Web Latenz im Transmission Control Protokoll

Dane Leube

Zusammenfassung

Dieses Paper behandelt das Thema Web Latenz. Die Grundlage ist eine Analyse von Google, wonach die Web Latenz sehr stark von Paket Losses beeinflusst wird. Diese Verluste führen zum Problem, wenn eine zuverlässige Verbindung via TCP aufgebaut, da die Pakete dann erneut gesendet werden. Diese Retransmission führt zu einer sehr großen Latenz wie eine Studie herausgefunden hat. In diesem Paper werden drei Lösungsansätze vorgestellt, die versuchen die Latenz zu reduzieren mit dem idealisierten Ziel einer Paket Recovery mit einem RTT.

1 Einleitung mit Problemstellung

Verzögerungen im Web kosten laut einer Studie von Amazon pro Verzögerung von 100ms ca. ein Prozent des Profits [1]. Somit ist das Thema Web Latenz ein sehr aktuelles Thema, womit sich jeder Anbieter eines Online-Shops auseinander setzen muss. Der Nutzer erwartet von einem Webshop, dass er schnell verfügbar ist und vor allem, dass er keine langen Wartezeiten hat, wenn er beispielsweise eine Abfrage an den Server schickt. Der einfachste Ansatz der Provider ist die Erhöhung der Backbone Hardware und der Pop´s [TFGo13]

Dieser Ansatz skaliert nicht unendlich, da es weitere Engpässe gibt. Ein weiterer Lösungsansatz kann die kritische Analyse des TCP Protokolls sein. Das TCP Protokoll ist der defakto Standard bei verbindungsorientieren und zuverlässigen Verbindungen im Web. Bei der Spezifizierung dieses Protokolls wurden solch große Durchflussraten, wie sie heutzutage zustande kommen nicht berücksichtigt. Google hat dieses Problem erkannt und genauer analysiert. Es wurden große Datenmengen mittels TCP Verbindung über das Web transferiert und ausgewertet. Die Auswertung zeigt, dass bei mindestens 10 Prozent der Übertragungen ein Paket Verlust passiert. Weiterhin wurde festgestellt, dass bei Übertragungen mit mindestens einem Verlust bis zu fünf Mal länger bei der Übertragung benötigen als welche ohne Verlust. Dieses Ergebnis zeigt, wie wichtig es ist, dass die Pakete bei einer Verbindung verlustfrei übertragen werden, damit die Web Latenz verringert werden kann. Der Faktor fünf ist sehr hoch, wenn man bedenkt, dass es sich nur um ein einzelnes Paket handelt welches verloren geht und somit neu übertragen werden muss. Die Studie hat weiterhin herausgefunden, dass 77 Prozent dieser Verluste immer am Anfang einer Verbindung auftreten, sogenannte Tail-Losses.

Die Ergebnisse der Studie von Google zeigen die Wichtigkeit von verlustfreien TCP Übertragungen, damit die Web Latenz reduziert wird. An dieser Stelle setzt dieses Paper an. Das Ziel für die Optimierung ist es bei einem Paketverlust die Retransmission mit einem Versuch erfolgreich abzuschließen. Wenn also ein Paket verloren geht, dann soll dieses beim nächsten Senden ankommen. Dieses Ziel ist sehr hoch gefasst und kann wahrscheinlich nicht immer erreicht werden. Aus diesem Grund versuchen wir eine Näherung an dieses Idealziel zu erreichen. Die drei Ideen, die dieses Ziel erreichen sollen, haben unterschiedliche Ansätze. Die drei Ansätze sind aus dem Paper *Reducing Web Latency: the Virtue of Gentle Aggression entnommen.* Die Ansätze Reactive, Corrective und Proactive werden nach einem kurzen Grundlagenkapitel genauer erläutert und anschließend gegeneinander abgewägt.

[1] Quelle: http://sites.google.com/site/glinden/Home/StanfordDataMining.2006-11-28.ppt, 2006

2 Grundlagen

In diesem Abschnitt geht es um eine kurze Einführung in die TCP Kommunikation, damit die Problemstellung klarer herausgearbeitet werden kann.

2.1 Transmission Control Protokoll

Das Transmission Control Protokoll (im folgenden nur noch TCP genannt) ist ein verbindungsorientiertes Transportprotokoll. Es sitzt in der OSI-Layer Architektur auf dem 4. Layer, dem Transportlayer. Die wichtigste Eigenschaft von TCP ist seine Zuverlässigkeit. Zur Erfüllung dieser Eigenschaft werden bei TCP Pakete, die auf dem Weg zum Empfänger verloren gehen erneut gesendet. Die zweite Eigenschaft, die Verbindungsorientierung von TCP baut eine Vollduplex Verbindung zwischen beiden Kommunikationspartnern auf, die ein Senden von Paketen in beide Richtungen zulässt. Im Folgenden soll der Ablauf einer TCP Verbindung dargestellt werden und die einzelnen Phasen erläutert werden, damit das Problem der Web Latenz im TCP Protokoll genauer erklärt werden kann.

Bei TCP erfolgt ein Verbindungsaufbau über einen 3-Way-Handshake. Bei jedem Paket wird der TCP Header an das Datenpaket angehängt. Der komplette Header von TCP ist in Abbildung 1 zu sehen. Für diese Arbeit relevant ist die sogenannte Sequenznummer, die beiden Flags ACK und SYN sowie das sogenannte CTRL Bit, dass angibt um welchen Typ von Paket es sich handelt. Beim Start einer neuen Verbindung sendet der Sender ein Paket mit einem gesetzten SYN Flag und einer zufällig generierten Sequenznummer. Diese dient der eindeutigen Identifizierung eines Paketes und ist die Basis zum Erkennen eines verloren gegangen Paketes. Sie wird immer fortlaufend pro neuem Paket erhöht und bei jeder Übertragung mitgeschickt. Zur Bestätigung des erhaltenen Paketes antwortet der Empfänger mit einem gesetzten ACK und SYN Flag. Das ACK enthält die um eins inkrementierte Sequenznummer des ersten Kommunikationsteilnehmers. Für den 3-Way-Handshake wird eine erneute Bestätigung vom Sender verschickt. Die ACK ist die um eins inkrementierte Sequenznummer des vorherigen Paketes vom Empfänger. Nachdem das ACK beim Empfänger angekommen ist, beginnt die Datenübertragung. Somit ist der Verbindungsaufbau von TCP abgeschlossen. Im Folgenden können Datenpakete übermittelt werden. Wichtig hierbei ist der sogenannte Slow-Start. Damit es nicht zu Stauungen und überflüssigen Paketverlusten kommt, wird bei einer Datenübertragung nie sofort das komplette Fenster übertragen. Es wird sich langsam an das Maximum, was die MTU hergeben kann angenähert. Dieser langsame Aufbau hat den Vorteil, dass die Leitungen weniger überlastet und somit schnell abgebrochen werden kann, falls die Verbindung nicht stabil genug sein sollte. Bei großen Datenmengen kann dies jedoch zu einem Problem führen, da die Geschwindigkeit beim Übertragen der Daten deutlich reduziert ist. Aus diesem Grund muss man beim Einsatz des Slow-Starts Mechanismus immer die konkrete Problemstellung im Auge haben.

Die Zuverlässigkeitseigenschaft von TCP ist für dieses Paper von einer sehr großen Bedeutung. Die Zuverlässigkeit wird in TCP umgesetzt, indem eine erneute Übertragung des Paketes durchgeführt wird, sobald ein Paket auf dem Weg zum Empfänger verloren geht. Anhand der Sequenznummer wird ein Paketverlust identifiziert. Auf der Senderseite startet ein Timer, sobald das Paket losgeschickt wurde. Wenn der Timer abgelaufen ist und kein ACK Paket vom Empfänger erhalten wurde, wird das Paket erneut gesendet. Die Optimierung dieses Retransmission Timeouts (RTT) ist ein wichtiger Prozess bei der Optimierung von TCP.

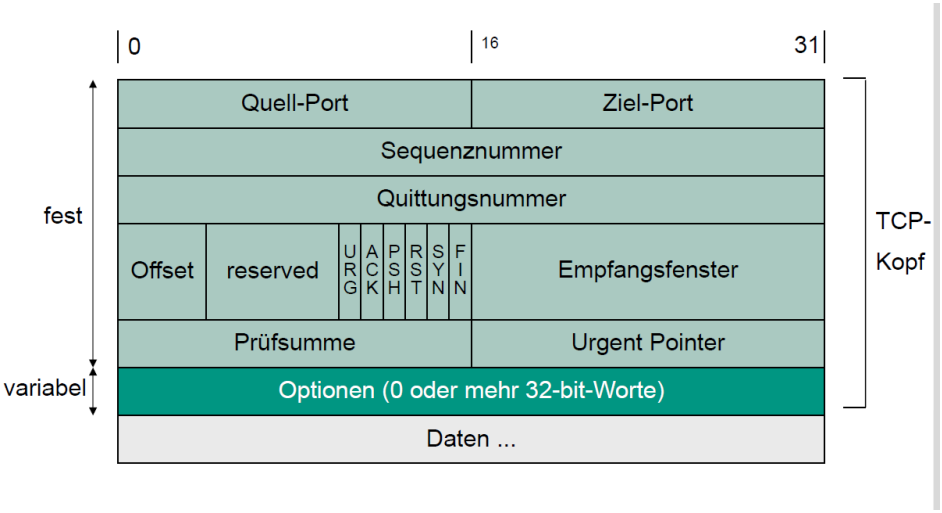

Abbildung 1: Aufbau des TCP Headers Quelle:[TELE]

2.2 Fehlerkorrigierende Codes

Kurze Erläuterung der fehlercorrigenden Codes, damit der Corrective Ansatz verstanden werden kann.

3 Googles Analyse

Nachdem die TCP Mechanismen erklärt wurden, ist die Frage nach der konkreten Problemstellung zum Thema Web Latenz zu klären. Grundlage der Analyse ist ein Forschungsergebnis von Google, wonach die Web Latenz deutlich ansteigt, je mehr Pakete auf dem Weg zum Empfänger verloren gehen. Dies ist soweit nichts neues. Interessant an der Studie ist zum Einen die Menge der Daten, die erhoben wurde und zum Anderen die Erkenntnis, dass bei vielen Paketverlusten ein exponentieller Anstieg der Web Latenz zu beobachten ist. In Abbildung 2 ist ein Balkendiagramm dargestellt, was diesen Sachverhalt verdeutlicht. Man sieht, dass bereits bei einem Paketverlust die Weblatenz um den Faktor zehn zunimmt. Weiterhin ist interessant zu sehen, dass man sehr nahe an der idealen RTT ist, wenn man keinen Paketverlust hat. Dadurch, dass bei verbindungslosen Protokollen wie UDP keine erneute Übertragung der Pakete stattfindet, begrenzt sich das Problem mit den Retransmissions auf Protokolle, die auf Zuverlässigkeit gebaut sind. Für unseren Ansatz ist das TCP Protokoll interessant, welches im vorigen Abschnitt erläutert wurde. Damit die Web Latenz im TCP Protokoll verringert werden kann ist eine Optimierung beim Retransmission Algorithmus von TCP zu suchen.

Weitere Ergebnisse der Messung von Google zeigen, dass die meisten Verluste in der Start-Phase vorkommen. Also nachdem der 3-Way-Handshake zum Verbindungsaufbau durchgeführt wurde und der Slow-Start mit den ersten Datenpaketen durchgeführt wird. Sobald die Verbindung einmal stabil steht und schon eine gewisse Menge an Daten durchgeflossen sind, treten laut Studie weniger Paketverluste auf. Aus dieser Erkenntnis folgt, dass die Übertragung von kleineren Datenmengen kritischer zu begutachten ist als große Datenströme. Weiterhin wurde beobachtet, dass 77 Prozent der Retransmission Probleme aufgrund von zu geringen RTO Zeiten zustande kommen [Laem]. Somit muss bei der späteren Analyse auch auf das Retransmission Timeout geachtet werden. Bzw. die RTO beim Lösungsentwurf herangezogen werden um eine Optimierung zu erhalten.

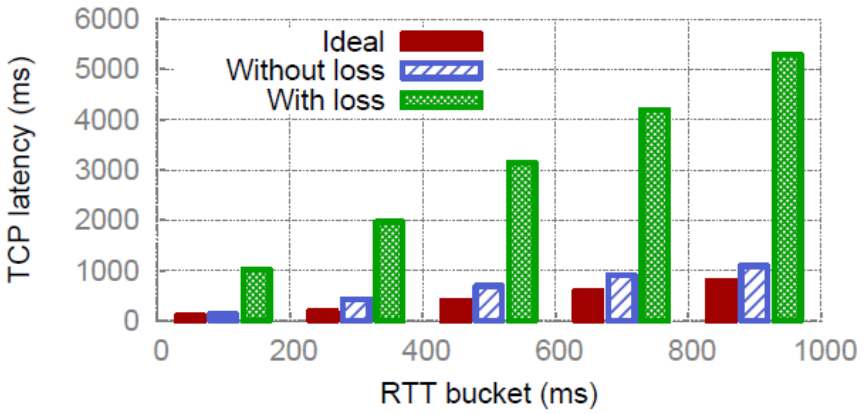

Abbildung 2: Googles Auswertung der Paketverluste und deren Konzequenz für die Web Latenz:[TFGo13]

4 Zielsetzung - Zieldefinition 1RTT

Zielsetzung des Papers erläutern

5 Lösungsvarianten

Nachdem unser Ziel definiert wurde, kümmern wir uns um die Wege, wie wir das Ziel Paket-Recovery innerhalb einer RTT. Hierzu wurden im Paper [] die drei Ansätze *Reactive*, *Corrective* und *Proactive* vorgestellt. Der *Reactive* beugt dem Problem des nicht erhaltenen ACKs vor indem es mehrere ACK Pakete zum Empfänger schickt. Somit ist die Wahrscheinlichkeit höher, dass der Empfänger eines dieser Pakete erhält und damit bestätigen kann. Der Corrective Ansatz versendet direkt mehrere Pakete. Durch diese Redundanz soll sichergestellt werden, dass die Pakete bei Verlust im Notfall beim Empfänger aus den vorhandenen Informationen wiederhergestellt werden. Die Proactive Lösung geht den krassesten Weg und versendet zu 100 Prozent redundant seine Daten. Somit wird das Paket doppelt aufgebläht. Wie die drei Ansätze im Detail funktionieren, wird in den folgenden Abschnitten behandelt.

5.1 Reactive

Lösungsansatz beschreiben

5.2 Corrective

Lösungsansatz beschreiben

5.3 Proactive

Lösungsansatz beschreiben

6 Lösungsbewertung anhand von Vor-und Nachteilen

Gegenüberstellung mit einer Tabelle

7 Ausblick und Fazit

Verweis auf die anderen Lösungsansätze bringen sowie ein endgültiges Fazit der Sache bringen.

Literatur

[Laem] Philipp Laemmel. Der Retransmission Timeout von TCP.

[TFGo13] Andreas Terzis Barath Raghavan Neal Cardwell Yuchung Cheng Ankur Jain Shuai Hao Ethan Katz-Bassett Tobias Flach, Nandita Dukkipati und Ramesh Govindan. Reducing Web Latency: the Virtue of Gentle Aggression. 2013.