

Introduction to KENS: KAIST Educational Network System

TA: Hyunju Jin
NCLab, KAIST

KENS' Objective

For OS,
You can learn locking,
paging, and scheduling in
a book,

NACHOS

and you can EXPERIENCE
them by building up
NACHOS

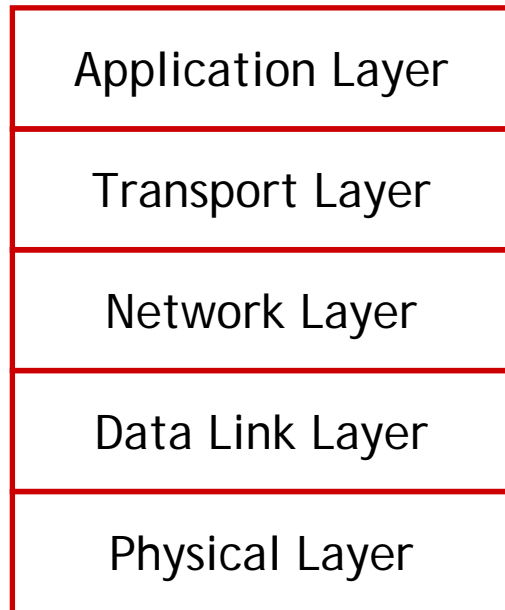
For Network,
You can learn sliding
windows and congestion
control in a book

KENS

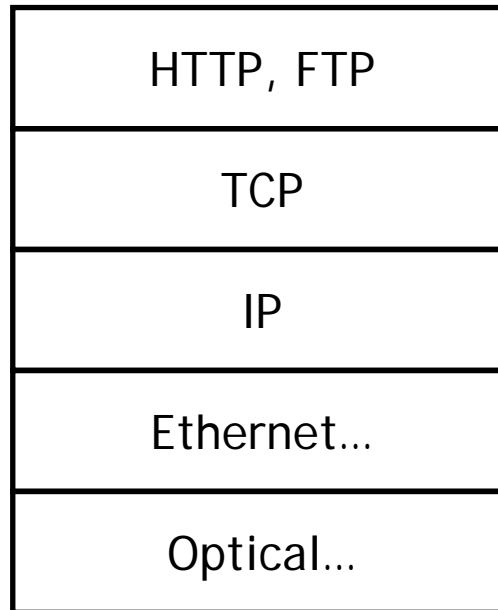
and you can EXPERIENCE
them by building up
KENS

Your KENS

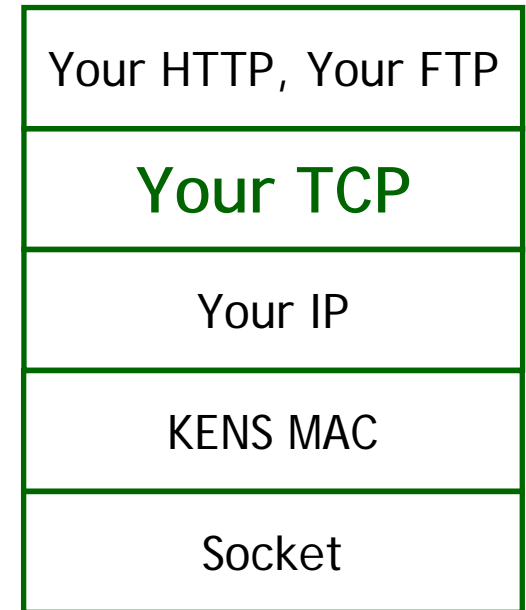
Internet Protocol Stack



In Practice



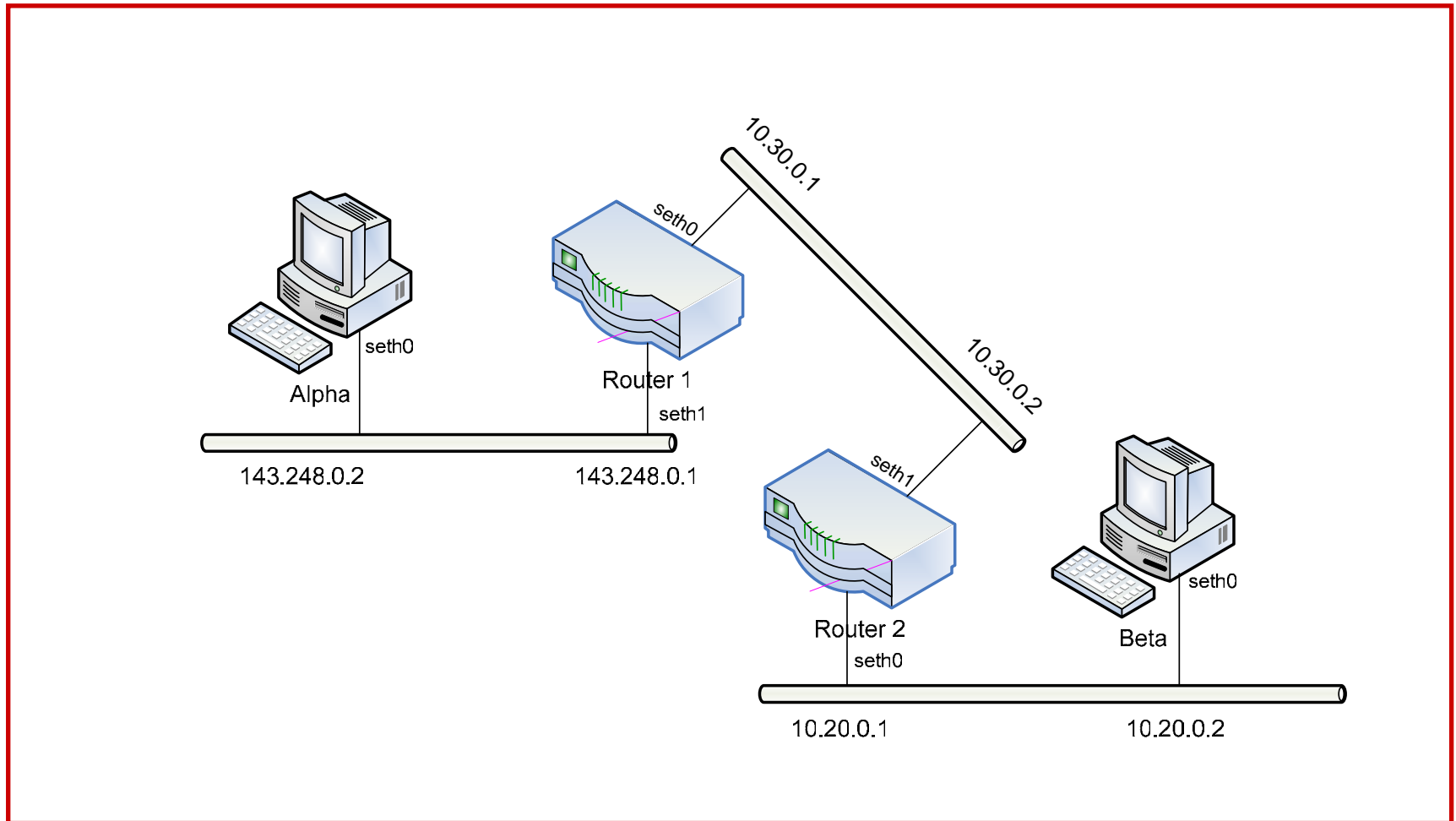
KENS



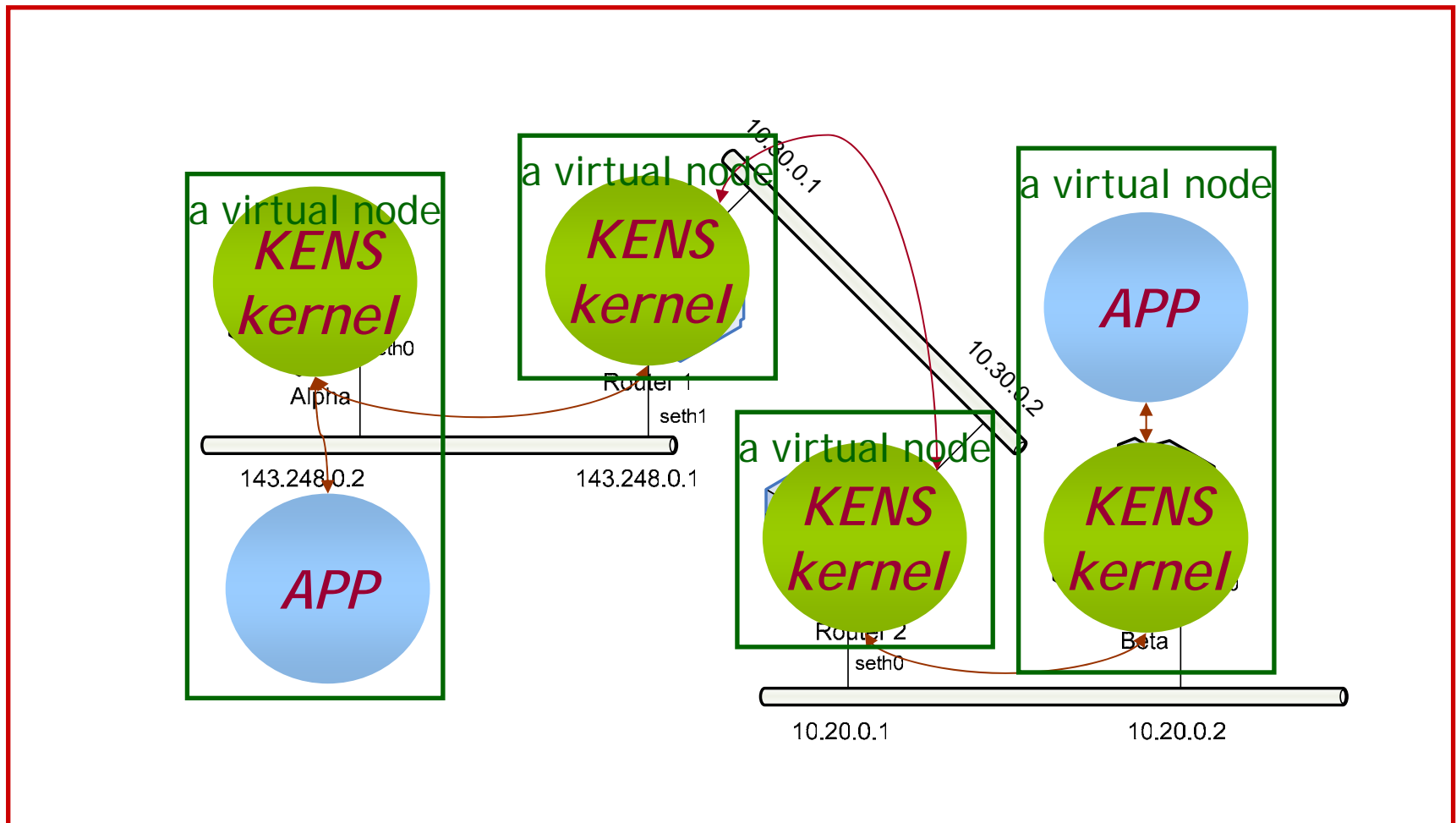
What is KENS?

- KAIST Educational Network System
- KENS is a programming environment that simulates the basic components of an operating system kernel, as well as the transport, network and data link layers, so that students can implement TCP/IP stack without modifying the kernel.

KENS Network



Process and virtual node models In KENS Network

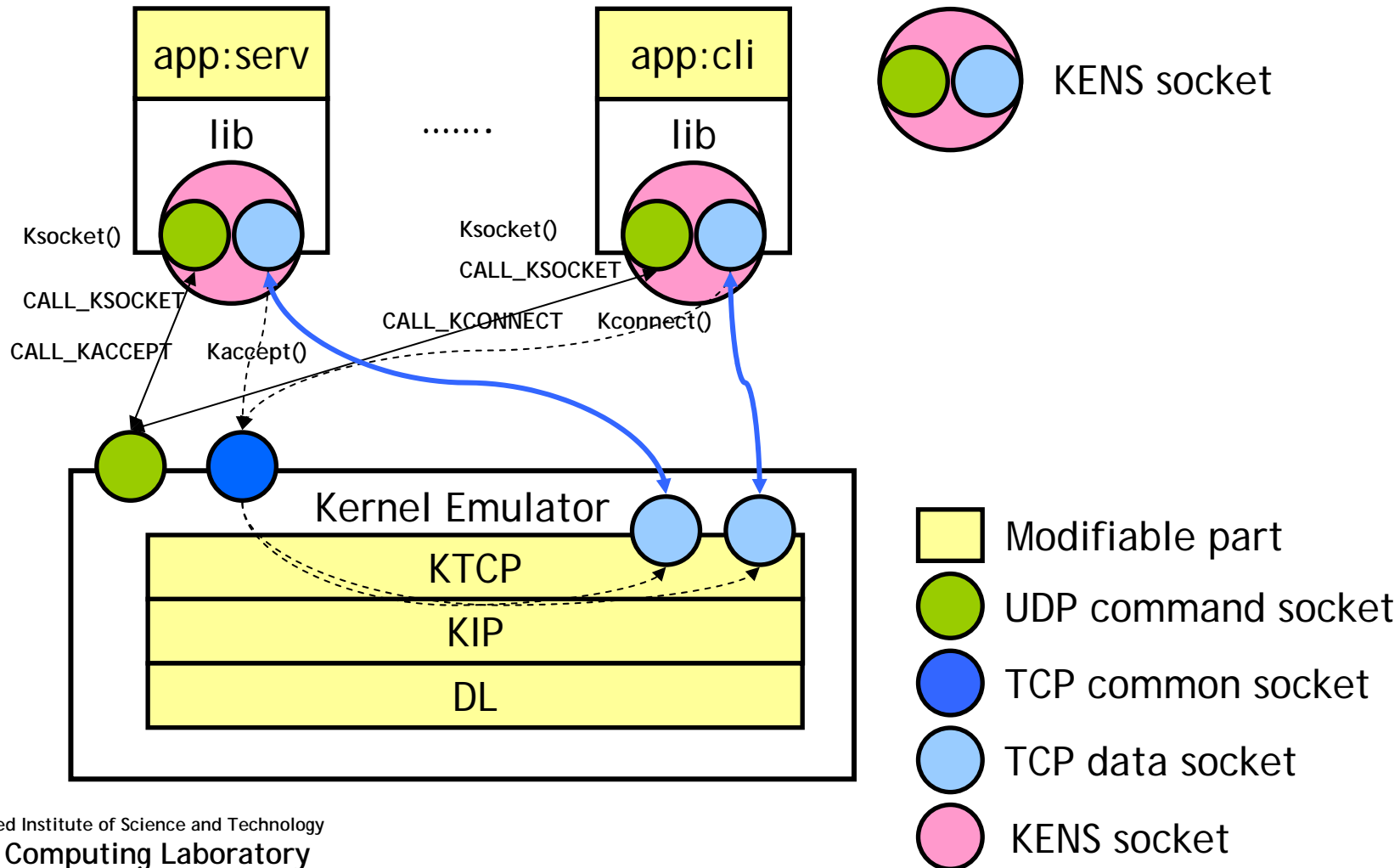


KENS Components

- Kernel Emulator
 - Emulates system call
 - Dispatch system call, data from application layer and data link layer
- KTCP
- KIP
- Data Link

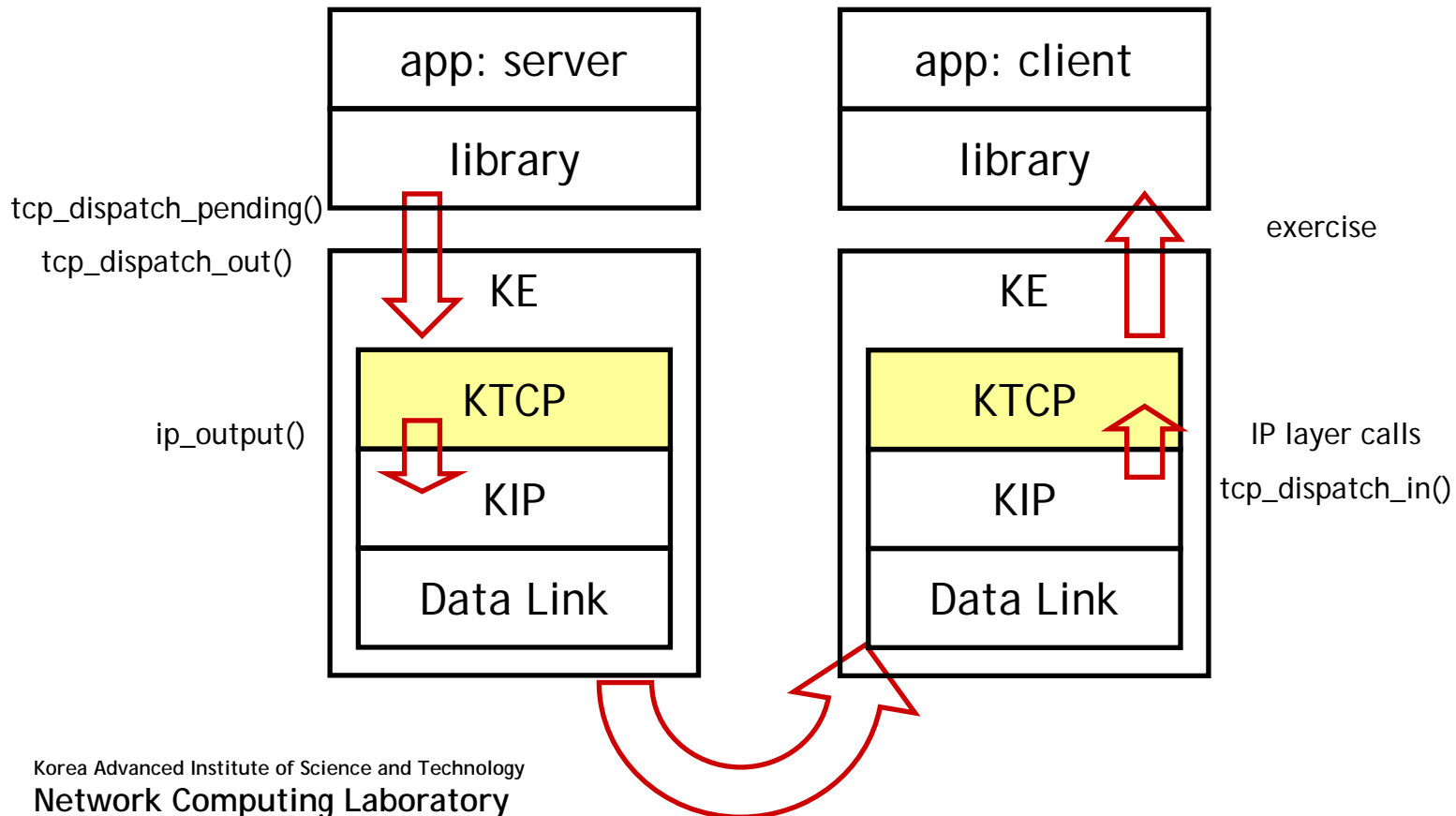
- KENS Library
 - Creates and maintains KENS socket

KENS Architecture



Implementing ktcp.c

- See also kernel_main.c, kip.c



How to start KENS

- Prepare your own Linux machine
- Download KENS code skeleton
- Extract source code
 - `tar zxvf kens.x.x.x.tar.gz`
- Configure: use default configuration
 - `./configure`
- Make
 - `./make`
- Make install
 - `./make install`

- Read KENS Install Guide

How to write KENS application

- Use KENS library instead of Berkeley socket library
- Compile with KENS library

How to run KENS

- Make configuration files
 - Do not modify configuration files in this project
 - Configuration files: test/*.conf
- Configuration & start four KENS kernel
 - [home/kens/test] ../bin/kens srv.conf &
 - [home/kens/test] ../bin/kens cli.conf &
 - [home/kens/test] ../bin/kens gw1.conf &
 - [home/kens/test] ../bin/kens gw2.conf &
- Set environment
 - [home/kens/test] source srv.sh
 - [home/kens/test] source cli.sh
 - [home/kens/test] source gw1.sh
 - [home/kens/test] source gw2.sh
- Run application
 - [home/kens/test] ./echosvr -p 8080
 - [home/kens/test] ./echocli 192.168.0.2:8080
- Read KENS User Guide

Specification

- PA#2: reliable mode
 - Network layer delivers all packet which are ordered by sequence number
 - tcp connection state
 - tcp connection setup, tear down
 - 3-way handshaking: SYN, SYN_ACK, SYN_ACK
 - 2*2-way handshaking: FIN, ACK, FIN, ACK
 - packet segmentation
 - tcp_hdr, sequence number, receive window
 - Sliding window

Specification

- In the *.conf file, add the following line
unreliable=true
- PA#3: unreliable mode
 - Packet miss, delay
 - time out, packet reordering, retransmission
- PA#4: congestion control
 - Congestion window

Evaluation

- You have to provide Demo
- Add functions to print messages which show that you KTCP works.
- We may use our own application to test

Reference

- KENS Install Guide
- KENS User Guide
- TCP/IP Illustrated, Stevens
- UNIX Network Programming, Stevens

Backup slides

Mapping KENS to the textbook

- `ip_output = udt_send(pkt)`
- `tcp_dispatch_out()` or
`tcp_dispatch_pending() = rdt_send(pkt)`
- `tcp_dispatch_in() = rdt_rcv(pkt)`