Karlstads Universitet
Institutionen för matematik och datavetenskap

Datakommunikation
DVGB02

Labboration 2: Pålitlig transport protocol

Skriven av:
Alexander Florean
florean.alexander@gmail.com
Emanuel Svensson
emansven100@student.kau.se

Handledare:
Alexander Rabitsch

9 mars 2020

Innehåll

1 Inledning

Labbens uppgift går ut på att implementera ett pålitligt transport protocol med hjälp av en förprogramerad simulation av nätverks miljön samt applikations lagret. Detta cirkulerat kring tanken att det ska fodra en bättre insikt om hur communikation fungerar mellan olika lager samt spara tid genom att inte behöva implementera alla lager.

Labben inleddes med att skapa ett tillstånds diagram av den tänkta strukturen för en sändare och ett annat tillstånds diagram för motagaren av packet, diagrammen finns inkluderade i bilagor [].

En sändare, samt en motagare av data skall programmeras. För att lösa detta inleddes labben med att skapa ett tillstånds diagram av sändaren, samt motagaren. Med hjälp av tillstånds diagramen programmerade vi sändaren och motagaren.

2 Problem

För att implementera tillstånden från grafen används globala variabler då vi inte kan förändra API:n till funkionerna till att ta emot tillståndet som parameter. Samma lösning användes för återsänding av paket. För att kunna återsända paket efter timeravbrott skapades en global pekare till det senaste sända paketet för att spara det.

// Checksum valdes att göras på det enkla sättet, där varje charactär konverterades till ascii, och sumerades, samt laddes ack och sekvensnummret på summan för att skapa checksum.

3 Design

För att skapa ett pålitligt transsportprotokoll för att överföra data används en kombination av checksum, ACK med bytande bit, och timer. Varje paket tilldelas en checksum som används för att se ifall paketet har blivit korrumperat på vägen. För att hålla koll på vilket paket man skickar eller väntar på har varje paket med data ett sekvensnummer som byter mellan 0 och 1. Efter att paketet har skickats väntar man tills man får ett ACK med samma nummer som sekvensnummret. Om ett mottagaren skickar alltid en ACK som svar till det paket den får med samma ACK som paketets sekvensnummer. Efter att en specifierad tid har gått utan att man har fått ett ACK aktiveras ett timeravbrott och paketet skickas igen.

4 Implementation

4.1 Checksum

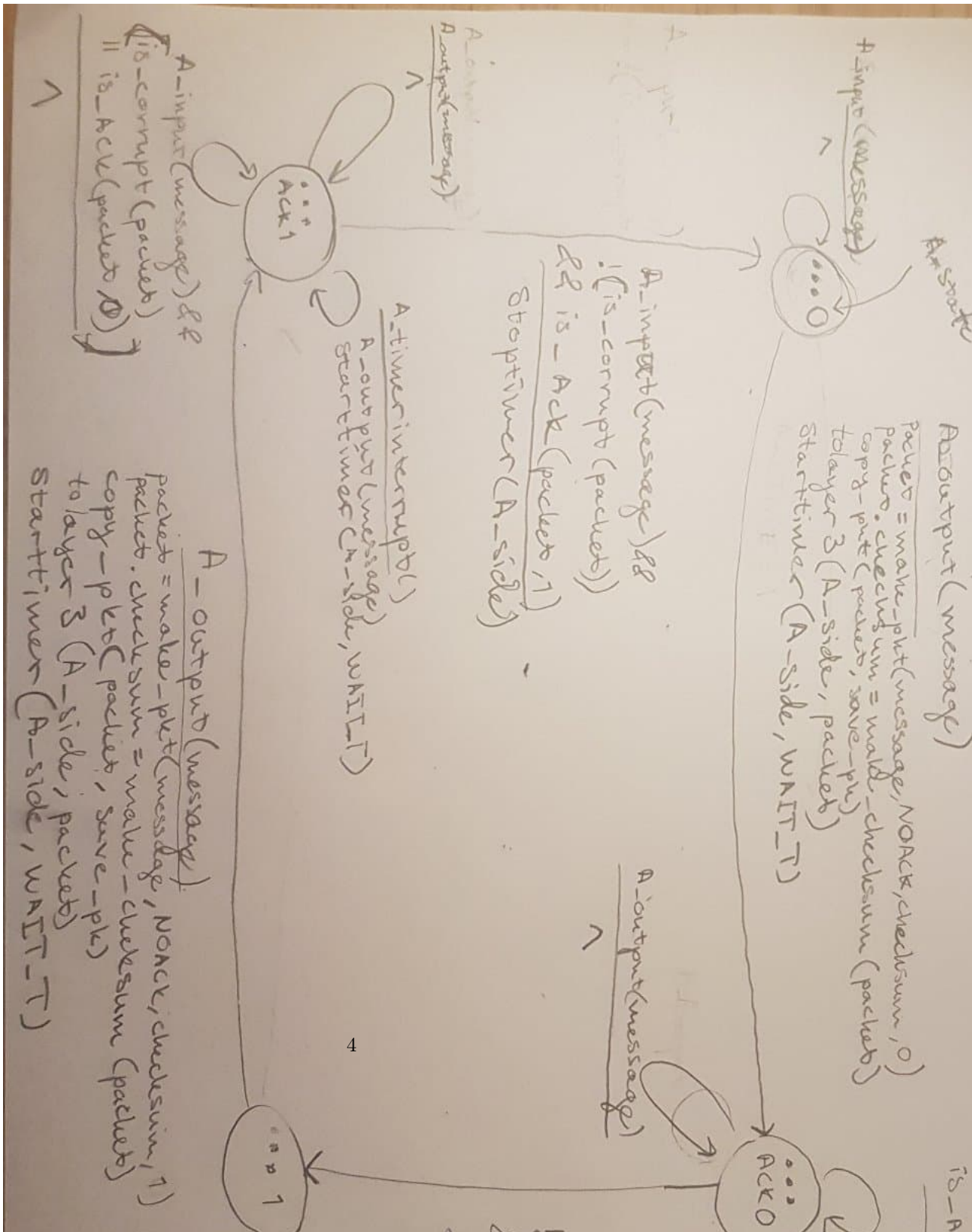
Checksum implementerades genom att skapa ett 16-bit tal som har summan av sekvensnumret, acknumret och varje char i paketets payload. För att kolla efter korruption i ett paket skapas en ny checksum av dess innehåll och jämförs med det paketet kom med. Om checksumen inte stämmer är det korrumperat.

4.2 Timer

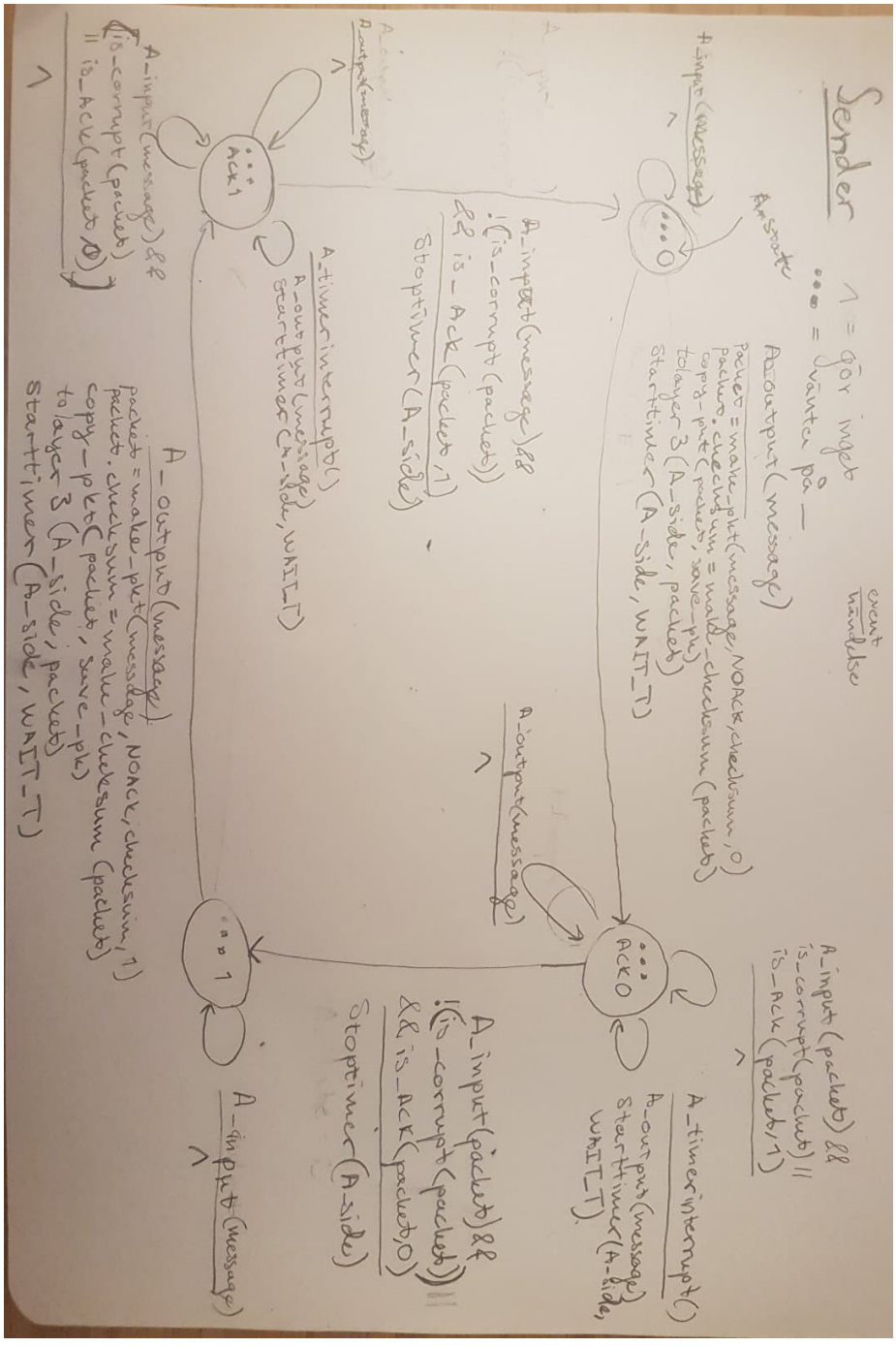
Timer var redan implementerad i simulatorn och det behövdes endast kodas in beteende vid avbrott. Macrot WAIT_T reglerar hur länge timern väntar och bestämdes genom prövning av flertal värden tills ett optimalt värde fanns.

5 Sammanfattning

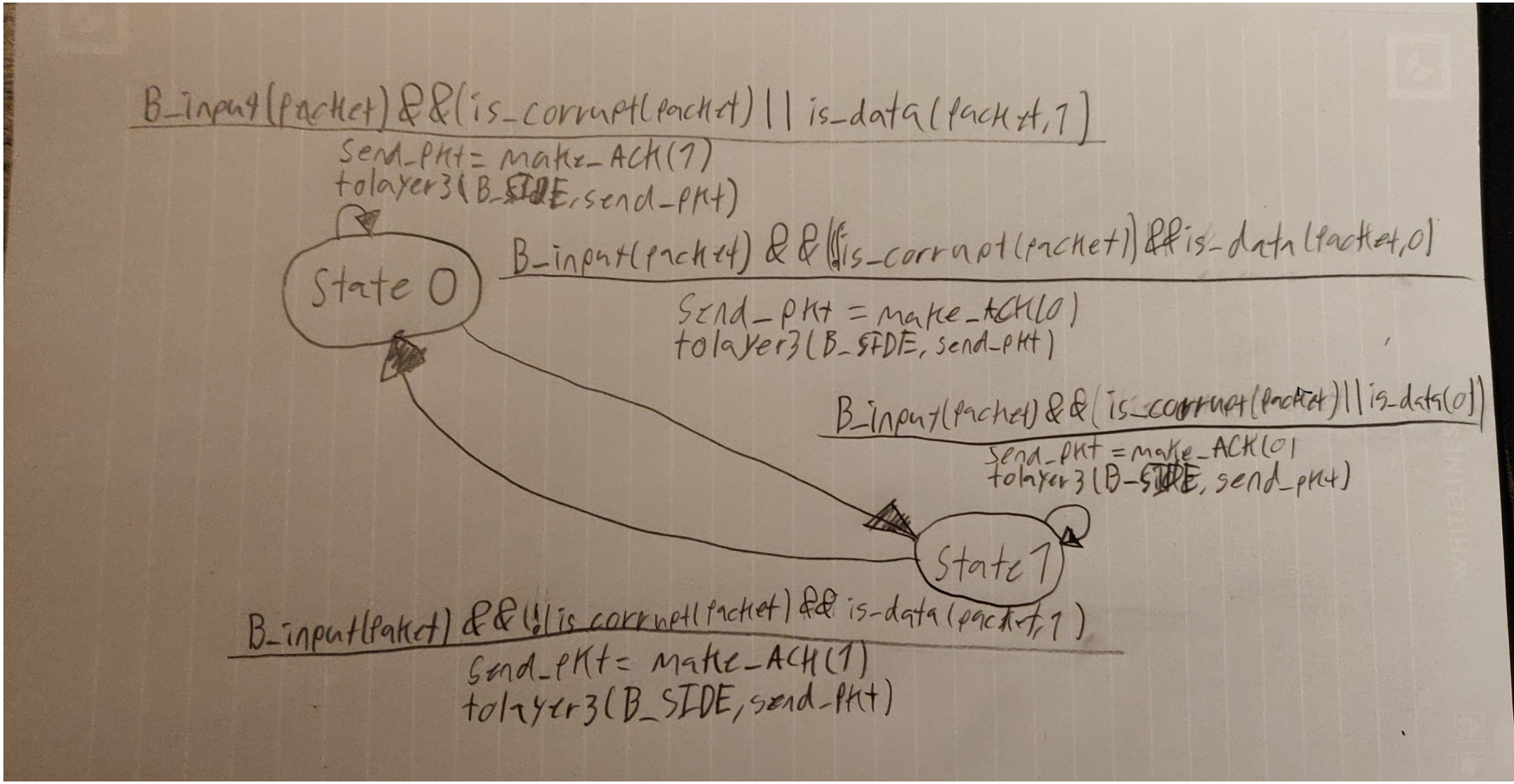
6 Bilagor



Figur 1: Sändar diagrammet



Figur 2: Sändar diagrammet


B_input(packet) && (is_corrupt(packet) || is_data(packet,1)
send_pkt = make_ACK(1)
tolayer3(B_SIDE, send_pkt)
State 0
B_input(packet) && (!is_corrupt(packet)) && is_data(packet,0)
send_pkt = make_ACK(0)
tolayer3(B_SIDE, send_pkt)
B_input(packet) && (is_corrupt(packet) || is_data(0))
send_pkt = make_ACK(0)
tolayer3(B_SIDE, send_pkt)
State 1
B_input(packet) && (!is_corrupt(packet) && is_data(packet,1))
send_pkt = make_ACK(1)
tolayer3(B_SIDE, send_pkt)

Figur 3: Mottagar diagrammet

7 Referenser

Referenser

[1] *Computer Networking: A top-down approach 7th-ed.* , James Kruose, Keith Ross, PEARSON, 2017