Zugangskontrolle für einen Videoverteildienst mit IP Multicast

Roland Balmer und Torsten Braun

Institut für Informatik und angewandte Mathematik (IAM), Universität Bern,
Neubrückstrasse 10, CH-3012 Bern, Schweiz
{balmer,braun}@iam.unibe.ch
http://www.iam.unibe.ch/~rvs/

Zusammenfassung Für die Übertragung von Information über das Internet, bei der gleichzeitig eine grosse Zahl von Empfängern erreicht werden soll, wird am besten IP-Multicast eingesetzt. Um die Übertragung zu empfangen, muss jeder Empfänger der entsprechenden Multicast Gruppe beitreten. Wird auf diese Art ein kommerzieller Dienst, wie z.B. Pay-TV, angeboten, muss sichergestellt werden können, dass die Interessen, wie z.B. Urheberrechte und Qualität, des Dienstanbieters sichergestellt sind. Um die Urheberrechte zu schützen kann man die Daten verschlüsseln, während man für die Stabilität des Dienstes die Übertragung im Internet mit bestimmten Prioritäten ausstatten muss. Um die Kosten möglichst gering zu halten, müssen diese Prioritäten auf die zahlenden Kunden, die auch anonym sein können, beschränkt werden. In diesem Beitrag wird ein Konzept vorgestellt, bei dem sich die Empfänger gegenüber dem Netzwerkbetreiber identifizieren, so dass diese von der privilegierten Datenübertragung profitieren können.

1 Einleitung

Mit den weit verbreiteten und immer leistungsfähigeren Zugangsmöglichkeiten zum Internet ist dieses Medium zum Verteilen von grossen Datenmengen interessant geworden. Vor allem werden immer öfters Dienste, wie z.B. Videokonferenzen, genutzt, die empfindlich auf grosse Verzögerungen oder auf Paketverlust reagieren. Damit solche Dienste problemlos funktionieren, müssen die im Internet transportierten Daten privilegiert behandelt werden. Techniken, um die bestimmten Dienstgüten (Quality-of-Service, QoS) zu garantieren existieren in der Form von Integrated und Differentiated Services. Das Aufsetzen dieser QoS-Unterstützung für eine Ende-zu-Ende Verbindung (Unicast) bereitet keine grossen Probleme, wenn die beiden Endpunkte bekannt sind. Dann ist der Pfad zwischen Sender und Empfänger bekannt und alle dazwischen liegenden Knoten können entsprechend konfiguriert werden. Dieser Ansatz kann auch für eine kleine Anzahl von Empfängern eingesetzt werden. Jedoch wächst mit jedem weiterem Empfänger die benötigte Bandbreite und dies kann vor allem beim Sender zu einem Engpass führen.

Um Netzwerkressourcen zu sparen, bietet sich die Verwendung von IP-Multicast für den Datentransport an. Dabei ist das Grundkonzept für IP-Multicast relativ einfach: Jeder Empfänger, der Daten empfangen will, tritt der entsprechenden Multicast-Gruppe bei und erhält die an die Multicast-Adresse gesendeten Daten. Dabei werden die Daten mit Hilfe des Multicast-Baumes über das Internet transportiert. Wird für den Multicast-Dienst ebenfalls QoS benötigt, so kann man auch hier die gleichen Techniken wie für Unicast nutzen, z.B. Differentiated Services. Bei IP-Multicast entstehen dabei jedoch neue Probleme. Solange im Multicast-Baum alle Empfänger gleich behandelt werden, erhalten alle Verbindungen im Multicast-Baum die gleichen Ressourcen zugewiesen. Im Falle eines kostenpflichtigen Dienstes sollten diese Ressourcen aber nur an zahlende Kunden vergeben werden. Wenn aber

anonyme Empfänger erlaubt sein sollen, kann der Dienstanbieter nicht angeben, wo im Multicast-Baum sich alle autorisierten Kunden befinden. Um trotzdem den Zugang zur Multicast-Gruppe einzuschränken, was bei Multicast unüblich ist, wird eine Zugangskontrolle benötigt, mit der festgestellt werden kann ob ein Mitglied der Multicast-Gruppe auch ein zahlender Kunde ist. Mit dieser Zugangskontrolle kann man dann die Ressourcen auf die Verbindungen beschränken, die von autorisierten Kunden benötigt werden. Mit dieser Zugangskontrolle kann man auch ein weiteres Problem kontrollieren. So kann, im Gegensatz zu Unicast, bei Multicast die Anzahl und die Position der Empfänger variieren und es ändern sich dadurch auch die Ressource-Bedürfnisse für einen bestehenden Service. Sobald ein neuer Multicast-Ast dazu kommt, kann das Problem, dass auf dieser Verbindung keine Ressourcen für den Dienst zur Verfügung sind, entstehen. Mit der Zugangskontrolle kann man nun die Zuteilung der Ressourcen für diese Verbindung blockieren, bis sichergestellt ist, dass die bestehenden Vereinbarungen nicht verletzt werden.

Ein Zugangskontrollmechanismus ist Gegenstand dieses Beitrages und im Abschnitt 4 wird eine Architektur für eine Zugangskontrolle für einen Multicast basierten Videoverteildienst vorgestellt.

Mit der Zugangskontrolle könnte man auch den Schutz der transportierten Daten kontrollieren, aber hierfür kann man viel einfacher auf konventionelle Verschlüsselungsmethoden zurückgreifen. In Abschnitt 2 wird ein Videoverteildienst vorgestellt, der eine solche Methode verwendet und als Grundlage für die Zugangskontrolle fungiert. Im Abschnitt 3 werden einige andere Konzepte vorgestellt und im Abschnitt 5 wird eine prototypische Implementierung und die damit erzielten Ergebnisse präsentiert. Abschnitt 6 schliesst den Beitrag ab.

Im nächsten Abschnitt wird jedoch zunächst ein Überblick über das Szenario des Videoverteildienstes gegeben, welches entscheidend die Anforderungen an die QoS-Unterstützung bestimmt hat.

2 Szenario des Videoverteildienstes

Die Grundprinzipien eines Videoverteildienstes sind unabhängig vom Medium, das verwendet wird, um die Daten zu transportieren. Bei bestehenden Pay-TV Betreibern steht dabei im Vordergrund, dass nur die zahlenden Kunden die ausgestrahlten Sendungen empfangen, bzw. betrachten können. Dabei profitieren die Betreiber von zwei günstigen Rahmenbedingungen. Erstens verfügen Sie über ein Transportmedium, z.B. Satellit oder Kabelfernsehen, das ihnen exklusiv zur Verfügung steht. Zweitens benützt jeder Kunde einen vom Dienstanbieter zur Verfügung gestellten Empfänger, der am Fernseher angeschlossen werden muss. Um sicherzustellen, dass nur die zahlenden Empfänger die Sendungen mitverfolgen können, werden die Daten verschlüsselt übertragen. Auch die Bezahlung ist einfach, da die Empfänger eine monatliche Gebühr bezahlen.

Für die Kunden ist dieses Geschäftsmodell nicht attraktiv, da sie auch bezahlen, wenn sie den Service nicht benutzen. Um dieses Bedürfnis der Kunden zu befriedigen, kann man ein Pay-Per-View Szenario verwenden. Dabei zahlt der Kunde nur noch für die Sendungen, die er auch angesehen hat. Die Sendungen werden aber weiterhin nach einem festen Zeitplan ausgestrahlt. Durch die immer höher werdende Kapazität des Internet, ist es auch attraktiv dieses Medium zu nutzen. Verwendet man das Internet als Medium, so bietet sich an auf eigene Empfänger-Hardware zu verzichten, dafür die vorhandene Hardware der Kunden zu nutzen und nur die entsprechende Software zur Verfügung zu stellen. Verwendet man aber nur noch Software, über deren endgültige Verteilung man keine Kontrolle mehr hat, so braucht man einen Mechanismus, um die Interessen des Betreibers zu schützen. Ausserdem soll es ebenfalls möglich sein, dass Benutzer gegenüber dem Betreiber

anonym bleiben können, bzw. mit dem Betreiber keinen direkten Vertrag haben. Der letzte Fall trifft vor allem dann ein, wenn man das Geschäftsmodell so erweitert, dass die Betreiber der Pay-Per-View Infrastruktur diesen Service nicht mehr selbst vermarkten, sondern dies über Internet-Händler tun. Dabei kaufen die Kunden die Sendungen, die sie sehen wollen, bei einem Drittanbieter ein, erhalten von ihm alle Informationen, um dann die Sendung direkt vom Betreiber empfangen zu können..

Im Interesse des Betreibers und der Kunden müssen folgende drei Bedingungen erfüllt sein:

- Der Betreiber soll für den Dienst korrekt entschädigt werden. Die notwendigen Transaktionen sollen sicher sein.
- Da ein gemeinsam genutztes Medium verwendet wird, müssen die Sendungen privilegiert transportiert werden, so dass keine Datenverluste auftreten und die Verzögerungen möglichst gering gehalten wird.
- Die Urheberrechte der gesendeten Videodaten müssen geschützt werden.

Aus Grund dieser Bedingungen wurde ein Szenario entwickelt, das in Abbildung 1 dargestellt wird. Dieses Szenario kann in verschiedene Teile unterteilt werden, die

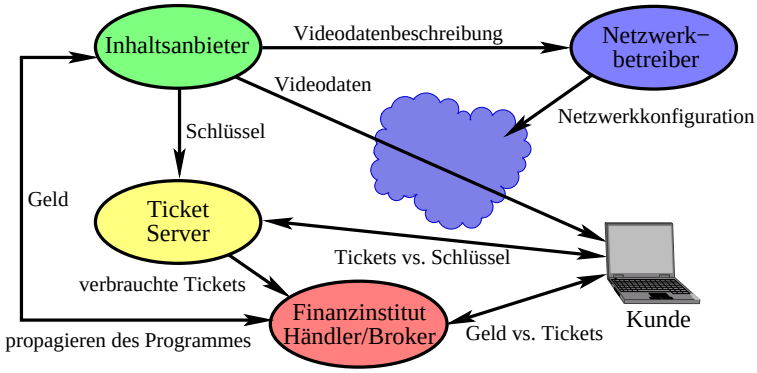

Abbildung 1. Alle Komponenten und deren Relationen innerhalb des Videoverteiler-Szenario.

Zeitlich getrennt sind. Am Anfang propagiert der Inhaltsanbieter sein Programm an die Händler die seine Produkte vermarkten. Sucht ein Kunde nach einer bestimmten Sendung, so kontaktiert er die entsprechenden Händler, bis er die Sendung mit dem ihm passenden Konditionen gefunden hat. Sobald er die Sendung angewählt hat, bezahlt er den Dienst und erhält dafür Tickets, eine Art Gutschein, mit denen er nachweisen kann, dass er für den Service bezahlt hat. Ausserdem erhält er alle relevanten Informationen um die Sendung zu empfangen. Die Bezahlung selbst kann z.B. mit Kreditkarte erfolgen, ist aber in diesem Zusammenhang nicht von Bedeutung.

Sobald der Kunde im Besitz der Tickets ist, beginnt für ihn die nächste Phase. Da die Sendung verschlüsselt ist, benötigt der Kunde die entsprechenden Schlüssel zum Entschlüsseln. Vom Händler erfährt der Kunde, welcher Ticket Server für die entsprechenden Datenübertragung zuständig ist und ihm die dazugehörigen Schlüssel übermitteln kann. Für jedes Ticket, das an den Ticket Server geschickt wird, erhält der Kunde einen Schlüssel. Dieser Schlüssel ist nur eine beschränkte Zeit gültig, und vor Ablauf dieser Zeit muss der nächste Schlüssel mit dem nächsten Ticket angefordert werden. Dies ermöglicht dem Kunden die Übertragung zu verlassen und sich gegebenenfalls das Geld, für die nicht genutzten Tickets, zurückerstatten zu lassen.

Die Tickets sind so aufgebaut, dass man aus dem neusten Ticket seinen Vorgänger berechnen kann. Auf diese Art kann der Ticket Server verifizieren, ob das neue Ticket korrekt ist. Die Übertragung der Schlüssel erfolgt mit einer abgesicherten Verbindung zwischen dem Ticket Server und dem Kunden. Die Schlüssel erhält der Ticket Server direkt vom Inhaltsanbieter, liefert diesem jedoch keine Informationen zurück. Von Zeit zu Zeit schickt der Ticket Server alle gebrauchten Tickets an das Finanzinstitut, welches dann die Abrechnung durchführt und das Guthaben an die verschiedenen Inhaltsanbieter weiterleitet. Mehr Informationen über die finanzielle Transaktion sind in [BB01] zu finden.

Damit ist die erste Bedingung erfüllt. Die zweite Bedingung kann erfüllt werden, indem der Multicast-Baum die entsprechenden Prioritäten erhält. Für die Zuweisung der Ressourcen zum Multicast-Baum ist der Netzwerkbetreiber zuständig. Dafür erhält er direkt vom Inhaltsanbieter die Informationen, die den Datenstrom beschreiben. Darin enthalten sind Informationen über die benötigte Bandbreite, die maximal erlaubte Verzögerung und natürlich die Absender- und die Empfängeradresse. Da die Übertragung, wie bereits erwähnt, mit IP-Multicast erfolgt, nützt die Empfängeradresse dem Netzwerkbetreiber nicht viel. Das Problem ist, dass der Inhaltsanbieter keine weiteren Informationen über die Empfänger hat. Ausserdem soll die Reservierung der Ressourcen vor Beginn der Sendung erfolgen, obwohl zu diesem Zeitpunkt der Multicast-Baum noch nicht existiert. Die einfachste und ein

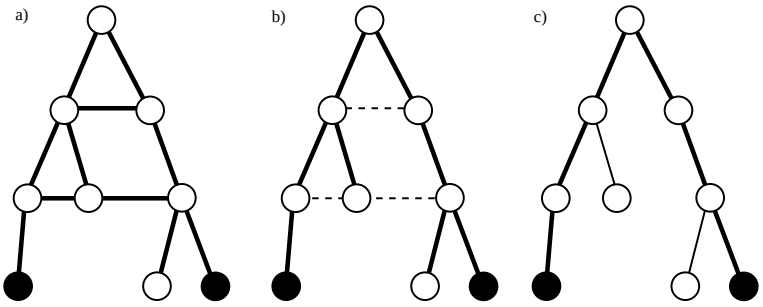

Abbildung 2. Die verschiedenen Phasen bei der Zuweisung der Ressourcen. a) Die Ressourcen werden für alle möglichen Verbindungen reserviert. b) Die Ressourcenreservierung, nachdem die Übertragung gestartet wurde. c) Das endgültige Ziel, dass nur die Kunden reservierte Ressourcen verbrauchen. Kunden sind mit schwarzen Kreisen markiert.

wenig unsaubere Lösung dafür ist, dass man die Reservierung provisorisch für alle möglichen Verbindungen einrichtet, was Abbildung 2.a) entspricht. Sobald der Dienst gestartet ist, z.B. mit einem Testbild oder einer Begrüssungsseite, und die Kunden dem Dienst beigetreten sind, kann man dem Multicast-Baum die Ressourcen zuweisen und die Reservierung für alle nicht benutzten Verbindungen freigeben (Abbildung 2.b)). Auf diese Art kann man die Ressourcen relativ schnell auf den Multicast-Baum beschränken, jedoch hat man weiterhin das Problem, dass sich Empfänger im Multicast-Baum befinden können, die keine zahlenden Kunden sind, die aber trotzdem Ressourcen verbrauchen. Um das Problem der störenden Empfänger und des unsauberen Anfangs zu lösen, kann man eine Zugangskontrolle einsetzen. Dabei müssen die Kunden dem Netzwerkbetreiber gegenüber beweisen, das sie autorisierte Kunden sind und somit kann, das in Abbildung 2.c) dargestellte Ziel, erreicht werden. Es soll weiterhin die Möglichkeit bestehen, dass die Kunden anonym sind. Es sollen also keine persönlichen Informationen ausgetauscht werden. Mit dieser Zugangskontrolle kann ausserdem das Problem abgefangen werden, das entsteht, wenn ein Kunde erst später der Übertragung beitritt. Dabei kann das Pro-

blem auftauchen, dass ein neuer Multicast-Ast entsteht, für den nicht mehr genug Ressourcen vorhanden sind und deshalb die Vereinbarung zwischen dem Inhaltsanbieter und dem Netzwerkbetreiber verletzt werden würde. Diese Problem wird unter anderem in [BW02] und [Sch01] tiefer erläutert. Wie eine solche Zugangskontrolle aussehen kann, ist in Abschnitt 4 beschrieben. Weitere Informationen über die Zuweisung der Ressourcen sind in [BGB01] und [SB03] enthalten.

Der dritte Punkt ist der Schutz des Urheberrechtes der übertragenen Videodaten. Wie bereits vorher erklärt, werden die Daten verschlüsselt übertragen. Damit kann aber nur sichergestellt werden, dass die Daten nur von berechtigten Kunden betrachtet werden können. Will man aber auch weitergehende Urheberrechte, wie z.B. das unerlaubte Verbreiten von Kopien verhindern, muss der Schutz erweitert werden. Dies geschieht, indem in die übertragenen Daten Wasserzeichen eingefügt werden. Da das Verfahren zum Einfügen von Wasserzeichen Einfluss auf die Zugangskontrolle hat, wird diese hier ausführlicher besprochen.

Wie bereits erwähnt, dient das Wasserzeichen dazu, die Urheberrechte zu schützen. Vorallem soll es mit Hilfe des Wasserzeichens möglich sein, illegale Kopien zu identifizieren und somit den Urheber dieser Kopien festzustellen. Damit dies möglich ist, muss jeder Kunde eine einzigartige Kopie erhalten. Wie dies mit dem Wasserzeichen erreicht wird, wird in zwei Schritten erklärt. Zuerst wird gezeigt, wie das Wasserzeichen überhaupt erzeugt wird und im zweiten Schritt wird gezeigt, wie Sichergestellt wird, dass jeder Kunde eine einzigartige Kopie der Übertragung erhält.

Das Wasserzeichen wird direkt in jedes Bild eingefügt (siehe Abbildung 3). Das Wasserzeichen wird in einige der 8x8 Pixel grossen Blöcke eingefügt, die nach der DCT Transformation entstehen. Damit das Wasserzeichen nicht zu einfach entfernt werden kann, aber die Qualität des Videobildes nicht beeinträchtigt wird, wird es nur in einem kleinen Teil des DCT-Blockes eingefügt. Um dies näher zu erläutern,

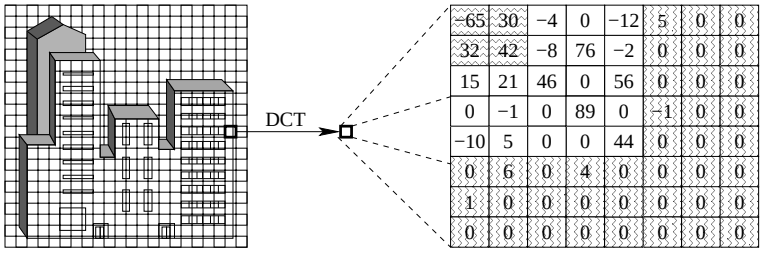

Abbildung 3. Die drei verschiedenen Regionen innerhalb eines DCT transformierten 8x8 Pixel grossen Block: major, markiert und bulk.

wird der 8x8 Pixel Block in drei Regionen aufgeteilt.

- **(major):** Diese Region enthält die wichtigsten Werte des Blockes. Das Einfügen des Wasserzeichens in diesem Bereich würde die Qualität des Videobildes stark beeinträchtigen.
- **(bulk):** Diese Region enthält die unwichtigsten Werte des Blockes. Meistens sind diese Werte nahe bei Null. Ein Wasserzeichen in dieser Region könnte ohne Beeinträchtigung des Videobildes entfernt werden, z.B. in dem alle Werte einfach auf Null gesetzt werden. Ausserdem ist die Wichtigkeit dieser Werte so gering, dass diese Werte unverschlüsselt übertragen werden können.
- **(markiert):** Diese Region repräsentiert diejenigen Werte, die wichtig genug sind, so dass sie nicht ohne sichtbaren Qualitätsverlust entfernt werden können. Sie sind aber nicht zu wichtig, so dass eine kleine Änderung die Qualität des Videobild nicht entscheidend beeinträchtigt. Ausserdem ist diese Region gross

genug, damit man das Wasserzeichen nicht zu leicht erkennen kann und die mögliche Anzahl der Wasserzeichen gross genug ist.

Das Wasserzeichen ist relativ robust gegenüber einer Manipulation, da es relativ gut im Bild versteckt ist. Wenn man nicht genau weiss wie das Wasserzeichen (Bitfolge) aussieht, ist es schwer entfernbar, ohne dass die Qualität beeinträchtigt wird. Jedoch ist es auf diese Art noch nicht möglich, dass jede Kopie einzigartig ist, da ja jeder Kunde die gleiche Kopie erhält. Um nun sicherzustellen, dass jeder Kunde eine einzigartige Kopie der Übertragung erhält, werden die Daten auf einen spezielle Art transportiert. Dafür wird das Videobild in verschiedene Teile unterteilt. Der grösste und unwichtigste Teil (bulk) wird ohne spezielle Vorverarbeitung in einem eigenen Multicast-Baum gesendet. Würde dieser Bulk-Teil, der etwa 90% der Daten ausmacht, alleine betrachtet, so würde man ein grösstenteils schwarzes Video bekommen.

Der Rest der Videodaten wird anders behandelt. Abbildung 4 zeigt die vier Schritte die auf die wichtigen Videodaten angewendet werden.

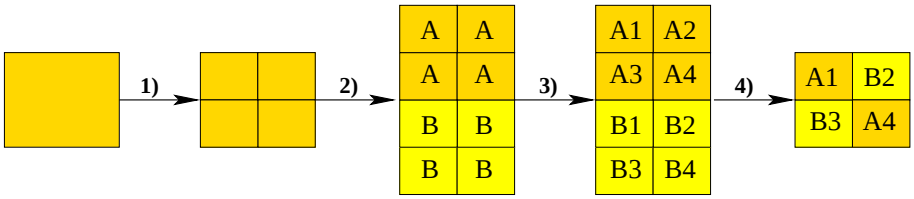

Abbildung 4. Die vier Schritte von einem Bild zu einem einzigartigen und markierten Bild.

1. Jedes Bild wird in verschiedene Segmente unterteilt. Im Beispiel werden so vier verschiedene Segmente erzeugt.
2. Jedes dieser Segmente wird kopiert. Im Beispiel existieren nun zwei Kopien von jedem der vier Segmente. Zu diesem Zeitpunkt sind die beiden Kopien A und B noch identisch.
3. Jedes Segment erhält ein individuelles Wasserzeichen. Nun sind die Segmente A1 und B1 voneinander unterscheidbar.
4. Jeder Kunde erhält eine individuelles Kombination von Segmenten. Im Beispiel erhält der Kunde die Kombination (A1,B2,B3,A4). Verwendet man zwei verschiedenen Kopien des Videos, sind $2^4 = 16$ verschiedene Kombinationen möglich.

Für den Transport wird jedes Segment (z.B. A1) verschlüsselt und mit Multicast transportiert. Für alle Segmente (z.B. A1) mit dem gleichen Wasserzeichen wird ein eigener Datenstrom erzeugt (in unserem Beispiel werden acht neue Ströme erzeugt). Da nur ca. 10% der Daten kopiert werden erhöht sich die Bandbreite in unserem Beispiel aber nur um ca. 10%. Mit den Schlüsseln vom Ticket Server kann der Kunde nur die Segmente entschlüsseln, die für ihn vorgesehen sind. Damit erhält jeder Kunde eine eigene Kopie. Für weitere Information über die Videobehandlung wird auf [KT00][TK00] verwiesen.

3 Andere Ansätze

Alternativ hätte auf bestehende Ansätze und Entwicklungen zurückgegriffen werden können. Aufgrund der Popularität von IP-Multicast im Forschungsbereich existieren einige Arbeiten auf diesem Gebiet.

Dabei gibt es zwei Hauptrichtungen. Die eine Richtung besteht darin die Übertragung der Multicast-Daten sicherer zu gestalten. Diese Richtung wird vorallem durch die Secure Multicast Arbeitsgruppe der IETF (SMuG) [IETb] forciert. Die Idee dahinter ist, dass der Empfänger von Daten verifizieren kann, dass diese auch vom korrekten Sender kommen. Dabei ist der Hauptaspekt auf Seiten des Senders, der sich gegenüber der Multicast-Gruppe identifizieren muss. Da in unserem Fall die Daten sowieso verschlüsselt übertragen werden, erübrigt sich hier ein weiterer Vergleich. Ein ähnlicher Ansatz wird auch in [MON] verfolgt. In unserem Szenario, wo die Daten ebenfalls verschlüsselt übertragen werden, ist die Identität verifiziert, wenn die Daten entschlüsselt werden können.

Eine weitere Richtung besteht darin, den Zugang zur Multicast-Gruppe zu regeln. Auch in diesem Bereich ist IETF mit der Multicast and Anycast Group Membership Arbeitsgruppe (magma) [IETa] aktiv. Das Hauptaugenmerk liegt dabei auf der Frage, ob auf dem entsprechenden Link ein autorisierter Empfänger vorhanden ist oder nicht. Dies wird erreicht, indem das Protokoll für den Zugang zur Multicast-Gruppe (IGMP) angepasst wird, was bei unserem Ansatz nicht nötig ist. Ein ähnlicher Ansatz wird in [IYT01] vorgestellt. Auch in diesem Ansatz wird eine Zugangskontrolle durch die Anpassung von IGMP gemacht. Dieser Ansatz erzeugt Probleme, da dadurch die Rückwärtskompatibilitat erschwert wird. Ausserdem kann der Urheber der Multicast-Gruppe Daten erfassen, die Rückschluss auf den Kunden geben. Solange die Kunden mit dem Sammeln dieser Informationen einverstanden sind, ist dies unproblematisch. Will er aber anonym bleiben, entsteht ein Problem.

4 Zugangskontrolle

Der für die Zugangskontrolle verantwortliche Netzwerkbetreiber verfügt nicht über mehr Informationen, als er vom Inhaltsanbieter bekommt. Im besonderen besitzt er keine Informationen darüber, wer berechtigt ist, entsprechenden Ressourcen für seine Datenübertragung zu bekommen. Es wird also ein Mechanismus benötigt, der es ermöglicht zu verifizieren, ob ein Empfänger ein autorisierter Kunde ist. Wie ein solcher Mechanismus aussehen kann, wird in diesem Abschnitt beschrieben.

In unserer Architektur (Abbildung 5) wird ein Protokoll zwischen dem Inhaltsanbieter und dem Empfänger verwendet, um die Authentifizierung durchzuführen.

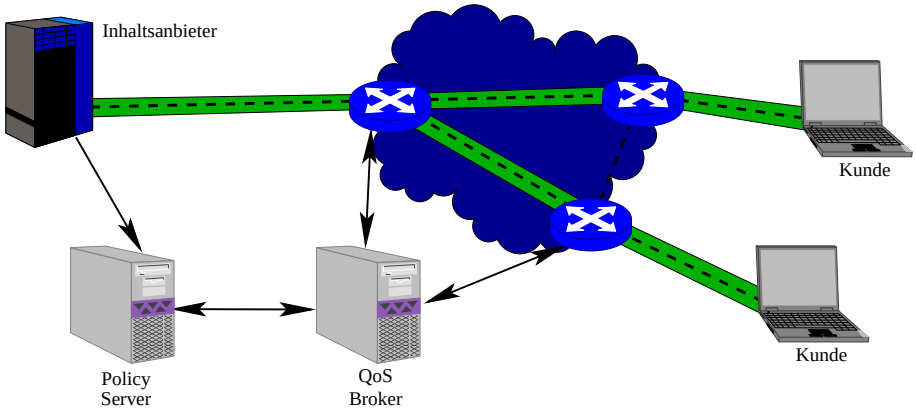

Abbildung 5. Alle Komponenten, die an der Zugangskontrolle beteiligt sind. Der Netzwerkbetreiber ist in die zwei logischen Einheiten Policy Server und QoS Broker aufgeteilt.

Dabei generiert der Inhaltsanbieter auf Verlangen des Netzwerkbetreibers eine An-

frage, die an alle Empfänger verschickt wird. Nur autorisierte Kunden können eine gültige Antwort erzeugen und senden diese an den Inhaltsanbieter zurück. Der Netzwerkbetreiber kann in ausgewählten Knoten kontrollieren, ob die Antworten korrekt sind und entsprechend die Ressourcen reservieren. Diese Arbeit wird auf zwei eigenständige Komponenten aufgeteilt. Dabei führt der Policy Server die Zugangskontrolle aus, während der QoS Broker für die Umsetzung der Reservierung verantwortlich ist. Die Aufteilung hat den Vorteil, dass die Last auf mehrere Komponenten verteilt wird. Als Basis für das Protokoll wird das Resource Reservation Protocol (RSVP) [BZB+97] verwendet, dies vor allem weil es alle Ansprüche an das zu verwendende Protokoll erfüllt. Erstens arbeitet RSVP mit Multicast zusammen und da die Übertragung IP-Multicast verwendet, ist dies zwingend nötig. Ausserdem verwendet RSVP ein Request-Response-Verfahren, das auch für die Datenübertragung bei einer Autorisierungsanfrage gebraucht wird. Dabei wird sichergestellt, dass die Daten in beiden Richtungen auf dem gleichen Weg transportiert werden. Drittens verwendet RSVP Soft-States, so dass die Anfrage regelmässig wiederholt werden muss, damit eine Zugangsberechtigung aufrecht erhalten werden kann. Der Abstand zwischen zwei Anfragen beträgt dabei 30 Sekunden, wobei zwei Bestätigungen ausbleiben dürfen, bevor die Verbindung als beendet betrachtet wird. Die vierte wichtige Eigenschaft von RSVP ist, dass es modular aufgebaut ist und im Design schon vorgesehen war, das Protokoll mit neuen Objekten zu erweitern.

Die andere Frage ist, welche Daten verwendet werden sollen um die Autorisierung durchzuführen. Dabei gibt es zwei unterschiedliche Möglichkeiten. Bei der ersten Möglichkeit werden die gesendeten Videodaten verwendet. Bei der zweiten Möglichkeit werden für die Autorisierung eigene Daten gebraucht. Wie die Autorisierung in beiden Fällen gemacht werden kann, wird nachfolgend beschrieben.

Verwendung der erhaltenen Videobilder: Der Inhaltsanbieter wählt ein Bild der Übertragung aus, das für die Autorisierung verwendet werden soll. Zusätzlich gibt er vor, welcher Teil vom Bild verwendet werden soll, indem er z.B. die Koordinaten eines Ausschnittes angibt. Bevor nun der Kunde die Anfrage beantworten kann, muss er zuerst das Video zusammensetzen. Danach wird der Teil des Videobildes verwendet und daraus werden die Authentifizierungsdaten generiert. Dies kann z.B. passieren, indem ein Hash-Code davon erzeugt wird. Dieser Hash-Code wird wieder zurück zum Inhaltsanbieter gesendet. Die Antwort wird auf dem Rückweg vom Netzwerkbetreiber verifiziert, der ebenfalls über die gültigen Hash-Codes verfügt.

Durch die Auswahl des Ausschnittes im Bild lassen sich zwei sinnvolle Abstufungen für die Authentifizierung erzeugen. Die am stärksten eingeschränkte Authentifizierung erhält man, wenn ein Bereich ausgewählt wird, in dem sich das Wasserzeichen befindet. In diesem Fall wird für jedes Bildsegment ein individueller Hash-Code erzeugt, mit dem man die Authentifizierung nur für die Datenströme machen kann, die man auch entschlüsseln kann. In diesem Fall ist es auch sinnvoll für jeden Datenstrom einen eigenen Ausschnitt zu wählen, sonst entsteht das Problem, das der Hash-Code eines Kunden individuell ist, aber für alle seine Datenströme gültig ist. Bei einem grossen Dienst bekäme man relativ schnell ein Skalierungsproblem. Diese Einschränkung auf einen Ausschnitt der Wasserzeichen enthält, muss nur gebraucht werden, wenn davon auszugehen ist, dass Kunden absichtlich noch andere, nicht für sie bestimmte, Bildsegmente empfange. Sonst genügt es, wenn man einen Bereich wählt, der kein Wasserzeichen enthält. Dieser Hash-Code kann dann für alle Segmente gleichzeitig benutzt werden. Ausserdem sollten die Koordinaten des gewählten Ausschnitts über längere Zeit konstant bleiben. Auf diese Art kann der Aufwand beim Empfänger klein gehalten werden, da weniger Daten zwischengespeichert

werden müssen. Die Zwischenspeicherung von Bilder, bzw. Ausschnitten, ist nötig, da die Anfrage nach dem korrespondierenden Bild eintreffen kann.
Für den Netzwerkbetreiber ist es auch entscheidend, welche Daten für die Authentifizierung verwendet werden. Je globaler der Hash-Code ist, um so kleiner ist der Aufwand. Der Aufwand für den Netzwerkbetreiber ist klein, wenn es genau einen Hash-Code für jeden Multicast-Strom gibt. Das grössere Problem des Netzwerkbetreibers ist es, die gültigen Hash-Codes zu generieren. Dies kann z.B. gemacht werden, indem der Policy Server auch die Übertragung mitverfolgt, jedoch würde auch hier das Problem der Skalierbarkeit auftreten. Der Netzwerkbetreiber könnte diese Aufgabe auch ausserhalb des Policy Servers betreiben, aber in diesem Fall wäre es einfacher die Hash-Codes direkt vom Inhaltsanbieter zu bekommen.
Im Fall der Verwendung der Videodaten gibt es noch einen Spezialfall. Wird beim Transport der Daten ein Verfahren verwendet, das alle transportierten Datenpakete eigenständig verschlüsselt, so könnte man ein solches Paket auswählen, um davon den Hash-Code zu erzeugen. In diesem Fall wäre auch der Aufwand für den Netzwerkbetreiber sehr gering, da nicht mehr die ganze Übertragnug bearbeitet werden muss, sondern nur noch die entsprechenden Pakete.

Verwendung eines Handshake-Verfahrens: Die zweite Möglichkeit besteht in der Verwendung eines reinen Request-Response-Szenarios. In diesem Fall wird die Authentifizierung unabhängig von den gesendeten Videodaten durchgeführt. Alle entscheidenden Informationen werden mit RSVP-Nachrichten verschickt. Das Problem ist hier, dass für die Authentifizierung Schlüssel verwendet werden müssen. Bei anonymen Kunden ist der Austausch von Schlüssel schwierig, da der Policy Server nicht weiss, wohin er diese schicken soll. Eine möglicher Lösung ist, die neuen Schlüssel mit dem gleichen Schlüssel zu verschlüsseln, der auch für die Verschlüsselung der Videodaten gebraucht wird. Ausserdem könnte diese Arbeit an den Inhaltsanbieter delegiert werden. Wenn man aber bereits die Verschlüsselung durch den Inhaltsanbieter durchführen lässt, so kann man den Austausch von Schlüsseln umgehen, indem man vom Inhaltsanbieter die Daten für das Handshake-Verfahren verschlüsseln lässt.

Beide Verfahren haben ihre Vorteile. Die zweite Möglichkeit ist von den transportierten Daten unabhängig, aber für anonyme Kunden wird der Austausch von sicherheitsrelevanten Informationen schwierig. Bei der ersten Möglichkeit besteht das Problem, dass die korrekten Hash-Codes irgendwie erzeugt werden müssen. Diese können aber auch vom Inhaltsanbieter erhalten werden, da dieser ja auch ein Interesse daran hat, dass möglichst wenig Ressourcen verwendet werden und somit der Profit gesteigert werden kann. Um die Nähe zum Videoverteildienst zu behalten, verwenden wir die erste Möglichkeit.

5 Prototypische Implementierung

Abbildung 6 zeigt das Szenario, das für die prototypische Implementierung verwendet wurde. Um die Beschreibung des Szenarios zu vereinfachen wird nur ein Kunde verwendet. In der oberen Hälfte befinden sich die Teilnehmer, die an der Übertragung beteiligt sind. Die Videodaten werden vom Sender direkt an den Kunden gesendet. Die dazwischen liegenden Knoten leiten die Daten normal weiter, ohne die Daten speziell zu bearbeiten. In der unteren Hälfte befinden sich die Komponenten, die für die Zugangskontrolle gebraucht werden. Beim Sender der Videodaten wird eine einfache RSVP-Anwendung, die Injektor genannt wird, verwendet, um die Authentifizierungsanfragen als RSVP-PATH-Nachricht zu generieren. Beim Kunden wird die selbe Anwendung verwendet, um die Antworten als RSVP-RESV-Nachricht zu generieren. Auf dem Pfad zwischen dem Sender und dem Kunden

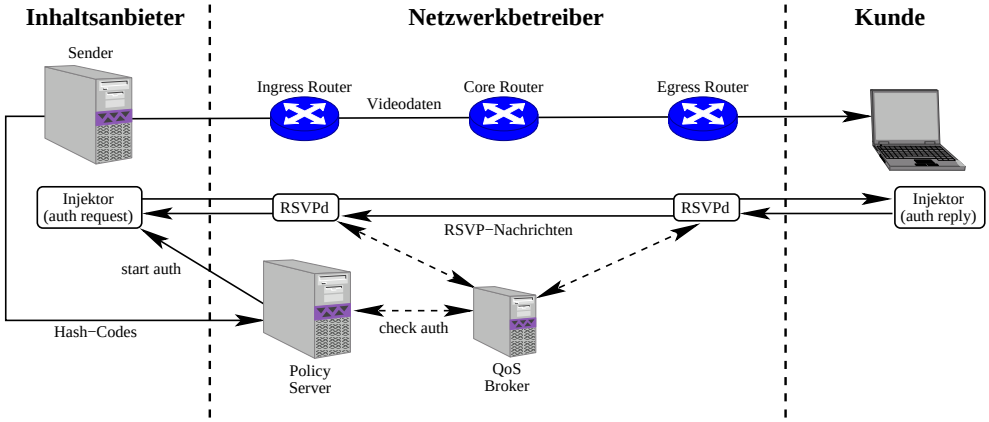

Abbildung 6. Das Szenario, wie es für die prototypische Implementierung vorgesehen ist.

werden die RSVP-Nachrichten durch einen erweiterten RSVP-Daemon bearbeitet. Um den Aufwand möglichst gering zu halten, laufen diese Daemone nur auf den Routern, die sich am Rande des Netzwerkes befinden, also auf dem Ingress- und dem Egress-Router. Dies ist ausreichend um alle für die Vergabe der Ressourcen benötigten Informationen zu erhalten.

Für die Authentifizierung werden die Videodaten, bzw. ein Ausschnitt, verwendet, um den entsprechenden Hash-Code zu erzeugen. Der korrekten Hash-Code erhält der Policy Server direkt vom Inhaltsanbieter.

Um die Prototypen genauer zu beschreiben, kann die Implementierung in verschiedene Bereiche unterteilt werden. Zuerst werden die Anpassungen des RSVP-Protokolls beschrieben, dann die Anwendung zur Erzeugung der RSVP-Nachrichten und zuletzt die Verbindung zwischen dem RSVP-Protokoll und dem Policy Server.

5.1 Erweiterung des RSVP-Protokolls

Zuerst betrachten wir die Anpassungen, die innerhalb des RSVP-Protokolls benötigt werden. Der Vorteil von RSVP ist, dass Policy-Objekte schon bei der ersten Definition [BZB+97] berücksichtigt wurden. Die Definition der Policy-Objekte wurde später in [Her00] verfeinert. Diese Definition ist flexibel gehalten und erlaubt das Einfügen von neuen Objekten. Dabei sind die gültigen Policy-Objekte in drei Bereiche unterteilt. Der erste und der zweite Bereich sind standardisiert. Der letzte Bereich hingegen ist offen für neue und proprietäre Erweiterungen.

Für die prototypische Implementierung wurden neue Policy-Objekte eingeführt, die für den Transport der Informationen der betreffenden Bildern, des betreffenden Ausschnittes, sowie der Paketnummer verwendet werden. Ausserdem wurde ein Objekt generiert, um den generierten Hash-Code zu transportieren.

5.2 Erzeugen der RSVP-Nachrichten

Die Injektor-Anwendung wird für die Initialisierung der RSVP-Nachrichten benötigt. Die Anwendung hat zwei verschiedene Modi. Der erste Modus wird *ask* genannt. In diesem Modus generiert die Anwendung die Anfrage für die Authentifizierung (PATH-Nachricht). Dabei erhält sie vom Policy Server die genaue Beschreibung der verlangten Daten. In diesem Szenario umfasst dies nur die Bildnummer und die Koordinaten des Ausschnittes. Der zweite Modus wird *reply* genannt. In diesem Modus erzeugt der Injektor automatisch die richtige Antwort (RESV-Nachricht) auf die Anfrage in der erhaltenen PATH-Nachricht.

Um dem Problem vorzubeugen, das entsteht, wenn die Anfrage nach dem entsprechenden Videobild ankommt, verwaltet der Injektor eine Liste mit den letzten 1024 Videobildern, bzw. den entsprechenden Ausschnitten, was etwa 40 Sekunden des Videos entspricht. Dies reicht aus, da der normale Abstand zwischen zwei Anfragen normalerweise 30 Sekunden beträgt. Normalerweise ist es ausreichend, wenn nur der aktuelle verwendete Ausschnitt gespeichert wird und nicht das ganze Bild. Im Fall, dass der Ausschnitt verschoben wurde und die Anfrage nach dem entsprechenden Videobild eintrifft, erzeugt man keine Antwort. Dies geht problemlos, da bis zu zwei Antworten verloren gehen dürfen, bevor die Reservierung beendet wird. Dies sollte aber selten vorkommen.

5.3 Verbindung zwischen dem RSVP-Protokoll und dem Policy Server

Die dritte Instanz beim Austausch der RSVP-Nachrichten ist der RSVP-Daemon. Der Daemon läuft auf dem Ingress- und dem Egress-Routern des Netzwerkbetreibers. Er hat die Aufgabe die RSVP-Nachrichten korrekt zu bearbeiten und leitet ausserdem ankommende RESV-Nachrichten (Antworten der Kunden) an den Policy Server weiter. Mit der Bearbeitung der RSVP-Nachrichten wird unter anderem ermöglicht, dass die RSVP-Nachrichten auf dem gleichen Pfad hin und her transportiert werden. So weiss jeder RSVP-Daemon, wer sein Vorgänger und wer sein Nachfolger im Pfad ist.

Wenn der RSVP-Daemon die Anfrage an den QoS Broker und den Policy Server weiterleitet, ergänzt er diese mit den Informationen über den Ingress- und Egress-Router des Multicast-Baumes. Mit diesen Informationen ist es dem QoS Broker möglich den entsprechenden Multicast-Ast zu berechnen. Der QoS Broker handelt aber erst, nachdem der Policy Server die Korrektheit der Daten bestätigt hat. Ist die Anfrage gültig, so kann der QoS Broker die Reservierung einrichten bzw. verlängern. Ist die Anfrage mehrmals hintereinander ungültig, kann der QoS Broker die Reservierung aufheben. Für den QoS Broker ist es ausreichend, wenn er die Antwort nur einmal, also entweder vom Ingress- oder vom Egress-Router erhält, da ja beide über die gleichen Informationen verfügen.

Der Policy Server kontrolliert nur, ob die erhaltene Antwort gültig ist oder nicht. Das Resultat gibt nur an, ob der Benutzer in der Lage ist sich zu authentifizieren, sagt aber nichts über den Benutzer selbst. Wenn der Policy Server die Hash-Codes für die Authentifizierung nicht direkt vom Inhaltsanbieter bekommt, so muss auch er eine Liste mit den entsprechenden Ausschnitten verwalten, die von einem Empfänger erzeugt werden, der vom Netzwerkbetreiber selber betrieben wird. Im anderen Fall verwaltet er nur eine Liste mit den gültigen Hash-Codes.

5.4 Resultate

Die prototypische Implementierung dient dazu, zu zeigen, dass das Konzept funktioniert und um einige einfache Leistungsmessungen durchzuführen.

Um den Ressourcenverbrauch der Anwendungen wie dem Injektor zu verringern, wurden die Anwendungen komplett neu implementiert. Für die komplexeren Anwendungen wie den Videosender, den Videoempfänger oder den RSVP-Daemon wurden bestehende Implementierungen verwendet. Der RSVP-Daemon basiert auf der Implementation von ISI [ISI] und einem RSVP/DiffServ-Gateway [BBBG00]. Für den Videoteil wurden ein selbst entwickelter Sender und Empfänger verwendet, die so angepasst wurden, dass die benötigten Ausschnitte der Videobilder an den Injektor weitergeleitet wurden.

Der Prototyp wurde in einem Netzwerk aus Linux-PCs mit 533MHz AMD K6-2 Prozessoren getestet. Der Einfachheit halber liefen der Injektor und der Sender auf dem selben Rechner wie der Ingress-Router. Da der Injektor auf dem Ingress-Router

lief, wurde dort kein zusätzlicher RSVP-Daemon verwendet. Die Systemauslastung für den Videosender und Videoempfänger betrug ca. 50 %. Also beeinflussen diese Anwendungen die Arbeit der anderen Applikationen nicht. Da die Systemauslastung relativ gering war, wurde zur Erzeugung der Hash-Codes ebenfalls ein Videoempfänger auf dem Policy Server betrieben. Der Nebeneffekt war, dass der Policy Server ebenfalls eine Liste mit den letzten 1024 Ausschnitten verwaltet hat.

Zur Abschätzung der Leistungsfähigkeit wurden Zeitmessungen durchgeführt. Dabei wurde die Zeiten gemessen, die für eine einzelne Aktion benötigt wurden. Im Falle des Policy Servers ist das die Zeit, die zwischen der Ankunft der Anfrage und der entsprechenden Antwort vergeht. Für den Injektor hängt dies vom Arbeitsmodus ab. Im Reply-Modus (auf der Empfängerseite) wird die Zeit gemessen, die benötigt wird, um die korrekte Antwort zu erzeugen. Im Ask-Modus (auf der Senderseite) wird die gesamte Zeit gemessen, die zwischen dem Senden der Authentifizierungsanfrage und dem Empfang der entsprechenden Antwort vergeht.

Da der QoS Broker entweder die Anfrage des Ingress- oder des Egress-Routers benötigt wurde beim Test nur die Anfragen des Ingress-Router berücksichtigt. Dabei wurden folgende Ergebnisse erzielt:

- Das Konzept funktioniert und es ist möglich auf einfach Weise die Informationen, die für die Zuweisung der Ressourcen gebraucht werden, zu ermitteln.
- Die Injektor-Anwendung hat einen kleine Ressourcenbedarf und somit behindert sie weder den Videosender, noch den Videoempfänger.
- Die Verzögerung zwischen dem Senden der Anfrage und dem Erhalt der Antwort beträgt $1.5ms$. Für den Transport der Daten werden davon $0.3ms$ (round trip time, RTT gemessen mit ping) benötigt. Der Injektor auf der Kundenseite benötigt $0.6ms$ um die Entsprechende Antwort zu generieren.
- Der Policy Server benötigt $0,42ms$, um zu die Authentifizierungsdaten zu kontrollieren. Dies benötigt kaum Zeit, da nur ein binärer Vergleich zwischen den eigenen Daten und der Anfrage gemacht wird. Extrapoliert man die Werte, erhält man eine Rate von ca. 2000 Anfragen pro Sekunde. Ausserdem wurde 60 % dieser Zeit für die Aktualisierung der Datenbank mit den Authentifizierungsinformationen benötigt.
- Es ist zwingend notwendig, dass die Sammlung der Hash-Codes ausserhalb des Policy Servers erfolgt, da die Leistung bei mehr als einem Datenstrom zusammenbricht. Damit die Datenbank klein gehalten werden kann, ist die Mitarbeit des Inhaltsanbieters zwingend nötig, so dass nur die benötigte Hash-Codes gespeichert werden müssen.

6 Zusammenfassung und Ausblick

In diesem Beitrag wurde ein funktionierendes Konzept zur Zugangskontrolle für einen Videodienst, der DiffServ und IP-Multicast verwendet, vorgestellt. Das Konzept ist flexibel, so dass es auch für andere Szenarios eingesetzt werden kann.

So könnte man das ganze Videoverteilszenario im e-learning Bereich einsetzen, um zu kontrollieren ob an Pflichtveranstaltungen teilgenommen wurde. So kann die Funktionalität der Tickets dazu benutzen werden, um festzustellen, wie lange die Veranstaltung verfolgt wurde. Die Zugangskontrolle könnte dazu benutzt werden, dem Studenten Zwischenfragen zu stellen und so zu Überprüfen, ob er die Veranstaltung auch wirklich mitverfolgt.

Auch das Konzept der Zugangskontrolle alleine kann erweitert werden, um andere Informationen, die vom QoS Broker und/oder vom Policy Server benötigt werden, auszutauschen. Authentifizierungsinformationen sind nur ein kleiner Teil, der mit Policy-Objekten transportiert werden können.

7 Danksagung

Dieser Beitrag wurde im Rahmen des Projekt Quality of Service Support for the Internet Based on Intelligent Network Elements - QuINE (Project 2100-055789.98/1) mit finanzieller Unterstützung des Schweizerischen Nationalfonds zur Förderung der Wissenschaftlichen Forschung (SNF) erstellt.

Literatur

[BB01] Levente Buttyan and Naouel Ben Salem. A Payment Scheme for Broadcast Multimedia Streams. 6th IEEE Symposium on Computers and Communications, Hammamet, Tunisia, July 2001.

[BBBG00] Roland Balmer, Florian Baumgartner, Torsten Braun, and Manuel Günter. A Concept for RSVP over DiffServ. The 9th International Conference on Computer Communication and Network, Las Vegas, USA, October 2000.

[BGB01] Roland Balmer, Manuel Günter, and Torsten Braun. Video Streaming in a DiffServ/IP Multicast Network. Workshop of Advanced Internet Charging and QoS Technology at Informatik 2001, Vienna, Austria, September 2001.

[BW02] Roland Bless and Klaus Wehrle. IP Multicast in Differentiated Service Networks, November 2002. draft-bless-diffserv-multicast-05.txt.

[BZB+97] Bob. Braden, Lixia Zhang, Steve Berson, Shai Herzog, and Sugih Jamin. Resource Reservation Protocol, September 1997. Internet RFC 2205.

[Her00] Shai Herzog. RSVP Extension for Policy Control, January 2000. Internet RFC 2750.

[IETa] IETF. Multicast and Anycast Group Membership magma. `http://www.ietf.org/html.charters/magma-charter.html`.

[IETb] IETF. The Secure Multicast Research Group (SMuG). `http://www.securemulticast.org/smug-index.htm`.

[ISI] ISI. ISI RSVP implementation release 4.2a4-1. `ftp://ftp.isi.edu/rsvp/release/rsvpd.rel4.2a4-1.tar.gz`.

[IYT01] Norihiro Ishikawa, Nagatsugu Yamanouchi, and Osamu Takahashi. An Architecture for User Authentication of IP Multicast and its Implementation. IPSJ Journal Vol. 40, No. 10, May 2001.

[KT00] Dimitri Konstantas and Dimitris Thanos. Commercial Dissemination of Video over Open Networks: Issues and Approaches. Internet Objects, Centre Universitaire d'Informatique, University of Geneva, September 2000. `http://cuiwww.unige.ch/OSG/publications/OO-articles/TechnicalReports/OO%/videoCommerc.pdf`.

[MON] MONET Research Group, University of Illinois. Engineering and Evaluation of A Watermark-Aware Secure Multicast System. `http://cairo.cs.uiuc.edu/security/`.

[SB03] Günther Stattenberger and Torsten Braun. Performance of a Bandwidth Broker for DiffServ Networks, March 2003. Kommunikation in verteilten Systemen (KiVS03), Leipzig, Germany.

[Sch01] Matthias Scheidegger. StreamCom QoS Reservation Architecture: Design of Network Elements, November 2001. Project Deliverable, `http://www.iam.unibe.ch/~rvs/publications/scom-design-ne.pdf`.

[TK00] Dimitris Thanos and Dimitri Konstantas. COiN-Video: A Model for the Dissemination of Copyrighted Video Streams Over Open Networks. Internet Objects, Centre Universitaire d'Informatique, University of Geneva, September 2000. `http://cuiwww.unige.ch/OSG/publications/OO-articles/TechnicalReports/OO%/COiN_Video.pdf`.