Hier soll der Titel hin

Niklas Weimann

8. Mai 2020

Inhaltsverzeichnis

1 Einleitung

Was ist Fuchsia OS? "Pink + Purple == Fuchsia (a new Operating System)"[2] So beschreibt Google ein neues Betriebssystem, das auf Modularität und Was sind die Ziele?

2 Zircon

Fuchsia setzt auf den Microkernel Zircon, der vor seiner Umbenennung Magenta hieß. Zircon entstand aus Little Kernel, einem Projekt von Travis Geiselbrecht. Der Kernel wurde so designet, dass er sowohl auf IoT Hardware, als auch auf Desktop Computern eingesetzt werden kann. [1] Die Systemaufrufe, die Zircon zur Verfügung stellt, sind bis auf wenige Ausnahmen alle asynchron und blocken somit nicht den aufrufenden Thread, dies ermöglicht eine effizientere Abarbeitung der Jobs.

2.1 Kernel Objects und System Calls

Die Kommunikation zwischen Kernel Objekten, und dem Usermode findet ausschließlich über Handles statt. Handles werden von System Calls zurückgegeben, falls diese ein Kernel Objekt erstellt haben (z.b. zx_event_create() + zx_process_create() + zx_thread_create()). Im Usermode ist dieses Handle eine 32 Bit Zahl, wobei die letzten 2 LSB bei validen Handles immer gesetzt sind und die Zahl 0 ein ungültiges Handle repräsentiert. Handle hat nur für den jeweiligen Prozess eine Bedeutung, der es angefordert hat, in einem anderen Prozess kann dieses Handle auf ein anderes Kernel Objekt zeigen oder ungültig sein. Ein Prozess kann mehrere Handles haben, um auf das selbe Kernel Objekt zuzugreifen, wobei sich hierbei dann meist die jeweiligen Zugriffsrechte unterscheiden. Die Zugriffsrechte, die ein Handle auf ein Kernel Objekt hat, werden im Kernel gespeichert. Der Kernel hat für jedes Handle ein C++ Objekt, das erstens eine Referenz auf das Kernel Objekt hat, zweitens eine Liste mit allen Rechten die das Handle hat und drittens eine Referenz auf den Prozess, zu dem es gehört. Wie funktioniert die Rechte Verteilung? Welche Vorteile hat es, dass alle Aufrufe (außer Wait) Async sind? Wie funktionieren Syscalls im Gegensatz dazu im Linux (Android) Kernel?

2.2 Scheduling

Vergleich zwischen Fuchsia (Fair Scheduler) und Linux (Android) Scheduling

3 Komponenten v2

3.1 Komponenten Einführung

Was ist eine Komponente? Was ist das Komponenten Manifest?

3.2 Komponenten Manager

Was sind die Aufgaben des Komponenten Manager?

3.3 Capabilities

Was sind Capabilities (Service, Protocol, Directory, Storage)? Was ist Capability Routing und wie funktioniert es?

3.4 Vergleich zu Komponenten

Welche Vorteile bietet es, dass alles eine Komponente ist im Gegensatz zu Linux (Android)?

4 Treiber

4.1 Verwaltung und Zugriff

Wie werden Treiber geladen? Wie wird er Zugriff auf die Geräte ermöglicht?

4.2 Vergleich Treiber

Wie werden Treiber unter Linux (Android) verwaltet?

5 Dateisystem

5.1 Komponentenbasierter Speicher

Wie sind Dateisysteme und das Dateimanagement realisiert?

5.2 Cloudunterstützung

Was ist Ledger?

5.3 Vergleich Filesystems

Wie unterscheidet sich das Dateisystem von Linux (Andorid)?

6 Zusammenfassung

Grobe Zusammenfassung über die Unterschiede zu Linux (Android)

Literatur

[1] Dave Altavilla. Google's mysterious fuchsia os developer site debuts with new fascinating details. *Forbes*, 2019, 30. Juni 2019.

[2] Google LLC. Get started.