Seminararbeit

Routingverfahren in Content Delivery Networks

Lothar Bremkes

Seminar
Content Delivery Networks

Universität Manheim
Lehrstuhl für Praktische Informatik IV: Prof. Dr. Wolfgang Effelsberg
Betreuer: Christian Liebig

7.12.2004

Inhalt

1 End To End Routing im Internet

Im Grundsätzlichen werden heute zwei Verfahrensweisen des Routing unterschieden:

-statisches Routing
-adaptives dynamisches Routing

Beim statischen Routing werden die Router mit festen (statischen) Routen ausgestattet. Die Vorteile dieser Methode sind der geringe Aufwand bei der Einrichtung (= geringere Kosten) und die Nachvollziehbarkeit der Routen, da die Pakete immer den gleichen Weg nehmen. Die Nachteile ergeben sich zum Teil direkt hieraus, denn der Router kann so offensichtlich nicht auf Überlastungen oder Ausfälle reagieren und jede Änderung der Netzwerktopologie ist mit einem sehr hohen administrativen Aufwand verbunden. Aufgrund der geringen Möglichkeit der Einflußnahme wird dieses Verfahren im Weiteren auch nicht betrachtet.

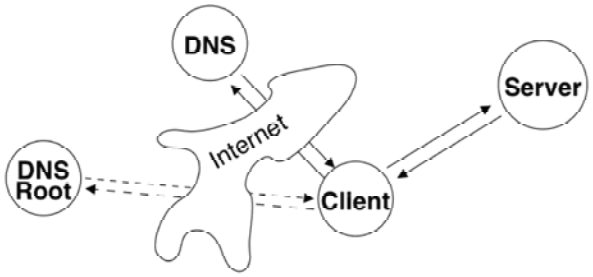

Abb. 1: konventionelles Content Routing

Beim adaptiven dynamischen Routing errechnen die Router selbständig die Routen für Datenpakete und tauschen diese via Routingtabellen untereinander aus. So können Router effektiv auf Änderungen der Netzwerktopologie oder auch Überlastungen und Ausfälle reagieren. Nachteile dieses Verfahrens sind die erhöhten Kosten für die Einrichtung des Routers und die Abhängigkeit von Routingprotokollen Änderungen der Metrik in eine Änderung der Wegwahl umsetzen zu können (Konvergenz), sowie daraus resultierende Routingfehler [8].
Die hier Verwendeten Routingprotokolle werden in zwei Arten unterteilt, die "interior gateway protocols" (OSPF, RIP), die mit Hilfe detaillierter Informationen über Topologie, Bandbreite und Link Delays innerhalb eines AS (Autonomous System) Routen ermitteln und das "exterior gateway protocol" (BGP), das für die Vernetzung individueller Netzwerke zum Internet genutzt wird.

2 Bottlenecks

Die kritischen Stellen für Content Routing sind die sog. Bottlenecks, Engpässe, die die Performance bremsen, nach Akamai [9]:

-First Mile
-Peering Points
-Backbone
-Last Mile

First Mile bezieht sich auf den Host Server und dessen Anbindung ins Netz. Der Gedanke des CDN an sich ist dieses problem durch Dezentralisierung der Inhalte zu lösen.

Die Peering Points (Verbindungen zwischen den Individuellen Netzwerken) sind ein weiterer Engpaß. Die meisten Netzwerkbetreiber haben wenig Ambitionen, die Peering Points auszubauen, da sie im grunde keine Einnahmen verursachen, aber erhebliche Kosten. So werden Peers oftmals anhand der vorhandenen Nutzung ausgebaut und haben keine oder nur geringe Reserven mit anwachsendem Datenverkehr mitzuhalten. Zudem weisen voll ausgelastete Links oft hohe Verlustraten bei Paketen und hohe Latenzzeiten (aufgrund „Router Queuing Delays“) auf.

Auch der Backbone kann sich als Engpaß etablieren, wobei hier das Problem nicht in der Kapazität der Kabelleitungen manifestiert, sondern - wie bei den Peers - in der Kapazität der Backbone-Router. Es ist heute üblich, daß Datenpakete durch das Internet über ein oder mehrere Backbones laufen, daher sind auch heute erreichte Spitzenwerte von ca. 400-500 Gbps Kapazität nicht ausreichend für die multimediale Zukunft. Ein Beispiel einer Video-On-Demand Kochshow zeigt, das bei nur 100,000 gleichzeitigen Zugriffen und einer Datenrate von nur 300 kbps diese Anwendung 30 Gbps Bandbreite benötigt [9]. Denkt man jetzt nur einen Schritt weiter zum Video-On-Demand Fernsehen wird deutlich, wie schnell ein Backbone überlastet würde.

Das bekannte Last Mile Problem bezieht sich auf die oft nur sehr langsame Anbindung an den Provider über 56k Modem oder ISDN, o. ä. Es ist allerdings ein wenig kurzsichtig zu sagen, wenn alle Anwender über schnelle Kabelanbindungen verfügen wäre das Problem gelöst, da es sich dann nur auf die anderen drei verlagert.

3 Redirection Mechanisms

Ein effizientes Routingverfahren für CDN muß also in der Lage sein, diese Engpässe zu minimieren. In CDN werden Inhalte auf unterschiedliche Server, sog. Surrogates verteilt um eben dieses Problem zu lösen. Jedoch stellt sich nun die Frage, wie eine Anfrage auf welchen dieser Surrogates umgelenkt werden soll. Hierzu gibt es mehrere Verfahren:

3.1 HTTP Request Forward

Eine Standardlösung zur Weiterleitung der Anfrage auf einen anderen Server ist ein HTTP Request Forward. Es gibt verschiedene Methoden, eine derartige Weiterleitung zu erreichen.

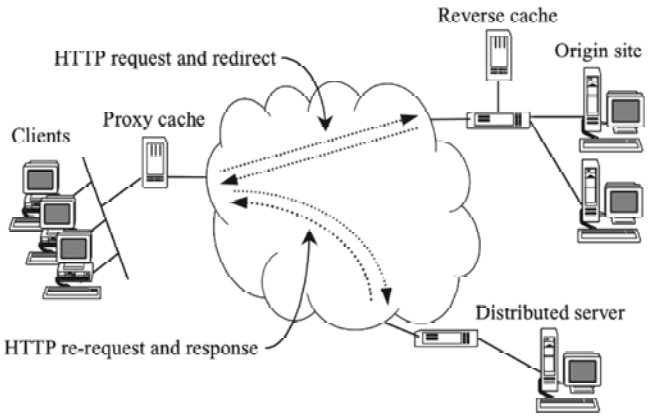

Abb. 2: Ablauf einer mittels HTTP Redirection umgeleiteten Anfrage

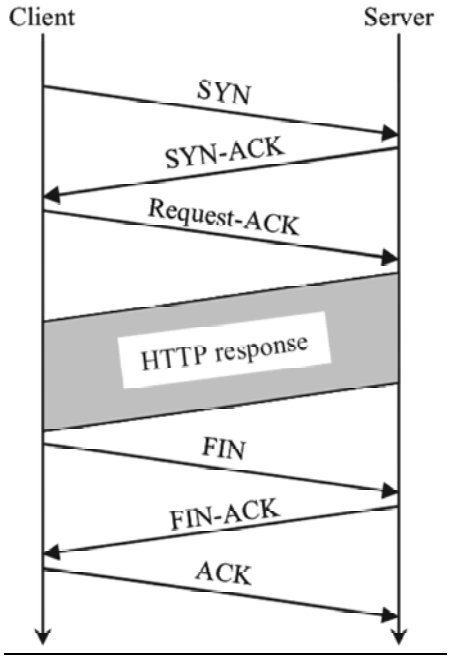

Abb. 3: Schematische Darstellung von HTTP Request und Response

3.1.1 DNS Redirection

Bei DNS Redirection werden in DNS(domain name system)-Servern verschiedene IP-Adressen zu einem Hostnamen gespeichert. Es gibt nun verschiedene Möglichkeiten, nach welchen Algorithmen der DNS-Server die IP(s) ermittelt, die er an die Anfrage sendet.
LBP (load balanced policy): die Anfragen werden gleichmäßig auf die verschiedenen Server verteilt.
MIN: die Anfrage wird an den Server mit der kleinsten zu erwartenden RTT (round trip time) geleitet.
MIN2: die Anfragen werden gleichmäßig auf die 2 Server mit der kleinsten zu erwartenden RTT verteilt (es hat sich gezeigt, daß für MIN3 keine oder kaum verbesserungen zu erwarten sind [7]).
WGT: 80% der Anfragen werden an den Server mit der kleinsten zu erwartenden RTT geleitet, die restlichen 20% werden auf alle anderen Server verteilt.
DNS Redirection wird oft für load balancing in Server Clustern benutzt

3.1.2 TCP Connection Splicing

Bei TCP Splicing schaltet sich ein "fremder" Rechner in eine bereits bestehende Verbindung ein und übernimmt diese. Hierzu schaut der Router in den TCP und IP Headern der Anfrage nach, welche Verbindungsinformationen - wie IP-Adresse und TCP-Portnummer - dort angegeben sind. Mithilfe dieser Informationen nutzt der Router Verfahren, wie NAT (network adress translation) oder PAT (port adress translation), um die Verbindung umzuleiten. Zur Verbesserung der Performance wird die Verbindung, ist sie ersteinmal aufgebaut, durchgeschleift ("spliced"), wobei die Datenpakete vom Switch auf der Network Layer weitergeleitet werden und nicht mehr von der TCP Layer bis zur Application Layer verarbeitet werden müssen. Es gibt auch Verfahren, die Verbindung nicht auf TCP Layer zu "splicen", sondern auf Socket-Layer. Beim socket level TCP splice wird mehr Rechenleistung und Speicher für das TCP Layer Processing benötigt, kann diesen Nachteil aber

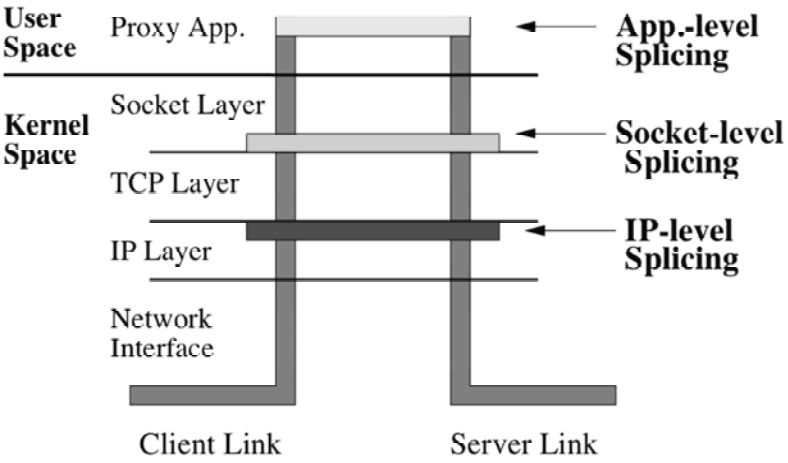

Abb. 4: Verschiedene Ebenen für „Splicing“ von Verbindungen

dadurch mehr als ausgleichen, daß die Daten nicht mehr zwischen Kernel und Benutzerpuffern kopiert werden müssen.

3.1.3 HTTP Redirection

Bei HTTP Redirection wird anstelle der HTTP Response eine "HTTP Redirection Message" (HTTP 302 message) zurückgegeben, in der die Adresse des Servers steht, von dem der Client das angeforderte Objekt erhält. Hierbei gilt es zu beachten, daß die HTTP Redirection Response eventuell die gleiche Prozeßlast verursacht, wie die reguläre HTTP Response, was auf einem ohnehin schon überlasteten Server kein wünschenswerter Effekt ist. Daher ist es sinnvoll, diese Umleitungen mitunter schon von den Routern vor dem Server ausführen zu lassen. Hierzu bilden die Router Routingtabellen, in denen URL und dazugehörige IP Adressen gespeichert werden. Diese werden entweder durch andere Router oder die Content Server selbst aktualisiert.

3.2 Multihoming

Multihoming ist definiert als ein Netzwerk oder AS mit mehr als einer externen Verbindung, entweder zu einem oder mehr ISP (Internet service provider). Der Ansatz geht davon aus, daß sich durch die Anbindung an mehrere ISP oder AS sowohl Bandbreite als auch Stabilität bei den Verbindungen erhöhen. Untersuchungen haben ergeben, daß für k-Multihoming Performanceverbesserungen von

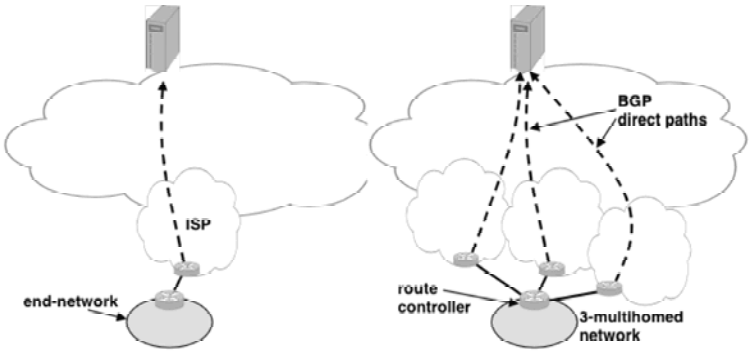

Abb. 5: ein ISP, BGP Routing (1-Multihoming) und Multihoming mit 3 ISPs (3-Multihoming)

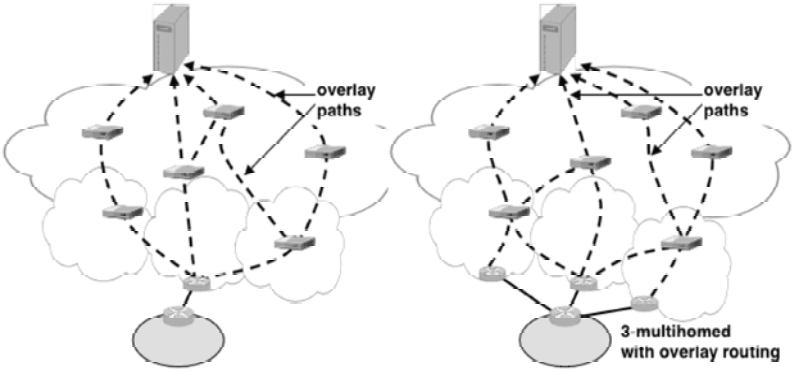

Abb. 6: ein ISP, Overlay Routing (1-Overlay) und Overlay Routing mit Multihoming (3-Overlay)

durchschnittlich 25% möglich sind für k≤4. Für k>4 sind die Performancezugewinne nur noch marginal ausgefallen. Außerdem hat sich gezeigt, daß die Wahl der richtigen Provider für Multihoming ein sehr wichtiger Faktor ist, um eine Performanceverbesserung zu erreichen [15].

3.3 Overlay Routing

Overlay Routing legt eine eigene Netzwerkstruktur über die vorhandene Infrastruktur des Internet, Resilient Overlay Networks (RON), um Verbindungen stabiler und eigenständig aufzubauen, bzw. auftretende Störungen zu erkennen und schnell durch die Nutzung anderer Routen zu umgehen. RON sondieren und überwachen beständig die Verbindungen zwischen den einzelnen Knoten und tauschen die Informationen über Verbindungsqualität zwischen Knoten selbständig aus. Daraus werden unter

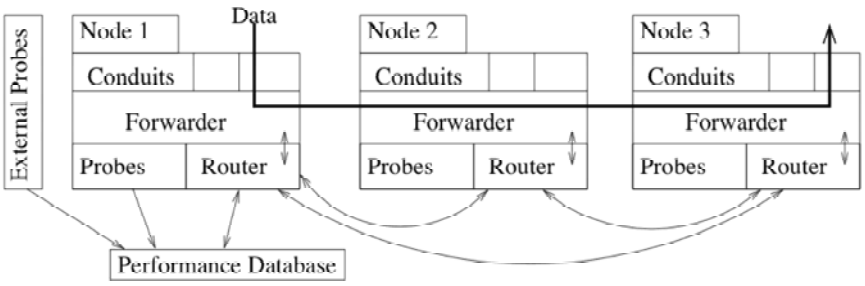

Abb. 7: RON Architektur

Berücksichtigung von Pfadmetrik, Latenz, Verlustrate und Bandbreite Tabellen zur Weiterleitung erstellt. Da die RON selbst die Pfadmetriken mit einer Kombination aus aktiven Sondierungsproben und passiver Überwachung des fließenden Datenverkehrs erstellen, sind diese Netzwerke nur für eine begrenzte Größe (2-50 Knoten) vorgesehen, um einer Auslastung der Bandbreite des Internet selbst durch die aktiven Proben vorzubeugen.

4 URL Router Architecture

Die Architektur des Routers kann einen nicht zu unterschätzenden Einfluß auf den Datendurchsatz im Netzwerk haben. Bei einem üblichen einarmigen URL Router übernimmt eine CPU alle Funktionen, also sowohl den Aufbau der TCP Verbindungen als auch die Auflösung der URL. Unzureichender

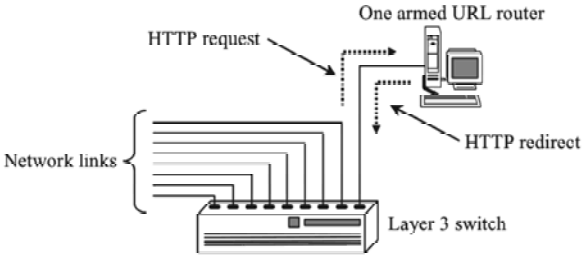

Abb. 8: Einarmiger URL-Router

Hauptspeicher, wenn die Routingtabellen nicht in den Speicher passen und auf die Festplatte ausgelagert werden müssen, kann die Performance noch weiter verschlechtern.
Ein spezieller URL Router hat zwei Prozessoren, einen für die TCP Verbindungen und einen zum auflösen der URL. Der Arbeitsspeicher sollte ausreichend groß sein, es gibt im folgenden aber noch Strategien, die Routingtabellen zu verkleinern.

4.1 Routing Tables

Mithilfe von Routingtabellen wird ermittelt, wohin eine Anfrage umgeleitet wird. In der Routingtabelle stehen URL und dazugehörige Adressen der Inhalte, sowie mitunter Auslastung und Distanz. Dabei wird mit Statusinformationen und dem Wissen um die Lage des Clients die bestmögliche Quelle für den gewünschten Inhalt ausgewählt.
Routingtabellen müssen in regelmäßigen Abständen aktualisiert werden, wobei die Router eigenständig Updates austauschen. Diese Updates können je nach Lebensdauer und Umfang auch einen merklichen Anteil am Datenverkehr zwischen den Routern ausmachen, weshalb man versucht ist, diese Datenmengen möglichst klein zu halten. Im allgemeinen verwendet man dazu Signaturen, die aus den URLs gebildet werden, um die URL zu verkürzen, z.B. CRC-Codes [3]. Die Routingtabellen werden dann mithilfe von Hashtabellen aus diesen Signaturen gebildet. Bei diesen automatisch generierten Signaturen muß allerdings beachtet werden, daß in seltenen Fällen, zwei unterschiedliche URLs auf die gleiche Signatur abgebildet werden. In diesem Fall spricht man von einer Kollision oder einem "False Hit". Es gibt drei Möglichkeiten zur Handhabung eines False Hits, einerseits kann man die Signaturen beim Generieren auf Eindeutigkeit überprüfen und gegebenenfalls eine Änderung zu erzwingen, anderenfalls kann man mittels einer weiteren HTTP redirection wieder auf die Originalseite verweisen.
Bei der dritten Möglichkeit wird ein sogenannter „Chained Hash Table“ erstellt, bei dem der kollidierende Hashwert ebenfalls wieder auf eine Liste zeigt, in der die einzelnen Hits hintereinander gespeichert sind.

URL 1	IPaddr1 (state), IPaddr2 (state), … IPaddrM_1 (state)
URL 2	IPaddr1 (state), IPaddr2 (state), … IPaddrM_2 (state)
URL 3	IPaddr1 (state), IPaddr2 (state), … IPaddrM_3 (state)
⋮	⋮
URL N	IPaddr1 (state), IPaddr2 (state), … IPaddrM_N (state)

Abb. 9: Struktur einer URL-Routing Tabelle

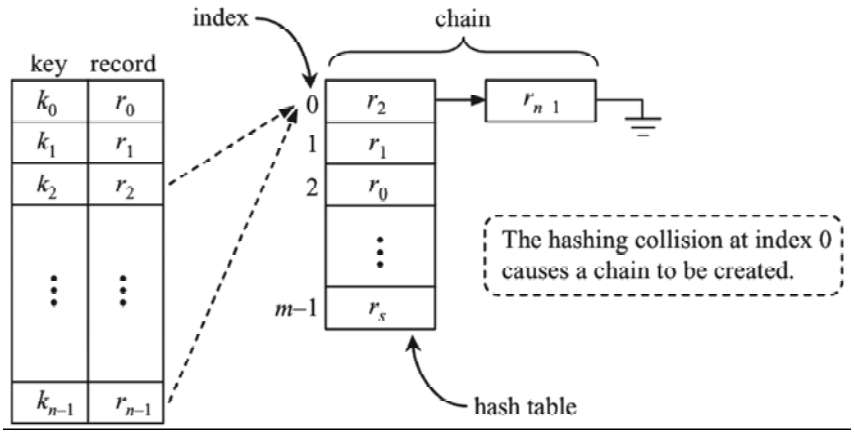

Abb. 10: Hash Kollision mit verketteter Hash Liste

4.2 Hashing Schemes

An welchen Server soll die Anfrage umgeleitet werden? Ein CDN ist meist aus geographisch sehr verteilten Surrogates zusammengesetzt. Das hat zur Folge, das klassische Routingverfahren für Cluster nicht so einfach adaptiert werden können, da diese leicht den aktuellen Status des Clusters ermitteln und zu grunde legen können. Eine Lösung ist, Hashingverfahren zu entwickeln, die URLs auf kleine Wertetabellen abbilden.
Dies hat einerseits den Vorteil, daß die Kommunikation zwischen den umleitenden Routern wegfällt, da die Wertetabellen nicht unabhängig davon, welcher Router die Anfrage umleitet, gleich sind, zum anderen kann der Umfang der Wertetabelle beeinflußt werden, um weniger Speicher anstelle von größerer Genauigkeit zu nutzen.

4.2.1 Modulo Hashing

Der klassische Ansatz, bei dem die URL abgebildet wird auf einen Wert modulo der Anzahl der Server. Der Vorteil dieser Methode besteht in der geringen Rechenleistung für den Hash Algorithmus. Allerdings erweist sich dieses Modell als unpassend, sobald sich das Set von Servern ändert. So veruracht ein zusätzlicher Server im Set einen erheblichen Aufwand.

4.2.2 Consistent Hashing (C-Hash)

Die URL wird in einen kreisförmig angeordneten Zahlenraum abgebildet, so wie die Namen der Server. Die URL wird demjenigen Server zugeordnet, der ihrem Hash-Wert im Kreis am nächsten liegt. Wenn ein Server in diesem Verfahren ausfällt, wird die Auslastung auf seine Nachbarn umgeleitet. Auf diese Weise verursacht das Hinzufügen oder Herausnehmen eines Servers nur lokale Änderungen der Zuordnung.

4.2.3 Highest Random Weight (HRW)

Erstellt und sortiert eine Liste von Hash-Werten aus den URL und den Namen aller Server. Jede URL zeigt nach einer festgelegten Reihenfolge auf die Server, so daß diese Liste durchlaufen wird, bis ein

angemessen ausgelasteter Server gefunden wird. Das Verfahren benötigt eine höhere Rechenleistung als Consistent Hashing, allerdings hat jede URL ihre eigene Reihenfolge, so daß ein ausgefallener Server seine Last auf alle anderen gleichmäßig verteilt.

4.3 Request Redirection Strategies

Im folgenden gibt es verschiedene Möglichkeiten, auf welche Art und Weise Anfragen an das CDN auf die einzelnen Surrogates verteilt werden können.

4.3.1 Statisches Server Set

Ein statisches Server Set besteht aus einer festen Anzahl Kopien jeder URL, die auf die Server im CDN verteilt werden, wobei die Anzahl der Kopien im regelfall kleiner als die Anzahl der Server ist.
Beim Replicated Consistent Hashing (RC-Hash) ist jede URL einer Gruppe von kopierten Servern zugeordnet, wobei die Anzahl der Kopien fest ist. Die URL wird in einen kreisförmig angeordneten Zahlenraum abgebildet, die Serverkopien werden, ausgehend vom Original, gleichmäßig auf diesen Zahlenraum verteilt. Das Netzwerk wird dabei wie eine einzige geographische Region behandelt.
Für das Replicated Highest Random Weight (R-HRW) Verfahren werden die Kopien der URL auf die ersten n Server, die nach dem HRW Verfahren ermittelt wurden, verteilt. Gegenüber dem RC-Hash wird dieses Verfahren weniger wahrscheinlich den gleichen Satz von Kopien für zwei verschiedene URLs generieren, wodurch weniger gefragte URLs, die einige Kopien auf den gleichen Servern wie gefragte URLs haben, auch Kopien auf weniger ausgelasteten Servern haben.

4.3.2 Load-Aware Server Set

Während beim statischen Server Set die Auslastung verteilt wird, indem die Anfragen zufällig auf die einzelnen Kopien verteilt werden, wird beim load-aware Server Set an dieser Stelle ein Verfahren eingesetzt, bei dem die Router lokale Abschätzungen der Server-Auslastung unterhalten und damit den Server mit der geringsten Last auswählen. Wie auch schon beim statischen Server Set gibt es auch hier zwei Varianten von Hash Verfahren, basierend auf RC-Hash oder R-HRW, LRC-Hash bzw. das Gegenstück LR-HRW.

4.3.3 Dynamisches Server Set

Hier wird die Anzahl der Kopien dynamisch angepaßt. Dadurch wird das Arbeitsvolumen auf den Servern reduziert und als Folge davon ein besseres Caching-Verhalten erzielt.
Beim Coarse Dynamic Replication Verfahren (CDR) wird die Anzahl der Kopien anhand von Serverauslastung und Nachfrage angepaßt. CDR basiert auf HRW und benutzt grobe Informationen über die Auslastung der Server, um den ersten "verfügbaren" Server aus der Liste auszuwählen. Dieser Prozeß wird von jedem Redirector (Router, der die Anfrage umleitet) unabhängig durchgeführt und nutzt den Auslastungsstatus der möglichen Server. Um nicht mit hohem Kommunikationsaufwand den Status des Servers abfragen zu müssen nimmt CDR die lokale Information, die jeder Redirector beobachtet, als Näherungswert. Im einfachsten Falle wird dabei auf die Anzahl der

aktiven Verbindungen zurückgegriffen, wobei dies leicht mit Latenz, verbrauchter Bandbreite o. ä. zu kombinieren ist. Wo Anfangs immer der gleiche Server gewählt wird, geht das Verfahren bei steigender Auslastung dazu über, die Anfragen über mehrere Server zu verteilen. Dies hat auch zur

```
find_server(url, S) {
        foreach server si in server set Si
                weighti = hash(url, adress(si));
        sort weight;
        foreach server sj in decreasing order of weightj {
                if satisfy_load_criteria(sj) then {
                        targetServer <- sj;
                        stop search;
                }
        }
        if targetServer is not valid then
                targetServer <- server with heighest weight;
        route request url to targetServer;
}
```

Abb. 11: Coarse Dynamic Replication

```
find_server(url, S) {
        walk_entry <- walkLenHash(url);
        w_len <- walk_entry.length;
        foreach server si in server  set S,
                weighti = hash(url, adress(si));
        sort weight;
        scandidate <- least-loded server of top w_len servers;
        if satisfy_load_criteria(scandidate) then {
                targetServer <- scandidate;
                if (w_len > 1 &&
                        timenow() - walk_entry.lastUpd > chgThresh)
                        walk_entry.length --;
        } else {
                foreach rest server sj in decreasing weight order {
                        if satisfy_load_criteria(sj) then {
                                targetServer <- sj;
                                stop search;
                        }
                }
                walk_entry.length <- actual search steps;
        }
        if walk_entry.length changed then
                walk_entry.lastUpd <- timenow();
        if targetServer is not valid then
                targetServer <- server with heighest weight;
        route request url to targetServer;
}
```

Abb. 11: Fine Dynamic Replication

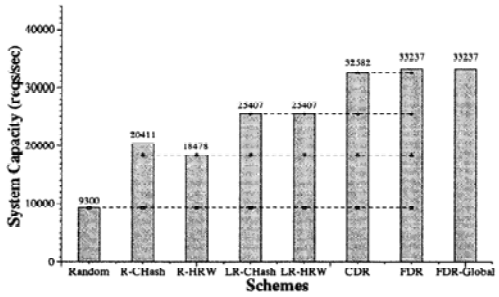

Figure 5: Capacity Comparison under Normal Load

Abb. 12: Vergleich der Kapazität unter normaler Auslastung

Category	Strategy	Hashing Scheme	Dynamic Server Set	Load Aware
Random	Random			No
Static	R-CHash	CHash	No	No
	R-HRW	HRW	No	No
Static +Load	LR-CHash	CHash	No	Yes
	LR HRW	HRW	No	Yes
Dynamic	CDR	HRW	Yes	Yes
	FDR	HRW	Yes	Yes
	FDR-Global	HRW	Yes	Yes
Network Proximity	NPR-CHash	CHash	No	No
	NPLR-CHash	CHash	No	Yes
	NP-FDR	HRW	Yes	Yes

Table 1: Properties of Request Redirection Strategies

Abb. 13: Eigenschaften von Request Redirection Verfahren

Folge, daß gefragte Dokumente von mehr Servern verteilt werden, als weniger gefragte, wobei es vorkommen kann, daß einige weniger gefragte Dokumente einfach deshalb kopiert werden, weil sie primär auf relativ ausgelasteten Servern liegen. Wenn einzelne Dokumente eine sehr große Nachfrage ereilen kann es dazu kommen, daß alle Server im System diese anbieten.
Das Fine Dynamic Replication Verfahren (FDR), greift zusätzlich auf Informationen zur Nachfrage nach einer URL, um die Anzahl der Kopien präziser als CDR anzupassen. Hierbei wird auch das Problem der unnötigen Kopien gelöst. Mit der Einführung einer feineren Buchführung soll der Möglichkeit eines sogenannten "ripple effects" in CDR entgegengewirkt werden, bei dem eine sehr große Nachfrage einer URL zur überlastung des ersten Servers führt, was wiederum die Auslastung anderer Server erhöht. Wenn diese auch überlastet werden, werden Anfragen nach Dokumenten auf diesen Servern auf sekundäre Server umgeleitet. Bei hoher Auslastung im gesamten System führt dies zum Verlust der Vorteile des Verfahrens bei der Verfügbarkeit. FDR nutzt eine Hilfsstruktur in jedem Redirector, die "walk length", die für jede URL angibt, wie viele Server sie nutzen sollte. Kommt eine Anfrage, nutzt der Redirector die "walk length" um den Server mit der niedrigsten Auslastung zu ermitteln. Wenn dieser auch ausgelastet ist, wird die "walk length" erhöht und der Prozeß wiederholt.
Beim "fdr with network proximity" (NP-FDR) wird anstelle der einfachen Näherungswerte für die Serverauslastung eine "effektive Auslastung" berechnet. Hierzu wird der Näherungswert der Auslastung mit einer "normalisierten" Distanz zwischen Redirector und Server multipliziert. Die reine Distanz zum Server wird mit "Round Trip Time" (RTT) Routing Hops o. ä. gemessen und durch die minimale beobachtete Distanz, gemessen mithilfe von Ping, Traceroute etc., geteilt. Durch die Nutzung der effektiven Auslastung werden nähere Server bevorzugt, wodurch die globale Netzwerkauslastung reduziert wird.

4.4 Content Routers

Content Routers (CR) sind eine erweiterte Form von Routern, die sowohl als IP-Router als auch als Name Server fungieren. Am sinnvollsten sind wohl Firewalls, Gateways und BGP-Router zu erweitern, anstatt alle Router zu CRs auszubauen. Die Name-Server Unterstützung funktioniert über das Internet Name Resolution Protocol (INRP), das rückwärtskompatibel zu DNS ist. Erreicht eine Anfrage den CR, schaut er in der Name Routing Tabelle nach dem nächsten CR entsprechend den vorhandenen Informationen bezüglich der Route, so daß insgesamt die "bestmögliche" Route ausgewählt wird. Wenn der CR in unmittelbarer Nachbarschaft zum besten Content Server erreicht ist, sendet dieser eine Antwort, die die Adresse des Content Servers enthält. Wird keine Antwort zurückgegeben, können

die angesprochenen CRs alternative Routen auswählen oder die Namenssuche erneut versuchen. Auf diese Art wird der "beste" Weg ausgewählt und gleichzeitig können Serverausfälle oder veraltete Routing Informationen ausgeglichen werden. CRs sollten Informationen aus dem IP-Routing auch auf das Content Routing anwenden, zum Beispiel wenn ein Peer unerreichbar wird, gilt das auch für alle Inhalte, die über diesen Peer erreichbar sind. Desweiteren müssen die IP-Routingmethoden und die Content-Routingmethoden einheitlich sein, so daß Entscheidungen auf Content-Ebene auch auf der IP-Ebene ausgeführt werden können.

4.4.1 Name Based Routing

Das Name Based Routing Protocol (NBRP) ist in seiner Struktur dem BGP sehr ähnlich, wo das BGP über verschiedene AS (Autonomous System) hinweg über die Erreichbarkeit von Adress-Präfixen Informiert, verteilt NBRP Informationen über die Erreichbarkeit von Namensendungen an CRs. Eine NBRP-Routingankündigung enthält jeweils den Weg der CRs bis zu einem Content Server. Im einfachsten Falle enthält ein BGP Protokoll einen Adressbereich, die Adresse des nächsten Routers und eine Liste von AS über die die angekündigte Route den Datenverkehr schleust, im Falle von NBRP sind dies die Adresse des gewünschten Inhalts, die Adresse des nächsten CR oder des Content Servers und der Pfad der Router über den der Zugriff läuft. Content Servers können dem noch Daten über die eigene Auslastung hinzufügen, die den Inhalt weiter entfernt scheinen lassen aus Sicht der CR.
Ein Vorteil in der Nutzung von INRP und NBRP ist, daß eine Anfrage innerhalb einer RTT zu einem Content Server in der Nähe des Clients geleitet wird.
NBRP unterstützt die Gruppierung von Adressen, die auf die gleiche Routing Information verweisen, wodurch Routing Updates erheblich verkleinert werden. Diese "routing aggregates" (RA) enthalten eine Seriennummer, anhand derer sich Änderungen bemerken lassen. Die Inhalte lassen sich über eine INRP Anfrage an den Urheber des RA ermitteln.

5 Andere Ansätze

Content Delivery ist ein sehr weitläufiges Thema, daß auf verschiedenste Arten zu lösen versucht wird. Im Folgenden werden noch einige Beispiele vorgestellt, die sich mit anderen Methoden zu diesem Thema befassen.

5.1 Kombination von CDN und P2P Netzwerken

Wenn ein Inhalt im lokalen Netz verfügbar ist, soll er durch ein P2P Netzwerk übertragen werden, wenn nicht, wird die Anfrage an einen übergeordneten Cache-Server außerhalb des P2P Netzes weitergeleitet. Y. Fujita et al. stellen einen statischen Algorithmus wie folgt vor [4]:

Schritt 1:
Auf die Anfrage eines Clients wird ein Cache Server ermittelt, der die benötigten Inhalte bereitstellt. Der Client hält die erstellte TCP Verbindung über die Anfrage aufrecht.

Schritt 2:
Anschließend sendet der Client die Titelanfrage an das P2P Netzwerk. Über einen distributet Hash-

table (DHT) findet der Client einen Peer, der den gewünschten Inhalt anbietet.

Schritt 3:
Unter Berücksichtigung einiger Auswahlkriterien wie RTT, wird eine Verbindung ausgewählt, von der die Inhalte geladen werden, wobei P2P aufgrund der Lokalität mit recht hoher wahrscheinlichkeit gewählt wird.

In einem weiteren dynamische Verfahren wird der letzte Schritt um einige methoden wereitert [4]. Es wird ein Agent auf Client und Server installiert, der den Durchsatz in Intervallen überwacht, zwischen Client, Peer und Server Informationen bezüglich Data Mining austauscht und den Status des Netzwerkpfades beständig Überwacht. Die Verbindung zwischen Cache Server und Client unterhält eine dummy-Verbindung mit geringer Bandbreite, um den Status in echtzeit zu überwachen. Letzteres Verfahren ist hilfreich, um schnell die Verbindung umschalten zu können, wenn der Peer ausfällt.
Ein weiterer interessanter Ansatz ist der gedanke, P2P Verbindungen mit mehreren Peers oder Replica Servern gleichzeitig auf CDN umzusetzen [5].

5.2 TRIAD

"translating relaying Internet architecture integrating active directories" oder TRIAD ist ein "vielversprechender Kandidat für die nächste Generation der Internet-Architektur" [16]. TRIAD versucht nicht das Content Delivery Problem über Algorithmen im bestehenden Netz zu lösen, sondern stellt eine komplette neue Architektur vor. Es gibt eine eigene Content Layer, die explizit Verfahren zum Content Delivery bereitstellt, aber auch eine neue Namensadressierung und eigene Routingprotokolle.

5.3 Onion Routing

Onion Routing setzt sich weniger mit der Suche nach einer optimalen Verbindung zwischen Server und Client auseinander, sondern stellt ein Verfahren zur Anonymisierung von Datenverkehr dar. Die einzelnen Datenpakete ("Onions" fester Größe, meist 128 Byte) werden von jedem speziellen Router auf dem Weg mit dem eigenen Public Key entschlüsselt. Die entschlüsselten Daten enthalten einige Kontrollinformationen, die Identität des nächsten Routers auf dem Weg und die eingebettete Onion, die an den nächsten Router weitergeleitet wird. Eine Onion ändert bei jedem Durchlauf durch einen Knoten den Inhalt bzw. das Bitmuster (aufgrund der Verschlüsselung), aber nicht die Größe, so daß ein Lauscher den Paketen nicht durch das Netzwerk folgen kann.

Literatur

[1] L. Wang, V. Pai, and L. Peterson. The effectiveness of request redirection on CDN robustness. In Proceedings of The Fifth Symposium on Operating Systems Design and Implementation (OSDI'02), Boston, MA, December 2002.

[2] GRITTER, M., AND CHERITON, D. An architecture for content routing support in the Internet. In Proc. USENIX USITS (Mar. 2001).

[3] Z.G. Prodanoff and K. J. Christensen. Managing routing tables for URL Routers in content distribution networks. International journal of network management 2004, 14: 177-192.

[4] Yoshikatsu Fujita, Jun Yoshida, Kenichi Yoshida and Kazuhiko Tsuda.Network Information Mining for Content Delivery Route Control in P2P Network. Knowledge-Based Intelligent Information and Engineering Systems: 8th International Conference, KES 2004, Wellington, New Zealand, September 20-25, 2004, Proceedings.

[5] Elisa Turrini, Fabio Panzieri. Using P2P Techniques for Content Distribution Internetworking: A Research Proposal. Proceedings of the Second International Conference on Peer-to-Peer Computing Page: 171 IEEE Computer Society 2002

[6] A Comparison of Overlay Routing and Multihoming Route Control
Aditya Akella (CMU), Jeffrey Pang (CMU), Anees Shaikh (IBM Research), Bruce Maggs (CMU), Srinivasan Seshan (CMU). Proceedings of SIGCOMM 2004

[7] L. Amini, A. Shaikh, and H. Schulzrinne, "Modeling Redirection in Geographically Diverse Server Sets", Proc. of 12th International World Wide Web Conference (WWW 2003), May 2003.

[8] V. Paxson. End-to-End Routing Behavior in the Internet. IEEE/ACM Transactions on Networking, 5(5):601--615, Oct. 1997.

[9] Akamai. Internet Bottlenecks. 2000 Akamai Technologies Inc. www.akamai.com

[10] David Goldschlag, Michael Reed, and Paul Syverson. Onion routing for anonymous and private Internet connections. Communications of the ACM, 42(2):39--41, 1999.

[11] P. Syverson, G. Tsudik, M. Reed, and C. Landwehr. Towards an analysis of onion routing security. In Workshop on Design Issues in Anonymity and Unobservability, July 2000.

[12] Marcel Rosu and Daniela Rosu, "Evaluation of TCP splice benefits in Web proxy Servers," Tech. Rep. RC 22159, IBM Research Report, August 2001.

[13] ANDERSEN, D. Resilient overlay networks. Master's thesis, Department of EECS, MIT, May 2001.

[14] A. Cohen, S. Rangarajan, and H. Slye. On the Performance of TCP Splicing for URL-Aware Redirection. In Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems, Boulder, CO, Oct. 1999.

[15] A Measurement-Based Analysis of Multihoming
Aditya Akella, Bruce Maggs, Anees Shaikh, Ramesh Sitaraman, Srinivasan Seshan
Proceedings of SIGCOMM 2003

[16] D. R. Cheriton and M. Gritter. TRIAD: A new next generation Internet architecture, March 2000.