I protocolli di VII livello

Fabio Vitali

Introduzione

Qui esaminiamo in breve i protocolli di VII livello, ed in particolare quelli basati sul testo e connessi con lo scambio di posta elettronica, per il loro impatto su HTTP.

- Cosa sono i protocolli a livello
- Il protocollo SMTP ed ESMTP
- Lo standard MIME
- Brevemente, i protocolli POP, IMAP e NNTP.

I protocolli a livello

La comunicazione tra computer avviene attraverso protocolli (regole) di comunicazione.

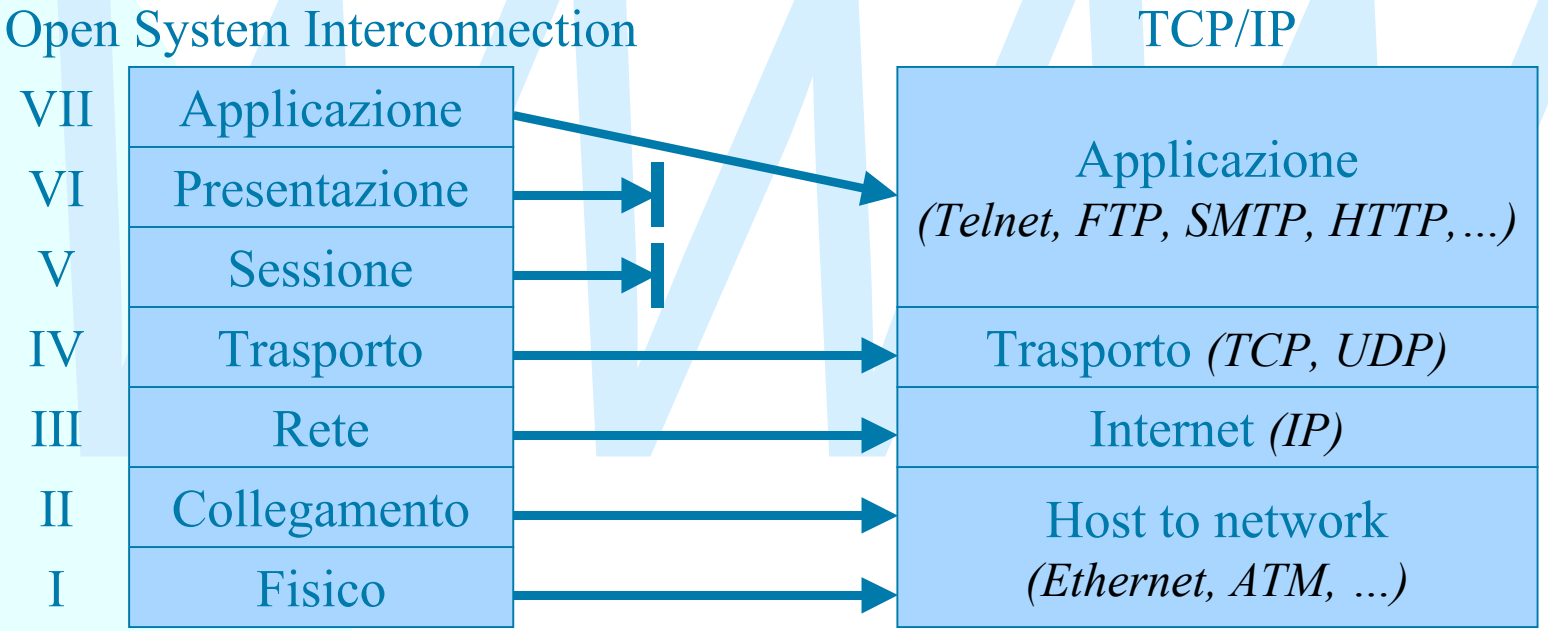
Questi sono tipicamente divisi in livelli, al fine di:

- Dividere le problematiche i tronconi affrontabili
- Incapsulare i requisiti in blocchi indipendenti
- Rendere soluzioni diverse per un livello interscambiabili

I livelli forniscono via via un'astrazione maggiore procedendo dal basso verso l'alto, dalla scelta del canale fisico al meccanismo di indirizzamento, alla creazione del meccanismo di trasporto più adeguato, alle applicazioni che richiedono la comunicazione

I modelli ISO-OSI e TCP/IP

Sono i due stack di protocolli più famosi, anche se OSI è rimasto sostanzialmente un modello su carta.



La nostra attenzione è sui protocolli di VII livello.

I protocolli di VII livello

- Al VII livello esistono i protocolli di applicazione, che svolgono un lavoro direttamente utile alle applicazioni utente
- Anche al VII livello dobbiamo distinguere tra
  - Protocolli di applicazione vera e propria: forniscono il servizio agli utenti finali (SMTP, NNTP, HTTP, telnet, FTP, ecc.)
  - Protocolli di servizio: forniscono servizi non direttamente agli utenti, ma alle applicazioni utente (SNMP, DNS, ecc.)
- Ovviamente i protocolli di servizio non costituiscono un livello a sé, poiché non sono frapposti tra il protocollo di applicazione e il protocollo di trasporto.
- Noi ci occupiamo di SMTP e di alcuni protocolli connessi, per la loro importanza rispetto ad HTTP

End-to-End argument (1)

L'End-to-end argument è uno dei principi progettuali più importanti di Internet, responsabile della scalabilità e flessibilità di Internet negli ultimi vent'anni.Venne espresso per la prima volta nel 1981 da Saltzer, Reed e Clark.

Quando si implementano protocolli a livelli, una particolare attenzione va rivolta alla decisione di quali funzionalità mettere in quale livello, e come possono i livelli inferiori aiutare i livelli superiori nello svolgimento dei loro compiti.

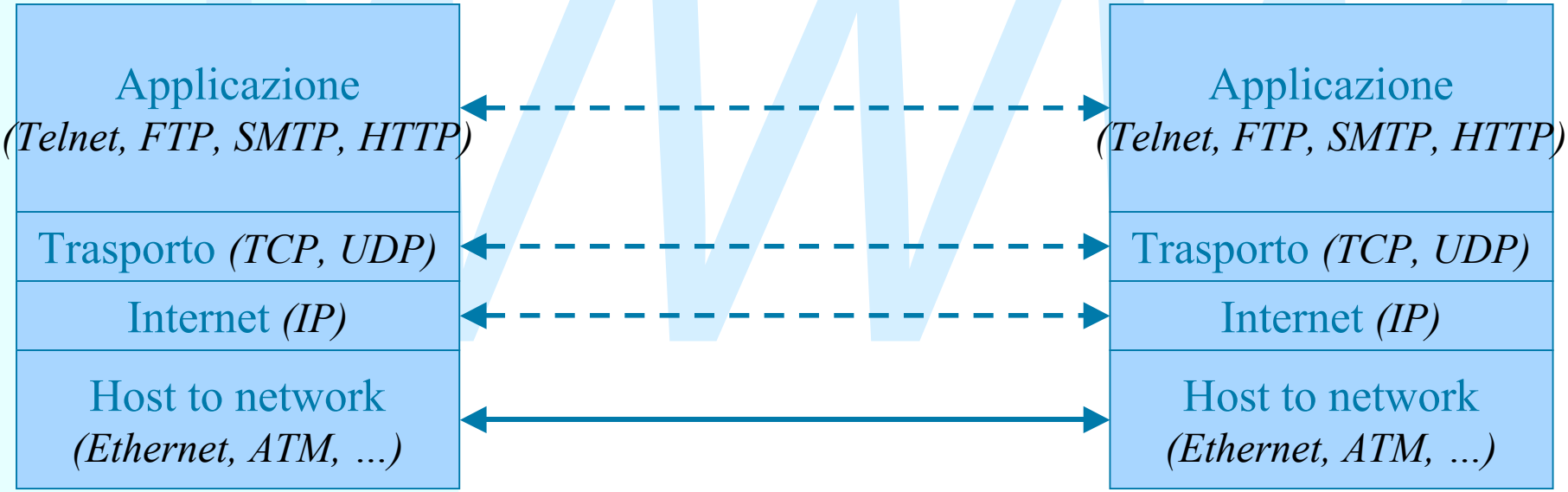
Il principio dell'end-to-end argument è che i livelli inferiori non possono e non debbono fornire funzionalità di livello applicazione, perché dovrebbero avere conoscenza di quali siano queste applicazioni e di come funzionino.

Questo ha degli effetti su quello che viene chiamata la sotto-rete di comunicazione.

A seguire: End-to-End argument (2)

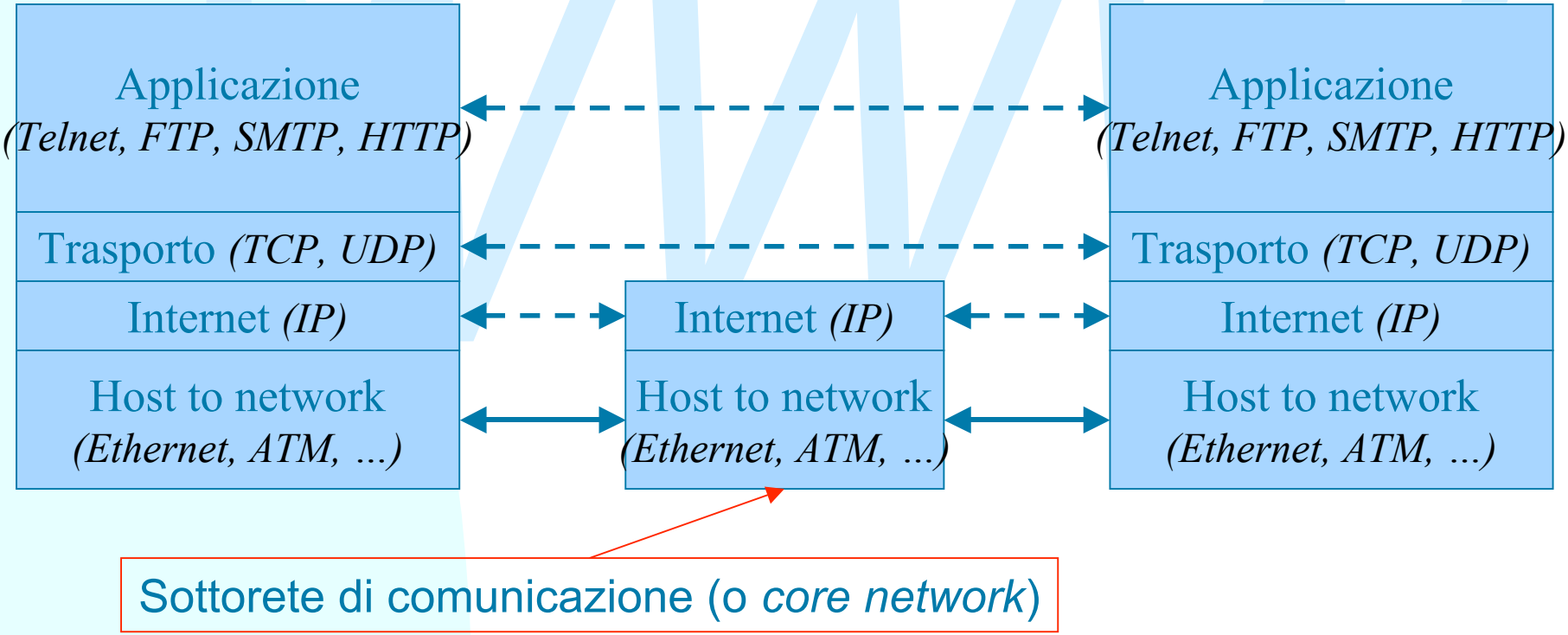
End-to-End argument (2)

Lo schema di comunicazione tra livelli è spesso disegnato così:



End-to-End argument (3)

Ma in realtà lo schema di comunicazione tra livelli è così:



A seguire: End-to-End argument (4)

End-to-End argument (4)

Un altro modo per esprimere l'end-to-end argument è di dire che la sottorete di comunicazione (ovvero tutti i protocolli e macchinari compresi tra i due host finali) non possono e non debbono occuparsi dei dettagli dell'applicazione, ma soltanto del completamento con successo (ed efficienza) della trasmissione.

Tra i vantaggi:

- La complessità del core network è drasticamente ridotta
- La generalità della rete aumenta la possibilità che un nuovo servizio venga creato senza modifiche al core network
- Le applicazioni non debbono dipendere da cattive implementazioni o limiti realizzativi di servizi intermedi su cui non hanno controllo, con ciò migliorando la loro affidabilità.

Questa semplicità e generalità ha grandemente giustificato l'evoluzione e il successo di Internet negli anni passati.

End-to-End argument (5)

Problemi con l'end-to-end argument:

- Applicazioni più esigenti:
  - l'end-to-end necessariamente implica il *best-effort service*, in cui la qualità di servizio di ogni applicazione dipende dal traffico esistente, indipendentemente dalla sua importanza. Scaricare MP3 può rallentare la trasmissione video di un operazione chirurgica in remoto.
- Differenziazione nei servizi degli ISP
  - Gli ISP attualmente competono su aspetti di commodity nella fornitura di rete: **prezzo** e **velocità**; questi sono ambiti di competizione infami e senza gloria, in cui sopravvivono i più spietati e potenti.
  - Piacerebbe invece agli ISP poter fornire abbonamenti collegati all'elenco di servizi disponibili, e non di prezzo o velocità. Ma questo implica inibire a livello di host certi servizi per cui uno non è abbonato.
  - Il caso di America On-line (e Microsoft Network)
  - Il caso dei cellulari GSM e GPRS

A seguire: End-to-End argument (6)

End-to-End argument (6)

Problemi con l'end-to-end argument:

- Lavorare in un mondo non fidato
  - Caratteristica dell'end-to-end è che i due host finali collaborano volontariamente al buon fine della conversazione, senza che il core network faccia niente per filtro o controllo
  - Questo ha portato a spam, virus, attacchi di hacker, siti web pirata, ecc., che assumono la neutralità del core network e approfittano dell'inadeguatezza dell'altro estremo per proprio vantaggio
- Utenti non esperti
  - L'end-to-end argument lascia completamente all'utente finale il compito di configurare correttamente i servizi. Con la crescita del numero di utenti e l'abbassamento del loro livello di conoscenze tecniche, questo può portare a problemi di inutilizzo dei servizi stessi.
  - Software e servizi configurati centralmente invece permetterebbero al fornitore di servizi di provvedere sempre indipendentemente dalle capacità tecniche dell'utente.

A seguire: End-to-End argument (7)

End-to-End argument (7)

Problemi con l'end-to-end argument:

- Coinvolgimento di terze parti
  - C'è una crescente esigenza da parte di terze parti di intervenire nello scambio di dati tra i due estremi:
    - **Censura**: Organizzazioni con i loro dipendenti, ISP con i loro abbonati, genitori con i loro figli, stati dittatoriali con i loro cittadini, vogliono poter controllare che tipo di dati vengono scambiati e poter bloccare in anticipo gli scambi sgraditi.
    - **Indagini di polizia**: anche gli stati democratici possono voler controllare (magari a posteriori) certe comunicazioni identificate e specificate da un giudice, se questo serve a portare a termine indagini di polizia. Non parliamo poi dei servizi segreti.
    - **Tassazione**: i governi o i proprietari di diritti tipicamente vogliono una fetta di guadagno da ogni transazione commerciale, in particolare se internazionale. Sono dunque interessati a scambi di dati che comportano trasferimenti di valore e a stabilire esattamente quali valori sono stati scambiati e come ottenerne una parte in tassazione.
  - Ovviamente tutte queste situazioni richiedono che lo scambio di dati venga monitorato e, sulla base del contenuto dello stesso, bloccato o registrato, possibilmente a livello di infrastruttura (core network)

A seguire: End-to-End argument (8)

End-to-End argument (8)

Possibili risposte

- Il ruolo delle leggi
  - E' possibile intervenire a livello di leggi, imponendo tasse e divieti e controlli.
  - Però la legge interviene sono a posteriori, e in molti casi solo se ne vale individualmente la pena.
  - Inoltre ha un effetto limitato, soprattutto sulle relazioni internazionali (il cattivo di turno sposta semplicemente il suo end-point in uno stato in cui questi controlli non ci sono).
- Modifica degli end-point
  - Si impone ai costruttori di hardware e software (ad esempio ai fabbricanti di browser) di creare nuovi prodotti in cui i meccanismi di filtro e controllo sono built-in
  - Problema: non si applica ai vecchi prodotti (bisogna fornire un fattore motivante per passare al nuovo), e fa nascere un mercato parallelo di prodotti che non ne tengono conto.
  - Altro problema: complica la realizzazione di nuovi software, il che riduce il numero e la flessibilità dei nuovi servizi.

A seguire: End-to-End argument (9)

End-to-End argument (9)

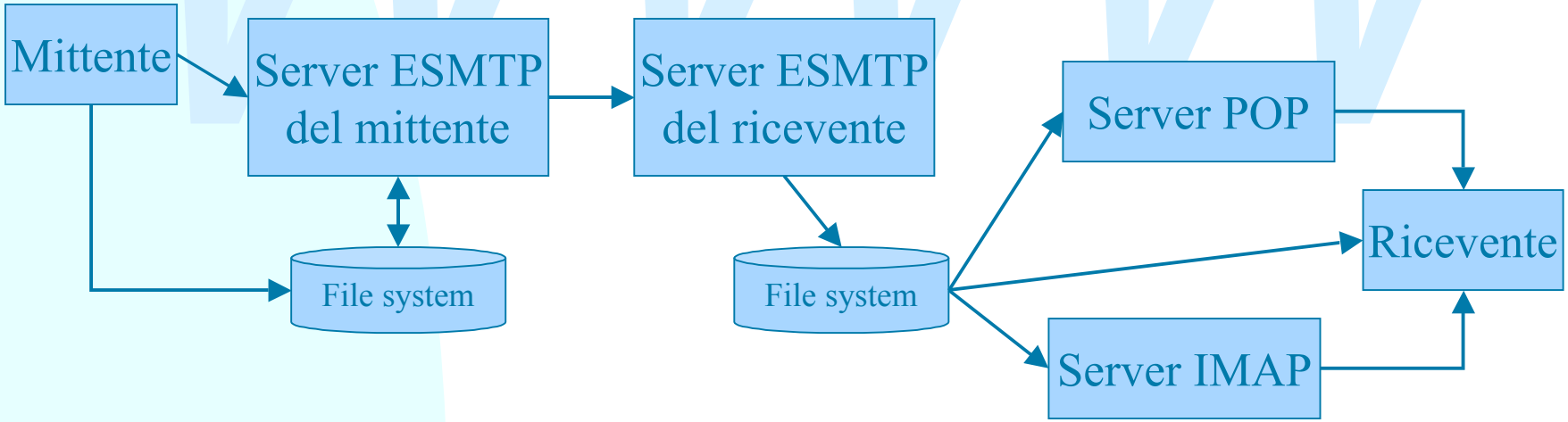
Possibili risposte

- Aggiungere funzioni al core network in sottoreti ad hoc
  - Di fatto, oggi firewall, filtri sul traffico e NAT sono modificatori del core network che permettono di effettuare controlli e blocchi a priori. Creano delle sottoreti (ad esempio, Intranet aziendali) all'interno delle quali l'end-to-end argument non vale.
  - Nuovi servizi vengono esaminati caso per caso e approvati solo se rispondono ai criteri globali di sicurezza/controllo/censura stabiliti. Ovviamente questo comporta una riduzioone della qualità e della flessibilità della sottorete, ma non incide sulla rete globale.
- Realizzare domini di servizio
  - Terze parti fidate per entrambi gli estremi vengono utilizzate come intermediari per lo scambio di messaggi, fornendo quei meccanismi di controllo che si reputa necessari.
  - La comunicazione non avviene più in maniera diretta, ma mediata dal nodo intermedio, con cui si viene a creare un sottodominio spesso indipendente dalla configurazione fisica della rete
  - L'adesione può essere volontaria (etichette PICS per i contenuti Web, anonimizzatori, ecc.), costretta dall'ISP (Server SMTP) o dalle leggi, ma non richiede modifiche al meccanismo fondamentale della rete.

La posta elettronica (1)

La posta elettronica è basata sull'applicazione di 3 protocolli di VII livello:

- SMTP/ESMTP *(host-to-host, client-to-host)*
- POP *(host-to-client)*
- IMAP *(host-to-client)*



A seguire: La posta elettronica (2)

La posta elettronica (2)

SMTP è descritto in due documenti IETF

- RFC 821: il modello di comunicazione, i comandi SMTP, i codici d'errore
- RFC 822: Il formato dei messaggi, il formato degli indirizzi, il formato delle date

MIME è descritto da cinque documenti IETF:

- RFC 2045, 2046, 2047, 2048, 2049

ESMTP è descritto da vari documenti IETF:

- RFC 1869: un meccanismo generale di estensione di SMTP
- RFC 1652, 1870, 1830, 2197, 1891, 1985, 2034, 2487: varie estensioni ad SMTP

SMTP è stato aggiornato nell'aprile 2001:

- RFC 2821 e 2822 aggiornano risp. RFC 821 e RFC 822

Altri protocolli:

- RFC 1939: descrive Post Office Protocol version 3 (POP3)
- RFC 2060: descrive Internet Message Access Protocol (IMAP4)

SMTP

Simple Mail Transfer Protocol

SMTP è un protocollo text-based, per lo scambio di messaggi di posta e la verifica dei destinatari dei messaggi.

Una connessione SMTP è composta da una apertura, uno o più sequenze di comandi, ed una chiusura.

Ad ogni comando corrisponde una risposta composta da un codice numerico ed una stringa leggibile.

- `MAIL FROM:<Smith@alpha.com>`
  `250 OK`
- `RCPT TO:<Green@beta.com>`
  `550 No such user here`

SMTP - RFC 821 (1)

L’apertura avviene con il comando HELO

- 220 alpha.com Simple Mail Transfer Service Ready
  HELO beta.com
  250 alpha.com says: Nice to meet you beta.com

La chiusura avviene con il comando QUIT

- QUIT
  221 alpha.com Service closing transmission channel

SMTP - RFC 821 (2)

La spedizione di un messaggio avviene attraverso l'identificazione del mittente (`MAIL FROM`), del/dei destinatari (`RCPT TO`), e del messaggio da trasmettere (`DATA`)

- `MAIL FROM:<Smith@alpha.com>`
  `250 OK`
- `RCPT TO:<Green@beta.com>`
  `550 No such user here`
- `RCPT TO:<Brown@beta.com>`
  `250 OK`
- `DATA`
  `354 Start mail input; end with <CRLF>.<CRLF>`
- `Blah blah blah…`
  `etc. etc. etc.`
  `.`
  `250 OK`

SMTP - RFC 821 (3)

Il forwarding avviene quando il destinatario non è corretto ma il server ricevente conosce l'indirizzo corretto:

- `RCPT TO:<Green@beta.com>`
  `250 OK` **(forward silenzioso)**
- `RCPT TO:<Green@beta.com>`
  `251 User not local; will forward to <Green@gamma.com>`
- `RCPT TO:<Green@beta.com>`
  `551 User not local; please try <Green@gamma.com>`

La verifica e l'espansione permettono di cercare un destinatario o di espandere il contenuto di un destinatario multiplo. Entrambi sono disabilitabili per sicurezza.

- `VRFY Smith`
  `250 Fred Smith <smith@beta.com>`
- `EXPN MailList`
  `250 Fred Smith <smith@beta.com>`
  `250 John Green <green@beta.com>`

A seguire: SMTP - RFC 821 (4)

SMTP - RFC 821 (4)

I codici di risposta

- Ogni comando di SMTP prevede un codice di risposta, della forma di un numero di tre cifre (utile per le applicazioni) e una stringa leggibile (utile per gli umani).
- Risposte su più linee debbono ripetere su ciascuna linea il codice numerico.
- I codici sono organizzati per categorie, a seconda della prima cifra:
  - 1xx: Risposta preliminaria positiva (nessun codice)
  - 2xx: Risposta di completamento positivo
  - 3xx: Risposta positiva parziale
    (il server si aspetta altri comandi successivi)
  - 4xx: Risposta di completamento negativo transiente
    (il server non ha accettato il comando per un problema non definitivo; es.: Mailbox busy)
  - 5xx: Risposta di completamento negativo definitivo
    (il server non ha accettato il comando e non lo accetterà mai in questa forma)

A seguire: SMTP - RFC 822 (1)

SMTP - RFC 822 (1)

I messaggi trasmessi su un canale SMTP sono composti da intestazione e corpo, separati da una riga vuota:

- `message = fields (CRLF text*)*`

L'intestazione è composta da campi posti su righe autonome. Ogni campo ha la sintassi

- `<nome_del_campo>": "<valore_del_campo> CRLF`

Il corpo è composto da qualunque sequenza di caratteri ASCII con l'eccezione della stringa CRLF-punto-CRLF, e con i seguenti limiti:

- La lunghezza massima del messaggio è di 1 Mb
- I caratteri accettati sono solo i caratteri ASCII a 7 bit
- Ogni messaggio deve contenere una sequenza CRLF ogni 1000 caratteri o meno (cioè deve essere diviso in righe di massimo 998 caratteri).

A seguire: SMTP - RFC 822 (2)

SMTP - RFC 822 (2)

L'intestazione è composta da date, origine, destinatari, ecc.:

- fields = source
  date
  destination +
  optional *

L'origine precisa il mittente, dati di risposta, e dati di traccia:

- source = "From: " mailbox CRLF
  ["Reply-To: " mailbox CRLF]
  ["Return-path: " mailbox CRLF]
  received *
- received = "Received: "
  ["from " sendingDomain]
  ["by " receivingDomain]
  ["with " linkOrMailProtocol]*
  ["id " receiverMsgId]
  … CRLF

SMTP - RFC 822 (2)

La data è quella di spedizione:

- `date =  "Date: " date-time CRLF`

La destinazione contiene uno o più destinatari principali, uno o più in carbon copy, ed uno o più in blind carbon copy:

- `destination =  "To: " mailbox + CRLF`
  `["Cc: " mailbox + CRLF]`
  `["Bcc: " mailbox+ CRLF]`

I campi opzionali contengono informazioni non necessarie per il buon successo della trasmissione:

- `optional =  ["Message-Id: " msg-Id CRLF]`
  `["In-Reply-To: " msg-id * CRLF]`
  `["Subject: " text* CRLF]`
  `["References: " msg-id * CRLF]`
  `…`

A seguire: SMTP - RFC 822 (3)

SMTP - RFC 822 (3)

Le date hanno il formato:

- `date-time = [day ",”] date time`
- `date = dd mmm yyyy`
- `time = hh ":" mm ":" ss (zzz | ("+"/"-") hhmm))`

Gli indirizzi hanno il formato:

- `mailbox = address | word+ "<" address ">`
- `address = local "@" domain`
- `local = word ("." word)*`
- `domain = subdomain ("." subdomain)*`
- `subdomain = word`

Limiti di SMTP

Ricapitoliamo i limiti fondamentali di SMTP:

- La lunghezza massima del messaggio è di 1 Mb
- I caratteri accettati sono solo ASCII a 7 bit
- Ogni messaggio deve contenere una sequenza CRLF ogni 1000 caratteri o meno (alcune antiche implementazioni lo aggiungevano automaticamente se non lo trovavano).

Questi limiti impediscono la trasmissione di documenti binari:

- Un file binario usa tutti i 256 tipi di byte
- Un file binario può facilmente essere più lungo di 1 Mb
- In un file binario la sequenza CRLF è una sequenza come tutte le altre, e può esserci o mancare senza vincoli. Introdurla artificialmente può corrompere il file.

MIME permette di bypassare questi limiti all'interno di SMTP

MIME (1)

Multipurpose Internet Mail Extensions

RFC 822 definisce con sufficiente dettaglio il formato degli header dei messaggi SMTP, ma specifica in modo molto generico che il corpo di un messaggio deve essere semplice testo US-ASCII.

MIME ridefinisce il formato del corpo di RFC 822 per permettere:

- Messaggi di testo in altri set di caratteri al posto di US-ASCII
- Un insieme estensibile di formati per messaggi non testuali
- Messaggi multi-parte
- Header con set di caratteri diversi da US-ASCII.

A seguire: MIME (2)

MIME (2)

Gli RFC su MIME sono divisi come segue:

- RFC 2045 specifica gli header SMTP per messaggi MIME
- RFC 2046 definisce il meccanismo di tipi di MIME
- RFC 2047 definisce estensioni a RFC 822 per header che non usano US-ASCII
- RFC 2048 definisce le procedure di registrazione a IANA per i tipi MIME e le altre caratteristiche estensibili di MIME
- RFC 2049 definisce i livelli di conformità e fornisce esempi di uso di formati MIME.

Noi guardiamo soltanto gli RFC 2045 e 2046.

MIME - RFC 2045 (1)

MIME introduce alcuni nuovi header SMTP:

- **Content-Type**: il tipo MIME del contenuto. Serve per permettere al ricevente di scegliere il meccanismo più adatto per presentare i dati. Specifica la natura del dato tramite la specificazione di tipo, sottotipo e ulteriori parametri utili.
  - `Content-Type: text/plain; charset=ISO-8859-1`
- **Content-Transfer-Encoding**: il tipo di codifica utilizzata per trasmettere i dati. Serve per la trasmissione su canale SMTP di dati che non sono naturalmente corretti secondo le regole di SMTP: 7bit, sequenze CRLF ogni 1000 caratteri o meno. Sono valori accettabili “7bit” (default), “8bit”, “binary”, “quoted-printable”, “base64” o altre stringhe definite nel registro IANA
  - Content-Transfer-Encoding: base64

A seguire: MIME - RFC 2045 (2)

MIME - RFC 2045 (2)

- **MIME-Version**: la versione di MIME attualmente utilizzata. L'unico valore accettabile attualmente è 1.0
  - `MIME-Version: 1.0`
- **Content-ID**: un meccanismo per permettere a più messaggi SMTP di far riferimento gli uni agli altri. Questo soprattutto è usato per entità esterne ed alternative dello stesso messaggio.
  - `Content-ID: stringa identificativa unica`
- **Content-Description**: utile per associare informazioni descrittive ad un blocco binario
  - `Content-Description: Immagine di Marte`
- Sono possibili altri header MIME purché inizino con il prefisso "Content-".

MIME - Quoted printable

Uno dei due tipi di content transfer encoding definiti da MIME.

Viene usata per la trasmissione di dati che contengono grosse quantità di byte nel set US-ASCII, e solo poche eccezioni

- Ad esempio, documenti testuali in lingue europee.

Codifica dunque solo quei pochi byte non conformi. Per esempio:

- Un codice superiore al 127 o inferiore al 32 viene codificato con la sintassi “=“ + codice esadecimale. Ad esempio “ICSE’99” diventa “ICSE=B499”
- Righe più lunghe di 76 caratteri vengono interrotte con “soft breaks”, cioè con un uguale come ultimo carattere della linea.

MIME - Base 64

Base 64 è il tipo di transfer encoding MIME suggerito per dati binari.

Viene identificato un sottoinsieme di 64 caratteri di US-ASCII sicuri (hanno la stessa codifica in tutte le versioni di ISO 646). Questi sono le lettere maiuscole (26), minuscole (26), i numeri (10) più i caratteri + e /.

Ogni flusso di dati viene suddiviso in blocchi di 24 bit (3 byte).

A loro volta questi 24 bit sono suddivisi in 4 blocchi di 6 bit ciascuno e codificati secondo una tabella prefissata in uno dei 64 caratteri già descritti.

La stringa risultante viene divisa in righe di 76 caratteri (tranne l'ultima, che è lunga quanto deve essere).

I codici CR e LF sono da ignorare nella decodifica.

MIME - RFC 2046 (1)

- MIME introduce il concetto di Content-Type per
  - permettere all'applicazione ricevente di identificare il modo migliore di presentare le informazioni ricevute
  - Permettere all'applicazione di dividere, riunire o ottenere parti di messaggio unite, divise o non trasmesse.
- MIME specifica il tipo con una coppia tipo/sottotipo più parametri opzionali.
  - I tipi principali sono
    - `text`
    - `audio`
    - `application`
    - `multipart`
    - `image`
    - `video`
    - `message`
- Tipi ulteriori possono essere introdotti registrandoli presso lo IANA o prefissandoli con "x-" (per *experimental*)

MIME - RFC 2046 (2)

`message`: un corpo di tipo “message” è esso stesso un messaggio completo incapsulato (con intestazioni ecc.) che può a sua volta contenere altri messaggi, ecc.

- Il sottotipo “rfc822” permette di specificare che il messaggio è esso stesso un messaggio del tipo definito in RFC 822.
- Il sottotipo “partial” permette di frammentare messaggi troppo lunghi per passare indenni in un canale SMTP. Sono definiti parametri per identificare i frammenti e riordinarli correttamente.
- Il sottotipo “external-body” permette di specificare un corpo di grandi dimensioni attraverso un puntatore ad una fonte di dati esterna. Sono definiti parametri per specificare metodo di accesso e identificatore della risorsa.

A seguire: MIME - RFC 2046 (3)

MIME - RFC 2046 (3)

`multipart`: un corpo di tipo “multipart” contiene nel corpo blocchi di dati di tipo diverso. Ogni blocco viene preceduto da una riga di delimitazione (*boundary line*), da righe di intestazione simili a quelle di RFC 822, per definire le caratteristiche specifiche del blocco, e viene seguito da un’altra boundary line.

- Il sottotipo “mixed” serve per segnalare che le parti sono indipendenti e di tipi diversi.
- Il sottotipo “alternative” serve per segnalare che le parti sono di tipi diversi ma identiche per contenuto, e che quindi l’applicazione finale può scegliere la versione che preferisce.
- Il sottotipo “parallel” serve per segnalare che le parti vanno mostrate contemporaneamente dall’applicazione finale (ad es. suono e video)
- Il sottotipo “digest” permette di precisare collezioni di testi sotto forma di digest (RFC 934).

A seguire: ESMTP - RFC 1869 (1)

ESMTP - RFC 1869 (1)

- SMTP è uno dei protocolli più robusti e utilizzati su Internet. Tuttavia è possibile che vi sia la necessità di estendere le sue capacità.
- E' da ricordare che l'estensione di SMTP va fatta con molta cautela. L'esperienza insegna che i protocolli con poche opzioni tendono all'ubiquità, quelli con troppe opzioni tendono all'oscurità.
- ESMTP non è un'estensione di SMTP, ma un meccanismo per realizzare estensioni. Esso include:
  - Un nuovo comando SMTP (`EHLO`)
  - Parametri addizionali per i comandi SMTP
  - Un registro di estensioni "ufficiali" ad SMTP.

A seguire: ESMTP - RFC 1869 (2)

ESMTP - RFC 1869 (2)

Il comando EHLO va usato invece del comando HELO, per indicare che si usa il protocollo ESMTP invece che SMTP.

I caso: il server **non** supporta ESMTP:

- ```
  220 beta.com SMTP service ready
  EHLO alpha.com
  500 Command not recognized: EHLO
  HELO alpha.com
  250 beta.com says hello!
  …
  ```

II caso: il server supporta ESMTP ma non ha estensioni:

- ```
  220 beta.com SMTP service ready
  EHLO alpha.com
  250 beta.com says hello!
  …
  ```

A seguire: ESMTP - RFC 1869 (3)

ESMTP - RFC 1869 (3)

III caso: il server supporta ESMTP ed ha alcune estensioni:

- ```
  220 beta.com SMTP service ready
  EHLO alpha.com
  250-beta.com says hello!
  250-EXPN
  250-HELP
  250-8BITMIME
  250-XONE
  250 XVRB
  ...
  ```

In questo caso, il server supporta i comandi opzionali EXPN e HELP, l'estensione ufficiale 8BITMIME e due estensioni non standard e non registrate, XONE e XVRB.
  ```

ESMTP - altri RFC

Esiste un registro presso lo IANA che contiene tutte le estensioni registrate ad SMTP. Tra le altre:

EXPN	Expand the mailing list	[RFC821]
HELP	Supply helpful information	[RFC821]
TURN	Turn the operation around	[RFC821]
8BITMIME	Use 8-bit data	[RFC1652]
SIZE	Message size declaration	[RFC1870]
CHUNKING	Chunking	[RFC1830]
BINARYMIME	Binary MIME	[RFC1830]
CHECKPOINT	Checkpoint/Restart	[RFC1845]
PIPELINING	Command Pipelining	[RFC2197]
DSN	Delivery Status Notification	[RFC1891]
ETRN	Extended Turn	[RFC1985]
ENHANCEDSTATUSCODES	Enhanced Status Codes	[RFC2034]
STARTTLS	Start TLS (SSL)	[RFC2487]

A seguire: SMTP - RFC 2821 e 2822

SMTP - RFC 2821 e 2822

Nell'aprile 2001 sono stati pubblicati due nuovi RFC che ridefiniscono 821 e 822 prendendo in considerazione le modifiche proposte nel frattempo.

Di fatto questi RFC contengono e organizzano tutto il materiale di SMTP ed ESMTP, chiarificando l'organizzazione del protocollo ed eliminando aspetti ormai obsoleti.

Non introducono niente di nuovo, ma costituiscono lettura integrata ed unica, ad esclusione degli altri RFC,

Non sono ancora standard, ma draft standard.

Sicurezza in SMTP ed ESMPT (1)

Impersonificazione

- Il meccanismo di trasporto di SMTP è inerentemente insicuro. E' facilissimo realizzare messaggi che sembrino, per l'utente inesperto, provenire da altri mittenti, ed è comunque possibile realizzare messaggi che sembrino provenire da altri mittenti anche agli utenti più esperti.
- L'unica soluzione sicura in assoluto è ignorare le questioni di sicurezza al livello trasporto e risolverle a livello di messaggio (crittografia e firma digitale)

BCC

- Alcune implementazioni reinseriscono in header sperimentali le informazioni del livello trasporto SMTP, così vanificando il senso delle copie cieche (*blind copies*). Questo invece è un uso legittimo del protocollo di posta elettronica, e va rispettato.
- Si suggerisce ai sender di realizzare un messaggio autonomo per ogni destinatario del messaggio, così da impedire questo tipo di problemi.

A seguire: Sicurezza in SMTP ed ESMPT (2)

Sicurezza in SMTP ed ESMPT (2)

VRFY e EXPN

- Questi sono meccanismi di debug e controllo di cui si può abusare per ottenere dati privati (utenti abilitati, composizione di gruppi riservati, ecc.)
- Spesso sono disabilitate per mantenere la privacy degli utenti e delle organizzazioni interne della organizzazione in questione.
- Tuttavia è necessario che il server non produca informazioni false (ad es. affermi di verificare un indirizzo e poi non lo faccia).

Sfruttamento malizioso del relay

- Il relay permette ad un client SMTP di richiedere ad un altro di trasmettere per proprio conto messaggi.Questo può essere sfruttato maliziosamente per nascondere la vera origine dei messaggi SMTP.
- Per questo motivo molte implementazioni di server non accettano il relay da parte di client che provengano da fonti non note o controllate.

POP3 - RFC 1939 (1)

Post Office Protocol (version 3)

- SMTP si disinteressa di come il ricevente acceda alla sua mailbox. Si supponeva all'epoca che tutti avessero accesso via file system alla directory con le mailbox.
- POP3 permette ad un'applicazione utente di accedere alla mailbox posta su un altro sistema.
- POP3 non permette manipolazioni complesse sulla mailbox, ma soltanto la possibilità di scaricare e cancellare mail. Per operazioni più complesse si utilizza IMAP4.

A seguire: POP3 - RFC 1939 (2)

POP3 - RFC 1939 (2)

Una connessione POP3 è composta dalle seguenti parti:

- Greeting: riconoscimento reciproco di client e server
- Authorization: identificazione del client presso il server
- Transaction: uno o più comandi richiesti dal client
  - LIST: informazioni su uno o più messaggi
  - RETR: richiesta di un messaggio
  - DELE: cancellazione di un messaggio
  - TOP (opzionale): richiesta delle intestazioni del messaggio
  - UIDL (opzionale): richiesta di un numero univoco e perenne che identifichi un messaggio
- Update: il server aggiorna e rilascia le risorse acquisite durante la transazione e chiude la comunicazione

RFC 2449 elenca alcune estensioni di POP che non sono ancora standardizzate.

- Introdotte da CAPA, che permette al server di elencare le estensioni implementate

IMAP - RFC 2060

Internet Message Access Protocol (v. 4rev1)

- IMAP4 permette un controllo più sofisticato della propria mailbox anche se posta su un server remoto.
- IMAP4 permette operazioni di creazione, cancellazione e cambio di nome a mailbox; verifica di nuovi messaggi; cancellazione di messaggi; ricerca per contenuto ed attributi; scaricamento selettivo di attributi, parti e messaggi.
- IMAP inoltre è in grado di fare parsing di header RFC 822 e MIME, separare messaggi multipart e settare alcune flag inter-sessione.

NNTP - RFC 977 (1)

Network News Transfer Protocol

- Le news nascono come generalizzazione delle mailing list pubblica. Con la mailing list una copia di ogni messaggio viene creata dal server originante e consegnata ad ad ogni appartenente alla lista.
- Abbonati multipli appartenenti sullo stesso server ricevono una copia a testa del messaggio.
- Questo è un carico eccessivo e ridondante di lavoro per il server d'origine e per tutti i server condivisi.
- Con le news, invece, il server di origine spedisce una sola copia del messaggio, e questo viene diffuso in unica copia a tutti i server interessati tramite il *flooding*.
- Il flooding è un meccanismo di diffusione dei messaggi di news secondo un processo progressivo e non deterministico.

A seguire: NNTP - RFC 977 (2)

NNTP - RFC 977 (2)

- Ogni server di news possiede una lista di news host “amici”. Ad intervalli regolari si collega con uno di loro e confronta i messaggi ricevuti dall’ultimo confronto (comando IHAVE). Ogni messaggio mancante viene scambiato ed alla fine i server hanno la stessa lista di messaggi.
- Se un server ha solo un news host amico, allora riceverà da esso i nuovi messaggi dal mondo, e trasmetterà ad esso i messaggi generati localmente.
- Ma se un server ha più news host amici, allora la lista di messaggi disponibili sarà di volta in volta determinata anche dall’ordine delle connessioni trascorse.
- Quindi la disponibilità di messaggi dipende in massima parte dall’ordine delle connessioni effettuate, fino al caso particolare di ricevere una risposta prima della domanda che l’ha generata.

Estensioni ad NNTP

Nel corso del tempo, molte sono state le estensioni di NNTP non ufficiali ma standardizzate de facto dalla pratica comune.

L'RFC 2980 ne contiene una lista con suggerimenti di implementazione e adozione.

- Streaming (MODE STREAM, TAKETHIS, CHECK), per spedire grosse quantità di messaggi tutte insieme e non singolarmente
- Replicazione (XREPLIC): per duplicare esattamente una struttura di newsgroup tra un server e l'altro
- Autenticazione (AUTHINFO) il client informa il server delle credenziali dell'utente che sta accedendo alle news
- Ricerca di messaggi (XPATH, XHDR, XOVER, ecc.) per cercare specifici messaggi basati su pattern
- Ricerca di newsgroup (LIST, LISTGROUP, ecc.) per cercare newsgroup basati su pattern o altre caratteristiche.

Conclusioni

Qui abbiamo parlato di protocolli basati su testo, specialmente per lo scambio di posta elettronica. Va notato che:

- le connessioni avvengono con ruoli rigorosi (client e server)
- I comandi e risposte avvengono in modo testo (telnettabili)
- Si cerca di eliminare la complicazione, o di localizzarla fortemente
- I codici di risposta sono sia numerici (machine-readable) che in testo (human-readable).

Riferimenti

- ***Wilde, Wilde's WWW, capitoli 1.4 e 11***

Altri testi:

- A. Tanenbaum, *Reti di computer*, Prentice Hall, capitoli 1.4, 7.4 e 7.5.
- Tutti gli RFC citati (`http://www.ietf.org/rfc/rfc###[#].txt`)