Computer System Sicherheit
Notizen WS20/21

Felix Marx

10. November 2020

Inhaltsverzeichnis

1 Begriffe **2**
1.1 Security 2
1.2 Safety 3

2 Verlässliche Systeme **3**
2.1 Strategien zur Fehlervermeidung/toleranz 4

1 Begriffe

Unterscheidung Betriebssicherheit (safety) und Angriffssicherheit (security).
safety bezeichnet den Schutz vor inneren Fehlern, deren Eintrittswahrscheinlichkeit durch probabilistische Techniken ermittelt wird.
security den Schutz gegen aktive Angreifer.

1.1 Security

Trends und Herausforderungen:
Mobilität, der Übergang zu "smart devices"

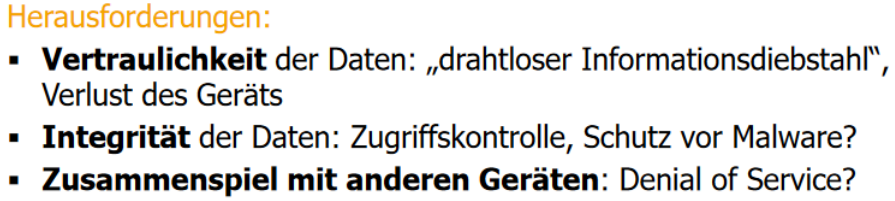


Abbildung 1: Herausforderungen bei der Mobilität

Vernetzung, der Übergang zu "always on" und Automatisierung von Systemen

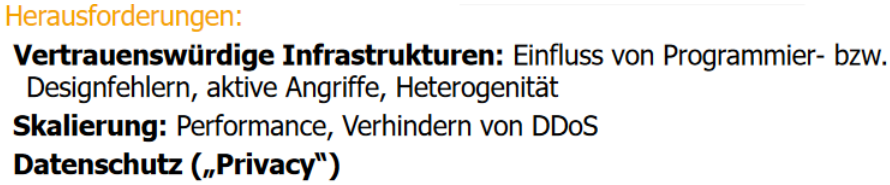


Abbildung 2: Herausforderungen bei der Vernetzung

Miniaturisierung, Produktion kleinerer Chips mit Trend zu Einwegchips

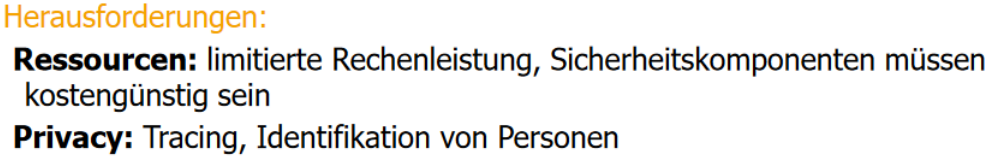
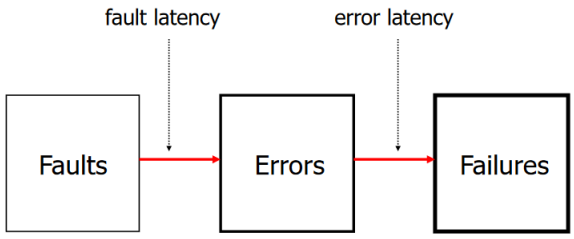


Abbildung 3: Herausforderungen bei der Mobilität

Social Engineering bezeichnet das Erlangen von vertraulichen Daten durch psychologische Techniken
Man-in-the-middle Angriffe, hier werden Nachrichten vor der gesicherten Kommunikation abgefangen.
Lösungen für Phishing umfassen die Verifikation durch Transaktionsnummern (TANs) oder Hardwaretokens was allerding nicht die Integrität der Transaktion gewährleistet.
Schutzmechanismen:
Passive Authentication: Sicherstellung der Authentizität durch Einsatz einer digitalen Signatur
Basic Access Control: elektronische gespeicherte Daten werden erst übermittelt wenn das Lesegerät einen aufgedruckten maschinenlesbaren Code kennt.



- **Fault:** Abnormal condition that can cause an element or an item to fail
- **Error:** Discrepancy between a computed, observed or measured value or condition, and the true, specified, or theoretically correct value or condition.
- **Failure:** Termination of the ability of an element to perform a function as required

Abbildung 4: Ablauf bis ein System versagt

Active Access Control: Verhindert 1:1 Kopien indem ein geheimer Schlüssel in einem gesicherten Chip-Bereich gespeichert wird.
Extended Access Control: Schutz von sensiblen Daten (Fingerabdruck, Iris, etc.) durch das Abgleichen von Zertifikaten die nur eine kurze Zeit gültig sind.

1.2 Safety

Im Fehlerfall sollte ein System immer in den sicheren Zustand übergehen.
Zum Beispiel zeigt die Fahranweisung von Zügen nach oben, sodass es bei einem Schaden am Mechanismus in den Haltezustand nach unten fällt (Schwerkraft). Ein Watchdog (hier Schwerkraft) überwacht quasi das System auf Fehler und resettet es bei einem Fehler.

2 Verlässliche Systeme

Ein zuverlässiges System erfüllt seinen Zweck auch falls Fehler auftreten. Als Beispiel des Ablaufes lässt sich die Einwirkung von kosmischer Strahlung auf eine DRAM-Zelle (Fault) was zum Wechsel eines Wertes führt (Error), wodurch die Berechnung ein falsches Ergebnis liefert (Failure).
In der Zuverlässigkeit wird unterschieden nach:
Availability: Verfügbarkeit eines Systems gemessen in Prozent, d.h.

$$\frac{\text{Total Up Time}}{\text{Total (Up + Down Time)}} = \frac{\text{MTTF}}{\text{MTTF + MTTR}}$$

Reliability: Zuverlässigkeit eines Systems, d.h. die Wahrscheinlichkeit dass ein System über einen gewissen Zeitraum korrekt funktioniert.
Wir bezeichnen mit MTTF die Mean Time To Failure und mit MTTR Mean Time To Recovery. Bei der Berechnung wird für die Up/Downtime nur die Zeiten im vereinbarten Betriebszeitraum gezählt!
Die Wahrscheinlichkeit, dass ein System bis zum Zeitpunkt t fehlerhaft wird lässt sich mittels einer Verteilerfunktion F und der Lebenszeit des Systems T berechnen:

$$F(t) = P(T \leq t),\ R(t) = P(T > t) = 1 - F(t)$$

Die Funktion R hingegen ermittelt die Wahrscheinlichkeit, dass ein System bis zum Zeitpunkt t korrekt funktioniert. Geht man davon aus, dass zukünftige Ausfälle unabhängig davon passieren, wann der letzte Ausfall war, lässt sich die Wahrscheinlichkeit mit einer

Fehlerrate λ so ausdrücken:

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0 \\ 0 & x < 0 \end{cases}$$

$$F(t) = \int_{-\infty}^{t} f(x)dx = \qquad \begin{cases} 1 - e^{-\lambda t} & t \geq 0 \\ 0 & t < 0 \end{cases}$$

$$R(t) = 1 - F(t) = \qquad \begin{cases} e^{-\lambda t} & t \geq 0 \\ 1 & t < 0 \end{cases}$$

Mean Time To Failure (MTTF) kann als **Erwartungswert** der Zeit bis zu einem Ausfall bestimmt werden:
Lebenszeit bis Ausfall ist Zufallsvariable $T \sim F$

$$MTTF = E[T] = \int_0^\infty 1 - F(t)\, dt - \overbrace{\int_{-\infty}^{0} F(t)\, dt}^{=0} = \int_0^\infty R(t)\, dt$$

Für das exponentielle Modell gilt daher: $MTTF = \int_0^\infty e^{-\lambda t}\, dt = \frac{1}{\lambda}$

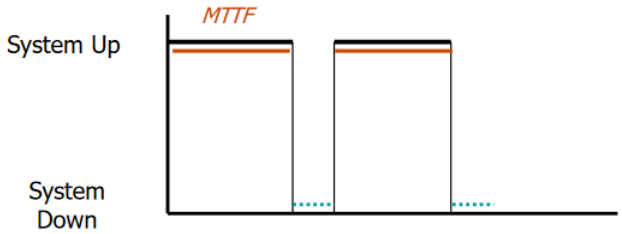


Abbildung 5: Berechnung MTTF mit dem exponentiellen Model

Bei in Reihe geschalteten Komponenten wird die Wahrscheinlichkeit dass das gesamte System korrekt funktioniert R durch $R_{ges}(t) = \prod_{i=1}^{n} R_i(t)$ im exponentiellen Modell wird dann für $\lambda = \lambda_{ges} = \sum_{i=1}^{n} \lambda_i$ verwendet.
Bei Parallelschaltung ist die Wahrscheinlichkeit dass das System korrekt funktioniert:

$$R_{ges}(t) = R_1(t) + R_2(t) - R_1(t) \cdot R_2(t), F_{ges}(t) = F_1(t) \cdot F_2(t)$$

Bei mehr als zwei Komponenten ist die allgemeine Formel:

$$R_{ges}(t) = 1 - \prod_{i=1}^{n}(1 - R_i(t)), F_{ges}(t) = \prod_{i=1}^{n} F_i(t)$$

2.1 Strategien zur Fehlervermeidung/toleranz

- Fehlervermeidung (fault avoidance)
  - Design des Systems stellt sicher, dass Fehler nicht auftreten
  - Beispiel: Testen, Verifikation,...
- Wiederherstellung aus Fehlerzustand (fault recovery)
  - Strategien, um ein System beim Auftreten eines Fehlers wieder in einen korrekten Systemzustand zu bringen

- Fehlertoleranz (fault tolerance)
  - Wenn Fehler nicht vermieden werden können, dann soll das System Fehler tolerieren
  - Beispiele: Redundanz, Safety, ...

Arten von Redundanzen:

- Physikalische Redundanz (physical redundancy)
  - Zusätzliche Ressourcen bzw. Komponenten
  - Berechnungen werden auf mehreren Komponenten ausgeführt und verglichen
  - Statische Redundanz
    - N Systeme laufen parallel
    - $N-1$ Systeme laufen im "stand-by-mode"
    - Bei einem Fehler wird im Betrieb umgeschaltet
    - Dazu muss man den Fehler natürlich zuerst erkennen!
  - Dynamische Redundanz
    - N Systeme laufen parallel
    - Ausfallsicherhere Komponente vergleicht die Resultate
    - Mehrheitsentscheidung (zB. 2-out-of-3)
    - Erkennt Fehler "automatisch"
- Zeitliche Redundanz (temporal redundancy)
  - Berechnungen auf gleicher Hardware-Plattform wiederholen
- Redundanzen durch (Zusatz-)Information (information redundancy)
  - Hinzufügen von zusätzlichen Daten (Checksummen, ...)
  - Erlaubt Datenfelder bei Übertragung oder Speicherung zu erkennen

Bei dynamischer Redundanz wird das Ergebnis parallel berechnet und dann per Mehrheitsentscheidung das korrekte ausgewählt. Problematisch falls im Vergleich ein Problem auftritt. Softwarefehler sind von der Hardware-Redundanz nicht abgedeckt. Zusatzinformationen können die Integrität gesendeter Daten durch Checksummen sicherstellen, allerdings auch nur begrenzt.

Nachteile von Redundanzen sind:

- Schlechtere Performanz (bei temporaler Redundanz)
- Synchronisation erforderlich
- Hohe Kosten durch mehrfache Hardware bzw. mehrfache Implemenation (falls überhaupt möglich)
- Benötigt Mechanismen zur Fehler-Erkennung, welche selbst wieder Fehleranfällig sein können