Seminar Concurrent Programming

```
/*
 * Meshy
 *
 * Simple Mesh Switch in C
 * using only libc and Pthreads
 *
 * written by Daniel Ritz
 */
```

Agenda

- Aufgabenstellung
- Design
- Implementation
- Test
- Fazit
- Interessante Codestellen und Fragen

Aufgabenstellung

Grobe Anforderung

- Für alle die gleiche Aufgabe
- Verwendung von Pthreads
- Kein globaler Lock
- Plain old C

Design

Grober Aufbau

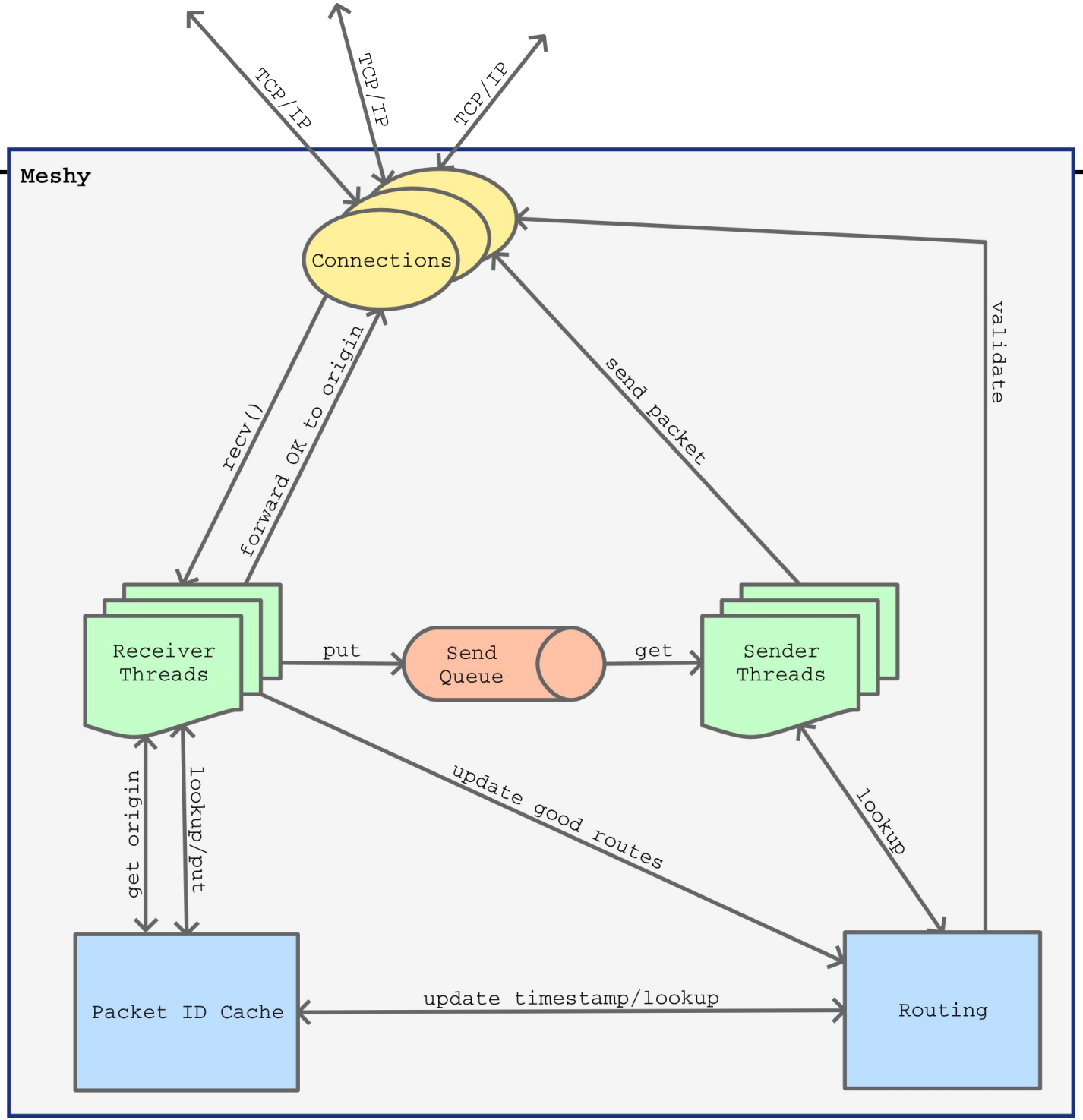
- main() wartet auf Verbindungen
- Ein Thread pro Verbindung
  - empfangen der Pakete - blockierend
  - senden der OK-Pakete
  - weiterleiten von Paketen via Queue
- Queue
  - blockierend falls leer oder voll
- x Worker Threads lesen von Queue
  - senden der Pakete als Broadcast oder Unicast

Design

Weitere Komponenten

- Verwaltung der Verbindungen
- Packet ID Cache
  - Zwischenspeichern von ID/Adresse
  - Abweisen von mehrfach erhaltenen Paketen
  - Weg zurück für OK-Pakete finden
- Routing

Design


Meshy
TCP/IP
TCP/IP
TCP/IP
Connections
recv()
forward OK to origin
send packet
validate
Receiver Threads
put
Send Queue
get
Sender Threads
get origin
lookup/put
update good routes
lookup
Packet ID Cache
update timestamp/lookup
Routing

Implementation

- Buildprozess mit automatischen Dependencies
- Pro Komponente ein .c und .h
  - .h: Definiert „public“-Schnittstelle einer Komponente
  - .c: alles was nicht zur Schnittstelle gehört ist „`static`“
- Abstraktion einer Verbindung: `connection_t`
  - Socket file descriptor
  - Status
  - Mutex
  - Referenzzähler
  - 2. Mutex zur Serialisation von `write()`

Implementation

- Queue:
  - Einfaches Array als Ring
  - Blockierendes get() wenn leer
  - Blockierendes add() wenn voll
- Packet ID Cache:
  - Kombination von *Array* als Ring und *Hashtable*
  - O(1) für Einfügen/Überschreiben
  - Schneller Lookup via Hash
  - Insert/Remove: kein malloc()/free()
  - Embedded circular doubly linked lists

Test

- Testscript

```
#! /usr/bin/env python

...

# add path to your mesh implementation to this array

implementations = [ ]

implementations += [ './../meshy/src/meshy' ]
```

- Resultat

```
$ ./test.py 200

Test passed
```

Fazit

- Erfolgreiche Umsetzung
- Gute Übung:
  - Pthreads
  - BSD Sockets
  - C
- Willkommene Abwechslung zu Java EE

Interessante Codestellen / Fragen

- Code: wer möchte was sehen?
- Sonstige Fragen