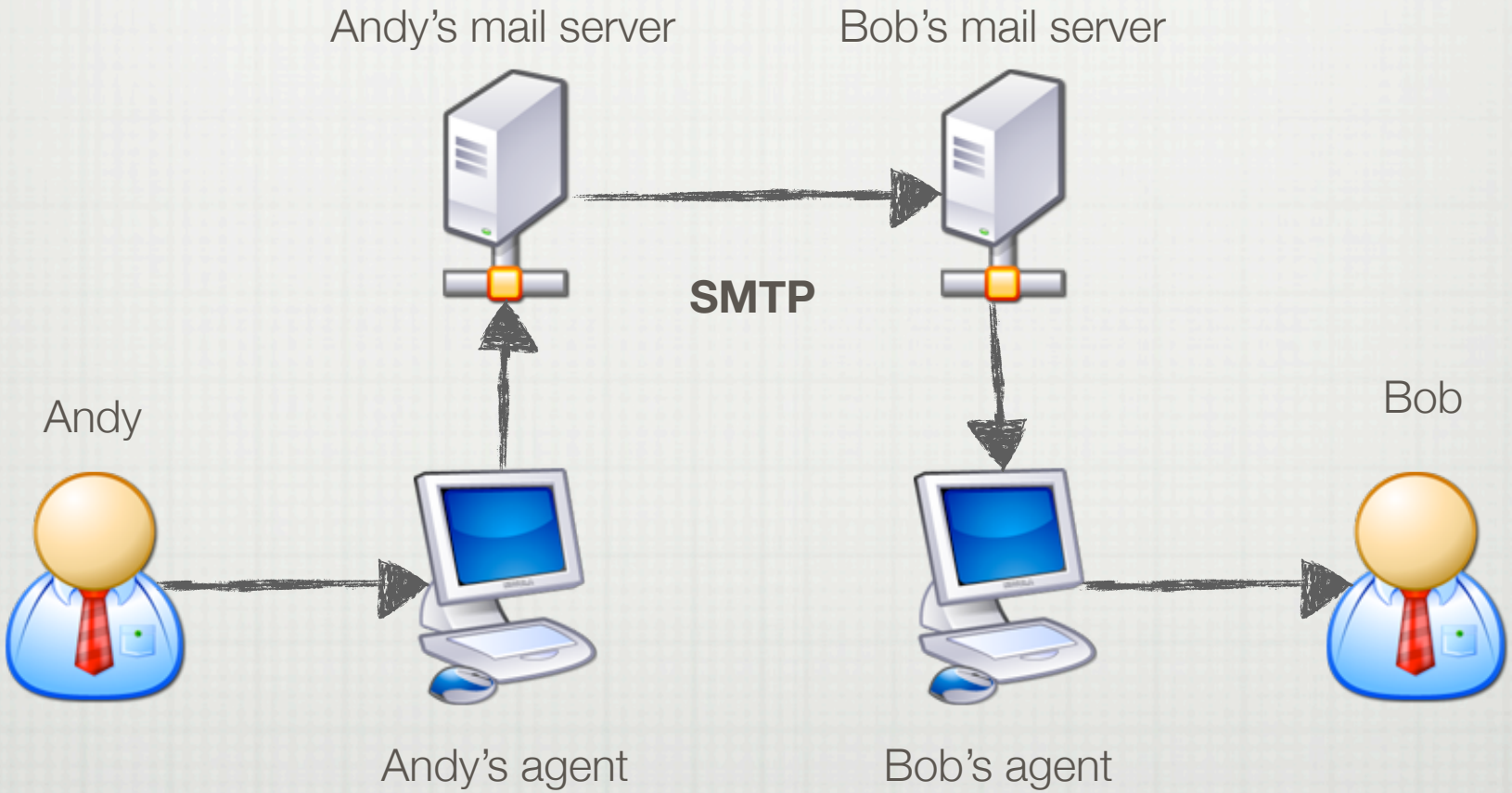
PROGRAMMING ASSIGNMENT I

TE-CHUAN CHIU

SMTP OVERVIEW

- Simple Mail Transfer Protocol
  - Application layer protocol for e-mail delivery
  - Client/Server pair
    - Client maps user requests into SMTP commands
    - Server pushes mails onto relay servers for delivery
  - RFC 821/RFC 5321

SMTP SCHEME


Andy's mail server
Bob's mail server
SMTP
Andy
Bob
Andy's agent
Bob's agent

- Try with ***telnet cnmail.csie.org 25***

SCENARIO

```
S: 220 Welcome CN 2011 Mail Server
C: HELO foo.com
S: 250 cnmail.csie.org
C: MAIL FROM:<andy@cnmail.csie.org>
S: 250 2.1.0 Ok
C: RCPT TO:<bob@foo.com>
S: 250 2.1.5 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: Subject: This is a test
C: Hello World!
C: .
S: 250 2.0.0 Ok: queued as 532043C0046
```

COMMANDS

- RFC 821 requires minimum implementation to include
  - `HELO` - Sender identification
  - `MAIL` - Assign sender
  - `RCPT` - Assign recipients
  - `DATA` - Assign mail context
  - `RSET` - Reset connection
  - `NOOP` - Require receiver responding OK
  - `QUIT` - Connection termination

SMTP ENHANCED

- SMTP requires mail header/content to contain ASCII only
- Attachments are appended to the mail body
  - MIME is used to distinguish different types of attachments
  - Attachments must be *base64-encoded*
  - Attachments are separated by a delimiter
  - Encoded strings should not exceed 76 chars per line

BASE-64 ENCODING (1/2)

- Base-64 encoding uses chars including A-Z, a-z, 0-9, +, /
- Transform every 3 8-bit chars into 4 6-bit base-64 chars

	N								T								U							
	78								84								85							
	0	1	0	0	1	1	1	0	0	1	0	1	0	1	0	0	0	1	0	1	0	1	0	1
	19				37				17				21											
	T				l				R				V											

BASE-64 ENCODING (2/2)

- If source string is not 3-divisible, padding 0s are used to lengthen the string, and the padding part is encoded as '='

	C								N																
	67								78								0								
	0	1	0	0	0	0	1	1	0	1	0	0	1	1	1	0	0	0	0	0	0	0	0	0	
	16				52				56				0												
	Q				0				4				=												

MIME HEADER (1/2)

- Encoding the mail body
  - Without attachment

```
MIME-Version: 1.0
Content-type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 7bit
```

  - With attachment

```
MIME-Version: 1.0
Content-type: multipart/mixed; boundary=”BOUNDARY”

--BOUNDARY
Content-type: text/plain; charset=US-ASCII
Content-Transfer-Encoding: 7bit
```

MIME HEADER (2/2)

- Encoding the attachments

```
--BOUNDARY
Content-type: application/x-msdownload; name=”filename”
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=”filename”

[Base-64 encoded string...]
```

- Termination

```
--BOUNDARY--
```

E-MAIL STRUCTURE

From: <sender's e-mail>

To: <receiver1's e-mail>, <receiver2's e-mail>, ...

Cc: <recipient1's e-mail>, <recipient2's e-mail>, ...

Subject: mail subject

[MIME Header]

[Mail body]

[Attachments]

.

REQUIREMENTS (1/2)

- Your program should be ...
  - GCC compilable
  - Capable of adding multiple recipients/attachments
  - Automatically generates mail/MIME headers
  - As user friendly as possible
  - Correctly handles error responses

REQUIREMENTS (2/2)

- Naming
  - b97902xxx.c
- Execution
  - ./SimpleMail *hostname port*
- Compression
  - tar -zcvf b97902xxx.tar.gz b97902xxx/

GRADING POLICIES

- Successful mail delivery (50%)
- Mail delivery with attachments (20%)
- Clarity of your C program code (comments!) (5%)
- Report (10%)
  - Flow chart, execution instruction
  - What you do, and how you do it
- Demo (15%)

REMINDERS

- Do not cheat! You cheat, you fail!
- Do not copy codes from the Internet
- Homework due
  - **2011/04/13 23:59 +0800**
- Demo
  - 2011/04/16 10:00 - 12:00 & 14:00 - 17:00

HAPPY CODING ^O^