



UNIVERSITÀ
DI PARMA

DIPARTIMENTO DI SCIENZE MATEMATICHE, FISICHE ED INFORMATICHE
Corso di Laurea in Informatica

II Livello Applicativo – Parte A

Applicativi UDP: TFTP e DNS

RETI DI CALCOLATORI - a.a. 2023/2024

Roberto Alfieri

Livello Applicativo: sommario

PARTE A

- ▶ Applicativi UDP: TFTP e DNS

PARTE B

- ▶ I servizi di posta elettronica: SMTP, POP e IMAP.

PARTE C

- ▶ Il World Wide Web

PARTE D

- ▶ Multimedia

TFTP e FTP

TFTP è la versione semplificata (Trivial) di FTP (File Transfer Protocol)

<http://openskill.info/topic.php?ID=87>

Per noi è interessante perché è un esempio di come viene gestito a livello applicativo il controllo del flusso.

Il protocollo FTP (RFC 959) è stato sviluppato per trasferimento affidabile ed efficiente dei dati, per questo motivo si basa TCP. Il server FTP offre anche un servizio di autenticazione per l'accesso al file-system, la gestione delle directory (navigazione, creazione, cancellazione) e dei file.

Altra caratteristica peculiare è quella di usare due porte: la TCP/21 (comandi) e la TCP/20 (dati). Le modalità di funzionamento sono due: attiva e passiva.

Nella modalità attiva il client apre il canale comandi verso il server (porta 21 del server), mentre per la trasmissione dati il client svolge la funzione di server, ovvero rimane in ascolto sulla porta > 1024 mentre il server si comporta da client utilizzando la porta 20.

In genere le politiche di sicurezza impediscono l'accesso alle porte dei client bloccando questa modalità.

Nella modalità passiva il server indica al client la porta > 1024 da utilizzare per il trasferimento dei dati.

TFTP non supporta l'autenticazione e utilizza UDP, con il server in ascolto sulla porta 69. Questo significa che la gestione del flusso (numerazione dei pacchetti, Ack, gestione degli errori) viene realizzata a livello applicativo, all'interno di TFTP.

Il protocollo TFTP

Ogni trasferimento inizia con una richiesta di read (comando get) o write (comando put).

GET: Il server risponde con un file frammentato in datagrammi numerati.

Ogni datagramma deve essere riscontrato.

Il block-size di default è di 512 byte

Un pacchetto di dimensione inferiore rappresenta l'ultimo pacchetto trasmesso

Opcode (16 bit)

1 RRQ (Read Request)

2 WRQ (Write Request)

3 DATA

4 ACK

5 ERROR

2 bytes	string	1 byte	string	1 byte
Opcode	Filename	0	Mode	0

RRQ/WRQ Packet

2 bytes	2 bytes	up to 512 bytes of data
Opcode	Block#	Data

DATA Packet

2 bytes	2 bytes
Opcode	Block#

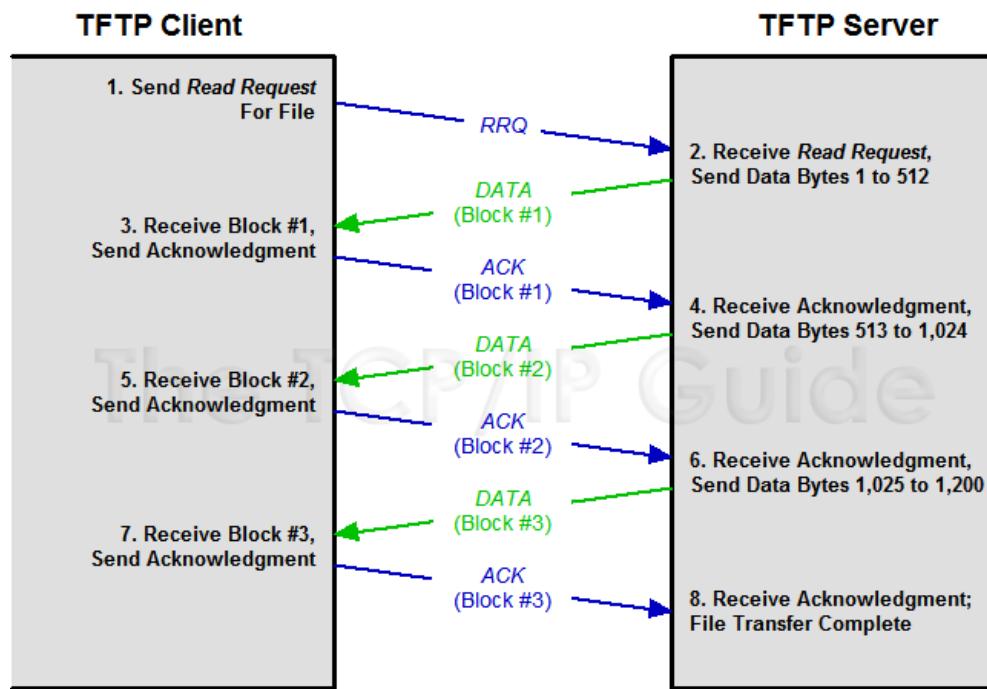
ACK Packet

2 bytes	2 bytes	string	1 byte
Opcode	Block#	ErrMsg	0

ERROR Packet

Le fasi di una sessione TFTP

- Il client contatta il server inviando un pacchetto di tipo RRQ o WRQ
- Il server risponde inviando/ricevendo pacchetti DATA di 512 byte.
Per ogni DATA inviato/ricevuto viene inviato/ricevuto un ACK (o un ERROR)
- I pacchetti vengono trasferiti finché la loro lunghezza non è inferiore a 512 byte;
- Termine della connessione;



Esempi:

```
> tftp -v <nome server> -c put /etc/hosts hosts  
> tftp -v <nome server> -c get hosts
```

DNS: Domain Name System

DNS è il protocollo applicativo più importante tra quelli che si appoggiano su UDP.

Scopo del sistema DNS:

- ▶ gestire uno spazio univoco dei nomi per i nodi della rete
- ▶ Fornire agli utenti di IP un servizio per la traduzione nome-numero e numero-nome

Lo spazio dei nomi è strutturato in modo gerarchico, come il file-system, ma la radice è a destra. *Esempio: didattica-linux.unipr.it.*

Il primo elemento è il nome locale del nodo, mentre gli elementi successivi (domini) rappresentano il percorso nella gerarchia e sono separati da punto ('.').

Un nome completo termina sempre con un punto, che rappresenta la radice della gerarchia.

Il dominio più a destra (**it** nell'esempio) è detto Top Level Domain (TLD).

I TLD sono gestiti dall'organismo internazionale [ICANN](#) (attraverso la sua emanazione [IANA](#)) che li assegna alle organizzazioni che ne fanno richiesta, mentre i livelli successivi sono gestiti in modo autonomo dalle organizzazioni assegnatarie.

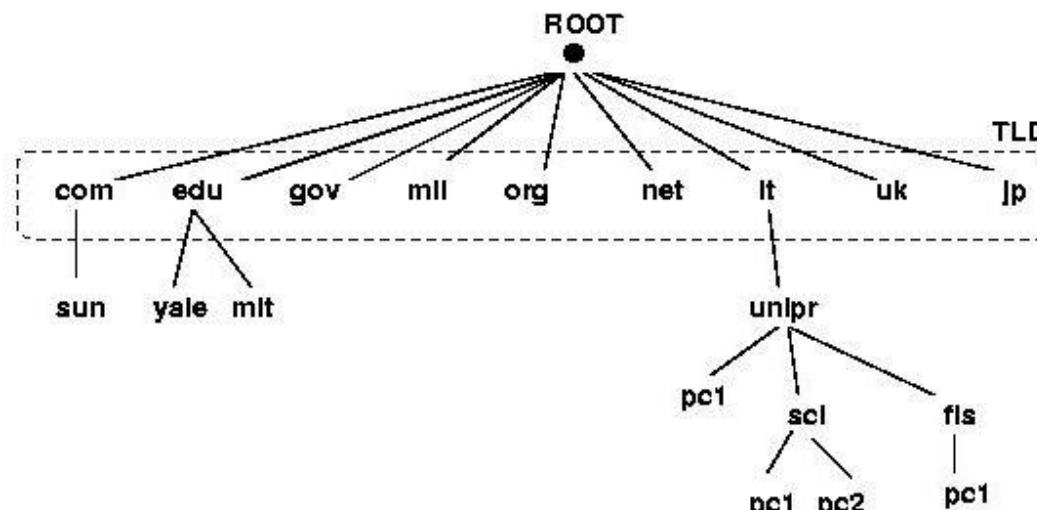
DNS: i Top Level Domain

Inizialmente Internet era composta unicamente da nodi americani, per cui esistono alcuni TLD che rispecchiano la strutturazione Statunitense originale (1985): **com (commerciali), edu (istituzioni educative), gov (governo federale US), mil (forze armate US), net (provider di rete), org (organizzazioni no-profit)**.

Successivamente, con la diffusione di Internet, sono nati i TLD geografici nazionali: **it, es, fr, de, gr ca, at, au, be, nl, pt, ch, ecc**

A partire dal 2000 ICANN ha approvato diversi nuovi nomi TLD generici quali: **biz (business), info (informazioni), name (nome di persona), pro (professionisti), coop (cooperative), museum (musei), travel (viaggi), aero (aerotrasporti)**

Attualmente sono attivi seguenti TLD <http://www.iana.org/domains/root/db/>



Risoluzione Inversa

La risoluzione inversa consiste nella risoluzione del nome a partire dall'indirizzo IP. Viene usata ad esempio per produrre un output leggibile nei file di log, oppure per controlli di autenticazione (e.g. richiedere che il client sia registrato). Il nome dei domini di Reverse è composto dai numeri della rete (in ordine rovesciato), seguiti dalla stringa “in-addr.arpa” (TLD per la risoluzione inversa). Il rovesciamento del numero consente di ricercare i numeri nello stesso albero dei nomi, utilizzando lo stesso procedimento di parsing da destra verso sinistra.

Ad esempio:

10.48.78.160.in-addr.arpa

è il nome di

caio.cce.unipr.it

nel ramo della risoluzione inversa.

provare il comando:

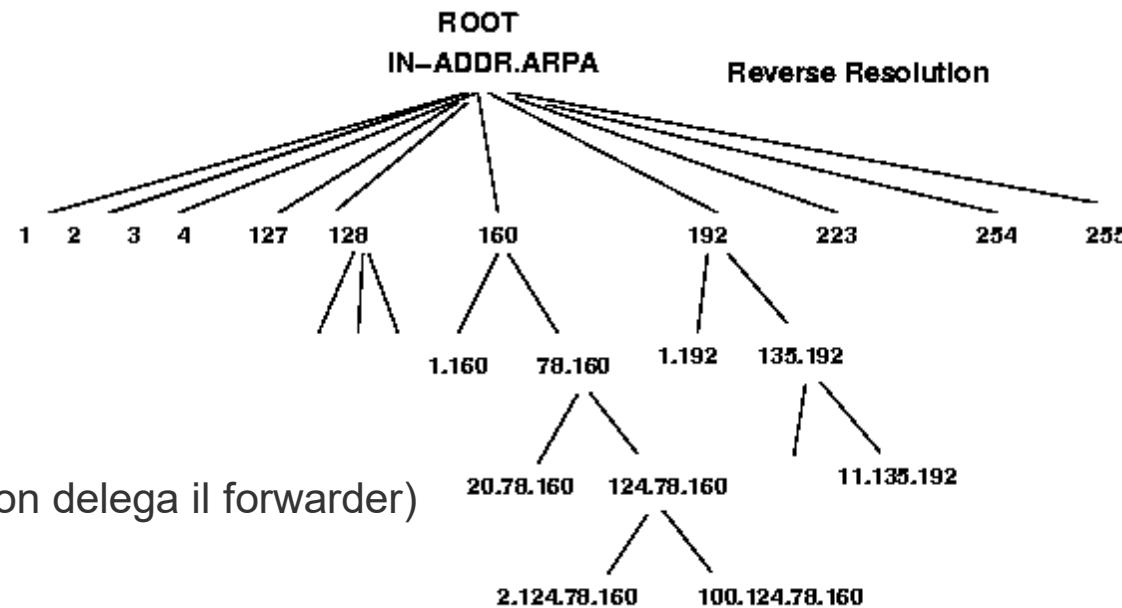
dig PTR 10.48.78.160.in-addr.arpa

dig -x 160.78.48.10

dig -x 160.78.48.10 +trace

-x → semplifica il reverse search

+trace → dig esegue query iterative (non delega il forwarder)



DNS client

Un DNS client contiene un componente software, detto Resolver, che ha il compito di risolvere la richiesta. Il Resolver deve essere configurato con l'indicazione di almeno un **server DNS forwarder** a cui rivolgersi per le risoluzioni.

Il server DNS forwarder recupera l'informazione (interagendo eventualmente con gli altri server DNS) e la comunica al Resolver.

Nei sistemi Linux questa configurazione viene stabilita nel file `/etc/resolv.conf`.

Esempio:

`nameserver 160.78.48.10`

la configurazione del DNS può essere impostata dinamicamente dal protocollo DHCP assieme alle altre informazioni di configurazione (indirizzo IP, Gateway, netmask, ..)

Vista la criticità del servizio DNS un client dichiara tipicamente almeno 2 DNS Forwarder per ridondanza.

In un programma applicativo il ruolo di Resolver è svolto dalla funzione **gethostbyname()**.

Esistono anche specifici client a linea di comando come **nslookup** e **dig**

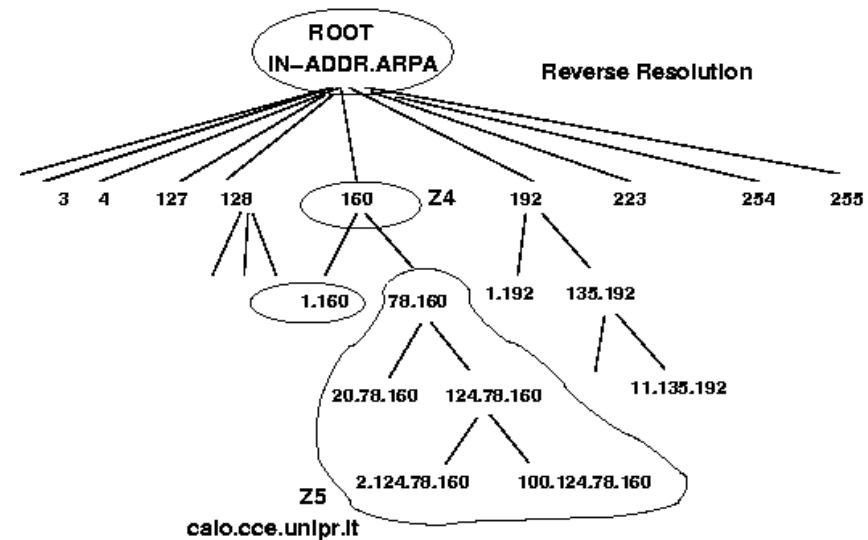
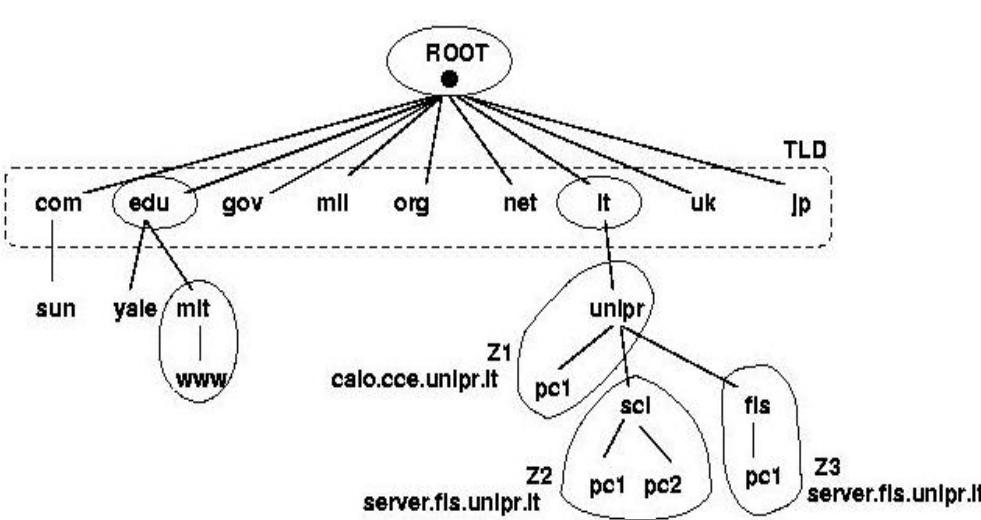
Le Zone del DNS

I TLD possono creare sottodomini di livello 2, i quali a loro volta possono gestire domini di livello 3 e così via. *Ad esempio il dominio a.b.com. può creare c.a.b.com*

Lo spazio dei nomi è gestito in modo distribuito suddividendolo in “Zone”. Ogni Zona include una porzione dell'albero che gestisce in modo autonomo con un DNS server Primario e uno o più DNS server Secondari che ne replicano i dati (per sicurezza e prestazioni).

Una Zona si forma attraverso la delega che il server della Zona superiore assegna mediante il record NS

Ogni nuovo host viene inserito tipicamente sia nella zona per la risoluzione diretta che nella zona per la risoluzione inversa.



Il servizio DNS: il server

Il **servizio DNS** è definito dall'RFC1034 e RFC1035

BIND è il nome del demone DNS più comunemente usato sui sistemi Unix/Linux.

Un server può essere configurato per diversi tipi di funzionamento:

- ▶ **Server Autoritativo di Zona Primario o Secondario**

L'amministratore di Zona aggiorna i dati sul Primario.

I server Secondari si sincronizzano con il primario replicando tutta la Zona (“Zone Transfer”) attraverso un Data-Pull sulla porta 53/TCP.

Risponde alle query che riceve (53/UDP).

Un server Autoritativo è generalmente anche Forwarder NS.

- ▶ **Server Forwarder (o Caching Name Server):**

Riceve query dai client (53/UDP)

Ottiene la risposta interrogando i Server Autoritativi

Mantiene copia locale in Cache delle risposte ottenute

Invia la risposta al client

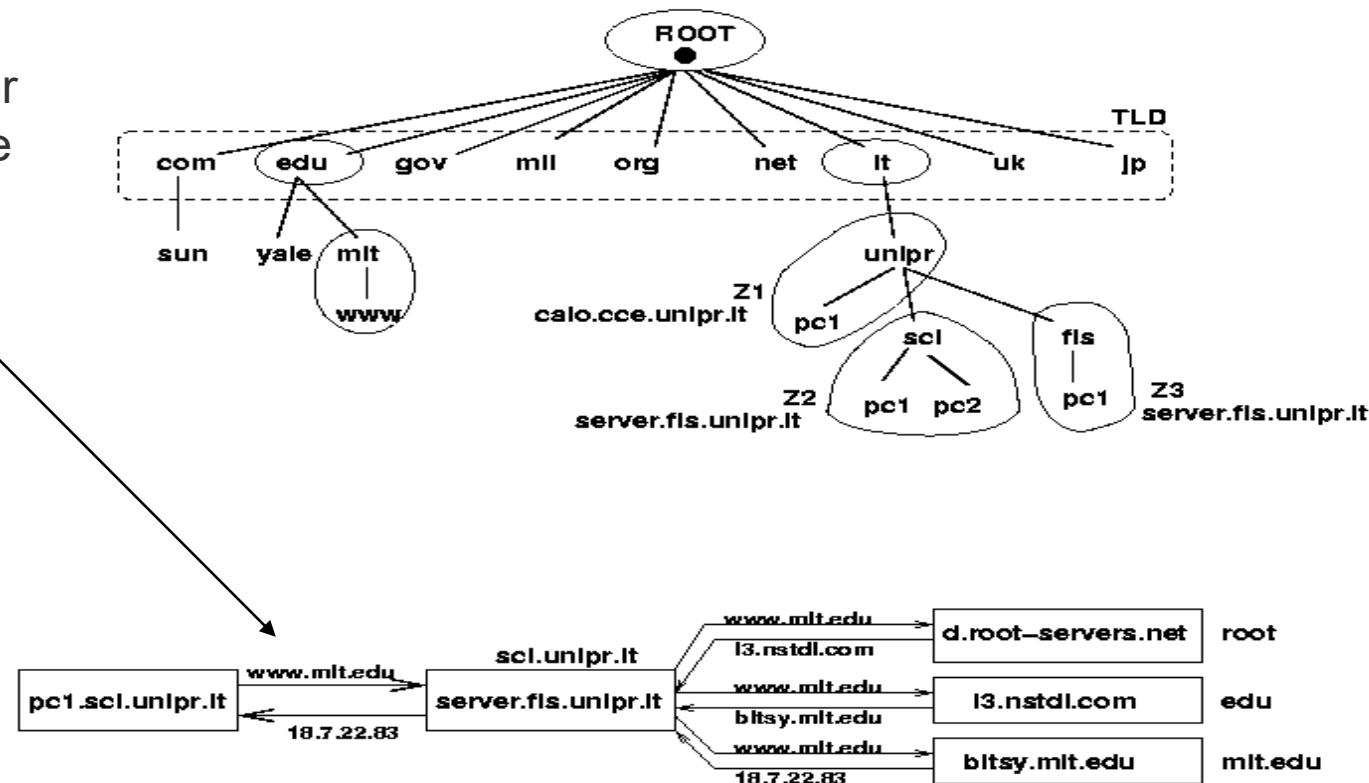
La risoluzione Ricorsiva e Iterativa

Se il server che riceve a richiesta è autoritativo per il dato richiesto risponde direttamente, altrimenti occorre attraversare l'albero passando attraverso i server autoritativi coinvolti. L'attraversamento può essere ricorsivo o iterativo.

Modalità Ricorsiva: Se il server interrogato non è autoritativo per il dato richiesto, passa la richiesta al server successivo e così via in modo ricorsivo.

Modalità Iterativa: Il server restituisce al client l'indirizzo del server successivo. In questo modo è il DNS locale che contatta direttamente i server coinvolti.

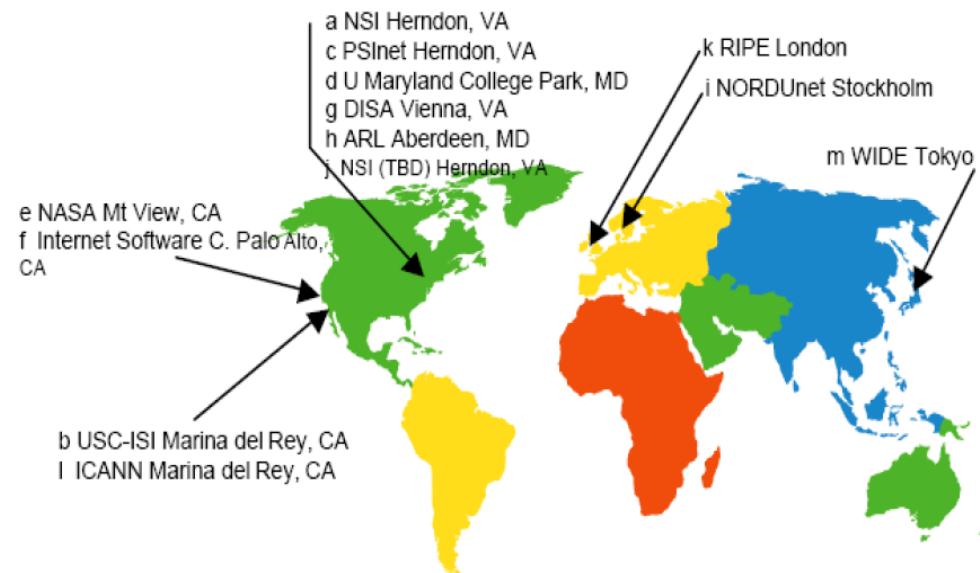
Generalmente un server ammette query ricorsive solo per i client locali.



I Root Server

L'albero DNS possiede 13 **Root Server** che contengono le informazioni riguardo i domini di primo livello. Questi server vengono contattati ogni volta che un client chiede informazioni relative ad altri TLD, quindi ogni DNS Forwarder deve possedere una lista aggiornata dei Root Server (file named.ca di Bind).
I Root NS sono iterativi: se non hanno la risposta forniscono l'indirizzo del server del TLD da contattare.

A.ROOT-SERVERS.NET.	3600000 IN	A	198.41.0.4
B.ROOT-SERVERS.NET.	3600000 IN	A	192.228.79.201
C.ROOT-SERVERS.NET.	3600000 IN	A	192.33.4.12
D.ROOT-SERVERS.NET.	3600000 IN	A	128.8.10.90
E.ROOT-SERVERS.NET.	3600000 IN	A	192.203.230.10
F.ROOT-SERVERS.NET.	3600000 IN	A	192.5.5.241
G.ROOT-SERVERS.NET.	3600000 IN	A	192.112.36.4
H.ROOT-SERVERS.NET.	3600000 IN	A	128.63.2.53
I.ROOT-SERVERS.NET.	3600000 IN	A	192.36.148.17
J.ROOT-SERVERS.NET.	3600000 IN	A	192.58.128.30
K.ROOT-SERVERS.NET.	3600000 IN	A	193.0.14.129
L.ROOT-SERVERS.NET.	3600000 IN	A	199.7.83.42
M.ROOT-SERVERS.NET.	3600000 IN	A	202.12.27.33



Resource Record

Le informazioni relative alla zona vengono memorizzate come **Resource Record (RR)**.

Il formato generico di un RR dispone dei seguenti campi:

- **Nome**: Il nome di dominio a cui questo RR si riferisce.
- **TTL**: Tempo di vita del RR nella cache dei server DNS prima di essere scartato.
- **Classe**: Identifica la famiglia di protocollo. **IN** che indica il sistema Internet.
- **Tipo**: Il tipo di RR. I tipi principali sono:
 - ✓ **A**: Il più usato. Indica l'indirizzo IPv4 per il nome specificato.
 - ✓ **AAAA**: Indica l'indirizzo IPv6 (eventuale) associato al nome.
 - ✓ **CNAME**: Record Canonical Name, usato per indicare un nome di alias.
 - ✓ **MX**: Mail eXchanger, indica un host che gestisce la posta per il dominio.
 - ✓ **NS**: Un server DNS per il dominio specificato.
 - ✓ **PTR**: Usato nella risoluzione inversa per associare un indirizzo IP al nome.
 - ✓ **SOA**: Start of Authority, un RR che indica il server DNS dove risiedono i dati autoritativi per questo dominio ed alcuni dati amministrativi.

Esempio di Record

```
; Authoritative data for unipr.it. (Zone1)
;nome      classe    tipo     valore          (serial  refresh  retry  expire  min_life)
unipr.it   IN        SOA      caio.cce.unipr.it  manager (20050818 8H 2H 1W 1D)
unipr.it   IN        TXT      "Univ Parma"
unipr.it   IN        MX       1           mail.unipr.it.
unipr.it   IN        MX       2           mail2.unipr.it.
fis        IN        NS       server.fis.unipr.it. ; Zone 2

dns        IN        CNAME    server1.unipr.it.
mail       IN        CNAME    server2.unipr.it.
mail2      IN        CNAME    server1.unipr.it.
pop        IN        CNAME    server2.unipr.it.
www        IN        CNAME    server1.unipr.it.

server1    IN        A        160.78.124.1
            IN        HINFO    Server1 Linux

server2    IN        A        160.78.124.2
            IN        HINFO    Server2 Linux

pc1        IN        A        160.78.124.101
            IN        HINFO    Client1 Win2000

pc2        IN        A        160.78.124.102
            IN        HINFO    Client2 Win2000
```

DNS e la Posta elettronica

Per ogni dominio che è in grado di ricevere posta il DNS fornisce una lista di server SMTP a cui inviare il messaggio.

In questo modo, se il mail server principale del destinatario non è operativo, è possibile inviare il messaggio verso un server di backup in grado di gestire la posta del dominio.

Queste informazioni sono contenute nei record MX (Mail eXchanger).

unipr.it

IN

MX

1

icaro.cce.unipr.it.

unipr.it

IN

MX

2

ipruniv.cce.unipr.it.

Il campo numerico indica la priorità (il numero più basso ha maggiore priorità)

Provare il comando:

➤ dig -t mx unipr.it +short

0 unipr-it.mail.protection.outlook.com.

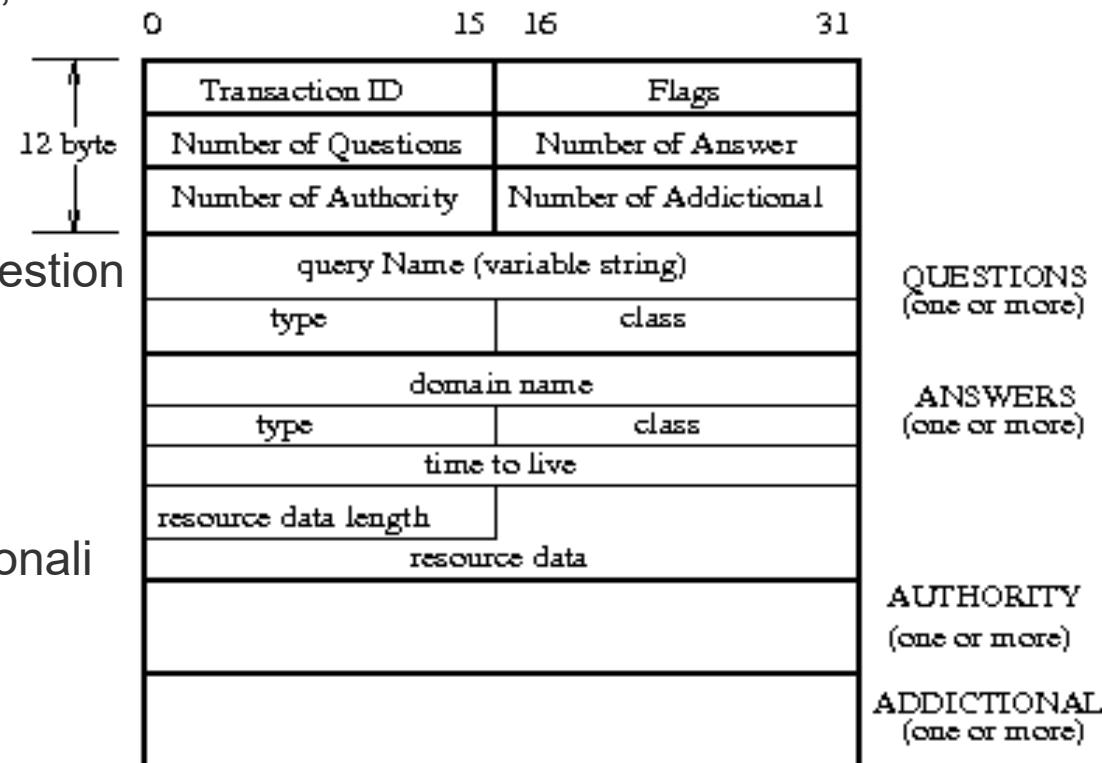
Il formato del pacchetto DNS

Tutti i pacchetti DNS hanno lo stesso formato di figura, composto da:

Transaction ID: serve per associare domanda a risposta

16 Flags tra cui: QR (domanda (0) / risposta (1)) - OPCODE (4 bits, tipo di query)
RD (Recursion Desired) - RA (Recursion Available) - RCODE (4 bits, esito della risposta)

4 sezioni: QUESTIONS, ANSWERS, AUTHORITY e ADDITIONALS



QUESTIONS:

domande per il DNS.

ANSWERS:

elenco di RR che rispondono alla Question

AUTHORITY:

elenco di RR degli NS autoritativi
che portano più vicino alla risposta.

ADDITIONAL:

elenco di RR con informazioni addizionali

DNS dinamico (dDNS)

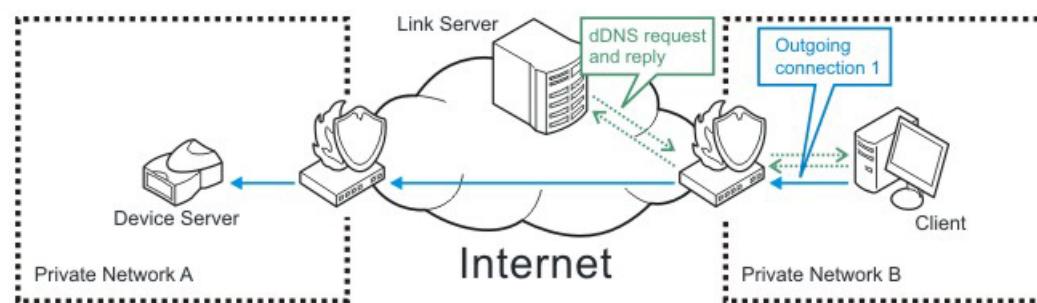
Il DNS Dinamico è una tecnologia che permette ad un nome DNS in Internet di essere sempre associato all'indirizzo IP di uno stesso host, anche se l'indirizzo cambia nel tempo (tipicamente computer portatili).

Il servizio è costituito da una popolazione di client dinamici (host con indirizzo IP dinamico che vogliono che il loro IP attuale sia registrato nel DNS), da uno o più server DNS dinamici e da un protocollo di comunicazione tra le due parti.

[nsupdate](#) è una utility disponibile ai client per l'aggiornamento del DNS.

L'aggiornamento dinamico del DNS può essere fatto direttamente dal server dhcp:

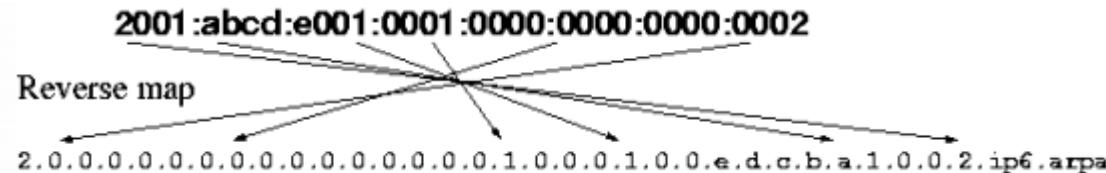
<http://semicomplete.com/articles/dynamic-dns-with-dhcp>



DNS e IPv6

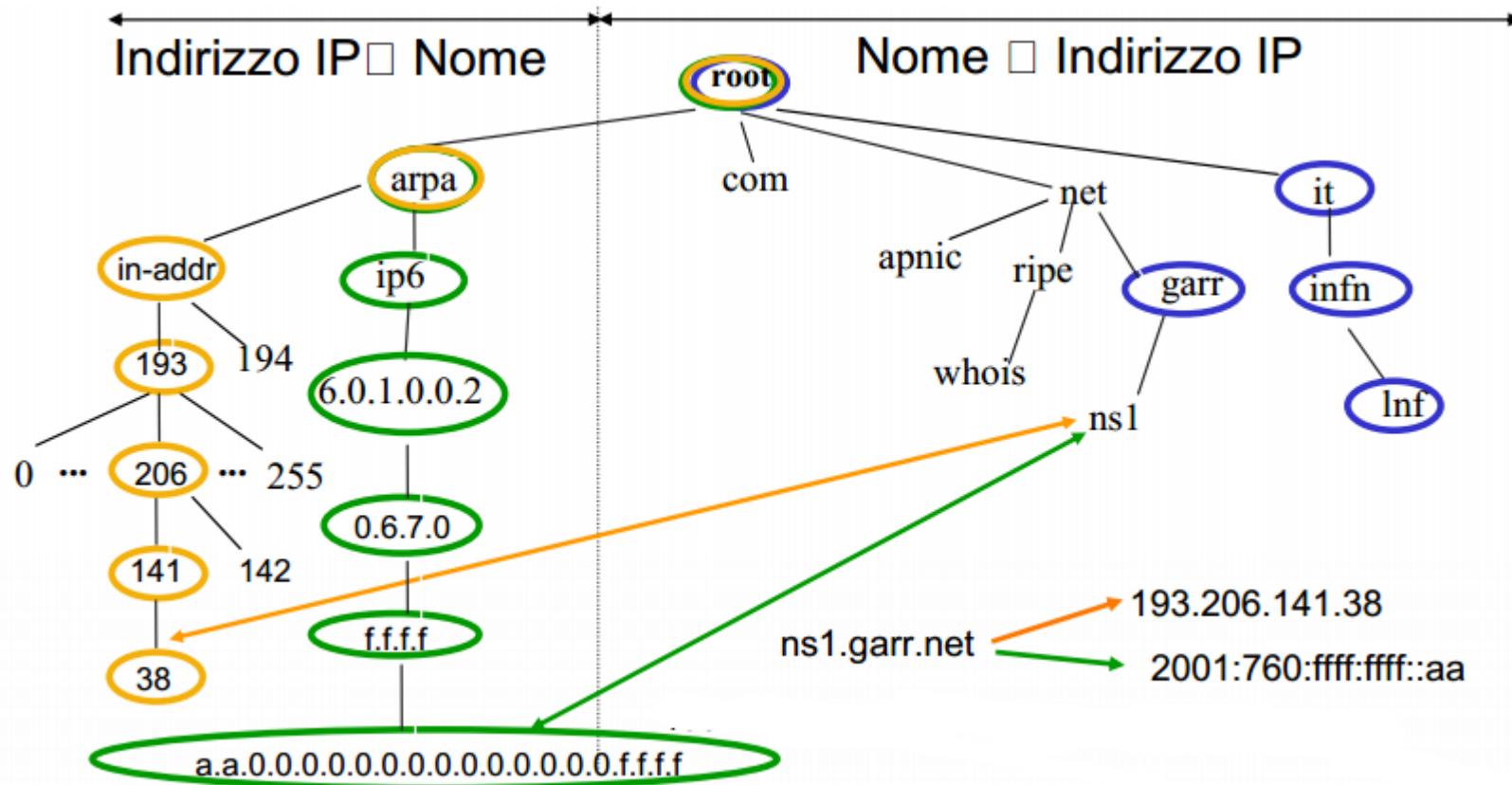
L'utilizzo di IPv6 non modifica i meccanismi di base del DNS, ma è solo stato necessario introdurre alcuni nuovi elementi:

- Un nuovo Resource Record AAAA
 - Un nuovo dominio per la risoluzione inversa : ip6.arpa (RFC 3152)



Riferimenti: Tutorial del GARR http://www.garr.it/eventiGARR/ws9/pdf/Gallo_Valli_pres.pdf

Il name space IPv4 e IPv6



Riferimenti: Tutorial del GARR: http://www.garr.it/eventiGARR/ws9/pdf/Gallo_Valli_pres.pdf

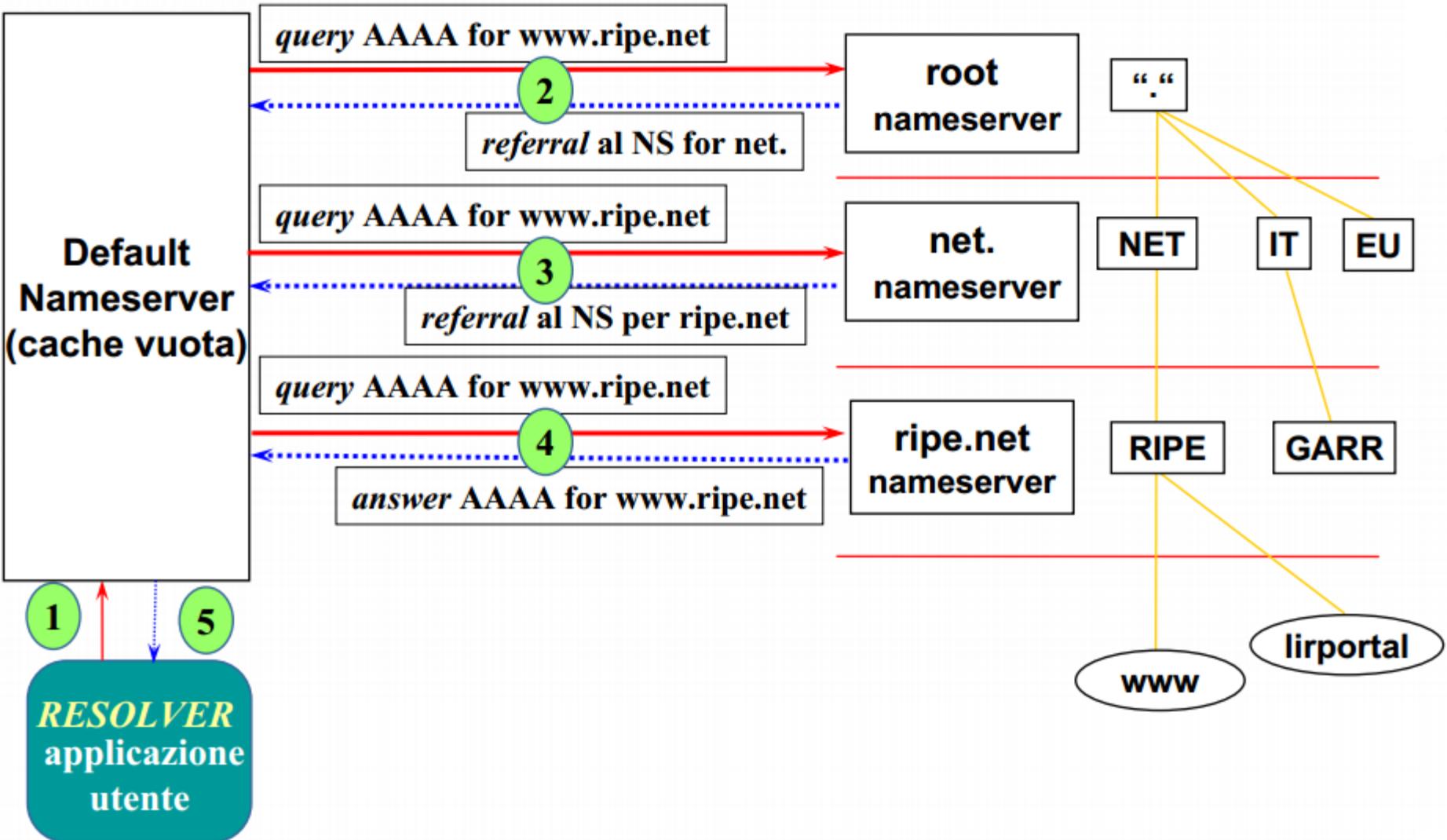
IPv6 e i root NS

Lettera	Vecchio nome	Indirizzo IPv6	Indirizzo IPv4	Operatore	Luogo geografico
A	ns.internic.net	2001:503:ba3e::2:30	198.41.0.4	VeriSign	Dulles, Virginia, USA
B	ns1.isi.edu	2001:500:84::b	192.228.79.201	USC Information Sciences Institute	Marina del Rey, California, USA
C	c.psi.net	2001:500:2::c	192.33.4.12	Cogent	distribuito in anycast
D	terp.umd.edu	2001:500:2d::d	199.7.91.13	University of Maryland	College Park, Maryland, USA
E	ns.nasa.gov	2001:500:a8::e	192.203.230.10	NASA	Mountain View, California, USA
F	ns.isc.org	2001:500:2f::f	192.5.5.241	ISC	distribuito in anycast
G	ns.nic.ddn.mil	2001:500:12::d0d	192.112.36.4	NIC del DoD USA	Vienna, Virginia, USA
H	aos.arl.army.mil	2001:500:1::53	128.63.2.53	U.S. Army Research Lab	Poligono di Aberdeen, Maryland, USA
I	nic.nordu.net	2001:7fe::53	192.36.148.17	Autonomica Archiviato il 1° maggio 2001 in Internet Archive.	distribuito in anycast
J	-	2001:503:c27::2:30	192.58.128.30	VeriSign	distribuito in anycast
K	-	2001:7fd::1	193.0.14.129	RIPE NCC	distribuito in anycast
L	-	2001:500:9f::42	198.32.64.12	ICANN	Los Angeles, California, USA
M	-	2001:dc3::35	202.12.27.33	Progetto WIDE	distribuito in anycast

Al momento non è possibile aggiungere altri nomi di server, a causa di un problema di ottimizzazione del protocollo: un pacchetto UDP deve poter contenere tutti i nomi dei server e con un quattordicesimo nome si supererebbe la dimensione massima del pacchetto.

Riferimenti: Wikipedia https://it.wikipedia.org/wiki/Root_nameserver

Processo iterativo per risoluzione dei nomi



Riferimenti: Tutorial del GARR http://www.garr.it/eventiGARR/ws9/pdf/Gallo_Valli_pres.pdf

Sicurezza del DNS

Le specifiche originali del DNS (RFC 882, 1034 e 1035) non prevedono autenticazione, integrità dei dati o cifratura e espongono il servizio a violazioni della sicurezza. Il principale problema di sicurezza riguarda la possibilità di dirottare la richiesta di traduzione del nome di un server verso un IP fraudolento ([DNS spoofing](#)).

Una tecnica molto diffusa è il [DNS cache poisoning](#) (avvelenamento della cache).

L'attaccante gestisce un server DNS autoritativo per un dominio fake. Quando viene interrogato risponde includendo informazioni false (con un TTL molto grande) riguardo un dominio target, che vengono memorizzate in cache ed utilizzate quando successivamente vengono richieste traduzioni sul target, dirottando la richiesta verso un IP fraudolento.

La soluzione principale è stata l'introduzione del DNSSEC (RFC 9364).

DNS Security Extensions (DNSSEC)

I Domain Name System Security Extensions (DNSSEC) sono una serie di specifiche dell'IETF ([RFC 9364](#)) per garantire la sicurezza e affidabilità delle informazioni fornite dai sistemi DNS.

Servizi:

- Autenticazione: garanzia sull'origine dei dati DNS
- Integrità dei dati ricevuti (non la riservatezza)

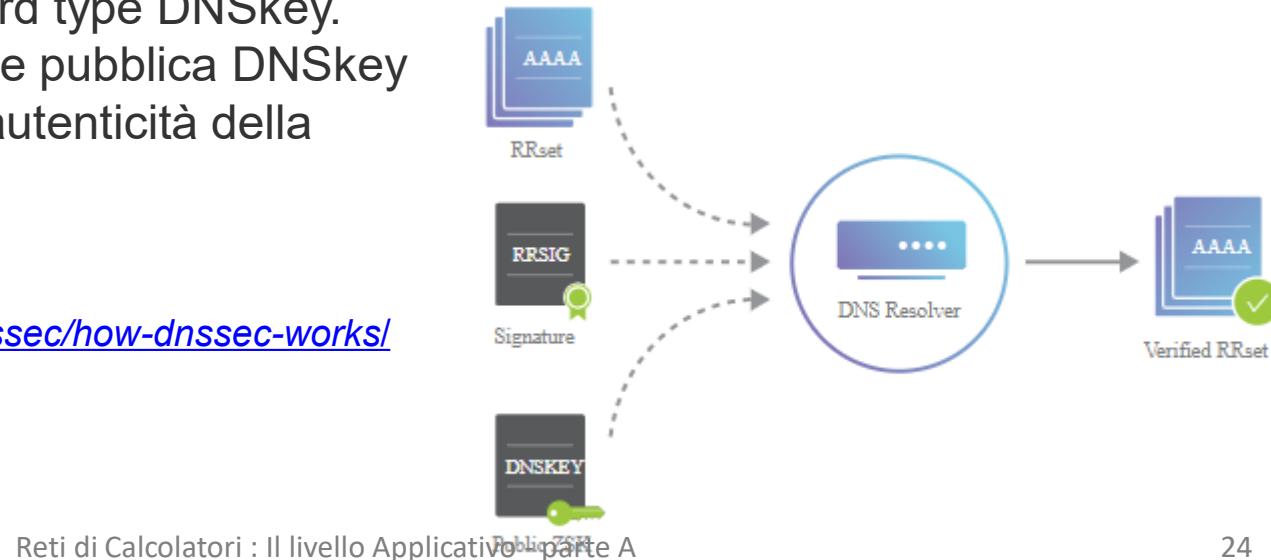
Funzionamento:

Ogni server DNSSEC possiede una coppia di chiavi crittografiche, una pubblica e una privata. La chiave privata viene utilizzata per firmare ogni Resource Record (RRset) generando un nuovo Record type, il RRsig. Il server pubblica anche la chiave pubblica introducendo il nuovo record type DNSkey.

Il client, utilizzando la chiave pubblica DNSkey del server può verificare l'autenticità della firma RRsig.

Riferimenti:

<https://www.cloudflare.com/dns/dnssec/how-dnssec-works/>





UNIVERSITÀ
DI PARMA

DIPARTIMENTO DI SCIENZE MATEMATICHE, FISICHE ED INFORMATICHE
Corso di Laurea in Informatica

Il Livello Applicativo – Parte B

La posta elettronica

RETI DI CALCOLATORI - a.a. 2023/2024

Roberto Alfieri

Livello Applicativo: sommario

PARTE A

- ▶ Applicativi UDP: TFTP e DNS

PARTE B

- ▶ I servizi di posta elettronica: SMTP, POP e IMAP.

PARTE C

- ▶ Il World Wide Web

PARTE D

- ▶ Multimedia

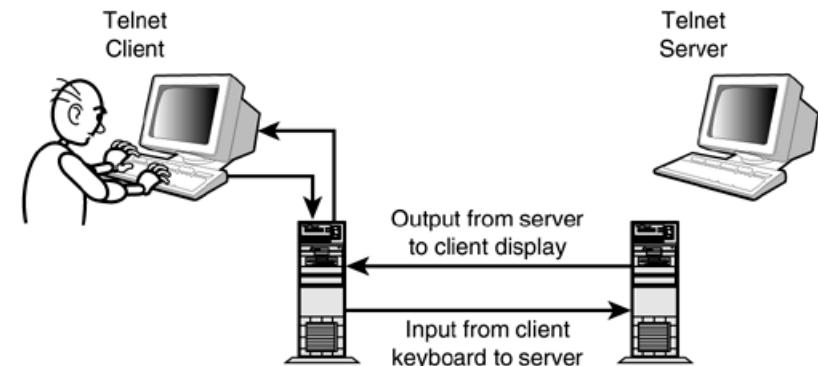
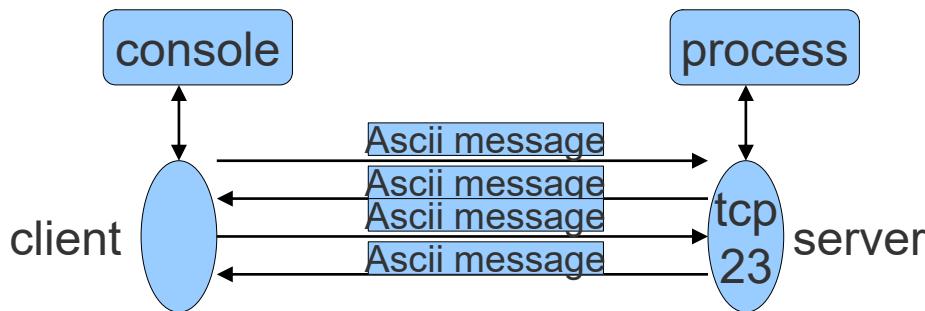
Il protocollo Telnet: emulazione di terminale remoto

Il telnet è un protocollo storico di TCP/IP (RFC 854) creato per l'accesso remoto alla console testuale di un host, basandosi su una connessione TCP tra client e server.

La porta assegnata da IANA è la 23/TCP. Operazioni:

- **Client.** Il client legge lo stream dallo standard input (tastiera) e lo gira al server. Lo stream proveniente dal server viene inviato allo standard output (video).

- **Server.** Il server legge le linee provenienti dal client, le interpreta come comandi di console e invia al client l'output del comando. Il dialogo inizia con la richiesta delle credenziali (username/password).



Telnet è un protocollo testuale (richieste e risposte sono spedite in ASCII, compresa la password) ha quindi un livello di sicurezza molto basso ed è stato sostituito da protocolli più sicuri (SSH).

Viene comunque ancora utilizzato per casi speciali (console di apparati di rete) o come semplice strumento di analisi di altri protocolli testuali (SMTP, POP3, IMAP, HTTP, ...)

La Posta Elettronica

Definita nel 1982 con gli RFC821, RFC822 ed estesa con RFC2821 e RFC2822
Ogni utente possiede una MailBox in cui gli altri utenti possono liberamente inserire messaggi.

Nel 2015 lo spam rappresentava il 55% del flusso mondiale di mail
(<https://securelist.com/kaspersky-security-bulletin-spam-and-phishing-in-2015/73591/>)

Un indirizzo di posta è costituito da due identificativi: nome del server che gestisce la mailbox e identificativo dell'utente: <utente>@<server>

Dove <server> è l'indirizzo IP o un suo identificativo nel DNS.

Il record MX del DNS consente di creare caselle di posta in un dominio, a cui possiamo associare uno o più server di posta per la sua gestione.

Ad esempio il seguente record DNS (ottenuto con dig -t MX unipr.it):
unipr.it. 86400 IN MX 0 unipr-it.mail.protection.outlook.com.

Consente di creare indirizzi di posta del tipo <utente>@unipr.it
Le cui caselle verranno gestite dal server unipr-it.mail.protection.outlook.com.

La Posta Elettronica : l'architettura

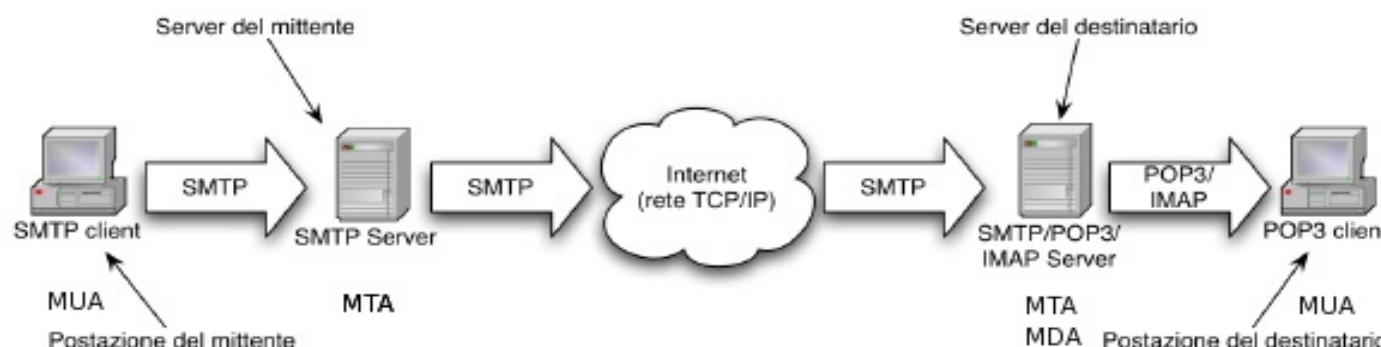
Ogni utente ha bisogno di uno MUA - “Message User Agent”, un programma che gli consente di inviare messaggi o leggere i messaggi dalla propria Mailbox.

Il MUA consegna il messaggio ad un MTA - “Message Transfer Agent” che ha il compito di trasportare il messaggio a destinazione. Il messaggio può attraversare diversi MTA prima di arrivare nella MailBox del destinatario.

Il primo MTA è detto anche Submission server perché è utilizzato dai MUA per la sottomissione delle mail e dovrebbe supportare autenticazione (SMTP AUTH).

Sull'ultimo MTA è presente anche l'MDA - “Message Delivery Agent” che si occupa della consegna del messaggio nella MailBox dell'utente.

Se il MUA del destinatario e MDA sono su host diversi occorre un protocollo per la lettura dei messaggi (POP3 o IMAP)



Il formato dei messaggi

Il formato dei messaggi è stato codificato nel 1982 attraverso l'RFC 822, superato nel 2001 dall'RFC 2822 e poi nel 2008 dall'RFC 5322.

I messaggi sono codificati in ASCII standard (7bit) in righe di max.1000 caratteri

Formato:

Intestazione	(Header)
<cr><lf>	(riga vuota)
corpo del messaggio	(Body)

Intestazione: in ogni riga la coppia <campo>:<valore>

Principali campi:

- ▶ To: indirizzi dei destinatari
- ▶ Cc: indirizzi destinatari secondari
- ▶ Bcc: destinatari secondari con indirizzi nascosti
- ▶ From: La persona che ha creato il messaggio (obbligatorio)
- ▶ Received: Riga aggiunta da ogni MTA attraversato
- ▶ Date: data e ora di invio del messaggio
- ▶ Subject: Breve riepilogo del messaggio
- ▶ Message-ID: Identificatore unico del messaggio (generato automaticamente).
- ▶ User-Agent: Client di posta utilizzato dall'utente

Esempio messaggio

MTA
Attraversati

Received: from smtp.unipr.it (sud.cce.unipr.it [160.78.48.162])

by unipr.it (8.12.6/8.12.6) with ESMTP id j4CCXorj032665

for <destinatario@unipr.it>; Thu, 12 May 2005 14:33:50 +0200



Received: from smtp2.mathworks.com (smtp2.mathworks.com [144.212.95.218])

by smtp.unipr.it (8.12.10/8.12.10/SuSE Linux 0.7) with ESMTP id j4CCTeOW001983

for <destinatario@unipr.it>; Thu, 12 May 2005 14:29:45 +0200

Received: from mail-vif.mathworks.com (fred-ce0.mathworks.com [144.212.95.18])

by smtp2.mathworks.com (8.12.11/8.12.11) with ESMTP id j4CCVPR5009740

for <destinatario@unipr.it>; Thu, 12 May 2005 08:31:25 -0400 (EDT)

Received: from telesto.mathworks.com (telesto.mathworks.com [144.212.95.234])

by mail-vif.mathworks.com (8.11.7/8.11.7) with ESMTP id j4CCVOH06340

for <destinatario@unipr.it>; Thu, 12 May 2005 08:31:25 -0400 (EDT)

Message-ID: <74127410.1115901084851.JavaMail.wwwadmin@telesto.mathworks.com>

Date: Thu, 12 May 2005 08:31:24 -0400 (EDT) #Nota: EDT US East Coast

From: sender@mathworks.com

User-Agent: Mozilla Thunderbird 1.0.2 (Windows/20050317)

Reply-To: sender@mathworks.com

To: destinatario@unipr.it

Subject: Greetings

Greetings from Mathworks

bye

MIME

Il formato RFC822 del 1982 era stato pensato per messaggi in cui corpo era esclusivamente testo espressi in ASCII standard.

Questo schema non ammette lettere accentate, alfabeti non latini, messaggi multimediali ecc.

La soluzione è stata proposta da MIME RFC1341, oggi ampiamente utilizzata in Internet.

MIME (Multipurpose Internet Mail Extensions) introduce 5 nuove intestazioni:

- 1) **MIME-version**
- 2) **Content-description**
- 3) **Content-id**
- 4) **Content-transfer-encoding**. Il nome dello schema di codifica utilizzato per trasformare il messaggio in ASCII standard. Principali valori:

Ascii 7bit: nessuna codifica. Linee fino a 1000 caratteri

Ascii 8bit: viola la versione originale del protocollo, ma probabilmente funziona.

Quoted-printable: messaggi ASCII non standard. I caratteri superiori al 127 sono codificati con = seguito dal codice ASCII in esadecimale. (città -> citt=9A)

base64: Per dati binari. Ogni sequenza di 6 bit viene trasformato in un carattere ASCII grazie ad una codifica di 64 simboli (sprecati 2 bit ogni 6, i dati codificati occupano il 35% in più)

Esempio di codifica di una immagine: `openssl base64 -e -in immagine.png -out immagine.b64`

Esempio di decodifica: `openssl base64 -d -in immagine.b64 -out immagine.png`

5) Content-type. Natura del corpo del messaggio

Espresso nella forma type/subtype (esempio Content-Type: text/plain)

Utile per attivare automaticamente il Viewer corretto (esempio video/mpeg)

Tipi e sottotipi sono definiti da IANA - <http://www.iana.org/assignments/media-types/index.html>

Principali Content-types

Type	Subtype	Description
Text	Plain	Unformatted text
	Enriched	Text including simple formatting commands
Image	Gif	Still picture in GIF format
	Jpeg	Still picture in JPEG format
Audio	Basic	Audible sound
Video	Mpeg	Movie in MPEG format
Application	Octet-stream	An uninterpreted byte sequence
	Postscript	A printable document in PostScript
Message	Rfc822	A MIME RFC 822 message
	Partial	Message has been split for transmission
	External-body	Message itself must be fetched over the net
Multipart	Mixed	Independent parts in the specified order
	Alternative	Same message in different formats
	Parallel	Parts must be viewed simultaneously
	Digest	Each part is a complete RFC 822 message

Esempio messaggio con MIME

Received: from ...

Message-ID: ...

Date: ...

From: ...

To: ...

Subject: ..

MIME-Version: 1.0

Content-Type: multipart/mixed; boundary="-----ThIs-RaNdOm-StRiNg-/_=.468328521:"

-----ThIs-RaNdOm-StRiNg-/_=.468328521:

Content-Transfer-Encoding: 7bit

Content-Type: text/plain

Ecco l'allegato

ciao

-----ThIs-RaNdOm-StRiNg-/_=.468328521:

Content-Transfer-Encoding: base64

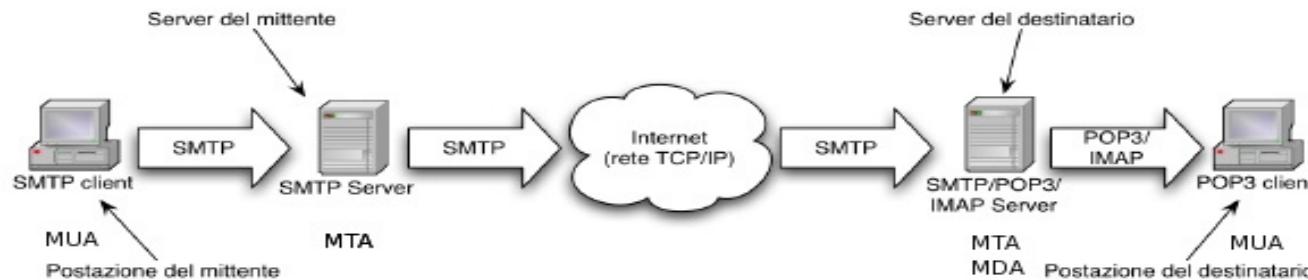
Content-Type: application/msword

BgAAAAAAAGYGAAAAAAAABIAwAAPAEAAIBAAAAAAAASAMAAAAAAAACAQAAAAAAEgD
AAAAAAA6gkAAAAAAAAAAAAAGYGAaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
AAAAAAAASAMAAAAAAADqCQAAAAAAAGYGAABIwAAZgYAAAAAAAAAAAAA
AAAAAK4JAAAAAAAAGEAAAAAAACAQAAAAAAAASAMAAAAAAADqCQAAAAAAAGYGAABIwAAZgYAAAAAAAAAAAAA
-----ThIs-RaNdOm-StRiNg-/_=.468328521:--

SMTP

SMTP è un protocollo applicativo (25/TCP) che si occupa del trasferimento di un messaggio da MUA a MTA o da MTA a MTA. L'MTA può essere destinatario finale (MTA+MDA) o di trasferimento (Mail Relay, Mail scanner, ...).

Il protocollo è codificato **ASCII Standard** ovvero prevede uno scambio di dati testuali.



Principali **comandi** del client SMTP:

HELO: indirizzi dei destinatari

MAIL FROM: mittente del messaggio

RCPT TO: destinatario del messaggio

DATA: corpo del messaggio

QUIT: fine del messaggio

RSET: reset

HELP: nome del comando

Principali **risposte** del server SMTP:

220: Servizio pronto

250: Comando richiesto completato

251: Utente non locale, il messaggio sarà inoltrato

221: Chiusura canale di trasmissione

421: Servizio non disponibile

500: Errore di sintassi

501: Errore di sintassi nei parametri

554: Transazione fallita

Formato della Mailbox

Esistono 2 formati principali:

Mbox

I messaggi ricevuti sono accodati in un singolo file per ogni utente

Su Unix si trova in /var/spool/mail/utente

Ogni messaggio inizia con una linea

“From sender@domain..”

From sender1@domain1
Messaggio1

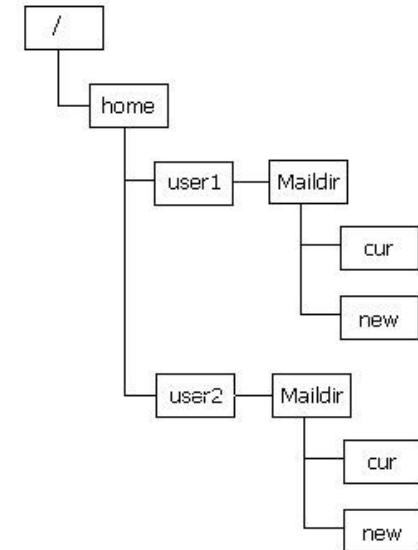
From sender2@domain2
Messaggio2

...

Maildir

Si utilizza una directory per ogni utente

All'interno della directory viene creato un file (di testo) per ogni messaggio ricevuto.



MUA : Mail User Agent

Un MUA è una applicazione che viene usata dall'utente per inviare e ricevere e-mail.

comando «mail»

MUA di base dei sistemi Linux. Non gestisce allegati MIME, POP e IMAP.
E' utile per l'invio automatico di messaggi all'interno di processi.

L'invio è affidato all'MTA su localhost. Esempio di utilizzo:

```
mail -s test user@domain<< EOF  
prova di spedizione  
EOF
```

comando «alpine»

MUA testuale, ma con gestione dello schermo. Software libero per Linux e MacOS.
Gestisce gli allegati MIME, può gestire la Mailbox da remoto con il protocollo IMAP

«Mozilla Thunderbird»

E' un MUA open software con interfaccia grafica disponibile per Windows, Linux e MacOS. Gestisce POP3/ IMAP e un filtro bayesiano anti spam.

Esempio di dialogo SMTP con telnet

telnet localhost 25 ...

Client

```
HELO      client.com  
MAIL FROM: <src@com>  
RCPT TO:   <dest@com>  
DATA
```

```
From: tizio  
To: caio  
Subject: prova
```

```
testo  
testo
```

MESSAGGIO

```
QUIT
```

Server

```
220  srv.com SMTP service ready  
250  client.com OK  
250  sender src@com OK  
250  Recipient dest@com OK
```

```
250  Message Accepted  
221  client.com closing connection
```

Un MTA è un server che accetta nel campo RCPT TO indirizzi diversi dall'host locale.

ESMTP (Extended SMTP)

Visto che con il passare degli anni vengono richieste sempre nuove funzionalità ad SMTP, con l'RFC 1869 del 1995 è stata definita una generale struttura standard di SMTP, denominata ESMTP, in grado di gestire le estensioni presenti e future.

Per utilizzare ESMTP occorre presentarsi con EHLO (anziché HELO). Se EHLO è accettato il server risponde con la lista delle estensioni supportate. Esempio:

```
> EHLO client.com
250-8BITMIME          (8 bit data transmission)
250-SIZE              (Message Size Declaration)
250-DSN               (Delivery Status Notification)
250-AUTH              (SMTP autenticato)
250-STARTTLS          (comunicazione cifrata con StartTLS)
....
```

Le estensioni più interessanti sono AUTH e STARTTLS, che introducono autenticazione e cifratura dei dati in fase di sottomissione del messaggio.

Sicurezza dell' SMTP

Poiché il protocollo SMTP non prevede autenticazione chiunque può contattare un MTA per

- spedire mail verso chiunque (**spam**)
 - Ingannare la vittima per indurlo a rivelare informazioni sensibili o svolgere azioni dannose (**phishing**)
 - furto di identità, ovvero impersonare l'identità di altri (**spoofing**)
 - Sfruttare gli allegati della mail per diffondere malware (virus, trojan, worm, ecc)
- Inoltre, poichè manca anche la riservatezza le mail sono esposte ad **intercettazione**.

Alcune semplici tecniche per mitigare:

- Filtro indirizzi IP sul server SMTP per accettare esclusivamente messaggi in cui il mittente o il destinatario sono locali.
- porte 25 bloccate dal firewall perimetrale per forzare l'utilizzo di MTA istituzionali (ad esempio dotati di antivirus e antispam).
- Verifica della registrazione DNS diretta e inversa del client
- Utilizzo del Resource Record SPF ([Sender Policy Framework](#)) sul DNS per identificare quali sono gli indirizzi IP abilitati a spedire per il nostro dominio.

Sicurezza ESMTP

Sottomissione dei Messaggi con SMTP-AUTH

Extended SMTP è una estensione (RFC 1869) che include nuove funzionalità, tra cui l'autenticazione con SMTP-AUTH per la fase di sottomissione di e-mail attraverso l'introduzione di un MTA dedicato denominato MSA (Message Submission Agent)

Tramite l'estensione SMTP AUTH un MSA può:

- richiedere le credenziali (user/pass o certificati) del client
- cifrare le comunicazioni tra MUA e MSA
- utilizzare una porta di ascolto diversa: 587/tcp

La cifratura SMTP-AUTH riguarda solo la consegna del messaggio all'MSA. Se si vuole una cifratura end-to-end occorre utilizzare una estensione di MIME denominata Secure MIME (S/MIME).

SMTP server

Il server SMTP ha il compito di ricevere e gestire messaggi di posta elettronica.
Opportunamente configurato può svolgere la funzione di

- MSA se riceve messaggi dal MUA offrendo servizi di autenticazione e riservatezza
- MTA se invia il messaggio ricevuto verso altri server SMTP
- MDA se gestisce caselle di posta in cui deposita i messaggi ricevuti

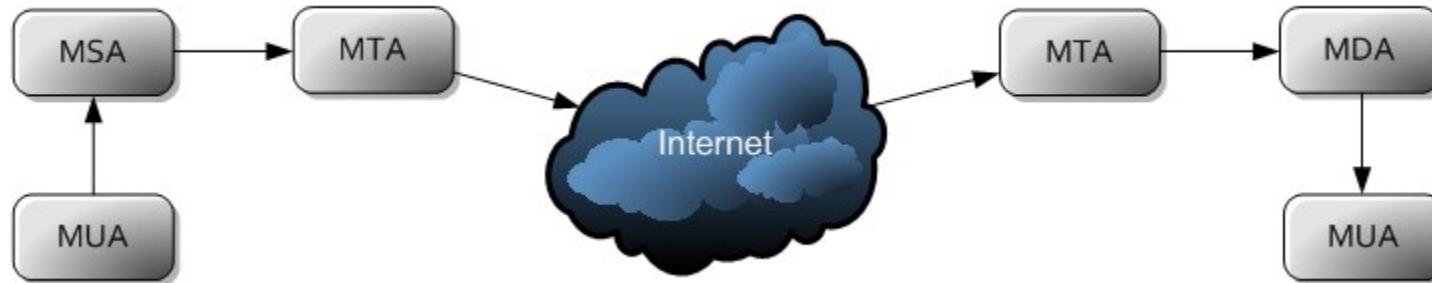
Esistono diverse distribuzioni open source di ESMTP server; le più diffuse sono Sendmail e Postfix.

Lettura dei Messaggi

La consegna del messaggio della mailbox utente è gestita dall'MDA (Message Delivery Agent) che ha anche il compito di consegnare il messaggio al MUA dell'utente previo opportuno meccanismo di autenticazione.

Quando è nato il protocollo SMTP gli utenti lavoravano sulla stessa macchina dove risiedevano le Mailbox, che quindi il MUA poteva accedere direttamente.

Con l'avvento dei PC il MUA si è disaccoppiato dall'MDA ed è nata la necessità di nuovo protocollo di rete per la comunicazione MUA-MDA. Due possibili protocolli: POP3 o IMAP.



POP3

POP3 (Post Office Protocol v.3, RFC 1939) è un protocollo ASCII con autenticazione per il trasferimento dei messaggi dal MailBox allo User Agent utilizzando un servizio TCP sulla porta 110.

Dopo la connessione TCP il protocollo attraversa 3 fasi:

Autenticazione: invio delle credenziali (USER e PASS)

Transazioni: Esecuzione dei comandi (LIST, RETR, DELE, QUIT)

Aggiornamento: Dopo il QUIT il server cancella effettivamente i messaggi eliminati e interrompe la connessione

POP3 è utilizzato tipicamente da Home Users, connessi via modem o ADSL all'ISP, per trasferire (RETR) tutti i nuovi messaggi, che vengono poi eventualmente cancellati dal server (DELE). Il MailBox server funziona così da area di transito per i messaggi, che vengono gestiti sull'Hard Disk dell'utente.

Riferimenti: http://openskill.info/release/guida_ai_protocolli_internet/i_protocolli_pop3_e_imap.htm

Poiché tutte le comunicazioni, incluse le credenziali di autenticazione, avvengono in chiaro, a POP3 è stato affiancato **il protocollo sicuro POP3S** in cui la comunicazione è cifrata grazie all'utilizzo del layer SSL/TLS. POP3S utilizza una porta diversa, la 995/TCP.

Esempio di dialogo POP3 con telnet

telnet localhost 110 ...

Client

USER nomeutente

PASS password

STAT

LIST

TOP 1

TOP 2

RETR 2

DELE 2

QUIT

Server

+OK POP3 ready

+OK

+OK

+OK 2 1000 #(2 mess – 1000 bytes)

+OK

1 500

2 500

.

mostra intestazione messaggio 1

mostra intestazione messaggio 2

recupera messaggio 2

.....

.....

.

+OK Marked to be deleted

+OK Logging Out, message deleted

IMAP

IMAP (Internet Message Access Protocol, RFC 2060), è un protocollo alternativo a POP3 per consentire all'user agent la gestione dei messaggi ricevuti, utilizzando il servizio TCP sulla porta 143.

A differenza di POP3 presume che i messaggi debbano rimanere sul server. Per questo fornisce la possibilità di gestire cartelle di posta sul server in cui archiviare i messaggi ricevuti.

IMAP è adatto per utenti che accedono alla posta utilizzando diversi MUA (casa, lavoro, portatile,...).

Riferimenti: http://openskill.info/release/guida_ai_protocolli_internet/i_protocolli_pop3_e_imap.htm

Poiché tutte le comunicazioni, incluse le credenziali di autenticazione, avvengono in chiaro, a IMAP è stato affiancato il **protocollo sicuro IMAPS** in cui la comunicazione è cifrata grazie all'utilizzo del layer SSL/TLS. IMAPS utilizza una porta diversa, la 993/TCP.

Esempio di dialogo IMAP con telnet

telnet localhost 143 ...

Client

a login <username> <password>
a list "/*" /* * *

Server

OK Dovecot ready
OK Logged in

LIST "/" saved-messages
LIST "/" sent-mail-feb-2018
LIST "/" sent-mail
LIST "/" INBOX
OK List completed.

a examine inbox

OK [PERMANENTFLAGS ()] Read-only mailbox.
6 EXISTS
0 RECENT
OK [UNSEEN 2] First unseen.
OK [UIDVALIDITY 1520015846] UIDs valid
OK [UIDNEXT 7] Predicted next UID
OK [READ-ONLY] Examine completed (0.004 secs).

a logout

BYE Logging out
OK Logout completed.
Connection closed by foreign host.



UNIVERSITÀ
DI PARMA

DIPARTIMENTO DI SCIENZE MATEMATICHE, FISICHE ED INFORMATICHE
Corso di Laurea in Informatica

Il Livello Applicativo – Parte C

WWW

RETI DI CALCOLATORI - a.a. 2023/2024

Roberto Alfieri

Livello Applicativo: sommario

PARTE A

- ▶ Applicativi UDP: TFTP e DNS

PARTE B

- ▶ I servizi di posta elettronica: SMTP, POP e IMAP.

PARTE C

- ▶ Il World Wide Web

PARTE D

- ▶ Multimedia

World Wide Web

WWW (World Wide Web) è una architettura client/server per la consultazione di documenti multimediali e ipertestuali distribuiti in rete.

L'architettura è nata nel 1989 al CERN di Ginevra e dal 1994 il suo sviluppo è gestito dal consorzio **W3C** (accordo CERN-MIT).

HTML (HyperText Markup Language) è il formato con cui vengono descritti gli ipertesti.

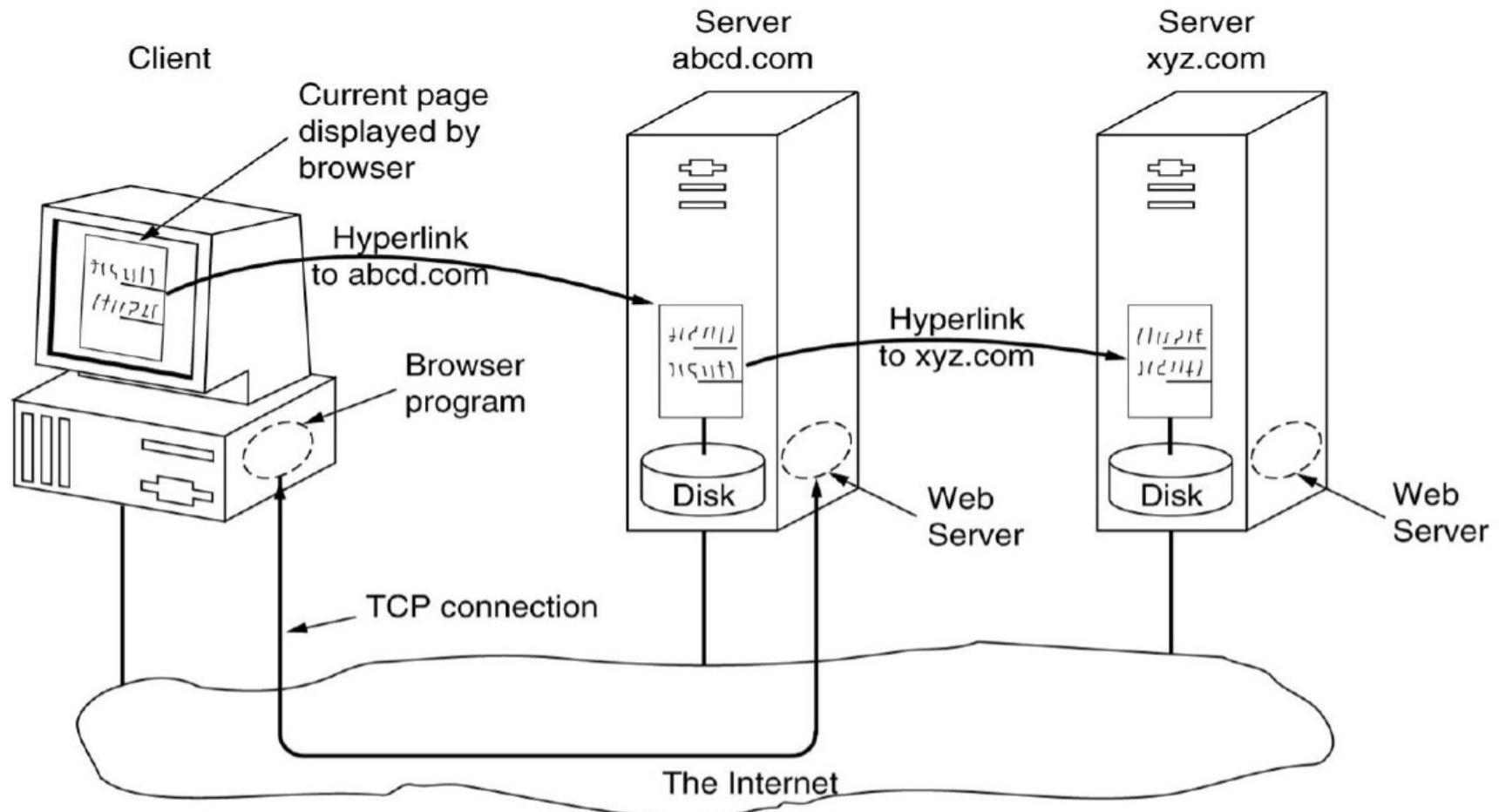
HTTP (HyperText Transfer Protocol) è il protocollo principale per la comunicazione tra **client** e **server**, anche se l'architettura WWW consente l'utilizzo di protocolli diversi.

Ogni documento WWW o singolo oggetto multimediale (file audio, immagine, ..) è identificato mediante un indirizzo univoco in Internet (**URL**) e può contenere riferimenti ipertestuali ad altri documenti.

I documenti possono essere statici (già presenti sul server al momento della richiesta) o dinamici (generati al momento della richiesta dal server o dal client)

L'utente accede ai documenti fornendo l'URL al programma client (Web Browser).

Architettura WWW



URL

Gli URL (Uniform Resource Locator) sono identificatori univoci di documenti WWW.
Sono composti da 4 parti schema://NomeServer:port/NomeLocale

- **schema** è il protocollo per raggiungere il documento
- **NomeServer** è il nome DNS del server che contiene il documento
- **Port** è la porta di ascolto del server
- **NomeLocale** è l'identificatore del documento sul server

Ad esempio: <http://www.company.com:81/a/b/c.html>

http è il protocollo più utilizzato per l'accesso a documenti WWW.

Altri schema diffusi sono:

- ✓ https è la versione cifrata del protocollo http
- ✓ ftp (vedi ad esempio qui: <https://www.gnu.org/prep/ftp.html>)
- ✓ file (esempio <file:///C:/> identifica i file in C:)

Riferimenti:

https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Identifying_resources_on_the_Web

Tipi Mime

Oltre ai documenti ipertestuali HTML l'architettura WWW supporta un numero sempre crescente di altri formati, denominati MIME-Types

Elenco di Mime-Type comuni:

https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types/Common_types

Per ogni Mime-Type il Browser avrà associata la modalità di gestione

Per alcuni formati il Browser conterrà l'interprete necessario per visualizzarli, per altri formati si appoggerà a componenti software esterne, i Plug-in o gli helper:

- ✓ - Un plug-in è un modulo esterno che il browser installa come estensione di se stesso
- ✓ - L'Helper (o applicazione di supporto) è un programma autonomo eseguito come processo separato a cui il browser passa il file da visualizzare.

I Web Client

Prevalentemente **browser grafici** (Firefox, Chrome, IExplorer, ecc)

Dispongono di una Cache su disco per i documenti visitati di recente

Sequenza di operazioni:

- ▶ Input dell'URL
- ▶ Verifica se il documento è presente in Cache
- ▶ Risoluzione del nome DNS nell'indirizzo IP
- ▶ Apertura della connessione TCP
- ▶ Invio della richiesta al server mediante il protocollo indicato nello schema
- ▶ Download del documento
- ▶ Parsing del documento
- ▶ eventuale richiesta di documenti collegati
- ▶ Visualizzazione del documento direttamente o mediante viewer esterni
- ▶ Rilascio della connessione TCP (se non vi sono altre richieste entro breve termine)

Un client Web può essere utilizzato per il **download di documenti**.

Esempi in ambiente Linux o MacOsX: wget e curl

I Web Server

E' un programma che si occupa di fornire, su richiesta del client browser, una pagina WWW.

The Apache Software Foundation è il nome di un gruppo di lavoro
(<http://www.apache.org/>) che sta portando avanti diversi progetti Open Source tra cui due
tra i più diffusi Server Web: **Apache** e **Tomcat**.

Altri server meno diffusi: **NGINX**

Sequenza di operazioni base del server:

- ▶ Accetta la connessione TCP da un client
- ▶ Determina dall'URL il percorso del documento richiesto
- ▶ Accede al documento su disco
- ▶ Invio al client di intestazione (Mime-Type, ..) e contenuto del documento
- ▶ Rilascio della connessione TCP

Web Caching

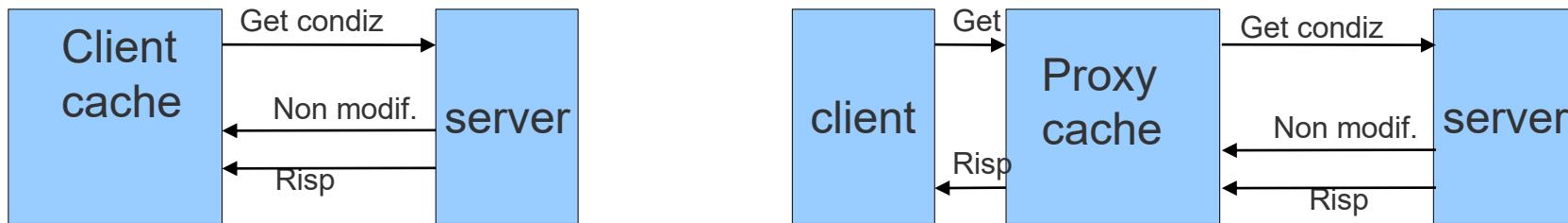
Fare Caching significa memorizzare temporaneamente le pagine in un punto più vicino al client per una **visualizzazione più rapida** e per **ridurre il carico del server**.

I **Browser** operano automaticamente caching sul disco locale dei documenti visitati.

Se il documento richiesto è presente sul disco locale viene fatto un GET condizionale (if-modified-since) in cui si chiede se esiste una versione più recente del documento.

Esempio:

If-Modified-Since: Tue, 17 Mar 2015 07:00:00 GMT



Una LAN può organizzare un servizio di caching disponibile per tutti gli utenti della rete.
Per utilizzare il servizio l'utente deve attivarlo esplicitamente.

Questo servizio di cache viene denominato **Proxy** poiché l'accesso al server che contiene il documento richiesto viene realizzato da Proxy/Cache server.

Il **Reverse Proxy** è un tipo di Proxy che recupera i contenuti per conto di un client da uno o più server. Questi contenuti sono poi trasferiti al client come se provenissero dallo stesso Reverse Proxy, che quindi appare al client come un Web server.

Cache Expiration

Alcuni documenti HTML possono contenere un'intestazione “Expires” che indica il tempo di validità del documento e verrà utilizzata per decidere se utilizzare la copia o recuperare il documento dal server.

```
<head>
<META HTTP-EQUIV="expires" CONTENT="Mon, 24 Mar 2008 08:21:57 GMT">
</head>
```

Altri documenti usano l'intestazione “no-cache” per impedire che il documento venga inserito nella cache (tipicamente pagine dinamiche).

```
<head>
<META HTTP-EQUIV="Pragma" CONTENT="no-cache">
</head>
```

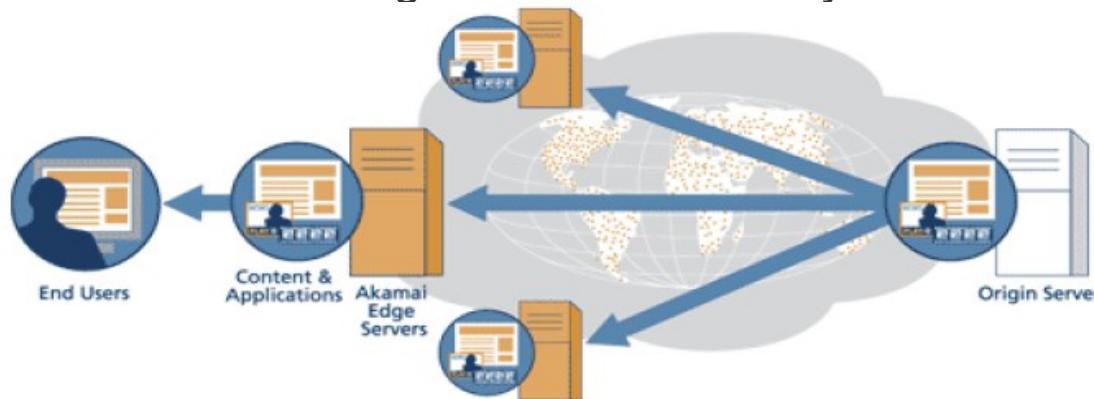
Content Delivery Network (CDN)

Per contenuti che devono essere distribuiti su scala globale (downloads, streaming, ..) esistono 2 approcci: Web caching e CDN.

Con il Web caching è il client a attivare le copie mentre con CDN è il provider che distribuisce copie dei contenuti in un insieme di nodi in differenti posizioni e redirige i client in modo che usino un nodo a lui vicino.

Possibili architetture per realizzare una rete CDN:

- **DNS redirection:** Il name server della CDN è gestito dalla CDN stessa. Il client ottiene dal DNS l'indirizzo Unicast della copia più vicina a lui.
- **Routing Anycast:** i mirror server hanno gli stessi indirizzi Anycast e il compito di trovare il server più vicino



Una delle reti CDN più note è Akamai, con oltre 29000 server sparsi su circa 70 nazioni.

Riferimenti: <https://www.akamai.com/it/our-thinking/cdn/what-is-a-cdn>

HTML

La maggior parte dei documenti WWW sono scritti in HTML.

HTML è un linguaggio di markup, ovvero contiene direttive di formattazione.

I contenuti sono testo formattato (font, colore, liste, tabelle, ecc) e eventuali riferimenti a oggetti esterni (immagini, video, suoni, hyper-link ad altri documenti)

Le direttive per la formattazione sono dette TAG e sono racchiuse tra parentesi angolari (esempio **< b >** testo evidenziato **< /b >**)

Il documento HTML è delimitato dal TAG **< html > .. < /html >** e comprende una intestazione **< head > .. < /head >** e un corpo **< body > .. < /body >**

Esempio:

```
<html>
<head>
<title> Le informazioni in Head non appaiono nel documento </title>
</head>
<body>
<b> Nel Body viene scritto il testo formattato del documento </b> <p>
<a href="http://www.unipr.it"> Questo e' un collegamento ipertestuale </a>
<p>
Le immagini vengono incluse nel documento nel seguente modo: <p>

</body>
</html>
```

Standard HTML

HTML continua ad evolversi.

HTML 2.0 è il primo vero standard di HTML rilasciato nel 1995 da **W3C**

HTML 3.2, rilasciato nel Gennaio 1997, è stato adottato in diversi browser.

HTML 4.01, rilasciato nel 1999 da W3C, è un'altra release con molte implementazioni

HTML 5, rilasciato da W3C nel 2014.

I Browser e Server Web hanno generalmente implementato le varie release HTML in modo più o meno rigoroso, sorvolando spesso su errori di formattazione e in alcuni casi, aggiungendo TAG non standard, funzionanti solo su alcune piattaforme.

XHTML è un linguaggio di markup che associa alcune proprietà dell'XML nell'HTML.

Un file XHTML è un pagina HTML scritta in conformità con lo standard XML.

Style Sheet

HTML è un linguaggio di Markup che mescola il contenuto alla formattazione. Infatti alcuni TAG descrivono il contenuto del documento, indipendentemente dalla sua rappresentazione finale (esempio `` oppure ``), mentre altri TAG descrivono il modo in cui il documento dovrà apparire al lettore (esempio ``)

Con la crescente complessità e varietà di utilizzo dei documenti Web è sorta la necessità di separare i 2 aspetti del documento. Per questo motivo sono stati introdotti i “Fogli di Stile” (Style Sheet) che sono file associati ad un documento in cui vengono confinate le informazioni di formattazione.

CSS (Cascading Style Sheet) sono i fogli di stile supportati da HTML introdotti nel 1996 da W3C.

Esempio: mystyle.css

```
.piccolo {font-style:normal; font-size:12px; }
```

Esempio: mydoc.html

```
<html>
<head> <link rel="stylesheet" type="text/css" href="mystyle.css"> </head>
<body> <div class="piccolo"> ciao </div> </body>
</html>
```

Il protocollo HTTP

E' un protocollo testuale di tipo request/response che utilizza il servizio 80/TCP.

Riferimenti: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>

Prima versione V0.9 rilasciata dal HTTP Working Group nel 1991 (obsoleto)

Versini successive: HTTP/1.0 rilasciato nel 1996 (obsoleto),

HTTP/1.1 del 1997 , HTTP/2 del 2015 e HTTP/3 del 2022

Deve trasportare un messaggio di richiesta dal client al server ed un messaggio di risposta dal server al client con il documento richiesto. Ogni messaggio è formato da una intestazione (Header) ed un corpo (Body) separati da riga vuota (CR LF).

<METHOD> <URI> HTTP/1.0

<Header>: <Value>

<Header>: <Value>

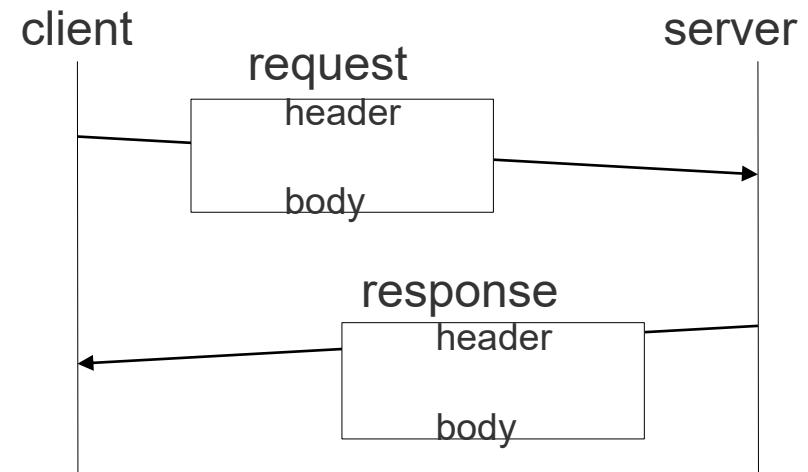
<BODY>

HTTP/1.0 <STATUS_CODE><REASON>

<Header>: <Value>

<Header>: <Value>

<BODY>



Metodi di richieste HTTP

GET : richiede tutte le informazioni disponibili per un determinato URL.
Il body del messaggio non è utilizzato.

```
GET /index.html HTTP/1.0
```

HEAD: Richiede solo l'header, senza la risorsa (il file HTML, l'immagine, ecc.).
Usato soprattutto per diagnostica.

```
HEAD /index.html HTTP/1.0
```

POST: è stato concepito in origine per inviare al server molte informazioni (nel Body della richiesta) , senza un limite sulla quantità di dati da trasmettere e sul tipo, ed in modo non visibile da URL.

```
POST /prog.cgi HTTP/1.0
Content-length: 10

01234566789
```

Metodi di richieste HTTP

OPTIONS: Richiede l'elenco dei metodi permessi dal server

```
OPTIONS * HTTP/1.0  
[riga vuota]  
...  
Allow: GET,HEAD,POST,OPTIONS,TRACE
```

TRACE: Traccia una richiesta, visualizzando come viene trattata dal server.

DELETE: Cancella una risorsa (file) sul server. L'utente con cui gira il web server deve poter avere permessi in scrittura sul file indicato e il server deve essere configurato per poterlo fare.

PUT: Upload di un file sul server con il nome indicato e i contenuti specificati nel body.

Principali intestazioni HTTP

Header	Type	Contents
User-Agent	Request	Information about the browser and its platform
Accept	Request	The type of pages the client can handle
Accept-Charset	Request	The character sets that are acceptable to the client
Accept-Encoding	Request	The page encodings the client can handle
Accept-Language	Request	The natural languages the client can handle
Host	Request	The server's DNS name
Authorization	Request	A list of the client's credentials
Cookie	Request	Sends a previously set cookie back to the server
Date	Both	Date and time the message was sent
Upgrade	Both	The protocol the sender wants to switch to
Server	Response	Information about the server
Content-Encoding	Response	How the content is encoded (e.g., gzip)
Content-Language	Response	The natural language used in the page
Content-Length	Response	The page's length in bytes
Content-Type	Response	The page's MIME type
Last-Modified	Response	Time and date the page was last changed
Location	Response	A command to the client to send its request elsewhere
Accept-Ranges	Response	The server will accept byte range requests
Set-Cookie	Response	The server wants the client to save a cookie
If-modified-since	Request	Allows a 304 Not Modified to be returned if content is unchanged.

http://en.wikipedia.org/wiki/List_of_HTTP_header_fields

Risposte del Server

La prima riga della risposta del server contiene un codice che classifica la risposta:

Code	Meaning	Examples
1xx	Information	100 = server agrees to handle client's request
2xx	Success	200 = request succeeded; 204 = no content present
3xx	Redirection	301 = page moved; 304 = cached page still valid
4xx	Client error	403 = forbidden page; 404 = page not found
5xx	Server error	500 = internal server error; 503 = try again later

Esempio di richiesta/risposta con telnet

Richiesta del Client:

```
telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^].
GET /index.html HTTP/1.0
[riga vuota]
```

Risposta del server:

```
HTTP/1.1 200 OK
Server: nginx/1.18.0 (Ubuntu)
Date: Thu, 24 Nov 2022 11:34:55 GMT
Content-Type: text/html
Content-Length: 612
Last-Modified: Wed, 23 Nov 2022 08:24:08 GMT
Connection: close
ETag: "637dd8a8-264"
Accept-Ranges: bytes

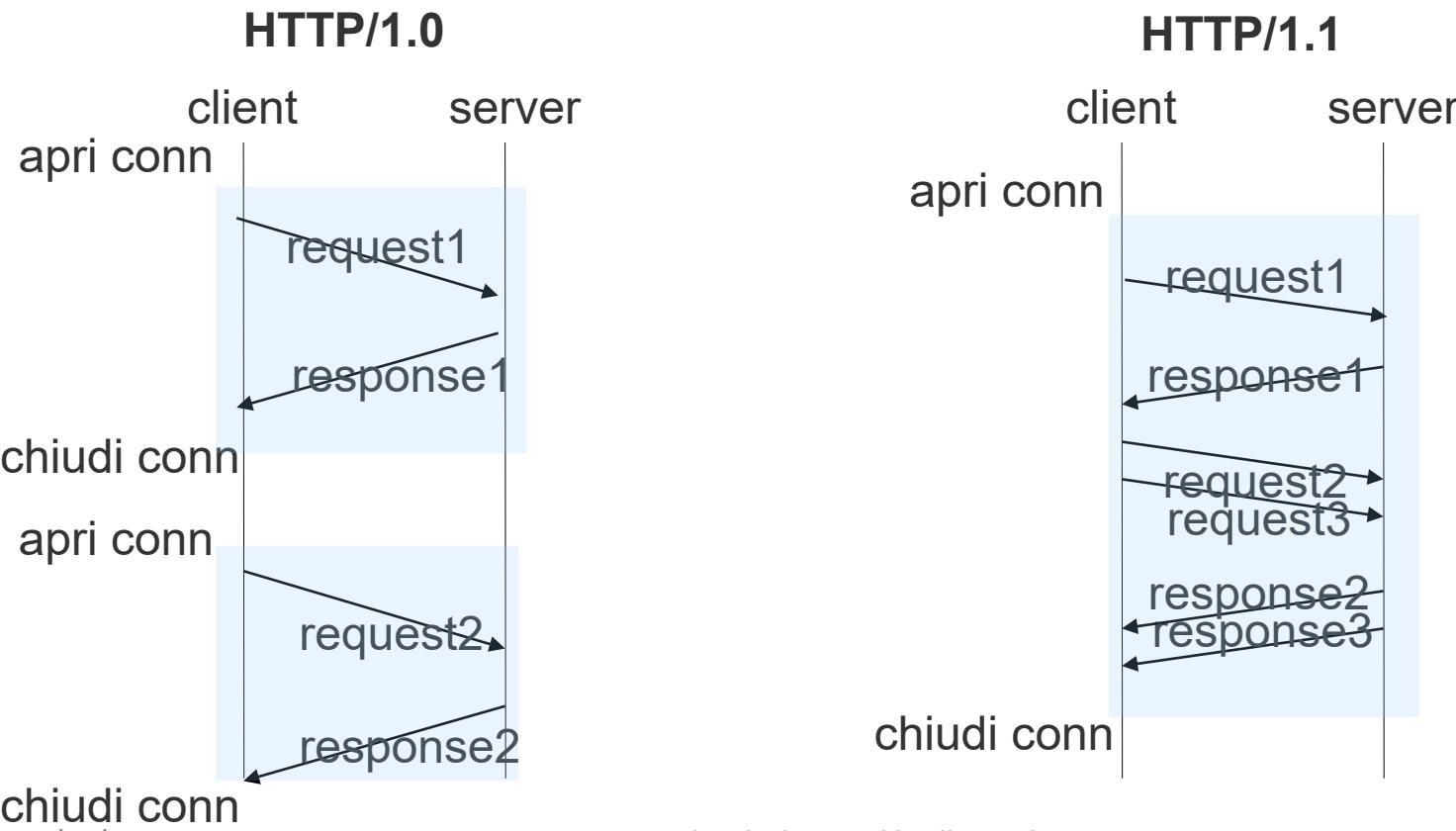
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully
working. Further configuration is required.</p>
```

HTTP 1.1: connessioni persistenti e parallele

HTTP 1.0 richiede la disconnessione TCP tra 2 richieste successive.

HTTP 1.1 supporta connessioni persistenti (riutilizzo di una conn. per diverse richieste)

HTTP 1.1 supporta anche richieste parallele



HTTP 1.1: Chunked Transfer Encoding

In HTTP 1.0 chi spedisce un dato include nell'intestazione la dimensione in byte del dato stesso, mediante l'intestazione **“Content-length: nn”**.

Chunked Transfer Encoding è una modalità di trasferimento introdotta in HTTP 1.1 in cui i dati vengono inviati in una serie di “chunk”.

Viene utilizzata per la trasmissione di dati generati dinamicamente, di cui non conosciamo la lunghezza prima di iniziare la trasmissione, come ad esempio gli eventi in streaming. Si utilizza **“Transfer-Encoding: chunked”** invece di “Content-length: nn”

La dimensione di ogni chunk viene inviata appena prima del chunk stesso in modo che il ricevente possa capire quando ha finito di ricevere il chunk.

Il trasferimento termina con un chunk finale pari a 0.

Riferimento:

http://en.wikipedia.org/wiki/Chunked_transfer_encoding

```
HTTP/1.1 200 OK
Date: Mon, 22 Mar 2004 11:15:03 GMT
Content-Type: text/html
Transfer-Encoding: chunked
```

```
29
<html><body><p>The file you requested is
5
3,400
23
bytes long and was last modified:
1d
Sat, 20 Mar 2004 21:12:00 GMT
13
.</p></body></html>
0
```

Mancanza di stato e Cookie

Il Web è privo di stato: se un Browser richiede più documenti da un server ogni richiesta è indipendente; il server non ricorda i contatti precedenti.

In alcuni casi però sarebbe utile avere “memoria”. Ad esempio se l’accesso ai documenti richiede autenticazione , autorizzazione, ecc.

I Cookie sono stati introdotti da Netscape e formalizzati in RFC 2109 per risolvere il problema.

I Cookie sono generati dal server e scaricati assieme al documento.

Il browser li memorizza in una opportuna directory, ma volendo li può disabilitare.

Informazioni contenute in un Cookie:

Domain	Path	Content	Expires	Secure
toms-casino.com	/	CustomerID=497793521	15-10-02 17:00	Yes
joes-store.com	/	Cart=1-00501;1-07031;2-13721	11-10-02 14:22	No
aportal.com	/	Prefs=Stk:SUNW+ORCL;Spt:Jets	31-12-10 23:59	No
sneaky.com	/	UserID=3627239101	31-12-12 23:59	No

Contenuto dei Cookie

Contenuto (Nome/valore) è una variabile ed un campo obbligatorio.

Expire (Scadenza) è un attributo opzionale che permette di stabilire la data di scadenza del cookie. Può essere espressa come data, come numero massimo di giorni oppure come Now (adesso) (implica che il cookie viene eliminato subito dal computer dell'utente in quanto scade nel momento in cui viene creato) o Never (mai) (implica che il cookie non è soggetto a scadenza e questi sono denominati persistenti).

Dominio e Percorso definiscono l'ambito di visibilità del cookie, indicano al browser che il cookie può essere inviato al server solo per il dominio e il percorso indicati. Se non specificati, come predefiniti prendono il valore del dominio e del percorso che li ha inizialmente richiesti.

Secure (Sicuro) indica se il cookie debba essere trasmesso criptato con HTTPS.

Come funzionano i Cookie

La prima volta che viene richiesto un URL il server invia i cookie al client inserendoli nell'intestazione, assieme al documento. Ad esempio:

```
HTTP/1.0 ....
```

```
Set-Cookie: tuocodice=1234567; expires=Tue, 18-Mar-08 18:43:09 GMT
```

```
Set-Cookie: tuonome=Mario; expires=Tue, 18-Mar-08 18:43:09 GMT
```

Il client memorizza i cookie ricevuti.

Quando l'utente torna a visitare la pagina il Browser cerca tra i Cookie che ha memorizzato un Cookie (non scaduto) con lo stesso dominio dell'URL.

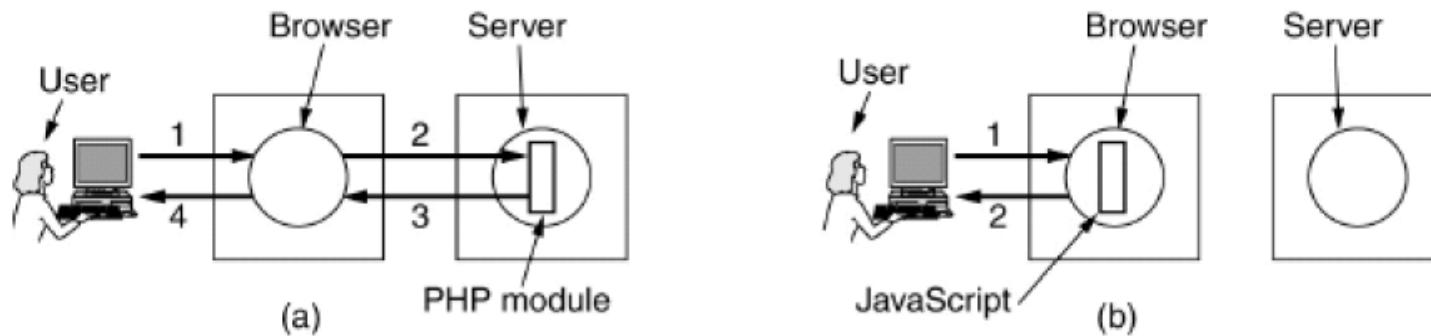
Se esiste viene fatto Upload del Cookie nell'Header assieme alla richiesta:

```
GET .... HTTP/1.0
```

```
Cookie: nome1=valore1 ; nome2=valore2
```

Pagine Web statiche e dinamiche

- ▶ **Le pagine statiche** sono file sul disco del server che vengono spediti al client assieme ad una intestazione (content-type ecc)
- ▶ **Pagine dinamiche:** il documento viene generato in tempo reale, su richiesta. La generazione è eseguita da un programma che può essere eseguito
 - **dal server**
 - via CGI (programmi esterni richiamati dal server)
 - scripting PHP, JSP, ASP (codice incorporato in HTML interpretato dal server)
 - **dal client (pagine attive)**
 - Javascript
 - Java Applet (richiede JVM sul client)
 - ActiveX (tecnologia Microsoft, codice compilato per Intel)



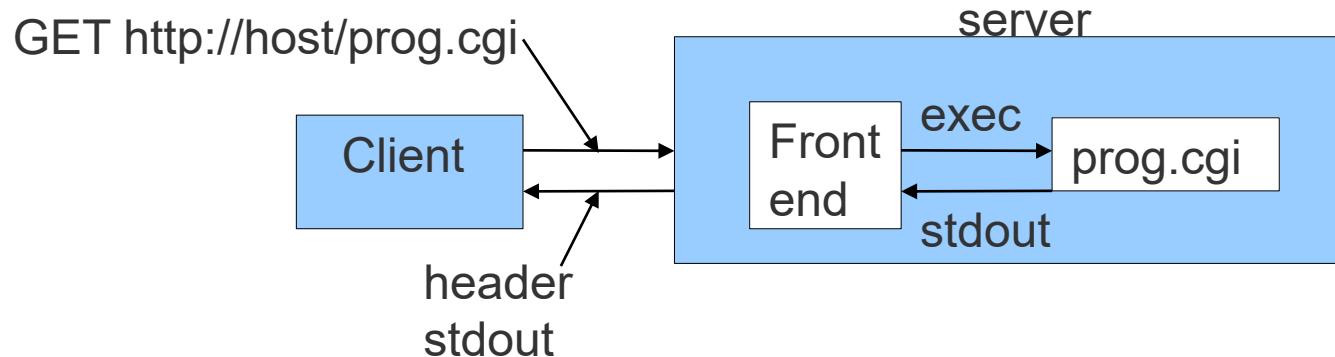
Pagine dinamiche con CGI

Il protocollo **CGI** (Common Gateway Interface, RFC 3875) consente di mettere in esecuzione un programma eseguibile sul server e di redirigere lo standard output del programma verso il client il quale lo interpreterà come una normale risposta http.

Per attivare un programma il client utilizza lo stesso modello URL utilizzato per il riferimento alle pagine statiche, con i metodi GET o POST.

Il server può riconoscere un programma cgi in base alla sua estensione (e.g. .cgi) o alla sua posizione (es /cgi-bin/..)

Normalmente l'output è in formato HTML, ma può assumere anche altre forme (immagini, dati binari, istruzioni particolari per il browser, ..)



Passaggio dei parametri

L'esecuzione di una pagina dinamica deve prevedere la possibilità di passare parametri o dati all'eseguibile.

Questo può avvenire con 2 metodi alternativi: GET e POST.

-**Il metodo GET** codifica i parametri all'interno della stringa URL

Esempio: `http://host/prog.cgi?param`

-**Il metodo POST** utilizza la parte Body della richiesta.

L'HTML fornisce diversi TAG per la codifica di parametri nella forma NOME=valore da passare con GET o POST

Ad esempio:

```
<FORM ACTION="http://host/prog.cgi" METHOD=GET>  
PARAM1: <INPUT TYPE="text" NAME="param1">  
PARAM2: <INPUT TYPE="text" NAME="param2" >  
<INPUT TYPE="submit" VALUE="Invia">  
</FORM>
```

PARAM1: val1	PARAM2: val2	Invia
--------------	--------------	-------

Genera la URL: `http://host/prog.cgi?param1=val1¶m2=val2`

Passaggio di dati con il metodo GET

Con il metodo GET la stringa di query (input) è inserita in coda alla URI del documento preceduta dal carattere “?”

GET http://host/prog.cgi?param1=val1¶m2=val2 HTTP/1.0

header..

header..

<cr><lf>

Il programma riceve la stringa attraverso la variabile di ambiente QUERY_STRING:

QUERY_STRING="param1=val1¶m2=val2"

Passaggio di dati con il metodo Post

I parametri passati con il metodo POST vengono inseriti nel Body della richiesta HTTP e il programma li riceve attraverso lo Standard Input.

Ad esempio, se scegliamo il metodo POST nella FORM dell'esempio precedente:

```
<FORM ACTION="http://host/prog.cgi " METHOD=POST> ...
```

Otteniamo una richiesta HTTP del tipo:

POST http://host/prog.cgi HTTP/1.0

header..

Content-length: 23

<cr><lf>

param1=val1¶m2=val2

Questo metodo ha 2 vantaggi nel passaggio dei parametri rispetto al metodo GET:

- ✓ i parametri non compaiono nella URI (e quindi non vengono tracciati nei file di log)
- ✓ è possibile trasferire non solo parametri, ma anche dati

Pagine dinamiche lato server con PHP

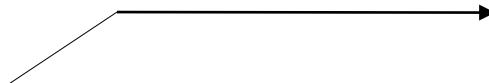
I server web supportano anche la possibilità di incorporare piccoli script all'interno del codice HTML che verrà eseguito al momento della consultazione della pagina.

Questo consente di realizzare documenti in cui solo una parte è dinamica.

Un linguaggio molto utilizzato per questo scopo è il PHP.

Esempio:

```
<html>
<head>
</head>
<body>
<FORM ACTION="http://host/prog.php" METHOD=GET>
PARAM1: <INPUT TYPE="text" NAME="param1">
PARAM2: <INPUT TYPE="text" NAME="param2" >
<INPUT TYPE="submit" VALUE="Invia">
</FORM>
</body>
</html>
```



```
<html>
<body>
<h1>Dati inseriti:</h1>
<? php echo $param1; ?>
<? php echo $param2; ?>
</body>
</html>
```

Pagine dinamiche lato client: javascript

In altre applicazioni è utile che il codice venga eseguito lato client.

Anche in questo caso viene incorporato codice di script nella pagina HTML.

Il linguaggio più popolare lato client è JavaScript. Esempio:

```
<html>
<head>

<script language="javascript" type="text/javascript">
Function response (test_form) {
...
}

</script>

</head>
<body>
<form>
PARAM1: <INPUT TYPE="text" NAME="param1">
PARAM2: <INPUT TYPE="text" NAME="param2" >
<input type="button" value="submit" onclick="response(this.form)">
</form>
</body>
</html>
```

Pagine dinamiche lato client: Applet

Un altro metodo molto diffuso è l'utilizzo di Applet Java.

E' necessario che il browser includa una JVM (quasi tutti i Browser)

Le Applet sono più portabili perché la JVM è la stessa su diverse piattaforme, mentre il supporto JavaScript può differire da un Browser all'altro.

Le Applet possono essere incorporate nelle pagine HTML:<applet> ... </applet>

Esempio:

```
<html>
<body>
<applet code="PrimoApplet.class">
</applet>
</body>
</html>

import java.applet.*;
import java.awt.*;

public class PrimoApplet extends Applet
{
    public void paint (Graphics g)
    {
        g.drawString("Ciao, io sono il primo applet.",0,50);
    }
}
```



UNIVERSITÀ
DI PARMA

DIPARTIMENTO DI SCIENZE MATEMATICHE, FISICHE ED INFORMATICHE
Corso di Laurea in Informatica

Il Livello Applicativo – Parte D

Multimedia

RETI DI CALCOLATORI - a.a. 2023/2024

Roberto Alfieri

Livello Applicativo: sommario

PARTE A

- ▶ Applicativi UDP: TFTP e DNS

PARTE B

- ▶ I servizi di posta elettronica: SMTP, POP e IMAP.

PARTE C

- ▶ Il World Wide Web

PARTE D

- ▶ **Multimedia**

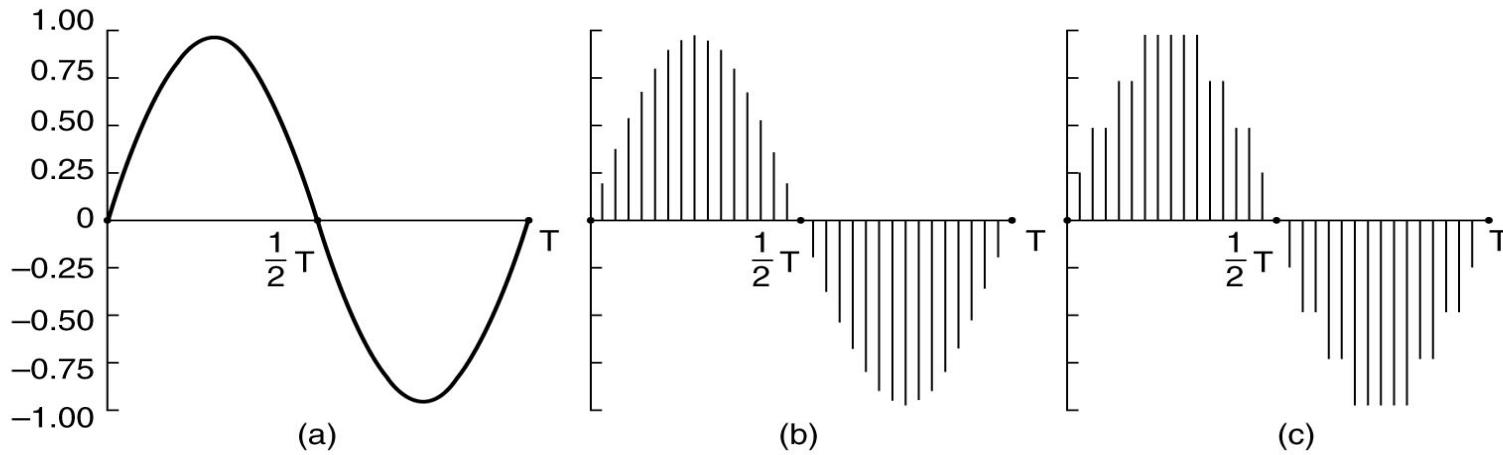
Multimedia

I dati multimediali hanno caratteristiche peculiari rispetto ai dati «classici» che utilizziamo nelle reti di calcolatori e hanno requisiti che richiedono un diverso QoS:

- ✓ I dati sono in origine analogici in 1 (audio) o 2 (video) dimensioni spaziali ed eventualmente una dimensione temporale e devono quindi subire una discretizzazione che introduce inevitabilmente una approssimazione.
- ✓ I dati possono essere pesanti (in particolare i video) e richiedono quasi sempre un processo di compressione/decompressione
- ✓ Sono tollerati (piccoli) errori di trasferimento, ma è richiesta una comunicazione con alto bit-rate e basso jitter.
- ✓ La rete internet, nonostante esistano tecniche per il supporto della QoS (vedi i Servizi Differenziati) attualmente è sostanzialmente best effort.
- ✓ Non esiste quindi un approccio standard per la gestione dei dati multimediali in rete ma vendono utilizzate in molti casi soluzioni proprietarie.

Audio Analogico e Digitale

Onda sonora percepita dall'uomo: onda monodimensionale con frequenze comprese tra 20 e 20.000 Hz (voce sotto i 3KHz).

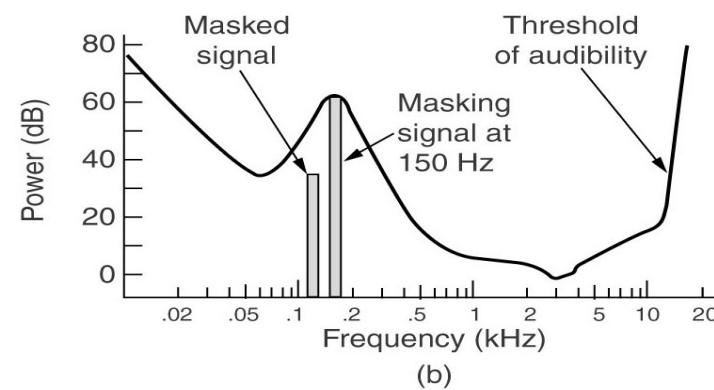
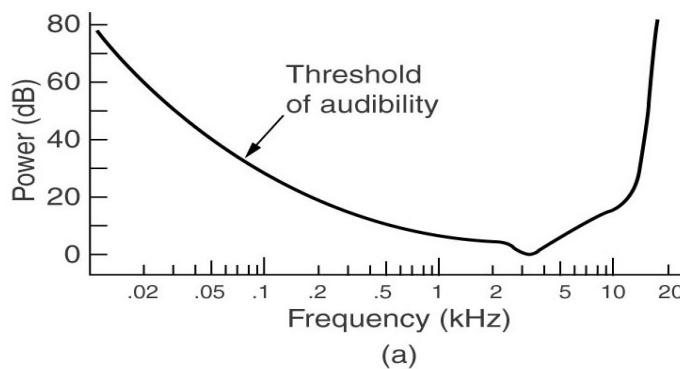


Conversione analogico/digitale:

- ▶ **La frequenza di campionamento è determinata dal teorema di Nyquist:**
- ▶ Freq. di campionamento = $2 * \text{Freq. Max del segnale da digitalizzare}$
- ▶ **quantizzazione:** numero discreto di valori possibili per le letture nel campionamento (8 bit = 256 valori, 16 bit = 65.536)
- ▶ Il sistema telefonico usa 8 bit per 8.000 volte al secondo (max 4KHz)
- ▶ CD audio: 44.100 campioni al secondo di 16 bit => 1,411Mbps (stereo)

Compressione Audio

- ▶ Per la trasmissione su internet è necessaria forte compressione.
- ▶ La compressione audio si può fare in 2 modi:
 - **Conversione per forma d'onda:** il segnale viene convertito, usando la trasformata di Fourier, nelle sue componenti nel dominio delle frequenze; ogni componente viene codificata con la minima quantità di bit
 - **Codifica percettiva:** sfrutta alcuni limiti del sistema uditivo umano (psicoacustica) per codificare il segnale in modo che sembri lo stesso ad un ascoltatore umano pur essendo diverso dal segnale originario, utilizzando la tecnica del mascheramento.



I formati MP3 e AAC eseguono la trasformata di Fourier, quindi codificano solo le frequenze non mascherabili.

CD audio: si arriva a circa 100 Kbps (1.4Mbps non compresso, Compressione 14:1)

Video digitale

- ▶ **Video digitale:** sequenza di fotogrammi, ognuno dei quali composto da una griglia rettangolare di elementi detti **pixel**
- ▶ La geometria è quella del video analogico con la differenza che le linee di scansione vengono sostituite da righe di pixel discreti
- ▶ Per avere un *movimento fluido* servono **almeno 25 fotogrammi** al secondo
- ▶ Per un flusso video alla risoluzione di 640x480, con 24 bit per pixel e 30 fotogrammi al secondo serve una linea di comunicazione a 200 **Mbps**

- ▶ La **compressione** è l'unica possibilità per riuscire ad inviare filmati video su internet.
- ▶ Servono due algoritmi: uno di **codifica** per la compressione all'origine, uno di **decodifica** per la decompressione alla destinazione
- ▶ Il sistema di codifica/decodifica può essere *irreversibile (lossy)*: a costo di una piccola **perdita di informazioni** si ottiene un fattore di compressione elevato

JPEG

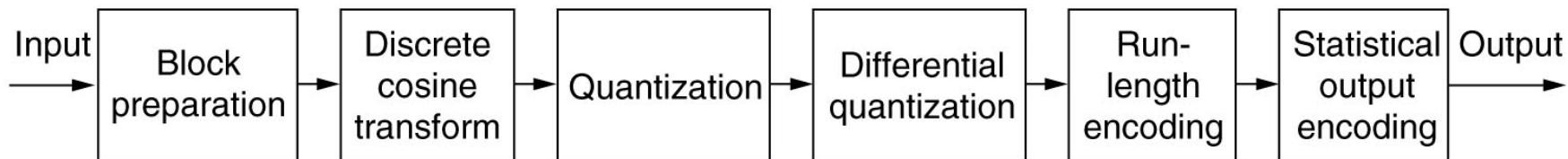
Un video è essenzialmente una sequenza di immagini con l'audio.

un algoritmo per la codifica Video è la codifica in successione di ogni singola immagine, con o senza perdite di informazione

Lo standard JPEG esegue la compressione di immagini statiche a toni continui (come le foto)

Processo di compressione:

- ▶ L'immagine viene suddivisa in blocchi di 8x8 pixel
- ▶ Per ogni blocco si passa dallo spazio tempo allo spazio delle frequenze tramite DCT
- ▶ vengono eliminati i coefficienti DCT meno importanti
- ▶ Si passa ad una rappresentazione differenziale di un blocco rispetto al blocco precedente
- ▶ Linearizzazione dei 64 elementi di un blocco
- ▶ Codifica di Huffman per assegnare codici più brevi ai numeri più frequenti

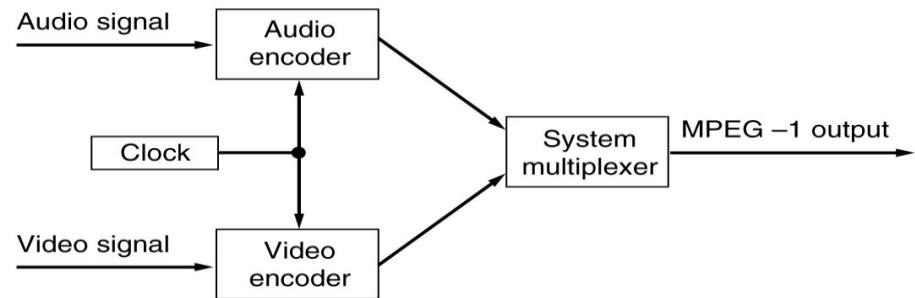


MPEG

E' un gruppo di lavoro (Motion Picture Experts Group) che dal 1993 si occupa di algoritmi e formati per le immagini in movimento.

- ▶ MPEG-1: pubblicato nel 1993, è ancora molto usato. Il suo scopo era quello di produrre un output di qualità simile a quella di un videoregistratore (352x240 per NTSC) usando un bit rate pari a 1,2Mbps con compressione 40:1 (richiederebbe circa 50,7Mbps non compressi)

- ▶ E' composto da tre parti:
 - Audio
 - Video
 - Multiplexer di sistema

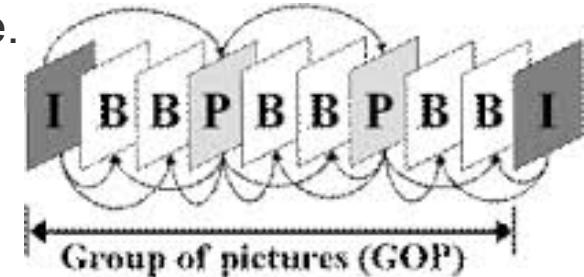


- ▶ Audio e video lavorano indipendentemente e sono sincronizzati usando un segnale comune a 90 KHz che emette il segnale del tempo corrente verso entrambi i codificatori
- ▶ **Ridondanza spaziale:** si usa la codifica JPEG per ogni singolo fotogramma. Utile soprattutto per accedere in modo casuale ad ogni singolo fotogramma
- ▶ **Ridondanza temporale:** si cerca di trarre vantaggio dal fatto che i fotogrammi consecutivi sono quasi identici

MPEG-1 e MPEG-2

MPEG-1 è composto da 3 tipi di fotogrammi:

- ▶ **I-frame** (Intracodificati): immagini statiche codificate in JPEG. Vengono inviate ad intervalli regolari (ad esempio ogni 10 frame).
- ▶ **P-frame** (Predittivi): E' una immagine che dipende dal frame precedente. Viene determinata calcolando le differenze blocco per blocco con l'ultimo fotogramma. Utile se non ci sono troppi cambiamenti rispetto alla precedente.
- ▶ **B-frame** (Bidirezionali): dipende sia dal precedente che dal successivo



MPEG-2: rilasciato nel 1996, è progettato per comprimere video con bit-rate compresi tra 4 e 6 Mbps, per essere inserito in trasmissioni NTSC e PAL per poi arrivare anche a risoluzioni superiori (HDTV)

- ▶ In prima approssimazione MPEG-2 è un superset di MPEG-1
- ▶ Utilizza fotogrammi I,P,B
- ▶ La trasformata DCT utilizza blocchi 10x10 invece di blocchi 8x8
- ▶ Supporta sia immagini progressive che interallacciate
- ▶ Supporta più livelli di risoluzione:

Low (352x240 compatibile MPEG-1), Main (720x480) High-1440 (1440x1152), High (1920x1080)

Dati multimediali in rete

Esistono diversi modalità di utilizzo del multimedia in internet, con diverse requisiti di rete:

Streaming di contenuti registrati (e.g. Youtube, Netflix)

- Numerosi flussi singoli. Riproduzione durante la ricezione.

Realtime streaming (e.g. radio Internet, dirette sportive)

- Ridurre il ritardo per minimizzare lo scostamento temporale rispetto alla trasmissione via etere.
- Centinaia o migliaia di utenti contemporanei

Conferenza in tempo reale (e.g. Skype)

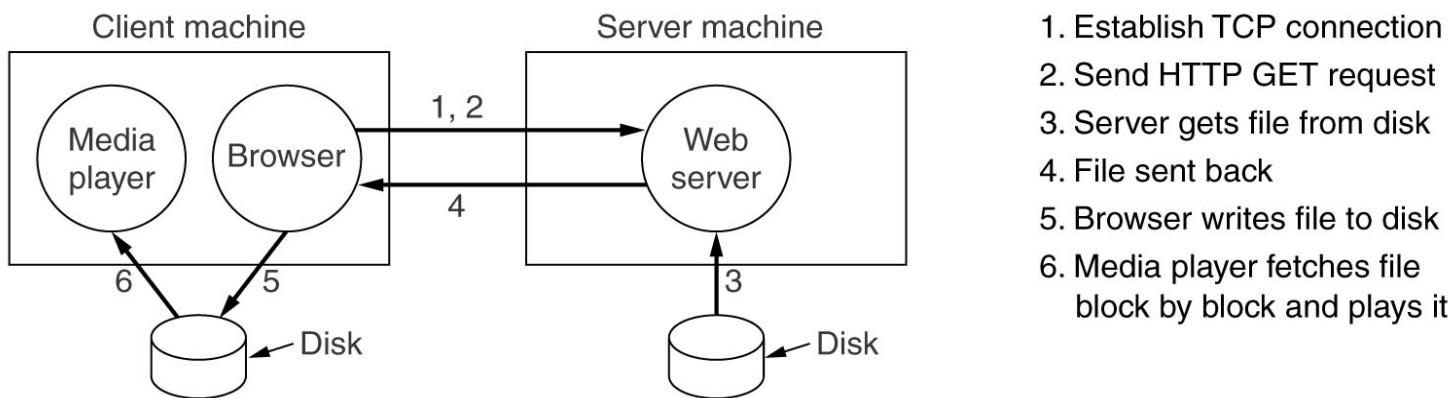
- L'interattività richiede latenze molto contenute.
- Numero di partecipanti tipicamente limitato ma in alcuni casi può essere elevato.

Streaming di contenuti registrati: Download and play

La tecnica più semplice per l'utilizzo di contenuti multimediali registrati su un server remoto è download and play:

il file deve essere scaricato completamente prima di poter essere riprodotto utilizzando il sistema "classico" del MIME

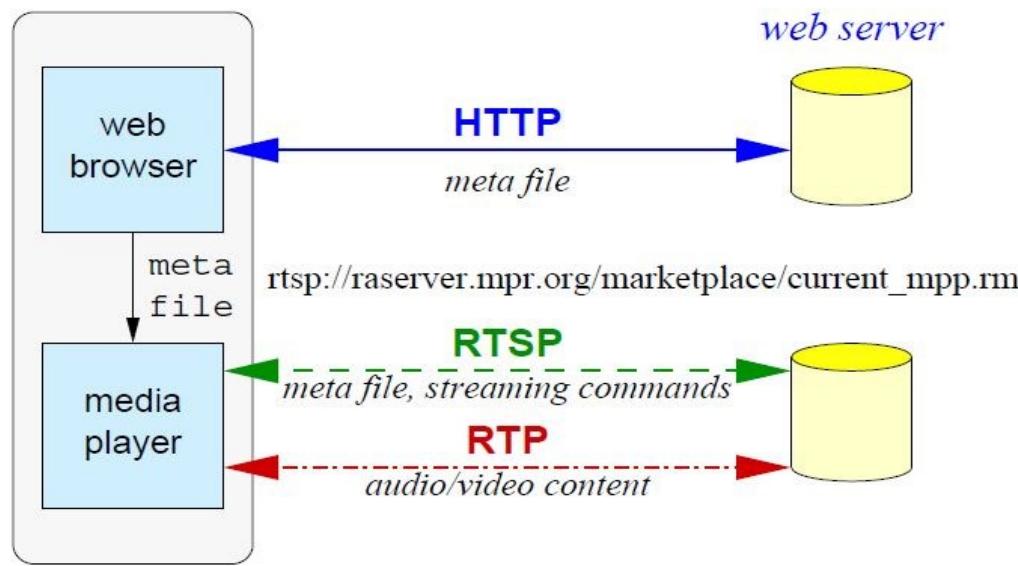
(esempio Content-type: video/mp4) e delle helper applications



Sistema poco efficiente: si deve scaricare il file prima di iniziare l'ascolto.

Streaming di contenuti registrati: Streaming on download

La soluzione generalmente adottata (Audio Streaming) è la seguente : il file collegato al titolo non è quello contenente l'audio ma un **metafile** che rimanda al file vero e proprio da ascoltare (il contenuto potrebbe essere la sola riga rtsp://server/file.mp3). Il browser, una volta passato il **metafile** all'applicazione esterna, non fa più parte del ciclo di comunicazione



L'accesso al file multimediale (e.g. `rtsp://server/file.mp3`) avviene mediante un protocollo per la gestione dell'interfaccia utente (Play/Record/Pause/ ecc) denominato Real Time Streaming Protocol (RTSP).

Il trasporto del flusso multimediale avviene su un canale separato basato sul protocollo RTP (Real-time Transport Protocol) o HTTP.

Real Time Streaming Protocol

- Il protocollo RTSP supporta un set di comandi che vengono inviati al server mediante un scambio testuale simile all'HTTP

Command	Server action
DESCRIBE	List media parameters
SETUP	Establish a logical channel between the player and the server
PLAY	Start sending data to the client
RECORD	Start accepting data from the client
PAUSE	Temporarily stop sending data
TEARDOWN	Release the logical channel

Vedi un esempio di dialogo: http://en.wikipedia.org/wiki/Real_Time_Streaming_Protocol

*Nota: YouTube usa HTTP/TCP. RTSP è comunque utilizzata per la rete dei cellulari
<http://m.youtube.com> I video sono memorizzati nella CDN di Google.*

Realtime Transport Protocol

RTP (RFC 3350) è un protocollo client/server per il trasporto di flussi anche multipli di dati audio e video. E' basato su UDP e può funzionare sia in unicast che in multicast.

I campi principali dell'intestazione RTP sono:

- Payload type: rende possibile l'identificazione del contenuto
(esempi: 26 → JPEG, 32 → MPEG1, 33 → MPEG2)
- Sequence Number: numerazione progressiva per il riordino
- Timestamp: istante di campionamento del primo byte nel payload.
- Synchronized Source ID: identificatore della sorgente di stream, per distinguere diversi flussi contemporanei.

Con l'aiuto di un piccolo buffer il ricevente può ricostruire il flusso nella sequenza temporale corretta e rifasare eventuali flussi contemporanei.

Payload Type	Sequence Number	Timestamp	Synchronization Source Identifier	Miscellaneous Fields
--------------	-----------------	-----------	-----------------------------------	----------------------

RTP Header

Alcuni fornitori preferiscono implementare protocolli di trasporto brevettati.

Ad esempio i server di Realnetworks utilizzano il protocollo di trasporto [Real Data Transport \(RDT\)](#) di proprietà della RealNetworks stessa.

Realtime streaming

Streaming in tempo reale di eventi (sportivi o musicali) a cui assiste un elevato numero di utenti.

Requisiti:

- **Ridurre il ritardo** per minimizzare lo scostamento temporale rispetto alla trasmissione via etere.
 - **Centinaia o migliaia di utenti contemporanei**

Soluzioni:

- Multicast con protocolli RTP/UDP Vedi [FastWeb multicast](#)
Buona scalabilità, ma sia multicasting che la porta RTP possono incontrare problemi nel supporto da parte dei provider.
 - Dynamic adaptive streaming over HTTP, in unicast.

Dynamic Adaptive Streaming over HTTP

Dynamic Adaptive Streaming è una tecnica particolare, basata su HTTP, che adegua la qualità dello streaming video alle risorse disponibili nel dispositivo dell'utente, quali le condizioni della rete (bandwidth) o la capacità della CPU, avendo come risultato il miglioramento della QoS dell'utente.

Funziona bene se il numero di utenti è moderato. Necessario avere il server con buona connettività internet, eventualmente in CDN.

Principali tecnologie: MPEG-DASH e HLS

Riferimenti: https://en.wikipedia.org/wiki/Dynamic_Adaptive_Streaming_over_HTTP

<https://www.cloudflare.com/learning/video/what-is-mpeg-dash/>

Conferenze in tempo reale

Comunicazioni multimediali tra più utenti in tempo reale

Requisiti:

- L'interattività richiede latenze molto contenute.
- Numero di partecipanti tipicamente limitato ma in alcuni casi può essere elevato.

Latenza: La rete telefonica considera accettabili latenze in una direzione entro **150 ms**

Componenti principali della latenza:

- Il ritardo di propagazione. *Ad esempio in una fibra ottica per 8000 Km (Seattle-Amsterdam) è di 40ms.*
- Ritardo dovuto al riempimento del pacchetto. *Un pacchetto da 1KB impiega 125ms per riempirsi a 64Kbps. Se si utilizzano pacchetti piccoli (160 bytes) è possibile ridurre la latenza, accettando un degrado di larghezza di banda.*
- Ritardo per compressione/decompressione. Soprattutto per il video.

Buffer: è ancora necessario per evitare riproduzioni a scatti e per il riordino dei pacchetti, ma deve essere piccolo per limitare la latenza.

Protocolli: Occorre un protocollo per il trasporto dei dati, che generalmente è RTP/UDP. Il protocollo di controllo deve gestire anche la **Segnalazione**, ovvero le problematiche di attivazione e gestione delle chiamate.

Voice over IP (VoIP)

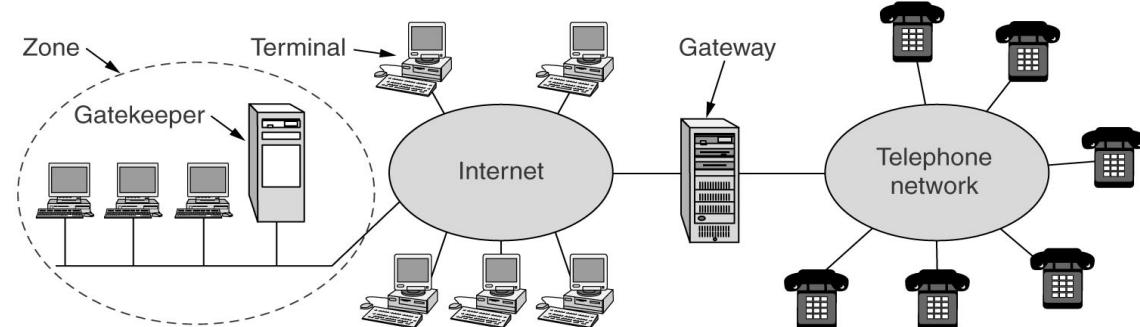
Voice over IP indica una tecnologia che rende possibile effettuare una conversazione, analoga a quella che si potrebbe ottenere con una rete telefonica, sfruttando una connessione Internet.

Per il **trasporto dei dati**, nella grande maggioranza delle implementazioni VoIP, viene adottato RTP (Real-time Transport Protocol) su UDP su IP.

Per quanto riguarda la **segnalazione** il processo di standardizzazione non si è ancora concluso. Al momento, sono coinvolti diversi enti internazionali di standardizzazione tra cui l'ITU (International Telecommunications Union) e l'IETF con due proposte alternative: **H.323** di ITU e **SIP** di IETF.

VoIP con H.323

H.323, creato nel 1996 da ITU, rappresenta una panoramica architetturale (dal punto di vista dell'industria telefonica) della telefonia su Internet. Non emette proprie specifiche ma fa riferimento a diversi protocolli specializzati: *codifica del parlato, impostazione delle chiamate, segnalazione, trasporto di dati, etc.*



Speech	Control				
	G.7xx	RTCP	H.225 (RAS)	Q.931 (Call signaling)	H.245 (Call control)
RTP					
UDP				TCP	
IP					
Data link protocol					
Physical layer protocol					

VoIP con SIP

SIP (Session Initiation Protocol) è la risposta di IETF (RFC 3261) ad H.323, considerato un prodotto tipico per telecomunicazioni (grande, complesso e poco flessibile), con un protocollo più semplice e modulare per la telefonia via Internet

Describe come impostare le telefonate via internet, le videoconferenze e altre connessioni multimediali. Gestisce solamente la segnalazione, ovvero l'impostazione, la gestione e la terminazione delle sessioni, mentre per il trasporto si usa RTP.

Riferimenti:

http://www.garr.it/eventiGARR/ws9/pdf/Sommani_pres.pdf

Protocollo SIP

Un SIP-URI (RFC 3261) rappresenta lo schema di indirizzamento SIP per chiamare un altro soggetto attraverso il protocollo SIP. In altre parole, un SIP URI è il recapito telefonico SIP di un utente. Il SIP URI assomiglia ad un indirizzo e-mail scritto nel seguente formato: {sip|sips}:[user-part@[domain-part[:port]]

Esempio: *sip:alfierir@ekiga.net* Queste URI possono contenere indirizzi IPV4, IPV6 o numeri di telefono veri e propri: <sips:+004437612234@sip-proxy.org:5062>

Il protocollo è basato su UDP/5060 con transazioni richiesta/risposta in ASCII (simile ad HTTP). Una transazione inizia con una Request inviata da uno User Agent Client ad un Proxy e termina con una Final Response inviata in senso inverso.

L'RFC 3261 definisce i seguenti metodi:

Method	Description
INVITE	Request initiation of a session
ACK	Confirm that a session has been initiated
BYE	Request termination of a session
OPTIONS	Query a host about its capabilities
CANCEL	Cancel a pending request
REGISTER	Inform a redirection server about the user's current location

