Ad-hoc Chatsystem für mobile Netze

Gruppe 3 (Adbee)

Softwareentwicklungspraktikum
Sommersemester 2007

Testdokumentation

Auftraggeber
Technische Universität Braunschweig
Institut für Betriebssysteme und Rechnerverbund
Prof. Dr.-Ing. Lars Wolf
Mühlenpfordtstraße 23, 1. OG
38106 Braunschweig
Deutschland

Betreuer: Sven Lahde, Oliver Wellnitz
Hiwi: Wolf-Bastian Pöttner

Auftragnehmer: Gruppe 3

Name	E - Mail
Ekrem Özmen	e.oezmen@gmail.com
Celal Özyalcin	c.oezyalcin@tu-bs.de
Thorben Schulze	thorben.schulze@tu-bs.de
Danny Melching	danny.melching@tu-bs.de

Phasenverantwortlicher: Danny Melching

Braunschweig, 07.07.2007

Versionsübersicht

Version	Datum	Autor	Status	Kommentar
1.0	09.07.07	Ekrem, Celal, Danny, Thorben		Erste Version der Testdokumentation

I n h a l t s v e r z e i c h n i s

1 Testplan

1.1 Zu testende Komponenten

GUI

Nachricht

Routing

SendenEmpfangen

Sicherheit

Verwaltung

1.2 Zu testende Funktionalitäten

/F10/ Nachrichten verschlüsselt im geschlossenen Kanal senden
/F20/ Nachrichten im anonymen Kanal senden
/F30/ Nachrichten unverschlüsselt im offenen Kanal senden
/F40/ Nachrichten im geschlossenen Kanal empfangen
/F50/ Nachrichten im anonymen Kanal empfangen
/F60/ Nachrichten im offenen Kanal empfangen
/F70/ Nachrichten weiterleiten
/F80/ Geschlossenen Kanal erstellen
/F90/ Offenen Kanal erstellen
/F95/ Kanäle anzeigen
/F100/ Kanal verlassen
/F110/ Geschlossenen Kanal beitreten
/F120/ Offenen Kanal beitreten
/F130/ Jemand anderen in einen geschlossenen Kanal einladen
/F140/ Zertifikat anfordern
/F150/ Zertifikat versenden
/F160/ Gemeinsamen Schlüssel für geschlossenen Kanal anfordern
/F180/ Aktualisieren der Netzstruktur
/F170/ Gemeinsamen Schlüssel für geschlossenen Kanal versenden
/F175/ Speichern und späteres erneutes Senden von Nachrichten
/F190/ In den Infrastrukturmodus wechseln
/F200/ Partitionierung und Verschmelzung von zwei Netzen

/W10/ Jeder Benutzer kann eine Liste aller gerade im Netz vorhandenen Benutzer anzeigen lassen

Die Musskriteren werden durch diese Funktonalitäten oder teilweise durch den JUnitTest der Komponenten abgedeckt. Allgemein decken die Testfälle in Kapitel 3 alle Funktionen, außer /F175/, ab. Diese wird durch den JUnit-KomponentenTest abgedeckt, weil ein Nachrichtenverlust ohne Beeinflussung des Programms schwer zu simulieren ist.
Die Gewährleistung der Interoperabilität erfolgt durch einen Interoperabilitätstest mit den anderen Gruppen. Die Funktion /W10/ ist dadurch erfüllt, dass die Liste aller gerade im Netz vorhandenen Benutzer der Teilnehmerliste des anonymen Kanal entspricht.

1.3 Nicht zu testende Funktionalitäten

/F210/ Peerverwaltungsliste
/F220/ Teilnehmerliste
/F230/ Kanalliste
/F240/ Kanalteilnehmerliste
/W20/ Neue Funktionen, die die Handhabung erleichtern oder erweitern, sollten sich möglichst einfach einbinden lassen
/W30/ Eine Portierung auf andere Betriebssysteme sollte auch ohne größere Änderungen erfolgen können
/W40/ Die Benutzeroberfläche sollte vom jeweiligen Benutzer in sinnvollem maß veränderbar sein

Die Listen (/F210/ - /F240/) werden teilweise durch die anderen Testfälle und die JUnit-Tests abgedeckt und brauchen deshalb keinen extra Testfall.

1.4 Vorgehen

Der Komponententest erfolgt durch JUnit-Tesfälle. In den Tests werden die Klassen mit deren Methoden selbst und, falls Objekte durch mehrere Klassen in einer Komponente laufen, die Komponenten im Ganzen getestet. Vor jedem Testfall sollten die JUnit-Testfälle erfolgreich durchgeführt worden sein. Ebenfalls dienen sie, um nach Änderungen im Quellcode, die bisherigen Methoden-Funktionalitäten zu gewährleisten. Da die Komponenten teilweise Klassen und Informationen, der anderen Komponenten benötigen, sind für die Komponenten keine Blackbox-Testfälle vorhanden. Bei den JUnit-Testfällen handelt es sich deshalb vorwiegend um Whitebox-Testfälle. Die Komponententests testen hauptsächlich die

umfangreichen Methoden, die Fehler werfen können. Get- und Set-Methoden sind nicht getestet.
Der Integrationstest erfolgt ebenfalls durch JUnit-Testfälle. Dabei werden die Schnittstellenklassen in den Testpaketen abgeleitet und bei den Testläufen ersetzen diese die richtigen Klassen. Mit diesen wird dann geprüft, ob Objekte korrekt an andere Komponenten weitergegeben worden sind. Ebenfalls wird beim Testen der Funktionalitäten(Kapitel 4) die Integration der Komponenten geprüft.
Der Funktionstest erfolgt direkt mit dem vollständigen Programm, mit dem in Kapitel 2 beschriebenen Testskript und in Kapitel 3 beschriebenen Testfällen.
Der Interoperabilitätstest entspricht dem Funktionstest. Er wird jedoch mit Programmen der anderen Gruppen ausgeführt.

1.5 Kriterien für erfolgreiche bzw. fehlgeschlagene Testläufe

Kriterien für einen erfolgreichen Testlauf sind die Erfüllung der getesteten Funktionalität und das Abfangen und Anzeigen von falschen Eingaben durch den Benutzer. Kriterien für einen fehlgeschlagenen Testlauf sind Programmabbrüche, Fehlermeldungen oder fehlgeschlagene Testläufe durch die beim Testen verwendete Entwicklungsumgebung.

1.6 Testumgebung

Die Tests werden unter der Entwicklungsumgebung Eclipse 3.2 durchgeführt. Die Klassen werden mit JUnit 4.1 getestet. Zum Testen der Funktionalitäten dient der Infrastruktur Modus, in dem eine virtuelle Netzstruktur aufgebaut werden kann. Getestet wird auf den Betriebssystemen Ubuntu Linux, Mac OS X und Microsoft Windows XP.

2 Testskript

Für die Durchführung der Tests werden je nach Test mindestens zwei Instanzen auf einem oder mehreren Rechnern gestartet. Beim Start wird der Infrastruktur Modus eingeschaltet und der Empfangsport gewählt. Anschließend werden durch Angabe von IP und Port im Infrastrukturmodus-Fenster die Instanzen untereinander sichtbar gemacht. Nun können die jeweiligen Tests durchgeführt werden. Verliefen die Tests erfolgreich, werden sie ohne Infrastrukturmodus mit den Programmen der anderen Gruppen wiederholt, um die Interoperabilität zu gewährleisten.

2.1.1 Testziel

Sinn dieses Dokumentes ist es, alle verbleibenden Fehler im laufenden Programm zu finden, dazu dienen folgende Testfälle.

/T10/ Nachrichten Übermittlung /F10/ - /F70/
/T20/ Kanäle erstellen/betreten/verlassen /F80/ - /F130/
/T30/ Zertifikat/Schlüssel anfordern/versenden /F140/ - /F170/
/T40/ Partitionierung und Verschmelzung von zwei Netzen /F180/ /F200/
/T50/ Falsche Benutzer Eingaben abfangen

2.1.2 Spezielle Anforderungen

Die einzelnen Instanzen sollten korrekt gestartet sein. D.h. die Zertifikate und privaten Schlüssel für die einzelnen Instanzen sollten gewählt sein und nicht doppelt verwendet werden. Ebenfalls ist ein eindeutiger Empfangsport pro Instanz zu wählen, wenn zwei oder mehr Instanzen auf einen Rechner gestartet werden.

2.1.3 Testablauf

Dies soll ein allgemeingültiges Testskript sein, was bedeutet, dass bei den einzelnen Tests unterschiedlich viele Instanzen des Programms gestartet werden müssen und je nach Test auch unterschiedliche Netze aufgebaut werden. Zum Beispiel müssen sich beim Senden im Anonymen Kanal, wenn das Obscure senden getestet werden soll, mindestens 3 Instanzen/Benutzer in diesem befinden. Möchte man die Verschmelzung von zwei Netzen testen, sollten mindestens 4 Instanzen gestartet werden und zwar jeweils zwei die sich

gegenseitig sehen. Dann müssen gleichnamige Kanäle erstellt werden und die zwei Instanz-Blöcke dann gegenseitig sichtbar gemacht werden.

- Vorbereitung
    - Aktuelle Revision des SVN /trunk/Implementierung unter Eclipse einrichten
    - Eclipse auf Java 1.5 einstellen
    - Die nötigen externen *.jar's einbinden
        - bcprov-jdk15-136.jar
        - jdom.jar (v. 1.0)
        - (JUnit 4.1)

- Ausführung
    - Die nötige Anzahl von Instanzen starten
    - Über „Menu" -> „Start das Programm" starten, hier Zertifikat und privaten Schlüssel wählen, Infrastruktur Modus isOn wählen und Empfangsport wählen
    - Über „Settings" -> „Infrastruktur Modus" unter Angabe von IP und Port die Instanzen sichtbar machen

- Beobachtung
    - Zu beobachten ist, ob die gesendeten Nachrichten korrekt empfangen und verarbeitet werden (bspw. Kanal erstellen, erscheint dieser auch in den anderen Instanz)
    - Die Eclipse Console dient ebenfalls zur Beobachtung, hier werden alle wichtigen gesendeten und empfangenen Messages ausgegeben

- Abbruch
    - Sollte es während des Test zu Fehlern kommen die entweder in der Console sichtbar sind oder ein erwartetes Ereignis nicht in im Programm eingetreten ist wird, wird der Test gestoppt der Fehler repariert oder gegebenenfalls der Verantwortlich der Komponente informiert, in der der Fehler aufgetreten ist und anschließen der Test wiederholt.

3 Testfall

Im Folgenden finden sich die einzelnen Testfälle, das genaue vorgehen ist im Punkt Eingaben beschrieben.

3.1 Testfall /T10/

Nachrichten Übermittlung /F10/ - /F70/

3.1.1 Eingaben

Zwei Instanzen werden gestartet und gegenseitig sichtbar gemacht, anschließend wird ein offener Kanal erstellt, dem beide beitreten. Nun werden von jeder Instanz Nachrichten im anonymen, offenen Kanal oder geschlossenen Kanal an die andere geschickt. Es wird eine dritte Instanz gestartet, die nur mit einer der beiden vorherigen sichtbar gemacht wird, es wird wieder von jeder zu jeder Instanz gesendet.

3.1.2 Ausgaben

Jede gesendete Nachricht sollte bei den anderen Instanzen angezeigt werden, so ist sichergestellt, dass das Senden und Empfangen funktioniert und zwar direkt, Hop-by-Hop und obscure. Letzteres wird über die Eclipse Console ersichtlich. Anhänge gehören ebenfalls zu Nachrichten, wobei die Vollständigkeit der übertragenen Dateien geprüft wird.

3.1.3 Testumgebung

Eclipse 3.2 auf den Plattformen Ubuntu Linux, Mac OS X und Microsoft Windows XP unter Verwendung des Infrastruktur Modus.

3.1.4 Abhängigkeiten

Der Testfall ist von /T20/ und /T30/ abhängig, da Kanäle erstellt werden müssen und für das empfangen von Nachrichten im offenen Kanal die Zertifikate des Senders vorliegen müssen. Ebenfalls muss für den geschlossenen Kanal der gemeinsame Schlüssel vorhanden sein.

3.2 Testfall /T20/

Kanäle erstellen/betreten/verlassen /F80/ - /F130/

3.2.1 Eingaben

Drei Instanzen werden gestartet und in einer Pipe (a sieht b, b sieht a und c, c sieht b) sichtbar gemacht. Anschließend erstellt eine beliebige Instanz einen Kanal und beide verbleibenden treten bei, an dieser Stelle sollte ein Nachrichten austausch stattfinden. Dann verlassen alle Instanzen diesen Kanal wieder.

3.2.2 Ausgaben

Ist ein Kanal erstellt worden, sollte den anderen Instanzen dieser in der Kanalliste angezeigt werden mit einem leeren Rechteck, nach dem Beitritt mit einem vollen Rechteck und ein Reiter für diesen Kanal sollte erstellt sein. Nachrichten für diesen Kanal sollten nun empfangen werden können. Befinden sich keine Teilnehmer mehr in dem Kanal, so sollte er aus der Kanalliste verschwinden und es sollten keine Channel Announcements mehr für diesen Kanal verschickt werden.

3.2.3 Testumgebung

Eclipse 3.2 auf den Plattformen Ubuntu Linux, Mac OS X und Microsoft Windows XP unter Verwendung des Infrastruktur Modus.

3.2.4 Abhängigkeiten

Der Testfall ist teilweise von /T10/ abhängig, da zur Sicherstellung eines gelungenen Tests Nachrichten verschickt werden sollten.

3.3 Testfall /T30/

Zertifikat/Schlüssel anfordern/versenden /F140/ - /F170/

3.3.1 Eingaben

Es können beliebig viele Instanzen(mindestens 2, bei Anonym mindestens 3) erstellt und gegenseitig sichtbar gemacht werden. Der Ordner „data“ in dem Programmverzeichnis muss leer sein, da dort alle Zertifikate liegen die angefordert worden sind. Es erfolgt ein Beitreten in einen Kanal und dort wird eine Nachricht gesendet oder es wird direkt im Anonymen Kanal gesendet. Für den Key wird dies im geschlossenen Kanal getan.

3.3.2 Ausgaben

In der Konsole muss ein Austausch von Anforderungs-Nachrichten erkennbar sein. Ebenfalls werden entsprechende Zertifikate im Ordner Data abgelegt. Falls Zertifikate nicht vorhanden sind muss das Senden einer Nachricht diese spätestens anfordern.
Für den Austausch des Key für den geschlossenen Kanal sind ebenfalls Konsolenausgaben vorhanden. Ein Erkennen der Nachricht beim jeweils anderen sollte aber für den Test des Austausches genügen.

3.3.3 Testumgebung

Eclipse 3.2 auf den Plattformen Ubuntu Linux, Mac OS X und Microsoft Windows XP unter Verwendung des Infrastruktur Modus.

3.3.4 Abhängigkeiten

Der Testfall ist von /T10/ abhängig, da zur Sicherstellung eines gelungenen Tests Nachrichten verschickt werden sollten.

3.4 Testfall /T40/

Partitionierung und Verschmelzung von zwei Netzen /F200/

3.4.1 Eingaben

Es werden 5 Instanzen gestartet, jeweils 2 können sich gegenseitig sehen, die fünfte bleibt zunächst „ungesehen“. Die beiden Instanz Blöcke erstellen jeweils einen geschlossenen und einen offenen Kanal, mit gleichen Namen wie der andere Instanzblock. Nun wird Instanz 5 für beide Blöcke sichtbar gemacht, Nachrichten werden verschickt und Instanz 5 wird wieder aus dem Netz entfernt.

3.4.2 Ausgaben

Nachdem Instanz 5 für beide Blöcke sichtbar ist, erhalten alle Instanzen Channel Announcements für Kanäle die bei Ihnen lokal schon existieren aber mit anderer ID, nun sollten offene Kanäle gemerged werden und geschlossene Kanäle weiterhin getrennt behandelt werden. Resultierend daraus sollten in der Kanalliste ein jeweils neuer Kanal erscheinen mit dem Namen des schon vorhandenen Kanals plus [„Nummer“] dahinter. Der offene Kanal sollte einmal bestehen bleiben und alle Nachrichten die in diesem Kanal verschickt werden sollte jede Instanz erreichen.
Nachdem Instanz 5 das Netz wieder verlassen hat sollten die Kanäle die Nummern tragen wieder verschwinden und die Teilnehmerliste des offenen Kanals entsprechend aktualisiert sein.

3.4.3 Testumgebung

Eclipse 3.2 auf den Plattformen Ubuntu Linux, Mac OS X und Microsoft Windows XP unter Verwendung des Infrastruktur Modus.

3.4.4 Abhängigkeiten

Der Testfall ist von /T10/, /T20/ und /T30/ abhängig, da Kanäle erstellt werden müssen, Nachrichten verschickt und für das empfangen von Nachrichten im offenen Kanal die Zertifikate des Senders vorliegen müssen.

3.5 Testfall /T50/

Falsche Benutzer Eingaben abfangen

3.5.1 Eingaben

Es müssen zwei Instanzen gestartet werden, wobei alle eingaben falsch getätigt werden

- Ungültiges Zertifikat und privaten Schlüssel wählen
- Ungültigen Port wählen
- Kanal erstellen mit Namen, der nicht vorgaben konform ist, oder Description die nicht UTF-8 konform ist
- Kanal erstellen mit Namen, der schon vorhanden ist
- Falsche Eingaben im Infrastruktur Fenster tätigen
- Nicht UTF-8 konforme Nachricht eingeben und verschicken

3.5.2 Ausgaben

Alle falsch eingaben sollten dem Benutzer angezeigt werden und ihn zur neu Eingabe zwingen, es darf keine falsche Eingabe vom Programm akzeptiert werden. Nicht konforme Text Eingabe beim versenden von Nachrichten sollte automatisch umgeformt werden.

3.5.3 Testumgebung

Eclipse 3.2 auf den Plattformen Ubuntu Linux, Mac OS X und Microsoft Windows XP unter Verwendung des Infrastruktur Modus.

3.5.4 Abhängigkeiten

Dieser Test setzt voraus, dass alle anderen Tests größtenteils positiv verlaufen sind. Bspw. Macht es keinen Sinn einen Kanal zu erstellen mit ungültigem Namen, wenn noch gar nicht sichergestellt ist, dass ein Kanal erstellt werden kann.

4 Testprotokoll

Das Testprotokoll enthält Angaben über alle ausgeführten Testfälle, deren Ergebnisse und evtl. Abweichungen vom erwarteten Ergebnis. Die Plattformen für die Tests sind Ubuntu Linux, Mac OS X und Microsoft Windows XP.

4.1 Testprotokoll /P1/

4.1.1 Beschreibung

Die Testdurchführung erfolgte gemäß Testskript und Beschreibung des jeweiligen Testfalls. Alle hier durchgeführten Tests werden im Infrastruktur Modus bearbeitet.

4.1.2 Aufstellung der durchgeführten Testfälle

- /T10/
- /T20/
- /T30/
- /T40/
- /T50/

4.1.3.1 Ausführung

- /T10/
- Nachrichten Übermittlung in allen drei Kanaltypen, Netzstruktur zunächst mit zwei Instanzen anschließend mit drei in einer Pipe

4.1.3.2 Ergebnis

Der Test verlief erfolgreich es gab keine erkennbaren Fehler, alle Nachrichten wurden korrekt übertragen.

4.1.3.3 Unerwartete Ereignisse

Es traten keine unerwarteten Ereignisse auf.

4.1.4.1 Ausführung

- /T20/
- Kanäle erstellen/betreten/verlassen für je einen offenen und geschlossenen Kanal mit drei Instanzen in einer Pipe

4.1.4.2 Ergebnis

Jeder Kanaltyp konnte erstellt werden, dieser wurde den anderen Instanzen korrekt angezeigt. Betreten und Verlassen durch andere Instanzen war ebenfalls möglich, sowie auch Einladen in einen geschlossenen Kanal. Damit war der Test erfolgreich.

4.1.4.3 Unerwartete Ereignisse

Beim Verlassen eines geschlossenen Kanals, der keine weiteren Teilnehmer mehr beinhaltet, wird dieser nicht von der Kanalliste gestrichen, dies liegt daran, dass beim Verlassen eines solchen man sich nicht wirklich aus dem Kanal entfernt, sondern nur den dazugehörigen Reiter zu diesem entfernt und keine Nachricht mehr erhält. Grund dafür ist, dass man eine Einladung in diesen Kanal nicht wieder Rückgängig macht.

4.1.5.1 Ausführung

- /T30/
- Zertifikat/Schlüssel anfordern/versenden in einer Pipe und in einem vollvermaschten Netz mit je drei Instanzen
- Es wurde ein geschlossener Kanal erstellt, die beiden anderen Instanzen eingeladen und dann von jeder zu jeder Instanz gesendet.

4.1.5.2 Ergebnis

Der Test in einem vollvermaschten Netz verlief erfolgreich.

4.1.5.3 Unerwartete Ereignisse

In der Pipe kam es zu einem Fehler beim weiterleiten der GetKeyMessage, dieser konnte unmittelbar behoben werden und in einem anschließenden Test traten keine Fehler mehr auf.

4.1.6.1 Ausführung

- /T40/
- Partitionierung und Verschmelzung von zwei Netzen

4.1.6.2 Ergebnis

Die Kanäle konnten korrekt gemerged, bzw. bei geschlossenen Kanälen mit neuem internen Namen getrennt weiter behandelt werden.

4.1.6.3 Unerwartete Ereignisse

Beim Beenden der Instanz, die die beiden Netze verbindet, erfolgte keine Partitionierung, es war zu beobachten, dass sich die hops zu dieser Instanz immer weiter erhöht haben. Dies liegt daran, dass bspw. Instanz a Instanz b und Instanz c über einen Hop erreicht, somit erreicht b c über 2 Hops, verlässt nun b das Netz kann a b nicht mehr erreichen, bekommt aber von c die Nachricht, dass c b über 2 Hops erreichen kann und baut nun b in die Routingtabelle so ein, als könne sie b über drei Hops erreichen, so kommt es zu einem Count-To-Infinity Problem.

4.1.7.1 Ausführung

- /T50/
- Falsche Benutzer Eingaben abfangen

4.1.7.2 Ergebnis

Der Test verlief erfolgreich, nahezu alle Benutzer Angaben konnten entsprechend abgefangen werden.

4.1.7.3 Unerwartete Ereignisse

Bei der Eingabe eines falschen Empfangsports für den Infrastruktur Modus wurde dem Benutzer zwar angezeigt, dass dieser ungültig ist, aber er wurde trotzdem an die anderen Komponenten weitergeleitet.

4.2 Testprotokoll /P2/

Das Protokoll protokolliert die Interoperabilitätstests

4.2.1 Beschreibung

Die Testdurchführung erfolgte gemäß Testskript und Beschreibung des jeweiligen Testfalls. Alle hier durchgeführten Tests werden mit den Programmen der anderen Gruppen ausgeführt. Teilweise laufen diese im Infrastrukturmodus und teilweise im normalen Modus ab. Die Tests sind hauptsächlich mit zwei Instanzen ausgeführt worden.

4.2.2 Aufstellung der durchgeführten Testfälle

- /T10/
- /T20/
- /T30/

Testfälle worden mit Gruppe 1 und 2 durchgeführt, wobei manches wegen fehlender Implementierungen noch nicht möglich gewesen ist. Tests mit Gruppe 4 sind noch nicht durchgeführt worden.

4.2.3.1 Ausführung

- /T10/
- Test mit Gruppe 1
- Nachrichten Übermittlung in allen drei Kanaltypen, Netzstruktur sind zwei Knoten die sich sehen.

4.2.3.2 Ergebnis

Der Test verlief in offenen und geschlossenen Kanälen erfolgreich. Ebenfalls war das Senden und Empfangen von Nachrichten mit Anhängen in geschlossenen und offenen Kanälen möglich.

4.2.3.3 Unerwartete Ereignisse

Ein Testen des anonymen Kanals war zum Testzeitpunkt noch nicht möglich.

4.2.4.1 Ausführung

- /T20/
- Test mit Gruppe 1
- Kanäle erstellen/betreten/verlassen für je einen offenen und geschlossenen Kanal mit zwei Programmen

4.2.4.2 Ergebnis

Jeder Kanaltyp konnte erstellt werden und wurde den anderen Instanzen korrekt angezeigt. Das Beitreten und Verlassen durch andere Instanzen war ebenfalls möglich. Außerdem funktionierte das Einladen in einen geschlossenen Kanal und die Anfrage zum Einladen.

.

4.2.4.3 Unerwartete Ereignisse

Es trat das gleiche Problem wie im nicht interoperablen Test auf, dass im geschlossenen Kanal keine schon eingeladenen Benutzer mehr aus der Kanalteilnehmerliste entfernt werden.
Außerdem kamen von der anderen Gruppe mehrere Anfragen zum Einladen. Erst durch das Einladen des Benutzers wurden diese gestoppt.

4.2.5.1 Ausführung

- /T30/
- Test mit Gruppe 1
- Testen des Anfordern der Zertifikate und Schlüssel

4.2.5.2 Ergebnis

Der Test verlief erfolgreich es gab keine erkennbaren Fehler, alle Nachrichten wurden korrekt übertragen.

4.2.5.3 Unerwartete Ereignisse

Es traten keine unerwarteten Ereignisse auf.

4.2.6.1 Ausführung

- /T10/
- Test mit Gruppe 2
- Nachrichten Übermittlung in allen drei Kanaltypen. Netzstruktur sind zwei Knoten die sich sehen für offenen und geschlossenen Kanal. Der anonyme Kanal wurde im Infrastrukturmodus mit zwei Instanzen von unserem Programm und einer Instanz von deren Programm in einer dreifachen Pipe getestet.

4.2.6.2 Ergebnis

Der Test verlief in offenen und geschlossenen Kanälen erfolgreich. Man konnte ein entschlüsseln der anonymen Nachricht erkennen. Das verschicken von Anhängen hat teilweise funktioniert. Der Test verlief also nur teilweise erfolgreich ab.

4.2.6.3 Unerwartete Ereignisse

Es traten manchmal Probleme beim Empfang einer Nachricht mit Anhang bei der anderen Gruppe auf.

Die entschlüsselte anonyme Nachricht wurde zwar korrekt ausgegeben, aber kam mehrmals an.

4.2.7.1 Ausführung

- /T20/
- Test mit Gruppe 2
- Kanäle erstellen/betreten/verlassen für je einen offenen und geschlossenen Kanal mit zwei Programmen.

4.2.7.2 Ergebnis

Jeder Kanaltyp konnte erstellt werden und wurde den anderen Instanzen korrekt angezeigt. Das Beitreten und Verlassen durch andere Instanzen war ebenfalls möglich. Außerdem funktionierte das Einladen in einen geschlossenen Kanal und die Anfrage zum Einladen.

.

4.2.7.3 Unerwartete Ereignisse

Es trat das gleiche Problem, wie im nicht interoperablen Test auf, dass im geschlossenen Kanal keine schon eingeladenen Benutzer mehr aus der Kanalteilnehmerliste entfernt werden.

4.2.8.1 Ausführung

- /T30/
- Test mit Gruppe 2
- Testen des Anfordern der Zertifikate und Schlüssel

4.2.8.2 Ergebnis

Der Test verlief erfolgreich es gab keine erkennbaren Fehler, alle Nachrichten wurden korrekt übertragen.

4.2.8.3 Unerwartete Ereignisse

Es traten keine unerwarteten Ereignisse auf

5 Zusammenfassung

Allgemein ist die Qualität des Programms, wenn es nicht mit Programmen der anderen Gruppen kommuniziert, gut. Alle Testfälle wurden ausgeführt und aufgetretene Probleme konnten teilweise oder ganz behoben werden.
Die Interoperabilität besitzt eine mittelmäßige Qualität, weil die meisten Standartfunktionen ausgeführt werden können, aber teilweise noch unerwartete Fehler auftreten. Ebenfalls konnten noch nicht alle Testfälle vollständig ausgeführt werden.
Zu den aufgetretenen Problemen lässt sich folgenden sagen:
Das Problem, dass ein geschlossener Kanal vorhanden bleibt, wenn man ihn als letztes verlässt wurde dadurch beseitigt, dass eine Abfrage eingebaut wurde, mit der geprüft wird, ob man der letzte verbleibende Teilnehmer dieses Kanals ist, dann wird der Kanal wirklich verlassen und somit terminiert.
Die Möglichkeit einen ungültigen Empfangsport einzugeben wurde durch umpositionieren eines if-Blocks erreicht.
Das Count-to-Infinity Problem wurde teilweise dadurch umgangen, dass keine Routinginformationen mehr an einen Zielhost geschickt werden, wenn diese informationen von diesem Zielhost stammen. Dadruch kann es nur noch in größeren Netzen zu diesem Problem kommen.
Nach Beseitigung aller aufgetretenen Probleme wurden die Tests auf allen drei genannten Plattformen wiederholt, dort liefen alle Tests erfolgreich, daher ist es nicht notwendig für diese noch die Testprotokolle aufzuführen.