UNIVERSITÄT
KOBLENZ · LANDAU

Fachbereich 4: Informatik

Moderne Instant-Messaging-Systeme als Plattform für sicherheitskritische kollaborative Anwendungen

Dissertation

zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften
(Doctor rerum naturalium)

vorgelegt von
Dipl.-Informatikerin Anastasia Meletiadou

Gutachter: Prof. Dr. Rüdiger Grimm und Prof. Dr. J. Felix Hampe

Inhaltsverzeichnis

Abbildungsverzeichnis

Zusammenfassung

Während unter dem Begriff Instant Messaging (IM) ursprünglich nur der Austausch von Textnachrichten (Messaging, Chat) verstanden wurde, bieten heutige Instant-Messaging-Systeme (IM-Systeme, IMS) darüber hinaus viele weitere Funktionen wie Sprachtelefonie, Videokonferenz oder Dateiübertragung an. Aktuelle IM-Systeme werden vor allem in der privaten unverbindlichen Kommunikation eingesetzt. Diese Systeme eignen sich – wie in dieser Arbeit gezeigt wird – aber auch für den Einsatz im geschäftlichen Kontext. Zu den Vorzügen, die sich aus der Nutzung eines IM-Systems ergeben, gehören beispielsweise die spontane, flexible und kostengünstige Nutzung unabhängig von Ort und Zeit, die einfache Verteilung von Informationen an mehrere Teilnehmer und die Verfügbarkeit verschiedener Kommunikationskanäle für unterschiedliche Aufgaben.

Der Ursprung dieser Systeme (kurzer informeller Nachrichtenaustausch), ihr modularer Aufbau, die unkontrollierte Installation auf Clients sowie der daraus entstehende Informationsfluss führten zu zahlreichen Fragen und Herausforderungen bezüglich ihrer Sicherheit. So kann beispielsweise nicht ohne weiteres kontrolliert werden, welche Informationen ein Unternehmen auf diesem Weg verlassen. Anders als bei anderen elektronischen Kommunikationsmedien (E-Mail, Web) sind hier noch keine bekannten Sicherheitsmechanismen etabliert (z. B. Spam-Filter, Sicherheitsfunktionen von Webbrowsern). Diese Fragestellungen sind zu betrachten und durch geeignete Maßnahmen zu beantworten, wenn solche Systeme für die Arbeit mit sensiblen und vertraulichen Informationen eingesetzt werden sollen.

Ziel der hier vorgestellten Arbeit ist es, Instant-Messaging-Systeme so zu gestalten, dass durch die Konzeption (bzw. Erweiterung) von geeigneten Sicherheitsmechanismen einerseits die einfache und spontane Nutzung der Systeme möglich bleibt und andererseits trotzdem eine sichere, authentifizierte und verbindliche Kollaboration unterstützt wird.

Dazu wird im ersten Teil eine Einführung in die Thematik gegeben (Kapitel 1), die Motivation erläutert (Kapitel 2) und das betrachtete Forschungsproblem definiert (Kapitel 3). Der erste Teil schließt mit einem Überblick über den Beitrag und die Struktur der Arbeit (Kapitel 4).

Als Grundlage für die weiteren Betrachtungen wird im zweiten Teil ein Überblick über den betrachteten Gegenstand und den allgemeinen Stand der Forschung im Bereich IM-Systeme gegeben. Dazu werden zunächst mit Hilfe eines Featuremodells die Bestandteile, der logische Aufbau und typische Funktionalitäten moderner Instant-Messaging-Systeme vorgestellt und strukturiert (Kapitel 6). Dann wird ein besonderes Augenmerk auf die Sicherheitsaspekte von IM-Systemen gerichtet (Kapitel 7). Anschließend wird ein Überblick über Literatur mit direktem Bezug zur vorliegenden Arbeit (Related Work) einschließlich der dort realisierten Sicherheitsmechanismen gegeben (Kapitel 8 und 9).

Die durch diese Betrachtungen aufgeworfenen Fragen werden im dritten Teil untersucht. Nach einem Überblick über den Lösungsansatz (Kapitel 10) werden zwei konkrete Mechanismen vorgestellt (Kapitel 11): (i) ein Mechanismus zur gegenseitigen Authentifikation innerhalb einer Gruppe (ein Zugangsberechtigungsmechanismus für den Eintritt einer weiteren Person in die Gruppe) und (ii) ein Mechanismus für die Realisierung einer verbindlichen Kollaboration, ein-

schließlich einer langfristigen Aufbewahrung der Kollaborationsergebnisse, basierend auf einem „Separation of Duty“-Konzept. Beide Verfahren werden durch geeignete Verschlüsselungs- und Signierungskonzepte unterstützt. Als erster Schritt hin zu einer konkreten Anwendung werden diese Ansätze in einer generischen Plattform für sicherheitskritische Anwendungen auf IM-Basis strukturiert (Kapitel 12). Zur weiteren Konkretisierung und als „Proof of Concept“ für die entworfenen Verfahren wird die Kollaborationsanwendung „Entscheidungsprozesse in kleinen Gruppen“ genutzt (Kapitel 13). Betrachtet werden dabei Entscheidungsprozesse in verteilten Gruppen, die aus einer Diskussionsphase und einer anschließenden elektronischen Wahl bestehen. Für die Phase der elektronischen Wahl wird ein Protokoll für geheime Wahlen vorgestellt, welches jedem Teilnehmer die eigenständige Berechnung des Wahlergebnisses erlaubt und somit unabhängig von einer separaten Wahlkommission verwendet werden kann. Es folgt die prototypische Realisierung des konkreten Szenarios eines Entscheidungsprozesses mit Hilfe und durch Erweiterung des IM-Systems Spark/Openfire, welches auf dem XMPP-Standard aufbaut (Kapitel 14).

Die Arbeit schließt mit einem Ausblick auf (und einer Aufwandsschätzung für) die Realisierung und den Einsatz derartiger IM-Systeme (d. h. unter Beachtung sicherheitskritischer Aspekte und zur Realisierung kollaborativer Prozesse) im unternehmerischen Kontext (Kapitel 15).

Abstract

Many Instant Messaging (IM) systems like Skype or Spark offer extended services, e.g., file sharing, VoIP, or shared whiteboard functionality. IM applications are predominantly used for a spontaneous text-based communication for private purposes.

In addition, there is a potential to use such applications in a business context. In particular, the discussion in this dissertation shows that IM systems can serve as platforms for secure collaborative applications (e.g., electronic contract negotiation, e-payment or electronic voting). On the one hand, such applications have to deal with many challenges, e.g., time constraints (an "instant" communication is desired), the integration of multiple media channels and the absence of one unifying "sphere of control" covering all participants. On the other hand, instant messaging systems provide many advantages, e.g., (i) a spontaneous and flexible usage, (ii) easy distribution of information to many participants and (iii) the availability of different channels for the tasks at hand.

The original intention of these systems (spontaneous free-flowing information exchange), their modular construction, the unsupervised installation and the ability to easily transmit information over a multitude of channels raise many questions and challenges for IT security. For example, one needs to consider how to contain confidential information, how to verify the authenticity of a communication partner, or how to ensure the non-repudiation of statements.

This thesis aims to design security mechanisms that allow to use IM systems as a platform for a collaboration that is *both* (i) spontaneous and flexible as well as (ii) secure, authentic and non-repudiable. Example applications where such collaboration platforms could be used are the electronic negotiation of contracts, electronic payments or electronic voting.

The first part of the dissertation gives an introduction (Section 1) and motivation of the topic (Section 2). It then defines the focused research topic, research questions and the applied methodology (Section 3) as well as the contribution and structure of the thesis (Section 4).

The second part analyses the state of the art and related work. It starts with a summary of the underlying research areas (Section 5). Then, a feature model is used to discuss the state of the art of IM technology, including a categorization of typical components, functionality and architectures of IM systems (Section 6). The next section discusses security goals and mechanism for IM systems (Section 7), which leads to the identification of open challenges. The final sections of this part provide overviews of the related work on security of IM systems (Section 8) and on secure collaborative applications (Section 9).

The third part provides the main contribution of this dissertation. As a starting point, a consideration of the results of the preceding sections leads to a definition and structure of the intended research (Section 10). Then, as a conceptual contribution, two security mechanism are designed: (i) An authentication mechanism for groups, which is based on the mutual recognition of the participants in a video transmission and the protection of the communication channels with an extended Diffie-Hellman (DH) key agreement and (ii) a mechanism for a non-repudiable long-term archival of the results of this collaboration, which is based on a „Separation of Duty" prin-

ciple (Section 11). To evaluate these results and to provide a "proof of concepts" these generic concepts are applied to a more specific case, the electronic voting. As a first step towards this goal, a generic platform for secure collaboration based on IM systems with application-specific plug-ins is designed (Section 12).

The idea is to find a way to support collaboration processes, such as decisions in committees with subsequent voting, even in situations where the participants are distributed across various locations (Section 13). With the intended technical approach (i.e., implemented with IM technologies), the challenge is to provide the same level of security and to support the same process, just like in a conventional approach (i.e., the decision is made on-site in a face-to-face meeting). In principle, in such groups all participants have equal rights. That means that potentially every participant can become the initiator of the decision process. Moreover, every participant should be able to retrace the e-voting method and to reproduce the e-voting results. The developed concepts are realized in a prototypical implementation based on the open source applications Spark (IM client) and Openfire (IM server) (Section 14).

The dissertation concludes with a discussion of the results (Section 15) and an outlook on potential topics for future work (Section 16).

Teil I Einleitung und Problemdefinition

1 Einleitung

Instant-Messaging-Systeme (IM-Systeme, IMS) existieren in der heutigen Form seit Mitte der 1990er Jahre, ausgehend von einer Öffnung des Internets für einen größeren Nutzerkreis außerhalb von Forschungsinstitutionen. Das erste IM-System, welches eine größere Verbreitung fand war ICQ („I seek you“) der Firma Mirabilis. Es handelt sich dabei um ein Netzwerk (strukturiert nach dem Client-Server-Modell), welches den Teilnehmern die Möglichkeit bietet, mit Hilfe eines grafischen ICQ-Clients Nachrichten untereinander, fast synchron, auszutauschen oder in sogenannten Chatrooms textbasierte Unterhaltungen zu führen.

Seitdem wurden diese Systeme in unterschiedliche Richtungen weiterentwickelt. So bieten sie mittlerweile weitere Dienste, wie die Nutzung von Voice over IP (VoIP), Video over IP oder Dateiübertragung (Gross und Koch 2007). Ihre Architektur basiert je nach System auf dem Client-Server-Modell, einem Peer-to-Peer-Modell oder einer Mischform aus beidem. Eine Eigenschaft, die alle IM-Systeme gemeinsam haben und die zu ihrer Verbreitung beigetragen hat, ist die Angabe der Präsenz der Teilnehmer in unterschiedlichen Ausprägungen (Tran, Yang und Raikundalia 2005). Aktuelle Instant-Messaging-Systeme werden vor allem in der privaten unverbindlichen Kommunikation eingesetzt. Nach Aussagen verschiedener Marktstudien (Gartner Group 2007; www.golem.de 2007; Schmidt 2006; BERR und PricewaterhouseCoopers 2008) zeigt sich allerdings, dass Instant Messaging nach und nach zu einem festen Bestandteil der Kommunikationsmedien in der Geschäftswelt wird.

Es existieren zwei Varianten solcher Systeme: interne firmenspezifische IM-Lösungen, die hauptsächlich die Chat-Funktion (textbasierte Nachrichten) implementieren, und öffentlich zugängliche Lösungen (Open Source oder kostenlose Systeme), die aus Werbemitteln finanziert werden) und meistens auch weitere Funktionen wie VoIP oder Dateiaustausch erlauben.

Ein weiterer Unterschied dieser Varianten, besteht in der Art und Weise der Einführung dieser Systeme in den Unternehmensalltag. Während die Einführung der ersten Variante (firmenspezifische Softwarelösungen) eine bewusste Entscheidung der Geschäftsleitung ist, basiert die Einführung der zweiten Variante oft auf der Initiative der Mitarbeiter. Somit folgt die Verbreitung einem Bottom-up-Prinzip, indem einzelne Mitarbeiter eine private Kopie auf dem Arbeitsplatzrechner installieren, um mit Teilnehmern aus dem privaten Umfeld zu kommunizieren oder arbeitsbezogene Informationen mit Kollegen auszutauschen (Cameron und Webster 2005). Diese zweite Variante ist aus Sicht der Forschung und für die vorliegende Arbeit die interessantere Alternative, da sie schon im privaten Bereich in Bezug auf die Funktionalität mächtiger ist und auf einer verteilten Architektur mit unterschiedlichen Administrationsbereichen beruht, wie in den nächsten Kapiteln erläutert wird.

2 Motivation

In der Forschung existieren zahlreiche Veröffentlichungen über die Nutzung von IM-Systemen. Die Studien beschäftigen sich mit einzelnen Kommunikationskanälen (Nardi, Whittaker und Bradner 2000; Herbsleb et al. 2002) oder stellen Vergleiche zwischen den unterschiedlichen angebotenen Kommunikationsmedien an, z. B. in einer Gegenüberstellung von Chat und Audio (Löber, Grimm und Schwabe 2005). Einige wenige Publikationen widmen sich, aus wissenschaftlicher Sicht der Sicherheitsproblematik von IM-Systemen (Borisov, Goldberg und Brewer 2004; Zimmermann, Johnston und Callas 2007; Biondi und Desclaux 2006). Ein Grund für die vergleichsweise geringe Anzahl an Publikationen zum Thema Sicherheit von IM-Systemen könnte in der vorwiegend privaten Nutzung liegen – die transportierten Inhalte werden eventuell als nicht wichtig genug erachtet, um ausführliche Sicherheitsbetrachtungen zu rechtfertigen. Ein weiterer Grund könnte die fehlende Verfügbarkeit von Informationen über die genaue Arbeitsweise solcher Systeme sein, da diese teilweise proprietäre und „Closed Source“ Protokolle verwenden.

Im Hinblick auf die Nutzung von IM-Systemen für eine geschäftliche Kommunikation sind die Betrachtung der Sicherheit solcher Systeme und die Erfüllung eines gesetzten Sicherheitsniveaus sowie die Einhaltung von Policies von großer Bedeutung. Diese Systeme bieten Möglichkeiten für neue flexible Arbeitsformen und dynamische Gruppenstrukturen, denen auf technischer Ebene, d.h. bei der Netzwerkinfrastruktur und den verwendeten Sicherheitsmechanismen, Rechnung getragen werden muss (Stein und Hampe 2008). Diese „Arbeitsformen“ müssen sich nicht nur auf eine einfache Kommunikation zwischen zwei Parteien begrenzen, sondern können unterschiedlichen kollaborativen Szenarien entsprechen.

In diesem Kontext zielt die hier beschriebene Dissertation auf Anwendungen mit hohen Sicherheitsanforderungen, die Instant-Messaging-Systeme als Plattform nutzen. Es soll gezeigt werden, dass IM-Systeme das Potenzial haben, eine spontane, authentifizierte und langfristig verbindliche Kooperation über die Grenzen zentral organisierter Unternehmen hinweg zu realisieren. Dadurch können IM-Systeme genutzt werden, um sicherheitskritische Anwendungen zu realisieren, wie beispielsweise die Durchführung eines Entscheidungsprozesses, einer elektronischen Abstimmung oder einer geschäftlichen Transaktion. Der Hauptvorteil einer solchen Lösung ist die Durchführung sicherer Anwendungen ohne die Nutzung einer großen Kollaborationsplattform und den entsprechenden zeitlichen und finanziellen Aufwand für deren Realisierung (siehe Kapitel 5.3).

Um die Eignung von IM-Systemen für derartige Anwendungsszenarien zu untersuchen, werden hier in einem ersten Schritt die Eigenschaften dieser Systeme identifiziert, um in einem zweiten Schritt die Herausforderungen für das vorliegende Vorhaben und die Anforderungen an eine sichere Kooperation definieren zu können. Im Folgenden werden die charakteristischen Eigenschaften von IM-Systemen beschrieben:

Spontane Nutzung – Die Nutzung eines IMS verlangt, zumindest aus Sicht des Anwenders, keine aufwendige Installation und Konfiguration; nach einer Registrierung mit einer Benutzerkennung kann jeder Teil des IM-Netzes sein.

Zeitliche Anforderungen – Die Kommunikation findet je nach gewähltem Kommunikationskanal in Echtzeit (Video/Audio) oder innerhalb weniger Sekunden (Chat) statt.

Unterschiedliche Kanäle – Ein IMS bietet durch Chat, Audio oder Video vielfältige Alternativen für eine Kommunikation und Kooperation. Allerdings bietet ein solches System durch die größere Vielfalt an technischen Mechanismen und Übertragungsprotokollen auch eine größere Angriffsfläche.

Verteilung von Informationen bei mehreren Teilnehmern – Ein IMS bietet die Möglichkeit, gleichzeitig mehrere Teilnehmer durch die verschiedenen Kommunikationskanäle zu erreichen und Informationen (als Wissen oder auch in Form von Dokumenten) zu verteilen.

Fehlende einheitliche Kontrollbereiche – Je nach Einsatzszenario können sich die Nutzer eines IMS in unterschiedlichen Administrationsbereichen befinden. Dementsprechend ist im Allgemeinen die Durchsetzung eines einheitlichen Sicherheitskonzepts nicht ohne weiteres möglich.

Flexiblere Nutzung unabhängig von Ort (und Zeit) – Durch die Verwendung von IMS als Infrastruktur können Gruppenmitglieder zusammenarbeiten, obwohl sie sich an verschiedenen Orten befinden. Teilweise ist auch eine asynchrone Zusammenarbeit möglich, z. B. wenn Text- oder Sprachnachrichten vom Empfänger erst später gelesen bzw. abgehört werden.

Transport der Kommunikation über offene Rechnernetze – IMS nutzen häufig das weltweit zugängliche Internet. Dieses ist in seiner ursprünglichen Auslegung unsicher, kann aber um zahlreiche Sicherungsmechanismen erweitert werden.

3 Herausforderungen und Ziele

3.1 Problemstellung

Bei der Realisierung einer spontanen und langfristig verbindlichen Kommunikation auf Grundlage des Mediums IMS stellen die oben genannten Eigenschaften eine Herausforderung dar – insbesondere dann, wenn derartige Anwendungen mit einem angemessenen Sicherheitsniveau realisiert werden sollen. Zu den Sicherheitsanforderungen, die an eine IMS-Kooperationsplattform gestellt werden, gehören:

- Authentizität der Gesprächspartner
- Integrität der Kommunikationsinhalte
- Vertraulichkeit einer Gruppenkommunikation
- Verbindlichkeit der Kommunikationsergebnisse

Diese Eigenschaften sollten für jede sichere Anwendung realisiert werden – unabhängig davon, welche konkrete Anwendung ein Instant-Messaging-System als Plattform nutzen soll. Ferner gibt es auch Sicherheitsanforderungen, welche anwendungsspezifisch sind. Dazu gehören beispielsweise die Anonymität, z. B. für die Durchführung von elektronischen Wahlen oder die Pseudonymität, z. B. bei Anwendungen der E-Participation.

Daraus folgen für das vorliegende Forschungsvorhaben Fragestellungen, die im folgenden Kapitel genauer aufgeschlüsselt werden.

3.2 Fragestellung

Die zentrale Fragestellung des hier beschriebenen Forschungsvorhabens lässt sich folgendermaßen zusammenfassen:

> *„Wie kann man die grundlegenden Eigenschaften von Instant-Messaging-Systemen so mit etablierten Verfahren der IT-Sicherheit kombinieren, dass man dadurch erstens eine spontane Kollaboration kurzfristig absichern kann und dass zweitens Ergebnisse dieser spontanen Kollaboration eine langfristige Verbindlichkeit haben?“*

Um diese zentrale Fragestellung zu beantworten, werden Schritt für Schritt die daraus resultierenden Fragen untersucht:

Frage 1: Wie kann man die Absicherung einer spontanen laufenden IM-Kommunikation *ohne* Rückgriff auf eine Public-Key-Infrastruktur (PKI) erreichen und dabei eine authentifizierte, integere und vertrauliche Kommunikation etablieren?

Frage 2: Wie kann man die spontan erzeugten Sicherheitselemente (Schlüssel, Zertifikate) für eine langfristig nicht-abstreitbare, verbindliche Kommunikation nutzen?

Frage 3: Wie sehen algorithmische Lösungen und Protokolle für solche Lösungen aus?

Frage 4: Wie sieht eine Plattform aus, die solche Lösungen realisiert?

Frage 5: Wie werden externe Anwendungen in eine solche Plattform integriert?

Frage 6: Wie zeigt man, dass das gewünschte Sicherheitsniveau erreicht wurde?

Darüber hinaus gibt es weitere Fragen, die in weiterführenden Arbeiten behandelt werden sollten (Teilaspekte dieser Fragen werden ggfs. auch im vorliegenden Forschungsvorhaben bearbeitet):

Frage 7: In welcher Weise und bis zu welchem Grad ist eine (nachträgliche) Anpassung einer konkret existierenden Anwendung auf eine IMS-Umgebung möglich?

Frage 8: Welche Änderungen im Prozessablauf des zugrunde liegenden Szenarios sind notwendig, um die Verwendung der geeigneten Sicherheitsmechanismen zu ermöglichen?

Frage 9: Welche der Kommunikationskanäle eines IMS sind für die jeweilige Anwendung geeignet und mit welchem Aufwand können sie adaptiert werden?

Frage 10: Wie beeinflussen die Eigenschaften der unterschiedlichen Kommunikationskanäle die Sicherheit des Gesamtsystems?

Frage 11: Wie kann formal nachgewiesen werden, dass die eingesetzten Mechanismen das angestrebte Sicherheitsziel auch gewährleisten (Konsistenz und Vollständigkeit)?

3.3 Methode

Das Forschungsvorhaben dieser Arbeit, die Realisierung einer sicheren, authentifizierten, zertifizierten und verbindlichen Kooperation auf Basis eines IM-Systems, kann schwerpunktmäßig der Forschungsmethode „Design Research“ (auch Design Science) zugeordnet werden. Grundsätzlich besteht die Vorgehensweise dieser Methode aus folgenden Schritten (Vaishnavi und Kuechler 2004/05; Hevner et al. 2004):

- *Erkennen eines Problems (Awareness of Problem):* Als Ausgangspunkt für die weiteren Bestrebungen wird ein Problem erkannt und näher analysiert. Dabei können beispielsweise Forschungsberichte oder Industrieentwicklungen einfließen, um neue Erkenntnisse zu gewinnen. Das Ergebnis dieser Phase ist die Definition des Forschungsproblems.
- *Lösungsvorschlag (Suggestion):* Dies ist die kreative Phase dieser Forschungsmethode, in der ein Lösungskonzept passend zum oben definierten Forschungsproblem skizziert wird.
- *Entwicklung eines Artefakts (Development):* Mit diesem Artefakt wird der Lösungsvorschlag in greifbare Ergebnisse umgesetzt. Ein Artefakt ist, zumindest im Kontext des Design Researchs, ein Konstrukt, ein Modell, eine Methode oder eine entsprechende Instanz in einer bestimmten Umgebung. Diese Phase führt auch zu einem besseren Verständnis des Problems und kann daher Auswirkungen auf die vorhergehende Phase „Erkennen eines Problems“ haben. So kann man z. B. zu der Erkenntnis gelangen, dass bei der Entwicklung eines Prototyps (als Artefakt) nicht alle Annahmen oder gewünschte Eigenschaften realisierbar sind. In diesem Fall muss das Vorhaben (d. h die Problemdefinition) unter Umständen revidiert werden.
- *Bewertung des Artefakts (Evaluation):* In dieser Phase wird das Artefakt entsprechend der Ziele, Annahmen oder Hypothesen (aus der Definition des Forschungsproblems) evaluiert. Die hier entstandenen Ergebnisse können entweder in die weitere Durchführung des gleichen Vorhabens (über die Phase des Erkennens des Problems) in den Lösungsvorschlag und das Artefakt einfließen oder separat für die weitere Forschung dokumentiert werden.
- *Abschließendes Fazit (Conclusion):* In diesem Schritt werden sowohl die Ergebnisse des aktuellen Forschungsvorhabens zusammengefasst als auch seine Grenzen und möglichen Erweiterungen sowie offenen Fragen für die weitere Forschung dokumentiert.

Die Methode des Design Researchs hat ihren Ursprung in den Ingenieurwissenschaften (Simon 1996). Die Adaption und Anwendung dieser Methode in der Wirtschaftsinformatik war u.a. eine Antwort auf die langjährige Diskussion über Relevanz und wissenschaftlichen Anspruch der Wirtschaftsinformatik und ihrer Methoden. Eine große Rolle bei der Adaption und Verbreitung spielte der Artikel von (Hevner et al. 2004). Dieser Artikel wurde unter Wirtschaftswissenschaftlern mit der Begründung kritisiert, dass Hevner anstrebt, den Forschungsbereich der Wirtschaftsinformatik als eigenständige Disziplin mit eigenen Methoden und Theorien zu etablieren und dass das Design Research an sich eine unzureichende theoretische Basis hat. Außerdem wurde angemerkt, dass die designwissenschaftlichen Forschungsergebnisse durch den schnellen Fortschritt der Entwicklung leicht vergänglich sein können. Mehr Details zu dieser Thematik finden sich in (Lehner und Zelewski 2007; Frank 2006).

Eine andere Sicht auf derartige Forschungsmethoden präsentiert (Frank 2006). Er schlägt ein allgemeines abstraktes Framework für die Nutzung von Forschungsmethoden in der Wirtschaftsinformatik sowie eine konkrete Instanziierung für die konstruktionsorientierte Forschung vor. Nach seiner Ansicht sollten, um die Wissenschaftlichkeit dieser Methode zu unterstützen, alle Annahmen, Anforderungen, Entwurfsentscheidungen sowie Evaluationsergebnisse, die in die Konstruktion einfließen, begründet werden – anhand von geeigneten Theorien, Stellungnahmen aus der Literatur und einer vergleichenden Bewertung möglicher Alternativen. Dies bedeutet, dass für die oben aufgelisteten Schritte und Vorgehensweise des Design Researchs

unterschiedliche Methoden (z. B. Literaturreviews, hermeneutische oder behavioristische Ansätze usw.) verwendet werden sollten. Für die vorliegende Arbeit wird dies im folgenden Kapitel skizziert.

4 Eigener Beitrag und Struktur der Arbeit

Das resultierende Konstrukt dieser Arbeit – als Artefakt im Sinne des Design Researchs – ist ein algorithmisches und technisches Konzept für eine spontane, sichere und verbindliche Gruppenkollaboration auf Grundlage von IM-Systemen. Es wird nicht nur die Authentizität und Vertraulichkeit der Kommunikation mit entsprechenden Sicherheitsmechanismen gewährleistet, sondern es wird auch eine Lösung vorgeschlagen, um das Ergebnis dieser Kommunikation langfristig nachvollziehbar zu speichern.

Diese Konzepte sind *nicht* auf die übliche IMS-Kommunikation (Chat, VoIP) begrenzt. Das Ziel der Arbeit ist nicht die Absicherung der IM-Systeme, sondern die verteilte IM-Infrastruktur zu nutzen, um sicherheitskritische kollaborative Anwendungen wie elektronische Transaktionen oder elektronische Entscheidungsprozesse zu realisieren. Um diese Eignung darzustellen, werden die empfohlenen Konzepte mit Hilfe einer IMS-Plattform und der beispielhaften Adaption und Realisierung einer Anwendung verfolgt.

Der konkrete Lösungsansatz gliedert sich in vier Schritte (vgl. Abbildung 1):

1. Analyse von Instant-Messaging-Systemen und existierenden IT-Sicherheitsmechanismen (vgl. Kapitel 6 und Kapitel 7)
2. Entwicklung von algorithmischen Lösungen für eine authentifizierte, verbindliche Kooperation (vgl. Kapitel 11)
3. Vorstellung der konzipierten Plattform (vgl. Kapitel 12)
4. Exemplarische Anwendung: Entscheidungsprozesse und Wahlen (vgl. Kapitel 13)

Als **erster Schritt** zur Beantwortung der Fragestellungen (vgl. Kapitel 6) wird zunächst eine Analyse und konzeptionelle Strukturierung des Themengebietes „Instant-Messaging-Systeme“ vorgenommen. Dies geschieht mit Hilfe eines **Featuremodells**, das typische Charakteristika von IM-Systemen aus *funktionaler* Sicht beschreibt, sowie einer Systemarchitektur, die den *strukturellen* Aufbau solcher Systeme darstellt. Weiterhin wird ein Überblick über die Thematik der IT-Sicherheit gegeben, in dem eine **Sicherheitsanalyse** von Instant-Messaging-Systemen (Kapitel 7) vorgenommen wird und gewünschte Sicherheitsanforderungen möglichen Sicherheitsmechanismen gegenüber gestellt werden.

Im **zweiten Schritt** wird eine Modellanwendung vorgestellt, welche die verteilte Kooperation darstellt. Die Modellanwendung umfasst eine Gruppe von n Teilnehmern ($n > 2$) in einer Kooperation über das IM-System und seine unterschiedlichen Kanäle. Diese Gruppe führt einen Entscheidungsprozess oder eine geschäftliche Vertragsverhandlung durch (Kapitel 11.2). Um eine **spontane Zertifizierung** innerhalb einer solchen Sitzung gewährleisten zu können, wird im Rahmen des hier beschriebenen Forschungsvorhabens ein **neuer Zugangsberechtigungsmechanismus** für den Eintritt in die Anwendungsgruppe vorgestellt (Kapitel 11.1.1, 11.1.2). Dieser orientiert sich an Verfahren zur Feststellung der Authentizität bei Präsenz-

Gruppenanwendungen ohne die Nutzung von IT (z. B. die Durchführung von Entscheidungsprozessen in einem Standort), durch die Wiedererkennung seitens der anderen Mitglieder des Gremiums. Dieses „Wiedererkennen“ wird durch eine technische Lösung nachgebildet, die zwei Mechanismen miteinander kombiniert: (1) eine Absicherung des Übertragungskanals mit Hilfe des Diffie-Hellman-Verfahrens (Diffie, van Oorschot und Wiener 1992) und (2) die gegenseitige Wiedererkennung der Teilnehmer im Videobild. Durch dieses Verfahren soll die Authentizität, Vertraulichkeit und Integrität der Kooperation erreicht werden. Um der Verbindlichkeit der Aussagen und Kooperation innerhalb einer Sitzung Rechnung zu tragen, wird darüber hinaus ein weiterer Lösungsansatz vorgestellt, welcher nicht nur die Nicht-Abstreitbarkeit der Aussagen innerhalb der Gruppe erfüllt, sondern auch die langfristige Verbindlichkeit Dritten gegenüber sicherstellt (Grimm 1994). Dabei wird die Verwendung von Schlüsseln und Signaturen *während* der Kommunikation durch ein Verfahren zur **Langzeitarchivierung** des Ergebnisses *nach* der Kommunikation ergänzt (Kapitel 11.12). Dadurch soll ermöglicht werden, zu einem späteren Zeitpunkt die Richtigkeit und Gültigkeit des Ergebnisses nachzuweisen. Der Mechanismus basiert auf einer kooperativen Aufgabenteilung („Separation of Duty“) zwischen einem Signierer und den anderen Mitgliedern des Gremiums, welche die signierten Aussagen aufbewahren. Diese Aufgabentrennung liefert den entscheidenden Baustein zur langfristigen Nicht-Abstreitbarkeit. Durch einen geeigneten Einführungsprozess können dabei sogar Personen mit einbezogen werden, die nicht allen Gruppenmitgliedern vorab bekannt waren.

Zur Umsetzung der definierten Sicherheitsziele wird im **dritten Schritt** eine generische IMS-Plattform definiert, die eine sichere, zertifizierte und langfristig verbindliche Kommunikation sicherstellt. Diese Plattform entsteht durch Erweiterung von Instant-Messaging-Systemen durch weitere Systemkomponenten einschließlich der Schnittstellen und weiterer Sicherheitsmechanismen, abhängig von der externen Anwendung, die auf einer IMS-Architektur adaptiert werden soll (Kapitel 12).

Um die Konzepte aus dem vorangegangenen Schritt weiter auszuarbeiten und zu überprüfen, wird in einem **vierten Schritt** die vorgestellte generische Plattform für einen konkreten Anwendungsfall präzisiert (Kapitel 13). Als Beispiel werden dazu Entscheidungsprozesse innerhalb von kleinen Gruppen ausgewählt. Solche Entscheidungsprozesse treten in verschiedenen Varianten auf und enthalten, je nach Fall, eine Diskussions- und eine Abstimmungsphase.

- Die Diskussion findet immer als eine offene (namentliche) Diskussion statt. Dies kann weitgehend durch übliche Mittel eines IMS realisiert werden, durch Kommunikation über Chat oder Audio-Video. Soll das Ergebnis der Diskussion in strukturierter Form festgehalten werden, zum Beispiel als Grundlage für eine nachfolgende Abstimmung, müssen hierfür unter Umständen geeignete Mechanismen ergänzt werden.
- Der Abstimmungsprozess dagegen, kann je nach Variante, sowohl unter Angabe der Identität (namentlich) als auch geheim ablaufen. Hierzu werden die vorher diskutierten generischen Sicherheitsmechanismen für die Anwendung des elektronischen Wählens spezifiziert.

Das Beispiel von Entscheidungsprozessen in kleinen Gruppen ist deswegen ausgewählt worden, da es in der offenen Diskussion den typisch unverbindlichen und spontanen Anteil von Kommunikation enthält, den IMS bereits jetzt gut unterstützen, und diesen verknüpft mit einer Ab-

stimmung als einem hochsensiblen und verbindlichen Kommunikationsanteil, der von IMS bisher nicht unterstützt wird. Beide Anteile sollen reibungslos ineinander spielen. Darin liegt eine innovative Anwendungsperspektive der vorliegenden Forschungsarbeit. Solche Entscheidungsprozesse finden bei herkömmlicher Vorgehensweise (d. h. ohne IMS) vor Ort statt, z. B. im Bereich von universitären Gremien oder im Programmkomitee einer Konferenz. Sie unterliegen bestimmten Regelungen, welche Auswirkungen auf den Ablauf und auf die Sicherheitsanforderungen dieses Vorgangs haben. Ein Unterprozess der vorgestellten Szenarien ist das eigentliche E-Voting. Es handelt sich dabei um elektronisches Wählen innerhalb von kleinen Gruppen, bei denen eine zentrale Instanz – etwa in Form einer Wahlkommission – wegen des unnötigen bürokratischen Zusatzaufwandes hinderlich wäre. Um eine dezentrale Organisation des Vorganges zu ermöglichen, ist es daher wünschenswert, dass jeder Teilnehmer das Verfahren nachvollziehen und das Ergebnis selbst ermitteln kann. Dazu können Wahlprotokolle ohne zentrale Instanz verwendet werden, die auf Secret-Sharing (Benaloh und Yung 1986) oder Vermischen von Voten (DeMillo, Lynch und Merritt 1982) basieren. Um die Übereinstimmung zwischen IMS (als Anwendungsplattform) und Entscheidungsprozessen (als Anwendung) zu erreichen, wird die vorher vorgestellte Plattform verwendet und für Realisierung für von Entscheidungsprozessen mit Diskussion und E-Voting adaptiert. Das entwickelte System (Kapitel 14) ist auf Grundlage des IM-Clients „Spark“ und des entsprechenden Servers „Openfire“ realisiert. Diese wurden ausgewählt, weil sie als Open-Source-Projekte verfügbar sind, standardisierte Protokolle (XMPP/Jabber) verwenden und Mechanismen zur Erweiterung anbieten.

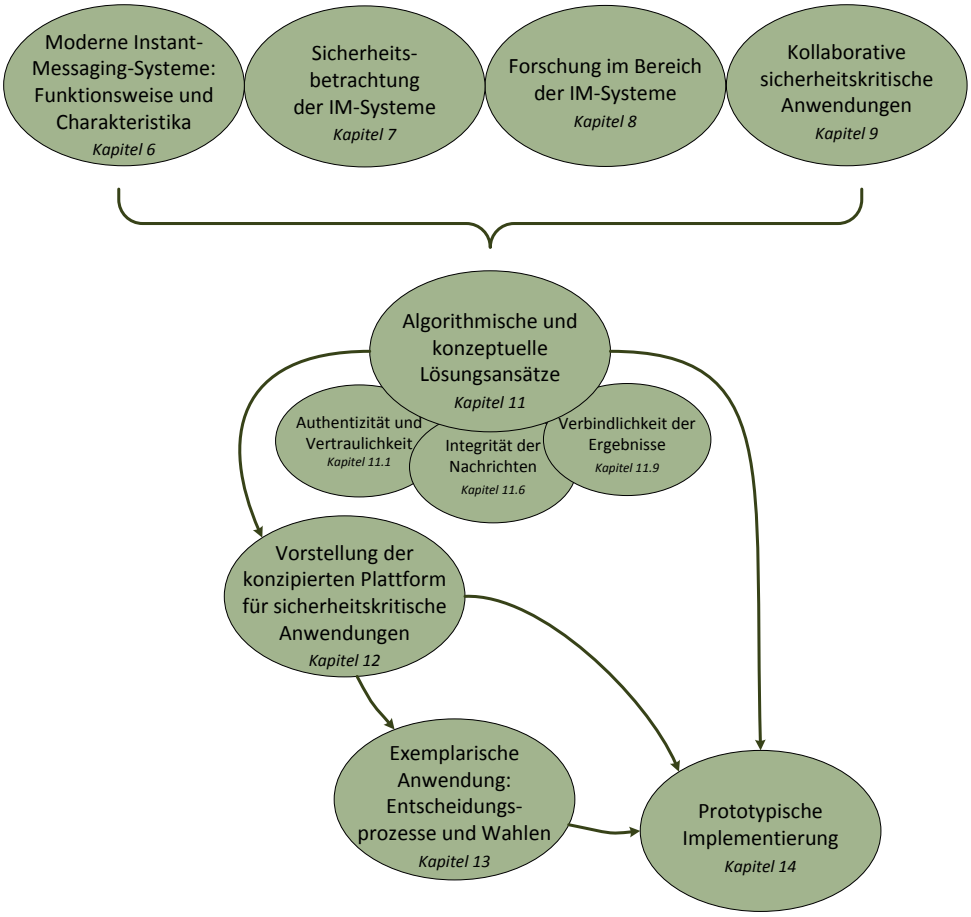

Abbildung 1 – Struktur der Arbeit

Teil II Stand der Forschung und Related Work

5 Begriffsklärung

Betrachtet man den Forschungsbereich der asynchronen und synchronen Kommunikation (auch semi-synchron genannt) und Kooperation (Avrahami, Fussell und Hudson 2008)) mit dem Ziel, Werkzeuge und Plattformen zur Verfügung zu stellen, die eine kooperative Zusammenarbeit unabhängig von Ort und Zeit erlauben, trifft man auf Begriffe wie Unified Communication (UC), Echtzeitkommunikationssysteme (Real Time Communication, RTC) oder CSCW (Computer Supported Cooperative Work). Unter diesen Begriffen wird die umfangreiche Thematik der Interaktion von Menschen innerhalb von Gruppen einschließlich der Unterstützung dieser Interaktion durch geeignete informationstechnische Werkzeuge aus unterschiedlichen Blickpunkten betrachtet. In den nächsten Kapiteln werden diese Begriffe kurz erläutert und eingeordnet.

5.1 Computer Supported Cooperative Work

Computer Supported Cooperative Work (CSCW) ist ein interdisziplinäres Forschungsgebiet, das sich mit der IT-Unterstützung von Gruppenzusammenarbeit beschäftigt. Dabei wird der Einfluss von technischen, sozialen und ökonomischen Aspekten auf die Gestaltung und Durchführung einer Gruppenzusammenarbeit betrachtet. Wegen dieser zahlreichen Perspektiven, gibt es auch zur genauen Abgrenzung und Definition des Begriffs „CSCW“ viele unterschiedliche Varianten. Es werden je nach Blickwinkel unterschiedliche Aspekte betont. Einen Überblick über die möglichen Ausprägungen geben (Gross und Koch 2007).

Da für die vorliegende Arbeit vor allem die technischen Aspekte einer Zusammenarbeit von Bedeutung sind, wird an dieser Stelle die Definition von (Ellis, Gibbs und Rein 1991) übernommen: *“CSCW looks at how groups work and seeks to discover how technology (especially computers) can help them work”*. Ellis et al. haben auch den Begriff „Groupware“ geprägt. Demnach handelt es sich bei Groupware um *“computer-based systems that support groups of people engaged in a common task (or goal) and that provide an interface to a shared environment”*. Einen Überblick über die im Markt existierenden Groupwaresysteme geben (Riemer, Arendt und Wulf 2005). Der Unterschied von Groupware im Vergleich zu anderer Software ist demnach, dass damit versucht wird, die Isolation der Benutzer voneinander zu reduzieren und das Bewusstsein zu erhöhen, dass jeder ein Teil einer Gruppe ist. Weiterhin sollen solche Systeme anpassbar sein, sodass Benutzer und Gruppen sie auf unterschiedliche Art und Weise nutzen können (Koch 2009).

Groupware-Systeme lassen sich unterschiedlich klassifizieren. (Gross und Koch 2007) fassen die verschiedenen Klassifikationen folgendermaßen zusammen:

- *Raum – Zeit:* Die Unterscheidung der Systeme ist abhängig von Ort und Zeit der Interaktion zwischen den Teilnehmern (Johansen 1991). So gehören zum Beispiel Gruppenmoderationssysteme, die dann zum Einsatz kommen, wenn sich alle Teilnehmer am gleichen Ort befinden, in die Kategorie „gleicher Ort – gleiche Zeit". Demgegenüber stehen Videokonferenz-Systeme oder Instant-Messaging-Systeme, die eine Kommunikation zwar zur gleichen Zeit, aber von unterschiedlichen Orten aus ermöglichen. Diese gehören somit in die Kategorie „verschiedene Orte – gleiche Zeit".
- *Personen – Artefakt:* Bei dieser Klassifikation geht es darum, die Art der Beziehung zwischen den Teilnehmern oder zwischen Teilnehmer und System zu beschreiben (Dix et al. 1993). Die unterschiedlichen Beziehungen können folgende Formen haben: die direkte Kommunikation, das Verstehen, die Rückmeldung und die Steuerung. Nach dieser Klassifizierung gehören zum Beispiel Sitzungs- und Entscheidungssysteme in die Kategorie des gemeinsamen Verständnisses, wohingegen Instant-Messaging-Systeme der direkten Kommunikation zuzuordnen wären.
- *3K-Model:* Bei dieser Kategorisierung wird die Interaktion der Teilnehmer in Kommunikation (Verständnis), Koordination (Aufgabenteilung) oder Kooperation (gemeinsames Ziel) unterteilt (Teufel et al. 1995). Die zuvor genannten Sitzungs- und Entscheidungssysteme gehören nach dieser Kategorisierung in den Bereich der Kooperationsunterstützung, Konferenzsysteme werden der Kommunikationsunterstützung zugeordnet und Workflowsysteme der Koordinationsunterstützung.
- *Anwendungsorientiert:* Ein letzter Klassifizierungsvorschlag kommt von (Ellis, Gibbs und Rein 1991), sie klassieren die Systeme direkt nach der Art der Anwendung in Nachrichtensysteme, Gruppeneditoren, elektronische Besprechungszimmer, Computerkonferenzen, Informationsräume, Intelligente Agenten und Koordinationssysteme. Instant-Messaging-Systeme gehören hier in die Gruppe der Nachrichtensysteme.

Innerhalb dieser Klassifikationsgruppen gibt es weitere Aspekte, die eine detaillierte Unterteilung erlauben. Dazu gehören (Steinmetz 2000):

- *Zeit:* Danach wird zwischen einer synchronen (z. B. Instant Messaging) und asynchronen (z. B. E-Mail) Kommunikation/Kooperation unterschieden.
- *Benutzergruppen:* Hier spielt einerseits die Anzahl der Teilnehmer eine Rolle (findet die Kommunikation zwischen zwei oder mehreren Teilnehmern statt?) und andererseits die Art und Weise, wie die Gruppe gebildet wird: In einer statischen Gruppe stehen die Identität und die Anzahl der Teilnehmer vorher fest, während sich in einer dynamischen Gruppe beides ändern kann. Weiterhin können innerhalb einer Gruppe unterschiedliche Rollen existieren. Man kann z. B. unterscheiden zwischen einem einfachen Mitglied einer Gruppe und einem Initiator einer Gruppensitzung, einem Verwalter eines Tokens (für die Steuerung einer Sitzung) und einem Beobachter.
- *Steuerung:* Die Steuerung und Kooperation einer Sitzung kann zentralisiert oder verteilt sein. Bei der zentralisierten Steuerung übernimmt ein Koordinator Aufgaben wie die Festlegung von Anforderungen, die Formulierung von Anfragen oder das Versenden von Berichten. Bei der verteilten Steuerung übernimmt jeder Teilnehmer einen Teil solcher Aufgaben.

5.2 Unified Communication

Der Unified-Communication-Ansatz (UC) zielt auf eine Integration von asynchronen und synchronen Kommunikationsmedien (wie Telefon, Fax, E-Mail oder VoIP) samt den dazu notwendigen Geräten in einer einheitlichen Umgebung. Die Motivation dazu war der hohe Aufwand, der notwendig ist, um den richtigen Kooperationspartner im Rahmen einer verteilten Zusammenarbeit zu erreichen. Dieser (unnötig) hohe Aufwand wurde durch die Nutzung von unterschiedlichen Geräten (Telefon, Laptop, PDA, Arbeitsrechner) mit unterschiedlichen Kennungen (Konten, Profile, Telefonnummern) verursacht. Unified-Communication-Systeme sollen die Anwender bei der Verwaltung von Medien und Geräten unterstützen, indem sie mit Hilfe eines Profils (identifiziert z. B. durch eine Telefonnummer) und eines vordefinierten Regelwerks (z. B. Zeitpunkt, Kontaktperson, Gerät) den Verbindungsaufbau mit dem gewünschten Kontaktes automatisieren (Riemer, Arendt und Wulf 2005; Riemer 2007; Elliot 2008). Eine Weiterentwicklung führte zu einer Integration von UC-Funktionen in unternehmensspezifische Applikationen wie ERP- oder CRM-Systeme. Durch diese Medienintegration soll nicht nur die Erreichbarkeit von Mitarbeitern, sondern auch der Informationsfluss und der Zugriff auf gemeinsam genutzte Daten und somit die gesamte Kollaboration erleichtert werden (Picot, Riemer und Taing 2008; Burton et al. 2007; Elliot 2008). IM-Systeme können in diesem Zusammenhang eine Medienvariante sein.

5.3 Real Time Communication/Collaboration

Dem Akronym RTC werden in der Literatur zwei Bedeutungen zugeordnet: erstens RTC als Real Time Communication und zweitens RTC als Real Time Collaboration. Die unterschiedlichen Bedeutungen hängen von der Zeit der Entstehung des Begriffes, dem Betrachtungsbereich der Autoren und der erfolgten Erweiterung der Systeme mit verschiedenen Kommunikationskanälen ab. Dabei geht es teilweise darum ganze Unternehmenslösungen wie zum Beispiel OpenScape von Siemens (Siemens Enterprise Communications Inc. 2009) oder One-X von Avaya (Avaya 2009) bereitzustellen, welche auf einer unternehmensweiten Client-Server-Architektur basieren. Teilweise wird auch versucht, die Medienintegration eines Unified-Communication-Ansatzes (siehe Kapitel 5.2) um Präsenzinformationen zu erweitern (Riemer 2007; Riemer, Frößler und Klein 2007; Lazar 2006). Somit haben RTC-Produkte folgende Eigenschaften (vgl. Abbildung 2):

- *Unified-Communication-Merkmale*, wie die Bereitstellung und Integration von unterschiedlichen Kommunikationskanälen
- *Präsenzangaben*, die abhängig vom Nutzer angeben, wann oder wie er erreicht werden möchte
- *Kontext-sensitive Informationen*, die eine Echtzeitkommunikation abhängig von den zugrundeliegenden (Unternehmens-)Prozessen erlauben
- *Groupware-Funktionen*, die eine allgemeine Echtzeitkooperation unterstützen

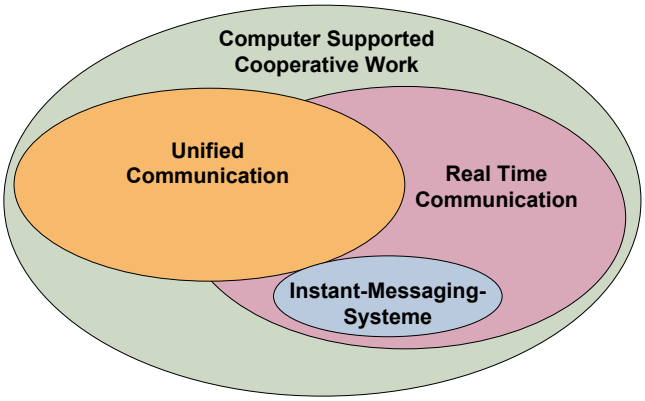

Abbildung 2 – Begriffsabgrenzung

Bei Betrachtung der oben genannten Eigenschaften, der bereitgestellten Funktionalität und den Erweiterungsmöglichkeiten können Instant-Messaging-Systeme auch als RTC-Lösungen gesehen werden (Riemer und Frößler 2006). Es handelt sich dabei in der Regel nicht um firmeneigene Entwicklungen, sondern um offene Anwendungen, die von einem Softwarehersteller oder einer Community zum Download angeboten werden, z. B. Skype oder ICQ (Skype 2008; ICQ 2008). Der größte Unterschied zu üblichen RTC-Anwendungen ist in der fehlenden internen Kontrolle, Installation und Wartung der Systeme zu sehen (Perey 2004; Hollis 2004). Die daraus resultierende Sicherheitsproblematik ist auch einer der Gründe für den geringen „offiziellen“ Einsatz von IM-Systemen innerhalb von Unternehmen (Schulze 2007; Schmidt 2006).

5.4 Instant-Messaging-Systeme

Laut eines Berichtes des Marktforschungsinstituts Gartner Inc. werden Instant-Messaging-Systeme bis zum Jahr 2011 eines der wichtigsten Werkzeuge für die Kommunikation innerhalb von Unternehmen sein. Bis 2013 sollen ca. 95% der Mitarbeiter in Großunternehmen IM-Systeme für die Echtzeitkommunikation und Unterstützung von Geschäftsprozessen verwenden (Gartner Group 2007).

Instant Messaging (IM) *in der ursprünglichen Form*, ist ein Kommunikationsdienst, welcher den Benutzer befähigt, einen Chatroom für zwei oder mehr Personen zu öffnen, in dem in Echtzeit miteinander kommuniziert werden. In der Regel geschieht dies über Textnachrichten, jedoch wird der Begriff „chatten“ mittlerweile auch auf Video- und Audio-Kommunikation angewandt. Die Verbindung zwischen den Kommunikationsteilnehmern wird über das Internet oder ein lokales Netzwerk hergestellt. Voraussetzung dafür ist eine Softwareanwendung, ein Instant Messenger, welcher das jeweilige IM-Protokoll unterstützt. Ein Instant Messenger meldet neue Ereignisse, wie etwa die Neuverbindung, den Verbindungsabbruch eines anderen Benutzers im gleichen Chatroom oder neue private Nachrichten über Textmeldungen in einem Protokoll (Chat Log), Pop-up-Fenster oder mit akustischen Signalen. Es ist auch möglich, mit anderen Benutzern neue Chat-Sessions außerhalb des Chatrooms zu eröffnen, um privat zu kommunizieren.

Als erstes Tool für Instant Messaging gilt das Unix-Programm „talk“, welches in den 1970er Jahren den textuellen Nachrichtenaustausch zwischen zwei Benutzern verschiedener Computer (Peer-to-Peer) in einem Netzwerk erlaubte. 1988 wurde dann ein Client-Server-Internetdienst

mit dem Namen „Internet Relay Chat“ (IRC) erfunden und eingeführt. Um mehreren Benutzern die Möglichkeit anzubieten, untereinander zu kommunizieren, finden IRC-Gespräche in virtuellen themenbezogenen Konferenzräumen (Channels) statt. Dabei können mehrere Gruppen getrennt kommunizieren und ein Benutzer kann in verschiedenen Channels gleichzeitig angemeldet sein. Es wird zwischen öffentlichen, moderierten und sogenannten „invite-only“-Channels unterschieden. Meistens wird über ein festgelegtes Topic (Thema), welches jedem Benutzer beim Betreten des Channels automatisch mitgeteilt wird, diskutiert. Weiterhin können IRC-Server miteinander vernetzt werden, um noch mehr Benutzer auf einmal zu verbinden. Diese bilden diverse Netzwerke (z. B. IRCnet, DALnet3), die teilweise, auch heute noch existieren (Schweitzer 2002; Dittmann 2002).

Der nächste Meilenstein des Instant-Messagings war 1996 die Entwicklung der Softwareanwendung der Firma Mirabilis bekannt als ICQ (I seek you) (ICQ 2008). Dies bot im Vergleich zu den bisherigen Entwicklungen eine übersichtliche GUI-Oberfläche (Graphical User Interface), was zur Verbreitung des Tools beigetragen hat. Später wurde ICQ und das dazugehörige Protokoll OSCAR (Open System for Communication in Realtime) von AOL aufgekauft, die darauf basierend einen eigenen Dienst namens AOL Instant Messenger (AIM) entwickelten. Der ICQ-Dienst wurde weiterhin angeboten und gleichzeitig die Kompatibilität der Nachrichten zwischen beiden Systemen realisiert (AOL 2009). Seitdem entwickelten weitere Anbieter proprietäre oder offene Protokolle und entsprechende Clients. Die bekanntesten Vertreter sind Microsofts Live Messenger (Microsoft 2009b), Yahoo! Messenger (Yahoo! 2009), Google Talk (Google 2009), Skype (Skype 2008) und Jabber (Jabber.org 2009). Die meisten dieser Systeme erlauben durch die proprietäre Entwicklung eine Kommunikation nur zwischen Clients des gleichen IM-Netzes. Diese Nische füllen sogenannte Multiprotokoll-IM-Systeme (auch Meta-Messenger-genannt) und erlauben ferner die Nutzung und Verwaltung von Accounts unterschiedlicher Systeme. Aktuelle Beispiele der Multiprotokoll-Clients sind Pidgin (Pidgin 2009), Trillian (Trillian-Messenger.net 2009) und Miranda (Miranda 2009).

Typische Funktionen solcher Systeme sind das An- und Abmelden der User, das Versenden von Nachrichten, die Angabe der eigenen Verfügbarkeit (z. B. Online Status) und, je nach Mächtigkeit der Tools, Audio-Video-Telefonie oder Filesharing (vgl. auch Kapitel Featuremodell 6.1). In diesem Kontext existiert eine Vielzahl an Protokollen für die textbasierte oder Audio/Video-Kommunikation. In den nächsten Kapiteln wird daher ein Überblick über existierende Architekturen und typische Funktionen von IM-Systemen gegeben. Zur Darstellung werden dabei so genannte Featuremodelle verwendet.

6 Moderne Instant-Messaging-Systeme: Funktionsweise und Charakteristika

Betrachtet man unterschiedliche IM-Systeme, haben alle eine ähnliche Funktionsweise: Der Benutzer muss als Erstes das gewünschte Werkzeug installieren (das ist sein IM-Client) und sich an dem Authentifikationsserver registrieren. Es wird ein entsprechendes Profil angelegt, welches aus einer Kennung und einem Passwort besteht, den Login-Daten für die Nutzung des Instant-Messaging-Systems. Diese Daten werden durch den Authentifikationsserver des IM-Netzes gespeichert. Da sowohl die Bezeichnung dieses zentralen Servers als auch die Anzahl

der Server, die für die Authentifikation der Teilnehmer zuständig ist, bei den unterschiedlichen Realisierungen variiert, wird dieser in der Abbildung 3 dieser als Instant-Messaging-Server dargestellt. Ein Profil kann auch weitere personenbezogene Daten enthalten, wie Wohnort, Geschlecht, Geburtsdatum, usw. Ist der Benutzer einmal angemeldet, kann er andere schon registrierte Benutzer kontaktieren. Dafür werden bei der Anmeldung die Verbindungsinformationen (z. B. die IP-Adresse und die dem Client lokal zugewiesene TCP-Portnummer) sowie die Kontaktliste (Freunde, Buddies des Benutzers) an den sogenannten Präsenzserver übermittelt.

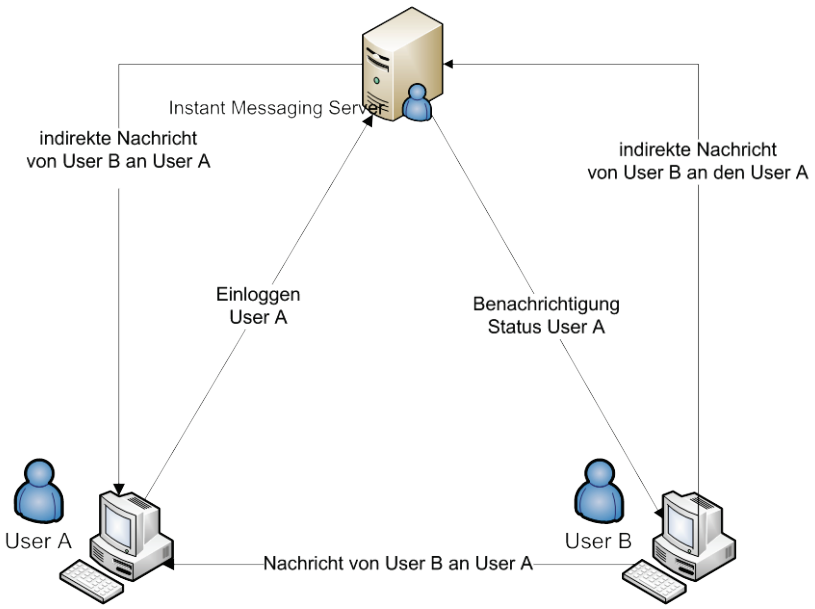

Abbildung 3 – Kommunikationsaufbau im IM-Netz

Dieser ist je nach Implementation ein Teil des IM-Servers oder ein separater Server. Der Präsenzserver überprüft, ob Benutzer, die in der Kontaktliste eingetragen sind, angemeldet sind. Je nach Ergebnis sendet der Server die entsprechenden Informationen an den angemeldeten Benutzer zurück. Ist einer der gewünschten Kontakte „online“, kann durch das Auswählen der entsprechenden Kennung eine Verbindung aufgebaut werden und die Beteiligten können Nachrichten austauschen. Um dies zu ermöglichen, erhält der Sender (der Anfrager einer Verbindung) die IP-Adresse und Portnummer des Empfängers (gewünschter Kommunikationspartner) vom Server übermittelt. Die Nachrichten werden direkt zwischen den Clients übertragen oder über den Server geroutet. Meldet sich ein Benutzer ab, erfolgt über den Präsenzserver eine Benachrichtigung der angeschlossenen Clients.

Der genaue Kommunikationspfad für die Übermittlung der Nachrichten ist von System zu System unterschiedlich. Ausschlaggebend dafür sind das verwendete Protokoll und die entsprechende Architektur. Der Nachrichtenaustausch kann zentral über den Server (vgl. Abbildung 3, indirekte Nachricht) oder nach dem Peer-to-Peer-Prinzip erfolgen (direkte Nachricht).

Im Folgenden wird ein detaillierter Überblick über die Bestandteile und Funktionsweise eines IM-Netzes gegeben.

6.1 Featuremodell von Instant-Messaging-Systemen

Durch die Realisierung unterschiedlicher Protokolle sowie die verschiedenen Entstehungsgeschichten und Schwerpunkte gibt es ein breites Spektrum von Funktionalitäten bei Instant-Messaging-Systemen. Um in den späteren Kapiteln dieser Arbeit Aussagen über die Sicherheit und Erweiterbarkeit treffen sowie Konzepte zur Realisierung einer spontanen, sicheren Kooperation vorschlagen zu können, werden an dieser Stelle mit Hilfe von Modellierungsansätzen technische Aspekte von IM-Systemen vorgestellt und strukturiert. Dabei wird zugunsten einer Gesamtübersicht von protokollspezifischen Charakteristika und Implementierungsdetails abstrahiert.

Eine Disziplin, die sich mit der Modellierung und dem Entwurf von Familien ähnlicher Systeme in einer Anwendungsdomäne beschäftigt, ist das *Domain Engineering*. In diesem Kontext gibt es unterschiedliche Methoden wie FODA (Feature Oriented Domain Analysis) und dessen Erweiterungen wie FOPLE (Feature Oriented Product Line Software Engineering) oder ODM (Organization Domain Modeling). Eine Übersicht entsprechender Ansätze ist in (Czarnecki und Eisenecker 2000) und (Kovačević et al. 2007) zu finden.

Alle diese Methoden haben den Begriff des „Features“ (dt. Merkmal) als gemeinsamen Nenner. Als „Feature“ definieren (Czarnecki und Eisenecker 2000) *“a distinguishable characteristic of a concept (e.g. system, component and so on) that is relevant to some stakeholder.”* Es handelt sich um ein Merkmal eines Systems das aus einer bestimmten Sicht relevant ist, z. B. aus der Sicht eines Auftraggebers oder der eines Nutzers. Featuremodellierung (dt. Merkmalsmodellierung) ist eine Methode, um Eigenschaften von Systemen und deren Varianten beschreiben zu können. Die grafische Darstellung eines Featuremodells visualisiert die Features eines Systems und die Beziehungen zwischen den Features. Je nach Ansatz und Notation kann ein solches Featuremodell unterschiedliche Elemente enthalten. Für die folgende Arbeit wird die FODA-Notation verwendet, da sie in der Forschung eine große Verbreitung genießt (Kang et al. 1990; Czarnecki und Eisenecker 2000).

Ein wesentlicher Aspekt der Featuremodellierung ist, dass die im Modell angegebenen Features nur *potenziell* in einem beschriebenen System der zugrunde liegenden Domäne vorhanden sind. Ein konkretes System wird daher durch *eine* mögliche Konfiguration des Featuremodells, mit selektierten und eliminierten Features festgelegt. Das Featuremodell als Ganzes (ohne Konfiguration) beschreibt durch seine Notationselemente eine Menge von legalen Konfigurationen.

Zu den Elementen eines Featuremodells gehören *mandatorische Features*, die immer in einem System enthalten sind, *optionale Features*, die je nach Produkt auftreten können sowie die Verknüpfung von Features in Form von *XOR*- oder *OR-Gruppen* (auch Alternative-Gruppen und Oder-Gruppen genannt). Man vergleiche die Notation in Abbildung 4. Die Blätter in einer Alternative und einer Oder-Gruppe können wiederum mandatorisch oder optional sein.

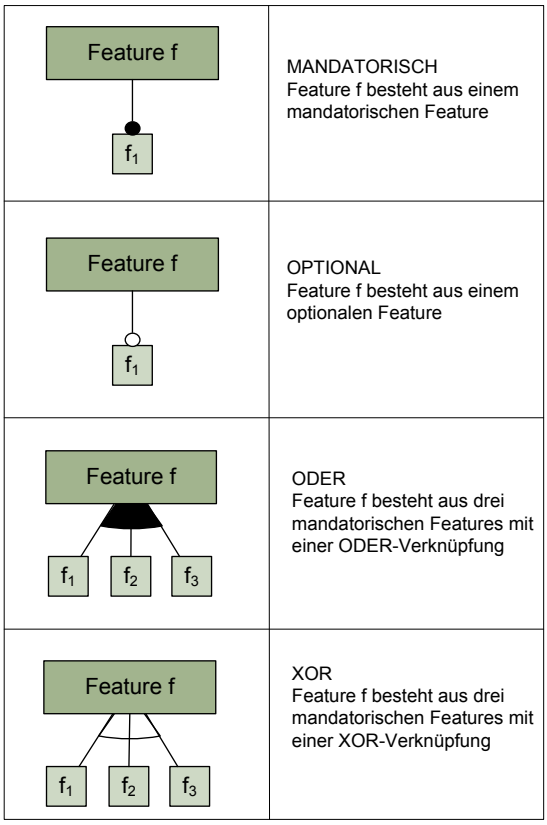

Abbildung 4 – Featuremodell Notation (in Anlehnung an (Czarnecki, Helsen und Eisenecker 2005))

Featuremodelle sind in der Regel als Hierarchie strukturiert und werden daher auch entsprechend als Baum visualisiert. Im Rahmen der vorliegenden Arbeit wurde durch Analyse der Domäne „Instant Messaging“ ein Featuremodell extrahiert und verfeinert, das die Familie der IM-Systeme beschreibt. In diesem Modell sind Gemeinsamkeiten und Unterschiede solcher Systeme zusammengefasst. Es sollte an dieser Stelle angemerkt werden, dass es sich hierbei – wie bei jedem anderen Modell eines Realitätsausschnitts auch – nicht um die einzig mögliche Strukturierungsform handelt. Vielmehr gibt dieses Modell (1) die Sichtweise der Verfasserin wieder und (2) fokussiert auf die, im Rahmen dieser Arbeit relevante Aspekte.

Auf der ersten Ebene unterhalb der Wurzel des Featuremodells (siehe Abbildung 5) sind die grundsätzlichen Eigenschaften eines IM-Systems zu finden: Jedes IM-System verfügt über eine Architektur (Topologie), ein verwendetes Protokoll (welches die unterschiedlichen Kommunikationskanäle und Arten der Nachrichten unterstützt), einen Client (den der Nutzer für seine Kommunikation verwendet) und optional einen Server (der den Verbindungsaufbau zwischen unterschiedlichen Clients übernimmt). Weiterhin realisieren IM-Systeme verschiedene Sicherheitsanforderungen, z. B. wird die Authentifikation der Benutzer von allen IM-Systemen durch einen Login-Prozess implementiert. Diese Eigenschaften werden durch die erste Ebene, (Farbe ■ apricot in Abbildung 5), dargestellt.

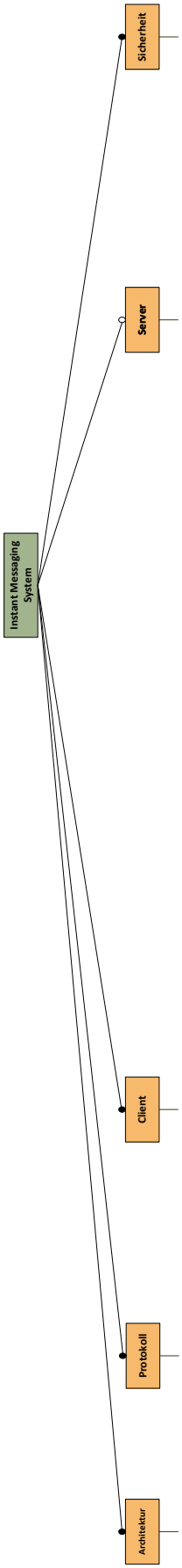

Abbildung 5 – Featuremodell (erste Ebene)

Jeder dieser Knoten führt zu einem Unterbaum, der detailliert die weiteren Merkmale beschreibt. Im nachfolgenden Kapitel werden die einzelnen Aspekte, d. h. die Unterbäume des Featurebaums, genauer betrachtet. Die Reihenfolge der Bearbeitung erfolgt dabei von links nach rechts.

6.1.1 Architektur

Viele Instant-Messaging-Systeme nutzen ein eigenes proprietäres Protokoll und eine entsprechende Netzwerkarchitektur. Bei grundsätzlichen Architekturprinzipien kann zwischen Peer-to-Peer- und Client-Server-Architektur unterschieden werden.

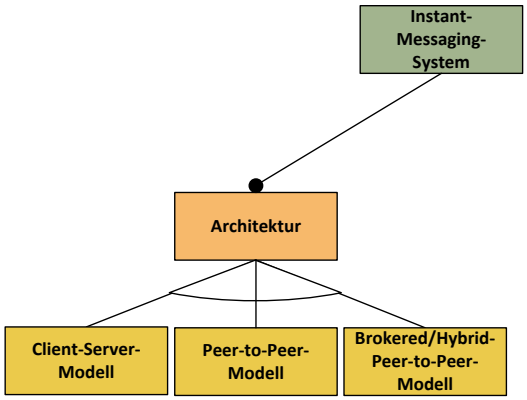

Abbildung 6 – Featuremodell - Architektur

6.1.1.1 Client-Server-Modell

Die erste Alternative ist eine Client-Server-Architektur, welche zum Beispiel bei Diensten wie ICQ, MSN, Yahoo! oder AOL genutzt wird. Der Server verfügt über Informationen zu den jeweiligen Clients und den angebotenen Diensten (unterschiedliche Präsenzangaben, Kontaktliste, Kommunikationskanäle). Je nach konkretem System existieren unterschiedliche Server für die Authentifikation, das Routing der Daten und die Verwaltung von Präsenzinformationen. Textnachrichten (Chat-Nachrichten) werden über den Routing-Server an den Empfänger geleitet und auch zwischengespeichert, wenn der Empfänger gerade nicht im System angemeldet ist oder die Verbindung unterbrochen wurde. Der Umfang der verfügbaren Funktionen wird vom Server gesteuert. So kann der Server zum Beispiel eine VoIP-Verbindung ablehnen, wenn nach seinen Informationen einer der Teilnehmer aufgrund einer älteren Client-Software diese Kommunikationsform nicht unterstützt.

6.1.1.2 (Reines) Peer-to-Peer-Modell

Die zweite Alternative basiert auf einem Peer-to-Peer-Modell (P2P), d. h. einem Kommunikationsmodell gleichberechtigter Einheiten, welches eine direkte Verbindung (Friend-to-Friend) oder indirekte Verbindung (Web-of-Trust) zwischen den Teilnehmern erlaubt. In einer reinen Peer-to-Peer-Topologie existiert kein Server, der Peers miteinander bekannt macht. Ein Peer muss daher dafür sorgen, dass er mindestens einen Peer eines Netzes erreichen kann. Dafür müssen die IP-Adressen der Teilnehmer (Peers) bekannt sein (Bischofs, Hasselbring und Warns 2006; Eberspächer und Schollmeier 2005).

Nur wenige Instant Messenger nutzen diese reine Variante von P2P, da alle Teilnehmer des IM-Netzwerks als Index (Liste) an jeden Peer gesendet und verwaltet werden. Geht man davon aus, dass kein Client für einen anderen Client Nachrichten zwischenspeichert, können bei dieser Variante nur Clients miteinander kommunizieren, die zu einem bestimmten Zeitpunkt gleichzeitig online sind. Diese Variante ist auch als Pure (dt. „reines“) P2P bekannt und wird hauptsächlich von dezentralen Filesharing-Diensten eingesetzt. Im Bereich Instant Messaging sind Systeme wie das WASTE-Projekt (WASTE 2008) und das Torchat-Projekt (Tor 2008) zwei Beispiele für einen derartigen Ansatz. WASTE ermöglicht den Aufbau eines IM-Netzes in kleinen Gruppen mit bis zu 50 Teilnehmern, unterstützt One-to-one Instant Messaging, Group Chat und Filesharing. Torchat ist ein Dienst der Tor-Software, welche den Aufbau eines anonymen Netzes ermöglicht. Die Anonymität soll dadurch gewährleistet werden, dass eine Verbindung über mehrere Proxy-Server mit einer Mix-Funktionalität geleitet wird. Durch die prinzipiellen Eigenschaften des zugrundeliegenden Tor-Netzes erhält man mit Torchat einen anonymen Peer-to-Peer-Instant Messenger, da der Ursprung der Nachrichten nicht nachvollziehbar ist.

6.1.1.3 Brokered Peer-to-Peer-Modell

Beim Brokered P2P-Modell (dt. „vermitteltes“ Modell, auch als zentralisierte P2P-Infrastruktur bezeichnet) besteht die Infrastruktur aus einem Server, der eine Koordinationsfunktion übernimmt und einer Anzahl von Clients zwischen denen die eigentliche Kommunikation über P2P-Protokolle stattfindet (Steinmetz und Wehrle 2005; Hess, Anding und Schreiber 2002). Im Gegensatz zu einem Client-Server-Modell, sind bei diesem Ansatz die Aufgaben des Servers begrenzt. Er stellt Informationen zur Verfügung, z. B. welche Peers mit welchem Status verfügbar sind. Gegebenenfalls leitet der Server Anfragen direkt an die entsprechenden Peers weiter. Die eigentliche Kommunikation (Chat-Nachrichten) erfolgt nach dem P2P-Modell. Dadurch, dass sie durch einen Verwaltungsserver koordiniert wird, kommt der Name „brokered“ zu Stande. Dieses Modell ist das meist genutzte in Instant-Messenger-Diensten. Dabei werden je nach Protokoll Ergänzungen vorgenommen und Varianten eingeführt. So nutzen z. B. Dienste wie ICQ, MSN oder AIM die Kommunikation über den Server, um Text-Nachrichten zwischenzuspeichern und zu versenden. Größere Daten, wie Bilder, MP3s oder Videos werden dagegen direkt zwischen den Peers übertragen.

6.1.1.4 Hybrides Peer-to-Peer-Modell

Das hybride P2P-Modell (auch als Super-Node-P2P bekannt) ist eine Erweiterung der zentralisierten Variante (Kapitel 6.1.1.3 Brokered P2P) und besteht aus *mehreren* zentralisierten P2P-Netzen (Steinmetz und Wehrle 2005). Dabei haben die Super-Nodes die gleiche Koordinationsaufgabe (z. B. Indizierung oder Suchanfragen) wie ein Server im Client-Server-Modell, aber sie koordinieren jetzt nicht einzelne Peers sondern ganze Subnetze. Der Vorteil gegenüber einem brokered P2P-Netz ist, dass man von der Zuverlässigkeit eines einzelnen Verwaltungselements (einem zentralen Server) unabhängig ist und trotzdem Suchanfragen schnell und effizient ablaufen, ohne dass alle Peers in diese Aufgabe involviert werden müssen. Ein Vertreter dieser Architektur ist Skype.

6.1.2 Protokolle

Eines der wichtigsten Merkmale eines Instant-Messaging-Systems ist das verwendete Protokoll, das für die Nachrichtenübermittlung zuständig ist und somit implizit die angebotene Dienste und Funktionen festlegt. Man unterscheidet zwischen proprietären und standardisierten Protokollen (vgl. Abbildung 7).

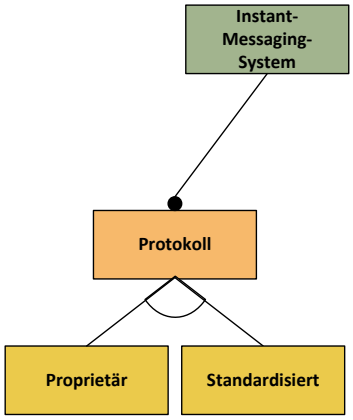

Abbildung 7 – Featuremodell - Protokolle

Eine weitere Klassifizierung von IM-Systemen erfolgt anhand der Dienste, welche die Protokolle anbieten, wie etwa Protokolle für Textnachrichten, für Multimediadaten oder für Statusinformationen. Man unterscheidet zwischen proprietären Protokollen und Standards. In die letzte Kategorie gehören auch die Erweiterungen von Standards, die noch nicht standardisiert sind, und Vorschläge für die Aufnahme in einen Standard-Status. In die Kategorie proprietäre Protokolle gehören sowohl Protokolle mit Open-Source-Code als auch solche, die als Closed-Source-Code entwickelt worden sind. Auf den nächsten Seiten wird eine Auswahl von proprietären Protokollen (OSCAR, MSNP, Skype) und Standards in diesem Bereich (XMPP, SIP, RTP) detaillierter erläutert. Die Auswahl basiert auf der Popularität der Nutzung und Entwicklung von IM-Clients.

6.1.2.1 Open System for Communication in Realtime (OSCAR)

OSCAR steht für „Open System for Communication in Realtime“ und ist das Protokoll, welches sowohl von ICQ („I seek you“) als auch von AIM (AOL Instant Messenger) unterstützt wird (AOL Developer Network 2009). Trotz des gleichen Basisprotokolls unterscheiden sich diese Systeme in den Konfigurationsfunktionen, wie zum Beispiel Kennungen oder Präsenzangaben. Während ICQ eine (mindestens fünfstellige) Zahl für die Identifikation von Nutzern verwendet (als Identifikator und sozusagen als Benutzername), fängt der Benutzername bei AIM mit einem Buchstaben an. Auch Länge und Art der Übertragung des dazugehörigen Passworts ist bei beiden Systemen unterschiedlich. Der Benutzer hat bei ICQ die Möglichkeit, detaillierte Angaben zu seiner Präsenz zu machen wie etwa „online“, „offline“, „nicht verfügbar“, „nicht stören“, während AIM nur zwischen einem „anwesend“- und „abwesend“-Status unterscheidet.

Das OSCAR-Protokoll war bis 2008 nur als Closed Source verfügbar. Das führte dazu, dass viele Drittanbieter und Entwickler von Multiprotokoll-Clients das Protokoll mit Hilfe von Reverse-Engineering-Verfahren analysierten und zahlreiche Clients entwickelten. Die Antwort von

AOL darauf war die Entwicklung eines weiteren Protokolls namens TOC-Protokoll (Talk to OSCAR) (AOL 1998), welches eine kleine, jedoch echte Teilmenge von OSCAR-Funktionen bot. Dies legte AOL für die Weiterentwicklung und Nutzung durch andere Anbieter offen. Allerdings hat sich dieses Protokoll wegen des eingeschränkten Funktionsumfangs nicht durchgesetzt. 2005 stellte AOL den Support für TOC ein und veröffentlichte 2008 das OSCAR-Protokoll, jedoch mit Auflagen für andere Entwickler. Das TOC-Protokoll wird an dieser Stelle wegen seiner geringen Nutzung und Bedeutung nicht weiter betrachtet.

Entsprechend der allgemeinen Funktionsweise von Instant-Messaging-Systemen, die in der Abbildung 3 dargestellt wurde, verfügt ein Benutzer bei der Nutzung des ICQ-Netzes und des OSCAR-Protokolls über eine Benutzerkennung. Diese enthält eine Identifikationsnummer (z. B. ICQ# 557013871) und ein Passwort. Während der Anmeldung werden diese Daten an den Authentifikationsserver (z. B. api.screenname.aol.com und api.oscar.aol.com) gesendet. Der Authentifikationsserver antwortet mit den Angaben zum Kommunikationsserver (Instant-Messaging-Server). Im Fall von ICQ ist das der sogenannte BOSS-(Basic Oscar Server Service-) Server. Die Antwort enthält die IP-Adresse und den Port des BOSS-Servers sowie einen temporären Schlüssel (den sogenannten Authorization Cookie). Der Client baut mit diesen Informationen eine Verbindung zum BOSS-Server auf. Letzterer hat auch das Authorization Cookie vom Authentifikationsserver erhalten und vergleicht diesen Wert mit dem vom Client übermittelten Wert. Stimmen die Werte überein, ist der Client erfolgreich im IM-Netz angemeldet und kann Nachrichten mit anderen Kontakten austauschen. Während dieser ersten Verbindung zum BOSS-Server werden auch andere Informationen und Einstellungen des Clients übertragen (z. B. sein Präsenzstatus, seine Profilinformationen, seine Kontakte aus der Kontaktliste oder ob er Peer-to-Peer-Verbindungen mit anderen Kontakten erlaubt). Die letzte Angabe ist insoweit erwähnenswert, da ICQ sowohl eine Peer-to-Peer-Kommunikation als auch eine Kommunikation über den BOSS-Server realisiert. Das OSCAR-Protokoll verwendet (sozusagen als Subprotokolle) die TCP-basierten Protokolle FLAP (Frame Layer Application Protocol) für die Erstellung der Frames und SNAC (Simple Network Atomic Communication) für die OSCAR-spezifischen Nachrichten zwischen Client und Server. (Schlildt 2005; AOL Developer Network 2009; Rittinghouse und Ransome 2005)

6.1.2.2 Mobile Status Notification Protocol (MSNP)

Das Mobile Status Notification Protocol (MSNP) wurde von Microsoft für den MSN Messenger (auch bekannt als .NET Messenger) entwickelt und in der Version 2 im Jahre 1999 über die Internet Engineering Task Force (IETF) veröffentlicht (Movva und Lai 1999). 2003 wurde der Support für diese Version eingestellt. Weitere Versionen des Protokolls sind nicht mehr öffentlich. In der aktuellen Version mit dem Namen Windows Live Messenger 2009 (Microsoft 2009a) wird MSNP 18 verwendet. Daher basieren die folgenden Aussagen auf Informationen, die durch Reverse-Engineering-Analysen gewonnen wurden (Mintz und Sayers 2003; Schlildt 2005; MSNPiki 2009; Williams und Ly 2004).

Im Vergleich zu den anderen IM-Protokollen gilt MSNP als ein komplexeres Protokoll, unter anderem aufgrund der Nutzung unterschiedlicher Server für die Authentifizierung der Benutzer und die Übermittlung der Nachrichten. Der Authentifikationsmechanismus basiert auf der .NET-Passport-Technologie (Microsoft TechNet 2005) und einem Challenge-Response-

Verfahren für die Überprüfung der Log-in-Daten des Clients. Bei .NET-Passport handelt es sich um ein Single-Sign-On-System, d. h., es existiert eine einheitliche Kennung für mehrere Dienste (Eckert 2009)Dabei wird eine E-Mail-Adresse als Identifier verwendet, um einen personalisierten Zugang zu Passport-aktivierten Diensten und Websites zu ermöglichen. Im Falle des Windows Live Messengers ist der Identifikator des .NET-Passports und des IM-Dienstes eine E-Mail-Adresse der Form xyz@hotmail.com. Möchte ein Benutzer den Windows Live Messenger benutzen, muss er sich zunächst an den Dispatch-Server (messenger.hotmail.com) wenden. Dieser ist zuständig für die Mitteilung der IP-Adresse und des Ports eines sogenannten Notification-Servers. So ein Notification-Server steuert dann den gesamten Datenfluss während der Nutzung des Live Messengers, überprüft die Passport-Identität des Benutzers und handelt Parameter wie die Version des Clients oder die Version des verwendeten Protokolls mit dem Server aus. Dafür erhält der Client vom Notification-Server einen Challenge-String und die IP-Adresse des Authentication-Servers (.Net-Passport-Servers). Kann der Benutzer dem Authentication-Server seine Identität (Passport, Passwort, Challenge-String) nachweisen, erhält er einen Schlüssel (Ticket), mit dem er die Anmeldung am Notification-Server abschließen kann. Ist die Anmeldung erfolgreich, werden weitere Daten wie sein Präsenzstatus und die Kontaktliste zwischen Client und Server ausgetauscht.

Ein weiterer Server (Switchboard-Server) übernimmt den Nachrichtenaustausch zwischen den Clients. Das MSNP erlaubt keine Peer-to-Peer-Übertragung. Möchte ein Client eine Chat-Nachricht an einen seiner Kontakte verschicken, erhält er vom Notification-Server die Adresse des Switchboard-Servers. Das bedeutet, dass jeder Benutzer, der eine Chat-Nachricht verschickt, Teil einer sogenannten Switchboard-Session ist. Ein Benutzer kann an mehreren Sessions teilnehmen, wenn er mit mehr als einem Teilnehmer eine Chat-Verbindung unterhält. Andererseits kann eine Session mehrere Teilnehmer haben, wenn es sich um einen Gruppenchat handelt. Über das Switchboard werden auch andere Dienste wie Spiele oder der Filesharing verwaltet. Beim MSNP sind alle Inhalte (Textnachrichten, Sprache, Video) in MIME kodiert. Das MSNP verwendet TCP-Verbindungen und den festen Port 1863, welcher nicht im Client konfigurierbar ist (Piccard et al. 2005; MSNPiki 2009).

6.1.2.3 Skype

Skype wurde 2003 von den Gründern der Firma Skype Technologies entwickelt. Es handelt sich dabei um ein proprietäres Closed-Source-Protokoll für Instant Messaging und Multimedia-Kommunikation. Wie bei den oben genannten Protokollen basieren die hier gemachten Angaben auf Reverse Engineering und Protokollanalysen mit Hilfe von Werkzeugen wie Ethereal (Ethereal 2006) oder NetPeeker (NetPeeker 2009). Das von Skype benutzte Protokoll ist eines der wenigen, welches hauptsächlich Peer-to-Peer-Übertragung verwendet. Dies erklärt sich unter anderem durch den Ursprung von Skype als ein weiteres Produkt von Niklas Zennström und Janus Friis, welche auch für die Entwicklung anderer Peer-to-Peer-Anwendungen wie Kazaa und Napster verantwortlich waren (Baset und Schulzrinne 2006; Piccard et al. 2005).

2005 wurde Skype von eBay gekauft, mit der Intention, durch Klingeltöne, Werbung und eine Integration in PayPal neue Geschäftsbereiche zu erschließen (Skype 2008). Aktuell ist allerdings ein möglicher Verkauf von Skype im Gespräch. Der Verkauf ist durch einen Gerichtsprozess bedroht, der durch die ursprünglichen Gründer von Skype angestrebt wird. Sie behaupten,

dass sie beim Verkauf an eBay die Rechte an einer Schlüsseltechnologie für Skype behalten hätten, und klagen mit dem Vorwurf der Urheberrechtsverletzung (Poletti 2009; Gohring 2009).

Bei Skype handelt sich um ein modernes IM-System, dessen Kernfunktion die Unterstützung von Voice-over-IP-Telefonie ist, was auch seine starke Verbreitung erklärt. Die Architektur und der Verbindungsaufbau von Skype basieren auf einem Hybrid-P2P-Netzwerk (vgl. Kapitel 6.1.1.4). Dieses Netz besteht aus Skype-Clients, Super-Nodes und den Log-in-Servern.

Möchte ein Nutzer von Skype mit anderen Teilnehmern kommunizieren, muss er wie bei anderen IM-Systemen über die Client-Software und eine Kennung verfügen. Bei Skype kann für eine Kennung eine beliebige Zeichenkette (aus A-Z, a-z, 0-9 und den Sonderzeichen . , @ - _) gewählt werden (Skype 2006). Die Aufgabe der Registrierung neuer Skype-Accounts und Authentifizierung vorhandener Benutzer übernimmt der Skype–Log-in-Server. Dieser stellt sicher, dass jede Kennung nur einmal vergeben wird. Seit der Version 1.2 des Clients verwaltet er auch die Kontaktlisten jedes Nutzers.

Ein Super-Node ist ein spezieller Client, der Signalisierungs- und Erreichbarkeitsinformationen im Skype-Netz verteilt und Informationen wie Präsenz oder Kontaktlisten der Skype-User synchronisiert. Jeder Client mit einer öffentlichen IP-Adresse und einem installierten Skype-Client kann Super-Node werden, wenn er hohe Upload- und Speicherressourcen sowie eine öffentlich routbare IP-Adresse mit hoher Netzverfügbarkeit besitzt. Ein Benutzer hat keinen Einfluss darauf, ob sein Client ein Super-Node wird oder nicht. Dieses Vorgehen ist Teil der AGBs, die man vor der Nutzung von Skype akzeptieren muss. Während des Einloggens versucht der Skype-Client, zunächst eine Verbindung zu einem Super-Node aufzubauen. Eine Liste der erreichbaren Super-Nodes ist auf dem Client gespeichert. Diese Liste kann bis zu 200 Eintragungen enthalten. Ist ein solcher Versuch nicht erfolgreich, versucht der Client mit einer der sieben IP-Adressen und Ports, die fest im Client codiert sind, eine Verbindung herzustellen. Erst nachdem eine erfolgreiche Verbindung zu einem Super-Node feststeht, kann sich der Client auf dem Log-in-Server authentifizieren. Skype unterstützt sowohl TCP-Pakete als auch UDP-Datagramme und nutzt dafür unterschiedliche Ports. Die Default-Einstellung ist Port 1387, aber erlaubt werden zum Beispiel auch Verbindungen über Ports, die typischerweise für HTTP-Übertragung verwendet werden (Port 80 und 443).

6.1.2.4 Jabber/ Extensible Messaging and Presence Protocol (XMPP)

Bei den bisherigen Protokollen handelte es sich um proprietäre Protokolle, die zum einen in der Regel nicht offengelegt (Closed Source) und zum anderen untereinander nicht kompatibel sind. Als Konsequenz können Nutzer eines IM-Systems andere Kontakte nur dann erreichen, wenn diese im gleichen System angemeldet sind. Eine Lösung für dieses Problem strebt das Jabber-Protokoll an, welches von der Internet Engineering Task Force (IETF) unter der der Bezeichnung „Extensible Messaging and Presence Protocol" (XMPP) als Standard-Protokoll für Instant Messaging verabschiedet wurde. Das Jabber-Protokoll wurde ursprünglich von Jeremy Miller vorgestellt und hatte das Ziel, ein sicheres, Spam-freies, dezentralisiertes und quelloffenes IM-System zu realisieren. Im Jahr 2000 wurde der erste Server (jabberd-Server) mit den wichtigsten Basisprotokollen (XML-Streaming, Instant Messaging, Presence Awareness, Kontaktlisten usw.) veröffentlicht (Saint-Andre 2004a; Rittinghouse und Ransome 2005).

Im August 2001 wurde die Jabber Software Foundation (JSF) gegründet mit der Aufgabe, alle XML-Protokolle, die im Rahmen des Jabber-Protokolls verwendet wurden, zu dokumentieren und weiterzuentwickeln. 2002 wurden die Basisprotokolle von Jabber bei der IETF zur Standardisierung eingereicht. Die Arbeitsgruppe „Extensible Messaging and Presence Protocol“ (XMPP) übernahm die Spezifikation des Protokolls, welches 2004 unter dem gleichen Namen veröffentlicht wurde (Adams 2002; Jabber.org 2009).

Jabber ist ein Framework, welches unterschiedliche Clients und Server unterstützt, die auf dem gleichnamigen Protokoll (auch XMPP) basieren. Als Identifikator nutzt das XMPP einen sogenannten Jabber Identifier (JID), der einer E-Mail-Adresse in der Form xyz@server.com, ähnelt. Der Benutzername ist hier durch xyz angegeben, server.com bezeichnet den Authentifikationsserver, über den die Registrierung und Anmeldung abgewickelt wird. Dadurch ist es möglich, Benutzernamen mehrfach zu vergeben, solange die Benutzer unterschiedliche Server verwenden. Eine Besonderheit von Jabber ist die gleichzeitige Anmeldung eines Benutzers an unterschiedlichen Geräten. Diese werden auch als Ressourcen bezeichnet. In diesem Fall wird die JID-Adresse dann folgendermaßen angegeben: xyz@server.com/ressource. Um bei der parallelen Anmeldung an unterschiedlichen Ressourcen, Probleme bei der Zustellung von Nachrichten zu vermeiden, werden die Ressourcen priorisiert. Die Nachrichten werden bei einer fehlenden Ressourcenangabe der Ressource mit der höchsten Priorität zugestellt. Sind mehrere Ressourcen mit gleicher Priorität angemeldet, stellt der Server die Nachricht der Instanz zu, die sich als letzte an dem Server angemeldet hat. Der XMPP-Server bietet verschiedene Schnittstellen, mit denen sich nicht nur Clients sondern auch weitere Server verbinden können. Dabei werden auch Server unterstützt, die nicht auf dem XMPP basieren. Diese Protokollunabhängigkeit, ist einer der größten Vorteile von Jabber/XMPP. Für die Kommunikation zwischen Client und Server werden (in einer Standard-Konfiguration) eine TCP-Verbindung und der Port 5222 verwendet, für die Kommunikation zwischen Servern dagegen der TCP-Port 5269. Für den Authentifizierungsvorgang werden Benutzername und Passwort, wie vom Benutzer eingegeben, mit den Daten im sogenannten Authentication Data Store (einem Modul des Servers) verglichen. Andere Informationen wie etwa Kontaktlisten oder Profilinformationen werden ebenfalls auf dem Server gespeichert. (Saint-Andre, Smith und Tronçon 2009)

6.1.2.5 Session Initiation Protocol (SIP)

SIP (Session Initiation Protocol) ist ein Protokoll, welches die Übertragung multimedialer Daten steuern kann. Es kontrolliert Aufbau, Steuerung, Modifikation und Abbau von Sitzungen zwischen zwei oder mehreren Teilnehmern – übernimmt aber nicht die eigentliche Übertragung der Multimediadaten. Die kontrollierten Sitzungen können Anwendungen wie VoIP, Video-over-IP oder die Übertragung von Textnachrichten und Filesharing sein. Somit ist es für SIP nicht wichtig *welche* Daten übertragen werden, sondern *wie* die Daten einer Anwendung übertragen werden. Die eigentliche Übertragung der multimedialen Daten wird z. B. durch das Real-time Transport Protocol (RTP)- übernommen (vgl. Kapitel 6.1.2.7). Das SIP-Protokoll wurde von der IETF (Internet Engineering Task Force) entwickelt und in der ersten Version 1999 (RFC 2543) veröffentlicht (Handley et al. 1999). In neueren RFCs (RFC 3261-3265) werden Erweiterungen und Verbesserungen zum ursprünglichen Protokoll festgelegt (Campbell et al. 2002; Roach 2002; Rosenberg et al. 2002).

Im Zusammenhang mit Medienströmen können Anwendungen über eine *direkte* Verbindung (Ende-zu-Ende-Verbindung) oder *indirekt* über einen sogenannten SIP Application Server kommunizieren. SIP ist ein Client-Server-Protokoll, dessen Nachrichtenaufbau und Ablauf dem HTTP ähnelt. Jedes SIP-Endsystem kann die Rolle des Servers (User Agent Server, UAS) oder Clients (User Agent Client, UAC) übernehmen. Die Kontaktadresse eines Endsystems wird durch einen SIP URI (Uniform Resource Identifier) oder eine SIP-URL-Adresse (Uniform Resource Locator) festgelegt. Es handelt sich dabei um eindeutige Benutzerbezeichnungen, welche an den syntaktischen Aufbau einer E-Mail-Adresse erinnern. Sie haben Formate wie sip:user@domain, sip:user@host oder sip:user@ip_adress (Kanbach 2005; Trick und Weber 2005).

Ähnlich wie bei der Kommunikation zwischen HTTP-Client und -Server, besteht die Kommunikation zwischen UAS und UAC aus zwei Arten von Nachrichten: *Requests* und *Responses*. Requests sind Nachrichten an einen Empfänger, um für die Kommunikation notwendige Transaktionen anzufordern oder durchzuführen. Beispiele dafür sind der erste Schritt beim Verbindungsaufbau oder die Abfrage von Informationen über Endsysteme oder bestehende Sessions. Die Nachrichten unterscheiden sich durch ihre sogenannte *Method* (Methode). Die wichtigsten Methoden (Grundnachrichten) sind:

- *INVITE* – Beginn einer SIP-Session
- *ACK* – Bestätigung der finalen Statusinformation
- *BYE* – Die Session wird terminiert
- *CANCEL* – Abbrechen; bei Nutzung von VoIP: „klingeln lassen“
- *OPTIONS* – Abfrage der unterstützten Funktionen der Gegenseite
- *REGISTER* – Registrierung mit lokalen Diensten

Darüber hinaus gibt es auch erweiterte Nachrichten, die vom Sender verschickt werden, zum Beispiel um ein weiteres Endsystem zu kontaktieren, eine schriftliche Kurzmitteilung zu schicken, Präsenzinformationen mitzuteilen oder Parameter innerhalb einer Session zu aktualisieren. Response-Nachrichten sind definiert als Empfangsbestätigungen oder Reaktionen durch den Empfänger. Jede Request-Nachricht erhält automatisch eine Antwort in Form einer Response. Responses werden durch Status-Codes gekennzeichnet, zum Beispiel:

- *100* – Trying
- *180* – Ringing
- *200* – OK
- *400* – Bad request

Für eine direkte Kommunikation zwischen zwei SIP User Agents muss dem UAC die IP-Adresse des Empfängers bekannt sein. Die Verbindung wird aufgebaut, indem eine INVITE-Nachricht vom Sender verschickt wird. Der Empfänger nimmt ab und antwortet mit der Statusinformation „200 – OK“. Nach der Versendung einer ACK-Nachricht durch den Sender kann die eigentliche Session und Übertragung der multimedialen Daten starten. Nach einer erfolgreichen Kommunikation wird die Session mit einer BYE-Nachricht und der entsprechenden OK-Nachricht abgeschlossen. Dieses Grundmuster des Verbindungsaufbaus bleibt gleich, unabhängig davon, ob es eine VoIP- oder Videoverbindung ist. Je nach SIP-Infrastruktur können zwi-

schen den Endsystemen Proxy-Server (für Authentifikation und Autorisierung von Nachrichten), ein Redirect-Server (für die Umleitung von Nachrichten) oder ein Location-Server (mit Informationen über den Aufenthaltsort des Teilnehmers) agieren (Kanbach 2005; Rupp, Siegmung und Lautenschlager 2002).

Das SIP nutzt unterschiedliche andere Protokolle für seine Funktion. Dazu gehört unter anderem das SDP (Session Description Protocol), welches verwendet wird, um Parameter bei der Verbindungsaushandlung zu definieren (Handley und Jacobson 1998). Über IMPP (Instant Messaging and Presence Protocol) wird der Status von User Agents beobachtet und mitgeteilt (Day, Rosenberg und Sugano 2000). Anschließend wird RTP (Real-time Transport Protocol) verwendet, um die eigentlichen multimedialen Daten zu übertragen (Schulzrinne et al. 2003)

6.1.2.6 Jingle

Das Jingle-Protokoll wurde als eine Erweiterung von XMPP konzipiert, um Peer-to-Peer-Sitzungen mit Multimedia-Inhalten zu unterstützen, zum Beispiel in Videokonferenzen (Ludwig et al. 2009a). Es erlaubt eine Kommunikation zwischen zwei XMPP-Instanzen und agiert als ein Signalisierungsprotokoll für eine Session. Der Aufbau einer Sitzung läuft ähnlich wie beim SIP-Protokoll ab:

- *Session initiate* – Beginn einer Sitzung
- *Session accept* – Explizite Bestätigung einer Sitzung
- *ACK* – Bestätigung der letzten Information
- *Session terminate* - Terminierung einer Sitzung

Während dieses Aufbaus wird ausgehandelt, welche Art von Daten übertragen werden sollen (z. B. Video- oder Desktop-Sharing) und über welches Übertragungsprotokoll (UDP, TCP) dies geschehen soll. Nachdem eine Sitzung aufgebaut ist, können über Jingle die Inhalte unterschiedlicher Anwendungen wie Voice- und Video-Daten, Filesharing oder Applikation-Sharing übertragen werden. Dafür wird aktuell an entsprechenden Vorschlägen für die Erweiterung des Protokolls gearbeitet. Beispiele hierfür sind Jingle RTP, Jingle XML Streams, oder Jingle File Sharing (Ludwig et al. 2009b; Saint-Andre, Karneges und Meyer 2009; Saint-Andre 2009). Ein bekannter Instant-Messaging-Vertreter, der das Jingle-Protokoll nutzt ist Google Talk (Google 2009).

6.1.2.7 Real-time Transport Protocol (RTP)

Das Real-time Transport Protocol (RTP) ist bei der Nutzung eines IM-Systems für die Übertragung der Multimedia-Datenströme zuständig, nachdem mit Hilfe von SIP (Kapitel 6.1.2.5) oder Jingle (Kapitel 6.1.2.6) eine Session aufgebaut wurde (Schulzrinne et al. 2003).

Das Protokoll wurde erstmals 1996 im RFC 1889 beschrieben. 2003 wurde es durch RFC 3550 abgelöst, welcher auch das RTCP (RTP Control Protocol) beschreibt (Schulzrinne et al. 2003). Zu den Aufgaben des RTP gehören die Übermittlung der Sprachdaten, die Garantie der richtigen Reihenfolge der Sprachpakete und der Transport der Daten über unterschiedliche Formate, z. B. bei der Weiterleitung eines Anrufs vom VoIP-Netz auf das Festnetz. Eine Teilaufgabe des RTP ist die Überwachung der Übertragungsqualität (Quality of Service), die durch das Teilprotokoll RTCP dadurch erfüllt wird, dass es Kontrollinformationen (z. B. über die Bandbreite)

einer Verbindung übermittelt. RTP und RTCP stellen *keine* Prozeduren zum Aufbau oder Abbau von Sessions bereit. Daher muss diese Aufgabe von anderen Protokollen wie SIP oder H.323 übernommen werden. Diese wiederum erledigen dann die Steuerung der Kommunikation, die Lokalisierung der Nutzer und den Austausch von anderen Parametern, welche für die RTP-Kommunikation notwendig sind. Der Transport der Audio-/Video-Daten erfolgt bei RTP über UDP. Um die Dienstsynchronität zu gewährleisten, nutzt RTP eine Nummerierung der Pakete (Sequenznummer) und Zeitstempel (Timestamp). RTP hat keinen festgelegten Port, ein Vorschlag ist der Port 5004. (Schulzrinne et al. 2003; Grimm, Meletiadou und Hundacker 2008; Badach 2007)

6.1.2.8 Weitere Protokolle

Es gibt viele weitere Protokolle, die Teilaufgaben von Instant-Messaging-Systemen erfüllen. Dazu gehören unter anderem Protokolle, die in größeren Kommunikationslösungen wie etwa Real-Time-Communication-Anwendungen verwendet werden und diese um IM-Eigenschaften erweitern. Derartige Anwendungen stehen nicht im Fokus der vorliegenden Arbeit und werden daher hier nicht im Detail betrachtet. Zur Vollständigkeit jedoch ein kurzer Überblick:

- *SIMPLE-Protokoll* – Das SIMPLE-Protokoll (SIP for Instant Messaging and Presence Leveraging Extensions) wurde als Standard für Präsenzinformationen und Instant Messaging im Kontext von SIP entwickelt. Dabei wird das SIP, welches für die Signalisierung in großen VoIP-Lösungen verwendet wird, auch für die Übermittlung von Präsenzangaben und IM-Nachrichten eingesetzt. Das Protokoll wurde von einigen großen IP-PBX-Herstellern übernommen, zum Beispiel von Avaya, Nortel, Siemens und Cisco. (Rosenberg 2006; Roach 2002)
- *IMPS-Protokoll* – Das IMPS-Protokoll ist ein weiteres Präsenzprotokoll, welches 2001 von der Wireless Village Initiative (bestehend aus Ericsson, Nokia und Motorola) beschrieben wurde, um eine gemeinsame Spezifikation für mobiles Instant Messaging zu definieren (Ericsson, Motorola und Nokia 2002). Später übernahm die Open Mobile Alliance (OMA) diese Gruppe und die Protokolle wurden unter dem Namen Instant Messaging and Presence Service (IMPS) weiterentwickelt. (Open Mobile Alliance 2007b, 2007a)
- *H.232-Protokoll* – Der H.323-Standard wurde von der ITU-T entwickelt und definiert ein komplettes Framework für die Übertragung von Audio- und Video-Daten. Er definiert Einzelheiten zur Signalisierung, Paketierung, Kodierung und zum Bandbreitenmanagement einer Verbindung. Die Signalisierungsdaten und die Beschreibung einer Sitzung werden über die Protokolle H.225.0 und H.245 übertragen (Badach 2007).

6.1.3 IM-Client

Nachdem Architektur und Protokolle als Merkmale eines IM-Systems betrachtet wurden, wird als Nächstes das dritte mandatorische Merkmal eines IM-Systems, der Client, fokussiert.

Ein IM-Client selbst ist in dem Featuremodell durch zwei (Unter-)Merkmale beschrieben: eine Plattform, und die angebotenen Funktionalitäten (Farbe apricot, 2. Ebene in der Abbildung 8).

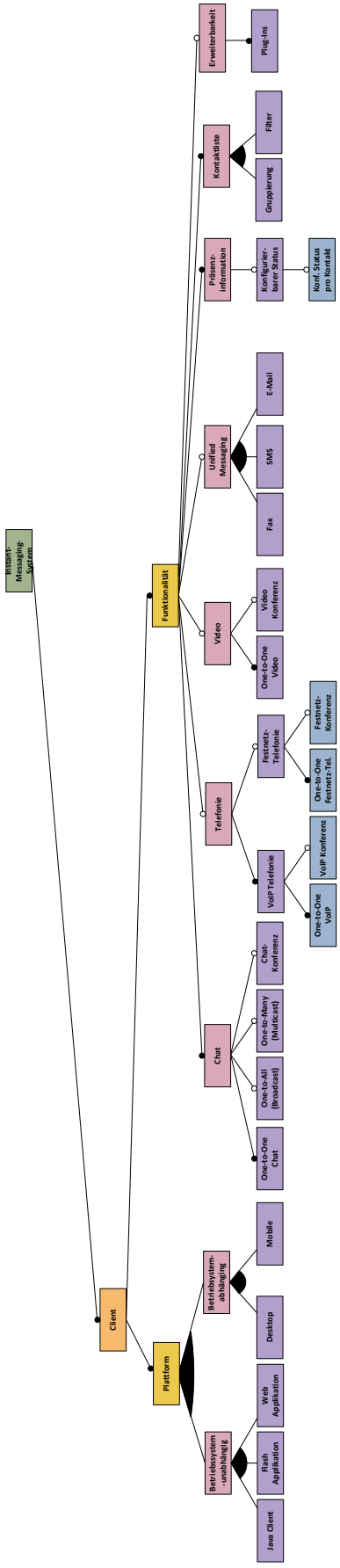

Abbildung 8 – Featuremodell - IM-Client

6.1.3.1 Plattform

Das erste Merkmal eines IM-Clients ist die Plattform. Es wird zwischen betriebssystemabhängigen und -unabhängigen Plattformendifferenziert. Bei den betriebssystemabhängigen Varianten wird zwischen Desktop- und mobilen Versionen unterschieden. Die mobilen IM-Clients werden von manchen Systemen angeboten, bieten aber häufig weniger Funktionen als die Desktop-Version (z. B. bietet die mobile Version von Skype keine Videoübertragung). Mobile Versionen stehen für die Systeme Symbian, Windows Mobile, Linux Client oder Java ME zur Verfügung. Die meisten Desktop-IM-Systeme werden für Microsoft Windows angeboten. Es existieren aber auch Varianten für Linux-Distributionen oder Mac OS. Je nach IM-System und Marktverbreitung werden für die verschiedenen anderen Betriebssysteme unterschiedliche Funktionen realisiert. So sind teilweise IM-Client-Versionen für bestimmte Betriebssysteme mit weniger Funktionalität ausgestattet oder neue Funktionen werden mit Verzögerung portiert.

In die Kategorie der betriebssystemunabhängigen Varianten gehören Java-, Web-Entwicklungen oder Flash-Applikationen. Um die Installation einer Software auf dem eigenen Rechner zu vermeiden oder die Nutzung von einem fremden Rechner aus zu ermöglichen (z. B. in einem Internet-Café), bieten manche Systeme wie der (Yahoo! 2009) oder (Ignite Realtime Community 2009b) auch eine Web-Version an. Für die Verwendung einiger Web-Clients werden weitere Komponenten auf Client-Seite benötigt, z. B. ein installierter Flash-Player von Adobe oder die Aktivierung von JavaScript im Webbrowser.

6.1.3.2 Funktionalität

An dieser Stelle muss zwischen den *Kommunikationsdiensten* (den Kommunikationskanälen), die ein IM-Netz seinen Teilnehmern zur Verfügung stellt, und den Funktionen, die als *Konfigurationsoptionen* clientseitig angeboten werden, unterschieden werden.

Zur ersten Kategorie, den Kommunikationsdiensten, gehören Dienste wie das Versenden von Nachrichten (Chat-Nachrichten), der Aufbau einer Audio-/Video-Kommunikation oder Filesharing. Darüber hinaus werden je nach Anwendung auch Spiele, Desktop-Sharing oder weitere individuelle Applikationen angeboten (Farbe ■ orange, 3. Ebene in der Abbildung 8).

Im Bereich der Chat-Nachrichten gibt es außer der direkten Eins-zu-eins-Kommunikation zwischen Teilnehmern auch sogenannte Chatrooms, früher in Form von permanenten Chatrooms oder Channels (z. B. im IRC), heute als Konferenzschaltungen oder MUCs (Multi-User-Chats), die eine multidirektionale Kommunikation ermöglichen. Eine solche Gruppenkommunikation wird nicht nur in Form von Text-Nachrichten, sondern heute auch verstärkt in Form von Voice-Kommunikation mit Audioübertragung genutzt. Jeder kann die Nachrichten/Sprachdaten der anderen Teilnehmer empfangen, aber eine Sitzung wird von einem (oder mehreren) Gruppenleitern administriert. Diese haben – je nach Implementation – spezielle Rechte für die Koordination der Gruppenkonversationen (wie z. B. Sitzung anlegen, Sitzung schließen oder Personen aus dem Chatroom entfernen). Während in der textuellen Gruppenkommunikation die Teilnehmeranzahl häufig unbegrenzt ist, unterliegt die Voice- und Video-Kommunikation wegen technischer Einschränkungen (z .B. Bandbreite der übertragenden Netzwerke, Rechnerleistung für Audio-/Video-Kompression) einer starken Begrenzung bezüglich der Teilnehmeranzahl.

Zur zweiten Kategorie der Funktionen, den *Konfigurationsoptionen,* gehören die Erstellung einer Kontaktliste, die Verwaltung von Präsenzinformationen, Suchfunktionen und eine Historie der Kontakte. Die Kontaktliste („Buddylist“) ist eine Art Adressbuch für die Kontakte, die über dieses Medium erreichbar sind. Die Kontaktliste und die Mitteilung der eigenen Präsenzinformationen an die anderen Teilnehmer in dieser Liste ist eine der wichtigen Funktionen von Instant-Messaging-Systemen. Meldet sich ein Benutzer beim Server an, wird dies den anderen Teilnehmern, welche auch im System angemeldet sind, signalisiert. Je nach Anwendung ist die Verwaltung der Kontakte unterschiedlich: Manche erlauben eine Gruppierung der Kontakte auf Protokollebene, andere überlassen diese Funktionalität den Clients. So wird das Versenden einer Nachricht an unterschiedliche Personen durch eine Gruppennachricht – im Vergleich zum separaten Senden an viele einzelne Personen – erleichtert. Die Präsenz ist die Funktion, über welche die Erreichbarkeit eines Benutzers den anderen Benutzern mitgeteilt wird. Diese Option ist auch einer der Vorteile und Hauptmerkmale eines Instant-Messaging-Systems, da direkt erkennbar ist, ob und unter welchen Voraussetzungen ein gewünschter Partner erreichbar ist. Je nach Anwendung existieren unterschiedlich detaillierte Ausprägungen an Präsenzinformationen wie Verfügbarkeit des Teilnehmers, gewünschte Regeln zur Kontaktaufnahme („Bitte nicht stören“ oder „Skype me!“), Stimmungslage, Aufenthaltsort bei mobilen Teilnehmern und so fort.

Zu den optionalen Eigenschaften eines Instant-Messaging-Systems gehört auch die Erweiterbarkeit solcher Systeme. So bieten einige Systeme spezifizierte Schnittstellen, an denen weitere Funktionen in Form von sogenannten „Plug-ins“ angedockt werden können. Diese Möglichkeit nutzen zum Beispiel Drittanbieter, um, basierend auf der schon realisierten Infrastruktur des IM-Systems, weitergehende Funktionen wie Spiele oder Whiteboards zu implementieren.

6.1.4 IM-Server

Das Feature eines IM-Servers ist ein optionales Feature, denn wie im Kapitel 6.1.1 erläutert wurde, existieren auch Systeme, die ohne einen Server auskommen.

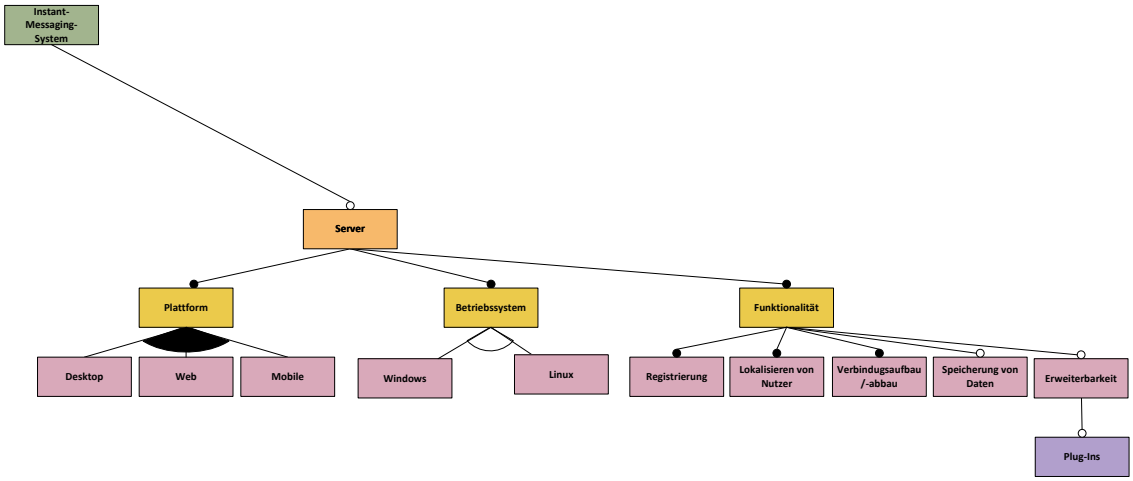

Abbildung 9 – Featuremodell - Server

Basiert die IM-Infrastruktur auf der Nutzung des Server-Modells, kann der Server unterschiedliche Aufgaben *(Funktionen)* realisieren. Man unterscheidet zwischen dem Authentifikationsserver, der den Log-in-Prozess jedes IM-Nutzers übernimmt, dem Präsenzserver, der die Informa-

tionen über den Status eines Kontaktes und die entsprechende Kontaktliste verwaltet, und dem Messenger-Server, der die Zwischenspeicherung und das Routing von Nachrichten erlaubt. Die *Betriebssysteme* der IM-Server decken das ganze Spektrum von Windows, Linux und Mac OS ab. Entwickelt man IM-Plug-ins für die IM-Clients, wird je nach Applikationstyp und Nachrichtenübertragung auch das entsprechende Plug-in für den Server unterstützt. Somit kann man die Funktionalität der Server *erweitern (Erweiterbarkeit)*.

6.1.5 IM-Sicherheit

Jedes IM-System bietet bezüglich Sicherheitsmechanismen zu mindeste einen Mechanismus für die Authentifikation der registrierten Benutzer (mandatorischer Mechanismus). Darüber hinaus können optional unterschiedliche Verbindungen der IM-Infrastruktur mit Mechanismen zur Vertraulichkeit geschützt werden. Dazu gehören z. B. eine SSL-Verbindung mit dem Authentifikationsserver oder eine Verschlüsselung der Peer-to-Peer-Kommunikation für die Chat-Nachrichten oder VoIP-Datenströme.

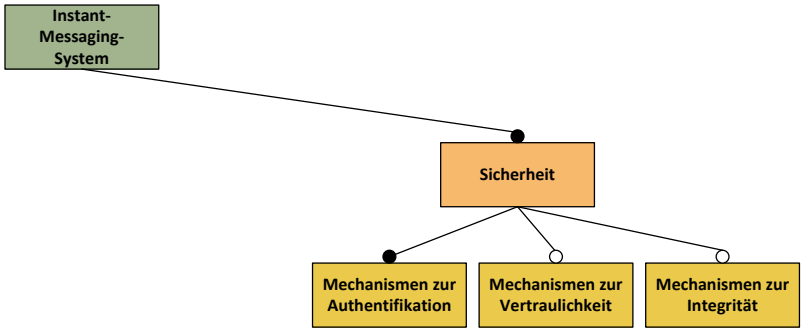

Abbildung 10 – Featuremodell - Sicherheit

Der erste Schritt für die Nutzung eines IM-Tools (nach dem Download des Tools) ist das Anlegen eines Accounts (Profils). Dafür wird eine Benutzerkennung mit Benutzername und Passwort für den Log-in-Server des Netzes angelegt. Die Benutzerkennung wird je nach Produkt entweder vom Benutzer gewählt oder vom System zugewiesen. Beispiele für die Art der Kennungen wurden in Kapitel 6.1.2 bei der Vorstellung der Protokollegenannt. Die Auswahl der Kennung ist beliebig, solange diese nicht schon für einen anderen Benutzer vergeben ist. Allerdings gilt, dass keine erkenntliche Zuordnung zwischen Personen und Kennungen existiert. Geht ein Benutzer nicht vorsichtig mit der Auswahl und Aufbewahrung seines Benutzernamens und Passworts um, kann jemand anderes diese Daten und den Zugang missbrauchen. Im Allgemeinen gilt, dass der Log-in-Server ein Verwaltungsknoten des Netzwerks ist, auf dem sich je nach Produkt detaillierte Informationen zu einem Profil oder auch zum aktuellen Status eines Kontaktes befinden. Die meisten Tools bieten ein automatisches Einloggen oder Speicherung der Kennung, um die Nutzung des IM-Systems zu vereinfachen. Da die Sicherheit der IM-Tools wesentlicher Teil dieser Arbeit ist, werden Kapitel 7 die weiteren Sicherheitsanforderungen sowie existierende Mechanismen betrachtet. Insgesamt ergibt sich also für die IM-Systeme folgendes Featuremodell:

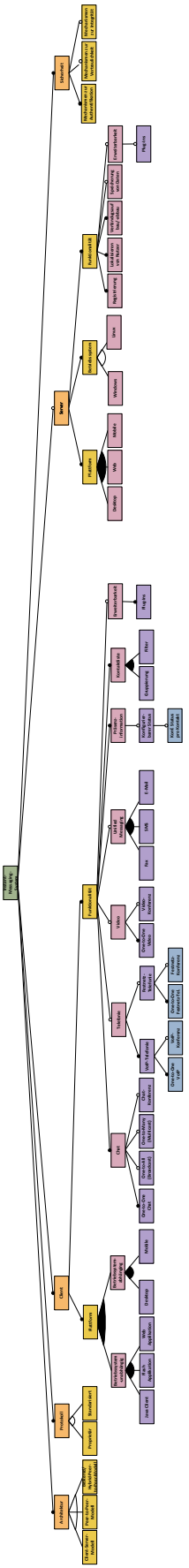

Abbildung 11 – Featuremodell - Überblick

7 Sicherheitsbetrachtungen der IM-Systeme

Ziel des vorliegenden Kapitels ist es, basierend auf den Grundlagen der IT-Sicherheit, eine Sicherheitsanalyse der Dienste der IM-Systeme durchzuführen und existierende Projekte und Sicherheitsmechanismen für die IM-Kommunikation zu identifizieren. Die Kernaufgabe lautet, offene Fragen bzgl. der Sicherheit und Erweiterung der Instant-Messaging-Systeme aufzuzeigen. Dies bildet auch die Basis für das Konzept einer sicheren, verbindlichen Kooperation über Unternehmensgrenzen hinweg, welches im dritten Teil dieser Arbeit realisiert wird.

7.1 IT-Sicherheit und ihre Anforderungen (Grundlagen der IT-Sicherheit)

Unter dem Begriff IT-Sicherheit wird eine Eigenschaft eines Systems verstanden, die die als bedeutsam angesehenen Bedrohungen, die sich gegen schützenswerte Güter richten, durch besondere Maßnahmen so weit ausschließt, dass das verbleibende Risiko akzeptiert wird (Amann und Atzmüller 1992). Je nach Anwendung und Umgebung gibt es unterschiedliche Anforderungen, die an ein System gestellt werden, um ein gewünschtes Sicherheitsniveau zu erreichen. Es gibt unterschiedliche Meinungen, welche minimalen Anforderungen ein System erfüllen muss, um dies als ein *sicheres System* zu bezeichnen. In den meisten Publikationen gehört die Erfüllung der Anforderungen „Vertraulichkeit“, „Integrität“ und „Verfügbarkeit“ zu den wichtigsten und minimalen Eigenschaften eines sicheren Systems. Diese drei Sicherheitsanforderungen sind auch als CIA-Triangle bekannt (confidentiality, integrity, and availability) (Common Criteria Recognition Agreement 2009; Bundesamt für Sicherheit in der Informationstechnik 2007; Avizienis et al. 2007). Darüber hinaus werden – je nach Ansicht der Forscher und Anwendungsgebiete – auch Eigenschaften wie Authentizität, Anonymität, Robustheit, Privatheit, Verbindlichkeit usw. gefordert (Avizienis et al. 2007; Grimm et al. 2007; Dierstein 2004a; Common Criteria Recognition Agreement 2009).

So ergänzt zum Beispiel (Dierstein 2004b, 2004a) die minimalen CIA-Anforderungen um die Zurechenbarkeit und Rechtsverbindlichkeit für die Definition des Begriffs der Sicherheit. Dabei unterscheidet er zwischen zwei Arten von Sicherheit: 1) Verlässlichkeit (Sicherheit des Systems) und 2) Beherrschbarkeit (Sicherheit vor dem System bzw. die Sicherheit der Betroffenen).

Die Verlässlichkeit eines Systems ist gegeben, wenn die Funktionen eines Systems aus technischer Sicht den gesetzten Anforderungen entsprechen. Hierzu gehören z. B. die Anforderungen Vertraulichkeit, Integrität oder Verfügbarkeit.

Die Beherrschbarkeit eines Systems dagegen ist aus Sicht des Betroffenen zu betrachten. Ein System ist beherrschbar, wenn seine Funktionen und Ergebnisse der auslösenden Instanz zugeordnet werden können und nachweisbar sind. Hierzu gehören die Anforderungen der Zurechenbarkeit oder Rechtsverbindlichkeit. Diese Art der Zuordnung ist keine starre Untergliederung der skizzierten Begriffe. Vielmehr soll hier ein Verständnis dafür erzeugt werden, dass der Begriff Sicherheit sowohl als technischer Begriff für die Systeme agiert als auch die Konsequenzen für die Umgebung darlegt.

Eine andere Definition der Sicherheit und ihrer Anforderungen kommt von den englischsprachigen Communities. Eine Zusammenfassung bieten (Avizienis et al. 2007). Sie führen die

Begriffe Sicherheit (security) und Verlässlichkeit (dependability) ein. Während unter Sicherheit die klassischen CIA-Anforderungen verstanden werden, ist der Begriff der Verlässlichkeit umfangreicher. Er umfasst die Sicherheitsziele Verfügbarkeit (availability), Integrität (integrity), Zuverlässigkeit (reliability), Sicherheit (safety), und Wartbarkeit (maintainability).

Diese Gegenüberstellung der unterschiedlichen Sichtweisen soll die Vielfältigkeit des Begriffes Sicherheit darstellen. Unabhängig von der Einordung der Anforderungen oder deren Klassifizierung, werden die einzelnen Anforderungen von allen Forschern ähnlich definiert. Es wird an dieser Stelle davon abgesehen, alle möglichen Sicherheitsanforderungen zu definieren. Stattdessen werden die Anforderungen definiert, die für die vorliegende Arbeit relevant sind (Eckert 2009; Dierstein 2004b; Avizienis et al. 2007; Grimm 1994).

- Integrität: Es darf keine unbefugte, nicht erkannte Änderung der Daten oder Funktionen des Systems stattfinden. Ein Datum ist deshalb nur dann integer, wenn an ihm nur zulässige Veränderungen vorgenommen wurden. Dies gilt sowohl für die Übertragung der Daten als auch für die Verarbeitung an den jeweiligen Netzkomponenten.
- Verfügbarkeit: Die Verfügbarkeit eines Systems bezeichnet die Eigenschaft, ein funktionsfähiges System (= erwartete oder geforderte Form und Qualität des Systems) zum geforderten Zeitpunkt für authentifizierte und autorisierte Subjekte anzutreffen.
- Vertraulichkeit: Ein System gewährleistet Vertraulichkeit, wenn die in ihm enthaltenen Informationen nur berechtigten Subjekten zur Verfügung stehen. Eine Information ist vertraulich, wenn sie nur berechtigten Subjekten zur Kenntnis gelangt.
- Verbindlichkeit: Verbindlichkeit ist die Eigenschaft eines Versprechens oder einer Anweisung, dass seine Erfüllung bzw. ihre Ausführung unter gesellschaftlicher Kontrolle steht.
- Authentizität: Die Authentizität ist erfüllt, wenn gleichzeitig Originalität (= Übereinstimmung der angeblichen Identität mit der tatsächlichen Identität) und Integrität erfüllt sind.

Ob eine Sicherheitsanforderung bei der Realisierung oder Überprüfung eines IT-Systems betrachtet werden soll, hängt vom Anwendungsbereich und der Systemumgebung ab. Zur Spezifizierung wird eine Sicherheitsanalyse durchgeführt. Die Vorgehensweise resultiert aus der oben genannten Definition zur IT-Sicherheit nach (Amann und Atzmüller 1992) und wird in der folgenden Abbildung 12 – Sicherheitsanalyse (in Anlehnung an (Bundesamt für Sicherheit in der Informationstechnik 1999; Grimm et al. 2007)) dargestellt:

Die Werte eines Systems (Anwendung oder Unternehmen) werden identifiziert und alle Bedrohungen bzw. Angriffe, die sich gegen die Werte richten, betrachtet. Als Werte sind technische, organisatorische oder personelle Komponenten eines Systems (Unternehmens) zu verstehen. Typische Angriffe sind das Belauschen einer Kommunikation, das Verändern von Daten oder die unerlaubte Nutzung von Ressourcen. Durch die Zuordnung von Werten und Bedrohungen (Bedrohungsanalyse) werden die Sicherheitsanforderungen für die zu betrachtenden Werte definiert. Anschließend werden die Sicherheitsmaßnahmen und -mechanismen ausgewählt, die die geforderten Sicherheitsanforderungen unterstützen.

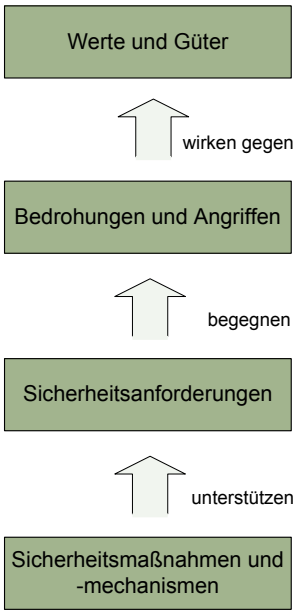

Abbildung 12 – Sicherheitsanalyse (in Anlehnung an (Bundesamt für Sicherheit in der Informationstechnik 1999; Grimm et al. 2007))

In den nächsten Kapiteln werden diese Sicherheitsbetrachtungen konkret auf das Beispiel der IM-Systeme bezogen. Das Ziel dabei ist es, den Stand der Forschung in diesem Bereich darzustellen und die Forschungslücken für die vorliegende Arbeit zu identifizieren.

7.2 Sicherheitsanalyse der Instant-Messaging-Systeme

Entsprechend des Sicherheitsanalyse-Schemas aus dem vorherigen Kapitel, sollen in diesem Kapitel die Werte eines Instant-Messaging-Systems, entsprechende Bedrohungen und existierende Mechanismen vorgestellt werden (vgl. Abbildung 12)

7.2.1 Werte/Güter

Als zu schützende Werte gelten:

- die IM-Kommunikation – Diese beinhaltet die übertragenen Daten zwischen Sender (Client oder Server) und Empfänger (Client oder Server), unabhängig vom Kommunikationskanal (Chat-Nachrichten, VoIP-Nachrichten, Video over IP-Nachrichten). Es wird bei jedem IM-Dienst zwischen den Signalisierungsdaten für einen Dienst und den Inhaltsdaten (z. B. verschickte Chat-Nachrichten) unterschieden. In die letzte Kategorie fallen auch die Daten anderer Anwendungen, die eine IM-Architektur als Kommunikationsbasis verwenden, wie im Rahmen dieser Arbeit die Daten des elektronischen Wahlprozesses (vgl. Kapitel 3.2 und Kapitel 9).
- der IM-Client – Hier geht es um die Originalität des Clients und der Daten, die er verwaltet wie. Kontaktlisten, Passwörter usw. Als wichtigste Daten gelten die Authentifizierungsdaten (Kennung und Passwort), denn diese sind sensible Pflichtdaten (Angaben), über die jeder Benutzer verfügen muss, wenn er den Dienst eines IM-Systems in Anspruch nehmen möchte. Die Manipulation dieser Daten (z. B. durch Ausspähen) hätte nicht nur Auswirkungen auf den „Wert“ des IM-Clients sondern auch auf die IM-Kommunikation an sich.

Eine vertrauliche und integere Kommunikation wäre somit nicht möglich. Der Schutz anderer Daten, die auf dem Client gespeichert sind, wie die Kontaktliste oder die Daten zum persönlichen IM-Profil des Nutzers sind wichtig für die Gewährleistung der Privatsphäre des Nutzers.

- der IM-Server – Von der Funktionsbereitschaft und Verfügbarkeit des Servers sind, je nach System, die Verfügbarkeit der profilbezogenen Daten eines Nutzers sowie der Daten zum Aufbau einer Kommunikation und die Erreichbarkeit von Kontakten abhängig.
- der Client – Er ist das System auf dem die IM-Software läuft, z. B. der Rechner oder PDA einen Benutzers.

7.2.2 Bedrohungen und Angriffe

Bedrohungspotenzial für eine Instant-Messaging-Anwendung kann man in jeder Ebene des TCP/IP-Stacks finden (Leavitt 2005; Mannan und van Oorschot 2005). In Abbildung 13 ist als Orientierung der TCP/IP-Stack mit konkreten Protokollen einer IM-Infrastruktur angegeben.

Auf der Netzwerkschicht sind MAC-Spoofing-Attacken für ankommende Verbindungen denkbar, welche als mögliche Voraussetzung für Lauschangriffe dienen können. Ein ARP-Spoofing ist auf dieser Ebene ebenfalls möglich, wodurch bestimmte Teilnehmer unerreichbar werden.

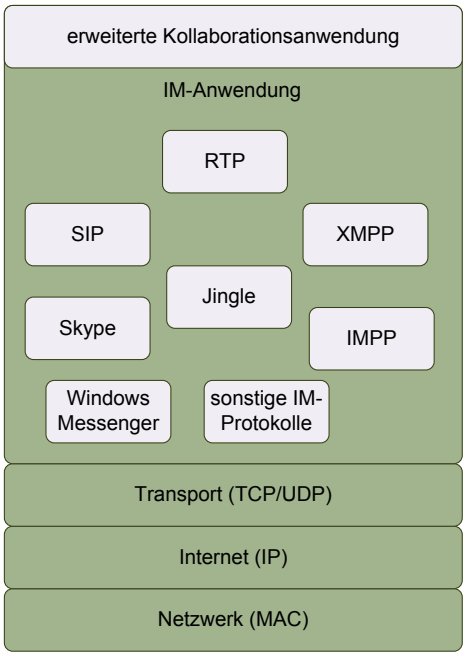

Abbildung 13 – TCP/IP-Stack

In der Internetschicht ist ein IP-Spoofing-Angriff denkbar, indem ein Angreifer die IP-Pakete mit einer vorgetäuschten Quell-IP-Adresse an einen Zielrechner sendet. So kann eine Kommunikationsbeziehung aufgezwungen und entsprechend ein IM-Client oder IM-Server angegriffen werden. Ein Man-in-the-Middle-Angriff ist durch das Abhören und Verändern der Angaben im Header der IP- und ICMP-Pakete bei der IM-Kommunikation ebenfalls realisierbar.

In der Transportschicht sind Portscanning-Angriffe möglich, um die Nutzung bestimmter IM-Systeme und -Dienste ausfindig zu machen und in einem zweiten Schritt mit bestimmten Schadensprogrammen (wie z. B. Backdoors) die Schwachstellen dieser Anwendungen auszunutzen.

In der Anwendungsschicht sind protokollabhängige Angriffe ausführbar. So sind für das Signalisierungsprotokoll SIP (vgl. Kapitel 6.1.2.5) Angriffe bekannt, die zur Unterbrechung einer bestehenden Verbindung oder einer gerade initiierten Verbindung führen. Dies ist durch Versendung einer CANCEL-Nachricht oder einer BUSY-Nachricht z. B. durch eine Malware durchführbar. Ein URI-Spoofing, d. h. die falsche Adressierung eines Kontaktes bei der Nutzung des SIP, ist genauso möglich (funktioniert ähnlich einem URL-Spoofing bei Webseiten). Bei der Nutzung des RTP (Kapitel 6.1.2.7) für die Übertragung von Mediadaten sind Man-in-the-Middle-Attacken bekannt. Durch die sogenannte RTP-Insertion bzw. RTCP-Insertion kann der Angreifer fiktive RTP- oder RTCP-Pakete zu einer RTP-Session hinzufügen. Das Abhören und die Aufzeichnung von Mediadaten ist mit geeigneten Tools ebenfalls realisierbar (Badach 2007; Eren und Detken 2007; Piccard et al. 2005; Sicker und Lookabaugh 2004).

Auf dieser Schicht sind weitere Schadprogramme zu erwähnen, die speziell für manche IM-Systeme implementiert wurden und entsprechende Schwachstellen ausnutzen. Deren negative Auswirkungen können nicht nur das jeweils befallene IM-System, sondern den ganzen Client (Gerät des Nutzers) beeinflussen. Beispiele hierfür sind der W32.Simic.Worm für Windows Messenger, der Komponenten von Visual Basic installierte, und der W32.Upering.Worm, der das Adressbuch der AOL AIM nutzt, um sich zu verbreiten (Stone und Merrion 2004; Piccard et al. 2005; Leavitt 2005; Williamson, Parry und Byde 2004). Für das IM-System Skype wurde ein Trojaner veröffentlicht, mit dessen Hilfe angeblich die Audio-Daten der Gespräche heimlich mitgeschnitten und als MP3-Dateien auf einen externen Server geladen werden (Schmidt 2009).

Die Verbreitung von solchen Schadprogrammen findet zusätzlich zu den üblichen Wegen (Besuch von bösartigen Internetseiten oder unvorsichtiger Download von Programmen) über sogenannte SPIM-Dienste statt. Diese Dienste – Spam over Instant Messaging – nutzen Benutzerkennungen, um unerwünschte Nachrichten zu verschicken und Phishing-Attacken oder die Installation von Dialern bzw. Trojanern zu erzwingen. Ähnlich wie bei Spam-E-Mail-Nachrichten werden über die Kontaktlisten oder die Suchfunktion eines IM-Systems Benutzernamen identifiziert und entsprechende Nachrichten versendet (Perey 2004). Die Vorteile für einen Angreifer bei einem SPIM-Angriff im Vergleich zu einem Spam-Angriff bestehen darin, dass diese Art des Angriffes noch nicht so bekannt ist (außer vielleicht SPIM als Werbemedium) und viele Benutzer aus Interesse und Gewohnheit schneller auf eine IM-Nachricht antworten als auf eine E-Mail-Nachricht.

Weitere Bedrohungen sind unter den Begriff „organisatorische Bedrohungen“ zu fassen. Hierzu gehört zum Beispiel der unvorsichtige Umgang mit den Benutzerdaten (z. B. Auswahl eines leichten Passwortes oder die Weitergabe des Passwortes). Ein weiterer Angriff, der auf den unvorsichtigen Umgang mit dem IM-System oder auf das Software-Design solcher Systeme zurückzuführen ist, ist die richtige Wahl eines Benutzernamens und die fehlende Zuordnung zwischen Person und Benutzerprofil. Möchte ein Benutzer einen gewünschten Gesprächspartner erreichen, kann er über die angebotene Suche nach seinem vermeintlich richtigen Partner suchen. Seine Trefferquote hängt von der Bereitschaft des Kontaktes ab, genügend Informationen

über seine Person zur Verfügung zu stellen. Diesen Umstand kann der Angreifer ausnutzen und den entsprechenden Kontaktnamen auswählen. So kann er z. B. Alice123 statt Alice231 wählen, um einen IM-Nutzer zu verwirren und an geheime Daten in Form von Chat-Nachrichten oder Filesharing zu gelangen. Lösungen für diesen Angriff werden in Kapitel 8.2.1 und Kapitel 11 vorgestellt.

Die oben genannten Bedrohungen gelten sowohl für die Nutzung der Instant-Messaging-Systeme als einfaches Kommunikationsmittel als auch für eine darauf basierende Anwendung. Denn diese neue Anwendung (z. B. elektronische Entscheidungsprozesse) nutzt die IM-Architektur (vgl. Abbildung 13) und ist anfällig sowohl für Angriffe auf die Basis als auch für spezielle Angriffe auf die Anwendung an sich.

7.2.3 Sicherheitsanforderungen

Durch den Einzug von Instant-Messaging-Systemen auch in Unternehmen und Institutionen (Quan-Haase 2008; Gartner Group 2007) gewinnt der sichere Umgang mit dem System und den darin übertragenen Daten mehr und mehr an Bedeutung. Während am Anfang der IM-Ära die Überlegungen bzgl. sicherer Kommunikation höchstens in Prototypen von Forschungsinstituten und Universitäten realisiert wurden, bieten mittlerweile immer mehr IM-Systeme Sicherheitsmechanismen für die Erfüllung bestimmter Sicherheitsanforderungen an (McCullagh 2008).

Die Sicherheitsanforderungen, die man an eine sichere Kommunikation und Kollaboration stellen könnte, sind:

- Vertraulichkeit – Ziel ist der vertrauliche Austausch von Chat-Nachrichten genauso wie von Telefonaten (Voice over IP) oder Video-Konferenzen (Video over IP). Die Benutzerdaten, Konfigurationsangaben oder Kontaktlisten sollen vertraulich in einem IM-Client oder -Server verwaltet werden.
- Integrität – Hierbei handelt es sich um die Anforderung, die oben genannten Daten unverändert verschicken zu können. Die Integrität bezieht sich sowohl auf die Signalisierungsdaten, die für die Etablierung einer Verbindung benötigt werden, als auch auf die Inhaltsdaten.
- Authentizität – Mit Hilfe von Authentifikationsmechanismen soll die Authentizität der Sender- und Empfänger-Identitäten überprüft werden.
- Verfügbarkeit – Der Abbruch der Verbindung zum IM-Server oder zu einem IM-Kommunikationspartner sollte mit geeigneten Mechanismen unterbunden werden. DOS-Attacken oder die Versendung von SPIM-Nachrichten sind so minimal wie möglich zu halten, um einen reibungslosen Ablauf einer Kollaboration gewährleisten zu können.
- Verbindlichkeit – Im Rahmen einer Kommunikation/Kollaboration bedeutet die Einhaltung der Verbindlichkeit einerseits, dass Aussagen, die ein Teilnehmer verbindlich abgibt, durchsetzbar sind (Verbindlichkeit der Befehle), und andererseits, dass deren Beweisbarkeit mit einem Ursprungs- und Integritätsbeweis ausgestattet sein sollte (Verbindlichkeit des Versprechens).
- Abstreitbarkeit – Aber auch die explizite Nicht-Einhaltung der Verbindlichkeit kann zugunsten der Privatsphäre (Privatheit, Privacy) eine der gewählten Sicherheitsanforderungen

sein, z. B. wenn nach der Beendigung einer Kommunikation die Inhalte nicht mehr bestimmten Personen zuzuordnen sind.

- Anonymität/Pseudonymität – Je nach Einsatzbereich des IM-Systems kann z. B. für den Bereich E-Participation oder Fürsorge Personen die Gelegenheit gegeben werden, anonym mit den entsprechenden Stellen Kontakt aufzunehmen. Bezüglich einer Gruppenkollaboration kann diese Anforderung nur für einen Teil der Kollaboration gesetzt werden, z. B. wie bei der Durchführung einer elektronischen Wahl über ein IM-System. Denn hier muss am Anfang der Kommunikation zwar die Authentizität der Teilnehmer überprüft werden, aber für die eigentliche Wahl ist die anonyme Abgabe einer Stimme wünschenswert.

Für die Sicherstellung dieser Anforderungen bietet die Forschung im Bereich der IT-Sicherheit unterschiedliche Mechanismen an wie Verschlüsselungsverfahren für die Gewährleistung der Vertraulichkeit, digitale Signaturen oder Hashfunktionen für die Erfüllung der Integrität oder den Einsatz von Firewalls oder Antiviren-Programmen auf Applikations-Ebene für die Gewährleistung der Verfügbarkeit oder Privatheit. An dieser Stelle werden keine allgemeinen Sicherheitsmechanismen vorgestellt, diese können unter (Raymond 2004; Whitman und Mattord 2004; Harris 2007; Eckert 2009) nachgelesen werden. Hier werden auf den nächsten Seiten als Erstes Mechanismen erläutert, die im Rahmen von Standards oder bereits verwendeten IM-Lösungen realisiert wurden (siehe nächstes Kapitel 7.2.4). Im späteren Kapitel 8 werden weitere Forschungsprojekte vorgestellt, die sichere IM-Alternativen darstellen.

7.2.4 Sicherheitsmaßnahmen und -mechanismen

Betrachtet man die IM-Infrastruktur zweier Teilnehmer (vgl. Abbildung 14) gibt es außer den IT-Instanzen (Client und Server) auch drei Arten von Verbindungen, die mit Hilfe von Sicherheitsmechanismen bestimmte Sicherheitsanforderungen erfüllen sollen:

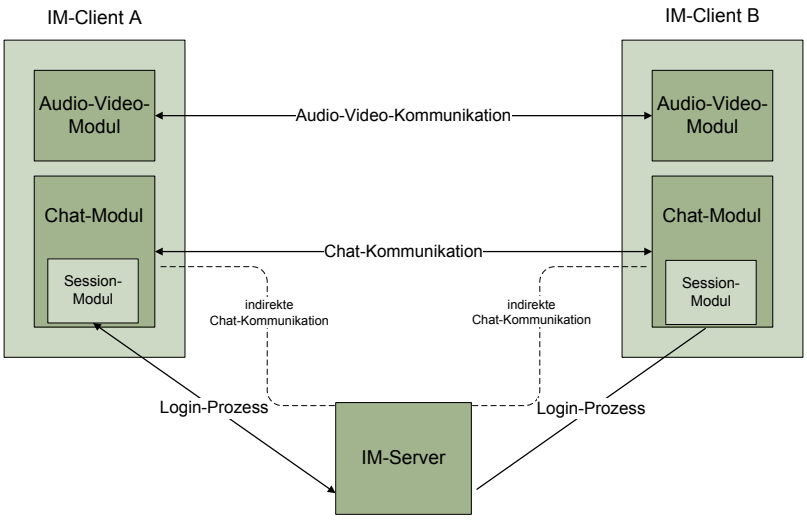

Abbildung 14 – IM-Infrastruktur

- Verbindung Client–Server, die z. B. während der Registrierung, Authentifikation eines Benutzers oder bei der Übertragung von Chat-Nachrichten oder Signalisierungsdaten entsteht.
- Verbindung Server–Server, die z. B. dann hergestellt wird, wenn unternehmensspezifische IM-Lösungen miteinander kombiniert werden (ist nicht in Abbildung 14 enthalten).
- Verbindung Client Client, die z. B. bei der Übertragung von Chat-Nachrichten aufgebaut wird.

Für die Absicherung dieser IM-Verbindungen gibt es Mechanismen, die aus der allgemeinen IT-Sicherheitsforschung stammen, wie z. B. die Verwendung einer SSL-Verbindung oder der Aufbau eines Virtual Privat Networks für die sichere Übertragung von Authentifikationsdaten. Die Verwendung von speziellen Protokollen, die z. B. aus dem Bereich der VoIP-Security stammen, eignet sich für die Absicherung von Nachrichten in einem Instant-Messaging-System. In diesem Kapitel werden die Absicherungskonzepte vorgestellt, die die aktuellen IM-Protokolle aus Kapitel 6.1.2 realisieren.

7.2.4.1 Chat-Protokolle (OSCAR, TOC, MSNP)

Ein Blick auf die gängigen Instant-Messaging-Systeme, die hauptsächlich aus dem Bereich der Chat-Systeme stammen, macht die mangelhafte Realisierung von Sicherheitsmaßnahmen deutlich.

Einzig die Übermittlung des Passworts während der Anmeldung auf dem IM-Server erfolgt je nach Implementierung mit einer XOR-Verknüpfung, dem MD5-Algorithmus oder sogar mit einer SSL-Verbindung verschlüsselt. Alle übertragenen Nachrichten zwischen den Teilnehmern sind durch die fehlende Existenz eines Verschlüsselungsmechanismus leicht abhörbar. Auch die Verletzung der Datenintegrität und der Authentizität der Sender ist durch Man-in-the-Middle-Attacken unter Ausnutzung von Session-Hijacking oder Spoofing-Tools und -Methoden möglich. Viele dieser Systeme übertragen die Parameter jeder Sitzung im Klartext. Ist einmal die eigene IP-Adresse bekannt, kann der Angreifer dies nutzen, um einen Angriff auf die Verfügbarkeit des Systems zu starten (z. B. DOS-Angriff). (Rittinghouse und Ransome 2005; Schlildt 2005; Piccard et al. 2005)

Viele IM-Systeme und IM-Hersteller „lösen“ diese Probleme, indem sie Nutzungspolicies veröffentlichen, dass über diese Systeme keine vertrauliche Kommunikation stattfinden soll. Dazu empfehlen sie bestimmte Einstellungen für die Systeme wie z. B. nur die Kontaktanfragen bekannter Personen zu erlauben. Darüber hinaus existieren Drittanbieter wie z. B. Trillian (Cerulean Studios 2009), die z. B. Verschlüsselungs-Plug-ins für verschiedene IM-Systeme anbieten. Allerdings müssen beide Kommunikationspartner über den gleichen Typ des Accounts und die gleiche Konfiguration verfügen, z. B. beide Teilnehmer sind z. B. über einen ICQ-Account eingeloggt. Eine Verschlüsselung zwischen unterschiedlichen Systemen ist nicht möglich.

7.2.4.2 Skype

Skype ist eines der wenigen IM-Tools, welches nach eigenen Angaben starke Sicherheitsmechanismen für alle Kommunikationskanäle anbietet. Allerdings basieren die Erkenntnisse über sein Sicherheitsniveau durch das proprietäre und nicht offen gelegte Protokoll auf den Angaben von Sicherheitsexperten der Firma Skype Technologies (Skype 2009) oder sonstigen Protokoll-

analysen (Ehlert et al. 2006; Baset und Schulzrinne 2006). Demnach wird durch die Verwendung von Zertifikaten und des symmetrischen Verfahrens (AES) die Authentifikation, Vertraulichkeit und Integrität sowohl für die Verbindung zwischen Client und Server als auch zwischen Client und Client realisiert. Bei der ersten Registrierung eines Benutzers auf dem Authentifikationsserver von Skype werden ein eindeutiger Benutzername und das gewählte Passwort (Letzteres als Hashwert mit SHA1) mit dem öffentlichen Schlüssel des Skype-Servers verschlüsselt. Dieser öffentliche Schlüssel wird bei der Installation des Skype-Clients installiert. Der Skype-Server entschlüsselt die empfangenen Daten mit seinem privaten Schlüssel und speichert diese für die weiteren Anmeldungen. Während jeder Anmeldung eines Nutzers im Skype-Netz findet grob folgender Ablauf statt: Die Client-Software generiert ein 1024-bit-Schlüsselpaar (privater und öffentlicher Schlüssel). Weiterhin wird clientseitig ein symmetrischer AES-Schlüssel (256 bit) erzeugt. Mit diesem symmetrischen Schlüssel werden der Benutzername, das Passwort (Hashwert mit SHA1) und der generierte öffentliche Schlüssel verschlüsselt und an den Server übertragen. Letzterer entschlüsselt die Daten und vergleicht diese mit den durch die Registrierung gespeicherten Daten. Ist die Überprüfung der Daten vom Server erfolgreich, wird der öffentliche Schlüssel und der Benutzername des Clients signiert (Erstellung von Zertifikaten). Der Server schickt dieses Zertifikat an den Client/Benutzer zurück. Möchten jetzt zwei Benutzer miteinander kommunizieren, werden diese Zertifikate ausgetauscht. Dies passiert in einem Challenge-Response-Verfahren von beiden Seiten. Ist diese gegenseitige Authentifikation erfolgreich, wird ein 256-bit-Sitzungsschlüssel (AES) zwischen den beiden Teilnehmern verabredet. Zur Erstellung dieses Schlüssels trägt jeder Client 128 bit bei. Dafür generiert jeder Teilnehmer einen 128-bit-Schlüssel, verschlüsselt diesen mit dem öffentlichen Schlüssel des Gegenübers und sendet diesen seinem Gesprächspartner zu. Der zusammengeführte 256 bit generierte Schlüssel wird für die Verschlüsselung von Voice-, Video- oder Chat-Nachrichten verwendet (Berson 2005; Baset und Schulzrinne 2006; Ehlert et al. 2006; Biondi und Desclaux 2006).

Leider bleiben an dieser Stelle viele Fragen offen: z. B., wie genau werden die Schlüssel generiert? Sind die Zertifikate von den Geräten, auf denen eine Skype-Version läuft, abhängig? Sind es X.509-Zertifikate? Kann ein Benutzer die Zertifikate überprüfen? Wie ist es mit der Gefährdung durch eine Schadenssoftware, die die Generierung der Schlüssel clientseitig manipuliert?

Immer wieder wird die Frage gestellt, ob überhaupt eine Verschlüsselung verwendet wird und warum das Verfahren nicht veröffentlicht wird, wenn es so gut funktioniert. (Garfinkel 2005) Zudem gibt es Veröffentlichungen über die gewollte Nutzung von Trojanern innerhalb von Skype, welches bestimmten Institutionen das Abhören der Kommunikation erlaubt (Krempl und Kuri 2008; Schmidt 2009). Die unerlaubten Abfragen von Rechnerdaten, wie z. B. Angaben zum Bios oder Motherboard eines Rechners, werden den Funktionen dieses Skype-Trojaners ebenso zugeschrieben (pagetable.com 2006; Schmidt 2006).

7.2.4.3 XMPP

Das XMPP bietet aufgrund der Offenheit des Protokolls und der Mitwirkung vieler Beteiligter unterschiedliche Mechanismen, um gesetzte Sicherheitsanforderungen zu gewährleisten (xmpp.org 2009; Saint-Andre, Smith und Tronçon 2009):

SSL/TLS: Das Konzept SSL/TLS (Secure Socket Layer/Transport Layer Security,), welches seinen Ursprung im Aufbau von sicheren HTTP-Verbindungen hat, hat die Aufgaben der Au-

thentifikation von Kommunikationspartnern, des Aufbaus einer vertraulichen Verbindung zwischen zwei Punkten und der Sicherstellung der Integrität. Dieses Verfahren basiert auf der Verwendung von Zertifikaten und der Nutzung eines gemeinsamen Sitzungsschlüssels (Freier, Karlton und Kocher 1996; Dierks und Rescorla 2008). XMPP bietet die Verwendung von SSL sowohl in der Kommunikation zwischen Client und Server als auch zwischen Servern. Während der Registrierung oder Anmeldung werden die Daten für die Überprüfung der Identität eines Benutzers (Benutzername, Passwort) damit verschlüsselt übertragen und gleichzeitig wird durch das Zertifikat des Servers die Echtheit des Servers überprüft. Wurde eine TLS-Verbindung erfolgreich aufgebaut, kann ein Angreifer diese nur schwer kompromittieren. Die einzige Möglichkeit für einen Angriff besteht beim Verbindungsaufbau durch die Nutzung von falschen Zertifikaten. Dies ist aber ein Schwachpunkt von SSL, unabhängig vom Anwendungsgebiet.

SASL: Eine weitere Möglichkeit ist die Verwendung des Simple Authentication and Security Layer Frameworks (Myers 1997; Melnikov und Zeilenga 2006). Dieses Framework bietet konkrete Sicherheitsmechanismen, um entsprechende Mechanismen in Applikationen einsetzen zu können, ohne sie jedes Mal neu zu implementieren. Diese sind:

- PLAIN – Die Übertragung von Benutzername und Passwort wird in Plain-Text übertragen, also ohne jegliche Sicherheitsmechanismen. Dies wird nur dann realisiert, wenn vorher eine TLS-Verbindung aufgebaut wurde.
- DIGEST-MD5 – Dieses Verfahren basiert auf der HTTP Digest Authentication. Während des Authentifikationsprozesses werden die Daten des Clients (Benutzername, Passwort) in Form eines Challenge-Response-Verfahrens mit Hash- und Verschlüsselungsalgorithmen (MD5) verschlüsselt an den Server übertragen. Der Server sendet einen zufälligen String an den Client. Dieser berechnet aus dem String und seinem Passwort einen Hashwert, den er zurück an den Server übermittelt. Er berechnet erst selbst den Hashwert (anhand der gespeicherten Daten) und vergleicht diesen dann mit dem empfangenen Wert (Leach und Newman 2000). Somit wird die Vertraulichkeit und Integrität der Daten gewährleistet, da einerseits die Daten verschlüsselt übertragen werden und andererseits nur derjenige, der die Daten kennt, den richtigen Hashwert berechnen kann.
- EXTERNAL – Dieser Mechanismus erlaubt die Verwendung einer etablierten, sicheren Verbindung mittels IPSec oder SSL/TLS. Das bedeutet, der Schutzmechanismus findet außerhalb des Frameworks statt.
- GSSAPI – Der Generic Security Service Application Program Interface-Mechanismus stellt weitere Authentifizierungsmechanismen innerhalb des SASL zur Verfügung. Das bekannteste unterstützende Verfahren hierbei ist das Kerberos-Verfahren (Zhu, Jaganathan und Hartman 2005). Dadurch kann die Identität des Senders und die Integrität der übertragenen Daten sichergestellt werden. Der Authentifikationsdienst wird von einem vertrauenswürdigen Server übernommen (trusted third party), auf dem die benötigten Daten für die Überprüfung einer Identität gespeichert sind. Kerberos basiert auf symmetrischen Schlüsseln, die der Client sicher aufbewahrt. Es wird in Single-Sign-On-Umgebungen so eingesetzt, dass unter Angabe nur einer Kennung ein Benutzer Zugang zu unterschiedlichen Systemen erhalten kann (Harris 2007; Eckert 2009).

- ANONYMOUS – Mit diesem Mechanismus kann die Nutzung des Systems ohne das Anlegen eines Profils gewählt werden, wie z. B., wenn Unternehmen den Kunden die Möglichkeit bieten, anonym mit ihnen zu kommunizieren (Zeilenga 2006).

Diese SASL-Mechanismen können theoretisch auch für die Kommunikation zwischen Servern eingesetzt werden. Dies geschieht innerhalb von Unternehmensgrenzen, in denen Zertifikate oder die notwendigen Authentifikationsdaten auf den Servern hinterlegt werden können. Geht es um die Verbindung zwischen Servern unterschiedlicher Unternehmen, wird für die Absicherung der Kommunikation von Teilnehmern aus verschiedenen Domänen eine Kombination der EXTERNAL-Methode und TLS eingesetzt. Ist der Austausch von Zertifikaten nicht möglich, wird eine schwächere Identitätsüberprüfung des Servers namens „Server Dialback“ vorgenommen. In diesem Fall kann der Empfänger die Korrektheit des DNS-Namens des Senders überprüfen. Dies soll verhindern, dass ein Angreifer eine Verbindung zum Empfänger aufbaut und sich als ein anderer Server ausgibt. Dieser unidirektionale Mechanismus kann nur zwischen Servern verwendet werden und basiert auf dem Austausch von symmetrischen Schlüsseln zwischen Sender- und Empfänger-Server nach einer DNS-Überprüfung. (Miller, Saint-Andre und Hancke 2009)

S/MIME: Dieser Mechanismus hat seinen Ursprung im MIME-(Multipurpose Internet Mail Extension-)Standard, welches ein Verfahren für die Kodierung und den Aufbau von Nachrichtenformaten ist. S/MIME bietet eine sichere Ende-zu-Ende-Verbindung durch die Erfüllung der Vertraulichkeit, Integrität und Authentifikation der involvierten Parteien (Ramsdell 2004). XMPP unterstützt zwar diesen Mechanismus, allerdings ist diese Methode aufgrund der Komplexität und der fehlenden Existenz von Zertifikaten in den Clients nicht besonders verbreitet (Saint-Andre 2004b).

OpenPGP: In der Erweiterung XEP-0027 Current Jabber OpenPGP Usage der XMPP Standards Foundation (XSF) des XMPP wird die Möglichkeit geboten, mit Hilfe von OpenPGP und der Einbindung von existierenden Schlüsselpaaren die Vertraulichkeit, Authentizität und Integrität sicherzustellen. (Muldowney 2006)

7.2.4.4 SIP

SIP bietet in der Grundausstattung des Protokolls keine Sicherheitsmechanismen an, allerdings gibt es Vorschläge und Protokollerweiterungen, die einen sicheren Einsatz von SIP erlauben (Badach 2007; Thermos und Takanen 2007; Eren und Detken 2007; Dunte und Ruland 2007; Bilien et al. 2005).

HTTP-Digest-Authentification: Dieses Verfahren wird für die Registrierung und Initialisierung einer Sitzung verwendet. So kann es z. B. vom SIP-Server (registrar) an den Teilnehmer einer Challenge (Nonce-Zahl genannt) gesendet werden, um diesen zu authentifizieren. Der Teilnehmer antwortet mit einem MD5-Hashwert anhand des Benutzernamens, des Passwortes, der gesendeten Zahl und des Servers. Ist die Authentifizierung erfolgreich, aktualisiert der Server die gespeicherten Daten des Nutzers (IP-Adresse oder URI) und kann mit dem Aufbau einer Session mit dem gewünschten Kommunikationspartner starten. Da der Benutzername des zu erreichenden Servers, die Nonce-Zahl und die Art des Hash-Algorithmus klar übertragen werden, kann ein Angriff nicht ausgeschlossen werden. Ein solches Verfahren kann nur mit Benutzern einer Domäne durchgeführt werden. Diese Methode wird je nach Implementation bei der Re-

gistrierung, Initiierung oder dem Abbau einer Sitzung verwendet. Um Replay-Attacken oder eine Maskierung von Nachrichten zu vermeiden, müsste man allerdings nicht nur die Registrierung, sondern alle Nachrichten, die Bestandteil einer sind, mit diesem Verfahren schützen (Franks et al. 1999).

IPsec: Eine weitere Möglichkeit ist es, das SIP innerhalb eines VPN (Virtual Privat Network) zu nutzen. IPsec bietet einen Mechanismus für die Gewährleistung der Vertraulichkeit, Integrität und Authentizität zwischen zwei Instanzen (Kent 1998). Diese können zwei Netze oder ein Client und ein Netz sein, z. B. wenn ein Außendienstmitarbeiter Informationen aus dem Unternehmensnetz benötigt. Dabei erfolgt die verschlüsselte Übertragung nur bis zum Eingang des verbundenen Netzes (z. B Gateway des Unternehmensnetzwerkes), innerhalb des LAN werden die Daten unverschlüsselt übertragen. Für die LAN-Kommunikation müssen dafür andere Sicherheitsmaßnahmen getroffen werden. Es existieren Studien, die belegen, dass die Absicherung der Signalisierungsdaten über diesen Weg ineffektiv ist, denn zu jedem Server, der in diese Verbindung involviert ist, muss ein IPsec-Tunnel aufgebaut werden (Hop-to-Hop-Verbindungen). Als effizienter hat sich die Lösung erwiesen, die Mediendaten über IPSec zu versenden, da in diesem Fall eine Ende-zu-Ende gesicherte Verbindung realisiert werden kann. Für die Verwendung einer IPSec-Lösung werden feste Zugangsdaten (z. B. eine PKI) benötigt, darüber hinaus ist diese Lösung für verteilte Applikationen, wie in einer IM-Konferenz, wegen der Komplexität nur bedingt anwendbar.

SSL/TLS: In den Spezifikationen des SIP über die Nutzung des TLS-Protokolls werden folgende Erweiterungen vorgeschlagen (Dierks und Rescorla 2006): Neben dem Protokollnamen, er heißt jetzt SIPS (SIP over TLS), werden die Adressierung sips:alice@domain.com und der verwendete Port (jetzt Port 5061) geändert. Dieses Verfahren kann nur für Implementierungen verwendet werden, bei denen das SIP mit TCP realisiert wird, da der TLS-Standard keine Lösung für die Absicherung von UDP-Paketen zur Verfügung stellt. SIPS kann über die typischen Sitzungsverwaltungsaufgaben hinaus auch dazu verwendet werden, Schlüssel für die spätere Absicherung der Mediendaten zu übertragen, z. B. durch das SDescriptions-Verfahren für SRTP (vgl. Kapitel 7.2.4.6). Das TLS-Verfahren gewährleistet, ähnlich der Verwendung von IPsec, nur eine sichere Punkt-zu-Punkt-Verbindung.

S/MIME: Es gibt zwei Arten, wie S/MIME in SIP verwendet wird: Es wird nur der Body einer SIP-Nachricht verschlüsselt, sodass die beteiligten Server die Nachricht verarbeiten können, oder es wird die SIP-Nachricht samt Header abgesichert. Dafür wird das gesamte SIP-Paket in einen Body gepackt und dazu ein neuer Header hinzugefügt. Der neue Header kann vom Server bearbeitet werden, der Original-Header ist dagegen jetzt verschlüsselt. Im Vergleich zu anderen Verfahren kann S/MIME sowohl für SIP-Implementationen, die auf TCP, als auch für solche, die auf UDP basieren, implementiert werden. Aufgrund der notwendigen PKI beschränkt sich sein Einsatz auf Prototypen, die im wissenschaftlichen Bereich entwickelt wurden.

7.2.4.5 Jingle

Es existiert für Jingle kein spezieller Sicherheitsmechanismus. Allerdings wird aktuell an einer Erweiterung des Protokolls durch den TLS-Standard gearbeitet. Das sogenannte XTLS-Konzept definiert eine Methode um zwischen zwei XMPP-Entities, die das Jingle-Protokoll implementieren, eine Ende-zu-Ende verschlüsselte Verbindung zu etablieren. Damit kann man nicht nur

eine sichere VoIP-Verbindung, sondern auch weitere Applikationen wie Chat und Filesharing initiieren. (Meyer 2009; Ludwig et al. 2009a)

7.2.4.6 RTP/SRTP

Das RTP besitzt in seiner ursprünglichen Form keine Sicherheitsmechanismen, deswegen wurde eine Erweiterung namens SRTP (Secure Real-time Transport Protocol) entworfen, welche Mechanismen zur Vertraulichkeit und Integrität der Nachrichten sowie zur Authentifizierung der Kommunikationspartner bietet. Dies ist im RFC 3711 beschrieben (Baugher et al. 2004). Es handelt sich um ein Ende-zu-Ende-Protokoll, welches durch Kapselung der RTP-Pakete die oben genannten Sicherheitsanforderungen erfüllt. Für die Verwendung des SRTP wird ein Schlüsselmanagement-Protokoll benötigt, mit dessen Hilfe die Mediendaten verschlüsselt oder signiert werden. Es werden zwei Arten von Schlüsseln verwendet: ein Master Key und verschiedene Session Keys. Der erste Schlüssel ist ein gemeinsamer, geheimer Schlüssel, der von beiden Partnern berechnet, nicht ausgetauscht, wird. Dies ist z. B. durch das Diffie-Hellman-Verfahren möglich (Diffie, van Oorschot und Wiener 1992). Die Session Keys, welche vom Master Key stammen, werden zur Verschlüsselung und Authentifizierung übertragener RTP- und RTCP-Pakete verwendet. Für die Verschlüsselung wird der AES-Algorithmus (Advanced Encryption Standard) in bestimmten Ausprägungen für Mediendaten (sogenannte Modi: AESf8 und AES-CTR) verwendet. Zur Gewährleistung der Integrität und Authentizität wird der Message-Authentication-Mechanismus, z. B. in Form von HMAC-SHA1, eingesetzt.

Im RFC 3711 werden drei Schlüsselaustauschverfahren genannt: MIKEY (Arkko et al. 2004), KEYMGT (Arkko et al. 2006) und SDMS (Andreasen, Baugher und Wing 2006), ohne eine besondere Empfehlung, welches verwendet werden sollte. MIKEY (Multimedia Internet KEYing) und SDMS (Security Descriptions for Media Streams, auch als SDescriptions bekannt) wurden mittlerweile als Standards verabschiedet. Beide wickeln den Schlüsselaustausch während der Signalisierungsphase ab. Dabei gilt SDMS als das einfachste Verfahren, da der Schlüssel als Teil der SIP-Nachricht übertragen wird. Dies ist nur empfehlenswert, wenn vorher eine TLS-Verbindung etabliert ist (Thermos und Takanen 2007; Badach 2007; Bundesamt für Sicherheit in der Informationstechnik 2005).

7.2.5 Zusammenfassung

Betrachtet man das Bedrohungspotenzial von modernen Instant-Messaging-Systemen und existierenden Sicherheitsmechanismen, kann man Folgendes zusammenfassen:

- Die existierenden proprietären IM-Systeme bieten in ihren Default-Einstellungen kaum Sicherheitsmechanismen (außer Skype). Ausnahme ist die Realisierung der Vertraulichkeit bei der Übertragung des Benutzernamens und Passwortes.
- XMPP ist das einzige Chat-Protokoll, welches über Sicherheitsmechanismen zur Vertraulichkeit, Integrität und Authentizität der Kommunikation verfügt. Manche der vorgestellten Konzepte sind entweder zu aufwendig für die Realisierung in IM-Clients, wie z. B. S/MIME), oder es bleiben trotzdem Angriffspunkte übrig, wie bei der Nutzung von SSL und der Überprüfung der Zertifikate (Man-in-the-Middle-Angriff).

- VoIP-Protokolle bieten verschiedene Ansätze für die Sicherung der Kommunikation, allerdings liegt der Schwerpunkt dieser Betrachtungen in der bilateralen Kommunikation.

Somit bleibt viel Raum für die Forschung im Bereich der modernen IM-Systeme (vgl. Kapitel 8), um z. B. Fragen z. B. der Performance der aktuellen Sicherheitsmechanismen, Lösungen für die IM-Gruppenkommunikation mit mehr als zwei Teilnehmern, die Realisierung weiterer Sicherheitsanforderungen oder Methoden zur Absicherung aller drei IM-Kanäle zu untersuchen.

8 Forschung im Bereich der Instant-Messaging-Systeme (Related Work)

Die Forschung im Bereich „IM-Systeme“ kann man grob in zwei Bereiche unterteilen: (1) die Betrachtung der sozialen oder wirtschaftlichen Aspekte und (2) die Betrachtung technischer Ansätze. Im Folgenden werden beispielhaft Projekte und Arbeiten aus diesen beiden Bereichen vorgestellt. Dabei wird der technische Bereich stärker fokussiert, da dort Konzepte und Technologien zu Sicherheit und Erweiterung von IM-Systemen anzusiedeln sind, welche im Kernbereich der vorliegenden Arbeit liegen.

8.1 Wirtschaftliche und soziale Aspekte

In der Forschung zur Computer-Supported Cooperative Work (CSCW, siehe Kapitel 5.1) existiert eine Reihe von Studien, die sich mit IM-Systemen und dort hauptsächlich mit der Chat-Funktionalität beschäftigen.

Während anfängliche Arbeiten die Nutzung von IM-Systemen am Arbeitsplatz für eine informale und eher private Kommunikation belegen (Whittaker, Frohlich und Daly-Jones 1994), zeigen spätere Untersuchungen in etwa 60 bis 70 % der Fälle eine Verwendung für arbeitsbezogene Themen, wenn IM-Systeme innerhalb von Unternehmen eingesetzt werden (Isaacs et al. 2002; Handel und Herbsleb 2002; Herbsleb et al. 2002; Cameron und Webster 2005). In diesem Zusammenhang wurde auch die Änderung des Kommunikationsverhaltens oder die Wahl der Sprache bei Chat-Nachrichten im Gegensatz zu sonstiger Büro-Kommunikation von Person zu Person untersucht (Zhou 2005; Baron 2005; Simon 2006).

Weitere Forschungsprojekte vergleichen den Einsatz von IM-Systemen mit dem von E-Mail-Systemen und kommen zu dem Ergebnis, dass, wenn sowohl IM als auch E-Mail genutzt werden, kein signifikanter Unterschied bei der Qualität der Gruppenzusammenarbeit zu verzeichnen ist. Einzig eine Präferenz der jüngeren Arbeitnehmer, IM-Systeme zu nutzen, ist erkennbar (Huang, Hung und Yen 2007). Bei den Regelungen zur Nutzung von E-Mail und IM von Seiten des Arbeitgebers sind jedoch Unterschiede erkennbar. Während für E-Mail-Nutzung konkrete Policies existieren (z. B. erlauben 68 % von 300 in einer Studie befragten Unternehmen eine kontrollierte private Nutzung), gibt es für die Nutzung von IM-Systemen entweder gar keine Regelungen (35 % der Unternehmen), oder nur ca. 40 % der Unternehmen erlauben hier eine private Nutzung am Arbeitsplatz. 16 % der Unternehmen verbieten sogar die Nutzung des IM Mediums, während die Zahlen bei Unternehmen, die das private Telefonieren oder den Austausch von E-Mail-Nachrichten verbieten, bei 3 bis 5 % liegen (Swartz 2005).

Während also zur Nutzung von E-Mail und Chat-Nachrichten einige Studien existieren, sind Arbeiten, die sich mit den weiteren Kanälen eines modernen IM-Systems beschäftigen, eher begrenzt. Der Grund liegt hier darin, dass erst in den letzten Jahren derartige Funktionen wie VoIP oder Filesharing eine stärkere Verbreitung in IM-Systemen gefunden haben. Einige Betrachtungen über die Nutzung und Eignung aller Kommunikationskanäle eines IM-Systems im Hinblick auf eine Verwendung im alltäglichen Geschäftsleben finden sich in (Turek 2004; Riemer, Frößler und Klein 2007; Frößler 2008; Joisten 2007)

8.2 Technische Aspekte

Neben den Untersuchungen aus einer eher wirtschaftlichen und sozialen Perspektive spielen bei einem technologieorientierten Thema wie IM-Systemen auch technische Aspekte eine wichtige Rolle. In der Forschung sind hier häufig prototypische Entwicklungen zu finden, wie etwa IM-Systeme, welche eine Visualisierung und Gruppierung von Kontakten erlauben (Neustaedter, Greenberg und Carpendale 2002), die Verwendung von kinetischer Typografie unterstützen (Animation, fliegender Text), um Gefühle besser als mit Emoticons auszudrücken (Bodine und Pignol 2003), oder IM-Systeme, die über mobile Eigenschaften verfügen (Isaacs, Walendowski und Ranganathan 2002).

Andere Entwicklungen zielen auf die Erschließung weiterer Einsatzgebiete für IM-Systeme, wie z. B. die Nutzung von IM als einem weiteren Kommunikationskanal für E-Learning-Umgebungen, um die Zusammenarbeit von Schülern zu unterstützen (Hrastinski 2006; Contreras-Castillo, Perez-Fragoso und Favela 2006), oder als Medium für Interviews (Stieger und Göritz 2006; Zalinger, Freier und Dutko 2009). Instant-Messaging-Systeme wurden aber auch in sensibleren Bereichen getestet, wie für den Informationsaustausch zwischen Ärzten in medizinischen Umgebungen (Sachpazidis et al. 2006).

Je sensibler die Daten sind, die über eine IM-Verbindung übertragen werden, desto wichtiger ist deren Absicherung. Wie in Kapitel 7.2.4 ausgeführt wurde, gibt es erste Vorschläge für Mechanismen. Allerdings sind diese nicht für alle Szenarien und die entsprechenden Sicherheitsanforderungen passend. Im Folgenden werden einige dieser Projekte vorgestellt, welche im Rahmen dieser Arbeit teilweise Lösungen zu den in Kapitel 7.2.5 aufgeworfenen Fragen bieten.

8.2.1 ZRTP

ZRTP ist eine Entwicklung von Phil Zimmermann für die Absicherung von VoIP-Daten, die von Softphone-Lösungen (Software für Telefonie) übertragen werden (Zimmermann, Johnston und Callas 2007). Auch wenn dabei durch den Namen der Eindruck erweckt wird, dass es sich hierbei um eine sichere Variante des RTP (ähnlich dem SRTP) handelt, ist ZRTP eher ein Konzept für ein Schlüsselaustauschverfahren ähnlich wie SDescriptions oder MIKEY, welche den sicheren Austausch von Sitzungsschlüsseln ermöglichen (vgl. Kapitel 7.2.4.6). Der Schlüsselaustausch findet hier nicht über SIP, sondern über RTP statt. Die Funktionsweise ist folgende:

Als Erstes wird eine unverschlüsselte SIP-Verbindung zwischen den beiden Teilnehmern initiiert und eine dazugehörige unverschlüsselte RTP-Session aufgebaut. Die ZRTP-Session startet nach der Signalisierungsphase und nutzt die gleichen Ports wie das RTP. Dies ist so vorgesehen, damit eine Kommunikation zwischen den Teilnehmern auch dann zu Stande kommt, einer der

beiden kein ZRTP unterstützt. Dafür sendet der ZRTP-Client zunächst ein RTP-Paket mit einem Flag, um herauszufinden, ob bei dem Kommunikationspartner auf der anderen Seite ZRTP unterstützt wird. ZRTP-Pakete werden also sozusagen in RTP-Pakete eingebettet.

Auf Grundlage dieser Verbindung wird das Diffie-Hellman-Schlüsselaustauschverfahren (DH) verwendet, um Parameter für die Erzeugung eines gemeinsamen Schlüssels auszutauschen. Anschließend berechnet jeder der Teilnehmer aus dem ausgehandelten Schlüssel über eine Hashfunktion einen der sogenannten Short Authentication Strings (SAS), die auf dem Display des jeweiligen Clients angezeigt werden. Die Teilnehmer lesen sich gegenseitig diese Werte über den Audiokanal vor und gewährleisten damit, dass die Verbindung nicht durch eine Man-in-the-Middle-Attacke manipuliert wurde, da sie sich gegenseitig sehen oder hören. Als Alternative gibt es auch die Möglichkeit, diesen Hashwert automatisch über den Software-Client zu vergleichen, ohne das Zutun der Teilnehmer. Bei jedem weiteren Kontakt fließt der vorherige Hashwert in die Erzeugung des neuen Wertes (mit dem gleichen Teilnehmer) mit ein. Dies bedeutet, dass Teilnehmer, die zu einem späteren Zeitpunkt erneut miteinander telefonieren möchten, den SAS-Vergleich nicht wiederholen müssen. Das bedeutet auch, dass, wenn einmal erfolgreich eine sichere Kommunikation zwischen zwei Partnern stattgefunden hat, der Angreifer den Schlüssel dieser Sitzung und die Schlüssel aller folgenden Sitzungen nicht errechnen kann. Eine weitere Eigenschaft des Verfahrens ist die sogenannte „Perfect Forward Privacy“. Bei diesem Verfahren werden die verwendeten Schlüssel nach einer Sitzung zerstört. Das heißt, sollte es einem Angreifer doch gelingen, einen Sitzungsschlüssel herauszufinden, so kann er davon ausgehend weder auf vorherige noch auf folgende Sitzungsschlüssel schließen. (Zimmermann, Johnston und Callas 2007; Sotillo 2006; Petraschek 2007)

Es sollte hier angemerkt werden, dass ZRTP nur die Mediendaten verschlüsselt, und zwar mittels AES. Für die ansonsten unverschlüsselten Signaldaten kann z. B. eine Absicherung durch eine TLS-Verbindung verwendet werden. Das ZRTP-Konzept ist im Zfone-Projekt implementiert und frei verfügbar (Zimmermann 2006).

8.2.2 Instant Messaging Key Exchange Protokoll (IMKE)

Ein weiterer Ansatz, der sich mit der Sicherheit von IM-Systemen beschäftigt, ist das Instant Messaging Key Exchange Protokoll (IMKE), welches von Mannan und van Oorshot entwickelt wurde (Mannan und van Oorschot 2006b; Mannan und Van Oorschot 2006a). Ziel dieses Protokolls ist eine sichere Verbindung zwischen Server und Client (Authentifikation, Vertraulichkeit, Integrität) sowie eine sichere *abstreitbare* Kommunikation zwischen den Clients. Obwohl zwischen den Clients paarweise keine Geheimnisse ausgetauscht werden, bietet IMKE eine sichere und private Kommunikation zwischen den Parteien, indem ein vertrauenswürdiger IM-Server verwendet wird, auf dem keine der ausgetauschten Nachrichten gespeichert wird. Das Protokoll basiert auf einer starken Authentifikation mit Hilfe eines dauerhaften Geheimnisses zwischen Server und Client (dies entspricht einem Passwort des Clients) und einem allgemein bekannten Zertifikat des Server. Dabei liegt der Schwerpunkt bei IMKE auf der One-to-one-Kommunikation zwischen zwei Teilnehmern.

Den Ablauf des IMKE-Protokolls kann man in drei Schritte unterteilen: (1) Authentifikation und Schlüsselaustausch zwischen Server und Client, (2) Verteilung der Schlüssel zwischen den

Clients und (3) Festlegung eines Sitzungsschlüssel für die Client-zu-Client-Kommunikation (vgl. Abbildung 15).

Bei IMKE verfügt der IM-Server über einen öffentlichen und einen privaten Schlüssel (ein Schlüsselpaar). Der öffentliche Schlüssel wird in Form eines Zertifikats während der Installation auf dem Client gespeichert und kann jederzeit online verifiziert werden, ähnlich wie die Verifikation von Zertifikaten in einem Browser. Für alle Hashberechnungen in den folgenden Berechnungen (gesendete Schlüssel oder Parameter) werden unterschiedliche One-way-Funktionen verwendet.

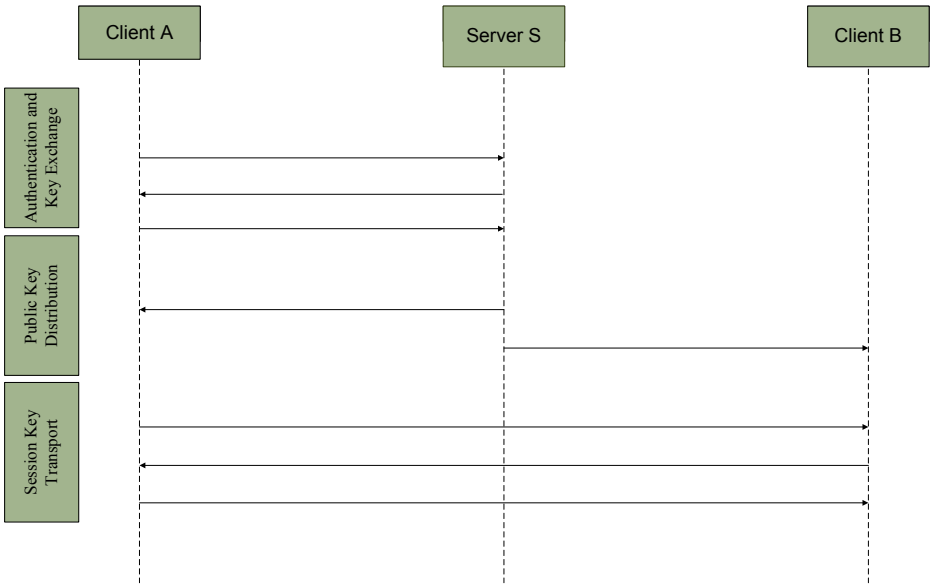

Abbildung 15 – IMKE-Protokoll (in Anlehnung an (Mannan und Van Oorschot 2006a))

Der Client (Benutzer) verfügt über eine Kennung und ein Passwort, welche bei der Registrierung generiert und verwendet werden, über kurzzeitige, selbst generierte Schlüsselpaare (eines für die Authentifikation mit dem Server und je eines für die Kommunikation mit einem anderen Client) sowie über kurzfristige symmetrische Schlüssel für die Kommunikation mit dem Server und entsprechenden MACs, und je einen kurzfristigen symmetrischen Schlüssel für die Kommunikation mit jedem anderen Client einschließlich entsprechenden MACs.

(1) Die Authentifikation und der Schlüsselaustausch zwischen Server und Client finden in einem vierstufigen Handshake statt (vgl. Abbildung 15, Authentication and Key Exchange):

(1a) Der Client authentifiziert sich dem Server gegenüber mit seinem dauerhaften Schlüssel (dem Passwort zur Kennung) ID_A , P_A und verifiziert den öffentlichen Schlüssel des Servers. Er generiert einen öffentlichen und privaten Schlüssel KU_A, KR_A und einen zufälligen symmetrischen Schlüssel K_{AS}. Er verschlüsselt diesen symmetrischen Schlüssel mit dem öffentlichen Schlüssel des Servers $\{K_{AS}\}_{E_S}$. Mit dem symmetrischen Schlüssel verschlüsselt er seinen öffentlichen Schlüssel sowie einen Hashwert des Passworts $\{KU_A, f_1(P_A)\}_{K_{AS}}$ und sendet diese Daten an den Server weiter.

(1b) Der Server berechnet seinerseits ebenfalls den Hashwert des gespeicherten Passworts des Clients und vergleicht diesen Wert mit den gerade vom Client erhaltenen Daten. Ist diese Überprüfung erfolgreich, berechnet er mit einer anderen Hashfunktion wieder einen Hashwert des

Passworts $\{f_2(P_A)\}_{K_{AS}}$ und verschlüsselt dies mit dem vorhin übertragenen symmetrischen Schlüssel. Er generiert eine Zufallszahl R_S, die er dann mit dem kurzzeitigen öffentlichen Schlüssel des Clients verschlüsselt, und sendet diese zusammen mit dem Hashwert an den Client.

(1c) Der Client überprüft jetzt zum letzten Mal den neuen Hashwert des Passworts und berechnet einen symmetrischen Sitzungsschlüssel für die Verschlüsselung der Kommunikation mit dem Server und einen MAC-Wert. Diese beiden Werte (symmetrischer Schlüssel K_{AS}^{S} und MAC K_{AS}^{m}) werden aus einer Kombination der Zufallszahl vom Server, die im letzten Schritt übertragen wurde und dem symmetrischen Schlüssel, den der Client am Anfang der Kommunikation generiert hat.

(1d) An den Server wird ein neuer Hashwert der Zufallszahl, die in dem Schritt zuvor an den Client gesendet wurde, geschickt. Ist die Überprüfung des Hashwertes erfolgreich kann er selbst den symmetrischen Schlüssel für die Verschlüsselung und die MACs berechnen ohne die Daten über die Leitung schicken zu müssen.

(2) Nachdem diese Kommunikation zwischen Server und Client abgeschlossen ist, folgt als zweiter Schritt die Schlüsselverteilung zwischen zwei Clients. Dafür würfelt jeder Client ein neues Schlüsselpaar und überträgt den öffentlichen Schlüssel an dem Server. Dieser leitet diese Informationen an die Clients weiter. Hier wird also der Server als vertrauenswürdigt vorausgesetzt.

(3) Da die dabei ausgetauschten Informationen auch die IP-Adressen der Clients beinhalten, können diese sich jetzt gegenseitig kontaktieren und mit einem ähnlichen Handshake-Verfahren wie oben in (1) beschrieben den dritten Schritt durchführen und einen neuen Session-Key vereinbaren.

Nach der Aushandlung der Sitzungsschlüssel zwischen Client-Server und Client-Client werden die Zufallszahlen und die symmetrischen Schlüssel gelöscht, damit die Abstreitbarkeit der Kommunikation bewahrt bleibt. Eine Beispiel-Implementation wurde basierend auf dem Jabber-Protokoll realisiert. Dafür wurden sowohl auf dem Client als auch auf dem Server Änderungen vorgenommen.

8.2.3 Off-the-Record (OTR)

Das Off-the-Record-(OTR-)Protokoll konzentriert sich auf die Nachbildung eines privaten Vieraugengesprächs mit technischen Mitteln. Dabei wird eine vertrauliche, integere und authentifizierte Verbindung unter Gewährleistung der „Perfect Forward Secrecy“ und der Abstreitbarkeit aufgebaut. Während der Kommunikation wird einerseits gewährleistet, dass Nachrichten *während der Kommunikation* vom richtigen Absender kommen, aber andererseits *nach der Kommunikation* nicht bewiesen werden kann, dass eine Nachricht tatsächlich von einem bestimmten Absender stammt. Dies ist dadurch möglich, dass jeder andere Teilnehmer Nachrichten mit einem gefälschten Absender generieren kann. Dies wird zusätzlich durch den Einsatz eines Verschlüsselungsalgorithmus mit kurzfristigen, nicht gespeicherten Schlüsseln unterstützt, sodass gespeicherte Gespräche nicht entschlüsselt werden können.

OTR verwendet mehrere kurzzeitige Schlüssel für die Verschlüsselung der Nachrichten, ein langzeitiges Schlüsselpaar (jeweils eines für jeden Teilnehmer) für die Signierung von Schlüs-

seln und MACs für die Signierung von Nachrichten und die Realisierung der Integrität und Authentifizierung der Teilnehmer.

Für die Authentifizierung der Gesprächspartner wird das Digital-Signature-Algorithm-Verfahren (DSA) verwendet (National Institute of Standards and Technology 2007) (vgl. Kapitel 11.7.3.1). Die Teilnehmer verfügen über ein Schlüsselpaar und müssen die öffentlichen Schlüssel verifizieren. Dies findet am Anfang einer Session statt. Danach wird für jene Nachricht ein DH-Schlüsselaustausch durchgeführt. Damit nicht zu viele Nachrichten übertragen werden, also nicht separat die Parameter des DH-Schlüsselaustausches und die eigentliche Nachricht übertragen werden, wird folgendermaßen vorgegangen:

Die ersten öffentlichen kurzzeitigen Schlüssel, die aufgrund des DH-Verfahrens generiert und übertragen werden, sind mit den DSA-öffentlichen Schlüsseln (den langzeitigen Schlüsseln) der Teilnehmer signiert. Jede weitere Nachricht enthält einen neuen öffentlichen DH-Schlüssel und die eigentliche Nachricht, welche mit dem Schlüssel des vorangegangenen DH-Austauschs verschlüsselt ist. Bei jeder Nachricht wird angegeben, mit welchem Diffie-Hellman-Schlüsseln diese Nachricht verschlüsselt ist. Dafür werden die DH-Schlüssel durchnummeriert. Um die Abstreitbarkeit zu ermöglichen, kann nicht mit digitalen Signaturen weitergearbeitet werden. Stattdessen werden, um die Integrität der Nachrichten sicherzustellen, die Daten mit einem MAC signiert. Der MAC wird als Hashwert des gemeinsamen, authentifizierten DH-Schlüssels berechnet. Damit lässt sich überprüfen, ob die Nachrichten mit einem gültigen MAC auch von der Person stammen, mit welcher der initiale, authentifizierte Schlüsselaustausch stattgefunden hat. Abstreitbar ist bei OTR nur der *Inhalt* eines Gesprächs und nicht die Tatsache, dass zwischen zwei Teilnehmern eine Kommunikation stattgefunden hat. Es sollte noch erwähnt werden, dass alte Schlüssel immer erst dann verworfen werden können, wenn der Kommunikationspartner den neuen Schlüssel benutzt hat. (Borisov, Goldberg und Brewer 2004; Stedman, Yoshida und Goldberg 2008)

Im Vergleich zum IMKE-Protokoll (vgl. Kapitel 8.2.2) ist OTR als eine Add-on-Lösung für den GAIM IM-Client implementiert worden. GAIM ist ein Multiprotokoll-IM-Client für verschiedene IM-Systeme. Inzwischen ist die Software bekannt unter dem Namen „Pidgin“ (Pidgin 2009). OTR wird auch von weiteren IM-Systemen implementiert.

8.2.4 Weitere Arbeiten zum Thema sichere IM-Kommunikation

Einen weiteren Ansatz für eine sichere IM-Kommunikation, wobei unter dem Begriff „sicher“ unterschiedliche Sicherheitsanforderungen verstanden werden (vgl. Kapitel 8.2.1, 8.2.2, 8.2.3), stellt das Protokoll von (Kikuchi, Tada und Nakanishi 2004) dar, welches nur die Vertraulichkeit einer Client-zu-Client-Kommunikation gewährleistet. Ziel ist es, die Inhalte einer Gruppenkommunikation über den Chatkanal (oder auch in einem Chatroom) vor dem IM-Server (und dem zugehörigen Administrator) geheim zu halten. Das Protokoll beruht auf einer Verwendung von Diffie-Hellman-Verfahren, um private und öffentliche Schlüssel zu erzeugen. Diese werden über den Server verteilt und anschließend verwendet, um eine vertrauliche Peer-to-Peer-Kommunikation zu etablieren.

Ein weiteres stärkeres Protokoll ist das Secure Instant Messaging and Presence Protocol (SIMPP), das von (Yang et al. 2008) entwickelt wurde. SIMPP basiert auf IMKE (vgl. Kapitel

8.2.2), hat aber das Ziel, eine schnellere Berechnung der Schlüssel zu erzielen, und verwendet für die Berechnungen Algorithmen, die auf elliptischen Kurven basieren.

Über die hier genannten Verfahren hinaus, gibt es nur wenige weitere Arbeiten, die sich mit Sicherheitsaspekten von Instant-Messaging-Systemen beschäftigen.

8.3 Zusammenfassung

Ziel der vorliegenden Arbeit ist nicht nur die einfache *Betrachtung* der Sicherheitsaspekte von IM-Systemen, sondern auch deren *Nutzung als Plattform* für eine spontane sichere Kollaboration. Die Verbreitung von IM-Systemen innerhalb von Unternehmen sowie ihre einfache und spontane Handhabung ist Motivation, auch weitere Aufgaben mit diesen Systemen zu lösen.

Wie in den vorangegangenen Kapiteln gezeigt wurde, gibt es einige Ansätze und Standards, die Sicherheitsanforderungen an eine sichere Kommunikation erfüllen. Allerdings beschränken sich diese Lösungen auf konkrete Kommunikationskanäle, wie z. B. die Verschlüsselung einer VoIP-Verbindung oder die Absicherung von Chat-Nachrichten. Es existieren jedoch keine Überlegungen für (1) die sichere Verknüpfung und Kombination mehrerer Medien und (2) die Nutzung eines IM-Systems als Plattform für die Realisierung von sicherheitskritischen Anwendungen, die gleichzeitig einen Gruppen- und Kollaborationscharakter haben.

Die existierenden Sicherheitsmechanismen, die marktübliche IM-Systeme anbieten, sind je nach Szenario und Anwendungszweck ausreichend, wenn es um eine einfache informelle Kommunikation zwischen Partnern geht. In solchen Fällen ist häufig die Sicherstellung der Vertraulichkeit (durch einen Verschlüsselungsalgorithmus) oder die Sicherstellung der Identität der Teilnehmer (durch die Verwendung von Zertifikaten) ausreichend. Sie sind aber nicht für eine formelle und verbindliche Kommunikation ausgerichtet.

Außerdem sind derartige Lösungen auf eine One-to-one-Kommunikation ausgerichtet und bieten keine Grundlagen für die Realisierung einer Gruppenkommunikation. Möchte man aber, wie in dieser Arbeit motiviert (vgl. Kapitel 2), IM-Systeme verwenden, um eine Kollaboration (1) spontan ohne einen davor geschalteten Authentifikationsprozess (z. B. in Form von Zertifikaten) und (2) innerhalb einer Gruppe durchzuführen und zwar so, dass (3) das Ergebnis langfristig beweisbar ist, reichen diese Methoden nicht aus.

Im Rahmen dieser Arbeit wird später ein Ansatz vorgestellt, der diese Anforderungen erfüllt (vgl. Teil III). Zunächst werden im folgenden Kapitel jedoch Anwendungen betrachtet, die eine solche Infrastruktur als Plattform nutzen, um auf dieser Grundlage eine Kollaboration über Unternehmensgrenzen hinweg zu realisieren. Ziel der vorliegenden Arbeit ist nicht nur die Betrachtung der Sicherheitsaspekte der IM-Systeme, sondern deren Eignung als Plattform für eine spontane sichere Kollaboration. Die Verbreitung dieser Systeme innerhalb der Unternehmen und deren einfache Handhabung ist eine Motivation, noch ergänzende Aufgaben mit diesen Systemen lösen zu können.

Wie in den Kapiteln zuvor gezeigt wurde, gibt es ein paar Ansätze oder sogar Standards, die die Sicherheitsanforderungen einer sicheren Kommunikation erfüllen. Allerdings begrenzen sich diese Lösungen auf konkrete Kommunikationskanäle, wie z. B. auf eine VoIP-Verbindung oder die Absicherung der Chat-Nachrichten. Es existieren keine Überlegungen für die Verknüpfung

und Nutzung beider Medien oder die Erweiterung solcher Systeme für die Durchführung weiterer Anwendungen, die diesem Gruppen- und Kollaborationscharakter entsprechen.

Die Sicherheitsmechanismen, die IM-Systeme anbieten, sind je nach Meinung ausreichend, wenn es sich um eine informelle Kommunikation zwischen zwei Partnern handelt. In solchen Fällen ist vielleicht die Erfüllung der Vertraulichkeit und die damit verbundene Implementierung eines Verschlüsselungsalgorithmus oder die Verwendung von Zertifikaten, um die Identität der Teilnehmer zu überprüfen, ausreichend. Außerdem sind diese Lösungen auf eine One-to-one-Kommunikation ausgerichtet und bieten keine Überlegungen für die Realisierung einer Gruppenkommunikation mit mehr als zwei Personen. Möchte man aber IM-Systeme verwenden, um spontan eine Kollaboration durchzuführen, und zwar so, dass das Ergebnis langfristig beweisbar ist, reichen diese Methoden nicht aus. Bevor ein solcher Lösungsansatz in Teil III präsentiert wird, sollen in dem folgenden Kapitel Anwendungen vorgestellt werden, die mit Hilfe einer IM-Infrastruktur zu einer Kollaboration über Unternehmenswege hinweg führen können.

9 Forschung im Bereich kollaborativer sicherheitskritischer Anwendungen (Related Work)

Die Globalisierung und die damit zusammenhängende räumliche und organisatorische Verteilung von Arbeitsgruppen sowie die Notwendigkeit zu breiten Kollaborationen, z. B. bei der Beantragung von größeren Projekten in öffentlichen Ausschreibungen, zwingen Unternehmen und Mitarbeiter dazu, nach Lösungen zu suchen, welche die Organisation, Entscheidung und Bearbeitung von solchen Projekten unterstützen. Neben der Nutzung des Telefons oder von E-Mail als Kommunikationsmedium, finden immer mehr moderne IM-Systeme Verbreitung. Bis jetzt werden derartige Systeme vor allem als günstigere Hilfsmittel (Chat oder VoIP-Kommunikation sind kostengünstiger als ein herkömmliches Telefonat) oder zur schnelleren Reaktion (im Vergleich zu E-Mail) eingesetzt. Aus ähnlichen Gründen entdecken Unternehmen solche Systeme als Mittel zur Erleichterung des Kundenkontaktes (Wenger, Merten und Teufel 2006; Turek 2004).

Alle bisherigen Systeme dienen aber nur der Vorbereitung sicherheitskritischer Anwendungen, da ihnen für die Anwendungen selbst die erforderlichen Sicherheitsmechanismen fehlen. Dabei könnte man durch diese Systeme weitere Anwendungsfelder erschließen, wenn man sie ausreichend mit Sicherheitsmechanismen ausstattete. Beispiele sind hier die sichere Aushandlung eines Vertrags, die Durchführung einer elektronischen Transaktion bei der Beschaffung von Waren oder eine Gruppendiskussion mit zugehöriger Abstimmung.

Manche dieser Anwendungsfälle sind derzeit entweder nur bei physikalischer Präsenz der Beteiligten, durch das Versenden von Dokumenten oder die Vermittlung einer dritten Partei (z. B. einer Bank) durchführbar.

Teilweise existieren auch technische Lösungen, die solche Anwendungen über das Internet realisieren. Beispiele hierfür sind Real Time Communication-Lösungen oder branchenspezifische Lösungen (z. B. Online-Shops), die technische Mechanismen verwenden, um für die Ge-

schäftspartner ein bestimmtes Sicherheitsniveau anzubieten oder juristische Auflagen wie das Telemediengesetz (Bundesministerium der Justiz 2009b) zu erfüllen.

Findet eine Geschäftstransaktion oder Gruppendiskussion mit Abstimmung an einem bestimmten Standort statt, gilt für jeden Geschäftspartner ein gegebenes Sicherheitsniveau. Man steht vor dem Gesprächspartner und erkennt ihn (Authentizität), die Kommunikation ist (in einem ausreichenden Maße) vertraulich und integer, wenn sie zum Beispiel in einem geschlossenen Raum abseits von möglichen dritten Zuhörern stattfindet. Kommt es zum Abschluss einer Transaktion, wird eine Dokumentation des Vorgangs erstellt. Beispielsweise kann ein schriftlicher Vertrag geschlossen und unterschrieben werden, womit die langfristige Verbindlichkeit des Inhaltes der Kollaboration bekundet wird. Dafür müssen unter anderem die Unterschriften der Personen und die Zuordnung von Person und Unterschrift bekannt sein.

Die rechtlichen Grundlagen für derartige Vereinbarungen sind gesetzlich geregelt, in Deutschland zum Beispiel im Bürgerlichen Gesetzbuch (BGB) (vgl. BGB Buch 1 „Allgemeiner Teil“, Kapitel 3 „Rechtsgeschäfte“ (BGB 2009)) oder im Handelsgesetzbuch (HGB). In vielen Fällen gibt es für die unterschiedlichen Vertragstypen zusätzliche Regelungen, etwa für Kauf- oder Tauschverträge nach BGB (vgl. §§ 433-479 BGB) oder für Handelsgeschäfte nach HGB (vgl. HGB Buch 4 „Von den Handelsgeschäften“).

Als Beispiel-Anwendung für die vorliegende Arbeit wurde eine Abstimmung mit vorangehender Diskussion und kollaborativer Entscheidungsfindung gewählt. Ein solcher Prozess enthält die Schritte Problemstellung, Lösungssuche, Bewertung der Lösungsalternativen und Auswahl einer Lösung durch Abstimmung. Dieser Anwendungstyp eignet sich besonders gut zur Diskussion und Demonstration der entwickelten Konzepte, da er eine *spontane* Kollaboration einerseits und *verbindliche*, nicht-abstreitbare Entscheidungen andererseits kombiniert.

Im Hinblick auf die vorher erwähnten gesetzlichen Regelungen sollte angemerkt werden, dass die Abstimmungen keine elektronischen Verträge im Sinne des § 129a BGB darstellen – unter anderem deswegen, weil hier keine qualifizierte elektronische Signatur im Sinne des Signaturgesetzes (SigG) verwendet wird (Bundesministerium der Justiz 2009a). Vielmehr soll mit den entworfenen (und später implementierten) Verfahren die grundsätzliche Kombination von Spontaneität und Verbindlichkeit demonstriert werden.

Bei herkömmlicher Gruppenarbeit (d. h. an einem Standort) werden unterschiedliche Kreativitätsmethoden wie Brainstorming, Brainwriting oder Mindmapping verwendet, um passende Lösungen zu finden. Dabei haben die Teilnehmer der Gruppe die Möglichkeit, die infrage kommenden Alternativen zu diskutieren, bis sie die Anzahl dieser Lösungsansätze ausreichend zusammenstellen und definieren können, sodass eine Abstimmung möglich ist. Je nach den Anforderungen des Szenarios und den entsprechenden Regelungen wird diese Abstimmung entweder geheim oder offen durchgeführt. Agiert eine Gruppe hingegen verteilt an unterschiedlichen Standorten, beschränkt sich der kollaborative Prozess auf die Nutzung einer Telefonanlage oder eines E-Mail-Systems. Diese Medien können nur dann ausgewählt werden, wenn eine offene Abstimmung unter den Teilnehmern durchgeführt werden soll. Dabei ist die Sicherstellung der Vertraulichkeit nur bedingt möglich. So müssen beispielsweise für eine sichere E-Mail-Kommunikation alle Teilnehmer *vorher* entsprechende Schlüssel ausgetauscht haben, etwa mit einer Public Key Infrastructure (PKI) oder Ansätzen wie „Pretty Good Privacy“ (PGP). Ein

Telefonat gilt als schwer manipulierbar, zumindest für normale Bürger. Einige Behörden haben dagegen solche Manipulationsmöglichkeiten, zum einen technisch (etwa durch Zugriff auf Vermittlungsstellen des Telefonsystems) und zum anderen rechtlich (als Beispiel vgl. StPO 1. Buch 8. Kapitel „Beschlagnahme, Überwachung des Fernmeldeverkehrs, Rasterfahndung, Einsatz technischer Mittel, Einsatz Verdeckter Ermittler und Durchsuchung“, insbesondere §§ 100c und 100d) (Bundesministerium der Justiz 2009c).

Für die vorliegende Arbeit wird der Entscheidungsprozess in zwei Teile unterteilt, die Diskussion und die elektronische Wahl. Im Folgenden wird davon ausgegangen, dass die Diskussion immer offen stattfindet, d. h., jeder Teilnehmer kann erkennen, von welchem anderen Teilnehmer Diskussionsbeiträge stammen (Angabe der Identität). In anderen Szenarien sind zwar auch anonyme Diskussionsverfahren denkbar, diese spielen aber bei den weiteren Betrachtungen keine Rolle. Die Abstimmung kann je nach Szenario entweder offen oder geheim realisiert werden. In einer offenen Abstimmung wird unter Angabe der Identität gewählt. Jeder Teilnehmer ist dann sowohl in der Lage, das Ergebnis selbst zu berechnen, als auch jede Stimme einer Person zuzuordnen. Bei einer geheimen Abstimmung ist eine der wichtigsten Anforderungen die Anonymität des Wählers. Dies ist nicht ganz einfach zu realisieren, da zwar jede Stimme verdeckt (geheim) abgegeben werden soll (d. h. niemand kann eine Stimme einer Person zuordnen), trotzdem soll aber jeder Wähler sicher sein, dass seine eigene Stimme in das Ergebnis einfließt, und jeder Wähler soll die Entstehung des Ergebnisses nachvollziehen können. Für diesen zweiten Teil des Entscheidungsprozesses, die elektronische Wahl, sollen an dieser Stelle ein paar Grundlagen erläutert werden, die später auch die Grundlage für Entwurfsentscheidungen bei der Realisierung des Prototyps sind (vgl. Kapitel 14).

Aktuell ist die Forschungsgemeinschaft im Bereich der elektronischen Wahlen sehr aktiv, unter anderem aufgrund des Urteils des Bundesgerichtshofes bezüglich der Verwendung elektronischer Geräte bei der Wahl und der Nachvollziehbarkeit der Ergebnisse (Weddeling et al. 2007; Hupf und Meletiadou 2009) Auch wenn das langfristige Ziel die Erforschung von geeigneten Protokollen für elektronische parlamentarische Wahlen ist, konzentriert sich die Forschung derzeit auf die Analyse bestimmter Aspekte einer Wahl (wie etwa Mechanismen zur Authentifizierung, Verifikation) oder die Realisierung von elektronischen Wahlen in kleineren Kontexten, etwa in Vereinen oder Universitäten (Alkassar, Krimmer und Volkamer 2005; Grimm et al. 2009; Grimm et al. 2006). Spricht man von *elektronischen Wahlen*, unterscheidet man zwei Formen: (1) die Wahl mit *Wahlgeräten*, die Stimme wird hier an einem elektronischen Wahlgerät im Wahllokal abgegeben, und (2) Online-Wahlen, die Stimme wird hier über ein offenes Netz, das Internet, übertragen. Für das hier betrachtete Szenario, den Einsatz in einer kollaborativen Umgebung und die Integration mit einem IM-System, ist die zweite Form mit den zugehörigen Konzepten und Techniken relevant. Hier wird wiederum unterschieden zwischen (2a) Systemen, die über eine zentrale vertrauenswürdige Instanz koordiniert werden, und (2b) Systemen, die ohne eine solche Instanz auskommen.

Im Vergleich mit der herkömmlichen nicht elektronischen Durchführung einer Wahl übernimmt diese zentrale Instanz ungefähr die Aufgaben einer Wahlkommission, d. h., sie überprüft die Wahlberechtigung der Wähler, verfolgt den Wahlprozess, gewährleistet die Richtigkeit des Wahlprotokolls und berechnet und verkündet das Ergebnis. Es gibt wenige solche Protokolle, die eine administrativlose (ohne zentrale Instanz) elektronische Wahl realisieren. Der Grund

liegt in der hohen Komplexität und dem hohen Berechnungsaufwand sowie in der Abhängigkeit der Protokollschritte von den einzelnen Teilnehmern. Dies führt zu einer Begrenzung der Anzahl möglicher Wähler. Somit sind diese Protokolle nicht für kommunale oder parlamentarische Wahlen geeignet.

Um die Anforderungen an eine anonyme Wahl realisieren zu können, stützen sich diese Konzepte auf das Vermischen von Stimmen oder das Secret-Sharing-Verfahren: Das *Vermischen von Stimmen* basiert auf den Mix-Netzen von (Chaum 1981). Diese Verfahren zielen darauf, durch die Verwendung von kryptografischen Verfahren und das Vermischen (Mixen) von Nachrichten die Anonymität der Sender zu bewahren. Das Funktionsprinzip beruht auf dem Empfang von verschlüsselten Nachrichten und der Entschlüsselung und Änderung der Sortierung für die weitere Abarbeitung, sodass zwischen der Reihenfolge der eingehenden Nachrichten und der Reihenfolge der ausgehenden Nachrichten keine nachvollziehbare Verbindung existiert. Die Anzahl der Knoten, die dieses Vermischen übernehmen, variiert je nach Anwendung zwischen einem und mehreren Mixern. Dieses Verfahren wird zum Beispiel in Wahlprotokollen von (DeMillo, Lynch und Merritt 1982; Alkassar, Krimmer und Volkamer 2005; Pfitzmann und Waidner 1992) verwendet. Das *Secret-Sharing-Verfahren* ist eine Entwicklung von (Shamir 1979) mit dem Ziel, eine Information so auf unterschiedliche Stellen zu verteilen, dass die Information nur bei Zusammenarbeit dieser Stellen wiederhergestellt werden kann. In Bezug auf elektronische Wahlen bedeutet dies, dass jeder Wähler seine Stimme auf unterschiedliche Stellen (Server, Teilnehmer) verteilen muss. Der jeweilige Teil der Stimme ist mit dem öffentlichen Schlüssel des empfangenden Servers verschlüsselt. Für die Berechnung des Ergebnisses ermittelt jeder Server die empfangenen Teilstimmen. Nur der Zusammenschluss aller Teilergebnisse ergibt dann das Endergebnis. Ein administrationsloses Protokoll, welches auf dieses Verfahren und den Homomorphismus von kryptografischen Verfahren setzt, ist in (Benaloh und Yung 1986) beschrieben. Für weitere Details und Referenzen zu den verschiedenen theoretischen Konzepten und implementierten Techniken für elektronische Wahlen sei an dieser Stelle auf die Arbeiten von (Schlifni 2000; Smith 2005; Ondrisek 2008; Volkamer 2009) verwiesen.

Die Entscheidung, in dieser Arbeit elektronische Wahlen in kleinen Gruppen als Beispiel für kollaborative Applikation zu verwenden, ist dadurch zu begründen, dass für die Realisierung dieser elektronischer Wahlen die Sicherheitsanforderungen zu erfüllen sind, die auch in der Problemstellung der Arbeit analysiert wurden. So stellen beispielsweise die im Grundgesetz verankerten Wahlgrundsätze einer allgemeinen, unmittelbaren, freien, gleichen und geheimen Wahl (vgl. GG § 38 Abs. 1) hohe Anforderungen an die genutzten Protokolle (Bundesministerium der Justiz 1949; Helbach et al. 2007; Bundesministerium der Justiz 1999; Grimm et al. 2006). Dadurch entstehen folgende Anforderungen:

- *Anonymität* – Jeder Wähler darf frei seine Stimme abgeben, er sollte seine Stimme keinem anderen mitteilen und es darf keine nachvollziehbare Zuordnung zwischen Stimme und Wähler existieren.
- *Authentifizierung* – Es dürfen nur wahlberechtigte Personen wählen. Als Konsequenz daraus muss also beim Zugang zur Wahl eine Überprüfung der Identität durchgeführt werden.

- *Integrität* – Es darf keine Veränderung einzelner Stimmen oder des Gesamtergebnisses möglich sein. Dazu gehört auch die korrekte Berechnung des Ergebnisses aus den einzelnen Stimmen.
- *Verifizierbarkeit* – Die Ergebnisse und ihre Berechnung sollen für die Wähler nachvollziehbar sein.
- *Verfügbarkeit* – Das System muss Mechanismen implementieren, die in angemessener Weise die Durchführung des Wahlvorganges ermöglichen.
- *Langfristige Aufbewahrung* – Die Ergebnisse und die Wahlunterlagen sollen für einen vorher bestimmten Zeitraum unter der Gewährleistung der anderen oben genannten Anforderungen aufbewahrt werden.

Diese Anforderungen werden je nach Szenario und den entsprechenden Regelungen mit unterschiedlichen Gewichtungen von den einzelnen Protokollen realisiert. Um einen Konsens in den Sicherheitsanforderungen für Online-Wahlen zu erzielen, wurde im Jahr 2008 ein Common Criteria-Schutzprofil vom Bundesamt für Sicherheit veröffentlicht (Common Criteria Recognition Agreement 2009). Dieses definiert einen Basissatz von Sicherheitsanforderungen an Online-Wahlprodukte. Die darin beschriebenen Anforderungen, die von allen zertifizierten Produkten zu erfüllen sind, sind (Volkamer und Vogt 2008):

- Jede Stimme ist nur dem zugehörigen Wähler bekannt (Wahlgeheimnis).
- Es muss nicht geheim gehalten werden, *welche* Wähler gewählt haben – so lange geheim bleibt, *wie* sie gewählt haben.
- Kein Wähler darf in der Lage sein, seine Wahlentscheidung Dritten gegenüber zu beweisen.
- Nur registrierte Wähler dürfen eine Stimme abgeben.
- Jeder Wähler darf nur eine Stimme abgeben.
- Während der Wahldurchführung darf kein Zwischenergebnis ermittelt werden.

Ein solches Profil definiert keine konkreten Mechanismen, welche die genannten Anforderungen sicherstellen. Dies wird erst durch die Verwendung und Kombination von bekannten Mechanismen erreicht. Zum Einsatz kommen können beispielsweise Passwörter, TANs oder biometrische Merkmale (für die Überprüfung der Identität), blinde Signaturen, Separation of Duty oder ein Homomorphismus (für die Gewährleistung der Anonymität) und Verschlüsselungsverfahren wie Zero Knowledge oder RSA (für die sichere Übertragung).

Die Auswahl derartiger Mechanismen als Grundlage für eine spontane, sichere, langfristig verbindliche Gruppenkommunikation auf Grundlage eines IM-Systems ist Gegenstand des nächsten Teils.

Teil III Integration sicherheitskritischer Anwendungen in einer IM-Umgebung

10 Überblick über das Forschungsvorhaben

In den bisherigen Kapiteln wurden der Forschungsbereich Instant-Messaging-Systeme und die dazugehörigen aktuellen Forschungsaktivitäten vorgestellt. Anhand der Charakteristika dieser IM-Systeme wurde eine Klassifizierung vorgenommen und ausgehend von einer allgemeinen Architektur eines IM-Netzes eine Sicherheitsanalyse durchgeführt. Diese Betrachtungen haben den Einsatz von IM-Systemen als *Plattform* fokussiert, die dazu dient, eine spontane, authentifizierte und langfristig verbindliche Kooperation über die Grenzen zentral organisierter Unternehmen hinweg zu realisieren. Eine Kooperation kann dabei zum Beispiel (1) eine Diskussion oder Ideensammlung (Brainstorming) sein, bei der eine verbindliche Aufbewahrung des Ergebnisses von Bedeutung ist, (2) eine elektronische Abstimmung, bei der großer Wert auf die Geheimhaltung der Stimmen gelegt wird, oder (3) eine elektronische Transaktion, bei der ein sicherer Austausch zwischen Ware und Bezahlung erfolgen soll. Dabei sollten Sicherheitsmechanismen gewählt werden, mit denen man spontan eine sichere Kommunikation etablieren kann und die gleichzeitig auch die Basis für eine langfristige Archivierung der Ergebnisse (Aussagen, Entscheidungen) bilden. Die Sicherheitsanforderungen, die durch die Forderung nach einer „sicheren Kommunikation/Kollaboration“ impliziert werden, sind die Authentifikation der Teilnehmer, die Integrität der ausgetauschten Nachrichten, die Vertraulichkeit der Nachrichten und die Verbindlichkeit von Kommunikationsergebnissen.

Wie im Kapitel Stand der Forschung gezeigt wurde (siehe Kapitel 8), existieren zwar Mechanismen und Standards, um eine sichere IM-Kommunikation aufzubauen. Diese erfüllen jedoch nicht die hier angestrebten Anforderungen:

1. Sie bieten nur partielle Lösungen, so kann z. B. nur ein Kommunikationskanal – entweder Chat oder VoIP – abgesichert werden, jedoch nicht beide.
2. Sie orientieren sich ausschließlich an einer bilateralen Kommunikation.
3. Sie sind so kompliziert und aufwendig (z. B. durch Notwendigkeit einer PKI), dass ein Einsatz für spontane Kollaboration und kleine Gruppen nicht realistisch erscheint.
4. Sie lassen relevante Sicherheitsanforderungen außer Acht, wie etwa die Nicht-Abstreitbarkeit bezüglich der Kommunikation und der Ergebnisse.

Im vorliegenden Teil III dieser Arbeit wird diese Lücke folgendermaßen gefüllt:

Es wird zunächst ein Zugangsberechtigungsmechanismus diskutiert, durch den die Überprüfung der *Authentizität der Teilnehmer* und die *Vertraulichkeit der Kommunikation* erreicht werden (Kapitel 11.1). Es werden unterschiedliche Alternativen präsentiert und bewertet. Das Ziel ist dabei die Etablierung einer kurzfristig sicheren Gruppenkommunikation. Als nächste Sicherheitsanforderungen werden die *Integrität der versendeten Nachrichten* (Kapitel 11.6) und die *Verbindlichkeit der Ergebnisse* betrachtet (Kapitel 11.10).

Anschließend wird erläutert, wie ein IM-System so erweitert werden kann, dass es als Plattform für sicherheitskritische Anwendungen dienen kann. Dazu wird eine typische IM-Architektur um die vorher erarbeiteten Sicherheitskonzepte und Schnittstellen für die Adaptierung von Kollaborationsanwendungen erweitert (Kapitel 12). Diese Konzepte (algorithmische Lösungsansätze und ein IM-System als Plattform) werden anschließend anhand der exemplarischen Kollaborationsanwendung „Entscheidungsprozesse innerhalb von Gremien“ konkretisiert (Kapitel 14). Derartige Entscheidungsprozesse eignen sich besonders gut als Beispielfall, da sie eine spontane, meist offene Diskussion mit einer geheimen und sicherheitskritischen Anwendung, dem elektronischen Wählen kombinieren. Es werden unterschiedliche Varianten dieser Entscheidungsprozesse vorgestellt. Abschließend werden die vorher erarbeiteten Konzepte in einer prototypischen Realisierung demonstriert (Kapitel 14). Durch den Einsatz der spontan angewendeten Sicherheitsmechanismen, die erst während einer IM-Kommunikation instanziiert werden, um die externe Applikation (hier den Entscheidungs- und Wahlprozess) abzusichern, soll die Eignung von IM-Systemen als Plattform für sichere langfristige Kollaboration gezeigt werden.

11 Algorithmische/Konzeptuelle Lösungsansätze

Für die weiteren Betrachtungen wird, als Modellanwendung, eine Gruppenkommunikation und -kollaboration mit n Teilnehmern $(n > 2)$ zugrunde gelegt. Es wird davon ausgegangen, dass jeder Teilnehmer über einen üblichen Instant-Messaging-Client, eine Webkamera und einen festen Internetzugang verfügt. Die Anforderung der Verfügbarkeit dieser Infrastruktur und der darauf aufsetzenden Anwendungen wird im Rahmen dieser Arbeit nur am Rande behandelt und die dadurch entstehenden Limitierungen an den entsprechenden Stellen erwähnt. Es wird angenommen, dass das IM-Netz ständig verfügbar ist, einschließlich der Verbindung zwischen Client und Server bzw. Peer-to-Peer-Verbindungen zwischen Clients. Es wird allerdings keine detaillierte Betrachtung der Verfügbarkeit vorgenommen.

In diesem Kapitel werden der Reihe nach algorithmische und konzeptuelle Lösungsansätze für folgende Sicherheitsanforderungen diskutiert:

- Authentizität der Gesprächspartner (wird durch den Zugangsberechtigungsmechanismus im Kapitel 11.1 realisiert)
- Vertraulichkeit einer Gruppenkommunikation (wird durch den Zugangsberechtigungsmechanismus im Kapitel 11.1 erreicht)
- Integrität der Kommunikationsinhalte (wird durch die Integritätsmechanismen im Kapitel 11.6 gewährleistet)
- Verbindlichkeit der Kommunikationsergebnisse (wird durch ein Rollenvergabe-Konzept im Kapitel 11.3 sichergestellt)

11.1 Authentizität der Gesprächspartner, Vertraulichkeit der Kommunikation

Jedes Instant-Messaging-System bietet einen Zugangsberechtigungsmechanismus einschließlich Registrierung und Authentifikation an. Dieser erlaubt es einem Benutzer, das IM-System zu nutzen und über diese Plattform erreichbar zu sein. Die allgemeine Funktionsweise dieses Zugangsmechanismus (erste Phase, Kapitel 11.1.1) und deren Anpassung an die gesetzten Sicherheitsanforderungen für die Realisierung der vorliegenden Szenarien (zweite Phase, Kapitel 11.1.2) ist Gegenstand der folgenden Betrachtungen. Das Hauptziel dieses Mechanismus ist es, die *Authentizität* der kommunizierenden Parteien zu gewährleisten. Durch die vorgeschlagene Lösung wird auch eine weitere Sicherheitsanforderung, die der *Vertraulichkeit* der Gruppenkommunikation, realisiert.

11.1.1 Erste Phase der Zugangsberechtigung

Die Überprüfung der Authentizität durch den „eingebauten“ Authentifikationsmechanismus des zugrundeliegenden IM-Systems ist der erste Schritt dieses Prozesses (vgl. Kapitel 6). Dafür benötigt der Benutzer ein Benutzerkonto (Benutzername und Passwort), welches auch unabhängig von den vorliegenden Szenarien angelegt werden kann.

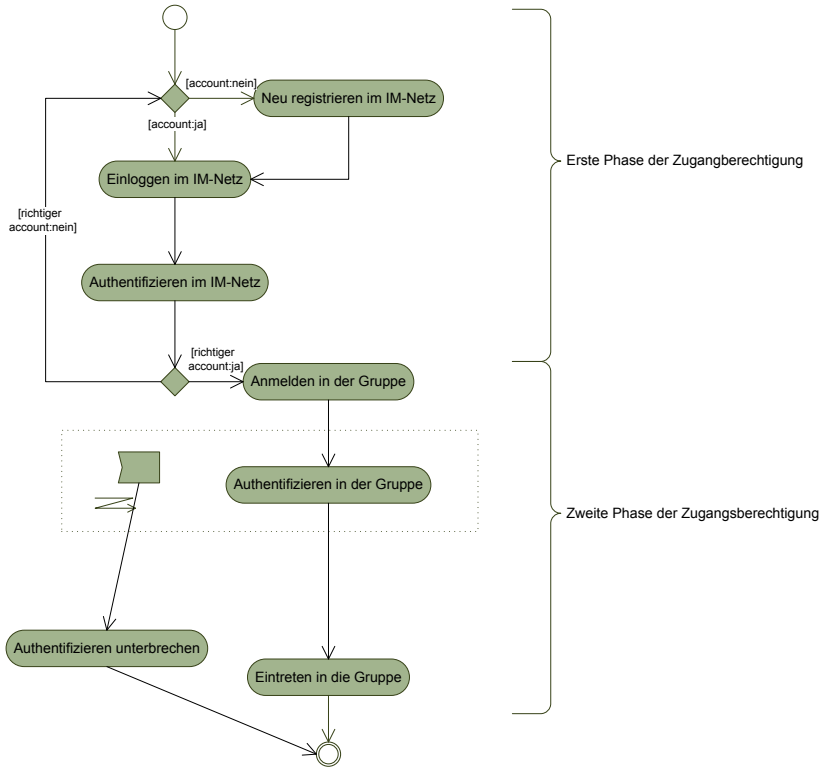

Abbildung 16 – Ablauf der Zugangsberechtigung

Er startet die Gruppensitzung, indem er entweder ein neues Benutzerkonto anlegt oder sich mit seinem existierenden Profil anmeldet. Ist die Authentifizierung im IM-Netz erfolgreich (vgl. erste Phase in Abbildung 16, „richtiger account: ja“), kann der Benutzer eine Chat-Konferenz

initiieren oder in eine existierende Konferenz eintreten. Ist die erste Phase des Authentifizierungsvorgangs nicht erfolgreich, hat der Benutzer z. B. eine falsche Kennung eingegeben, muss er entweder seine Eingaben wiederholen oder eine neue Kennung beantragen.

Um Nachrichten zwischen den registrierten Nutzern verschicken zu können, muss jeder Nutzer eine Kontaktliste mit den Benutzerkennungen seiner gewünschten Kommunikationspartner aufbauen. Dafür gibt es zwei Wege:

- Die erste Möglichkeit ist die Verwendung der Suchfunktion des IM-Systems, zum Beispiel durch Eingabe des Namens oder Wohnortes des gesuchten Benutzers. Diese Vorgehensweise ist aus Sicht der Anwendungssicherheit problematisch, da die Suchergebnisse nicht eindeutig sind und auch ein gefundener, anscheinend korrekter Treffer nicht dem gewünschten Gesprächspartner entsprechen muss. Die Möglichkeit, gefundene Treffer eindeutig zu identifizieren, hängt von der Bereitwilligkeit der Teilnehmer ab, genügend persönliche Informationen preiszugeben. Es kann also passieren, dass die falsche Person angesprochen wird. Es ist darüber hinaus denkbar, dass ein Angreifer absichtlich ein passendes, gefälschtes Profil anlegt, um selbst „gefunden“ zu werden.
- Eine zweite, verlässlichere Möglichkeit ist die Weitergabe der genauen Benutzerkennung über ein anderes Medium (etwa über eine persönliche Mitteilung, E-Mail, Telefon, Visitenkarte oder im Briefkopf).

In beiden Fällen kann die Authentizität des Nutzers nicht ohne weiteres garantiert werden. Dazu kommt, dass aktuelle IM-Systeme die Zuordnung von echten Personen zu gewählten Kennungen häufig nicht überprüfen. Jeder darf eine Kennung anlegen und den entsprechenden Namen frei wählen (soweit dieser noch nicht vergeben ist). Es ist somit möglich, dass ein „Andreas Maier“ sich eine Kennung mit Namen „peter.mueller“ erstellt.

Die Überprüfung der Authentizität der Teilnehmer ist aber eine wichtige Sicherheitsanforderung für die Durchführung von sicheren Kollaborationsszenarien. Es sollten nur Teilnehmer der IM-Sitzung beitreten, die gemäß den Regelungen und Vorschriften der jeweiligen Kooperation über diese Berechtigung verfügen. So kann beispielsweise das Risiko eines Man-in-the-Middle-Angriffs oder einer Identitätsvortäuschung minimiert werden.

11.1.2 Zweite Phase der Zugangsberechtigung

Da also der übliche Authentifikationsmechanismus eines IM-Systems für die Durchführung einer sicheren Kollaboration nicht ausreicht und angestrebt wird, die hier technisierten Abläufe so nah wie möglich an einer herkömmlichen Präsenz-Gruppenkommunikation und -kollaboration zu realisieren, wird an dieser Stelle eine weitere Zugangsberechtigungsphase eingeführt. Der Ansatz nutzt eine Kombination aus (zweite Phase der Zugangsberechtigung in Abbildung 16):

- einem Diffie-Hellman-Schlüsselaustausch,
- einer Absicherung des verwendeten Übertragungskanals mit Hilfe des DH-Schlüssels und
- der persönlichen Wiedererkennung der Gesprächspartner im Videobild.

Zusammengefasst läuft diese zweite Zugangsberechtigungsphase folgendermaßen ab:

Zwei Teilnehmer nutzen den Chat-Kanal als erstes Kommunikationsmedium. Bei der Einladung zu einer gemeinsamen Chat-Konferenz wird ein Diffie-Hellman-Key-Agreement ausgeführt (Diffie, van Oorschot und Wiener 1992). Nach diesem Aushandeln von asymmetrischen Schlüsseln (Parameter des DH-Algorithmus) generiert jeder Kommunikationspartner den DH-Schlüssel; es handelt sich dabei um einen symmetrischen Schlüssel. Mit diesem gemeinsamen Schlüssel haben die Gesprächspartner jetzt die Möglichkeit, die weitere Kommunikation zu verschlüsseln.

In einem weiteren Schritt bauen sie eine Video-Konferenz auf. Dadurch haben beide die Möglichkeit, ihren Gesprächspartner anhand des Videobilds zu erkennen. Um sicherzustellen, dass man mit dem gewünschten Partner – den man auch im Video-Kanal als Bild sieht – gerade auch das Diffie-Hellman-Verfahren durchgeführt hat, wird eine weitere Maßnahme implementiert: Beide Gesprächspartner berechnen einen Hashwert des gerade generierten symmetrischen Schlüssels und lesen sich diesen gegenseitig vor. Stimmen beide Hashwerte überein, dann haben die Gesprächspartner einen weiteren Beweis erbracht, dass die Kommunikation nicht durch einen Dritten manipuliert wurde. Durch dieses Verfahren werden also folgende Ziele erreicht: Die Teilnehmer sind gegenseitig authentifiziert, sie verwenden einen integeren Schlüssel und die Kommunikation ist zuverlässig und exklusiv, das bedeutet integer und vertraulich.

Im Folgenden wird dieses Verfahren genauer erläutert und dessen Anpassung für Gruppen diskutiert.

11.1.3 Zugangsberechtigung bei zwei Teilnehmern

Wie im Kapitel 11.1.1 erläutert wurde, wird in der ersten Phase der Überprüfung der Zugangsberechtigung (also bei der Anmeldung und ersten Überprüfung der Identität) das Instant-Messaging-Netz und dessen Authentifikationsserver genutzt (siehe Abbildung 17, Einloggen Person A bzw. B).

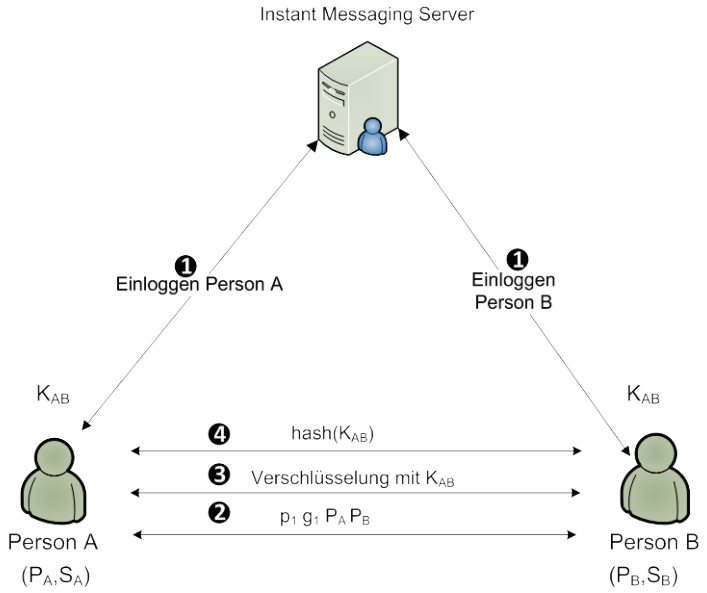

Abbildung 17 – Zugangsberechtigung bei zwei Teilnehmern

Nachdem die Teilnehmer A und B Schritt ❶ durchgeführt haben, sind beide mit ihren Clients in das IM-Netz eingeloggt (vgl. dazu nochmal den früher gezeigten Ablauf in Abbildung 16). Durch Nutzung des Chat-Kanals zur Datenübertragung über das öffentliche Netz sind sie nun in der Lage, den Diffie-Hellman-Schlüsselaustausch durchzuführen.

Nachdem sich die Teilnehmer auf die öffentlichen Parameter p_1 und g_1 geeinigt haben (Schritt ❷), erzeugt jeder Teilnehmer X einen privaten Schlüssel (S_X) und leitet mit Hilfe der Parameter daraus seinen öffentlichen Schlüssel (P_X) ab:

$$(1)\ P_A = {g_1}^{S_A} mod\ p_1$$

$$(2)\ P_B = {g_1}^{S_B} mod\ p_1$$

Daraus berechnet jeder Teilnehmer für sich einen symmetrischen Schlüssel K_X

$$(3)\ K_A = {P_B}^{S_A} mod\ p_1$$

$$(4)\ K_B = {P_A}^{S_B} mod\ p_1$$

Diese beiden Schlüssel sind identisch, denn es gilt

$$(5)\ K_A = {P_B}^{S_A} mod\ p_1 = ({g_1}^{S_B} mod\ p_1)^{S_A} mod\ p_1 = {g_1}^{S_B S_A} mod\ p_1 = ({g_1}^{S_A} mod\ p_1)^{S_B} mod\ p_1 = {P_A}^{S_B} mod\ p_1 = K_B$$

Somit haben die zwei Kommunikationspartner über einen offenen Kanal einen gemeinsamen symmetrischen Schlüssel vereinbart. Dieser Schlüssel wird verwendetet, um mit Hilfe eines symmetrischen Verschlüsselungsalgorithmus (z. B. AES) eine vertrauliche Verbindung aufzubauen (Schritt ❸).

Mit diesem Mechanismus können zwar verschlüsselte Nachrichten versendet werden, es wird aber nicht die Authentizität des Senders und des Ursprungs der Nachrichten gewährleistet.

Ein typischer Angriff auf das DH-Verfahren ist ein Man-in-the-Middle-Angriff. Angenommen eine Person E will einen solchen Angriff durchführen. Dazu generiert der Angreifer E zwei Schlüsselpaare und positioniert sich im Netzwerk zwischen A und B. Er tritt dann mit seinen eigenen Schlüsseln jeweils einem Kommunikationspartner (A oder B) gegenüber und lässt die beiden dabei in dem Glauben, dass sie direkt miteinander reden, obwohl die Kommunikation in Wirklichkeit indirekt über ihn abgewickelt wird. Dadurch werden zwei unterschiedliche symmetrische Schlüssel generiert: der Schlüssel K_{AE} für die Verbindung zwischen Person A und Person E und der Schlüssel K_{BE} für die Verbindung zwischen Person B und Person E. Ohne weitere Maßnahmen, bemerken die beiden ursprünglichen Kommunikationspartner A und B keine Unregelmäßigkeiten in der Kommunikation.

Eine Maßnahme, die es erlaubt, einem solchen Angriff entgegenzuwirken, ist die Nutzung einer Public-Key-Infrastruktur (PKI). Jeder Kommunikationspartner lässt über eine zentrale vertrauenswürdige Instanz (Certificate Authority) den eigenen öffentlichen Schlüssel zertifizieren und veröffentlichen. Durch die Erstellung und den Austausch von Zertifikaten sowie die Signierung der Nachrichten, kann der Empfänger die Signierung jeder Nachricht und damit den Ursprung der Nachricht überprüfen.

Die Anwendung einer PKI einschließlich digitalen Signaturen ist aber aus folgenden Gründen nicht geeignet für eine spontane Interaktion im IM-Netz, wie sie hier angestrebt wird:

- Der Aufbau und die Integration einer PKI in die weitere Kommunikationsinfrastruktur sind mit einem nicht unerheblichen zeitlichen und monetären Aufwand verbunden.
- Damit die digitalen Signaturen einer PKI genutzt werden können, müssen *alle* Teilnehmer über die entsprechenden Schlüssel verfügen. Das bedeutet, eine Kooperation mit Teilnehmern aus verschiedenen Organisationen kann nur dann durchgeführt werden, wenn die entsprechenden Organisationen auch über eine PKI verfügen.
- Um die Authentizität gewährleisten zu können, müssen die Teilnehmer schon vor der eigentlichen Kommunikation die öffentlichen Schlüssel austauschen. Die Langwierigkeit des Prozesses zur Beantragung, Erstellung und zum Austausch von Zertifikaten ist für die vorliegenden Anwendungsfälle ein limitierender Faktor.

Im Rahmen dieser Arbeit wird daher ein Verfahren für die Identitätsüberprüfung der Teilnehmer vorgeschlagen, das nicht auf einer PKI basiert, sondern auf der entsprechenden Vorgehensweise in einer Präsenz-Kollaboration: Die Personen werden anhand der körperlichen Merkmale und persönliche Bekanntheit („Wiedererkennung“) erkannt. Das Aussehen (Bild) bekannter Personen und die Existenz des Namens auf der Anwesenheitsliste einer Konferenz entscheiden über die Zulassung zur Gruppe. Diese Wiedererkennung wird in einer Instant-Messaging-Architektur über den Audio- oder Video-Kanal realisiert.

Nachdem die Teilnehmer A und B, wie oben beschrieben, einen verschlüsselten Chat-Kanal aufgebaut haben, schalten sie die Webkameras ein und eröffnen mit den gleichen Verschlüsselungsparametern einen Video-Kanal. Jetzt kann jeder seinen Gesprächspartner über das Video sehen und ihn dementsprechend „wiedererkennen“. Bis jetzt gibt es aber keinen Zusammenhang zwischen dem Video-Kanal (dem Bild des Gesprächspartners) und der gesicherten Chat-Kommunikation. Konkret bedeutet dies, dass ein Angreifer als Man-in-the-Middle die Chat-Nachrichten manipulieren könnte, obwohl der gewünschte Partner in der zugehörigen Videoaufnahme gezeigt wird. Daher ist es notwendig hier eine Verknüpfung zwischen dem abgesicherten Chat-Kanal und dem Video-Bild herzustellen. Dies wird durch den Sitzungsschlüssel K_{AB} erreicht. Jeder Teilnehmer hat auf seiner Seite einen symmetrischen Schlüssel berechnet. Wenn die Kommunikation samt Schlüsselaustausch integer ist, sich also kein Angreifer E zwischen A und B geschoben hat, muss gelten:

$$(6)\quad K_A = K_B = K_{AB}$$

Um dies zu überprüfen, berechnet jeder Teilnehmer einen Hashwert des errechneten sels $H(K_A)$ bzw. $H(K_B)$ (vgl. Abbildung 17, Schritt ❹). Die jeweiligen Hashwerte werden über Audio-Video-Kanal durch Vorlesen verglichen. Die Teilnehmer überprüfen dabei die Lippenbewegungen des Gesprächspartners in Verbindung mit den gesprochenen alphanumerischen Bestandteilen des Hashwertes. Stimmen die von beiden Teilnehmern vorgelesenen Werte überein, bedeutet dies, dass die Kommunikation integer ist und beide über den gleichen symmetrischen Schlüssel $K_A = K_B = K_{AB}$ verfügen. Sind die Werte unterschiedlich bedeutet dies, dass ein Man-in-the-Middle-Angriff stattgefunden hat. Daraufhin wird die Kommunikation unterbrochen.

Bei diesen Überlegungen wird davon ausgegangen, dass es mit den heutigen Entwicklungen im Forschungsbereich der Spracherkennung und Lippenbewegung nicht möglich ist, in Real-time

eine Manipulation der gesprochenen Worte und des dazugehörigen Videobildes unter Bewahrung der Synchronität vorzunehmen. Selbst der Versuch einer solchen Manipulation unter Vorgabe von beliebiger Zeit ist aktuell nur sehr eingeschränkt realisierbar (Kanellos 2006; Wang 2009). Für den Anwendungsbereich dieser Arbeit würde dies bedeuten dass bei der spontanen Videoüberprüfung der Identität ein Angreifer z. B. dem echten Teilnehmer andere Hashwerte in den Mund legt. So beschreiben (Ezzat, Geiger und Poggio 2002) ein Projekt, in dem im Bild dargestellte Teile des Gesichtes, z. B. der Ausschnitt, der den Mund zeigt, so manipuliert werden, dass andere Bewegungen zu sehen sind und andere Sprachinhalte geäußert werden. Diese Manipulation ist aber noch nicht für einen Videostream in Echtzeit möglich, da das Prozedere des Ausschneidens und Änderns dieser Gesichtsmerkmale sehr langwierig ist. Somit wird für das vorliegende Konzept der Aufbau eines verschlüsselten Audio-Video-Kanals als ausreichend sicher betrachtet.

Zusammenfassend lässt sich also sagen, dass der Beitrag dieser Arbeit (an dieser Stelle) darin besteht, dass Chat-Kanal und Audio-Video-Kanal so kombiniert werden, dass eine sichere IM-Konferenz unter Gewährleistung der Authentizität der Teilnehmer aufgebaut werden kann. Im Gegensatz zum Konzept von (Zimmermann, Johnston und Callas 2007) wird hier die Chat-Verbindung sozusagen als „Signalisierungsprotokoll“ für den Austausch der Diffie-Hellman-Parameter und die Bestimmung des symmetrischen Schlüssels verwendet. Der dadurch entstandene Schlüssel wird für die Verschlüsselung des Audio-Video-Kanals verwendet. Somit wird eine Integration dieser und weiterer Kanäle realisiert. Weitere Unterschiede zwischen der vorliegenden Arbeit und dem Stand der Forschung werden in der laufenden Arbeit an entsprechender Stelle erwähnt.

11.2 Mögliche Ansätze für die Gewährleistung der Authentizität und Vertraulichkeit in einer Gruppe

Bisher wurde der prinzipielle Zugangsberechtigungsmechanismus für zwei Teilnehmer beschrieben. Da aber eine Gruppenkommunikation und die später vorgestellten konkreten Szenarien (Kapitel 13) nur selten einer bilateralen Kommunikation entsprechen, liegt in den folgenden Kapiteln die Herausforderung darin, diesen Mechanismus für eine größere Gruppenkommunikation anzupassen. Hier sind unter anderem die folgenden Fragen zu beantworten:

- Wie verhalten sich die Schlüsselgenerierung und der Schlüsselaustausch, wenn mehr als zwei Teilnehmer an einer Kommunikation teilnehmen?
- Wie findet das Aushandeln des Sitzungsschlüssels statt und wie viele Schlüssel sind für die Absicherung der Kommunikation notwendig?
- Wie kann man mehrere Personen gleichzeitig verifizieren?
- Wie werden weitere Kommunikationskanäle integriert und abgesichert?

Es werden unterschiedliche Varianten für die Schlüsselgenerierung und -überprüfung vorgestellt und bewertet (Meletiadou 2009). Der Mechanismus zur Überprüfung der Zugangsberechtigung besteht aus dem Prozess des Diffie-Hellman-Schlüsselaustauschs, der Echtheitsüberprüfung der

Schlüssel durch die visuelle Erkennung der Teilnehmer, der Verschlüsselung der Nachrichten und der eigentlichen Datenübertragung (siehe auch Abbildung 18).

Die einzelnen Bestandteile sind folgendermaßen zu verstehen:

- Wer führt mit wem einen **DH-Schlüsselaustausch** durch?

  Dabei geht es um die Topologie des DH-Austausches, also die Frage, zwischen welchen Parteien die benötigten DH-Parameter und öffentlichen Schlüssel ausgetauscht werden. In Abbildung 18 sind dazu die Alternativen (Jeder mit jedem, Zentrale Instanz, Web-of-Trust) dargestellt.

- Wer führt mit wem eine Echtheitsüberprüfung der Schlüssel durch?

  Nach dem Schlüsselaustausch kann, genau wie im einfachen Verfahren für zwei Teilnehmer, die Identität der Teilnehmer durch visuelle Wiedererkennung überprüft werden. Je nach Variante müssen dabei aber nicht für alle paarweisen Kombinationen solche Überprüfungen stattfinden sondern man kann sich auf eine ausgewählte Stichprobe beschränken, wenn das dadurch erreichte Sicherheitsniveau akzeptabel ist.

- Welche Schlüssel werden für die Vertraulichkeit der Kommunikation genutzt (**Verschlüsselung**)?

  Diese Frage hängt mit der Übertragung der Nachrichten und den durchgeführten DH-Key-Agreements zusammen. Je nach Verfahren gibt es die Möglichkeit, für jede zweiseitige Kommunikation einen Schlüssel K_{XY} oder einen gemeinsamen Schlüssel K_{ALL} für die gesamte Kommunikation zu berechnen.

- Welche Topologie der **Datenübertragung** wird verwendet?

  Hiermit wird festgelegt, ob die Kommunikationsdaten (Chat und Video) über Peer-to-Peer-Verbindungen zwischen den Teilnehmern versendet werden oder ob eine zentrale Instanz einen Weiterleitungsdienst (evtl. inklusive Videomixer) anbietet.

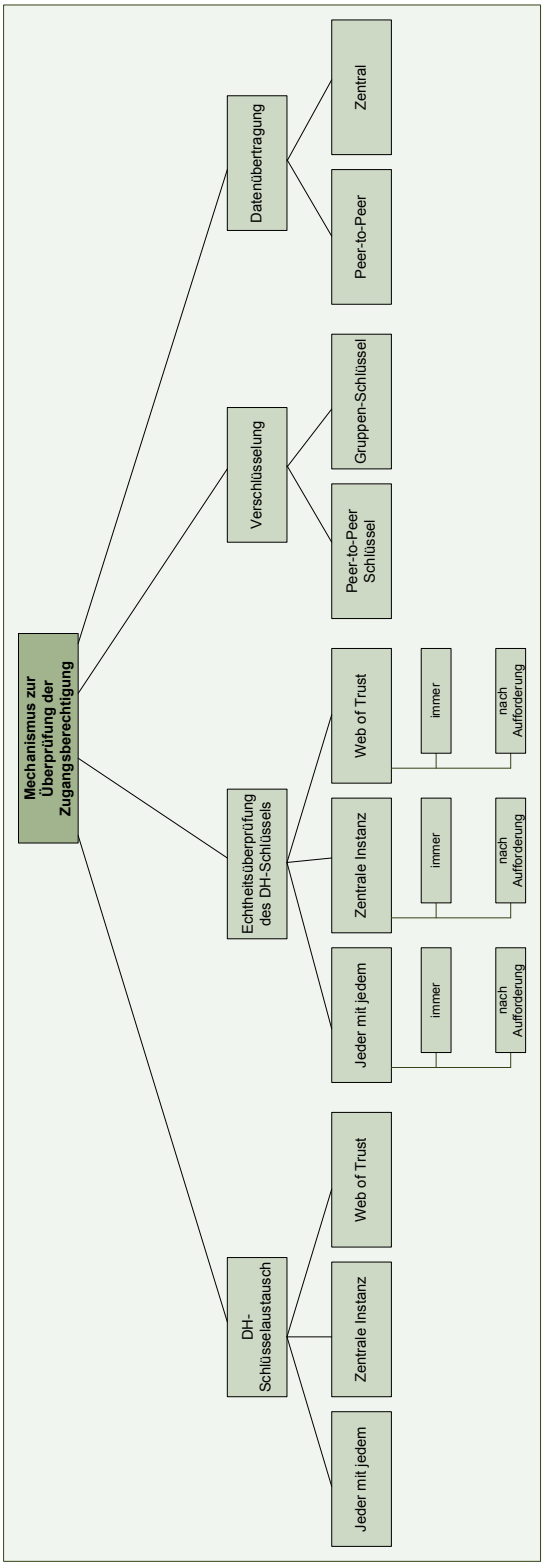

Abbildung 18 – Bestandteile der Zugangsberechtigung zur Anwendergruppe

Aus der Abbildung 18 entstehen unterschiedliche Kombinationen eines Zugangsmechanismus und des entsprechenden Ablaufs, welche für die Durchführung von verschiedenen kollaborativen Anwendungen verwendet werden können. Um je nach Anwendung die richtigen Bestandteile für die Realisierung der Überprüfung der Authentizität, die Verschlüsselung der Daten und

die Datenübertragung zu wählen, muss einerseits die Art der Anwendung und deren Ablauf und andererseits das erzielte Sicherheitsniveau berücksichtigt werden. Für Letzteres wird das sogenannte Vertrauensmodell eingeführt.

11.2.1 Vertrauensmodell

Das Vertrauensmodell legt fest, wie viele Komponenten vertrauenswürdig sein müssen, um einen sicheren Ablauf der Kommunikation gewährleisten zu können. Dabei wird zwischen ehrlichen und nicht ehrlichen Komponenten (Teilnehmern) unterschieden. Ein ehrlicher Teilnehmer verhält sich gemäß der Sicherheitspolicy des Systems. Er führt den Protokollablauf korrekt durch, schützt seine Daten (einschließlich der generierten und berechneten Sicherheitselemente) und hat nicht die Absicht, die Kommunikation zu manipulieren. Alle anderen gelten als unehrliche Teilnehmer. Dabei spielt es keine Rolle, ob ein Teilnehmer die Regeln absichtlich oder aus Fahrlässigkeit verletzt.

Für jedes sicherheitstechnische Verfahren kann ein Vertrauensmodell ermittelt werden. Ein Verfahren hat ein *Vertrauensmodell von* x, wenn x ehrliche Teilnehmer genügen, um jeden internen Angriff auf dieses Verfahren zu erkennen. Dies bedeutet umgekehrt, dass bei einer Gruppe mit insgesamt n Teilnehmern, die übrigen $n - x$ keinen unbemerkten Angriff durchführen können. Man beachte, dass es für die Verhinderung eines Angriffs bei den hier vorgesehenen Anwendungsszenarien ausreichend ist, wenn der Angriff zwar zunächst erfolgreich ist (z. B. wenn ein übertragenes Votum manipuliert und zunächst als legitimes Votum gezählt wird), aber im Nachhinein erkannt wird (weil dann z. B. das Ergebnis einer Wahl als ungültig erklärt wird).

Für eine Gruppe von n Teilnehmern, kann das Vertrauensmodell folgende Werte annehmen:

$x = 1$	Es genügt bereits *ein* ehrlicher Teilnehmer, um einen Angriff zu erkennen. Selbst wenn alle anderen $n - 1$ Teilnehmer unehrlich sind und zusammenarbeiten kann der eine ehrliche Teilnehmer, die Manipulation immer noch entdecken.
$1 < x < n$	x Teilnehmer müssen kollaborieren, um eine authentifizierte und vertrauliche Kommunikation gewährleisten zu können und einen Angriff zu verhindern.
$x = n$	*Alle* Teilnehmer müssen kollaborieren, um eine Manipulation zu erkennen. Umgekehrt bedeutet dies, dass bereits ein unehrlicher Teilnehmer ausreicht, um die Kommunikation anzugreifen.

Darüber hinaus gibt es noch einen besonderen Fall des Vertrauensmodells, wenn *keiner* der Teilnehmer notwendig ist, um einen internen Angriff zu erkennen oder zu verhindern, weil ein Angriff technisch nicht möglich ist. Man könnte in diesem Fall sagen, dass das Verfahren ein Vertrauensmodell von 0 hat.

Man beachte außerdem, dass es für das Vertrauensmodell nicht sinnvoll ist, von Werten größer als n zu sprechen, da die Gruppe nur n Teilnehmer hat. Sobald alle n Teilnehmer ehrlich sind (d. h. der letzte Fall in der oben stehenden Auflistung vorliegt), gibt es keinen anderen Teilnehmer mehr, der einen internen Angriff versuchen könnte.

Im Allgemeinen ist x nicht nur eine einfache Zahl sondern ein Vektor, dessen Bestandteile sich auf verschiedene Teilnehmer-Rollen beziehen. So drückt beispielsweise ein k-resilience Wert von $(1 + 1\ out\ of\ \{Signierer, Aufbewahrer\})$ aus, dass 1 ehrlicher Signierer und 1 ehrlicher Aufbewahrer ausreichen, um das gewünschte Sicherheitsziel der verbindlichen langfristigen Speicherung zu realisieren. Dies ist der k-resilience Wert des Vertrauensmodells, welches im Abschnitt 11.4.4 erläutert wird.

Somit kann man die Überlegungen zum Zugangsmechanismus (vgl. Abbildung 18) um das Vertrauensmodell erweitern. Das resultierende Modell kann in zwei Richtungen interpretiert werden: Man kann entweder (1) erst ein gewünschtes Vertrauensmodell bestimmen und anhand dessen die gewünschte Variante des Zugangsberechtigungsmechanismus wählen oder (2) erst die Variante bestimmen und dann das daraus resultierende Vertrauensmodell betrachten.

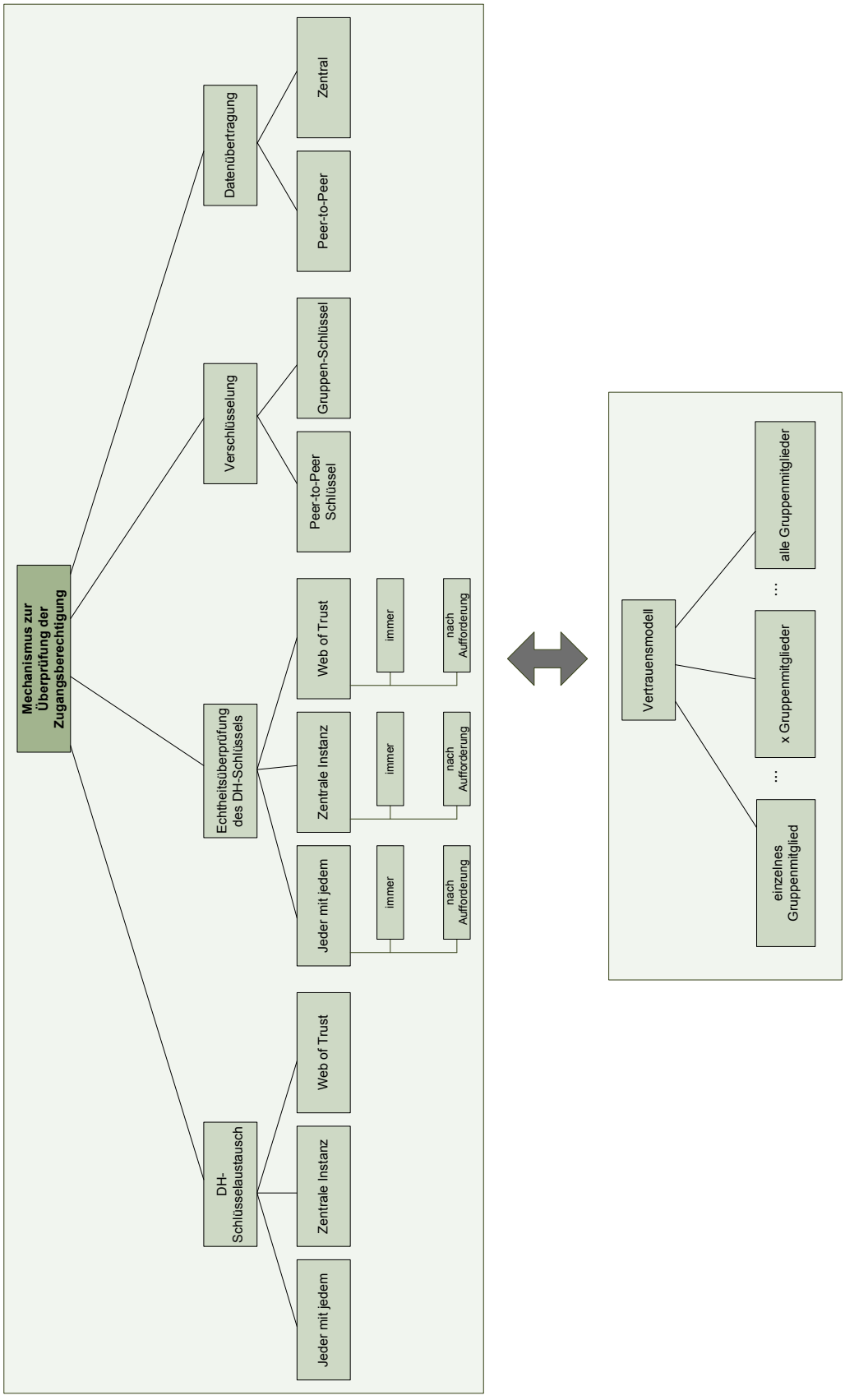

Abbildung 19 – Bestandteile der Zugangsberechtigung bzgl. Anwendergruppe u. Vertrauensmodell

Dabei sind nicht alle Kombinationen der obigen Darstellung realisierbar. Zum Beispiel kann eine Überprüfung nach Aufforderung nur dann stattfinden, wenn vorher in dieser Verbindung das entsprechende Schlüsselaustauschverfahren durchgeführt wurde.

Es werden folgende Varianten behandelt:

- „Jeder mit jedem“
- „Jeder mit jedem nach Aufforderung“
- „Zentrale Instanz“
- „Zentrale Instanz nach Aufforderung“
- „Jeder mit jedem nach Aufforderung und zentrale Pflicht-Überprüfung“
- „Web-of-Trust“
- „Web-of-Trust nach Aufforderung“

Wie diese Varianten zu verstehen sind, wird in den kommenden Abschnitten genauer erläutert. An dieser Stelle (vgl. Abbildung 20) wird zunächst ein Überblick über die Varianten (Spalten, z.B. „Jeder mit Jedem“) und ihre Merkmale (Zeilen, z.B. verschiedene Formen von „DH-Schlüsselaustausch“) gegeben

	Jeder mit jedem	Jeder mit jedem nach Auf-forderung	Zentrale Instanz	Zentrale Instanz nach Auf-forderung	Jeder mit jedem nach Auf-forderung, Pflicht-Überprüf.	Web of Trust	Web of Trust nach Auf-forderung
DH-Schlüsselaustausch							
Jeder mit Jedem	●	●			●		
Zentrale Instanz			●	●			
Web of Trust						●	●
Echtheit des DH-Schlüssels							
Jeder mit Jedem	●						
Zentrale Instanz			●	●	●		
Web of Trust						●	●
nach Aufforderung		●		●	●		●
Verschlüsselung							
Peer-to-Peer-Schlüssel	●	●			●	●	●
Gruppen-Schlüssel	●	●	●	●	●	●	●
Datenübertragung							
Peer-to-Peer	●	●		●	●	●	●
Zentral			●				
Vertrauensmodel							
Kein Teilnehmer			●				
Ein Teilnehmer	●	●		●	●		●
Mehrere Teilnehmer						●	

Abbildung 20 – Überblick der Varianten

11.2.2 Variante „Jeder mit jedem“

In dieser Variante wird der Authentifikationsprozess für eine Gruppe mit n Teilnehmern realisiert. Dazu wird das Konzepts der bilateralen Authentifikation (Kapitel 11.1.3) auf jede paarweise Verbindung angewendet. Genau wie bei einer einzelnen paarweisen Verbindung werden für jedes Paar ein Diffie-Hellman-Key-Agreement und eine gegenseitige Erkennung durchgeführt (siehe Abbildung 21).

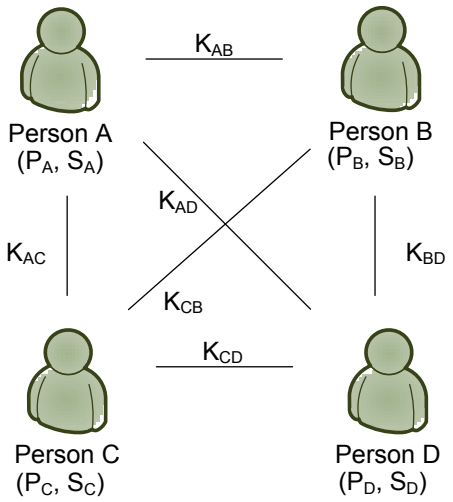

Abbildung 21 – Authentifikation „Jeder mit jedem“

So wird z. B. zwischen den Teilnehmern A und B der erste Sitzungsschlüssel K_{AB} über den Chat-Kanal ausgehandelt. Dieser Schlüssel wird genutzt, um den Kommunikationskanal mit Hilfe eines symmetrischen Verschlüsselungsalgorithmus zu verschlüsseln. Die Teilnehmer bauen jeweils paarweise eine Video-Konferenz auf und lesen sich gegenseitig die entsprechenden Hashwerte der Schlüssel vor. Somit können beide die Identität des Gesprächspartners verifizieren. Sind die Hashwerte gleich, bedeutet dies, dass die so entstandene Kommunikation nicht nur verschlüsselt, sondern auch mit der richtigen Person etabliert ist *(Echtheitsüberprüfung des DH-Schlüssels = Jeder mit jedem).*

Jeder Teilnehmer der Gruppe muss mit jedem anderen Teilnehmer einen Sitzungsschlüssel K erzeugen. Es sind $n * (n-1)/2$ bilaterale Sitzungsschlüssel und $n * (n-1)/2$ gegenseitige Erkennungsprozeduren notwendig, um jede Verbindung zwischen den Mitgliedern zu verschlüsseln und eine gegenseitige Authentifikation durchzuführen. Bei einer Gruppe mit vier Teilnehmern (wie in Abbildung 21 gezeigt) werden insgesamt sechs Verbindungen aufgebaut und überprüft. Jeder Teilnehmer muss dabei drei verschlüsselte Kanäle bedienen (sowohl für die Audio-Video-Kommunikation als auch für die Chat-Kommunikation). So muss zumindest für das Vorlesen der Hashwerte jede Nachricht drei Mal (verschiedene Schlüssel, verschiedene Verbindungen) verschlüsselt bzw. entschlüsselt werden *(Schlüsselaustausch = Jeder mit jedem).*

Für den weiteren Nachrichtenaustausch können entweder die unterschiedlichen paarweisen Schlüssel verwendet werden ($K_{AB}, K_{AC}, K_{AD}, K_{BC}, K_{BD}, K_{CD}$) oder die Gruppe kann sich auf einen gemeinsamen Sitzungsschlüssel (K_{ALL}) einigen, der über die existierenden sicheren Verbindungen verteilt wird. *(Verschlüsselung = Peer-to-Peer oder Gruppen-Schlüssel).*Alle Nachrich-

ten werden gemäß einer Peer-to-Peer-Topologie übertragen *(Datenübertragung = Peer-to-Peer)*.

Betrachtet man das Vertrauensmodell dieser Alternative, reicht ein ehrlicher Teilnehmer, um eine eventuelle Manipulation aufzudecken. Er würde durch die gegenseitige verpflichtende Überprüfung der Teilnehmer die unrechtmäßige Teilnahme einer Person an einer Sitzung erkennen *(Vertrauensmodell = ein Teilnehmer)*.

Eine weitere Möglichkeit dieser Variante ist, eine "One-to-one-Kommunikation" (flüstern) zwischen den Mitgliedern, solange die Kommunikation über die separaten bilateralen Schlüssel und nicht über einen gemeinsamen Gruppenschlüssel stattfindet. Bei der praktischen Umsetzung ist dabei zu beachten, dass die Eingabegeräte (Mikrofon, Videokamera) entsprechend nur für diese „geflüsterte" Kommunikation verwendet und die Daten nicht an andere Teilnehmer gesendet werden. Es kann also immer nur mit *einem* anderen Teilnehmer „geflüstert" werden.

Dagegen ist eine geheime und verschlüsselte Chat-Kommunikation zwischen Partnern einfacher realisierbar. So kann jederzeit eine weitere Chat-Kommunikation parallel zu der existierenden Chat-Kommunikation etabliert werden: Es reicht aus, ein zweites Chat-Fenster – außerhalb des Chatraums – mit dem gewünschten Partner zu öffnen und dort „flüsternd" Informationen weiterzugeben. Diese Verbindung könnte wiederum mit einem DH-Verfahren verschlüsselt werden. Jedoch erlauben aktuelle IM-Systeme derzeit keine zwei voneinander unabhängigen Audio-Video-Verbindungen, um damit die Hashwerte vergleichen zu können. Die hier beschriebene Variante des Verfahrens ist in der folgenden Abbildung 22 dargestellt. Dabei sind die selektierten Bestandteile grün und die eliminierten Bestandteile grau abgebildet.

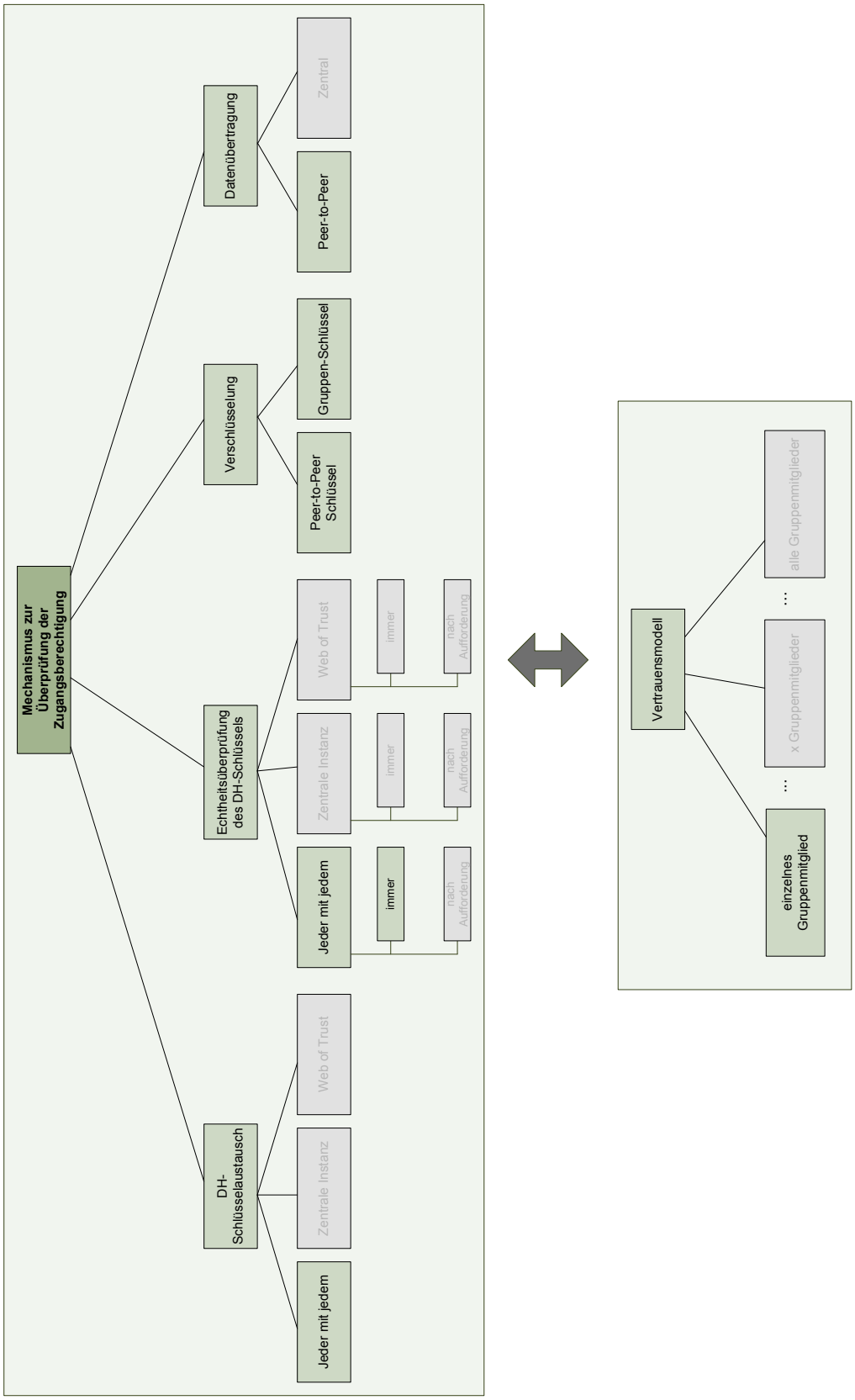

Abbildung 22 – Variante „Jeder mit jedem“

11.2.3 Variante „Jeder mit jedem nach Aufforderung“

Das vorher vorgestellte Verfahren kann variiert werden, wenn die Überprüfung der Authentizität nur nach Aufforderung stattfindet (im Gegensatz zu „immer“, d. h. mit jedem Teilnehmer). Ein Teil der Zugangsberechtigung – das DH-Key-Agreement und die bilaterale Verschlüsselung – erfolgt genau wie, in der vorherigen Variante (Kapitel 11.2.2, gestrichelte Linien). Der zweite Teil – die paarweise Wiedererkennung zwischen allen Mitgliedern – entfällt. Stattdessen findet eine Wiedererkennung über den Audio-Video-Kanal nur dann statt, wenn eine explizite Aufforderung von einem Teilnehmer ausgesprochen wird. Dies kann entweder direkt am Anfang der Gruppensitzung oder während einer gestarteten Sitzung geschehen. In der Abbildung wird dies durch die durchgezogene Linie zwischen Teilnehmer A und D dargestellt.

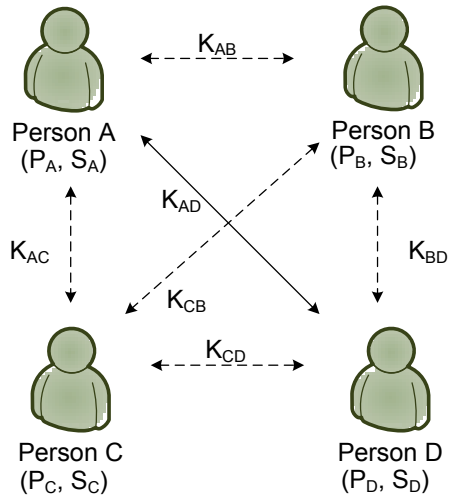

Abbildung 23 – Authentifikation „Jeder mit jedem nach Aufforderung“

Jeder Teilnehmer kann, etwa aufgrund des Inhalts der Chat-Kommunikation und der ausgetauschten Nachrichten, die Identität eines anderen Teilnehmers anzweifeln. Ein solches Szenario kann beispielsweise dann auftreten, wenn die Audio-Video-Kommunikation nicht benötigt wird und sich der Nachrichtenaustausch nur auf den Chat beschränkt. In diesem Fall fordert der zweifelnde Teilnehmer seinen Gesprächspartner auf, einen verschlüsselten Videokanal aufzubauen und den Hashwert des zuvor ausgetauschten Schlüssels vorzulesen. Je nach Ergebnis kann die weitere Diskussion oder Abstimmung fortgeführt bzw. abgebrochen werden, wenn ein Angriff auf die Kommunikation vermutet wird *(Echtheitsüberprüfung des DH-Schlüssels = nach Aufforderung).*

In diesem Szenario bleibt die Generierung von $n * (n-1)/2$ Sitzungsschlüssel notwendig, um jede Verbindung zwischen den Mitgliedern zu verschlüsseln und später auf Anfrage eine eventuelle gegenseitige Authentifikation durchzuführen. Allerdings verkürzt sich der Gesamtprozess durch die entfallende verpflichtende Überprüfung der Hashwerte. Die Anzahl der durchgeführten Wiedererkennungsprozeduren variiert, je nachdem, wie viele Teilnehmer eine solche Prozedur verlangen, zwischen 0 Mal und $n * (n-1)/2$ Mal *(Schlüsselaustausch = Jeder mit jedem).*

Die visuelle Wiedererkennung und der Abgleich der Hashwerte sind ein wichtiger Bestandteil der Zugangsberechtigung, weil nur dadurch die Authentizität des Senders gewährleistet wird. Wenn die Überprüfung nur nach Aufforderung durchgeführt wird, steigt die Wahrscheinlichkeit,

dass eine bilaterale Kommunikation durch einen Man-in-the-Middle-Angriff manipuliert werden kann.

Bezüglich der Verschlüsselung kann, je nach gefordertem Sicherheitsniveau, entweder jede Zweierkommunikation mit dem ausgehandelten DH-Schlüssel abgesichert werden oder ein gemeinsamer Sitzungsschlüssel für die Gesamtkommunikation verwendet werden. Im letzteren Fall ist durch die kleinere Anzahl an Ver- und Entschlüsselungen der Berechnungsaufwand geringer. Der Sender muss eine Nachricht nur einmal verschlüsseln und an alle Teilnehmer (Peer-to-Peer) verschicken. Andererseits werden dadurch die direkten Verbindungen zwischen zwei Teilnehmern anfällig für Abhören oder Manipulationen durch einen weiteren Teilnehmer, der auch den gemeinsamen Schlüssel kennt *(Verschlüsselung = Peer-to-Peer oder Gruppen-Schlüssel)*.

Eine „One-to-one“ Kommunikation wäre hier unter den gleichen Limitierungen wie im Abschnitt vorher möglich. Das Gleiche gilt auch für das Vertrauensmodell. Zwar ist hier eine Überprüfung nicht notwendig, aber ein ehrlicher Teilnehmer würde eine Überprüfung verlangen, wenn er Zweifel an der Identität eines Gesprächspartners hat *(Datenübertragung = Peer-to-Peer, Vertrauensmodell = Ein Teilnehmer)*.

Auch diese Variante lässt sich folgendermaßen darstellen (Abbildung 24).

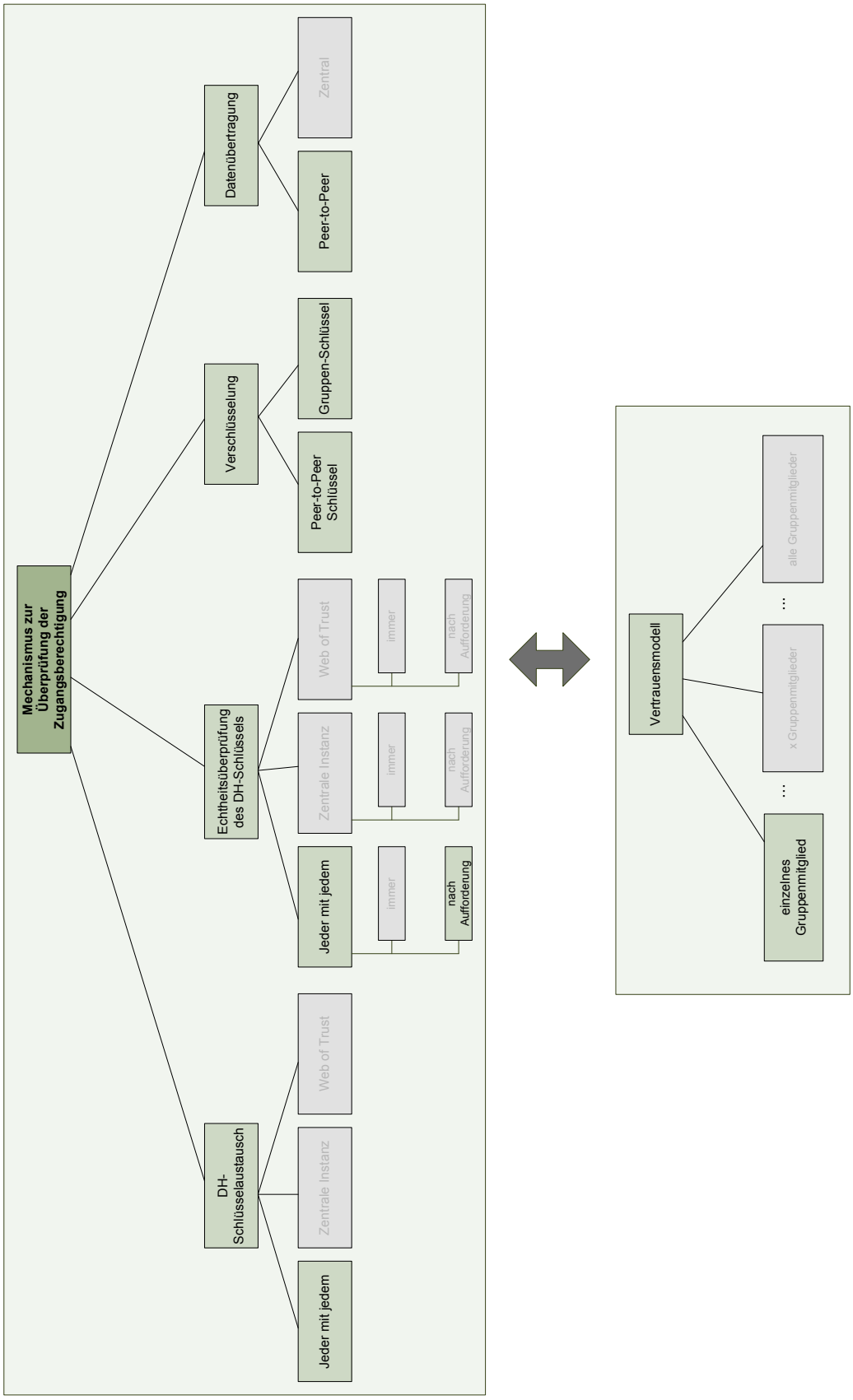

Abbildung 24 –Variante „Jeder mit jedem – nach Aufforderung“

11.2.4 Variante „Zentrale Instanz“

Eine weitere Möglichkeit zur Realisierung der Zugangsberechtigung ist die Einführung einer zentralen Instanz. Dazu erhält ein ausgewähltes Mitglied der Gruppe mehr Privilegien und die Aufgabe, jeden Teilnehmer zu authentifizieren und dessen Identität über den Audio-Video-Kanal zu überprüfen. Dies entspricht den Aufgaben eines Vorsitzenden bei traditioneller Durchführung einer Sitzung *(Echtheitsüberprüfung des DH-Schlüssels = zentrale Instanz*).

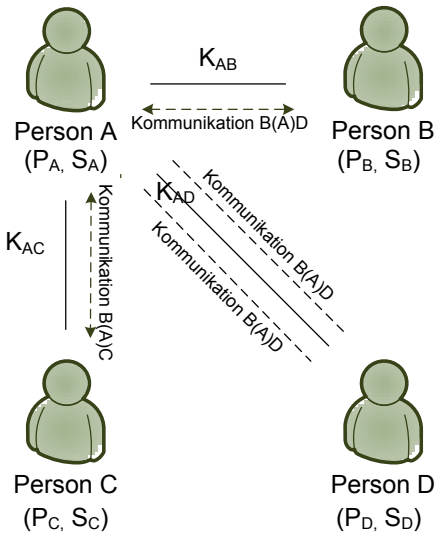

Abbildung 25 – Authentifikation „Zentrale Instanz“

Abbildung 25 zeigt ein Beispiel bei dem Person A als zentrale Instanz fungiert. Sie führt mit jedem Teilnehmer das DH-Key-Agreement durch und überprüft die entsprechenden Hashwerte durch die Wiedererkennungsprozedur (in der Abbildung durch die durchgezogene Linie dargestellt). Dabei werden die Schlüssel K_{AB} K_{AC} K_{AD} erzeugt und genau drei verschlüsselte Verbindungen aufgebaut *(Schlüsselaustausch = zentrale Instanz).*

Jede Mitteilung von A erreicht Person B, C und D über unterschiedlich verschlüsselte Kanäle. Jeder Mitteilung von Person B erreicht Person A direkt und Person D indirekt über den verschlüsselten Kanal, den A mit D etabliert hat *(Datenübertragung = zentral).* Die indirekte Kommunikation wird durch die gestrichelte Linie dargestellt.

Insgesamt werden $n-1$ Sitzungsschlüssel erzeugt und $n-1$ Mal die Authentifizierung über das Videobild durchgeführt. Die Übertragung der Nachrichten erfolgt über die zentrale Instanz als Verteiler. Um den Aufwand für eine n-fache Verschlüsselung zu vermeiden, kann auch in dieser Variante ein symmetrischer Schlüssel (K_{ALL}) erzeugt und verwendet werden *(Verschlüsselung = Peer-to-Peer-Schlüssel oder Gruppen-Schlüssel).* Der Gruppen-Schlüssel wird hier nur von der zentralen Instanz verwendet, um den Rechenaufwand zu minimieren. Nachrichten zwischen den übrigen Teilnehmern werden nicht direkt übermittelt.

Diese Variante reduziert die Anzahl der notwendigen Authentifikationsvorgänge und Schlüssel, hat aber auch einige Nachteile: Aufgrund ihres Wissens über die Verschlüsselungsparameter und durch ihre Rolle als zentraler Verteiler wäre die zentrale Instanz (im Beispiel Person A) in der Lage, jede bilaterale Kommunikation zwischen zwei anderen Teilnehmern abzuhören oder zu manipulieren (vgl. Abbildung 26 , die selektierten Bestandteile sind grün). Für eine zuverläs-

sige Zugangsberechtigung spielt die Ehrlichkeit der zentralen Instanz eine große Rolle, da die gesamte Kommunikation über sie weitergeleitet wird. Ein einzelner ehrlicher Teilnehmer hat in dieser Alternative keine Möglichkeit eine authentifizierte Kommunikation zu gewährleisten, wenn die zentrale Instanz unehrlich ist *(Vertrauensmodell = ein einzelner bestimmter Teilnehmer, der Initiator).*

In dieser Variante ist die Kommunikation vertrauenswürdig, wenn der Initiator ehrlich ist. Ein anderer einzelner Teilnehmer oder eine Gruppe von Teilnehmern kann eine Manipulation nicht aufdecken. Für das oben beschriebene Vertrauensmodell wird zunächst nur die Rolle der Teilnehmer betrachtet. Eine detailliertere Betrachtung des Vertrauensmodells erfolgt später in Abschnitt 11.4.4 gegeben.

Kollaborationen, die auf der Korrektheit der Handlungen von einzelnen Komponenten basieren und bei denen nur in Problemfällen eine weitere Partei Einfluss ausübt, sind bekannt als „optimistische Verfahren“ (Schunter 2000). Im vorliegenden Fall, liegt der Optimismus darin, dass man davon ausgeht, dass sich die zentrale Instanz beim gesamten Prozedere des Eintritts in die Gruppe und Überprüfung der Authentizität ehrlich verhält.

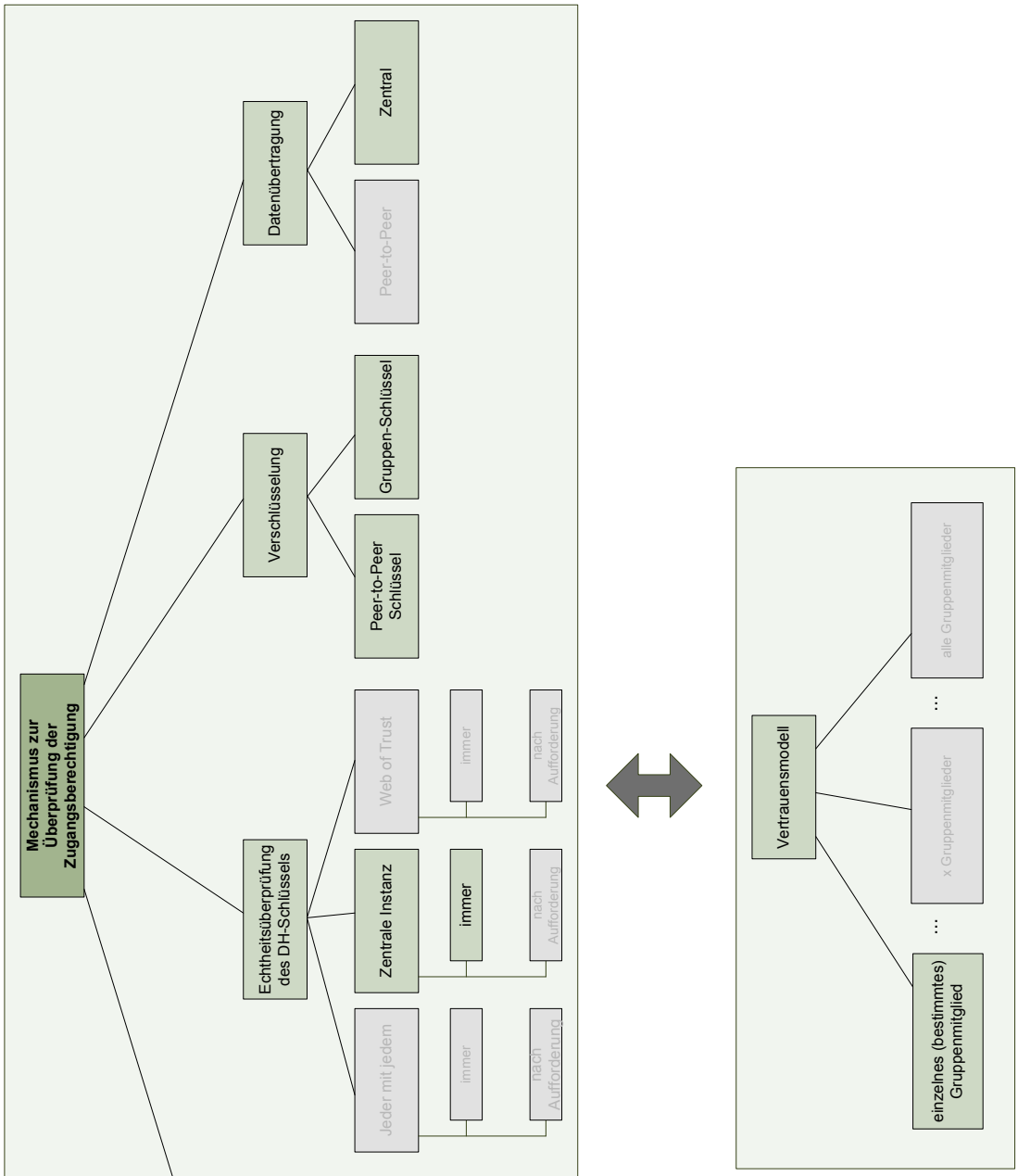

Abbildung 26 – Variante „Zentrale Instanz“

Eine ähnliche Variante entsteht, wenn der Zugangsberechtigungsmechanismus (DH-Key-Agreement und Video-Erkennung, durchgezogene Linien $K_{AB}\ K_{AC}\ K_{AD}$) zwar von der zentralen Instanz durchgeführt wird, aber die Nachrichten über Peer-to-Peer-Verbindungen (gestrichelte Linie) zwischen den Teilnehmern übertragen werden *(Schlüsselaustausch = zentrale Instanz, Echtheitsüberprüfung des DH-Schlüssels = zentrale Instanz, Datenübertragung = Peer-to-Peer).*

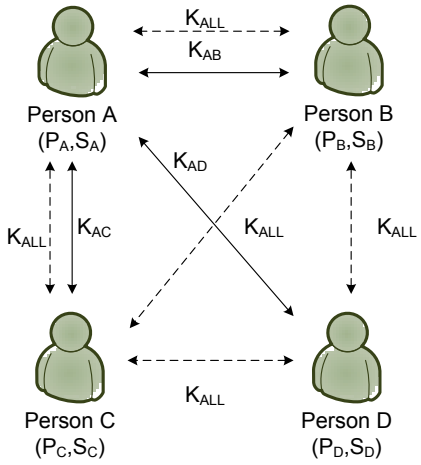

Abbildung 27 – Authentifikation „Zentrale Instanz mit P2P-Übertragung“

Die Anzahl der Schlüsselaustausch-Verfahren und der persönlichen Wiedererkennungen beträgt jeweils $n-1$. Zusätzlich wird nach einer erfolgreichen Authentifizierung aller Mitglieder ein gemeinsamer symmetrischer Schlüssel gewürfelt, der für die Verschlüsselung aller weiteren Nachrichten verwendet wird *(Verschlüsselung = Gruppen-Schlüssel).* In dieser Variante können keine bilateralen Schlüssel verwendet werden, da nicht zwischen allen Paaren ein DH-Schlüsselaustausch stattfindet, aber trotzdem eine Peer-to-Peer-Übertragung gewünscht ist.

Trotz der Peer-to-Peer-Übertragung, ist das Vertrauensmodell dieser Alternative immer noch stark von der Ehrlichkeit der zentralen Instanz abhängig. Allerdings wird der zentralen Instanz durch die Peer-to-Peer-Übertragung der Nachrichten eine Manipulation erschwert. Das bedeutet: man benötigt hier mehr Teilnehmer, um das Vertrauensmodell zu erfüllen. Eine Kooperation von ehrlichen Teilnehmern kann eine eventuelle Manipulation der zentralen Instanz aufdecken *(Vertrauensmodell =x Teilnehmer oder eine zentrale Instanz).*

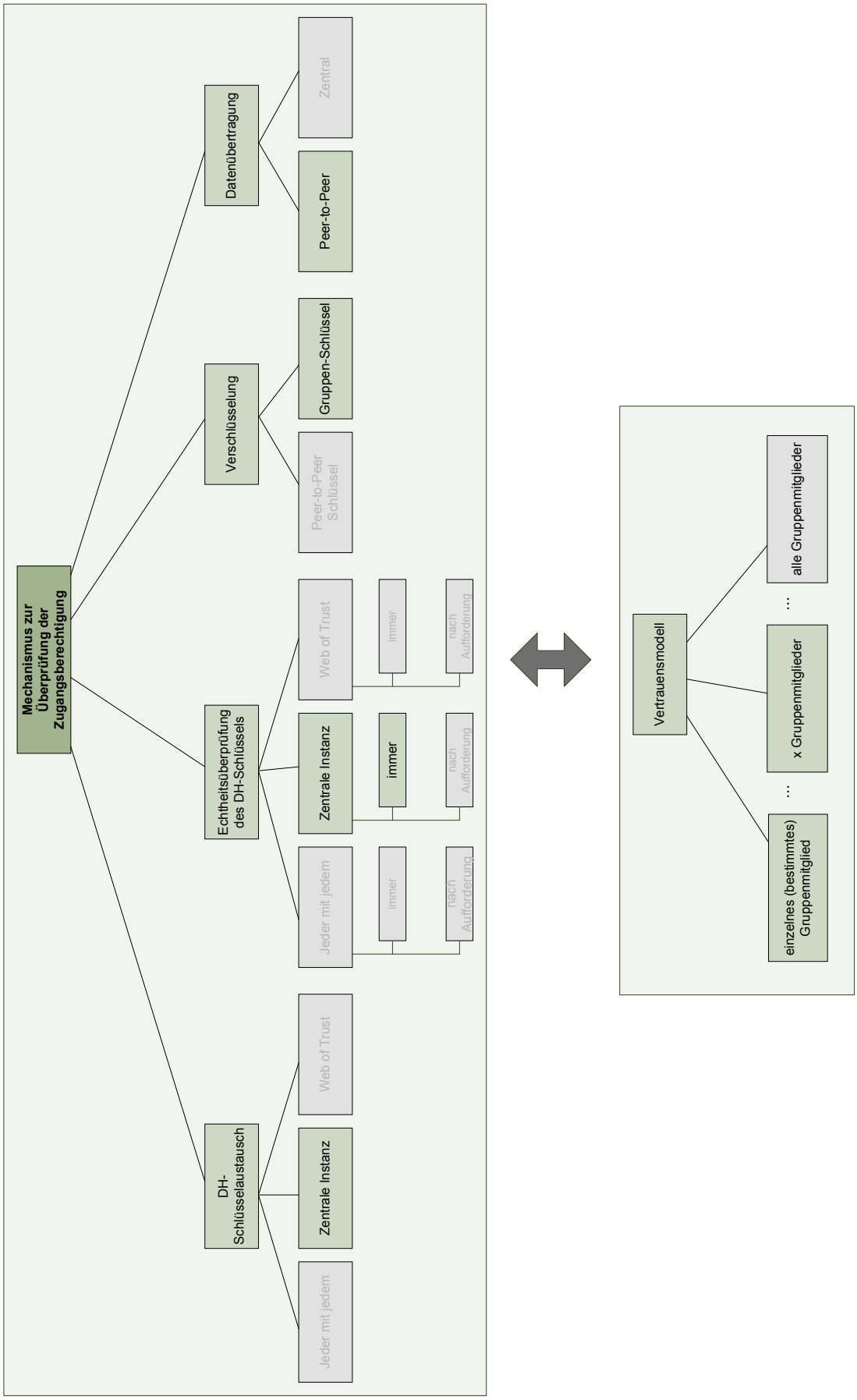

Abbildung 28 –Variante „Zentrale Instanz mit P2P-Übertragung“

11.2.5 Variante „Zentrale Instanz nach Aufforderung“

Bei der Variante der zentralen Instanz ist es möglich die „Macht“ der zentralen Instanz zu vermindern und dadurch das Vertrauensmodell zu optimieren. Dafür findet als Erstes der Schlüsselaustausch und die dazugehörige Überprüfung der Hashwerte wie in der Variante zuvor statt *(Schlüsselaustausch = zentrale Instanz, Echtheitsüberprüfung des DH-Schlüssels = zentrale Instanz, Datenübertragung= Peer-to-Peer, Verschlüsselung = Gruppen-Schlüssel).*

Allerdings wird jetzt eine weitere Überprüfungsmöglichkeit eingeführt: Wenn z.B. Teilnehmer B die Echtheit des Teilnehmers D anzweifelt, kann er anhand des Benutzernamens von D (der ihm aus der ersten Phase der Zugangsberechtigung über das IM-Netz bekannt ist) und seiner IP-Adresse (die ihm aus der Initiierung der Gruppenkommunikation bekannt ist), einen Video-Kanal aufbauen und sich mit oder ohne Durchführung eines bilateralen DH-Schlüsselaustauschs von der Identität seines Gesprächspartners überzeugen. Das bedeutet, es wird zusätzlich zu der Überprüfung der Identitäten durch die zentrale Instanz eine weitere Überprüfung durch zufällige Teilnehmer durchgeführt.

Dadurch wird das Vertrauensmodell verändert (vgl. Abbildung 29). Durch diese Methode kann ein ehrlicher Teilnehmer die authentifizierte, vertrauliche Kommunikation, auch dann gewährleisten, wenn der Initiator im ersten Schritt unehrlich ist (*Vertrauensmodell = einzelner Teilnehmer).*

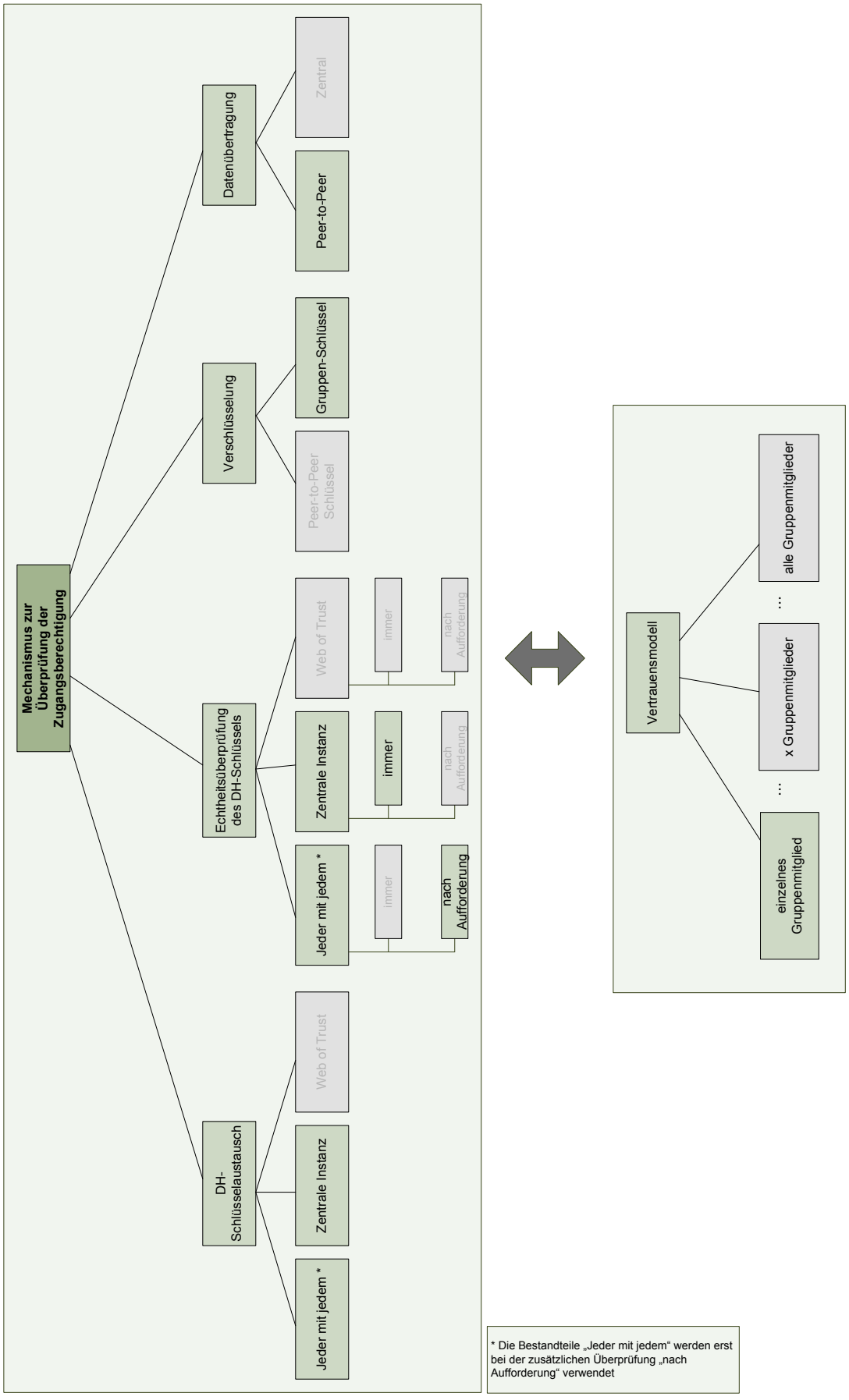

Abbildung 29 –Variante „Zentrale Instanz nach Aufforderung“

11.2.6 Variante „Jeder mit jedem nach Aufforderung und zentraler Pflicht-Überprüfung“

Eine weitere Variante ist eine Kombination der Varianten „Jeder mit jedem nach Aufforderung“ und „Zentrale Instanz“. Das Ziel ist es, die Teilnehmer zu zwingen, mindestens einmal eine Echtheitsüberprüfung der Schlüssel durchzuführen.

Wie in Abbildung 30 zu sehen ist, führt hier jeder Teilnehmer mit allen potenziellen Gesprächspartnern einen DH-Schlüsselaustausch durch (gestrichelte Linie). So entstehen die Schlüssel $(K_{AB}, K_{AC}, K_{AD}, K_{BC}, K_{BD}, K_{CD})$. In einer Gruppenkommunikation werden $n * (n-1)/2$ Schlüssel genau so wie in der Variante „Jeder mit jedem nach Aufforderung“ generiert *(Schlüsselaustausch =Peer-to-Peer)*.

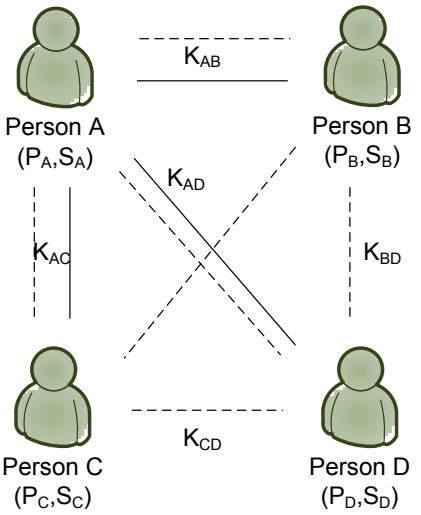

Abbildung 30 – Authentifikation „Jeder mit jedem nach Aufforderung und zentraler Pflicht-Überprüfung“

Die Prozedur der persönlichen Wiedererkennung wird nicht dem Zufall – wie nach Aufforderung – überlassen, sondern jeder der Teilnehmer tätigt diese Überprüfung mit dem Initiator der Konferenz, es werden also $n-1$ Überprüfungen durchgeführt (vgl. Abbildung 30). Dadurch ist zwar jeder Teilnehmer gezwungen, mindestens einmal seinen Hashwert in Verbindung mit seinem Videobild vorzulesen, der zeitliche Aufwand für die persönliche Wiedererkennung ist aber geringer *(Echtheitsüberprüfung des DH-Schlüssels = zentrale Instanz)*. Die Verschlüsselung findet paarweise mit den per DH-Verfahren erzeugten symmetrischen Schlüsseln statt. Damit nicht jeder Teilnehmer Ver- und Entschlüsselungen durchführen muss, kann – wie bei anderen Varianten – die Wahl eines gemeinsamen Sitzungsschlüssels das Prozedere vereinfachen. Dies birgt jedoch Gefahren bei der Integrität der ausgetauschten Nachrichten *(Verschlüsselung = Peer-to-Peer oder Gruppen-Schlüssel)*.

Der Vorteil dieser Variante ist, dass, wenn während der Kommunikation ein Man-in-the-Middle-Angriff vermutet wird, eine bilaterale Videoerkennung zwischen den Teilnehmern durchgeführt werden kann, um dies zu überprüfen. Das Gleiche gilt, wenn nach der ersten Überprüfung der Identitäten durch den Initiator der Konferenz der Video-Kanal nicht weiter benötigt wird. Meint ein Teilnehmer, Unregelmäßigkeiten in den ausgetauschten Chat-

Nachrichten zu bemerken, kann er jederzeit eine spontane Authentifikation anfordern. Denn unabhängig davon, welche Art der Verschlüsselung verwendet wird (bilateral oder mittels eines gemeinsamen Sitzungsschlüssels), existiert durch den Diffie-Hellman-Schlüsselaustausch für jede paarweise Verbindung ein symmetrischer Schlüssel, der über den Video-Kanal nachträglich überprüft werden kann.

Bezüglich des Vertrauensmodells gelten hier die gleichen Aussagen wie in der „Jeder mit jedem nach Aufforderung“-Alternative. Ein ehrlicher Teilnehmer muss sich nicht auf die Ehrlichkeit des Initiators verlassen und kann selbst die anderen Teilnehmer überprüfen *(Vertrauensmodell = ein Teilnehmer)*.

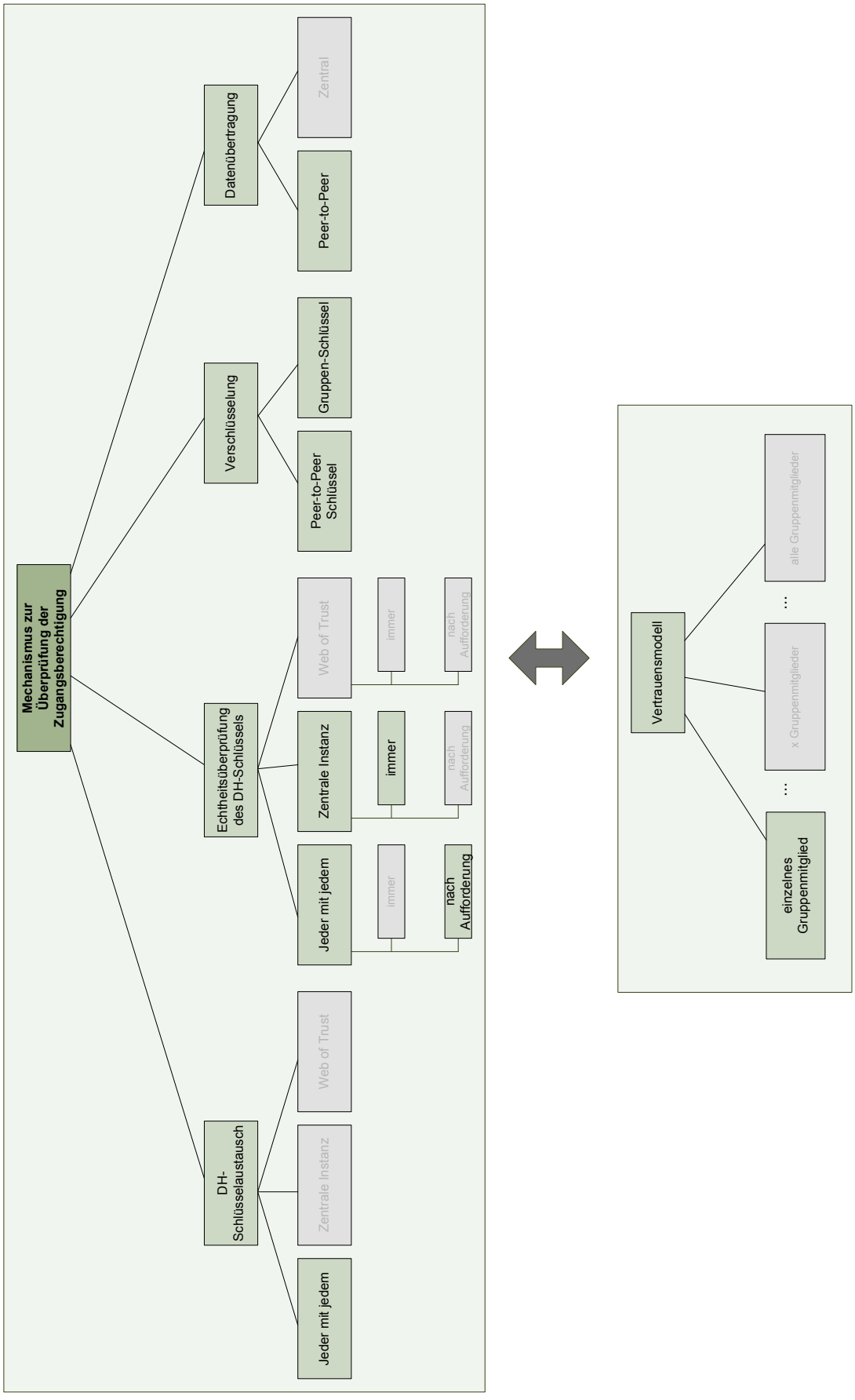

Abbildung 31 – Variante „Jeder mit jedem nach Aufforderung und zentraler Pflicht-Überprüfung“

11.2.7 Variante „Web-of-Trust“

Die letzte Variante des Zugangsmechanismus ähnelt einem „Web-of-Trust“ Konzept, welches von Phil Zimmermann erstmals 1992 in der Spezifikation von PGP erläutert wurde. (Eckert 2009; Schwenk 2005)Wie in den vorher geschilderten Varianten erfolgt auch hier die Überprüfung der Zugangsberechtigung über das Diffie-Hellman-Verfahren, die Erkennung über den Audio-Video-Kanal und die Verschlüsselung paarweise (vgl. Abbildung 32).

Im Unterschied zu früheren Varianten erfolgt hier die Verbindung (d.h. DH-Verfahren, Wiedererkennung und P2P-Verschlüsselung) immer nur mit einem Teilnehmer, der schon eine Zugangsberechtigung hat und Teil des Netzwerkes („Web“) der authentifizierten Teilnehmer ist (durchgezogene Linie). Im Unterschied zu Varianten mit einer zentralen Instanz erfolgt diese Verbindung nicht zwingend mit dieser zentralen Instanz sondern mit einem beliebigen authentifizierten Teilnehmer *(Schlüsselaustausch = Web-of-Trust, Echtheitsüberprüfung des DH-Schlüssels = Web-of-Trust).*

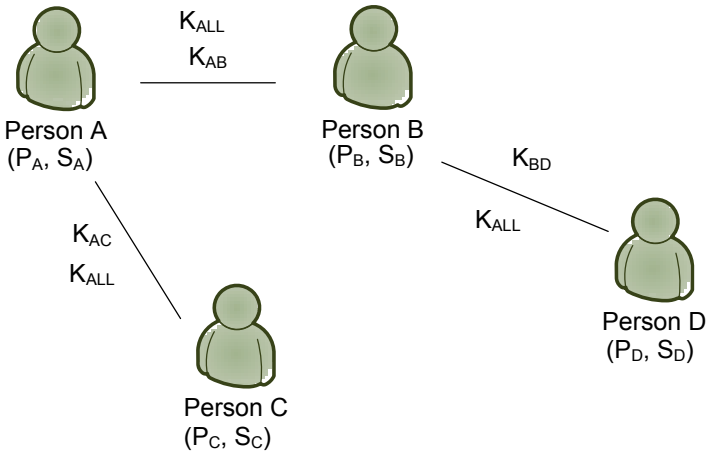

Abbildung 32 – Authentifikation „Web-of-Trust“

Die Anzahl der benötigten DH-Key-Agreements und jeweils visuellen Überprüfungen ist mindestens $n-1$, wenn jeder nur ein Mal überprüft, was hier der Fall ist. Auch hier kann die Verschlüsselung der Peer-to-Peer-Kommunikation entweder durch die Nutzung der bilateralen Schlüssel oder durch die Wahl eines symmetrischen Schlüssels K_{All} erfolgen, welcher durch die ersten beiden Teilnehmer der Gruppe vereinbart wird *(Verschlüsselung = Peer-to-Peer-Schlüssel oder Gruppen-Schlüssel, Datenübertragung = Peer-to-Peer).* Der Vorteil dieser Variante ist, dass sie auch in Szenarien eingesetzt werden kann, in denen sich nicht alle Teilnehmer gegenseitig kennen (vgl. Abbildung 33). Das Vertrauensmodell entspricht dem eines typischen „Web-of-Trust“-Modells und basiert auf der Ehrlichkeit derjenigen, die einen Zugangsberechtigungsmechanismus durchführen. Im umfangreichsten Fall („worst case“), bei dem das „Web-of-Trust“ die Form einer langen Kette von paarweisen Verbindungen annimmt, beruht das Vertrauensmodell auf der Ehrlichkeit aller Teilnehmer *(Vertrauensmodell = mehrere Teilnehmer).*

Eine direkte Kommunikation ist nur mit denjenigen Teilnehmern möglich, mit denen man vorher die bilaterale Authentifikation durchgeführt hat.

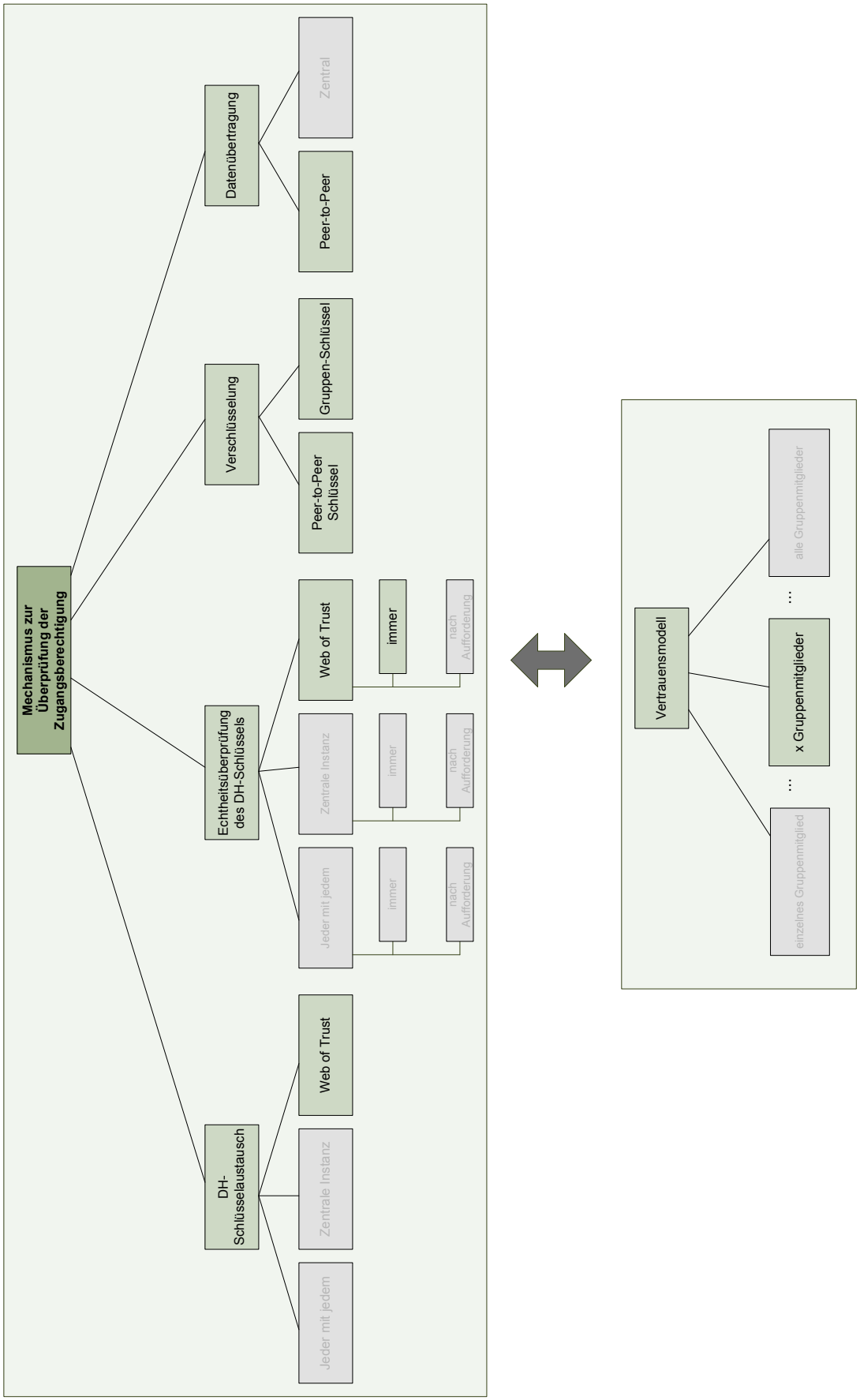

Abbildung 33 – Variante „Web-of-Trust“

11.2.8 Variante „Web-of-Trust nach Aufforderung“

Für diese Variante gelten die gleichen Aussagen wie in der Variante „Web-of-Trust“ (Verschlüsselung = Peer-to-Peer-Schlüssel oder Gruppen-Schlüssel, Datenübertragung = Peer-to-Peer, Vertrauensmodell=mehrere Teilnehmer).

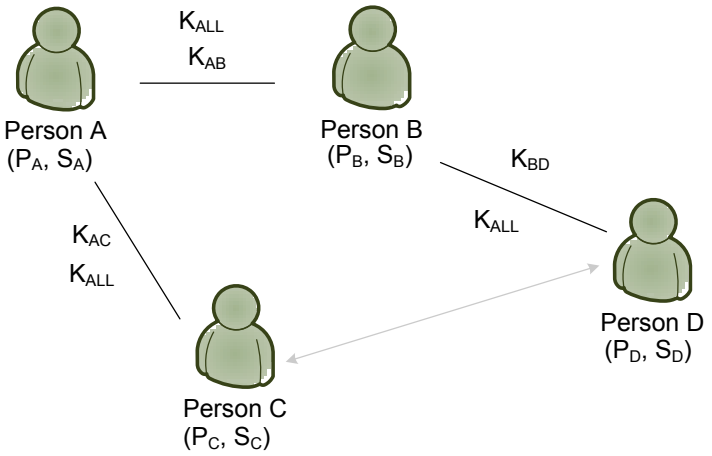

Abbildung 34 – Authentifikation „Web-of-Trust nach Aufforderung“

Ähnlich wie bei der Variante „Zentrale Instanz nach Aufforderung“ kann hier allerdings auch eine weitere Überprüfungsmöglichkeit eingeführt werden (vgl. Abbildung 34). Dies ist dann sinnvoll, wenn ein Teilnehmer zwei unterschiedlichen Personen vertraut und den Authentifikationsprozess mit der gegenseitigen Überprüfung mit einem Teilnehmer durchgeführt. Bezweifelt er (z. B. C) die Echtheit der Identität des anderen „bekannten“ Teilnehmers D, kann er anhand des Benutzernamens von D (der ihm aus der ersten Phase der Zugangsberechtigung über das IM-Netz bekannt ist) und dessen IP-Adresse (die ihm aus der Initiierung der Gruppenkommunikation vorliegt), einen Video-Kanal aufbauen und sich mit oder ohne Durchführung eines bilateralen DH-Schlüsselaustauschs von der Identität seines Gesprächspartners überzeugen (graue durchgezogene Linie).

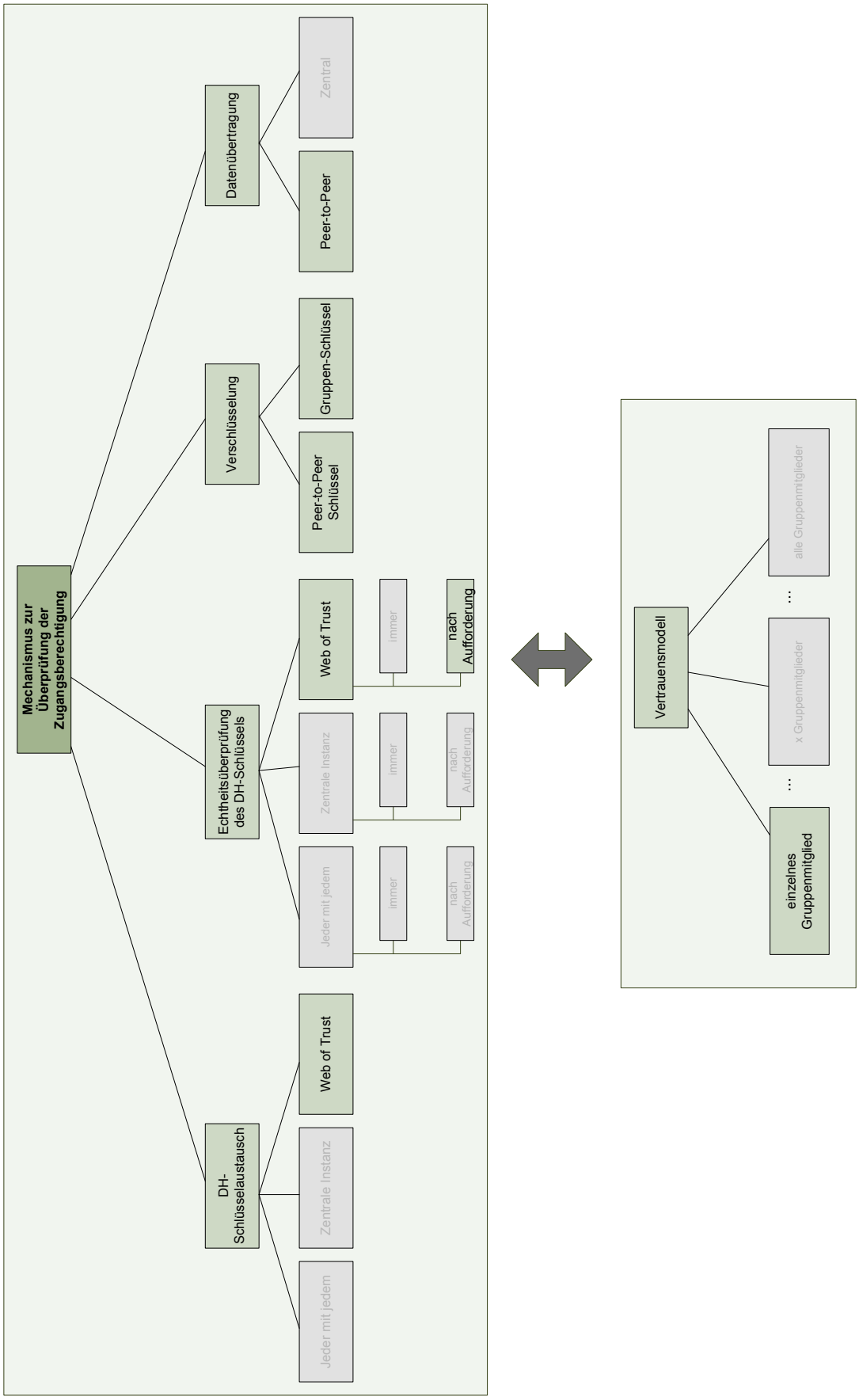

Abbildung 35 – Variante „Web-of-Trust nach Aufforderung“

Durch diese zusätzliche Überprüfung wird das Vertrauensmodell so geändert, dass, wenn ein Teilnehmer alle Mitglieder der Gruppenkommunikation kennt, diese auch überprüfen kann (vgl. Abbildung 35). Also wäre in diesem Fall das Vertrauensmodell = ein Teilnehmer. Allerdings wird das Web-of-Trust-Konzept durch die Limitierung abgeschwächt, dass ein Teilnehmer alle anderen Gesprächspartner kennen muss.

11.2.9 Zusammenfassung

Eine eindeutige Auswahl einer besten Variante ist nicht möglich, da je nach Kontext und Anwendung unterschiedliche Ansätze besser geeignet sind. Vielmehr sollten die genannten Möglichkeiten als Wahlalternativen verstanden werden, die je nach Kollaborationsanwendung, Einsatzumgebung, Akzeptanz für zusätzliche Sicherheitsprozeduren und dem erstrebten Sicherheitsniveau samt Vertrauensmodell realisiert werden können.

In den nächsten Kapiteln wird die zu implementierende Variante genauer betrachtet und einer Sicherheitsanalyse unterzogen. Dabei lassen sich aus der Analyse und Diskussion der dabei angetroffenen Merkmale Rückschlüsse auf andere Varianten mit ähnlichen Merkmalen ziehen.

Die hier gezeigten unterschiedlichen Varianten einer sicheren, d. h. authentifizierten und vertraulichen Kommunikation innerhalb einer Gruppe, stellen eine Erweiterung des Ansatzes von (Zimmermann, Johnston und Callas 2007) dar, der eine Lösung vorsieht, die auf eine bilaterale Audio-Video-Kommunikation und ein bestimmtes Sicherheitsniveau begrenzt ist.

11.3 Ausgewählter Ansatz für die Gewährleistung der Authentizität und Vertraulichkeit in einer Gruppe

Wie in der Einleitung und im Kapitel Stand der Forschung (Kapitel 9) erläutert, sollen in dieser Arbeit die vorgestellten Ansätze verwendet werden, um ein Verfahren für eine elektronische Wahl im Rahmen von Gruppensitzungen (mehr dazu in Kapitel 13, 14) auf Basis eines Instant-Messaging-Systems zu konzipieren. Dafür wird die Variante der zentralen Instanz nach Aufforderung (vgl. Abbildung 29 im Kapitel 11.2.5) aus folgenden Gründen realisiert:

- *Anwendungsangemessen*: Die entwickelte informationstechnische Lösung orientiert sich an den etablierten Verfahren mit Präsenz vor Ort. Hier gibt es ebenfalls einen Vorsitzenden, der im Vergleich zu normalen Teilnehmern mehr Rechte und Pflichten bezüglich der Authentifizierung der Mitglieder und des Ablaufs des Gesamtprozesses hat. Als Vorsitzender muss er mindestens alle Teilnehmer kennen (er kann also als eine zentrale Instanz agieren). Die gewählte Variante bildet die herkömmliche (d. h. nicht-IT-gestützte) Durchführung einer Gruppenkollaboration und das entsprechende, akzeptable Sicherheitsniveau gut ab.
- *Kostengünstig*: Durch die Verwendung einer zentralen Instanz für die Überprüfung der Identitäten entfallen langwierige Überprüfungen zwischen allen Teilnehmern. Der Berechnungsaufwand und die benötigte Zeit für die Durchführung des Authentifikations- und Verschlüsselungsverfahrens bleiben in akzeptablen Grenzen.
- *Ausgewähltes Vertrauensmodell*: Dies geschieht unter Realisierung eines hohen Sicherheitsniveaus, da jeder Teilnehmer die Möglichkeit hat, bei Bedarf seinen Gesprächspartner

zu überprüfen (optimistischer Ansatz). Das heißt: jedes neue Mitglied ist ein „optimales“ ehrliches Mitglied, welches wohlwollend gegenüber dem System agiert.

- *Vereinfachte Implementation*: Durch die Nutzung einer Peer-to-Peer-Übertragung wird die Implementierung vereinfacht, da keine Anpassung des IM-Servers notwendig ist. Die Entwicklung einer zusätzlichen Komponente für den IM-Server wird vermieden.

Nachdem diese Variante nun ausgewählt wurde, wird der entsprechende Ablauf einer Gruppensitzung etwas detaillierter erläutert (Abbildung 36).

Einer der Teilnehmer eröffnet eine Gruppenkonferenz, indem er über den Server des IM-Systems Einladungen an die gewünschten Gesprächspartner verschickt. Die Rolle des Initiators wird einem der Mitglieder am Anfang der Kooperation zufällig zugewiesen, es ist z. B. der letzte Kooperationsberechtigte, der mit seinem IM-Client den Status „online“ hat (alle Teilnehmer müssen „online“ sein).

Jeder Teilnehmer tritt in die Konferenz ein und führt ein DH-Key-Agreement mit dem Initiator der Konferenz durch. Dabei entstehen für jeden Teilnehmer X ein privater Schlüssel S_X, ein öffentlicher Schlüssel P_X und die entsprechenden Parameter. Für jede Zweierbeziehung XY wird aufgrund des DH-Verfahrens ein gemeinsamer Schlüssel K_{XY} berechnet. Alle benötigten Parameter werden über den Chat-Kanal ausgetauscht

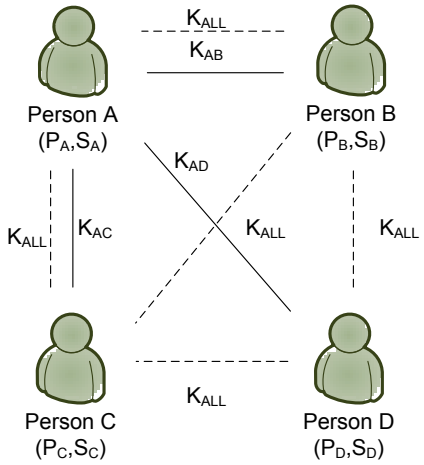

Abbildung 36 – Realisierter Zugangsberechtigungsmechanismus

So werden im Beispiel mit vier Teilnehmern (A, B, C, D) insgesamt folgende Schlüssel generiert:

$$(7)\ K_{AB} = {P_A}^{S_B}(mod\ p_1) = {P_B}^{S_A}(mod\ p_1)$$

$$(8)\ K_{AC} = {P_A}^{S_C}(mod\ p_2) = {P_C}^{S_A}(mod\ p_2)$$

$$(9)\ K_{AD} = {P_A}^{S_D}(mod\ p_3) = {P_D}^{S_A}(mod\ p_3)$$

Dementsprechend führt jeder der Teilnehmer Y mit dem Initiator (im Beispiel Person A) eine Verifikation der Identität über den Audio-Video-Kanal durch, indem er den Hashwert des bilateralen symmetrischen Schlüssels K_{AY} mit A abgleicht. Ist der Initiator von der Identität aller Gesprächspartner überzeugt, wird ein gemeinsamer symmetrischer Schlüssel K_{ALL} gewählt und

den authentifizierten Teilnehmern mitgeteilt. Dieser neue gemeinsame Schlüssel K_{ALL} wird dann für die Verschlüsselung der weiteren Gruppenkommunikation verwendet.

Zweifelt nach einiger Zeit einer der Teilnehmer die Identität eines Gesprächspartners an, kann er folgendermaßen vorgehen: Er kann anhand des Benutzernamens (d. h. der Kennung im IM-Netz) und der IP-Adresse seines Gesprächspartners, über die er durch die Initiierung der Gruppenkommunikation verfügt, einen Video-Kanal aufbauen und sich mittels bilateralem DH-Schlüsselaustausch und visueller Erkennung von dessen Identität überzeugen. Würde beispielsweise der Teilnehmer B die Identität des Gesprächspartners D anzweifeln, könnte er eine nachträgliche Überprüfung der Identität von D verlangen. Dabei hätte er auch die Möglichkeit, einen neuen K_{ALL} zu würfeln und diesen über die bis dahin existierenden bilateralen vertraulichen Kommunikationen zu verteilen. Im Fall des vorgestellten Beispiels mit dem Teilnehmer B sind das die Verbindung mit dem Initiator (Verbindung AB) und die gerade von ihm überprüfte Verbindung BD. Für die Weiterverteilung dieses gemeinsamen Schlüssels wäre der Initiator zuständig. Entdeckt allerdings Teilnehmer B bei der visuellen Erkennung, dass mit der Identität des Teilnehmers D etwas nicht stimmt, wird die Kommunikation unterbrochen.

Wenn man diesen Ansatz noch weiter ausdehnt, könnte der anfragende Teilnehmer (im Beispiel Teilnehmer B) den Zugangsberechtigungsmechanismus auch mit jedem anderen Teilnehmer durchführen und somit die Rolle des Initiators übernehmen. Diese Vorgehensweise stärkt das Vertrauensmodell der Kommunikation, da eine Manipulation durch den ersten Initiator entdeckt würde. Hier ist jedoch zu bedenken, dass dieses Vorgehen den Aufwand für den Schlüsselaustausch und den Wiedererkennungsprozess erheblich erhöhen würde. Das wiederum ist aber eines der Charakteristika einer optimistischen Kollaboration. Im Normalfall geht man davon aus, dass das entsprechende Verfahren richtig funktioniert, da alle Teilnehmer wohlwollend sind. In einem Problemfall müsste dann mehr Aufwand betrieben werden, um die Korrektheit der Handlung gewährleisten zu können. Hier müssen insbesondere der Entwurf eines angemessenen Protokolls für die Gesamtkommunikation und der Umgang mit den ständig wechselnden Schlüsseln berücksichtigt werden.

Mit den bis hier beschriebenen Ansätzen wird das erste Forschungsziel einer spontanen authentifizierten und vertraulichen Gruppenkommunikation und -kollaboration erreicht. Durch das DH-Key-Agreement und die Verifikation über den Video-Kanal wird die Authentizität der Kommunikation sichergestellt. Die Vertraulichkeit der Kommunikation Dritten gegenüber wird – je nach Variante – durch einen gemeinsamen symmetrischen Sitzungsschlüssel K_{ALL} oder durch spontane bilaterale DH-Schlüssel gewährleistet. Anstelle der Verwendung eines gemeinsamen Schlüssels besteht durch die bilateralen Schlüssel die Möglichkeit, auch innerhalb der Gruppe eine vertrauliche Kommunikation sicherzustellen, wenn dies im konkreten Anwendungsfall erforderlich ist.

11.4 Bedrohungen der Authentizität der Teilnehmer für die ausgewählte Variante

In den nächsten zwei Kapiteln wird für die ausgewählte Variante der zentralen Instanz nach Aufforderung eine Bedrohungsanalyse der Authentizität und der Vertraulichkeit vorgenommen.

Dabei werden Hinweise zur Gewährleistung dieser Sicherheitsanforderungen auch für die anderen Varianten abgeleitet. Da der Zugangsmechanismus die Basis sowohl für die Überprüfung der Authentizität der Teilnehmer als auch für die Vertraulichkeit der Kommunikation ist, werden in diesem und dem folgenden Kapitel mögliche Angriffe auf diese zwei Anforderungen vorgestellt.

Für eine genauere Klassifizierung der Angriffe wird die Ehrlichkeit der Teilnehmer betrachtet. Es wird zwischen ehrlichen und unehrlichen Teilnehmern und zwischen externen und internen Angreifern unterschieden. Ein *ehrlicher* Teilnehmer verhält sich gemäß der Sicherheitspolicy des Systems (korrekte Nutzung, sorgsamer Umgang mit den Kommunikationsdaten). Ein *unehrlicher* Teilnehmer ist einer, der entweder seine Sorgfaltspflicht verletzt oder bewusst eine Gruppenkommunikation manipulieren möchte. Dabei wird der Initiator (derjenige, der die Konferenz startet und die Überprüfung der Identitäten übernimmt) separat betrachtet, da er einen großen Einfluss auf die Authentifikationsphase hat. Dies ist wichtig für die Ausführung des Vertrauensmodells. Als *extern* wird ein Angreifer bezeichnet, der kein Teilnehmer der Gruppenkommunikation ist und mit oder ohne Wissen der internen Architektur eine Manipulation vornehmen möchte. Demgegenüber steht der *interne* Angreifer. Dieser ist ein Teilnehmer (Mitglied der Gruppe), der mehr Wissen über die Kommunikation hat und dieses zu seinen Gunsten ausnutzt. Somit werden je Anforderung die möglichen Kombinationen dieser Kriterien und Kooperationsangriffe betrachtet. Für die Authentizität werden folgende Fälle samt internen und externen Angriffen analysiert:

- Unehrlicher Initiator
- Unehrlicher Teilnehmer
- Ehrlicher Teilnehmer

11.4.1 Unehrlicher Initiator

Einer der Schwachpunkte der gewählten Variante ist der Initiator. Er übernimmt diese Aufgabe zwar spontan am Anfang der Kommunikation, trotzdem ist seine Rolle wichtig für die Sicherheit des Systems. Wenn der Initiator (Person A) der Gruppenkommunikation (siehe Abbildung 36) nicht vertrauenswürdig (unehrlich) ist, dann sind folgende Manipulationen möglich:

- Der Initiator verweigert einer berechtigten Person den Eintritt in die Gruppe und somit die Teilnahme an der Gruppenkollaboration. Wenn er keine andere Person (z. B. einen Angreifer) in die Kommunikation einschleust, wird den übrigen Gruppenmitgliedern das Fehlen des Teilnehmers auffallen, da zumindest die Anzahl der Gruppenmitglieder allen Teilnehmern bekannt ist. Wenn der Initiator statt des berechtigten Teilnehmers einer anderen Person Eintritt gewährt und somit die erwartete Anzahl von Mitgliedern an der Kommunikation teilnimmt, dann gilt der nächste Fall.
- Der Initiator verhält sich bewusst „falsch“ und erlaubt einer Person den Zutritt in die Gruppe, obwohl die DH-Authentifikation und die Verifikation über Bild fehlgeschlagen ist (bewusste Zulassung eines Angreifers). Ähnlich gestaltet sich der Angriff, wenn die Instanz ihre Sorgfaltspflicht verletzt und einer Person den Zutritt in die Gruppe (richtige Verifizierung über Stimme oder Video) erlaubt, die dazu keine Berechtigung hat.

Wird für die Gruppenkommunikation die Audio-Video-Verbindung weiter benötigt, ist eine solche Manipulation schnell aufdeckbar, wenn die anderen Teilnehmer durch Erkennen der Stimme oder des Videobildes den unberechtigten oder „falschen“ Teilnehmer identifizieren.

Sollte der Audio-Video-Kanal für die weitere Kollaboration nicht benötigt werden, ist eine solche Manipulation schwieriger aufzudecken. Eine Aufdeckung ist dann möglich, wenn andere Teilnehmer einen semantischen Widerspruch in den Aussagen des Angreifers erkennen. Das ist der Fall, wenn der Angreifer Inhalte über den Chat-Kanal verbreitet, die nicht zum „echten“ Teilnehmer (dessen Identität übernommen wurde) passen.

Diesem wirkt der Aufbau einer bilateralen Verbindung zur Überprüfung nach Aufforderung (vgl. Kapitel 11.2.5) entgegen. Ein Teilnehmer führt dann anhand des Benutzernamens und der IP-Adresse des Gesprächspartners ein DH-Verfahren durch und lässt seinen Gesprächspartner über einen Video-Kanal seine Identität nachweisen.

11.4.2 Unehrlicher Teilnehmer

Eine weitere Angriffsmöglichkeit basiert auf der Unehrlichkeit eines beliebigen Teilnehmers. Ein Teilnehmer kooperiert mit einem Angreifer und überlässt ihm seine Position in der Konferenz, etwa indem er ihm seine Client-Software überlässt.

Hier sind prinzipiell drei Stufen möglich: Im einfachsten Fall überlässt der unehrliche Teilnehmer dem Angreifer *vor Beginn der Sitzung* seine Zugangsdaten des IM-Netzes (Benutzername und Kennwort aus der ersten Phase der Zugangsberechtigung, siehe Kapitel 11.1.1). Diese Manipulation würde direkt während der Video-Verifikation der Hashwerte auffallen.

Die nächste Stufe besteht darin, dass der berechtigte Teilnehmer dem Angreifer seine Position (z. B. seinen IM-Client) erst *nach der Überprüfung der Identität* überlässt, dann ist es schwierig, diese Manipulation zu erkennen. Immer vorausgesetzt, dass für die weitere Kommunikation keine Video-Verbindung benötigt wird.

Noch weiter geht ein unehrlicher Teilnehmer, der zwar selbst in der elektronischen Sitzung *präsent* ist, z. B. indem er sich im Video zeigt, *aber den Angreifer* die Inhalte seiner Äußerungen und Entscheidungen *bestimmen lässt*. Dies kann zum Beispiel dadurch geschehen, dass er dem Angreifer – für die anderen unsichtbar – Maus und Tastatur überlässt, oder dadurch, dass der Angreifer dem Teilnehmer indirekt mitteilt, welche Meinungen er äußern und wie er abstimmen soll. Da der Teilnehmer selbst immer noch an der Sitzung teilnimmt, ist es kaum möglich, diese Manipulation zu erkennen. Ein solcher Angriff ist immer durchführbar, es gibt dagegen keine sicherheitstechnischen Mechanismen. Der Angreifer hat hier jederzeit die Hilfe und Unterstützung des Teilnehmers. Je weniger der Teilnehmer mit dem Angreifer kooperiert, desto höher ist die Wahrscheinlichkeit, den Angriff zu erkennen.

11.4.3 Externer Angreifer und ehrliche Teilnehmer

In diesem Szenario sind alle Teilnehmer ehrlich. Ohne das Wissen und die Mithilfe eines Teilnehmers ist ein externer Angriff auf die Authentizität von Teilnehmern zwar sehr aufwendig, aber prinzipiell möglich.

Der externe Angreifer könnte den Datenstrom auf Ebene eines Anwendungsprotokolls oder auf TCP/IP-Ebene manipulieren, indem er, z. B. mit Hilfe einer Malware, den Client eines Mitg-

lieds infiziert. So könnte er z. B. über eine zwischengeschaltete Komponente (Proxy) eingehende und ausgehende Nachrichten für das Gruppenmitglied schon während der Zugangsberechtigungsphase beeinflussen und den gemeinsamen Schlüssel berechnen. Spätestens bei der Verifikation über das Video-Bild wäre seine Manipulation aber erkennbar.

Mit entsprechenden Werkzeugen kann ein Angreifer einen Avatar (d. h. eine elektronisch generierte Figur) herstellen, der das Opfer nachahmt. Zur Erzeugung eines solchen gefälschten Videobildes gibt es unterschiedliche Ansätze, z. B. eine Gesichtsanimation mit Sprachausgabe über Audio-Dateien oder Text-to-Speech-Verfahren. Eine entsprechende Zusammenfassung gibt (Tümmler 2007). Zum Zeitpunkt der Wiedererkennung im Videobild muss bei ordnungsgemäßem Ablauf ein getäuschter Teilnehmer X eigentlich sein Gegenüber Y im bewegten Videobild sehen und im Audiokanal hören, während Y die richtigen Hashwerte vorliest. Ein Angreifer, der X täuschen will, müsste also für die abgefangene Verbindung und die zugehörigen Parameter des Key-Agreements künstlich ein passendes vertontes Videobild erzeugen, in dem (es so scheint als ob) Teilnehmer Y den entsprechenden Hashwert vorliest. Bleibt die Videoverbindung nach dieser Authentifikation bestehen, müsste auch im weiteren Verlauf der Sitzung diese Täuschung mittels künstlichen Videobildes von Y aufrechterhalten werden.

Um diesen Angriff erfolgreich abzuschließen, müsste der Angreifer über digitale Bilddaten seines Opfers verfügen. Er müsste während des Vorlesens der Hashdaten entsprechende Video- und Audiodaten übertragen und seinen DH-Austauschpartner von seiner Identität überzeugen. Ähnliches gilt für die anderen Mitglieder der Gruppe im weiteren Verlauf der Sitzung. Die Erstellung und Nutzung solcher Avatare ist zwar prinzipiell möglich, bedarf aber eines großen Ressourcenaufwands, welcher den Sinn des Angriffes relativiert. Weiterhin ist eine Erkennung eines solchen Angriffes aufgrund der sozialen Interaktion (etwa während des Smalltalk zu Beginn der Sitzung) oder durch die schlechte Qualität des künstlichen Bildes sehr wahrscheinlich. Dafür sprechen unter anderem die nicht ausgereifte Bildbearbeitungstechnologie und die für einen erfolgreichen Angriff notwendige Erzeugung und Übertragung in Echtzeit (Schreer et al. 2008).

11.4.4 Vertrauensmodell der Authentizität

Das oben beschriebene Bedrohungspotenzial, welches durch das Verhalten der Teilnehmer motiviert ist, kann durch das Vertrauensmodell angegeben werden, wie im Abschnitt 11.2.1 erläutert wurde. Dieses Vertrauensmodell wird formal für jedes einzelne Sicherheitsziel durch den *k-resilience*-Wert dargestellt. Der *k-resilience*-Wert gibt für ein Sicherheitsziel an, wie viele ehrliche Komponenten eines Systems erforderlich sind, um Angriffe auf eine realisierte Sicherheitsanforderung zuerkennen oder sogar zu verhindern (Volkamer 2009; Volkamer und Grimm 2009).

Dabei kann eine Komponente eine technische Komponente, wie z. B. ein Server oder eine Person mit bestimmten Aufgaben sein. Als Beispiel kann man hier einen Authentifikationsserver im IM-Netz betrachten. Angenommen, dieser Server wird von zwei gleichberechtigten Administratoren gewartet. Wenn man ausdrücken möchte, dass es ausreicht, wenn einer dieser zwei Administratoren ehrlich ist, um den Server in einem sicheren Zustand zu halten, spricht man von einem k-resilience-Wert von $1\ out\ of\ \{Admin_1, Admin_2\}$.

Der k-resilience Wert wird durch folgende Formel gegeben:

$$(10) \quad \left(k_{1,1} + \cdots + k_{m,1}\ out\ of\ N_{1,1} \cup \ldots \cup N_{m,1}\right), \ldots, \left(k_{1,n} + \cdots + k_{m',n}\ out\ of\ N_{1,n} \cup \ldots \cup N_{m',n}\right)$$

Dabei ist $k_{i,j}$ die erforderliche Mindestanzahl der ehrlichen Komponenten aus der gesamten Menge $N_{i,j}$ von Komponenten dieses Typs.

Die Operatoren + sowie ∪ können dabei als „und“ interpretiert werden, die Verknüpfung mit Komma als „oder“. Ein k-resilience Wert von $(1 + 1\ out\ of\ A \cup B)$ bedeutet, dass eine Komponente vom Typ A *und* eine Komponente vom Typ B benötigt werden. $(1\ out\ of\ C), (1\ out\ of\ D)$ bedeutet, dass eine Komponente vom Typ C *oder* eine Komponente vom Typ D benötigt werden. Je mehr „und“ Operatoren vorkommen, desto *unsicherer* ist ein System (es werden alle Komponenten benötigt). Je mehr „oder“ Operatoren vorkommen, desto *sicherer* ist ein System (es reicht, wenn eine der Alternativen zutrifft).

Dieses generische Modell lässt sich auf den hier diskutierten speziellen Fall anwenden, wenn Komponenten als Teilnehmer und Komponententypen als Rollen interpretiert werden. Dann ist $k_{i,j}$ die erforderliche Mindestanzahl an ehrlichen Teilnehmern aus der Menge $N_{i,j}$ von Teilnehmern mit dieser Rolle. Haben alle Komponenten die gleiche Rolle (Verantwortlichkeiten), dann entspricht dies mit einer Aussage $k\ out\ of\ N$ dem einfachsten Fall des k-resilience-Wertes so wie er oben in Abschnitt 11.2.1 als Vertrauensmodell eingeführt wurde.

Bezüglich der Bewahrung der Authentizität werden zwei Rollen unterschieden, dem *Initiator* und den *Teilnehmern*. Wie oben gezeigt wurde, ist die Authentizität gegeben, wenn der Initiator ehrlich ist und nur berechtigte und richtig verifizierte Personen als Teilnehmer zulässt *oder* wenn mindestens einer der n Teilnehmer ehrlich ist und beim Verdacht eines Angriffs eine neue gegenseitige Authentifizierung nach Aufforderung verlangt. Entsprechend ist der k-resilience-Wert gemäß der Formel (10) also

$$(11) \quad 1\ out\ of\ \{Initiator\}, 1\ out\ of\ \{Teilnehmer_1, \ldots, Teilnehmer_n\}$$

11.5 Bedrohungen der Vertraulichkeit der Kommunikation für die ausgewählte Variante

Ein Angriff auf die Vertraulichkeit einer Kommunikation kann durchgeführt werden, wenn der Angreifer die verwendeten Schlüssel für die Verschlüsselung erraten kann oder der Verschlüsselungsalgorithmus zu schwach ist und/oder als gebrochen gilt. Da die in den vorliegenden Szenarien verwendeten Algorithmen (AES, ElGamal) zum Zeitpunkt der Fertigstellung dieser Arbeit als sicher gelten, werden in den folgenden Fällen Angriffe beschrieben, die nicht auf der Berechnung oder dem Erraten der symmetrischen oder asymmetrischen Schlüssel basieren.

11.5.1 Unehrlicher Teilnehmer

Ein erster Angriff auf die Vertraulichkeit der Chat- oder Video-Kommunikation ist möglich, wenn ein Teilnehmer mit einem Angreifer kooperiert und ihn bewusst durch die Weitergabe des gemeinsamen Schlüssels K_{ALL} und/oder Nennung weiterer Parameter oder Informationen unterstützt (z. B. durch Benennung des Verschlüsselungsalgorithmus). Damit kann ein externer Ang-

reifer mit geeigneten Werkzeugen Inhalte der Kommunikation abhören und gegebenenfalls speichern und analysieren. Er wäre somit in der Lage, geheime Inhalte zu einem späteren Zeitpunkt zu entschlüsseln und zu veröffentlichen. Ein solcher Angriff kann leider nur indirekt erkannt werden, etwa wenn später geheime Informationen an falscher Stelle auftauchen. Es handelt sich um einen typischen Angriff auf die Vertraulichkeit, wenn vertrauliche Informationen explizit weitergeleitet werden.

11.5.2 Externer Angreifer und ehrliche Teilnehmer

Im Gegensatz zur vorherigen Angriffsmöglichkeit kann der Angreifer unter Umständen auch ohne das Wissen oder die Mitarbeit eines Teilnehmers an den gemeinsamen symmetrischen Schlüssel K_{ALL} gelangen. In diesem Fall gelten die Teilnehmer als ehrlich und der Angreifer als extern. Dabei müsste der Angreifer als Erstes einen DH-Schlüsselaustausch angreifen, wie dies im Kapitel 11.4.3 skizziert wurde. Anschließend müsste er die Absicherung durch die Videoverifikation überwinden.

Ist der Angreifer dann im Besitz des gemeinsamen Schlüssels K_{ALL}, könnte er die Kommunikation zwischen Teilnehmern aktiv verfolgen und eventuell manipulieren (das wäre allerdings ein Angriff auf die Integrität, siehe dazu Kapitel 11.6). Der Aufwand und die Erfolgsaussichten dieses Angriffes wurden im Kapitel 11.4.3 erläutert und unter Berücksichtigung des angestrebten Sicherheitsniveaus kann man dieses Risiko als akzeptabel bewerten.

Ähnliche Angriffe sind auch bei den Varianten möglich, in denen bilaterale Schlüssel für die Peer-to-Peer-Kommunikation verwendet werden. In diesen Fällen wäre aufgrund der verschiedenen Schlüssel nur ein Teil der Kommunikation angreifbar.

11.5.3 Vertrauensmodell der Vertraulichkeit

Bezüglich der Vertraulichkeit der Kommunikation haben alle n Teilnehmer, einschließlich des Initiators, die gleiche Rolle. Um die Vertraulichkeit der Kommunikation zu gewährleisten, müssen alle Teilnehmer ehrlich sein. Denn schon durch einen unehrlichen Teilnehmer, der den Inhalt der Kommunikation verrät, kann die Kommunikation angegriffen werden. Also ist der k-resilience Wert:

$$n\ out\ of\ \{Teilnehmer_1 \dots Teilnehmer_n\} \tag{12}$$

Im vorliegenden Beispiel mit vier Teilnehmern ist der k-resilience Wert $4\ out\ of\ \{A, B, C, D\}$.

11.6 Integrität der Nachrichten

In diesem Abschnittgeht es um den Schutz der Integrität der versendeten verschlüsselten Nachrichten. Als allgemeine Maßnahmen für die Wahrung der Integrität von Daten gelten Hashfunktionen, wie Modification Detection Codes (z. B. MD-5, SHA-1), Message Authentification Codes (z. B. HMAC) oder digitale Signaturen (Schneier 1996; Eckert 2009). In den weiteren Kapiteln werden die unterschiedlichen Mechanismen zunächst kurz vorgestellt. Anschließend werden Angriffe auf die Integrität identifiziert und die Nutzung entsprechender Sicherheitsmechanismen (innerhalb einer Gruppenkommunikation über ein IM-System) bewertet.

An dieser Stelle wird zwischen multimedialen Nachrichten (z. B. Videodaten) und Textnachrichten (Chat-Nachrichten) unterschieden. Der Grund dafür liegt einerseits in den unterschiedlichen Integritätsschutz-Methoden für beide Nachrichtenarten und andererseits in der Schwierigkeit verschlüsselte Videodaten mit aktuellen Methoden in Echtzeit zu manipulieren. Auf dieser Voraussetzung beruhen auch die Überlegungen der vorliegenden Arbeit.

Ein relevanter Angriff auf die Integrität der Audio-Video-Verbindung im Rahmen dieser Szenarien ist folgender: Der Angreifer manipuliert (wie im Kapitel 11.4.3 erläutert wurde), die gesamten Bilddaten indem er seinen Avatar in die Kommunikation einschleust. Beim aktuellen Stand der Technik gehen einschlägige Experten aus dem Forschungsbereich Videoprocessing und Multimedia-Daten davon aus (persönliche Kommunikation mit Prof. Dr. Paulus, Universität Koblenz, 30.7.2009), dass ein Angriff mit Hilfe von manipulierten Videodaten derzeit unmöglich ist, da ein solcher Avatar sehr schnell als künstlich erzeugt und manipuliert erkannt würde (Kanellos 2006; Wang 2009). Darüber hinaus gibt es auch Versuche, nur partielle Teile des Videomaterials zu manipulieren. Im Szenario der spontanen Videoüberprüfung der Identität könnte so ein Angreifer dem echten Teilnehmer z. B. andere Hashwerte in den Mund legen. So beschreiben (Ezzat, Geiger und Poggio 2002) ein Projekt, in dem im Bild dargestellte Teile des Gesichtes, z. B. der Ausschnitt, der den Mund zeigt, so manipuliert werden, dass andere Bewegungen zu sehen sind und andere Sprachinhalte geäußert werden. Diese Manipulation ist aber noch nicht für ein Videostream in Echtzeit möglich, da das Prozedere des Ausschneidens und Änderns dieser Gesichtsmerkmale sehr langwierig ist. Somit wird für das vorliegende Konzept der Aufbau eines verschlüsselten Audio-Video-Kanals als ausreichend sicher betrachtet.

Da Multimediadaten häufig als Datenstrom übertragen werden, sind Integritätsmechanismen, welche die gesamte Datenmenge in einer Nachricht blockweise verarbeiten, ungeeignet. Stattdessen werden Verfahren benötigt, die eine kontinuierliche Verarbeitung (und Überprüfung der Integrität) ermöglichen. Dabei sollte als Eingabe für das Verfahren nicht der ganze Bitstrom verwendet werden, da sich die Syntax (die konkret übertragenen Daten) verändern kann, ohne dass die Semantik (der relevante Inhalt) beeinflusst wird. Beispielsweise würde nach Kompression von Mediadaten eine Integritätsverletzung angezeigt werden, obwohl die Semantik der Daten unverändert ist (Dittmann, Steinmetz und Steinmetz 1999). In ähnlicher Weise ist es problematisch, dass bei einer Signierung der gesamten Daten die Gesamtdaten als manipuliert gelten, sobald nur ein kleiner Fehler in einem einzigen Teilbild auftaucht. Aus diesem Grund werden in der Forschung Verfahren wie z. B. inhaltsbasierte Signaturen verwendet (Dittmann und Wohlmacher 2000; Ramaswamy und Rao 2006). Andere Methoden für die Gewährleistung der Integrität sind fragile Wasserzeichen (Dittmann 2000). Für Realtime-Applikationen gibt es entsprechende Ansätze, welche die besonderen Anforderungen solcher Anwendungen berücksichtigen (Kang, Choi und Choi 2004).

Die weiteren Ausführungen und Lösungsmaßnahmen konzentrieren sich auf die Chat-Kommunikation (im Gegensatz zum Audio-Video-Kanal).

11.7 Mögliche Ansätze zur Überprüfung der Integrität in einer Gruppe

11.7.1 Message Detection Codes

Eine erste Möglichkeit, die Integrität einer Nachricht zu gewährleisten, ist die Nutzung von dedizierten Hashfunktionen in so genannten Modification Detection Codes (Schmeh 2001) wie MD-5 oder SHA-1. Dabei berechnet der Sender einer Nachricht M einen Hashwert und übermittelt diesen gemeinsam mit der Nachricht seinem Gesprächspartner. Die verwendeten Hashfunktionen sind so entworfen, dass der berechnete Hashwert, im Vergleich zur Nachricht, nur wenig Speicherplatz (=Übertragungskapazität) benötigt, aber trotzdem eine Veränderung der Nachricht zu einer Veränderung des Hashcodes führt.

Nach Empfang der Nachricht berechnet der Sender seinerseits mit dem gleichen Algorithmus einen Hashwert $H(M')$ der empfangenen Nachricht. Sind $H(M)$ und $H(M')$ gleich, kann der Empfänger davon ausgehen, dass die empfangene Nachricht korrekt übertragen wurde.

Dieser Mechanismus schützt gegen eine unbeabsichtigte Veränderung der Nachricht, etwa durch Übertragungsfehler. Allerdings steht damit der Ursprung der Nachricht nicht fest. So könnte ein Angreifer sehr einfach eine Nachricht manipulieren und – durch Anwendung des bekannten Hash-Algorithmus – mit einem neuen gültigen Hashwert versehen. Der Empfänger kann dann bei einer Überprüfung des Hashwertes diese Manipulation nicht erkennen, da der Hashwert zur übertragenen Nachricht passt.

11.7.2 Message Authentification Codes

Eine zweite Möglichkeit zur Gewährleistung der Nachrichtenintegrität ist die Verschlüsselung mit sogenannten Message Authentification Codes (MAC) wie z. B. HMAC. Dabei handelt es sich um Hashfunktionen, bei denen der Hashwert einer Nachricht mit einem symmetrischen Schlüssel verknüpft wird (Eckert 2009) . Geht man von einer bilateralen Kommunikation aus, können die Teilnehmer einer Kommunikation eine MAC-Hashfunktion dazu verwenden, die Unverfälschtheit der Daten und die Authentizität der Urheber zu überprüfen. Der Sender (Person A) sendet die Nachricht M mit dem Hashwert $MAC(M, K)$, welcher mit Hilfe des geheimen Schlüssels K erzeugt wurde, den nur Sender und Empfänger kennen. Dieser Schlüssel kann z. B. aus dem DH-Key-Agreement K_{AB} resultieren. Der Empfänger (Person B) berechnet – wie beim Fall der MDC – den Hashwert der empfangenen Nachricht unter Berücksichtigung des geheimen Schlüssels. Eine Übereinstimmung beweist die Integrität der Daten.

Die Anwendung von MAC-Funktionen innerhalb eines IM-Netzes würde die Integrität der Nachrichten in folgenden Fällen gewährleisten: (1) Alle Nachrichten, die innerhalb der Gruppe ausgetauscht werden, werden mit einer MAC-Funktion und dem gemeinsamen Sitzungsschlüssel K_{ALL} kombiniert. (2) Alle Nachrichten werden Peer-to-Peer zwischen den Teilnehmern übertragen und mit einer MAC-Funktion und einem bilateralen Schlüssel K_{XY} kombiniert.

Der erste Ansatz ist geeignet, wenn Angriffe auf die Integrität von *externen* Angreifern abgewehrt werden sollen. Innerhalb der Gruppe wird durch diesen Mechanismus nur bestätigt, dass der Erzeuger einer Nachricht *ein* Teilnehmer dieser Gruppe ist, da alle Teilnehmer den Schlüs-

sel K_{ALL} kennen. Um darüber hinaus einen *internen* Angriff erkennen zu können, müsste man für jede Zweierkommunikation einen symmetrischen Schlüssel aushandeln. Ob dies möglich ist, hängt von der jeweils gewählten Variante ab (siehe Kapitel 11.2, Merkmal „Verschlüsselung“).

Für beide Fälle gilt, dass nicht eindeutig erkannt werden kann, ob eine bestimmte Quelle (Absender) einer Nachricht auch der Erzeuger einer Aussage/Nachricht ist. MACs sind daher durch dritte Personen, die nicht Mitglieder der Gruppe sind, nicht überprüfbar und können deshalb auch nicht zum Nachweis der Nicht-Abstreitbarkeit der Herkunft verwendet werden.

11.7.3 Digitale Signatur

Digitale Signaturen beruhen auf der Verwendung eines Schlüsselpaares, bestehend aus einem öffentlichen und einem privaten Schlüssel. Jeder Kommunikationspartner verfügt über einen geheimen Schlüssel für die Signierung der Nachrichten und einen öffentlichen Schlüssel, der für die Verifizierung der Nachrichten seitens der anderen Kommunikationspartner verwendet wird. Dementsprechend existieren die Rollen des Signierers und Verifizierers (Menezes, van Oorschot und Vanstone 2001). Bekannte Signaturverfahren sind RSA (Rivest, Shamir und Adleman 1978), DSA (National Institute of Standards and Technology 2007) oder ElGamal (ElGamal 1985).

In dem hier beschriebenen Verfahren werden solche Schlüsselpaare während des DH-Authentifikationsverfahrens generiert. Jeder Teilnehmer hat dem Initiator während des DH-Verfahrens seinen öffentlichen Schlüssel P_X mitgeteilt (siehe Kapitel 11.1.3). Somit besitzt der Initiator (im Beispiel der Teilnehmer A) alle öffentlichen Schlüssel der Teilnehmer.

Außerdem wurde durch das DH-Verfahren zwischen dem Initiator A und jedem Teilnehmer eine sichere Verbindung etabliert. Über diese Verbindung kann der Initiator jedem Teilnehmer nun die öffentlichen Schlüssel aller anderen Teilnehmer mitteilen.

Anschließend liegen folgende öffentliche und private Schlüssel vor (hier am Beispiel der Gruppenkommunikation mit vier Teilnehmern und zentraler Kontrolle durch den Initiator):

Verbindung AB

$$S_A , S_B \tag{13}$$

$$P_{A1} = {g_1}^{S_A} (mod\ p_1) \tag{14}$$

$$P_B = {g_1}^{S_B} (mod\ p_1) \tag{15}$$

Verbindung AC

$$S_A , S_C \tag{16}$$

$$P_{A2} = {g_2}^{S_A} (mod\ p_2) \tag{17}$$

$$P_C = {g_2}^{S_C} (mod\ p_2) \tag{18}$$

Verbindung AD

$$S_A , S_D \tag{19}$$

$$P_{A3} = {g_3}^{S_A} (mod\ p_3) \tag{20}$$

$$P_D = {g_3}^{S_D} (mod\ p_3) \tag{21}$$

Man beachte, dass A zwar einen gleich bleibenden geheimen Schlüssel S_A nutzt, aber für jede Verbindung i unterschiedliche Parameter p_i, g_i verwendet und unterschiedliche P_{Ai} generiert. Der Initiator verfügt also nach der Authentifikation mit n Teilnehmern über $n-1$ unterschiedliche öffentliche Schlüssel. Dies ist unproblematisch für das eigentliche Zugangsberechtigungsverfahren, erschwert aber die Anwendung von digitalen Signaturen, da alle öffentlichen Schlüssel verteilt und eventuell signiert werden. Um diesem Problem zu begegnen, sind folgende Alternativen möglich:

1. Der Initiator nutzt für jede bilaterale Kommunikation die gleichen Parameter und verfügt somit über das gleiche Schlüsselpaar und vor allem den gleichen öffentlichen Schlüssel für jede Kommunikation. Die Realisierung erfolgt mit einer Kombination aus DH-, DSA-Algorithmus und einen AES-Schlüssel.
2. Ein ähnliches Konzept kann verfolgt werden, indem für den Schlüsselaustausch, die Verschlüsselung und die digitale Signatur der ElGamal-Algorithmus verwendet wird.
3. Während des Zugangsverfahrens generiert der Initiator zwar für die Verschlüsselung unterschiedliche bilaterale DH-Schlüsselpaare, für die Signierung der Daten würfelt jedoch jeder Teilnehmer ein RSA-Schlüsselpaar, welches über die ganze Kommunikation gleich bleibt. Dieses RSA-Schlüsselpaar wird wiederum über den Initiator an die anderen Teilnehmer verteilt.

In den nächsten Kapiteln werden diese Alternativen genauer erläutert und bewertet.

11.7.3.1 Diffie-Hellman, Digital Signature Algorithm

Für diese Wahlmöglichkeit wird eine Kombination aus (1) Diffie-Hellmann (DH) für den Schlüsselaustausch und (2) Digital Signature Algorithm / Digital Signature Standard (DSA/DSS) für die Signierung der Nachrichten verwendet (National Institute of Standards and Technology 2007). Der Initiator verwendet für alle DH-Verbindungen den gleichen öffentlichen Schlüssel. Um dies zu erreichen, muss er für alle Verbindungen die gleichen Parameter p_i, g_i wählen.

In diesem Fall werden die Schlüssel der bilateralen Verbindungen, hier wieder am Beispiel der vier Teilnehmer, folgendermaßen berechnet:

Verbindung AB

$$S_A, S_B \quad (22)$$

$$P_A = g^{S_A} (mod\ p) \quad (23)$$

$$P_B = g^{S_B} (mod\ p) \quad (24)$$

Verbindung AC

$$S_A, S_C \quad (25)$$

$$P_A = g^{S_A} (mod\ p) \quad (26)$$

$$P_C = g^{S_C} (mod\ p) \quad (27)$$

Verbindung AD

$$S_A, S_D \quad (28)$$

$$P_A = g^{S_A} (mod\ p) \quad (29)$$

$$P_D = g^{S_D} (mod\ p) \quad (30)$$

Danach nutzt er den DSA-Algorithmus, um die Signatur zu erzeugen. Dafür werden die DH-Parameter verwendet, um weitere einmalige, wechselnde öffentliche und private Schlüssel zu erzeugen und damit jede neue Nachricht zu signieren.

Betrachtet man als Beispiel die Versendung einer signierten Nachricht M von Teilnehmer B, gilt Folgendes: Er verfügt über die oben genannten Schlüssel

$$S_B \quad (31)$$

$$P_B = g^{S_B} (mod\ p) \quad (32)$$

Der Signierer B generiert Zufallszahlen als einmaligen privaten Schlüssel K_B und den zugehörigen einmaligen öffentlichen Schlüssel:

$$R_B = g^{K_B} mod\ p (mod\ q) \quad (33)$$

Für den Signaturvorgang wird der Hashwert der Nachricht $H(M)$ gebildet und die Signatur berechnet:

$$Sig = (H(M) + P_B * R_B) * {K_B}^{-1}\ (mod\ q) \quad (34)$$

Der Teilnehmer B (Signierer) verschickt das Zahlentripel (M, R_B, Sig) an alle Teilnehmer. Ein Empfänger dieser Nachricht (z. B. Teilnehmer C) erhält diese Nachricht (M, R_B, Sig) und übernimmt die Rolle des Verifizierers. Er verfügt über den öffentlichen Schlüssel P_B und kennt die Parameter p, g und q. Er berechnet

$$T_C = {S_C}^{-1} (mod\ q) \quad (35)$$

und erstellt den Hashwert $H(M)$ der Nachricht M und berechnet den Term für die Verifikation:

$$V_C = {P_B}^{R_B * T_C} * g^{H(M) * T_C} \big(mod\ p (mod\ q)\big) \quad (36)$$

Ergibt sich eine Übereinstimmung $V_C = R_B$, dann ist die Signatur authentisch und somit die Integrität der Nachricht gegeben.

11.7.3.2 ElGamal

Eine zweite Möglichkeit, eine digitale Signatur zu realisieren, ist der ElGamal-Algorithmus. Dies ist ein asymmetrisches Verfahren, welches einen Schlüsselaustausch, Verschlüsselung und Signierung ermöglicht. Für das weitere Vorgehen wird nur die Signierung betrachtet. Prinzipiell wäre es auch möglich, die Überprüfung der Authentizität, die Verschlüsselung und die Signierung mit einem ElGamal-Verfahren durchzuführen (ElGamal 1985). Dabei wird wie auch in der DSA-Alternative ein festes Schlüsselpaar pro Teilnehmer erzeugt und darauf aufbauend einmalige Schlüsselpaare für die Signierung der Nachrichten abgeleitet.

Analog zum vorhergehenden Kapitel, wird hier zur Erläuterung die Signierung der Nachrichten von Teilnehmer B betrachtet. Er generiert einen einmaligen privaten Schlüssel K_B und berechnet den öffentlichen Schlüssel

$$R_B = g^{K_B} mod\ p \quad (37)$$

Im nächsten Schritt berechnet er den Hashwert $H(M)$ seiner Nachricht M und anschließend die Signatur

$$(38) \qquad Sig = (H(M) - P_B * R_B) * {K_B}^{-1} \, (mod\ p - 1)$$

Er sendet das Zahlentripel (M, R_B, Sig) an alle Teilnehmer. Der Empfänger (z. B. der Teilnehmer C) erhält diese signierte Nachricht (M, R, Sig) und übernimmt wiederum die Rolle des Verifizierers. Er kennt den öffentlichen Schlüssel P und die Parameter p und g. Die erhaltene Nachricht ist dann authentisch wenn gilt

$$(39) \qquad g^{H(M)} \equiv {R_B}^{Sig} * {P_B}^{R_B} (mod\ p)$$

11.7.3.3 Rivest-Shamir-Adleman

Die dritte Implementierungsalternative ist die Nutzung des RSA-Algorithmus (Rivest, Shamir und Adleman 1978), der sowohl für die Verschlüsselung einer Kommunikation als auch für die Erstellung einer digitalen Signatur genutzt werden kann. Für den vorliegenden Fall würde sich sein Einsatz auf die Signierung der Nachrichten begrenzen. Dafür müsste jeder Teilnehmer ein individuelles Schlüsselpaar generieren und seinen öffentlichen Schlüssel den anderen Teilnehmer zugänglich machen, etwa durch Verteilung über den Initiator. Dieses RSA-Schlüsselpaar steht in keinem Zusammenhang zu den Schlüsseln, die durch das DH-Verfahren während des Zutrittsverfahrens generiert worden sind.

Die Signierung der Nachrichten läuft nach folgendem Verfahren ab. Zunächst generiert jeder Teilnehmer seine Schlüssel (im vorliegenden Beispiel der Teilnehmer B) und verteilt diese während des Zugangsberechtigungsmechanismus über den Initiator:

$$(40) \qquad S_B, P_B$$

Er berechnet den Hashwert (z. B. mit Hilfe der SHA-1 Hashfunktion) $H(M)$ seiner Nachricht M und anschließend die Signatur

$$(41) \qquad Sig = \big(H(M)\big)^{S_B} * (mod\ P_B)$$

Er sendet die Sig und die Nachricht M, somit kann jetzt der Verifizierer seine Berechnungen durchführen. Gilt für den Verifizierer folgende Formel, dann ist Sig die Signatur der von B gesendeten Nachricht:

$$(42) \qquad Sig^e \equiv H(M) * (mod\ P_B)$$

Dabei ist e ein systembekannter öffentlicher Parameter, der zum öffentlichen Schlüssel P_B von B passt.

11.8 Ausgewählter Ansatz für den Schutz der Integrität in einer Gruppe

Für die Realisierung einer sicheren und verbindlichen Gruppenkommunikation und -kooperation wird der Mechanismus der digitalen Signatur gewählt. Denn wie schon im Kapitel über die unterschiedlichen Integritätsmechanismen analysiert wurde (Abschnitt 11.7), bietet die digitale Signatur als einzige einen Integritätsschutz sowohl vor externen als auch von internen Angreifern. Dabei kann je nach Präferenz des Programmierers entweder eine DSA-Signatur

oder eine RSA-Signatur verwendet werden. Bei der DSA-Signatur basieren die verwendeten Signaturschlüssel auf den authentifizierten Schlüsseln, die während der Schlüsselvereinbarung mit DH ausgetauscht wurden. Bei der RSA-Signatur müsste ein neues Signaturpaar erzeugt und über einen sicheren Kanal an alle Teilnehmer verteilt werden. Nachdem der Initiator mit jedem Teilnehmer den DH-Schlüsselaustausch, die Verschlüsselung der Zweierverbindung und die Wiedererkennung durch das Videobild durchgeführt hat, verteilt er über diese sichere bilaterale Verbindung, die er mit den Teilnehmern hat, die öffentlichen Schlüssel P_X .

Dieses Konzept der spontanen Erzeugung von digitalen Signaturen wurde gewählt, da damit sowohl die Integrität der Chat-Kommunikation, als auch die langfristige Verbindlichkeit des Ergebnisses einer Kooperation (mehr dazu später in Kapitel 11.3) gewährleistet werden kann.

Weiterhin kann dieser Lösungsansatz der spontanen digitalen Signaturen auch für andere Anwendungen, die auf Grundlage einer IM-Plattform realisiert werden, verwendet werden. So wird etwa im Fall der Entscheidungsprozesse (Kapitel 13.2) dieser Mechanismus verwendet, um die Voten zu signieren (vgl. Kapitel 13.5 Wahlprotokoll).

11.9 Bedrohungen der Integrität der Nachrichten für die ausgewählte Variante

In der Bedrohungsanalyse bezüglich der Authentizität in Kapitel 11.4 wurden auch Hinweise zu Angriffen auf die Integrität gegeben. An dieser Stelle soll diese Thematik vertieft werden. Ein Angreifer kann die ausgetauschten Nachrichten manipulieren, eine Verbindung stören oder gar neue Nachrichten einschleusen, wenn er z. B. Kenntnis des Schlüssels K_{ALL} hat oder sogar ein unsicherer Verschlüsselungsalgorithmus verwendet wird.

Unter der Annahme, dass alle Teilnehmer ehrlich sind und sich konform zur Sicherheitspolicy der Kommunikation verhalten, ist es für einen Angreifer schwierig die verschlüsselten Inhalte unbemerkt zu manipulieren. Die verwendeten symmetrischen kryptografischen Verfahren (wie z. B. AES oder ElGamal) gelten aktuell als sicher. Nichtsdestotrotz wird an dieser Stelle das Angriffspotenzial bezüglich der Integrität der Nachrichten diskutiert, um auch dieses Restrisiko einer Manipulation identifizieren zu können.

11.9.1 Unehrlicher Teilnehmer

Hält sich ein Teilnehmer nicht an die Sicherheitspolicy der Gruppenkommunikation und (1) überlässt seine Position bewusst einem externen Angreifer oder (2) verrät den symmetrischen Schlüssel der Kommunikation, hat das folgende Auswirkungen auf die Integrität:

Im ersten Fall wird zwar die Authentizität des Nachrichtensenders angegriffen (der Angreifer sendet, nicht der richtige Teilnehmer), die Integrität der übermittelten Nachrichten bleibt aber gewahrt (Kapitel 11.4.2) (die Nachrichten des Angreifers erreichen unverändert das Ziel).

Ähnliches gilt für den zweiten Fall, es wird zwar die Vertraulichkeit der Verbindung durch die Weitergabe des symmetrischen Schlüssels zerstört (Kapitel 11.4.2), aber die Nachrichtenübertragung bleibt integer. Auch wenn der Angreifer die Nachrichten des echten Teilnehmers abhören kann, erreichen diese zumindest unverändert ihr Ziel.

Ein weiterer Angriff auf die Integrität der ausgetauschten Nachrichten kann durch einen unehrlichen Teilnehmer erfolgen, der gleichzeitig als interner Angreifer auftritt. Angenommen, der Teilnehmer B möchte den Ausgang einer Kooperation manipulieren, indem er die Verbindung zwischen Person C und Person A angreift. Dadurch, dass er Mitglied dieser Kooperation ist, befindet sich der allgemeine Schlüssel K_{ALL} schon in seinem Besitz. Somit ist er in der Lage, übertragene Nachrichten zu ver- und entschlüsseln. Ein Empfänger kann anhand der Verschlüsselung nicht erkennen, von wem genau in der Gruppe die Daten stammen. Er kann nur sicher sein, dass die Nachricht von *einem* Besitzer des gemeinsamen symmetrischen Schlüssels kommt. Wird dieser Angriff auf den Chat-Kanal ausgeübt, kann dieses Vorgehen erfolgreich sein, weil sich textuelle Daten leichter als Audio- oder Video-Daten in der notwendigen Zeit (Echtzeit) manipulieren lassen. Der Aufwand für einen solchen internen Angreifer auf die Video-Daten ist wesentlich größer und nicht ohne Entdeckung durchzuführen. Siehe hierzu auch die Diskussion in Kapitel 11.4.3. Eine Variante, die leichter durchzuführen ist, ist anders als eine Einspeisung von Bildern eine Störung der Kommunikation.

Alle diese Angriffe sind theoretisch auch dann möglich, wenn für die Verschlüsselung bilaterale Schlüssel K_{XY} verwendet werden. In diesem Fall ist es für den internen Angreifer allerdings schwierig, die Kommunikation zu manipulieren, da er den symmetrischen Schlüssel der Zweierkommunikation nicht kennt. Um diesen zu berechnen, bedarf es eines ähnlichen Aufwands, den ein externer Angreifer investieren müsste, um K_{ALL} zu erraten.

11.9.2 Externer Angreifer und ehrliche Teilnehmer

Durch den Zugangsberechtigungsmechanismus einschließlich der gegenseitigen Authentifikation der Teilnehmer, die Verschlüsselung der Kommunikation und die Tatsache, dass die Kommunikation in Realtime stattfindet, bleiben einem externen Angreifer nur wenige Möglichkeiten für einen Angriff ausschließlich auf die Integrität der Nachrichten, dabei gelten alle Teilnehmer als ehrlich.

Eine Manipulation wäre zum Beispiel dann möglich, wenn der externe Angreifer einen der Clients der Teilnehmer durch Malware verändert. Dazu könnte er beispielsweise ein modifiziertes Plug-in (das angeblich die sichere Anwendung implementiert) zum Download anbieten. Er kann nur dann einen Angriff auf die Integrität der Kommunikation ausüben, wenn die Übernahme des Clients schon *vor* der Überprüfung der Zugangsberechtigung erfolgreich ist. Er hat dann auch Zugriff auf Mechanismen und Schlüssel, welche eigentlich die Integrität gewährleisten sollen. Dieser Angriff bezieht sich über die Integrität hinaus auch auf die Authentizität des Erzeugers und wurde im Kapitel 11.4.3 erläutert.

Eine Manipulation des Clients und des Nachrichtenaustausches, *nachdem* der Zugangsberechtigungsprozess samt Überprüfung der Authentizität und Verschlüsselung der Kommunikation abgeschlossen ist, ist nicht möglich. Der Angreifer müsste die gerade laufende Applikation mit seiner angepassten (manipulierten) Version ersetzen, und das in Echtzeit.

11.9.3 Vertrauensmodell der Integrität

Betrachtet man das Vertrauensmodell und seinen k-resilience Wert im Zusammenhang mit der Integrität der Nachrichten, müssen zwei Rollen unterschieden werden: die Rolle des Initiators und die eines Teilnehmers. Für die Integrität ist die Rolle des Initiators wichtig, weil dieser für die Weiterleitung der öffentlichen Schlüssel verantwortlich ist. Verhält er sich ehrlich, dann muss noch die Ehrlichkeit der Teilnehmer überprüft werden. Durch die Verwendung der digitalen Signaturen hat keiner der Teilnehmer die Möglichkeit die Nachrichten zu manipulieren. Somit ist der k-resilience Wert

$$1\ out\ of\ \{Initiator\} \tag{43}$$

11.10 Verbindlichkeit der Kooperationsergebnisse

Im vorangegangenen Kapitel wurde erläutert, wie mit Hilfe eines geeigneten Authentifikationsverfahrens und den dazugehörigen kurzfristig erzeugten Sicherheitselementen die Kommunikation innerhalb einer spontan gebildeten Gruppe abgesichert werden kann.

Ein weiterer Beitrag dieser Arbeit ist die Entwicklung eines Konzepts zur Gewährleistung der Verbindlichkeit der Kooperation. Basierend auf der Verwendung der kurzfristig erzeugten Sicherheitselemente, soll in diesem Kapitel die Durchführung einer langfristig verbindlichen Kooperation realisiert werden. Ziel ist dabei, dass das Ergebnis dieser Kooperation (z. B. das Ergebnis einer Wahl) zu einem *späteren Zeitpunkt* (nach der Gruppensitzung) überprüfbar und nicht abstreitbar ist. Dazu wird zunächst das Modell einer herkömmlichen Kooperation (vor Ort, ohne technisches System) mit Diskussion und Abstimmung in einer kleinen Gruppe betrachtet. Solche Prozesse finden täglich in Firmen, Universitäten oder Vereinen statt. Dabei werden eine Zusammenfassung der Diskussion und die Ergebnisse von Abstimmungen in Form eines *Sitzungsprotokolls* auf Papier festgehalten. Dieses wird – je nach Sicherheitsanforderungen der konkreten Anwendung – von allen Teilnehmern oder nur vom Vorsitzenden der Sitzung unterschrieben und von einer weiteren Instanz, z. B. einem Sekretariat, in einer zentralen Dokumentenablage aufbewahrt.

Wenn einer der Teilnehmer das Ergebnis zu einem späteren Zeitpunkt nachvollziehen möchte, vertraut er auf die Ehrlichkeit des „Aufbewahrers“ (z. B. des Sekretariats). Das heißt, er wird in der Regel davon ausgehen, dass erstens der „Aufbewahrer“ das Dokument mit dem Ergebnis und den zugehörigen Unterschriften nicht manipuliert hat und zweitens sich kein Unbefugter Zugang verschaffen konnte. Will er die Authentizität des Dokumentes nachprüfen, muss er entweder darauf vertrauen, dass er selbst die vorhandenen Unterschriften erkennen und den richtigen Mitgliedern zuordnen kann, oder sich von den Mitgliedern nachträglich Unterschriftenproben zum Vergleich beschaffen. Um ein höheres Sicherheitsniveau zu erreichen könnte das Protokoll in gleichem Wortlaut auch an mehreren Stellen aufbewahrt werden.

11.11 Mögliche Ansätze für die Verbindlichkeit in einer Gruppe

Um ein ähnliches Konzept zu entwickeln und mindestens das gleiche Sicherheitsniveau in einer IT-gestützten Gruppenkooperation – auf der Basis eines Instant-Messaging-Systems – zu errei-

chen, müssen an dieser Stelle die Eigenschaften von IM-Systemen (siehe Kapitel 6.1 Featuremodell) bezüglich ihrer Auswirkung auf die Verbindlichkeit der Kommunikation und der dabei gewonnenen Ergebnisse geprüft werden. Bei modernen IM-Systemen wird die Chat-Kommunikation je nach Implementierung über einen zentralen Server übertragen, die Historie einer Diskussion aber *client*seitig gespeichert. Das bedeutet, jeder Teilnehmer einer Gruppenkommunikation speichert die ausgetauschten Nachrichten lokal und kann diese prinzipiell nach Belieben bearbeiten. Diese Funktion, d. h. das automatische Speichern eines Protokolls, ist bei den meisten Systemen für die Chat-Kommunikation eine Standardeinstellung. Der Videostream einer Video-Konferenz wird allerdings während der Übertragung nur temporär im RAM zwischengespeichert und ist damit schon kurz nach dem Anzeigen nicht mehr verfügbar. Abweichend davon erlauben externe Werkzeuge oder Plug-ins trotzdem einen Mitschnitt der Videoübertragung. Somit ist auch für die Videokommunikation eine clientseitige Speicherung des Nachrichtenstroms möglich. Vergleicht man das mit dem Szenario der Protokollerstellung während einer (herkömmlichen) Gruppensitzung, würde dies bedeuten, dass jeder Teilnehmer über ein eigenes Protokoll verfügt. Im günstigsten Fall sind alle diese Protokolle gleich (wenn kein Übertragungsfehler aufgetreten ist, alle Teilnehmer ehrlich sind und kein externer Angriff stattgefunden hat), im ungünstigsten Fall würden bis zu n unterschiedliche Protokolle existieren. Also besteht die Herausforderung bei der Erstellung eines Lösungsansatzes für die Gewährleistung der Verbindlichkeit darin, aus unterschiedlichen bei der Kollaboration ausgetauschten Protokolldaten (z. B. einzelne Aussagen oder Ergebnisse) das eigentliche „offizielle Ergebnis“ zu identifizieren und dies unter Sicherstellung der Nicht-Abstreitbarkeit aufzubewahren.

In der vorliegenden Arbeit werden für diese typische Aufgabe eines Protokollführers innerhalb der Gruppe verschiedene Rollen vergeben. Jede dieser Rollen ist mit unterschiedlichen Aufgaben verbunden. Diese Aufgabenteilung (engl. „Separation of Duty“) (Clark und Wilson 1987) wird genutzt, um das gewünschte Sicherheitsniveau dadurch zu gewährleisten, dass sich für eine Manipulation mehrere Personen zusammentun müssten. Die Verteilung der folgenden drei Rollen wird spontan am Anfang des virtuellen Treffens unter den Mitgliedern vereinbart:

- Der *Signierer* unterzeichnet das aufzubewahrende Ergebnis mit seiner digitalen Unterschrift. Dafür benötigt er einen entsprechenden Schlüssel bzw. ein Zertifikat. An dieser Stelle wird der Sicherheitsmechanismus eines spontan hergestellten Zertifikats gewählt. Dieses wird erzeugt, indem ein öffentlicher Schlüssel eines Teilnehmers durch einen anderen Teilnehmer signiert wird.
- Der *Zertifikat-Ersteller* hat die Aufgabe, den öffentlichen Schlüssel des Signierers zu überprüfen, zu signieren und somit das Zertifikat zu erstellen.
- Der *Aufbewahrer* des Ergebnisses hat die gleiche Aufgabe, wie sein Pendant in der Vor-Ort-Variante, d.h. er bewahrt das Ergebnis (Datensatz der gesamten Chat-Verkehrs, des Videostroms und das Ergebnis der elektronischen Wahl) samt den benötigten Schlüsseln für eine eventuelle Überprüfung auf.

Je nachdem, wie diese Rollen verteilt werden und zusammenspielen, lassen sich – mit unterschiedlichem Aufwand – unterschiedliche Sicherheitsniveaus realisieren.

In den folgenden Kapiteln werden vier solcher Lösungsansätze mit unterschiedlichen Rollenverteilungen vorgestellt, die entsprechend unterschiedliche Sicherheitsniveaus erreichen. Anschlie-

ßend werden die Eigenschaften und Limitierungen dieser vier Varianten herausgearbeitet. Die Varianten unterscheiden sich vor allem in der Anzahl der Personen, welche die oben genannten drei verschiedenen Rollen übernehmen. Im einfachen Fall ist es akzeptabel, wenn eine Person alle Rollen übernimmt. Um aber ein höheres Sicherheitsniveau zu erreichen, müssen die Rollen auf mehrere Teilnehmer verteilt werden, um Manipulationen zu erschweren.

11.11.1 Variante „eine Person, drei Rollen“

In diesem ersten Szenario werden alle Rollen von einem Teilnehmer übernommen. Betrachtet man das Beispiel einer Gruppe mit vier Teilnehmern könnten, all diese Aufgaben z. B. von der Person D übernommen werden (vgl. Abbildung 37). Nach einer Chat-Diskussion oder einer Anwendung wie z. B. einer elektronischen Wahl (vgl. Abschnitt 13.4) würde Person D dann das Protokoll der Chat-Kommunikation oder das (eventuell selbst berechnete) Wahlergebnis langfristig aufbewahren.

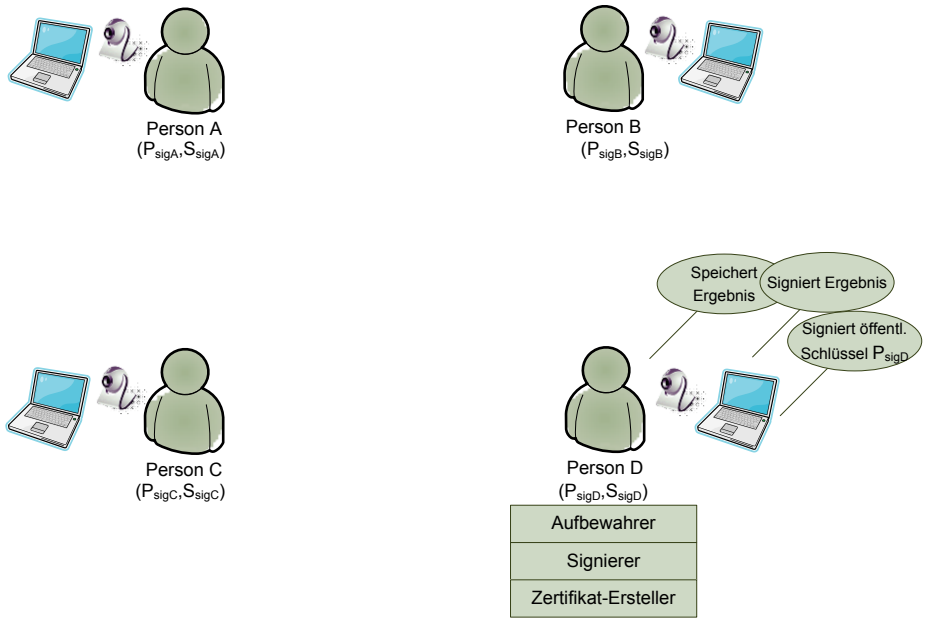

Abbildung 37 – Rollenübernahme für die langfristige Archivierung durch einer Person

Ob D dieses Ergebnis mit seinem Schlüsselpaar aus dem Zugangsberechtigungs- und Integritätsmechanismus $P_{sigD,}\ S_{sigD}$ signiert oder es sogar mit einem selbst signierten Zertifikat unterschreibt, bleibt für die Gewährleistung der Verbindlichkeit gleich. Der Aufbewahrer kann das Ergebnis jederzeit neu erstellen und speichern – auch mit veränderten Werten. Möchte ein weiteres Mitglied (der *Anfrager*) zu einem späteren Zeitpunkt das Ergebnis der Kommunikation erfragen, muss er dem Aufbewahrer vertrauen, da er keine weitere Möglichkeit hat, diese Aussage zu überprüfen. Durch die Übernahme aller Rollen durch eine einzige Instanz kann man hier nicht wirklich von einer Aufgabenteilung sprechen. Hier existiert letztendlich nur eine Rolle, die des Aufbewahrers. Dieser Fall wird erstens der Vollständigkeit halber erwähnt und zweitens weil dieser Fall der typischen Aufbewahrung von Text-Nachrichten in einer IM-Kommunikation entspricht, die viele IM-Clients als Standardeinstellung speichern. Dabei kann jeder Teilnehmer einer Chat-Kommunikation kann selbstständig entscheiden, für welchen Zeit-

raum er eine durchgeführte Kommunikation speichern möchte, da dies auf seinem eigenen Client passiert.

11.11.2 Variante „zwei Personen, drei Rollen“

In der zweiten Variante wird – zur Erhöhung des Sicherheitsniveaus – die Rolle des Aufbewahrers entmachtet, indem die Rollen des Signierers und Zertifikat-Erstellers durch eine weitere Person ausgefüllt werden. Durch die vorangegangenen Schritte, d.h. die Anwendung der Sicherheitsmechanismen, wird für jeden Teilnehmer ein Schlüsselpaar P_{sigX}, S_{sigX} erzeugt, welches (im Rahmen der Verbindlichkeit) für eine spätere Überprüfung der Ergebnisse benötigt wird. Da es sich bei den verwendeten Schlüsseln um spontane DH-Schlüssel handelt (vgl. Kapitel 11.8), ist das DSA-Signierverfahren zu verwenden (Schneier 1996; National Institute of Standards and Technology 2007). Eine Alternative ist die zusätzliche Generierung neuer Signaturschlüssel (z.B. RSA-Signaturschlüssel) und deren sichere Übertragung über den DH-Kanal (vgl. Kapitel 14.3 SecureMUC). Diese Schlüssel werden über den Moderator der Gruppenkommunikation verteilt, sobald die Konferenz initiiert ist und sich alle Teilnehmer authentifiziert haben.

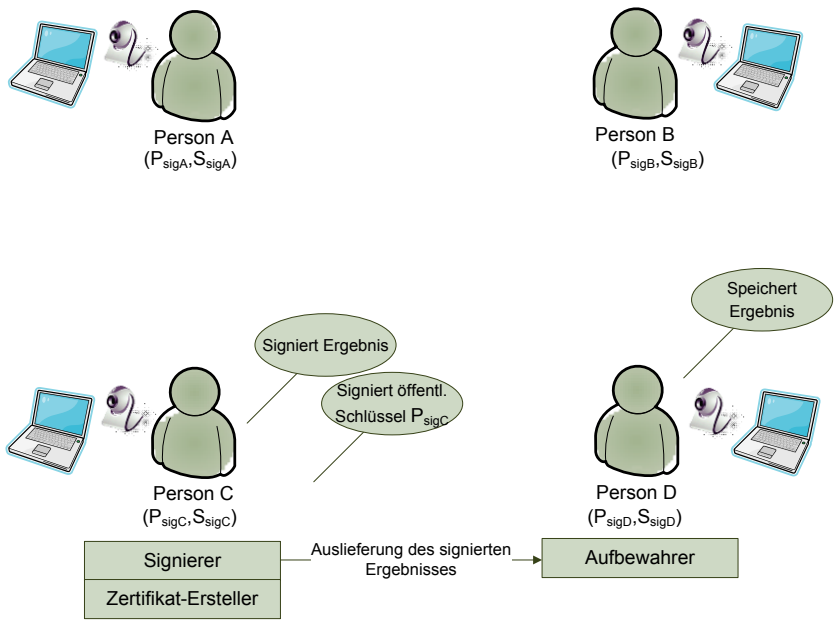

Abbildung 38 – Rollenübernahme für die langfristige Archivierung durch zwei Personen

Der Signierer hat – wie jedes der Mitglieder – das Ergebnis Res der Kommunikation vorliegen. Dies kann beispielsweise als gepacktes Dateiarchiv realisiert werden, das Chat-Nachrichten, eventuell Videomitschnitte der Diskussion und die Ergebnisse der Abstimmung (z. B. als XML-Datei) enthält. Er signiert dieses Ergebnis mit seinem privaten Signaturschlüssel S_{sigC} und schickt es einem weiteren Teilnehmer, der die Rolle des Aufbewahrers übernommen hat:

$$(44) \qquad sigS_{sigC}(Res), Res, P_{sigC}$$

In diesem Fall gilt genauso wie in dem Fall zuvor (Variante „eine Person, drei Rollen“, Abschnitt 11.11.1), dass es für die Gewährleistung der Verbindlichkeit keine Rolle spielt, ob die Person C zusätzlich ein Selbstzertifikat erstellt.

Der Aufbewahrer seinerseits verifiziert anhand des ihm bekannten öffentlichen Schlüssels des Signierers P_{sigC} die Echtheit des Ursprungs der Nachricht und überprüft weiterhin, ob das signierte Ergebnis, welches er gerade erhalten hat, mit seinem eigenen selbst ermittelten Ergebnis (oder Protokoll der gesamten Kommunikation) übereinstimmt. Sind beide Überprüfungen erfolgreich, speichert er den öffentlichen Schlüssel von P_{sigC} und das Ergebnis und hält diese Daten für spätere Nachfragen und Überprüfungen bereit.

Um das Szenario abzuschließen, muss jetzt noch der letzte Schritt betrachtet werden, wenn ein Mitglied der Gruppenkommunikation (der sogenannte *Anfrager*) zu einem späteren Zeitpunkt das Ergebnis der Sitzung nachvollziehen möchte. Die eigentliche Kollaboration ist zu diesem Zeitpunkt lange abgeschlossen; die Gruppe hat sich aufgelöst. Daher wendet sich der Anfrager an den Aufbewahrer. Dieser stellt dem Anfrager das Ergebnis der Kommunikation zur Verfügung (das Archiv mit Chat-Nachrichten, Videostream und sonstigen Ergebnissen). Der Aufbewahrer bekräftigt seine Aussage durch die Verifizierung des Ergebnisses anhand des gespeicherten öffentlichen Schlüssels (oder des selbsterstellten Zertifikats).

11.11.3 Variante „drei Personen, drei Rollen“

In einer dritten Variante werden (nachdem durch den Zugangsberechtigungsmechanismus alle Schlüsselpaare P_{sigX}, S_{sigX}, die für die Signierung verwendet werden, über den Moderator verteilt sind) die drei Rollen des Aufbewahrers, Signierers und Zertifikat-Erstellers spontan drei Personen zugeordnet.

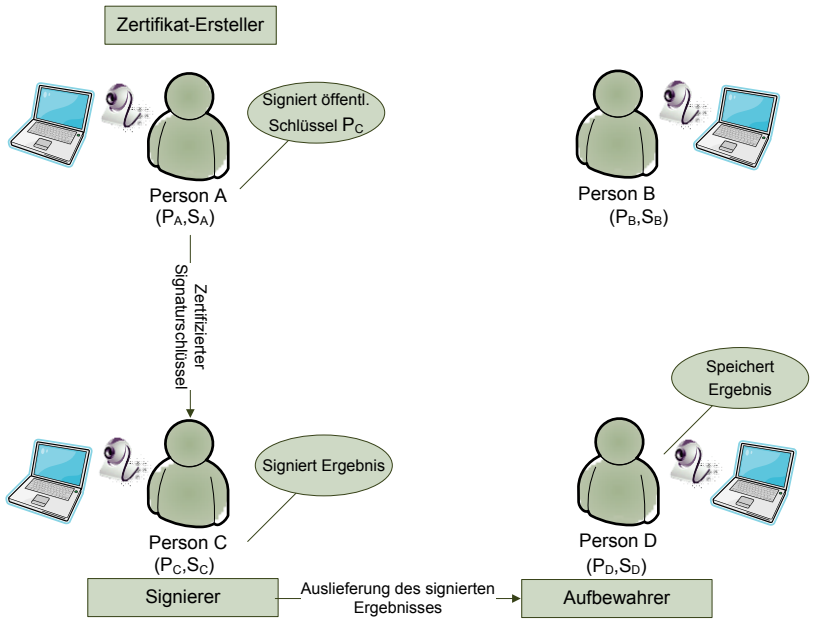

Abbildung 39 – Rollenübernahme für die langfristige Archivierung durch drei Personen

Erst in diesem Szenario ist eine klare Aufgabenteilung, eine wirkliche „Separation of Duty“ möglich. In Abbildung 39 wird ein Beispiel mit vier Teilnehmern gezeigt, in dem Person C die Rolle des Signierers übernimmt. Der Signierer lässt seinen öffentlichen Schlüssel P_{sigC} von dem Teilnehmer (Person A) signieren, der die Rolle des Zertifikat-Erstellers übernommen hat, dadurch entsteht das Zertifikat zP_{sigC} .

Ähnlich wie im Fall zuvor (Variante „zwei Personen, drei Rollen, Abschnitt 11.11.2) ist der Ablauf wie folgt: Der Signierer signiert das Ergebnis Res mit seinem privaten Signaturschlüssel S_{sigC} und schickt es zusammen mit seinem Zertifikat (dem signierten öffentlichen Schlüssel zP_{sigC}) an einem weiteren Teilnehmer, der die Rolle des Aufbewahrers übernommen hat:

$$(45) \qquad sigS_{sigC}\,(Res), Res, zP_{sigC}$$

Der Aufbewahrer verifiziert als Erstes anhand des ihm bekannten öffentlichen Schlüssels des Signierers P_{sigC} und des gerade erhaltenen Zertifikats zP_{sigC} das signierte Ergebnis. Ist dies identisch mit dem Ergebnis, welches er selbst berechnet oder mitprotokolliert hat dann speichert er das Zertifikat und das Ergebnis.

Die Verbindlichkeit ist dadurch gegeben, dass die Ergebnisse der Diskussion und der Abstimmung (genauer: die Inhalte der Archivdatei) zu einem späteren Zeitpunkt nachvollzogen werden können. Für die langfristige Nicht-Abstreitbarkeit des Zertifikats ist es wichtig, dass der Aufbewahrer glaubwürdig bezeugen kann, dass er das Zertifikat und das Ergebnis zum Zeitpunkt der Speicherung geprüft hatte.

In der folgenden Abbildung werden die Daten, die für den Aufbau der verbindlichen Kommunikation und die langfristige Aufbewahrung notwendig sind, noch einmal zusammengefasst:

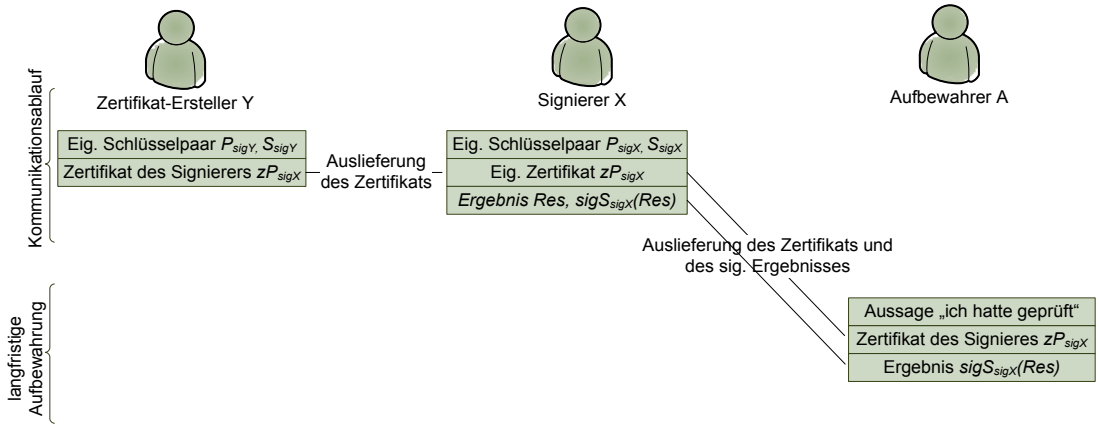

Abbildung 40 – Langfristige Archivierung ohne gesondertes Speichern durch den Zertifikat-Ersteller

Durch die einseitige Aufbewahrung der Daten durch einen Aufbewahrer sind unterschiedliche Angriffe auf die Verbindlichkeit denkbar. Diese werden später in Kapitel 11.13 diskutiert. Um manchen dieser Angriffe vorzubeugen, kann zusätzlich die Aufbewahrung der Zertifikate durch den Zertifikat-Ersteller gefordert werden. Wenn er zusätzlich die Zertifikate aufbewahrt, die er erstellt, dann kann die Glaubwürdigkeit der Zertifikate, die der Aufbewahrer speichert, durch einen Dritten von einem weiteren unabhängigen Aufbewahrungsort überprüft werden. Dabei wird das zuvor dargestellte Verfahren folgendermaßen angepasst (Abbildung 41):

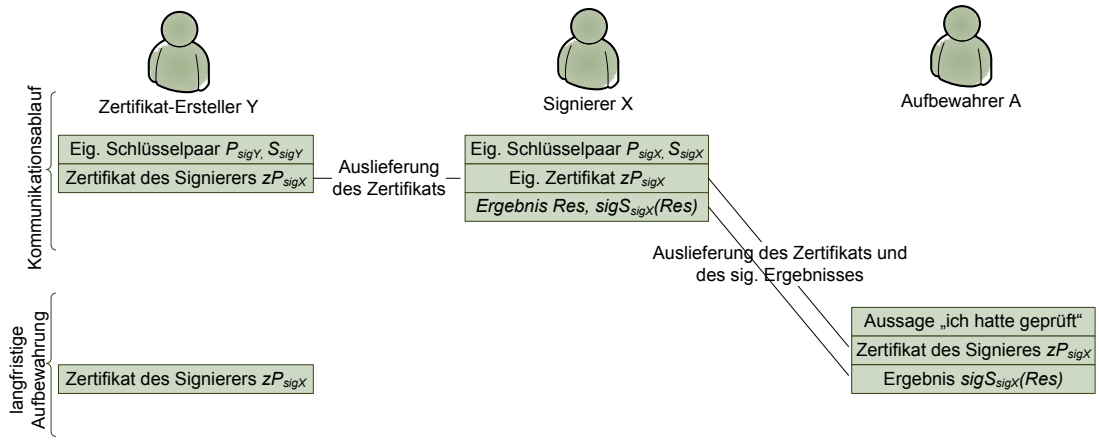

Abbildung 41 – Langfristige Archivierung mit gesondertem Speichern durch den Zertifikat-Ersteller

11.11.4 Variante „n Personen, drei Rollen“

Um die Glaubwürdigkeit der Aussage des Aufbewahrers zu erhöhen und einen eventuellen Kooperationsangriff von Zertifikat-Ersteller und Aufbewahrer zu vermeiden, kann man eine weitere Erweiterung vornehmen: Man führt mehrere Aufbewahrer ein, die dann bei einer späteren Überprüfung die signierten Ergebnisse und zugehörigen Zertifikate sozusagen nebeneinander legen können. Bei einwandfreier Durchführung des Verfahrens müssen alle Ergebnisse gleich lauten (vgl. Abbildung 42):

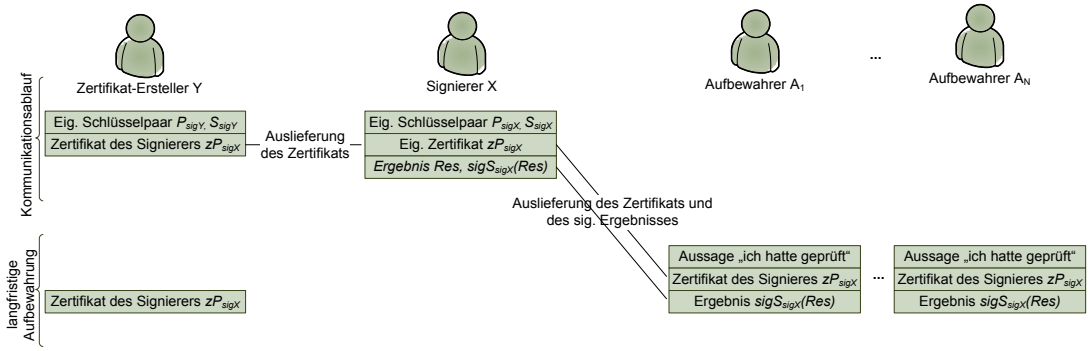

Abbildung 42 – Langfristige Archivierung mit mehreren Aufbewahrern (und gesondertem Speichern der Zertifikate durch den Zertifikat-Ersteller)

11.12 Ausgewählter Ansatz für die Verbindlichkeit in einer Gruppe

Die Entscheidung, welche Variante für die Realisierung der Verbindlichkeit gewählt wird, hängt von der konkreten Anwendung und dem dafür gewünschten Sicherheitsniveau ab. Für die exemplarische Anwendung (die in Kapitel 13 näher erläutert wird) kann beispielsweise die Variante „drei Personen, drei Rollen“ ohne ein separates Speichern des erstellten Zertifikats gewählt werden, da dies das herkömmliche Verfahren bei der Protokollierung von einfachen Sitzungen widerspiegelt und ein ausreichendes Sicherheitsniveau für diese Szenarien bietet. Eine Alternative mit erhöhter Sicherheit ist die Variante „drei Personen, drei Rollen“ mit Speicherung des Zertifikats durch den Zertifikat-Ersteller (vgl. Abbildung 42). Für die weiteren

Betrachtungen wird diese letztere Variante („drei Personen, drei Rollen, mit separater Speicherung“) gewählt.

Grundsätzlich lässt sich festhalten, dass ein höheres Sicherheitsniveau gegen den Aufwand für ein komplexeres Verfahren abgewogen werden muss. Als Richtschnur müssen dabei die Anforderungen der jeweiligen Anwendung gelten. Zum Beispiel wird man bei der langfristigen Speicherung von Wahlunterlagen einen höheren Verteilungsaufwand mit mehreren Aufbewahrern treiben, als bei der Protokollierung einer Unterrichtsstunde. Das hier beschriebene Verfahren erlaubt es dabei, verschiedenen Sicherheitsanforderungen flexibel gerecht zu werden.

11.13 Bedrohungen der langfristigen Verbindlichkeit

Sind alle Mitglieder einer Gruppenkommunikation ehrlich, dann wird sowohl (1) die *Durchführung* der gewünschten Kommunikation und Kooperation (z. B. ein Entscheidungsprozess und die zugehörige Wahl) als auch (2) die *Aufbewahrung* des entsprechenden Ergebnisses unter Gewährleistung aller Anforderungen durchgeführt.

An dieser Stelle sollen die Angriffe auf die Verbindlichkeit der aufbewahrten Ergebnisse (zu 2) erläutert werden. Der Wert der Kollaboration, der hier geschützt werden muss, ist das unverfälschte Ergebnis selbst, das durch Angriffe auf seine Authentizität, d.h. auf seinen Wortlaut (*Integrität*) und seine Herkunft (*Originalität*), bedroht wird. Dieser Wert kann theoretisch vom Signierer oder dem Aufbewahrer (bzw. den Aufbewahrern) angegriffen werden. Die möglichen Fälle werden in den folgenden Abschnitten diskutiert.

11.13.1 Unehrlicher Zertifikat-Ersteller

Wenn der Zertifikat-Ersteller Zertifikate auf Namen berechtigter Teilnehmer ausstellt und sie unberechtigten Teilnehmern zuweist, dann ist die Authentizität der Teilnehmer und die Vertraulichkeit der Gruppenkommunikation gefährdet. Die Integrität des langfristigen *Kooperationsergebnisses* ist aber trotzdem so lange gewahrt, wie der Signierer das richtige Ergebnis signiert und der Aufbewahrer das signierte Ergebnis bei der Speicherung verifiziert hat und beide ehrlich sind. Ein unehrlicher Zertifikat-Ersteller könnte allerdings zu einem späteren Zeitpunkt das erstellte Zertifikat widerrufen (bzw. ein neues anderes erzeugen) und damit die Integrität des echten Ergebnisses, das beim Aufbewahrer gespeichert wurde, in Zweifel ziehen.

11.13.2 Unehrlicher Signierer

Ein unehrlicher Signierer kann das Ergebnis manipulieren und das falsche Ergebnis signieren. Dies wird in den oben beschriebenen Verfahren dadurch aufgedeckt, dass der Aufbewahrer eine Überprüfung des Inhalts und des Zertifikats vornimmt, bevor er das Ergebnis speichert: Er vergleicht sein (selbst ermitteltes) Ergebnis der Kooperation mit dem vom Signierer gesendeten Ergebnis. Da sich zu diesem Zeitpunkt noch alle Mitglieder in der Gruppenkonferenz befinden, kann diese Manipulation vor der langfristigen Speicherung aufgedeckt werden.

11.13.3 Unehrlicher Aufbewahrer

Der Aufbewahrer speichert das Zertifikat und das signierte Ergebnis. Ein unehrlicher Aufbewahrer könnte beim Speichervorgang ein neues selbst signiertes Zertifikat erstellen, damit das

ursprüngliche Zertifikat ersetzen und ein manipuliertes Ergebnis neu signieren. Diese Manipulation würde dann Erfolg haben, wenn das Zertifikat an keiner anderen Stelle aufbewahrt wird. Für die hier ausgewählte Variante verfügt der Zertifikat-Ersteller über eine unmanipulierte Kopie des echten Zertifikats und kann somit eine Manipulation seitens des Aufbewahrers aufdecken. Eine ähnliche Aufdeckung eines solchen Manipulationsversuches ist auch möglich, wenn weitere (ehrliche) Aufbewahrer eingeführt werden.

11.13.4 Unehrliche Signierer und Zertifikat-Ersteller

Wenn ein unehrlicher Signierer in Absprache mit einem unehrlichen Zertifikat-Ersteller ein Zertifikat für einen falschen Teilnehmernamen verwendet, ist zwar die Herkunft des signierten Ergebnisses nicht mehr gesichert (falscher Name des Signierers), die Integrität des Ergebnisses ist dadurch aber nicht beeinträchtigt, denn der Aufbewahrer verifiziert gemäß Protokoll zum Zeitpunkt der Speicherung die Integrität des Ergebnisses persönlich. Nachträglich können Signierer und Zertifikat-Ersteller keine Fälschung beim Aufbewahrer mehr vornehmen. Hier gilt die Glaubwürdigkeit des Aufbewahrers.

11.13.5 Unehrliche Signierer und Aufbewahrer

Schließen sich ein unehrlicher Signierer und ein unehrlicher Aufbewahrer zusammen, um die langfristige Aufbewahrung und Verbindlichkeit zu brechen, so haben sie zwei Möglichkeiten, dies zu tun: (1) Sie signieren das korrekte Ergebnis mit einem falschen, neu erstellten, selbst signierten Zertifikat oder (2) sie generieren ein falsches neues Ergebnis und signieren dies mit dem richtigen Zertifikat. Diese Manipulationen würden dann Erfolg haben, wenn das Zertifikat und das Ergebnis nur bei einem Aufbewahrer aufbewahrt werden. Bei der ausgewählten Variante „drei Personen, drei Rollen“ mit Speicherung des Zertifikats beim Zertifikat-Ersteller (vgl. Abbildung 42) wird die erste Manipulation (1) aufgedeckt, da der Zertifikat-Ersteller auch das Zertifikat speichert. Die zweite Manipulation (2) wird nur dann aufgedeckt, wenn es weitere (ehrliche) Aufbewahrer gibt, da sie das Ergebnisprotokoll immer vor der Speicherung mit dem eigenen vergleichen.

11.13.6 Unehrliche Aufbewahrer und Zertifikat-Ersteller

Bei einer Kombination aus unehrlichem Aufbewahrer und unehrlichem Zertifikat-Ersteller können diese beiden das Ergebnis manipulieren und einen neuen manipulierten öffentlichen Schlüsse (den der Zertifikat-Ersteller zertifiziert) zur Signierung verwenden. Diese Manipulation hat dann Erfolg, wenn das Zertifikat nur beim Aufbewahrer und beim Zertifikat-Ersteller aufbewahrt wird. Sie wird dann aufgedeckt, wenn es weitere (ehrliche) Aufbewahrer gibt.

Wie diese Diskussion zeigt, ist der Schwachpunkt dieser Gruppenkommunikation eine Kollaboration zwischen dem Aufbewahrer und entweder dem Zertifikat-Ersteller oder dem Signierer. Diesem Schwachpunkt kann begegnet werden, indem mehrere Aufbewahrer eingesetzt werden. Diese Lösung ist organisatorischer Art und gehört zum Konzept einer Aufgabenteilung („Separation of Duty“). Dies erhöht zwar das Sicherheitsniveau, verursacht aber gleichzeitig auch eine größere zu speichernde Datenmenge und ein aufwendigeres Kommunikationsverfahren sowohl bei der Speicherung der signierten Ergebnisse, als auch bei ihrer Überprüfung.

11.13.7 Vertrauensmodell der Verbindlichkeit

Auch für die Verbindlichkeit kann ein Vertrauensmodell durch einen k-resilience Werten formalisiert werden: Im Bezug auf die Verbindlichkeit gibt es drei Rollen (Signierer, Aufbewahrer, Zertifikats-Ersteller), die jeweils einer Komponenten (Teilnehmer) zugeordnet werden.

In dem ausgewählten Ansatz (ein Aufbewahrer, ein Signierer und der Zertifikat-Ersteller, der das Zertifikat speichert, vgl. Abbildung 41) kann eine Kooperation des Aufbewahrers mit dem Signierer oder Zertifikat-Ersteller zu einer erfolgreichen Manipulation führen.

Es genügt aber ein ehrlicher Aufbewahrer oder eine ehrliche Kombination aus Zertifikat-Ersteller und Signierer, um einen Angriff zu erkennen. Somit ist gemäß des Vertrauensmodell aus den Abschnitten 11.2.1 und 11.4.41 der k-resilience-Wert:

$$(46) \quad (1\ out\ of\ \{Aufbewahrer\}), (2\ out\ of\ \{ZertifikatErsteller, Signierer\})$$

Betrachtet man die Gewährleistung der Verbindlichkeit durch die Einführung von mehreren Aufbewahrern (erhöhtes Sicherheitsniveau), ist das Ergebnis verbindlich, wenn alle Aufbewahrer das richtige Ergebnis speichern. Eine manipulative Kooperation von m Aufbewahrern ($m < n$) kann zwar stattfinden, aber bei einem Vergleich mit den Ergebnissen der $n - m$ übrigen Aufbewahrer wird die Manipulation auffallen. Hier ist noch zu bedenken, dass zwar erkannt wird, *dass* eine Manipulation vorliegt. allerdings nicht klar erkennbar sein wird, welche Gruppe (die m oder die $n - m$ Aufbewahrer) die Manipulation vorgenommen hat. Ähnliches gilt, wenn ein Aufbewahrer mit einer der anderen Rollen kooperiert. Die Manipulation wird nicht erfolgreich sein, da das gespeicherte Ergebnis von den anderen Ergebnissen abweichen wird. Also für den k-resilience Wert gilt:

$$(47) \quad (1\ out\ of\ \{Aufbewahrer_1, \dots, Aufbewahrer_n\}), \\ (2\ out\ of \{ZertifikatErsteller, Signierer\}).$$

Abschließend werden in der folgenden Abbildung die unterschiedlichen Vertrauensmodelle der Varianten der Verbindlichkeit zusammengefasst.

Variante der Verbindlichkeit	Vertrauensmodell	Erläuterung
Variante "eine Person, drei Rollen"	1 out of {Aufbewahrer}	Die verbindliche Speicherung des richtigen Ergebnisses hängt davon ab, dass der **einzige Aufbewahrer** ehrlich ist. Kein anderer kann eine Manipulation erkennen.
Variante "zwei Personen, drei Rollen"	2 out of {Signierer, Aufbewahrer}	Die verbindliche Speicherung des richtigen Ergebnisses hängt davon ab, dass die Kombination aus **Signierer** und **Aufbewahrer** ehrlich ist. Eine **Kombination** ihres Wissen kann zur Erkennung einer Manipulation führen.
Variante "drei Personen, drei Rollen"	1 out of {Aufbewahrer}, 2 out of {Signierer, Zertifikat-Ersteller}	Die verbindliche Speicherung des richtigen Ergebnisses hängt davon ab, dass der **einzige Aufbewahrer** oder die Kombination aus **Signierer und Zertifikat-Ersteller** ehrlich ist. Das Wissen des Aufbewahrers oder die **Kombination** des Wissens des Signierers und Zertifikat-Erstellers können zur Erkennung einer Manipulation führen.
Variante "n Personen, drei Rollen"	1 out of {Aufbewahrer$_1$,…,Aufbewahrer$_n$}, 2 out of {Signierer, Zertifikat-Ersteller}	Die verbindliche Speicherung des richtigen Ergebnisses hängt davon ab, dass **mindestens ein Aufbewahrer** oder die Kombination aus **Signierer und Zertifikat-Ersteller** ehrlich ist. Das Wissen eines Aufbewahrers oder die **Kombination** des Wissens des Signierers und Zertifikat-Erstellers können zur Erkennung einer Manipulation führen.

Abbildung 43 – Übersicht über das Vertrauensmodell für die Verbindlichkeit

12 Vorstellung der konzipierten IM-Plattform für sicherheitskritische Anwendungen

Durch den Zugangsberechtigungsmechanismus einschließlich der gegenseitigen Authentifikation der Teilnehmer, die Verschlüsselung der Kommunikation und die Tatsache, dass die Kommunikation in Realtime stattfindet, bleiben einem externen Angreifer nur wenige Möglichkeiten für einen Angriff ausschließlich auf die Integrität der Nachrichten.

Eine Manipulation wäre zum Beispiel dann möglich, wenn der externe Angreifer einen der Clients der Teilnehmer durch Malware verändert. Dazu könnte er beispielsweise ein modifiziertes Plug-in (das angeblich die sichere Anwendung implementiert) zum Download anbieten. Er kann nur dann einen Angriff auf die Integrität der Kommunikation ausüben, wenn die Übernahme des Clients schon *vor* der Überprüfung der Zugangsberechtigung erfolgreich ist. Er hat dann auch Zugriff auf Mechanismen und Schlüssel, welche eigentlich die Integrität gewährleisten sollen. Dieser Angriff bezieht sich über die Integrität hinaus auch auf die Authentizität des Erzeugers und wurde im Kapitel 11.4.3 erläutert.

Eine Manipulation des Clients und des Nachrichtenaustausches *nachdem* der Zugangsberechtigungsprozess samt Überprüfung der Authentizität und Verschlüsselung der Kommunikation abgeschlossen ist, ist nicht möglich. Der Angreifer müsste die gerade laufende Applikation mit seiner angepassten (manipulierten) Version ersetzen, und das in Echtzeit.

12.1 Überblick über die Systemarchitektur

Eine typische Architektur eines Instant-Messaging-Systems mit den Kommunikationskanälen für Chat und Audio-Video wurde in Kapitel 6 folgendermaßen dargestellt (vgl. Abbildung 44).

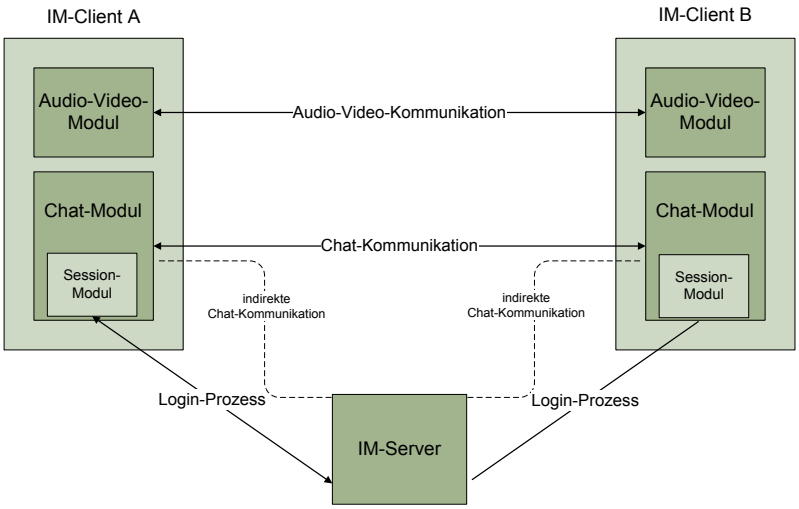

Abbildung 44 – IM-Architektur

Im Kapitel 11.2 wurden Möglichkeiten erläutert, wie eine IM-Umgebung auch für eine sichere (vertrauliche, integer, authentifizierte) Gruppenkommunikation eingesetzt werden kann. Es wurden verschiedene Alternativen vorgestellt, die, abhängig von der Anwendung, dem gewünschten Sicherheitsniveau und dem notwendigen Aufwand, ausgewählt werden können.

Wenn diese Verfahren auf Grundlage eines IM-Systems umgesetzt werden sollen, muss die oben dargestellte Architektur um zwei Komponenten erweitert werden: die eigentliche kollaborative Anwendung (vgl. Kapitel 9) und die notwendigen Sicherheitsmechanismen (vgl. Kapitel 11). Die erweiterte Architektur, ergänzt um ein Applikationsmodul (violett) sowie Sicherheitsmechanismen (orange), ist in Abbildung 45 dargestellt.

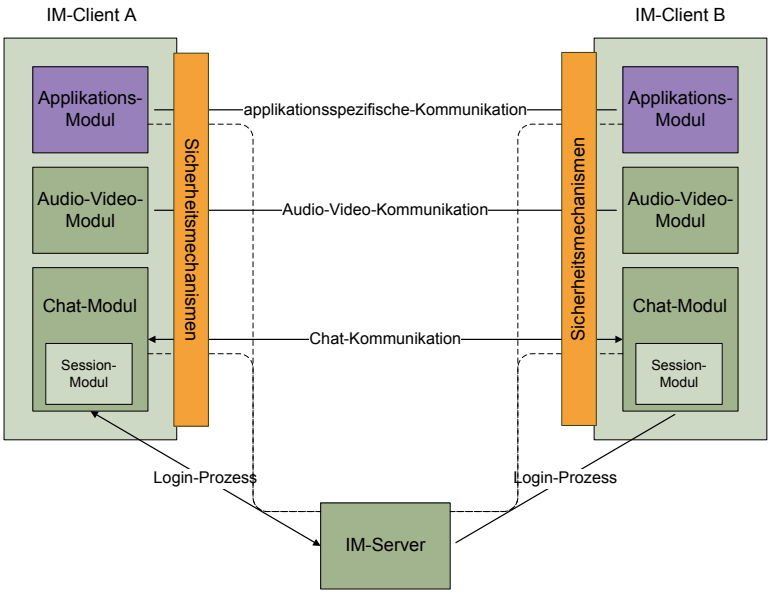

Abbildung 45 – Erweiterte IM-Architektur

Die Sicherheitsmechanismen werden beim Anwendungsstart initialisiert (z. B. werden Schlüssel generiert und ausgetauscht) und auf die übliche IM-Kommunikation angewendet (z. B. wird die Chat-Verbindung verschlüsselt).

In der später beschriebenen Referenzimplementierung (Kapitel 14) wird auch die Kommunikation, welche für die Anwendung der Sicherheitsmechanismen notwendig ist, über das Chat-Protokoll (im Prototyp ist dies XMPP) übertragen. So wird auf diesem Wege beispielsweise die Parametrisierung des Diffie-Hellman-Verfahrens abgewickelt und die Signatur- und Sitzungsschlüssel übertragen. Diese Schlüssel werden anschließend dazu verwendet, die Sicherheitsmechanismen für die applikationsspezifische Kommunikation zu realisieren (z. B. zum Signieren eines Vertragsabschlusses oder zum Verschlüsseln eines geheimen Votums in einer elektronischen Wahl). Der Nachrichtenaustausch der spezifischen kollaborativen Anwendung kann entweder nach dem Peer-to-Peer-Modell (horizontale Linien) oder über den IM-Server (gestrichelte Linie) übertragen werden. Für die Realisierung der prototypischen Anwendung (Kapitel 14) werden die spezifischen Applikationsdaten über Peer-to-Peer-Verbindungen zwischen den Gruppenmitgliedern übertragen.

Im nächsten Schritt wird nun die Architektur eines (um Anwendungsmodul und Sicherheitsmechanismen) erweiterten IM-Clients genauer betrachtet. Auf diese Weise sollen Komponenten identifiziert werden, die bei der Realisierung einer sicheren kollaborativen Anwendung über IM anzupassen wären. Gleichzeitig werden dabei Funktionen identifiziert, die eine Anwendung ihrerseits bereitstellen muss, um in eine IM-Umgebung integriert werden zu können.

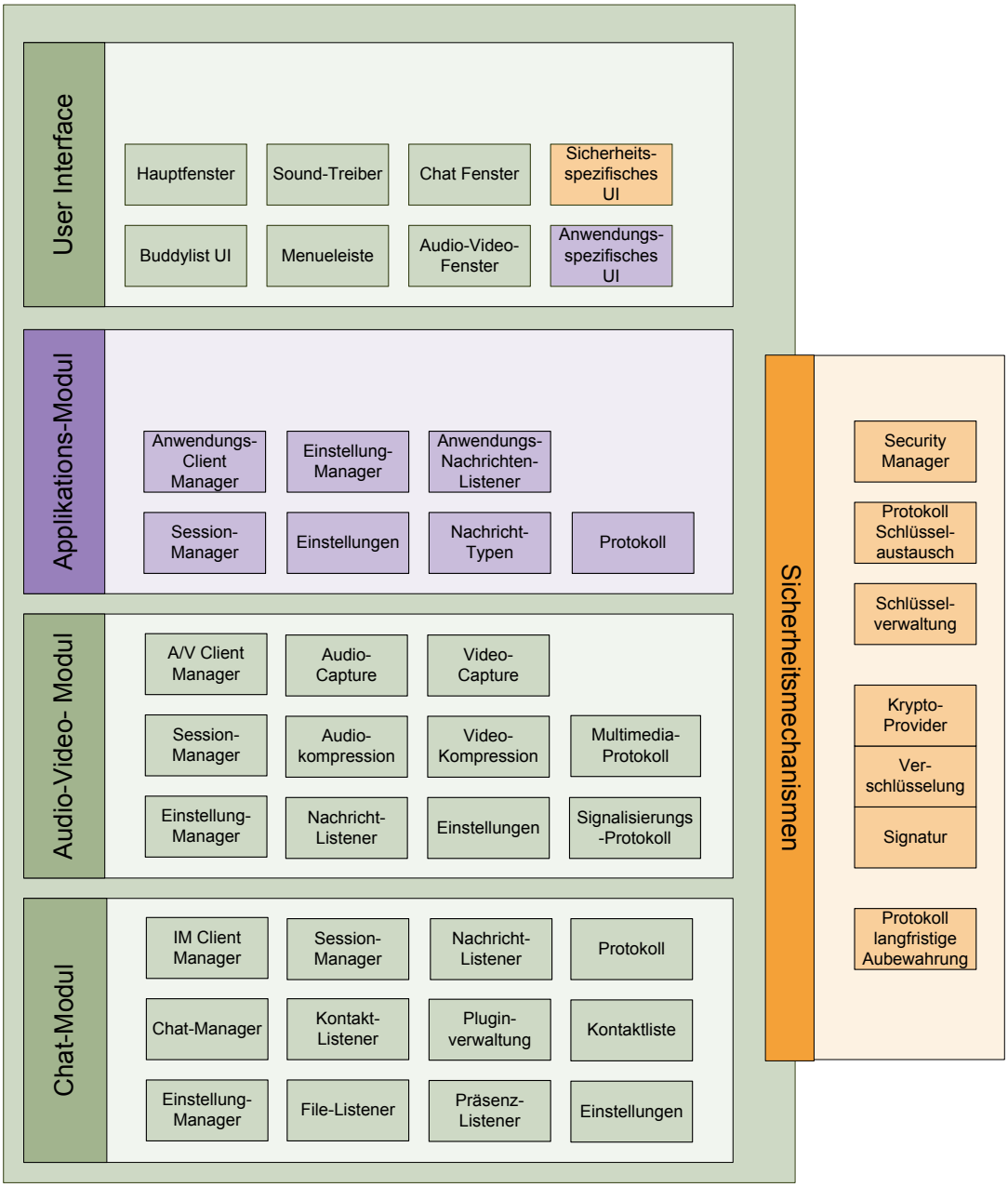

Abbildung 46 – Komponente eines erweiterten IM-Clients

Abbildung 46 ist folgendermaßen zu interpretieren: Entsprechend der vorhergehenden Diskussion (vgl. Kapitel 6.1.1) verfügt ein IM-Client über ein Chat-Modul, ein Audio-Video-Modul und über eine Benutzeroberfläche (engl. User Interface). Bei der hier betrachteten Erweiterung des Systems werden ein Applikations-Modul (violett) und zusätzliche Sicherheitsmechanismen (orange) hinzugefügt.

Die drei Module Chat, Audio-Video und Anwendung sind ähnlich aufgebaut: Jedes davon enthält eine eigene Manager-Komponente (Chat-Manager, A/V-Client-Manager und Anwendungs-Client-Manager). Jeder Manager ist eine Komponente (z. B. eine Java-Klasse), die verantwortlich ist für die Koordination der anderen Komponenten innerhalb eines Moduls. Dazu gehört beispielsweise die Initialisierung der Komponenten oder die Verteilung von Aufgaben bei eintretenden Ereignissen.

Zusätzlich gibt es noch einen IM-Client-Manager, der die Gesamtkoordination übernimmt. Dieser ist im Chat-Modul angesiedelt. Das lässt sich damit erklären, dass IM-Systeme ihren Ursprung in puren Chat-Applikationen haben und weitere Module (AV usw.) als Systemerweiterungen konzipiert werden. Der IM-Client-Manager übernimmt die Koordination aller drei Module sowie zentrale Aufgaben wie die Kontrolle des File-Listeners, des Kontakt-Listeners, die Plug-in-Verwaltung oder die Verwaltung der allgemeinen Einstellungen. Dagegen übernimmt der Chat-Manager die Chat-spezifische Kommunikation wie die Steuerung des Session-Managers für die Sitzungsverwaltung oder die Steuerung des Nachrichten-Listeners für das Empfangen von Chat-Nachrichten. Die Manager und Listener in den anderen Modulen funktionieren auf ähnliche Weise.

Neben der Strukturierung des *statischen* Aufbaus durch die Software-Architektur muss auch das *dynamische* Verhalten des Systems entworfen und festgelegt werden. Bei dem hier fokussierten Anwendungstyp liegt ein Schwerpunkt auf der Kollaboration und somit auf technischer Ebene auf der Kommunikation über Netzwerke. Daher kann das Verhalten von Anwendungen, die auf Grundlage eines IM-Systems als Plattform realisiert werden, gut durch die Beschreibung von Protokollen festgelegt werden. Beispiele für solche Protokolle sind das Chat-Protokoll, das anwendungsspezifische Protokoll, das Protokoll des Schlüsselaustausch-Verfahrens oder das Protokoll für die langfristige Aufbewahrung. Hier entscheidet sich, ob und wie eine bestimmte Anwendung in eine IM-Umgebung integriert werden kann. Durch das Protokoll wird z. B. definiert, ob die Nachrichten über Peer-to-Peer-Verbindungen oder den Server verteilt werden, wie die Adressierung verschiedener Teilnehmer erfolgt, welches Austauschformat dafür zuständig ist und welche Komponente aufgerufen wird, um die eigentliche Anwendungsfunktionalität zu realisieren. In der Komponente „Protokoll“ des Anwendungs-Moduls wird für die vorliegende Arbeit unter anderem das genaue Wahlprotokoll implementiert (vgl. Kapitel 13.5).

Für die Umsetzung einer *sicheren* Kollaboration auf Basis einer IM-Plattform ist die Auswahl der geeigneten Sicherheitsmechanismen von wesentlicher Bedeutung (vgl. Modul „Sicherheitsmechanismen“ in Abbildung 46). Diese Sicherheitsmechanismen werden durch den Security-Manager koordiniert. Dieser übernimmt die Auswahl der zu verwendenden Sicherheitsmechanismen (z. B. der Verfahren für Schlüsselerzeugung und -austausch), initiiert beim Systemstart die anderen Komponenten und übernimmt die Weiterleitung von eingehenden Nachrichten an die richtigen Komponenten zur Weiterverarbeitung. So sorgt der Sicherheitsmanager beispielsweise dafür, dass, wenn über das Netzwerk ein öffentlicher Schlüssel eines anderen Teilnehmers eintrifft, dieser entsprechend gespeichert wird, um später signierte Nachrichten dieses Teilnehmers überprüfen zu können.

Je nach angestrebtem Sicherheitsniveau und den dafür ausgewählten Mechanismen kann der Security-Manager unterschiedliche *Varianten* von Protokollen realisieren. So kann beispielsweise für einen einfachen anwendungsspezifischen Vorgang eine unverschlüsselte Übertragung gewählt werden, für einen geheimen Vorgang wird hingegen eine bestimmte Verschlüsselung gewählt und entsprechende Nachrichtentypen übertragen. Für eine flexible Nutzung der hier vorgestellten Anwendungsplattform ist es hilfreich, wenn die Konfiguration der verwendeten Sicherheitsmechanismen und Protokolle nicht fest in der Implementation des Security-Managers kodiert wird, sondern über eine Konfigurationsdatei geladen und somit je nach Szenario ausgetauscht werden kann.

Die zuvor beschriebene Aufgabenverteilung ist in einer idealisierten Architektur zu finden, bei der die implementierten Mechanismen streng nach logisch-funktionaler Zuordnung organisiert sind. In einer konkreten Implementation kann es aber sein, dass die genannten Aufgaben an sehr unterschiedlicher Stelle von vielen verschiedenen Komponenten realisiert werden, ohne dass eine separate koordinierende Instanz verwendet wird.

Der erste hier relevante Sicherheitsmechanismus ist die Implementierung des Schlüsselaustauschverfahrens und die Überprüfung der Echtheit der Identitäten und der Schlüssel. Für die in dieser Arbeit fokussierten Gruppenkollaborationen wurde das Diffie-Hellman-Verfahren ausgewählt, da dies die sichere Aushandlung von Schlüsseln über ein unsicheres Medium erlaubt, ohne dass dabei der eigentliche Schlüssel übertragen wird (vgl. auch Kapitel 11.1.2, 11.2.5). Da das vorgestellte Konzept (vgl. Kapitel 11.3) nur dann durchgeführt werden kann, wenn sich die Kollaborationspartner kennen, könnte an dieser Stelle auch ein anderes Schlüsselaustauschverfahren für weitere Anwendungen gewählt werden.

Die zweite Komponente ist die clientseitige Schlüsselverwaltung. Hier wird bestimmt, wie viele Schlüssel verwendet und wie diese gespeichert, verteilt und aktualisiert werden. Die dritte Komponente ist der Krypto-Provider (vgl. Abbildung 46). Diese Komponente ist ein separates Framework, welches die Implementationen von Kryptoalgorithmen (auch für die Berechnung des Diffie-Hellman-Schlüssels) bereitstellt. Über diese Komponente werden die Verfahren für die Verschlüsselung und Signierung der Daten gewählt und realisiert. Dies kann sowohl unter Einsatz asymmetrischer als auch symmetrischer Verfahren erfolgen. Die letzte Komponente ist das Protokoll für die langfristige verbindliche Speicherung des Ergebnisses (vgl. Kapitel 11.10.). Diese Komponente realisiert das vorgestellte Verfahren unter Verwendung der Verschlüsselungsmechanismen und der Signierungsverfahren, die von den vorherigen Komponenten (z. B. dem Krypto-Provider) bereitgestellt werden. Weiterhin definiert dieses Protokoll die Reihenfolge des Nachrichtenaustausches zwischen den unterschiedlichen Rollen und zugehörigen Aufgaben.

Der letzte Bestandteil der Plattform ist die Benutzeroberfläche (engl. User Interface). Hier müssen zusätzlich zu den üblichen Komponenten eines IM-Systems (wie Chat-Fenster, Audio-Video-Fenster, Menüleiste, usw.) auch Interaktionselemente für die Erweiterungen (die spezifische Anwendung und die Sicherheitsmechanismen) realisiert werden. So werden zum Beispiel nach der Durchführung des Diffie-Hellman-Verfahrens die vereinbarten Schlüssel durch *Anzeige* von Hashwerten und Vorlesen in einem Video*fenster* überprüft (vgl. Kapitel 11.3 und Kapitel 14). Zusätzlich können weitere Sicherheitselemente in die Benutzeroberfläche aufgenommen werden. Wenn zum Beispiel das Versenden oder der Empfang von RSA-Schlüsseln für die Signierung von Nachrichten (Kapitel 14) auch in der Benutzeroberfläche angezeigt wird, trägt dies zum besseren Verständnis bei, beeinflusst aber nicht unbedingt den sicheren Kommunikationsablauf.

12.2 Kommunikation zwischen den Systemkomponenten

Nachfolgend soll die prinzipielle Kommunikation zwischen den Komponenten der Plattform (bei der Umsetzung der vorher beschriebenen algorithmischen Lösungen) erläutert werden. Dafür werden die Komponenten innerhalb des Clients B separat dargestellt, der Client A hinge-

gen als zusammenhängendes System (vgl. Abbildung 47). B erhält von A eine Anfrage, die ihn auffordert, das Diffie-Hellman-Key-Agreement durchzuführen. Hier nimmt der Chat-Manager die Anfrage entgegen und leitet sie an den Security-Manager weiter. Dieser generiert (mit den von A gesendeten Parametern q, p und P_A) mit Hilfe des Krypto-Providers den privaten und öffentlichen Schlüssel des Clients B für das DH-Verfahren. Außerdem generiert er den Signaturschlüssel – entweder zufällig (bei RSA) oder abgeleitet aus den DH-Parametern (bei DSA), je nachdem welches Verfahren verwendet wird.

1.) Sind die entsprechenden Schlüssel von B für die Verschlüsselung und Signatur berechnet, werden sie dem Chat-Manager übergeben und von dort an A weitergeleitet (vgl. Abbildung 47, 1. Generiere Schlüssel und Signatur).

2.) Nach der Berechnung des gemeinsamen DH-Schlüssels K_{AB} durch beide Seiten (diese Aufgabe übernimmt wieder der Security-Manager in Zusammenarbeit mit dem Krypto-Provider) wird eine verschlüsselte Verbindung zwischen den Chat-Managern beider Clients hergestellt (vgl. Abbildung 47, 2. Öffnen der verschlüsselten Chat-Verbindung).

3.) Jeder Client berechnet über den Security-Manager den Hashwert des bilateralen Schlüssels K_{AB} und fordert den AV-Client-Manager auf, ein Fenster für die Video-Überprüfung des Schlüssels zu öffnen. Damit die Überprüfung durchgeführt werden kann, übergibt der Chat-Manager den symmetrischen Schlüssel K_{AB} dem AV-Client-Manager, der dann eine verschlüsselte Audio-Video-Verbindung aufbaut (vgl. Abbildung 47, 3. Öffnen der verschlüsselten AV-Verbindung).

4.) Im nächsten Schritt wird durch einen AV-Client-Manager (je nach Initiative von A oder B) die Anfrage für den Aufbau einer AV-Verbindung geschickt. Über diese Verbindung wird dann der Hashwert vorgelesen (vgl. Abbildung 47, 4. Vorlesen Hashwert in AV-Verbindung).

5.) Nach diesem Schritt kann jetzt die jeweilige realisierte Gruppenapplikation gestartet werden. Diese Initiierung übernimmt der Chat-Manager mit der Anweisung „Anwendung öffnen“. Bevor jedoch der eigentliche anwendungsspezifische Nachrichtenaustausch zwischen Client A und B erfolgt, werden vom Chat-Manager noch die generierten und ausgetauschten Schlüssel für die Verschlüsselung und Signatur übergeben (vgl. Abbildung 47, 5. Sicherheitskritische Anwendung).

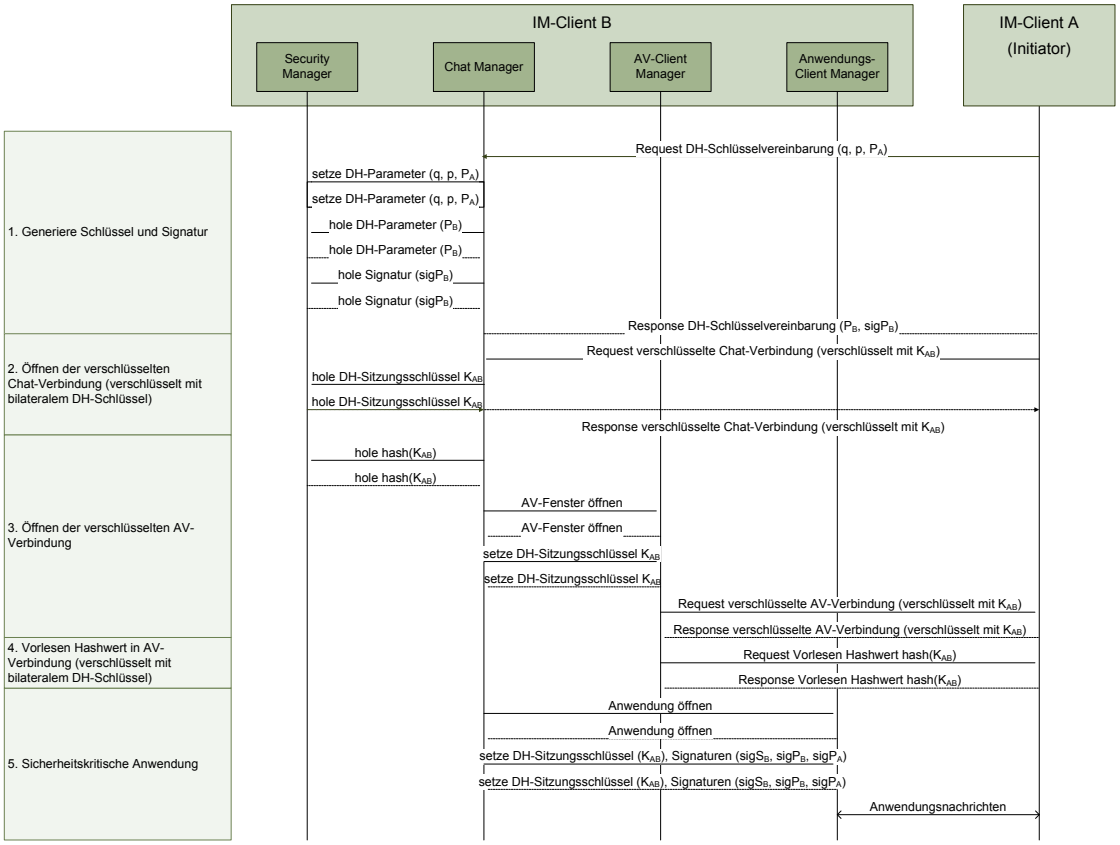

Abbildung 47 – Sequenzdiagramm einer bilateralen Kommunikation

Über diese grundlegenden Schritte hinaus ist auch noch das Protokoll für die Verbindlichkeit zu realisieren. Dafür werden, wie im Kapitel 11.11 diskutiert drei Rollen (Signierer, Aufbewahrer, Zertifikat-Ersteller) vergeben Bei einer Gruppenanwendung, in die nur zwei Partner involviert sind, müssen diese Rollen auf die beiden Teilnehmer verteilt werden. Dies bedeutet zwangsläufig, dass ein Teilnehmer zwei Rollen übernimmt. Um dabei eine sichere langfristige Aufbewahrung gewährleisten zu können, müssen die Rollen des Zertifikat-Erstellers und des Signierers von der einen Person und die Rolle des Aufbewahrers von der anderen übernommen werden. Eine andere Verteilung der Rollen würde die sichere Aufbewahrung des Ergebnisses aufgrund der möglichen Kollaborationen, wie im Kapitel 11.10 erläutert wurde, gefährden. Zur besseren Übersicht wird das weitere Vorgehen bei der Realisierung der Verbindlichkeit im Anschluss an die folgende Abbildung mit drei Teilnehmern erläutert.

Der folgende Ablauf zwischen drei Teilnehmern (hier A, B und C) unterscheidet sich vom Vorgehen für zwei Teilnehmer (vgl. Abbildung 47), dadurch, dass erstens zusätzlich zum paarweisen Aushandeln der DH-Schlüssel (z. B. K_{AB} und K_{AC}) über diese paarweisen DH-Schlüssel auch gemeinsame Schlüssel für den Nachrichtenaustausch zwischen allen Teilnehmern (K_{ALL}) vereinbart werden und zweitens die Rollenverteilung für die Verbindlichkeit der Ergebnisse festgelegt wird.

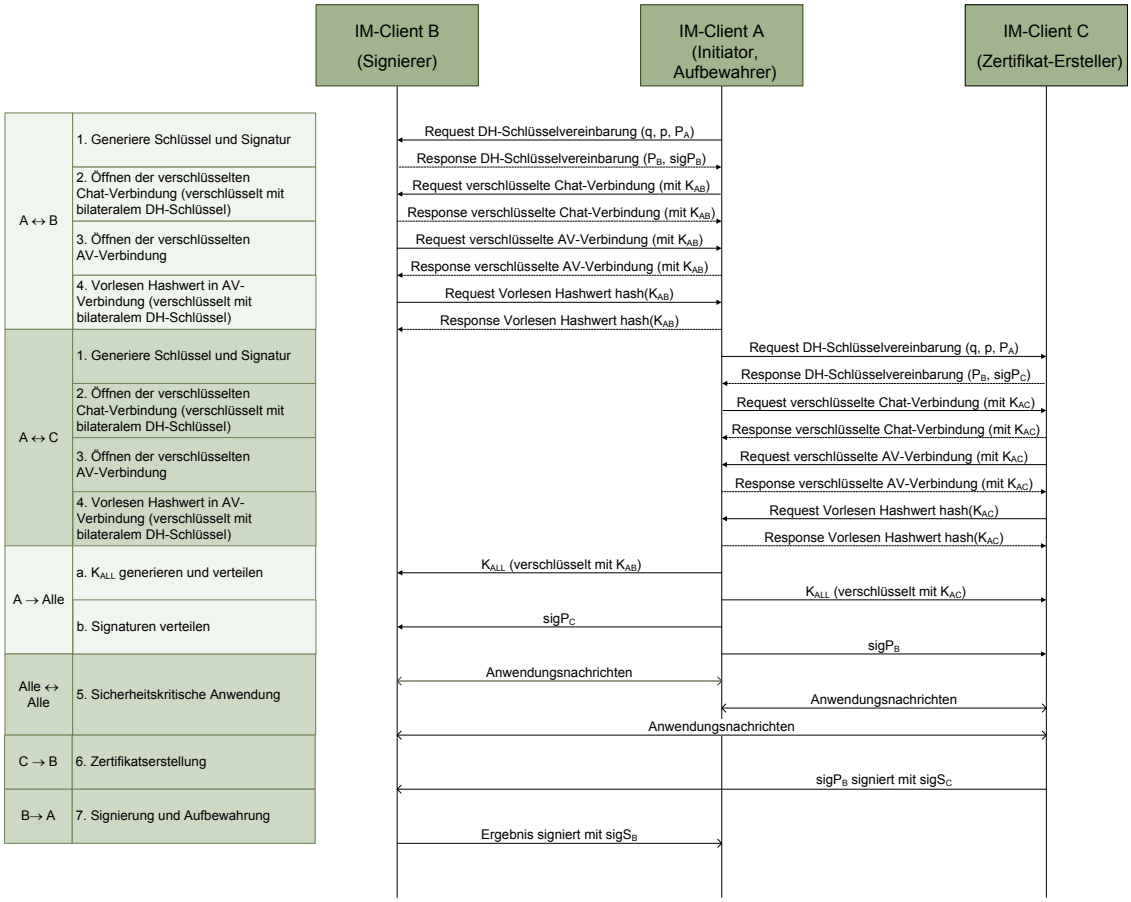

Abbildung 48 – Sequenzdiagramm einer Gruppenkommunikation

a.) Nachdem der Initiator das oben beschriebene Prozedere mit jedem der Teilnehmer durchgeführt hat (vgl. Abbildung 48, A↔B, A↔C), generiert er mit Hilfe seines Security-Managers einen symmetrischen Schlüssel K_{ALL} und verteilt diesen über die bilateralen Verbindungen an die Clients B und C.

b.) Danach leitet der Initiator auch die Signaturschlüssel der verschiedenen Teilnehmer weiter. Diese Schlüssel werden später dazu verwendet, die Integrität der Kommunikation und im letzten Schritt die Verbindlichkeit der Ergebnisse sicherzustellen. Dafür wird während der Kommunikation der Zertifikat-Ersteller definiert. Im Beispiel (vgl. Abbildung 48) ist dies Client C, welcher mit Hilfe seines Chat-Managers, Security-Managers und Krypto-Providers den Signaturschlüssel des Signierers (im Beispiel Client B) zertifiziert. Der Zertifikat-Ersteller C leitet dieses Zertifikat an den Signierer B weiter. Dieser unterschreibt damit das Ergebnis der Kommunikation und sendet es an den Aufbewahrer (im Beispiel A) weiter, der das Ergebnis und das zugehörige Zertifikat speichert.

Je nach Anwendung und gefordertem Sicherheitsniveau kann auch ein weiterer Aufbewahrer bestimmt werden oder der Zertifikat-Ersteller kann das Zertifikat zusätzlich speichern, um zu einem späteren Zeitpunkt die Aussage des Aufbewahrers überprüfen zu können. Man vergleiche hierzu die Diskussion in Kapitel 11.11.

13 Exemplarische Anwendung: Entscheidungsprozesse und Wahlen

In diesem Kapitel der Arbeit wird eine konkrete Anwendung als Paradigma für die Integration einer kollaborativen Anwendung in eine Instant-Messaging-Umgebung gewählt. Diese Anwendung ist ein Entscheidungsprozess von Gremien oder Ausschüssen, die gemeinsam eine Thematik besprechen und über Alternativen abstimmen. Anhand dieser Anwendung wird belegt, dass IM-Systeme nicht nur für eine informelle Kommunikation geeignet sind, sondern dass diese Plattform unter Berücksichtigung von passenden Sicherheitsmechanismen auch geeignet ist, sichere Kollaborationen zu realisieren.

Als Erstes werden die Entscheidungsprozesse vor Ort in Gremiengruppen vorgestellt und die aktuell realisierten Sicherheitsanforderungen identifiziert. Es handelt sich dabei um Entscheidungsprozesse in kleinen Gruppen (weniger als 20 Personen) mit der Voraussetzung, dass sich die meisten Teilnehmer untereinander kennen. Solche Anwendungsfälle kommen zum Beispiel im universitären Bereich (Berufungskommissionen, Besetzung der Vertreter in einem Gremium) oder in Unternehmen (Entscheidungen über die Vorgehensweise in einem Projekt) vor. Dabei steht die Definition der Sicherheitsanforderungen im Mittelpunkt.

Im Anschluss daran wird der Ablauf der obigen Entscheidungsprozesse auf den Einsatz in einer verteilten Umgebung übertragen, in der sich die Teilnehmer in unterschiedlichen Kontrollbereichen befinden (z. B. sie gehören unterschiedlichen Unternehmen an, oder sie gehören zwar der gleichen Institution an, befinden sich aber temporär in einer anderen Umgebung, z. B. an einem Arbeitsplatz zu Hause). Dafür wird der Prozessablauf der vollständigen Kommunikation dargestellt und die Schnittstellen zu den Sicherheitskonzepten einer sicheren IM-Kommunikation aus dem Kapitel 13.1 definiert.

Als Vorbereitungsschritt für eine prototypische Entwicklung wird am Ende dieses Kapitels das verwendete E-Voting-Protokoll samt der notwendigen Modifikationen z. B., um die spontan (kurzzeitig) generierten Sicherheitselemente aus der sicheren IM-Kommunikation für die Realisierung der Sicherheitsanforderungen der elektronischen Wahl zu verwenden.

13.1 Elektronische Entscheidungsprozesse

In diesem Kapitel werden drei Szenarien für Entscheidungsprozesse in kleineren Gruppen vorgestellt, wie sie in der Realität vor Ort durchgeführt werden. Jeder Prozess besteht jeweils aus einer Diskussionsphase und/oder einer anschließenden (Gruppen-)Abstimmung (= Wahl) realisiert durch eine Wahl. Sie unterscheiden sich in den Regeln, die für die Wahl oder Diskussion gelten. So kann eine Wahl beispielsweise geheim oder offen durchgeführt werden. Aus diesen Regeln ergeben sich unterschiedliche Sicherheitsziele (z. B. Anonymität) und entsprechende Anforderungen, die auch in einer technischen Umsetzung zu beachten sind. Darüber hinaus gilt, dass sich die Wähler in ihrer Mehrheit untereinander kennen.

An dieser Stelle muss erwähnt werden, dass die Festlegung der Wahlberechtigten (z. B. Erstellung eines Wählerverzeichnisses oder der Teilnehmerliste eines Gremiums) ein Prozess ist, der entweder einen separaten Entscheidungsprozess darstellt oder bei dem die Berechtigung/Teilnahme aufgrund einer Nominierung oder entsprechender Regelungen erfolgt. Für die

vorliegende Arbeit wird die Vorgabe einer Teilnehmerliste als gegeben betrachtet. Jeder Teilnehmer kennt im Voraus die Größe der Gruppe und mindestens die Namen (oder sogar die Identität) der Beteiligten. Eine Vorstellung der Szenarien findet sich auch in (Meletiadou 2008).

13.1.1 Szenario 1: keine Diskussion, geheime Abstimmung

Das erste Szenario beschreibt eine Wahl ohne Diskussion mit geheimer Abstimmung. Eine solche Wahl wird beispielsweise bei der Besetzung von universitären Gremien durchgeführt. Die Wahl ist als Urnenwahl mit personalisierter Verhältniswahl (wenn für eine Gruppe mehrere zugelassene Wahlvorschläge vorliegen und die Zahl der Vorgeschlagenen insgesamt über der Zahl der zu wählenden Mitglieder liegt) oder Mehrheitswahl (wenn nur ein oder kein Wahlvorschlag vorliegt oder die Zahl der Vorgeschlagenen insgesamt unter der Zahl der zu wählenden Mitglieder liegt) durchzuführen. Es wird je nach Institution auch eine Briefwahl angeboten. Die Aufgaben der Wahlkommission (bzw. des Wahlvorstands) sind dabei, die Vorschläge zu beschließen, die Stimmabgabe zu leiten und das Ergebnis festzustellen. Solche Wahlen finden vor Ort statt und werden von einer Wahlkommission oder einem Wahlvorstand organisiert und durchgeführt. Alle Stimmen werden gesammelt und ausgezählt; das Ergebnis wird veröffentlicht. Für eine solche Wahl gelten bestimmte Regeln. So wird zum Beispiel durch das Hochschulgesetz und die Grundordnung der Universität Koblenz-Landau für die Wahl des Senats oder eines Fachbereichsrats festgelegt, dass diese als freie, gleiche und geheime Wahl organisiert werden muss (Land Rheinland-Pfalz 2003; Universität Koblenz-Landau 2006, 2008). Das Kriterium der freien Wahl ist in diesem Zusammenhang durch die Existenz der Wahlkommission gegeben, die dafür sorgen muss, dass die Wähler unbeeinflusst ihre Stimme abgeben können. Gleiche Wahl bedeutet, dass alle Wahlberechtigten die gleiche Chance erhalten, ihre Stimme abzugeben, und dass jede Stimme das Ergebnis mit dem gleichen Gewicht beeinflusst. Geheime Wahl heißt, dass einem Wähler nicht nachgewiesen werden kann, welchem Kandidaten er seine Stimme gegeben hat, und dass zu keinem Zeitpunkt ein Zusammenhang zwischen seinem Namen und seiner Stimme besteht. Weiterhin wird durch diese Regularien (Wahlordnungen) definiert, wer wahlberechtigt ist, mit wie vielen Mitgliedern jedes Gremium besetzt wird und wann ein Beschluss rechtsverbindlich ist. Aus diesen rechtlichen Rahmenbedingungen ergeben sich entsprechende organisatorische Abläufe und Sicherheitsziele, die folgendermaßen in der Präsenzwahl realisiert werden:

Die *Festlegung der Wahlberechtigten* wird bisher durch die Wahlkommission gemäß den entsprechenden Regularien festgelegt.

Die *Authentizität* der Wahlberechtigten beim Wahlvorgang wird durch das Vorzeigen eines Ausweises und den Abgleich mit dem Wählerverzeichnis überprüft.

Die *Anonymität* ist durch das Vertrauensmodell der Wahlkommission gegeben. Die Durchführung einer solchen Wahl läuft ähnlich wie bei einer politischen Wahl ab. Wähler, die im Wählerverzeichnis registriert sind, erhalten einen Stimmzettel, den sie in einer Wahlkabine unter Ausschluss der Öffentlichkeit ausfüllen (*Vertraulichkeit*). Der Stimmzettel darf keine Informationen enthalten, die eine eventuelle Zuordnung von Stimme und Wähler erlauben. Zum Schluss wirft der Wähler seinen Stimmzettel in eine Urne. Die Urne wiederum muss vor dem Start der

Wahl leer sein und darüber hinaus ein einwandfreies Mischen der Stimmen gewährleisten, um das Wahlgeheimnis der Wähler zu schützen.

Die *Nicht-Abstreitbarkeit* kann in zwei Formen auftreten. Erstens, wenn der Wählende nicht abstreiten kann, *dass* er gewählt hat (Teilnahme nachweisbar), und zweitens, wenn der Wählende nicht abstreiten kann, *wie* er gewählt hat (Stimme nachweisbar). Die erstere Form ist zwar eventuell durch die Beobachtung der Öffentlichkeit und ein „Abhaken" im Wählerverzeichnis gegeben. Die zweite Form ist in diesem Szenario (geheime Wahl, Quittungsfreiheit) aber ausdrücklich auszuschließen (*Verfolgbarkeit*).

Eine dritte Eigenschaft ist die *Integrität* des Wahlergebnisses. Wenn ein Wähler die korrekte Erfassung der Stimmen anzweifelt, kann er, basierend auf dem Öffentlichkeitsprinzip, die Wahl mitverfolgen, indem er die Wahlprozedur vom Aufbau der Wahlurne bis zur Auszählung der Stimmzettel beobachtet. Darüber hinaus besteht die Möglichkeit der Anfechtung der Wahl. Dazu müssen genügend Beweise vorliegen, dass während der Vorbereitungsphase, der Wahlhandlung oder der Ermittlung der Ergebnisse die Wahl so manipuliert wurde, dass dies zu einem verfälschten Ergebnis geführt hat. In diesem Fall wird ein weiterer Ausschuss (Wahlprüfungsausschuss) die Beweise überprüfen und einen neuen Wahlvorgang anordnen.

Die *Ergebnisermittlung* erfolgt durch die Auszählung der anonymen Stimmzettel und Veröffentlichung durch die Wahlkommission.

13.1.2 Szenario 2: keine Diskussion, offene Abstimmung

Eine offene Abstimmung ohne vorherige Diskussion ist eine eher seltene Form eines Entscheidungsprozesses. Diese findet statt, wenn die Wahlalternativen durch einen vorherigen Entscheidungsprozess, der eventuell unterbrochen wurde, definiert wurden. Dieses Szenario entspricht einer speziellen Form des folgenden Falles einer offenen Diskussion mit offener Abstimmung. Da die gestellten Anforderungen bzgl. der Abstimmung gleich sind, wird an dieser Stelle auf das nächste Kapitel verwiesen.

13.1.3 Szenario 3: offene Diskussion, offene Abstimmung

Eine weitere Form eines Entscheidungsprozesses ist eine offene Abstimmung, welche im Anschluss an eine offene Diskussion folgt. Dies bedeutet, dass sowohl die Diskussion als auch die Abstimmung unter Angabe der Identität erfolgt. Ein Universitätsgremium könnte auf einer Sitzung z. B. über die Erweiterung des Curriculums um einen weiteren Studiengang abstimmen wollen. Die Mitglieder diskutieren zunächst über die denkbaren Möglichkeiten. Nach einer Diskussion und Bewertung der Alternativen mit ihren Vor- und Nachteilen erfolgt die eigentliche Abstimmung. Die hier geltenden organisatorischen Abläufe und Sicherheitsziele sind folgendermaßen definiert:

Ebenso, wie im vorigen Szenario, erfolgt die Festlegung der Stimmberechtigten zu einem früheren Zeitpunkt. Da solche Entscheidungsprozesse innerhalb einer Kommission oder eines Gremiums stattfinden, haben die Stimmberechtigten durch einen anderen Abstimmungsprozess (wie im Szenario 1) ihr Stimmrecht erhalten. Dies ist auch allen anderen Mitgliedern bekannt durch das Veröffentlichen der Ergebnisse des vorherigen vorangegangenen Wahlprozederes.

Die Authentifikation der Wahlberechtigten erfolgt in der Regel aufgrund persönlicher Bekanntschaft („man kennt sich“) ohne Vorlage von Ausweisen oder anderer technischer Identifikationsmerkmale und aufgrund der Stimmberechtigtenliste (Anwesenheitsliste). In diesem Szenario ist keine Anonymität oder Vertraulichkeit erforderlich sowohl die Phase der Diskussion als auch die Phase der Abstimmung finden namentlich und offen für alle Mitglieder statt, z. B. durch einfaches Handzeichen. Hier muss erwähnt werden, dass der Stimmzettel (mit den zur Wahl stehenden Alternativen) ein Ergebnis der Diskussionsphase ist. An dieser Stelle ist die Nicht-Abstreitbarkeit der Teilnahme, aber auch der Stimmen durch die Anwesenheit und Zeugenaussage der anderen Mitglieder gegeben. Sie können ein bestimmtes Verhalten bezeugen.

Das Ergebnis kann von jedem Anwesenden dadurch nachvollzogen werden, dadurch dass jeder selbst die Handzeichen zählen kann. Somit wird auch die Integrität des Ergebnisses erreicht. Die Ergebnisermittlung erfolgt durch die Bekanntgabe des Ergebnisses, und das, je nach Thema der Abstimmung, wird dies auch eventuell auch veröffentlicht wird. Weiterhin werden die Abstimmungsschritte und das Ergebnis in Form eines Protokolls während einer Sitzung festgehalten (Land Rheinland-Pfalz 2003; Universität Koblenz-Landau 2006).

13.1.4 Szenario 4: offene Diskussion, geheime Abstimmung

Eine Kombination des ersten und dritten Szenarios ist ein Entscheidungsprozess, der aus einer offenen Diskussion und einer geheimen Wahl besteht. Ein solcher Prozess kann beispielsweise bei Personalentscheidungen in entsprechenden Gremien und Ausschüssen einer Universität auftreten. Die Teilnehmer unterhalten sich zum Beispiel zunächst offen über die Besetzung der Stelle. Dann erfolgt in einer geheimen Abstimmung die Entscheidung. Der organisatorische Ablauf und die Sicherheitsziele entsprechen demnach einer Zusammenstellung der in Kapitel 13.1.1 und 13.1.3 genannten Punkte.

Die Festlegung der Stimm- und Wahlberechtigten resultiert aus der Mitgliedschaft einer Person in dem entsprechenden Gremium. Die Authentifikation der Wahlberechtigten basiert wie im vorherigen Szenario auf der sozialen Interaktion der Teilnehmer. In Bezug auf die Anonymität gilt, dass während der Diskussionsphase keine Anonymität erwünscht ist, wohingegen diese in der zweiten Phase der Abstimmung ausdrücklich gewünscht ist.

Die Nicht-Abstreitbarkeit der Teilnahme ist durch die Anwesenheit und Zeugenaussage der anderen Mitglieder gegeben. Die Nicht-Abstreitbarkeit der eigenen Stimme wird ausdrücklich nicht durchgesetzt, im Gegenteil: Die Wahl ist anonym (s. o.). Allerdings kann durch die vorangegangene Diskussion eine Tendenz der Teilnehmer erkennbar sein.

Für die Nachweisbarkeit der Integrität der Ergebnisse gelten ähnliche Aussagen wie im ersten Szenario (Kapitel 13.1.1). Die Teilnehmer können ihre Stimme geheim auf einem Stimmzettel abgeben (ähnlich wie bei der Urnenwahl). Für die Ergebnisermittlung gibt es zwei Varianten: In der ersten werden die Ergebnisse an eine ausgewählte Instanz weitergereicht (Vorsitzender), die die Stimmen auswertet und das Ergebnis verkündet. In der zweiten Variante werden die Ergebnisse offen ausgezählt.

Zweifelt ein Teilnehmer das Ergebnis an, führt dies zu einer Nachzählung. Hier ist der Vorgang weniger formell und sofort überprüfbar. Das Ergebnis wird in Form eines Protokolls während einer Sitzung festgehalten (Land Rheinland-Pfalz 2003; Universität Koblenz-Landau 2006).

Darüber hinaus gilt für alle Wahlformen, dass die Ergebnisse langfristig aufbewahrt werden müssen. Die Dauer ist abhängig von den jeweiligen gesetzlichen oder betrieblichen Regelungen und kann zwischen zwei und sechs Jahre betragen (Langer und Opitz-Talidou 2009).

13.2 Entscheidungsszenarien basierend auf der Instant-Messaging-Technologie

Gegenstand dieses Kapitels ist die Anpassung eines Entscheidungsprozesses an den Prozessablauf einer Kommunikation über ein Instant-Messaging-System. Nachdem im Kapitel 11 Sicherheitsmechanismen für eine spontane (kurzzeitige) sichere Kommunikation konzipiert wurden, geht es in diesem Kapitel um die Einbettung/Realisierung dieser Anwendung in einer IM-Plattform, und zwar so, dass die Sicherheitsanforderungen aus dem Kapitel 13.1 bestehen bleiben. Ein erster Entwurf des Gesamtprozederes wurde in (Meletiadou und Grimm 2009) beschreiben.

13.2.1 Systemablauf eines Entscheidungsprozesses

Betracht man einen sicheren vollständigen Kommunikations-/Kollaborationsprozess über ein Instant Messaging-System, wie er in Kapitel Algorithmische/Konzeptuelle Lösungsansätze vorgestellt wurde, werden nach der Installation der entsprechenden Lösung folgende Schritte identifiziert (vgl. Abbildung 49):

- Zutritt in die Gruppe – Dazu gehören konkret die Zugangsberechtigung (Registrierung und Einloggen) zu dem IM-System, die sichere Authentifizierung und der Eintritt in die Gruppe, wie in Kapitel 11.3. beschrieben.
- Interaktion der Teilnehmer – Dies entspricht der Kommunikation und Kooperation zwischen den Teilnehmern und ist abhängig von der jeweiligen realisierten Anwendung. In dem Fall eines Entscheidungsprozesses bilden die Diskussion und elektronische Wahl diese Interaktion (Kommunikation) zwischen den Teilnehmern.
- Aufbewahrung der Ergebnisse – Das Ergebnis der Kollaboration soll in einem weiteren Schritt langfristig aufbewahrt werden, um die Verbindlichkeit des Ergebnisses Dritten gegenüber zu beweisen (vgl. Kapitel 11.11)

Diese Unterteilung wurde vorgenommen, da der erste und letzte Schritt für jede denkbare Anwendung, die auf Basis eines IM-Systems erfolgt, gleichwertig realisiert werden kann. Der zweite Schritt (die Interaktion der Teilnehmer) ist dagegen anwendungsspezifisch. So ist es beispielsweise möglich, (i) die ganze Kommunikation (alle drei Schritte) über einen zentralen Server abzuwickeln oder (ii) die Registrierung und Authentifikation über einen zentralen Server und die Interaktion zwischen den Teilnehmern direkt (Peer-to-Peer) oder über unterschiedliche Server (Hop-by-hop) zu realisieren.

Für den vorliegenden Anwendungsfall der Entscheidungsprozesse folgt auf die erfolgreiche Installation des IM-Clients die Überprüfung der Zugangsberechtigung (vgl. Abbildung 49), so wie im Kapitel 11.1.3 erläutert wurde. In dieser Phase erfolgt die Interaktion/Kommunikation teilweise über den Server und teilweise als eine Peer-to-Peer-Kommunikation. Der Benutzer wird erstens anhand seiner Kennung vom IM-System authentifiziert und zweitens durch den

Initiator der Gruppenkommunikation oder einen anderen Teilnehmer. In der Phase der Interaktion während des Entscheidungsprozesses wird je nach implementiertem Szenario eine Diskussion durchgeführt. Diese Phase ist nach dem Peer-to-Peer-Prinzip realisiert. Dafür stehen sowohl der Chat-als auch der Audio-Video-Kanal zur Verfügung. In dieser Phase wird auch der Stimmzettel für die nachfolgende Abstimmung erzeugt und verteilt.

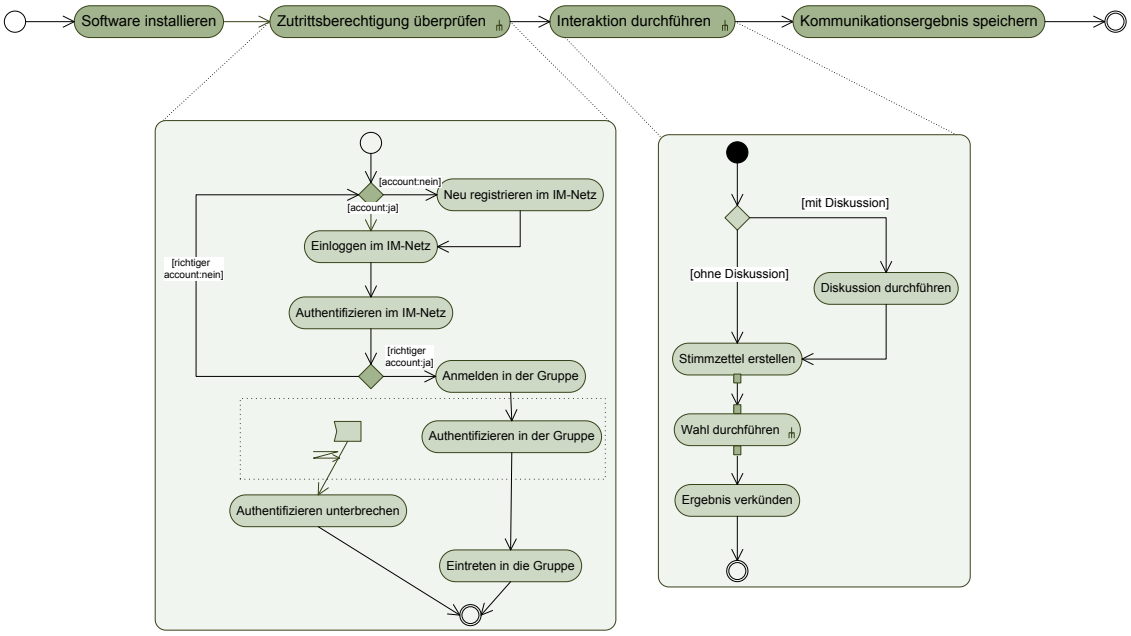

Abbildung 49 – Darstellung des Systemablaufs

In der Phase der Wahl wird ein Wahlprotokoll für elektronische Wahlen implementiert. Zum Schluss wird das Ergebnis der Abstimmung, z. B. ein Bericht, durch den in Kapitel 11.11 beschriebenen Mechanismus langfristig gespeichert.

13.2.2 Sicherheitsanforderungen der Entscheidungsprozesse

Wird ein Entscheidungsprozess in eine IM-Umgebung integriert, müssen die in den Kapiteln 13.1.1, 13.1.3, 13.1.4 vorgestellten Sicherheitsanforderungen weiterhin garantiert werden.

Um die richtigen Mechanismen für die Gewährleistung dieser Anforderungen zu realisieren und um zu zeigen, dass die in Kapitel 11 dargestellten Mechanismen genau diese Anforderungen erfüllen, wird in der folgenden Übersicht (Abbildung 50) eine Bedrohungsanalyse präsentiert. Diese identifiziert die Bedrohungen der Entscheidungsprozesse und setzt sie mit denjenigen Anforderungen in Bezug, welche dadurch verletzt werden.

Es wird zwischen den folgenden Teilprozessen einer Entscheidung unterschieden: der Diskussion – unabhängig vom Kommunikationskanal – und der Abstimmung. Ein weiteres Unterscheidungsmerkmal ist die Art der übertragenen Daten (Steuerdaten und Sitzungsdaten). Die Steuerdaten sind für den Kommunikationsaufbau und -abbau zuständig, während Sitzungsdaten die Inhalte der jeweiligen Kommunikation enthalten. Die Angriffe, des Teilbaums „Diskussion“ sind typische Angriffe auf die IM-Kommunikation. Beispiele für solche Angriffe wurden im Kapitel 7.2.2 gegeben.

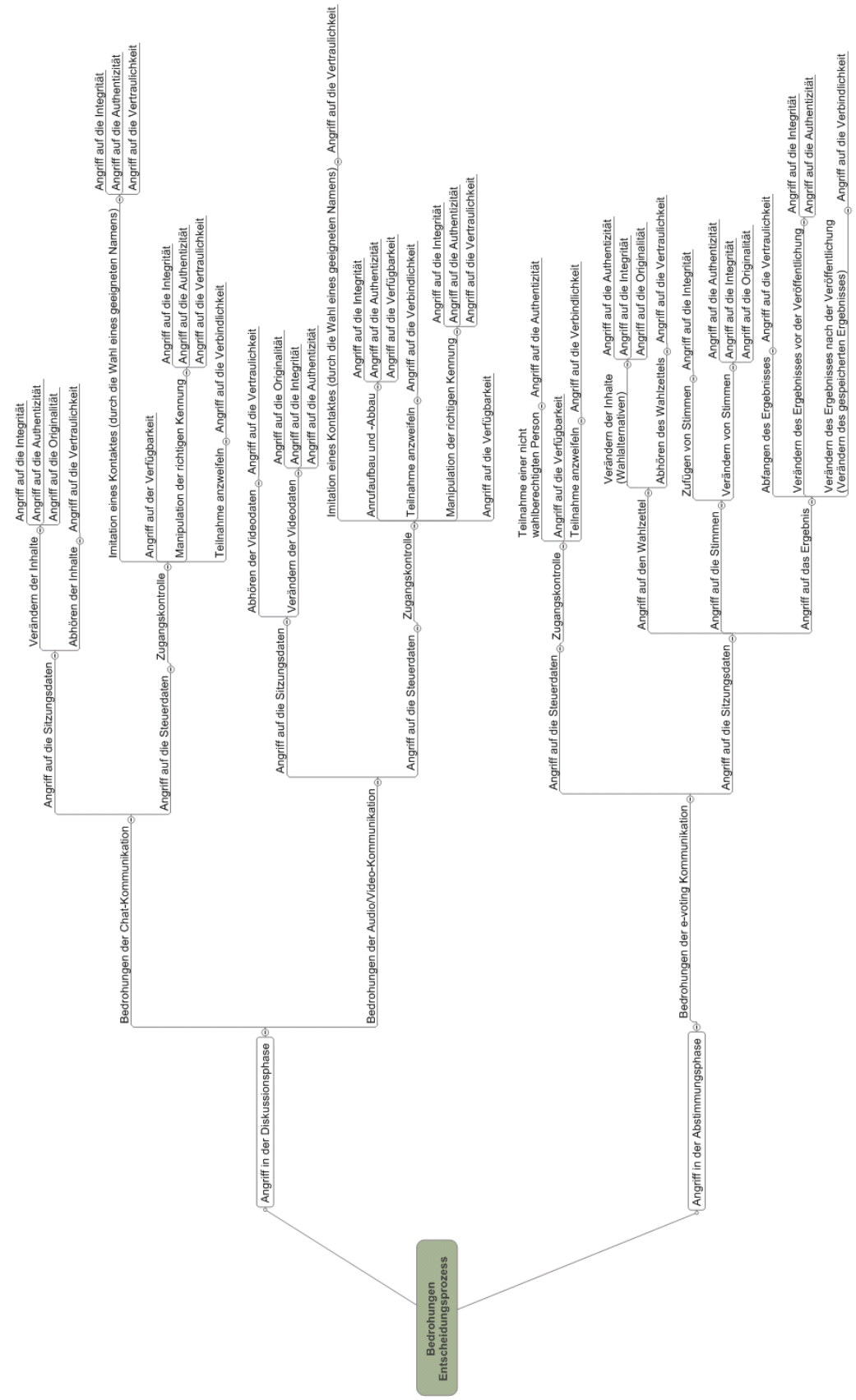

Abbildung 50 – Bedrohungen des gesamten Prozesses

Zusätzlich zu den Bedrohungen, die für jede Gruppenkommunikation und die darauf basierende kollaborative Applikation gelten, gibt es Angriffe, die speziell auf die Anwendung des elektronischen Wählens abzielen. Für die geheime Abstimmung gelten Dritten gegenüber (nicht Teilnehmer des Entscheidungsgremiums) die gleichen Anforderungen wie bei der Diskussion bezüglich der Vertraulichkeit und Integrität. Weiterhin muss sichergestellt werden, dass die einzelne Stimme (Votum) vertraulich und integer (jetzt gegenüber den anderen Teilnehmer) abgegeben ist und ebenfalls vertraulich und integer übermittelt wird. Es darf zu keinem Zeitpunkt ein Zusammenhang zwischen einem Teilnehmer (Wähler) und seiner Stimme erkennbar sein. Damit das Ergebnis der Abstimmung und dessen Veröffentlichung so korrekt wie in der Präsenz-Variante realisiert wird, muss sichergestellt werden, dass jede Stimme mitgezählt wird und dass kein Dritter oder ein anderer Teilnehmer Stimmen hinzufügen oder existierende Stimmen verändern kann. Zum Schluss muss gewährleistet werden, dass das Ergebnis protokolliert wird und dass das Berichtsprotokoll wie in den Szenarien davor verbindlich ist und langfristig gilt.

Im nächsten Kapitel wird jeder Teilprozess getrennt betrachtet und die Auswahl der Sicherheitsmechanismen für die Gewährleistung der entsprechenden Anforderungen erläutert.

13.3 Diskussion innerhalb einer Gruppe

Haben alle Mitglieder des Gremiums die Phase der Zutrittsberechtigung erfolgreich abgeschlossen, können sie (siehe Kapitel 13.1) in die Diskussionsphase einsteigen.

- Fall 1: Diskussionsphase ist nicht erforderlich. Gemeint ist hier eine Diskussionsphase, welche relevant für den Entscheidungsprozess ist und somit besonders gesichert werden soll. In so einem Fall kann nach der Authentifizierung und Verifikation der Identität über die Video-Verbindung die weitere Kommunikation nur auf Basis des Chat-Kanals als Übertragungsmedium erfolgen. Die Nachrichten für die Wahl basieren in der folgenden Implementation (Kapitel 14) auf dem gleichen Format wie die Chat-Kommunikation. Auf diese Weise wird z. B. Bandbreite gespart, und trotzdem können die Wahlberechtigten genau bestimmt werden. Um die Abstimmung zu realisieren, muss der Stimmzettel vorgegeben sein. Der Initiator der Kollaboration oder der Vorsitzende der Sitzung – diese Rollen müssen nicht ein und derselben Person zugewiesen sein – könnte, nachdem alle Mitglieder in der Gruppensitzung verifiziert wurden, den Stimmzettel in das dafür vorgegebene Dialogfeld eingeben und an alle verteilen.
- Fall 2: Diskussionsphase ist erforderlich. Wenn in Gremien eine Diskussionsphase durchgeführt wird, ist diese meist auch die Grundlage für die spätere Entscheidung einer Thematik. So werden während dieser Diskussion die Alternativen identifiziert, über die in einer nachfolgenden Abstimmung entschieden werden muss. Der Wahlzettel kann auch hier schon vorgegeben sein und von den Mitgliedern für die Abstimmung modifiziert werden. Wie im vorherigen Fall kann ein Mitglied die Erstellung und Modifizierung übernehmen. Das heißt. die Erstellung des Wahlzettels findet lokal bei einem Mitglied statt. Es erstellt nach den Vorschlägen des Gremiums den Wahlzettel und verteilt diesen anschließend an alle. Die andere Alternative ist die gemeinsame Erstellung des Wahlzettels. Hier kann jedes Mitglied den Wahlzettel nach seinem Ermessen erweitern und weiterleiten, bis alle Vorschläge übernommen sind. Erst dann wird der Wahlzettel zur Abstimmung freigegeben.

Mögliche Wahlzettel sind in der folgenden Abbildung dargestellt. Dabei entsprechen diese unterschiedlichen Arten von Entscheidungen und Wahlalternativen. Das Einzelvotum präsentiert die Entscheidung eines Teilnehmers. Dieses wird als XMPP-Nachricht gemäß dem implementierten Protokoll übertragen. Weitere Informationen zum Wahlzettel werden auch im Kapitel 14 gegeben.

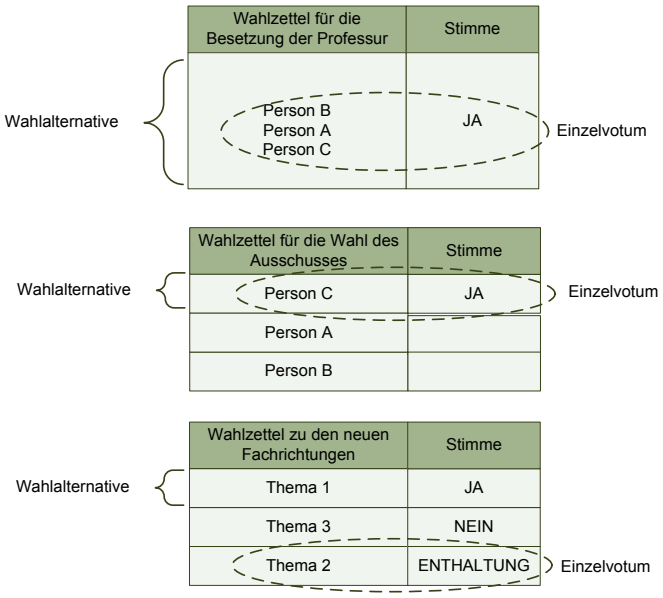

Abbildung 51 – Mögliche Wahlzettel

13.4 Abstimmung innerhalb einer Gruppe

Der Abstimmungsprozess (elektronische Wahl) ist der letzte Schritt eines Entscheidungsprozesses. Seine Abwicklung kann als offene oder geheime Abstimmung erfolgen.

- Fall 1: Offene Abstimmung. Diese Art der Abstimmung kann entweder als mündliche Abstimmung über den Audio-Video-Kanal, als gegenseitiges Versenden von Chat-Nachrichten oder über das entsprechende Voting-Dialogfeld erfolgen. Die Nutzung des A/V- oder Chat-Kanals würde zu einer unstrukturierten Kommunikation und Abstimmung führen, welche bezüglich der späteren Speicherung des Ergebnisses für die Verbindlichkeit der Abstimmung, nicht ratsam wäre. Geschickter wäre es für eine offene und geheime Abstimmung unterschiedliche Prozeduren zu realisieren, welche mit oder ohne kryptografische Funktionen auskommen. Denn für den vorliegenden Fall ist die Anonymität der einzelnen Stimme nicht erforderlich. Wichtig ist die Sicherheit der Kommunikation Dritten gegenüber, und dies wird durch den Zugangsmechanismus und die Vereinbarung eines Sitzungsschlüssels sowie die Gewährleistung der Integrität garantiert.
- Fall 2: Geheime Abstimmung. Wie in Kapitel 9 beschrieben, existieren unterschiedliche E-Voting-Protokolle. In der vorliegenden Arbeit wurde ein administrationsloses Protokoll (ohne die Existenz einer Wahlkommission) implementiert. Der Grund lag in dem Wunsch, so realitätsnah wie möglich Gruppenentscheidungen mit wenigen Teilnehmern zu realisieren, für die bislang nur die Möglichkeit besteht, diesen Prozess an einem Standort durchzuführen. In solchen Gremien gibt es je nach Fall zwar ein Mitglied, das die Rolle eines Vor-

sitzenden übernimmt, dieser hat aber eine eher koordinierende Rolle; seine Rechte bzgl. des Wahlprozederes entsprechen denen aller anderen Teilnehmer. Nichtsdestotrotz, es ist möglich, auch Voting-Protokolle zu realisieren, die auf der Existenz einer Wahlkommission in Form eines Servers aufbauen (vgl. die Diskussion in den Kapitel 15 und 16).

13.5 Wahlprotokoll

Für die Integration eines E-Voting-Protokolls als Realisierung einer sicherheitskritischen Anwendung in einer IM-System-Umgebung wurde an dieser Stelle das Protokoll von (Merritt 1983; Schneier 1996) in seiner ursprünglichen Form samt den Modifikationen der Variante von (Alkassar, Krimmer und Volkamer 2005) verwendet. Anhand eines Szenarios mit vier Teilnehmern wird in diesem Kapitel das Protokoll erläutert, bevor später (Kapitel 14) die konkrete Implementierung vorgestellt wird.

Das Wahlprotokoll durchläuft fünf Phasen, wie die folgende Abbildung 52 zeigt. Die Stimmen werden erstellt, verschlüsselt und in einer bestimmten Reihenfolge weitergereicht und bearbeitet (vgl. das Konzept der Mix-Nets (Chaum 1981)). Dann werden sie wieder in einer bestimmten Reihenfolge zwischen den Teilnehmern signiert und verteilt. Zum Schluss ist jeder Teilnehmer in der Lage, das Ergebnis selbst auszuwerten. Nach dem Mix-Prinzip bedeutet diese Weiterreichung und Bearbeitung von Stimmen, dass die Teilnehmer selbst als Mixknoten agieren (vgl. Kapitel 9).

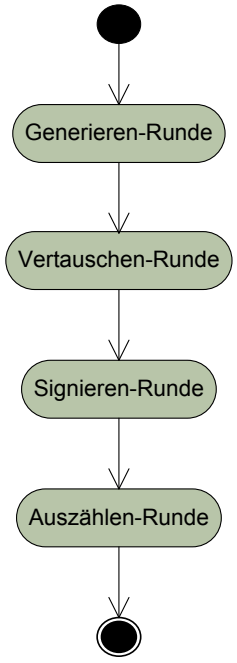

Abbildung 52 – Ablauf des Wahlprotokolls

Voraussetzung für den Ablauf dieses Protokolls ist die Existenz von zwei Schlüsselpaaren (öffentliche und geheime Schlüssel) für jeden Teilnehmer zum Zwecke der Verschlüsselung und Signierung. Wie im Kapitel 11.7 erläutert wurde, können folgende Alternativen für die konkrete Erzeugung der Schlüssel gewählt werden: Als Verschlüsselungsschlüssel können die DH-Schlüssel, die durch den Zugangsberechtigungsmechanismus erzeugt wurden, verwendet wer-

den. Statt der schwachen Verschlüsselung des DH-Mechanismus kann dafür auch der ElGamal-Algorithmus eingesetzt werden. In beiden Fällen kann der gewünschte Signaturschlüssel mit einer DSA-Signatur erzeugt werden. Eine weitere Alternative ist es, nach einer sicheren Überprüfung der Teilnehmeridentitäten, neue Schlüssel für Verschlüsselung und Signierung der Nachrichten mit Hilfe des RSA-Algorithmus zu generieren und diese in der Phase des DH-Parameteraustausches zu verteilen.

13.5.1 Generieren-Runde

Als Erstens wird eine zufällige Reihenfolge festgelegt, nach der die Teilnehmer (Wähler) die benötigten Nachrichten während des Protokollablaufs bearbeiten sollen. Als Beispiel wird hier die Reihenfolge (Person A, Person B, Person C, Person D) festgelegt. Jeder Wähler $x \in \{1 \dots n\}$ erhält den Stimmzettel, gibt seine Stimme V_X ab und verbindet diese Stimme mit einer Zufallszahl R_X. Somit generiert jedes Mitglied folgendes Tupel:

$$(48) \qquad (V_X , R_X)$$

Diese Zufallszahl wird benötigt, um in den späteren Schritten überprüfen zu können, ob das eigene Votum noch verarbeitet und mitgezählt wird.

In einem zweiten Schritt wird dieses Tupel mit den öffentlichen Schlüsseln aller Teilnehmer verschlüsselt. Dies geschieht in der umgekehrten Reihenfolge der Teilnehmer. Für das vorliegende Beispiel ist die Reihenfolge der Verschlüsselung mit den verwendeten öffentlichen Schlüsseln folglich: Person D, Person C, Person B, Person A (E_X steht für Verschlüsselung/Encryption). So erhält jeder den Wert:

$$(49) \qquad Inpart_X = E_A \left(E_B \left(E_C \left(E_D \left(V_X , R_X \right) \right) \right) \right)$$

Dieses Ergebnis wird erst in einer der späteren Runden weiterverarbeitet, indem entschlüsselt wird, um die Voten zu signieren und auszählen zu können.

Anschließend wird dieser Wert $Inpart_X$ mit den öffentlichen Schlüsseln der Teilnehmer und einer weiteren Zufallszahl verschlüsselt. Diese Zahlen erlauben es den Teilnehmer, eine Überprüfung des eigenen Votums durchzuführen. Jeder Teilnehmer berechnet jetzt folgenden Wert:

$$(50) \qquad V(X) = E_A \left(R_{XA} , E_B \left(R_{XB} , E_C \left(R_{XC} , E_D \left(R_{XD} , Inpart_X \right) \right) \right) \right)$$

Diese Nachricht schicken alle Wähler an den Ersten in der Reihenfolge, z. B. Person A. Alle Zwischenergebnisse zur Berechnung des obigen Wertes müssen zwischengespeichert werden, damit die Wähler bei der folgenden Prozedur jederzeit die Existenz des eigenen Votums überprüfen können.

Während im Protokoll von (Merritt 1983; Schneier 1996) eine asymmetrische Verschlüsselung verwendet wird, wird in der Erweiterung von (Alkassar, Krimmer und Volkamer 2005; Volkamer und Gessner 2003) eine hybride Verschlüsselung vorgeschlagen, um die Berechnung der Terme zu beschleunigen

13.5.2 Vertauschen-Runde

Der erste Teilnehmer in der Reihenfolge enthält aus der Generieren-Runde die vier zweifach verschlüsselten Voten (Verschlüsselung des inneren Terms $Inpart_X$ und Verschlüsselung der Nachrichten mit den Zufallszahlen)

$$(51) \quad V(A) = E_A\left(R_{AA}, E_B\left(R_{AB}, E_C\left(R_{AC}, E_D\left(R_{AD}, Inpart_A\right)\right)\right)\right)$$

$$(52) \quad V(B) = E_A\left(R_{BA}, E_B\left(R_{AB}, E_C\left(R_{BC}, E_D\left(R_{BD}, Inpart_B\right)\right)\right)\right)$$

$$(53) \quad V(C) = E_A\left(R_{CA}, E_B\left(R_{CB}, E_C\left(R_{CC}, E_D\left(R_{CD}, Inpart_C\right)\right)\right)\right)$$

$$(54) \quad V(D) = E_A\left(R_{DA}, E_B\left(R_{DB}, E_C\left(R_{DC}, E_D\left(R_{DD}, Inpart_D\right)\right)\right)\right)$$

Er entschlüsselt mit seinem privaten Schlüssel (dies ist möglich, weil sein öffentlicher Schlüssel als letzter bei der Verschlüsselung verwendet wurde) und entfernt die Zufallszahlen dieser Ebene (alle R_{XA}). Das Ergebnis sind vier Voten der Form

$$(55) \quad V(X) = E_B\left(R_{XB}, E_C\left(R_{XC}, E_D\left(R_{XD}, Inpart_X\right)\right)\right)$$

Er vertauscht die Reihenfolge der Terme (um das Wahlgeheimnis zu wahren) und schickt diese dem nächsten Empfänger gemäß der festgelegten Reihenfolge (in diesem Beispiel B) vom Typ

$$(56) \quad message = V(B), V(A), V(D), V(C)$$

Die gleiche Prozedur führt der nächste Empfänger (B) durch: Er entschlüsselt, entfernt die Zufallszahlen (R_{XB}) vertauscht die Voten und leitet die Nachricht an den nächsten Teilnehmer (Wähler C) weiter. Dieser erhält:

$$(57) \quad V(X) = E_C\left(R_{XC}, E_D\left(R_{XD}, Inpart_X\right)\right)$$

$$(58) \quad message = V(D), V(C), V(B), V(A)$$

Entsprechend agiert C mit den vier erhaltenen Voten. Der letzte Teilnehmer (D) leitet die Voten wieder an den Initiator A weiter. Die an Wähler A versendeten Daten entsprechen den Voten die innere Verschlüsselung (vgl. Gleichung (49)).

$$(59) \quad V(X) = Inpart_X$$

13.5.3 Signieren-Runde

Der erste Wähler in der Reihenfolge (Wähler A) erhält alle Voten von Teilnehmer D. Jedes Votum entspricht folgendem Format:

$$(60) \quad V(X) = Inpart_X = E_A\left(E_B\left(E_C\left(E_D\left(V_X, R_X\right)\right)\right)\right)$$

Er entschlüsselt seinerseits die vier Voten mit seinem privaten Schlüssel und erhält:

$$(61) \quad V(B) = Inpart_B = E_B\left(E_C\left(E_D\left(V_B, R_B\right)\right)\right)$$

$$(62) \quad V(C) = Inpart_C = E_B\left(E_C\left(E_D\left(V_C, R_C\right)\right)\right)$$

$$(63) \quad V(A) = Inpart_A = E_B\left(E_C\left(E_D\left(V_A, R_A\right)\right)\right)$$

$$(64) \quad V(D) = Inpart_D = E_B\left(E_C\left(E_D\left(V_D, R_D\right)\right)\right)$$

Er signiert diese und sendet die Nachricht an *alle* Teilnehmer. Damit kann jeder einzelner Teilnehmer das Ergebnis ermitteln. Die folgende Nachricht mit den signierten Voten wird zwar an alle verschickt, aber in der normalen Reihenfolge wie in der vorhergehenden Runde verarbeitet, d. h. im vorliegenden Beispiel von Teilnehmer B. Dabei entspricht $sig_X V(Y)$ dem signierten Votum von Y, welches von X signiert wurde (privater Signaturschlüssel von X) (vgl. Kapitel 13.5 und Kapitel 11.7)

$$(65) \quad message = sig_A V(B), sig_A V(A), sig_A V(D), sig_A V(C)$$

Der Teilnehmer B verfährt ähnlich: Er verifiziert und entfernt die Signatur von A. Er entschlüsselt die einzelne Voten, signiert diese und verschickt die Nachricht an alle.

$$(66) \quad message = sig_B V(B), sig_B V(A), sig_B V(D), sig_B V(C)$$

Nachdem Teilnehmer C die gleichen Schritte durchgeführt hat, erhält D folgende Terme:

$$(67) \quad V(B) = E_D\,(V_B\ , R_B)$$

$$(68) \quad V(C) = E_D\,(V_C\ , R_C)$$

$$(69) \quad V(A) = E_D\,(V_A\ , R_A)$$

$$(70) \quad V(A) = E_D\,(V_D\ , R_D)$$

D verfährt wie die vorangegangenen Teilnehmer. Danach erhält jeder Teilnehmer vom letzten Teilnehmer (D) alle unverschlüsselten Stimmen in der Form:

$$(71) \quad V(A) = (V_A\ , R_A)$$

$$(72) \quad V(B) = (V_B\ , R_B)$$

$$(73) \quad V(C) = (V_C\ , R_C)$$

$$(74) \quad V(D) = (V_D\ , R_D)$$

$$(75) \quad message = sig_D V(B), sig_D V(A), sig_D V(D), sig_D V(C)$$

In ihrem Erweiterungsprotokoll schlagen (Alkassar, Krimmer und Volkamer 2005; Volkamer und Gessner 2003) die Signierung nicht nur der Voten, sondern der Gesamtnachrichten vor und eine Signierung während des Gesamtprotokoll, um die Integrität der Nachrichten und Originalität der Teilnehmer sicherstellen zu können.

Weiterhin wird eine Konsistenzprüfung vorgenommen, in der jeder Wähler innerhalb der Signierrunde überprüft, ob die erhaltenen Voten im Schritt n konsistent zu den Voten von $n-1$ sind. Damit soll verhindert werden, dass unterschiedliche Ergebnisse entstehen, wenn ein Wähler eine Stimme manipuliert. Das kann passieren, weil in der Signierrunde die signierten Daten an alle Teilnehmer verschickt werden.

13.5.4 Auszählen-Runde

In dieser letzten Runde verifizieren alle Teilnehmer die Signatur von D. Ist sie korrekt, so sucht jeder nach seiner Zufallszahl R_X , um feststellen zu können, ob das eigene Votum noch vorhanden ist. Anschließend kann das Ergebnis ausgezählt werden.

13.6 Bewertung des Wahlprotokolls

An dieser Stelle wird das Wahlprotokoll einerseits mit Bewertungskriterien für elektronische Wahlen bewertet (Volkamer und Vogt 2008), um dessen Mächtigkeit zu überprüfen. Andererseits wird aufgezeigt, dass dieses Protokoll die gestellten Anforderungen aus dem Kapitel 13.2.2 erfüllt.

Vergleicht man die Basisanforderungen des Common Criteria Profiles (CC-Profils) für Online Wahlen und das oben beschriebene Protokoll, lässt sich feststellen, dass die Basisanforderungen erfüllt werden:

1. Wahlgeheimnis: Durch die Verschlüsselung des eigenen Votums (Generieren-Runde) und das Vertauschen der verschlüsselten Voten (Vertauschen-Runde) kann keine Zuordnung von Stimme und Wähler vorgenommen werden, wenn mindestens ein Wähler ehrlich ist.
2. Die berechtigten Wähler sind bekannt: Die Gremiumsmitglieder, die die geheime Abstimmung mit dem Instant-Messaging-System als Kollaborationswerkzeug durchführen, sind namentlich bekannt. Durch den Zugangsberechtigungsmechanismus und die gegenseitige Überprüfung der Identität über den Audio-Video-Kanal ist bekannt, wer an der Sitzung teilnimmt (Zugangskontrolle Abbildung 50). Dadurch ist auch gewährleistet, dass nur registrierte Wähler abstimmen können. Durch diese Erkennung kann auch ein Mitglied nicht im Nachhinein seine Teilnahme an der Sitzung anzweifeln. Darüber hinaus wird durch die Realisierung des Konzepts der Verbindlichkeit mit den verschiedenen Rollen nicht nur das Ergebnis, sondern auch die Liste der Wähler gespeichert (Vgl. Kapitel 11.11)
3. Kein Beweis der eigenen Stimme: Hier ist der Begriff der Quittungsfreiheit zu nennen. Man unterscheidet zwischen der Quittungsfreiheit seitens des Systems (schwache Quittungsfreiheit) und der Quittungsfreiheit seitens des Wählers. Der erste Fall wird vom vorgestellten Protokoll realisiert. Das System speichert zu keinem Zeitpunkt nachhaltig die Stimme des Wählers. Nur während der Erzeugung des eigenen Votums wird das Tupel (V_X, R_X) im Hauptspeicher gespeichert, um das eigene Votum in den nachfolgenden Runden zu vergleichen. Ist der Wähler ehrlich, wird er selbst seine Stimme nicht gegenüber Dritten beweisen wollen. Er hätte die Möglichkeit dies zu tun, wenn er vorsätzlich sein generiertes verschlüsseltes Votum und die Terme, die er für den Vergleich benötigt, längerfristig speichert. Damit wäre die zweite Form der Quittungsfreiheit nicht gegeben. Im Schutzprofil wird System-Quittungsfreiheit vorgeschrieben.
4. Nach dem Protokoll ist nur eine Stimme pro Wähler erlaubt: Ein Einschleusen oder Austauschen der Stimmen ist nicht möglich.
    a) *Hinzufügen von Stimmen:* Ein Teilnehmer versucht weitere Stimmen in das System einzuschleusen. Wenn dieser Angriff vom Initiator am Anfang der Wahl durchgeführt wird, wird der Zweite in der Reihenfolge (z. B. Teilnehmer B) den Angriff erkennen, da er mehr Voten als Teilnehmer erhalten wird. Wenn dieser Angriff von irgendeinem der Teilnehmer bei der Generieren-Runde initiiert wird, dann wird dies vom Initiator aufgedeckt, da er mehr Voten erhalten wird als die Anzahl der Personen, die an dieser Wahl teilnehmen.

b) *Austauschen von Stimmen:* Ein weiterer Angriff ist das Ersetzen von Voten. Dieser Angriff kann in unterschiedlichen Runden erkannt werden, spätestens aber in der Signieren-Runde. Angenommen, Person B versucht, eine Stimme zu ersetzen. Da sie nicht wissen kann, welche Stimme sie ersetzen wird, gibt es zwei Möglichkeiten: Sie ersetzt die Stimme eines der Teilnehmer, die nach ihr die Nachrichten verarbeiten (für das Beispiel C oder D). Dieser Versuch bleibt erfolglos, da der Teilnehmer C als Erstes überprüft, ob seine Stimme korrekt weitergegeben wird. Hier würde der Manipulationsversuch erkannt werden, allerdings ohne zu wissen, wer der Angreifer ist. Es kann jedoch nur jemand sein, der vor ihm in der Reihenfolge die Stimmen bearbeitet hat. Die zweite Möglichkeit ist, dass er die Stimme eines Teilnehmers erwischt, der vor ihm in der Reihenfolge die Nachrichten entschlüsselt und bearbeitet. In dem vorliegenden Beispiel wäre dies die Stimme z. B. von A. In diesem Fall wird Person A erst in der Signierrunde den Angriff entdecken, wenn sie nach ihrem Votum die empfangenen Nachrichten überprüft. Hier wäre sie auch nicht in der Lage, den Angreifer zu identifizieren. Wenn während der Signierrunde ein Votum manipuliert wird, kann dies direkt aufgedeckt werden. Der Grund dafür ist, dass jeder Teilnehmer einerseits alle signierten Voten erhält und sein Votum überprüfen kann und andererseits die Protokollschritte auf Konsistenz überprüfen kann. Wenn jemand hier entdeckt, dass sein eigenes Votum nicht dabei ist, kann er den ganzen Ablauf unterbrechen. Durch den Vergleich der Voten kann hier der Angreifer identifiziert werden.

5. Es ist kein Zwischenergebnis ermittelbar: Durch die Einhaltung einer bestimmten Reihenfolge der Stimmenverarbeitung und der entsprechenden Verschlüsselung kann kein Teilnehmer das Ergebnis vorherberechnen. Das Original-Protokoll hat den Schwachpunkt, das der letzte Teilnehmer die Ergebnisse als Erster kennt. In der aktuellen Version (vgl. Kapitel 14), fällt dieser Schwachpunkt durch die Real-time-Bearbeitung einer Instant-Messaging-Kommunikation nicht mehr ins Gewicht.
6. Verifikation der eigenen Stimme: Diese Anforderung ist eine optionale Anforderung des CC-Profils. Im vorliegenden Protokoll wird dies durch die Überprüfung der Existenz des eigenen Votums in jeder Runde und die Konsistenzprüfung in der Signieren-Runde erfüllt.

Externe Angriffe wie Spoofing oder Man-in-the-Middle-Angriffe auf die verwendeten Schlüssel oder Nachrichten werden durch die in Kapitel 11 beschriebenen Mechanismen verhindert. Durch den Einsatz des Diffie-Hellman-Verfahrens für die Erzeugung der Schlüssel und die Audio-Video-Überprüfung der Identität ist sowohl die Identität der Teilnehmer als auch die Authentizität der Schlüssel sichergestellt. Die Integrität der Nachrichten ist durch die Verwendung der Signatur sowohl in der Phase der Diskussion als auch beim Austausch der Wahldaten sichergestellt. Der Wahlzettel mit den Wahlalternativen wird verschlüsselt und signiert von dem Erzeuger zwischen den Teilnehmern ausgetauscht (verschlüsselt mit dem Sitzungsschlüssel, Erzeuger ist der Absender des Wahlzettels, dies kann der Initiator der Kommunikation oder ein zufälliger Teilnehmer sein). Das Ergebnis wird von jedem Teilnehmer selbst berechnet und kann mit den anderen Wählern verglichen werden. Durch das Konzept der Verbindlichkeit des Ergebnisses wird auf jeden Fall ein Vergleich der Ergebnisse zwischen dem Signierer und Aufbewahrer vollzogen (siehe Kapitel 11.10).

14 Realisierung der Entscheidungsprozesse

In diesem Kapitel wird die prototypische Anwendung „IMVote“ vorgestellt, welche die Konzepte aus den Abschnitte 11, 12 und 13 umsetzt und demonstriert. IMVote realisiert einen Entscheidungsprozess mit einer Diskussionsphase und einer anschließenden geheimen Wahl auf Basis eines IM-Systems. In den folgenden Abschnitten werden kurz die verwendeten Basis-Technologien vorgestellt, es folgt ein Überblick über das IMVote-Gesamtsystem, einschließlich einem Use-Case-Diagramm sowie eine Erläuterung der Bestandteile der Implementation. Anschließend werden die vier Plug-Ins beschrieben, aus denen IMVote besteht.

Wie beim Entwurf von Softwaresystemen üblich werden bei der Beschreibung des Gesamtsystems und der verschiedenen Plug-Ins in Unterabschnitten jeweils unterschiedliche Aspekte erläutert (Kruchten 1995; Jacobson, Booch und Rumbaugh 1999). Hierzu zählen unter anderem Use Cases, der strukturelle Aufbau der Implementation, Datenmodelle und Abläufe. Soweit nicht anders angemerkt, werden dazu UML-Diagramme verwendet.

14.1 Verwendete Technologien

Als Grundlage für die Implementation des Prototyps wurden zahlreiche Werkzeuge und Frameworks eingesetzt. Diese werden im Folgenden kurz zusammengefasst. Die Funktionsweise wird dabei nur insofern erläutert, als dies für das Verständnis der vorliegenden Arbeit notwendig ist. Weitere Details findet der interessierte Leser in den jeweils referenzierten Dokumentationen bzw. Websites.

14.1.1 Instant-Messaging-System

Als zentrale Basis für die Realisierung der prototypischen Implementierung wurde der IM-Client „Spark“ sowie der zugehörige Server „Openfire“ eingesetzt. Es handelt sich dabei um ein Projekt einer Open Source Community, welches auf dem Jabber/XMPP Protokoll basiert. Das Vorhaben wird unterstützt durch die Firma Ignite Realtime, die kommerzielle Erweiterungen und Dienstleistungen anbietet. Die hier relevanten Teile sind jedoch alle als Quellcode verfügbar. Die verwendete Lizenz entspricht ungefähr einer Apache Lizenz, d.h. ein Entwickler kann auf Grundlage des verfügbaren Quellcodes auch kommerzielle Anwendungen entwickeln.

Konkret wurden folgende Versionen von Openfire und Spark verwendet

- Ignite Realtime Spark 2.5 und 2.6 als IM-Client (Ignite Realtime Community 2008b)
- Ignite Realtime Openfire 3.6.3 als IM-Server (Ignite Realtime Community 2008a)
- Ignite Realtime Smack API 3.1.0, eine Klassenbibliothek für XMPP, (Ignite Realtime Community 2009a)

Zur Erläuterung des verwendeten Kommunikationsmodells (Client-Server vs. Peer-to-Peer) sollte an dieser Stelle betont werden, dass bei der prototypischen Umsetzung die Rolle des Openfire-Servers absichtlich so gering wie möglich gehalten wurde. Der Server übernimmt nur anfangs einen Teil der Authentifikation der Teilnehmer (Registrierung und Einloggen, vgl. Abschnitt 11.1.1 erste Phase). Die Mechanismen für Schlüsselvereinbarung, Verschlüsselung, Verifizierung der Schlüssel, Kommunikation (Chat und Audio-Video) während der Diskussion

und das Protokoll für die Wahl sind alle als Nachrichtenaustausch zwischen den Teilnehmern realisiert. Die Kommunikation für Authentifizierung, Schutz der Integrität und Realisierung der Verbindlichkeit erfolgt nach dem *Peer-to-Peer*-Prinzip.

14.1.2 Sonstige Frameworks

Neben dem IM-Client Spark und dem IM-Server Openfire wurden folgende Werkzeuge und Frameworks als Grundlage für die prototypische Implementation eingesetzt:

- Eclipse 3.5 („Galileo“) als allgemeine Entwicklungsumgebung (The Eclipse Foundation 2009)
- Sun Netbeans 6.5 zur Erstellung der Benutzeroberfläche (Sun Microsystems 2008)
- Evelopers Unimod 1.3.39 zur Modellierung von Protokollen mittels Zustandsmaschinen, sowie zur Ausführung dieser Protokolle zur Laufzeit (eVelopers Corporation 2009)
- Bouncy Castle Crypto APIs zur Bereitstellung von Standardverfahren zur Schlüsselerzeugung, Verschlüsselung und Signierung (bouncycastle.org 2009)
- JUnit 4 als Test-Framework (Junit.org 2009)
- Java Media Framework für die Übertragung von Anzeige von Multimedia-Daten, insbesondere die Videoübertragung für den Authentifikationsvorgang (Sun Microsystems 2009)

14.2 IMVote – das Gesamtsystem

Bevor auf die Details der verschiedenen Subsysteme genauer eingegangen wird, soll nun zunächst ein Überblick über das Gesamtsystem gegeben werden.

14.2.1 Use-Cases

Als Ausgangspunkt zur Festlegung der beim Software-Entwurf zu berücksichtigenden Szenarien wird zunächst ein Use-Case-Diagramm verwendet (Abbildung 53). Dort sind die Rollen der beteiligten Akteure dargestellt, sowie die Use-Cases, an denen sie teilnehmen.

Es treten Initiator, Aufbewahrer, Signierer, Zertifikatsersteller und Teilnehmer als Rollen auf. Alle anderen Rollen sind eine Spezialisierung der Rolle Teilnehmer. Dies durch die Konzeption der vorliegenden Arbeit begründet: Die angestrebte Kollaboration soll spontan und kurzfristig zu Stande kommen, ohne dass es notwendig ist, *vor* der eigentlichen Sitzung Rollen und explizite Berechtigungen festzulegen. Das bedeutet, dass potentiell jeder der Teilnehmer eines Gremiums die notwendigen Rollen übernehmen kann. Er übernimmt damit rollenspezifische Aufgaben, wie die Durchführung der Überprüfung der Identität und Echtheit der Diffie-Hellman-Schlüssel, das Signieren des Wahlergebnisses, die Aufbewahrung des Ergebnisses oder die Erstellung eines Zertifikats.

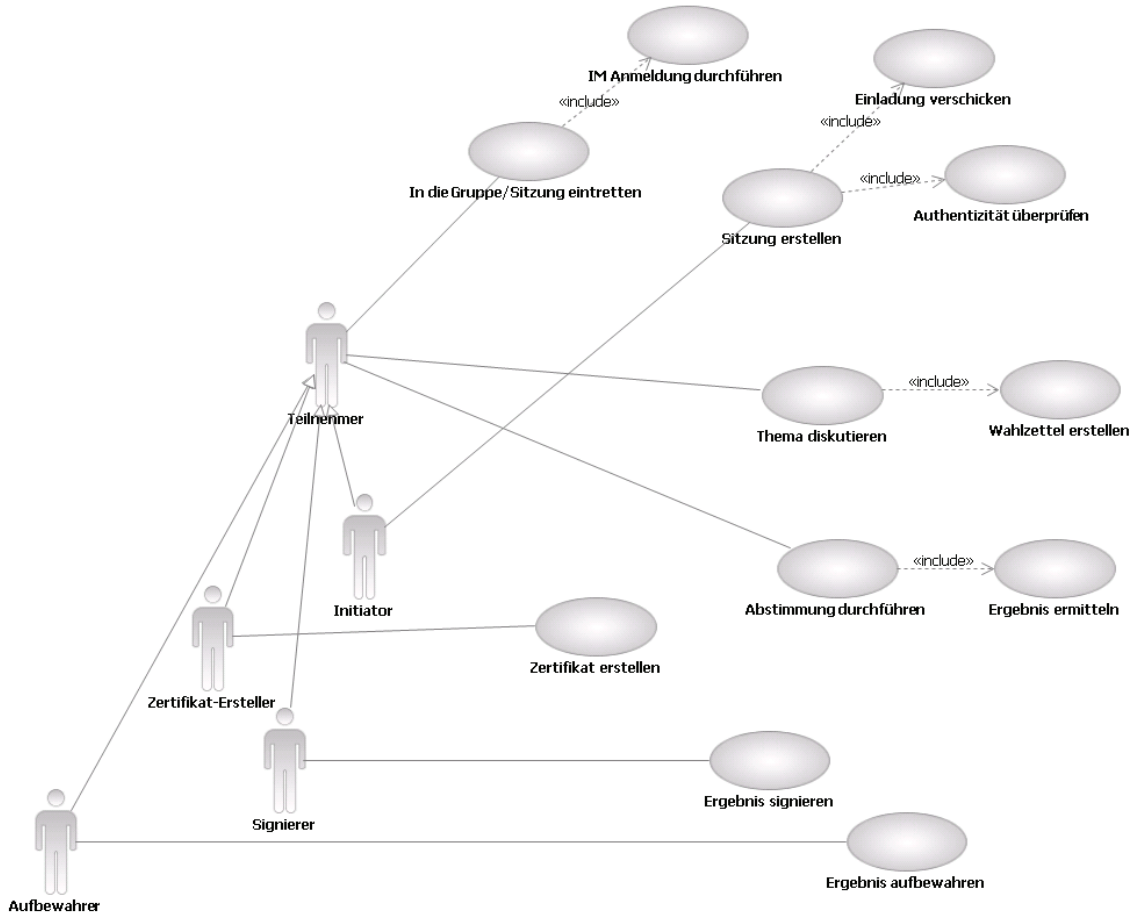

Abbildung 53 – Use-Case-Diagramm für Entscheidungsprozesse

Darüber hinaus gibt es Aufgaben, die alle Teilnehmer unabhängig von ihrer speziellen Rollen gleichberechtigt ausführen können, z.B. die Diskussion mittels Chat- oder Audio-Video-Kanal und die elektronische Wahl samt Berechnung des Wahlergebnisses.

14.2.2 Struktureller Aufbau des Gesamtsystems

Die Anwendung IMVote wurde in Form von Plug-Ins für Spark in der Programmiersprache Java realisiert. Die logische Strukturierung der Anwendung findet sich daher in der Struktur der Java Packages (Pakete) wieder. Bevor diese erläutert werden, sollte zum besseren Verständnis erwähnt werden, dass die Implementation von IMVote an vielen Stellen nach dem Model-View-Controller-Prinzip strukturiert ist:

- *Model* – Klassen, die Anwendungsdaten aus der Realität nachbilden (z.B. `Wahl`)
- *View* – Klassen, die entsprechende Benutzeroberflächen-Elemente realisieren (z.B. `WahlFrame`)
- *Controller* – Klassen, die koordinierende Funktionen übernehmen und die Anwendung steuern (z.B. `WahlController`)

Diese Begriffe finden sich später an vielen Stellen in den entsprechenden Package-Namen (z.B. `org.imvote.model`). Ein Überblick über die verwendeten Java-Pakete ist in der folgenden Abbildung dargestellt (vgl. Abbildung 54).

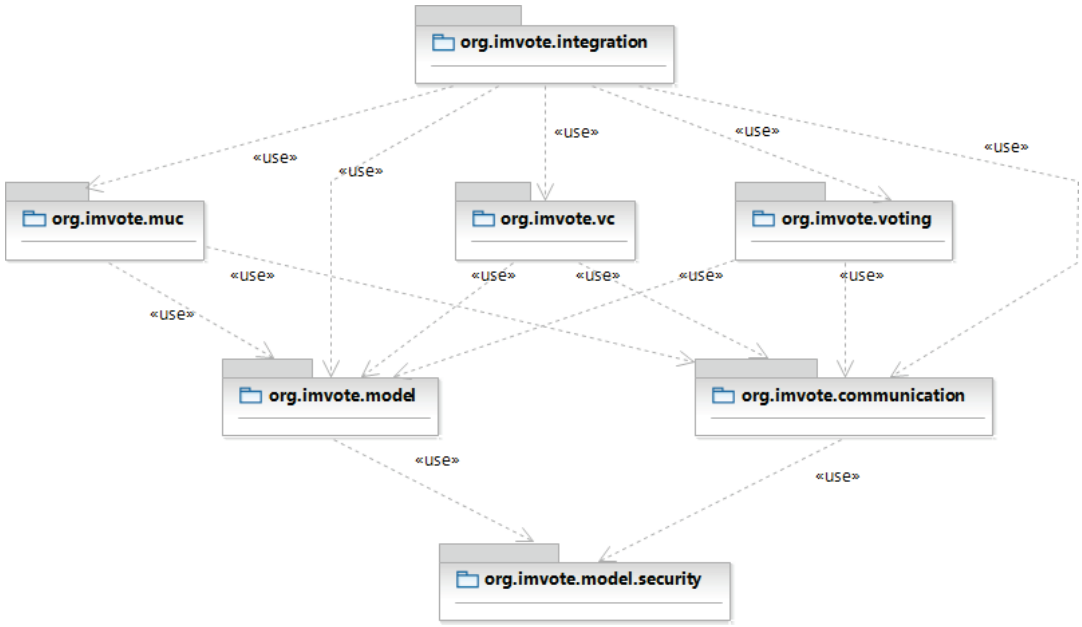

Abbildung 54 – Struktureller Aufbau (Gesamtüberblick)

Insgesamt besteht IMVote aus vier Spark-Plug-Ins. Diese werden durch Java Packages realisiert, die in den oberen beiden Ebenen in Abbildung 54 gezeigt sind:

- Das Integration-Plug-In org.imvote.integration implementiert die Gesamtanwendung „IMVote“ und integriert die anderen drei Plug-Ins. Die Anwendung IMVote wird dadurch gestartet, dass dieses Plug-In durch den Spark IM-Client initialisiert wird. Dabei werden die Erweiterungen von Spark installiert, die notwendig sind, um IMVote in die Spark-Infrastruktur zu integrieren, beispielsweise Klassen zum Empfangen von anwendungsspezifischen Nachrichtentypen.
- Das SecureMUC-Plug-In org.imvote.muc realisiert einen sicheren Multi-User-Chat sowie den dazu notwendigen Schlüsselaustausch auf Basis von Diffie-Hellman (vgl. Konzepte in Kapitel 11.3, Details zur Implementation in Kapitel 14.3).
- Das SecureVC-Plug-In org.imvote.vc erweitert den Authentifikationsvorgang auf Basis von Diffie-Hellman um eine Überprüfung mittels Vorlesen von Hashwerten und Erkennen von Identitäten im Audio/Video-Kanal (=Video Channel VC) (vgl. Konzepte in Kapitel 11.3, Details zur Implementation in Kapitel 14.4).
- Das E-Voting-Plug-In org.imvote.voting realisiert eine Anwendung für geheime, elektronische Wahlen über Instant-Messaging, einschließlich des dabei verwendeten Wahlprotokolls (vgl. Konzepte in Kapitel 13.5, Details zur Implementation in Kapitel 14.5). Weiterhin übernimmt dieses Plug-In die sichere, verbindliche Speicherung der Ergebnisse zur Realisierung der langfristigen Verbindlichkeit (vgl. Konzepte in Kapitel 11.12 Details zur Implementation in Kapitel 14.5).

Diese vier Plug-Ins bilden den Kern der Anwendung IMVote. Darüber hinaus gibt es weiter Packages, die Basisfunktionen implementieren:

- Das Paket org.imvote.model implementiert generische Basisklassen für Anwendungsdaten, die für alle oben genannte Plug-ins gemeinsam sind (z.B. BenanntesElement), die durch plug-in-spezifischen Klassen spezialisiert werden (z.B. Wahlzettel).
- Auf ähnliche Weise werden in org.imvote.communication generische Klassen zur Kommunikation über XMPP sowie die dafür notwendige Integration mit der zugrundeliegenden Spark-Infrastruktur beschrieben. Hierzu gehören unter anderem Listener zum Empfang von XMPP-Nachrichten und sogenannte IQProvider zum Parsen der Nachrichten.
- Generische Sicherheitsfunktionen werden in org.imvote.model.security realisiert. Hierzu gehören beispielsweise Klassen zur Verwaltung von Schlüsseln oder zur Signalisierung von Sicherheitsfunktionen bei der Kommunikation zwischen Teilnehmern (z.B. StartSchlüsselverteilung).

Betrachtet man abschließend die allgemeine Plattform, welche im Kapitel 12 vorgestellt wurde und die hier vorgestellten Komponenten der prototypischen Realisierung, lässt sich eine Zuordnung vornehmen und folgende Gemeinsamkeiten und Unterschiede festhalten.

Das Plug-In SecureMUC entspricht einer Erweiterung des Chat-Moduls der Plattform. Es verwendet existierende Nachrichtentypen des zugrunde liegenden IM-Systems (Spark) und erweitert dieses, indem es eigene speziellere Nachrichtentypen ableitet (z.B. zur Einladung in eine IMVote-Sitzung oder zum Aushandeln von DH-Parametern) und einen Ablauf der Kommunikation (Reihenfolge der gesendeten Nachrichten, Protokoll in der Abbildung 46) festlegt. Dieser gewünschte sichere Ablauf mit den dabei ausgetauschten Nachrichtentypen und der dazu notwendigen Koordination (Was passiert, wenn eine bestimmte Nachricht eingeht?) entspricht in der Plattform den Aufgaben des Security-Managers. Dazu gehört auch das Protokoll der Schlüsselaustausch nach dem DH-Verfahren nebst der Verwendung der Kryptoverfahren, die vom Kryptoproviders bereitgestellt werden (hier Bouncy Castle).

Das Plug-In SecureVC entspricht einer Erweiterung des Audio-Video-Moduls in der Plattform. Es baut eine Audio-Video-Verbindung nach RTP auf. Die Erweiterung besteht darin, dass zusätzlich die vorgeschlagenen Mechanismen des Kryptoproviders verwendet werden, um die Verschlüsselung der Kommunikation zu realisieren.

Das Voting-Plug-In entspricht dem Applikations-Modul in der Plattform. Die Art der anwendungsspezifischen Nachrichten und die Reihenfolge ihrer Abarbeitung werden in den Nachrichten-Typen und dem Protokoll in der Abbildung 46 spezifiziert. Darüber hinaus werden diese Nachrichten nach den festgelegten Mechanismen des Kryptoproviders verschlüsselt und signiert. Anschließend wird das Ergebnis der Wahl durch das Protokoll für die langfristige Aufbewahrung verbindlich gespeichert.

Das in der Plattform gezeigte Modul „User Interface“ findet sich in der prototypischen Realisierung nicht als eigenständige Systemkomponente wieder. Stattdessen enthalten die unterschiedlichen Plug-Ins jeweils gesonderte Pakete für die Realisierung der Benutzeroberfläche (z.B. org.imvote.muc.view und org.imvote.voting.view).

14.2.3 Integration der Plug-Ins

Die eigentliche Funktionalität von IMVote wird durch die drei Plug-Ins SecureMUC, SecureVC und Voting realisiert. Jedes einzelne dieser Systemkomponenten nutzt die Schnittstellen, die von Spark für solche Erweiterungen des IM-Clients vorgesehen sind. Dazu gehören beispielsweise Funktionen zur Initialisierung und zum Beenden eines Plug-Ins beim Start bzw. Beenden des Spark IM-Clients.

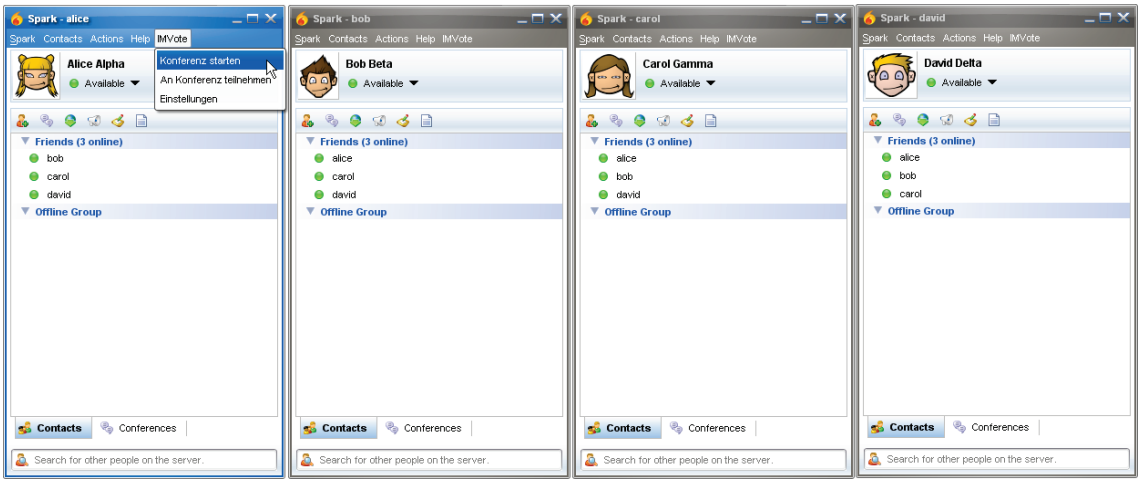

Abbildung 55 – Starten des IMVote Plug-Ins in einer Kollaboration von vier Teilnehmer

Diese Plug-In können also auch einzeln ausgeführt werden (was beispielsweise zu Testzwecken sinnvoll ist). Um jedoch die hier konzipierte Anwendung IMVote zu realisieren, müssen diese drei Plug-Ins zusammenarbeiten. Daher wurde ein weiteres Integration-Plug-In entwickelt, dieses übernimmt folgende Aufgaben:

- Es erzeugt je eine Instanz von SecureMUC, SecureVC und Voting-Plug-In und verbindet diese miteinander.
- Es repräsentiert gegenüber dem Spark IM-Client die Anwendung IMVote als ein zusammenhängendes Plug-In, mit den dazu notwendigen Methoden (Initialisierung, Beenden, usw.). Dabei werden intern die entsprechenden Methoden der anderen Plug-Ins aufgerufen, insbesondere bei der Initialisierung.
- Es erzeugt ein IMVote-Menü und fügt dies in die Benutzeroberfläche von Spark ein. Darüber kann die Anwendung durch den Benutzer gestartet werden (vgl. Abbildung 55, Menüleiste).
- Es registriert den Bouncy Castle Provider, so dass er für alle Plug-Ins zur Verfügung steht, wenn diese Instanzen von Kryptoalgorithmen anfordern.
- Es initialisiert die Logging-Funktion, die eine Protokollierung der Abläufe beim Entwickeln und der Fehleranalyse ermöglicht.

Die folgende Abbildung 56 zeigt einen Überblick über diese Plug-Ins und die wichtigsten Datenflüsse bei ihrer Zusammenarbeit.

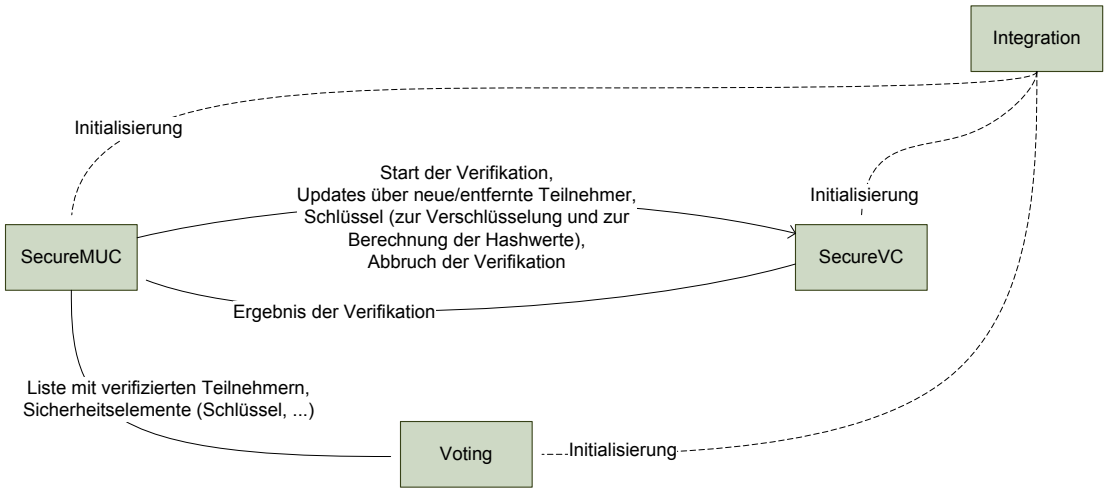

Abbildung 56 – Zusammenarbeit der verschiedenen Plug-Ins

Nachfolgend werden die drei Plug-Ins, welche die eigentliche Funktionalität von IMVote bereitstellen, detaillierter erläutert.

14.3 SecureMUC-Plug-In

Das SecureMUC-Plug-In (Dietz 2009) realisiert die Schlüsselgenerierung und -austausch mit Hilfe des Diffie-Hellman-Verfahrens, die Generierung der Schlüsselpaare für die Signierung der Nachrichten und die anschließende Generierung eines gemeinsamen Schlüssel, der als Sitzungsschlüssel für die Gewährleistung der Vertraulichkeit genutzt wird. Das Kürzel MUC steht als Abkürzung für „Multi User Chat“, weil mehrere Teilnehmer während und nach der Authentifikation über sichere Chat-Verbindungen kommunizieren.

14.3.1 Struktureller Aufbau

SecureMUC ist im Paket org.imvote.muc implementiert. Die Abbildung 57 zeigt die entsprechenden Pakete einer Übersicht.

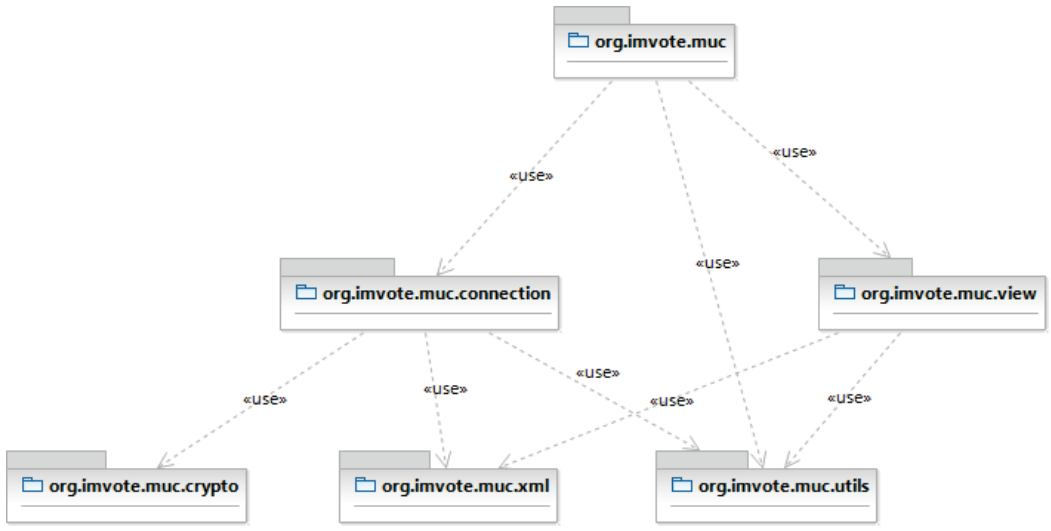

Abbildung 57 – Struktureller Aufbau (SecureMUC)

Die zentralen Klassen in org.imvote.muc nutzen die Pakete org.imvote.muc.connection und org.imvote.muc.view. In org.imvote.muc.connection werden Klassen implementiert, welche die Verbindungen zwischen einem Initiator und den Teilnehmern sowie zwischen den Teilnehmer verwalten. Weiterhin werden hier auch die Klassen zur Verfügung gestellt, die für die Integration von SecureMUC und SecureVC und somit für die Überprüfung der Echtheit eines Teilnehmers zuständig sind.

Das Paket org.imvote.muc.crypto realisiert das Diffie-Hellman-Key-Agreement, die spätere Verschlüsselung von Nachrichten mit Hilfe von symmetrischen Schlüsseln (mit AES) sowie die Signierung von Nachrichten (mit RSA).

Wenn ein Initiator einen Teilnehmer einlädt (und dieser die Einladung annimmt), werden zwei Schlüsselpaare erzeugt: das DH-Schlüsselpaar, welches für das Diffie-Hellman-Key-Agreement verwendet wird und ein RSA-Schlüsselpaar, welches für die Signierung der Nachrichten und später für die verbindliche Speicherung genutzt wird. Die Abbildung 58 zeigt einen Überblick über die verwendeten Verfahren und die entsprechenden Schlüssellängen. Für die Realisierung der kryptographischen Verfahren wird die Implementation des Bouncy Castle Providers zurückgegriffen. Je nach Sicherheitsanforderung können die Schlüssel länger gewählt werden oder aufwendigere Verfahren verwendet werden. So könnte beispielsweise für die AES-Verschlüsselung statt dem hier gewählten einfacheren ECB (Electronic Codebook) ein CBC (Cipher-Block Chaining) Verfahren (Schneier 1996; Dworkin 2001) eingesetzt werden, das jedoch einen Initialisierungsvektor benötigt und für die hier angestrebte Demonstration („Proof of Concept) nicht notwendig ist.

Verwendungszweck	Algorithmus	Schlüssellänge
Key-Agreement	Diffie-Hellman	1024 Bit
Verschlüsselung	AES/ECB/ PKCS5Padding	128 Bit
Signatur	RSA	1024 Bit
Hashwerte	SHA-1	

Abbildung 58 – Verwendete kryptographische Verfahren

Im Paket org.imvote.muc.xml werden alle für den Kommunikationsablauf des SecureMUC-Plug-Ins benötigten Nachrichten definiert. Dazu gehören die Benachrichtigung über kommende/gehende verifizierte Teilnehmer, Chat-Nachrichten für die Diskussion sowie die Aufforderung das Voting-Plug-In zu starten.

Im Paket org.imvote.muc.view wird die Benutzeroberfläche für die Initiierung des Multi-User-Chat-Plug-Ins, sowie die Verwaltung des Zustands während des Einlade- und Überprüfungsvorgangs implementiert. Dazu gehören auch die Erstellung der Liste der eingeladenen Teilnehmer und der verifizierten Teilnehmer. Beispiele für die implementierten UI-Elemente werden später bei der Erläuterung der Abläufe (Kapitel 14.3.3) gezeigt, um dort die Zuordnung zum jeweiligen Prozessschritt im Verfahren deutlich zu machen.

Unterschiedliche Hilfsklassen, die sowohl vom Paket org.imvote.muc.view als auch vom Paket org.imvote.muc.connection verwendet werden, sind in org.imvote.muc.utils realisiert. Beispiele dafür

sind Klassen zu Ermitteln der eigenen IP-Adresse (AddressParser) oder zum Kodieren der verschlüsselten Daten mit Base64-Kodierung (Base64Coder) um eine Übertragung in ASCII-Zeichensatz zu ermöglichen.

14.3.2 Datenmodell

Die folgende Abbildung 59 zeigt einen Ausschnitt aus dem Datenmodell mit den wichtigsten Klassen für SecureMUC-Plug-Ins.

- Die Klasse SecureMUC implementiert das SecureMUC-Plug-In und übernimmt die Kommunikation mit den anderen Plug-Ins (SecureVC, Voting) sowie die Integration in die Spark-Infrastruktur.
- Ein ConnectionManager verwaltet die Verbindung zu einem Kommunikationspartner. Dazu gehört beispielsweise die Kommunikation beim Diffie-Hellman-Key-Agreement-Verfahren oder die Benachrichtigung an alle verifizierten Teilnehmer, dass ein weiterer Teilnehmer verifiziert wurde und somit jetzt offiziell Teil der Gruppe ist. ConnectionManager ist eine abstrakte Klasse, d.h. sie wird spezialisiert, aber selber nicht als Instanz verwendet. Jeder ConnectionManager kennt die aktuelle Liste der verifizierten Teilnehmer.
- Ein ServerConnectionManager ist der Connection Manager, so wie er vom Initiator verwendet wird. Er kennt zusätzlich alle Teilnehmer, die eingeladen wurden, d.h. auch diejenigen, die noch nicht verifiziert wurden.

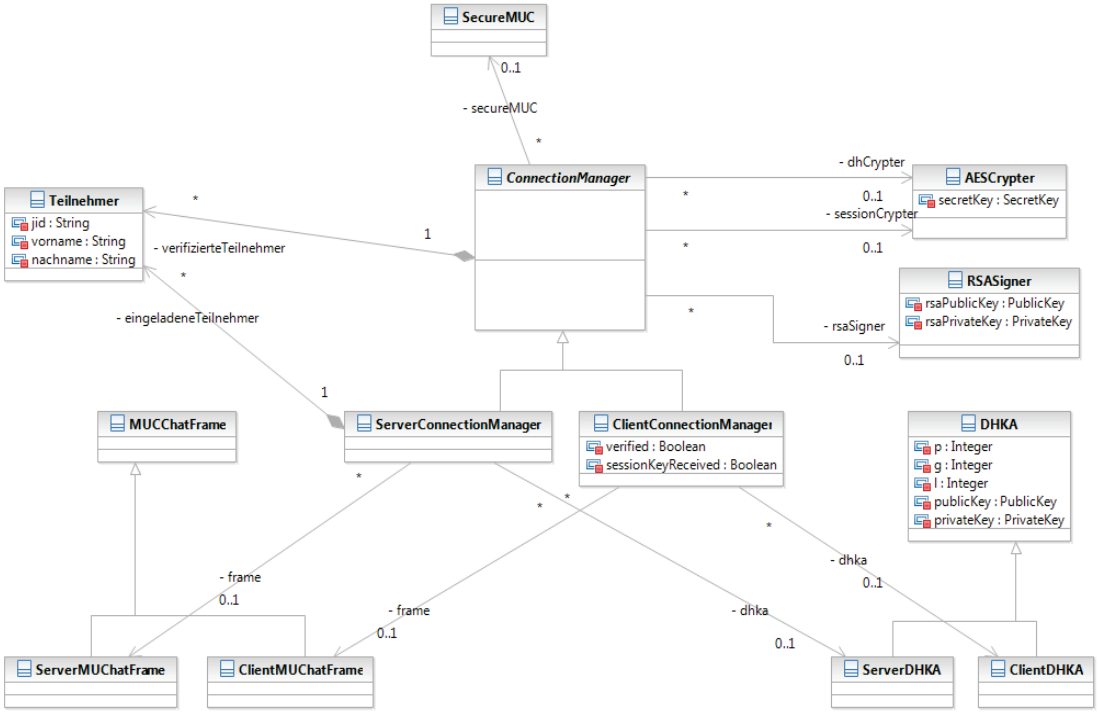

Abbildung 59 – Datenmodell von SecureMUC (Ausschnitt mit Überblick über die wichtigsten Klassen)

- Ein ClientConnectionManager ist der Connection Manager, wie er von anderen Teilnehmern verwendet wird. Ein ClientConnectionManager weiß, ob er bereits verifiziert wurde und ob er schon den Session-Key erhalten hat.
- Zu den verschiedenen Connection-Managern gibt es analog Fenster, welche den aktuellen Zustand (z.B. die Liste eingeladenen und verifizierten Teilnehmer) darstellen und Interaktionselement anbieten (z.B. einen Knopf „Teilnehmer verifizieren“). Diese Funktionen werden von MUChatFrame, ServerMUChatFrame, und ClientMUChatFrame implementiert.
- Eine ähnliche Struktur (d.h. abstrakte Klasse, Spezialisierung für Initiator und Teilnehmer) gibt es für die Klassen, welche das eigentliche Diffie-Hellman-Verfahren implementieren, also nicht die Kommunikation mit dem Gegenüber, sondern die Verwaltung der DH-Parameter $(p, q \ldots)$ und die Berechnung der Schlüssel. Diese Funktionen werden von DHKA, ServerDHKA und ClientDHKA implementiert.
- Die Verschlüsselung von zu übertragenden Daten wird von AESCrypter übernommen, der die AES-Implementation (Advanced Encryption Standard, (Housley 2004)) des Bouncy-Castle-Providers verwendet. Man beachte das AESCrypter in zwei Rollen verwendet wird: Einmal zur Verschlüsselung der paarweisen Verbindung (mit K_{AB}, als ConnectionManager.dhCrypter) und einmal zur Verschlüsselung der Sitzung, also für die Kommunikation innerhalb der ganzen Gruppe (mit K_{ALL}, als ConnectionManager.sessionCrypter).
- RSASigner implementiert die Signierung von Nachrichten mit RSA (Rivest, Shamir und Adleman 1978). Jeder ConnectionManager hat einen RSASigner, der bei der beim Systemstart mit dem Schlüsselpaar des Benutzers initialisiert wird. Der öffentliche Teil dieses Schlüsselpaars wird nach dem Ablauf des Zugangsberechtigungsverfahrens, den anderen Teilnehmern übertragen, damit diese die Signaturen von empfangenen Nachrichten überprüfen können. Außerdem werden diese Schlüssel später beim Verfahren zur verbindlichen Speicherung verwendet.

14.3.3 Ablauf

In diesem Abschnitt werden entsprechend den Abläufen der Gesamtkommunikation und der gewählten Variante des Zugangsberechtigungsmechanismus (vgl. Abschnitt 11.3 Variante „Zentrale Instanz nach Aufforderung“) der Ablauf bei der Nutzung des SecureMUC-Plug-Ins erläutert. Dazu gehören die Implementierung des Diffie-Hellman-Schlüsselaustausch-Verfahrens, die Erzeugung der Schlüssel für Verschlüsselung und Signierung von Nachrichten und die Verteilung dieser Schlüssel. Der Start einer IMVote-Sitzung mit SecureMUC läuft folgendermaßen ab:

*Initialisierung und Verbindungsaufbau:*Der Initiator startet die IMVote-Anwendung. Dabei wird zuerst das SecureMUC-Plug-in gestartet, weil es die Erzeugung und Verteilung von Schlüsseln steuert und den Zugangsberechtigungsmechanismus implementiert. Beim Start von SecureMUC wird auf Seiten des Initiators ein ServerConnectionManager gestartet. Der Initiator wählt aus seiner Kontaktliste die möglichen Teilnehmer einer Sitzung und verschickt an diese eine Einladung über XMPP (SMUCInvitation), welche beim Empfänger im Spark-Client den Start des SecureMUC-Plug-Ins auslöst. Dabei wird auf Seiten des Teilnehmers ein

ClientConnectionManagers gestartet. Dieser ClientConnectionManager des Teilnehmers kontaktiert daraufhin den ServerConnectionManager des Initiators. Dieser überprüft die JID des Clients, um sicherzustellen, dass der anfragende Teilnehmer eine Einladung erhalten hatte (siehe ServerConnectionManager.eingeladeneTeilnehmer in Abbildung 59). Damit wird nur überprüft, ob die JID des Clients auch in der Liste der eingeladenen Teilnehmer vorkommt. Die eigentliche Überprüfung, ob es sich dabei um den richtigen Teilnehmer handelt, wird erst später durch das Zugangsberechtigungsverfahren (siehe Kapitel 14.4) vorgenommen. Der ServerConnectionManager (für den Initiator) und der ClientConnectionManager (für den Teilnehmer) übernehmen die Koordination für das jetzt ablaufende Verfahren und implementieren auf diese Weise das Protokoll zur Schlüsselerzeugung und Überprüfung der Zugangsberechtigung. Der ServerConnectionManager verwaltet dabei für jeden Teilnehmer in der Liste einen separaten Satz von Daten (DH-Parameter, Verifikationsstatus, usw.).

Schlüsselerzeugung und –austausch: Um das Diffie-Hellman-Key-Agreement (DHKA) zu starten, erzeugt der Initiator zunächst die Parameter für das Diffie-Hellmann-Verfahren (ServerDHKA), sowie ein RSA-Schlüsselpaar (in RSASigner). Dann sendet er eine Nachricht (DHKARequest), die seine DH-Parameter, seinen öffentlichen DH-Schlüssel und seinen öffentlichen RSA-Schlüssel enthält. Der Teilnehmer berechnet anhand der empfangenen Daten seinen öffentlichen DH-Schlüssel sowie den gemeinsamen bilateralen DH-Schlüssel, den nur die beiden Kommunikationspartner kennen. Außerdem erzeugt der Teilnehmer selbst ein RSA-Schlüsselpaar für die spätere Signierung sowie die verbindliche Speicherung. Anschließend verschlüsselt der Teilnehmer seinen öffentlichen RSA-Schlüssel mittels AES und sendet alles an den Initiator (als DHKAResponse). Zur späteren Verschlüsselung wird dieser Schlüssel nicht direkt verwendet. Stattdessen leitet SecureMUC aus dem bilateralen DH-Schlüssel einen AES-Schlüssel ab, indem er den DH-Schlüssel zur Initialisierung des Generators für einen AES-Schlüssels verwendet. Dabei wird zur späteren Durchführung der Verschlüsselung ein AESCrypter erzeugt. Sobald das DH-Key-Agreement abgeschlossen ist, kann auch der Initiator diesen Schritt durchführen, so dass bei dann über einen gemeinsamen geheimen symmetrischen AES-Schlüssel für die paarweise Kommunikation verfügen. Damit wird dann der empfangene RSA-Public-Key des Teilnehmers entschlüsselt. Um dem Teilnehmer den erfolgreichen Abschluss des Key-Agreement mitzuteilen, verschlüsselt der Initiator seinen öffentlichen RSA-Schlüssel mit dem bilateralen Schlüssel und sendet ihn an den Client. Der Client entschlüsselt den empfangenen RSA-Public-Key des Initiators mit dem bilateralen Schlüssel. In der Diskussion der algorithmischen Lösung (siehe Kapitel 11.3) wurde dieser bilaterale Schlüssel (dort am Beispiel für zwei Teilnehmer A und B) als K_{AB} bezeichnet. Dort wurde nicht auf das Implementationsdetail eingegangen, dass aus dem DH-Schlüssel ein AES-Schlüssel abgeleitet wird. Daher wurden diese beiden Schlüssel dort nicht unterschieden.

*Ergebnis :*Sowohl der Initiator als auch jeder Teilnehmer verfügen jetzt über einen gemeinsamen bilateralen Schlüssel (ausgehandelt per DH, daraus als AES-Schlüssel abgeleitet) für die jeweilige paarweise Verbindung zum Initiator und jeweils den RSA-Public Key der Gegenseite.

Außerdem können sie den Hashwert des paarweisen DH-Session-Keys berechnen.

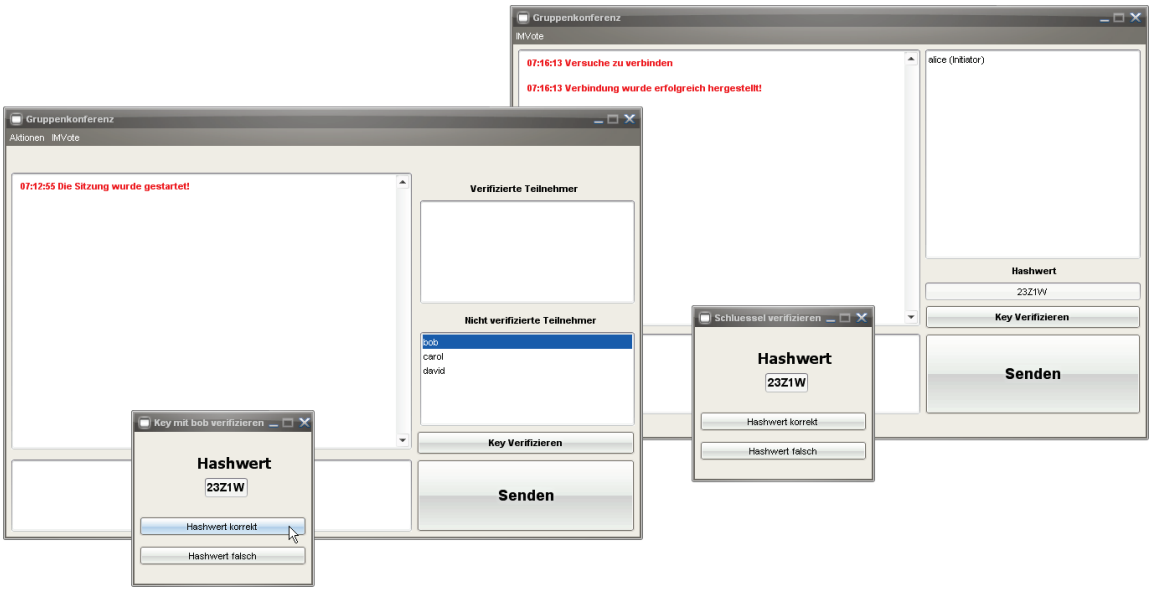

Abbildung 60 – Verifikation der Hash-Werte, links die Benutzeroberfläche für den Initiator, rechts für den normalen Teilnehmer

Diese Hashwerte sind die Basis für die spätere Verifikation des Diffie-Hellman-Schlüssels und indirekt für die Verifikation der Identitäten und der RSA-Signaturschlüssel. Die dabei verwendeten Abläufe werden später bei der Erläuterung des SecureVC-Plug-Ins beschrieben. Abbildung 60 zeigt die Benutzeroberfläche zur Überprüfung der Hashwerte und für den Initiator (links) und einen noch nicht verifizierten Teilnehmer (rechts).

14.4 SecureVC-Plug-In

Während der ersten Phase der Entwicklung von IMVote wurde das SecureVC-Plug-In zunächst als separates Projekt implementiert (Boll 2009) und dann später mit den anderen Plug-Ins integriert. SecureVC etabliert eine sichere Audio-Video-Verbindung zwischen den Teilnehmern, um die Überprüfung der Authentizität der Teilnehmer und Schlüssel zu ermöglichen und eine vertrauliche Konferenz zu gewährleisten. Das Kürzel VC steht als Abkürzung für „Video-Channel“, da in diesem Fall eine Videoübertragung als Medium für die Überprüfung der Authentizität und für die Kommunikation verwendet wird.

14.4.1 Struktureller Aufbau

Die Implementation von SecureVC ist in die Pakete org.imvote.vc, org.imvote.vc.configuration und org.imvote.vc.utils organisiert.

Im Paket org.imvote.vc wird das Plug-In für die Video-Konferenz implementiert (Klasse SecureVC), das die eigentliche Video-Konferenz realisiert. Außerdem werden hier Hilfsklassen zur Verwaltung von Parametern und Einstellungen realisiert.

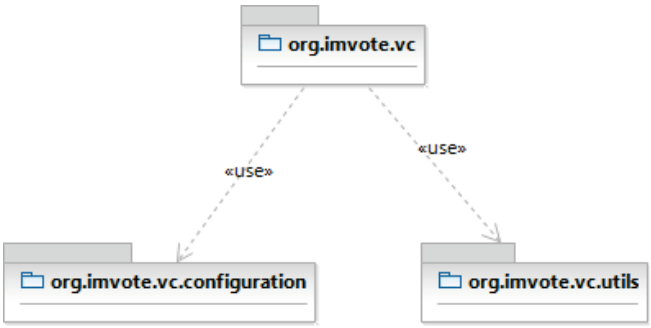

Abbildung 61 – Struktureller Aufbau (SecureVC)

Das Paket org.imvote.vc.configuration ermöglicht dem Benutzer bestimmte Einstellungen selber zu konfigurieren und diese Einstellungen zu speichern.

Im letzten Paket org.imvote.vc.utils wurden unterschiedliche Hilfsklassen, z.B. zum Parsen von JID-Adressen oder zum Ermitteln der öffentlichen IP-Adressen der Teilnehmer implementiert.

14.4.2 Datenmodel

Die folgende Abbildung 62 zeigt einen Überblick über die wichtigsten Klassen im Datenmodell von SecureVC.

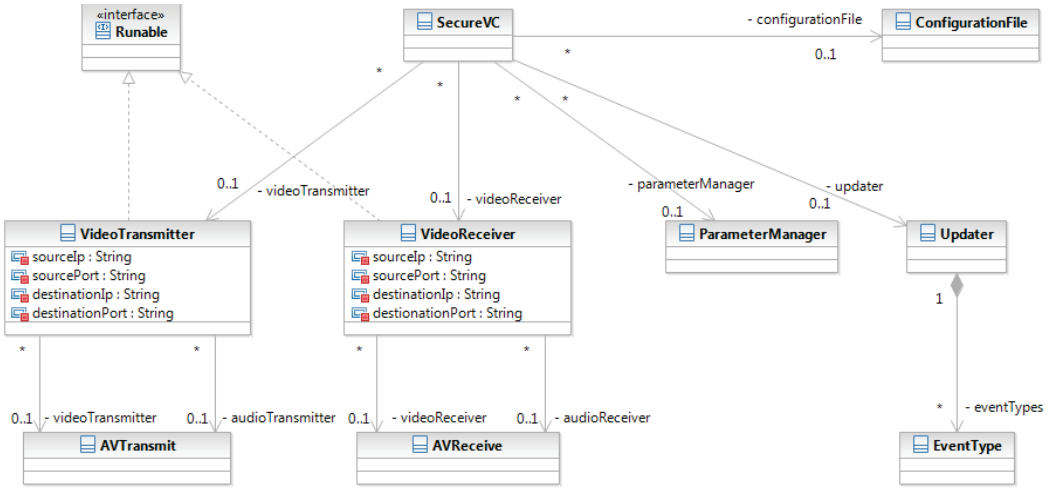

Abbildung 62 – Datenmodell von SecureVC (Ausschnitt mit Überblick über die wichtigsten Klassen)

Diese Klassen werden im Folgenden kurz erläutert:

- Die Klasse SecureVC implementiert das eigentliche Plug-In und übernimmt die Kommunikation mit den anderen Plug-Ins und der Spark-Infrastruktur.
- Die Logik des Ablaufprotokolls und die Übertragung und Verwaltung der Daten werden durch die Klassen VideoTransmitter und VideoReceiver realisiert. Diese implementieren das Interface Runnable und können daher in eigenen Threads ausgeführt werden. Für ihre Aufgaben verwenden sie AVTransmit und AVReceive, die auf dem Java Media Framework basieren. Man beachte, dass es jeweils unterschiedliche Instanzen für den Video- und den Audio-Kanal gibt.

- Das Plug-In bedient sich Hilfsklassen, wie dem ParameterManager. Dieser verwaltet allgemeine Parameter des SecureVC-Plug-Ins wie zum Beispiel IP-Adressen und Port-Nummern, Informationen über den Initiator und eine Liste von Schlüsseln zur Verschlüsselung von Videoverbindungen. Spezielle Parameter, welche für die Verwendung des RTP-Protokolls notwendig sind, werden durch die Klasse VCParameters repräsentiert (nicht in der Abbildung gezeigt).
- ConfigurationFile ermöglicht dem Benutzer Konfigurationseinstellungen zu ändern und zu speichern. Dazu gehören beispielsweise die Portnummer oder ein Video, dass als Ersatz abgespielt wird, wenn keine Webcam initialisiert werden konnte. Diese Funktion wurde zu Testzecken eingerichtet, um die Verifikationsfunktion auch dann testen zu können, wenn nicht genügend Webcams für das gerade durchgespielte Gruppenszenario vorhanden sind.
- Der Updater realisiert einen Aktualisierungsmechanismus realisiert, mit dem das SecureMUC-Plug-In dem SecureVC-Plug-In interne Nachrichten mit angehängten Daten schicken kann. Diese Nachrichten werden *nicht* über das Netzwerk verschickt, sondern als Daten direkt zwischen den Plug-Ins ausgetauscht. Darüber werden dem SecureVC-Plug-In die bilateralen Schlüssel (zur Berechnung des Hashwertes und zur Verschlüsselung während des Zugangsberechtigungsverfahrens) und später nach der Verifikation der gemeinsame Sitzungsschlüssel für die Gruppe (zur Verschlüsselung der Audio-Video-Verbindung) mitgeteilt. Dazu verwendet er eine Liste von EventTypes, ein Überblick findet sich in Abbildung 63.

Kategorie	Event Type	Erläuterung
Server	IsInitiator	Dieser Teilnehmer ist Initiator der Sitzung.
Server	MemberJoin	Ein verifizierter Teilnehmer ist in die Gruppe eingetreten . Wird übermittelt, wenn dieser Teilnehmer schon verifiziert (oder Initiator) ist.
Server	MemberLeave	Ein verifizierter Teilnehmer hat die Gruppe verlassen. Wird übermittelt, wenn dieser Teilnehmer schon verifiziert (oder Initiator) ist.
Server	VerificationStarted	Verifikation gestartet.
Server	VerificationSuccessful	Verifikation erfolgreich. Verschlüsselung auf gemeinsamen Schlüssel der Gruppe umschalten.
Server	EndOfSession	Sitzung wurde für alle Teilnehmer beendet.
Client	IsClient	Dieser Teilnehmer ist einfacher Teilnehmer der Sitzung.
Client	SessionKeyReceived	Verifikation war erfolgreich und der Teilnehmer hat den gemeinsamen Schlüssel der Gruppe. Verschlüsselung entsprechend umschalten.
Client	MemberJoin	Siehe oben
Client	MemberLeave	Siehe oben
Cleint	Disconnected	Verbindung zur Sitzung wurde für diesen Teilnehmer beendet (durch Teilnehmer selbst oder durch den Initiator).

Abbildung 63 – Ereignisse, die von SecureMUC an SecureVC geschickt werden.

14.4.3 Ablauf

In diesem Abschnitt wird der Ablauf bei der Nutzung des SecureVC-Plug-Ins erläutert. Dabei geht es vor allem um die Implementierung des Zugangsberechtigungsverfahrens in Zusammenarbeit mit dem SecureMUC-Plug-In.

Initialisierung und Verbindungsaufbau: Der Initiator sieht in der Benutzeroberfläche eine Liste der eingeladenen, aber noch nicht verifizierten Teilnehmer. Aus dieser Liste wählt der Initiator einen Teilnehmer, um die gegenseitige Überprüfung der Identität durchzuführen. Daraufhin wird an den Teilnehmer eine Einladung zu Videokonferenz verschickt und beim Teilnehmer ein Dialogfenster angezeigt. Bestätigt der Teilnehmer die Einladung, dann bestimmt der Initiator freie Ports für das Senden seiner Daten und den Empfang der Daten des neuen Kontaktes. Diese Portnummern werden zusammen mit der IP-Adresse des Servers in einer `INV_SERVERDATA` Nachricht via XMPP an den Teilnehmer geschickt. Der Client des Teilnehmers übernimmt diese Parameter in seine Verbindungstabelle, errechnet die eigenen Ports für die Verbindung und bestimmt seine eigene IP-Adresse. Diese Informationen schickt er dann mit einer `INV_CLIENTDATA` Nachricht an den Initiator zurück. Dieser bestätigt dem Client die Verbindung mit `INV_APPROVED` und baut auf seiner Seite die RTP-Verbindungen auf. Dabei werden noch einige Schritte durchgeführt, die eine direkte Verbindung ermöglichen, falls sich Initiator und Teilnehmer im gleichen lokalen Netz befinden. Für jede RTP-Verbindung werden jeweils ein eigener Port für jeden Video- und jeden Audiodatenstrom benötigt. Bei einer Konferenz von 4 Teilnehmern benötigt also jeder Teilnehmer 2 Ports zum Senden der eigenen Daten sowie 3×2 Ports für die Streams der anderen Teilnehmer. Als Basisport wird standardmäßig der Port 10200 verwendet, von dort werden die Portnummern in 2-er Schritten erhöht. Dies geschieht damit die jeweils ungeraden Portzahlen für RTCP-Verbindungen (Real Time Control Protocol) genutzt werden können. Zur Verschlüsselung der RTP-Daten wird, wie für die anderen Datenströme auch, AES verwendet.

Überprüfung des Videobilds und Vorlesen des Hashwertes: Über diese etablierten und verschlüsselten Verbindungen werden Audio- und Videostreams übertragen, so dass sich die beiden Teilnehmer gegenseitig im Videobild sehen und über Lautsprecher hören können. Erkennen sich die Teilnehmer über die Audio-Video-Verbindung, lesen sie sich gegenseitig den Hashwert vor, der aus dem bilateralen Schlüssel berechnet wird. Hier können sie sich verständigen, wie die Zeichenfolge vorgelesen wird, z.B. kann jeder einen Teil des Hashwertes vorlesen. Hier ist eine spontane und wechselnde Vorgehensweise sinnvoll, da sie einen Angriff durch vorher vorbereitete manipulierte Videofragmente erschwert (vgl. Kapitel 11.4, 11.5, 11.8). Kommen die beiden Teilnehmer zu dem Schluss, dass die beiden bei Ihnen angezeigten Hashwerte identisch sind und die Videoverbindung keine Anzeichen auf Manipulation (wie z.B. Sprünge im Bild oder einen schlecht synchronisierten Audiostream) aufweist, können die Teilnehmer durch die Betätigung eines Buttons ihre Zustimmung für die weitere Kommunikation geben. Wenn sowohl der Teilnehmer, als auch der Initiator diese Zustimmung erteilen, gilt der Teilnehmer als verifiziert und ein Man-in-the-Middle-Angriff als ausgeschlossen.

Verwaltung der Liste der verifizierten Teilnehmer: Bis die zuvor beschriebene Verifikationsphase abgeschlossen ist, sieht nur der Initiator alle Teilnehmer, die sich im IM-System angemeldet haben und seine Einladung zur Konferenz erhalten haben (siehe ServerConnection-

Manager.eingeladeneTeilnehmer Abbildung 59). Aus Sicherheitsgründen, können alle anderen Teilnehmer nur diejenigen Teilnehmer als sehen, die bereits verifiziert worden sind (siehe ConnectionManager.verifizierteTeilnehmer in Abbildung 59). Nach dem Eintreten jedes neuen Teilnehmers wird ihm der gemeinsame Gruppenschlüssel K_{ALL} durch den Initiator mitgeteilt. Somit kann er Chat-Nachrichten von allen Teilnehmern verschlüsseln und entschlüsseln. Dieser Schlüssel wird ab diesem Zeitpunkt auch auf die Video-Kommunikation angewendet. Dazu wird ein neuer verschlüsselter Kanal aufgebaut, damit sich alle Teilnehmer gegenseitig über diesen verschlüsselten Kanal sehen und hören können.

14.5 Voting-Plug-In

Das Voting-Plug-In implementiert eine Anwendung zur Durchführung von elektronischen Wahlen. Es ist mit den *generischen* Plug-Ins, SecureMUC und SecureVC so integriert, dass diese die Zugangsberechtigung überprüfen und dem anwendungs*spezifischen* Voting-Plug-In die notwendigen Sicherheitselementen zur Verfügung stellen, damit dieses eine spontane und trotzdem sichere Kollaboration realisieren kann. Weiterhin realisiert das Voting-Plug-In die Signier- und Aufbewahrungsfunktionen für die langfristig verbindlichen Abstimmungsergebnisse.

14.5.1 Struktureller Aufbau

Die Realisierung des Voting-Plug-In wird in dem Paket org.imvote.voting implementiert das alle anderen Klassen zur Durchführung von elektronischen Wahlen über IM koordiniert.

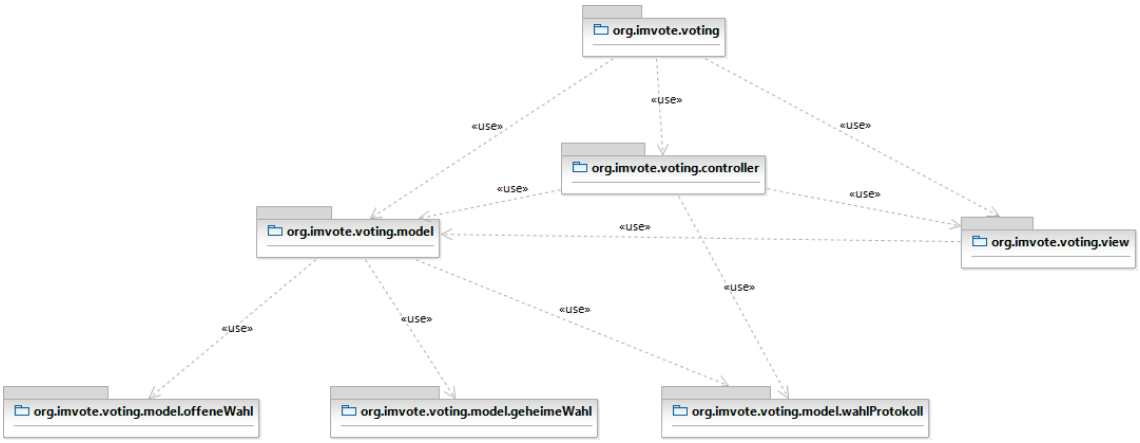

Abbildung 64 – Struktureller Aufbau (E-Voting-Plug-In)

org.imvote.voting.model enthält alle Klassen, die anwendungsspezifische Mechanismen realisieren und bildet dazu Konzepte aus dem betrachteten Realitätsausschnitt (Wahlen) nach. Dieses Package entspricht dem „Model“ im Model-View-Controller-Pattern. Es enthält beispielsweise die Klassen Wahl, Wahlzettel, Votum und Ergebnis.

Die Datenobjekte sind dabei erstens als JavaBeans (Sun Microsystems 2008) realisiert (um eine Bindung an UI-Elemente zur ermöglichen) und zweitens als Spezialisierungen der Klasse IQ realisiert (um eine Übertragung mittels IM-Protokollen und eine Integration in das Spark/Smack-Framework (Ignite Realtime Community 2009a) zur ermöglichen).

Dieses Subpackage org.imvote.voting.model.offeneWahl enthält diejenigen Klassen, die spezifisch für eine offene Wahl sind. Dabei werden teilweise Klassen aus org.imvote.voting.model spezialisiert, zum Beispiel Votum durch OffenesVotum.

Analog enthält das Subpackage org.imvote.voting.model.geheimeWahl Klassen, die spezifisch für eine geheime Wahl sind (z.B. GeheimesVotum). An dieser Stelle werden die algorithmischen Konzepte des gewählten Voting-Protokolls evote (siehe Kapitel 13.5, (Alkassar, Krimmer und Volkamer 2005)) als Java-Klassen implementiert (SignierElement, VertauschElement, usw.).

Das Subpackage org.imvote.voting.model.wahlProtokoll enthält Klassen zur Implementierung des dynamischen Ablaufs des eigentlichen Wahlprotokolls. Dabei geht es vor allem um die Integration des Zustandsmaschine (engl. „Finite State Machine“, FSM), die teilweise mit dem Unimod-Framework realisiert wurde. Die Koordination übernimmt der WahlFsmManager. Außerdem werden von Unimod benötigte Klassen bereitgestellt (WahlEventProvider, WahlControlledObjects-Manager) und Klassen zur Aktualisierung der Benutzeroberfläche bei Änderungen des Zustands implementiert (ControlledObjectWithDerivedProperties).

Die Klassen im Package org.imvote.voting.controller koordinieren das Model (z.B. eine Wahl) und die zugehörige Benutzeroberfläche (z.B. WahlFrame).

Die Benutzeroberfläche wird durch das Package org.imvote.voting.view realisiert. Diese besteht aus einem WahlFrame, der ein Fenster anzeigt sowie viele darin enthaltene Panels, die jeweils eine Ansicht auf einem Teil der anwendungsspezifischen Daten bereitstellen. So gibt es beispielsweise WahlzettelPanel, SchluesselbundPanel und ErgebnisPanel. Das besondere ZustandPanel gibt den Zustand der Zustandsmaschine – und somit den Zustand des Wahlprotokolls – wieder.

14.5.2 Datenmodel

Das Package org.imvote.model stellt die anwendungsspezifischen Konzepte als implementierte Klassen bereit. An dieser Stelle soll ein kurzer Überblick über die Strukturen gegeben werden, die zwischen diesen Klassen bestehen, siehe dazu das Datenmodell in Abbildung 65 in UML-Notation.

Alle Klassen sind Spezialisierungen von BenanntesElement zur besseren Übersichtlichkeit sind nicht alle diese Generalisierungsbeziehungen dargestellt. So ist z.B. auch ein WahlZettel ein BenanntesElement, das wird aber in der Abbildung nicht gezeigt. BenanntesElement ist eine Spezialisierung der Klasse IQ aus der Smack-Klassenbibliothek von Spark. Instanzen dieser Klasse können als Nachrichten via XMPP über das Instant-Messaging-Netzwerk verschickt werden. Da alle anderen anwendungsspezifischen Klassen wiederum von BenanntesElement abgeleitet sind, kann daher die XMPP-Infrastruktur als Grundlage für die anwendungsspezifische Kommunikation verwendet werden, beispielsweise zur Signalisierung des Wahlstarts (WahlStart) oder für das Übermitteln von Voten (Votum, GeheimesVotum).

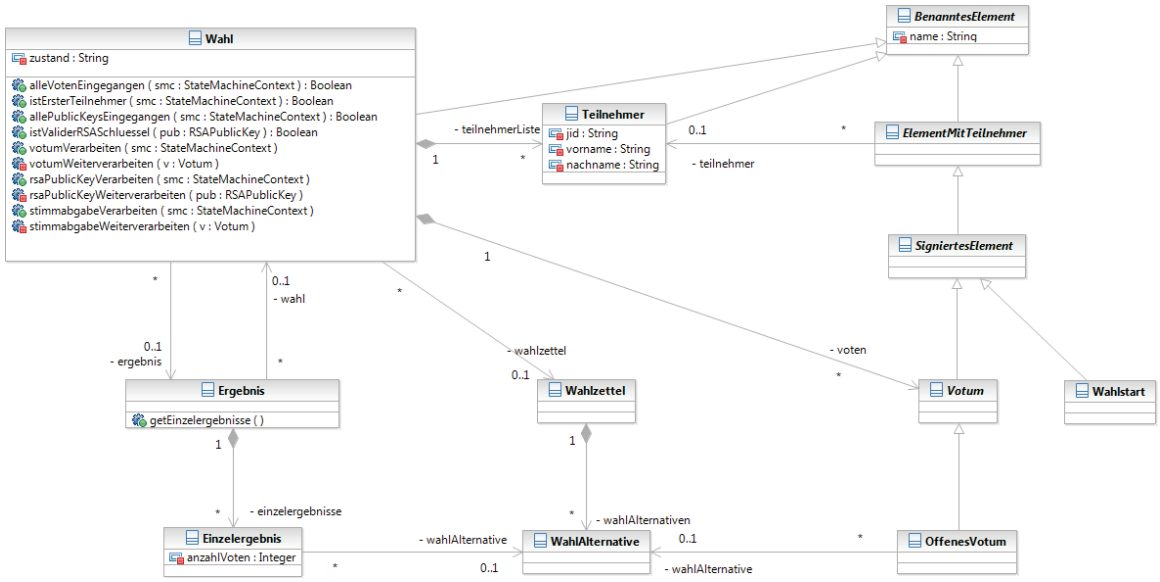

Abbildung 65 – Überblick über Anwendungsdaten des E-Voting-Plug-Ins

Alle Daten, die einem Teilnehmer zugeordnet werden können, sind Spezialisierungen von ElementMitTeilnehmer. Alle Daten, die signiert werden können sind Spezialisierungen von SigniertesElement. SigniertesElement und jede davon abgeleitete Klasse besitzt die Fähigkeit, den eigenen Inhalt als XML bereitzustellen und mit dem privaten Schlüssel des zugeordneten Teilnehmers zu signieren. Das Ganze, d.h. die signierte Daten und die zugehörige Signatur, wird dann als ein XML-Block bereitgestellt. Auf diese Weise kann der Empfänger die Authentizität der eingehenden Nachrichten überprüfen.

```
<offenesVotum xmlns="http://imvote.org/imvote/offenesVotum">
    <encapsulatingSignature>

        <signedInfo>
            <name>OffenesVotum von alice@ibm</name>
            <teilnehmer>alice@ibm</teilnehmer>
            <wahlAlternative>Peter Meier</wahlAlternative>
        </signedInfo>

        <signature>RspOvdUYsXRRU+8efZTW5BssBHQJ/HHSi7PVn7KH
        DQqcbiH8lWL1m6H4B/FTjlWX3uEnmjB6N9tevnrUFE+EnrqkeCy
        8dGpn+/iIHqausscFpLGQ772PcaHv9f5RXnhcilgEO0tBM9Gl9F
        zTF6ADA5jhKYTHSB1bFksZJoiTHFk=</signature>

    </encapsulatingSignature>
</offenesVotum>
```

Abbildung 66 – Encapsulating Signature, wie von der Klasse SigniertesElement erzeugt

Eine Wahl ist das wichtigste und daher auch das komplexeste Objekte bei den anwendungsspezifischen Daten. Eine Wahl enthält eine Liste von Teilnehmern, einen Wahlzettel, eine Liste von eingegangenen Voten und ein Ergebnis. Ein Wahlzettel enthält eine Liste von WahlAlternativen.

Es gibt verschiedene Formen von Voten, hier ist ein OffenesVotum gezeigt. Es gibt aber auch GeheimeVoten. Ein Votum ist (als Spezialisierung von ElementMitTeilnehmer) einem Teilnehmer zugeordnet und referenziert eine WahlAlternative aus dem Wahlzettel.

Das Ergebnis besteht aus vielen Einzelergebnissen. Ein Einzelergebnis gibt eine Anzahl der Voten an und referenziert eine WahlAlternative (z.B. 3 x „Merkel“). Es Ergebnis sollte genau ein

Einzelergebnis zu jeder WahlAlternative im Wahlzettel enthalten. Dies wird bei der Erzeugung des leeren Ergebnisses, vor dem Eintreffen der Voten sichergestellt. Es kann also Einzelergebnisse mit einer Null als Anzahl der Voten geben.

14.5.3 Ablauf

Zur Realisierung des Wahlprotokolls, wurden dessen Abläufe mit dem Unimod-Framework als Zustandsmaschine modelliert. Unimod verwendet zur Integration der Zustandsmaschine mit der restlichen Anwendung ein sogenanntes Connectivity Diagram (siehe Abbildung 67).

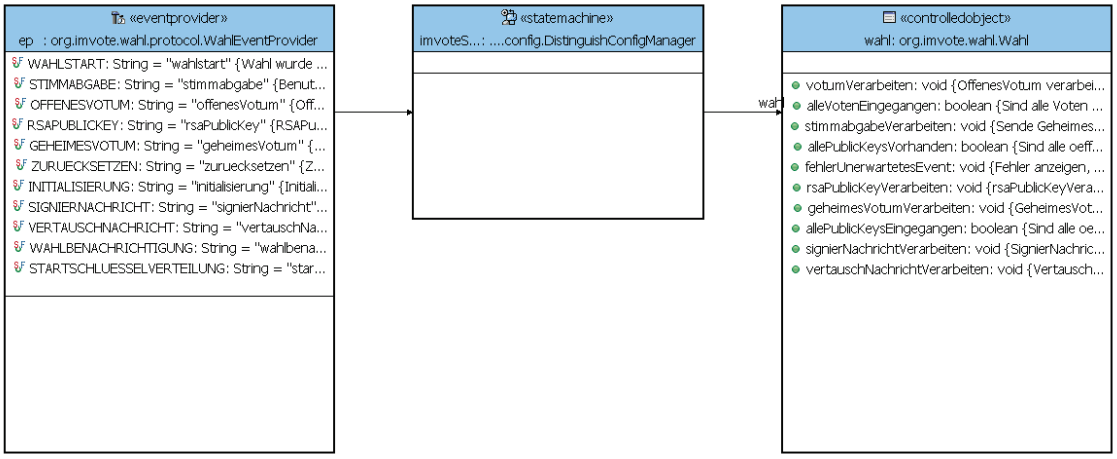

Abbildung 67 – Connectivity Diagram der Zustandsmaschine zur Definition des Wahlprotokolls

In diesem Connectivity Diagramm wird die Zustandsmaschine (Mitte von Abbildung 67) den zugehörigen Objekten zugeordnet. Das gehören EventProvider, die Ereignisse liefern (hier WahlEventProvider, links) und kontrollierte Objekte, deren Methoden verwendet werden um Bedingungen zu überprüfen und Aktionen auszuführen (hier Wahl, rechts).

Das eigentliche Wahlprotokoll wird in einem separaten Diagramm definiert, das typische Elemente für Zustandsmaschinen enthält (Abbildung 68). Ein abgerundetes Rechteck stellt einen Zustand dar, (z.B. InVorbereitung). Pfeile zwischen Zuständen stellen Transitionen dar. Diese sind Beschriftungen mit Ereignis[Bedingung]/Aktion.

Die Integration der Zustandsmaschine mit dem Rest der Anwendung funktioniert folgendermaßen: Man beachte, dass Bedingungen und Aktionen als Objektmethoden der kontrollierten Objekte beschrieben sind. So ruft beispielsweise die Bedingung wahl.allePublicKeysVorhanden eine Funktion der Klasse Wahl auf, die überprüft ob alle öffentlichen Schlüssel vorhanden sind. In ähnlicher Weise werden während dem „Feuern“ von Transitionen Aktionen ausgeführt, z.B. wahl.stimmAbgabeVerarbeiten als Aufruf der entsprechenden Methode des Wahl-Objektes.

Die Integration der Zustandsmaschine mit der Spark-Infrastruktur erfolgt in zwei Richtungen: Erstens werden eintreffende XMPP-Pakete als Ereignisse an die Zustandsmaschine gegeben (z.B. vertauschNachricht). Zweitens werden innerhalb der Aktionen, welche die Zustandsmaschine aufruft, wiederum neue XMPP-Pakete versendet (z.B. innerhalb von wahl.vertauschNachrichtVerarbeiten).

Nach dem Start geht die Zustandsmaschine in den Zustand Initialisierung. Dieser Zustand wurde eingeführt, damit der erste Zustand einen Namen hat, der in der Benutzeroberfläche angezeigt werden kann. Außerdem wird hier eine Fallunterscheidung zwischen zwei Varianten des Wahlprotokolls durchgeführt. Im ersten Fall (allePublicKeysVorhanden=false) übernimmt das Voting-Plug-in selbst die Schlüsselverteilung (Zustand Schluesselverteilung). Dieser erste Fall wird benötigt, wenn das Voting-Plug-In ohne SecureMUC betrieben werden soll, die Schlüssel sind dann nicht über den Zugangsberechtigungsmechanismus abgesichert (dies war für Testzwecken erforderlich).

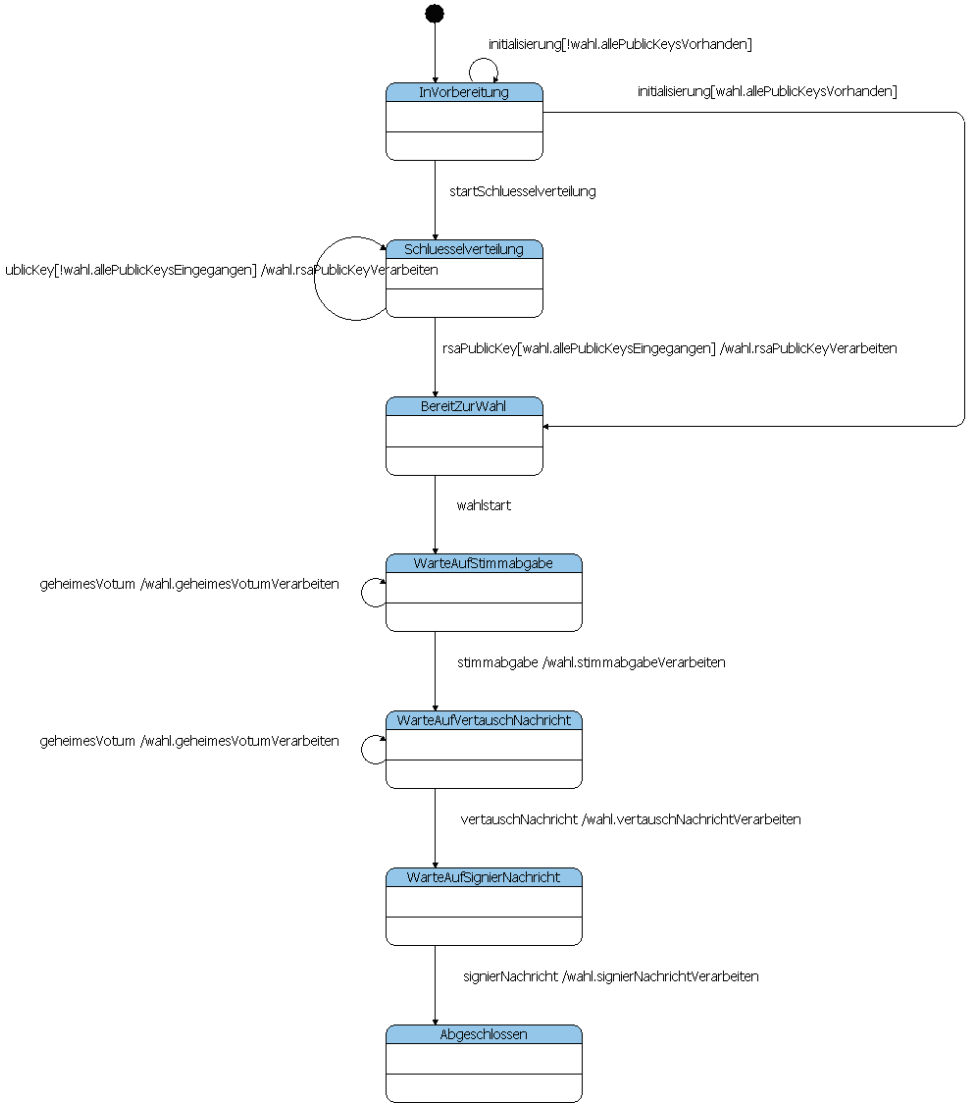

Abbildung 68 – Zustandsmaschine zur Definition des Wahlprotokolls

Im zweiten Fall (allePublicKeysVorhanden=true) überlässt das Voting-Plug-in die Schlüsselverteilung dem SecureMUC und verwendet die Schlüssel aus dem video-gestützten Zugangsberechtigungsverfahren. Es springt dann direkt in den Zustand BereitZurWahl. In der Abbildung 69 werden die entsprechenden Schlüssel dargestellt.

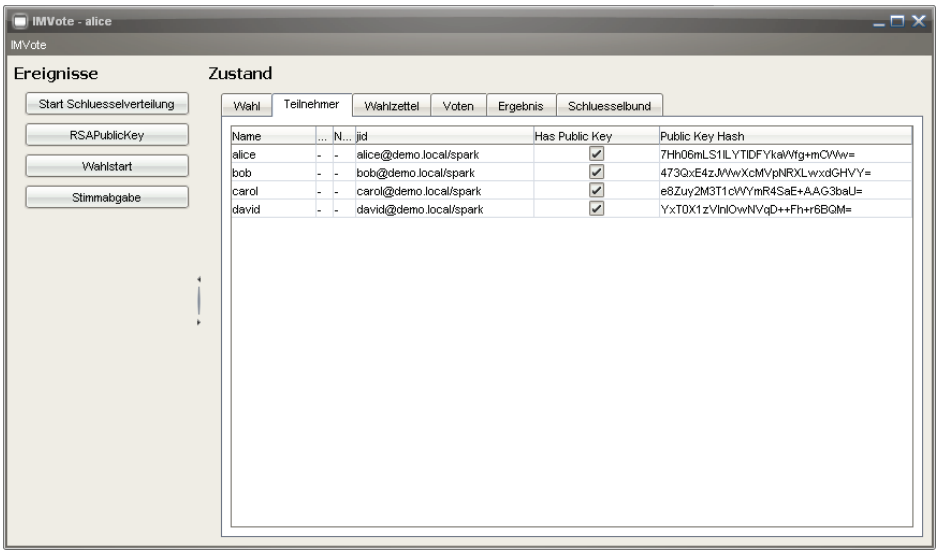

Abbildung 69 – IMVote Schlüsselverteilung

Sobald die Wahl beginnt wartet das Voting-Plug-in auf die Stimmabgabe durch den Benutzer (WarteAufStimmabgabe), vgl. Abbildung 70.

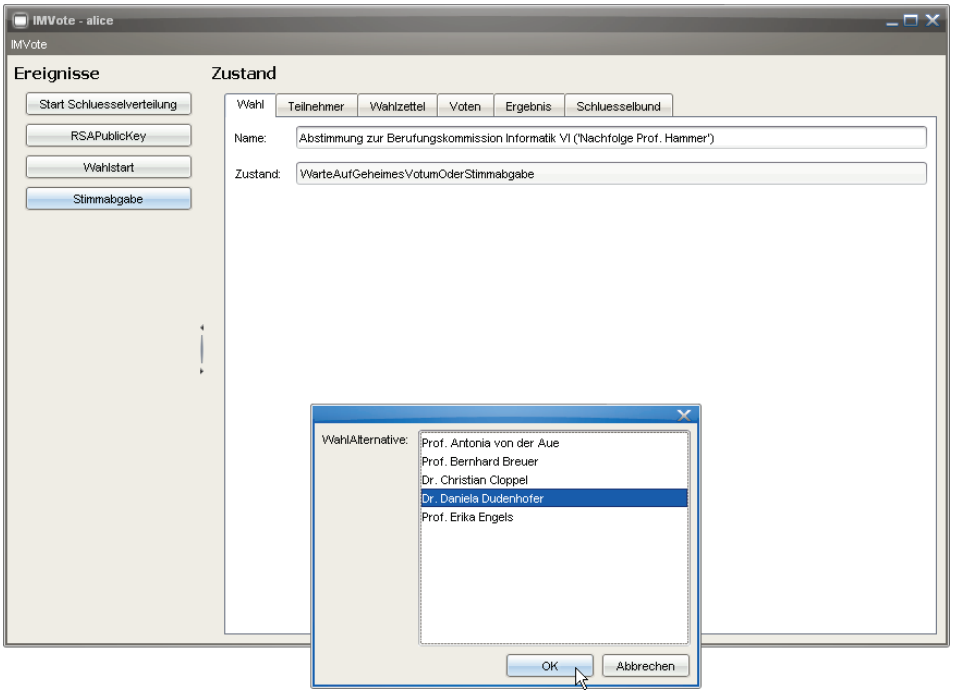

Abbildung 70 – IMVote Stimmabgabe

Nachdem der Benutzer seine Stimme abgegeben hat beginnt die Umsetzung des evote-Protokolls in zwei Phasen, der Vertauschphase (WarteAufVertauschNachricht) und der Signierphase (WarteAufSignierNachricht). Nachdem diese durchgeführt wurden, springt die Zustandsmaschine in den Zustand Abgeschlossen.

Nachdem eine Wahl abgeschlossen und das Ergebnis berechnet wurde, kann dieses Ergebnis entsprechend den algorithmischen Lösungsansätzen langfristig aufbewahrt werden. Dafür wurde im Abschnitt 11.12 das Konzept der Verbindlichkeit mit der Variante –„drei Personen, drei

Rollen“ ausgewählt. An dieser Stelle werden nun technische Mechanismen skizziert mit denen diese Konzepte in einer Anwendung umgesetzt werden können. Als „Proof-of-Concept“ wurden diese in der prototypischen Realisierung teilweise umgesetzt.

Die dafür benötigten Rollen Signierer, Zertifikat-Ersteller und Aufbewahrer werden zufällig vergeben. Jeder Teilnehmer kann prinzipiell jede Rolle übernehmen. Der Zertifikat-Ersteller erstellt ein Zertifikat für den Signierer (und speichert dieses Zertifikat). Der Signierer signiert Ergebnis und übersendet es an den Aufbewahrer. Der Aufbewahrer speichert das Ergebnis.

Das zu speichernde Sitzungsergebnis besteht mindestens aus

- der Liste aller Teilnehmer der Sitzung
- einem aufgezeichneten Protokoll aller anwendungsspezifischen, ergebnis-relevanten Nachrichten (hier die Voten)
- dem aggregierten Ergebnis, dass sich daraus ergibt (hier das Wahlergebnis)

Das Sitzungsergebnis könnte zusätzlich enthalten

- ein aufgezeichnetes Protokoll aller Chat-Nachrichten (ausgetauschte Diskussionsbeiträge)
- aufgezeichnete Audio-Video-Streams

In der prototypischen Realisierung wurde die Speicherung der Teilnehmerliste, der Voten und des Wahlergebnisses realisiert. Während die verbindliche Speicherung der Chat-Nachrichten prinzipiell mit ähnlichem Aufwand möglich wäre, wie für die oben genannten Daten (Teilnehmerliste, Voten, Wahlergebnis), ist die Speicherung und Vergleich von Video-Strömen nicht trivial. Wenn Video-Ströme über das Netz übertragen werden (mit potentiellen Verlusten von Frames) und dabei komprimiert/dekomprimiert werden (mit verlustbehafteter Kompression und potentiell unterschiedlichen Kompressionsparametern auf den verschiedenen Rechnern) ist es schwierig sicherzustellen, dass die gespeicherten Ergebnisse identisch sind.

Der Ablauf für die Realisierung einer verbindlichen Speicherung der Sitzungsergebnisse sieht folgendermaßen aus:

- Während der Sitzung zeichnen alle Teilnehmer die ausgetauschten Informationen auf und können selbst jederzeit das Ergebnis nachvollziehen.
- Nach Abschluss der Wahl wird die Zuordnung der Rollen durch die Anwendung zufällig vorgegeben. Falls möglich, d.h. wenn die Gruppe groß genug ist, werden die Rollen an unterschiedliche Teilnehmer vergeben. Diese Rollenverteilung wird allen Teilnehmern mitgeteilt.
- Der Zertifikat-Ersteller erzeugt ein Zertifikat, in dem er den öffentlichen Schlüssel des Signierers mit seinem privaten Schlüssel signiert und daraus ein X.509 Zertifikat (X509Certificate) erzeugt. Es werden dabei diejenigen Schlüssel verwendet, die während der Etablierung der Sitzung durch das SecureMUC Plug-In sowieso erzeugt wurden (siehe Kapitel 14.4). Der Zertifikat-Ersteller sendet dieses Zertifikat erstens als XMPP-Nachricht an den Signierer und zweitens speichert er das Zertifikat für eine eventuelle spätere Überprüfung.
- Der Signierer bildet das Ergebnis (bestehend aus Teilnehmerliste, Voten und Wahlergebnis), packt dieses als Datenstruktur (Sitzungsergebnis) zusammen, signiert dies mit seinem

privaten Schlüssel (passend zum Zertifikat) und schickt es als XMPP-Nachricht an den Aufbewahrer.

- Der Aufbewahrer, bildet ebenfalls aus den ihm vorliegenden Daten der Sitzung, auf die gleiche Weise das Ergebnis. Dieses kann er dann mit dem vom Signierer empfangenen Ergebnis vergleichen. Außerdem überprüft er die Signatur des Ergebnisses. Entscheidet er, dass das Ergebnis korrekt ist, so speichert er dies (zusammen mit dem erhaltenen Zertifikat) für eine spätere Überprüfung durch einen Anfrager.

Ein potentieller Anfrager kann sich zur ersten Überprüfung vom Aufbewahrer das Ergebnis sowie das am gleichen Ort gespeicherte Zertifikat vorlegen lassen. Damit ist jedoch nur eine grundlegende Überprüfung des Ergebnisses auf Veränderungen, z.B. durch technische Fehler oder Unachtsamkeiten, möglich. Ein höheres Sicherheitsniveau lässt sich erreichen, wenn der Anfrager nicht das beim Aufbewahrer gespeicherte Zertifikat verwendet, sondern stattdessen das Zertifikat vom Zertifikat-Ersteller abfragt. Nur so lässt sich beispielsweise ausschließen, dass der Aufbewahrer das Ergebnis manipuliert und ein dazu passendes gefälschtes Zertifikat erzeugt. Siehe hierzu auch die Diskussion in Kapitel 11.13 „Bedrohungen der langfristigen Verbindlichkeit“.

Teil IV Diskussion und Ausblick

15 Bearbeiteter Forschungsgegenstand

Der fokussierte Gegenstand der vorliegenden Arbeit ist die Nutzung von Instant-Messaging-Systemen als Plattform für sichere, kollaborative Anwendungen über Unternehmensgrenzen hinweg.

Die charakteristischen Eigenschaften solcher IM-Systeme, wie die einfache Installation und Konfiguration, die schnelle Registrierung, eine flexible Nutzung unabhängig vom Aufenthaltsort in Echtzeit, das Angebot unterschiedlicher Kommunikationskanäle (Chat, Audio, Video usw.) und die Möglichkeit, gleichzeitig mit mehreren Teilnehmern Kontakt aufnehmen zu können, bieten vielfältige Alternativen für eine kostengünstige und effiziente Kommunikation und Kollaboration. Da Instant-Messaging-Systeme ursprünglich für den Austausch spontaner einfacher Textnachrichten konzipiert wurden, lag die sichere Übertragung der übermittelten Nachrichten nicht im Fokus der Entwicklung. Allerdings sind für die Unterstützung einer Kollaboration, in der mit sensiblen Daten gearbeitet wird, Sicherheitsziele wie die Authentizität der Gesprächspartner, die Integrität der Kommunikationsinhalte, die Vertraulichkeit einer Gruppenkommunikation und die Verbindlichkeit der Kommunikationsergebnisse zu fordern. Dabei liegt die **Herausforderung** der hier geführten Betrachtungen in dem (scheinbaren?) Widerspruch zwischen einer spontanen, kurzzeitigen Kommunikation und deren Sicherheit. Daher wurde am Anfang der Arbeit die zentrale **Fragestellung** formuliert:

„*Wie kann man die grundlegenden Eigenschaften von Instant-Messaging-Systemen so mit etablierten Verfahren der IT-Sicherheit kombinieren, dass man dadurch erstens eine spontane Kollaboration kurzfristig absichern kann und dass zweitens Ergebnisse dieser spontanen Kollaboration eine langfristige Verbindlichkeit haben?*“

15.1 Diskussion der Vorgehensweise und Zwischenergebnisse

Die Beantwortung dieser Fragestellung hat sich an der Methode des „Design Researchs“ orientiert. Dabei wurde in **vier Schritten** vorgegangen.

Zunächst wurde **(1) der Forschungsbereich der modernen Instant-Messaging-Systeme** strukturiert sowie eine Analyse von Instant-Messaging-Systemen und eine Identifizierung der existierenden und fehlenden IT-Sicherheitsmechanismen vorgenommen (vgl. Kapitel 7 und Kapitel 8). Die Ergebnisse dieser Analyse wurden mit Hilfe eines Featuremodells dokumentiert und erläutert.

Der Mehrwert, den diese Analyse erbracht hat, war die Erkenntnis, dass existierende Ansätze wie ZRTP (vgl. Abschnitt 8.2.1) oder OTR (vgl. 8.2.3) zwar eine Teilmenge der hier definierten Anforderungen (z. B. Vertraulichkeit oder Authentizität) oder darüber hinaus weitere (hier nicht erforderliche) Eigenschaften wie die gewünschte Abstreitbarkeit von Aussagen bereits realisiert

haben. Für die hier angestrebte Zielsetzung greifen die existierenden Ansätze allerdings zu kurz, da sich die bereitgestellten Techniken entweder nur für einen Kommunikationskanal (Audio/Video *oder* Chat) eignen oder nur eine bestimmte Anzahl von Kommunikationspartnern bedienen können. Die sichere Nutzung der Systeme in größeren Gruppen oder die Gewährleistung einer langfristigen Verbindlichkeit der Kollaborationsergebnisse wurde von den existierenden Ansätzen nicht untersucht. Darüber hinaus hat keine dieser Arbeiten die Erweiterung von IM-Systemen als eine Plattform für sichere, kollaborative Anwendungen fokussiert.

Ein wichtiger Beitrag dieser Arbeit war somit **(2) die Entwicklung von algorithmischen Lösungen** für eine authentifizierte, verbindliche Kollaboration (Kapitel 11). Das Ziel war dabei, nicht nur alle existierenden Kommunikationskanäle eines IM-Systems (Chat, Video usw.) abzusichern, sondern diese zusätzlich mit Mechanismen auszurüsten, die einen **sicheren kollaborativen Einsatz von Anwendungen** ermöglichen, die auf das **IM-System als Plattform** aufsetzen. Dafür wurde ein **Zugangsberechtigungsmechanismus** entworfen, welcher, basierend auf spontan erzeugten Sicherheitselementen (Schlüsseln, Zertifikaten), die mit Hilfe eines Diffie-Hellman-Key-Agreements erzeugt werden, die Vertraulichkeit der übertragenen Daten (unabhängig vom Kommunikationskanal) und die Echtheit der Identitäten und der entsprechenden generierten Sicherheitselemente gewährleistet (gemäß den Sicherheitsanforderungen Authentizität und Vertraulichkeit). Diese Überprüfung wird durch die visuelle Erkennung der Gesprächspartner über den Videokanal und das Vorlesen eines Hashwertes der generierten Schlüssel realisiert. Dazu wurden unterschiedliche Varianten diskutiert, wie innerhalb einer Gruppe die Schlüsselgenerierung und -verteilung stattfinden kann. Die Auswahl der geeigneten Variante hängt von der konkreten Anwendung ab, da hierdurch das gewünschte Sicherheitsniveau und Vertrauensmodell vorgegeben sind.

Ein weiteres Ziel war die Realisierung einer gespeicherten und langfristig **verbindlichen Kommunikation** (Sicherheitsanforderung Verbindlichkeit). Dafür wurde ein „Separation of Duty"-Konzept entworfen, das sich abermals an den Abläufen einer herkömmlichen Kollaboration orientiert. Möchte man das Ergebnis einer Kollaboration langfristig aufbewahren, muss es eine Instanz geben, welche diese Aufbewahrung übernimmt und als Ansprechpartner fungiert, wenn das Ergebnis zu einem späteren Zeitpunkt überprüft werden soll. Soll das Ergebnis darüber hinaus auch verbindlich gespeichert werden, muss es von einer anerkannten Stelle unterschrieben werden. Da der hier präsentierte Ansatz auf eine spontane und kurzzeitige Kollaboration zielt, kann diese Aufgabe eben *nicht* von einer zentralen Stelle wie einer PKI (Public Key Infrastructure) übernommen werden. Um die angestrebten Anforderungen trotzdem erreichen zu können, werden die Rollen „Aufbewahrer", „Signierer" und „Zertifikat-Ersteller" eingeführt. Diese können am Anfang einer Kollaboration zufällig einzelnen Teilnehmern der Gruppe zugeordnet werden. Ähnlich wie bei dem Ansatz der Authentizität und Vertraulichkeit werden hier unterschiedliche Varianten zur Abwägung (Trade-off) von Einfachheit und Spontaneität gegenüber dem gewünschten Sicherheitsniveau und Vertrauensmodell vorgestellt.

In einem weiteren Schritt **(3)** wurde eine **allgemeine IM-Plattform** für sichere kollaborative Anwendungen skizziert und ihre wichtigsten Komponenten identifiziert, die bei der Erweiterung eines IM-Systems um konkrete Sicherheitsmechanismen und Anwendungsfunktionalität angepasst werden müssen. Die generischen Konzepte dieser Plattform wurden anschließend für ei-

nen bestimmten Typ von Anwendung, Entscheidungsprozesse samt Wahlphase, spezialisiert und weiter verfeinert.

Der letzte Schritt der Arbeit beinhaltet **(4)** einen **„Proof of Concept“** durch Konzeption und **prototypische Realisierung** einer Anwendung für **„Entscheidungsprozesse und Wahlen“**. Entscheidungsprozesse schließen im Rahmen der vorliegenden Arbeit zwei Phasen ein: eine Diskussion und eine elektronische Wahl (Abstimmung). In diesen Phasen sollten als Teilnehmer nur berechtigte Personen agieren; die Diskussion sollte offen unter den Mitgliedern ablaufen, während die elektronische Wahl sowohl offen als auch geheim stattfinden kann. Beide Phasen sollten Dritten gegenüber vertraulich ablaufen. Die Diskussion kann sowohl über den Chat-Kanal, durch die Eingabe von Textnachrichten, als auch über den Audio-Video-Kanal erfolgen. Für die elektronische Wahl in der konkreten prototypischen Umsetzung wurde ein Wahlprotokoll gewählt, das jedem Teilnehmer einer Gruppe erlaubt, das Ergebnis selbst zu berechnen. Es wird keine zentrale Wahlkommission benötigt. Dieses Protokoll erwies sich besonders gut geeignet, um reelle Gremiumssitzungen in kleinen Gruppen nachzubilden.

15.2 Diskussion der Ergebnisse in Bezug auf Vorteile und Einschränkungen

Durch die entworfenen Sicherheitsmechanismen und die Integration mit der sicherheitskritischen Applikation „elektronische Wahlen“ ergeben sich folgende **Vorteile**:

Die Teilnehmer des Gremiums können verteilt von unterschiedlichen Orten aus kommunizieren und kollaborieren. Die Teilnehmer können schnell die Anwendung installieren und ohne große Konfigurationsanpassungen unter Gewährleistung der definierten Sicherheitsanforderungen zusammenarbeiten. Durch den Zugangsberechtigungsmechanismus werden Schlüssel für die Kommunikation (sowohl für die Verschlüsselung der Chat-Nachrichten als auch des Audio-Video-Kanals) und auch für die Signierung der Chat-Nachrichten erzeugt. Durch das Vorlesen der Hashwerte der DH-Schlüssel über den Videokanal werden sowohl die Echtheit der Identität der Teilnehmer als auch die Echtheit der dabei verwendeten und übertragenen Sicherheitselemente (Schlüssel und Zertifikate) garantiert. Diese Schlüssel werden auch für die Anwendung (elektronische Wahl) verwendet, um die Integrität der anwendungsspezifischen Daten (z. B. der Voten) zu schützen. Alle diese Sicherheitselemente werden spontan und kurzfristig generiert. Eine zentrale PKI mit den damit verbundenen Nachteilen (Administrationsaufwand, Spontaneitätsverlust) wird nicht benötigt. Weitere Sicherheitsanforderungen (wie die Anonymität) werden durch das ausgewählte Protokoll realisiert, welches auf dem Mix-Verfahren von Chaum basiert (Chaum 1981).

Zur Überprüfung der entwickelten Konzepte wurde der Open-Source-IM-Client Spark mit dem entsprechenden Server Openfire verwendet. Von diesen Projekten wurde die Infrastruktur zur Anmeldung eines Benutzers beim IM-Server und zum Aufbau von Multi-Chat-Konferenzen genutzt. Anschließend wurden Komponenten, die von Spark bereitgestellt werden, erweitert oder durch eigene Implementationen ersetzt, um die hier konzipierten Verfahren zu realisieren (siehe Kapitel 14). Dazu gehören beispielsweise der DH-Schlüsselaustausch und die Versendung von Schlüsseln und Zertifikaten. Befinden sich alle Teilnehmer in einer Konferenz, haben

sie die Möglichkeit, vertraulich über Chat oder Audio-Video zu kommunizieren und anschließend eine Wahl zu starten.

Das dann ablaufende Wahlprotokoll wurde nicht manuell kodiert, sondern zunächst als Zustandsmaschine modelliert, welche die unterschiedlichen Zustände einer Wahl (Schlüsselverteilung, Stimmabgabe, ...) sowie Ereignisse (GeheimesVotumEingegangen, ...) und entsprechende Reaktionen beschreibt. Dieses Modell wird dann zur Laufzeit der Anwendung aus einer XML-Datei geladen und durch Interpretation ausgeführt. Diese Vorgehensweise unterstützt einen sauberen Entwurf der Anwendung und die bessere Wartbarkeit. So kann die Anwendung beispielsweise leichter an verschiedene Wahlprotokolle (z. B. offene Wahl, geheime Wahl, geheime Wahl mit mehreren Stimmen, ...) angepasst werden.

Anschließend kann ein Aufbewahrer das Ergebnis zur Realisierung der Verbindlichkeit speichern. Das ganze Vorhaben wurde als prototypische Implementierung durch vier Spark-Plug-ins realisiert (SecureMUC, SecureVC, Voting und die eigentliche Anwendung IMVote). Wo sinnvoll, werden standardisierte Verfahren verwendet (z. B. Übertragung via XMPP, Kodierung von anwendungsspezifischen Daten als XML-Nachrichten).

Ist das entworfene Konzept in Nutzung, gibt es ein paar **Einschränkungen** und Grenzen, die man beachten sollte:

Die Integration von Sicherheitsmechanismen und der externen Anwendung ist alles andere als trivial. Dies zeigt sich beispielsweise bei der Integration der verschiedenen Plug-ins, aus denen IMVote besteht. In den frühen Phasen der Entwicklung wurden zur ersten Evaluation des Zugangsberechtigungsmechanismus die Anwendungen SecureMUC und SecureVC als eigenständige Prototypen, d. h. ohne weitere sicherheitskritische Anwendung (Wahl), realisiert. Parallel wurde ein davon unabhängiges Voting-Plug-in für die Durchführung von elektronischen Wahlen implementiert. Im Zuge der weiteren Entwicklung hat sich herausgestellt, dass zum Zusammenfügen dieser Plug-ins einige Anpassungen notwendig waren. Hierzu gehören beispielsweise Mechanismen zur Übergabe der Schlüssel (die von SecureMUC und SecureVC erzeugt, ausgetauscht und über Video verifiziert werden) an die Anwendung für elektronische Wahlen. Auch wenn diese Integration auf konzeptioneller Ebene einfach scheint, sind bei der Implementation doch zahlreiche Feinheiten zu beachten, beispielsweise für die korrekte Zuordnung der Sicherheitselemente zu den Teilnehmern. Ähnliche Herausforderungen ergeben sich bei der Integration der (Übertragungs-)Protokolle der verschiedenen Phasen. Die Realisierung eines Entscheidungsprozesses mit Hilfe des hier vorgeschlagenen Verfahrens kann nur unter Teilnehmern stattfinden, die sich kennen.

Bei der Wahl des Schlüsselaustauschverfahrens (DH-Key-Agreement) gibt es einen Trade-off zwischen kryptografischer Stärke und Aufwand für die Generierung der Schlüssel. So kann je nach gewählter Schlüssellänge und Leistungsfähigkeit des Rechners die Erzeugung von Schlüsseln für das DH-Verfahren schnell länger als eine Minute dauern. Dies ist für eine spontane Kollaboration nicht gerade erstrebenswert. Hier sind Verfahren im Vorteil, die Schlüssel speichern und mehrfach verwenden.

Bezüglich der Anzahl der Teilnehmer ist der hier vorgeschlagene Ansatz unter anderem durch die Bandbreite beschränkt, die für die Audio-Video-Verbindung benötigt wird. Diese funktioniert in lokalen Netzwerken mit vier Teilnehmern in akzeptabler Geschwindigkeit, wurde aber

über WAN-Verbindungen bisher noch nicht ausführlich getestet. Hier sollte noch bemerkt werden, dass nur für die eigentliche Überprüfung der Hashwerte eine Videoverbindung notwendig ist. Um Bandbreite zu sparen, könnte bei sehr großen Gruppen danach auf eine Audioverbindung oder sogar Chat zurückgeschaltet werden.

Das E-Voting-Protokoll, das dem Voting-Plug-in zugrunde liegt, erlaubt aus technischer Sicht eine Nutzung bis zu 18 Teilnehmern. Eine grundsätzlichere Beschränkung ergibt sich durch die Zeit, die notwendig ist, um den Zugangsberechtigungsmechanismus für eine große Teilnehmerzahl durchzuführen, und die Praktikabilität einer Gruppensitzung (insbesondere bei virtueller Durchführung über Audio/Video) mit so vielen Teilnehmern.

15.3 Mögliche Adaptionen

Für die Anwendung der hier vorgestellten Applikation sind verschiedene Möglichkeiten zu betrachten.

- Der einfachste Fall ist die **Anwendung** in der vorliegenden Form **in einem geschäftlichen Kontext**. Hierzu müssen die Teilnehmer die notwendige Software installieren und sich auf einem bereitgestellten IM-Server registrieren. Dank der genutzten standardisierten XMPP-Infrastruktur können dabei auch Netzwerke von mehreren Servern aufgebaut und zusammengeschaltet werden. Je nach Anwendungskontext wäre zu überprüfen, welche Implikationen die lokale Speicherung der verbindlichen Daten auf den jeweiligen Client-Rechnern verursacht. Beispielsweise sind bei der Verwendung von Laptops entsprechende Vorsorgemaßnahmen für den Verlust eines Rechners (z. B. durch Diebstahl) zu treffen.
- Ein Fall, der wesentlich mehr Adaptionsaufwand verursacht, ist die **Anwendung mit einem anderen IM-Client** als Spark. Die prototypische Implementierung verwendet zahlreiche Mechanismen von Spark, z. B. für die Realisierung von anwendungsspezifischen Nachrichten auf Basis von XMPP. Trotzdem sind die entwickelten Konzepte und zugehörigen algorithmischen Lösungen unabhängig vom gewählten IM-Client allgemein anwendbar.
- Kommt ein anderer Typ von Anwendung zum Tragen sind zwei unterschiedliche Stufen zu unterscheiden. Im einfacheren Fall werden **andere Anwendungsprotokolle** verwendet (beispielsweise eine offene Wahl anstatt einer geheimen). Dieser Fall ist aufgrund des verwendeten Softwareentwurfs (Zustandsmaschine, modularer Aufbau) komfortabel durchführbar.
- Ein anspruchsvollerer Fall, der aber auch noch gut realisiert werden kann, ist der Einsatz einer **ganz anderen Anwendung** (anstatt elektronischer Wahlen). In anderen Worten, geht es hier um die Verwendung der vorgestellten generischen Plattform (Kapitel 12) für einen anderen Anwendungstyp. Damit sich eine Anwendung prinzipiell für den Einsatz mit der Plattform eignet, muss (1) es sich grundsätzlich um eine kollaborative Anwendung handeln, die (2) über asynchrone Nachrichten implementiert werden kann, und (3) Schlüssel und Zertifikate zur Absicherung verwendet, die den Teilnehmern der Gruppe zugeordnet werden können. Auch wenn es sich dabei um eine vollständig andere Anwendung handelt, so bieten doch die hier entwickelten Konzepte (Kapitel 11) und Technologien (Kapitel 14)

eine solide Grundlage. Beispielsweise können mit Hilfe der Modellierung und Implementierung als Zustandsmaschine viele reaktive Systeme abgedeckt werden.

16 Anschließende Forschungsthemen

Während der Erarbeitung des Themas sind einige Fragen neu aufgeworfen worden, die sich für **weitere Forschungsarbeiten** anbieten:

- Wie bewährt sich das vorgestellte Konzept bei einer langzeitigen Evaluierung? Die grundsätzliche *technische* Umsetzbarkeit der Konzepte ist durch die prototypische Realisierung demonstriert worden. Darüber hinaus sollte in einer praktischen Anwendung überprüft werden, welche Vorteile und Herausforderungen in einer Gruppenkommunikation bei der Nutzung eines erweiterten IM-Systems entstehen und wie die Akzeptanz seitens der Nutzer in einem realistischen Anwendungskontext ist. Man denke hier beispielsweise an die praktische Durchführung von Zustandsberechtigungsverfahren mittels Video über Verbindungen mit beschränkter Bandbreite.
- Wie kann man die Verbindlichkeit eines kontinuierlichen Audio-Video-Streams gewährleisten? Hier besteht eine Problematik bei der Signierung solcher Daten und dem potenziellen späteren Vergleich mit Referenzdaten. Bei solchen Multimediadaten verändert die Veränderung eines einzelnen Bits unter Umständen nicht die inhaltliche Bedeutung, stellt aber eine Veränderung der digitalen Daten dar – und würde je nach Hashverfahren als solche angezeigt. Hier ist zu untersuchen, welche Methoden zum Schutz der Integrität multimedialer Daten verwendet werden können. Eine wesentliche Fragestellung ist dabei die Unterscheidung einer signifikanten, d. h. sinnverändernden, Änderung von einer nichtsignifikanten. Diese Thematik erhält durch die Verwendung von verlustbehafteter Kompression (lossy-compression) zusätzliche praktische Bedeutung.
- Welche der Kommunikationskanäle eines IM-Systems sind für die jeweilige Anwendung geeignet und mit welchem Aufwand können sie adaptiert werden? Um diese Frage zu beantworten, sollten konkrete Überlegungen zu anderen kollaborativen Anwendungen angestellt werden sowie weitere Szenarien für die Realisierung der Sicherheitsmechanismen betrachtet werden. Als Beispiel sei hier der Fall genannt, dass sich die Teilnehmer einer Gruppe untereinander nur teilweise oder gar nicht kennen.
- Wie beeinflussen die Eigenschaften der unterschiedlichen Kommunikationskanäle die Sicherheit des Gesamtsystems? Ähnlich wie für die Überlegungen zur Anwendbarkeit von Kommunikationskanälen (voriger Punkt) müssten hier zur Erlangung weiterer Erkenntnisse weitere Anwendungen unterschiedlicher Art betrachtet werden. In der vorliegenden Anwendung wurde die Sicherheit des IM-Systems und der übertragenen Nachrichten durch die Kombination der verschiedenen Kommunikationskanäle (Verifikation im Videobild, Ansatz zur verbindlichen Speicherung) erhöht.
- Wie kann formal nachgewiesen werden, dass die eingesetzten Mechanismen das angestrebte Sicherheitsziel auch gewährleisten (Konsistenz und Vollständigkeit)? Hier sind formale Methoden anzuwenden (ähnlich wie (Grimm 2009)), um entsprechende Aussagen fundiert treffen zu können.

Literaturverzeichnis

Adams D.J. 2002. *Programming Jabber - Extending XML Messaging*: O' Reilly Media Verlag

Alkassar Ammar, Krimmer Robert und Volkamer Melanie. 2005. Online-Wahlen für Gremien. *Datenschutz und Datensicherheit (DuD)* 8 (29)

Andreasen Flemming, Baugher Mark und Wing Dan. 2006. RFC 4568: Session Description Protocol (SDP), Security Descriptions for Media Streams. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc4568.txt [Zugriff am 2009-06-06]

AOL. 1998. *Talk to Oscar (TOC 1.0)* http://terraim.cvs.sourceforge.net/*checkout*/terraim/terraim/src/toc/TOC1.txt [Zugriff am 2009-04-05]

———. 2009. *AOL Instant Messenger* http://www.aol.de/AIM [Zugriff am 2009-05-05]

AOL Developer Network. 2009. *OSCAR Protocol* http://dev.aol.com/aim/oscar/ [Zugriff am 2009-04-05]

Arkko Jari, Carrara Elisabetta, Lindholm Fredrik, Naslund Mats und Norrman Karl. 2004. RFC 3838: Multimedia Internet KEYing (MIKEY). In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc3830.txt [Zugriff am 2008-06-22]

Arkko Jari, Lindholm Fredrik, Naslund Mats, Norrman Karl und Carrara Elisabetta. 2006. RFC 4567: Key Management Extensions for Session Description Protocol (SDP) and Real Time Streaming Protocol (RTSP). In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc4567.txt [Zugriff am 2009-06-06]

Avaya. 2009. *Avaya Meeting Exchange® Enterprise* http://www.avaya.de/gcm/emea/de/products/offers/one-x_desktop_edition.htm [Zugriff am 2009-01-01]

Avizienis Algirdas, Laprie Jean-Claude, Randell Brian und Landwehr Carl. 2007. Basic Concepts and Taxonomy of Dependable and Secure Computing. *IEEE Transactions on dependable and secure computing* 1 (1):11-33

Avrahami Daniel, Fussell Susan R. und Hudson Scott E. 2008. IM Waiting: Timing and Responsiveness in Semi-Synchronous Communication. Computer Supported Cooperative Work CSCW'08, San Diego, USA, 285-294

Badach Anatol. 2007. *Voice over IP - Die Technik*. München, Wien: Carl Hanser Verlag

Baron N. 2005. Instant messaging and the future of language. *Communications of the ACM* 48 (7):29-31

Baset Salman A. und Schulzrinne Henning 2006. An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol. 25th Conference on Computer Communications INFOCOM 2006, Barcelona, Spanien, IEEE Communications Society http://www1.cs.columbia.edu/~salman/skype/

Baugher Mark, McGrew David, Naslund Mats, Carrara Elisabetta und Norrman Karl. 2004. RFC 3711: The Secure Real-time Transport Protocol (SRTP). In *Request for Comments, Network Working Group* http://www.faqs.org/rfcs/rfc3711.html [Zugriff am 2008-02-02]

Benaloh Josh Cohen und Yung Moti 1986. Distributing the power of a government to enhance the privacy of voters. 5th Annual ACM symposium on principles of distributed computing, Calgary, Alberta, Canada, 52 - 62 ACM New York, NY, USA http://portal.acm.org [Zugriff am 2008-04-28]

Berson Tom. 2005. Skype security evaluation ALR-2005-031 Anagram Laboratories http://www.anagram.com/index.html [Zugriff am 2008-05-05]

BGB. 2009. *Bürgerliches Gesetzbuch*: DTV-Beck Verlag

Bilien Johan , Eliasson Erik, Orrblad Joakim und Vatn Jon-Olov. 2005. Secure VoIP: call establishment and media protection. 2nd Workshop on Securing Voice over IP, Washington DC http://www.tslab.ssvl.kth.se/publications [Zugriff am 2006-05-05]

Biondi Philippe und Desclaux Fabrice. 2006. Silver Needle in the Skype. BlackHat Europe, Suresnes, France http://www.secdev.org/conf/skype_BHEU06.handout.pdf [Zugriff am 2006-04-18]

Bischofs Ludger, Hasselbring Wilchelm und Warns Timo. 2006. Peer-to-Peer-Architekturen. In *Handbuch der Software-Architektur*, R. Reussner und W. Hasselbring (Hrsg.) Heidelberg: dpunkt.verlag GmbH

Bodine Kerry und Pignol Mathilde. 2003. Kinetic typography-based instant messaging. Conference on Human Factors in Computing Systems CHI 2003 Ft. Lauderdale, Florida, USA, 914-915

Boll Matthias. 2009. Konzeption und Realisierung einer sicheren Audio-/Videogruppenkommunikation. Bachelorarbeit, Institut für Wirtschafts- und Verwaltungsinformatik, Universität Koblenz-Landau

Borisov Nikita, Goldberg Ian und Brewer Erik A. 2004. Off-the-record communication, or, why not to use PGP. 2004 ACM Workshop on Privacy in the Electronic Society, 77-84 http://www.cypherpunks.ca/otr/otr-wpes.pdf [Zugriff am 2008-06-15]

bouncycastle.org. 2009. *The Legion of the Bouncy Castle* http://www.bouncycastle.org/ [Zugriff am 2009-06-07]

Bundesamt für Sicherheit in der Informationstechnik. 1999. Gemeinsame Kriterien für die Prüfung und Bewertung der Sicherheit von Informationstechnik - Common Criteria 2.1 Bundesamt für Sicherheit in der Informationstechnik

———. 2005. *VoIPSEC Studie zur Sicherheit von Voice over Internet Protocol* www.bsi.de/literat/studien/VoIP/index.htm

———. 2007. BSI-Standard 100-2: IT-Grundschutz-Vorgehensweise. In *BSI-Standards* Bundesamt für Sicherheit in der Informationstechnik http://www.bsi.de/literat/bsi_standard/index.htm

Bundesministerium der Justiz. 1949. Grundgesetz für die Bundesrepublik Deutschland Bundesministerium der Justiz www.gesetze-im-internet.de/bundesrecht/gg/gesamt.pdf

———. 1999. Verordnung über den Einsatz von Wahlgeräten bei Wahlen zum Deutschen Bundestag und der Abgeordneten des Europäischen Parlaments aus der Bundesrepublik Deutschland (Bundeswahlgeräteverordnung - BWahlGV) Bundesministerium der Justiz http://bundesrecht.juris.de/bwahlgv/BJNR024590975.html [Zugriff am 2008-08-05]

———. 2009a. Gesetz über Rahmenbedingungen für elektronische Signaturen (Signaturgesetz - SigG) http://www.gesetze-im-internet.de/sigg_2001/index.html [Zugriff am 2009-08-08]

———. 2009b. Gesetz zur Vereinheitlichung von Vorschriften über bestimmte elektronische Informations- und Kommunikationsdienste (Elektronischer-Geschäftsverkehr-Vereinheitlichungsgesetz - ElGVG) http://www.bgblportal.de/BGBL/bgbl1f/bgbl107s0179.pdf [Zugriff am 2009-09-23]

———. 2009c. Strafprozeßordnung http://www.gesetze-im-internet.de/stpo/ [Zugriff am 2009-09-08]

Burton Jim, Parker Marty, Blair Pleasant. und Van Doren Don. 2007. Will 2007 Be the Year of Unified Communications? *Business Communications Review* März:20-26

Cameron Ann Frances und Webster Jane. 2005. Unintended Consequences of Emerging Communication Technologies: Instant Messaging in the Workplace. *Computers in Human Behavior* 21:85-103

Campbell Ben, Rosenberg Jonathan David, Schulzrinne Henning, Huitema Christian und David Gurle. 2002. RFC 3428: Session Initiation Protocol (SIP) extension for instant messaging. In *Request for Comments, Network Working Group* http://www.faqs.org/rfcs/rfc3428.html

Cerulean Studios. 2009. *Trillian Astra* http://www.trillian.im [Zugriff am 2009-09-09]

Chaum David. 1981. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Communication of the ACM* 24 (2)

Clark David und Wilson David. 1987. A comparison of comercial ans military computer security policies. IEEE Symposium on Security and Privacy, 184-194

Common Criteria Recognition Agreement. 2009. Common Criteria for Information Technology Security Evaluation - Common Criteria 3.1 Bundesamt für Sicherheit in der Informationstechnik http://www.commoncriteriaportal.org/thecc.html [Zugriff am 2009-08-08]

Contreras-Castillo Juan, Perez-Fragoso Carmen und Favela Jesus. 2006. Assessing the Use of Instant Messaging in Online Learning Environments. *Interactive Learning Environments* 14 (3):205 - 218

Czarnecki Krzysztof und Eisenecker Ulrich. 2000. *Generative Programming - Methods, Tools and Applications*: Addison-Wesley Verlag

Czarnecki Krzysztof, Helsen Simon und Eisenecker Ulrich. 2005. Staged configuration through specialization and multilevel configuration of feature models. *Software Process: Improvement and Practice, Special Issue: Special Issue on Software Product Lines* 10 (2):143-169

Day Marc, Rosenberg Jonathan David und Sugano Hiroyasu. 2000. RFC 2778: A Model for Presence and Instant Messaging. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc2778.txt [Zugriff am 2008-02-20]

DeMillo Richard A. , Lynch Nancy A. und Merritt Michael I. . 1982. Cryptographic Protocols. 14. ACM Symposium on the Theory of Computing, 383-400 http://theory.lcs.mit.edu/tds/papers/Lynch/stoc82.pdf

Dierks Tim und Rescorla Eric. 2006. RFC 4346: The Transport Layer Security (TLS) Protocol. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc4346.txt [Zugriff am 2009-03-04]

———. 2008. RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2. In *Request for Comments, Network Working Group* http://tools.ietf.org/html/rfc5246 [Zugriff am 2008-05-06]

Dierstein Rüdiger. 2004a. Duale Sicherheit – IT-Sicherheit und ihre Besonderheiten. In *Mehrseitige Sicherheit in der Kommunikationstechnik*, A. Pfitzmann und G. Müller (Hrsg.) Bonn: Addison Wesley Longman Verlag

———. 2004b. Sicherheit in der Informationstechnik – der Begriff IT-Sicherheit *Informatik-Spektrum* 27 (4):343-353

Dietz Florian. 2009. Absicherung und Verifikation einer Instant Messaging Kommunikation über DH-Key-Agreement. Bachelorarbeit, Institut für Wirtschafts- und Verwaltungsinformatik, FB4, Universität Koblenz-Landau

Diffie Whitfield, van Oorschot Paul C. und Wiener Michael J. 1992. Authentication and Authenticated Key Exchanges. *Designs, Codes and Cryptography* 2 (2):107-125

Dittmann Jana. 2000. *Digitale Wasserzeichen. Grundlagen, Verfahren, Anwendungsgebiete*: Springer Verlag

Dittmann Jana, Steinmetz Arnd und Steinmetz Ralf. 1999. Content-based Digital Signature for Motion Pictures Authentication and Content-Fragile Watermarking. International Conference on Multimedia Computing and Systems, Vol. 1, 574-579

Dittmann Jana und Wohlmacher Petra. 2000. Aspekte der Sicherheit multimedialer Daten und Anwendungen mittels Kryptographie und digitaler Wasserzeichentechniken. Sicherheit in Netzen und Medienströmen, Tagungsband des GI-Workshops "Sicherheit in Mediendaten", 107-123, Springer Verlag

Dittmann Miguel. 2002. *Sprachverwendung im Internet: Untersuchungen zu Sprache und Nutzung des Internet Relay Chat (IRC) in Deutschland und Frankreich*: Books on Demand GmbH

Dix Alan, Finley Janet, Abowd Gregory und Beale Russell. 1993. *Human-Computer Interaction*. New York: Prentice Hall Verlag

Dunte Markus und Ruland Christoph. 2007. Secure Voice-over-IP. *IJCSNS International Journal of Computer Science and Network Security* 7 (6):63-68

Dworkin Morris. 2001. Recommendation for Block Cipher Modes of Operation Methods and Techniques, NIST Special Publication 800-38A 2001 Edition NIST National Institute of Standards ans Technology http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf [Zugriff am 2009-09-06]

Eberspächer Jörg und Schollmeier Rüdiger. 2005. First and Second Generation of Peer-to-Peer Systems. In *Peer-to-Peer Systems and Applications* Berlin, Heidelberg: Springer Verlag

Eckert Claudia. 2009. *IT-Sicherheit, Konzepte - Verfahren - Protokolle*. München, Wien: Oldenbourg Verlag

Ehlert Sven, Petgang Sandrine, Magedanz Thomas und Sisalem Dorgham. 2006. Analysis and Signature of Skype VoIP Session Traffic. Fourth International Conference on Communications, Internet, and Information Technology (CIIT 2006), St. Thomas, US Virgin Islands http://user.cs.tu-berlin.de/~dukat/ehlert.name/publications/Ehlert_SkypeSignature_2006.pdf [Zugriff am 2008-07-07]

ElGamal Taher. 1985. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory* 31 (4):469-472

Elliot Bern. 2008. Magic Quadrant for Unified Communications, Gartner RAS Core Research Note G00160407 Gartner Inc. http://mediaproducts.gartner.com/gc/reprints/ibm/external/2008/volume3/article24/pdf/article24.pdf

Ellis Clarence A., Gibbs Simon J. und Rein Gail 1991. Groupware: some issues and experiences. *Communications of the ACM* 34 (1)

Eren Evren und Detken Kai-Oliver. 2007. *VoIP Security- Konzepte und Lösungen für sichere VoIP-Kommunikation*. München: Carl Hanser Verlag

Ericsson, Motorola und Nokia. 2002. Wireless Village: System Architecture Model, Version 1.1, WV Tracking Number: WV-020 Wireless Village Initiative http://www.openmobilealliance.org/tech/affiliates/wv/wv_architecture_v1.1.pdf [Zugriff am 2008-09-08]

Ethereal. 2006. *Ethereal - Network Protocol Analyzer* http://www.ethereal.com/ [Zugriff am 2009-04-05]

eVelopers Corporation. 2009. *Unimod* http://unimod.sourceforge.net/ [Zugriff am 2009-06-06]

Ezzat Tony, Geiger Gadi und Poggio Tomaso. 2002. Trainable videorealistic speech animation. 29th International Conference on Computer Graphics and Interactive Techniques SIGGRAPH 2002, San Antonio, Texas, 388 - 398

Frank Ulrich. 2006. Towards a Pluralistic Conception of Research Methods in Information Systems Research. In *ICB-Research Reports*, ICB-Research Report No. 7 Institut für Informatik und Wirtschafstinformatik Duisburg Essen http://www.icb.uni-due.de/fileadmin/ICB/research/research_reports/ICBReport07.pdf

Franks John, Hallam-Baker Phillip, Hostetler Jeffery, Lawrence Scott D., Leach Paul J., Luotonen Ari und Stewart Lawrence C. 1999. RFC2617: HTTP Authentication: Basic and Digest Access Authentication. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc2617 [Zugriff am 2009-05-05]

Freier Alan O., Karlton Philip und Kocher Paul C. 1996. Internet Draft: The SSL Protocol Version 3.0 http://tools.ietf.org/html/draft-freier-ssl-version3-01 [Zugriff am 2008-05-06]

Frößler Frank. 2008. A Practice Theoretical Analysis of Real Time Collaboration Technology - Skype and Sametimes in Software Development Projects. PhD, University College Dublin, Dublin

Garfinkel Simson L. 2005. VoIP and Skype Security http://www1.cs.columbia.edu/~salman/skype/SkypeSecurity_1_5_garfinkel.pdf [Zugriff am 2008-08-07]

Gartner Group. 2007. *Gartner Predicts Instant Messaging Will Be De Facto Tool for Voice, Video and Text Chat by The End of 2011* http://www.gartner.com/it/page.jsp?id=507731

Gohring Nancy. 2009. *Second lawsuit threatens Skype sale* http://www.computerworld.com/s/article/9138246/Second_lawsuit_threatens_Skype_sale [Zugriff am 2009-09-20]

Google. 2009. *Google Talk* http://www.google.com/talk/intl/de/ [Zugriff am 2009-05-05]

Grimm Rüdiger. 1994. *Sicherheit für offene Kommunikation - Verbindliche Telekooperation*. Mannheim: BI Wissenschaftlicher Verlag, Spektrum

———. 2009. A Formal IT-Security Model for a Weak Fair-Exchange Cooperation with Non-Repudiation Proofs. The Third International Conference on Emerging Security Information, Systems and Technologies SECURWARE 2009, Athens, Griechenland, IEEE Computer Society Press

Grimm Rüdiger, Krimmer Robert, Meissner Nils, Reinhard Kai, Volkamer Melanie und Weinand Marcel. 2006. Security Requirements for Non-political Internet Voting. 2nd International Workshop, Electronic Voting 2006, Bregenz, 203-212, Gesellschaft für Informatik, Lecture Notes on Informatics 86

Grimm Rüdiger, Mehr Farid, Meletiadou Anastasia, Pähler Daniel und Uerz Ilka. 2007. SOA-Security. In *Arbeitsberichte aus dem Fachbereich Informatik* Institut für Wirtschafts- und Verwaltungsinformatik Koblenz http://www.uni-koblenz.de/FB4/Publications/Reports/index_html [Zugriff am 2008-03-15]

Grimm Rüdiger, Meletiadou Anastasia und Hundacker Helge. 2008. Anwendungsbeispiele für Kryptographie (erweiterte Version). In *Arbeitsberichte aus dem Fachbereich Informatik*, 2/2008 Universität Koblenz-Landau U. Koblenz-Landau (Hrsg.) Koblenz http://www.uni-koblenz-landau.de/koblenz/fb4/publications/Reports/arbeitsberichte

Grimm Rüdiger, Reinhard Kai, Winter Cornelia und Witte Jule. 2009. Erfahrungen mit Online-Wahlen für Vereinsgremien. *Datenschutz und Datensicherheit.* 22 (3):97-101

Gross Tom und Koch. 2007. *Computer-Supported Cooperative Work*. M. Herczeg (Hrsg.), *Interaktive Medien*. München, Wien Oldenbourg Verlag

Handel Mark und Herbsleb James D. 2002. What is chat doing in the workplace? ACM Conference on Computer Supported Cooperative Work, 1-10, ACM Press http://conway.isri.cmu.edu/~jdh/collaboratory/research_papers/cscw-2002.pdf [Zugriff am 2008-06-03]

Handley M. und Jacobson V. 1998. RFC 2327: SDP: Session Description Protocol. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc2327.txt [Zugriff am 2009-05-06]

Handley Mark, Schulzrinne Henning, Schooler Eve und Rosenberg Jonathan David. 1999. RFC 2543: SIP: Session Initiation Protocol. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc2543.txt [Zugriff am 2009-05-06]

Harris Shon. 2007. *CISSP Certification All-in-One Exam Guide*: McGraw-Hill Osborne Media

Helbach Jörg, Krimmer Robert, Meletiadou Anastasia, Meissner Nils und Volkamer Melanie. 2007. Zukunft von Online-Wahlen. Aktuelle rechtliche, politische soziale und technisch-organisatorische Fragen *Datenschutz und Datensicherheit (DuD)* 6 (2007)

Herbsleb James D. , Atkins David L., Boyer David G , Handel Mark und Finholt Thomas A. 2002. Introducing instant messaging and chat in the workplace. Conference on Human Factors in Computing Systems, SIGCHI 2002, 171-178, ACM Press http://www.crew.umich.edu/Technical%20reports/Herbsleb_Atkins_Boyer_Handel_Finholt_Introducing_instant_messaging_12_10_01.pdf [Zugriff am 2008-06-03]

Hess Thomas, Anding Markus und Schreiber Matthias. 2002. Napster in der Videobranche? Erste Überlegungen zu Peer-to-Peer-Anwendungen für Videoinhalte. In *Peer-to-Peer*, D. Schoder, K. Fischbach und R. Teichmann (Hrsg.) Berlin, Heidelberg, et.al.: Springer Verlag

Hevner Alan R., March Salvatore T., Park Jinsoo und Ram Sudha. 2004. Design Science in IS Research. *MIS Quarterly* 28 (1):75-105

Hollis Emily. 2004. Get a handle on instant messaging. *Certification Magazine* Oktober

Housley Russell. 2004. RFC 3686: Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP). In *Request for Comments, Network Working Group* http://www.networksorcery.com/enp/rfc/rfc3686.txt [Zugriff am 2008-02-02]

Hrastinski Stefan. 2006. The relationship between adopting a synchronous medium and participation in online group work: An explorative study. *Interactive Learning Environments* 14 (2):137-152

Huang Albert H., Hung Shin-Yuan und Yen David C. 2007. An exploratory investigation of two internet-based communication modes. *Computer Standards & Interfaces archive* 29 (2):238-243

Hupf Katharina und Meletiadou Anastasia. 2009. End-to-End verifizierbare Wahlverfahren in Hinblick auf den Grundsatz der Öffentlichkeit der Wahl. Workshop Elektronische Wahlen, elektronische Teilhabe, Societyware, Informatik 2009, Lübeck, Vol. 154, 1847-1855, Lecture Notes in Informatics (LNI)

ICQ. 2008. *ICQ everybody everywhere*. ICQ LLC http://www.icq.com/ [Zugriff am 2008-08-06]

Ignite Realtime Community. 2008a. *Openfire - a jive software community*. Ignite Realtime Community http://www.igniterealtime.org/projects/openfire/ [Zugriff am 2008-03-28]

———. 2008b. *Spark - a jive software community*. Ignite Realtime Community http://www.igniterealtime.org/projects/spark [Zugriff am 2008-03-28]

———. 2009a. *Smack API - a jive software community*. Ignite Realtime Community http://www.igniterealtime.org/projects/smack/ [Zugriff am 2008-03-28]

———. 2009b. *SparkWeb - a jive software community*. Ignite Realtime Community [Zugriff am 2009-09-09]

Isaacs Ellen, Walendowski Alan und Ranganathan Dipti. 2002. Mobile instant messaging through Hubbub. *Communications of the ACM* 45 (9):68-72

Isaacs Ellen, Walendowski Alan, Whittaker Steve, Schiano Diane J. und Kamm Candace. 2002. The character, functions, and styles of instant messaging in the workplace. Conference on Computer Supported Cooperative Work 2002, CSCW 2002, 11–20, ACM Press http://dis.shef.ac.uk/stevewhittaker/IM-CSCW02-final.pdf

Jabber.org. 2009. *Jabber* http://www.jabber.org/ [Zugriff am 2009-05-05]

Jacobson Ivar, Booch Grady und Rumbaugh James. 1999. *The Unified Software Development Process*: Addison-Wesley Longman

Johansen Robert. 1991. Teams for tomorrow. 24. IEEE Hawaii International Conference On System Science, Kauai, HI, USA, S. 520-534, IEEE Computer Society Press

Joisten Martina. 2007. Renegotiating Interaction Routines: Adoption of Skype in the Workplace. Mensch & Computer 2007 - Konferenz für interaktive und kooperative Medien, München, 303-307, Oldenbourg Verlag

Junit.org. 2009. *JUnit.org Resources for Test Driven Development* http://www.junit.org/ [Zugriff am 2009-06-08]

Kanbach Andreas. 2005. *SIP - Die Technik: Grundlagen und Realisierung der Internet-Technik - Für VoIP, Videotelefonie, Instant Messaging und Presence Service*. Wiesbaden: Vieweg Verlag

Kanellos Michael 2006. *Are fake videos next?* http://news.cnet.com/Are-fake-videos-next/2100-1008_3-6113449.html [Zugriff am 2009-04-05]

Kang Kyo Chul, Cohen Sholom G., Hess James A., Novak William E. und Peterson Spencer A. 1990. Feature-Oriented Design Analysis (FODA) Feasibility Study, CMU/SEI-90-TR-21 Software Engineering Institute, Carnegie Mellon University Pittsburgh, PA

Kang Kyung-Pyo , Choi Yoon-Hee und Choi Tae-Sun. 2004. Real-Time Video Watermarking for MPEG Streams. Computational Science and Its Applications ICCSA 2004, Vol. 3046/2004, 348-358, Lecture Notes in Computer Science

Kent Stephen. 1998. RFC 2401:Security Architecture for the Internet Protocol. In *Network Working Group, Request for Comments,* http://www.ietf.org/rfc/rfc2401.txt [Zugriff am 2008-05-07]

Kikuchi Hiroaki, Tada Minako und Nakanishi Shohachiro. 2004. Secure instant messaging protocol preserving confidentiality against administrator. 8. International Conference on Advanced Information Networking and Applications, AINA 2004, Fukuoka, Japan, Vol. 2, 27-30, IEEE Computer Society

Koch Michael. 2009. Groupware In *Enzyklopädie der Wirtschaftsinformatik – Online-Lexikon*, K. Kurbel, J. Becker, N. Gronau, E. Sinz und L. Suhl (Hrsg.) http://www.enzyklopaedie-der-wirtschaftsinformatik.de/

Kovačević J., Aférez M., Kulesza U., Moreira A., Araújo J., Amaral V., Alves V.R., Rashid A. und Chitchyan R. 2007. AMPLE - Aspect-Oriented, Model-Driven, Product Line Engineering, Specific Targeted Research Project: IST- 33710, Survey of the state-of-the-art in Requirements Engineering for Software Product Line and Model-Driven Requirements Engineering, AMPLE: D1.1 http://ample.holos.pt/gest_cnt_upload/editor/File/public/Deliverable%20D1.1.pdf

Krempl Stefan und Kuri Jürgen. 2008. *Ein "Bayerntrojaner" zum Abhören von Internet-Telefonie?* Heise Security News http://www.heise.de/newsticker/Ein-Bayerntrojaner-zum-Abhoeren-von-Internet-Telefonie--/meldung/102375 [Zugriff am 2009-02-02]

Kruchten Philippe. 1995. Architectural Blueprints - The “4+1” View Model of Software Architecture. *IEEE Software* 12 (6):42-50

Land Rheinland-Pfalz. 2003. 223-41 Hochschulgesetz (HochSchG) Rheinland-Pfalz. In *Gesetz- und Verordnungsblatt für das Land Rheinland-Pfalz* http://www.uni-koblenz-landau.de/verwaltung [Zugriff am 2008-04-22]

Langer Lucie und Opitz-Talidou Zoi. 2009. Elektronische Aufbewahrung von Wahldokumenten bei Onlinewahlen *Datenschutz und Datensicherheit* 33 (7):418-422

Lazar Irwin. 2006. Integrating Telephony, IM, Video and Mobility with Presence. *Business Communications Review* (June 2006):28-31

Leach Paul J. und Newman Chris. 2000. RFC 2831: Using Digest Authentication as a SASL Mechanism. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc2831.txt [Zugriff am 2008-08-08]

Leavitt Neal. 2005. Instant Messaging: A new target for hackers. *Computer* 38 (7):20-23

Lehner Franz und Zelewski Stephan. 2007. Kann Wissenschaftstheorie behilflich für die Publikationspraxis sein? Eine kritische Auseinandersetzung mit den "Guidelines" von Hevner et al . In *Wissenschaftstheoretische Fundierung und wissenschaftliche Orientierung der Wirtschaftsinformatik*, F. Lehner und S. Zelewski (Hrsg.) Berlin: GITO

Löber Andreas, Grimm Sibylle und Schwabe Gerhard. 2005. Audio vs. Chat: The Effects of Group Size on Media Choice. 38. Hawaii International Conference on System Sciences, HICSS'05, Hawai, 40-41

Ludwig Scott, Beda Joe, Saint-Andre Peter, McQueen Robert, Egan Sean und Hildebrand Joe. 2009a. XEP-0166: Jingle XMPP Standards Foundation http://xmpp.org/extensions/xep-0166.html [Zugriff am 2009-07-07]

Ludwig Scott, Saint-Andre Peter, Egan Sean, McQueen Robert und Cionoiu Diana 2009b. XEP-0167: Jingle RTP Sessions XMPP Standards Foundation http://xmpp.org/extensions/xep-0167.html [Zugriff am 2009-07-07]

Mannan Mohammad und Van Oorschot P.C. 2006a. A Protocol for Secure Public Instant Messaging (Extended Version), TR-06-01 Carleton University http://www.scs.carleton.ca/research/tech_reports/2006/download/TR-06-01.pdf [Zugriff am 2007-04-04]

Mannan Mohammad und van Oorschot Paul C. 2005. On Instant Messaging Worms, Analysis and Countermeasures. 2005 ACM workshop on Rapid malcode, WORM '05, Fairfax, VA, USA, ACM Press [Zugriff am 2007-06-25]

———. 2006b. A Protocol for Secure Public Instant Messaging. In *Financial Cryptography and Data Security* Berlin / Heidelberg: Springer Verlag

McCullagh Declan 2008. *How safe is instant messaging? A security and privacy survey*. Politics and Law, CNET News http://news.cnet.com/8301-13578_3-9962106-38.html [Zugriff am 2009-07-08]

Meletiadou Anastasia. 2008. E-Voting auf Grundlage eines Instant Messaging Systems. Informatik 2008 - Beherrschbare Systeme - dank Informatik, München, Vol. Band 1, 405-410, Lecture Notes in Informatics 133

———. 2009. Instant Messaging Systeme als Plattform für elektronisches Wählen. D.A.CH Security 2009 - Bestandsaufnahme, Konzepte, Anwendungen, Perspektiven, Bochum, 77-84, syssec Verlag

Meletiadou Anastasia und Grimm Rüdiger. 2009. Using Instant Messaging Systems as a Platform for Electronic Voting. E-Technologies: Innovation in an Open World: 4. International MCTECH Conference on eTechnologies, Ottawa, Kanada, Vol. LNBIP 26, 13-24, Springer Verlag

Melnikov Alexey und Zeilenga Kurt. 2006. RFC 4422: Simple Authentication and Security Layer (SASL). In *Request for Comments, Network Working Group* http://tools.ietf.org/html/rfc4422 [Zugriff am 2008-08-08]

Menezes Alfred J., van Oorschot Paul C. und Vanstone Scott A. 2001. *Handbook of Applied Cryptography*: CRC Press

Merritt Michael I. 1983. Cryptographic protocols. Phd disseration GIT-ICS 83/6, Georgia Institut of Technology, Georgia Tech, Georgia

Meyer Dirk. 2009. Internet-Draft: XTLS - End-to-End Encryption for the Extensible Messaging and Presence Protocol (XMPP) Using Transport Layer Security (TLS). In *Network Working Group* http://tools.ietf.org/html/draft-meyer-xmpp-e2e-encryption-02 [Zugriff am 2009-09-09]

Microsoft. 2009a. *Windows Live Messenger* http://download.live.com/messenger [Zugriff am 2009-07-26]

———. 2009b. *Windows Live Messenger* http://messenger.live.de/ [Zugriff am 2009-05-05]

Microsoft TechNet. 2005. *.NET Passport Authentication* http://technet.microsoft.com/en-us/library/cc778966(WS.10).aspx [Zugriff am 2009-07-20]

Miller Jeremy, Saint-Andre Peter und Hancke Phillip. 2009. XEP-0220: Server Dialback. In *XMPP Extensions* XMPP Standards Foundation http://xmpp.org/extensions/xep-0220.html [Zugriff am 2009-09-09]

Mintz Mike und Sayers Andrew. 2003. *MSN Messenger Protocol* http://www.hypothetic.org/docs/msn/index.php [Zugriff am 2009-05-06]

Miranda. 2009. *Miranda* http://miranda-im.de/ [Zugriff am 2009-05-05]

Movva Ramu und Lai William. 1999. *Instant Messaging and Presence Protocol - MSN Messenger Service 1.0 Protocol* http://www.hypothetic.org/docs/msn/ietf_draft.txt [Zugriff am 2009-05-06]

MSNPiki. 2009. *MSNPiki - Unofficial MSN Protocol Documentation* http://msnpiki.msnfanatic.com/index.php/Main_Page [Zugriff am 2009-05-06]

Muldowney Thomas. 2006. XEP-0027: Current Jabber OpenPGP Usage. In *XMPP Extensions* XMPP Standards Foundation http://xmpp.org/extensions/xep-0027.html [Zugriff am 2008-05-05]

Myers John G. 1997. RFC 2222: Simple Authentication and Security Layer (SASL). In *Request for Comments, Network Working Group* http://tools.ietf.org/html/rfc2222 [Zugriff am 2008-08-09]

Nardi Bonnie, Whittaker Steve und Bradner Eric. 2000. Interaction and Outeraction: Instant Messaging in Action. Conference on Computer Supported Cooperative Work, CSCW 2000, 79-88 http://dis.shef.ac.uk/stevewhittaker/outeraction_cscw2000.pdf

National Institute of Standards and Technology. 2007. Digital Signatur Standard (DSS). In *Federal Information Processing Standards Publication*, FIPS PUB 186-2 http://csrc.nist.gov/publications/fips/fips186-2/fips186-2-change1.pdf

NetPeeker. 2009. *NetPeeker* http://www.net-peeker.com/ [Zugriff am 2009-05-05]

Neustaedter C., Greenberg S. und Carpendale S. 2002. IMVis: Instant messenger visualization. Conference on Computer Supported Cooperative Work, CSCW 2002 New Orleans, Louisiana, 6-7, ACM Press

Ondrisek Barbara. 2008. Sicherheit elektronischer Wahlen - Eine Methode zur Bewertung und Optimierung der Sicherheit von E-Voting-Systemen. Dissertation, Institut für Gestaltungs- und Wirkungsforschung, Fakultät der Informatik, Technische Universität Wien, Wien

Open Mobile Alliance. 2007a. Enabler Release Definition for IMPS 1.3, OMA-ERELD-IMPS-V1_3-20070123-A Open Mobile Alliance http://www.openmobilealliance.org/Technical/release_program/docs/IMPS/V1_3-20070123-A/OMA-ERELD-IMPS-V1_3-20070123-A.pdf

———. 2007b. IMPS Architecture 1.3, OMA-AD-IMPS-V1_3-20070123-A Open Mobile Alliance http://www.openmobilealliance.org/Technical/release_program/docs/IMPS/V1_3-20070123-A/OMA-AD-IMPS-V1_3-20070123-A.pdf [Zugriff am 2008-09-05]

pagetable.com. 2006. *Skype Reads Your BIOS and Motherboard Serial Number* http://www.pagetable.com/?p=27 [Zugriff am 2008-05-05]

Perey Christine. 2004. *Top 5 IM security risks*. Network World http://www.networkworld.com/research/2004/0628imfeat5.html [Zugriff am 2008-08-08]

Petraschek Martin. 2007. Schlüsselworte - Sichere Internet-Telefonie mittels ZRTP. *c't - magazin für computertechnik* 09 (2007):164-167

Pfitzmann Birgit und Waidner Michael. 1992. Unconditionally Untraceable and Fault-tolerant Broadcast and secret ballot election. In *Hildesheimer Informatikberichte*, 3/92 Universität Hildesheim, Institut für Informatik

Piccard Paul, Baskin Brian, Craig Edwards, Spillman George und Sachs Marcus. 2005. *Securing IM and P2P Applications for the Enterprise*. K. Beaver (Hrsg.): Syngress Media Verlag

Picot Arnold, Riemer Kai und Taing Stefan. 2008. Unified Communications. In *Enzyklopädie der Wirtschaftsinformatik – Online-Lexikon*, K. Kurbel, J. Becker, N. Gronau, E. Sinz und L. Suhl (Hrsg.) http://www.enzyklopaedie-der-wirtschaftsinformatik.de/

Pidgin. 2009. *Pidgin, the universal chat client* http://www.pidgin.im/ [Zugriff am 2009-05-05]

Poletti Therese 2009. *EBay manages to get amazing price for most of Skype* http://www.marketwatch.com/story/ebay-gets-big-bucks-for-most-of-skype-2009-09-01 [Zugriff am 2009-09-09]

Quan-Haase Anabel. 2008. Instant Messaging on Campus: Use ans Integration in University Students Everyday Communication. *The information society: an international journal* 24 (2):105-115

Ramaswamy Nandakishore und Rao K. R. . 2006. Video Authentication for H.264/AVC using Digital Signature Standard and Secure Hash Algorithm. International Workshop on Network and Operating System Support for Digital Audio and Video NOSSDAV '06, Vol. 21, 1-6 http://www.nandakishore.com/aboutme/Academics/MS_thesis_presentations/video-authentication-ramaswamy-rev1.pdf [Zugriff am 2009-07-06]

Ramsdell Blake. 2004. RFC 3851: Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc3851.txt [Zugriff am 2009-06-05]

Raymond Panko. 2004. *Corporate Computer and Network Security*: Pearson Education

Riemer Kai. 2007. Präsenzbasierte Echtzeitkommunikation - Eine prototypbasierte Untersuchung der Nutzbarkeit im Unternehmensberatungskontext,. 8. Internationale Tagung Wirtschaftsinformatik, Karlsruhe

Riemer Kai, Arendt Patrik und Wulf Andreas. 2005. *Marktstudie Kooperationssysteme - Von E-Mail über Groupware zur Echtzeitkooperation*. Göttingen: Cuvillier Verlag

Riemer Kai und Frößler Frank. 2006. Presence-based, Context-sensitive Real-Time Collaboration (RTC) - research directions for a new type of eCollaboration system. 19. Bled eConference, Bled

Riemer Kai, Frößler Frank und Klein Stefan. 2007. Real Time Communication - Modes of Use in Distributed Teams. 15. European Conference on Information Systems, St.Gallen (CH)

Rittinghouse John und Ransome James F. 2005. *IM Instant Messaging Security*. H. P. Schmidt (Hrsg.): Elsevier Digital Press Verlag

Rivest Roland L., Shamir Adi und Adleman Leonard. 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM* 21 (2):120-126

Roach Adam. 2002. RFC 3265: Session Initiation Protocol (SIP)-specific event notification. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc3265.txt

Rosenberg Jonathan David. 2006. RFC 4479 : A Data Model for Presence. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc4479.txt [Zugriff am 2009-05-07]

Rosenberg Jonathan David, Schulzrinne Henning, Camarillo Gonzalo, Johnston Alan, Peterson Jon, Sparks Robert, Handley Mark und Schooler Eve. 2002. RFC 3261: SIP: Session Initiation Protocol. In *Request for Comments, Network Working Group* http://www.faqs.org/rfcs/rfc3261.html

Rupp Stephan, Siegmung Gerd und Lautenschlager Wolfgang. 2002. *SIP Multimediale Dienste im Internet - Grundlagen, Architektur, Anwendungen*: dpunkt Verlag

Sachpazidis Ilias, Rohl Roland, Kontaxakis George und Sakas Georgios. 2006. TeleHealth networks: instant messaging and point-to-point communication over the Internet. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 2:631-634

Saint-Andre Peter. 2004a. RFC 3920: Extensible Messaging and Presence Protocol (XMPP). In *Request for Comments, Network Working Group* http://tools.ietf.org/id/draft-ietf-xmpp-core-24.txt

———. 2004b. RFC 3923:End-to-End Signing and Object Encryption for the Extensible Messaging and Presence Protocol (XMPP). In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc3923.txt [Zugriff am 2009-09-09]

———. 2009. XEP-0234: Jingle File Transfer XMPP Standards Foundation http://xmpp.org/extensions/xep-0234.html [Zugriff am 2009-07-07]

Saint-Andre Peter, Karneges Justin und Meyer Dirk 2009. XEP-0247: Jingle XML Streams XMPP Standards Foundation http://xmpp.org/extensions/xep-0247.html [Zugriff am 2009-07-07]

Saint-Andre Peter, Smith Kevin und Tronçon Remko 2009. *XMPP: The Definitive Guide Building Real-Time Applications with Jabber Technologies*. Beijing, Cambridge et.al.: O'Reilly Verlag

Schlifni Manhard. 2000. Electronic Voting Systems and electronic democracy - Participatory e-politics for a new wave of democracy, Technische Universität Wien, Wien http://members.chello.at/manhard.schlifni/Webpub/Inhalt/Inhalt.htm

Schlildt Holger. 2005. Sicherheitsaspekte von Instant Messaging, Fakultät für Informatik, Professur für Betriebssysteme, Teschnische Universität Chemnitz, Chemnitz http://archiv.tu-chemnitz.de/pub/2005/0091/index.html

Schmeh Klaus. 2001. *Kryptografie und Public-Key-Infrastukturen im Internet*. Heidelberg: dpunkt.Verlag

Schmidt Jürgen. 2006. Der Lochtrick Wie Skype & Co. Firewalls umgehen. *c't - magazin für computertechnik* 17 (2006):142- 145

———. 2009. *Skype-Wanze im Quelltext veröffentlicht*. Heise Security News http://www.heise.de/security/Skype-Wanze-im-Quelltext-veroeffentlicht--/news/meldung/144348 [Zugriff am 2009-09-09]

Schneier Bruce. 1996. *Applied Cryptography - Protocols, Algorithms and source code in C* New York: John Wiley and Sons, Inc.

Schreer Oliver, Englert Roman, Eisert Peter und Tanger Ralf. 2008. Real-Time Vision and Speech Driven Avatars for Multimedia Applications. *IEEE Transactions on Multimedia* 10 (3):352-360

Schulze Hendrik 2007. *Skype: Grenzenlose Kommunikation mit Gefahrenpotenzial*. VoIP magazin.de http://www.voipmagazin.de/magazin/artikel_1307_skype_grenzenlose_kommunikation_mit.html [Zugriff am 2008-01-13]

Schulzrinne Henning, Casner Stephen, Frederick Ron und Jacobson Van. 2003. RFC 3550: RTP: A Transport Protocol for Real-Time Applications. In *Request for Comments, Network Working Group* http://www.ietf.org/rfc/rfc3550.txt [Zugriff am 2009-05-06]

Schunter Matthias. 2000. Optimistic Fair Exchange. Doktorarbeit, Universität des Saarlandes, Saarbrücken

Schweitzer Douglas 2002. *Securing the Network from Malicious Code: A Complete Guide to Defending Against Viruses, Worms, and Trojans*. Indianapolis: Wiley Publishing Inc.

Schwenk Jörg. 2005. *Sicherheit und Kryptographie im Internet. Von sicherer E-Mail bis zu IP-Verschlüsselung* Wiesbaden: Vieweg und Teubner Verlag

Shamir Adi. 1979. How to Share a Secret. *Communications of the ACM* 22 (11):612–613

Sicker Douglas C. und Lookabaugh Tom. 2004. VoIP DDOS takes on a whole new meaning. *ACM Queue* 2 (2006):56-64

Siemens Enterprise Communications Inc. 2009. *OpenScape* http://www.enterprise-communications.siemens.com/northamerica/Products/Applications/Unified%20Communications.aspx [Zugriff am 2009-01-01]

Simon Andrew F. 2006. Computer-Mediated Communication: Task Performance and Satisfaction. *Journal of Social Psychology* 146 (3):349-379

Simon Herbert. 1996. *The Sciences of the Artificial*. Boston: MIT Press

Skype. 2006. SkypeWeb - Developers Notes, PUBDN0002 2006-05-19 https://developer.skype.com/Docs/Web?action=AttachFile&do=get&target=SkypeWebDevNotes.pdf

———. 2008. *Skype - take a deep breath*. Skype Technologies S.A www.skype.com [Zugriff am 2008-03-28]

———. 2009. *Skype Security - Detailed security section*. Skype Technologies S.A http://www.skype.com/security/detailed-security/ [Zugriff am 2009-05-05]

Smith Warren D. 2005. *Cryptography meets voting* http://www.math.temple.edu/~wds/homepage/cryptovot.pdf

Sotillo Samuel. 2006. Zfone: A New Approach for Securing VoIP Communication Infosecwriters.com http://www.infosecwriters.com/texts.php?op=display&id=466 [Zugriff am 2008-06-05]

Stedman Ryan, Yoshida Kayo und Goldberg Ian. 2008. A user study of off-the-record messaging. 4th Symposium on Usable Privacy and Security Symposium, SOUPS 2008, Pittsburgh, Pennsylvania, ACM Press

Stein Stefan und Hampe J. Felix. 2008. Providing Spontaneous WLAN Guest Access as a Mobile Value Added Service. 16. European Conference on Information Systems, ECIS 08, Galway, Ireland, 110-120 http://is2.lse.ac.uk/asp/aspecis/default5.asp

Steinmetz Ralf. 2000. *Multimedia-Technologie. Grundlagen, Komponenten und Systeme*. Berlin, Heidelberg: Springer Verlag

Steinmetz Ralf und Wehrle Klaus eds. 2005. *Peer-to-Peer Systems and Applications*, *Lecture Notes in Computer Science 3485/2005*. Berlin, Heidelberg: Springer Verlag

Stieger Stefan und Göritz Anja S. . 2006. Using Instant Messaging for Internet-Based Interviews. *CyberPsychology & Behavior* 9 (5):552-559

Stone John und Merrion Sarah. 2004. Instant Messaging or Instant Headache? . *ACM Queue* 2/2:72-80

Sun Microsystems Inc. 2008. *NetBeans* http://www.netbeans.org/ [Zugriff am 2008-09-07]

———. 2009. *Java Media Framework API (JMF)* http://java.sun.com/javase/technologies/desktop/media/jmf/ [Zugriff am 2008-11-06]

Swartz Nikki. 2005. Companies must manage IM, study says. *Information Management Journal* 39 (1):10-10

Teufel S., Sauter C., Muehlherr T. und Baucknecht K. 1995. *Computerunterstützung für die Gruppenarbeit*. Bonn: Addison-Wesley Verlag

The Eclipse Foundation. 2009. *Eclipse Galileo* http://www.eclipse.org/galileo/ [Zugriff am 2009-06-06]

Thermos Peter und Takanen Ari. 2007. *Securing VoIP Networks - Threats, Vulnerabilities, and Countermeasures*: Addison-Wesley Verlag

Tor. 2008. *The Tor Project* http://www.torproject.org/index.html.de [Zugriff am 2008-11-05]

Tran Minh Hong, Yang Yun und Raikundalia Gitesh K. . 2005. Supporting Awareness in Instant Messaging: An Empirical Study and Mechanism Design. Annual conference of the Australasian Computer-Human Interaction Special Interest Group OZCHI 2005, Canberra, Australia

Trick Ulrich und Weber Frank. 2005. *SIP, TCP/IP und Telekommunikationsnetze - Next Generation Networks und VoIP - konkret*: Oldenbourg Verlag

Trillian-Messenger.net. 2009. *Trillian - Your unofficial Trillian Resource Center* www.trillian-messenger.net/de [Zugriff am 2009-05-05]

Tümmler Jörn 2007. Avatare in Echtzeitsimulationen. Doktorarbeit, Fachbereich Elektrotechnik/Informatik, Universität Kassel, Kassel

Turek Melanie. 2004. Instant Messaging: Time For IT To Pay Attention. *Business Communications Review* Januar 2004:50-53

Universität Koblenz-Landau. 2006. Grundordnung der Universität Koblenz-Landau Universität Koblenz-Landau https://www.uni-koblenz.de/gesetze/dateien/uni/uni_grundo_20060323.pdf [Zugriff am 2008-27-22]

———. 2008. Wahlordnung für die Wahlen der Organe der Universität Koblenz-Landau, 21 Staatsanzeiger Staatsanzeiger (Hrsg.) http://www.uni-koblenz-landau.de/verwaltung [Zugriff am 2008-06-06]

Vaishnavi Vijay K. und Kuechler William. 2004/05. *Design Research in Information Systems* http://home.aisnet.org/displaycommon.cfm?an=1&subarticlenbr=279 [Zugriff am 2008-09-09]

Volkamer Melanie 2009. *Evaluation of Electronic Voting Requirements and Evaluation Procedures to Support Responsible Election Authorities*. W. van der Aalst, J. Mylopoulos, N. M. Sadeh, M. J. Shaw und C. Szyperski (Hrsg.). Vol. 30, *Lecture Notes in Business Information Processing*: Springer Verlag

Volkamer Melanie und Gessner Sandra. 2003. E-Mail-basierendes Wahlsystem für Universitätsgremien, Sicherheit und Kryptographie, Universität Saarland, Saarbrücken http://www2.dfki.de/fuse/content/blogcategory/17/31/

Volkamer Melanie und Grimm Rüdiger. 2009. Determine the Resilience of Evaluated Internet Voting Systems. First International Workshop on Requirements Engineering for E-voting Systems (RE-Vote09), Atlanta, Georgia

Volkamer Melanie und Vogt Roland. 2008. Common Criteria Schutzprofil für Basissatz von Sicherheitsanforderungen an Online-Wahlprodukte, BSI-CC-PP-0037 Bundesamt für Sicherheit in der Informationstechnik http://www.bsi.bund.de/zertifiz/zert/reporte/pp0037b.pdf [Zugriff am 2008-06-30]

Wang Weihong 2009. Digital Video Forensics. Ph.D. Dissertation, Department of Computer Science, Dartmouth College, Hanover http://www.cs.dartmouth.edu/farid/publications/

WASTE. 2008. *waste - anonymous, secure, encryped sharing* http://waste.sourceforge.net/ [Zugriff am 2008-11-05]

Weddeling Sonja, Volkamer Melanie, Paulsen Christian, Młyńczak Katarzyna , Meletiadou Anastasia, Meissner Nils, Krimmer Robert und Helbach Jörg. 2007. Verifiability in Electronic Voting - An Interdisciplinary view. 2. Internationales Rechtsinformatik Symposion, IRIS 2008, Universität Salzburg

Wenger Daniel, Merten Patrik und Teufel Stefanie. 2006. Sichere elektronische Geschäftsprozesse via Voice over IP. D.A.CH Security Conference, Düsseldorf, Germany, 422-432, syssec Verlag

Whitman Michael E. und Mattord Herbert J. 2004. *Principles of Information Security*: Cengage Learning Services

Whittaker Steve, Frohlich David und Daly-Jones Owen. 1994. Informal workplace communication: What is it like and how might we support it? 1994 SIGCHI Conference on Human Factors in Computing Systems, 131-137, ACM Press http://www.cc.gatech.edu/ccg/paper_of_week/p131-whittaker.pdf [Zugriff am 2008-06-03]

Williams Nigel und Ly Joanne 2004. Securing Public Instant Messaging (IM) At Work, Technical Report 040726A Swinburne University of Technology, Centre for Advanced Internet Architectures Melbourne, Australia http://caia.swin.edu.au/reports/040726A/CAIA-TR-040726A.pdf [Zugriff am 2008-02-05]

Williamson Matthew, Parry Alan und Byde Andrew. 2004. Virus throttling for Instant Messaging. Virus Bulletin Conference VB2004, Chicago, USA http://www.hpl.hp.com/techreports/2004/HPL-2004-81.html [Zugriff am 2008-05-06]

xmpp.org. 2009. *XMPP Standards Foundation* http://xmpp.org/ [Zugriff am 2009-09-09]

Yahoo! 2009. *Yahoo! Messenger,* http://messenger.yahoo.com/ [Zugriff am 2009-05-05]

Yang Chung-Huang, Kuo Tzong-Yih, Ahn TaeNam und Lee Chia-Pei. 2008. Design and Implementation of a Secure Instant Messaging Service based on Elliptic-Curve Cryptography. *Journal of Computers* 18 (4)

Zalinger Jason, Freier Nathan und Dutko Eric. 2009. Ethnochat: an instant messenger program for ethnography. 27th international conference extended abstracts on Human factors in computing systems, Boston, MA, USA

Zeilenga Kurt. 2006. RFC 4505: Anonymous Simple Authentication and Security Layer (SASL) Mechanism. In *Request for Comments, Network Working Group* http://tools.ietf.org/html/rfc4505 [Zugriff am 2009-02-03]

Zhou Lina. 2005. An empirical investigation of deception behavior in instant messaging. *IEEE Transactions on Professional Communication* 48 (2):147- 160

Zhu Larry, Jaganathan Karthik und Hartman Sam. 2005. RFC 4121: The Kerberos Version 5, Generic Security Service Application Program Interface (GSS-API). In *Request for Comments, Network Working Group* http://tools.ietf.org/html/rfc4121 [Zugriff am 2009-02-02]

Zimmermann Philip. 2006. *The Zfone™ Project* http://zfoneproject.com/index.html [Zugriff am 2007-06-05]

Zimmermann Philip, Johnston Alan und Callas Jon. 2007. Internet-Draft: ZRTP - Media Path Key Agreement for Secure RTP. In *Network Working Group* http://tools.ietf.org/html/draft-zimmermann-avt-zrtp-03 [Zugriff am 2008-05-15]