CARISSMA
Institute of Electric, Connected and Secure Mobility

Kapitel 3: Security Engineering Prozess in der Software-Entwicklung

Vorlesung „Security Engineering in der IT“

Prof. Dr.-Ing. Hans-Joachim Hof

Technische Hochschule Ingolstadt

CARISSMA
Institute of Electric, Connected and Secure Mobility

Kapitel 3.1: Security Requirements Engineering

Vorlesung „Security Engineering in der IT“

Prof. Dr.-Ing. Hans-Joachim Hof

Technische Hochschule Ingolstadt

Ziele der heutigen Veranstaltung

- **Studentinnen und Studenten können für ein Softwareprodukt sinnvolle Sicherheitsanforderungen erstellen und dokumentieren**
- **Studentinnen und Studenten kennen verschiedene Ansätze um Bedrohungen für Werte zu identifizieren und die Bedrohungen zu bewerten**
- **Studentinnen und Studenten können an einem konkreten System eine Bedrohungs- und Risikomodellierung vornehmen**

Breakout Groups zu Anforderungen

Fragestellungen:

1. **Definieren Sie den Begriff „Anforderung“. Welche Granularität von Anforderungen gibt es?**
2. **Welche Eigenschaften haben gute Anforderungen?**
3. **Was ist ein Akzeptanzkriterium? Wodurch zeichnet sich ein gutes Akzeptanzkriterium aus?**
4. **Was sind funktionale Anforderungen? Was sind nicht-funktionale Anforderungen?**
5. **Wie werden Anforderungen dokumentiert?**

- **Können funktionale oder nicht funktionale Anforderungen sein, meist handelt es sich um nicht-funktionale Anforderungen**
- **Beispiel funktionale Sicherheitsanforderungen:**
  „Es gibt eine Authentifizierungsfunktion“
- **Beispiel nicht-funktionale Sicherheitsanforderungen:**
  „Alle Benutzer müssen sich authentifizieren“
  „Die Bestimmungen des Datenschutzes werden eingehalten“
- **Nicht-funktionale Anforderungen wirken sich üblicherweise auf funktionale Anforderungen aus – stehen zu diesen z.B. im Konflikt**
  - Sicherheitsanforderungen, die erst im laufe des Projekts „erscheinen“ erzeugen erheblichen Aufwand

- **Oft Unklarheiten**
  - z.B.: „es gibt eine Authentifizierungsfunktion“ (funktional) und „alle Benutzer müssen sich authentifizieren“ (nicht funktional)
- **Akzeptanzkriterium bei nicht-funktionalen Anforderungen schwierig zu definieren**
- **Nicht-funktionale Sicherheitsanforderungen “verschwinden“ in der Praxis oft in den Akzeptanzkriterien von funktionalen Anforderungen**
  - Diskussion: Warum ist das ein Problem?

- **Abbildung nicht-funktionale Eigenschaft auf funktionale Eigenschaft**
- **Abbildung nicht-funktionale Eigenschaft auf funktionale Eigenschaft + leicht zu überprüfende nicht-funktionale Eigenschaft**
- **Keine Umwandlung möglich und Prüfung Eigenschaft nicht möglich: gehe nach Reihe von Prüfungen davon aus, dass Eigenschaft erfüllt**
  - heute häufigster Fall
  - leider: Akzeptanzkriterium = Test, der auch bei positivem Ausgang nichts über Korrektheit sagt

Sicherheitsanforderungen

Probleme in der Praxis

- **Sicherheitsanforderungen werden erst in späteren Projektphasen erhoben, üblicherweise erhebliche Änderungen an bestehenden Funktionen notwendig und eventuell an der Architektur notwendig**
- **Auftraggeber können meist Sicherheitsanforderungen nicht sinnvoll formulieren**
- **Mangelnde Sensitivität gegenüber Sicherheitsanforderungen (ja, auch heute noch!)**
- **Neben technischen Sicherheitsanforderungen existieren auch organisatorische Sicherheitsanforderungen – beiden werden oft nicht sinnvoll getrennt**
- **Abhängigkeiten der nicht-funktionalen Sicherheitsanforderungen zu funktionalen Anforderungen oft nicht direkt erkennbar.**

- **Requirements Engineering: Prozess zur Ermittlung von Anforderungen an ein Softwareprodukt**
- **Security Requirements Engineering: Prozess zur Ermittlung von Sicherheitsanforderungen im Rahmen des Requirements Engineerings**

- **Realistische Annahmen bezüglich Bedrohungslage und möglichen Angreifer sind notwendig aber nicht verfügbar**
- **Richtiges Sicherheitsniveau muss gefunden werden**
- **Ähnliche Probleme im Requirements Engineering?**

Qualitätsanforderungen an die Anforderungsanalyse nach [7] (nicht spezifisch für Security)

- **Ausnahmslos alle Anforderungen sind in Vorgaben oder Entscheidungen der Teilhaber (Stakeholder) bzw. Nutzererfordernissen begründet**
- **Jede Anforderung ist eindeutig. Sie ist klar, prägnant und unzweideutig, sie kann nur auf eine Weise interpretiert werden**
- **Jede Anforderung ist widerspruchsfrei zum Zweck und zu den Zielen des Systems/Produkts**
- **Jede Anforderung ist nachprüfbar, der Erfüllungsgrad kann bewertet werden**
- **Sämtliche Anforderungen können widerspruchsfrei in funktionale Anforderungen, Sicherheitsanforderungen oder nicht-funktionale Anforderungen unterschieden werden**
- **Funktionale Anfoderungen sind Entscheidungen der Stakeholder bzw. Nutzererfordernisse hinsichtlich der erwarteten Funktionalität**

Qualitätsanforderungen an die Anforderungsanalyse nach [7] (nicht spezifisch für Security)

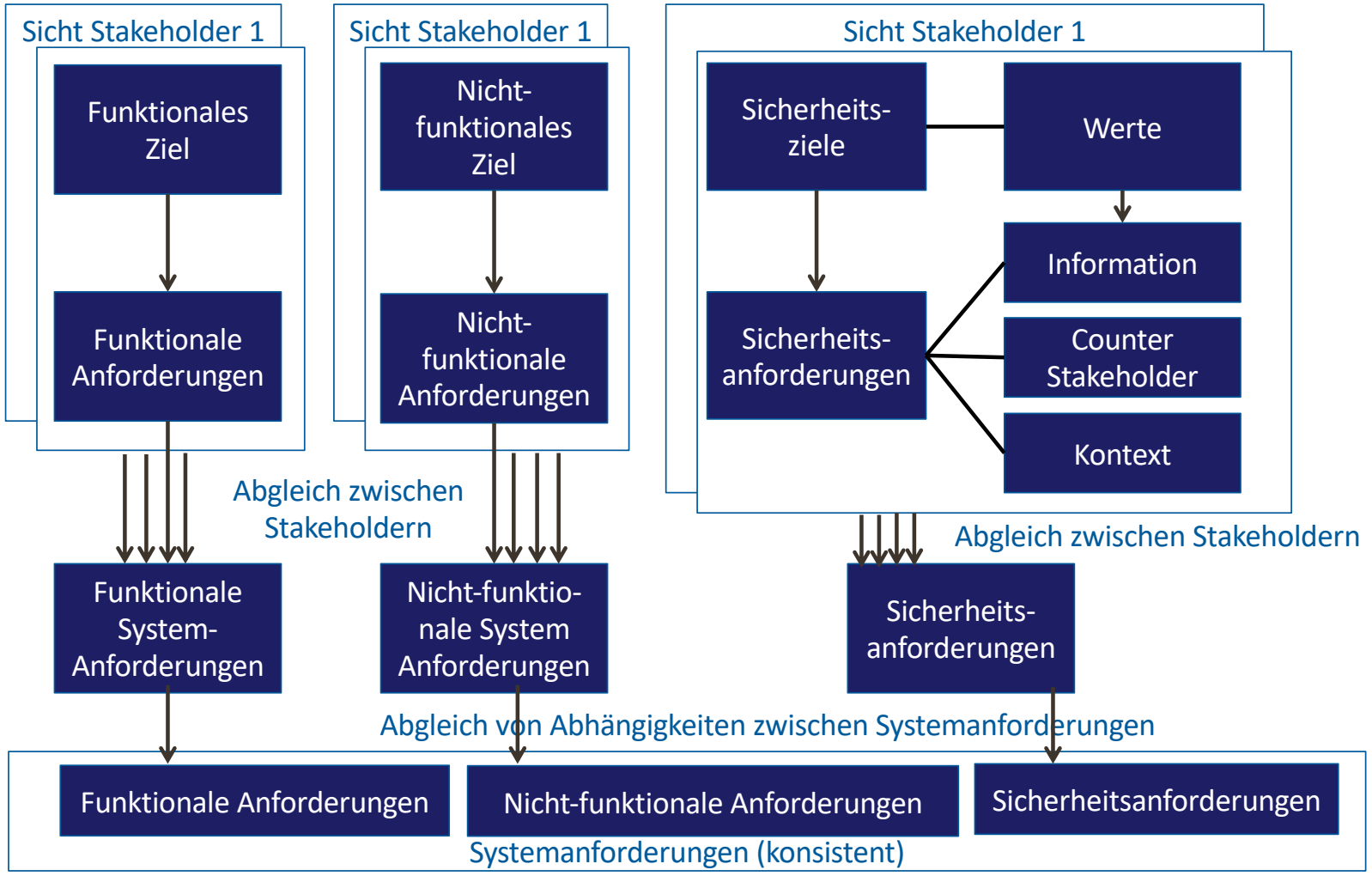
- **Sämtliche funktionale Anforderungen sind unabhängig voneinander (orthogonal)**
- **Jede der funktionalen Anforderungen kann durch eine Funktion oder Eigenschaft das Systems erfüllt werden.**
- **Ausnahmslos alle Sicherheitsanforderungen können auf Sicherheitsziele, Sicherheitsbedürfnisse der Stakeholder und Nutzer oder Bedrohungen schützenswerter Güter oder Personen zurückgeführt werden**
- **Sämtliche Sicherheitsanforderungen sind Restriktionen des technischen und technologischen Lösungsraums**
- **Sämtlichen Sicherheitsanforderungen liegt eine Risikoanalyse der zu schützenden Werte zugrunde**
- **Sämtliche Sicherheitsanforderungen sind unabhängig voneinander (orthogonal)**

Security Requirements Engineering

Qualitätsanforderungen an die Anforderungsanalyse nach [7] (nicht spezifisch für Security)

- **Sämtliche Sicherheitsanforderungen können durch Funktionen oder Eigenschaften des Systems oder der Systemumgebung erfüllt werden.**
- **Keine Sicherheitsanforderung darf durch zweifelhafte Annahmen an das System oder die Systemumgebung ganz oder teilweise erfüllt werden**
- **Nicht-funktionale Anforderungen betreffen ausschließlich die Funktions- bzw. Servicequalität.**
- **Sämtliche nicht-funktionalen Anforderungen sind explizit und hinsichtlich des Erfüllungsgrads spezifiziert.**

Modell Security Requirements Engineering angelehnt an [1]



- **Stakeholder: Individuum, Gruppe oder Organisation, die ein Interesse an dem System hat**
- **Counter Stakeholder: Der Stakeholder, gegen den sich ein Sicherheitsziel richtet.**
  - Nicht unbedingt nur Angreifer
  - Beispiel: Sicherheitsziel Vertraulichkeit: Counter Stakeholder = alle, die die Informationen nicht erlangen dürfen
- **Kontext: Beschreibung der Begebenheiten, in denen eine Sicherheitsanforderung erfüllt werden muss**
- **Modell folgt einem werte-basierten Ansatz (heute üblich).**

- **Strukturierte Vorgehensweise zur Ermittlung von Sicherheitsanforderungen unter Berücksichtigung Risiko**
  - Betrachtung Risiko stellt Wirtschaftlichkeit sicher

- **Vorgehen (Übersicht):**
  - Analyse der schützenswerten Güter (=Werte)
  - Identifikation möglicher Bedrohungen für diese Werte
  - Bewertung des mit der Bedrohung verbundenen Risikos

- **Gegenmaßnahmen basierend auf Risiko**
  - Nur für hohe/kritische Risiken
  - Dazu Anforderungen formulieren

Überblick

Modell kommt in vielen Abwandelungen vor!

Phase 1

Werte, Akteure und Operationen identifizieren

- **Werte = zu schützende Güter**
  - Wichtig: Sicht der Stakeholder berücksichtigen, Bedeutung von Werten unterschiedlich für Stakeholder
- **Akteure = Personen oder Prozesse, die Operationen auf den Werten durchführen**
- **Operationen = Handlungen, die Akteure auf den Werten durchführen**

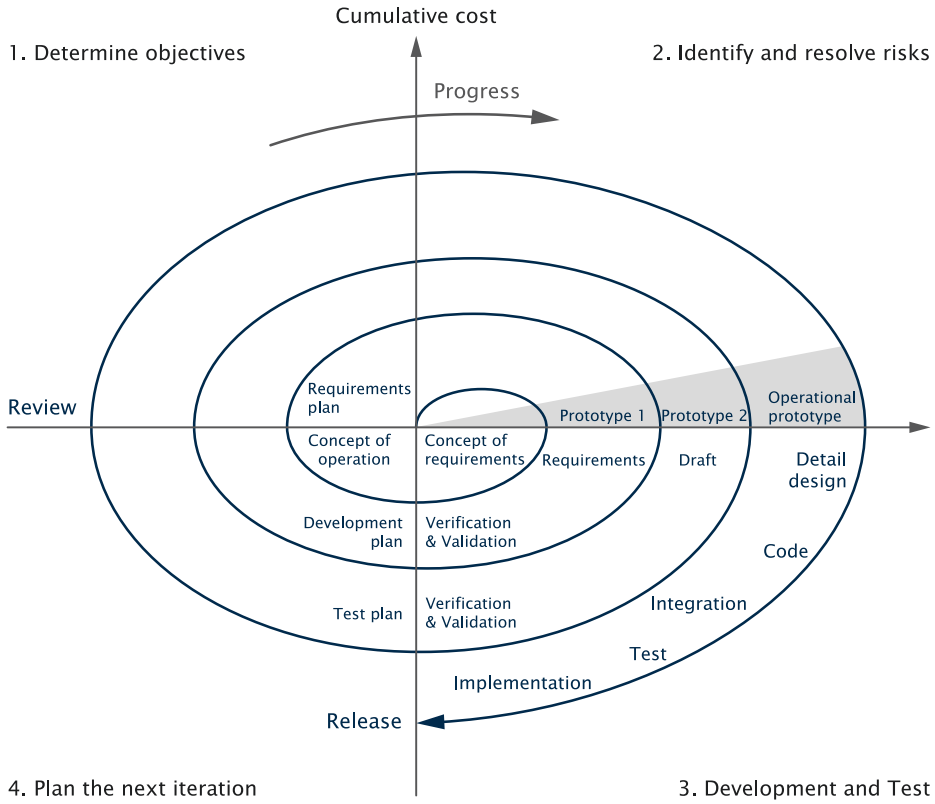
- **Konzentration auf die zu schützenden Werte reduziert die Problemstellung und verhindert falschen Fokus sowie Over Engineering**

- **Betrachtung der Akteure und deren Operationen fokussiert die Problemstellung weiter**

- **Bedrohungsanalyse erfordert grobe Architektur des zu betrachtenden Systems**
  - Existierendes System, das erweitert wird: Architektur liegt vor
  - Neues System: Henne-Ei-Problem: gute Architektur basiert auf Sicherheitsanforderungen, diese müssen erst noch ermittelt werden
- **Umgang mit Änderungen im Projekt**
  - Bedrohungsmodellierung iterativer Prozess
  - Sicherheitsvorgaben und Architektur iterativ verfeinern
- **Da Werte meist Daten sind Fokus bei Architektur auf:**
  - Datenhaltung
  - Datenflüsse

Einschub: iterative Systementwicklung

Spiralmodell nach Böhm [8] aus dem Jahr 1988 – Grundlage für viele heutige Prozesse



- **Diskutieren Sie, wie Sie eine Software-Architektur/System-Architektur geeignet darstellen lässt. Wie stellen Sie Datenhaltung und Datenflüsse dar?**

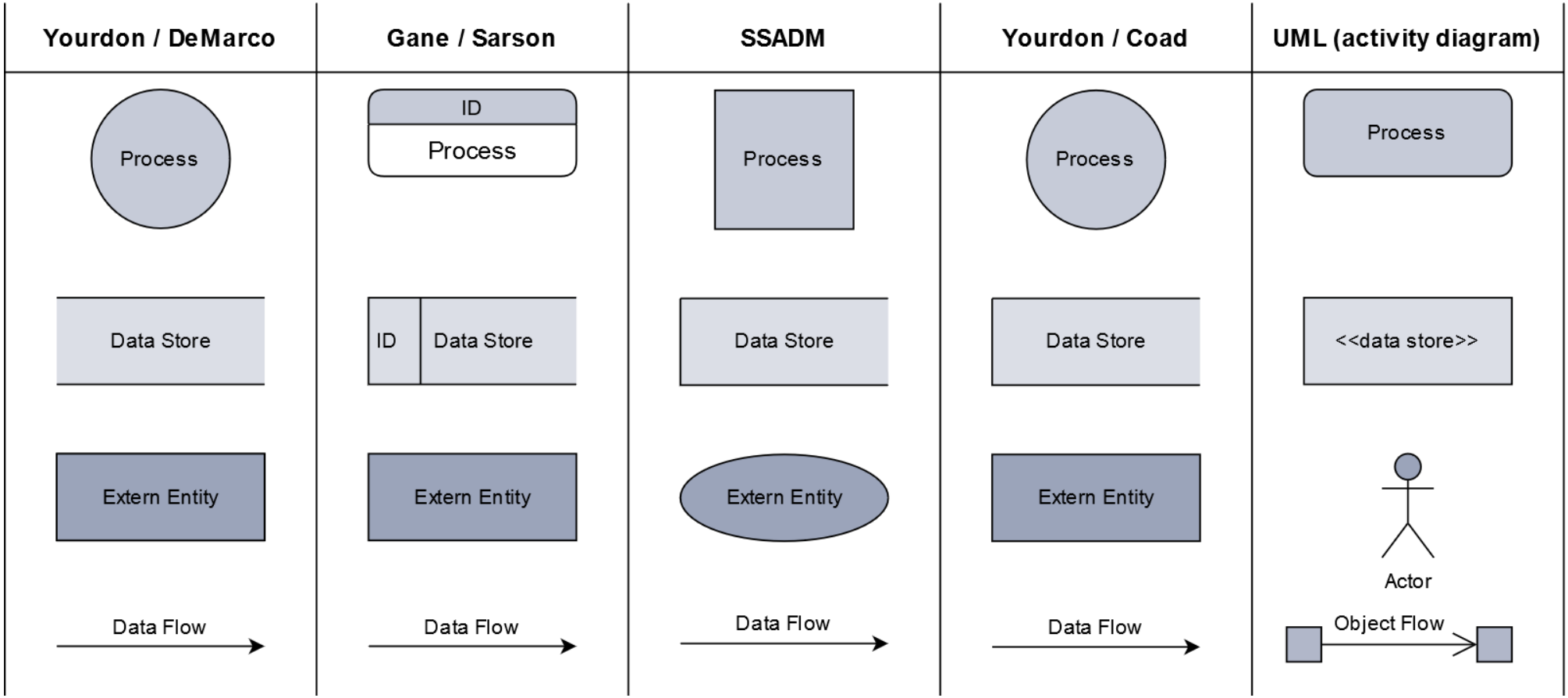
Datenflussdiagramm

Übersicht

- **Definition Datenflussdiagramm: Das Datenflussdiagramm stellt den Informationsfluss in einem System grafisch dar.**
- **Keine Datenflussdiagramme in ULM, dort Aktivitätsdiagramme, die aber vor allem den Kontrollfluss beschreiben**
- **Grundlegende Elemente:**
  - Verarbeitungsfunktionen (Processes)
  - Speicher/Datenbanken (Data Store)
  - Start/Endpunkte (External Entity)
  - Datenflüsse (Flow)

Datenflussdiagramm

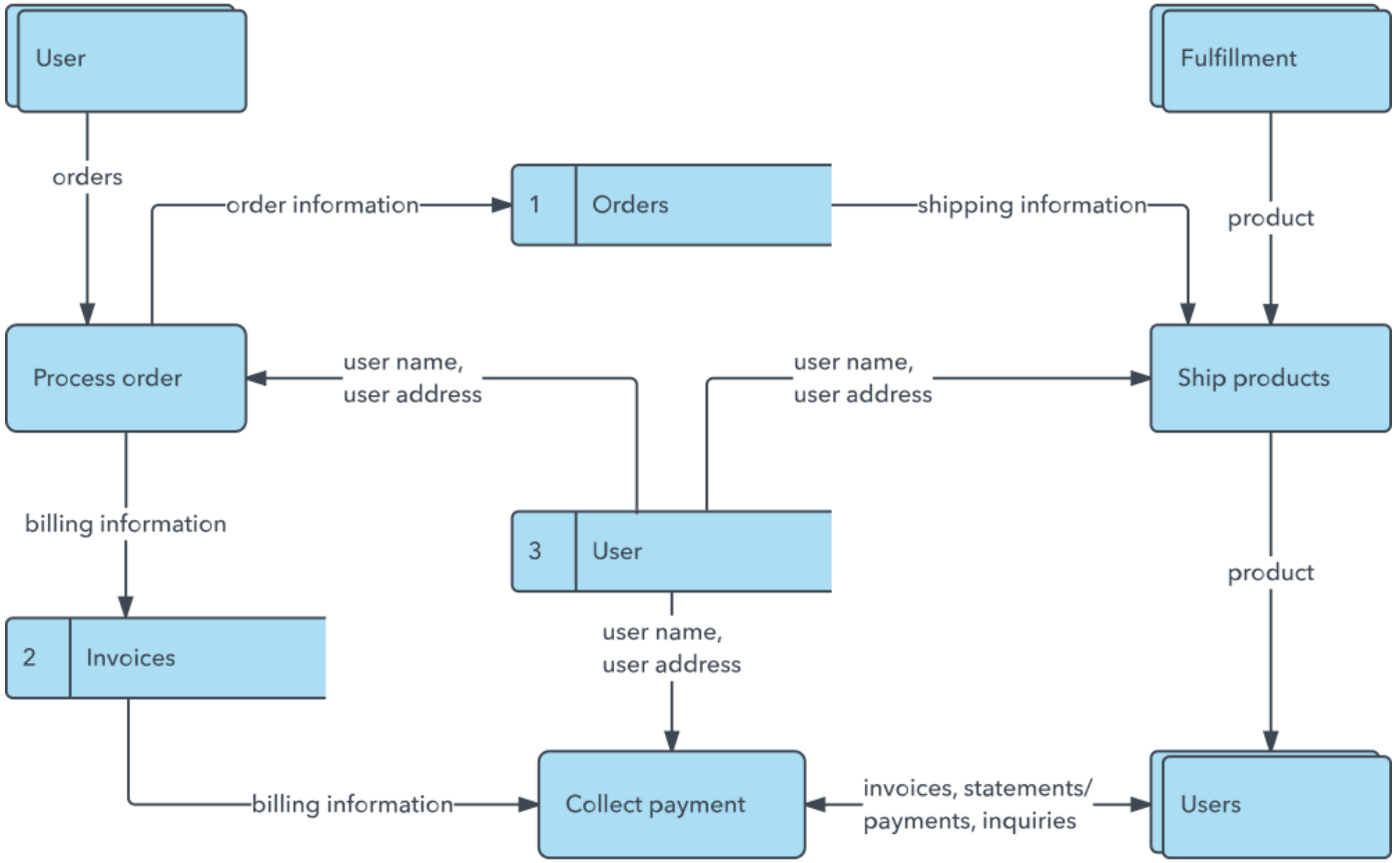
Notationen



Bildquelle: https://www.solcept.ch/de/blog/komplexe-systeme/datenflussdiagramme/

Datenflussdiagramm

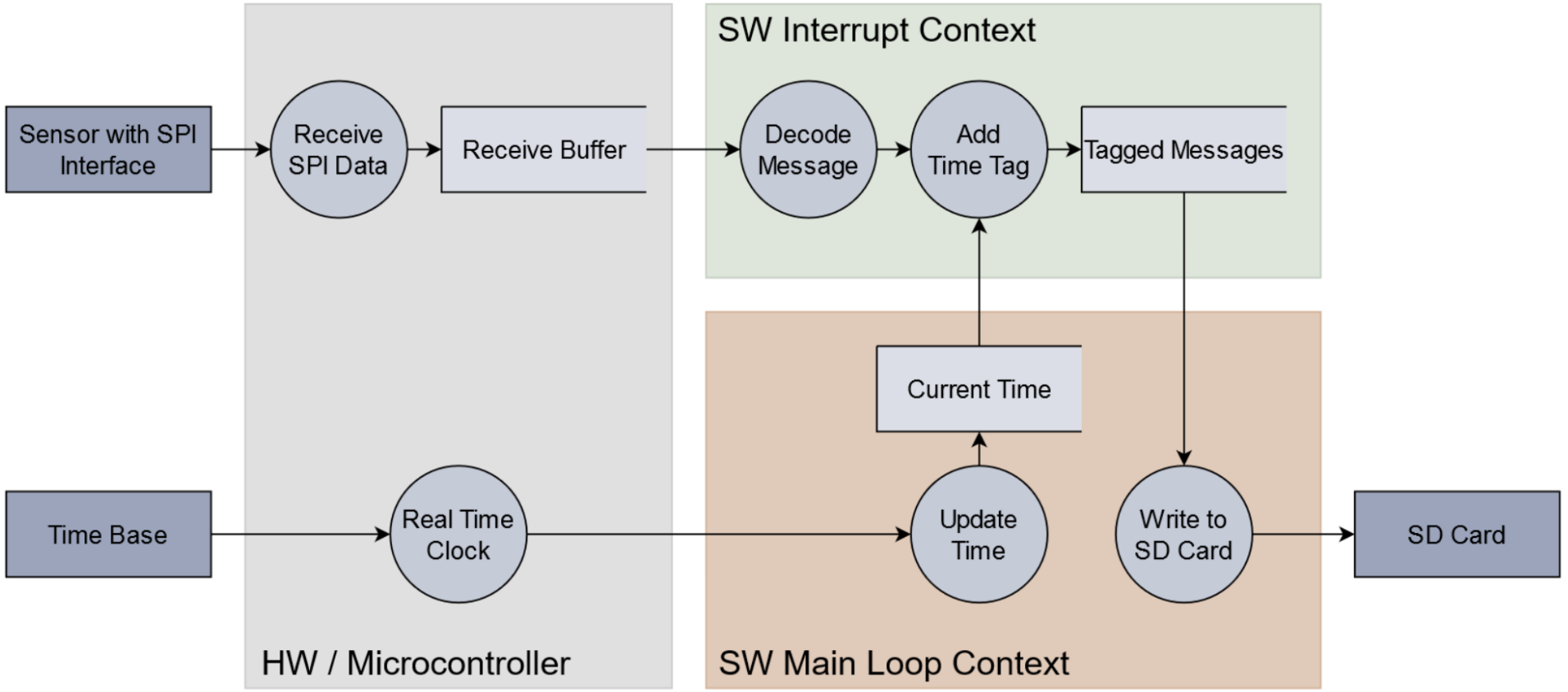
Beispiel nach Gane/Sarson (Verarbeitung einer Bestellung)



Bildquelle: https://www.lucidchart.com/pages/de/erstellen-sie-ein-datenflussdiagramm

Datenflussdiagramm

Beispiel nach Yourdan/Coad (einfacher Daten-Logger)



Bildquelle: https://www.solcept.ch/de/blog/komplexe-systeme/datenflussdiagramme/

Phase 2

Sinn und Zweck

- **Am Ende von Phase 2 liegt eine Architekturskizze vor, anhand derer klar ist, wo Werte gespeichert werden und über welche Systemkomponenten Datenflüsse, die Werte beinhalten, gesendet werden**

- **Eine Architekturskizze auch in frühen Phasen sorgt dafür, dass die Analyse konkret durchgeführt werden kann und somit brauchbare Ergebnisse liefern kann.**

Phase 3

System zerteilen

- **Definition Vertrauensgrenze: Eine Vertrauensgrenze ist eine tatsächliche oder gedachte Abgrenzung zwischen zwei oder mehr Komponenten in einem System. Sie trennt im System zwei oder mehr Bereiche mit unterschiedlichem Sicherheitsniveau.**

- **In Phase 3 werden Vertrauensgrenzen (engl. Trust Boundaries) identifiziert und dokumentiert**
  - Bereiche mit gleichem Sicherheitsniveau identifizieren
  - Grenzen identifizieren, über die Daten nur sehr kontrolliert oder gar nicht fließen sollen.
  - Mehr Vertrauensgrenzen erhöht Sicherheit, jedoch damit einhergehend höherer Aufwand

- **System entlang Vertrauensgrenzen zerteilen**

- **Eventuell Phase 2 wiederholen, dazu auf Granularität prüfen (weitere Zergliederung möglich?)**

- **Die Zerteilung des Systems aus Sicherheitssicht liefert einen ersten Überblick über die verschiedenen Sicherheits-Domänen. Vertrauensgrenzen sind Kandidaten für den Einsatz von Sicherheitsmaßnahmen und ebenfalls Ansatzpunkte für mögliche Bedrohungen.**

Phase 4

Bedrohungen identifizieren und dokumentieren

- **Definition Bedrohung [5]: „ Eine Bedrohung ist ganz allgemein ein Umstand oder Ereignis, durch den oder das ein Schaden entstehen kann. Der Schaden bezieht sich dabei auf einen konkreten Wert wie Vermögen, Wissen, Gegenstände oder Gesundheit. Übertragen in die Welt der Informationstechnik ist eine Bedrohung ein Umstand oder Ereignis, der oder das die Verfügbarkeit, Integrität oder Vertraulichkeit von Informationen beeinträchtigen kann, wodurch dem Besitzer bzw. Benutzer der Informationen ein Schaden entstehen kann. [...] Trifft eine Bedrohung auf eine Schwachstelle (insbesondere technische oder organisatorische Mängel), so entsteht eine Gefährdung.“**

- **Insbesondere Angriffe sind Bedrohungen (bzw. Gefährdungen) für Systeme**

- **Ziel in Phase 4: möglichst viele Bedrohungen für Werte aus Phase 1 identifizieren**

- **Bedrohungen pro Wert erheben, eventuell Werte gruppieren**
  - Welche zusätzlichen Operationen/unberechtigten Operationen auf einem Wert kann ein Angreifer durchführen?
  - Welche Angriffe auf einen Wert kann ein Angreifer durchführen?

- **Bedrohungen an dem Vertrauensgrenzen erheben**

- **Verschiedene Werkzeuge:**
  - Brainstorming
  - Checklisten
  - Datenflussanalyse
  - Angriffsbaum
  - Misuse Cases
  - …

- **Zu bearbeitende Fragestellung: „Gesetzt den Fall, dass die Software so entwickelt wird, wie es die Architekturskizze beschrieben ist, so dass die genannten Vertrauensgrenzen eingehalten werden: Wo würden Sie angreifen, um den Wert <konkreten Wert nennen> zu schädigen“**

- **Brainstorming-Regeln:**
  - Keine Kritik an Beiträgen anderer Teilnehmer
  - Keine Wertung oder Beurteilung eigener oder anderer Ideen
  - Jeder darf seine Ideen und Gedanken frei äußern
  - Je fantasievoller, desto besser

- **Danach: thematische Gruppierung und Bewertung der Relevanz**

- **Klassifizierung von Angriffen z.B. über STRIDE**

Klassifikation von Bedrohungen

STRIDE

- **Spoofing: Erschleichen einer Identität, oft für folgenden Angriff benutzt**
- **Tampering: Verändern von Informationen**
- **Repudiation: Abstreiten durchgeführte Aktion**
- **Information Disclosure: Veröffentlichen von Informationen**
- **Denial of Service: Angriff auf Verfügbarkeit**

- **Elevation of Privilege: Verschaffen von zusätzlichen Rechten**

- **Liste mit Angriffsarten, die einfach abgehakt werden können**
- **Vorteil:**
  - schnell durchzuführen
  - für manche Domänen sind ausführliche Checklisten vorhanden
- **Nachteil:**
  - Vollständigkeit nicht sichergestellt
  - Qualität nicht sichergestellt
  - Eignung für Domäne nicht sichergestellt
- **Gut geeignet als Ergänzung zu anderen Methoden**

Beispiel Checkliste

OWASP Top 10: die 10 häufigsten Bedrohungen für Web-Anwendungen

#	2003	2004	2007	2010	2013	2017	2021
1	Unvalidated Input	Unvalidated Input	Cross Site Scripting (XSS)	Injection	Injection	Injection	Broken Access Control
2	Broken Access Control	Broken Access Control	Injection Flaws	Cross Site Scripting (XSS)	Broken Authentication and Session Management	Broken Authentication	Cryptographic Failures
3	Broken Authentication and Session Management	Broken Authentication and Session Management	Malicious File Execution	Broken Authentication and Session Management	Cross Site Scripting (XSS)	Sensitive Data Exposure	Injection
4	Cross Site Scripting	Cross Site Scripting	Insecure Direct Object Reference	Insecure Direct Object Reference	Insecure Direct Object Reference	XML External Entities (XXE)	Insecure Design
5	Buffer Overflow	Buffer Overflow	Cross Site Request Forgery (CSRF)	Cross Site Request Forgery (CSRF)	Security Misconfiguration	Broken Access Control	Security Misconfiguration
6	Injection Flaws	Injection Flaws	Information Leakage and Improper Error Handling	Security Misconfiguration	Sensitive Data Exposure	Security Misconfiguration	Vulnerable and Outdated Components
7	Inproper Error Handling	Improper Error Handling	Broken Authentication and Session Management	Insecure Cryptographic Storage	Missing Function Level Access Control	Cross Site Scripting (XSS)	Identification and Authentication Failures
8	Insecure Storage	Insecure Storage	Insecure Cryptographic Storage	Failure to Restrict URL Access	Cross Site Request Forgery (CSRF)	Insecure Deserialization	Software and Data Integrity Failures
9	Application Denial of Service	Application Denial of Service	Insecure Communications	Insufficient Transport Layer Protection	Using Components with Known Vulnerabilities	Using Components with Known Vulnerabilities	Security Logging and Monitoring Failures
10	Insecure Configuration Management	Insecure Configuration Management	Failure to Restrict URL Access	Unvalidated Redirects and Forwards	Unvalidated Redirects and Forwards	Insufficient Logging & Monitoring	Server-Side Request Forgery

Beispiel Checkliste

Annex UNECE R155 (Zulassungsvorschrift Cybersicherheit für Fahrzeuge)

Bedrohungen für Backend Servern für Fahrzeuge	Bedrohungen für Kommunikation zwischen Fahrzeugen	Bedrohungen für Software Update	Angriffe auf Halter/Fahrer
• Kompromittierter Backend Service • Eingeschränkte Verfügbarkeit • Datendiebstahl	• Spoofing • Einspielen von Nachrichten • Session Hijacking • Replay Attacks • Abhören • Denial of Service • Manipulation Software/Daten auf dem Fahrzeug • Virus und andere schädliche Inhalte	• Missbrauch Software Update • Unterdrückung legitimes Update	• Halter/Fahrer führen unbeabsicht Cyberangriff aus

Beispiel Checkliste

Annex UNECE R155 (Zulassungsvorschrift Cybersicherheit für Fahrzeuge)

Bedrohungen durch externe Konnektivität	Bedrohungen für Code und Daten im Fahrzeug	Potentielle Verwundbarkeiten
• Manipulation der Konnektivität für Cyberangriff • Angriff über 3rd Party Softwsare • Angriff über Zugriff auf externe Schnittstelle (z.B. USB, OBD)	• Extraktion von Code/Daten • Manipulation von Code/Daten • Löschen von Code/Daten • Einspielen von Malware • Störung der Funktionalität • Manipulation der Parameter des Fahrzeugs	• Gebrochene/falsch verwendete kryptographische Methoden • Manipulierte Teile/Ersatzteile • Verwundbarkeiten in Software • Verwundbarkeiten im Netzwerkdesign • Unbeabsichtigter Datenabfluss • Physische Manipulation der Daten

Beispiel Checkliste

Annex UNECE R155 (Zulassungsvorschrift Cybersicherheit für Fahrzeuge)

Bedrohungen für Backend Servern für Fahrzeuge
• Kompromittierter Backend Service • Eingeschränkte Verfügbarkeit • Datendiebstahl

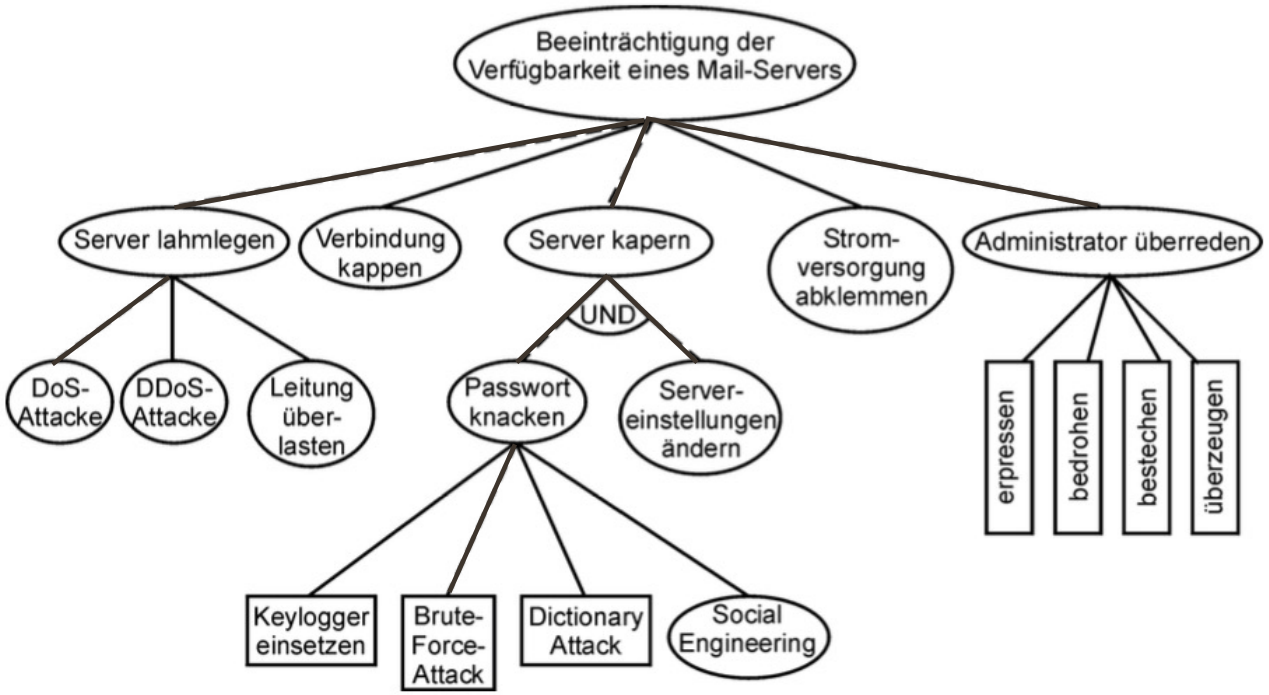
- **Datenflussdiagramme bereits in Phase 3 erstellt**
- **Betrachte Bedrohungen für diese Datenflüsse, Fokus auf**
  - Datenflüsse von externen Akteuren und Benutzern in das System („Angriffsoberfläche“)
  - Datenflüsse über eine oder mehrere Vertrauensgrenze hinweg
- **Blinder Fleck: Erkennt nur Bedrohungen für Werte, die fließende Informationen in irgendeiner Form enthalten**

Techniken zur Bedrohungsidentifikation

Angriffsbaum

- **Stellt Bedrohungen in einem Baum dar**
- **Ausgegangen wird von dem Grundwert, der geschützt werden soll (=Wurzel)**
- **Davon ausgehend werden verschiedene Bedrohungen für diesen Grundwert dargestellt (=Kinder)**
- **Kindknoten standardmäßig mit Oder-Semantik, Und-Semantik kann explizit angegeben werden**
- **Bedrohungen können weiter verfeinert werden**
- **Im Automotive Bereich aktuell Diskussion, ob dieses Modell mit den dort üblichen Fault-Trees kombiniert werden kann**

Beispiel



Quelle: [6]

- **Angelehnt an Use Cases, bekannt aus UML, Scrum.**
- **Erlauben es sehr intuitiv, Bedrohungen zu identifizieren**

- **Neue Elemente:**

Misuse Case (Aktion Angreifer)

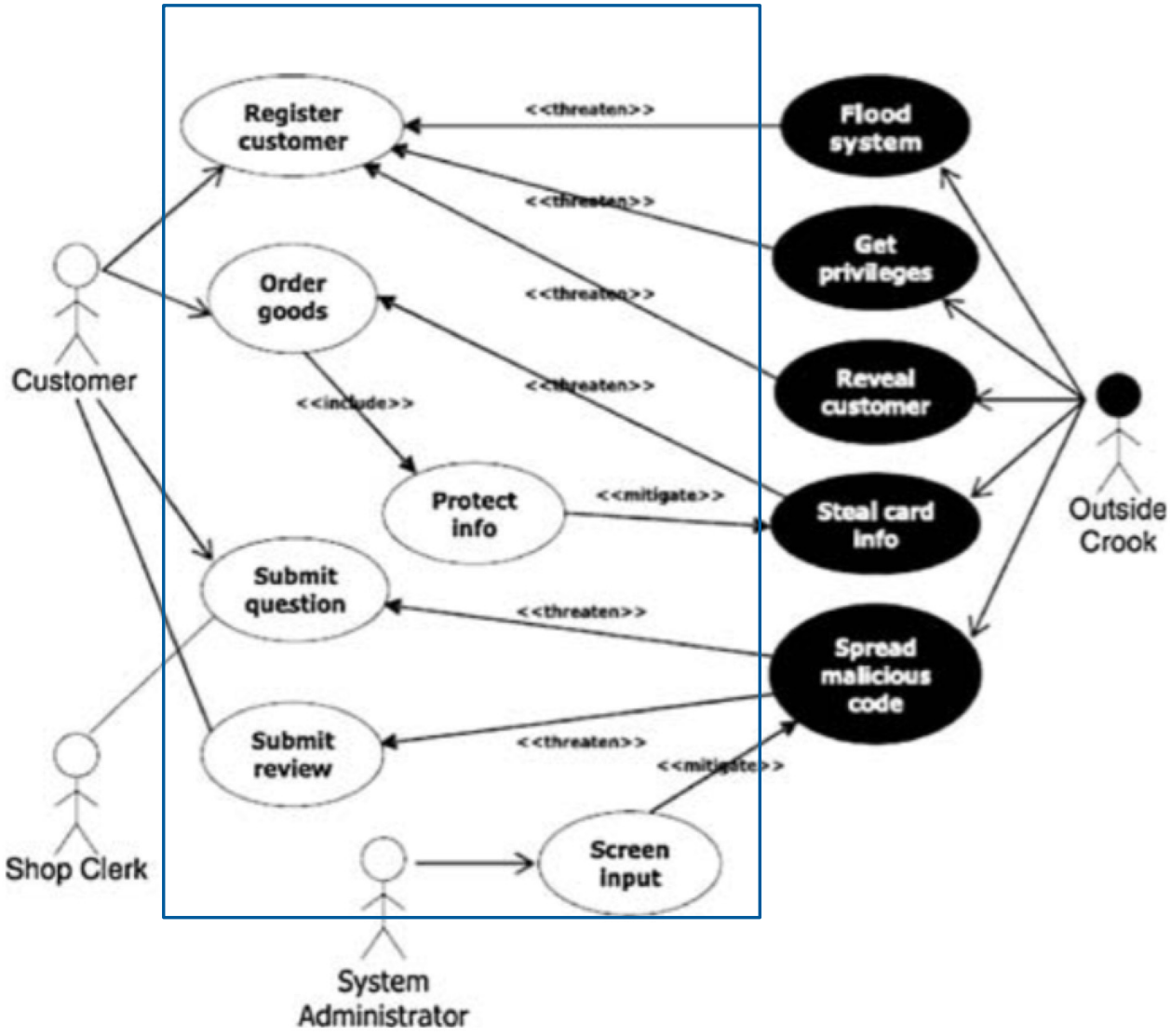
Misuser

  - Neue Stereotypen:
    - <<threaten>>: Verbindung Misuse Case u. bedrohter UseCase
    - <<mitigate>>: Verbindung Security Use Case und abgemilderter Misuse Case

- **Ansatz macht also Bedrohungen für Use Cases des Produkts sichtbar**

Beispiel

- **Quelle: [1]**

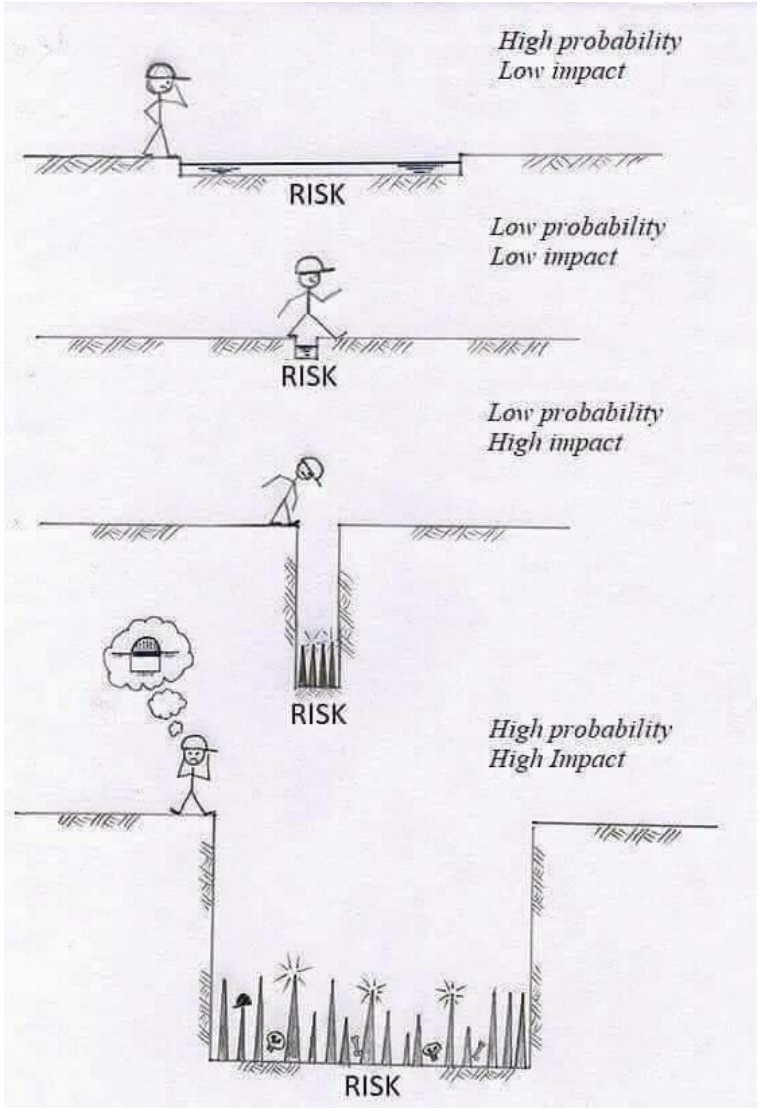


- **Nach Identifikation von Bedrohungen für Werte werden die Bedrohungen bezüglich der Relevanz bewertet und irrelevante Bedrohungen fallengelassen**
- **Hinweis: Checklisten wie OWASP Top 10 oder Annex 5 UNECE R155 müssen komplett betrachtet werden, da häufigste Bedrohungen (OWASP) bzw. gesetzlich verpflichtend (R155)**
- **Ergebnis: Pro Wert eine Liste mit Bedrohungen, die diesen Wert gefährden**

- **Der Fokus in dieser Phase wird auf (realistische) Bedrohungen für Werte gelegt, die im Folgenden weiter betrachtet werden können.**

- **Bewertung der Bedrohungen ermöglicht einen fokussierten Einsatz von Ressourcen**
- **Bewertet werden soll das Risiko einer Bedrohung für einen Wert**
- **Aus dem Risk Management: Risiko = Eintrittswahrscheinlichkeit x Schadenshöhe**
  - Jedoch: wie berechnet man die Eintrittswahrscheinlichkeit? Wie berechnet man die Schadenshöhe für alle Angriffe?

Risiko


High probability
Low impact
RISK
Low probability
Low impact
RISK
Low probability
High impact
RISK
High probability
High Impact
RISK

OWASP Risk Rating

Übersicht

- **Entwickelt der OWASP, siehe** https://owasp.org/www-community/OWASP_Risk_Rating_Methodology
  - Selbstlernaufgabe: Webseite anschauen und für die folgende Übung anwenden

- **Idee:**
  - Bestimme Risiko pro Bedrohung eines einzelnen Werts
  - Bestimme Eintrittswahrscheinlichkeit der Bedrohung über Fähigkeiten des wahrscheinlichen Angreifers und Schwere der Verwundbarkeit, welche der Bedrohung zugrunde liegt.
  - Bestimme Schadenshöhe der Bedrohung über die verletzten Schutzziele und die wirtschaftlichen Auswirkungen.

OWASP Risk Rating

Aufbau

- **Vier Themengruppen**
  - Threat Agent
  - Vulnerability
  - Technical Impact
  - Business Impact
- **Pro Themengruppe vier Faktoren**
- **Bewertung jedes Faktors zwischen 0 und 9**
  - Wert 0: bester Wert aus Sicht des Systems (Best Case)
  - Wert 9: schlechtester Wert aus Sicht des Systems (Worst Case)
  - Interpretation für jeden Faktor: siehe Veröffentlichung OWASP

- **Annahmen über wahrscheinlichen Angreifer für diese Bedrohung**
  - Wenn mehrere Angreifer wahrscheinlich: immer stärksten Angreifer betrachten

- **Faktoren:**
  - Skill Level: Welchen Wissensstand hat der Angreifer?
  - Motive: Wie stark ist Motiv Angriff durchzuführen?
  - Opportunity: Wie anspruchsvoll ist es für den Angreifer bezüglich Ressourcen und Zugang, die Verwundbarkeit zu entdecken und den Angriff durchzuführen?
  - Size: Wie groß ist Gruppe der möglichen Angreifer dieser Art?

- **Annahmen über Schwachstelle, welche die Bedrohung zu einer Gefährdung werden lässt**

- **Faktoren:**
  - Ease of Discovery: Wie leicht kann die Schwachstelle durch angenommenen Threat Agent gefunden werden?
  - Ease of Exploit: Wie leicht kann Schwachstelle genutzt werden?
  - Awareness: Wie bekannt ist die Schwachstelle
  - Intrusion Detection: Wie leicht wird der Angriff bemerkt?

- **Annahmen über das technische Schadensausmaß, falls Gefährdung wirksam wird**

- **Faktoren:**
  - Loss of Confidentiality: Wie viele Informationen würde veröffentlicht werden und wie vertraulich sind diese Informationen?
  - Loss of Integrity: Wie viel Information könnten korrumpiert werden?
  - Loss of Availability: Wie stark ist Verfügbarkeit eingeschränkt und wie wichtig ist diese?
  - Loss of Accountability: Können die Aktionen eines Angreifers einem Individuum zugeordnet werden?

- **Annahmen über das betriebswirtschaftliche Schadensausmaß falls Gefährdung wirksam wird**

- **Faktoren:**
  - Financial Damage: Welcher finanzielle Schaden entsteht durch den Angriff?
  - Reputation Damage: Welcher Imageschaden entsteht?
  - Non-compliance: Wie schwer wiegt ein Compliance Verstoß durch einen Angriff?
  - Privacy Violation: Wie viele personenbezogene Informationen werden veröffentlicht durch Angriff?

OWASP Risk Rating

Risikoermittlung

- **Gesamtwahrscheinlichkeit: Mittelwert Faktoren Threat Agent und Vulnerability**
- **Gesamtschaden: Mittelwert Faktoren Technical und Business Impact**
- **Für Gesamteinschätzung: Verwende nur Werte niedrig (<3), mittel (3<=x<6) und hoch (x>=6) für Gesamtschaden und -wahrscheinlichkeit**

Gesamtschaden \ Gesamtwahrscheinlichkeit	NIEDRIG	MITTEL	HOCH
NIEDRIG	NIEDRIG	NIEDRIG	MITTEL
MITTEL	NIEDRIG	MITTEL	HOCH
HOCH	MITTEL	HOCH	KRITISCH

- **Da Eintrittswahrscheinlichkeit und Schadenshöhe nicht direkt bestimmt werden kann, verwendet das OWASP Risk Rating indirekte Mittel zur Abschätzung dieser Werte**
- **Die Risikoeinschätzung der Bedrohungen dient der Priorisierung bei der Wahl der Schutzmaßnahmen.**
- **Nun Umgang mit Risiken möglich:**
  - Risiko akzeptieren
  - Risiko vermeiden
  - Risiko abmildern/verhindern

Phase 6

Gegenmaßnahmen planen

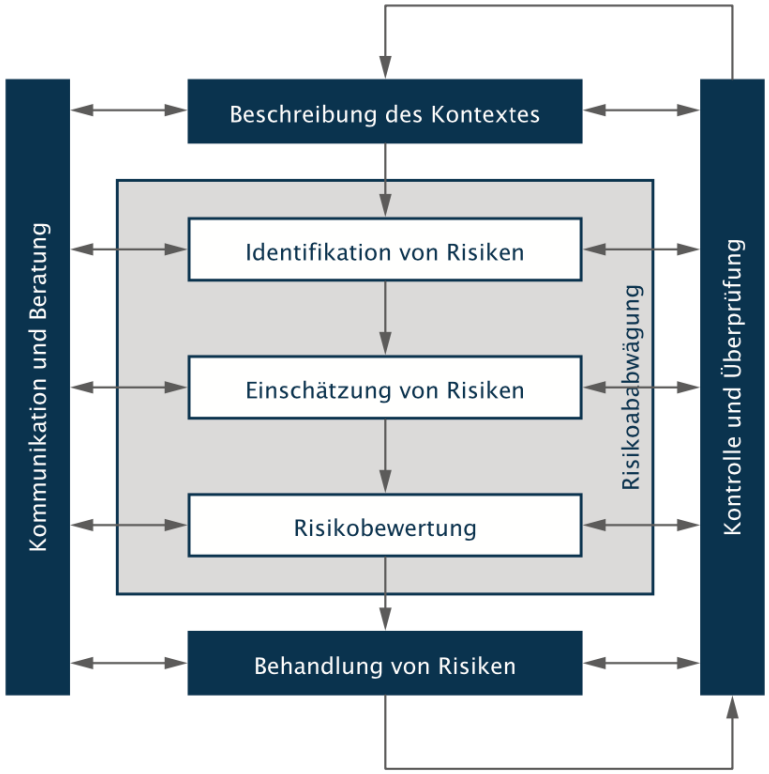
- **Angriffe nach Kritikalität ordnen (=nach Gesamtrisiko)**
- **Entscheidung, gegen welche Angriffe man schützen möchte (mindestens hoch und kritisch)**
  - Oft eine Frage des Budgets
- **Für jeden Angriff folgende Fragen stellen:**
  - Warum ist Angriff möglich? Was ist das Problem? Gewünschte Funktionalität, die ausgenutzt wird, Ursache des Problems oder logischer Fehler im Sicherheitskonzept?
  - Welche Änderung der Architektur kann das Problem lösen? Dadurch neue Schwachstellen?
  - Welche Angriffe haben die gleiche Schwachstelle im Softwareentwurf als Ursache? Können mehrere Angriffe zusammen abgewehrt werden?
  - Sind Vertrauensgrenzen neu zu ziehen?
- **Sicherheitsanforderungen formulieren, welche die hohen und kritischen Risiken behandeln**

- **Sicherheitsanforderungen als Konsequenzen aus Bedrohungen**
  - Subjektiv: Auswahl Bedrohungen, Werte, Angreifermodell [1]
  - Betrachtet vor allem Werte für den Betreiber der IT-Systeme, andere Stakeholder existieren [1]
- **Betrachtet nur bekannte Bedrohungen**
- **Motiviert nur dazu, das Nötigste zu machen (Risikobetrachtung aus wirtschaftlicher Sicht)**
- **Risikobewertung muss an Domäne angepasst werden. Beispiel: personenbezogene Daten spielen nicht in allen Systemen eine Rolle.**
- **Ketten von Vorfällen können mit der Methodik nicht bewertet werden**

Modellbasierte Risikoanalyse CORAS

Speziell für sicherheitskritische Systeme

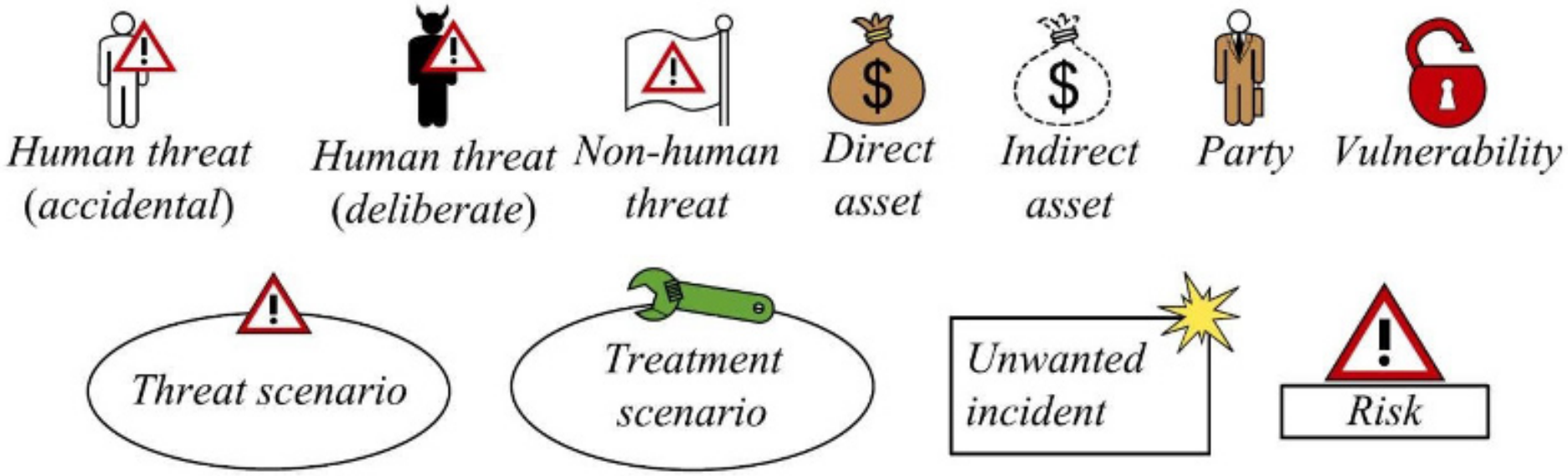
- **Grundlage: ISO 31000 Standardprozess für Risikomanagement (Grafik aus [9]):**



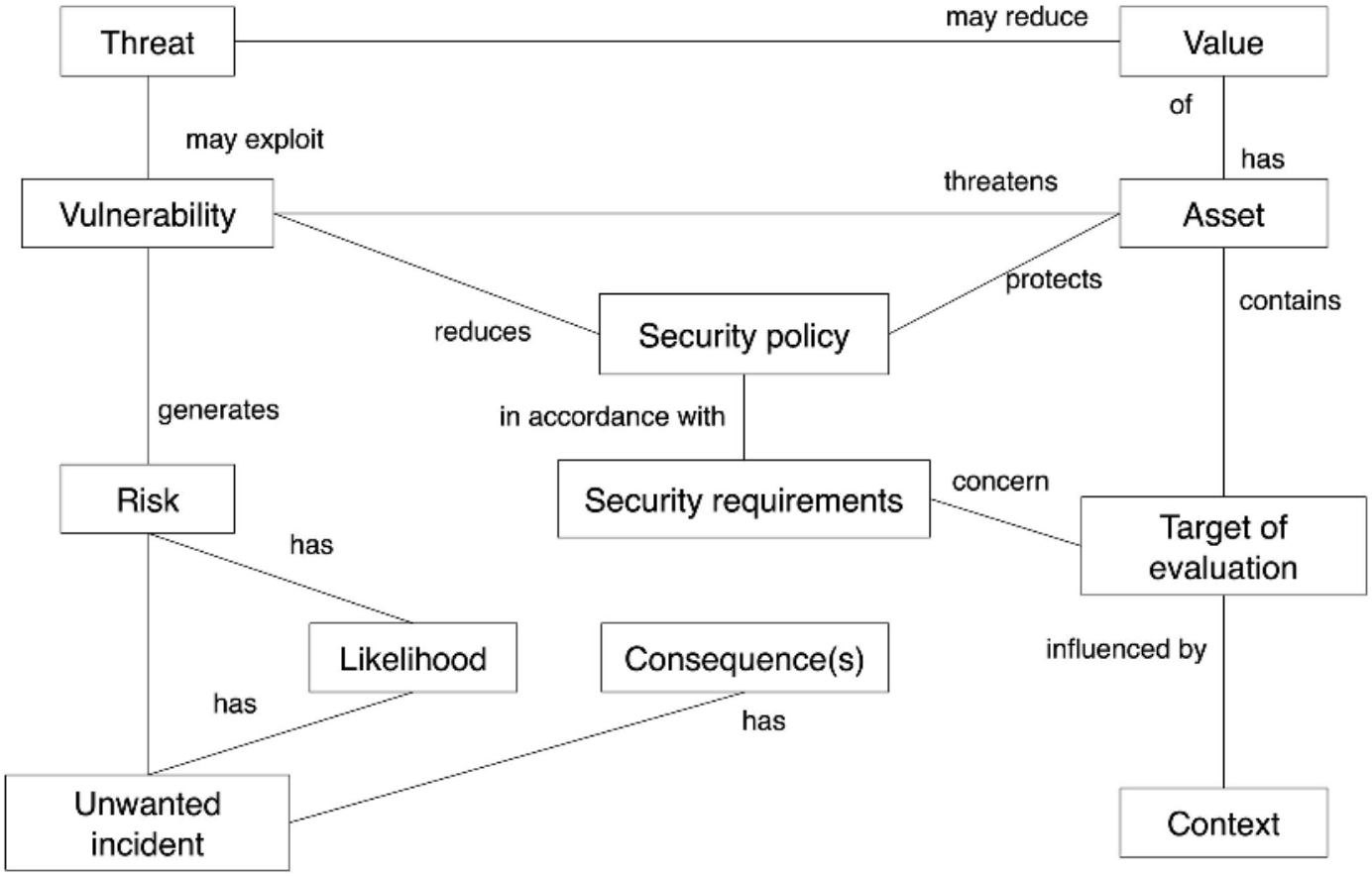
- **Threat: Potenzielle Ursache für ein unerwünschtes Ergebnis**
- **Threat Scenario: Kette oder Reihe von Ereignissen, die von einer Bedrohung ausgelöst werden und zu unerwünschtem Vorfall führen**
- **Vulnerability: Fehler oder Mangel, der Realisierung eines Threats ermöglicht oder von Bedrohung ausgenutzt werden kann, um Wert eines Assets zu schädigen**
- **Unwanted Incident: Ereignis, das Wert eines Assets beeinträchtigt**

- **Asset: etwas, dem eine Partei (Organisation/Gruppe/Institution/Person) einen materiellen oder ideellen Wert zuweist und für den die Partei Schutz benötigt**
- **Risk: Eintrittswahrscheinlichkeit eines unerwünschten Vorfalls und den daraus folgenden Auswirkungen auf einen Vermögenswert**
- **(Risk) Treatment: geeignete (organisatorische oder technische) Maßnahme zur Abwehr oder Reduzierung des Risk**

CORAS Modellelemente [9]

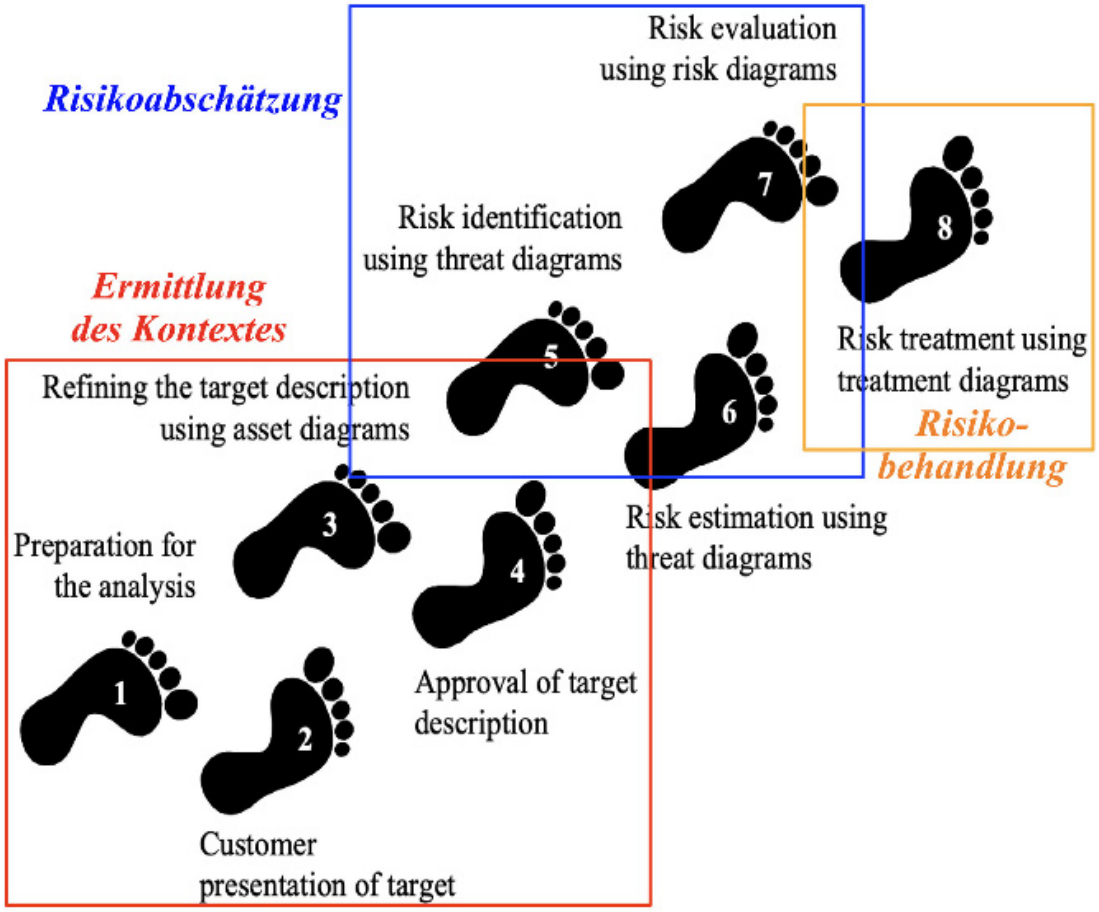


Terminologie CORAS ([9])



Coras

Vorgehen [9]

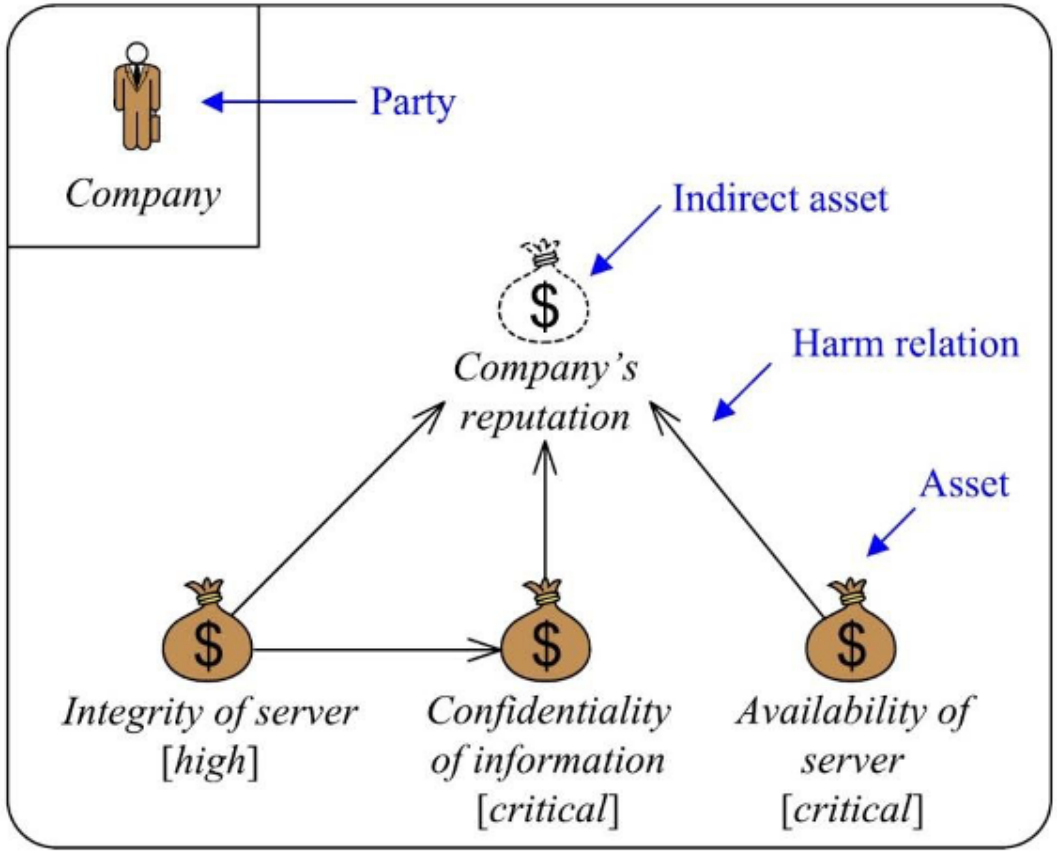


- **Asset-Diagramm**
- **Bedrohungsdiagramm**
- **Risikodiagramm**
- **Behandlungsdiagramm**
- **Behandlungs-Übersicht-Diagramm**

Asset Diagramm [9]

Beispiel

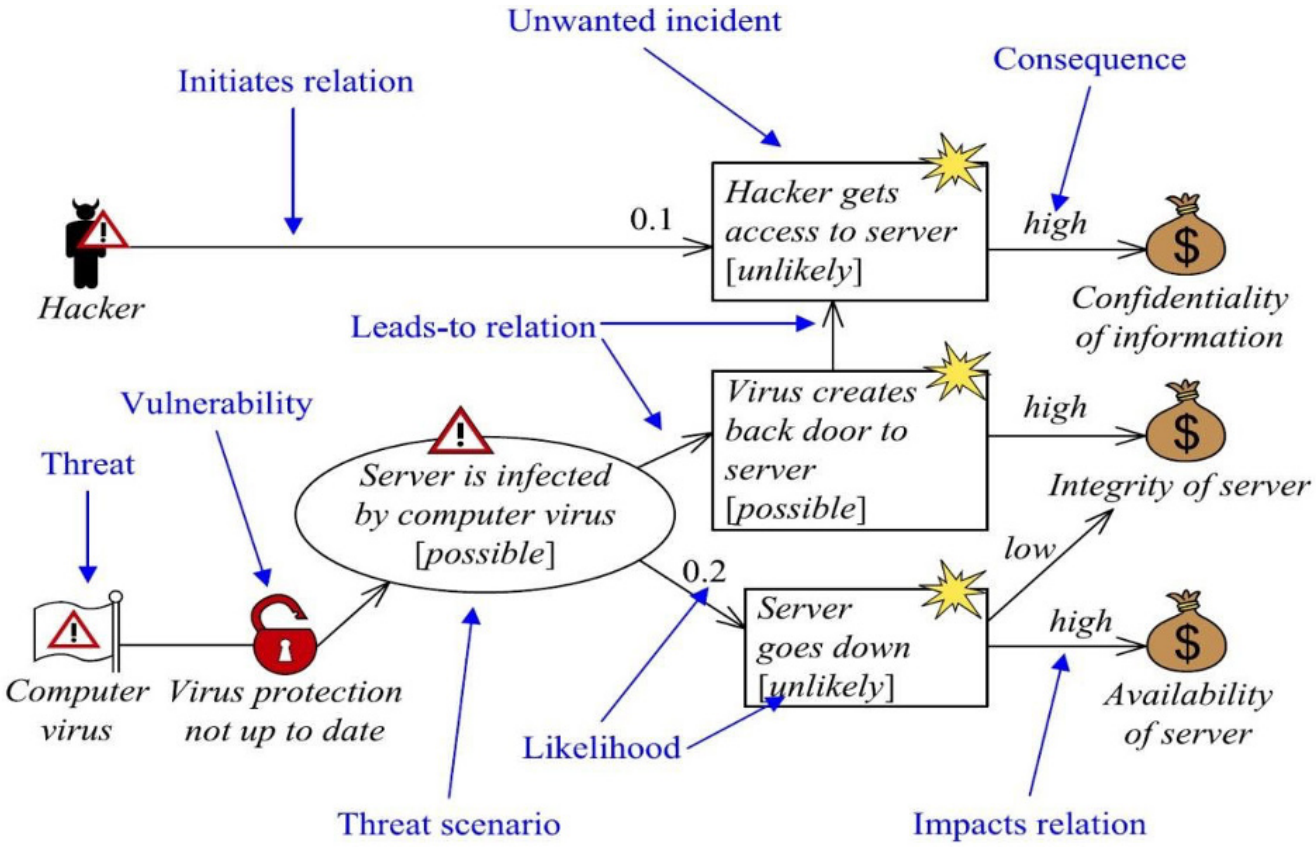
- **Die „Harm-Relation“ gibt Abhängigkeiten zwischen Assets an**
  - Beispiel: Wird die Verfügbarkeit des Servers beeinträchtigt, wird damit die Reputation des Unternehmens beeinträchtigt



Bedrohungsdiagramm [9]

Beispiel

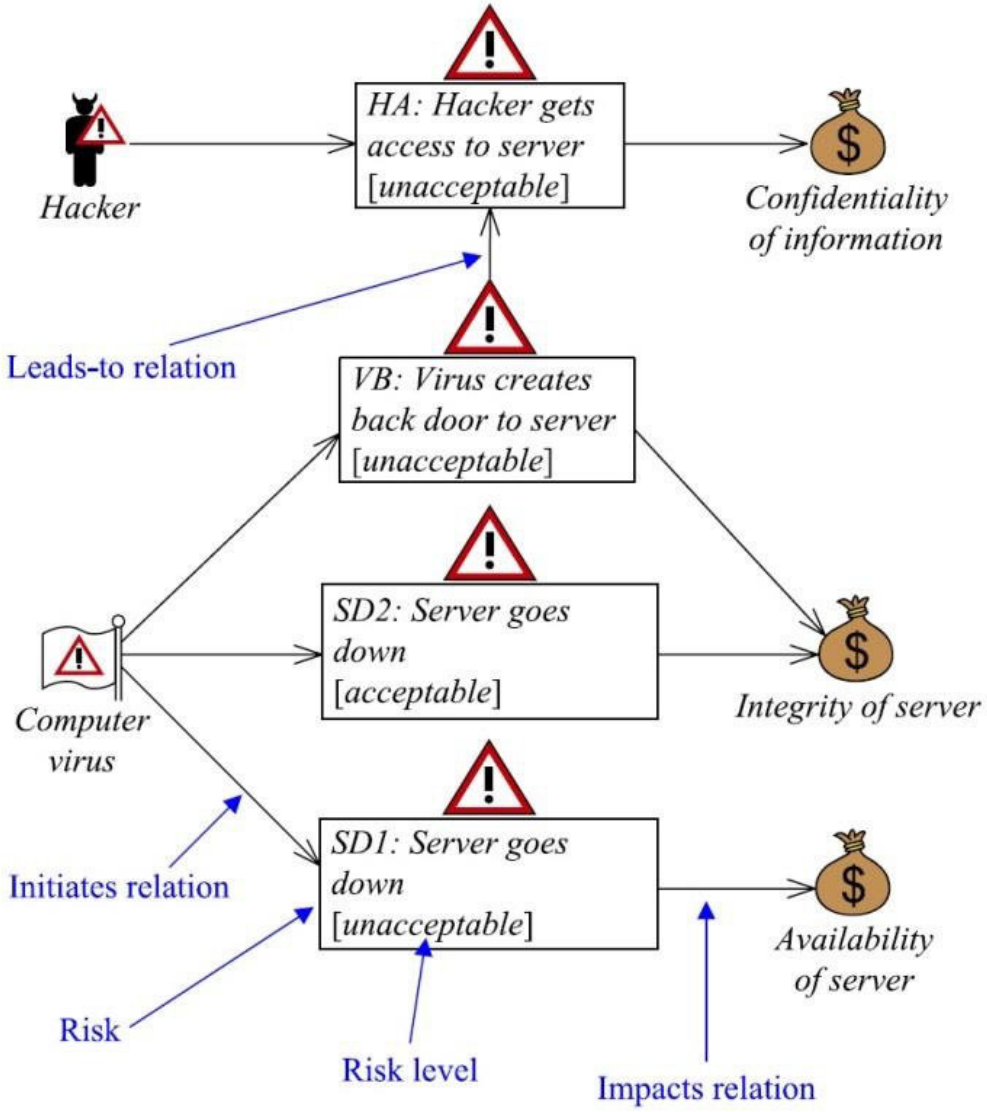
- **Bedrohungsdiagramm setzt Angreifer mit durch sie ausgelösten ungewünschten Aktionen und den daraus resultierenden Risiken in Verbindung**



Risikodiagramm [9]

Beispiel

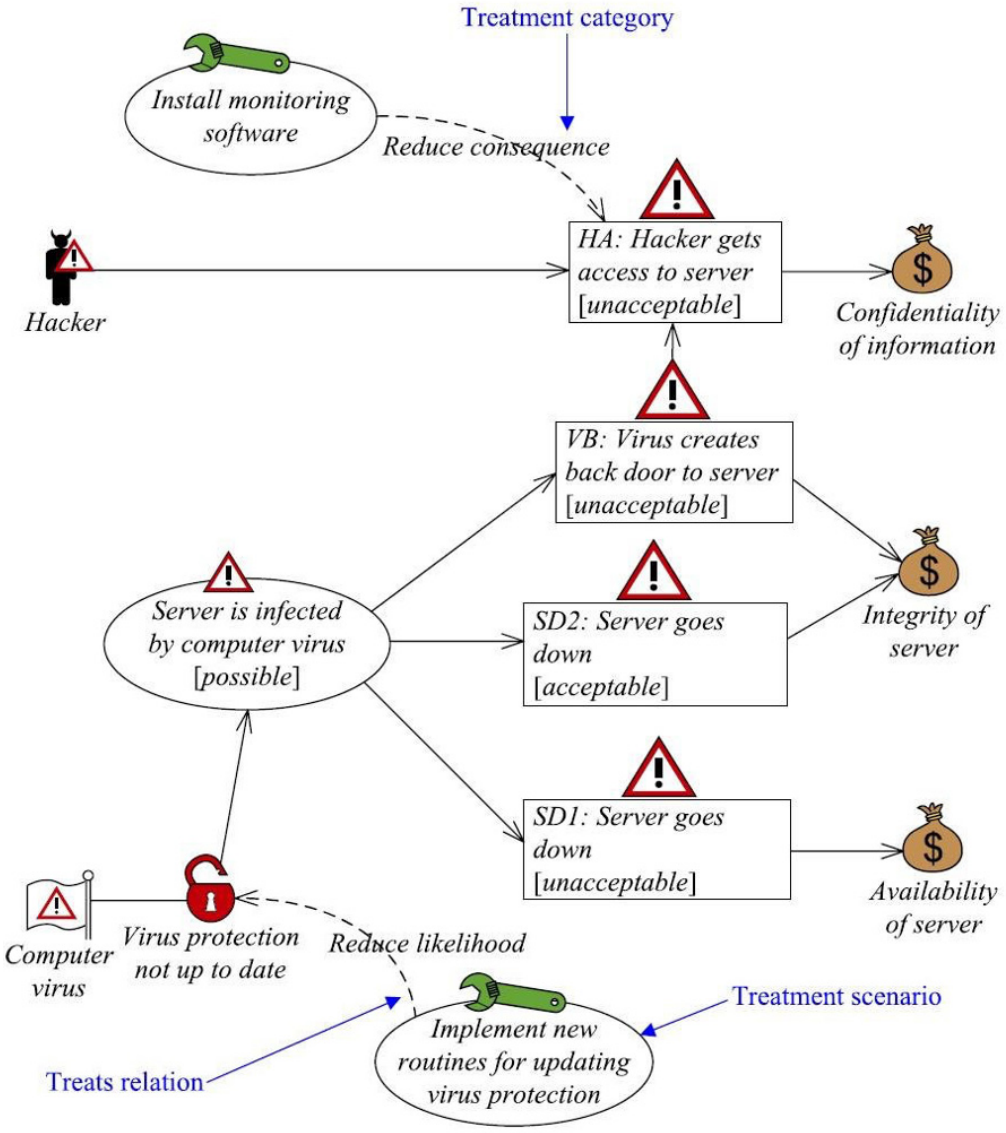
- **Das Risikodiagramm setzt Angreifer mit den durch sie ausgelösten Risiken und den dadurch bedrohten Werten in Verbindung**



Behandlungsdiagramm [9]

Beispiel

- **Das Behandlungsdiagramm zeigt, wie mit ungewünschten Vorfällen umgegangen wird, es setzt also die Sicherheitsmaßnahmen in Verbindung mit Bedrohungen**



Bestimmung des Risikos

Beispiel

- **Risk = Likelihood x Impact**

Schadensklassen für eine Online-Plattform

Auswirkung	Beschreibung
Katastrophal	Ausfallzeit $\geq$ 1 Woche
Groß	Ausfallzeit 1 Tag bis zu 1 Woche
Moderat	Ausfallzeit 1 Stunde bis zu 1 Tag
Gering	Ausfallzeit 1 Minute bis zu 1 Stunde
Vernachlässigbar	Ausfallzeit bis zu 1 Minute

Wahrscheinlichkeits-kategorien für Denial-of-Service Angriffe

Wahrscheinlichkeit	Beschreibung
Sicher	[>5] Vorfälle p.a.
Wahrscheinlich	[2, 5] Vorfälle p.a.
Möglich	[1, 2] Vorfälle p.a.
Selten	[< 1] Vorfälle p.a.
Unwahrscheinlich	[<1] Vorfälle in 5 Jahren

Risikoberechung über Matrix

Literatur

[1] Benjamin Fabian, Seda Gürses, Maritta Heisel, Thomas Santen, Holger Schmidt, „A comparison of security requirements engineering methods“, in „Requirements Engineering“, Volume 15, No 1, Springer Verlag, ISSN 0947-3602, 2010

[2] Sedan Gürses, Thomas Santen, „Contextualizing security goals—a method for multilateral security requirements elicitation“ In: Dittmann J (ed) Proceedings of Sicherheit 2006—Schutz und Zuverlässigkeit, ser. Lecture notes in Informatics. Gesellschaft für Informatik, 2006, pp 42–53

[3] Gürses S, Jahnke JH, Obry C, Onabajo A, Santen T, Price M, “Eliciting confidentiality requirements in practice“, In: CASCON ’05: Proceedings of the 2005 conference of the centre for advanced studies on collaborative research. IBM Press, 2005, pp 101–116

[4] Onabajo A, Weber-Jahnke J, „Stratified modeling and analysis of confidentiality requirements“ In: 41st Annual Hawaii international conference on system sciences“, 2008

[5] Bundesamt für Sicherheit in der Informationstechnologie „Cyber-Glossar“, https://www.bsi.bund.de/cyberglossar

[6] Bernhard C. Witt, „Grundlagen des Datenschutzes und der IT-Sicherheit“, Vorlesung an der Uni Ulm, 2010

[7] S. Tockey, „How to Engineer Software“, WileyPublishing, 2019

[8] B.W. Boehm „A Spiral Model of Software Development and Enhancement“ *IEEE Computer*, 21(5):61–72, May 1988. 5[

[9] M.S. Lund,B. Solhaug, K. Stølen „Model-Driven Risk Analysis – The CORAS Approach“, Springer, Berlin Heidelberg, 2011.