**Rob Johansen**
**u0531837**
**CS 4480 - Homework Assignment 1**

---

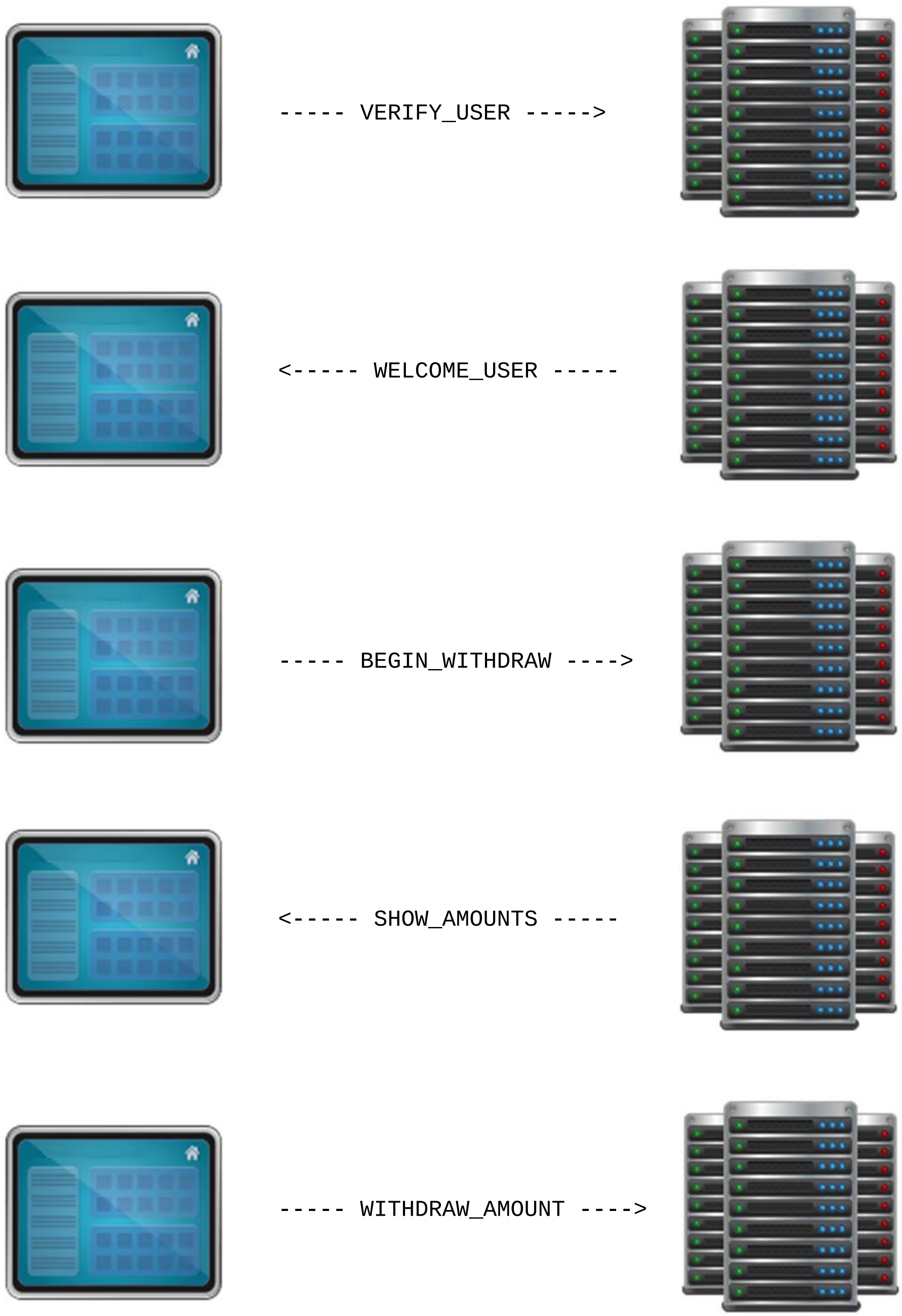
**P1**

Let the protocol be designated Automatic Teller Machine Protocol (ATMP). The following table lists the messages supported by ATMP, which direction each message travels, and the corresponding actions taken by either the ATM or the central bank computer (hereafter referred to as the server):

| Message Name | Direction | Action |
|---|---|---|
| VERIFY_USER | ATM -> server | The server looks up the user in the bank database and verifies that the PIN is correct. |
| WELCOME_USER | server -> ATM | The ATM displays a welcome message and presents its user interface because the PIN was correct. |
| INVALID_PIN | server -> ATM | The ATM informs the user that the PIN was incorrect and asks the user to try again. |
| GET_BALANCE | ATM -> server | The server looks up the user's account balance and sends it to the ATM via the SHOW_BALANCE message. |
| SHOW_BALANCE | server -> ATM | The ATM displays the account balance received from the server. |
| BEGIN_WITHDRAW | ATM -> server | The user has pressed the "Withdraw Money" button.<br><br>The server looks up the allowed withdrawal amounts (e.g. $20, $50, $100) and the user's balance. If the user's balance is larger than |

| | | |
|---|---|---|
| | | or equal to one or more of the allowed withdrawal amounts, the server sends those amounts back to the ATM via the SHOW_AMOUNTS message. If the balance is less than *all* the allowed withdrawal amounts, the server sends back the INSUFFICIENT_FUNDS message. |
| SHOW_AMOUNTS | server -> ATM | The ATM displays the allowed withdrawal amounts. |
| WITHDRAW_AMOUNT | ATM -> server | The server verifies that the requested withdrawal amount is equal to one of the allowed withdrawal amounts, and that the user has sufficient funds to make the requested withdrawal.<br><br>If the user has sufficient funds, the server debits the user's account and sends the ISSUE_MONEY message. Otherwise, the server sends the INSUFFICIENT_FUNDS message. |
| ISSUE_MONEY | server -> ATM | The ATM attempts to dispense the specified amount of money to the user. If the ATM has less money than requested by the user, the ATM sends the MONEY_FAILURE message to the server and displays an "Out of Service" message to the user. |
| INSUFFICIENT_FUNDS | server -> ATM | The ATM displays an "Insufficient Funds" message to the user. |
| MONEY_FAILURE | ATM -> server | The server credits the previously debited amount back to the user's account. |

The following diagram illustrates the ATMP traffic for a withdrawal with no errors:



<----- ISSUE_MONEY -----

ATMP makes the following assumptions about the underlying end-to-end transport service:

- It is reliable. Messages are transmitted to the correct host.
- It is encrypted. Plaintext messages cannot easily be deciphered by unauthorized parties.

## P3

a. It seems a circuit-switched network would be more appropriate for the following reasons:
   - Because the amount of time between transmissions is small, the overhead of rapidly and repeatedly exchanging packets for each N-bit data unit might be excessive. The reserved connection of a circuit-switched network would better lend itself to this application.
   - The traditional goal of a packet-switched network is to transmit as much data as possible, and to do it as quickly as possible. Because this application transmits a steady amount of data over a long period of time, a circuit-switched network is more appropriate.

b. Because the sum of *all* the data rates is less than the capacity of *each and every* link, no congestion control is necessary. All the applications combined simply do not generate enough data to overwhelm even a single link.

## P10

The total end-to-end delay can be expressed as follows:

$$total = (L/R_1 + d_1/s_1) + (L/R_2 + d_2/s_2) + (L/R_3 + d_3/s_3) + 2(d_{proc})$$

Because the transmission delay (L/R) is the same for all three links, the *total* transmission delay can be calculated as follows (L = 1,500 bytes = 12,000 bits):

$$\text{total transmission delay} = 3 * \frac{12{,}000 \text{ bits}}{2{,}097{,}152 \text{ bits / sec}} = .017 \text{ sec}$$

The propagation delay of each link can be calculated as follows, where $d_i$ is the distance of the given link in meters (*N* km * 1000):

$$\text{propagation delay}_i = \frac{d_i \text{ meters}}{250{,}000{,}000 \text{ meters / sec}}$$

Thus, the three propagation delays are:

$$\text{propagation delay}_1 = \frac{5{,}000{,}000 \text{ meters}}{250{,}000{,}000 \text{ meters / sec}} = .02 \text{ sec}$$

$$\text{propagation delay}_2 = \frac{4{,}000{,}000 \text{ meters}}{250{,}000{,}000 \text{ meters / sec}} = .016 \text{ sec}$$

$$\text{propagation delay}_3 = \frac{1{,}000{,}000 \text{ meters}}{250{,}000{,}000 \text{ meters / sec}} = .004 \text{ sec}$$

Because there are two processing delays, the total processing delay can be calculated as follows:

$$\text{total processing delay} = 2 * .003 \text{ sec} = .006 \text{ sec}$$

The total end-to-end delay can then be calculated as follows:

$$\begin{aligned}\text{total delay} = {} & \text{total transmission delay} + \\ & \text{propagation delay}_1 + \\ & \text{propagation delay}_2 + \\ & \text{propagation delay}_3 + \\ & \text{total processing delay}\end{aligned}$$

Thus, the total end-to-end delay is:

total delay = .017 + .02 + .016 + .004 + .006 = .063 sec

**P19**

(a) I performed traceroute to www.utah.edu from the SDV host (18 hops) and the WZ-Conseil host (20 hops). 11 of the links were the same in the two traceroutes, and it appears that the transatlantic link was the same (ae4.cr1.dca2.us.above.net).

(b) I performed traceroute to www.utah.edu from the SDV host (18 hops) and the HanNet host (25 hops). Only 4 of the links were the same in the two traceroutes, and it appears that the transatlantic link was different.

(c) I performed traceroute from the Stanford University server to the v-www.ihep.ac.cn host in China (the other two traceroute servers in China were not responding). After the 13$^{th}$ link, the requests repeatedly timed out.

**P31**

a. Without segmentation, the transmission delay to the first packet switch is calculated as follows:

$$\text{transmission delay} = \frac{8{,}000{,}000 \text{ bits}}{2{,}097{,}152 \text{ bits / sec}} = 3.81 \text{ sec}$$

The total time to move the message from source host to destination host is simply three times the transmission delay (because there are three links that have the same transmission rate of 2 Mbps). Thus, the total time is calculated as follows:

total time = 3 * 3.81 sec = 11.43 sec

b. For a packet of 10,000 bits, the delay to the first packet switch is calculated as follows:

$$\text{transmission delay} = \frac{10{,}000 \text{ bits}}{2{,}097{,}152 \text{ bits / sec}} = .005 \text{ sec}$$

This means the second packet will be fully received at the first switch after .01 seconds.

c. The first packet will be received at the destination host at .015 seconds (.005 seconds per link * 3 links). After that, another packet will be received at the destination host every .005 seconds. Thus, the total time for transmitting the file is calculated as follows:

   total time = .015 + 799(.005) = 4.01 seconds

   This is almost a third of the time required by part (a).

d. One benefit of message segmentation is that if a packet becomes corrupted in transit, only that one packet needs to be retransmitted. Another benefit is that multiple sources can transmit messages to multiple destinations in parallel on a first-come, first-served basis.

e. One drawback of message segmentation is the processing delay. Each packet must be processed at each router. Another drawback is that packets might arrive at the destination host out of order (i.e. the message must be reassembled from its packets).