TCP/IP Standard Applications for Electronic Mail

Email, SMTP, POP, IMAP, MIME

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

What is E-Mail ?

- **E-Mail (or "email") is the most widely used Internet application**
  - Note: email was one of TCP/IP's keys to success: developers wrote RFCs and exchange them quickly via email
- **user can communicate with each other**
  - on the same machine or across a network
- **using a mailbox principle**
  - a sender does not require the receiver to be online nor the recipient to be present
  - a user's mailbox can be maintained anywhere in the Internet on a server

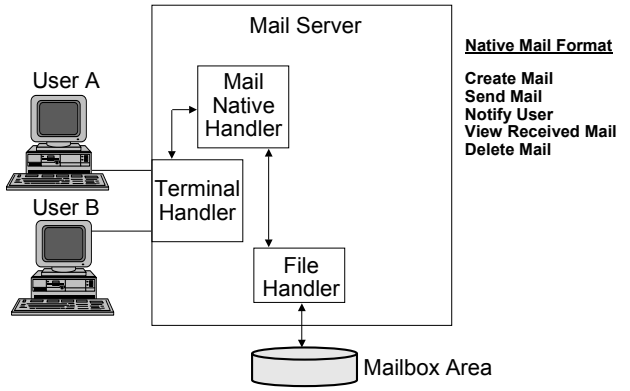
History

- **Electronic Mail has been invented**
  - in 1972 by Ray Tomlinson (note TCP in 1974)
- **initially started as a simple service that copied a file from one machine to another and appended it to the recipient's "mailbox" file**
- **problems to cope:**
  - several exchange techniques
  - several machine-dependent character sets
  - several mail content formats
  - demand for multi-media extensions
  - demand for encryption
- **1982: standardized mail format (RFC 822)**

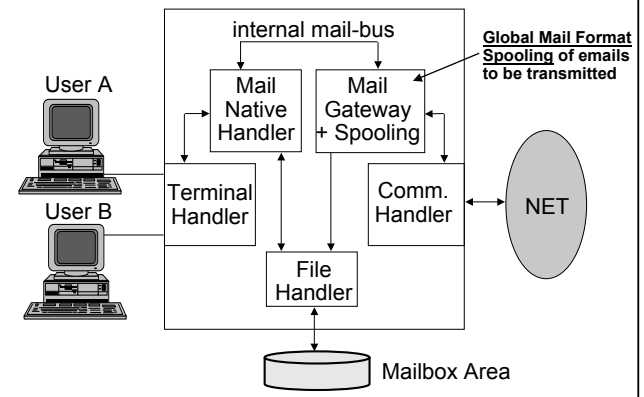
The "Electronic Mail System" (EMS)

- **there are <u>several</u> implementations of an EMS**
- **though "Internet-Mail" (using SMTP) is the most popular; standardized by IETF**
  - rather obsolete: Unix to Unix Copy (UUCP)
- **every user owns his own <u>mailbox</u> where he receives and stores messages from other users**
- **every user can be uniquely identified by an <u>email-address</u>**
- **outbound mails are intermediately stored using <u>spooling-resources</u>**
  - in case of non-standalone EMS

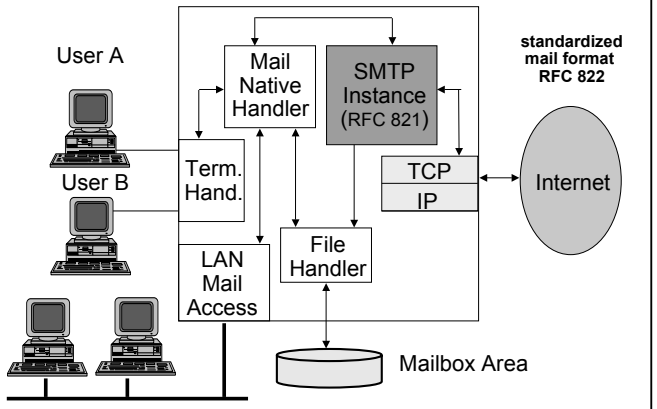
General EMS-Model Standalone



General EMS-Model Network



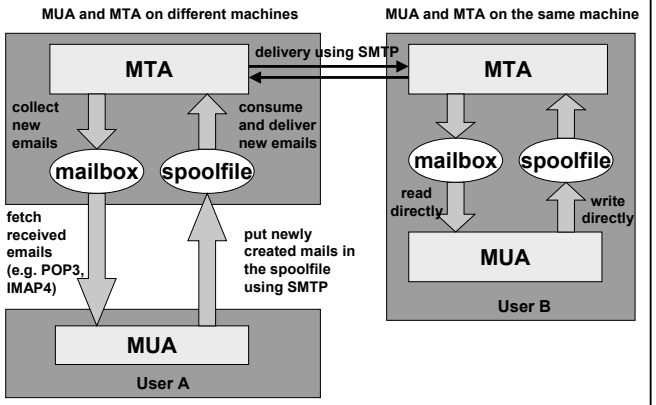
EMS-Model using SMTP



Basic Components

- **Mail User Agent (MUA)**
  - program to read and write emails
- **sender spool-file**
  - each message to be send is placed (appended) in a designated spool-file by the MUA
- **Mail Transfer Agent (MTA)**
  - program which reads emails from a spool-file in a consuming way
  - forwards these emails into the mailboxes of the recipients (e.g. using SMTP)
- **mailbox**
  - designated file owned by a receiver
  - delivered mails should be appended here

Basic Components



Basic Protocols

- **transport mechanisms to send mails from the sender's spooling memory to the receiver's mailbox:**
  - SMTP - Simple Mail Transfer Protocol (widely used)
  - X.400 (more sophisticated)
- **fetch mechanisms to move (copy) mails from a remote mailbox to a local host**
  - POP - Post Office Protocol
  - IMAP - Internet Message Access Protocol

Basic Protocols

- **multimedia attachment formats:**
  - MIME - Multipurpose Internet Mail Extensions
- **encryption standards:**
  - PGP - Pretty Good Privacy

Typical Mail Configurations

- **Local Delivery**
  - no network access - all users are directly attached on a local machine (e.g. via terminals)
- **Internet site using a "smarthost"**
  - mail is received directly using SMTP or fetched using POP or IMAP
  - outgoing mail is sent to a "smarthost" which is responsible for the proper delivery
  - smarthost optionally applies address-rewriting
  - typical for a dialup system
- **Internet site**
  - mail is sent and received directly using SMTP

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

Email Addresses

- **every mailbox can uniquely identified by an email address**
- **email addresses consists of character strings conforming the following format:**

  user@domain

  user: identifies the user or his/her mailbox of a domain

  domain: identifies some organization or a host-machine providing a mail-exchange service (DNS name)
- **example: lindner@ict.tuwien.ac.at**

Mail Routing in the Internet (Source Routing)

- **in the old days of the Internet sometimes it was necessary to specify the path a mail should take**
- **the path consists of a series of "mailbox-gateways"**
- **intermediate hops are given as a domain-list which precedes the mailbox-address**

  @domain1, @domain2, ... , @domainX:user@domain
- **today's usage of source-routing is discouraged**

Mail Routing in the Internet (DNS Based)

- **mail routing service of a mail server can be announced with the help of DNS**
  - DNS servers allow to identify a Mailbox Exchanger (MX) which is registered for a domain
  - using MX-records in the DNS database which specify the name(s) of such machine
    - each MX record is assigned a preference value (positive integer)
    - if several MX server exist for one domain, the MTA will try to transfer the message to the server with the lowest preference value
    - a MTA must not transfer mails to MX servers with a higher preference value than its own (safe way of avoiding mail loops)
  - DNS resolves for any given domain-name the machine's associated IP-address

Message Components (RFC 822, 2822) 1

- **Envelope or Header**
  - contains any information necessary for transmission and delivery
  - starts with a "From" expression in the first line
  - necessary for MUA's mail handling
    - not particular to any transport mechanism (though MTA's may use some information of the header)
  - contains well defined message information
    - about sender, receiver, intermediate stations, date and time, content-type, return-path (for error messages back to the sender), subject of the message, etc...

Message Components (RFC 822, 2822) 2

- **Body**
  - separated from the header by an empty line
  - contains the user's message
  - maximal 1000 characters
- **Signature**
  - separated from the body by two dashes "--"
  - contains personal information, jokes, PGP-keys or fingerprints, etc.
- **Very important:**
  - Header and Body must be represented with US-ASCII characters only to be RFC822 conform

Header Fields (1)

- **From:**
  - sender's email address and (frequently) her "real name"
  - many formats are used here
- **To:**
  - recipients email address
- **Subject:**
  - what the message is about (to the sender's opinion)
- **Date:**
  - the date the mail *was sent*
- **Reply-To:**
  - hint for the recipient which email address should be used for a reply

Header Fields (2)

- **Organization:**
  - hint which organization (company, etc) the user belongs to
- **Message-ID:**
  - a string, generated by the initial MTA
  - identifies a message uniquely
- **Received:**
  - every site (including sender and recipient) which processes this email inserts such a field in the header
  - several information can be stated here: site name, message-id, time, IP-address, software name
- **X-anything:**
  - used to implement additional features
  - no MUA or MTA should complain about this lines

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

Simple Mail Transfer Protocol

- **RFC 821, 2821**
- **client-server principle**
  - SMTP relies on TCP, well-known port number 25
- **end-to-end communication**
  - sender (SMTP client) talks directly to the receiver (SMTP server)
  - local deleting condition: mail must successfully arrive at the receiver
- **commands and message-contents are transferred in ASCII format**
  - printable 7-bit US-ASCII (=character values 33-126) plus CR and LF

ASCII-Code

American Standard Code for Information Interchange

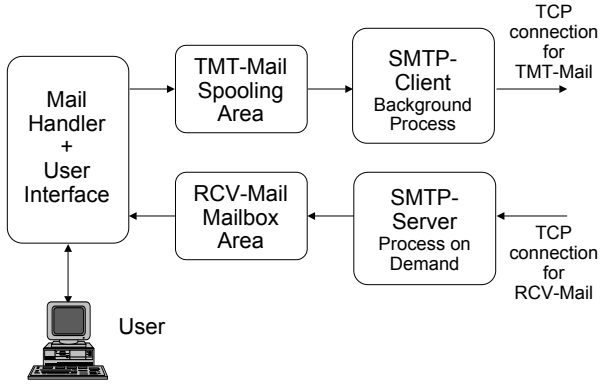
Bit Positions 7 6 5	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0 0	Null	DLE	SP	0	@	P	\	p
0 0 0 1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	STX	DC2	“	2	B	R	b	r
0 0 1 1	ETX	DC3	#	3	C	S	c	s
0 1 0 0	EOT	DC4	$	4	D	T	d	t
0 1 0 1	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	ACK	SYN	&	6	F	V	f	v
0 1 1 1	BEL	ETB	`	7	G	W	g	w
1 0 0 0	BS	CAN	(	8	H	X	h	x
1 0 0 1	HT	EM	)	9	I	Y	i	y
1 0 1 0	LF	SUB	*	:	J	Z	j	z
1 0 1 1	VT	ESC	+	;	K	[	k	{
1 1 0 0	FF	FS	,	<	L	\	l	|
1 1 0 1	CR	GS	-	=	M	]	m	}
1 1 1 0	SO	RS	.	>	N	^	n	~
1 1 1 1	SI	US	/	?	O	_	o	DEL
4 3 2 1								

Transmission Control | Format Control | Printable Character | Information Separator | Others

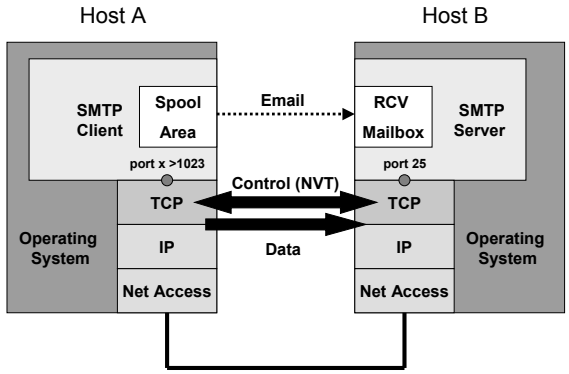
Simple Mail Transfer Protocol - SMTP

- **multiple receivers:**
  - client must establish a separate TCP connection to every receiver's server-process
- **generally, a client background process tries to empty the whole spooling area**
- **mails that cannot be delivered keep waiting in the spooling area**
  - client process will repeat its delivery attempts periodically
  - the user will be noticed about each delivery failure
  - after several repetitions the mail will be removed from the spooling area

SMTP-Model



SMTP Model



SMTP - Commands and Replies

HELO	Authentication
MAIL	Sender's Name
RCPT	Receiver's Name
DATA	Beginning transmission
SEND	Send directly
SOML	„Send or Mail"
RSET	Reset all Buffers
QUIT	Finnish session

Client → Server Command

220	Service ready
250	Request mail action ok
354	Start mail input
421	Service not available
450	Request action aborted
500	Syntax Error
550	Requested action not taken
551	User not local
554	Transaction failed

Server → Client Reply

SMTP - Commands: Client -> Server

- HELO.....for client authentication
- MAIL.....specifies sender's name (FROM-line)
- RCPT.....specifies receiver's name; can be repeated if there are several recipients on the receiver's system
- DATA.....indicates beginning of mail transmission
- SEND..... this email should be send directly to the terminal of the specified user
- SOML..... first act like SEND; if the user's terminal cannot be reached use that user's mailbox ("Send Or MaiL") *
- RSET.....resets all buffers, TCP connection remains open though
- QUIT.....finishes this client-server session

SMTP - Replies: Server -> Client

- 220 <domain> service ready
- 250 <domain> requested mail action okay, completed
- 354 start mail input, end with CR,LF,.,CR,LF
- 421 <domain> service not available, closing trans.cha.
- 450 request action aborted, local error in processing
- 500 syntax error, command unrecognized
- 550 requested action not taken (mailbox not found)
- 551 user not local
- 554 transaction failed

- **error numbers are very similar like those of FTP**
- **both commands and replies are completed with a CR, LF sequence**

SMTP Example (1)

C: (opens TCP connection to port 25 of the server)
S: 220 tuwien.edu Simple Mail Transfer Service ready
C: HELO tugraz.edu
S: 250 OK
C: MAIL FROM: josef@tugraz.edu
S: 250 OK
C: RCPT TO:hans@tuwien.edu
S: 550 no such user there
C: RCPT TO:manfred@tuwien.edu
S: 250 OK
C: DATA
S: 354 start mail input, end with CR LF . CR LF

SMTP Example (2)

C: sends message im RFC 822 Format

Date: Sun 17 April 94 09:10:22
From: Josef Maier <josef@tugraz.edu>
Subject: Greetings
To: manfred@tuwien.edu

Did this email reach you?
Josef

C: CR , LF , . , CR , LF
S: 250 OK
C: QUIT
S: 221 tuwien.edu closing transmission channel

SMTP Example (3)

```
Return-Path: josef@tugraz.edu
Posted-Date: Sun 17 April 94 09:10:22 PDT
Received-Date: Sun 17 April 94 09:11:43 PDT
Received: from tugraz.edu by tuwien.edu
        id AA07832; Sun 17 April 94 09:11:43 PDT
Date: Sun 17 April 94 09:10:22 PDT
From: Josef Maier <josef@tugraz.edu>
Subject: Greetings
To: manfred@tuwien.edu
(additionally, here may appear some Logging Information
 caused by SMTP processes having forwarded this mail)

Did this email reach you?
Josef
------
```

message conforming to the RFC 822 format, seen at the receiver

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

Post Office Protocol (POP3, RFC 1939, 2449)

- **very often a user reads and writes his emails on a local PC but has his mailbox on a server machine**
  - running a SMTP server process for receiving email
    (probably running also a SMTP client process for sending email)
  - is permanently connected with the Internet
- **POP3 lets a user fetch his emails from a remote mailbox (client-server principle)**
  - the machine with the mailbox (SMTP-server) runs also a POP3 server process
  - the POP3 client on the user's workstation is able to load and delete emails from that server and also to save them on the local disk

POP3 Principles

- **POP3 relies on TCP**
  - well-known port number 110
  - again commands and error-/state-messages are exchanged using ASCII characters
  - communication procedure is similar to SMTP
- **Some examples of "LAN Mail Access Modules and/or Native Mail Systems"**
  - Pegasus Mail (DOS/Windows)
  - Eudora
  - Groupwise (Novel, originally IPX based)
  - MS Exchange
  - MS Outlook
  - Lotus Notes

POP3 Commands 1

- USER name … user name for authentication
  - attention: cleartext
- PASS password … password for authentication
  - attention: cleartext
- STAT … to get the number of messages and total size of the messages
- LIST [msg] … if a message number is specified, the size of this mail is listed (if it exists), if not all messages will be listed with the message sizes
- RETR msg .. sends the whole message to the client
- DELE msg … deletes the specified message

POP3 Commands 2

- NOOP … the server does not do anything, just sends a positive response.
- RSET … this command cancels previous delete requests
- QUIT … if entered in the authorization state, it merely ends the TCP connection; if entered in the transaction state, it first updates the mailbox (deletes any messages requested previously) and then ends the TCP connection

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

Internet Message Access Protocol (IMAP4)

- **RFC 3501**
- **client-server principle**
- **relies on TCP, well-known port 143**
- **IMAP4 is similar to POP3 but more sophisticated**
  - allows a client to access and manipulate emails and mailboxes on a server
  - includes operations for creating, deleting, and renaming mailboxes
  - commands for selective fetching of message attributes
    - ALL
    - BODY
    - BODY<section> (get single pages of a "multipart message"),

IMAP4

- commands for selective fetching of message attributes (cont.)
  - BODYSTRUCTURE (get MIME-1 body structure of a message), ENVELOPE
  - FLAGS (get only the flags that are set for this message)
    - \Seen … Message has been read
    - \Answered … Message has been answered
    - \Flagged … Message is marked for special attention.
    - \Deleted … Message is deleted for later permanent removal.
    - \Draft … Message has been completed.
    - \Recent … Message has arrived recently and this is the first session after its arrival, this flag cannot be changed by the client.
  - FULL
  - RFC822 (get message in RFC822 format)
  - UID (get the unique identifier for this message)

IMAP4

- search-command
  - searches a mailbox for messages that match a given criteria (search keys)
- examine-command:
  - enables *read-only* mailboxes
- maintains several *flags* for each message
  - SEEN, ANSWERED, DRAFT, DELETED, FLAGGED

- **RFC 1733**
  - specifies „Distributed Electronic Mail Models in IMAP4“
    - offline use model
    - online use model
    - disconnected use model

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

SMTP and Binary Data Sources

- **RFC 822 format**
  - allows only US-ASCII characters in the message body
- **For including binary data like pictures, images, executable files in an RFC 822 conform email**
  - they first must be prepared for an ASCII-transmission
    - conversion into 7-bit-Bytes represented by printable ASCII characters
- **several ad hoc methods were used before MIME**
  - UUENCODE and UUDECODE
    - Unix-to-Unix
  - pure hexadecimal representation
  - Andrew Toolkit Representation (ATK)
  - many others

Multipurpose Internet Mail Extensions

- **MIME is a mechanism**
  - for specifying and describing the format of message bodies (content-type) in a standardized way
  - but leaves message body as ASCII text
- **using MIME now emails can contain**
  - images
  - audio-content
  - videos
  - HTML pages
  - application specific data
- **it is necessary that**
  - MUA can identify and support the associated content-type

Multipurpose Internet Mail Extensions

- **MIME is realised using**
  - MIME-Version header field
  - Content-Type header field
    - type and subtypes of data in the body
    - this describes how the object within the body is to be interpreted
    - the default value is text/plain; charset=us-ascii,
  - Content-Transfer-Encoding header field
    - this describes how the object within the body was encoded so that it could be included in he message in a mail-safe form (US-ASCII-code)
  - Content-Description header field (optional)
    - for additional plain-text data description
  - Content-ID header field
    - a world-unique identifier for the content of this part of the message

7 Standard Content-Types

- **1) text**
  - plain (unformatted text) charset=us-ASCII
    - 7 bit (position 0 - 127 in the code table)
  - plain (unformatted text) charset= iso-8859-x (x = 1 - 9)
    - us-ascii plus national characters (position 128 - 255 in the code table)
  - html and enriched
- **2) image**
  - jpeg, gif
- **3) audio**
- **4) video**
  - mpeg

7 Standard Content-Types (cont.)

- **5) application**
  - postscript
  - octet stream
- **6) multipart**
  - mixed:
    - different body parts sequentially presented to the receiver
  - parallel:
    - same as mixed but no order how to presented the different parts to the receiver
  - alternative:
    - different body parts are alternatives of the same information
    - can be presented depending on capabilities of the receiver
    - e.g. email as text/plain or text/html

7 Standard Content-Types (cont.)

- **7) message**
  - the body is an encapsulated message or part of one
  - rfc822
    - encapsulated message is RFC822 conform
  - partial
    - large mail fragmented in smaller pieces
  - external-body
    - pointer to a object existing elsewhere accessible via ftp, tftp, local file, mail-server
- **private types not falling into categories above**
  - starts with a type/subtype X-
    - e.g. X-Mailer (MS Outlook, Novell GroupWise, etc.)
    - e.g. X-Priority (Normal, High, Low)

5 Standard Content-Transfer-Encodings

- **1) 7-bit encoding**
  - body contains strict US-ASCII with maximal length of 1000 characters
- **2) 8-bit encoding**
  - possible SMTP agents support the SMTP service extension for 8-bit MIME transport
    - EHLO instead of HELO
  - still maximal length of 1000 characters
- **3) binary encoding**
  - binary with length greater than 1000 characters
  - currently only usable for type=message subtype=external-body

5 Standard Content-Transfer-Encodings

- **4) quoted-printable encoding**
  - real encoding
  - leaves text files largely readable in their encoded form
  - it represents non-mail safe characters by the hexadecimal representation of their ascii-characters
  - non-text characters are replaced by three byte sequence
- **5) Base64 encoding**
  - real encoding
  - for binary data
  - three 8-bit input words -> grouped to 24 bits
  - 24 bits -> grouped to four 6-bit words (bbbbbb)
  - each of it padded to 8-bit (00bbbbbb) word
  - 8-bit word converted with Base64-table to be mail-safe

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

X.400

- **X.400 was designed by the ITU (formerly known as CCITT) initially for telephone and X.25 networks**
  - in the 80s, many governments preferred international standards over rather randomly growing IETF protocols
  - "X.400" is short for the protocol family X.400 to X.440
- **X.400 has more features than RFC822-Mail**
  - e.g.: delivery notifications, receipt notifications, security functions, 3-level priority markers, deferred delivery, protocol conversions, reliable transfer service

X.400

- **X.400 uses a binary oriented messages structure**
  - (+) easy to embed other binary stuff (e.g. images) without encoding it for the transfer
  - (-) relies on error-less transmissions !
  - (-) debugging is complicated !
    - consider SMTP where you can simply telnet to a server and verify the operations
  - (+) but mail-faking is also complicated
  - (-) X.400 is decoded by an ASN.1-compiler; new commands violate its syntax specification and will cause error messages
    - whereas RFC822's text-based messages can be displayed in any case; even new features will not prevent revealing the basic information

X.400 features

- **store-and-forward delivery method**
  - mail is deleted locally after it was transferred to the first mail-relay-machine
    (other than SMTP's end-to-end communication)
- **address scheme uses a more general set of "attributes" which are used to look up the recipient's host in an X.500 directory server**
  - e.g.: G=Robert; S=Scott; O=southpole; OU=notfarfrom; PRMD=polarnet; ADMD=polarnet; C=ax
    (compared to IETF Mail: Robert.Scott@notfarfrom.southpole.ax)
  - actually, X.500 integration is specified but not used very often

X.400 features

- **Delivery Notification**
  - sender gets a delivery report saying that the message has been delivered to the specified address
- **Receipt Notification**
  - telling the sender, that the mail reached the receiver and she "will probably read it" -- *generated automatically*
  - receiver can issue such a receipt-notification *also manually*, telling the sender that she *did* read the message already

X.400 Features

- **priority markers (3 levels)**
  - forces "important" mails to be send earlier
  - useful if only low-bandwidth connections are available (compared to the daily mail volume)
- **conversion**
  - e.g. Teletex to plain text or embedding fax images
  - counterpart to the MIME-idea
- **reliable transfer service**
  - provides the ability to continue the transmission if it gets interrupted
  - very useful when many interrupts can be expected (and also the ideal case transmission time is relatively long)

IETF-Mail Features Missing in X.400

- **SMTP check recipients for validity before transmitting the message (receiver's existence)**
- **optionally, IETF mail can check if a message is too large before sending it**
- **ability to insert arbitrary data in the mail header**
  - using "X-....." syntax
  - promotes development of special additional features
  - additional information can be exchanged
- **with MIME Multipart/Alternative function, several representations of the same message content can be transmitted at once**
  - to ensure that any recipient is able to read it

Agenda

- **Introduction**
- **Email Address, Routing, Format**
- **SMTP**
- **POP**
- **IMAP**
- **MIME**
- **X.400**
- **RFCs**

RFCs

- **Mail:** RFC 822 (obsolete), RFC 2822
- **SMTP:** RFC 821 (obsolete), RFC 2821
- **POP2:** RFC 937
- **POP3:** RFC 1081, RFC 1225, RFC 1460, RFC 1725, RFC 1939
- **POP3 Authentication:** RFC 1734
- **APOP:** RFC 1460, RFC 1725, RFC 1939
- **RPOP:** RFC 1081, RFC 1225
- **IMAP2, IMAP2BIS:** RFC 1176, RFC 1732
- **IMAP4:** RFC 1730, RFC 1731, RFC 1732, RFC 2060, RFC 2061, RFC 3501
- **MIME:** RFC 2045, 2046, 2047, 2048, 2049