One-Way Delay Determination Techniques

Mislav Boras
Betreuer: Dirk Haage
Seminar Innovative Internet-Technologien und Mobilkommunikation WS09/10
Institut für Informatik, Lehrstuhl Netzarchitekturen und Netzdienste
Technische Universität München
Email: boras@in.tum.de

Kurzfassung
Diese Seminararbeit beschreibt Techniken und Methoden welche genutzt werden um den One-Way Delay (OWD) zu messen. Dabei wird auch erklärt was der One-Way Delay ist und wo er zum Einsatz kommt. Des weiteren werden auch Methoden und Techniken erklärt welche notwendig sind um die Messtechniken anzuwenden welche hier beschrieben werden um den OWD zu bestimmen. Da es verschiedene Messtechniken gibt, werden diese auch miteinander verglichen um herauszufinden welche der Messtechniken sich gut und welche sich weniger gut eignen.

Schlüsselworte
Delay, One-Way Delay, One-Way, Measurement, Determination, Techniques, Einwegverzögerung, Messtechniken, Messungen.

1. OWD
In dieser Seminararbeit geht es vorrangig um den One-Way Delay (OWD). Das OWD beschreibt die Verzögerung die Auftritt wenn ein Paket von einem Computer zum anderen Computer geschickt wird.

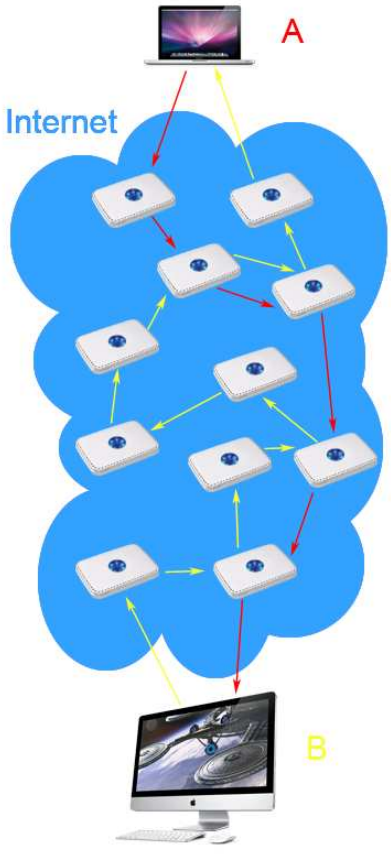

Abbildung 1 One-Way

In Abbildung 1 ist der Weg zu sehen, welchen die Pakete zurück legen, um von einem Computer zum anderen zu gelangen. Computer A sendet ein Paket an Computer B. Computer B sendet dabei auch ein Paket an Computer A. Deutlich zu sehen ist hierbei, dass die Pakete nicht die selbe Route einschlagen. Die blaue Wolke soll das Internet Demonstrieren. Klar wird hier, dass der OWD von A nach B nicht derselbe sein kein wie der von B nach A. Es soll nun herausgefunden werden wie groß der OWD in beiden Fällen ist. Die folgenden Abschnitte werden beschreiben wie dieser OWD gemessen werden kann und welche Methoden und Techniken dafür notwendig sind.

1.1 OWD Motivation
Um zu verstehen warum der OWD gemessen wird muss erst betrachtet werden, wo der OWD von Wichtigkeit ist. Im Grunde genommen spielt der OWD überall dort eine Rolle wo es nur darum geht ein Paket in eine Richtung zu schicken. Eine Antwort ob das Paket angekommen ist, wird nicht benötigt, da das Paket automatisch seine Bedeutung verliert wenn es zu spät ankommt. Deshalb ist es aber umso wichtiger herauszufinden wie lange das Paket braucht um anzukommen. Typische Anwendung bei welchen der OWD von Bedeutung ist, sind VoIP oder das Streaming von Videodaten. In beiden Fällen müssen die Pakete so schnell wie möglich zum gegenüber gelangen, da sie sonst nicht mehr aktuell sind. Grad bei diesen beiden Szenarien ist es wichtig, dass die Pakete zur Echtzeit übertragen werden.

Beim Video-Streaming haben wir einen einseitigen Datenaustausch. Die Empfänger schicken keine Daten zum Sender, sondern empfangen nur das Video-Signal. Hier spielt der OWD also nur in eine Richtung eine Rolle. Bei VoIP haben wir ein anderes Szenario. Beide Seiten schicken gleichzeitig Pakete zueinander. Wie in Abbildung 1 zu sehen ist können diese Pakete verschiedene Routen durch das Internet nehmen. Wir werden also in beide Richtungen unterschiedliche OWDs messen.

Um zu erforschen wie man Video-Streaming und VoIP in Zukunft noch verbessern kann ist es also notwendig den OWD zu minimieren. Da für diese Verbesserung zwanghaft notwenig ist, dass man den OWD weiß müssen auch effektive Messverfahren entwickelt werden.

1.2 OWD-Messmethoden
Der einfachste Ansatz um den OWD zu bestimmen, wäre die RTT (Round Trip Time) zu messen und diese durch 2 zu teilen (RTT/2). Wie in Abbildung 1 aber schon zu sehen ist, ist es nicht gewährleistet, dass das Paket auf den gleichen Weg zurück geroutet wird, wie es hin geroutet wurde. Wir können deshalb nicht davon ausgehen das die beiden Pfade auch den gleiche Verzögerung besitzen. Der OWD wäre somit verfälscht und würde nicht ein exaktes Ergebnis liefern. Den Ansatz den OWD durch die RTT zu bestimmen muss deshalb verworfen werden. In Abschnitt 4 werden diese Messmethoden genauer erklärt.

2. Zeit-Synchronisation

Der OWD wird in Sekunden gemessen. Es handelt sich dabei um die Verzögerung die Auftritt, wenn ein Paket von einem Punkt zum anderen Geschickt wird. Die Zeit spielt für eine Messung eine wichtige Rolle. Um die OWD richtig zu messen sollten die Zeiten beim Sender und Empfänger synchronisiert sein. Sollten die Uhren nicht synchron sein könne es vorkommen, dass ein negativer Wert für das OWD gemessen wird. Es gibt verschiedene Verfahren, welche die Synchronisation von den beiden Zeiten ermöglichen. Um ein besseres Verständnis für die Messung des OWD zu haben werden diese Verfahren in diesem Abschnitt vorgestellt.

2.1 GPS-Synchronisation

GPS kurz für Global Positioning System ist ein System bei welchem 30 Satelliten in der Erdlaufbahn kreisen. Die Satelliten kreisen so, dass immer 4 Satelliten erreichbar sind von jedem Punkt der Erde. Diese Satelliten senden ununterbrochen ein Signal aus, welches von einem GPS-Empfänger empfangen werden kann. Zu sehen ist dieses in Abbildung 2. Dieses Signal beinhaltet die Uhrzeit. Die GPS-Satelliten sind alle synchronisiert. Empfängt man also von zwei verschiedenen Satelliten das Signal, so erhält man dieselbe Uhrzeit. Man kann GPS also auch dazu verwenden Uhren zu synchronisieren. GPS hat aber ein entscheidendes Problem. Das GPS-Signal kann in Gebäuden nur schwer empfangen werden. Des weiteren braucht man jeweils einen GPS-Empfänger am Sender und am Empfänger.

Abbildung 2 GPS [3]

2.2 Christians Algorithm [1]

Bei diesem Algorithmus wird ein Zeitserver benötigt. Wichtig dabei ist zu beachten das die RTT von einem Paket geringer sein muss als die gewünschte Genauigkeit. Veröffentlich wurde der Algorithmus 1989 von Flavie Christian.

Der Algorithmus von Christian verläuft zwischen einem Prozess P und einem Zeitserver S, welcher die Quelle für die exakte Zeit (UTC) ist.

1. P erfragt die Zeit von S zum Zeitpunkt t_0
2. Die Anfrage wird von S zum Zeitpunkt t_1 empfangen
3. Die Anfrage wird von S verarbeitet; dies benötigt eine Zeitspanne l. Zum Zeitpunkt t_2 wird an P die UTC-Zeit von S gesendet
4. Die Antwort wird von P zum Zeitpunkt t_3 empfangen
5. P wird auf die Zeit von t_2+RTT/2 bzw. $t_2+((t_1-t_0)+(t_3-t_2))/2$ gesetzt

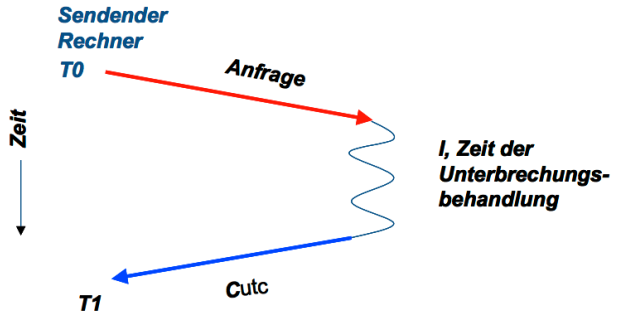

Abbildung 3 Algorithmus von Christian [3]

In Abbildung 3 verbildlicht den Algorithmus von Christian. T1 steht in dieser Abbildung für den Zeitpunkt T3.

Um den Algorithmus genauer zu verstehen empfiehl es sich ihn der Literatur nachzulesen [2]

2.3 Berkeley Algortihmus [4]

Ähnlich wie bei dem Algorithmus von Christian wird auch hier ein Zeitserver benötigt, welcher auch oft als Zeitdaemon bezeichnet wird. Entwickelt wurde dieser Algorithmus 1988 an der Universität Berkeley.

Der Zeitserver welcher hier verwendet wird, unterscheidet sich von dem Zeitserver bei dem Algorithmus von Christian. Während der Zeitserver bei dem Algorithmus von Christian passiv agiert, indem er eine Antwort senden wenn er angefragt wird, arbeitet der Zeitserver bei dem Berkeley Algorithmus aktiv. Das bedeutet das sich der Zeitserver um die Verteilung der Zeit kümmert und nicht der Client.

Der Algorithmus läuft in 3 Phasen ab. [4]

1. Der Zeitserver sendet in regelmäßigen Abständen seine lokale Uhrzeit an alle Maschinen.
2. Diese errechnen dann die Differenz der empfangenen Zeit mit ihrer lokalen Zeit und schicken die Differenz dem Zeitdaemon. Der Zeitserver errechnet aus allen Antworten das arithmetische Mittel.
3. Der Zeitserver teilt den Rechnern im Netz die jeweilige zeitliche Differenz mit, um welche die lokalen Uhrzeiten umgestellt werden müssen. Geht die Uhr des Clients vor (z.B. Client Zeit 1:20, Serverzeit 1:00), dann verlangsamt dieser seine Uhr solange bis Server- und Client-Zeit wieder übereinstimmen. Geht die Uhr des Clients nach (z.B. Client Zeit 1:00, Serverzeit 1:20), dann beschleunigt dieser seine Uhr solange bis Server- und Client-Zeit wieder übereinstimmen.

Abbildung 4 und Abbildung 5 verbildlichen den Algorithmus. In beiden Abbildungen sind drei Clienten zu sehen, welche mit dem Zeitserver verbunden sind.

In Abbildung 4 ist zu sehen wie der Zeitserver seine Zeit an die beiden Clienten schickt. Dieses ist gekennzeichnet durch die blauen Pfeile. Die roten Pfeile symbolisieren die Zeit, welche die Clienten an den Zeitserver übertragen wird. Nachdem der Zeitserver alle Zeiten empfangen hat berechnet er die Differenzen welche Zwischen der aktuellen Zeit und der Zeit welche ihm von den einzelnen Clienten übertragen wurde. Diese überträgt er dann wieder an die Clienten damit diese ihre Uhren umstellen können. Zu sehen ist dieses in Abbildung 5.

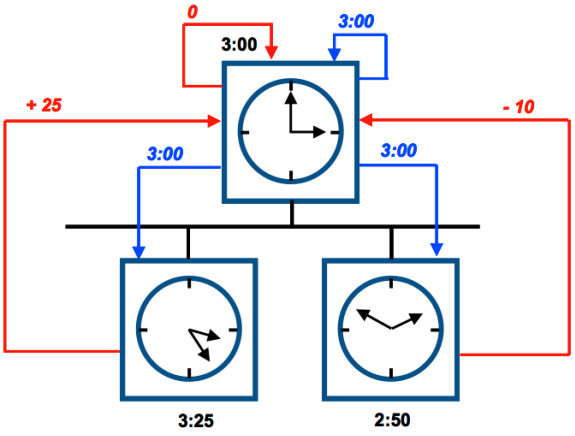

Abbildung 4 Berkeley Algorithmus vor Synchronisation [3]

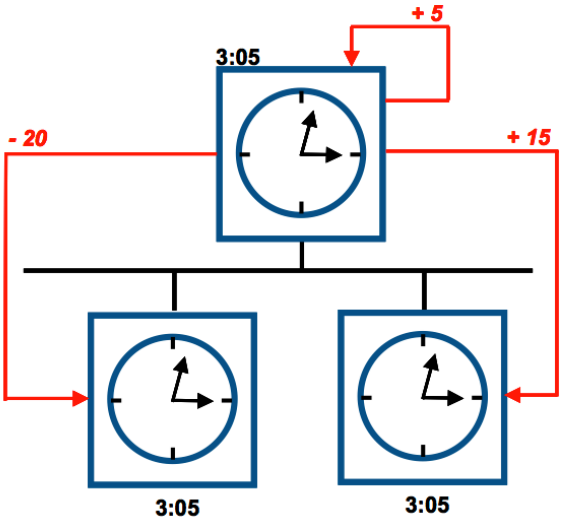

Abbildung 5 Berkeley Algorithmus nach Synchronisation [3]

2.4 Network Time Protocoll NTP [5]

NTP ist ein Standart welcher entwickelt wurde, um die Zeit in Computersystem zu synchronisieren. Entwickelt wurde NTP von David Mills 1985. NTP ermöglicht es die Uhren mehreren Computer in einem System mit hoher Genauigkeit zu synchronisieren. Die Algorithmen die dem Protokoll zugrunde liegen wurden in mehreren RFCs [6] veröffentlicht. Aktuell gibt es zwei Versionen von NTP. NTP v3 und NTP v4. Diese Versionen können auch miteinander verwendet werden. Es gibt auch eine einfachere Version dieses Protokolls, welches für kleinere Netzwerke (Heimnetzwerke) gedacht ist. Dieses nennt sich SNTP, Simple Network Time Protocoll. Im Gegensatz zu NTP hat SNTP einfachere Algorithmen implementiert, was dazu führt, dass die Zeit nicht so genau synchronisiert wird. Die Struktur der beiden Algorithmen ist jedoch die gleiche. Diese Struktur beinhalten 3 Komponenten.

1. **Client:** holt sich die Referenzzeit von einem oder mehreren Servern.
2. **Server:** stellt seine eigene Zeit anderen NTP-Clients im Netzwerk zur Verfügung.
3. **Peer**: vergleicht er seine eigene Zeit mit mehreren anderen NTP-Peers, die sich schließlich auf eine gemeinsame Zeit einigen, nach der sich alle richten.

Abbildung 6 zeigt ein Szenario des NTP. Zu sehen in diesem Bild sind die Server, Client und Peers. Die Grünen Pfeile symbolisieren die Anfrage der Clients an die NTP Server. Die roten Pfeile symbolisieren die Antwort der Server für die Client. Zu sehen ist auch, dass die Peers miteinender verbunden sind um sich auf eine Zeit zu einigen.

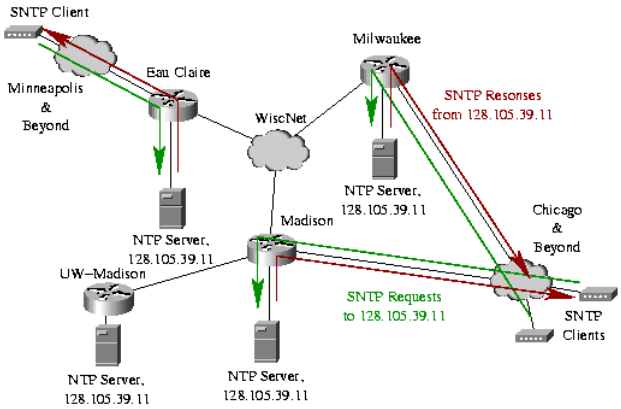

Abbildung 6 Network Time Protocoll Szenario [7]

2.5 Precision Time Protocoll PTP [8]

Im Gegensatz zu NTP ist PTP von der IEEE [10] entwickelt worden. Der Aufgabenbereich der beiden Protokolle ist unterschiedlich. Während NTP mehr für größere Netzwerke gedacht ist PTP eher für lokale Netzwerke Anzuwenden. Der Vorteil bei PTP liegt in seiner Genauigkeit. PTP schafft in seiner Hardwareausführung die Genauigkeit von Nanosekunden und in seiner Sofwareausführung die Genaugikeit von Microsekunden.

PTP besteht aus einem Netzwerk von Uhren die miteinander kommunizieren. PTP lässt über dieses Netzwerk den BMC (Best Master Clock) Algorithmus laufen. Dabei wird ermittelt welche der Uhren die exakteste Zeit hat. Dieses Uhr nennt sich dann Grandmaster Clock und wird als Master verwendet. Die anderen Uhren agieren als Slave und ermitteln die Zeit vom Master. Zu sehen ist dieses Szenario in Abbildung 7. Der Master senden ein Sync Message an seine Slaves mit der Zeit. Die Slaves bestimmen die Empfangszeit an der eigenen Zeit. Unabhängig davon senden die Slaves Delay-Request an die Master. Der Master bestimmt die Emfpangszeit dieser Pakete an der eigenen Zeit. Diese Empfangszeit wird mittels eines Delay-Response an den Slave zurück gesendet. Aus diesen ermittelten Zeiten wird nun der Master-to-Slave Delay und der Slave-to-Master Delay bestimmt. Der Mittelwert der beiden ermittelten Zeiten ermöglicht es dem Master die Uhren zu synchronisieren.

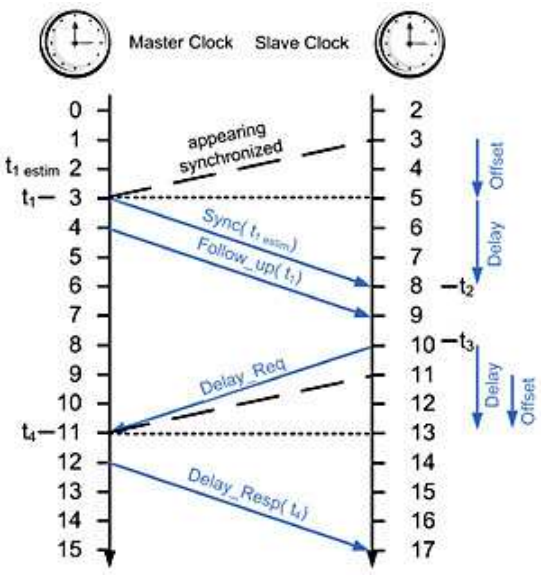

Abbildung 7 PTP Beispiel [9]

2.6 Andere Methoden

Die oben genannten Methoden zur Synchronisation der Uhren sind nicht die einzigen Methoden die dazu verwendet werden können. Sie beschreiben aber vor allem Methoden die oft benutzt und weiterentwickelt werden, was man auch daran sieht, dass einige Methoden standardisiert sind. Auch wenn die Methoden alle das gleiche Ziel verfolgen so sind ihre Einsatzgebiete doch unterschiedlich. Einige eignen sich besser für Große Netzwerke während andere sich eher für kleinere Netzwerke einigen. Es gibt noch einige Methoden die hier nicht beschrieben wurden. Bei Interesse empfiehlt es sich diese nachzuschauen bei folgender Literatur [2] [3].

2.7 Methoden Vergleich

Für genau Messung ist die Genauigkeit und Zuverlässigkeit der einzigen Verfahren von enormer Bedeutung. Deswegen werden hier die vorgestellten Verfahren miteinander verglichen und bewertet.

Tabelle 1 Genauigkeit

Verfahren	Genauigkeit
NTP	+-10ms
PTP	+-10ns
GPS	+-1µs
Berkeley Algorithmus	+-10ms
Christian Algorithmus	+-50ms
Funkuhr	+-50ms

In Tabelle 1 ist die Genauigkeit der einzelnen Algorithmen im Vergleich zu sehen. Je genauer des Verfahren desto besser eignet sich das Verfahren für OWD-Messungen. Leider können nicht alle Algorithmen in allen Szenarien angewandt werden.

Tabelle 2

Verfahren	Vorteile	Nachteile
NTP	Genauigkeit, Weit verbreitet	Komplexer Aufbau
PTP	Sehr genau	Nur für kleine Netzwerke
GPS	Überall vorhanden	Sichtkontakt, Empfänger
Berkeley Algorithmus	Genauigkeit	Komplexer Aufbau
Christian Algorithmus	Aufbau	Ungenau
Funkuhr	Überall verfügbar	Ungenau, Empfänger

Tabelle 2 hilft hierbei einen Vergleich herzustellen, wann sich welche Verfahren besonders gut eignen. Bei genauerer Betrachtung der Tabellen wird deutlich, dass man kein Verfahren als das optimale Verfahren zur Zeitsynchronisierung bzw. zur Messung des OWD bezeichnen kann. Es muss, dass Verfahren gewählt werden, welches sich an die Gegebenheiten am besten anpasst.

3. OWD Messtechniken

Der OWD wurde schon in Punkt 1 beschrieben. Hier in Punkt 3 wird nun auf die Messtechniken eingegangen welche genutzt werden um den OWD zu bestimmen. Die Messtechniken können in 2 Gruppen eingeteilt werden. Messungen des OWD mit synchronisierter Uhr und Messungen des OWD ohne synchronisierte Uhr.

3.1 OWD ohne Clock

Nicht jeder Computer verfügt über eine synchronisierte Uhr. In vielen Szenarien, wie zum Beispiel eine Telefongespräch mit VoIP, sitzen zwei Endverbraucher vor ihren Computer und kommunizieren miteinander. Es besteht die Möglichkeit, dass beide Verbraucher keine synchronisierte Uhr besitzen. Nichts desto trotz wäre es interessant herauszufinden wie groß die OWDs in beiden Richtungen sind. Im folgenden wird ein Verfahren beschrieben welches dieses tut.

3.1.1 One-Way Delay Estimation without Clock-Synchronisation [11]

Das Paper von Dongkeun Kim und Jaiyoung Lee beschreibt ein Verfahren wie der OWD ohne synchronisierte Uhren gemessen werden kann.

Es werden mehrere Pakete versendet. Dieses wird k-mal wiederholt je nachdem wie genau die Messungen werden sollen. Durch die verschiednen RTTs die gemessen werden ist es den beiden gelungen Algorithmus zu entwickeln durch welchen sie eine geringe Fehlerrate als RTT/4 erzielen. Der Algorithmus ist sehr komplex und nicht in weniger Worten zu beschreiben. Deshalb empfiehlt es sich dieses Pater genauer zu betrachten.

3.1.2 One-Way Delay Estimation Using Network-Wide Measurements[17]

Wie das voriger Paper behandelt auch dieses Paper die OWD-Messung ohne synchronisierte Uhren.

Die Art und weise wie die Messungen durchgeführt werden unterscheided sich jedoch von der vorigen. Im Grunde genommen werden hier kleine Messungen der RTT zwischen Knotenpaaren zusammengeführt. Dieser Algorithmus ist wie der vorige sehr komplex. Deswegen empfiehlt es sich auch hier das Paper zu lesen um diesen zu verstehen. Im Rahmen dieses Seminar gab es auch eine Seminararbeit zu diesem Paper, welche von Phillip Lowack angefertigt wurde [18]. Diese Seminarausarbeitung ist auf deutscher Sprache und kompakter gehalten als das Paper.

3.2 OWD mit Clock

Um den One-Way Delay zu bestimmen ist es hilfreich synchronisierte Uhren zu haben. Der Vorteil dabei liegt darin, dass man bei synchronisierten Uhren davon ausgehen kann, dass die Zeiten bei Empfänger und Sender identisch sind. Im Folgenden wird vorgestellt wie der Delay mit Hilfe von synchronisierten Uhren ermittelt werden kann.

3.2.1 One-Way Delay Measurement [13]

In diesem Paper wird den OWD mit Hilfe von PTP oder NTP gemessen werden. Wie schon in Abschnitt 2 beschrieben sind PTP und NTP Methoden zum synchronisieren von Uhren. Ist ein Netzwerk mit PTP oder NTP synchronisiert so hat man auf einfachen Weg die Möglichkeit den OWD herauszufinden indem man einfach ein Paket vom Sender zum Empfänger schickt. Beim absenden dieses Pakets wird die Zeit mit gesendet. Der Empfänger muss dann nur noch die Differenz zwischen seiner Zeit und der Zeit Des Pakets berechnen und hat dadurch den OWD. NTP wird verwendet, wenn Empfänger und Sender weit voneinander entfern sind. PTP wird verwendet, wenn Sender und Empfänger nicht weit voneinander entfernt sind. Die Genauigkeit von PTP lässt nach, wenn die Empfänger und Sender weit voneinander entfernt sind. Es existieren zwei Möglichkeiten wie man PTP und NTP zur Messung des OWD verwenden kann.

3.2.1.1 Standart NTP/PTP

Abbildung 9 zeigt einen Versuchaufbau um den OWD zu meseen. Bei diesem Versuchaufbau haben wir 2 Test-Units und einen Zeitserver. Beide Test-Units beziehen ihre Zeit von dem Zeitserver. Rot gekennzeichnet ist hier der Weg über welchen die Pakete zum Messen des OWD gesendet werden. Zu beachten ist bei diesem Aufbau das es zu Ungenauigkeiten kommen kann wen der Weg vom Zeitserver zur Test-Unit lang ist. Dadurch ist die Synchronisierung ungenauer und der OWD wird verfälscht.

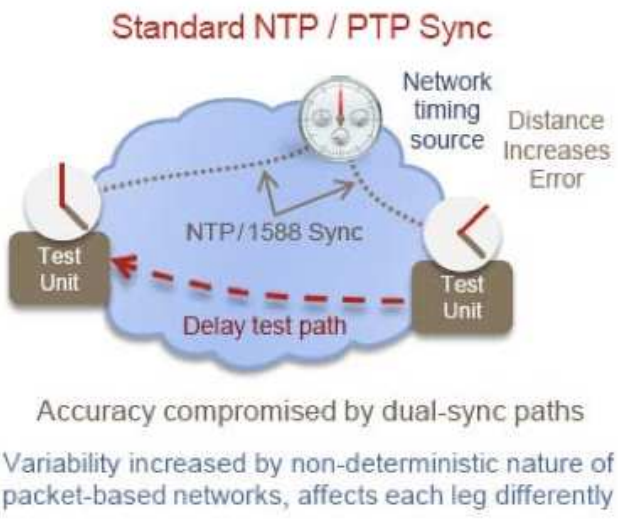

Abbildung 8

3.2.1.2 Enhanced NTP/PTP

Wie schon in Punkt 3.2.1.1 beschrieben kann es zu Fehlern kommen wenn der Zeitserver weit von den Test-Units entfernt ist. Abhilfe schafft hier der Versuchsaufbau aus Abbildung 10. Statt für beide Test-Units einen Zeitserver aufzusetzen ist hier eine Test-Unit gleichzeitig der Zeitserver. Dieser Versuchaufbau erzielt doppelt so gute Ergebnisse wie der einfache Versuchaufbau.

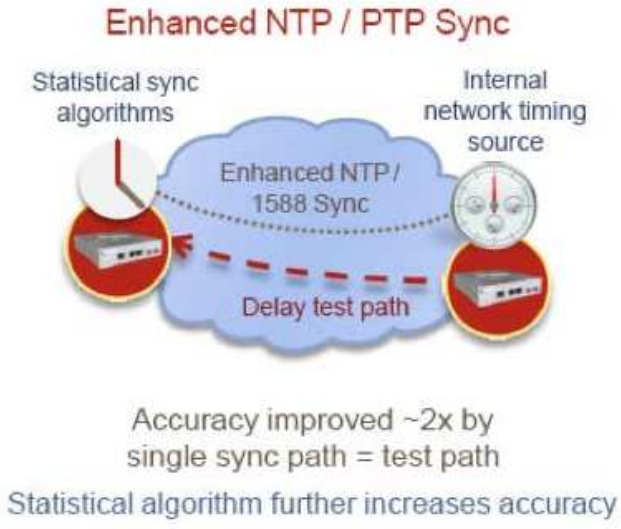

Abbildung 9

3.3 OWD mit anderen Verfahren

Im vorigen Punkt wurde die Zeiten mittels NTP oder PTP synchronisiert. Es besteht aber auch die Möglichkeit die Uhren auf andere Weise zu synchronisieren. In Punkt zwei haben wir andere Verfahren kennengelernt. Es kann genauso GPS, Berkeleys Algorithm oder Christians Algorithm zur synchronisation der Zeit verwendet werden. Der Versuchaufbau wäre der Selbe. Lediglich die Zeit würde anders bestimmt werden. Obwohl GPS bezüglich der Zeit sehr genau ist eignet es sich jedoch nicht für die Zeitsynchronisierung innerhalb von Gebäuden, da man für die optimale Zeitmessung Sichtkontakt zum Satelitten braucht.

3.4 OWAMP [12]

OWAMP, kür für One-Way Acitve Measurement Protocoll, ist ein Protokoll um den OWD im Netzwerk zu ermitteln. Es ist sowohl ein Kontroll-Protokoll als auch ein Test-Protokoll. OWAMP ist eine klassische Client-Server Application.

Abbildung 11 zeigt den Aufbau eines OWAMP Testbeds. Der Client kontaktiert der Server um eine Verbindung aufzubauen. Der Server kann nun entscheiden ob er diesen Request annehmen will oder ob er ihn ablehnen will. Nimmt er an so kommt es zu den Testpaketen, welche untereinander ausgetauscht werden um den OWD zu bestimmen. OWAMP benutzt NTP um die Uhren zu synchronisieren. Um zu verstehen wie OWAMP im einzelnen funktioniert empfiehlt es sich das RFC [13] durchzulesen.

4. Zusammenfassung

In diesem Paper wurde beschrieben, was der OWD ist und wie man in Messen kann. Festzuhalten bleibt, dass für eine genaue Messung des OWDs eine Sychronisierung der Zeit notwendig ist. Es existieren zwar Verfahren, welche die Möglichkeit bieten ohne Synchronisierte Uhren den OWD zu messen, jedoch sind diese Verfahren sehr kompliziert und sind mit der Genauigkeit der Verfahren die mit synchronisierten Uhren arbeiten nicht zu vergleichen. Außerdem gibt es zu diesen Verfahren noch keine Standards. Bei den Verfahren mit synchronisierten Uhren gibt es sogar RFCs welche eine umfangreiche Dokumentation anbieten.

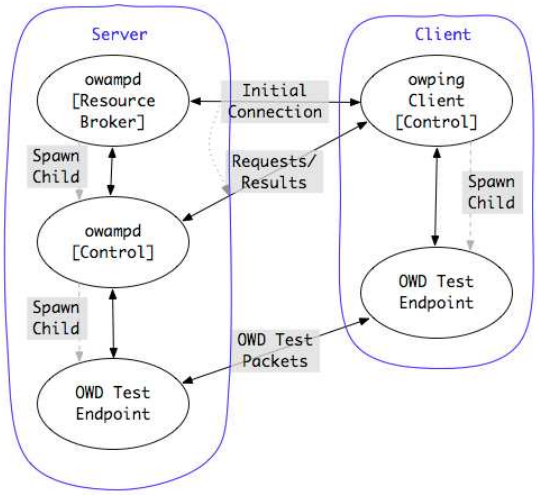

Abbildung 10 OWAMP Details

5. Literatur

[1] Algorithmuss von Christian http://de.wikipedia.org/wiki/Algorithmus_von_Cristian

[2] Andrew S. Tanenbaum und Maarten van Steen: *Verteilte Systeme – Grundlagen und Paradigmen.* Pearson Studium, 2003

[3] Uhrensynchronization http://www2.informatik.hu-berlin.de/~richling/emes2003/16-clocks.pdf

[4] Berkley Algorithmus http://de.wikipedia.org/wiki/Berkeley-Algorithmus

[5] NTP http://www.meinberg.de/german/info/ntp.htm

[6] NTP RFCs http://www.meinberg.de/german/info/ntp.htm#rfc

[7] NTP Szenario http://pages.cs.wisc.edu/~plonka/netgear-sntp/

[8] PTP http://de.wikipedia.org/wiki/Precision_Time_Protocol

[9] PTP IEEE 1588 http://www.nohau.co.uk/products/technologies/ieee1588_tech.htm

[10] IEEE 1588 http://www.ieee1588.com

[11] Dongkeun Kim und Jaiyoung Lee One-way Delay Estimation http://www.jstage.jst.go.jp/article/elex/4/23/717/_pdf

[12] OWAMP http://www.internet2.edu/performance/owamp/details.html

[13] ACCEDIAN http://www.accedian.com/modules/accedian/images/File/One-Way%20Delay%20Measurement%20Techniques.pdf

[14] Conger., S., and Loch, K.D. (eds.). Ethics and computer use. Commun. ACM 38, 12 (entire issue).

[15] Mackay, W.E. Ethics, lies and videotape... in Proceedings of CHI '95 (Denver CO, May 1995), ACM Press, 138-145.

[16] Schwartz, M., and Task Force on Bias-Free Language. Guidelines for Bias-Free Writing. Indiana University Press, Bloomington IN, 1995.

[17] One-Way Delay Estimation Using Network-Wide Measurements http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.7282&rep=rep1&type=pdf

[18] Phillip Lowack, Ermittlung der unidirektionaler Paketverzögerung durch netzwerkweite Messungen, 2010