PRACTICAL ISSUES

I'll use Piazza to distribute you:

- Slides
- Links to recordings
- Notes, updates, etc.

piazza.com/uniroma1.it/fall2020/sapienza1044404

PRACTICAL ISSUES

Exam is done in two parts

- Privacy part
- Security part

- You can take them together, or one at a time.
- Your mark will be registered only when both parts are completed
- You MUST finish both parts by end of 2021

E-MAIL: A RICH INTRODUCTION

Leonardo Querzoni

querzoni@diag.uniroma1.it

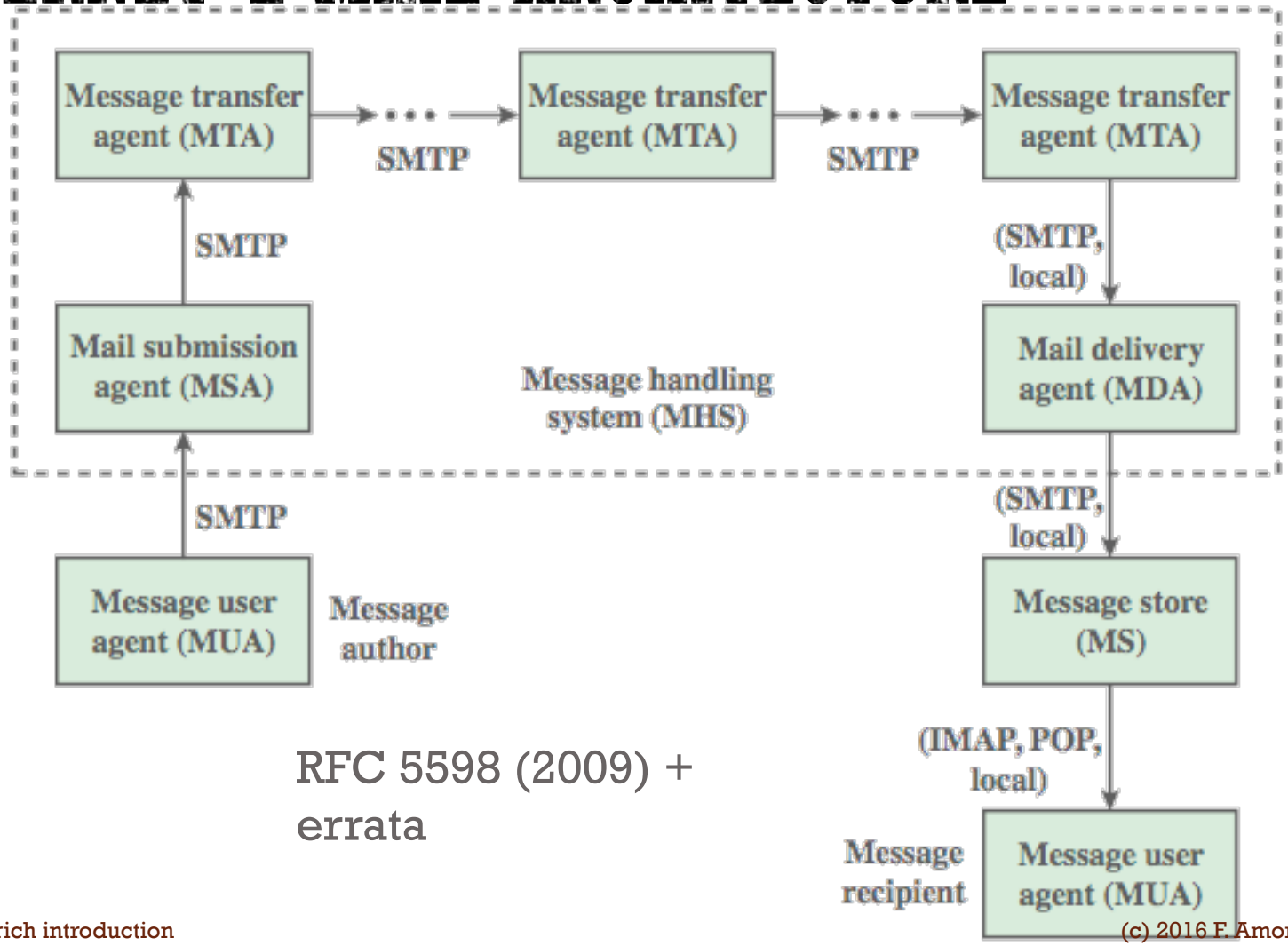
Slides by Fabrizio d'Amore – DIAG, CIS

THE INTERNET E-MAIL

The Internet e-mail system

- architecture and functioning
- extensions (MIME)
- phishing, spamming, SPF, DKIM
- intro to e-mail forensics (tracking)
- secure e-mailing: PGP

INTERNET E-MAIL ARCHITECTURE



RFC 5598 (2009) + errata

THE E-MAIL SYSTEM

- e-mail is a method of exchanging digital messages from an author to one or more recipients, operating across the Internet/ intranet
- modern e-mail systems are based on a **store-and-forward model**: e-mail servers accept, forward, store and deliver messages
- neither the users nor their computers are required to be online simultaneously

THE E-MAIL SYSTEM

- an e-mail message consists of three components
  - the **message envelope**
  - the **message header**, containing control information (originator's email address, one or more recipient addresses, a subject, a message submission date/time stamp etc.)
  - the **message body**
- originally a text-only (7-bit ASCII, or US-ASCII) communications medium
- extended to carry multi-media content attachments, a process standardized in RFC 2045 through 2049 (Multipurpose Internet Mail Extensions - MIME)

E-MAIL EXCHANGE

- e-mail transmission across IP networks is carried by the Simple Mail Transfer Protocol (SMTP, RFC 821, 1982)
  - last update: RFC 5321 (2008). Includes the extended SMTP (ESMTP) additions
- SMTP communicates delivery parameters using a message envelope separate from the message (header and body) itself
- an Internet e-mail address is a string of the form localpart@exampledomain
  - the part before the @ sign is the local part of the address
  - the part after the @ sign is a domain name (PS+1) or a fully qualified domain name
- user-level client mail applications only use SMTP for sending messages to a mail server for relaying
- to access their mail box accounts, client applications usually use either the Post Office Protocol (POP) or the Internet Message Access Protocol (IMAP) or a proprietary system (such as Microsoft Exchange or Lotus Notes/ Domino)

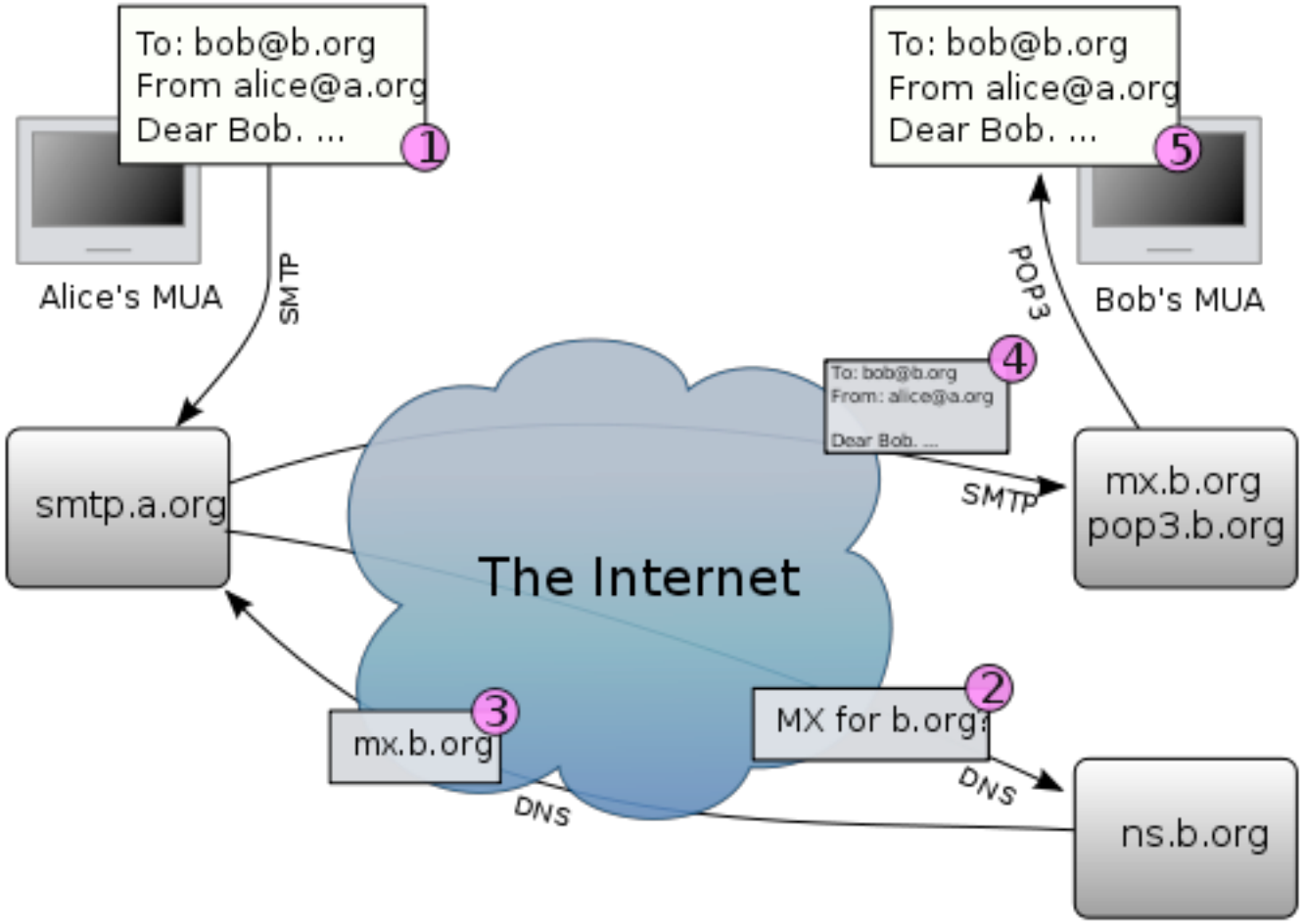
MUA, MSA, MTA

- MUA: computer program used to access and manage a user's e-mail
- MSA: computer program or software agent that receives e-mail messages from a MUA and cooperates with a mail transfer agent (MTA) for delivery of the mail
  - it uses a variant of the Simple Mail Transfer Protocol (SMTP), as specified in RFC 6409
  - its functions are a subset of those of MTA, since it deals with senders (as for input) and MTA (as for output)
  - it makes it easer for a MTA to deny relaying
- MTA: software that transfers e-mail messages from one computer to another using a client–server application architecture
  - MTAs implements both the client and server portions of the Simple Mail Transfer Protocol

MDA, MRA

- MDA: computer software component that is responsible for the delivery of e-mail messages to a local recipient's mailbox
  - within the Internet mail architecture, local message delivery is achieved through a process of handling messages from the message transfer agent, and storing mail into the recipient's environment (typically a mailbox)
- MRA: computer application that retrieves or fetches e-mail from a remote mail server and works with a mail delivery agent to deliver mail to a local or remote email mailbox
  - MRAs may be external applications or be built into a bigger application like an MUA
  - The concept of MRA is not standardized in e-mail architecture. Although they operate like mail transfer agents, MRAs are technically clients when they retrieve and submit messages

OPERATION OVERVIEW



OPERATION OVERVIEW

1. Alice composes a message using her mail user agent (MUA); she enters the e-mail address of her correspondent, and hits the "send" button
2. the MUA formats the message in email format and uses the Submission Protocol (variant of SMTP, see RFC 6409) to send the message to the local message submission agent (MSA), in this case smtp.a.org, run by Alice's ISP
3. the MSA looks at the destination address provided in the SMTP protocol, in this case bob@b.org and resolves a domain name to determine the fully qualified domain name of the mail exchange server in the Domain Name System (DNS)
4. the DNS server for the b.org domain, ns.b.org, responds with any MX records listing the mail exchange servers for that domain, in this case mx.b.org, a message transfer agent (MTA) server run by Bob's ISP

OPERATION OVERVIEW

5. smtp.a.org sends the message to mx.b.org using SMTP
   - this server may need to forward the message to other MTAs before the message reaches the final message delivery agent (MDA), which delivers it to the mailbox of Bob
6. Bob presses the "get mail" button in his MUA, which picks up the message using either the Post Office Protocol (POP3) or the Internet Message Access Protocol (IMAP4)

VARIANTS AND OTHER NOTES

A few alternative possibilities and complications can occur

- Alice or Bob may use a client connected to a corporate email system, such as IBM Lotus Notes or Microsoft Exchange and the entire transaction may be completely local
- Alice may not have a MUA on her computer but instead may connect to a webmail service
- Alice's computer may run its own MTA, so avoiding the transfer at step 1
- Bob may pick up his email in many ways, for example logging into mx.b.org and reading it directly, or by using a webmail service
- Domains usually have several mail exchange servers so that they can continue to accept mail when the main mail exchange server is not available
- Email messages are not secure if email encryption is not used correctly
- Number of open e-mail relays is decreasing (to prevent spam)

MESSAGE FORMAT

- Message format is defined by RFC 5322
  - support to MIME (RFC 2045 through RFC 2049), collectively called Multipurpose Internet Mail Extensions
- Internet e-mail messages consist of two major sections:
  - **Header** — Structured into fields such as From, To, CC, Subject, Date, and other information about the email.
  - **Body** — The basic content, as unstructured text; sometimes containing a signature block at the end. This is exactly the same as the body of a regular letter.
- The header is separated from the body by a blank line

HEADER

- Each message has exactly one header, which is structured into fields. Each field has a name and a value. RFC 5322 specifies the precise syntax
- Informally, each line of text in the header that begins with a printable character begins a separate field and its name starts in the first character of the line and ends before the separator character ":"
- The separator is then followed by the field value (the "body" of the field). The value is continued onto subsequent lines if those lines have a space or tab as their first character. Field names and values are restricted to 7-bit ASCII characters. Non-ASCII values may be represented using MIME encoded words
- Email header fields can be multi-line, and each line must be at most 76 characters long. Header fields can only contain US-ASCII characters; for encoding characters in other sets, a syntax specified in RFC 2047 can be used

MANDATORY HEADER FIELDS

- **From:**
  The email address, and optionally the name of the author(s)

- **Date:**
  The local time and date when the message was written

OTHER (SUGGESTED) HEADER FIELDS

- **Message-ID**

  Also an automatically generated field; used to prevent multiple delivery and for reference in In-Reply-To:

- **In-Reply-To**

  Message-ID of the message that this is a reply to. Used to link related messages together. This field only applies for reply messages

- RFC 3864 describes registration procedures for message header fields at the IANA; it provides for permanent and provisional message header field names

Common Header Fields

To:

The email address(es), and optionally name(s) of the message's recipient(s). Indicates primary recipients (multiple allowed)

Subject:

A brief summary of the topic of the message. Certain abbreviations are commonly used in the subject, including "RE:" and "FW:"

Bcc:

Blind Carbon Copy; addresses added to the SMTP delivery list but not (usually) listed in the message data, remaining invisible to other recipients.

Cc:

Carbon copy; Many email clients will mark email in your inbox differently depending on whether you are in the To: or Cc: list.

Content-Type:

Information about how the message is to be displayed, usually a MIME type.

COMMON HEADER FIELDS

- **Precedence:**
  commonly with values "bulk", "junk", or "list"; used to indicate that automated "vacation" or "out of office" responses should not be returned for this mail
  - With modern high-bandwidth networks delivery priority is less of an issue than it once was.
- **References:**
  Message-ID of the message that this is a reply to, and the message-id of the message the previous reply was a reply to, etc.
- **Reply-To:**
  Address that should be used to reply to the message
- **Sender:**
  Address of the actual sender acting on behalf of the author listed in the From: field (secretary, list manager, etc.)
- **Archived-At:**
  A direct link to the archived form of an individual email message

- Note that the To: field is not necessarily related to the addresses to which the message is delivered. The actual delivery list is supplied separately to SMTP, which may or may not originally have been extracted from the header content. In the same way, the "From:" field does not have to be the real sender of the email message.

TRACE INFORMATION OF A MESSAGE

SMTP defines the trace information of a message, which is also saved in the header using the following two fields:

- **Received:**
  when an SMTP server accepts a message it inserts this trace record at the top of the header (last to first)
- **Return-Path:**
  when the delivery SMTP server makes the final delivery of a message, it inserts this field at the top of the header

Other header fields that are added on top of the header by the receiving server may be called trace fields, in a broader sense

- **Authentication-Results:**
  when a server carries out authentication checks, it can save the results in this field for consumption by downstream agents
- **Received-SPF:**
  stores the results of SPF checks
- **Auto-Submitted:**
  is used to mark automatically generated messages
- **VBR-Info:**
  claims VBR whitelisting

SAMPLE SMTP INTERACTION

as of RCF 821

Party	SMTP commands and status codes	Explanation
Server:	220 smtp.example.com ESMTP Postfix	After the connection has been established, the SMTP server answers
Client:	HELO relay.example.com	The SMTP client logs on with its hostname
Server:	250 smtp.example.com, hello	The server confirms the login
Client:	MAIL FROM:<john@doe.com>	The client specifies the sender address of the MUA
Server:	250 OK	The server confirms
Client:	RCPT TO:<boss@workplace.com>	The client specifies the recipient address
Server:	250 OK	The server confirms
Client:	DATA	The client initiates the transmission of the e-mail
Server:	354 End data with <CR><LF>.<CR><LF>	The server begins the reception and indicates that the e-mail text should be closed with a dot (".")
Client:	From: "John Doe" <john@doe.com> To: Boss Workplace <boss@workplace.com>	The client transmits the e-mail text, highlights it with a line

SAMPLE SMTP INTERACTION

as of RCF 821

Client:	DATA	The client initiates the transmission of the e-mail
Server:	354 End data with <CR><LF>.<CR><LF>	The server begins the reception and indicates that the e-mail text should be closed with a dot (".")
Client:	From: "John Doe" <john@doe.com> To: Boss Workplace <boss@workplace.com> Date: Monday, March 12 2018 10:03:42 Subject: Sick note Hello boss, Unfortunately, I am sick today and cannot come into work. Thank you for your understanding, John Doe	The client transmits the e-mail text, highlights it with a line break after "Subject: Sick note" and ends it with the desired dot
Server	250 OK: queued as 15432	The server confirms it has successfully received the e-mail and puts it in a queue
Client:	QUIT	The client signals the end of the session
Server:	221 Goodbye	The server terminates the connection

OTHER SMTP COMMANDS

as of RCF 821

- **RSET** current mail transaction is to be aborted
- **SEND**, **SAML**, **SOML** announce message to terminal and/or mailbox
- **VRFY** asks to confirm that the argument identifies a user
- **EXPN** asks to confirm that the argument identifies a mailing list (to be expanded)
- **HELP** for general or specific help
- **NOOP** no operation
- **TURN** reverses the SMTP communication, by exchanging the role of the parties

some of the above commands may be not implemented

ESMTP

- RFC 821 (1982) was obsoleted by RFC 5321 (2008), where ESMTP, an extended version of SMPT, was introduced
- software agents should stick to ESMTP but for backward compatibility reason a client connecting by SMTP will be also served
- the greeting command for ESMTP is **EHLO**, which gets a (possibly multiline) response listing the supported extended commands

S: 220 foo.com Simple Mail Transfer Service Ready
C: EHLO bar.com
S: 250-foo.com greets bar.com
S: 250-8BITMIME
S: 250-SIZE
S: 250-DSN
S: 250-VRFY
S: 250 HELP

OTHER ESMTP COMMANDS

from Wikipedia

- 8BITMIME — 8 bit data transmission, RFC 6152
- ATRN — Authenticated TURN for On-Demand Mail Relay, RFC 2645
- AUTH — Authenticated SMTP, RFC 4954
- CHUNKING — Chunking, RFC 3030
- DSN — Delivery status notification, RFC 3461 (See Variable envelope return path)
- ETRN — Extended version of remote message queue starting command TURN, RFC 1985
- HELP — Supply helpful information, RFC 821
- PIPELINING — Command pipelining, RFC 2920
- SIZE — Message size declaration, RFC 1870
- STARTTLS — Transport layer security, RFC 3207 (2002)
- SMTPUTF8 — Allow UTF-8 encoding in mailbox names and header fields, RFC 6531
- UTF8SMTP — Allow UTF-8 encoding in mailbox names and header fields, RFC 5336 (deprecated)

SOFTWARE FOR ESMTP

- according to a survey (from Security Space), sendmail, Microsoft Exchange Server, Postfix, and Exim together control over 90% of market share for SMTP service in 2014
  - http://www.securityspace.com/s_survey/data/man.201403/mxsurvey.html

MIME (MULTIPURPOSE INTERNET MAIL EXTENSIONS)

- Internet standard that extends the format of email to support:
  - Text in character sets other than ASCII
  - Non-text attachments
  - Message bodies with multiple parts
  - Header information in non-ASCII character sets
- MIME's use has grown beyond describing the content of email to describe content type in general (web, storage)
- Virtually all human-written Internet email and a fairly large proportion of automated email is transmitted via SMTP in MIME format
- MIME is specified in six linked RFC memoranda

MIME

Important RFCs

- RFC-822 Standard for the format for ARPA Internet text messages
- RFC-2045 MIME Part 1: Format of Internet Message Bodies
- RFC-2046 MIME Part 2: Media Types
- RFC-2047 MIME Part 3: Message Header Extensions
- RFC-2048 MIME Part 4: Registration Procedure
- RFC-2049 MIME Part 5: Conformance Criteria

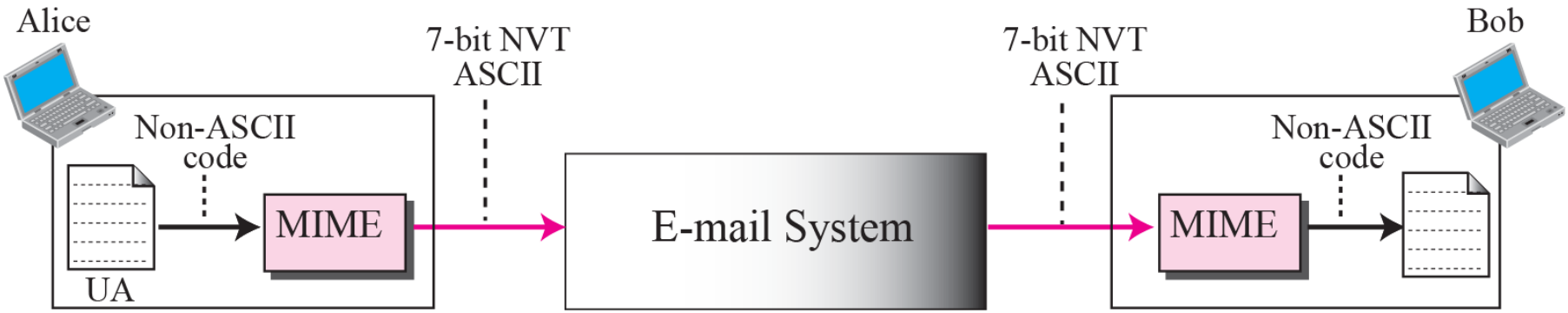
MIME – WHAT IS IT?

- MIME refers to an official Internet standard that specifies how messages must be formatted so that they can be exchanged between different email systems.
- MIME permits the inclusion of virtually any type of file or document in an email message.
- Specifically, MIME messages can contain
  - text
  - images
  - audio
  - video
  - application-specific data
    - spreadsheets
    - word processing documents

MIME FEATURES

- Support of character sets other than ASCII
- Content type labeling system
- Support of non-text content in e-mail messages
- Support for compound documents

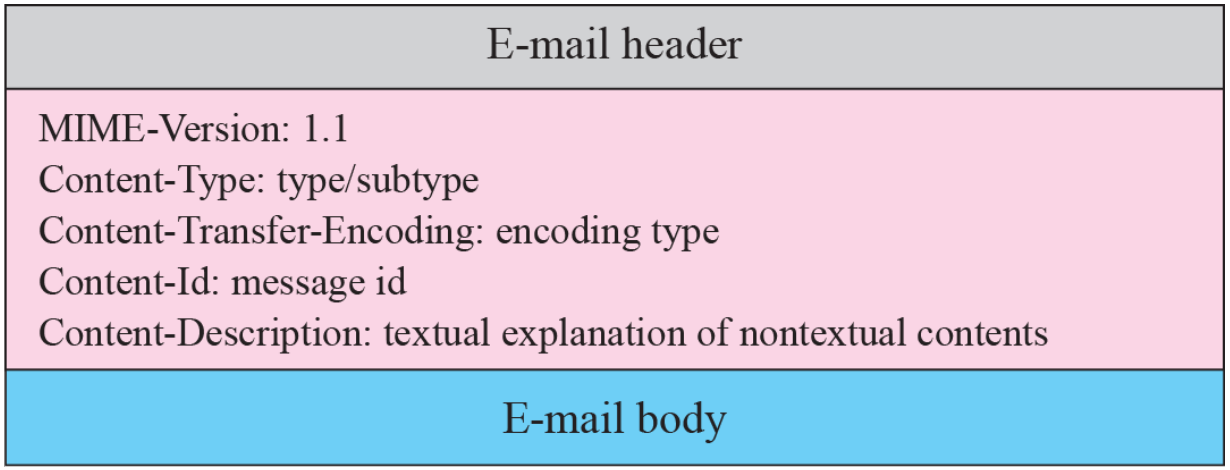
MIME SCHEME



NVT = network virtual terminal

MIME HEADERS

MIME headers



NON-ASCII CHARACTER SET SUPPORT

- Message header
  - content-type field
    - put in the header by the client program creating the e-mail for use by the client program used to display the received message
    - charset= optional parameter
      - if absent ASCII is assumed
- Content-Type: text/plain; charset="ISO-8859-1"
  - ISO-8859-1 extends the basic character set of ASCII to include many of the accented characters used in languages such as Spanish, French, German and Italian.
  - US-ASCII is the standard character set used in the US

CONTENT LABELING

- a set of registered MIME Types that map to specific file types
  - MIME Types consist of :
    - a primary type
    - a sub type separated by a / (as text/html)
- Common Mime Types:

FileExtension	MIME Type	Description
.txt	text/plain	Plain text
.htm	text/html	Styled text in HTML format
.jpg	image/jpeg	Picture in JPEG format
.gif	image/gif	Picture in GIF format
.wav	audio/x-wave	Sound in WAVE format
.mp3	audio/mpeg	Music in MP3 format
.mpg	video/mpeg	Video in MPEG format
.zip format	application/zip	Compressed file in PK-ZIP

MIME TYPES/SUBTYPES

Type	*Subtype*	*Description*
Text	Plain	Unformatted
	HTML	HTML format (see Appendix E)
Multipart	Mixed	Body contains ordered parts of different data types
	Parallel	Same as above, but no order
	Digest	Similar to Mixed, but the default is message/RFC822
	Alternative	Parts are different versions of the same message
Message	RFC822	Body is an encapsulated message
	Partial	Body is a fragment of a bigger message
	External-Body	Body is a reference to another message
Image	JPEG	Image is in JPEG format
	GIF	Image is in GIF format
Video	MPEG	Video is in MPEG format
Audio	Basic	Single channel encoding of voice at 8 KHz
Application	PostScript	Adobe PostScript
	Octet-stream	General binary data (eight-bit bytes)

CONTENT-TRANSFER-ENCODING

Type	*Description*
7bit	NVT ASCII characters and short lines
8bit	Non-ASCII characters and short lines
Binary	Non-ASCII characters with unlimited-length lines
Base64	6-bit blocks of data are encoded into 8-bit ASCII characters
Quoted-printable	Non-ASCII characters are encoded as an equal sign plus an ASCII code

NON-TEXT CONTENT

To be sent through the e-mail system non-textual content is converted (encoded) to ASCII for transmission and decoded back to its original format for display upon receipt

- originally done via uuencode
- MIME uses base 64 encoding (RFC 2045)
  - binary to text encoding scheme
  - targets A-Z, a-z, 0-9, +,/
- scheme:
  - take three bytes of data, put into a 24 bit buffer
  - extract 4 six-bits values
  - use each value as an index into: ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/
  - this yields 4 ASCII characters
  - use zero, one or two = symbols for padding (at the end)

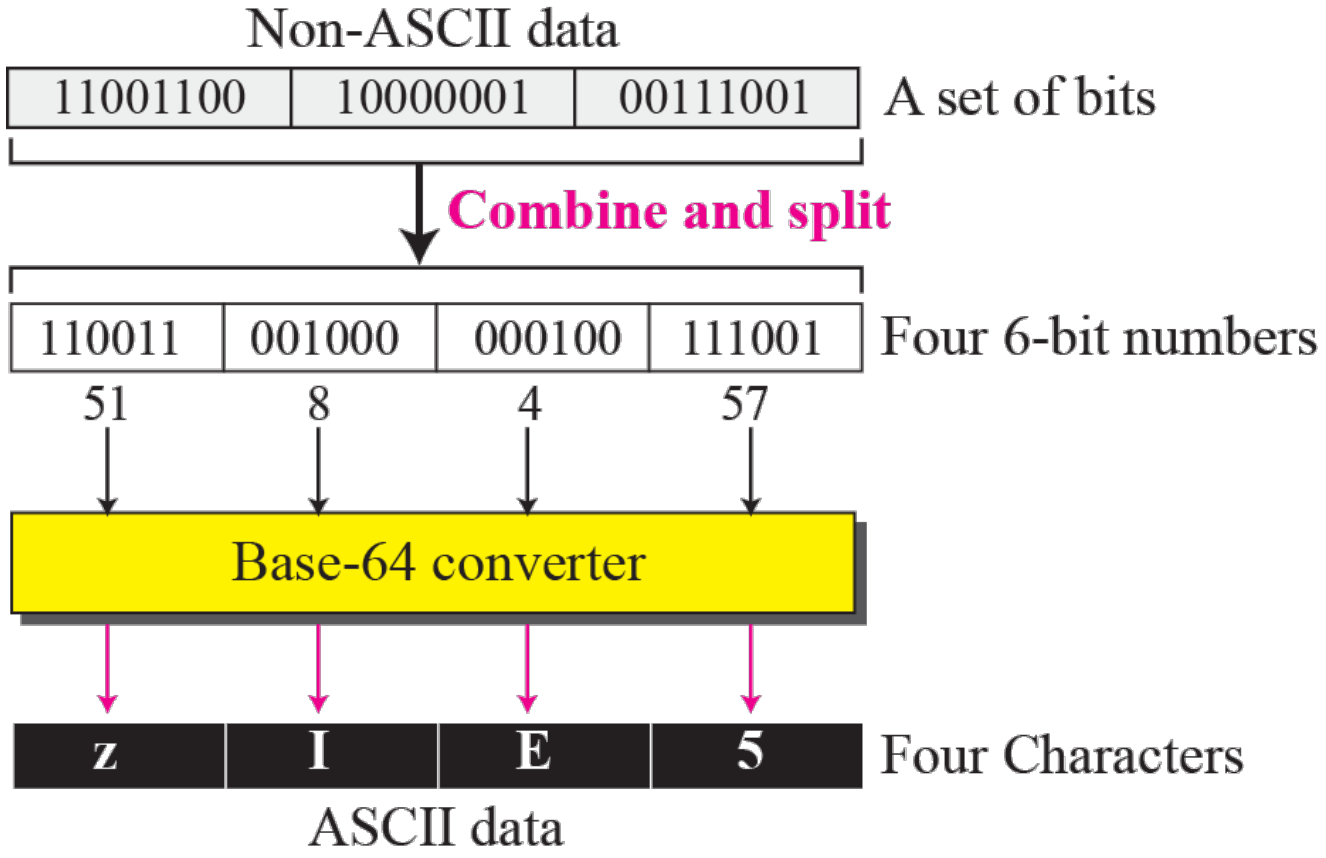
BASE-64 ENCODING EXAMPLE

Man is distinguished, not only by his reason, but by this singular passion from other animals, which is a lust of the mind, that by a perseverance of delight in the continued and indefatigable generation of knowledge, exceeds the short vehemence of any carnal pleasure.

base64 encoded:

TWFuIGlzIGRpc3Rpbmd1aXNoZWQsIG5vdCBvbmx5IGJ5IGhpcyByZWFzb24sIGJ1dCBieSB0aGlzIHNpbmd1bGFyIHBhc3Npb24gZnJvbSBvdGhlciBhbmltYWxzLCB3aGljaCBpcyBhIGx1c3Qgb2YgdGhlIG1pbmQsIHRoYXQgYnkgYSBwZXJzZXZlcmFuY2Ugb2YgZGVsaWdodCBpbiB0aGUgY29udGludWVkIGFuZCBpbmRlZmF0aWdhYmxlIGdlbmVyYXRpb24gb2Yga25vd2xlZGdlLCBleGNlZWRzIHRoZSBzaG9ydCB2ZWhlbWVuY2Ugb2YgYW55IGNhcm5hbCBwbGVhc3VyZS4=

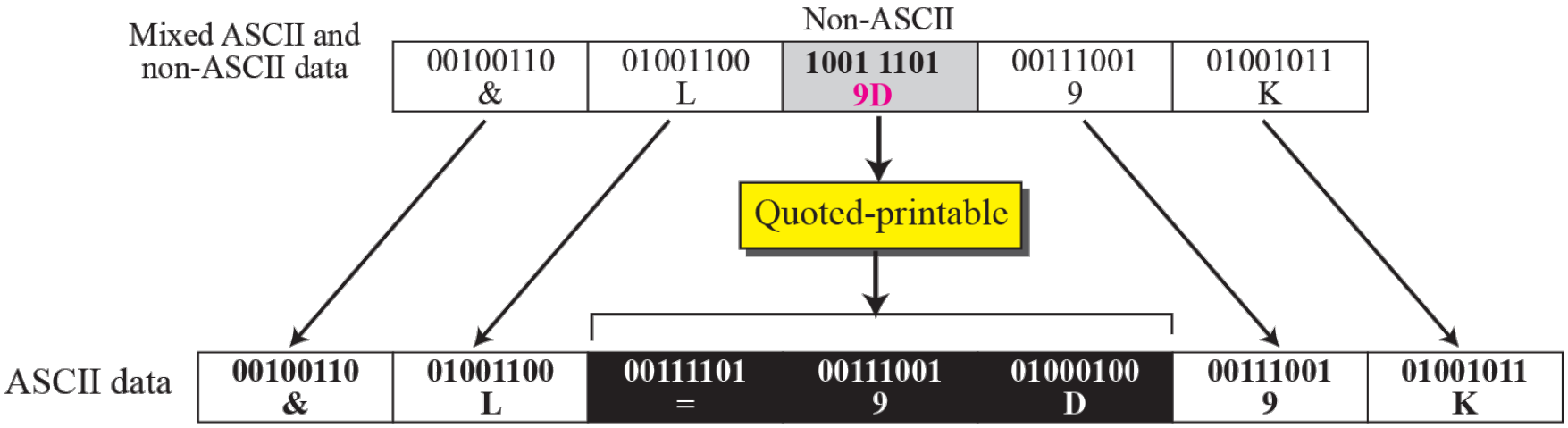
BASE-64 ENCODING SCHEMA



BASE-64 CONVERTING TABLE

Value	*Code*	*Value*	*Code*	*Value*	*Code*	*Value*	*Code*	*Value*	*Code*	*Value*	*Code*
0	**A**	11	**L**	22	**W**	33	**h**	44	**s**	55	**3**
1	**B**	12	**M**	23	**X**	34	**i**	45	**t**	56	**4**
2	**C**	13	**N**	24	**Y**	35	**j**	46	**u**	57	**5**
3	**D**	14	**O**	25	**Z**	36	**k**	47	**v**	58	**6**
4	**E**	15	**P**	26	**a**	37	**l**	48	**w**	59	**7**
5	**F**	16	**Q**	27	**b**	38	**m**	49	**x**	60	**8**
6	**G**	17	**R**	28	**c**	39	**n**	50	**y**	61	**9**
7	**H**	18	**S**	29	**d**	40	**o**	51	**z**	62	+
8	**I**	19	**T**	30	**e**	41	**p**	52	**0**	63	/
9	**J**	20	**U**	31	**f**	42	**q**	53	**1**		
10	**K**	21	**V**	32	**g**	43	**r**	54	**2**		

QUOTED-PRINTABLE ENCODING



- any 8-bit byte value may be encoded with 3 characters: an '=' followed by two hexadecimal digits (0–9 or A–F) representing the byte's numeric value
- non 8-bit byte values are ASCII chars from 33 to 126 (excluded 61, the '=' sign)
- special cases for SPACE and TAB

MULTIPART SUBTYPES

- **Mixed**. For sending files with different "Content-Type" headers.
- **Digest**. To send multiple text messages.
- **Message**. Contains any MIME email message, including any headers
- **Alternative**. Each part is an "alternative" version of the same (or similar) content (e.g., text + HTML)

- more subtypes…

MIME TYPES/SUBTYPES

```
From: Some One <someone@example.com>
MIME-Version: 1.0
Content-Type: multipart/mixed;
        boundary="XXXXboundary text"

This is a multipart message in MIME format.

--XXXXboundary text
Content-Type: text/plain

this is the body text

--XXXXboundary text
Content-Type: text/plain;
Content-Disposition: attachment;
        filename="test.txt"

this is the attachment text

--XXXXboundary text--
```

MIME TYPES/SUBTYPES

```
MIME-Version: 1.0
X-Mailer: MailBee.NET 8.0.4.428
Subject: test subject
To: kevinm@datamotion.com
Content-Type: multipart/mixed;
       boundary="XXXXboundary text"

--XXXXboundary text
Content-Type: multipart/alternative;
       boundary="XXXXboundary text"

--XXXXboundary text
Content-Type: text/plain;
       charset="utf-8"
Content-Transfer-Encoding: quoted-printable

This is the body text of a sample message.
--XXXXboundary text
Content-Type: text/html;
       charset="utf-8"
Content-Transfer-Encoding: quoted-printable
<pre>This is the body text of a sample message.</pre>

--XXXXboundary text
Content-Type: text/plain;
name="log_attachment.txt"
Content-Disposition: attachment;
filename="log_attachment.txt"
Content-Transfer-Encoding: base64

TUlNRS1WZXJzaW9uOiAxLjANClgtTWFpbGVyOiBNYWlsQmVlLk5FVCA4LjAuNC40MjgNClN1Ympl
```

PLAIN TEXT AND HTML

- modern graphic email clients allow use of HTML for the message body
  - HTML email messages often include an automatically generated plain text copy as well
- HTML messages should have an additional header: "Content-type: text/html". Most email programs insert this header automatically
- advantages of HTML include the ability to include in-line links and images, set apart previous messages in block quotes, wrap naturally on any display, use emphasis such as underlines and italics, and change font styles
- disadvantages include the increased size of the email, privacy concerns about web bugs, abuse of HTML email as a vector for phishing attacks and the spread of malicious software
- some mailing lists recommend that all posts be made in plain-text, with 72 or 80 characters per line for all the above reasons, but also because they have a significant number of readers using text-based email clients
- some Microsoft email clients allow rich formatting using RTF (portability issues)

UNWANTED E-MAIL MESSAGES

- SPAM = unwanted ads (?)
  - both normal and low quality merchandize (drugs, pharmacy, dating, online sex, pirated software/multimedia etc.)
- frauds/malware
  - "write here your username/password"
  - "write here your credit card number"
  - "help me to retrieve $ 20 000 000 …"
  - "you haven't claimed your € 500 prize"
  - loans and funds at lowest rates
  - "I'm so lonely and looking for love…"
  - "you won the lottery"
  - "the message you have sent is undeliverable"
  - "invoice to be paid: click here"
- e-mail chain letters
  - exponential growth
- all of above, joint to low-quality automatic language translation

we'll use the generic terms spam or junk for denoting unwanted or undesirable e-mail messages

WHAT DO SPAMMERS WANT?

- sell products/services (aggressive marketing)
- sell low-qualities/fake/expired goods/medicines (low prices)
- distribute/spread malware (viruses, worms, Trojan horses, backdoors, rootkits etc.) and grayware (adware, spyware, dialers etc.)
  - computer can be enrolled/controlled for participation in (future) attacks
  - Internet activity (browsing, instant messaging and other social activity) can be monitored, users can be profiled
  - audio/video sessions can be recorded
  - collect (any) data on you and on your contacts (databases are built to the purpose of digital identity thefts)
- phishing
  - username/password stealing, credit card data capture, frauds etc.
  - often based on malicious links
- validate e-mail addresses
  - can be re-sold at a higher price
  - based on HTML images and links
- this list is non-exhaustive

BASIC E-MAIL NONALOGUE

- disable HTML messages or, at least, disable download of remote images
  - prevent the sender to validate our e-mail address
- don't click links (specially if tiny or IP-based URLs)
  - could redirect to bad web sites containing malware/spyware
- don't open unknown/unexpected attachments
  - they may contain malware/spyware
  - executables (.exe, .app, .bat etc.), documents(.doc, .pdf etc.) and others (.src, …)
- activate local anti-spam filter
- don't participate with chain letters
  - google their contents!
- protect and respect privacy of other recipients
  - be careful in e-mail forwarding (don't uselessly disclose e-mail addresses)
- even if non-Windows user, activate anti-virus for protecting your (Windows) recipients
- don't provide your personal/sensitive data
  - identity thefts!
- don't click "delete me"
  - may validate your email address
  - OK with known senders

HTML IMAGES

- could be used for e-mail address validation
- prepare in remote http server a file image to be referred in HTML formatted message (e.g., "small.gif")
- image may be a small dot having same color as background
- for each e-mail address make symbolic link (alias) to image file and define a corresponding table

address	alias
a@b.c	s1.gif
d@e.f	s2.gif
g@h.i	s3.gif

- print/merge HTML messages using the above table (with each recipient use appropriate filename)
- send messages, then grep logs of remote http server

EXPANSION OF TINY URLS

how to?

- click & see
  - risky!
- use analyzing tools
  - where can we find them?
- ad hoc services on the Web
  - e.g.: https://urlex.org
  - good results?

URL CHECKING

twitted (twitpic) on March 18, 2013

Google Chrome

- **Sends the name of the file you're downloading** to Google for whitelist checking; **stores your IP address** associated with the file for a few weeks
- Asks you to login to your Google account, so your **browsing tabs, history, etc. is stored on Google servers**
- **Every URL** you even begin to type in the address bar is **sent to Google**, in whole or in fragments, for auto-completion purposes
- **Connects to websites** in the background before you are even finished typing them in, **without your explicit instruction**
- **Connects to Google every 30 minutes** to download a list of malicious URLs, so the fact that you even have Chrome open is transmitted to Google

Summary: there is nothing, *nothing*, you can do in Chrome that isn't transmitted to Google through some channel.

Welcome to the Botnet.

E-MAIL VALIDATION SYSTEMS

- Sender Policy Framework (SPF)
  - prevents e-mail spam by detecting email spoofing through verification of sender IP addresses
  - well known project
    - http://www.openspf.org/Project_Overview
  - RFC 4408
- DomainKeys Identified Mail (DKIM)
  - allows to check that incoming mail from a domain is authorized by that domain's administrators and that the email (including attachments) has not been modified during transport
  - RFC 4871
- Vouch by Reference (VBR)
  - implements sender certification by third-party entities
  - RFC 5518
  - certification based on DomainKeys Identified Mail (DKIM)

SENDER POLICY FRAMEWORK

- SPFv1 (or SPF Classic) protects the sender address (in envelope) by allowing the owner of a domain to specify a mail sending policy, namely which mail servers are authorized to send mail from the domain, using special DNS records (SPF, type 99)
- Receivers verifying the SPF records may reject messages from unauthorized sources before receiving the body of the message
- If server accepts the sender, and also accepts recipients and body of message, it should insert a Return-Path field in the message header in order to save the sender address
  - While the address in the Return-Path often matches other originator addresses in the mail header such as From or Sender, this is not necessarily the case, and SPF does not prevent forgery of these other addresses

SPF EXAMPLE

Bob owns domain example.net. He also sometimes sends mail through his GMail account and contacted GMail's support to identify the correct SPF record for GMail.

Since he often receives bounces about messages he didn't send, he decides to publish an SPF record in order to reduce the abuse of his domain in e-mail envelopes:

example.net TXT "v=spf1 mx a:pluto.example.net include:aspmx.googlemail.com -all"

SPF EXAMPLE CONT'D

example.net TXT "v=spf1 mx a:pluto.example.net include:aspmx.googlemail.com -all"

SPF record items	explanation
v=spf1	SPF version 1
mx	the incoming mail servers (MXes) of the domain are authorized to also send mail for example.net
a:pluto.example.net	the machine pluto.example.net is authorized, too
include:aspmx.googlemail.com	everything considered legitimate by gmail.com is legitimate for example.net, too
-all	all other machines are not authorized

DOMAIN VS IP CHECK

- mail servers (e.g. Gmail) that are contacted by clients may simply check client's IP against sender's domain name: on mismatch, message is rejected

```
<XXXX.YYYY@gmail.com>: host gmail-smtp-in.l.google.com[173.194.78.26]
    said: 550-5.7.1 [aa.bb.cc.dd] The IP you're using to send mail is not
    authorized to 550-5.7.1 send email directly to our servers. Please use the
    SMTP relay at your 550-5.7.1 service provider instead. Learn more at 550
    5.7.1 http://support.google.com/mail/bin/answer.py?answer=10336
    fl4si3665795wib.12 - gsmtp (in reply to end of DATA command)
```

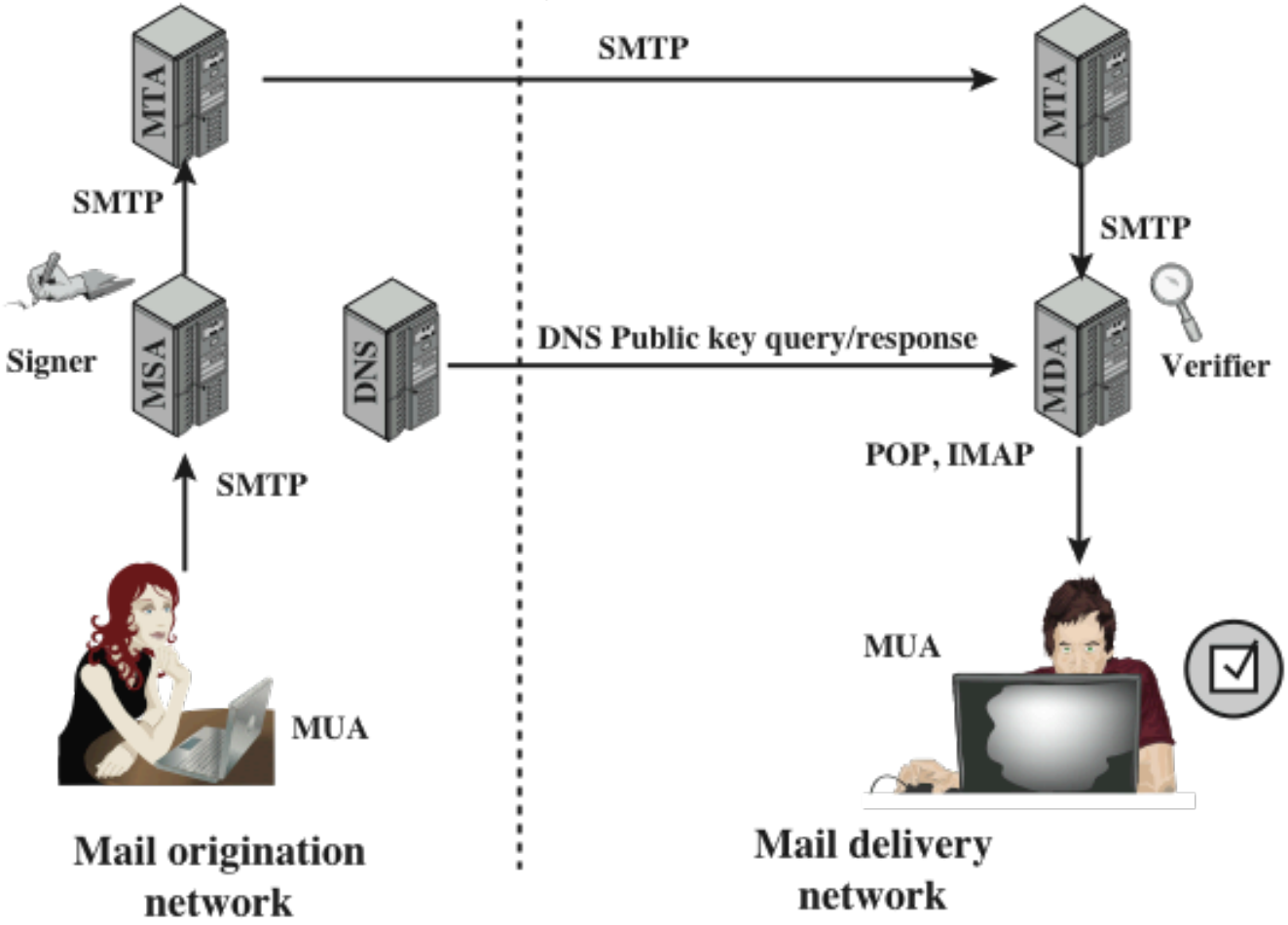
SPF LIMITATIONS

- Keeping SPF records updated as companies change service providers and add mail streams is difficult.
- Just because a message fails SPF, doesn't mean it will always be blocked from the inbox — it's one of several factors email providers take into account.
- SPF breaks when a message is forwarded (true only for some forwarding methods).
- SPF does nothing to protect companies against cybercriminals who spoof the display name or "header from" address in their message, which is the more frequently spoofed "from" address since it's the address most visible to the email recipient.

DKIM

- DomainKeys Identified Mail (DKIM) is a specification for cryptographically signing e-mail messages, permitting a signing domain to claim responsibility for a message.
- Message recipients (or agents acting in their behalf) can verify the signature by querying the signer's domain directly to retrieve the appropriate public key and thereby can confirm that the message was attested to by a party in possession of the private key for the signing domain.
- DKIM is a proposed Internet Standard (RFC 4871: DomainKeys Identified Mail (DKIM) Signatures).
- DKIM has been widely adopted by a range of e-mail providers, including corporations, government agencies, gmail, yahoo, and many Internet service providers (ISPs).

POSSIBLE DKIM DEPLOYMENT



CANONICALIZATION

- e-mail servers and relay systems may modify email in transit, potentially invalidating a signature
- headers are subjected to a canonicalization algorithm
  - relaxed (tolerating) or simple (strict)
- bodies are also subjected to a canonicalization algorithm
  - choices for header/body are independent
- see RFC 4871 for details

SELECTORS

- To support multiple concurrent public keys per signing domain, key namespace is subdivided using **selectors**
  - for example selectors might indicate the names of office locations, the signing date, or even the individual user
- Selectors are useful to implement some important use cases
  - domains that want to delegate signing capability for a specific address for a given duration to a partner, such as an advertising provider or other outsourced function
  - domains that want to allow frequent travelers to send messages locally without the need to connect with a particular MSA.
  - "affinity" domains (e.g., college alumni associations) that provide forwarding of incoming mail, but that do not operate a MSA for outgoing mail

DKIM EXAMPLE

a = Hash/signing algorithm

q = Algorithm for getting public key

d = Signing domain

i = Signing identity

s = Selector

c = Canonicalization algorithm

t = Signing time (seconds since 1/1/1970)

x = Expiration time

h = List of headers included in signature; dkim-signature is implied

b = The signature itself

bh = The hash of the canonicalized body part of the message

```
Received: by mail-wg0-f44.google.com with SMTP id dr12so5400749wgb.35
        for <damore@dis.uniroma1.it>; Mon, 18 Mar 2013 14:17:04 -0700 (PDT)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
        d=gmail.com; s=20120113;
        h=x-received:mime-version:in-reply-to:references:from:date:message-id
         :subject:to:content-type;
        bh=I7Gc1zNUyy13QDKdzeRoGrgVCJaaKCpVqUjIPSV24P8=;
        b=u1yT9znzpgvzRm4/hiXZKtrq77auuYbqT7HjzpKAL4siHsKKlCZNgElIiPXLHk6Y6l
         7daYBXnicBUiZLkU5jaoo/uK+IocGZNbCEJ0nC0A42mNxX4GkL84JiMNjXvdd4wMTvMF
         IUUgjQLk710OZYas9rCSMCkK48e8SeVbTFnAF42BhqF4rIXbHN/9PhlUy7AXuqnE1SSy
         BtRfS28eSlO7xjRR7Lkg+VHgsAIhMRn/SNVle1T09lXwWIJSXayjlPzQREb1DYQM8B6n
         xSuqSIwztkshtTd2BjC2JrORKXa+tUeTBZjA3vzDKiG7dMqEMJxMN9i2GN8VK2IiAR69
         Kkig==
```

DMARC

Domain-based Message Authentication, Reporting, and Conformance, is a technical standard (RFC 7489) that helps protect email senders and recipients from spam, spoofing, and phishing.

DMARC allows an organization to publish a policy that defines its email authentication practices and provides instructions to receiving mail servers for how to enforce them.

Specifically, DMARC establishes a method for a domain owner to:

- Publish its email authentication practices
- State what actions should be taken on mail that fails authentication checks
- Enable reporting of these actions taken on mail claiming to be from its domain

DMARC itself is not itself an email authentication protocol, but it builds on key authentication standards SPF and DKIM.

HOW DOES DMARC WORK?

1) A domain administrator publishes the policy defining its email authentication practices and how receiving mail servers should handle mail that violates this policy.

This DMARC policy is listed as part of the domain's overall DNS records.

A DMARC record is included in an organization's DNS database. It is a specially-formatted version of a standard DNS TXT record with a particular name, namely "_dmarc.mydomain.com"

HOW DOES DMARC WORK?

DMARC record example

```
_dmarc.mydomain.com. IN TXT "v=DMARC1\; p=none\; rua=mailto:dmarc-
aggregate@mydomain.com\; ruf=mailto:dmarc-afrf@mydomain.com\; pct=100"
Reading left-to-right in plain English, this record says:
```

- **v=DMARC1** specifies the DMARC version
- **p=none** specifies the preferred treatment, or DMARC policy
- **rua=mailto:dmarc-aggregate@mydomain.com** is the mailbox to which aggregate reports should be sent
- **ruf=mailto:dmarc-afrf@mydomain.com** is the mailbox to which forensic reports should be sent
- **pct=100** is the percentage of mail to which the domain owner would like to have its policy applied

HOW DOES DMARC WORK?

DMARC record example

```
_dmarc.mydomain.com. IN TXT "v=DMARC1\; p=none\; rua=mailto:dmarc-
aggregate@mydomain.com\; ruf=mailto:dmarc-afrf@mydomain.com\; pct=100"
Reading left-to-right in plain English, this record says:
```

The DMARC specification provides three choices for domain owners to use to specify their preferred treatment of mail that fails DMARC validation checks. These "p= policies" are:

- none: treat the mail the same as it would be without any DMARC validation
- quarantine: accept the mail but place it somewhere other than the recipient's inbox (typically the spam folder)
- reject: reject the message outright

HOW DOES DMARC WORK?

2) When an inbound mail server receives an incoming email, it uses DNS to look up the DMARC policy for the domain contained in the message's "From" (RFC 5322) header.

The inbound server then checks evaluates the message for three key factors:

- Does the message's DKIM signature validate?
- Did the message come from IP addresses allowed by the sending domain's SPF records?
- Do the headers in the message show proper "domain alignment"?

HOW DOES DMARC WORK?

2) When an inbound mail server receives an incoming email, it uses DNS to look up the DMARC policy for the domain contained in the message's "From" (RFC 5322) header.

"Domain alignment" is a concept in DMARC that expands the domain validation intrinsic to SPF and DKIM. DMARC domain alignment matches a message's "from" domain with information relevant to these other standards:

- For SPF, the message's From domain and its Return-Path domain must match
- For DKIM, the message's From domain and its DKIM d= domain must match

HOW DOES DMARC WORK?

3)With this information, the server is ready to apply the sending domain's DMARC policy to decide whether to accept, reject, or otherwise flag the email message.

4) After using DMARC policy to determine the proper disposition for the message, the receiving mail server will report the outcome to the sending domain owner.

HOW DOES DMARC WORK?

DMARC reports are generated by inbound mail servers as part of the DMARC validation process. There are two formats of DMARC reports:

- Aggregate reports, which are XML documents showing statistical data about the messages received that claimed to be from a particular domain. Data reported includes authentication results and message disposition. Aggregate reports are designed to be machine-readable.
- Forensic reports, which are individual copies of messages which failed authentication, each enclosed in a full email message using a special format called AFRF. Forensic report can be useful both for troubleshooting a domain's own authentication issues and for identifying malicious domains and web sites.

CHECK OUTCOMES

Outcomes from protocol checks are reported in the mail headers

```
Authentication-Results: mx.google.com;
       dkim=pass header.i=@enisa.europa.eu header.s=enisadkim
header.b=B8Ea0tk+;
       spf=pass (google.com: domain of prokopios.drogkaris@enisa.europa.eu
designates 139.91.222.30 as permitted sender)
```

VOUCH BY REFERENCE

- VBR is a protocol used in Internet mail systems for implementing **sender certification by third-party entities**
  - Independent certification providers vouch for the reputation of senders by verifying the domain name that is associated with transmitted electronic mail
- **Email sender**. A user of a VBR email certification service signs its messages using DomainKeys Identified Mail (DKIM) and includes a VBR-Info field in the signed header
  - The sender may also use the Sender Policy Framework to authenticate its domain name
  - The VBR-Info: header field contains the domain name that is being certified, the type of content in the message, and a list of one or more vouching services, that is the domain names of the services that vouch for the sender for that kind of content:
    VBR-Info: md=domain.name.example; mc=type;
    mv=vouching.example:vouching2.example

VOUCH BY REFERENCE

- **Email receiver**. An email receiver can authenticate the message's domain name using DKIM or SPF, thus finding the domains that are responsible for the message. It then obtains the name of a vouching service that it trusts, either from among the set supplied by the sender or from a locally configured set of preferred vouching services. Using the DNS, the receiver can verify whether a vouching service actually vouches for a given domain. To do so, the receiver queries a TXT resource record for the name composed:
  domain.name.example._vouch.vouching.example
- The returned data, if any, is a space-delimited list of all the types that the service vouches, given as lowercase ASCII. They should match the self-asserted message content. The types defined are transaction, list, and all. Auditing the message may allow to establish whether its content corresponds. The result of the authentication can be saved in a new header field, according to RFC 6212, like so:
  Authentication-Results: receiver.example; vbr=pass
  header.mv=vouching.example header.md=domain.name.example

MORE INSIGHT

- "The World of E-mail Authentication" http://www.openspf.org/Related_Solutions
- "Vouch By Reference specification" http://www.domain-assurance.org/protocol-specification.phtml
- Wikipedia

BASIC E-MAIL ANALYSIS

topics

- **e-mail spoofing**

- **intro to e-mail forensics (e-mail tracking)**

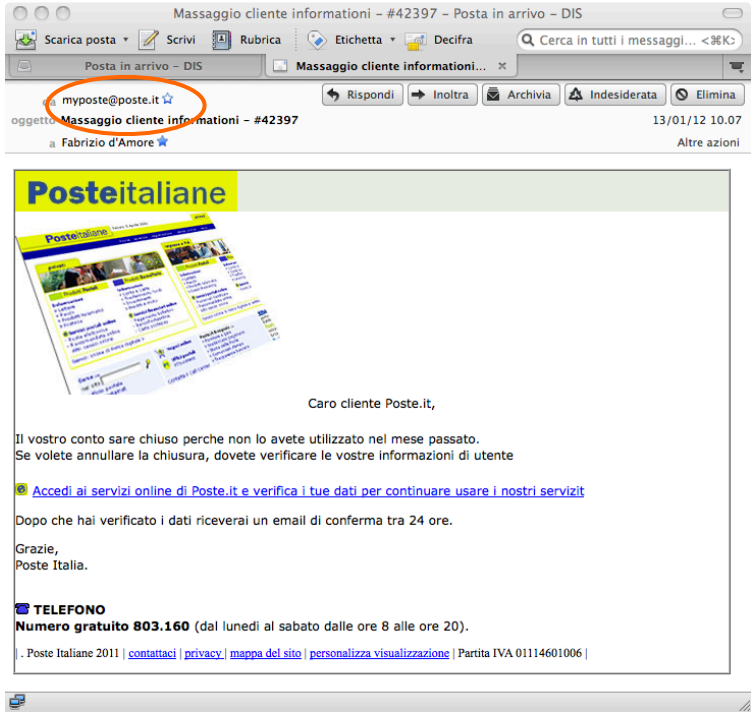
E-MAIL SPOOFING

- activity of altering the e-mail's sender address to the purpose of making the message looking like originated from other sender
  - the spoofer will possibly alter other fields
- easy in the plain Internet e-mail system, since original SMTP doesn't provide any authentication
  - later, a few mechanisms for authentication have been introduced, such as SMTP-AUTH
- most of spam/phishing e-mail messages are spoofed

A TYPICAL EXAMPLE

Sorgente di: imap://damore@imap.dis.uniroma1.it:993/fetch%3EUID%3E/Junk%3E47620

```
Return-Path: <root@periscope.hu>
X-Original-To: damore@dis.uniroma1.it
Delivered-To: damore@dis.uniroma1.it
Received: from localhost (webmail.dis.uniroma1.it [151.100.59.69])
        by mail.dis.uniroma1.it (Postfix) with ESMTP id B7AB628203
        for <damore@dis.uniroma1.it>; Fri, 13 Jan 2012 10:14:32 +0100 (CET)
Received: from webmail.dis.uniroma1.it ([127.0.0.1])
 by localhost (webmail [127.0.0.1]) (amavisd-new, port 10024) with ESMTP
 id 08099-11 for <damore@dis.uniroma1.it>;
 Fri, 13 Jan 2012 10:14:29 +0100 (CET)
Received: from morpheus.periscope.hu (shosting-26.84.nethub.hu [87.229.26.84])
        by webmail.dis.uniroma1.it (Postfix) with ESMTP id DBC23154AA1
        for <damore@dis.uniroma1.it>; Fri, 13 Jan 2012 10:14:28 +0100 (CET)
Received: by morpheus.periscope.hu (Postfix, from userid 0)
        id B3108A147102; Fri, 13 Jan 2012 10:07:01 +0100 (CET)
To: damore@dis.uniroma1.it
Subject: Massaggio cliente informationi - #42397
From: myposte@poste.it
Content-Type: text/html
Message-Id: <20120113090701.B3108A147102@morpheus.periscope.hu>
```



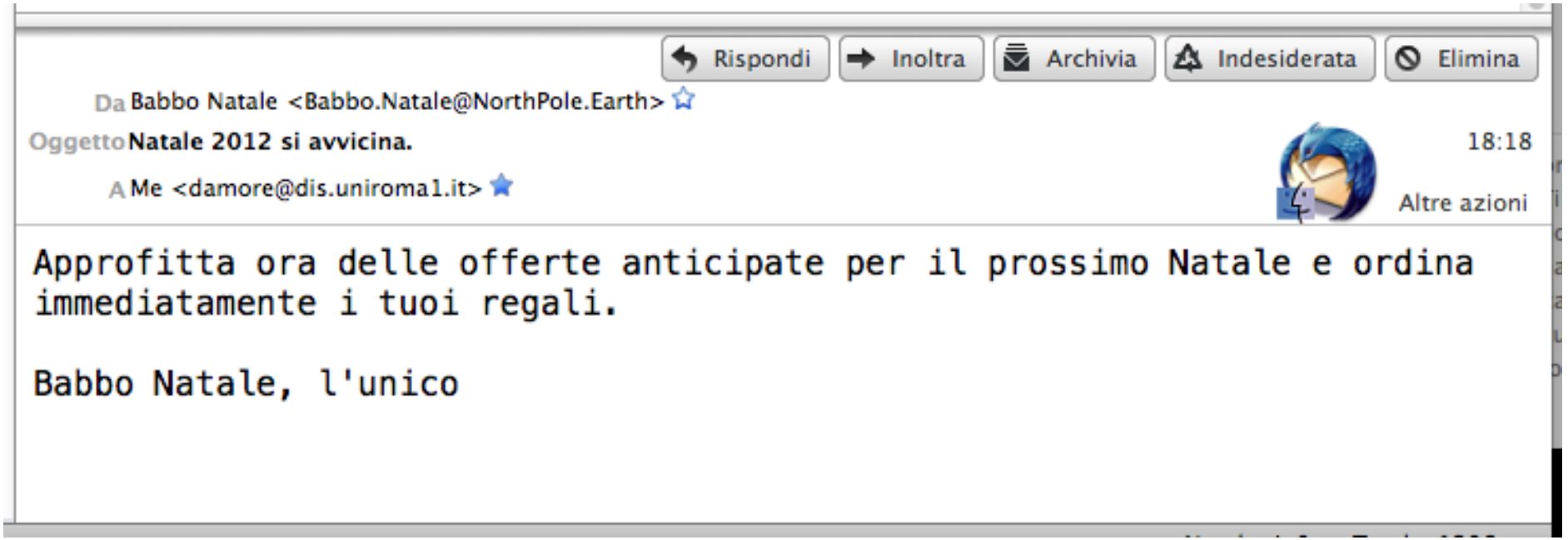
SIMPLE SPOOFING SESSION

```
telnet mail.dis.uniroma1.it 25
Trying 151.100.59.100...
Connected to mail.dis.uniroma1.it.
Escape character is '^]'.
220 Mail Server ESMTP
helo babbonatale
250 mail.dis.uniroma1.it
mail from:<Babbo.Natale@NorthPole.Earth>
250 Ok
rcpt to:<damore@dis.uniroma1.it>
250 Ok
data
354 End data with <CR><LF>.<CR><LF>
Message-ID: 4F525268.40304@NorthPole.Earth
Date: Sat, 03 Mar 2012 18:18:32 +0100
From: Babbo Natale
<Babbo.Natale@NorthPole.Earth>
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac
OS X 10.7; rv:10.0.2) Gecko/20120216
Thunderbird/10.0.2
MIME-Version: 1.0
To: Fabrizio d'Amore <damore@dis.uniroma1.it>
Subject: Natale 2012 si avvicina.
Content-Type: text/plain; charset=ISO-8859-15
Content-Transfer-Encoding: 7bit

Approfitta ora delle offerte anticipate per il
prossimo Natale e ordina immediatamente i tuoi
regali.

Babbo Natale, l'unico
.
250 Ok: queued as AF3B722FDD
quit
221 Bye
Connection closed by foreign host.
```

SPOOFING RESULT

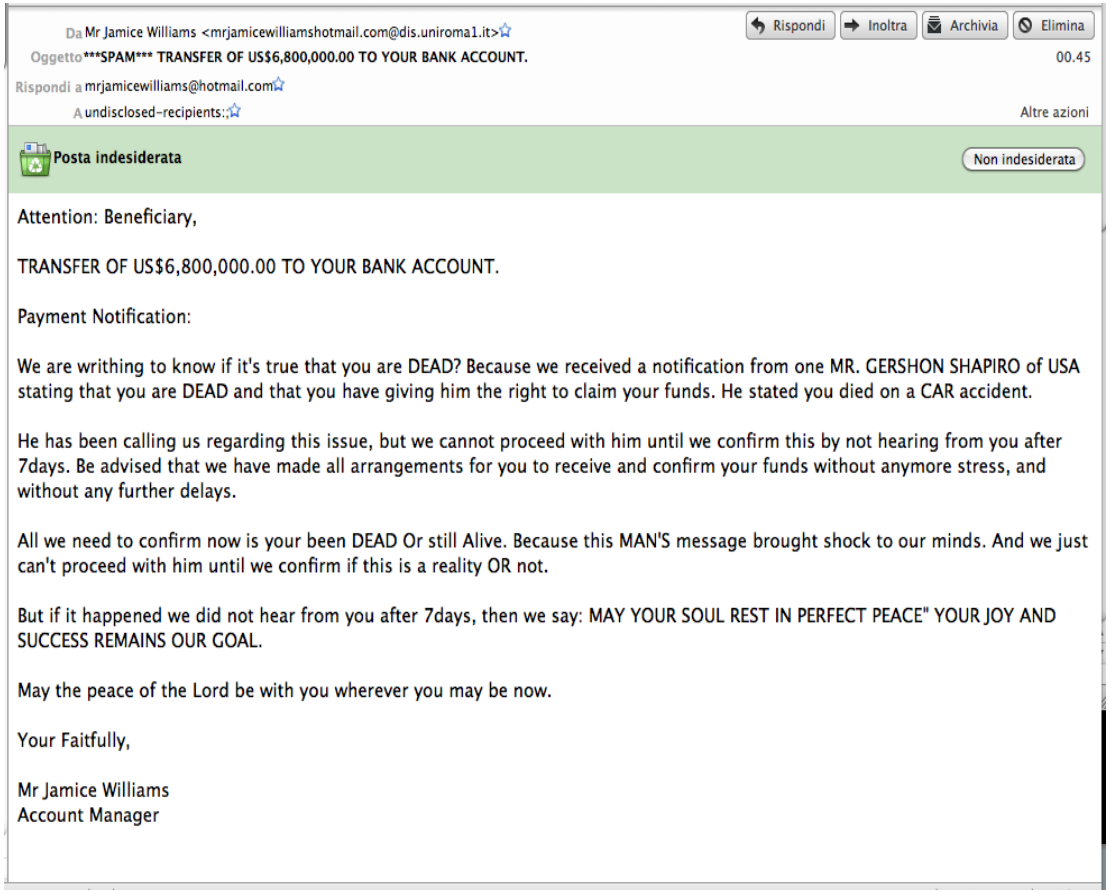


HOW TO CHECK FOR SPOOFING

- no success-guaranteeing techniques
  - it is often easy to detected spoofed messages
  - sometimes it is hard or almost impossible
- a good chance is to analyze the complete message (full header + body)
  - standard e-mail clients normally hide most of the header, since considered uninteresting
  - the analyst has to get the integral and original message: no standard GUI, IMAP can be good means
  - check fields From, Return-Path, Reply-To, Received
    - compare values (not all fields necessarily present in header)
    - lookup IP numbers (if any) and check domain names
      - many tools available for that

SPAM EXAMPLE

- message delivered to official e-mail address, published in web site
- Thunderbird labeled it as spam
- sender looks to be "Mr Jamice Williams"
- delivered to multiple hidden recipients (BCC)
- in Thunderbird (Mac OS) source (full text) of message can be quickly obtained by pressing CMD-U



Da Mr Jamice Williams <mrjamicewilliamshotmail.com@dis.uniroma1.it>
Oggetto ***SPAM*** TRANSFER OF US$6,800,000.00 TO YOUR BANK ACCOUNT.
Rispondi a mrjamicewilliams@hotmail.com
A undisclosed-recipients:;

Rispondi | Inoltra | Archivia | Elimina
00.45
Altre azioni

Posta indesiderata
Non indesiderata

Attention: Beneficiary,

TRANSFER OF US$6,800,000.00 TO YOUR BANK ACCOUNT.

Payment Notification:

We are writhing to know if it's true that you are DEAD? Because we received a notification from one MR. GERSHON SHAPIRO of USA stating that you are DEAD and that you have giving him the right to claim your funds. He stated you died on a CAR accident.

He has been calling us regarding this issue, but we cannot proceed with him until we confirm this by not hearing from you after 7days. Be advised that we have made all arrangements for you to receive and confirm your funds without anymore stress, and without any further delays.

All we need to confirm now is your been DEAD Or still Alive. Because this MAN'S message brought shock to our minds. And we just can't proceed with him until we confirm if this is a reality OR not.

But if it happened we did not hear from you after 7days, then we say: MAY YOUR SOUL REST IN PERFECT PEACE" YOUR JOY AND SUCCESS REMAINS OUR GOAL.

May the peace of the Lord be with you wherever you may be now.

Your Faitfully,

Mr Jamice Williams
Account Manager

SPAM ANALYSIS

a few interesting headers

Sorgente di: imap://damore@imap.dis.uniroma1.it:993/fetch%3EUID%3E/Junk%3E48069

```
Return-Path: <mrjamicewilliamshotmail.com@uictech.com.cn>
X-Original-To: damore@dis.uniroma1.it
Delivered-To: damore@dis.uniroma1.it
Received: from localhost (webmail.dis.uniroma1.it [151.100.59.69])
        by mail.dis.uniroma1.it (Postfix) with ESMTP id 9333B22174
        for <damore@dis.uniroma1.it>; Sat, 10 Mar 2012 00:47:47 +0100 (CET)
Received: from webmail.dis.uniroma1.it ([127.0.0.1])
 by localhost (webmail [127.0.0.1]) (amavisd-new, port 10024) with ESMTP
 id 28570-13 for <damore@dis.uniroma1.it>;
 Sat, 10 Mar 2012 00:47:42 +0100 (CET)
Received: from mial.uictech.com.cn (unknown [121.52.214.219])
        by webmail.dis.uniroma1.it (Postfix) with SMTP id 1BD9026AF0A
        for <damore@dis.uniroma1.it>; Sat, 10 Mar 2012 00:47:01 +0100 (CET)
Received: from user ([41.203.64.150])
        (envelope-sender <mrjamicewilliamshotmail.com>)
        by 121.52.214.219 with ESMTP
        for <damon@eurca-gazette.com.au>; Sat, 10 Mar 2012 07:45:31 +0800
Reply-To: <mrjamicewilliams@hotmail.com>
From: "Mr Jamice Williams" <mrjamicewilliamshotmail.com@dis.uniroma1.it>
Subject: ***SPAM*** TRANSFER OF US$6,800,000.00 TO YOUR BANK ACCOUNT.
Date: Fri, 9 Mar 2012 15:45:36 -0800
MIME-Version: 1.0
Content-Type: text/html;
        charset="Windows-1251"
Content-Transfer-Encoding: 7bit
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 6.00.2600.0000
X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2600.0000
X-Antivirus: avast! (VPS 120309-0, 03/09/2012), Outbound message
X-Antivirus-Status: Clean
Message-Id: <20120309234701.1BD9026AF0A@webmail.dis.uniroma1.it>
To: undisclosed-recipients:;
X-Virus-Scanned: by amavisd-new at dis.uniroma1.it
X-Spam-Status: Yes, hits=9.2 tagged_above=-99.0 required=8.0 tests=BAYES_50,
 FORGED_HOTMAIL_RCVD2, FORGED_MUA_OUTLOOK, FORGED_OUTLOOK_HTML,
 FORGED_OUTLOOK_TAGS, HTML_MESSAGE, MIME_HTML_ONLY, MSOE_MID_WRONG_CASE,
 RCVD_IN_BL_SPAMCOP_NET, RCVD_IN_SORBS_WEB, RDNS_NONE, SUBJ_ALL_CAPS,
 US_DOLLARS_3
X-Spam-Level: *********
X-Spam-Flag: YES
```

Riga 41, col. 17

FIRST HOP

questions

a) whom 41.203.64.130 is registered to?

b) whom 121.52.214.219 is registered to?

c) whom euroa-gazette.com.au is registered to?

d) are these data compatible?

first hop basic data

```
Received: from User
([41.203.64.130]) (envelope-sender
<mrjamicewilliamshotmail.com>) by
121.52.214.219 with ESMTP for
<damon@euroa-gazette.com.au>; Sat,
10 Mar 2012 07:45:31 +0800
```

IP Information for 41.203.64.130

IP Location:	Nigeria Abuja Glo-mobile
ASN:	AS37148
IP Address:	41.203.64.130 W R P D T

```
inetnum:        41.203.64.0 - 41.203.65.255
netname:        GLOBACOM
descr:          GLO-Mobile Network Services
country:        NG
admin-c:        PA2-AFRINIC
tech-c:         PA2-AFRINIC
status:         ASSIGNED PA
mnt-by:         GLO-ONLINE-ADMIN
source:         AFRINIC # Filtered
parent:         41.203.64.0 - 41.203.95.255

person:         Prasoon Agarwal
nic-hdl:        PA2-AFRINIC
address:        1- Mike Adenuga Close, Victoria Island
address:        Lagos
address:        Lagos
address:        Nigeria
e-mail:         michael.okoduwa@gloworld.com

phone:          +2348055571050
phone:          +2348055570601
source:         AFRINIC # Filtered
```

S

IP Information for 121.52.214.219

IP Location:	China Beijing Beijing Topnew Info&tech Co .ltd
ASN:	AS4808
IP Address:	121.52.214.219 W R P D T
Reverse IP:	2 websites use this address. (examples: tanchengtax.com uictech.com.cn)

```
inetnum:        121.52.208.0 - 121.52.223.255
netname:        TopnewNET
descr:          Beijing Topnew Info&Tech co,.LTD.
descr:          No.9 A JintailiJiaÃ£Â¬Chaoyang DistrictÃ£Â¬Beijing China
country:        CN
admin-c:        HG335-AP
tech-c:         CL1725-AP
mnt-by:         MAINT-CNNIC-AP
mnt-lower:      MAINT-CNNIC-AP
mnt-routes:     MAINT-CNNIC-AP
status:         ALLOCATED PORTABLE
changed:        hm-changed@apnic.net 20071107
source:         APNIC

person:         Hongbo Gao
nic-hdl:        HG335-AP
e-mail:         gao@topnew.cn
address:        No.9 A JintailiJia£¬Chaoyang District£¬Beijing China
phone:          +86-10-52081277
fax-no:         +86-10-52081280
country:        CN
changed:        ipas@cnnic.net.cn 20071106
mnt-by:         MAINT-CNNIC-AP
source:         APNIC

person:         Chaocheng Li
nic-hdl:        CL1725-AP
e-mail:         lcc@topnew.cn
address:        No.9 A JintailiJia£¬Chaoyang District£¬Beijing China
phone:          +86-10-52081208
fax-no:         +86-10-52081280
country:        CN
changed:        ipas@cnnic.net.cn 20071106
mnt-by:         MAINT-CNNIC-AP
source:         APNIC
```

moreover

- euroa-gazette.com.au is registered to "Euroa Gazette Newspaper", an Aussie company
- the website of "The Euroa Gazette" for long time (about 2 years) showed news of October 13, 2009 (message has been sent on March 10, 2012)

courtesy of

DomainTools
The recognized leader in Internet domain name intelligence

RESULT OF FIRST-HOP ANALYSIS

message has been **sent** from a host registered to some Nigerian organization and **received** by a Chinese organization, that has been also informed that the **final recipient** belongs to an Aussie organization

SECOND HOP

questions

a) whom mial.uictech.com.cn is registered to?

b) why IP 121.52.214.219 is labeled as unknown?

c) what compatibility between such data?

second hop basic data

```
Received: from mial.uictech.com.cn
(unknown [121.52.214.219])
        by webmail.dis.uniroma1.it
(Postfix) with SMTP id 1BD9026AF0A
        for <damore@dis.uniroma1.it>;
Sat, 10 Mar 2012 00:47:01 +0100
(CET)
```

SECOND-HOP ANALYSIS

>whois uictech.com.cn

Domain Name: uictech.com.cn
ROID: 20061205s10011s12255687-cn
Domain Status: ok
Registrant ID: hc812883321-cn
Registrant Organization: 北京联友创嘉科技发展有限公司
Registrant Name: 陈文杰
Registrant Email:
Sponsoring Registrar: 北京万网志成科技有限公司
Name Server:dns11.hichina.com
Name Server:dns12.hichina.com
Registration Date: 2006-12-05 16:32:09
Expiration Date: 2012-12-05 16:32:09
Dnssec Deployment: N

after three attempts (first ones were void):

>nslookup uictech.com.cn

Non-authoritative answer:

Name: uictech.com.cn

Address: 121.52.214.219

data are compatible!

RESULT OF ANALYSIS

- message from Nigeria to China (with claimed final destination in Australia), then from China to Italy looks scarcely convincing
  - in particular there seems to be no reason why the Chinese server has delivered it to server in Sapienza (no explicit recipients of Sapienza are written in message)
- identity of Chinese server appears to be reasonably assured, since it is confirmed by Sapienza server
  - if Sapienza server was been captured, confirmation is unreliable
- initial Nigerian origin is only attested by Chinese server

MESSAGE IS COMPATIBLE WITH A PHISHING ATTEMPT ORIGINATED IN CHINA AND DELIVERED WITH SPOOFING TECHNIQUES AND ADULTERATED HEADERS

E-MAIL SECURITY NEEDS

- **confidentiality**: protection from disclosure
- **authentication** of sender of message
- **message integrity**: protection from modification
- **non-repudiation** of origin: protection from denial by sender

SECURING E-MAIL BY PGP

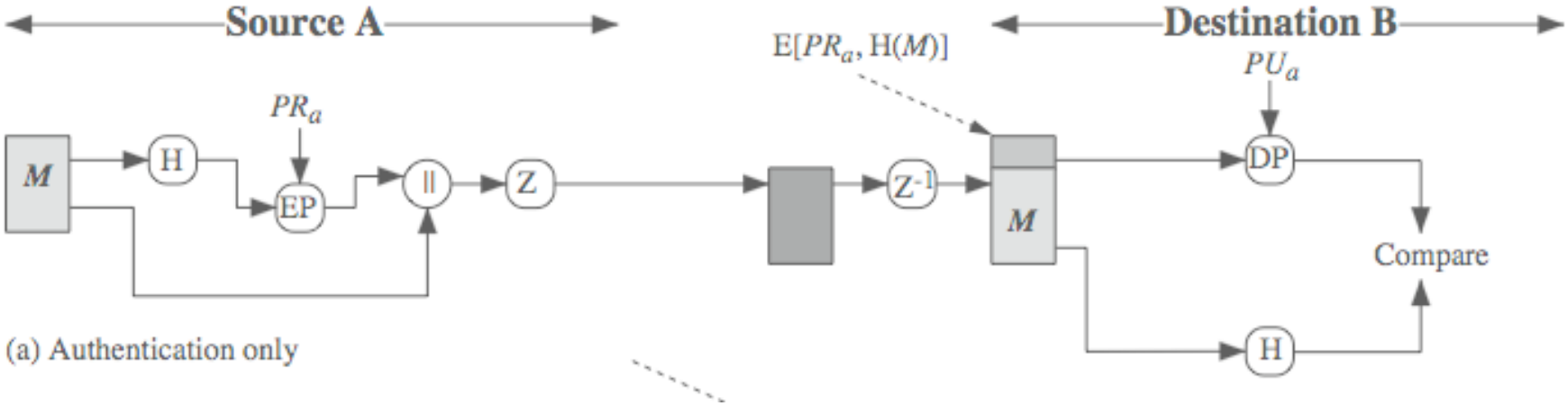
- Pretty Good Privacy is a standard created by Phil Zimmermann in 1991
  - "PGP empowers people to take their privacy into their own hands. There has been a growing social need for it. That's why I wrote it." See Why I wrote PGP https://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html
- The slides on PGP are inspired to the well-known textbook Cryptography and Network Security, 5/e, by William Stallings, Chapter 18 – "Electronic Mail Security"

PRETTY GOOD PRIVACY (PGP)

- well known and widely used since the 90s
- using best available crypto algorithms
- integrated into a single program
  - Linux/Unix, PC, Macintosh and other systems
- originally free, now owned by Symantec (www.pgp.com)
- open version (OpenPGP) standardized in RCF 4880
  - several implementations, e.g., Gnu Privacy Guard (www.gnupg.org)

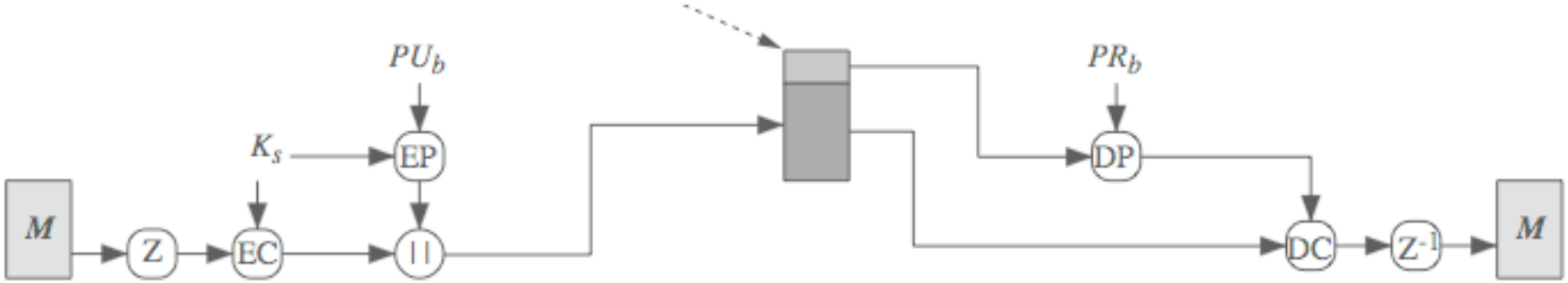
PGP AUTHENTICATION

1. sender creates message
2. make SHA-1 160-bit hash of message
3. attached RSA signed hash to message
4. receiver decrypts & recovers hash code
5. receiver verifies received message hash



PGP CONFIDENTIALITY

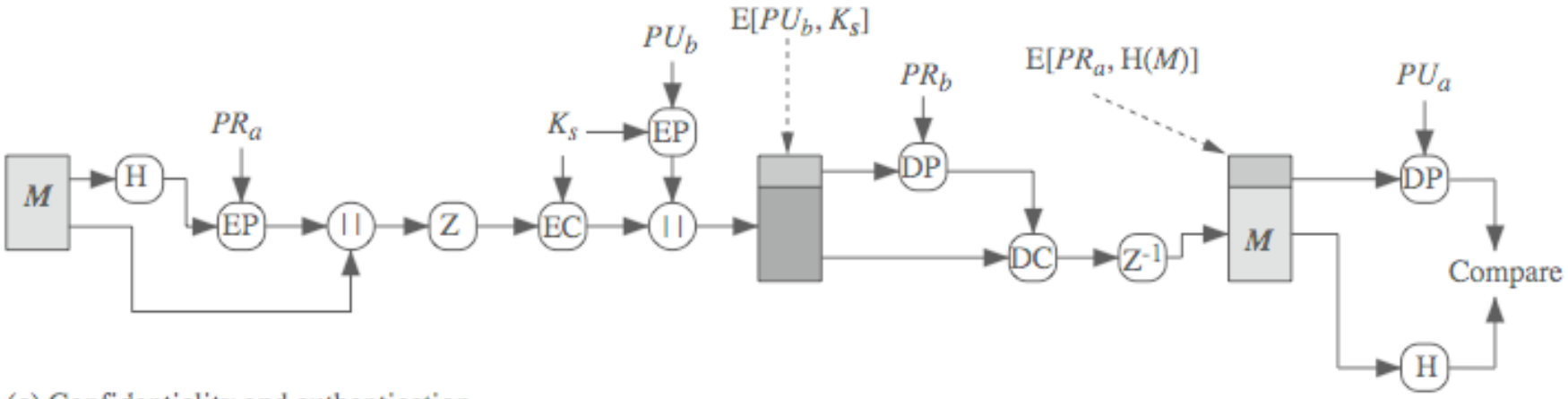
- sender forms 128-bit random session key
- encrypts message with session key
- attaches session key encrypted with RSA
- receiver decrypts & recovers session key
- session key is used to decrypt message



(b) Confidentiality only

CONFIDENTIALITY & AUTHENTICATION

- can use both services on same message
  - create signature & attach to message
  - encrypt both message & signature
  - attach RSA/ElGamal encrypted session key



(c) Confidentiality and authentication

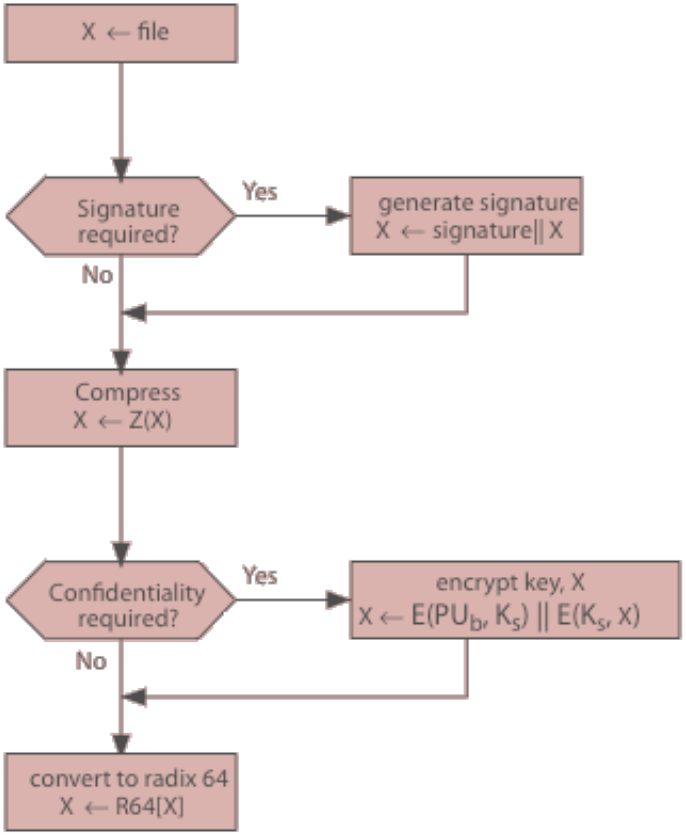
COMPRESSION

- by default PGP compresses message after signing but before encrypting
  - so can store uncompressed message & signature for later verification
  - because compression is non deterministic (if verification requires compression it may fail on legitimate messages)
- uses ZIP compression algorithm

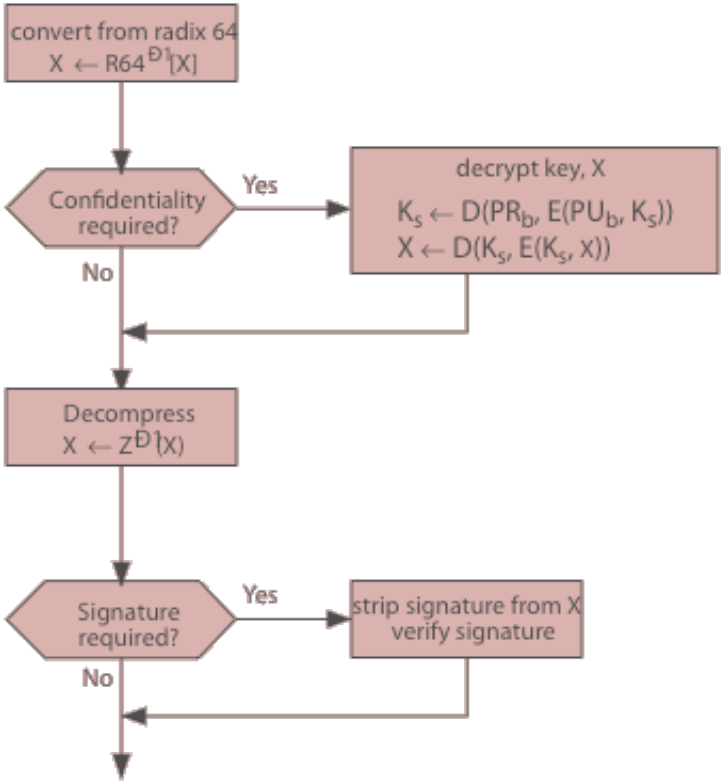
EMAIL COMPATIBILITY

- when using PGP will have binary data to send (encrypted message etc.)
- however email was designed only for text
- hence PGP must encode raw binary data into printable ASCII characters
  - uses Base-64 encoding and appends a 24-bit CRC
- PGP also segments messages if too big

PGP OPERATION – SUMMARY



(a) Generic Transmission Diagram (from A)



(b) Generic Reception Diagram (to B)

RFC 4880 - PUBLIC KEY

9.1. Public-Key Algorithms

ID	Algorithm
1	- RSA (Encrypt or Sign) [HAC]
2	- RSA Encrypt-Only [HAC]
3	- RSA Sign-Only [HAC]
16	- Elgamal (Encrypt-Only) [ELGAMAL] [HAC]
17	- DSA (Digital Signature Algorithm) [FIPS186] [HAC]
18	- Reserved for Elliptic Curve
19	- Reserved for ECDSA
20	- Reserved (formerly Elgamal Encrypt or Sign)
21	- Reserved for Diffie-Hellman (X9.42, as defined for IETF-S/MIME)
100 to 110	- Private/Experimental algorithm

Implementations MUST implement DSA for signatures, and Elgamal for encryption. Implementations SHOULD implement RSA keys (1). RSA Encrypt-Only (2) and RSA Sign-Only are deprecated and SHOULD NOT be generated, but may be interpreted. See Section 13.5. See Section 13.8 for notes on Elliptic Curve (18), ECDSA (19), Elgamal Encrypt or Sign (20), and X9.42 (21). Implementations MAY implement any other algorithm.

RFC 4880 - SYMMETRIC KEY

9.2. Symmetric-Key Algorithms

ID	Algorithm
0	Plaintext or unencrypted data
1	IDEA [IDEA]
2	TripleDES (DES-EDE, [SCHNEIER] [HAC] - 168 bit key derived from 192)
3	CAST5 (128 bit key, as per [RFC2144])
4	Blowfish (128 bit key, 16 rounds) [BLOWFISH]
5	Reserved
6	Reserved
7	AES with 128-bit key [AES]
8	AES with 192-bit key
9	AES with 256-bit key
10	Twofish with 256-bit key [TWOFISH]
100 to 110	Private/Experimental algorithm

Implementations MUST implement TripleDES. Implementations SHOULD implement AES-128 and CAST5. Implementations that interoperate with PGP 2.6 or earlier need to support IDEA, as that is the only symmetric cipher those versions use. Implementations MAY implement any other algorithm.

added support for Camelia in RFC 5581 (2009)

RFC 4880 - HASH FUNCTIONS

9.4. Hash Algorithms

ID	Algorithm	Text Name
1	- MD5 [HAC]	"MD5"
2	- SHA-1 [FIPS180]	"SHA1"
3	- RIPE-MD/160 [HAC]	"RIPEMD160"
4	- Reserved	
5	- Reserved	
6	- Reserved	
7	- Reserved	
8	- SHA256 [FIPS180]	"SHA256"
9	- SHA384 [FIPS180]	"SHA384"
10	- SHA512 [FIPS180]	"SHA512"
11	- SHA224 [FIPS180]	"SHA224"
100 to 110	- Private/Experimental algorithm	

Implementations MUST implement SHA-1. Implementations MAY implement other algorithms. MD5 is deprecated.

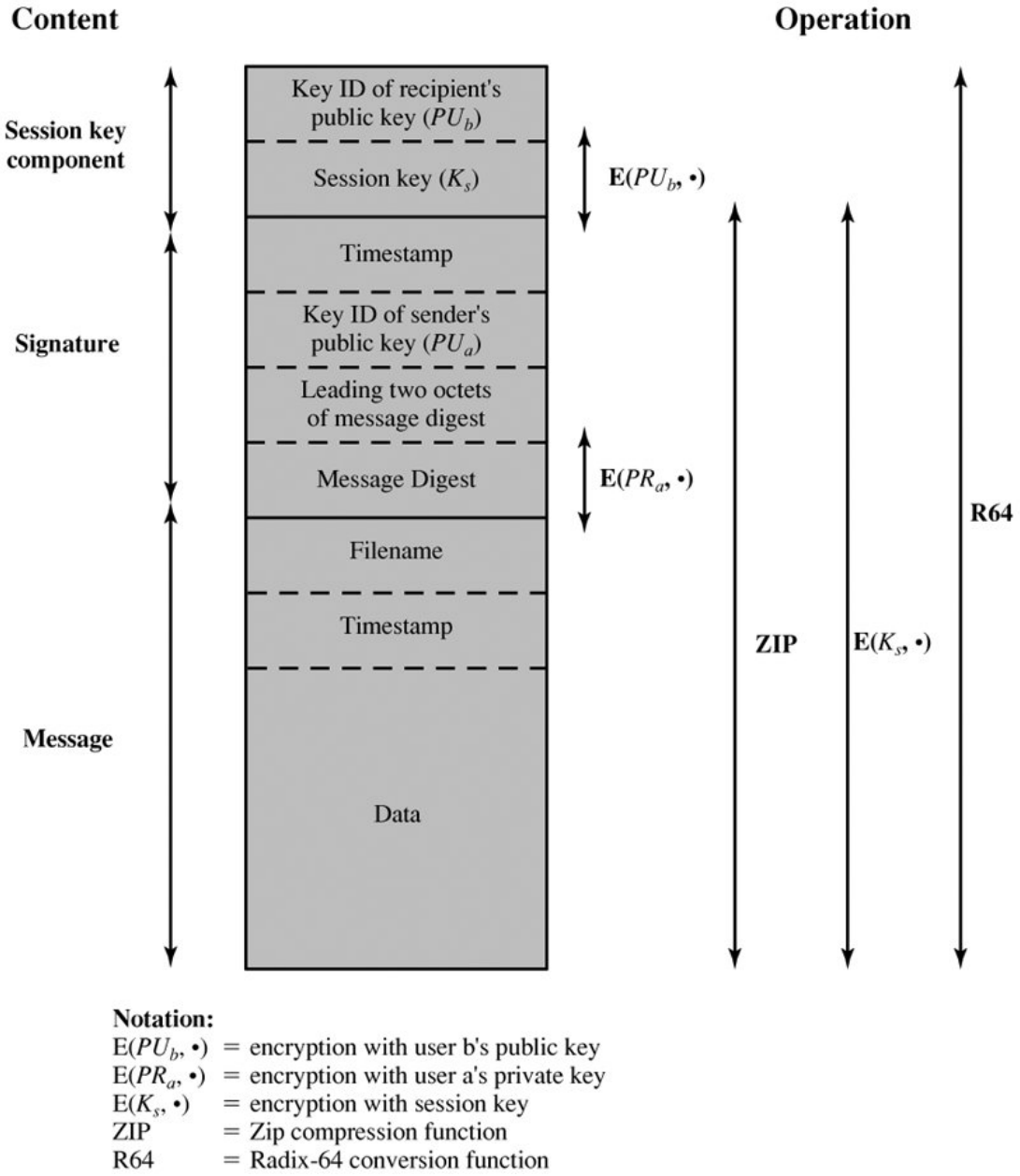
PGP SESSION KEYS

- PGP needs a session key for each message
- PGP maintains a 256-byte buffer of random bits. Each time PGP expects a keystroke, it records the time, in 32-bit format, at which it starts waiting. When it receives the keystroke, it records the time the key was pressed and the 8-bit value of the keystroke. The time and keystroke information are used to generate a key, which is, in turn, used to encrypt the current value of the random-bit buffer
- Pseudorandom number generation makes use of a 24-octet seed and produces a 16-octet session key, an 8-octet initialization vector, and a new seed to be used for the next pseudorandom number generation.
- The algorithm is based on the X9.17 algorithm but uses CAST-128 instead of triple DES for encryption

PGP PUBLIC & PRIVATE KEYS

- since many public/private keys may be in use (by one user), need to identify which is actually used to encrypt session key in a message
  - could send full public-key with every message
  - but this is inefficient
- rather use a key identifier (ID) based on key
  - is least significant 64-bits of the key
  - will very likely be unique
- also use key ID in signatures

PGP MESSAGE FORMAT



PGP KEY RINGS

- each PGP user has a pair of keyrings:
  - public-key ring contains all the public-keys of other PGP users known to this user, indexed by key ID
  - private-key ring contains the public/private key pair(s) for this user, indexed by key ID & encrypted keyed from a hashed passphrase
- security of private keys thus depends on the passphrase security

PGP KEY RINGS

Private Key Ring

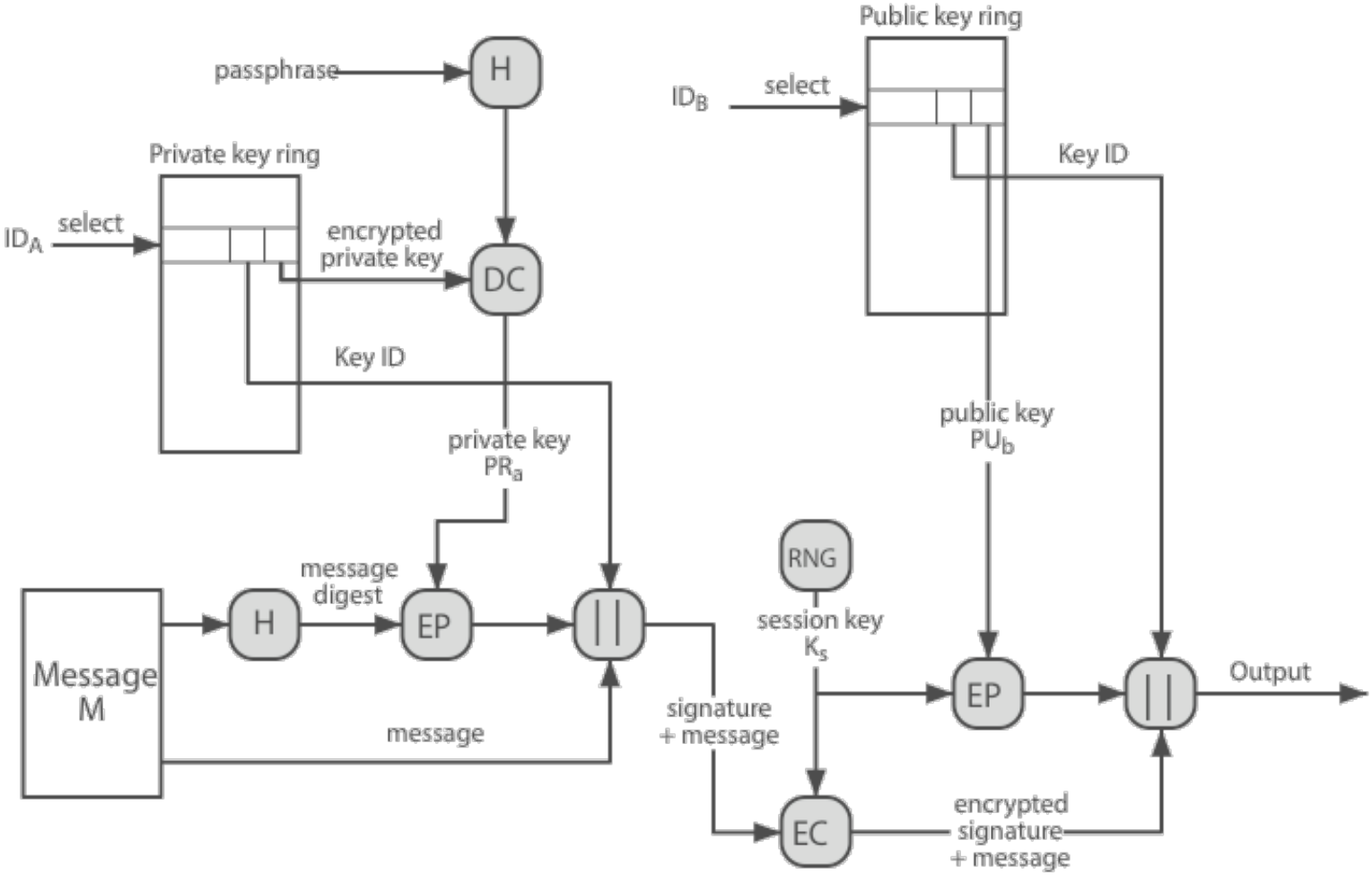
Timestamp	Key ID*	Public Key	Encrypted Private Key	User ID*
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•
Ti	PU_i mod 2^{64}	PU_i	E(H(P_i), PR_i)	User *i*
•	•	•	•	•
•	•	•	•	•
•	•	•	•	•

Public Key Ring

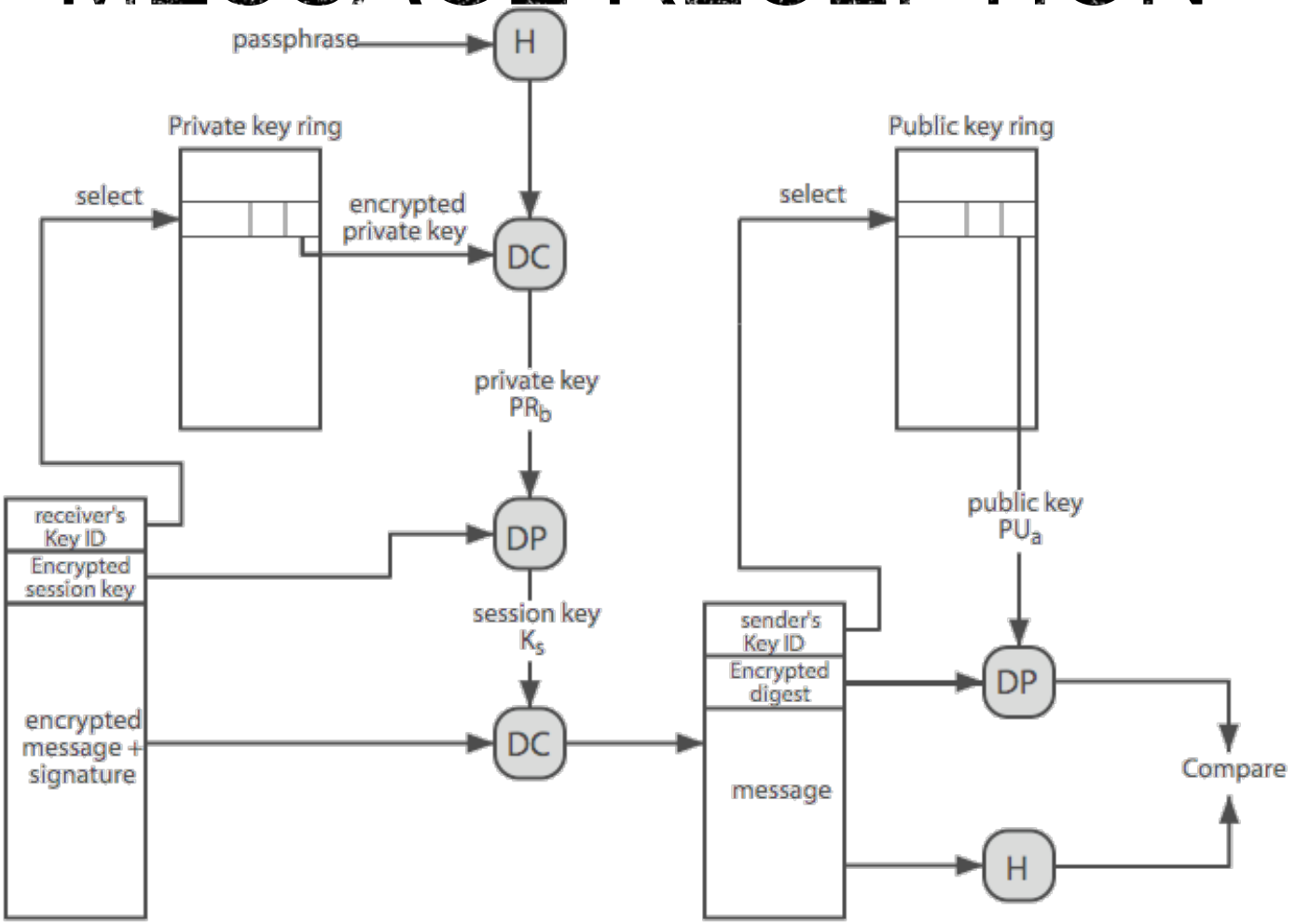
Timestamp	Key ID*	Public Key	Owner Trust	User ID*	Key Legitimacy	Signature(s)	Signature Trust(s)
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
Ti	PU_i mod 2^{64}	PU_i	$trust\_flag_i$	User *i*	$trust\_flag_i$		
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•
•	•	•	•	•	•	•	•

* = field used to index table

PGP MESSAGE GENERATION



PGP MESSAGE RECEPTION



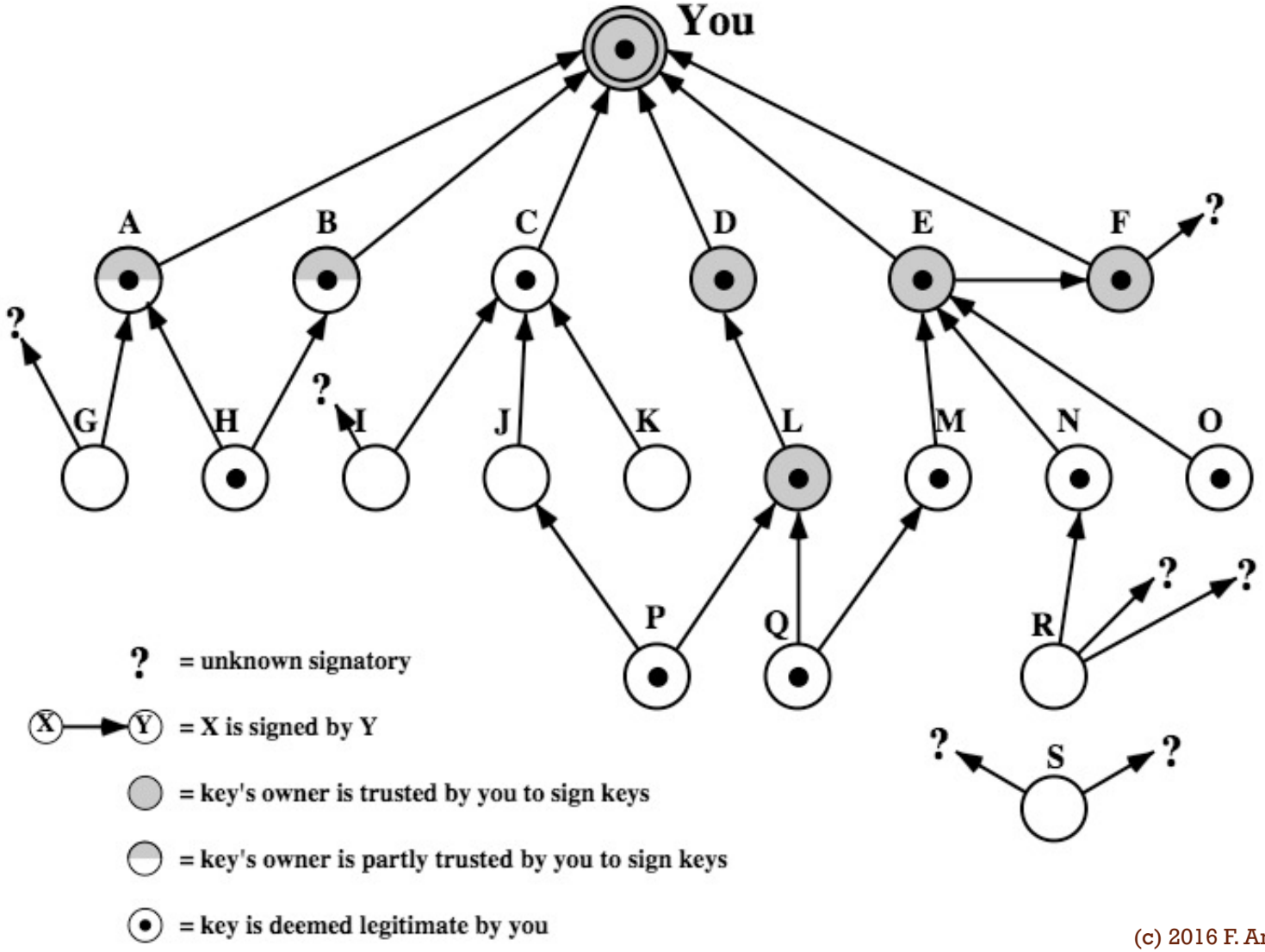
PGP KEY MANAGEMENT

- rather than relying on certificate authorities
- in PGP every user is own CA
  - can sign keys for users they know directly
- forms a "web of trust"
  - trust keys have signed
  - can trust keys others have signed if have a chain of signatures to them
- key ring includes trust indicators
- users can also revoke their keys
- a possible key-sign procedure http://herrons.com/keysigning-party-guide/

WEB OF TRUST (ZIMMERMANN)

As time goes on, you will accumulate keys from other people that you may want to designate as trusted introducers. Everyone else will each choose their own trusted introducers. And everyone will gradually accumulate and distribute with their key a collection of certifying signatures from other people, with the expectation that anyone receiving it will trust at least one or two of the signatures. This will cause the emergence of a decentralized fault-tolerant web of confidence for all public keys.

PGP TRUST MODEL EXAMPLE



PGP TODAY

- OpenPGP is an Internet standard (RFC 4880, 2007)
- many e-mail clients provide OpenPGP-compliant email security
- best known implementations of OpenPGP
  - PGP by Symantec Inc.
  - GNU Privacy Guard (GnuPG or GPG) by The Free Software Foundation. Open-source

ENIGMAIL EXAMPLE

- supported email clients
  - Mozilla Thunderbird
  - Mozilla SeaMonkey
  - Eudora 1.0 OSE
  - Postbox
- GnuPG Software
- EnigMail plugin (http://enigmail.mozdev.org)
  - language packs available

Gestione componenti aggiuntivi

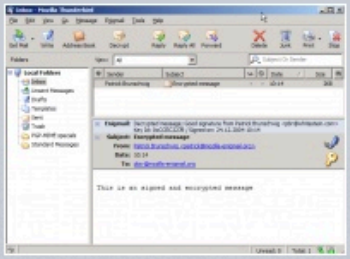
Posta in arrivo - DIS | Gestione componenti aggiuntivi

Cerca tra i componenti aggiuntivi

- Esplora
- Estensioni
- Aspetto
- Plugin

Enigmail 1.4.1

di Patrick Brunschwig



OpenPGP message encryption and authentication

Enigmail adds OpenPGP message encryption and authentication to your email client. It features automatic encryption, decryption and integrated key management functionality.
Enigmail requires GnuPG (www.gnupg.org) for the cryptographic functions. Note: GnuPG is not part of the installation.
The addon offered here supports Windows, Linux (32 and 64-bit) and Mac OS X. Versions for more platforms are available from the homepage.

Aggiornamento automatico	(•) Predefinito ( ) Attivo ( ) Disattivato
Ultimo aggiornamento	21 aprile 2012
Sito web	http://enigmail.mozdev.org/
Voto	★★★★☆ 133 recensioni

Preferenze | Disattiva | Rimuovi