PIK, Vol. 10, pp. 121–129, June 2010 · Copyright © by Walter de Gruyter · Berlin · New York. DOI 10.1515/piko.2010.019

TCP-Verkehrsklassifizierung mit Markov-Modellen

Gerhard Münz

Lehrstuhl für Netzarchitekturen und Netzdienste
Institut für Informatik
Technische Universität München
85748 Garching bei München
Germany
muenz@net.in.tum.de

Gerhard Münz ist wissenschaftlicher Mitarbeiter am Lehrstuhl „Netzarchitekturen und Netzdienste“ der Technischen Universität München und war zuvor ebenfalls als wissenschaftlicher Mitarbeiter am Lehrstuhl „Rechnernetze und Internet“ der Universität Tübingen tätig. Seine Forschungsarbeit beschäftigt sich mit passiven Verkehrsmesstechniken sowie der Analyse der gesammelten Verkehrsdaten mit dem Ziel der Anomalieerkennung und der Verkehrsklassifizierung. Darüber hinaus ist Gerhard Münz in der Internet-Standardisierung (IETF) aktiv. Sein Studium der Elektrotechnik absolvierte er im Jahr 2003 an der Universität Stuttgart und an der Ecole Nationale Supérieure des Télécommunications in Paris.

Lothar Braun

Lehrstuhl für Netzarchitekturen und Netzdienste
Institut für Informatik
Technische Universität München
85748 Garching bei München
Germany
braun@net.in.tum.de

Lothar Braun ist seit 2008 wissenschaftlicher Mitarbeiter am Lehrstuhl „Netzarchitekturen und Netzdienste“ der Technischen Universität München. Er studierte Informatik an der Universität Tübingen und schloss sein Studium im Jahr 2008 ab. Im Rahmen seiner Forschungstätigkeit beschäftigt er sich mit passiven Verkehrsmessungen mit dem Ziel der Verkehrsklassifikation und der Erkennung von Malware und Botnetzen.

Hui Dai

Lehrstuhl für Netzarchitekturen und Netzdienste
Institut für Informatik
Technische Universität München
85748 Garching bei München
Germany
dai@in.tum.de

Hui Dai hat 2009 sein Informatik-Studium an der Technischen Universität München abgeschlossen. In seiner Diplomarbeit sowie als Hilfswissenschaftler am Lehrstuhl für „Netzarchitekturen und Netzdienste“ beschäftigte er sich mit statistischen Verfahren zur Verkehrsklassifizierung. Im Januar 2010 nimmt er eine Stelle in der Entwicklungsabteilung eines mittelständischen Unternehmens im Bereich Avionik an.

Georg Carle

Lehrstuhl für Netzarchitekturen und Netzdienste
Institut für Informatik
Technische Universität München
85748 Garching bei München
Germany
carle@net.in.tum.de

Prof. Dr.-Ing. Georg Carle ist Inhaber des Lehrstuhls „Netzarchitekturen und Netzdienste“ der Technischen Universität München. Er studierte Elektrotechnik an der Universität Stuttgart und der Ecole Nationale Supérieure des Télécommunications in Paris und erwarb an der Brunel University in London einen Master of Science in Digital Systems. Am Institut für Telematik der Universität Karlsruhe promovierte er 1996 als Stipendiat des Graduiertenkollegs „Beherrschbarkeit komplexer Systeme“. 1997 war er mit einem Stipendium der Europäischen Gemeinschaft am Institut Eurécom in Sophia Antipolis tätig. Beim Fraunhofer Institut für Offene Kommunikationssysteme (FOKUS) in Berlin leitete er das Kompetenzzentrum ‚Global Networking‘, von wo aus er im Dezember 2002 an die Universität Tübingen auf den neu geschaffenen Lehrstuhl „Rechnernetze und Internet“ wechselte.

Zusammenfassung

Die Klassifizierung von Verkehrsströmen anhand von Portnummern ermöglicht heutzutage keine zuverlässigen Aussagen bezüglich der ursächlichen Anwendungen. Auch die Verwendung von anwendungsspezifischen Signaturen stößt an ihre Grenzen, sobald der Verkehr verschlüsselt ist. Ein aktuelles Forschungsgebiet beschäftigt sich daher mit Verfahren, die es ermöglichen, Verkehrsströme mit Hilfe von alternativen Flow- und Paketeigenschaften einer Anwendung zuordnen zu können.

In diesem Kontext stellen wir ein Klassifizierungsverfahren vor, welches die Abfolge von gesetzten TCP-Flags und Paketgrößen innerhalb einer TCP-Verbindung mit Hilfe von Markov-Modellen modelliert. Stehen Markov-Modelle für verschiedene Anwendungen zur Verfügung, kann eine neue TCP-Verbindung durch Bestimmung der maximalen A-posteriori-Wahrscheinlichkeit einer dieser Anwendungen zugeordnet werden. Ein Vergleich des Verfahrens mit dem viel zitierten Ansatz von Bernaille [1] zeigt, dass sich mit unse-

rem Ansatz ähnlich gute und bei geeigneter Parametrisierung sogar bessere Klassifikationsergebnisse erzielen lassen.

1 Einleitung

Eine möglichst genaue Klassifizierung des Internet-Verkehrs nach Anwendungen bzw. Anwendungsklassen ist für Netzbetreiber aus mehrfacher Hinsicht von Interesse. Sie ermöglicht beispielsweise, den Verkehr in verschiedene Dienstgüteklassen einzuteilen, anomale oder bösartige Verkehrsströme zu erkennen oder Informationen über die Nutzung des Netzes zu erheben, woraus sich Schlüsse für die zukünftige Netzplanung ziehen lassen. Früher war die Klassifizierung des Internet-Verkehrs relativ einfach, da sich der Verkehr im Wesentlichen aus Anwendungen wie Web, FTP, SMTP und Telnet zusammensetzte, deren Protokolle bei der Internet Assigned Numbers Authority (IANA) mit unterschiedlichen, so genannten „well-known“ oder „registered“ Portnummern registriert sind.

Heutzutage nimmt die Klassifikationsgenauigkeit, die mit diesem Verfahren erreicht werden kann, immer weiter ab, weil viele neue Protokolle keinen bei der IANA registrierten Port verwenden. Außerdem gibt es neue Anwendungen, die Portnummern dynamisch festlegen, wie z. B. die Internet-Telefonie-Software *Skype*. Viele Peer-to-Peer-Anwendungen (P2P) verwenden Ports, die für andere Anwendungen registriert wurden, um Firewall-Regeln zu umgehen. In einer Arbeit von 2005 konnte anhand der Portnummer ein Anteil von 69% des Verkehrsvolumens bezogen auf die Byte-Anzahl bzw. 71% bezogen auf die Paketanzahl korrekt einer Anwendung zugeordnet werden [2]. In einer weiteren Studie aus dem Jahr 2008 funktionierte dies je nach Netzwerk bei 71% bis 95% der Flows [3]. Für klassische Anwendungen wie Web, Mail, DNS, Chat und SSH liegt die Genauigkeit deutlich über 90%, für FTP, P2P, Streaming-Anwendungen und Online-Spiele teilweise weit unter diesem Wert.

Ein anderes klassisches Klassifizierungsverfahren untersucht die Paketinhalte des Internet-Verkehrs. Eine große technische Herausforderung ist hierbei, mit den immer höheren Paketraten im Internet umgehen zu können. Daneben nimmt mit der Zahl der Anwendungen auch die Zahl der Signaturen stetig zu, wodurch die Klassifizierung mehr Speicherplatz und einen höheren Rechenaufwand je Paket benötigt. Die Notwendigkeit zusätzlicher Signaturen kann sich auch schon bei einer neuen Version einer Anwendung ergeben, woraus sich ein hoher Anpassungsbedarf ergibt. Bei verschlüsseltem Internet-Verkehr ist zudem eine inhaltsbasierte Klassifizierung gar nicht möglich.

Neben den technischen Problemen müssen bei der Untersuchung von Paketinhalten auch rechtliche Aspekte berücksichtigt werden. Aus datenschutzrechtlichen Gründen ist es häufig nicht zulässig, den Inhalt des Verkehrs zu analysieren und zu kontrollieren.

Angesichts der Einschränkungen der port- und inhaltsbasierten Klassifizierung versuchen seit einigen Jahren verschiedene Wissenschaftler, neue Methoden zu finden, die auf statistischen Verkehrseigenschaften beruhen. Zum großen Teil werden Methoden des maschinellen Lernens (ML) angewandt, wie z. B. Clustering-Algorithmen oder naive Bayes-Klassifikatoren [4]. Einige Ansätze verwenden Hidden-Markov-Modelle [5, 6]. In Abschnitt 2 werden wir die wichtigsten verwandten Arbeiten kurz vorstellen.

Im Anschluss stellen wir in Abschnitt 3 ein neues Verfahren vor, das Markov-Modelle zur Klassifizierung von TCP-Verbindungen einsetzt [7]. Die betrachteten Eigenschaften sind die Längen der Nutzdaten und die TCP-Flags der ersten Pakete einer TCP-Verbindung. Ein Vergleich mit dem viel zitierten Ansatz von Bernaille [1] zeigt, dass unser Verfahren ähnlich gute und teilweise sogar bessere Ergebnisse erzielt. Dabei ist unser Verfahren einfacher zu implementieren, weniger rechenaufwendig und leichter zu parametrisieren. In Abschnitt 4 präsentieren wir unsere Untersuchungsergebnisse, die auf Verkehrsdaten basieren, die in echten Netzen gesammelt wurden. Abschnitt 5 schließt unsere Ausführungen mit einer Diskussion der Ergebnisse und einem Ausblick auf mögliche Verbesserungen des Verfahrens ab.

2 Stand der Forschung

Für die Verkehrsklassifizierung wurden mehrfach ML-Techniken vorgeschlagen, die in Trainingsdaten anwendungsspezifische Muster und Unterscheidungsmerkmale finden sollen. Die betrachteten Eigenschaften beziehen sich dabei entweder auf den gesamten Flow, wie z.B die mittlere Paketlänge, oder auf einzelne Pakete innerhalb eines Flows oder einer Verbindung, wie z. B. die Paketgröße oder der zeitliche Abstand zum vorangegangenen Paket (Zwischenankunftszeit). Nguyen und Armitage liefern einen ausführlichen Vergleich dieser Verfahren [4].

Im Folgenden gehen wir auf Arbeiten ein, die ähnlich wie unser Ansatz Markov-Modelle verwenden oder sich auf die Untersuchung der ersten Pakete einer Verbindung bzw. eines Flows beschränken, ohne dabei die übertragenen Nutzdaten zu betrachten.

Bernaille et al. klassifizieren TCP-Verkehr anhand der Längen der ersten vier bis fünf Datenpakete einer Verbindung [1]. Datenpakete sind dabei diejenigen Pakete, die Nutzdaten enthalten. Zur Klassifizierung werden die unüberwachten Cluster-Algorithmen k-Means und GMM (Gaussian Mixture Models) verwendet, obgleich die Trainingsdaten nach Anwendungen markiert sind (engl. „labeled“) und von daher auch überwachte ML-Methoden verwendet werden könnten. Die gefundenen Cluster werden anschließend derjenigen Anwendung zugeteilt, aus der die Mehrzahl der zugeordneten Verbindungen stammt. Die Autoren sprechen in diesem Zusammenhang von halbüberwachtem (engl. „semi-supervised“) Lernen. Bei der Klassifizierung einer neuen Verbindung wird deren Cluster-Zugehörigkeit bestimmt und die entsprechende Anwendung angenommen. Weiterhin untersuchen Bernaille et al. den Einsatz von *Hidden-Markov-Modellen* (HMM) und so genannten spektralen Clustering-Methoden [1, 8].

Wright et al. [5] sowie Dainotti et al. [6] bestimmen mit Hilfe von Trainingsdaten je ein HMM für jede Anwendung. Ihr Klassifizierer ordnet dann eine neue beobachtete Paketfolge derjenigen Anwendung zu, für die das HMM die größte

A-posteriori-Wahrscheinlichkeit ergibt. Beide Forschergruppen modellieren dabei nicht nur die Paketlängen sondern auch die Zwischenankunftszeiten. Wright et al. betrachten beide Richtungen von TCP-Verbindungen und verwenden Links-Rechts-HMMs mit einer großen Zustandsanzahl und einer diskreten Emissionswahrscheinlichkeitsverteilung. Dainotti et al. verwenden dagegen ergodische HMMs mit vier bis sieben Zuständen und gamma-verteilten Emissionswahrscheinlichkeiten. Betrachtet werden in dieser Arbeit nur Datenpakete (d. h. Pakete mit Nutzdaten), die vom Client zum Server fließen.

Im Gegensatz zu den genannten Arbeiten verwenden wir keine Markov-Modelle mit versteckten Zuständen sondern solche, bei denen die Zustände mit den Beobachtungen übereinstimmen. Motiviert wurden wir dabei von Estevez-Tapiador et al., die solche Markov-Modelle zur Erkennung von Anomalien in TCP-Verbindungen verwendet haben [9]. Während die Zustände der Markov-Modelle bei Estevez-Tapiador nur die TCP-Flags der Pakete reflektieren, untersuchen wir auch die Berücksichtigung der Paketlängen. Weitere Details unseres Verfahrens werden im folgenden Abschnitt vorgestellt.

3 TCP-Verbindungsklassifizierung mit Markov-Modellen

Wie alle Arbeiten, die Verkehrsklassifizierung anhand statistischer Eigenschaften des Netzwerkverkehrs betreiben, gehen auch wir davon aus, dass bestimmte Eigenschaften des Netzverkehrs vom Verhalten der Anwendungen bestimmt werden, die den Verkehr erzeugen. Aus der Beobachtung von Verkehrsströmen können somit Rückschlüsse auf die Anwendungen gezogen werden. Wir modellieren die Eigenschaften von TCP-Verbindungen mit Hilfe von Markov-Modellen, deren Zustände direkt aus den beobachteten Paketen abgeleitet werden. Die betrachteten Paketeigenschaften sind TCP-Flags und Paketlängen.

Das Markov-Modell besteht aus einer Menge von n möglichen Zuständen $S = \{s_1, \ldots, s_n\}$, einem Vektor für die Anfangswahrscheinlichkeiten $\Pi = \{\pi_1, \ldots, \pi_n\}$ und einer $n \times n$-Matrix der Übergangswahrscheinlichkeiten $A = \{a_{s_i,s_j}\}$ zwischen den Zuständen. Übergänge werden durch Paketankünfte ausgelöst. Der neue Zustand kann direkt aus den gesetzten TCP-Flags bzw. der Paketgröße des neu eingetroffenen Paketes abgeleitet werden.

In einem ersten Ansatz betrachten wir wie Estevez-Tapiador [9] nur die TCP-Flags, die in einem Paket gesetzt sind. Je nach dem, welche der Flags URG, ACK, PSH, RST, SYN und FIN gesetzt sind, wird das Paket einem Zustand zugeordnet. Die 64 möglichen TCP-Flag-Kombinationen werden also auf 64 verschiedene Zustände abgebildet, wovon in einer typischen TCP-Verbindung aber nur ein kleiner Teil auftritt. Der übliche Ablauf einer TCP-Verbindung folgt dem Markov-Modell, das in Abbildung 1 dargestellt ist. Der Aufbau einer TCP-Verbindung durch den so genannten *Three-Way-Handshake* ist für alle Anwendungen gleich. Gleiches gilt im Allgemeinen auch für den Verbindungsabbau, wobei hier gewisse Formen des Verbindungsabbruchs charakteristisch für eine Anwendung sein können. Interessant für die Unterscheidung verschiedener Anwendungen sind vor allem die TCP-Flags in den Paketen, die zwischen Verbindungsaufbau und -abbau ausgetauscht werden, also das Auftreten von ACK- und ACK/PUSH-Paketen.

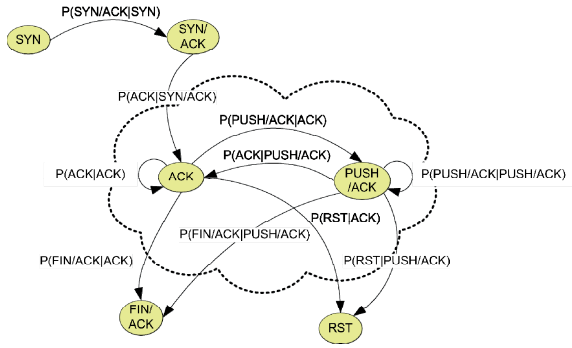

Abbildung 1 Markov-Modell mit TCP-Flags als Zustände.

In einem zweiten Ansatz betrachten wir nur die Paketlängen innerhalb einer TCP-Verbindung. Die Paketlänge ist nach unten durch den TCP/IP-Header und nach oben durch die MTU (*Maximum Transmission Unit*) begrenzt. Es macht nun keinen Sinn, jede mögliche Paketlänge auf einen diskreten Zustand abzubilden, weil die Auftrittswahrscheinlichkeiten sehr gering wären und der Zustandsraum sehr groß. Deshalb unterteilen wir die Menge der möglichen Paketlängen in Intervalle, die jeweils einen Zustand bilden.

Weiterhin ist es möglich, Zustände zu erzeugen, die sich aus einer Kombination aus TCP-Flags und Paketlänge ergeben. Wie sich zeigen wird, kann eine solche Kombination die Genauigkeit der Klassifikation gegenüber der Verwendung einer einzelnen Paketeigenschaft verbessern. Zusätzlich kann auch die Flussrichtung berücksichtigt werden, d. h. Pakete, die vom Client zum Server fließen, werden auf andere Zustände abgebildet als Pakete in entgegengesetzter Richtung. Als Client verstehen wir hierbei stets den Kommunikationspartner, der die TCP-Verbindung initiiert hat. In Abschnitt 4 untersuchen wir die verschiedenen Möglichkeiten, die Paketeigenschaften auf die Zustände des Markov-Modells abzubilden.

Nachdem die Zustände definiert sind, können die Anfangswahrscheinlichkeiten π_i und die Übergangswahrscheinlichkeiten a_{s_i,s_j} direkt aus Trainingsdaten geschätzt werden. Die Anfangswahrscheinlichkeiten ergeben sich aus den relativen Häufigkeiten der modellierten Eigenschaft in den ersten Paketen. Für die Übergangswahrscheinlichkeiten ermittelt man die Häufigkeiten der verschiedenen Zustandsübergänge $F(s_i, s_j)$ mit $s_i, s_j \in S$ in den TCP-Verbindungen einer Anwendung. Die Übergangswahrscheinlichkeit für den Zustandswechsel von s_i nach s_j ergibt sich dann wie folgt:

$$a_{s_i,s_j} = \frac{F(s_i, s_j)}{\sum_{k=1}^{n} F(s_i, s_k)}. \quad (1)$$

Somit erhält man für jede Anwendung ein eigenes Markov-Modell. Abbildung 2 zeigt beispielhaft die unterschiedlichen Markov-Modelle, die sich aus den Trainingsdaten für Webverkehr (HTTP) und Gnutella ergeben, wenn die TCP-Flags

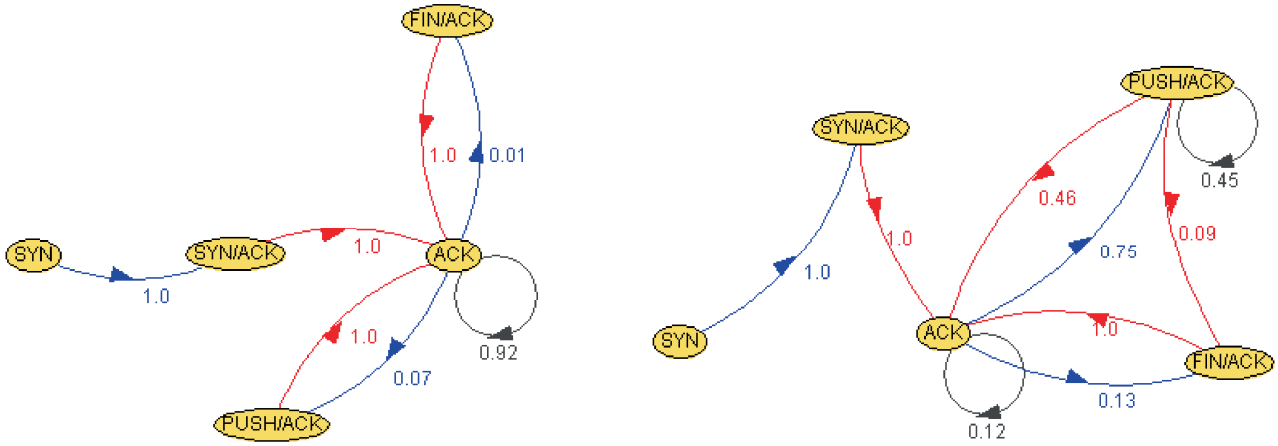

Abbildung 2 Markov-Modelle für HTTP (links) und Gnutella (rechts).

auf die Zustände abgebildet werden. Der SYN-Zustand ist hier der einzig mögliche Anfangszustand, da alle Verbindungen mit einem SYN-Paket beginnen.

Mit den aus Trainingsdaten erzeugten Markov-Modellen kann, wie in Abbildung 3 gezeigt, ein Klassifizierer aufgebaut werden, der neue TCP-Verbindungen derjenigen Anwendung zuordnet, für deren Markov-Modell sich die größte A-posteriori-Wahrscheinlichkeit (engl. „maximum a-posteriori probabilities“) ergibt. Die maximale A-posteriori-Wahrscheinlichkeit für ein Modell M und eine TCP-Verbindung mit der Paketfolge $O = \{o_1, o_2, \ldots, o_N\}$ lässt sich durch Logarithmieren in eine Summe umformen, die sich leicht berechnen lässt:

$$\log\{\Pr(O \mid M)\} = \log \pi_{o_1} + \sum_{i=1}^{N-1} \log a_{o_i, o_{i+1}}. \qquad (2)$$

Der Klassifizierer wählt nun dasjenige Modell aus, für das die logarithmierte A-posteriori-Wahrscheinlichkeit maximal ist.

Falls in der zu klassifizierenden TCP-Verbindung ein Übergang o_i, o_{i+1} vorkommt, der in den Trainingsdaten einer Anwendung nicht auftrat (d. h. $a_{o_i, o_{i+1}} = 0$), ist die A-posteriori-Wahrscheinlichkeit Null, d. h. das Markov-Modell ist nicht passend. Falls in sämtlichen Markov-Modellen ein unbekannter Übergang auftritt, kann die Verbindung keiner Anwendung zugeordnet werden. Sie wird daher als „unbekannt“ klassifiziert.

Abbildung 4 verdeutlicht das Vorgehen bei der Klassifizierung anhand einer HTTP-Verbindung. Hier ist der Verlauf der A-posteriori-Wahrscheinlichkeit für die Markov-Modelle von fünf verschiedenen Anwendungen aufgetragen. Dabei zeigt sich, dass das HTTP-Modell tatsächlich am besten zu der Verbindung passt, weil sich hierfür die größte Wahrscheinlichkeit ergibt. Diese Feststellung kann bereits nach wenigen Paketen getroffen werden. Für die Klassifikation muss daher nicht bis zum Ende der Verbindung gewartet werden, was eine wichtige Voraussetzung für die Online-Verkehrsklassifizierung darstellt. Interessant ist, dass das YouTube-Modell am zweitbesten abschneidet. Die Ähnlichkeit zwischen den Markov-Modellen der beiden Anwendungen lässt sich dadurch erklären, dass bei YouTube Videos über HTTP übertragen werden.

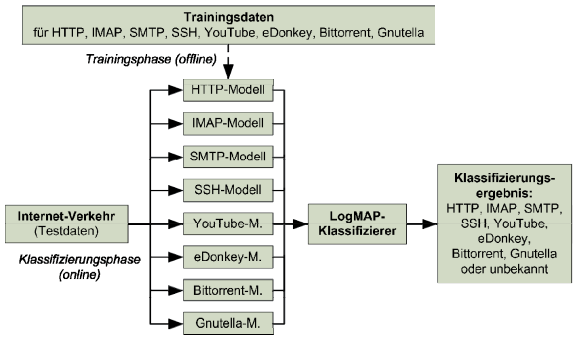

Abbildung 3 Schematische Darstellung der Klassifizierung.

Im folgenden Abschnitt evaluieren wir das vorgestellte Verfahren und untersuchen dabei verschiedene Möglichkeiten, TCP-Flags und Paketgrößen auf die Zustände der Markov-Modelle abzubilden. Zudem vergleichen wir das Verfahren mit dem Klassifizierungsansatz von Bernaille [1].

4 Evaluierung und Vergleich

Die Wirksamkeit des vorgestellten Klassifizierungsansatzes haben wir anhand von Verkehrsdaten evaluiert, die in echten Netzen aufgezeichnet wurden. Der folgende Unterabschnitt beschreibt, wo und wie die Daten aufgezeichnet wurden und nach welchen Gütemaßen wir unseren Klassifizierungsansatz bewerten. Danach fassen wir in den Unterabschnitten 4.2, 4.3 und 4.4 die Untersuchungsergebnisse zusammen, die sich bei der Berücksichtigung von TCP-Flags, Paketgrößen oder beiden Paketeigenschaften in den Zuständen der Markov-Modelle ergeben. In Unterabschnitt 4.5 folgt ein Vergleich mit dem Ansatz von Bernaille [1]. Für eine ausführlichere

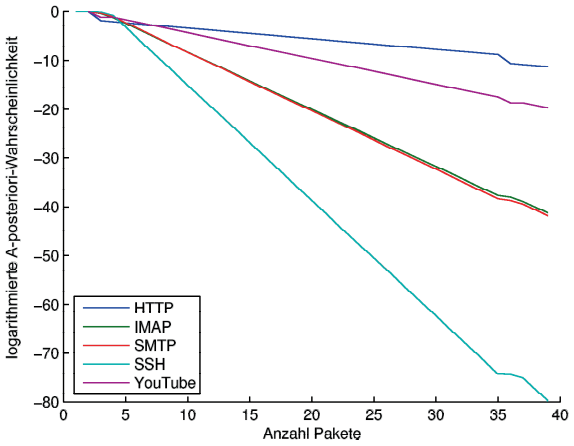

Abbildung 4 A-posteriori Wahrscheinlichkeit einer HTTP-Verbindung bzgl. verschiedener Markov-Modelle.

Darstellung der Ergebnisse verweisen wir auf unseren Beitrag zu MMBnet 2009 [7].

4.1 Verkehrsdaten und Bewertungskriterien

Die für die Evaluierung verwendeten Verkehrsaufnahmen wurden in den Netzwerken unserer Arbeitsgruppe gesammelt. Sie umfassen zum einen Daten klassischer Client-Server-Protokolle wie HTTP, IMAP, SMTP und SSH. Dieser Verkehr wurde an unseren eigenen Servern aufgenommen, sodass sichergestellt ist, dass die Messdaten der richtigen Anwendung zugeordnet wurden. Weitere Aufnahmen umfassen Verkehr zum Video-Portal YouTube, der von uns durch Aufrufen von Webseiten auf www.youtube.com erzeugt wurde. Da die Verkehrsklassifizierung im Speziellen für Dienste notwendig ist, die unbekannte bzw. dynamisch ausgehandelte Ports verwenden, untersuchen wir auch Verkehr aus Peer-to-Peer-Netzen. Durch eine Installation der entsprechenden Anwendungen auf unseren Rechnern waren wir in der Lage, Verkehr aus den Peer-to-Peer-Netzen eDonkey, BitTorrent und Gnutella zu generieren und aufzuzeichnen.

In der Trainingsphase werden von jeder Anwendung 300 Verbindungen verwendet, um jeweils ein Markov-Modell zu erzeugen. Die Testdaten beinhalten zwischen 311 und 761 Verbindungen je Anwendung. Alle Verbindungen enthalten mindestens vier Pakete mit Nutzdaten und mindestens acht Pakete insgesamt. Dadurch wird sichergestellt, dass nur solche Verbindungen untersucht werden, die vollständig aufgebaut wurden und bei denen Daten geflossen sind. Die Beschränkung auf Verbindungen mit mindestens vier Paketen mit Nutzdaten ermöglicht zudem einen direkten Vergleich mit dem Ansatz von Bernaille [1], der eine solche Mindestanzahl erfordert. Es sei an dieser Stelle hervorgehoben, dass unser Ansatz auch mit weniger Datenpaketen arbeiten kann.

In der Trainings- und Klassifizierungsphase betrachten wir jeweils eine vorgegebene Maximalanzahl an Paketen je Verbindung. In unserer Auswertung verwenden wir 10 als kleinste Maximalanzahl, was bedeutet, dass bei Verbindungen mit zehn Paketen oder weniger alle Pakete bei der Modellbildung und der Klassifizierung berücksichtigt werden, bei Verbindungen mit mehr als zehn Paketen nur die ersten zehn. Bei einer höheren Maximalanzahl werden entsprechend mehr Pakete am Anfang einer jeden Verbindung betrachtet.

Unsere Klassifikationsergebnisse bewerten wir anhand von Trefferquote und Genauigkeit und verwenden hierfür die üblichen englischen Begriffe *Recall* und *Precision*. Recall und Precision sind wie folgt definiert:

– Recall bezeichnet den Anteil aller Verbindungen der Anwendung A, die korrekt als Anwendung A klassifiziert werden:

$$\text{Recall} = \frac{\#\,(\text{korrekt als A klassifizierte Verbindungen})}{\#\,(\text{alle Verbindungen von A})}.$$

– Precision bezeichnet den Anteil der korrekt als Anwendung A klassifizierten Verbindungen bezüglich aller als A klassifizierten Verbindungen:

$$\text{Precision} = \frac{\#\,(\text{korrekt als A klassifizierte Verbindungen})}{\#\,(\text{alle als A klassifizierte Verbindungen})}.$$

Ziel der Klassifizierung ist es, sowohl für Recall als auch für Precision einen Wert nahe 100 Prozent zu erreichen.

Für die Berechnung dieser beiden Gütemaße wird zwischen korrekt und falsch klassifizierten Verbindungen unterschieden. Unter die falsch klassifizierten Verbindungen fallen solche, die einer falschen Anwendung zugeordnet werden, und solche, die aufgrund eines unbekannten Zustandübergangs keiner Anwendung zugeordnet werden können (vgl. Abschnitt 3).

4.2 Klassifizierung anhand von TCP-Flags

In diesem Unterabschnitt stellen wir die Ergebnisse für den Fall vor, dass die TCP-Flags und die Flussrichtung der Pakete die Zustände im Markov-Modell definieren. Dadurch ergibt sich ein Markov-Modell mit 128 Zuständen, 64 für jede Richtung. Tabelle 1 zeigt die Ergebnisse für verschiedene Maximalanzahlen von Paketen, die je Verbindung berücksichtigt werden. Die besten Recall- und Precision-Werte sind für jede Anwendung fett gedruckt. Um das Verfahren für die Online-Klassifizierung einsetzen zu können, sollten pro Verbindung möglichst wenige Pakete untersucht werden.

In Tabelle 1 ist zu sehen, dass bereits gute Ergebnisse erzielt werden können, wenn 20 Pakete pro Verbindung untersucht werden. Für 30 Pakete ergeben sich die besten Durchschnittswerte für Recall und Precision, bei mehr Paketen werden die Klassifikationsergebnisse schlechter. Am schlechtesten sind die Ergebnisse, wenn sämtliche Pakete untersucht werden. Diese Beobachtung legt nahe, dass die ersten Pakete einer Verbindung charakteristische Übergänge zwischen TCP-Flag-Kombinationen aufweisen, wodurch sich die Anwendungen recht gut unterscheiden lassen. Betrachtet man komplette Verbindungen, gehen auch spätere Übergänge, die weniger charakteristisch sind und nur in langen Verbindungen mit vielen Paketen auftreten, in die Markov-Modelle ein. Dadurch verwischen sich die Unterschiede zwischen den verschiedenen Anwendungen, was sich durch sehr ähnliche

Werte in den Übergangsmatrizen der zugehörigen Markov-Modelle bemerkbar macht.

Wichtig bei der Interpretation der Zahlenwerte ist, dass die Ergebnisse für die unterschiedlichen Anwendungen nicht isoliert betrachtet werden dürfen, sondern in engem Zusammenhang stehen. So ist beispielsweise ein guter Recall-Wert für eine Anwendung nicht nur darauf zurückzuführen, dass das zugehörige Markov-Modell die Anwendung gut charakterisiert, sondern auch darauf, dass die Markov-Modelle der übrigen Anwendungen weit weniger gut passen.

Dieselben Untersuchungen wurden auch für Markov-Modelle durchgeführt, bei denen die Flussrichtung der Pakete unberücksichtigt blieb, d. h. für Markov-Modelle mit 64 Zuständen für die unterschiedlichen TCP-Flag-Kombinationen. Unabhängig davon, ob alle Pakete einer Verbindung oder nur die Pakete einer Flussrichtung (von Client zu Server oder umgekehrt) bei der Modellbildung und Klassifizierung betrachtet wurden, ergaben sich schlechtere Klassifikationsergebnisse als in Tabelle 1. Die Flussrichtung stellte sich also in Zusammenhang mit den TCP-Flags als wichtiges Paketmerkmal heraus.

4.3 Klassifizierung anhand von Paketlängen

Nun gehen wir auf die Ergebnisse ein, die sich für Markov-Modelle ergeben, deren Zustände sich aus den Paketgrößen und der Flussrichtung ableiten. Wir bilden hierfür die Paketgrößen auf 16 Intervalle ab, die sich an der Nutzdatenlänge orientieren, d. h. der Paketlänge abzüglich des TCP/IP-Headers. Der erste Zustand repräsentiert Pakete ohne Nutzdaten (Nutzdatenlänge ist Null). Die folgenden 14 Zustände bilden Intervalle mit einer Länge von je 100 Byte ab: [1:100], [101:200], . . . , [1301:1400]. Der letzte Zustand steht für Pakete mit Nutzdatenlängen ab 1401 Byte. Diese Intervallaufteilung ist willkürlich, bietet aber einen Kompromiss zwischen einer feingranularen Unterscheidung von Paketgrößen und einer überschaubaren Zustandsanzahl. Zusammen mit der Flussrichtung ergeben sich somit 32 verschiedene Zustände.

Tabelle 2 liefert die Ergebnisse für verschiedene Maximalanzahlen von Paketen, die je Verbindung berücksichtigt werden. Ähnlich wie in Unterabschnitt 4.2 können wir feststellen, dass schon mit wenigen Paketen pro Verbindung ein gutes Klassifikationsergebnis erreicht werden kann. Die durchschnittlichen Werte für Recall und Precision sind zumeist deutlich besser als bei der Verwendung der TCP-Flags. Dies deutet darauf hin, dass die Übergänge zwischen verschiedenen Paketgrößen bzw. Nutzdatenlängen charakteristischer für eine Anwendung sind als die Übergänge zwischen den gesetzten TCP-Flag-Kombinationen. Bereits bei zehn bis 15 Paketen pro Verbindung ist eine hohe Klassifikationsgenauigkeit zu beobachten. Auch hier werden die Klassifikationsergebnisse schlechter, wenn mehr als 30 Pakete je Verbindung betrachtet werden, wobei die Paketanzahl mit dem größten Recall- oder Precision-Wert von Anwendung zu Anwendung variiert.

Wie im Fall der TCP-Flags haben wir auch bei den Paketlängen den Einfluss der Flussrichtung untersucht. Wieder wiesen Untersuchungen, die nur den Verkehr in eine Richtung betrachten oder die Flussrichtung der Pakete ignorieren, deutlich schlechtere Klassifikationsergebnisse auf [7]. Anschaulich kann man sich dies dadurch erklären, dass die Richtung des Datenflusses stark von den Protokollen und Anwendungen oberhalb von TCP geprägt ist. Bei Protokollen wie SMTP und IMAP werden beispielsweise zu Beginn der Verbindung verschiedene Parameter zwischen Client und Server ausgehandelt, was sich durch einen ständigen Wechsel der Flussrichtung bei den Paketen mit Nutzdaten bemerkbar macht.

4.4 Klassifizierung anhand von TCP-Flags und Paketlängen

Im Folgenden untersuchen wir die Kombination von Nutzdatenlängen und TCP-Flags als Basis der Verkehrsklassifizierung mit Markov-Modellen. Betrachtet man die TCP-Flags, die typischerweise in einer Verbindung auftreten (siehe Abbildung 1), stellt man fest, dass für die Unterscheidung verschiedener Anwendungen nur wenige TCP-Flag-Kombinationen von Interesse sind. So treten SYN- und SYN/ACK-Pakete nur zu Beginn einer Verbindung auf, das FIN- und RST-Flag nur am Verbindungsende. Dadurch tragen diese Flags kaum zur Unterscheidbarkeit verschiedener Anwendungen bei. Das URG-Flag hat so gut wie keine Bedeutung, weil es in der Praxis – wenn überhaupt – nur äußerst selten verwendet wird. In unseren Test- und Trainingsdaten war es in keinem einzigen Paket gesetzt. Am aussagekräftigsten ist das PUSH-Flag, welches immer in Kombination mit dem ACK-Flag auftritt. Für die kombinierte Betrachtung von TCP-Flags und Nutzdatenlängen bietet es sich daher an, nur das PUSH-Flag zu berücksichtigen, um die Zustandsanzahl gering zu halten.

Eine weitere Frage betrifft die Anzahl der Intervalle, in die man die Nutzdatenlängen einteilt. Neben der aus dem vorherigen Unterabschnitt bekannten Aufteilung in 16 Intervalle untersuchten wir unterschiedliche Unterteilungen in weniger Intervalle, um so die Zustandsanzahl der Markov-Modelle zu reduzieren. Interessanterweise ergaben sich für weniger Intervalle auch bessere Klassifikationsergebnisse.

Tabelle 3 zeigt die Ergebnisse für die folgenden vier Intervalle: [0:0], [1:299], [300:(M-1)], [M]. M bezeichnet hierbei die maximal mögliche Nutzdatenlänge, die der *Maximum Sequence Size* (MSS) der TCP-Verbindung entspricht. Zusammen mit Flussrichtung und dem PUSH-Flag ergeben sich insgesamt nur 16 Zustände, was einen geringeren Speicherbedarf für die Matrix der Übergangswahrscheinlichkeiten zur Folge hat. Der Vergleich mit den Werten in Tabelle 1 und 2 zeigt, dass die Kombination beider Paketeigenschaften bessere Ergebnisse liefert als die alleinige Betrachtung von Nutzdatenlänge oder TCP-Flags. Wieder reichen wenige Pakete pro Verbindung aus, um gute Klassifikationsergebnisse zu erzielen. Mit Ausnahme von eDonkey-Verkehr liegen die Recall-Werte bei maximal zehn untersuchten Paketen über 90 Prozent, für die meisten Anwendungen sogar über 95 Prozent, was ein sehr gutes Ergebnis ist. Der durchschnittliche Recall-Wert liegt bei 94,3 Prozent.

Tabelle 1 Klassifikationsergebnis bei Berücksichtigung von TCP-Flags und Flussrichtung.

	Recall und Precision in Abhängigkeit von der Anzahl untersuchter Pakete					
Maximale Paketanzahl	10	15	20	30	100	alle
eDonkey	41,2%, **87,8%**	51,9%, 65,6%	70,5%, 73,9%	**80,4%**, 76,2%	70,1%, 69,8%	17,8%, 59,4%
BitTorrent	65,8%, 69,5%	**83,7%**, 91,0%	81,3%, **93,5%**	80,9%, 92,3%	78,5%, 87,4%	7,79%, 31,5%
Gnutella	90,3%, 77,6%	88,4%, 80,2%	**94,2%**, **82,5%**	93,5%, 74,4%	92,6%, 69,1%	83,9%, 38,4%
HTTP	80,9%, 97,6%	**85,2%**, **97,8%**	84,6%, 95,9%	85,2%, 91,0%	80,2%, 80,3%	59,1%, 79,5%
IMAP	75,6%, 63,2%	71,7%, 68,6%	86,0%, 79,6%	**89,2%**, 92,9%	73,2%, **95,6%**	80,0%, 52,4%
SMTP	**96,9%**, 59,1%	86,1%, 66,6%	79,1%, 78,8%	88,9%, 90,4%	94,3%, **92,9%**	89,1%, 66,9%
SSH	62,3%, **96,3%**	84,0%, 87,3%	93,3%, 82,2%	**93,6%**, 89,8%	86,4%, 79,3%	27,3%, 28,3%
YouTube	92,9%, 92,6%	96,7%, **95,8%**	95,1%, 92,2%	95,5%, 95,1%	**99,3%**, 93,3%	97,4%, 91,2%
Durchschnitt	75,7%, 80,5%	81,0%, 81,6%	85,5%, 84,8%	**88,4%, 87,8%**	84,3%, 83,5%	57,8%, 56,0%

Tabelle 2 Klassifikationsergebnis bei Berücksichtigung von Nutzdatenlänge (16 Intervalle).

	Recall und Precision in Abhängigkeit von der Anzahl untersuchter Pakete					
Maximale Paketanzahl	10	15	20	30	100	alle
eDonkey	80,2%, **95,9%**	81,9%, 80,7%	81,1%, 94,8%	**85,5%**, 91,6%	45,0%, 86,5%	9,7%, 40,4%
BitTorrent	**96,5%**, 95,7%	95,2%, 95,2%	95,0%, 93,2%	93,9%, **97,7%**	89,8%, 95,8%	3,68%, 3,83%
Gnutella	82,6%, **92,4%**	70,5%, 85,2%	82,0%, 87,6%	**84,9%**, 88,9%	82,3%, 75,8%	81,0%, 25,6%
HTTP	87,5%, 95,6%	86,2%, 95,7%	**89,0%**, 96,7%	87,9%, **97,3%**	83,7%, 94,6%	32,0%, 65,4%
IMAP	90,9%, 90,4%	**95,9%**, 94,8%	95,8%, 94,3%	90,4%, **95,7%**	75,9%, 79,0%	65,7%, 58,6%
SMTP	**95,1%**, 85,3%	77,6%, 85,7%	86,6%, 91,7%	86,6%, **92,1%**	75,6%, 90,0%	65,2%, 91,7%
SSH	**98,6%, 99,2%**	97,6%, 92,9%	98,4%, 93,4%	97,6%, 94,8%	93,3%, 62,1%	91,5%, 56,0%
YouTube	95,1%, **100%**	95,1%, 97,0%	94,5%, 98,3%	93,8%, 98,6%	**95,8%**, 93,4%	38,2%, 86,2%
Durchschnitt	**90,8%**, 94,3%	87,5%, 90,9%	90,3%, 93,7%	90,1%, **94,6%**	80,2%, 84,6%	48,4%, 53,5%

Tabelle 3 Klassifikationsergebnis bei Berücksichtigung von Nutzdatenlänge (4 Intervalle) und PUSH-Flag.

	Recall und Precision in Abhängigkeit von der Anzahl untersuchter Pakete				
Maximale Paketanzahl	10	15	20	30	100
eDonkey	82,3%, 92,9%	81,5%, 92,2%	84,9%, 92,3%	**92,7%, 93,3%**	73,1%, 91,8%
BitTorrent	**94,5%**, 97,3%	94,3%, 96,2%	89,1%, 97,4%	88,5%, **98,0%**	86,1%, 97,0%
Gnutella	**95,1%**, 93,6%	90,3%, **94,0%**	90,3%, 92,4%	91,3%, 92,5%	89,7%, 83,0%
HTTP	**96,5%**, 99,0%	93,9%, **99,5%**	95,4%, 99,0%	93,6%, 99,0%	90,8%, 99,0%
IMAP	92,4%, 92,7%	**96,3%**, 92,4%	95,9%, 90,9%	95,1%, **96,4%**	91,2%, 89,9%
SMTP	**97,1%**, 88,0%	96,3%, 92,7%	94,3%, **95,2%**	92,6%, 93,9%	86,9%, 86,6%
SSH	97,3%, **100%**	**98,1%**, 96,8%	98,1%, 92,0%	96,8%, 87,5%	95,2%, 75,9%
YouTube	**99,0%**, 98,4%	98,7%, **99,0%**	96,7%, 98,3%	97,1%, 97,1%	98,3%, 92,7%
Durchschnitt	**94,3%**, 95,2%	93,7%, **95,4%**	93,1%, 94,7%	93,5%, 94,7%	89,9%, 89,5%

Tabelle 4 Klassifikationsergebnisse für Bernaille.

	Recall und Precision für jeden Durchlauf					
Durchlauf	1	2	3	4	5	6
Ergebnis	4 Pakete 29 Cluster	3 Pakete 37 Cluster	3 Pakete 29 Cluster	3 Pakete 33 Cluster	3 Pakete 28 Cluster	3 Pakete 28 Cluster
eDonkey	92,7%, 96,4%	91,0%, 96,9%	94,1%, 80,3%	91,2%, 96,0%	96,9%, 92,0%	92,5%, 92,9%
BitTorrent	99,1%, 86,4%	98,2%, 93,2%	82,0%, 100%	97,4%, 94,5%	97,4%, 98,2%	91,5%, 99,0%
Gnutella	79,4%, 66,4%	86,5%, 100%	93,2%, 85,3%	96,7%, 80,3%	98,4%, 64,7%	86,8%, 85,7%
HTTP	75,5%, 90,5%	80,2%, 88,1%	71,0%, 95,7%	80,0%, 91,3%	72,9%, 93,9%	41,1%, 72,4%
IMAP	96,8%, 91,5%	95,8%, 96,6%	92,6%, 85,1%	95,9%, 92,3%	89,9%, 98,8%	86,7%, 99,6%
SMTP	97,6%, 100%	81,3%, 100%	98,5%, 100%	94,3%, 100%	98,8%, 94,4%	98,0%, 94,3%
SSH	96,0%, 100%	99,7%, 78,0%	95,4%, 100%	97,3%, 98,3%	97,0%, 100%	98,6%, 100%
YouTube	78,1%, 75,2%	90,3%, 63,8%	83,9%, 58,6%	84,5%, 71,4%	83,6%, 78,3%	88,7%, 37,7%
Durchschnitt	89,4%, 88,3%	90,4%, 89,6%	88,8%, 88,1%	**92,2%, 90,5%**	91,8%, 90,0%	85,5%, 85,2%

4.5 Vergleich mit Verfahren von Bernaille

Im Folgenden vergleichen wir unseren Ansatz mit dem Verfahren von Bernaille [1], dessen Implementierung als Matlab-Quellcode von einer Projektwebseite heruntergeladen werden kann [10]. Wie in Unterabschnitt 4.1 erwähnt, haben wir die Trainings- und Testdaten so gewählt, dass jeweils mindestens vier Datenpakete pro Verbindung enthalten sind. Damit können wir Bernailles Verfahren auf die gleichen Daten anwenden. Unser Verfahren arbeitet aber grundsätzlich auch mit Verbindungen, die weniger als vier Datenpakete enthalten.

Die vorliegende Implementierung von Bernaille setzt GMM als Cluster-Algorithmus ein, um Verbindungen mit ähnlichen Nutzdatenlängen in den ersten Datenpaketen zu gruppieren. Dabei wird zu Beginn eine zufällige Cluster-Aufteilung initialisiert. Abhängig von dieser Initialisierung bildet der Cluster-Algorithmus unterschiedliche Cluster, wodurch sich dann auch unterschiedliche Klassifikationsergebnisse ergeben. Weiterhin muss der Cluster-Algorithmus mit der Anzahl betrachteter Datenpakete und der Anzahl zu bestimmender Cluster parametrisiert werden. Auch von diesen Parametern hängt das Klassifikationsergebnis ab.

Bernailles Matlab-Code enthält eine Kalibrierungsmethode, die den Cluster-Algorithmus mit verschiedenen Parametereinstellungen auf die Trainingsdaten anwendet und die Ergebnisse anhand des normalisierten mittleren Transinformationsgehalts (engl. *Normalized Mutual Information*) miteinander vergleicht [1]. Die Cluster-Aufteilung mit dem größten Transinformationsgehalt wird anschließend für die Klassifizierung der Testdaten verwendet. Tabelle 4 enthält die Klassifikationsergebnisse für sechs Durchläufe der Kalibrierungsmethode, wobei in jedem Durchlauf unterschiedliche Kombinationen aus drei und vier Datenpaketen sowie 30, 35 und 40 Clustern durchprobiert wurden. Die abweichenden Cluster-Zahlen in den Ergebnissen ergeben sich durch das automatische Entfernen kleiner Cluster, denen weniger als drei Verbindungen aus den Testdaten zugeordnet werden.

Wie man aus der Tabelle entnehmen kann, führen die Durchläufe teilweise zu recht unterschiedlichen Ergebnissen, was mit der zufälligen Initialisierung der Cluster-Aufteilung zusammenhängt. Häufig finden sich einzelne Anwendungen, die deutlich schlechter als die anderen Anwendungen klassifiziert werden. HTTP erreicht zum Beispiel im letzten Durchlauf nur einen Recall-Wert von 41 Prozent. Im Vergleich zum zweiten Durchlauf, wo 80 Prozent für diese Anwendung erreicht werden, ist dies ein sehr großer Unterschied. Ein ähnliches Beispiel liefert Gnutella, das im fünften Versuch mit einem Recall-Wert von über 98 Prozent sehr gut abschneidet, im ersten Versuch aber nur 79 Prozent erreicht.

Der beste durchschnittliche Recall-Wert liegt bei 92,2 Prozent, während wir mit unserem Verfahren über 94 Prozent erreichen konnten (siehe Tabelle 3). Auch bei nicht optimal gewählter maximaler Paketanzahl lagen bei uns die mittleren Recall-Werte meist deutlich über 90 Prozent. Ein wesentlicher Vorteil unseres Verfahrens ist, dass die Bildung der Markov-Modelle nicht von der Wahl geeigneter Startwerte abhängt sondern für gegebene Trainingsdaten deterministische Ergebnisse liefert. Auch sind bei unserem Verfahren keine größeren Ausreißer bei einzelnen Anwendungen zu beobachten.

5 Zusammenfassung und Ausblick

In der vorliegenden Arbeit haben wir ein Verfahren zur Verkehrsklassifizierung vorgestellt, das die charakteristische Abfolge von TCP-Flags und Paketlängen in den TCP-Verbindungen einer Anwendung in Markov-Modellen modelliert. Mit Hilfe von Trainingsdaten bestimmen wir ein Markov-Modell für jede Anwendung und verwenden diese Modelle zur Klassifizierung neuer TCP-Verbindungen. Anhand von echten Verkehrsaufnahmen konnten wir zeigen, dass sich die Markov-Modelle verschiedener Anwendungen insbesondere bei der kombinierten Betrachtung von TCP-Flags und Paketlängen ausreichend stark unterscheiden, um die Anwendungen gut voneinander abgrenzen und somit eine hohe Klassifikationsgenauigkeit erreichen zu können. Unsere Auswertung ergab zudem, dass die Untersuchung der ersten zehn Pakete einer TCP-Verbindung ausreicht, um die Klassifizierung durchzuführen.

In einem direkten Vergleich mit dem bekannten Verfahren von Bernaille erzielte unser Ansatz bessere Klassifikationsergebnisse. Ein wesentlicher Vorteil ist zudem, dass die Trainingsphase bei unserem Ansatz ein deterministisches Modell ergibt, während der von Bernaille verwendete Cluster-Algorithmus auf zufällig gewählte Initialisierungswerte angewiesen ist, die das Ergebnis unserer Erfahrung nach recht stark beeinflussen.

Unsere bisherige Evaluierung basiert auf Verkehrsdaten für eine kleine Anzahl von Client-Server- und Peer-to-Peer-Anwendungen. Wir haben vor, Untersuchungen mit weiteren Verkehrsdaten, die eine vielfältigere Verkehrszusammensetzung beinhalten, vorzunehmen. Da bei vielen Anwendungen die Eigenschaften eines Pakets wesentlich von seiner Position innerhalb der TCP-Verbindung abhängen, arbeiten wir zudem an einer Verbesserung des Verfahrens, die auch die Paketpositionen in den Markov-Modellen berücksichtigt [11]. Außerdem soll untersucht werden, inwieweit das Verfahren auch für die Klassifizierung von UDP-Verkehr geeignet ist.

Danksagung

Wir danken der Deutschen Forschungsgemeinschaft (DFG) für die Förderung des LUPUS-Projekts (DFG-Geschäftszeichen: CA 595/1-1), in dessen Rahmen die vorgestellte Forschungsarbeit durchgeführt wurde.

Literatur

[1] L. Bernaille, R. Teixeira and K. Salamatian, „Early application identification," in *Proc. of ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, Lisboa, Portugal, Dezember 2006.

[2] A. W. Moore and K. Papagiannaki, „Toward the accurate identification of network applications," in *Proc. of Passive and Active Network Measurement Workshop (PAM)*, Boston, USA, März 2005.

[3] H.-C. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos and K. Lee, „Internet traffic classification demystified: Myths, caveats, and the best practices," in *Proc. of ACM International Conference on Emerging Networking Experiments and Technologies (CoNEXT)*, Madrid, Spanien, Dezember 2008.

[4] T. T. T. Nguyen and G. Armitage, „A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, 2008.

[5] C. Wright, F. Monrose and G. Masson, „HMM profiles for network traffic classification (extended abstract)," in *Proc. of Workshop on Visualization and Data Mining for Computer Security (VizSEC/DMSEC)*, Fairfax, USA, Oktober 2004.

[6] A. Dainotti, W. D. Donato, A. Pescapè and P. S. Rossi, „Classification of network traffic via packet-level hidden markov models," in *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)*, New Orleans, USA, November 2008.

[7] H. Dai, G. Münz, L. Braun and G. Carle, „TCP-Verkehrsklassifizierung mit Markov-Modellen," in *Leistungs-, Zuverlässigkeits- und Verlässlichkeitsbewertung von Kommunikationsnetzen und Verteilten Systemen, 5. GI/ITG-Workshop MMBnet 2009*, Hamburg, September 2009.

[8] L. Bernaille, A. Soule, I. Akodjenou and K. Salamatian, „Blind application recognition through behavioral classification," CNRS LIP6, Paris, Frankreich, Tech. Rep., 2005.

[9] J. M. Estevez-Tapiador, P. Garcia-Teodoro and J. E. Diaz-Verdejo, „Stochastic protocol modeling for anomaly based network intrusion detection," in *Proc. of IEEE International Workshop on Information Assurance (IWIA)*, Darmstadt, März 2003.

[10] L. Bernaille, Homepage of early application identification, http://www-rp.lip6.fr/∼teixeira/bernaill/earlyclassif.html, 2009.

[11] G. Münz, H. Dai, L. Braun and G. Carle, „TCP Traffic Classification Using Markov Models", in *Proc. of Workshop on Traffic Monitoring and Analysis (TMA)*, Zürich, Schweiz, April 2010.