schulungen, . . . )

Integration in vorhandene Vorgehensmodelle erforderlich

Vertreter:

- Comprehensive Lightweight Application Security Process (CLASP)
- Microsoft Security Development Lifecycle (SDL)
- Software Assurance Maturity Model (SAMM)

## Sicherheit in der Analysephase

Analyse wird die Erhebung und Aufbereitung der Sicherheitsanforderungen für das zu entwickelnde System bezeichnet.

- Funktionale und nicht-funktionale Sicherheitsanforderungen
- Unterschiedliche/widersprüchliche Anforderungen von Stakeholdern (am System beteiligte Personen)
- Weitere Anforderungen durch Normen, Gesetze, Standards
- Bei sicherheitskritischen Systemen oft Evaluierungen:
    - Berücksichtigung der Evaluierungsanforderungen
    - Z.B. Evaluierungen anhand von Protection Profiles
- Sicherheit wird als selbstverständlich angesehen und nicht explizit von den Stakeholdern gefordert

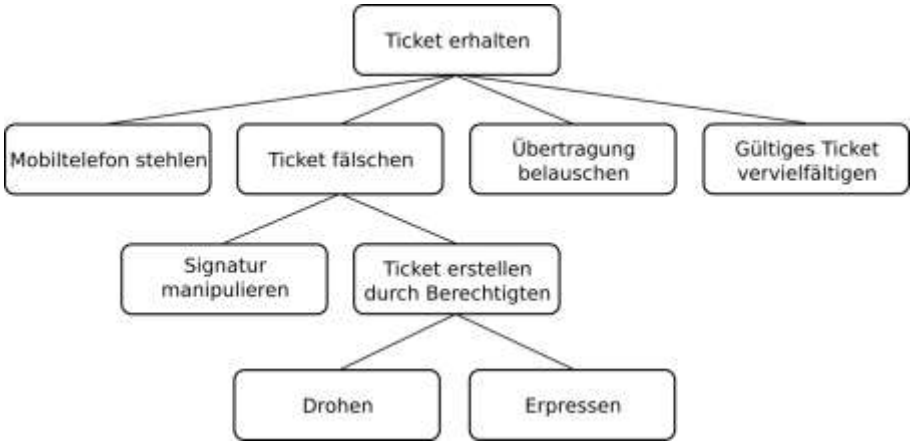
## Durchführung der Analyse

- Ermittlung der Bedrohungen und Risiken
- Vollständige Erfassung der Angriffsoberfläche
- Festlegung der Schutzziele für die verarbeiteten Daten
- Systematisches Vorgehen zur vollständigen Erfassung erforderlich
- Werkzeuge zur Durchführung vorhanden, z.B. Angriffsbäume oder Threat Modelling

## Angriffsbäume/Attack Trees

- Technik zur Dokumentation von Bedrohungen
- Identifizierung möglicher Bedrohungen mit iterativer Detaillierung

- Knoten/Kanten können zur Bewertung der Eintrittswahrscheinlichkeit einer Bedrohung verwendet werden
- Erstellung ist manuell oder automatisch möglich
- Bei größeren Systemen werden Angriffsbäume schnell unübersichtlich.




## Threat Modeling

- Methode von Microsoft zur Modellierung von Bedrohungen in der Software
- Frühzeitige Beschäftigung mit den möglichen Bedrohungen
- Schrittweise Modellierung und Detaillierung
- Bedrohungen identifizieren – Denken wie ein Angreifer/eine Angreiferin – und nicht: ”But no one would ever do that!“
- Wird derzeit von Microsoft eingesetzt (Teil des SDL – Security Development Lifecycle)

## Sicherheit in der Entwurfsphase

- Abbildung von Sicherheitsanforderungen durch den Entwurf
- Robuste Architektur bildet die Basis für ein sicheres System
- Berücksichtigung von Best Practices
    - **Sicherheitsmuster** (Security Patterns) stellen für immer wieder auftretende Herausforderungen im Sicherheitsbereich Entwurfslösungen zur Verfügung
    - **Angriffsmuster** (Attack Patterns) Beschreibung von Angriffen inklusive der Vorbedingungen welche erfüllt werden müssen. Ableitung von Sicherheitsaspekten für das eigene System möglich
    - **UML Spezielle Erweiterungen** zur Modellierung von Sicherheitsanforderungen vorhanden z.B. UMLsec

## 8 Design Principles

- Einfachheit der Schutzmechanismen (Economy of Mechanism)
    - Keep it simple
    - Design komplex -> Fehleranfällig
    - Kleine, einfache Codeteile können besser getestet werden
- Fail-Safe Defaults
    - Im Fehlerfall auf einen sicheren Zustand zurück gehen
    - Zugriff standardmäßig ablehnen
    - Input-Validierung durch Whitelists
- Vollständige Berechtigungsprüfung (Complete Mediation)
    - Sämtliche Zugriffe über einen Mediator prüfen

    - Java Security Manager prüft sämtliche Zugriffe auf potentiell gefährliche Funktionen wie zum Beispiel Dateizugriffe
- Offenes Design (Open Design)
    - Die Sicherheit eines Systems darf nicht auf der Geheimhaltung des Designs oder der Implementierung beruhen
    - **Kerckhoffs' Prinzip**: Sicherheit kryptographischer Verfahren basiert nur auf der Geheimhaltung des Schlüssels
    - Und nicht: Security by Obscurity
- 4-Augen-Prinzip (Separation of Privilege)
    - Zugriff auf Informationen durch z.B. 2 kryptographische Schl¨ussel gesch¨utzt
- Minimum an Rechten (Least Privilege)
    - Nur erforderliche Rechte verwenden/vergeben.
    - Beispiel: Kompromittierung eines Accounts mit niedrigen Rechten ermöglicht keinen Zugriff auf weitere Systemteile
- Minimum an gemeinsamen Mechanismen (Least Common Mechanism)
    - Abhängigkeiten minimieren
    - Gemeinsame Variablen/Dateien vermeiden
- Psychologische Akzeptanz (Psychological Acceptability)
    - Usability
    - Einsatz von Sicherheitsmechanismen kann durch einfach bedienbare Benutzerschnittstellen begünstigt werden
    - Reduzierung der Sicherheit, wenn Sicherheitssysteme nicht gut benutzbar.

## Sicherheit in der Implementierung

- Eine sichere Softwarearchitektur kann durch Programmierfehler
- unsicher werden
    - Fehlerhafte Prüfung von Zertifikaten
- Eine Software ohne Schwachstellen durch Programmierfehler kann Schwächen durch die Architektur aufweisen
- Fehlerfreie telnet Implementierung ist durch unverschlüsselte Kommunikation angreifbar

## Sichere Programmierung

- Auflistungen der häufigsten Sicherheitsfehler, z.B.
    - OWASP: Top Ten Web Vulnerabilities
    - CWE und SANS: TOP 25 Most Dangerous Programming Errors
- Programmierrichtlinien
    - CERT Secure Coding Standards
    - Secure Coding Guidelines for the Java Programming Language
- Sicherheitskonzepte von Programmiersprachen
    - Automatische Längenprüfung und Speicherverwaltung, . . .
    - Kenntnis über vorhandene Sicherheitskonzepte
    - Korrekte Anwendung der vorhandenen Sicherheitskonzepte
- Code Review

## Penetrate and Patch

Ansatz

"Penetrate and Patch" für sichere Software **nicht** ausreichend!

- Sicherheitsfehler werden nur im Betrieb gefunden und mit Patches behoben

- Wird von einigen Unternehmen noch immer betrieben Schwachstellen, welche von AngreiferInnen nicht gemeldet werden, können auch nicht behoben werden
- Patches beheben nicht immer die eigentliche Ursache, sondern oft nur Symptome
- Zeitfenster für Angriffe bleibt vorhanden, bis alle Systeme gepatcht werden. Bei einigen Systemen werden Patches nie eingespielt.
- Aus Patches kännen AngreiferInnen Details über die Schwachstelle ableiten

### SQL-Injection

**Gegenmaßnahmen**

- Jede Eingabe muss validiert werden. Es ist nicht ausreichend nur Formularfelder zu validieren. Injection ist auch zum Beispiel durch HTTP Header-Felder möglich.
- Ausfiltern von speziellen Datenbankzeichen wie zum Beispiel einfache Hochkomma -> Blacklist Filtering
- Zu bevorzugen ist eine Prüfung der Eingaben auf gültige Zeichen -> Whitelist Filtering
- Precompiled Statements verwenden. Das Statement wird in der Datenbank mit Platzhaltern kompiliert. Spätere Manipulation des Statements durch Eingaben nicht mehr möglich.
- Minimierung der erforderlichen Rechte in der Datenbank -> Principle of Least Privilege. Dies verhindert SQL Injection nicht, verkleinert jedoch den Schaden durch einen Angriff.

### Command Injection

Wie bei SQL Injection wird eine Eingabe ohne ausreichender Validierung für einen Systemaufruf verwendet. Dabei besteht die Möglichkeit den Aufruf zu manipulieren und beliebige Programme auszuführen.

Beispiel:
Durch site.php?prod=*** file; cat /etc/passwd; cat** wird das File /etc/passwd ausgegeben.

**Gegenmaßnahmen**

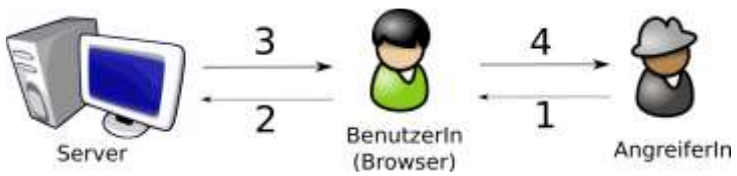
- Validierung der Eingaben!
- Verwendung der Funktionen von Programmiersprachen zum Lesen von Dateien
- Beschränkung der Zugriffsrechte auf Dateien durch Rechtevergabe oder Verwendung von chroot Umgebungen
- Minimierung der möglichen ausführbaren Dateien Berücksichtigung der System-Konfigurationen (z.B. sichere PHP Konfiguration)

### Cross-Site-Scripting (XSS)

Gefahren:

- Session-Hijacking
- Teile der Web-Seite können überschrieben werden
- Weiterleitung von Benutzern zu einer anderen Seite

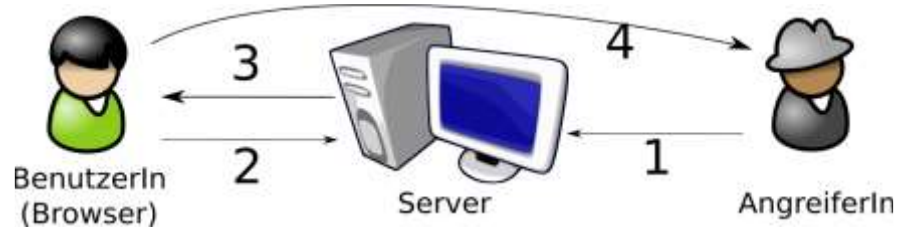
**Reflected XSS**




1. AngreiferIn sendet präparierte URL an BenutzerIn über weiteren Kanal (z.B. E-Mail)
2. BenutzerIn ruft Seite im Browser auf

3. Server liefert reguläre Seite mit zusätzlichen Inhalten aus präparierter URL zurück
4. Browser wertet Rückgabe von Server aus. Je nach Angriff sofort Übermittlung von Daten an Angreifer oder umleiten nach weiteren Eingaben des Benutzers/der Benutzerin zum Angreifer/zur Angreiferin

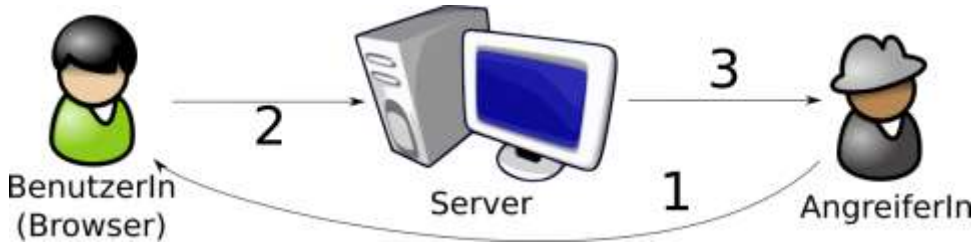
**Stored XSS**




1. AngreiferIn fügt präparierten Inhalt ein. Persistente Speicherung des Inhaltes durch Server BenutzerIn ruft Seite im Browser auf
2. Server liefert reguläre Seite mit zusätzlichen präparierten Inhalten des Angreifers/der Angreiferin zurück
3. Browser wertet Rückgabe von Server aus. Je nach Angriff sofort Übermittlung von Daten an AngreiferIn oder umleiten nach weiteren Eingaben des Benutzers/der Benutzerin zum Angreifer/zur Angreiferin

## Cross-Site-Request-Forgery (CSRF)

- CSRF ist eine Design-Schwachstelle
- Durch untergeschobene präparierte URL kann ein Angreifer/eine Angreiferin einen Benutzer/eine Benutzerin zu einer Aktion am Server missbrauchen
- Aktionen wie Logout, Einträge erstellen, Passwörter abändern, . . .
- http://server/changePassword.php?newPWD=angreiferIn
- Durch vorhandenes Authentifizierungs-Cookie beim Benutzer/bei der Benutzerin ist kein Login erforderlich
- AngreiferIn kann Änderungen über IP-Adresse des Benutzers/der Benutzerin durchführen (Verschleierung)




1. AngreiferIn sendet präparierte URL an BenutzerIn durch zusätzlichen Kanal (z.B. E-Mail)
2. BenutzerIn ruft Seite im Browser auf – Server führt Aktion direkt aus
3. (optional) Weiter Zugriff auf den Server, z.B. nach Passwortänderungen. Beispielsweise bei einer Überweisung kännte der Schritt entfallen.

**Gegenmaßnahmen**

- Hinzufügen eines **Shared Secrets** zu jeder internen URL und zu Formularen. Ein Shared Secret ist ein geheimer Token, den nur der Server und der Client kennen. Bei jeder Anfrage vom Client wird dieser Token dem Server übermittelt – und dort geprüft.
    - Bereits in einigen Web Frameworks enthalten

    - Bei unverschlüsselter Kommunikation kann ein Angreifer/eine Angreiferin diesen Token stehlen
- Verifizierung von Aktionen durch BenutzerInnen anfordern

### Buffer Overflows

Buffer-Overflows entstehen durch Programmierfehler, wenn keine oder falsche Längenüberprüfung beim Schreiben in Buffer durchgeführt wird.

- Buffer Overflow überschreibt bei zu langen Eingaben nachfolgende Speicherbereiche
- Überschreiben kann unterschiedliche Auswirkungen haben
- Absturz der Applikation mittels Exception/Segfault
- Ungewöhnliches Systemverhalten
- Keine Auswirkungen zum Beispiel wenn der zusätzlich überschriebene Speicher nicht mehr verwendet wird
- Ausführen von eingeschleusten Code
- Verwendung von low-level Programmiersprachen ohne automatisches Bounds-Checking

**Gegenmaßnahmen**

- Korrekte Input Validierung und Längenüberprüfung
- Verwendung von Programmiersprachen mit automatischer Längenüberprüfung
- Vermeidung von gefährdeten Funktionen (strcpy(), gets(), strcat(), ...)
- Verwendung von Testmethoden zum Finden von Buffer Overflows im Quellcode (Statische Codeanalyse, Fuzzing, ...)

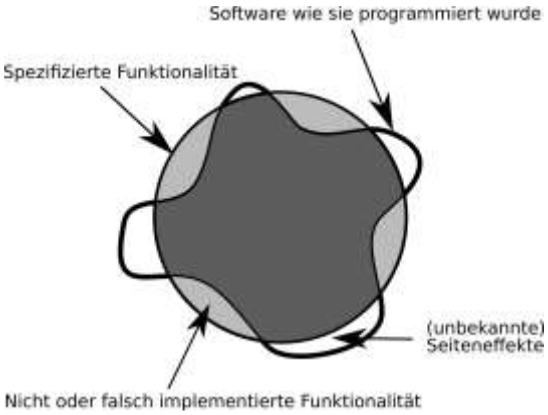
## Sicherheit in der Betriebsphase

- Es zeigt sich, ob Sicherheitsanforderungen tatsächlich erfüllt werden
- Kenntnis von Sicherheitsanforderungen für den Betrieb
- Definition und Einhaltung des Service Levels
- Bemerken von Sicherheitsvorfällen
- Mögliche Reaktionen auf Sicherheitsvorfälle
- ITIL (Information Technology Infrastructure Library)
- Organisatorische Sicherheitsanforderungen

# Testing

## Grundlagen des Testens

- Überprüfen des aktuellen Zustands mit einem geplanten Zustand
- Validierung: Erzeugen wir das richtige Produkt?
- Vergleich: Wünsche des Kunden - Anforderungen
- Verifikation: Erzeugen wir das Produkt richtig?
- Vergleich: Anforderungen - Applikation
- Test ist erfolgreich, wenn Fehler gefunden werden
- Unterscheidung nach Testziel
- Funktionalität, Sicherheit, Usability, . . .




## Unterschiede zu funktionalen Tests

- Funktionale Tests
  - Spezifikation als Grundlage
  - Erwarteter Output
  - Tests sind in sich abgeschlossen
- Sicherheitstests
  - Spezifikation nicht nutzbar als Grundlage
  - Suche nach ungewollter Funktionalität und Seiteffekte
  - Tests beeinflussen sich gegenseitig

## Security Features != Secure Features

- Sicherheitsrelevante Funktionalitäten
  - Anforderungen mit sicherheitskritischen Aspekten
  - Funktionale Fehler haben Auswirkung auf Sicherheit
  - Durchführen von funktionalen Tests
  - Beispiel: Authentifizierung, Public Key Infrastrukturen, . . .
- Sichere Funktionen
  - Sicherheitsfehler können in jeder Funktionalität auftreten
  - Testen der Robustheit gegen Angriffe
  - Beispiel: Fehlende Input Validierung
  - Wird oft verwechselt -> klare Definition und Abgrenzung erforderlich

## Sicherheitstests

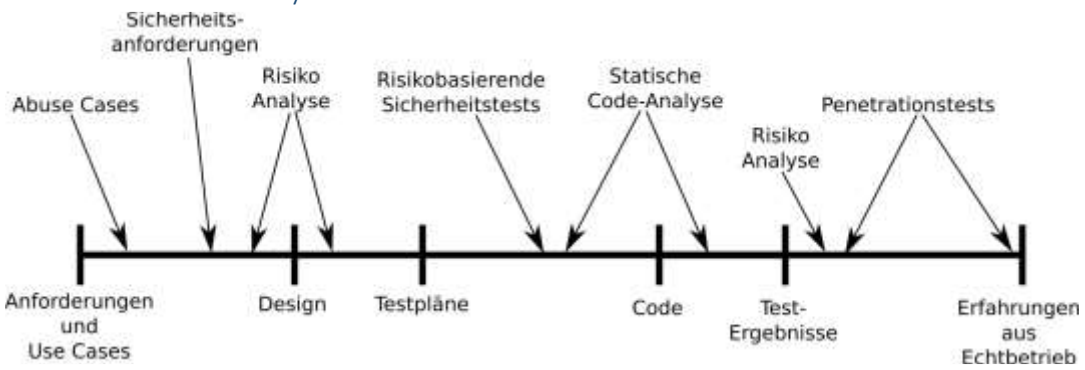
- Studie Carnegie-Mellon-Universität: ”tausend Zeilen Programm ca. fünf bis fünfzehn Bugs – nach dem Testen“
  - Diese bleiben oft unentdeckt
  - Haben keine Auswirkung auf die restliche Funktionalität
  - Stellen mögliche Sicherheitsfehler dar
  - Gezielte Suche nach diesen Fehlern erforderlich
- Test ist eine Momentaufnahme und zeitlich beschränkt – Angreifer hat beliebig Zeit
- Testen alleine nicht ausreichend für ein sicheres System
- Testen ist ein Teil zur Erstellung eines sicheren Systems

## Wer kann Sicherheitstests durchführen?

- Entwickler/Tester

  - Fehlendes Wissen über Sicherheit und aktuelle Schwachstellen -> Schwachstellen werden leicht übersehen
  - Augenmerk auf Funktionalität und testen dieser
  - Verwendung des Systems aus Benutzersicht -> Funktionen
- Sicherheitsexperte
  - Fehlendes Wissen über Domäne und die Implementierungsdetails
  - Wissen über Schwachstellen und aktuelle Entwicklungen im Sicherheitsbereich wie zum Beispiel Sicherheits-Testtools
  - Verwendung des Systems aus Angreifersicht -> Schwachstellen

### Sicherheitstests im Lebenszyklus




## Einteilung von Testtechniken

### White-Box-Tests

- Wissen über Systemdetails wird für Tests herangezogen
- Verwenden von Source Code, Dokumentationen, Konfigurationsdetails, ...
- Ableiten von Ein- und Ausgabewerten
- Testtechniken Statische Analyse, Code Reviews, ...
- Zugriff auf Informationen nicht immer möglich. Z.B. externe Programmbibliotheken

### Black-Box-Tests

- Details der Umsetzung werden für Tests nicht herangezogen
- Übergeben der Eingabedaten und Betrachtung der zurückgelieferten Ausgabe
- Vergleich der Ausgabe mit spezifizierten/erwarteten Ausgabe
- Erforderlich wenn kein Zugriff auf die Implementierung vorhanden ist
- Alle Datenvarianten nicht möglich
- Äquivalenzklassen: z.B. Verwendung eines Vertreters aus Menge der positiven und negativen Integer
- Grenzwertanalyse: z.B. Vertreter aus Grenzbereichen (MAX Integer und MIN Integer)

### Gray-Box-Testing

- Kombination von White-Box- und Black-Box-Testing
- Bevorzugendes Verfahren für die Testdurchführung
- Durch Informationen können Tests optimiert werden
- Ermittlung der Testabdeckung erleichtert
- Datenvarianten anhand interner Details identifizieren (White-Box-Tests)
- Tests mit ermittelten Datenvarianten durchführen (Black-Box-Tests)

# Statische Codeanalyse

## Secure Code Review

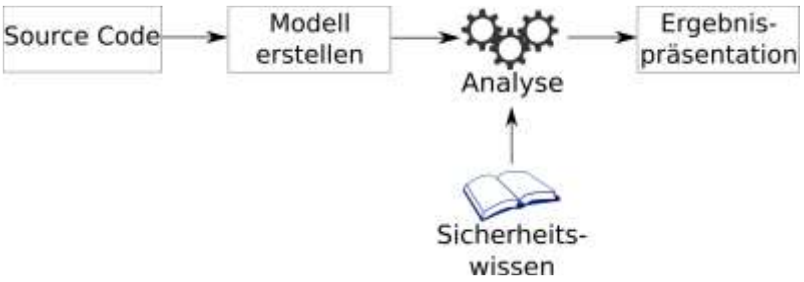
- Durchführung kostenintensiv aber sehr effektiv
- Top-Down Ansatz
  - Ausgangspunkt: Wissen über Schwachstellen
  - Erforderliches Detailwissen über Code gering
- Bottom-Up Ansatz
  - Ausgangspunkt: Wissen über Code
  - Höherer Aufwand als beim Top-Down Ansatz
- Walkthrough: Testfälle am Papier durchführen
- Inspection: Überprüfung anhand von Checklisten/Coding Standards
- Einfaches Design/Implementierung vereinfacht ein Review

## Statische Codeanalyse

- Untersuchung des Codes ohne Ausführung
  - Lauffähigkeit des Systems nicht erforderlich
  - Keine spezielle Testumgebung
- Zusätzlich zur Unterstützung von Secure Code Reviews verwendbar
- Unterschiedliche Algorithmen vorhanden
- Problem ist hohe false-positiv Rate
- Statische Codeanalyse wird auch bei Compilern und in IDEs verwendet
  - **Typprüfung** (Type Check), z.B. korrekte Verwendung von Typen
  - **Stilprüfung** (Style Check), z.B. Verwendung von Codierrichtlinien
  - **Fehlersuche** (Bug Finding), z.B. Ermittlung der Verwendung nicht initialisierter Variablen

### Prozess

- Format der Parameter und angewendete Techniken variieren
- Durchführung anhand Source Code, Byte Code oder Binary möglich
- Sicherheitswissen beschreibt Regeln
  - Eingabequellen
  - Fehleranfällige Funktionen wie strcpy()




### Varianten

- Signaturbasiert
  - Suchen nach definierten Mustern wie sprintf, strcpy, ...
  - Tools: Flawfinder, RATS, ITS4
- Kontrollfluss-Analyse

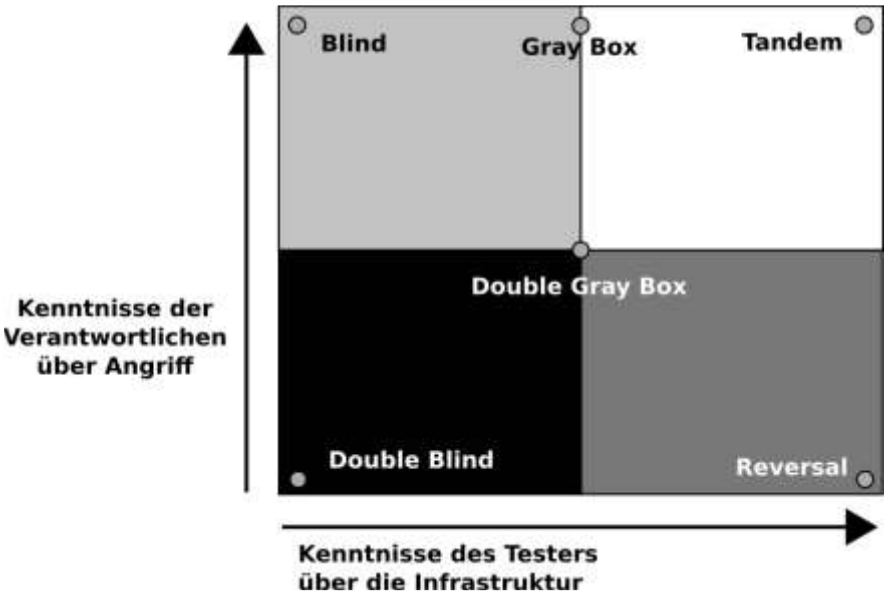
  - Kontrollfluss: Ausführungsreihenfolge der einzelnen Instruktionen
  - Intra-Prozedural vs. Inter-Prozedural (Call Graph)
- Datenfluss-Analyse
  - Zusammenhangs Erzeuger und Verbraucher von Daten
  - Vertrauenswürdigkeit der Quellen (Tainted)

## Penetration Testing

- Sicherheitsüberprüfung
  - Schwachstellen finden
  - Ausnutzbarkeit der Schwachstellen zeigen -> Beweis für Kunden/Management
  - Durchführung im Betrieb
- Überprüfung unterschiedlicher Ebenen des Systems
  - Überprüfung organisatorischer Sicherheit durch Social Engineering
- Spät im Lebenszyklus eines Systems -> höhere Kosten bei der Behebung
- Berücksichtigung des gesamten Systems (Betriebssystem, Datenbanken, Konfigurationen, ...)
- Teilweise automatisierbar
  - Vulnerability Scanner
  - Exploit Frameworks: z.B. Metasploit
- Wichtig: Rechtliche Aspekte beachten!
  - Abstürze oder Schäden am System unter Test
  - Gesetzliche Bedingungen -> Hacker Paragraph
  - Ist Auftraggeber berechtigt?

## Testabdeckung

- Ermittlung der Testabdeckung schwierig
  - Laufende Teilsysteme
  - Ports, Applikationen
- Einige methodische Vorgehensweisen zur Testdurchführung -> höhere Abdeckung möglich
  - BSI Durchführungskonzept für Penetrationstests
  - OSSTMM – Open Source Security Methodology Manual
  - Guideline on Network Security Testing (NIST)
  - OWASP Testing Project



### Prozess

- Vorbereitung u. Organisation: Zielvereinbarung, Vertrag
- Informationsbeschaffung: WWW, Foren, Stellenausschreibungen, Netzwerkscans, ...
- Testdurchführung: Durchführung von Angriffen
- Dokumentation: Protokollierung der Testdurchführung (z.B.
- Netzwerkmitschnitt), Testberichte, Präsentationen

### Vulnerability Scanning

- Suchen nach Schwachstellen ohne weitere Ausnutzung dieser
    - Ermitteln der Versionen
    - Bekannte Schwachstellen
    - Fehlkonfigurationen
    - Bekannte Dateien und Ordner
- Durchführung für jedes System wichtig
- Regelmäßige Wiederholung erforderlich
- Erster Schritt möglicher Angreifer -> schnell mit wenig Aufwand
- Automatisierbar, Tools für die Durchführung vorhanden

### Fuzzing

- Random Testing
- Automatisiertes Ausführen mit zufällig generierten oder vordefinierten Werten
- Testen der Datenfelder, Struktur und Ablauf
- Fuzzer für unterschiedliche Eingabeformat/Protokolle vorhanden
    - Dateien: Bilder, HTML, XML, DOC, SWF, . . .
    - Netzwerkprotokolle: TCP/IP, SIP, RTP, . . .

### Arten

- Template/Block Fuzzer
    - Für einfache Protokoll geeignet
    - Berücksichtigen der Prüfmechanismen eines Protokolls, z.B. Checksummen, Schemavalides XML, usw.
- Mutation Based Fuzzer
    - Verwendung bestehender Daten und Abwandlung dieser Daten
- Generation Based Fuzzer
    - Spezielle Implementierung pro Format/Protokoll
    - Aufwendiger bei der Erstellung
- Model Based Fuzzer
    - Umsetzung eines komplexeren Ablaufs, z.B. Handshake

### Eingabedatengenerierung

- Zufällig generierte Daten
    - Kein Wissen über das spezielle Format erforderlich
    - Optimierung vs. Vollständigkeit (256 unterschiedliche Varianten für ein Byte)
- Ableitung der Werte von typischen Angriffswerten -> Fuzz-Vektor
- XSS, SQL Injection, Buffer Overflow, ...

### Ausführung

- Erzeugung und Ausführung automatisierbar
    - Große Anzahl von Datenvarianten möglich/erforderlich. Microsoft SDL verlangt 100.000 Test-Dateien bei Datei-Fuzzer

- Fehlererkennung
  - Veränderungen eines Systems: Prozessorleistung, Fehlereinträge in Log-Dateien, . . .
  - Überwachen des Prozesses durch Debugger z.B. gdb oder OllyDbg
  - Fehler der Applikation beobachten -> segfault
- Fehlerlokalisierung
  - Welcher Testfall hat Fehler verursacht?

## Testen von Web Applikationen

- Überprüfung der Implementierung und Konfiguration des Web Servers
- Bekannte Schwachstellen der verwendeten Komponenten prüfen
- HTTP ist Stateless
  - Sessions ermöglichen Zustandsspeicherung
  - Testen auf Muster in Session ID
- Backup Dateien am Server: *.php.bak
- Nicht interpretierte Dateien: *.inc
- Standard Verzeichnisnamen: admin, include, . . .
- Implementierung der Web Applikation, z.B. OWASP Top Ten
- Unterschiedliche Encodings für Eingaben
  - < (ASCII)
  - %3c (URL Encoded)
  - < (HTML Entity)
  - < (Hex Encoding)

# Identität, Authentifizierung, Zugriffskontrolle

## Begriffe

**Identität** Wer jemand ist

**Authentisierung** (authentication) … Vorlage eines Nachweises zur Identifikation (z.B. Username/Passwort)

**Authentifizierung** (authentication) … Überprüfung eines Nachweises zur Identifikation

**Autorisierung** … Überprüfung, ob eine Person, IT-Komponente oder Anwendung zur Durchführung einer bestimmten Aktion berechtigt ist

**Zutritt** … Betreten von abgegrenzten Bereichen wie z.B. Räumen oder geschützten Arealen in einem Gelände

**Zugang** … Nutzung von IT-Systemen, System-Komponenten und Netzen

**Zugriff** … Nutzung von Informationen bzw. Daten

## Authentifizierung

### Gründe

- Nachweis der Identität (sowohl von Personen als auch Komponenten)
- Kontrolle des Zutritts, Zugangs
- Voraussetzung zur Zugriffskontrolle
- Nachweis Zurechenbarkeit/Verantwortlichkeit
- Herstellung von Vertrauen zwischen zwei KommunikationspartnerInnen

### Herausforderungen

- Valider Zugang soll einfach möglich sein
- Unberechtigte sollen zuverlässig vom Zugang abgehalten werden
- AngreiferInnen reicht u.U. ein einzelner Acount
- Authentisierungsmethoden haben unterschiedliche Vor- & Nachteile
- Kein einziger, bester Weg für Authentisierung
- Je nach Einsatzzweck geeignetes Verfahren auswählen
- Kombination von Methoden: 2-Faktor-Authentifizierung, wenn 2 Methoden miteinander kombiniert werden

## Methoden der Authentisierung

- Wissen
  - Herkömmliche textuelle und graphische Passwörter
  - PINs
- Besitz
  - Tokens
  - Chipkarten
- Biometrisches Merkmal
  - Gesicht
  - Fingerabdruck
  - Stimme
  - Tippverhalten

## Wissen

- Kennwörter wurden schon vor dem Einsatz in der Informationstechnik verwendet
  - Militär: Parole
  - Losungsworte
- Etablierte Methode zur Authentisierung
- Sicherheit abhängig von technischen und organisatorischen Faktoren
- Ausreichend groß Basis an verfügbaren Zeichen
- Verlust schwer oder erst zu spät erkennbar

### *Passwörter*

- Passwörter werden häufig verwendet
- Klein- und Großbuchstaben, Ziffern, Sonderzeichen1, möglichst lange
- graphische Passwörter (Bildfolge, Punkte auf Bildern, Gesichter, ...)
- leicht zu merken vs. schwierig zu erraten
- Anzahl der Passwörter steigt
- u.U. Wechsel in bestimmten Abständen, Passwort Historie
- u.U. Mindestlänge, Mindest-Vorkommen von einzelnen Zeichenklassen (Achtung bei 4 bzw. 6-stelligen Passwörtern, PINs)

## Regeln bei Verwendung

- Keine Default-Passwörter verwenden
- Passwörter sollen nicht aufgeschrieben werden (v.a. nicht am Monitor, Keyboard etc.)
- Passwörter sollen vom Benutzer/von der Benutzerin änderbar sein
- Anzahl der Versuche zur Passwort Eingabe beschränken (Achtung: Denial of Service)
- Prozess zur Änderung bei vergessenen Passwörtern

## Verwendung von schlechten Passwörtern

- Trotz aller Empfehlungen werden manche Passwörter häufig verwendet
- Sicherheitsbewusstsein stärken
- Beispiele für schlechte Passwörter
- Familie (eigener Vorname/Nachname, LebenspartnerInnen, Kinder, Haustiere, Geburtsdatum, ...)
- Stars (Namen von MusikerInnen, SchauspielerInnen,. . . )
- Username = Passwort
- 123456, passwort, schatz, hallo, qwertz, ab123
- Ab und zu können aber auch schlechte Passwörter ausreichen  nicht alles muss gleich gut geschützt werden

## Beispiele für Angriffe

- Raten
- Social Engineering
- Brute Force (lokal, remote)
- Wörterbuch Attacken
- Rainbow Tables
- Script Files, Trojaner, Default-Passwörter
- Passwortdateien/Passwörter in Swap-Files
- „Über die Schulter Schauen“, Wisch-Spuren auf Smartphones
- Sniffing (daher in Netzwerken z.B. Anwendung von Verschlüsselung)
- Elektromagnetische Strahlung

- Wireless Tastaturen
- Preisgabe von Information: „Username/Passwort falsch“

### Abwehrmethoden zu Angriffen

- Passwörter müssen am System sicher gespeichert werden, sie sollen nicht im Klartext gespeichert werden (Hash, Salt)
- Eigene Hash-Algorithmen mit bestimmten Eigenschaften zur Speicherung von Passwörtern (siehe Provos und Mazières bzw.
- Password Hashing Challenge), z.B. Argon2, bcrypt
- Zugriffsschutz der Passwortdatei
- Proaktive Passwort-Checker
- User-Training
- Lockout Mechanismen
- Passwort Wechsel
- Zeitliche Begrenzung der Gültigkeit der Passwörter
- Fehlanmeldungen/Letzte Anmeldung anzeigen

## Besitz

- Verwendung auch abseits von Einsatz in IT bzw. vor Einsatz in IT
  - Abzeichen
  - Dienstmarken
- Verlust besser erkennbar als bei Passwörtern
- Erzeugung von Einmal-Passwörtern
  - Passwort auf Basis von aktueller Zeit
  - Passwort auf Basis eines Counters
- Immer wieder Schutz durch PIN/Passwort (z.B. Bankomatkarte)
- Physischer Schutz erforderlich
- Vorgaben für die Verwendung von PINs
- Kosten höher als bei Passwörtern

### *Chipkarten*

- Anwendung z.B. mit kryptographischen Schlüsseln
- Gut: Privater Schlüssel verlässt die Chipkarte nicht
- U.U. geschützt durch PIN
- Gesamter Lebenszyklus relevant
- Einsatz z.B. Banken, Gesundheitswesen
- Angriffe
  - Side Channel Attacken (Stromverbrauch, Rechenzeit,. . . )
  - Physikalisch (Temperatur, Spannung,. . . )
  - Man in the Middle
  - Social Engineering

## Biometrisches Merkmal

- körperliches Merkmal
- Identifizierung von Menschen im zwischenmenschlichen Alltag
- biometrische Merkmale sind schwer/nicht zu ersetzen
- Unterscheidung: Identifikation und Validierung
- Weitergabe von biometrischen Merkmalen schwer/nicht möglich
- Fehlerrate (FAR  False Acceptance Rate, FRR  False Rejection Rate)

- Akzeptanz von biometrischer Authentisierung
- Kosten höher als bei Passwörtern oder Token
- Ausprägung biometrischer Merkmale nicht bei allen Menschen gleich
- Alternativen/Ausnahmen erforderlich

### Angriffe

- (Auch) Authentifizierung duch biometrische Merkmale nicht zu 100% sicher
- Stärken/Schwächen von biometrischer Authentisierung werden oft vernachlässigt
- Spoofing (Hochauflösende Fotos (z.B. 31C3), Modellierung von (gefundenen) Fingerabdrücken, ...)
    - Beispiel: Samsung Galaxy S5: Fingerabdrucksensor auch schon gehackt
- Replay Attacken
- Umgehen der Sensoren
- Brute Force

## Zugriffskontrolle

- Zugang ist nun kontrolliert
- Zugriff sollen nur die berechtigten Akteure erhalten
- Unberechtigter Zugriff soll verhindert oder zumindest erkannt werden
- Zugriffskontrolle bedeutet Auditierung
- Zwei unterscheidbare Fälle für Angriffe
    - AngreiferIn hat Zugriff auf Binärdaten -> Kryptographische Methoden erforderlich
    - Es existiert eine Schicht zwischen AngreiferIn und Binärdaten -> Zugriffskontrolle
- Umsetzung einer Sicherheitsstrategie (Security Policy)
- Verwendung von Zugriffskontrollmechanismen z.B. in
    - Betriebssystem
    - Datenbanken
    - Webserver
- Kernfrage der Zugriffskontrolle:
    - Wer darf auf was wie zugreifen.

**Zugriffskontrollstrategie** ist ein Regelwerk, das besagt was erlaubt/nicht erlaubt ist.

**Zugriffskontrollmodell** ist ein Formalismus, um eine Zugriffskontrollstrategie zu beschreiben.

- Verschiedene Zugriffskontrollmodelle für unterschiedliche Zugriffskontrollstrategien
- Ein Modell soll einfach, ausdruckstark, intuitiv, wartbar und umsetzbar sein.

### Begriffe

**Objekt** … passiver, zu schützender Informationsträger

**Subjekt** … aktive Elemente, die im Auftrag von AnwenderInnen Zugriffe auf Informationen ausführen

**Zugriffsoperation** … Art, um auf ein Objekt zuzugreifen (read, write, execute, ...)

**Zugriffsrecht** … Rechte für Zugriffe auf Dateien, Datenträger, Prozesse, ...

**Schutzdomäne** … Gruppierung von identischen Zugriffsrechten

**Zugriffsmonitor** setzt die Zugriffskontrollstrategie durch

## Zugriffskontrollmodell


## Grundmodell

Ob für ein Subjekt s der Zugriff der Art a auf ein Objekt o zulässig ist, ergibt sich aus einer Funktion f, die das Ergebnis des 3-Tupels (s;o;a), das wahr oder falsch sein kann, auswertet.

Dabei gilt

- s ϵ S, der Menge aller Subjekte
- o ϵ O, der Menge aller Objekte
- a ϵ A, der Menge aller möglichen Zugriffsarten

## Zugriffsmatrix

| | | Objekt | | | | |
|---|---|---|---|---|---|---|
| | | $\alpha$ | $\beta$ | $\gamma$ | $\delta$ | $\epsilon$ |
| Subjekt | $a$ | 0 | 0 | 1 | 0 | 1 |
| | $b$ | 1 | 1 | 1 | 1 | 0 |
| | $c$ | 0 | 0 | 0 | 0 | 1 |
| | $d$ | 0 | 1 | 1 | 1 | 0 |
| | $e$ | 1 | 1 | 0 | 1 | 0 |
| | $f$ | 0 | 1 | 0 | 0 | 1 |

- Zugriffsmatrix definiert die Zugriffsrechte
- Sichtweisen
  - Was darf ein Subjekt?
  - Was darf mit einem Objekt passieren?
- Umsetzung direkt als Matrix i.A. langsam, ineffizient

**Eigentümer** … Der Eigentümer hat uneingeschränkte Zugriffsrechte.

**Gruppe** … Alle innerhalb einer speziellen Gruppe haben dieselben Zugriffsrechte, z.B. Lesen eines Objekts.

**Jeder …** Allen Usern in einem System können bestimmte Zugriffsrechte gegeben werden.

## Discretionary Acess Control (DAC)

- Ziel: Der/Die BesitzerIn legt Berechtigungen auf Objekte selbst fest
- Zugriff auf Objekte allein von der Identität abhängig

- Jedes Objekt hat einen Besitzer/eine Besitzerin

- Der/Die BesitzerIn kann Rechte für die Durchführung von Operationen an andere BenutzerInnen vergeben

- Verwaltung von Zugriffsrechten aufwändig (Zugriffsrechte auf BenutzerInnen-Ebene - Wechsel von BenutzerInnen)
- Aufwändig festzustellen welche Rechte bestimmte BenutzerInnen haben

### Role-Based Acess Control (RBAC)

- Ziel: Der Zugriff auf Objekte wird durch die Rolle festgelegt

- BenutzerInnen haben Rollen
- von den Rollen hängen die Zugriffsrechte ab
- eine Rolle ist eine Sammlung von Funktionen, die für eine Arbeit gebraucht werden (z.B. LVA-LeiterIn)
- hierarchisches Rollenkonzept
- BenutzerInnen können mehrere Rollen haben, der Wechsel der Rolle ist möglich
- Rollen, die für eine Session gelten sollen, können aktiviert werden
- Rollen können andere Rollen ausschließen („Separation of Duty“)

### Mandatory Acess Control (MAC)

- Ziel: Zentrale Festlegung der (maximalen) Zugriffsrechte von BenutzerInnen
- Kontrolle des Informationsflusses
- Weitergabe von Rechten eingeschränkt möglich

- Objekte und BenutzerInnen haben Einstufungen (z.B. öffentlich, vertraulich, streng geheim)
- BenutzerInnen können lesend nur auf ein Objekt zugreifen, wenn ihre Einstufung mindestens der des Objekts entspricht
- Objekte können nur mit Einstufungen, die gleich- oder höherwertig der Einstufung des Benutzers/der Benutzerin ist, geschrieben werden
- Anwendung hauptsächlich im militärischen Bereich

# Organisatorische Sicherheit/Sicherheitsmanagement

## Technische vs. organisatorische Sicherheit

- Bestimmte Herausforderungen lassen sich besser bzw. nur durch organisatorische Maßnahmen lösen.
- Breiter Bogen unterschiedlicher Aspekte

## Beispiele für technische Sicherheitsmaßnahmen

- Firewall, Virenscanner, Verschlüsselungssysteme. . .
- Sicherheits-Patches und Systemhärtung
- Sandbox-Systeme
- Absicherung (drahtloser) Kommunikationswege (LAN, WLAN, etc.) mittels VPN/TLS
- Usernamen/Passwörter
- Elektronische Signaturen

## Organisatorische Sicherheitsaspekte

- Primär der Aufbau des IT-Sicherheits-Managements (Sicherheitsplanung)
- GF: Formulierung einer IT-Sicherheitsleitlinie (strategische Aussagen)
- Typische Aufgaben sind z.B.
    - IT-BenutzerInnen schulen und für Sicherheit sensibilisieren
    - Regelmäßige Audits
    - Alle Maßnahmen und Veränderungen dokumentieren
    - Eindeutige Definition von Verantwortlichkeiten inklusive Vertretungsregelungen sowie Kommunikationswege
- IT-Sicherheit muss vom Management, den AdministratorInnen und den IT-BenutzerInnen als fortlaufender Prozess verstanden und gelebt werden!

## Beispiele Organisatorischer Sicherheitsmaßnahmen

### Schulung von MitarbeiterInnen

- MitarbeiterInnen brauchen eine ausreichende Einführung und Schulung
- Betrifft sowohl den sicheren Umgang mit IT-Systemen, als auch Unternehmensdaten
- Aufgabe ist Etablierung eines Sicherheitsbewusstseins
- Regelmäßige Schulungen und Sicherheitssensibilisierungen
- Rechtzeitige Unterrichtung der MitarbeiterInnen über Bedrohungen

### Sicherung von Daten

- Verlust von Daten kann zu Einschränkungen im Betrieb und wirtschaftlichen Schäden führen
- Daher: aktive Datensicherung (Backup)
- Durchführung regelmäßiger Backups nach einem Backup-Plan
- Bei wichtigen Daten evtl. auch Sicherungskopien außerhalb des Unternehmens
- Regelmäßige Kontrolle/Test der Wiederherstellung gesicherter Daten (Restore)

### Sicheres Löschen von Daten

- Besonderer Schutz für sensible Daten
- Löschen nach Ende von Aufbewahrungsfristen, Hardwaretausch etc.
- Häufiger Fall: ausrangierte HDDs enthalten interne Zahlen, KundInneninformationen, Zugangsdaten, ...
- Einfaches Löschen nicht ausreichend - Daten können oft wiederhergestellt werden

- Verwendung von spezieller Soft- und/oder Hardware zur Zerstörung, z.B. thermisch, aktives Überschreiben
- Auslagerung an spezialisierte Unternehmen
- Durchgehende Anwendung von Verschlüsselung
- Berücksichtigung von Ausdrucken/Papier-Unterlagen
- -> Prozesse

### Updates

- Ständige Weiterentwicklung von Schadprogrammen, tw. mit sehr hohem Gefährdungsgrad
- Aufkommen neuer Technologien, die Angriffe ermöglichen oder die Angriffswahrscheinlichkeit erhöhen
- Laufende Updates der Risiko- und Bedrohungsanalysen
- Zeitnahe Installation von Sicherheitsupdates
- Aktueller Virenschutz
- Wartung eingesetzter Soft- und Hardware

### Dokumentation

- Stetig aktuelle Dokumentation bildet oft Grundlage für Aufgaben
- Inventar von Soft- und Hardware
- Konfigurationen der (wichtigsten) Systeme
- Servicenummern der (wichtigsten) IT-Lieferanten und IT-Dienstleister
- Vertragliche Vereinbarungen, SLAs
- Software-Artefakte
- Prozesse (Onboarding, Offboarding, …)
- Berechtigungen
- Möglichst hoher Automatisierungsgrad

### Überprüfung der Wirksamkeit von (organisatorischen) Sicherheitsmaßnahmen

- Organisatorische Sicherheitsmaßnahmen und Vorgaben sind regelmäßig auf Einhaltung und Wirksamkeit zu überprüfen
- -> Dokumentation
- Aktualität von Software/IT-Systemen
- Einhaltung von Richtlinien seitens der MitarbeiterInnen
- Aktualität der Notfallpläne und Dokumentationen
- Test durch externe Dienstleister, z.B. Penetrationstests
- Besteht das Erfordernis, müssen entsprechende Anpassungen vorgenommen werden

## Sicherheitsmanagementprozess

### Festlegung der IT-Sicherheitsverantwortlichen-Rollen

- Festlegung IT-Sicherheitsverantwortlicher (und Vertretung) durch Management
- Klare Regelung von Verantwortlichkeiten und Zuständigkeiten
- Ansprechpartner bei auftretenden Sicherheitsproblemen
- Typische Rollen sind beispielsweise
    - Information Security Manager (Zentrale Koordination und Ansprechperson für das Thema Sicherheit)
    - Information Risk Manager (Erkennung, Bewertung, Management von Risiken)
    - IT-Sicherheitsbeauftragter (Umsetzung von Sicherheitsmaßnahmen)
    - Datenschutzbeauftragter (Schutz personenbezogener Daten)

## Security Policies

„Die Sicherheitsrichtlinie (engl. security policy) eines Systems oder einer organisatorischen Einheit legt die Menge von technischen und organisatorischen Regeln, Verhaltensrichtlinien, Verantwortlichkeiten und Rollen sowie Maßnahmen fest, um die angestrebten Schutzziele zu erreichen.“

„Jede Organisation ist frei, die für ihren geschäftlichen Kontext als relevant erachteten Ziele individuell festzulegen. Diese Ziele sind zu dokumentieren, was meist im Überblick in einer Security Policy erfolgt - im Deutschen meist als Sicherheitsleitlinie bezeichnet.“

### Inhalte von (Security) Policies

- Angestrebte IT-Sicherheitsziele
- Verfolgte IT-Sicherheitsstrategie
- Anspruch und Aussage zugleich, dass das IT-Sicherheitsniveau auf allen Ebenen der Organisation erreicht werden soll
- Die Verantwortung für die Security Policy unterliegt der Unternehmens-/Behörden/...-leitung
- Eine Policy muss von der GF verabschiedet und vorgelebt werden

### Anforderungen an Policies

- Vollständigkeit der Policies
- Bekanntheit der Policies
- Verständnis und aktives Leben der Policies
- Laufende Überprüfung der Einhaltung von Policies

### Policies - Sicherheitsrichtlinien für MitarbeiterInnen

- Eines der größten Risiken ist Fehlverhalten von MitarbeiterInnen
- Policies enthalten Vorgaben, um diesem Risiko entgegenzuwirken, bzw. dieses zu mindern
- Beschreibung wie sicher und korrekt mit IT und deren Komponenten umgegangen wird
- Fortwährende Anpassung und Ergänzung aufgrund aktueller Entwicklungen wichtig (z.B. Smartphones, BYOD)

## PDCA-Modell


### Planung (Plan)

- Planung und Installation eines Sicherheitsmanagements anhand Risiko-, Sicherheits- und Unternehmenspolitik, der betrieblichen Anforderungen sowie der gesetzlichen Vorgaben
- Definition Vorgehensmodell zu Sicherheits-, Kontinuitäts und Risikopolitik
- Erhebung der Sicherheits- und Kontinuitätsanforderungen
- Erstellung von Sicherheits- und Kontinuitätsrichtlinien
- Entwicklung von Sicherheitskonzepten
- Durchführung von Maßnahmen

### Betreiben (Do)

- Einführung, Betrieb und Monitoring des Sicherheitsmanagements
- Schulung und Sensibilisierung von MitarbeiterInnen
- Betrieb unter Berücksichtigung der eingeführten Sicherheits- und Kontinuitätskonzepte
- Messung und Überwachung den Anforderungen entsprechend (Erkennung & Alarmierung)
- Sammlung von Controlling-Daten, Durchführung von Ist-Soll-Vergleichen

### Prüfung (Check)

- Prüfung des Sicherheitsmanagements in regelmäßigen Abständen
- Tests
- Audits
- Übungen
- Sicherheitsprüfungen
- Dokumentation, Analyse/Auswertung und Bericht der Ergebnisse

### Verbesserung (Act)

- Weiterentwicklung des Sicherheitsmanagements
- Identifizierung von Potenzial bzw. Änderungsbedarf
- Adaptierung beispielsweise auf Grund
    - neuer Erkenntnisse
    - neuer Gesetze
    - Veränderungen der Umwelt
- Priorisierung von Bedarfen und Erstellung einer Roadmap/eines Projektplans zur Verbesserung

## Standards und Normen zu IT-Sicherheit

### Erreichung von Vertrauen in IT-Sicherheit: Standards und Normen

- Zertifizierung
- Nachweis eines bestimmten Levels an IT-Sicherheit
- Einmaliger Nachweis vs. Regelmäßiger erneuter Nachweis der IT-Sicherheit
- Teilweise Grundlage für Vertragsbeziehungen
- Fokus auf unterschiedliche Aspekte der IT-Sicherheit
- Betriebs-Prozesse, Entwicklungs-Prozesse, Fertigungs-Prozesse, technische Maßnahmen

### ISO27k-Reihe

- 27000: Übersicht/Einführung ISO27k Standards inkl. Verwendeter Vokabeln
- 27001: Information Security Management System (ISMS) Anforderungen
- 27005: Information Security Risk Management Standard
- …

### ISO2001

- Spezifikation von Anforderungen an Informationssicherheitsmanagementsystemen (ISMS)
- Errichtung, Umsetzung, Betrieb, Überwachung, Überprüfung, Aufrechterhaltung und Verbesserung eines dokumentierten ISMS
- **Anwendbar** auf Bedürfnisse von **Organisationen jeglicher Art**, Größe oder Ausprägung oder Teile davon
- Entwickelt, um die Auswahl geeigneter und angemessener Sicherheitskontrollen durchzuführen

### Aspekte bei der Einführung eines ISMS nach ISO 27001

- ISMS-Richtlinie, die die Grundlagen des ISMS beschreibt
- Umfang des ISMS
- Prozeduren und Maßnahmen zur Pflege eines ISMS
- Beschreibung der Methoden zur Risikobewertung
- Risikobewertung: selbst
- Plan zum Umgang mit Risiken
- Dokumentation der Sicherheitsmaßnahmen und eine Beschreibung zum Nachweis der Effektivität
- Erklärung über die Anwendbarkeit der einzelnen Sicherheitsmaßnahmen

### IT-Grundschutz: Idee und Konzeption

- Herausgegeben vom BSI, Deutschland
- „Kochbuch“ für **normales** Schutzniveau
- Praktikable Durchführung von IT-Sicherheitsanalysen
- Kosteneffektive Erhöhung des IT-Sicherheitsniveaus
- **Schnelle Identifizierung** und **Umsetzung** von Sicherheitsmaßnahmen
- Angemessener Schutz durch **Kombination von organisatorischen, personellen, infrastrukturellen & technischen** Maßnahmen
- Verwendung eines Baukastenprinzips: Bausteine, Gefährdungen, Maßnahmen
- Soll-Ist-Vergleich empfohlene und realisierte Maßnahmen
- Einfache und arbeitsökonomische Erstellung von IT-Sicherheitskonzepten

### Information Technology Infrastructure Library (ITIL)

- ITIL Sicherheits-Management baut auf dem ISO 27001 Standard auf
- Hinweise zu **Best-Practices**
- Richtlinien was man wie umsetzen soll
- **Prozess-basiert**
- Optimierung des operativen Betriebs von IT-Services
- Anwendung auf nahezu alle IT-Infrastrukturen möglich

## Sicherheitskonzepte

### Inhalte

- Erforderliche Maßnahmen zur
- Realisierung und Aufrechterhaltung eines
- angemessenen und definierten Sicherheitsniveaus

### Fragen in einem Sicherheitskonzept

- Was will ich schützen?
- Wogegen soll ich mich schützen?
- Wie kann ich diesen Schutz erzielen?
- Kann ich mir diesen Schutz leisten?

### Business Continuity (BC)

- Mögliche und erforderliche Verfahren und Konzepte,
- die bei Einwirkung existenzieller Bedrohungen
- die **Erhaltung der Geschäftstätigkeit** ermöglichen.

# Betriebssystemsicherheit

## Grundlagen

„Ein Betriebssystem umfasst die Programme eines digitalen Rechensystems, die zusammen mit den Eigenschaften der Rechenanlage die Grundlage der möglichen Betriebsarten des digitalen Rechensystems bilden und insbesondere die Abwicklung von Programmen steuern und überwachen.“ (DIN44300)

## Sicherheitsziele von Betriebssystemen

Übliche Sicherheitsziele wie

- Vertraulichkeit
- Integrität
- Verfügbarkeit

## Maßnahmen zur Erhöhung der IT-Sicherheit bei Betriebssystemen

- Berechtigungssysteme
- Firewalls
- Speicherverwaltung, z.B.
  - Address Space Layout Randomization (ASLR)
  - Non Executable Stack
- Verschlüsselung von Filesystemen
- Jails/Sandboxes (z.B. chroot), Virtualisierung
- Härtung von Systemen

## Minimierung der Anzahl von Angriffsvektoren/der Attack Surface

- Grundgedanken für die Härtung eines Systems
  - Ein nicht installiertes Service kann nicht angegriffen werden.
  - Ein nicht vorhandenes Tool kann nicht für einen Angriff verwendet werden.
  - Ein nicht vorhandenes Recht kann nicht missbraucht werden.
- Ausgewählte Methoden der Minimierung der Anzahl von Angriffsvektoren
  - Ausschließlich Services installieren, die erforderlich sind
  - Ausschließlich Tools installieren, die erforderlich sind
  - Nur BenutzerInnen anlegen, die erforderlich sind
  - Rechte restriktiv vergeben (BenutzerInnen, Files, …->Concept of Least Privilege)

## Härtung von Systemen

„‚Härten‘ (engl. Hardening) bedeutet in der IT-Sicherheit die Entfernung aller Softwarebestandteile und Funktionen, die zur Erfüllung der vorgesehenen Aufgabe durch das Programm nicht zwingend notwendig sind.“ (BSI: Leitfaden IT-Sicherheit, 2012)

- Ziel: Verringerung der Möglichkeiten für Angriffe
- Hardening ist prinzipiell auf allen (konfigurierbaren) Systemen möglich, unabhängig vom Betriebssystem
- Hardening ist eine zusätzliche Maßnahme, kein Ersatz für andere Sicherheitsmaßnahmen (wie z.B. Security Patches etc.)

## … in einer idealen Welt

- Erstellung eines Minimal-Systems
  - Auswahl sicherer Hardware
  - Service(s)

    - (Remote) Admin Utilities
    - Support Libraries
    - Betriebssystem
    - System Monitoring
- Eingeschränkte, sichere Konfiguration
- Beispiel für ein Minimalsystem und gutes Know-How
- Linux From Scratch (LFS)

### … in der realen Welt

- … selten möglich
- Gründe dafür sind beispielsweise
    - Fehlendes Know-How
    - Budget
    - Zeit
    - 3rd Party Software
    - Proprietäre Systeme

### Umsetzung in der realen Welt

- Erforderliche Funktionen festlegen
- System installieren, updaten (nicht immer eine Option)
- Nicht erforderliche Software entfernen
- Falls nicht löschbar: nicht erforderliche Dienste deaktivieren
- Zugriffsrechte strenger setzen
- Konfigurationen überprüfen
- Default Passwörter/Geheimnisse ändern

## Beispiele für Unix/Linux Security Konzepte

- Berechtigungen (ls, chmod, chown, chgrp)
    - User, Group, Others
    - Weitere Modelle, unterschiedlich weit verbreitet
        - erweiterte Berechtigungen (lsattr, chattr)
        - ACL granulare Berechtigungen
        - SELinux/AppArmor
- Chroot/Jailroot, Linux Container, Docker
- suid/sgid
- Alle BenutzerInnen sind diesen Sicherheitskonzepten unterworfen, i.A. Ausnahme root
- Unterschiedliche Maßnahmen, um Speicher zu schützen

#### *Berechtigungen von root*

- Files/Directories Lesen
- Änderung von Zugriffsberechtigungen
- Wechsel der UID
- Prozesssteuerung
- Verändern von Systemparametern (Limits f. Prozesse, Filesysteme,. . . )
- User-Management
- System-Management (Shutdown, Reboot,. . . )
- Aktivierung Promiscous Mode v. Netzwerk Interfaces
- Filesysteme (un)mounten
- chroot

### *suid*

- Viele Systemprogramme nutzen das suid-Bit
- Theoretisch kein Sicherheitsproblem
- Praktisch jedoch immer wieder fehlerhaft implementiert (z.B. Buffer Overflows)
- suid wird oft verwendet, obwohl nicht erforderlich

### *chroot*

- change root: / wird geändert
  z.B. /var/www -> /
- Verwendung für Testumgebung
- Erhöhung der Sicherheit
- richtige Anwendung beachten – Benutzer root
- Minimierung der Tools in einer Jail-Umgebung

### *Bibliotheken und chroot*

- Bei dynamisch gelinkten Applikationen müssen die Bibliotheken alle aus dem Jail erreichbar sein, d.h. in der neuen durch chroot erzeugten Umgebung
- Statisch gelinkte Applikationen funktionieren ohne weitere Bibliotheken

### *Linux-Firewall*

- Backend derzeit iptables
- Frontends: Commandline, GUIs
- Überwacht und filtert gesamten Datenverkehr über TCP/IP
- Einfacher Packet-Filter oder Stateful Inspection
- Hohe Flexibilität

### *Remote-Zugriff auf Linux*

- SSH als Standard
  - Deaktiviertes X-Forwarding
  - Aktiviertes X-Forwarding
  - Verwendung von Programmen wie screen/tmux/. . .
- Weitere Tools wie Remote Desktop/VNC usw. ebenfalls möglich

## Principle of Least Privilege

Principle of Least Privilege: The principle of least privilege states that a subject should be given only those privileges that it needs in order to complete its task. (M. Bishop: Computer Security: Art and Science)

- Daraus folgt mindestens
  - Nicht-priviligierte User-Accounts
  - Priviligierte(r) Administrations-Account(s), z.B. root in Linux oder Administrator in Windows
- Für unterschiedliche Aktionen sind besondere Rechte erforderlich, z.B. Installation von Programmen oder Änderung von Passwörtern
  - z.B. sudo in Linux
  - z.B. Runas in Windows

## Grundlegende Strategie zur Sicherung von Windows

- Das Starten von Programmen/Services ist standardmäßig verboten und nur für zugelassene Programme erlaubt

- Ausgehender und eingehender Netzwerkverkehr ist grundsätzlich verboten und nur für festgelegte Ausnahmen erlaubt
- Die automatische Nutzung von administrativen Rechten für administrativen Konten ist deaktiviert
- Weitere unterstützende Maßnahmen:
- Durchführen von (kontrollierten) Windows Updates
- Isolierung von Software (AppContainer)
- Einsatz des Enhanced Mitigation Experience Toolkits (EMET)

## Beispiele für Windows Security Konzepte

### Windows Zugriffssteuerung

- Zugriffssteuerung ist das Autorisieren von BenutzerInnen, Gruppen und Computern für den Zugriff auf Objekte und beruht auf Berechtigungen, BenutzerInnenrechte und Objektüberwachung
- Berechtigungen
    - Festlegung des gewährten Zugriffs auf ein Objekt oder eine Objekteigenschaft für BenutzerInnen/Gruppen
    - Unterschiedlich, je nach Objekttyp
    - Prinzipiell Lesen, Ändern, Eigentümer ändern, Löschen
- BenutzerInnenrechte
    - BenutzerInnen und Gruppen erhalten innerhalb der Computerumgebung spezielle Privilegien und Anmelderechte
- Objektüberwachung
    - Überwachen von BenutzerInnenzugriffen auf Objekte
    - Sicherheitsereignisse werden im Sicherheitsprotokoll registriert

### BenutzerInnenverwaltung in Windows

- Zugriff auf Ressourcen, Dateisystem, Drucker, usw. mit Hilfe einer eigenständigen Sicherheitsdatenbank
- Lokale Nutzerverwaltung
    - Zugriffssteuerung von Nutzer und Gruppen nur auf lokale Ressourcen
    - "Administrator" (SID S-1-5-21*-500) ist der erste lokale Benutzer am System
- Verzeichnis-basierte Nutzerverwaltung - Active Directory
    - Nutzer, Gruppen, Computer in der Windows Domäne
    - Zugriffsteuerung von Nutzer und Gruppen auf lokale und Domänen-Ressourcen

### Administrative und nicht-administrative Gruppen

Die BenutzerInnenverwaltung unter Windows kann grob in privilegierte (administrative) Gruppen und nicht-privilegierte (nicht-administrative) Gruppen eingeteilt werden.

| | |
|---|---|
| Administrator | Vollzugriff, zuweisen von Rechten und Berechtigungen |
| Benutzer | allgemeine berechtigte Aufgaben |
| Gäste | temporäres Profil, standardmäßig deaktiviert |
| Hauptbenutzer | BenutzerInnenkonten erstellen, lokale Gruppen,. . . |

### Windows Access Token

- Nach erfolgreicher lokaler Anmeldung wird ein Windows-Zugriffstoken generiert
- Jeder vom System für den angemeldeten Nutzer erzeugte Prozess erhält eine Kopie

- Informationen aus dem Token werden benutzt, um Zugriffsrechte des Prozesses auf das jeweilige Zielobjekt zu ermitteln
- Auswertung durch System Einträge in der ACL des Zielobjektes und der Informationen im Zugriffstoken
- Nutzung von Kerberos Tickets und des Kerberos Authentifzierungsprotokolls in der Windows Domäne

### Windows Firewall

- Überwacht und filtert gesamten Datenverkehr über TCP/IP
- Ist eine **Stateful Inspection Firewall**
- Für alle Verbindungen, welche der PC nach außen initiiert, ist eine Rückverbindung nach innen zulässig
- Erwünschter eingehender Datenverkehr muss ausdrücklich zugelassen werden
- Ausnahmeliste mittels Portnummer, Anwendungsname oder Dienstname

### Gruppenrichtlinien

- Steuerung von (Sicherheis)einstellungen auf dem System
- Zentralisierte Verwaltung der Computer in einer Domäne
    - Konfiguruation über Gruppenrichtlinienobjekte (GPO)
    - Regelmässige Aktualisierung der Gruppenrichtlinien aus der Domäne
- Administrative Vorlagen für eine Vielzahl an Einstellungen, z.B.
    - Richtline für sichere Kennwörter
    - Konfiguration der Windows Firewall
    - Zugriffe auf Wechselmedien regulieren

### BenutzerInnenkontensteuerung (UAC)

- User Account Control, basiert auf Integrity Levels (MIC) und Access Tokens
- Nutzung von administrativen Rechten erst durch Zustimmungsabfrage
    - Keine automatische Nutzung administrativer Rechte
    - Rechtereduzierung beim Zugriff auf nicht vertrauenswürdiger Datenquellen
    - Jedoch kein Schutz vor Malware, die mit reduzierten Zugriffsrechten ausführbar ist

### Mandatory Integrity Control (MIC)

- Kernkomponente der Windows Sicherheitsarchitektur - Bekannt unter Integrity Level
- Definieren das maximale Zugriffsrecht, das ein Prozess auf ein Windows-Objekt ausüben darf
- Beruhen auf. . .
    - der Definition von Vertrauensstufen (Untrusted, Low, Medium, High und System)
    - zugelassenen Aktionen auf das zu schützende Objekt (Lesen, Ausführen und Schreiben)
- Ein Prozess kann nur Objekte derselben Vertrauensstufe oder Objekte mit niedrigerer Vertrauensstufe als der eigenen Vertrauensstufe beschreiben
- Prozess mit Rechten eines Standardbenutzers kann nicht ohne Zustimmungsabfrage einen Administrator-Token benutzen

### Logging, Auditing

- Linux
    - primär Logfiles, z.B. /var/log/
        - messages, syslog, auth, . . .
    - AIDE, Snort, . . .
    - Diverse System-Tools zum Auditing (lsof,. . . )

- Windows
  - Windows erzeugt Event Logs (Ereignisprotokolle)
  - Aufgetretene Ereignisse einsichtbar per Event Viewer (Ereignisanzeige)
    - Anwendungsereignisse
    - Sicherheitsbezogene Ereignisse
    - Setupereignisse
    - Systemereignisse
    - Weitergeleitete Ereignisse

### Windows Spezifisches: Absicherung von Remotedesktopzugängen

- Remote Desktop / WinRM ist für viele AdministratorInnen unverzichtbar für die tägliche Arbeit
- Zugriff auf den Windows-Desktop und Einstellungen mit den Privilegien des Benutzers
- Mögliche Maßnahmen zur Absicherung von RD Zugriffen:
  - Einschränkung der Reichweite des RD-TCP-Ports
  - Härten des RDP-Protokolls (Verschlüsselung, Ciphers)
  - Unterbinden administrativer Anmeldungen per Remotedesktop mittels RDP-ACLs
  - Aktivierung der Network Level Authentication

### Backdoors, Rootkits

„You can't trust code that you did not totally create yourself. ... No amount of source-level verification or scrutiny will protect you from using untrusted code. "

- Ständige Problematik von Hintertüren
- Absicht ist unentdeckt und/oder illegal
  - geschützte Daten einzulesen, und/oder
  - Änderungen an geschützten Daten vorzunehmen
- Lösung Open Source?

- Rootkit
  - Start direkt zu Beginn des Systems
  - Änderung von tiefgreifenden OS-Funktionen
  - möglichst unbemerkt von BenutzerIn

# Sicherheitsfaktor Mensch / Social Engineering

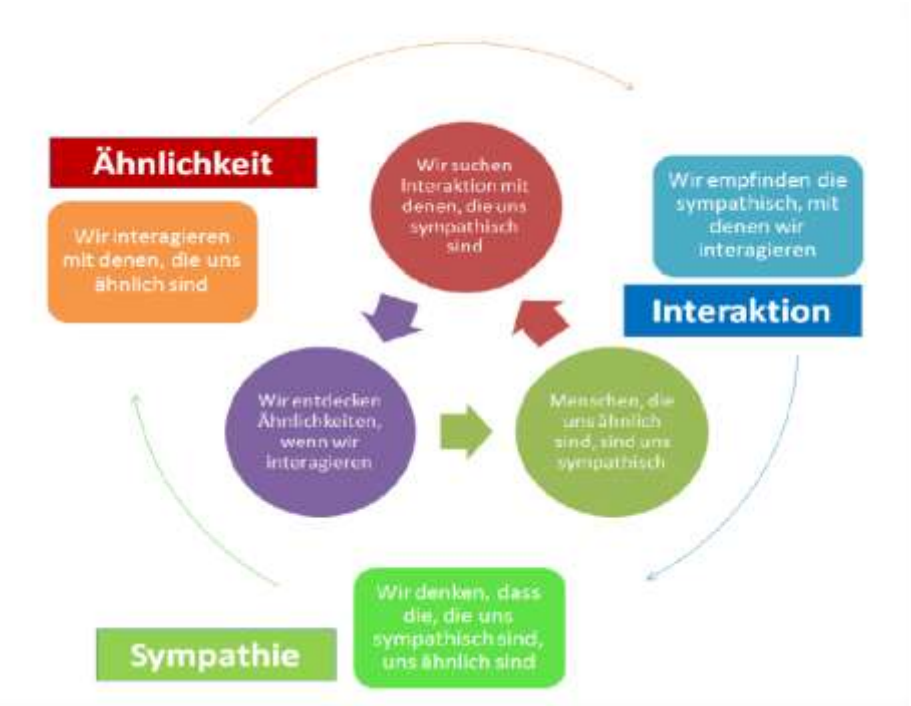
## Der Social Engineer

- Hohe soziale Fähigkeiten
- Hohe Selbstkontrolle
- In der Lage, spontan auf sich veränderte Situationen zu reagieren
- Selbstbewusst und willensstark
- Perfektionistisch veranlagt
- Ablehnung und Abscheu von Tradition, eingefahrenen Muster und Routine
- Tendenz, schnell von alten Sachen gelangweilt zu sein
- Ein Verlangen, sich mit Neuem zu beschäftigen und Neues zu erschaffen

## Menschliche Faktoren

| **Verhalten** | **Beschreibung** | **Beispiel** |
|---|---|---|
| Reziprozität | Man fühlt sich verpflichtet, Gefallen zurückzuzahlen | Nach einer Gratisprobe kauft man eine Ware |
| Konsistenz | Wir sind bestrebt, in Übereinstimmung mit früheren Verhalten zu handeln | Eine Zusage kann man schwer brechen |
| Soziale Bewährtheit | Wir orientieren uns an anderen | Produkte werden als Top Seller beworben |
| Sympathie | Sympathischen Menschen hilft man lieber | Attraktive Modells werden in der Werbung eingesetzt |
| Autorität | Anweisungen von Autoritäten werden weniger in Frage gestellt | Ein Arzt gibt seinen Patienten eine Anweisung |
| Knappheit | Knappe Ressourcen haben einen höheren Wert | Von einer Ware ist nur mehr ein Stück lagernd |

## Sympathie – Interaktion – Ähnlichkeit




## Angriff

### Allgemein

1. Informationsbeschaffung
2. Beziehungen aufbauen
3. Beziehungen ausnutzen
4. Ausführung

### Einschleichen

1. Situation ausforschen
2. Opfer ausforschen
3. Vertrauen gewinnen
    a. Qualifikation (Experte oder Kollege)
    b. Gutmütigkeit (Opfer vermeidet Schaden)
    c. Integrität (Erwartungen erfüllt)
4. Bindung erhalten

### Beeinflussen / Ausnutzen

1. Ziel identifizieren
2. Angriff planen
3. Vertrauen aufbauen
    a. Betrug / Täuschung (Wissen, Charme)
    b. Manipulation (Autorität, Kreativität)
    c. Überredekunst (Wissen, Ausdauer, Klugheit)
    d. Beeinflussung (Wissen, Autorität, Selbstbewusstsein)
4. Rolle einnehmen
5. Informationen erhalten
6. Sicherheitslücke ausnutzen (weiter mit 5.)

### Angriffsziele

- Geld

- Ego
- Unterhaltung
- Streitsachen
- Zugehörigkeit
- Status

## Klassifizierung

- Computer-based
- Human-based
    - Reverse SE

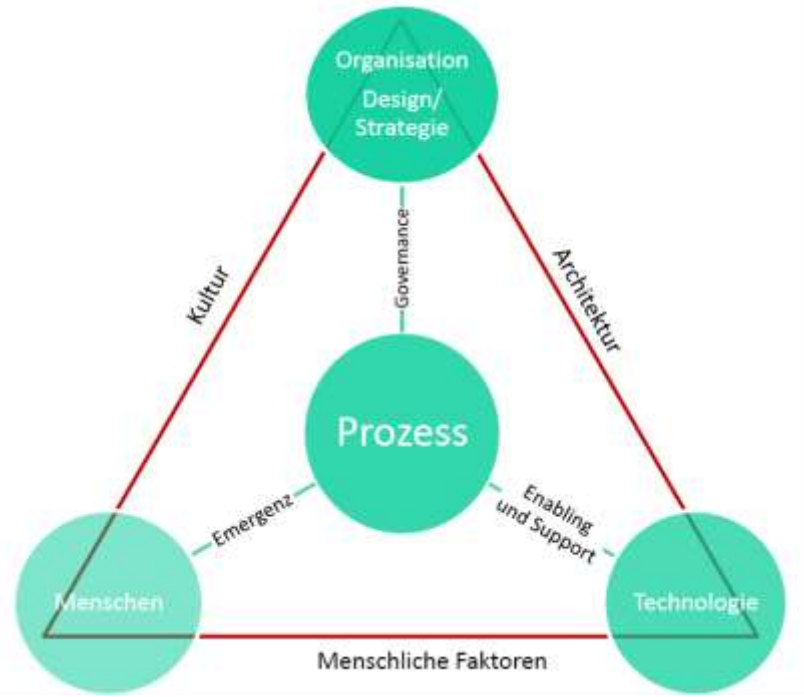
## Schutz

- User Awareness

## Forschung

- „Abgeschlossen“
    - IT-Governance
    - IT-Security
- Forschungsstand
    - Socaial Engeneering
    - Menschliche, technische und organisatorische Sicherheit
- Forschungslücke
    - Systemische Sicherheit
    - IT-Governance Modelle als systemische Sicherheit

## Systemic Security Management Model



# Inhalt