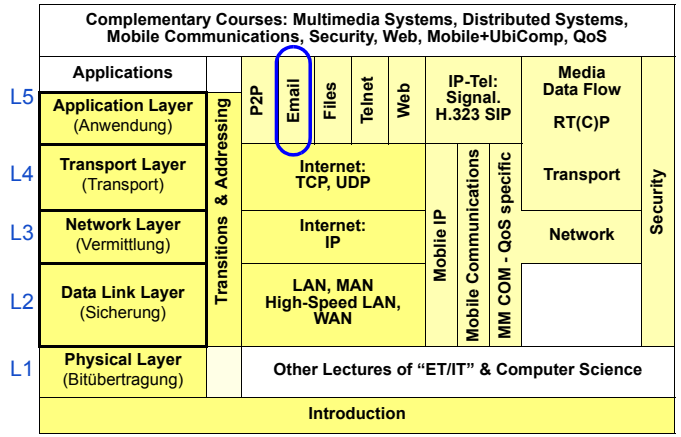
Communication Systems
Network Applications - Electronic Mail

Prof. Dr.-Ing. **Lars Wolf**

TU Braunschweig
Institut für Betriebssysteme und Rechnerverbund

Mühlenpfordtstraße 23, 38106 Braunschweig, Germany
Email: wolf@ibr.cs.tu-bs.de

Scope



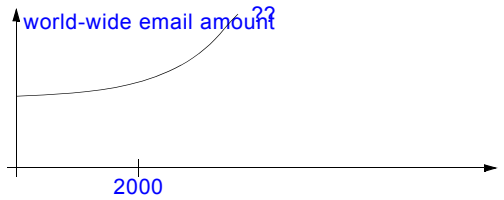
Overview

1. **Motivation and Overview**
2. **Simple Mail Transfer Protocol SMTP**
3. **X.400 Mail**
4. **Multipurpose Internet Mail Extensions (MIME)**
5. **Other concepts of Multimedia Electronic Mail**

1. Motivation and Overview

Function
- **"open memo"**
- **as in regular correspondence**

Some remarks
- **informal way to communicate**
- **cheap**
- **quantity typically approx. 5-50 emails per day**
  - in business well established
  - at home in progress
  - at many countries to be established



History

1972
- **first email sent between 2 systems**
- **Ray Tomlinson**
  - left MIT to join BBN, Boston, USA

email was THE application of the internet
- **until the web was introduced**
- **and, more recently, peer-to-peer communication is in place**

Users
- **previously:** **universities, research**
- **by now worldwide:** **companies, usually first within the engineering departments**
- **more and more:** **general population**

Email Address

Electronic mailbox
- **person/addressee is assigned to an electronic mailbox**
- **address' form is "MAILBOX@COMPUTER"**
  - unique
  - split in
    - "MAILBOX":
      Mailbox name assigned only locally
      in accordance with the respective local conventions

    @ at
    - "COMPUTER"
      for file transfer between systems
- **address today in Internet is usually "MAILBOX@DOMAINNAME"**
  - "DOMAINNAME"
    - name of the destination domain
    - "domainname" is assigned the appropriate "computer"
      by using the MX-record (MX = Mail eXchange)
      of the domain's DNS server

2. Simple Mail Transfer Protocol SMTP

SMTP (Simple Mail Transfer Protocol)
- **consists of**
  1. **message format (ASCII presentation)**
     - in 1982 defined in RFC 822
     - how the messages are structured
  2. **data transfer protocol (ASCII presentation)**
     - in 1982 defined in RFC 821
     - how the messages are transfered

2.1 SMTP - Message Format

Defined in RFC 822

Messages consist of:
- **an envelope; defined in RFC 821**
- **SMTP commands:**
  - HELO, MAIL, RCPT, DATA, QUIT,...
- **header fields (see the following table)**
- **one blank line**
- **message text**
  - originally only 7 bit, i.e. 0-127
  - (extension see also MIME)

SMTP - Message Format: Header Fields (2)

Header Field	Meaning
To:	Recipient's email address (several addresses may be given).
Cc:	Carbon Copy. Email address of second recipient (several addresses may be given).
Bcc:	Blind Carbon Copy. Email address of recipients not supposed to be visible to the other recipients (deleted before delivery).
From:	Originator of the message.
Sender:	Sender of the message.
Received:	Displays the route a message has followed until then. A new line is added for each transfer agent.
Return-Path:	May be used to list a path back to the sender.

- difference To: and Cc: solely psychologically
- difference Cc: and Bcc: bcc line will be removed from the message and is thus not visible for the recipient
- Sender: and From: if these are one & the same, then sender omitted
- Return-Path: optional

SMTP - Message Format: Other Optional Header Fields (3)

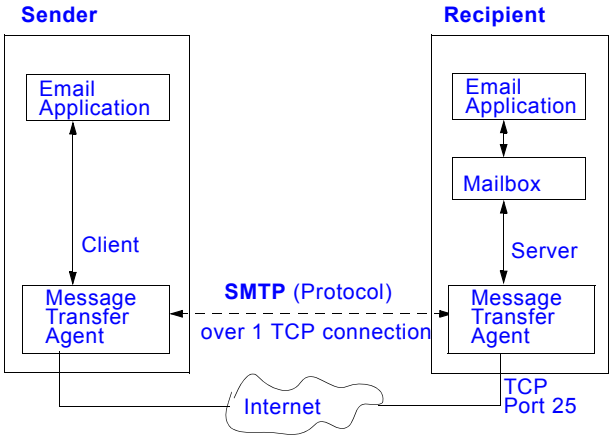
Header Field	Meaning
Date:	Day and time when message was sent.
Reply-To:	Email address to which the response is to be sent.
Message-Id:	Unique number by which the message may be identified.
In-Reply-To:	Id of the message to which this message is a reply.
References:	Other relevant message Ids.
Keywords:	User defined keywords.
Subject:	Short summary of the contents.

Based on RFC 822, additional (later defined) fields

- **may be defined**
- **these fields have to start with X**
- **examples:**
  - X-No-Archive:
  - X-Auth:
  - X-SPAM:

2.2 SMTP - Data/Mail Transmission

e.g. simple example (one hop in between)



SMTP - Data/Mail Transmission (2)

Steps

1. **sender: application**
   - generates the message in the correct format (often also the "mail user agent")
   - may store a copy of the message that was sent
2. **sender: transmission program**
   - distributes a copy of each message to each recipient
   - e.g., "sendmail" in UNIX systems
3. **receiver: email server**
   - receives message and files it in the appropriate mailbox
4. **receiver: application**
   - reads mailbox
   - converts the messages into an adequate presentation

Transfer protocol (RFC 821)

- in the internet email is transfered over a TCP connection to Port 25

Transfer Over Several MTAs

i.e., route sender to receiver
- **over several Message Transfer Agents (MTA)**

SMTP uses the store-and-forward principle to transfer messages
- **identifies the sender**
- **verifies if receiver's mailbox exists**

system name not always known, but domain is
- **address usually "mailbox@domainname"**
- **domain name server**
  - resource records:
    - information about the systems
  - among others: Mail eXchange Record (MX-Record) with
    - information about preferred system nodes for accepting mail
    - i.e., possibly different systems with different priorities

2.3 SMTP Characteristics

Characteristics
- **all transfered characters are 7 bit ASCII**
- **commands consist out of 4 letters**
- **forwarding option**
- **mailing list administration**
- **receiver confirms command with numerical value**

Example:

```
HELO mysystem.org                (establish contact)
   250 chianti.ibr.cs.tu-bs.de Hello ...
```

Problems:
- **message length limited to 64KB (in older versions)**
- **if sender and receiver have different timeouts**
  - it may result in misunderstandings
- **"mailstorms" may occur**
  - for example because mailing lists refer to each other

Improvements on some of the above mentioned SMTP problems
- **ESMTP (extended SMTP), defined initially in RFC 1425**
- **differentiation by contacting (same syntax as HELO)**

```
EHLO <systemname>
```


2.4 SMTP: Example Protocol of Direct Interaction

```
[hansa] >TELNET MAIL 25
   Trying 134.169.34.18...
   Connected to agitator.ibr.cs.tu-bs.de.
   Escape character is '^]'.
   220 agitator.ibr.cs.tu-bs.de ESMTP Sendmail 8.12.6/
   tubsibr; Tue, 28 Jan 2003 12:59:40 +0100

HELO MAIL.IBR.CS.TU-BS.DE
   HELO mail.ibr.cs.tu-bs.de
   250 agitator.ibr.cs.tu-bs.de Hello
   wolf@hansa.ibr.cs.tu-bs.de [134.169.34.81], pleased to
   meet you

MAIL FROM: <WOLF@IBR.CS.TU-BS.DE>
   250 2.1.0 <wolf@ibr.cs.tu-bs.de>... Sender ok

RCPT TO: <DIEDERICH>
   250 2.1.5 <diederich>... Recipient ok

DATA:
   500 5.5.1 Command unrecognized
```



```
DATA
   354 Enter mail, end with "." on a line by itself

TESTMAIL
THIS MAIL TESTS THE MAIL SYSTEM
.
   250 2.0.0 h0SC0jjo005641 Message accepted for delivery

QUIT
   221 2.0.0 agitator.ibr.cs.tu-bs.de closing connection

[Connection closed by foreign host.]
[hansa]~ >
```

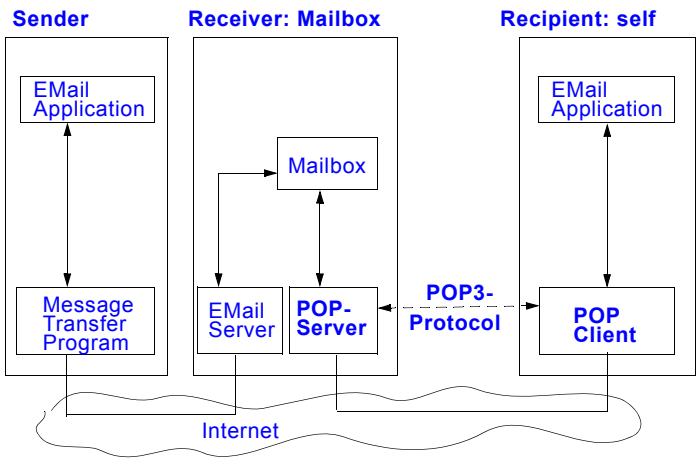
2.5 SMTP: Example Messages – Sent Message:

```
From - Fri Jan 03 12:08:51 2003
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00800000
Message-ID: <3E156F41.5070200@ibr.cs.tu-bs.de>
Date: Fri, 03 Jan 2003 12:08:49 +0100
From: Lars Wolf <wolf@ibr.cs.tu-bs.de>
Organization: IBR TUBS
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.1) Gecko/20020826
X-Accept-Language: de-de, de, en-us, en
MIME-Version: 1.0
To: Joerg Diederich <dieder@agitator.ibr.cs.tu-bs.de>
Subject: Re: Vorlesung
References: <3E0ADFD7.5020300@ibr.cs.tu-bs.de>
<200301030855.h038tWV29961@zwickel.ibr.cs.tu-bs.de>
Content-Type: text/plain; charset=ISO-8859-1; format=flowed
Content-Transfer-Encoding: 8bit

Hallo,
...
```


SMTP: Example of Received Message

```
From - Sun Jan 05 17:17:07 2003
X-UIDL: LpO!!4dn"!0=p!!]Q1!!
X-Mozilla-Status: 0001
X-Mozilla-Status2: 00000000
Return-Path: <owner-sigcomm-members@ACM.ORG>
Received: from mailhost.rz.uni-karlsruhe.de (exim@mailhost.rz.uni-karlsruhe.de
[129.13.64.98])
    by agitator.ibr.cs.tu-bs.de (8.12.6/8.12.6/Debian-6Woody) with ESMTP id h04L9xBT022208
    for <wolf@ibr.cs.tu-bs.de>; Sat, 4 Jan 2003 22:09:59 +0100
Received: from mail.listserv.dfn.de (mail.listserv.dfn.de [192.88.97.5])
    by mailhost.rz.uni-karlsruhe.de with esmtp (Exim 3.36 #1)
    id 18UvYB-0006qo-00 for lars.wolf@UNI-KARLSRUHE.DE; Sat, 04 Jan 2003 22:09:48 +0100
Received: from mail.listserv.dfn.de (192.88.97.5) by mail.listserv.dfn.de (LSMTP for
OpenVMS v1.1a) with SMTP id <11.7B2ADDFC@mail.listserv.dfn.de>; Sat, 4 Jan 2003 22:09:57
+0100
Date: Sat, 4 Jan 2003 16:09:36 -0500
Reply-To: Andreas Terzis <terzis@cs.jhu.edu>
Sender: ACM SIGCOMM organizational discussions <SIGCOMM-MEMBERS@ACM.ORG>
From: Andreas Terzis <terzis@cs.jhu.edu>
Subject: SIGCOMM MEMBERS List Digest: January 2003
To: SIGCOMM-MEMBERS@ACM.ORG
Message-Id: <E18UvYB-0006qo-00@mailhost.rz.uni-karlsruhe.de>
X-Spam-Status: No, hits=0.0 required=5.0 tests= version=2.20
X-Spam-Level:
X-UIDL: LpO!!4dn"!0=p!!]Q1!!

Dear SIGCOMM Community Members:
...
```


2.6 Post Office Protocol (POP3)



Post Office Protocol (POP3) (2)

Defined in RFC 1225, 1939, 2449

Motivation

- **user (mail recipient) uses different systems**
  - but his mailbox should always be the same
- **server has to run reliably for 24 hours**
  - but not necessarily his system
- **mailbox and applications**
  - often on different systems

Protocol for remote mailbox access:

- **user (usually) transfers mail for further processing**
  - to his local system
- **this transfer is defined in a protocol: Post Office Protocol (POP)**
- **characteristics**
  - access permitted only after authentication
  - can provide information about contents without actually transfering them
  - port
    - uses Port 110
    - SSL encrypted Port 995

Alternatively: Interactive Mail Access Protocol (IMAP)

IMAP: Interactive Mail Access Protocol
- **alternatively to POP**
- **RFC 1056**

Motivation
- **electronic letters remain on the server**
- **that means that server management is necessary**

characteristics
- **port**
  - port 143
  - SSL encrypted Port 993
- **security problem**
  - access to server data
  - possible actions: copy, delete, move

2.7 Electronic Mail: Critical Issues of Classical SMTP

With SMTP and original message format
- **sending a message to various recipients**
  - done by sending the same data to all of them individually
- **messages do not have an internal structure**

Makes automatic processing difficult
- **no acknowledgement:**
  - sender does not know whether the message he sent has actually been received by the recipient
- **message rerouting arduous?**
- **user interface not integrated in transfer system**
- **no way to send message containing a mixture of text, graphics and audio**
- **messages may contain ASCII characters only**
  - no accents or special characters ä, ö, ü, etc. (e.g. French, German)
  - no non-latin alphabets
    - e.g. Hebrew
  - no possibility to present languages without alphabets
    - e.g. Chinese, Japanese

3. X.400 Mail

History
- **defined 2 years after RFC 821 and RFC 822 (1984)**
- **idea: to correct the disadvantages of the above RFC's**

Supported by:
- **CCITT (ITU)**
- **telecommunication corporations, governments, industry**

Defacto today: X.400 not very widespread anymore
- **reasons:**
  - poor design
  - extremely complex
  - SMTP had prevailed

Pragmatic decision
- **simple but functioning system (YES) or**
- **beautiful but very complex functioning system**

4. Multipurpose Internet Mail Extensions (MIME)

Defined in RFC 1341 and RFC 1521

Possibilities:
- **messages may contain non ASCII character**
  - accents or special characters ä, ö, ü, etc. (e.g. French, German)
  - non-latin alphabets
    - e.g. Hebrew
  - languages without alphabet
    - e.g. Chinese, Japanese
- **messages that may contain audio data, video data or general data**

Idea:
- **using the format defined in RFC 822 for messages**
- **define a structure for the message text**
- **define rules for coding non-ASCII messages**

⇒ **only programs for generating and displaying messages to be modified**

⇒ **Programs for sending and receiving remain unmodified**

MIME Messages

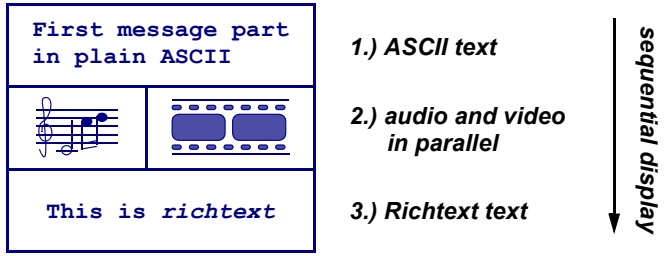
Chosen approach:
- MIME messages consist of multiple parts
- Each part may have a different type: text, audio, image, ...

Content types:
- Text (subtypes: plain, richtext)
- Image (subtypes: gif, jpeg)
- Audio (subtypes: basic)
- Video (subtypes: mpeg, h261)
- Message (subtypes: partial, external-body)
- Multipart (subtypes: mixed, alternative, parallel)
- Application (subtypes: postscript, oda)

Subtypes:
- Additional subtypes can be registered
- Designated subtypes for private usage

MIME Message: Example

Structure of an example message:



MIME message must include
- Data in multiple message parts
- Definition of content types of individual parts
- Boundaries between parts

MIME Message: Example (cont.)

```
...
Content-type: multipart/mixed;
boundary=unique-boundary-1
--unique-boundary-1
Content-type: text/plain
First message part in plain ASCII.
--unique-boundary-1
Content-type: multipart/parallel;
boundary=unique-boundary-2
--unique-boundary-2
Content-Type: audio-basic
Content-Transfer-Encoding: base64
... base64-encoded audio data goes here ...
--unique-boundary-2
Content-Type: image/jpeg
Content-Transfer-Encoding: base64
... base64-encoded image data goes here ...
--unique-boundary-2--
--unique-boundary-1
Content-Type: text/richtext
This is <italic>richtext.</italic>
--unique-boundary-1--
```

- Boundaries between message parts
- Definition of content types
- Data

4.1 MIME: Header Fields

Header Field	Meaning
MIME-Version:	Identifies the MIME version
Content-Description:	Legible description of the message
Content-Id:	Unique number to identify the message
Content-Transfer-Encoding:	Encoding type
Content-Type:	Message type

MIME version:
- is necessary to identify the message as a MIME message
- example:

```
MIME-Version: 1.0
```

Content description:
- example:

```
Content-Description: A picture of my guinea pig
```

Header Fields: Content-Transfer-Encoding

Content-Transfer-Encoding in 5 different types:

Type	Way
ASCII text	7-bit ASCII
ASCII text with 8 bit	8-bit ASCII violates protocol specification
binary	any desired 8-bit violates protocol specification
quoted-printable	ASCII presentation for short 8-bit information
base64 (ASCII armor)	ASCII presentation for 8 bit information

e.g. quoted-printable:

- **7-bit ASCII**
- **all characters > 127:**
  - presented as XXh
  - with XXh as a hexadecimal number representing the character

Header Fields: Content-Transfer-Encoding: base64 (2)

e.g. base64:

- **information viewed as a data stream**
- **64 characters are used (i.e. 2^6=64)**
  - = has special function, i.e.
    - == last group contained only 8 bits
    - = last group contained only 16 bits
- **3 bytes which need to be coded (24 Bit) are divided into four 6-bit groups**
- **line breaks are ignored**
- **example**

```
Content-type: application/msword; name="A000001.doc"
Content-Disposition: attachment; filename=A000001.doc
Content-transfer-encoding: base64

0M8R4KGxGuEAAAAAAAAAAAAAAAAAAAAAPgADAP7/
CQAGAAAAAAAAAAAAAAACAAAAhgAAAA
AAAAAAEAAAiAAAAAEAAAD+////AAAAAIQAAACFAAAA//////////////////////////
///
//////////////////////////////////////////////////////////////////////
///
//////////////////////////////////////////////////
spcEAcQAHBAAACBK/
AAAAAAAAEAAAAAAABAAAm0YAAA4AYmpianQrdCsAAAAAAAAAAAAAAAAAAAAAAAAAHBBY
AQo
0AABZBAQAWQQEANUIAAAAAAABlAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAD//
w8AAAAAAAAA
```


Header Fields: Content-Type (Examples)

Type	Subtype	Description
Text	Plain	Unformatted text
	Richtext	Text containing simple formatting commands
Image	Gif	Image in GIF format
	Jpg	Image in JPG format
Audio	Basic	Audio
Video	Mpeg	Video in MPEG format
Application	Octet-Stream	Uninterpreted byte stream
	Postscript	Printable document in Postscript format
Message	Rfc822	A MIME RFC 822 message
	Partial	This message has been split for transmission
	External-body	This message has to be retrieved from the network
Multipart	Mixed	Independent parts in the specified order
	Alternative	Same message but different formats
	Parallel	Parts have to be presented parallel
	Digest	Each part is a full RFC 822 message

4.2 MIME: Examples

Example:

```
Content-Type: text/targettext
"I am an <bold>owl </bold>", said the
<italic>walrus</italic>.
```

results in

"I am an **owl**", said the *walrus*.

MIME Example: Sent Text Message

```
From: matthias.hollick@saxophon.kom.e-technik.tu-darmstadt.de

To: diederich@ibr.cs.tu-bs.de

MIME-Version: 1.0

Message-Id: <200307011607.SAA20302@saxophon.kom.e-technik.tu-
darmstadt.de>

Content-Type: multipart/alternative; boundary= "------------
1DA8FCD5D4D"

This is a preamble, ignored by the user agent.

-----------1DA8FCD5D4D

Content-Type: text/targettext

"I am an <bold>owl</bold>", said the <italic>walrus</italic>.

The marabu nodded <italic>wisely</italic> and said:

"I am an owl, too!"
```


MIME Example: Sent Audio Message

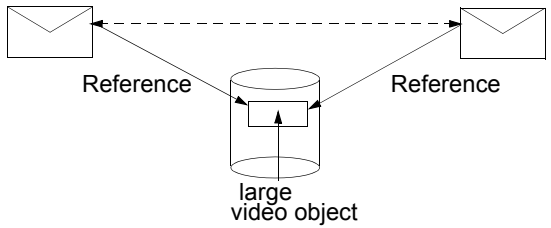
```
From: matthias.hollick@saxophon.kom.e-technik.tu-darmstadt.de

To: wolf@ibr.cs.tu-bs.de

MIME-Version: 1.0

Message-Id: <200307011607.XYAA20302@saxophon.kom.e-technik.tu-
darmstadt.de>

Content-Type: multipart/alternative; boundary= "------------
1DA8FCD5D4D"

This is the preamble, ignored by the user agent

-----------1DA8FCD5D4D

Content-Type: message/external-body;

access-type="anon-ftp";
site="ftp.kom.e-technik.tu-darmstadt.de";
directory="/pub/eulen";
name="am_owls_too.snd"

Content-Type: audio/basic
content-transfer-encoding: base64
```


5. Other concepts of Multimedia Electronic Mail

Problem:
- **many objects have a high amount of data (e.g. video)**
- **receiver has only a limited storage capacity**

Solution: global store
- **can be realized by url**
- **but:**

  contents may not necessarily be available



Secure Electronic Mail

Motivation
- **ASCII text is easy to read (by e.g. any sniffer)**
- **is the sender really the one he claims he is?**

S/MIME
- **based on strictly hierarchical certification, X.509 certificates**
- **(just like SSL)**

OpenPGP
- **Open Pretty Good Privacy**
- **based on “web of trust”**
  - user decides which certification entity he can trust