Hausarbeit

des Studiengangs Informationstechnik

der Dualen Hochschule Baden-Württemberg Mannheim

Sicherheit und Fehlermodelle von verteilten Systemen

Julian Fuchs, Marius Bröcker, Sebastian Wallat

November 15, 2020

Bearbeitungszeitraum:	24.10.2020-27.11.2020
Matrikelnummer, Kurs:	1708267, TINF18-IT1
Vorlesung:	Verteilte Systeme

Eidesstattliche Erklärung

Wir versichern hiermit, dass wir diese Hausarbeit mit dem Thema: "Sicherheit und Fehlermodelle von verteilten Systemen" selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

Datum, Ort

Unterschrift

Zusammenfassung

Thema

Contents

List of Figures

Abkürzungsverzeichnis

DLR **D**eutsches Zentrum für **L**uft- und **R**aumfahrt

1 Einführung

2 Sicherheit

2.1 Schutzziele

2.2 Angriffsvektoren

2.3 Schutzmaßnahmen

2.3.1 Verschlüsselung

2.3.2 Authentisierung

2.3.3 Autorisierung

3 Fehlermodelle

Sowohl System externe als auch interne Fehlerereignisse und Störungen können die Stabilität und Verfügbarkeit eines Systems komprimieren. Zu externen Störungen lassen sich unter anderem Natureinflüsse (wie Stromausfälle) oder gezielte Attacken auf die zuvor definierten Schutzziele zählen. Interne Störungen umfassen Hardware-Probleme (wie Festplatten-Ausfälle) und Software-Probleme zählen. Verteilte Systeme sind dabei durch den physikalisch getrennten Aufbau weniger Anfällig für Komplettausfälle. In der Regel sind bei Störungen einzelne Komponenten oder Teilsysteme betroffen. Gleichzeitig kann es durch den komplexeren Aufbau zu regelmäßigeren Ausfällen kommen (wie durch Wartungsarbeiten) und Fehler in einer Komponente könnten sich auf andere übertragen. Ein wichtiger Teil eines verteilten Systems umfasst damit auch die Vorbeugung von und den Umgang mit Ausfällen von solchen System-Komponenten.

3.1 Anforderungen

Ausfälle und Störungen eines Teilsystems gänzlich vorzubeugen ist sehr schwer – wenn nicht unmöglich. Ein zentrales Ziel muss daher sein, ein verteilten Systems so aufzubauen, dass es eine gewissen Fehlertoleranz besitzt. Die Fehlertoleranz beschreibt die Eigenschaft eines Systems bei Störungen eine korrekte Funktionsweise aufrecht zuhalten, ohne

Anforderung	Beschreibung
Fehlererkennung	Um fehlertolerant zu sein, muss ein System Ausfälle erkennen und auf diese reagieren können
Verfügbarkeit	Das System und dessen Dienste sind stets verfügbar und erreichbar
Zuverlässigkeit	Ein System wird durchgehend fehlerfrei ausgeführt
Funktionssicherheit	Störungen im System komprimieren nicht die Sicherheit von Daten oder führen nicht zu ungewollten Aktionen und Ergebnissen
Wartbarkeit	Beschreibt den Aufwand der Wartung eines Systems

Table 3.1: Anforderungen an ein System, um fehlertolerant zu sein (soll Tabelle bestehen bleiben oder in Klartext umgewandelt werden?)

schwerwiegende Leistungsminderungen einbüßen zu müssen. Um dieses Ziel gewährleisten zu können, muss ein verteiltes System mehrere Anforderungen erfüllen. Zu diesen zählen die Fähigkeiten des Systems einen Fehler zu erkennen, eine konstante Verfügbarkeit dieses, eine gewisse Zuverlässigkeit und Funktionssicherheit des Systems. Zuletzt spielt die Wartbarkeit eines solchen Systems ohne Unterbrechungen eine wichtige Rolle, damit ein System fehlertolerant und verlässlich sein kann. Die Anforderungen sind in Table 3.1 erneut aufgeführt und erläutert [1].

3.2 Arten von Fehlern und Störungen

Prinzipiell lassen sich die Störung, die ein verteiltes Systems erfahren kann, in fünf verschiedene Ausfallarten unterteilen: Absturz, Dienstausfall, zeitbedingter Ausfall, Antwortfehler und byzantinischer Ausfall [2][1].

- Beim *Absturz(-Ausfall)* stoppt die Ausführung des Systems. Bis zum entsprechenden Zeitpunkt hat dieses jedoch korrekt gearbeitet.
- Im Falle des *Dienstausfalls* reagiert ein Dienst nicht auf eingehende Befehle. Es wird dabei unterschieden, ob das System keine eingehende Anforderung erhält (Empfangsausfall) oder keine Antwort aussendet (Sendeauslassung).
- Von einem *zeitbedingten Ausfall* ist die Rede wenn ein System – auch Server – nicht im gegebenen Zeitintervall antwortet.
- Beim *Antwortfehler* ist die gesendete Antwort falsch. Es wird dabei unterschieden zwischen Wertfehlern (Der Wert der Antwort ist falsch) und Zustandsübergangsfehler, in dem das System vom Programmablauf abweicht.
- Zuletzt beschreibt ein *byzantinischer (zufälliger) Ausfall* den Fall, wenn ein System beliebige Antworten zu beliebigen Zeiten versendet.

Ferner lässt eine Störung, die einen solchen Ausfall verursachen kann, anhand einer zeitlichen Komponente einordnen. Dabei wird zwischen transiente, sich wiederholende und permanente Fehler unterschieden [2].
Transiente Fehler beschreiben dabei einmalig auftretende Störungen. Da diese oft nur unter einzigartigen oder sehr bestimmten Voraussetzungen eintreten, sind solche Fehler oft schwer vorherzusagen. Wiederholende Fehler treten periodisch oder bei bestimmten Voraussetzungen auf, die regelmäßig erfüllt werden. Permanente Fehler beschreiben Störungen, die nach Eintritt bestehen bleiben. In der Regel muss die betroffene Komponente repariert bzw. ersetzt wird.

3.3 Fehlerbehebung

Verteilte Systeme besitzen durch die dezentrale Natur viele Fehlerquellen. Daher sollte beim Aufbau eines solchen Systems besonderen Wert auf die Fehlertoleranz gelegt werden, um stets eine konsistente Funktionsweise gewährleisten zu können.
Das generell Vorgehen sollte dabei sein den Fehler oder die Störung zu maskieren und vor anderen Komponenten (wie dem Client) zu verbergen. Systemintern kann dann der Ausfall behoben werden – im Idealfall ohne Beeinträchtigung einer außenstehenden Komponente. Dabei spielt vor allem die Redundanz im System eine wichtige Rolle. Es wird dabei zwischen Informationsredundanz, zeitlicher und physischer Redundanz unterschieden.

- *Informationsredundanz* beschreibt die Eigenschaft eines Dienstes notwendige Informationen zu übertragen, obwohl Informationsteile auf dem Transportweg verloren gehen. Ein klassisches Beispiel wäre die Verwendung eines geeigneten Kodierungsverfahrens, um Bit-Fehler erkennen und korrigieren zu können (vgl. Hemming-Codes) (Quelle: https://de.wikipedia.org/wiki/Hamming-Code).
- Bei *zeitlicher Redundanz* kann ein fehlgeschlagener Prozess erneut durchgeführt werden. Ein Beispiel eines solchen Prozesses könnten Transaktionen wie `PUSH`- und `GET`-Anfragen an einen `HTML`-Server sein.
- *Physische Redundanz* zeichnet sich durch zusätzliche, dedizierte Hardware aus. Diese könnte in Form von vollständigen Servern oder einzelnen Komponenten sein, wie zusätzlicher Festplatten, die in einem Verbund – beispielsweise als *RAID*-System – geschaltet sind, um eine Ausfallsicherheit zu erhöhen. Mit einem solchen System könnten zudem eine erhöhte Datendurchsatzrate erzielt werden (Quelle: https://de.wikipedia.org/wiki/RAID).

- auf Replikation eingehen
- wie können einzelne Ausfallfälle behandelt werden ...

Bibliography

[1] Dr. C. Werner, "Verteilte systeme – 7. fehlertoleranz," Sommersemster 2011. Vorlesungs-Skript.

[2] Dr. Dillinger, "Verteilte systeme," Wintersemester 2020. Vorlesungs-Skript.

[3] A. Schill and T. Springer, "Verteilte systeme," *eXamen-press*, 2012.