

Technische Dokumentation – Netzwerk-Chat-Anwendung

Datum: 5. Juli 2025

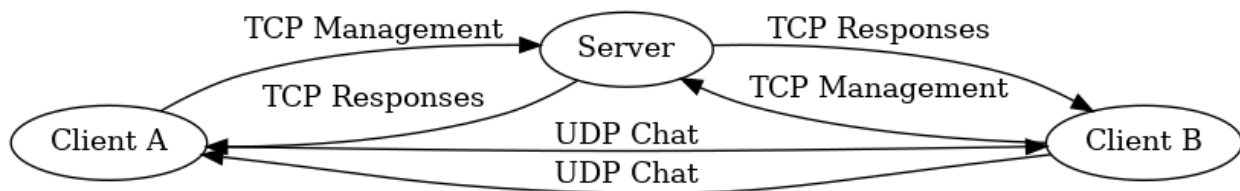
Autor: Yulia Korzhova

1 Einleitung

Die Dokumentation erklärt, wie die Netzwerk-Chat-Anwendung funktioniert. Das System folgt den Vorgaben des Aufgabenblatts "Rechnernetze – Netzwerkschicht". Es setzt eine Client-Server-Architektur mit einem TCP-basierten Management-Protokoll sowie einem UDP-basierten Chat-Protokoll um. Optional wurde eine grafische Benutzeroberfläche (GUI) erstellt.

2 Architekturübersicht

Die Gesamtarchitektur besteht aus einem zentralen Serverprozess und beliebig vielen Client-Instanzen (CLI oder JavaFX-GUI). Steuerbefehle (Registrierung, Login, Einladungen ...) werden über eine dauerhafte TCP-Verbindung ausgetauscht. Sobald zwei Clients einen Chat starten, kommunizieren sie P2P über UDP.



3 Management-Protokoll (TCP)

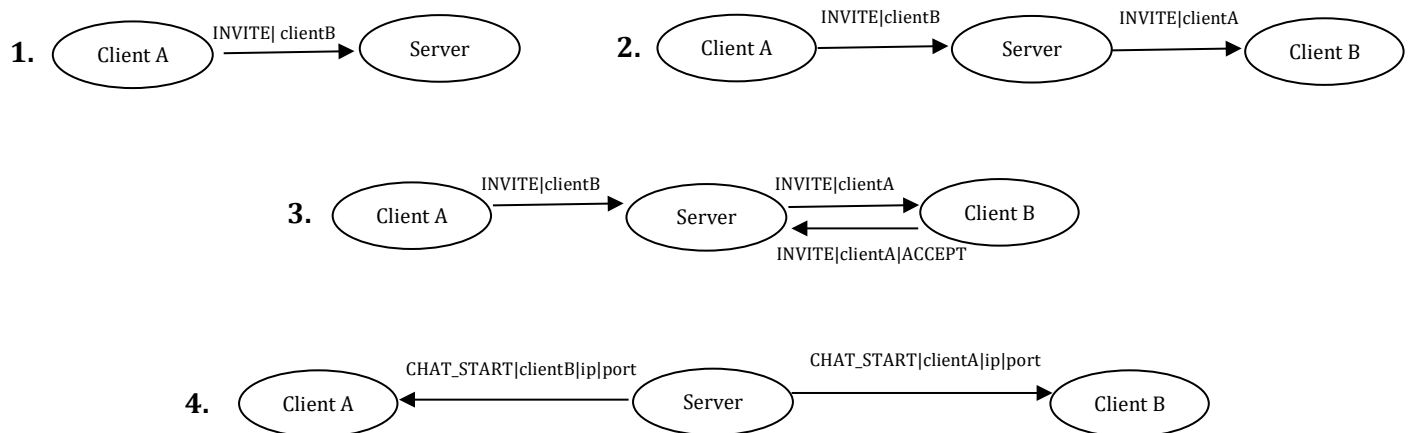
3.1 Ziel

Das Management-Protokoll regelt die Registrierung, die Anmeldung, die Verwaltung aktiver Teilnehmer und das Aushandeln von Chats.

3.2 Nachrichtenformate

Befehl-Typ	Richtung	Beschreibung	Befehl-Beispiel
REGISTER	Client -> Server	Registriert neuen Benutzer	REGISTER alice secret
LOGIN	Client -> Server	Meldet Benutzer an	LOGIN alice secret
UDPPORT	Client → Server	Meldet Port für UDP-Chat	UDPPORT 54321
LIST	Client -> Server	Liste aktiver Nutzer anfordern	LIST
INVITE <target>	Client -> Server	Einladung eines anderen Nutzers	INVITE bob
INVITE_RESP	Client -> Server	Antwort auf Einladung	INVITE_RESP alice ACCEPT
INVITE	Server → Client	Einladung erhalten	INVITE alice
INVITE_RESP	Server → Client	Einladungsantwort weitergeleitet	INVITE_RESP bob ACCEPT
CHAT_START <ip> <port>	Server -> Client	Teilt beiden Clients UDP-Daten mit	CHAT_START bob 10.0.0.2 45678
OK / ERR	Server -> Client	Allgemeine Antworten	

3.3 Ablauf (Sequenzdiagramm)



4 Chat-Protokoll (UDP)

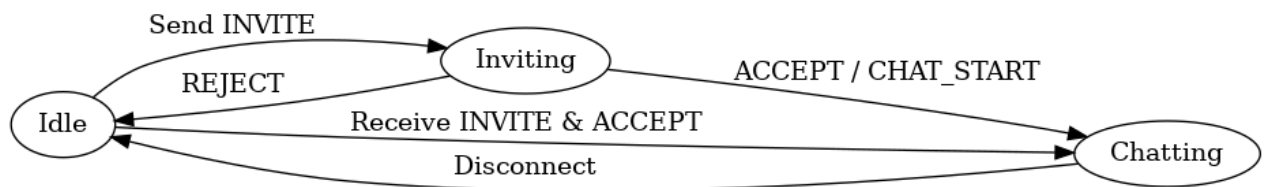
4.1 Paketformat

Textnachrichten werden als UTF-8-Strings versendet. In der Kommandozeilen-Variante werden die Nutzdaten vor dem Versand mit AES-128 verschlüsselt und anschließend Base64-kodiert.

4.2 Zuverlässigkeitsmechanismen

Es existieren keine expliziten ACK/Timeout-Mechanismen; das Protokoll folgt dem Best-Effort-Prinzip und nutzt die Benutzeroberfläche zur Anzeige verlorener Nachrichten.

4.3 Zustandsautomat



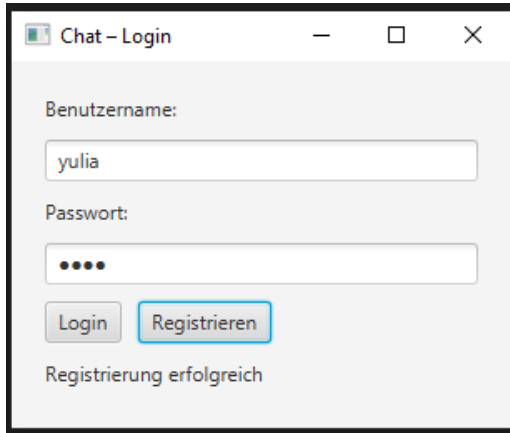
5 Sicherheit

- Passwörter werden auf dem Server mit SHA-256 gehasht gespeichert.
- UDP-Chat-Nachrichten der CLI werden mit einem statischen AES-128-Schlüssel verschlüsselt.
- Das Management-Protokoll wird unverschlüsselt übertragen; Integrität beruht auf TCP.
- Replay-Schutz sowie Perfect-Forward-Secrecy sind nicht implementiert.

6 Benutzeroberfläche (GUI)

Die GUI ist in JavaFX implementiert. Zentrale Elemente:

- Registrierung/ Login
- Anzeige aktiver Nutzer
- Einladung zum Chat
- Dynamische Chatfenster (ChatWindow)



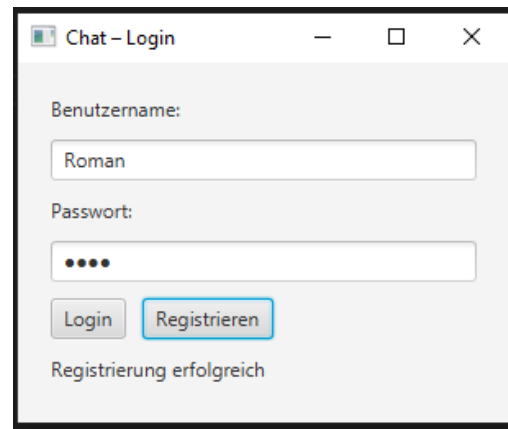
Chat - Login

Benutzername:
yulia

Passwort:
••••

Login Registrieren

Registrierung erfolgreich



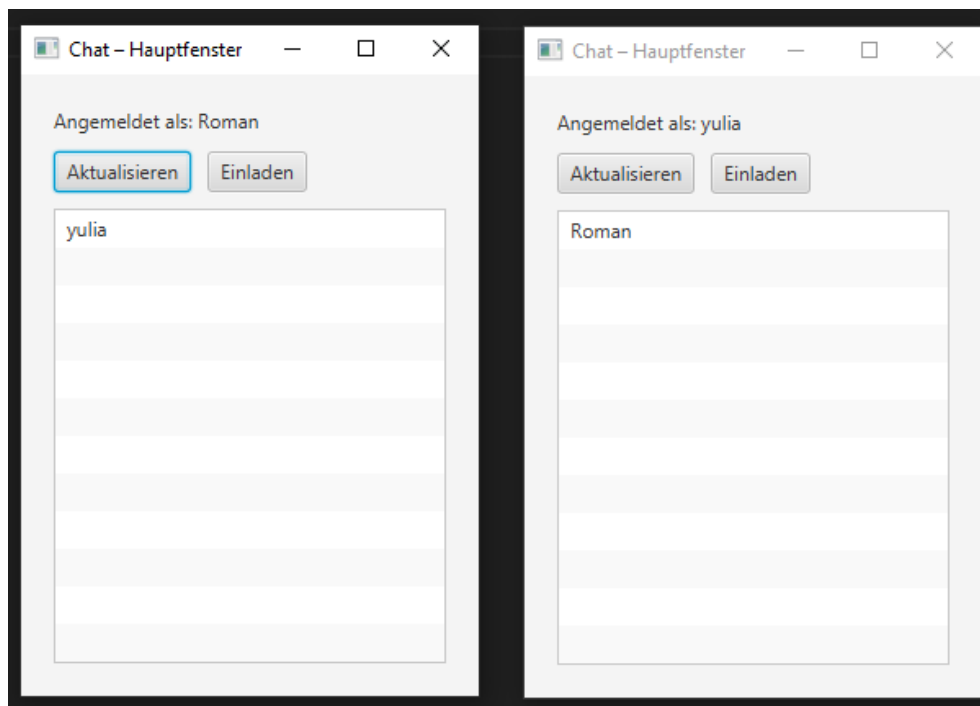
Chat - Login

Benutzername:
Roman

Passwort:
••••

Login Registrieren

Registrierung erfolgreich



Chat - Hauptfenster

Angemeldet als: Roman

Aktualisieren Einladen

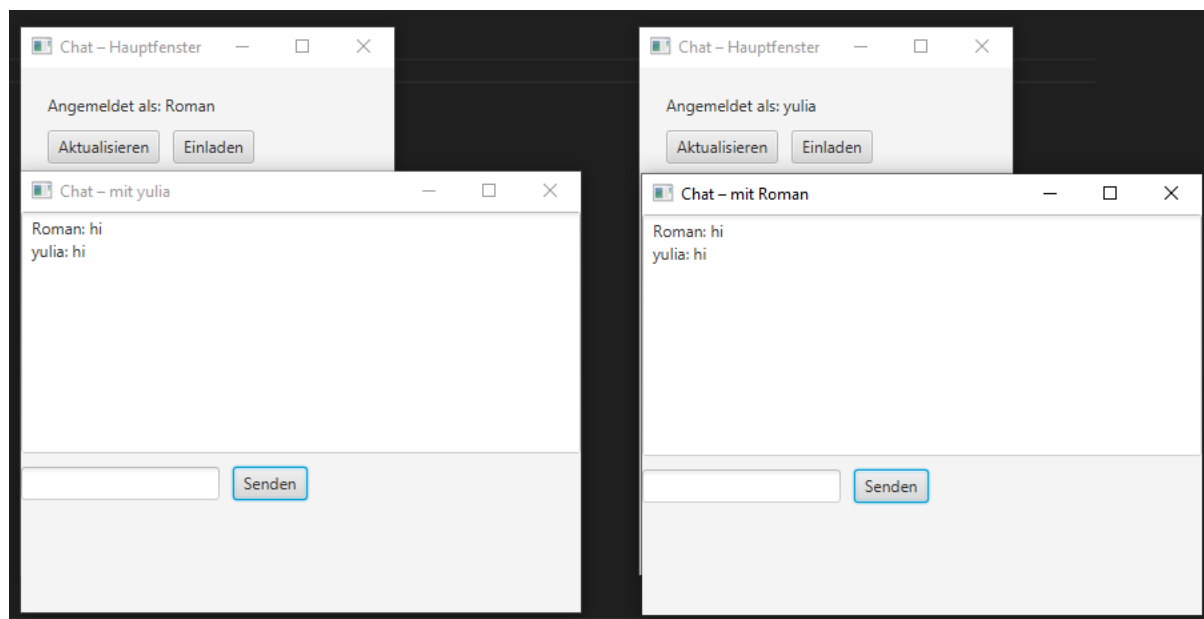
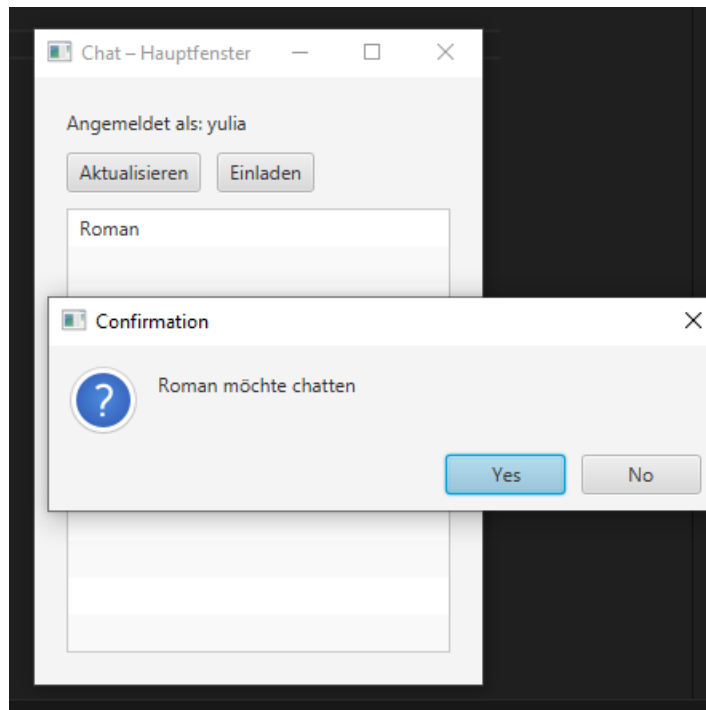
yulia

Chat - Hauptfenster

Angemeldet als: yulia

Aktualisieren Einladen

Roman



7 Erfüllungsgrad der Anforderungen

Anforderung	Status	Kommentar
Client-Server-Architektur	Erfüllt	
TCP-basiertes Management-Protokoll	Erfüllt	
Mehrere Clients parallel	Erfüllt	Thread-pro Client in `ChatServer`.
UDP-basiertes Chat-Protokoll mit Zuverlässigkeit	Teilweise	UDP implementiert, aber ohne ACK/Timeout.
Durchgehende Verschlüsselung aller Nachrichten	Erfüllt	AES in CLI-Variante und GUI.
JavaFX-GUI mit allen Funktionen	Erfüllt	Login, Liste, Einladen, Chat-Fenster vorhanden.