Untersuchung der Performanz beim Transfer unterschiedlicher Datenmengen und Größen zur Google Cloud Storage Plattform

Victor Cranz Matrikelnr. 727519, Pascal Bechtoldt Matrikelnr. 734060

May 18, 2017

1 Einleitung

Im Rahmen des Praktikums der Vorlesung "Speicher und Datennetze im IoT" wird die Plattform Google Cloud Storage (GCS) untersucht. Googles Plattform bietet eine große Bandbreite unterschiedlicher Lösungen um Daten verteilt in der Cloud abzulegen. Dieses Experiment fokussiert sich speziell auf das "Cloud Storage" Produkt Googles. Google bietet mit diesem Produkt eine Speicherplattform, welche regional, wie auch multiregionale Datenspeicherung ermöglicht. Bekannte Firmen, welche z.B. Steamingdienste anbieten, wie z.B. Spotify verwenden GCS als Speicherressource.[Pla]
Es wird die Antwort auf folgende Fragestellung gesucht:

Wird die Performance von GCS bei parallelen Zugriffen bei mehreren Verbindungen bei unterschiedlichen Datenmengen und Datengrößen beeinflusst?

2 Konzept

Um die oben genannte Fragestellung beantworten zu können, wird ein Programm entwickelt, welches in unterschiedlichen Iterationen binäre Daten an GCS transferiert.
Um GCS zu nutzen muss zuerst ein "Bucket" als Datenbehälter erstellt werden. Im Rahmen des Experiments wird ausschließlich das europäische Datencenter Googles genutzt um überregionale Latenzen zu vermeiden.
Interessante Attribute bei der Übertragung, welche im Test wechselseitig miteinander kombiniert werden sollen sind:

1. Kleine Dateien
2. Große Dateien

3. Viele Daten
4. Wenige Daten
5. Übertragung von Daten mit nur einer Verbindung (iterativ)
6. Übertragung von Daten mit mehreren Verbindungen (parallel, multithreaded)

2.1 Testdaten

Im Rahmen des Experiments sollen unterschiedlich große Datenmengen an GCS übertragen werden. Die Testdaten werden aus /dev/random mit in folgendem Abschnitt definierten Größen generiert.

2.1.1 Datengröße

Gemäß der bekannten Speichergrößengrenzen werden Datenpakete in Form von Zweierpotenzen, beginnend bei 0 genutzt. Manche Potenzen werden ausgelassen um Abweichungen signifikanter erscheinen zu lassen.
Paketgrößen: 0 Byte, 1 Byte, 512 Byte, 1 Kb, 512 KB, 1 MB, 512 MB, 1 GB, 4 GB

2.2 Datenmenge

Um zu überprüfen wie sich GCS bei der Übertragung mehrerer Daten verhält werden die zu testenden Datenmengen (Anzahl der zu übertragenden Dateien) wie folgt festgelegt:
Dateien: 0, 1, 5, 10, 100, 1000, 10000

2.3 Parallele Zugriffe

Es wird versucht zu festzustellen, wie viele simultane Verbindungen von GCM zum Datentransfer gestattet werden. Zur Referenz wird mit einer Verbindung begonnen. Bis zum Fund einer lokalen oder entfernten Limitation werden die Verbindungen dann ebenfalls in Zweierpotenzen inkrementiert.

2.4 Messungen

Die Performance von GCM wird anhand der verstrichenen Zeit zur Durchführung eines Testszenarios erfasst. Als Referenzzeit wird jeweils eine Operation in Ihrer einfachsten Form verwendet. Unter Annahme das die Dauer einer Operation proportional steigt wird in der Analyse geprüft, ob der Erwartungswert mit dem gemessenen Wert übereinstimmt.

2.5 Netzwerkverbindung

Alle erfassten Daten müssen unter Berücksichtigung und Notation der verwendeten Netzwerkverbindung betrachtet werden. Ein belegtes oder teilausgelastetes lokales Netz kann Messdaten verfälschen. Eine weitere Limitation stellt die verfügbare Bandbreite des Internet Service Providers (ISP) dar. Um aussagekräftigere Daten zu erhalten, wird

versucht die Testszenarien mit unterschiedlichen Netzwerken möglichst kabelgebunden durchzuführen.

3 Testszenarien

Zur Bestimmung der Performance von GCS definieren wir verschiedene Testszenarien, welche mit Hilfe von Java-Programmen umgesetzt und evaluiert werden. Die ersten drei Szenarien betrachten, wie sich das System bei Schreibzugriffen verhält. Dabei ist es interessant herauszufinden, ob das System bei einer großen Anzahl von Schreibzugriffen langsamer wird, oder eventuell vorhandene, überlaufende, Puffer bemerkbar werden. Das vierte Szenario beleuchtet die Performance von GCS bei lesenden Zugriffen. Insbesondere wird beobachtet, ob Cachingmechanismen verwendet werden.

3.1 Szenario: Einzelne Datensätze

Im ersten Szenario werden die Übertragungszeiten bestimmt, welche benötigt werden, um jeweils eine Datei jeder zuvor definierten Paketgrößen an GCM zu übertragen. Zwischen jeder Operation wird ausreichend Zeit gelassen, um zu Gewährleisten, das der entfernte Server die zuvor übertragene Datei fertig verarbeitet hat. Die gemessenen Werte werden gemäß der definierten Datenmengen für den folgenden Vergleich interpoliert da von einem proportionalen Zeitanstieg ausgegangen wird.

3.2 Szenario: Sequentielle Schreibzugriffe

In diesem Szenario wird das erste Szenario wiederholt, jedoch werden die Daten in diesem Experiment tatsächlich sequenziell an GCS übertragen. Die hier bestimmten Übertragungszeiten werden im Anschluss mit den interpolierten Referenzwerten verglichen.

3.3 Szenario: Parallele Schreibzugriffe

Hier wird der Umgang von GCS mit parallelen Schreibzugriffen betrachtet. Dafür werden aus der oberen Hälfte der definierten Datengrößen jeweils die spezifizierten Datenmengen über mehrere Verbindungen in die Cloud geladen. Jede Verbindung läuft in einen eigenen Thread. Auf diese Weise soll GCM bestmöglich ausgelastet und potenzielle Limitierungen aufgedeckt werden. Die maximale Anzahl von Verbindungen wird gemäß der Definition im Abschnitt "Parallele Zugriffe" durchgeführt.

3.4 Szenario: Lesender Zugriff

Im letzten Szenario wird das Verhalten bei lesenden Zugriffen betrachtet, bzw. ob GCS Caching verwendet. Dafür werden aus jeder Größenkategorie Dateien jeweils drei Mal gelesen und die verstrichene Zeit gemessen. Die zu lesenden Daten werden eine Woche vor dem Lesen an GCS übertragen, um zu vermeiden, dass diese schon beim ersten Lesezugriff in einem möglichen Cache liegen.

4 Erwartung

Unserer Erwartungshaltung ist, dass GCS darauf ausgelegt ist, mit solchen Daten umzugehen. Daher sollten die verschiedenen Testszenarien indifferent zueinander sein.

References

[Pla] Google Cloud Platform. *Cloud Storage*. `https://cloud.google.com/storage/?hl=de`. Accessed: 2017-05-1.