NANYANG TECHNOLOGICAL UNIVERSITY

SEMESTER 2 EXAMINATION 2013-2014

CZ3006/CSC302 – NET-CENTRIC COMPUTING

Apr/May 2014 Time Allowed: 2 hours

INSTRUCTIONS

1. This paper contains 7 questions and comprises 8 pages.
2. Answer **ALL** questions.
3. This is a closed-book examination.
4. Questions **DO NOT** carry equal marks.

---

1. (a) The signal travels 200 meters per micro second in the cable. Answer the following questions and explain your answers.

   (i) What is the minimum frame size of a CSMA/CD network running at 100 Mbps over a 1-km cable?

   (3 marks)

   (ii) Under what conditions can a CSMA/CD network run at 1Gbps and uses the same minimum frame size as calculated in Q1(a)(i)?

   (3 marks)

   (b) Briefly describe any two routing methods/algorithms and circumstances in which they can be used.

   (6 marks)

   (c) Suppose the TCP congestion window size is 40K bytes when a timeout occurs. How big will the window be if the next 10 transmission bursts are all successful? Explain your answer. Assume that the maximum segment size is 1 KB. (Hints: when a timeout occurs, the value of the slow start threshold is set to half of the current congestion window.)

   (8 marks)

2. Suppose machine A is communicating with machine B using the sliding window protocol in Assignment 1 (attached in the Appendix on pages 7 and 8). Suppose A has sent B frames 0, 1, 2, 4, and B has received frames 0 and 3. Answer the following questions:

- What sequence numbers are within the sending window of A?
- What sequence numbers are within the receiving window of B?
- Which frame(s) can B deliver to the network layer?
- Which frame(s) can B save in the buffer?
- Which sequence number can B positively acknowledge A?
- Which sequence number can B negatively acknowledge A?
- Which frame(s) will A retransmit to B?
- How will the retransmission of each lost frame be triggered?

(10 marks)

3. Frames of 1000 bytes are sent over a 1Gbps communication channel that has a 270-msec one-way delay. Suppose acknowledgements are always piggybacked onto data frames, and the headers are very short (so the overhead can be ignored). To achieve 100% channel utilization, how many bits are needed to represent the sequence numbers in a sliding-window protocol using Selective-repeat, and Go-back-n strategies, respectively? Explain your answers.

(10 marks)

4. Consider a TCP machine A sending windows of 65,535 bytes over a communication channel to another TCP machine B. How long does it take to send 64MB over this line under the following conditions?

- The round-trip time is 100-msec and the channel bandwidth is 100Mbps.
- The round-trip time is 200-msec and the channel bandwidth is 1Gbps.
- What conclusions can you draw from above results?

(10 marks)

5. (a) Describe how the HTTP cookies can be used to implement the sessions in PHP.

(2 marks)

(b) Several most commonly used methods in HTTP request include: GET, POST, HEAD, and PUT. Please explain the functionalities of above HTTP methods.

(4 marks)

6. In the wireless networking scenario shown in Figure Q6a, a wireless host sends data to the Internet. The data is first taken in an 802.11 frame to the wireless AP, and then forwarded in an Ethernet frame through the gateway router R1. There are five address fields in the two frames (address 1-5 in Figure Q6a). Please list the addresses used for the five fields. Explain how those addresses will be used in data delivery.

(8 marks)

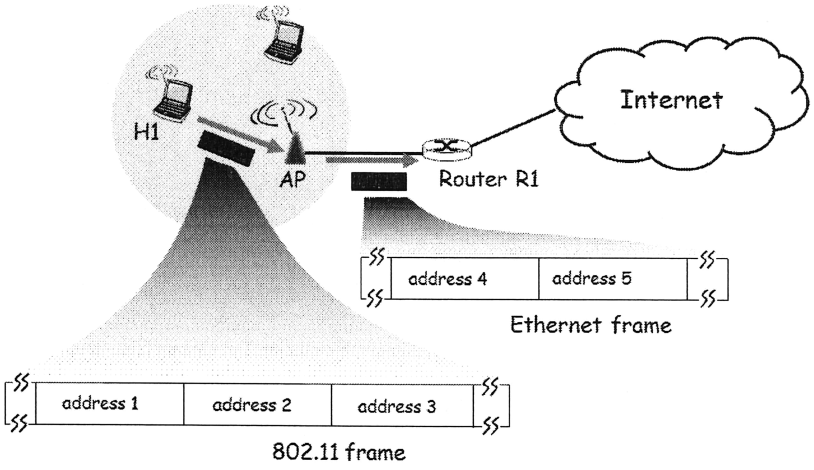

Figure Q6a

7. You are required to complete a PHP program for web registration. The registration contains four stages.

At the first stage, the client user will choose and input an ID for registration. Figure Q7a depicts the browser display at this stage. An ID registration textbox accepts the user ID input. A legitimate user ID is an **8 digit word** composed of **both** letters ('A-Z', 'a-z', and '_') and digit numbers ('0-9'). The user will be

Note: Question No. 7 continues on Page 4

asked to re-enter a valid user ID if the input does not follow the requirement. The textbox will be focused and the input will be selected. On clicking the "submit" button, the legitimate ID will be submitted to the server.

At the second stage, the server will examine the submitted user ID in a local file "record.txt" and ensure that it has **NOT** been registered by others. The file "record.txt" stores all registration information. Each entry includes a user ID, the corresponding password, and the user gender, separated by ":". Figure Q7b depicts the file organization of "record.txt". If the submitted user ID is found being used before, a form will be sent to the client asking for a different input. Figure Q7c depicts the browser display of the form. The form exchange continues until an unused ID is submitted to the server.

At the third stage, after the submitted user ID is verified unused, a new form will be sent to the client for collecting full registration data. Figure Q7d depicts the browser display of the form. In the first line, a textbox indicates the submitted user ID, which is **NOT** intended for user input. Any attempts to write into the textbox will be blurred. In the second line, a radio button group allows the user to choose the gender. "male" is checked as the default choice. In the third line, a password textbox accepts the user input for setting the password. In the fourth line, a password textbox accepts the user input for confirming the password. The two password inputs are compared and if they are not the same, the first password textbox will be focused and selected for re-input. All registration information will be submitted to the server after the "submit" button is clicked.

At the fourth stage, the server receives the current registration and appends a new entry at the end in "record.txt". The appended entry follows the record format in Figure Q7b. An "ID registration successful" notice is returned to the client.

A sole PHP program "registration.php" implements above functionalities. Complete the program "registration.php" in Figure Q7e by filling in the boxes **A1** to **A12 (2 points each)** and **B1** to **B4 (3 points each)**, each with a single statement or expression.

(36 marks)

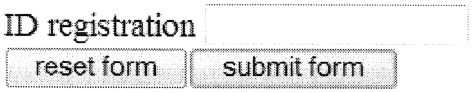

Figure Q7a

```
UserID:CZ3006EX:password:3006
UserID:CZ3002EX:password:23415
UserID:CSC30268:password:3213
UserID:123EM_22:password:11111
```

Figure Q7b

Note: Question No. 7 continues on Page 5

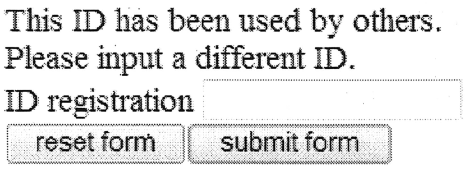

Figure Q7c

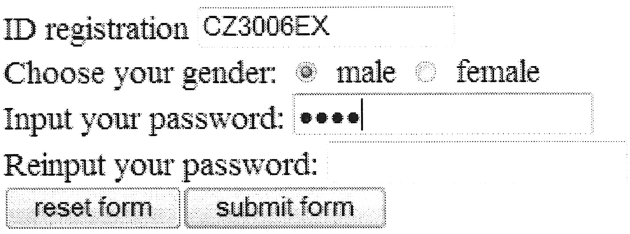

Figure Q7d

registration.php

```
<html xmlns="http://www.w3.org/1999/xhtml"><head><title>Registration</title>
<script type = "text/javascript">
<!--
      function chkID() {
            var inputID = document.getElementById("userID");
            var pos1 = inputID.value.search(      A1      );
            var pos2 = inputID.value.search(      A2      );
            if (      A3      ){
                  alert("Please re-enter a valid user ID");
                   A4
                  inputID.select();
                   return false;
                   }else
                              return true;
         }

      function chkPwd() {
            var init = document.getElementById("userPwd");
            var sec = document.getElementById("userPwd2");
            if (      A5      ) {
                  alert("The two passwords you entered are not the same. Please re-enter.");
                   init.focus();
                   A6
                   return false;
                   }else
                   return true;
      }
// --></script></head>

<body>
<form      A7      >
<?php
```

Note: Figure Q7e continues on Page 6

```
if (!isset($_POST["userID"])){
    print ("ID registration <input type = \"text\" name = \"userID\" id = \"userID\"
        A8 />");
    print ("<br/><input type = \"reset\" value = \"reset form\" />");
    print ("<input type = \"submit\" value = \"submit form\" />");
}
elseif( A9 ){
    $theID = $_POST["userID"];
    $duplicate = false;
    $file = fopen("record.txt","r");
    while(!feof($file)){
        $record = explode(":", fgets($file));
        if ($record[1] != "" && A10 ){
            $duplicate = true;
            break;
        }
    }
    fclose($file);
    if ($duplicate){
        print ("This ID has been used by others. Please input a different ID.
    <br/>");
        print ("ID registration <input type = \"text\" name = \"userID\" id =
    \"userID\"
            onchange = \"chkID();\" />");
    }else{
        print ( B1 );
        print ( B2 );
        print ( B3 );
        print ( B4 );
    }
    print ("<br/><input type = \"reset\" value = \"reset form\" />");
    print ("<input type = \"submit\" value = \"submit form\" />");
}
else{
    $output =
    "UserID:".$_POST["userID"].":password:".$_POST["userPwd"].":gender:"
    .$_POST["gender"]."\r\n";
    $file = fopen("record.txt","a");
    A11
    A12
    print ("ID registration successful.");
}
?>
</form></body></html>
```

Figure Q7e

Appendix: A Sliding-Window Protocol (for Question No. 2)

```
#define MAX_SEQ 7
#define NR_BUFS ((MAX_SEQ + 1)/2)
typedef enum {frame_arrival, cksum_err, timeout, network_layer_ready, ack_timeout}
event_type;
#include "protocol.h"
boolean no_nak = true;
seq_nr oldest_frame = MAX_SEQ + 1;

static boolean between(seq_nr a, seq_nr b, seq_nr c)
{   return ((a <= b) && (b < c)) || ((c < a) && (a <= b)) ||  ((b < c) && (c < a)); }

static void send_frame(frame_kind fk, seq_nr frame_nr, seq_nr frame_expected, packet
buffer[])
{
        frame s;  s.kind = fk;
        if (fk == data) s.info = buffer[frame_nr % NR_BUFS];
        s.seq = frame_nr;
        s.ack = (frame_expected + MAX_SEQ) % (MAX_SEQ + 1);
        if (fk == nak) no_nak = false;
        to_physical_layer(&s);
        if (fk == data) start_timer(frame_nr % NR_BUFS);
        stop_ack_timer();
}

void Sliding-Window-Protocol(void)
{
        seq_nr ack_expected, next_frame_to_send, frame_expected, too_far;
        int i;
        frame r;
        packet out_buf[NR_BUFS];
        packet in_buf[NR_BUFS];
        boolean arrived[NR_BUFS];
        event_type event;

        enable_network_layer(?);
        ack_expected = 0;
        next_frame_to_send = 0;
        frame_expected = 0;
        too_far = NR_BUFS;

        for (i = 0; i < NR_BUFS; i++) arrived[i] = false;
```

Note: Appendix continues on Page 8

```
while (true) {
  wait_for_event(&event);
  switch(event) {
    case network_layer_ready:
      from_network_layer(&out_buf[next_frame_to_send % NR_BUFS]);
      send_frame(data, next_frame_to_send, frame_expected, out_buf);
      inc(next_frame_to_send);
      break;
    case frame_arrival:
      from_physical_layer(&r);
      if (r.kind == data) {
        if ((r.seq != frame_expected) && no_nak)
          send_frame(nak, 0, frame_expected, out_buf); else
        start_ack_timer();
        if (between(frame_expected, r.seq, too_far) &&
                    (arrived[r.seq % NR_BUFS]==false)) {
          arrived[r.seq % NR_BUFS] = true;
          in_buf[r.seq % NR_BUFS] = r.info;
          while (arrived[frame_expected % NR_BUFS]) {
            to_network_layer(&in_buf[frame_expected % NR_BUFS]);
            no_nak = true;
            arrived[frame_expected % NR_BUFS] = false;
            inc(frame_expected);
            inc(too_far);
            start_ack_timer();
          } } }
      if((r.kind==nak) && between(ack_expected, (r.ack+1)%(MAX_SEQ+1),
                                  next_frame_to_send))
        send_frame(data, (r.ack+1) % (MAX_SEQ+1), frame_expected,
    out_buf);
      while (between(ack_expected, r.ack, next_frame_to_send)) {
        enable_network_layer(?);
        stop_timer(ack_expected % NR_BUFS);
        inc(ack_expected);
      }
      break;
    case cksum_err:
      if (no_nak) send_frame(nak, 0, frame_expected, out_buf);
      break;
    case timeout:
      send_frame(data, oldest_frame, frame_expected, out_buf);
      break;
    case ack_timeout:
      send_frame(ack, 0, frame_expected, out_buf);
  } } }
```

END OF PAPER

CZ3006 NET CENTRIC COMPUTING
CSC302 NET CENTRIC COMPUTING

Please read the following instructions carefully:

1. **Please do not turn over the question paper until you are told to do so. Disciplinary action may be taken against you if you do so.**

2. You are not allowed to leave the examination hall unless accompanied by an invigilator. You may raise your hand if you need to communicate with the invigilator.

3. Please write your Matriculation Number on the front of the answer book.

4. Please indicate clearly in the answer book (at the appropriate place) if you are continuing the answer to a question elsewhere in the book.