Zuerst ersch. in: Forum : das Forschungsmagazin der Hochschule Konstanz ; 2012/2013 (2012). - S. 52-58

ADAPTIVE NAT-TRAVERSIERUNG

Alexander Diener, Oliver Haase, Jürgen Wäsch, Thomas Zink

Alexander Diener

2007–2011 Studium der Wirtschaftsinformatik an der HTWG Konstanz. 2011–2012 Masterstudium Informatik mit Schwerpunkt Software Engineering ebenfalls an der HTWG Konstanz. Seit Abschluss des Studiums wissenschaftliche Hilfskraft im Projekt „Transparente Integration von NAT-Traversierungstechniken in Java".

Prof. Dr. Oliver Haase

Studium der Informatik an der Universität Karlsruhe, danach Promotion zum Dr.-Ing. an der Universität Siegen. 1998–2005 Industrieforschung, zuerst bei NEC Europe in Heidelberg, dann bei den Bell Labs in Holmdel, New Jersey. Seit Sept. 2005 Professor für Verteilte Systeme und Software Engineering an der Fakultät Informatik der HTWG Konstanz. Zahlreiche Konferenz- und Zeitschriftenpublikationen, Patente, Bücher und Buchkapitel. Seit März 2011 Leiter des Informations- und Medienzentrums der HTWG.

Prof. Dr.-Ing. Jürgen Wäsch

Studium der Informatik und Wirtschaftswissenschaften an der Universität Kaiserslautern. 1993–1999 GMD-Forschungszentrum Informationstechnik GmbH in Darmstadt. 1998–1999 externer Berater bei der Software AG. 1999 Promotion an der TU Darmstadt. 2000–2003 Bereichsleiter bei der e-pro solutions GmbH in Stuttgart. Seit 2004 Professor für E-Business Technologien und Datenbanksysteme an der HTWG Konstanz. 2008 Forschungssemester bei der SAP AG. Seit 2009 Studiendekan und Prodekan der Fakultät Informatik sowie Studiengangsleiter Wirtschaftsinformatik.

Thomas Zink

erhielt 2002 ein Diplom in Wirtschaftsinformatik von der ASW Saarland. Danach arbeitete er bis Herbst 2004 als Network Security Engineer bei einer IT Sicherheitsfirma. Nach einer Babypause kam er 2006 an die Universität Konstanz, um ein Studium zum M.Sc. in Computer Science durchzuführen, welches er im Frühjahr 2009 abschloss. Seither Promotionsstudent an der Universität Konstanz und seit Oktober 2011 an der HTWG als akademischer Mitarbeiter beschäftigt.

EINLEITUNG UND MOTIVATION

Die Kommunikation zwischen Rechnern im Internet basiert auf dem Internet Protokoll (IP). Ursprünglich und auch heute noch vorwiegend wird IPv4 eingesetzt. Da immer mehr Endgeräte eine IP-Adresse benötigen, wird die Anzahl der verfügbaren IPv4-Adressen in naher Zukunft zu Ende gehen. Aus diesem Grund wurde bereits 1995 der Nachfolger IPv6 definiert, der deutlich mehr IP-Adressen ermöglicht. Zur Überbrückung des Adress-Problems bis zur globalen Verbreitung von IPv6 wird Network Address Translation (NAT) eingesetzt. Mit diesem Prinzip können sich mehrere Rechner eine gemeinsame IP-Adresse teilen. Jedoch wird dadurch das Ende-zu-Ende Prinzip des Internets gebrochen und nicht jeder Rechner ist direkt adressierbar. Besonders in Peer-to-Peer (P2P) Netzwerken ist dies ein großes Problem, da sich hier die Peers meist hinter einem NAT befinden. Auch bei einer globalen Verbreitung von IPv6 werden die NAT-Geräte nicht verschwinden, da sich mit der Zeit viele Einsatzmöglichkeiten, wie die Abschottung von ganzen Netzwerken, ergeben haben. Diese tragen dazu bei, dass NAT auch weiterhin eingesetzt werden wird.

Um die Erreichbarkeit von Rechnern hinter einem NAT trotzdem zu ermöglichen, wurden verschiedene NAT-Traversierungstechniken vorgeschlagen. Der Erfolg einer solchen Technik ist abhängig von der konkreten NAT-Situation. Unter dem Begriff NAT-Situation wird die Kombination der NAT-Geräte verstanden, die sich zwischen den Peers befinden. Das NAT-Verhalten bezeichnet hingegen das NAT-Gerät, hinter dem sich ein einzelner Peer befindet. Um zu entscheiden, welche Traversierungstechniken erfolgreich eingesetzt werden können, sind fundierte Kenntnisse über die NAT-Situation und die Funktionsweise der Traversierungstechniken notwendig. Meist sind diese Kenntnisse jedoch nicht vorhanden. Aus diesem Grund ist ein Connectivity Manager notwendig, der die aktuelle NAT-Situation untersucht und optimale NAT-Traversierungstechniken auswählt. Die Anwendung bzw. der Benutzer benötigen keinerlei Kenntnisse über diesen Prozess, denn er geschieht für sie vollkommen transparent.

1 NETWORK ADDRESS TRANSLATION

Mittels NAT können verschiedene Adressräume miteinander verbunden werden. Innerhalb der einzelnen Adressräume werden für die Adressierung IP-Adressen verwendet, deren Gültigkeiten auf die jeweiligen Räume beschränkt sind. Die Verbindung dieser Bereiche geschieht durch ein NAT-Gerät, das Zuordnungen der Adressen erstellt und Adressinformationen im Header der Datenpakete entsprechend anpasst. Da das NAT in den meisten Fällen sowohl die IP-Adresse als auch die Portnummer verändert, wird von Network Address and Port Translation (NAPT) gesprochen. Jedoch werden die Begriffe NAT und NAPT in der Regel synonym verwendet.

Das NAT-Verhalten wird in Mapping- und Filterverhalten unterteilt. Das Mappingverhalten beschreibt den Umgang mit ausgehenden Verbindungen, so zum Beispiel die Strategie bei der Zuweisung von Portnummern. Das Filterverhalten gibt den Umgang mit eingehenden Verbindungen an. Es beschreibt, unter welchen Bedingungen sie erlaubt oder verboten werden. Prinzipiell können die Verhaltensweisen jeweils in die vier Strategien

- Endpoint-Independent,
- Address-Dependent,
- Address- and Port-Dependent und
- Connection Dependent

unterteilt werden. Das Verhalten hängt jedoch stark vom Hersteller und dessen Implementierung ab. Ursprünglich wurde NAT nur für einen vorübergehenden Einsatz konzipiert, weshalb weitestgehend auf eine Standardisierung verzichtet wurde. Aus diesem Grund ist es auch möglich, dass sich das Verhalten im Laufe der Zeit und unter Last verändert [1].

2 NAT-TRAVERSIERUNG

Um trotz der Präsenz von NAT-Geräten eine direkte Verbindung zwischen Peers aufbauen zu können, bedarf es spezieller Techniken. Diese führen jedoch nur in bestimmten NAT-Situationen zum Erfolg. Jede bisher entwickelte Technik zur NAT-Durchdringung benötigt eine öffentlich erreichbare Instanz, den Mediator, um den Verbindungsaufbau zu koordinieren und die Endpunkte zwischen den Peers auszutauschen. Aus diesem Grund ist eine bestehende Verbindung zum Mediator für den Einsatz der Techniken zwingend erforderlich. Die hier vorgestellten NAT-Traversierungstechniken wurden bereits im Rahmen des Projektes „Universal Connection Establishment“ der HTWG Konstanz realisiert [2].

2.1 Relaying

Relaying ist eine Traversierungstechnik, die in allen NAT-Situationen zum Erfolg führt. Sie umgeht die NAT-Geräte, indem die Peers mit einem Relay-Server eine Client-Server-Verbindung aufbauen und dieser die Daten zwischen den Peers weiterleitet [3]. Durch die Beteiligung einer externen dritten Partei ist diese NAT-Traversierungstechnik allerdings sehr „teuer“, da die Latenz im Gegensatz zu einer direkten Verbindung vergrößert wird und die Ressourcen des Relay-Servers beansprucht werden. Aus diesen Gründen eignet sich Relaying nur als Fallback-Technik. Sie wird nur eingesetzt, wenn mit keinem anderen Verfahren eine direkte Verbindung aufgebaut werden kann.

2.2 Connection Reversal

Befindet sich nur der Server hinter einem NAT und der Client ist öffentlich erreichbar, kann eine Verbindung mittels Connection Reversal hergestellt werden. Dabei tauschen Client und Server die Rollen und der Server baut eine Verbindung mit dem erreichbaren Client auf [3].

2.3 Hole Punching

Mit Hole Punching kann eine Verbindung zwischen zwei Peers aufgebaut werden, obwohl sich beide hinter einem NAT befinden. Dazu versuchen beide Seiten, gleichzeitig eine direkte Verbindung aufzubauen. Durch die ausgehenden Verbindungsversuche werden Mappings in den NAT-Geräten angelegt, es werden sozusagen Löcher in das NAT geschlagen. Verbindungsversuche des jeweils anderen Peers interpretiert das NAT als Antwort auf die zuvor ausgehende Verbindung. Allerdings führt dieses Vorgehen nicht bei allen Verhaltensweisen zum Erfolg. In [4, 5] wird die Funktionsweise von Hole Punching ausführlich erläutert sowie verschiedene Vorgehensweisen verglichen.

3 AUSWAHL OPTIMALER NAT-TRAVERSIERUNGSTECHNIKEN

Eine optimale NAT-Traversierungstechnik ist geeignet für die aktuelle NAT-Situation und konform zu den Anforderungen, die an sie gestellt werden. Um die optimalen NAT-Traversierungstechniken auswählen zu können, müssen zunächst Kriterien definiert werden. Anhand dieser Kriterien wird ein Auswahlprozess durchgeführt.

3.1 Kriterien

Die Auswahlkriterien können in gegeben und beeinflussbar unterteilt werden. Die gegebenen Kriterien werden durch die vorliegende NAT-Situation charakterisiert und können weder von der Anwendung, noch durch den Einsatz einer bestimmten Traversierungstechnik verändert werden. Für den Erfolg oder Misserfolg einer Traversierungstechnik sind

- das **Mappingverhalten** und
- das **Filterverhalten**

der NAT-Geräte entscheidend. Die konkrete Strategie kann mit Hilfe eines STUN-Servers gemäß [6] bestimmt werden.

Beeinflussbare Kriterien werden von der Anwendung oder den NAT-Traversierungstechniken vorgegeben.

- Anhand der **Dauer der Verbindung** kann entschieden werden, ob sich ein Wechseln der Verbindungen nach dem Auswahlprozess lohnt oder nicht. Als Grenze zwischen kurz und lang dauernden Verbindungen können 30 Sekunden angesehen werden.
- Die **ConnectionSetupTime** ist ein Maß für die Dauer eines Verbindungsaufbaus mit einer bestimmten Technik. Dazu dient die Anzahl an mindestens und maximal versendeten Nachrichten während dem Aufbau. Diese Anzahl ist im Gegensatz zu einer Messung in einer Zeiteinheit unabhängig von anderen Gegebenheiten wie zum Beispiel der Latenz.
- Die **Service-Klasse** orientiert sich an den Differentiated Services [7] und macht es möglich, verschiedene Quality-of-Service für eine Verbindung zu definieren. Daraus lassen sich Rückschlüsse auf die Art der Anwendung ziehen.
- Das Kriterium **Fallback-Technik** gibt für eine NAT-Traversierungstechnik an, ob es sich um eine Technik handelt, die in jeder NAT-Situation eine Verbindung herstellen kann. Diese Techniken stellen allerdings eine indirekte Verbindung her.

3.2 Auswahlprozess

Die Auswahl optimaler NAT-Traversierungstechniken findet in mehreren Schritten statt. Der Prozess ist in Abbildung 1 veranschaulicht. Zunächst wird, falls verfügbar, mit einer Fallback-Technik eine Verbindung zwischen den Peers hergestellt. Dadurch kann ein Datenaustausch ohne lange Wartezeit starten. Parallel dazu wird das Mapping- und Filterverhalten der NAT-Geräte zwischen Client und Server untersucht. Mit diesen Informationen kann mit einem Entscheidungsbaum bestimmt werden, welche NAT-Traversierungstechniken für die vorliegende NAT-Situation geeignet sind. Diese Menge von Techniken

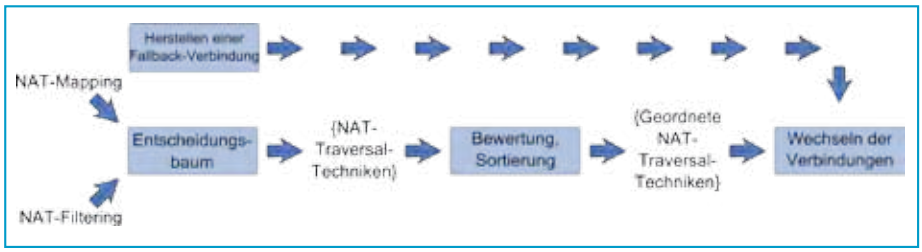

ABB. 1: Auswahlprozess optimaler NAT-Traversierungstechniken

wird anhand der ConnectionSetupTime und anhand von Statistiken bewertet und sortiert. Das Ergebnis ist eine geordnete Liste mit NAT-Traversierungstechniken, die sequentiell abgearbeitet wird, bis eine Verbindung zu Stande gekommen ist. Sobald dies der Fall ist, wird die Fallback-Verbindung durch die direkte Verbindung, die mittels einer NAT-Traversierungstechnik hergestellt wurde, ausgetauscht. Dieser Austausch der Socket-Verbindungen wird in [10] erläutert.

3.3 Entscheidungsbaum

Mit einem Entscheidungsbaum, wie in Abbildung 2, kann anhand einer gegebenen NAT-Situation eine Auswahl geeigneter Traversierungstechniken durchgeführt

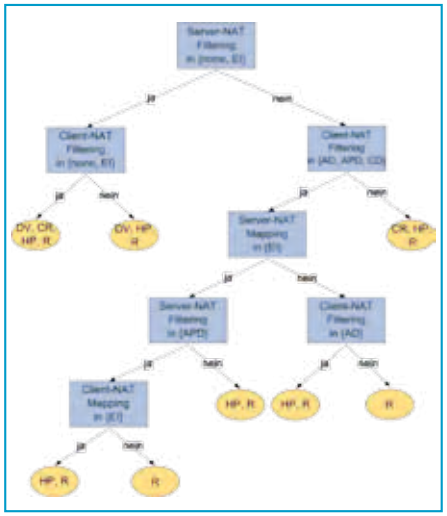

ABB. 2: Beispielhafter Entscheidungsbaum zur Auswahl geeigneter Traversierungstechniken; Abkürzungen: EI: Endpoint-Independent, AD: Address-Dependent, APD: Adress- and Port-Dependent, CD: Connection-Dependent, DV: Direktverbindung, CR: Connection Reversal, HP: Hole Punching, R: Relaying

werden. Genau betrachtet wird eine Klassifikation durchgeführt, deren Ergebnis eine Menge von NAT-Traversierungstechniken ist. Die Entscheidungen, die dabei getroffen werden, beziehen sich auf die NAT-Situation.

Durch die Generierung des Entscheidungsbaumes ist es möglich, nachträglich NAT-Traversierungstechniken hinzuzufügen und den Baum neu zu generieren. Die Generierung des Baumes erfolgt mit einer Menge von Trainingsdatensätzen, die von den Techniken selbst geliefert werden.

3.4 Bewertung und Sortierung

Das Ergebnis des Entscheidungsbaumes ist eine Menge von geeigneten NAT-Traversierungstechniken für die aktuelle NAT-Situation. Prinzipiell ist es möglich, mit diesen Verfahren parallel Verbindungsversuche durchzuführen. Dies erzeugt allerdings einen unnötigen Nachrichten-Overhead und eine hohe Ressourcenauslastung, denn einige Techniken sind schneller als andere und werden in der gegebenen NAT-Situation immer zuerst erfolgreich eine Verbindung aufbauen. Aus diesem Grund ist es sinnvoll, die Techniken zu bewerten und zu sortieren und die resultierende Liste sequentiell abzuarbeiten.

Die Bewertung und Sortierung der Menge erfolgt anhand der Connection SetupTime und anhand von Statistiken. Dafür werden zwei Rangfolgen erstellt. Eine spiegelt die ConnectionSetupTime wider, die andere basiert auf Statistiken und somit auf Erfahrungswerten. Für jede dieser beiden Rangfolgen kann die Anwendung bzw. der Benutzer eine Gewichtung angeben. Anhand der Rangfolgen und der Gewichtungen wird eine kombinierte Rangfolge berechnet. Diese wird sequentiell abgearbeitet, bis eine direkte Verbindung zwischen den Peers besteht. Die Berechnung der Rangfolgen wird in Abbildung 3 verdeutlicht.

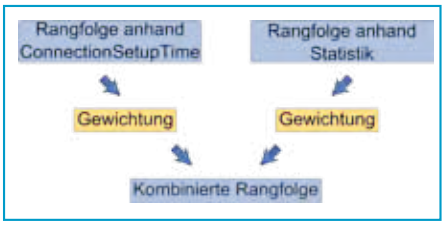

ABB. 3: Berechnung der kombinierten Rangfolge

Die theoretische Betrachtung einer NAT-Traversierungstechnik und der Einsatz in einer realen Umgebung können unterschiedliche Ergebnisse hervorbringen. Um dies auszugleichen, werden Statistiken berücksichtigt. Jedoch ist die Berücksichtigung von Statistiken erst sinnvoll, wenn eine entsprechend große Datenbasis zur Verfügung steht. Bei einer lokalen Lösung, bei der jeder Peer für sich Erfahrungswerte sammelt, ist dies erst mit zunehmender Aktivität möglich. Neue Peers müssen komplett ohne diese Daten auskommen. Um auch ihnen Zugriff auf gesammelte Erfahrungswerte zu geben, können die Daten in einer globalen Lösung allen Peers zur Verfügung gestellt werden. Dies kann durch Ablegen in einer zentralen Instanz, wie dem Mediator, oder in einer verteilten Hashtabelle geschehen.

3.5 Uninformierter Auswahlprozess

Bisher wurde davon ausgegangen, dass Informationen über die NAT-Situation verfügbar sind. Ein besonderer Fall tritt deshalb ein, wenn diese nicht zur Verfügung stehen, zum Beispiel wenn keine Verbindung zu einem STUN-Server hergestellt werden kann. In diesem Fall muss der Auswahlprozess ohne die Bestimmung von geeigneten Verfahren mittels Entscheidungsbaum erfolgen. In dieser Situation

ist es sinnvoll, alle vorhandenen NAT-Traversierungstechniken anhand der ConnectionSetupTime zu sortieren und diese Liste abzuarbeiten.

3.6 Lokalisierung

Nachdem der Auswahlprozess definiert wurde, stellt sich die Frage, wer ihn ausführt. Dafür kommen grundsätzlich die drei Parteien, Client, Server und Mediator, in Frage. Jedoch wird bei der Betrachtung des Ende-zu-Ende Prinzips deutlich, dass die Auswahllogik in einem der Peers liegen muss und nicht beim Mediator, als Vermittler, liegen darf.

Führt der Server den Auswahlprozess durch, benötigt der Client keinerlei Logik. Jedoch hat dies eine enorme Belastung für den Server zu bedeuten, falls dieser viele Clients bedienen muss.

Eine parallele Ausführung beim Client und beim Server führt durch die identische Logik zum selben Ergebnis. Durch die Verteilung wird der Prozess fehleranfälliger und schwerer nachzuvollziehen.

Eine Verlagerung des Prozesses auf den Client hat den Vorteil, dass der Server nicht weiter belastet wird. Außerdem hat der Client ein Interesse daran, sich mit dem Server zu verbinden, und der Prozess wird dadurch weniger komplex. Nach der Berechnung des Ergebnisses teilt der Client über den Mediator dem Server mit, welches Verfahren anzuwenden ist.

Bei allen vorgestellten Möglichkeiten müssen vor dem Auswahlprozess die verfügbaren NAT-Traversierungstechniken von Client und Server abgeglichen werden. Dies stellt sicher, dass nur Techniken eingesetzt werden, die beide Seiten unterstützen.

4 SICHERHEIT

Anwendungen, die mit sensiblen Daten umgehen und diese übermitteln müssen, stellen nicht nur Anforderungen an den Verbindungsaufbau, sondern auch an die Sicherheit einer Verbindung. Sicherheit in einem Netzwerk umfasst viele Bereiche und Aspekte. Jedoch sind in diesem Kontext nur solche von Interesse, die sich mit dem Verbindungsaufbau bzw. der Verbindung selbst befassen. Dies sind

- Authentizität,
- Integrität und
- Vertraulichkeit.

Die hier beschriebenen Konzepte zur Realisierung werden auf Anwendungsschicht (ausgehend vom OSI-Modell) ausgeführt. Dies ist allerdings nur notwendig, wenn auf den tieferen Ebenen unverschlüsselt kommuniziert wird. Bei Verwendung von IPSec wird grundsätzlich jede Kommunikation verschlüsselt, da IPSec auf einer niedrigeren Schicht arbeitet. Insofern ist eine redundante Sicherung der Verbindung auf Anwendungsschicht nicht mehr notwendig.

4.1 Authentizität

Die Authentizität sichert zu, dass ein Kommunikationspartner der ist, für den er sich ausgibt. In den meisten Fällen geschieht dies mittels Zertifikaten. Um diese nutzen zu können, muss allerdings eine entsprechende Infrastruktur in einem Netzwerk bereitgestellt sein. Zentrale Stellen müssen Zertifikate ausstellen und diese signieren.

Generell ist das Thema Authentizität in Netzwerken ein Problem. Besonders in P2P-Netzen existiert die notwendige Infrastruktur meist nicht. Jede bisher entwickelte NAT-Traversierungstechnik benötigt eine dritte Instanz. Die Peers müssen dieser Instanz vertrauen und davon ausgehen, dass die gelieferten Informationen, wie die Endpunkte, korrekt und nicht manipuliert sind.

4.2 Integrität

Integrität bezeichnet die Unversehrtheit einer Nachricht. Die Überprüfung geschieht mittels Keyed-Hash Message Authentication Code (HMAC). Dazu wird auf Basis einer kryptografischen Hash-Funktion, einem gemeinsamen geheimen Schlüssel und der Nachricht eine Prüfsumme berechnet. Diese Prüfsumme kann von beiden Peers berechnet und verglichen werden.

4.3 Vertraulichkeit

Eine Verbindung ist vertraulich, wenn die übermittelten Nachrichten nur von befugten Peers gelesen werden können. Um dies zu realisieren, werden die Nachrichten verschlüsselt. Dazu dient ein symmetrischer geheimer Schlüssel. Der Austausch dieses Schlüssels geschieht mit dem Diffie-Hellman-Schlüsselaustausch. Dieser ermöglicht es, einen symmetrischen Schlüssel sicher über ein unsicheres Medium auszutauschen. Der hier verwendete Schlüssel kann auch zur Berechnung des HMAC genutzt werden.

5 SOFTWAREARCHITEKTUR UND -DESIGN

Vor der Konzeption von Client, Server und Mediator stellt sich zunächst die Frage, wie Einsatzszenarien für den Connectivity Manager aussehen können. Beim Entwurf der Softwarearchitektur müssen besonders die Erweiterbarkeit um neue NAT-Traversierungstechniken und die Transparenz für eine Anwendung beachtet werden.

5.1 Einsatzszenarien

Der Connectivity Manager kann in zwei unterschiedlichen Szenarien eingesetzt werden. Zum einen ist dies das Einbinden als externe Bibliothek in eine bestehende Software und zum anderen die Bereitstellung als Service in einem lokalen Netzwerk.

Bei der Verwendung als Bibliothek muss der Connectivity Manager in jede Software eingebunden werden, die ihn verwenden möchte. Im Quellcode muss die Erzeugung der Sockets entsprechend der Bibliothek angepasst werden. Dieses Szenario bildet die Grundlage für die weitere Konzeption.

Mit einem Wrapper um die zuvor beschrieben Bibliothek kann der Connectivity Manager in einem lokalen Netzwerk als Service bereitgestellt werden. Alle Anwendungen nutzen dieselbe Instanz und müssen sich bei dieser registrieren. Der Wrapper übernimmt dabei die Verwaltung der registrierten Anwendungen. Der Service stellt für die Anwendungen Verbindungen zu anderen Peers her und fungiert somit selbst als Relay.

5.2 Kommunikationsprotokoll

Basis für die Kommunikation zwischen Client. Server und Mediator ist das Session Traversal Utilities for NAT (STUN) Protokoll, das in [8] definiert wird. Dieses erweiterbare Nachrichtenprotokoll macht es möglich Type-Length-Value-Kodierungen zu übertragen. Damit können auch eigene Attribute, wie die einzusetzende NAT-Traversierungstechnik, definiert werden.

5.3 Client

Der Client verfügt über alle Komponenten, die zum Verbindungsaufbau notwendig sind. Abbildung 4 stellt die Softwarearchitektur in der FMC-Blockdiagramm Notation dar. Jedoch sind nicht alle Komponenten für einen Verbindungsaufbau zwingend erforderlich. Ohne die optionalen Komponenten ist ein uninformierter Auswahlprozess, wie in Abschnitt 3.5 beschrieben, möglich.

Jede Komponente erfüllt eine fest definierte Aufgabe.

- Die **UCE Connection** stellt Sockets und Fabriken bereit, über die eine Anwendung den Connectivity Manager nutzen kann. Dadurch bleiben Logik und Funktionsweise für die Anwendung transparent.
- Der **Connection Manager** ist für die Koordinierung aller Aktivitäten zuständig.
- Der Wechsel von Socket-Verbindungen ist Aufgabe des **Socket Switcher.**

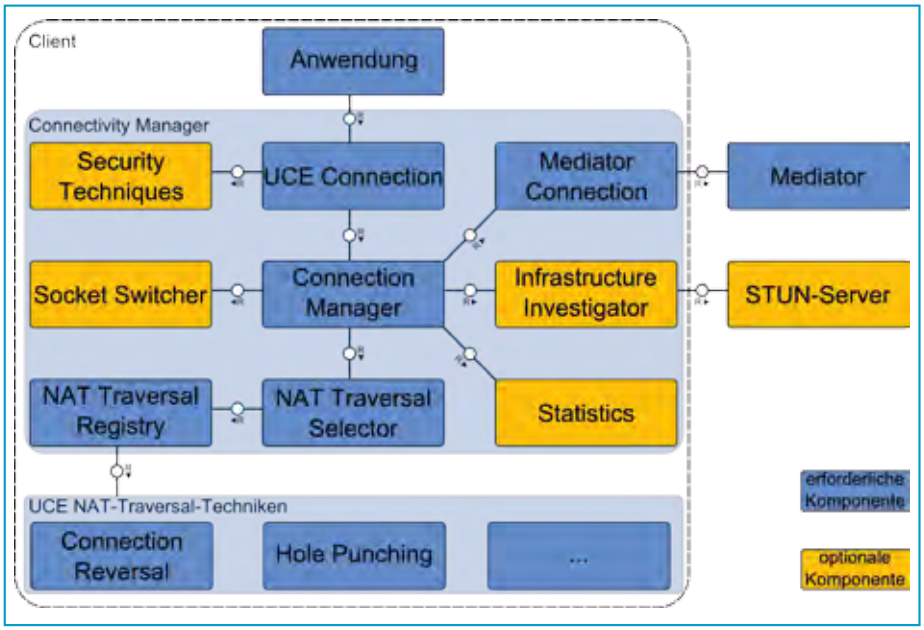

ABB. 4: Softwarearchitektur des Clients in der FMC-Blockdiagramm Notation.

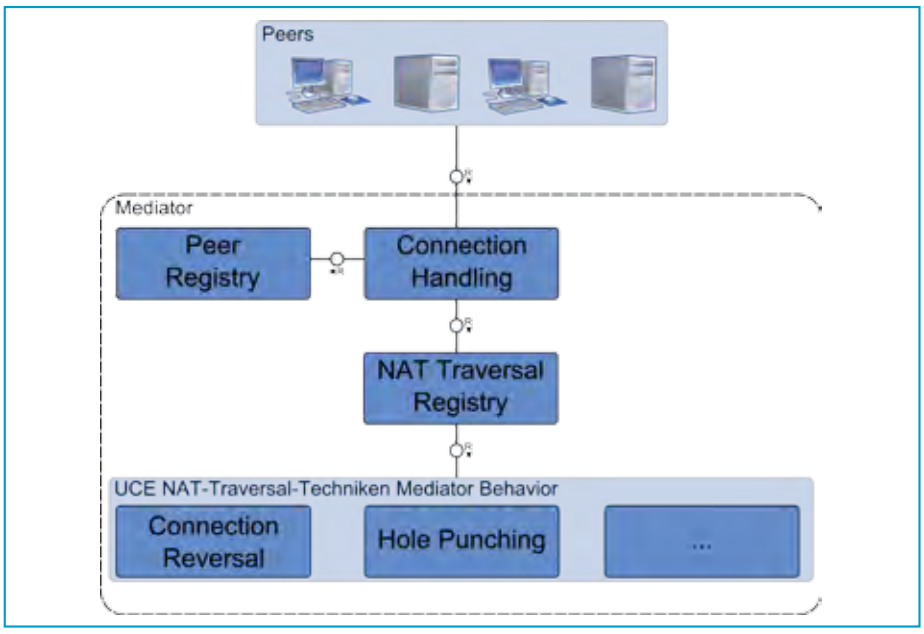

ABB. 5: Softwarearchitektur des Mediators in der FMC-Blockdiagramm Notation.

- Verbindungen zum Mediator werden über die **Mediator Connection** hergestellt.
- Der **Infrastructure Investigator** untersucht das NAT-Verhalten und kommuniziert dazu mit einem öffentlich erreichbaren STUN-Server.
- Die **NAT Traversal Registry** verwaltet verfügbare NAT-Traversierungstechniken und liefert einen einheitlichen Zugriff darauf.
- In der Komponente **Statistics** werden Erfahrungswerte gesammelt und ausgewertet.

- Techniken und Methoden zur Bereitstellung von sicheren Verbindungen werden in der Komponente **Security Techniques** zusammengefasst.

5.4 Server

Die Softwarearchitektur des Servers unterscheidet sich nur minimal von der des Clients. Lediglich die Komponenten **NAT Traversal Selector** und **Statistics** zur Auswahl optimaler NAT-Traversierungstechniken fallen weg, da der Auswahlprozess vom Client durchgeführt wird. Trotzdem gibt es keine separate Server-Bibliothek, da der Server wiederum in der Rolle eines Client einen anderen Server kontaktieren kann. In diesem Fall benötigt er die Komponenten für den Auswahlprozess.

5.5 Mediator

Abbildung 5 veranschaulicht die Softwarearchitektur des Mediators. Wie auch der Client und der Server muss der Mediator um neue NAT-Traversierungstechniken erweiterbar sein. Dies hat den Grund, dass das Verhalten des Mediators abhängig von der eingesetzten NAT-Traversierungstechnik ist.

5.6 Einbindung in Java RMI

Das Einbinden des Connectivity Manager in Java RMI erfolgt über Custom Socket Factories. Sie schreiben vor, welche Sockets, in diesem Fall ein UCE-Socket, für die RMI-Kommunikation verwendet werden. Abbildung 6 zeigt die Einbindung mittels Custom Socket Factories. Mit dieser Vorgehensweise kann der Connectivity Manager auch in die P2P-RMI integriert werden, die in [11] beschrieben wird. Es ist aber auch möglich, den Connectivity Manager direkt in einer Anwendung zu nutzen.

6 EVALUATION

Die Evaluation hat das Verhalten des Connectivity Manager in einer möglichst realen Umgebung gezeigt. Dazu wurde eine Testumgebung entwickelt, denn

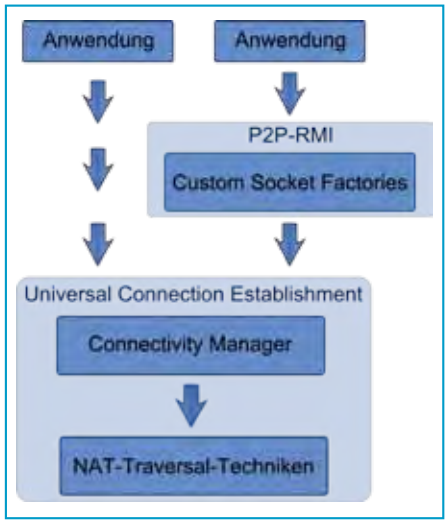

ABB. 6: Möglichkeiten zur Einbindung des Connectivity Manager

bestehende Umgebungen zur automatischen Simulation von verteilten Systemen stellen öffentlich erreichbare Knoten zur Verfügung. Sie blenden damit das Problem des Verbindungsaufbaus komplett aus.

6.1 Umgebung

Abbildung 7 zeigt die Umgebung für die Evaluation. Client und Server laufen je in einer virtuellen Maschine, aber auf unterschiedlichen Rechnern. Jeder Rechner besitzt zwei virtuelle Netzwerkschnittstellen, von denen eine als interne und eine als externe Schnittstelle verwendet wird. Dazu wird die Kommunikation der internen Schnittstelle über die externe Schnittstelle umgeleitet. Die Konfiguration des NAT-Verhaltens erfolgt mit iptables [9]. Zwei Demoanwendungen stellen mit dem Connectivity Manager eine Verbindung zueinander her und tauschen zur Überprüfung der Verbindung eine Kontrollnachricht aus.

6.2 Parameter

Für die Evaluation wurden verschiedene NAT-Verhalten verwendet. Diese sind

- ein NAT mit Endpoint-Independent Mapping und Filterung,
- ein NAT mit Endpoint-Independent Mapping und Address-Dependent Filterung,
- ein NAT mit Endpoint-Independent Mapping und Address- and Port-Dependent Filterung,
- ein NAT mit Connection-Dependent Mapping und Connection-Dependent Filterung und
- ein öffentlich erreichbarer Peer ohne NAT.

Evaluiert wurden alle 25 möglichen NAT-Situationen, die sich daraus bilden lassen. Für jedes Verfahren wurde ein Timeout von 30 Sekunden gesetzt. Wird innerhalb dieser Zeitdauer keine Verbindung aufgebaut, gilt der Versuch als gescheitert und die nächste NAT-Traversierungstechnik wird angewandt. Während der Evaluation wurde kein Wechseln der Socket-Verbindungen durchgeführt und auch auf den Einsatz von Statistiken verzichtet.

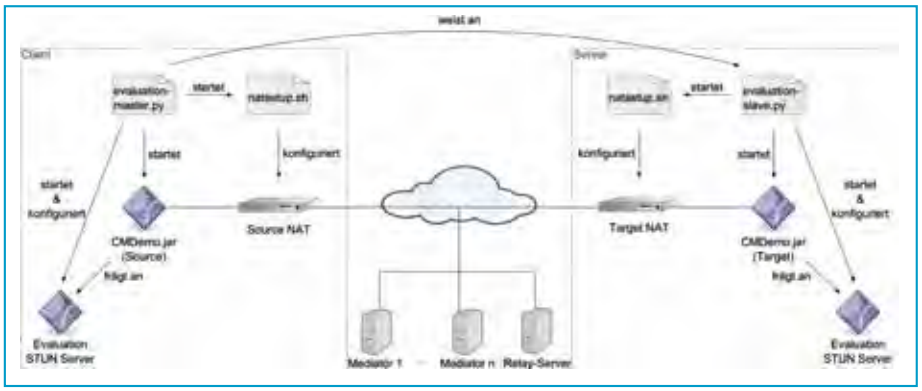

ABB. 7: Umgebung für die Evaluation

6.3 Ergebnisse

In der Evaluation wurde in allen Fällen erfolgreich eine Verbindung hergestellt. Der Grund dafür ist natürlich die Verfügbarkeit einer Fallback-Technik. Das Diagramm in Abbildung 8 zeigt die Erfolgsquote des Verbindungsaufbaus in Abhängigkeit des Versuches. In 60 % der Fälle konnte mit der ersten, vom Connectivity Manager berechneten, NAT-Traversierungstechnik erfolgreich eine Verbindung aufgebaut werden. Spätestens nach dem zweiten Versuch konnte in 88 % der Fälle erfolgreich eine Verbindung aufgebaut werden.

Im Durchschnitt dauerte der Verbindungsaufbau 7,6 Sekunden. Bei der Evaluation wurde kein Wechseln von Verbindungen durchgeführt, was bedeutet, dass die Anwendung in dieser Zeit gewartet hat. Bei der Betrachtung dieses Wertes muss allerdings die geringe Latenz zwischen Client und Server beachtet werden, wodurch die Verweildauer von Mappings in den NAT-Geräten zu keinem Problem wurde. Die Zeitdauern für den Verbindungsaufbau betrugen zwischen 0,5 und über 40 Sekunden. Dies resultiert aus dem Timeout von 30 Sekunden, der für jede NAT-Traversierungstechnik gesetzt wurde.

7 FAZIT

Mit dem Connectivity Manager ist es möglich, ohne Kenntnisse über NAT-Traversierungstechniken und die aktuelle NAT-Situation eine Verbindung zwischen Peers aufzubauen. Ist eine Fallback-Technik verfügbar, ist dies sogar in jeder denkbaren NAT-Situation möglich. Die Vorgehensweise und die eingesetzte NAT-Traversierungstechnik bleiben dabei für die Anwendung bzw. den Benutzer transparent. Durch die Erweiterbarkeit des Connectivity Manager ist es möglich, nachträglich neue, noch nicht bekannte NAT-Traversierungstechniken einzubinden und zu verwenden.

Die Evaluation hat dies bestätigt und es konnten wichtige Erkenntnisse über die

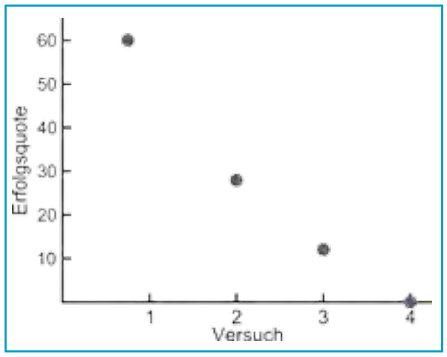

ABB. 8: Erfolgsquote in Abhängigkeit des Versuches

Möglichkeiten und den Einsatz in einer realen Umgebung gewonnen werden. Generell ist das Testen von verteilten Systemen schwierig. In diesem Fall lag dies besonders daran, dass es keine Testumgebungen gibt, die den Fokus auf den Verbindungsaufbau setzen. Dieses Problem wurde mit der Entwicklung einer eigenen Umgebung gelöst, die genau dieses Problem adressiert.

DANKSAGUNG

Diese Arbeit wurde im Kontext des FHprofUnt-Projekts „Transparente Integration von NAT-Traversierungstechniken in Java“ durchgeführt, das vom BMBF und der Firma Seitenbau GmbH, Konstanz, gefördert wird. Projektpartner sind die HTWG Konstanz, Seitenbau GmbH und die Universität Konstanz.

LITERATUR

[1] P. Srisuresh, B. Ford, D. Kegel: „State of Peer-to-Peer (P2P) Communication across Network Address Translators (NATs)“. RFC 5128, Internet Engineering Task Force, März 2008. Online: http://tools.ietf.org/html/rfc5128

[2] HTWG Konstanz: „Universal Connection Establishment“. Online: http://ice.in.htwg-konstanz.de/

[3] B. Ford, P. Srisuresh, D. Kegel: „Peer-to-peer Communication across Network Address Translators“. In: USENIX Annual Technical Conference, 2005, S. 179–192.

[4] D. Maier, O. Haase, J. Wäsch, M. Waldvogel: „A comparative analysis of NAT hole punching“. In: Forum, Forschungsmagazin der Hochschule Konstanz, 2011, S. 40–48.

[5] D. Maier, O. Haase, J. Wäsch, M. Waldvogel: „NAT Hole Punching Revisited“. In: Proceedings of the 36th IEEE Conference on Local Computer Networks (LCN), Bonn, Germany, October 4-7 2011.

[6] D. MacDonald, B. Lowekamp: „NAT Behavior Discovery Using Session Traversal Utilities for NAT (STUN)“. RFC 5780, Internet Engineering Task Force, Mai 2010. Online: http://tools.ietf.org/html/rfc5780

[7] J. Babiarz, K. Chan, F. Baker: „Configuration Guidelines for DiffServ Service Classes“. RFC 4594, Internet Engineering Task Force, Ausgust 2006. Online: http://tools.ietf.org/html/rfc4594

[8] J. Rosenberg, R. Mahy, P. Matthews, D. Wing: „Session Traversal Utilities for NAT (STUN)“. RFC 5389, Internet Engineering Task Force, Oktober 2008. Online: http://tools.ietf.org/html/rfc5389

[9] R. Spenneberg: „Linux Firewalls mit Iptables & Co“. Pearson Deutschland GmbH, 2006.

[10] S. Böckle: „Tauschen bestehender JavaSocket Verbindungen“. Bachelorarbeit, HTWG Konstanz, 2012.

[11] T. Zink, O. Haase, J. Wäsch, M. Waldvogel: „P2P-RMI: Transparent Distribution of Remote Java Objects”. In: International Journal of Computer Networks & Communications, 2012, S. 17–34.