Seminarausarbeitung „Leveraging BitTorrent for End Host Measurements“

Ralf Stange
(majere@cs.tu-berlin.de)

Betreuer
Oliver Hohlfeld

Seminar „Internet Measurement“
Technische Universität Berlin

WS 2008/2009 (Version vom 25. Januar 2009)

Zusammenfassung

Client-Systeme werden heutzutage gut geschützt. Unerwartete Datenpakete werden durch Firewalls und Intrusion Detection Systeme verworfen bevor sie das eigentliche Ziel erreichen. Viele Messverfahren benötigen jedoch gerade diese aktive Kommunikation mit dem Client um Informationen wie Bandbreite, Latenz oder Topologie über das Zielsystem zu sammeln.

In ihrer Arbeit „Leveraging BitTorrent for End Host Measurements“ [1] beschreiben die Autoren Isdal, Piatek, Krishnamurthy und Anderson eine Methode um Messdaten auch von geschützten Client-Systemen zu erhalten. Dabei verwenden sie einen modifizierten BitTorrent Client, der die normale und erwünschte Kommunikation der Teilnehmer des beliebten Peer-to-Peer Netzes für Messungen nutzt.

1 Einleitung

Das Wissen über den Aufbau und Struktur des Internets wächst längst nicht im gleichen Maße wie das Internet selbst. Dabei erlangen gerade Dienste und Anwendungen an Bedeutung, die auf eben jene Informationen angewiesen sind. Content Delivery Networks wie Akamai zum Beispiel versuchen durch Lastverteilung und möglichst kurze Datenwege eine schnellere Auslieferung von Online-Inhalten zu erreichen, benötigen dafür aber unter anderem Kenntnisse der Topologie der von ihnen belieferten Netze. Peer-to-Peer Netze wie Skype oder BitTorrent könnten durch Wissen über die Bandbreite der einzelnen Client-Systeme bessere Verteilungsstrategien anwenden und so die Performance verbessern.

Während man bestimmte Daten in Einzelfällen noch direkt von den Knotenpunkten des Internets durch Kooperation der jeweiligen Internet-Provider beschaffen kann, so ist dies jedoch für die große Zahl an Client-Systemen (End-Hosts) nahezu ausgeschlossen. Wir benötigen hier Messverfahren, die uns die gewünschten Informationen direkt von den Client-Systemen beschaffen können, ohne auf deren Kooperation angewiesen zu sein.

Dem steht aber leider die seit Jahren ansteigende Gefährdung der Clients im Internet entgegen. Um sich vor Angriffen zu schützen, werden Firewalls verwendet, Netze hinter NAT (Network Adress Translation) versteckt und Intrusion Detection Systeme verwerfen jedes unerwünschte oder unbekannte Datenpaket. Leider basieren gerade klassische Messverfahren darauf spezielle Datenpakete an die zu messende Client-Systeme zu schicken und aus den Antworten die Messergebnisse (wie z.B. Bandbreite oder Latenz des Clients) zu berechnen. Ärgerlich nur, wenn speziell diese Kommunikation von den Sicherheitssystemen der Clients verworfen wird.

Um auch Messdaten von unkoperativen Client-Systemen zu erhalten haben die Autoren dieser Arbeit das Tool *BitProbes* entwickelt. Dieses Tool nutzt das beliebte Filesharing Netz BitTorrent um an Messdaten zu gelangen. Für einen Teilnehmer dieses Peer-to-Peer Netzes ist es normal mit anderen BitTorrent-Teilnehmern zu kommunizieren und größere Datenmengen zu senden und zu empfangen. Diese „gewünschte“ Kommunikation nutzt *BitProbes* um seine Messdaten von BitTorrent-Teilnehmern zu sammeln.

Ein Einsatz eines Prototypen von *BitProbes* lieferte unter anderem die folgenden Ergebnisse:

- Sammlung von über 500.000 bestätigten IP-Adressen innerhalb einer Woche
- Auswertbare Datentransfers von ca. 20% dieser Clients
- die Upload-Bandbreite der untersuchten Clients
- Aussagen zur Effektivität von *BitProbes* beim Sammeln von Client IP-Adressen
- eine Schnittstelle für weitere TCP-Datenstrom orientierten Messverfahren
- Mitschnitt der BitTorrent-Protokollnachrichten zur späteren statistischen Auswertung von Nutzerverhalten oder Optimierung des BitTorrent Clients

Bevor wir auf die Funktionsweise von *BitProbes* in Kapitel 3 eingehen wird kurz die Arbeitsweise von BitTorrent beschrieben. Mögliche Auswertungsmethoden folgen in Kapitel 4. Die Ergebnisse des praktischen Einsatzes von *BitProbes* werden in Kapitel 5 beschrieben.

2 Grundlagen zu BitTorrent

BitTorrent ist ein beliebtes und weit verbreitetes kollaboratives Filesharing-Protokoll. Es eignet sich insbesondere für die schnelle Verteilung großer Datenmengen und nutzt dabei Peer-to-Peer Technik.

Das Server-Programm, der **Tracker**, besitzt die Rolle eines Koordinators. Er verwaltet Informationen von einer oder mehreren Dateien. Der Client erfährt vom Tracker, welche anderen Clients (**Peers**) noch die Datei herunterladen und verteilen.

Ein Anwender, der eine bestimmte Datei (**Torrent**) herunterladen möchte, benötigt eine sogenannte **Torrent-Datei**. Sie enthält den Hostnamen oder IP des Trackers, den gesuchten Dateinamen und eine Prüfsumme. BitTorrent besitzt keine globale Abfrageinstanz. Sollen Dateien zur Verfügung gestellt werden, so müssen dafür diese Torrent-Dateien erzeugt und bereitgestellt werden, die auf herkömmliche Kommunikationswege (z.B. Webseiten – **Torrent-Sites**) verteilt werden.

Der BitTorrent-Client kontaktiert mit Hilfe der Daten in der Torrent-Datei den Tracker und erfährt von ihm welche Peers aktuell die Datei (oder Teile davon) besitzen und verteilen.

Sobald ein Client ein komplettes Teilstück (**Piece**) der Datei heruntergeladen hat, meldet er dies den anderen Peers. Während er noch weitere Pieces herunterlädt, können nun andere Peers schon das erfolgreich geladene Teilstück von ihm laden.

Alle Clients die Teile einer Datei verteilen oder suchen und dem Trackers bekannt sind, nennt man zusammengefasst **Schwarm** (Swarm).

Spezielle Bezeichnungen für Clients sind:

Seeder Clients die die komplette Datei besitzen und anbieten

Free-Rider Clients die nur herunterladen ohne selber zu verteilen

Gerade die Fähigkeit die Upload-Bandbreite der Clients zu nutzen, bevor die gesamte Datei heruntergeladen wurde, ist einer der Gründe für die allgemein gute Performance von BitTorrent.

Eine deutlich umfassendere Beschreibung der Funktionsweise von BitTorrent findet sich unter anderen in [2] und [3].

2.1 Details zur Kommunikation

Um eine Datei über BitTorrent herunterzuladen benötigt der Client zwingend eine Torrent-Datei mit einem gültigen Tracker. Mit Hilfe der Informationen aus der Torrent-Datei findet die folgende (vereinfacht dargestellte) Kommunikation statt:

- Client kontaktiert den Tracker

- der Tracker unterhält eine regelmäßig (durch andere Clients) aktualisierte Liste von aktiven Peers mit der gewünschten Datei
- eine zufällige Auswahl der aktiven Peers wird vom Tracker an den initialen Client gesendet
- der Client verbindet sich mit den vom Tracker genannten Peers
- zwischen Client und Peers werden Informationen über den aktuellen Stand (wer besitzt schon welche Pieces der Datei) mit Hilfe von `BitField` Nachrichten [3] ausgetauscht
- außerdem teilen die Peers dem Client mit, ob sie bereit sind ihm Pieces zu schicken (unchoke, siehe nächstes Kapitel)
- falls ja, fordert vom Peer ein fehlendes Piece an (64-512 KB)
- Download des Pieces
- sobald ein Piece vollständig heruntergeladen wurde, wird an alle verbundenen Peers eine `have` Nachricht [4] gesendet

Der Schwarm besitzt damit regelmäßig aktualisierte Informationen, welcher Peer welche Teilstücke der Datei besitzt.

2.2 Choking Mechanismus

Ein Client entscheidet bei einer Anfrage von einem anderen Peer ob er diesem Daten schickt (**unchoke**) oder nicht (**choke**).

Um dabei eine möglichst gute Performance beim verteilen der Datei zu erreichen, werden Peers mit hohen Upload-Raten bevorzugt. Um durch diese Strategie nicht dauerhaft neue Peers auszuschließen wird zusätzlich ein sogenanntes **optimistic unchoke** durchgeführt. Dabei wählt ein Peer alle 30 Sekunden einen anfragenden Client zufällig und unabhängig von seiner Upload-Rate aus und setzt ihn auf unchoke. Neue (evtl. sogar performantere) Peers erhalten so eine Chance.

Speziell dieser Mechanismus der *optimistic unchokes* ist essentiell für die Funktionsweise des *BitProbes* Clients.

3 BitProbes

Um Messdaten über einen Client zu sammeln benötigt man dessen IP-Adresse und für Messungen auf Basis von TCP-Verbindungen je nach eingesetztem Tool eine Mindestmenge an übertragenen Daten.

Mit Hilfe des *BitProbes* Clients gelingt dies großflächig selbst von unkooperativen Client-Systemen (End-Hosts). Es nutzt dabei die starke Verbreitung von BitTorrent und verwendet zum Sammeln der Daten protokollkonforme Kommunikation mit den Clients des BitTorrent Netzes. Der einzelne Client merkt nicht, dass er mit *BitProbes* statt mit einem normalen BitTorrent-Client spricht.

Die Autoren sind leider nicht näher auf den Aufbau ihres Prototypen eingegangen (selbstgeschriebener oder modifizierter Client) und stellen ihn auch nicht für eigene Tests zur Verfügung.

3.1 Sammeln von gültigen IP-Adressen

Um die Adressen von möglichst vielen Systemen zu sammeln wurden beliebte Torrents mit großen Teilnehmerzahlen gesucht. Dazu wurden:

- bekannte und stark genutzte Torrent-Sites ausgesucht
- Torrents mit einer großen Zahl an Seedern und Peers ausgewählt

Auf Basis der ausgewählten Torrent Dateien kontaktiert *BitProbes* die Tracker und erhält die Listen der aktuellen Peers der einzelnen Schwärme. Damit besitzt er umfangreiche Listen an potenziellen „Messobjekten", die bereit sind mit BitTorrent-Clients zu kommunizieren.

3.2 Sammeln von Daten aus TCP-Verbindungen

Um die Peers dazu zu bringen größere Datenmengen zur späteren Auswertung zu senden wird ausschließlich BitTorrents *optimistic unchoke* Mechanismus verwendet (siehe Kapitel 2.2).

BitProbes fordert also Daten an, ohne selber welche anzubieten. Es werden auch keinerlei heruntergeladenen Daten gespeichert. Einzig das Zeitverhalten (Antwortzeiten, Downloadgeschwindigkeit, etc.) wird protokolliert.

Dieser strikte Grundsatz keine Daten zu speichern oder anzubieten ermöglicht es beliebige populäre Torrents zu nutzen, unabhängig von der Legalität der Quelle.

Um die Trefferquote bei der zufälligen Auswahl beim *optimistic unchoke* Mechanismus zu erhöhen wurden die Zahl der gleichzeitigen Verbindungen bei *BitProbes* auf 1000 für einen Schwarm vergrößert. Typische BitTorrent Clients besitzen ein Limit von 50-100.

Zur Erhöhung der Gesamtperformance der Datensammlung wird *BitProbes* auf mehrere Messknoten mit jeweils mehreren Instanzen betrieben. Zur Synchronisation der Clients untereinander und zur Vergrößerung der Anzahl möglicher Ziele leiten die Clients alle Informationen über Schwärme vom Tracker an einen zentralen *BitProbes*-„Shadow-Tracker" weiter.

Als weitere Maßnahme wurde die maximale Datenmenge die von einem Peer heruntergeladen wird auf 2 MB begrenzt, um die Belastung als „Free-Rider" im BitTorrent Netz gering zu halten

3.3 Sammeln von Protokolldaten

Zusätzlich werden noch für spätere Auswertungen sämtliche Protokoll-Nachrichten (`have`, etc.) in anonymisierter Form gespeichert.

4 Auswertungsverfahren

Prinzipiell eignet sich *BitProbes* für beliebige Auswertungen auf Basis von

- IP-Adressen
- Zeitverhalten von TCP-Verbindungen
- Analyse der BitTorrent Protokoll-Nachrichten

Dabei muss die eigentliche Speicherung der Messdaten nicht innerhalb von *BitProbes* erfolgen. Es kann vielmehr jedes beliebige Tool zur Analyse von TCP-Verbindungen verwendet werden. Es folgen einige Anwendungsmöglichkeiten durch *BitProbes*.

4.1 Messung der Upload-Bandbreite

Um Informationen über die Upload-Bandbreite eines Peers zu erhalten wurde das Tool MultiQ [5] eingesetzt.

MultiQ kann passiv anhand einer TCP-Verbindung die Bandbreite der Verbindung mit hoher Genauigkeit abschätzen. Voraussetzung ist eine Mindestmenge an übertragener Daten (z.B. die Upload-Bandbreite beim herunterladen eines Pieces). Für eine Berechnung der Download-Bandbreite reichen die wenigen übertragenen Kontrollpakete in die andere Übertragungsrichtung jedoch nicht aus. Zusätzlich wird angenommen, dass die geringste Bandbreite bei der Peer-Anbindung vorliegt. Sollten auf der Übertragungsstrecke andere Engpässe existieren, so verfälschen sie die Messergebnisse.

4.2 Messung der Download-Bandbreite

Diese Messung erfolgt durch eine Analyse der BitTorrent Protokoll-Nachrichten.

Der Peer sendet bei jedem vollständig heruntergeladenen Piece eine `have` Nachricht. Mit dem Wissen über die Größe der Pieces und dem Abstand der Nachrichten ist eine Schätzung über die Download-Bandbreite möglich ohne selber Daten hoch zu laden. Unberücksichtigt bleibt dabei jedoch die Auslastung der Anbindung durch andere Protokolle.

4.3 Weitere Messungen

Theoretisch sind weitere Messverfahren denkbar, wie sie z.B. in [6] (TCP-Sidecar) beschrieben sind. Also über die vorhandenen TCP-Verbindungen durch geplante Manipulation der Pakete weitere Informationen zu sammeln. Als Beispiel geben die Autoren das Variieren der TTL (Time-to-Live) von Paketen an um so Routinginformationen über das Ziel zu sammeln.

Durch das Sammeln sämtlicher BitTorrent-Protokollnachrichten werden auch statistische Auswertungen zum Verhalten von BitTorrent Clients (Schwarmgrößen, Teilnahmezeiten von Peers, etc.) möglich.

5 BitProbes im praktischen Einsatz

Für den Einsatz eines Prototypen von *BitProbes* galten die folgenden Bedingungen:

- es wurden 8 Server aus dem Netz der University of Washington genutzt
- auf jedem Server wurden 40 Instanzen von *BitProbes* gestartet
- Daten wurden eine Woche lang gesammelt (2.9. – 9.9.2006)
- `http://thepiratebay.org` und `http://www.mininova.org` waren Ausgangspunkt zum Sammeln von Torrents
- die beiden Torrent-Sites wurden alle 12h automatisiert nach Torrents mit möglichst vielen Teilnehmern (große Schwärme) durchsucht

Die in der Arbeit beschriebenen Messdaten beschäftigen sich zuerst mit der Effiktivität von *BitProbes*. In Kapitel 5.2 wird schließlich auf Messergebnisse der kontaktierten Peers eingegangen.

5.1 Verbindungs- und Mess-Rate

Ausgehend von den gesammelten Torrents erhält man (über die Tracker) Listen von IP-Adressen mit potenziellen Peers.

Für Clients die eingehende BitTorrent Verbindungen zulassen ist eine Verifikation der IP-Adresse einfach. *BitProbes* baut einfach eine Verbindung auf und fragt nach den verfügbaren Pieces.

Für Peers die keine eingehenden BitTorrent Verbindungen zulassen (NAT, Firewall, etc.) ist dies nicht möglich. Hier sind wir darauf angewiesen, dass der Peer *BitProbes* selbstständig kontaktiert. Die Adresse erhält er vom Tracker, der alle Peers regelmäßig über neue Schwarmmitglieder informiert. Diese Form des Verbindungsaufbaus geschieht in der Praxis sogar sehr häufig.

Sobald *BitProbes* in der Lage war Kontakt zum Peer aufzunehmen, bezeichnen wir eine IP-Adresse als bestätigt. Dabei spielt es keine Rolle ob die Verbindung zu weiteren Messungen verwendet werden konnte.

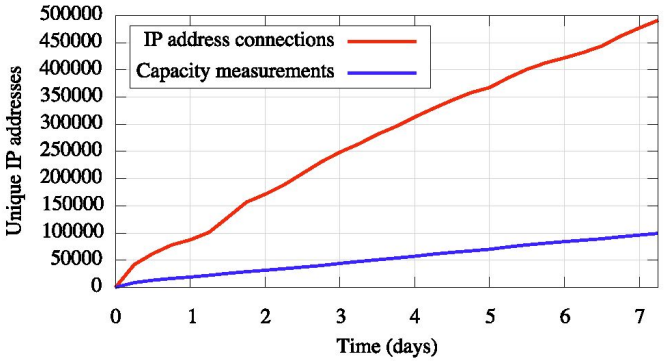

Abbildung 1: Anzahl bestätigter und gemessener IP-Adressen

Die rote Kurve in Abbildung 1 zeigt die Zahl der bestätigten IP-Adressen innerhalb des einwöchigen Messzeitraums. Insgesamt wurden ca. 500.000 unterschiedliche IP Adressen gefunden und bestätigt.

Der lineare Anstieg bis zum Ende des Messzeitraumes deutet darauf hin, dass das Mess-System mehr IP-Adressen zur Auswahl hatte, als es untersuchen konnte. Bei einem sich erschöpfenden IP-Pool müsste die Kurve abflachen.

Die TCP-Kommunikation der bestätigten Peers wurde für Kapazitätsmessungen durch das Tool MultiQ [5] genutzt. Alle Verbindungen mit mehr als 100 IP-Paketen wurden an MultiQ zur Analyse gegeben (176.487 Verbindungen).

Die blaue Kurve in Abbildung 1 zeigt die Zahl der Verbindungen die zur Bandbreitenabschätzung durch MultiQ verwendet wurden. Insgesamt waren es 96.080 Verbindungen. Die restlichen Verbindungen erfüllten nicht die relativ hohen Qualitätskriterien an die TCP-Daten zur Kapazitätsmessung durch MultiQ.

5.2 Netzwerkabdeckung

BitProbes ist also problemlos in der Lage sich mit zehntausenden Peers zu Verbinden und von ihnen analysierbare TCP-Verbindungen zu erhalten. Doch wie sieht es mit der Verteilung dieser Peers im Internet aus?

Dazu wurden anhand der bestätigten IP-Adressen der Peers die BGP Präfixe (Border Gateway Protokoll) und AS (Autonomous Systems) bestimmt. BGP ist ein verbreitetes Routing Protokoll zwischen unabhängigen Netzen (AS) und wird unter anderem von Internet-Providern eingesetzt.

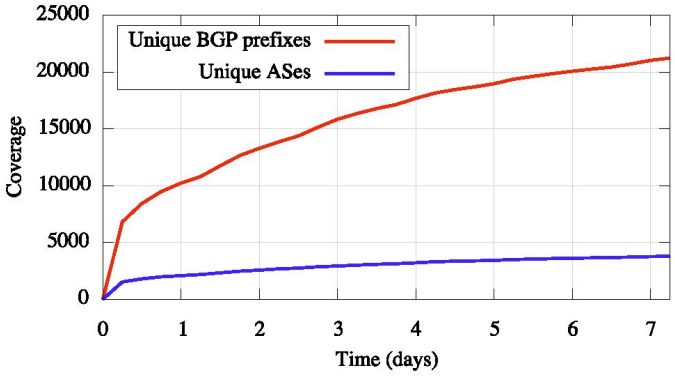

Abbildung 2: Netzwerkabdeckung der bestätigten IP-Adressen

Abbildung 2 zeigt die Zahl der gefundenen unterschiedlichen BGP Präfixe (rote Kurve) und autonomen Systeme (blaue Kurve) über den Messzeitraum und ergibt innerhalb nur einer Woche bei den autonomen Systemen eine Abdeckung von fast 20% (3.763 gefundene AS bei ca. 23.000 AS weltweit [7]).

5.3 Upload-Bandbreite der Peers

Neben der Erprobung von *BitProbes* und eine Abschätzung dessen Leistungsfähigkeit wurden auch Daten zur Upload-Bandbreite von Peers gesammelt.

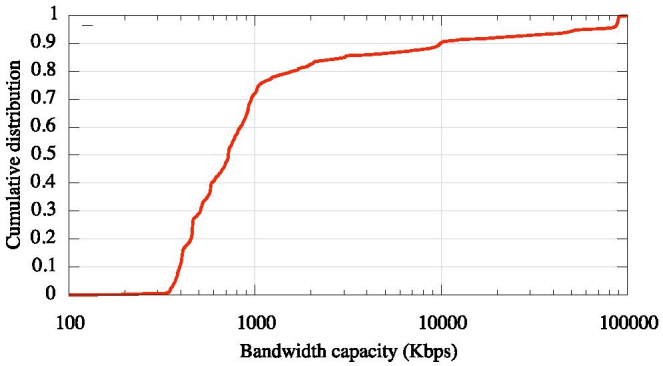

Abbildung 3: Verteilung Upload-Bandbreite

In Abbildung 3 ist die prozentuale Verteilung der Upload-Bandbreite der Peers dargestellt. Einige Zahlen aus der Grafik:

- 70% der Peers besitzen eine Upload-Bandbreite zwischen 0,35 und 1 Mbps
- 10% der Peers besitzen 10Mbps und mehr
- BitTorrent wird scheinbar erst mit höheren Bandbreiten interessant, niedrige Bandbreiten (unter 300 Kbps) sind kaum vertreten

Dabei stammen 64% der gesamten übertragenen Daten von 5% der Peers (mit Bandbreiten über 55 Mbps).

Während die Zahlen im Verhältnis untereinander eine gute Abschätzung über die Bandbreitenanbindung der Peers geben, sind die absoluten Werte mit Vorsicht zu betrachten. Aus den Messungen erschließt sich nicht, wieviel von der Upload-Bandbreite von anderen Applikationen oder BitTorrent-Verbindungen genutzt wird. Auch könnte der Peer absichtlich die zur Verfügung gestellte Bandbreite drosseln.

6 Zusammenfassung

Messungen auf Basis von TCP-Verbindungen von geschützten End-Hosts ohne deren Kooperation sind möglich!

Die Autoren haben mit ihrem Prototypen *BitProbes* im praktischen Versuch gezeigt, dass sich populäre Peer-to-Peer Netze mit ihren Millionen von Teilnehmern eignen Messdaten von den Teilnehmern zu sammeln.

Unter Nutzung des BitTorrent-Filesharing-Netzes war es dem *BitProbes* Client innerhalb einer Woche problemlos möglich

- über 500.000 bestätigte IP Adressen von BitTorrent Teilnehmern zu sammeln,

- bei fast 20% dieser End-Hosts eine längere TCP-Verbindung zum Datentransfer zu initiieren und das
- unbeeinträchtigt von Intrusion-Detection Systemen oder anderen Schutzsystemen (Firewall, NAT, etc.).

Das zur Analyse der Datentransfers eingesetzte Tool MultiQ ist dabei nur als Beispiel zu sehen. Jedes Analyse-Tool auf Basis von TCP-Streams kann eingesetzt werden. Zusätzlich sammelt der BitProbe Client alleine durch seine Teilnahme am BitTorrent Netz Informationen über das Nutzerverhalten.

Eine Vergrößerung der in die Messung einbezogenen End-Hosts ist ebenfalls problemlos möglich. Es müssen die Zahl der *BitProbes* Instanzen erhöht werden und ggf. weitere Torrent Quellen erschlossen werden. Die Durchdringung ist nur durch die Popularität und Verbreitung des BitTorrent Netzes beschränkt.

Prinzipiell sind die beschriebenen Methoden nicht nur auf BitTorrent beschränkt. Sind die Voraussetzungen gegeben (direkte Kommunikation der Clients auf TCP-Basis untereinander und eine Möglichkeit den Client zum Senden von Infomationen zu motivieren), kann die vorgestellte Idee auch für andere Peer-to-Peer Netze genutzt werden.

Außen vor bleiben bei allen Messungen natürlich jene Netzbereiche, die keine oder nur wenig Nutzer des verwendeten Peer-to-Peer Netzes besitzen (wie z.B. Firmennetze).

Literatur

[1] Tomas Isdal, Michael Piatek, Arvind Krishnamurthy, Thomas Anderson: *Leveraging BitTorrent for End Host Measurements*. Department of Computer Science and Engineering, University of Washington, 2007.

[2] J. Hawes: *MoreHawes: The BitTorrent Protocoll* `http://morehawes.co.uk/index.php?q=the-bittorrent-protocol`, 12.2008.

[3] *BitTorrent Protocol Specification v1.0*, `http://wiki.theory.org/BitTorrentSpecification`, 12.9.2006.

[4] B. Cohen: *BitTorrent Protocol Specifications v1.0*, `http://www.bittorrent.org/beps/bep_0003.html`, 28.2.2008.

[5] S. Katti, D. Katabi, D. C. Blake, E. Kohler, J. Strauss: *MultiQ: Automated detection of multiple bottleneck capacities along a path*, IMC, 2004.

[6] R. Sherwood, N. Spring: *Touring the Internet in a TCP Sidecar*, IMC, 2006.

[7] *Ripe, Statistics*, `http://www.ris.ripe.net/as-stat/2006/rrc00/200609.html`, 12.2008.