1. Livello di Applicazione

Connessioni stateless e stateful

Esistono due tipi di connessioni di stato:

Stateless: l'applicazione non tiene conto dei passi precedenti che l'utente ha fatto, sa solo quello che sta facendo in quell'istante

Stateful: La macchina conosce tutto il flusso di lavoro che l'utente ha svolto

1.1 Architettura di Rete

1.1.1 Rete di calcolatori

> *Una rete di calcolatori è un insieme di dispositivi elettronici (come computer, stampanti, server, ecc.) interconnessi tramite collegamenti fisici o wireless che permettono la comunicazione e lo scambio di dati. Queste reti utilizzano protocolli di comunicazione standard per garantire l'interoperabilità tra i dispositivi e facilitare la condivisione di risorse (come file, stampanti e connessioni a Internet). Le reti possono variare in scala, ad esempio dalle reti locali (LAN) alle reti geografiche (WAN), fino a reti globali come Internet.*

1.2 Modelli di comunicazione

1.2.1 Definizione

> *Il modello di comunicazione è una rappresentazione concettuale che descrive come i dati vengono scambiati e*

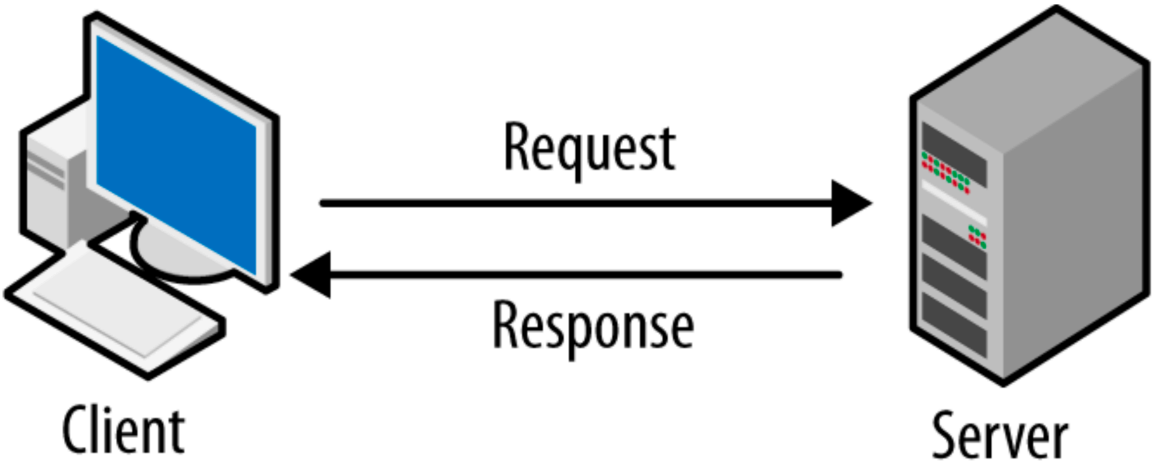
> *interpretati tra entità comunicanti. Ogni modello di comunicazione genera un processo di comunicazione.*

1.2.2 Modello Client-Server

Il modello Client-Server è un modello informatico utilizzato per l'accesso a delle informazioni dove un **client** prova ad accedere a dei dati, i quali sono memorizzati su un **server**. Può essere utilizzato anche quando client e server sono in due posti diversi.

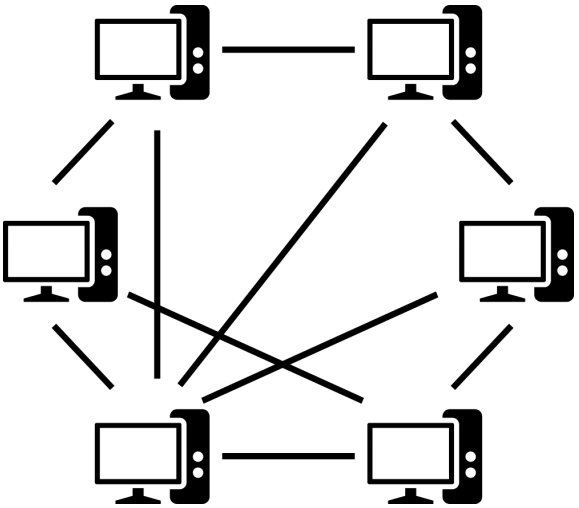
Analizzando meglio il modello client-server, può essere suddiviso in due processi:

- **Processo client**: un programma che invia una domanda al server e attende una risposta.
- **Processo server**: un programma che quando riceve la richiesta, esegue il lavoro o recupera i dati desiderati e restituisce una risposta.



1.2.3 Modello Peer to Peer

Nella forma di comunicazione peer-to-peer, gli individui che costituiscono un gruppo sono in grado di comunicare tra di loro. Ognuno può comunicare con chiunque all'interno del gruppo senza che vi sia una divisione predefinita tra server e client.



1.3.3 Servizi di trasporto per le applicazioni

Esistono diversi tipi di servizi di trasporto classificabili in una rete. Li possiamo ordinare in base a quattro fattori:

- Tipologia del trasferimento dati: affidabile e non affidabile
- Throughput
- Temporizzazione
- Sicurezza

Servizi

Un nuovo utente può visitare qualunque altro membro sia per vederne i contenuti sia per ottenere i nominativi di altri membri da ispezionare a loro volta.

1.2.4 Internet of Things

Questo modello di comunicazione, anche chiamato **ubiquitous computing**, è un modello dove la computazione è inglobata nella vita quotidiana. Si parla di comunicazione intelligente, nella vita di tutti i giorni. Questa rivoluzione, ancora in atto, si suppone connetterà in rete ogni dispositivo elettronico che compriamo.

E' molto più economico utilizzare un sensore per acqua, luce e gas rispetto a dei contatori da controllare manualmente. Come è molto più efficiente utilizzare

un sensore di fumo che avverte i vigili del fuoco rispetto ad un allarme domestico.

1.3 Comunicazione tra processi

1.3.1 Interfaccia tra processo e rete

Socket

Molte applicazioni oggi consistono in coppie di processi comunicanti che si scambiano messaggi. Un processo invia messaggi nella rete e riceve messaggi dalla rete attraverso un'interfaccia software detta socket. Quindi il socket è l'interfaccia presente tra il livello di applicazione e il livello di trasporto all'interno di un host.

API

Esistono anche le API (Application Programming Interface) tra l'applicazione e la rete, ovvero un insieme di regole e protocolli che permettono ai processi di comunicare tra di loro.

1.3.2 Indirizzamento

Su internet gli host vengono identificati attraverso degli **indirizzi** chiamati **indirizzi IP**. Un indirizzo IP è un numero di 32 bit che identifica univocamente l'host. Oltre a conoscere questo indirizzo però, che identifica l'host, il mittente deve anche identificare il processo destinatario: sapere quindi quale socket deve ricevere il dato. Il socket è identificato quindi da un **numero di porta di destinazione**.

1.3.3 Servizi di trasporto per le applicazioni

Esistono diversi tipi di servizi di trasporto classificabili in una rete. Li possiamo ordinare in base a quattro fattori:

- Tipologia del trasferimento dati: affidabile e non affidabile
- Throughput
- Temporizzazione
- Sicurezza

Servizi affidabili

Ogni servizio può ulteriormente essere caratterizzato in base **all'affidabilità**. Alcuni servizi sono **affidabili**, cioè non perdono mai dati e di solito vengono progettati in modo che il ricevente confermi la ricezione di ciascun messaggio così che il mittente sia informato. Un esempio di servizio affidabile orientato alla connessione è il trasferimento di file, poiché si vuole sapere che ogni bit sia arrivato a destinazione senza errori.

Esistono due varianti del servizio affidabile orientato alla connessione:

Message sequence

In questa variante i confini di messaggi vengono mantenuti, quindi due messaggi da 1024 byte arriveranno come due messaggi distinti da 1024 byte.

Inviare un libro alla stampante attraverso la rete ha la necessità di mantenere i confini tra le pagine.

Byte Stream

In questa versione la connessione è un singolo flusso di byte senza divisione di messaggi, quindi due messaggi da 1024 byte sono uguali a un messaggio da 2048 byte o come 2048 messaggi da 1 byte.

Quando si scarica un film, non è importante la distinzione tra i suoi byte.

Servizi non affidabili

Per quanto riguarda i servizi senza connessione, ci sono casi dove l'affidabilità è essenziale. Si può utilizzare il servizio datagram con conferma, cioè che come una raccomandata invia una ricevuta di ritorno.

Ci sono situazioni dove servizi non affidabili sono meglio di quelli affidabili, per esempio il contesto di un voice over IP dove è meglio sentire del rumore ogni tanto che avere del ritardo. Un altro esempio di questo tipo è il modello **request-reply** dove la conferma viene rappresentata dal messaggio ricevuto dal destinatario (comunemente implementato nei database con le query).

Throughput

> *Il throughput indica la quantità di dati che un sistema o una rete riesce a trasmettere o elaborare in un determinato intervallo di tempo. Solitamente, viene espresso in bit al*

> *secondo (bps), byte al secondo o in termini di pacchetti al secondo, e rappresenta un parametro fondamentale per valutare l'efficienza e le prestazioni di sistemi di comunicazione e informatici.*

Le applicazioni che hanno requisiti di throughput si chiamano **applicazioni sensibili alla banda**. Un esempio di questo tipo di applicazioni sono tutti i tipi di telefonia o videoconferenza.

Esistono anche le **applicazioni elastiche** che possono far uso di una quantità arbitraria di throughput, dipendentemente dalla disponibilità del momento.

Temporizzazione

Un protocollo a livello di trasporto può anche fornire garanzie di temporizzazione, che possono assumere varie forme: per esempio la garanzia potrebbe essere che ogni bit che il mittente invia sulla socket venga ricevuto dalla socket di destinazione non più di 100 ms più tardi.

In generale, molte applicazioni (come per il throughput) richiedono brevi ritardi per funzionare correttamente.

Sicurezza

La sicurezza ha come obiettivo, nella progettazione di rete, quello di difesa contro diversi tipi di minacce. Meccanismi come l'autenticazione impedisce a qualcuno di impersonare qualcuno che non è; oppure meccanismi per l'integrità prevengono cambiamenti nei messaggi, come l'alterazione del testo.

1.4 Livello di trasporto: introduzione

Internet, come ogni rete TCP/IP, mette a disposizione delle applicazioni due protocolli di trasporto: TCP e UDP. La prima decisione da prendere per realizzare un'applicazione internet è scegliere quale protocollo utilizzare.

1.4.1 TCP

Il TCP prevede un servizio orientato alla connessione e il trasporto più affidabile dei dati. Un'applicazione che utilizza il TCP ha due vantaggi:

1. **Ha un servizio orientato alla connessione**: client e server si scambiano messaggi a livello di trasporto prima che i messaggi a livello di applicazione comincino a fluire. Questa è una procedura chiamata **handshaking**, dove si preparano client e server alla partenza e alla ricezione dei pacchetti. Dopo che l'handshaking, si dice che esiste una connessione TCP tra i socket del client e del server di tipo full-duplex.
2. **Ha un servizio di trasferimento dati affidabile**: I processi comunicanti possono contare su TCP per trasportare i dati senza errori e nel giusto ordine, poiché non c'è perdita di dati.

1.4.2 UDP

L'UDP è un protocollo di trasporto leggero, dotato di un servizio minimo. Il protocollo UDP è un servizio non orientato alla connessione, non necessita infatti di handshaking e fornisce un trasferimento dati non affidabile. Quando la socket di partenza invia un messaggio, il protocollo UDP non garantisce che esso arrivi in perfetta condizione (e nemmeno che arrivi). Tuttavia, questo protocollo è utile per la velocità, si pensi a un servizio di streaming in real-time o a un servizio di videoconferenza come esempio di utilizzo.

1.4.3 Canale fisico

Il canale fisico può essere di 3 tipologie:

1. **Simplex**: singola direzione, un esempio è il telecomando di una televisione
2. **Half-duplex**: ci sono entrambe le direzioni, ma su un unico canale. Quindi non si possono mandare messaggi in contemporanea.
3. **Full-duplex**: ci sono entrambe le direzioni e hanno due canali di comunicazione differenti.

1.5 Protocolli a livello di applicazione

1.5.1 Cosa definiscono i protocolli

Un protocollo a livello di applicazione definisce come i processi di un'applicazione, in esecuzione su sistemi periferici diversi, si scambiano i messaggi. Definisce in particolare:

- I tipi di messaggi scambiati

- La sintassi dei vari tipi di messaggio
- La semantica dei campi
- Le regole per determinare quando e come un processo invia e risponde ai messaggi

Numeri di porte

> *Una **porta di rete** (o "numero di porta") è un valore numerico che, in ambito TCP/IP, viene utilizzato per identificare in modo univoco un processo o un servizio all'interno di un dispositivo connesso in rete. Quando un computer o un server riceve dati, li consegna a una specifica applicazione basandosi proprio sul numero di porta indicato nei pacchetti in arrivo.*

Le porte identificano il tipo di servizio da utilizzare (80 se si parla di http): `http://192.168.1.10:80`

1.5.2 Telnet

Telnet è un protocollo di rete che consente di accedere e controllare in remoto un sistema o dispositivo tramite una sessione testuale interattiva. Per prima cosa stabilisce una connessione TCP (solitamente sulla porta 23), dopodiché il client si connette al server remoto aprendo una sessione di terminale e una volta stabilita la connessione l'utente è in grado di inviare comandi al sistema remoto e ricevere le risposte in tempo reale.

1.5.3 HTTP

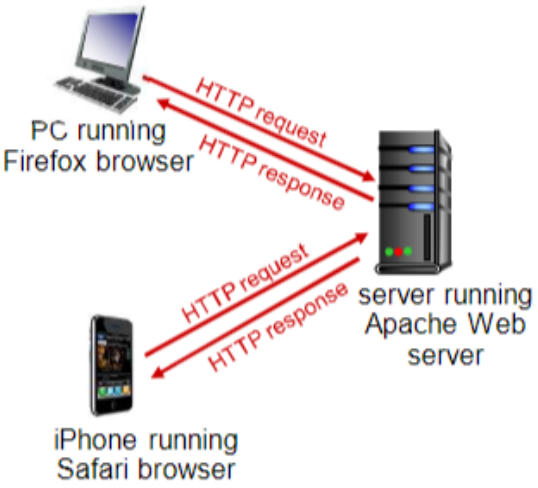
Le pagine web sono composte solitamente dall'indirizzo della pagina, l'URL (Uniform Resource Locator) che identifica la pagina tra i vari indirizzi IP (è un'astrazione), da degli oggetti che possono essere di tipo file (HTML, CSS, JavaScript, etc..), immagini o altro ancora memorizzato all'interno. Una pagina viene quindi identificata **dall'hostname** e dal suo **path**.

Esempio: `www.schoolname.edu/somedEpt/pic.gif`

Dove la parte in verde è l'hostname e la parte in rosso il path.

HTTP sta per HyperText Transfer Protocol, è un protocollo a livello di applicazione e costituisce il cuore del web moderno. HTTP utilizza TCP come protocollo di trasporto e fa i seguenti passaggi:

1. Il Client inizia una connessione con il server
2. Il server accetta la richiesta del client
3. Una volta stabilita, i processi client e server accedono a TCP attraverso le proprie socket
4. Si chiude la connessione TCP



Connessione persistenti e non

In molte applicazioni per internet, client e server comunicano per un lungo periodo di tempo. A seconda dell'applicazione e del suo impiego la serie di richieste potrebbe essere effettuata in sequenza periodicamente a intervalli regolari o in maniera intermittente. Su TCP questa differenza è importante, poiché si deve decidere se è necessario fare più connessioni separate o una unica. Nel primo caso si parla di **connessioni non persistenti** e nel secondo di **connessioni persistenti.**

Connessioni stateless e stateful

Esistono due tipi di connessioni di stato:

1. **Stateless**: l'applicazione non tiene conto dei passi precedenti che l'utente ha fatto, sa solo quello che sta facendo in quell'istante
2. **Stateful**: La macchina conosce tutto il flusso di lavoro che l'utente ha svolto

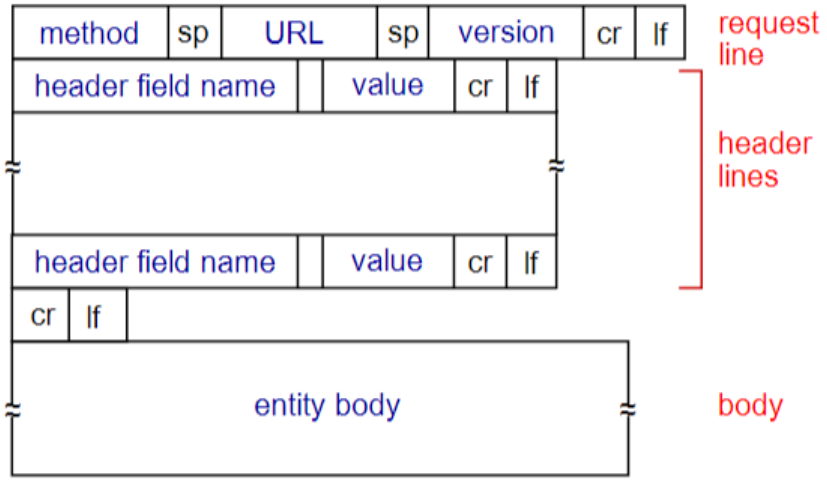
---

HTTP è una macchina stateless, il server non mantiene informazioni riguardo il passato dell'utente.

Formato dei messaggi

I messaggi di HTTP vengono divisi in righe:

- La prima riga è chiamata **riga di richiesta**
- Le altre, **righe di intestazione**



Riga di richiesta

Questa riga presenta tre campi: il campo URL, il campo versione di HTTP e il campo metodo che può assumere diversi valori (GET, POST, HEAD, PUT E DELETE).

Righe di intestazione

Le righe di intestazione sono una serie di informazioni di metadata. Specificano per esempio l'host o i cookie.

Versioni di HTTP

Differenti versioni di HTTP utilizzano diversi modi per recuperare gli oggetti:

- **HTTP/1.0**: chiude ogni connessione TCP dopo che l'oggetto richiesto è stato trasferito.
- **HTTP/1.1**: utilizza una connessione TCP persistente. Quindi diversi oggetti possono essere mandati in pipeline utilizzando la stessa connessione HTTP.
- **HTTP/2**: comprime l'header, permette il download parallelo. HTTP/2 introduce un livello di framing completamente binario che trasforma i messaggi HTTP in una sequenza di piccoli frame. Ogni frame ha un header fisso di 9 byte che specifica il tipo di frame, la lunghezza del payload, le flag e l'identificativo dello stream. Questo design permette di multiplexare più flussi di dati su una singola connessione TCP, migliorando l'efficienza e riducendo il problema del head-of-line blocking tipico di HTTP/1.1.

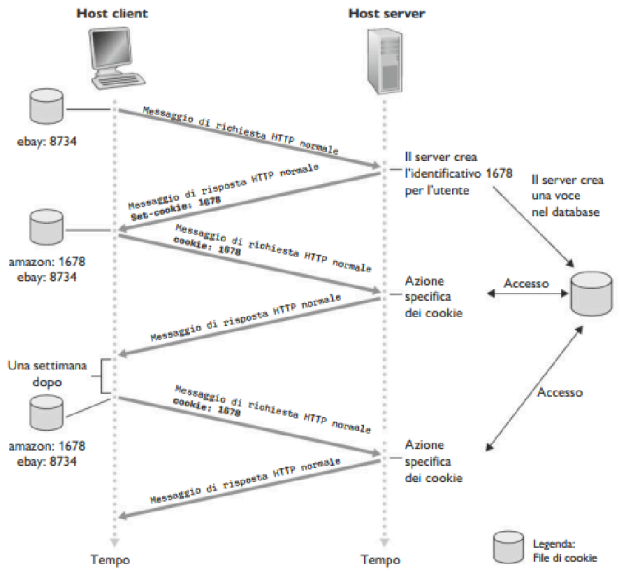
- **HTTP/3**: Usa UDP Quic invece di TCP, per garantire migliore efficienza.

Cookie

I server HTTP sono stateless, quindi non sanno il flusso di richieste che l'utente fa; molto spesso però, i server necessitano di autenticare gli utenti per ragioni di sicurezza e funzionalità. Per questo scopo, HTTP adotta i **cookie**.

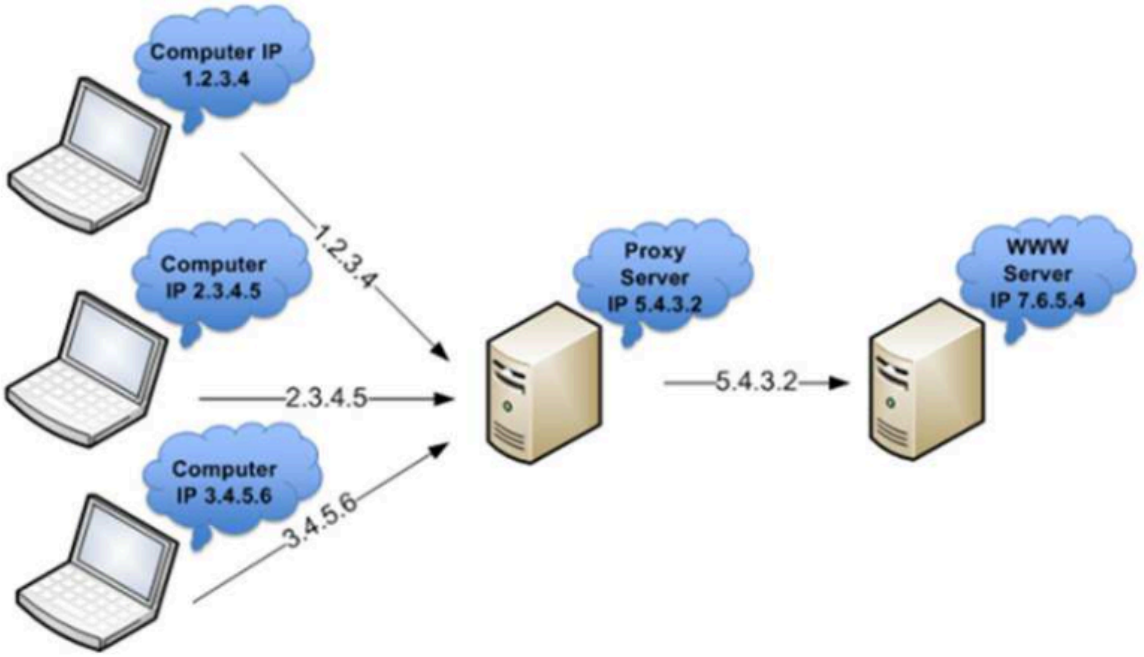
I cookie consentono di tener traccia degli utenti, essi presentano 4 componenti nella loro tecnologia:

1. Una riga di intestazione nel messaggio di risposta HTTP.
2. Una riga di intestazione nel messaggio di richiesta HTTP.
3. Un file mantenuto sul sistema dell'utente e gestito dal browser.
4. Un database sul sito



Proxy server

Esiste un tipo di **caching eseguito sul web**, chiamato **proxy server**, cioè un'entità di rete che soddisfa richieste HTTP al posto del web server. Il proxy ha una propria memoria sul disco in cui conserva copie di oggetti recentemente richiesti.



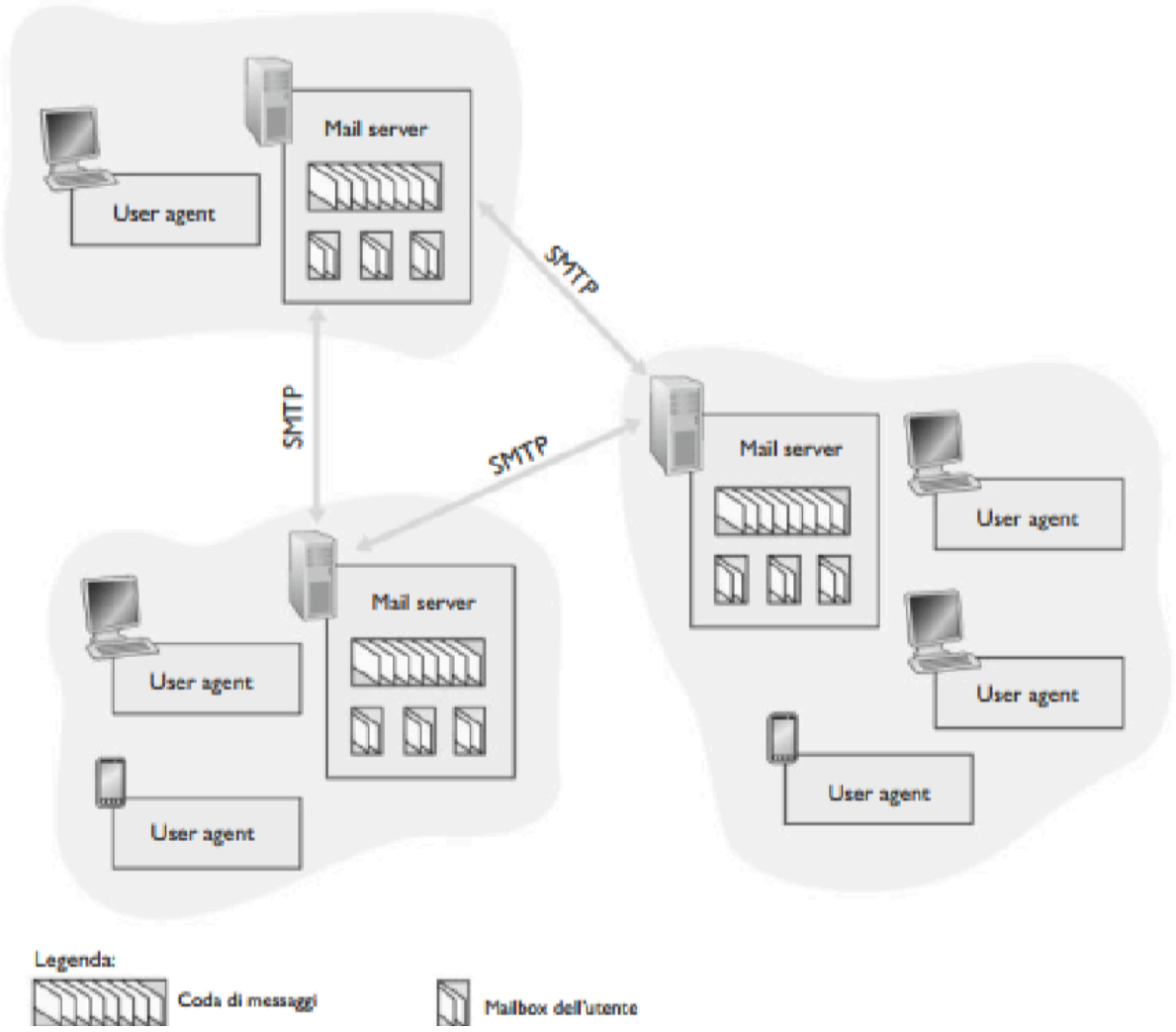
1.5.4 SMTP

Il protocollo SMTP è un protocollo utilizzato per spedire posta su internet; utilizza 3 componenti principali:

1. Lo **user agent**: consentono agli utenti di leggere, rispondere, inoltrare, salvare e comporre i messaggi.
2. I **mail server**: costituiscono la parte centrale dell'infrastruttura del servizio. Ogni utente ha una **mail box** collocata in un **mail server**. La mailbox gestisce e contiene messaggi inviati all'utente.
3. **Protocollo SMTP**: il protocollo principale utilizzato per spedire posta online.

Funzionamento

Il protocollo SMTP fa uso del trasferimento dati affidabile di TCP per trasferire la mail dal server del mittente a quello del destinatario. SMTP presenta un lato client, in esecuzione sul mail server del mittente, e un lato server, in esecuzione sul server del destinatario.



Ci sono 3 fasi nel trasferimento dei dati:

1. Handshaking tra i server
2. Trasferimento dei messaggi (posti in una coda di messaggi nel mail server)
3. Chiusura della connessione

Il mittente manda un messaggio in testo ASCII come comando, il destinatario invia come risposta un codice di stato. Ogni messaggio deve essere in ASCII 7 bit.

POP vs IMAP

Esistono due protocolli principali di ritirare la posta dal mail server. Il protocollo POP e il protocollo IMAP. Il primo si collega al server e scarica i contenuti, eliminandoli dalla memoria del server; il secondo, invece, fa connettere il client e fa visualizzare le immagini contenute al suo interno.

Feature	POP3	IMAP
Where is protocol defined?	RFC 1939	RFC 2060
Which TCP port is used?	110	143
Where is e-mail stored?	User's PC	Server
Where is e-mail read?	Off-line	On-line
Connect time required?	Little	Much
Use of server resources?	Minimal	Extensive
Multiple mailboxes?	No	Yes
Who backs up mailboxes?	User	ISP
Good for mobile users?	No	Yes
User control over downloading?	Little	Great
Partial message downloads?	No	Yes
Are disk quotas a problem?	No	Could be in time
Simple to implement?	Yes	No
Widespread support?	Yes	Growing

1.6 DNS: Domain Name System

1.6.1 Descrizione

Il DNS è un servizio che traduce i nomi dei vari siti in indirizzi IP a lunghezza fissa. Sono implementati grazie a un database distribuito implementando una gerarchia di DNS server e un protocollo a livello di applicazione che permette agli host di interrogare il database. Il protocollo DNS utilizza UDP e la porta 53. Viene comunemente utilizzato da HTTP e SMTP per tradurre i domini in indirizzi IP.

1.6.2 Distribuzione di DNS server

DNS centralizzato

I server DNS non sono centralizzati a un'unica macchina. Questo creerebbe diverse problematiche:

1. **Unico punto di vulnerabilità**: Se il server DNS si guasta, ne soffrirà l'intera rete Internet
2. **Volume di traffico elevato**: Un singolo server DNS dovrebbe gestire tutte le richieste e si creerebbe facilmente congestione e rallentamento.
3. **Database centralizzato distante**: Un singolo server DNS non può essere vicino a tutti i client e alcuni client impiegherebbero più tempo di altri (più vicini al server) per ricevere le risposte.

4. **Manutenzione**: Il server DNS dovrebbe tenere conto di tutti gli host utente provenienti da tutto il mondo (database estremamente vasto).

DNS Gerarchico e distribuito

Il DNS utilizza un grande numero di server, organizzati in maniera gerarchica e distribuiti nel mondo. Nessun DNS ha tutte le informazioni relative a tutti gli host esistenti. Sono invece distribuiti in tre classi:

1. **Root-Server**: ne esistono più di 1000 e sono dislocati nel mondo. Tali server sono copie di 13 differenti root server gestiti da 12 organizzazioni e forniscono gli indirizzi IP dei TLD server.
2. **Top-Level-Domain server (TLD)**: Questi server si occupano dei domini di primo livello (.com, .org, etc) relativi ai propri paesi.
3. **DNS server autoritativi**: Ogni organizzazione dotata di host pubblicamente accessibili tramite Intenet (web server e mail server) deve fornire record DNS pubblicamente accessibili che associno i nomi di tali host a indirizzi IP. Il DNS server autoritativo dell'organizzazione ospita questi record. Esistono DNS server autoritativi primari e secondari, che differiscono nelle autorizzazioni di modifica; il primo è l'originale mentre il secondo è una copia in sola lettura (questo aumenta anche la ridondanza).

1.6.3 Operazioni su DNS server

Query su DNS server

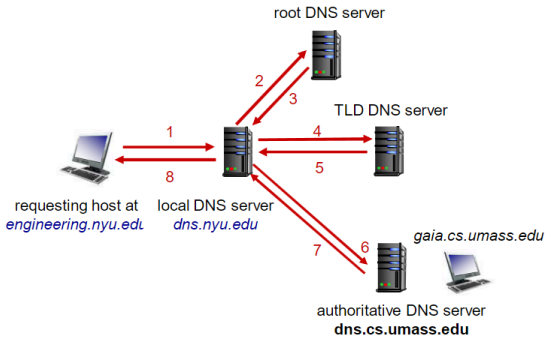
Esistono due tipi di query su server DNS, in base all'interrogazione che stiamo facendo.

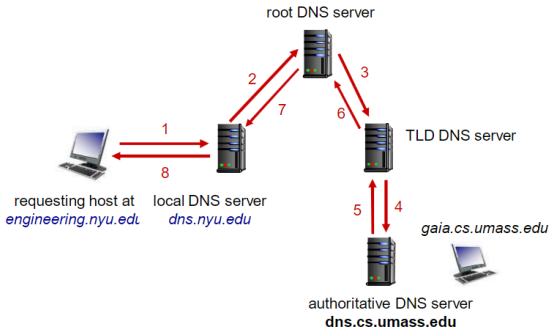
Query Iterativa

Questo tipo di query contatta il server locale che, se non conosce la risposta, interroga tutti i server DNS in ordine gerarchico finché non trova una risposta.

Query ricorsiva

Questo tipo di query contatta i server DNS in ordine gerarchico crescente e li interroga ricorsivamente.





Caching

Un DNS server locale può, per un certo tempo limitato, salvare informazioni richieste frequentemente in una cache della macchina. Questo è utile per siti molto visitati in una certa zona.

Record

I server che implementano il database distribuito di DNS memorizzano i cosiddetti **record di risorsa** dove sono presenti anche le corrispondenze tra nomi e indirizzi. Ogni messaggio di risposta trasporta uno o più record di risorse ed esso può contenere i seguenti campi:

`(Name, Value, Type, TTL)`

TTL è il Time To Live, ossia il tempo residuo di vita di un record e determina quando una risorsa vada rimossa dalla cache. Il significato di Name e Value dipende da Type:

1. **Type=A**: Name è il nome dell'host e Value è il suo indirizzo IP. Fornisce quindi la corrispondenza tra hostname e indirizzo IP.
2. **Type=NS**: Name è un dominio e Value è l'hostname del DNS server autoritativo che sa come ottenere gli indirizzi IP degli host nel dominio
3. **Type=CNAME**: Value rappresenta il nome canonico dell'host per il sinonimo Name. Questo record può fornire agli host richiedenti il nome canonico relativo a un hostname.
4. **Type=MX**: Value è il nome canonico di un mail server che ha il sinonimo Name. Questo tipo di record consente agli hostname dei mail server di avere sinonimi semplici.

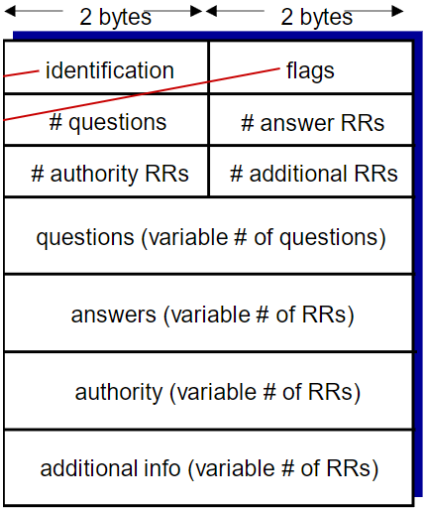
Messaggi DNS

Le richieste e le risposte di messaggi DNS hanno lo stesso formato: 12 byte (6 blocchi da 2 byte ciascuno) che rappresentano l'header.

La sezione delle domande contiene informazioni sulle richieste che stanno per essere effettuate.

La sezione autoritativa contiene i record di altri server autoritativi.

La sezione aggiuntiva racchiude altri record utili (come un record di tipo A come risposta e una richiesta MX).



1.7 SNMP: Simple Network Management Protocol

1.7.1 Introduzione a SNMP

SNMP, acronimo di *Simple Network Management Protocol*, è un protocollo essenziale per la gestione e il monitoraggio dei dispositivi di rete. Utilizzato in ambienti che spaziano da reti aziendali a infrastrutture di telecomunicazione, SNMP consente agli amministratori di ottenere informazioni dettagliate sullo stato dei dispositivi e intervenire tempestivamente in caso di anomalie. Il protocollo opera tipicamente su UDP, garantendo rapidità nelle comunicazioni anche se a scapito di alcune funzionalità di affidabilità.

1.7.2 Componenti e Architettura

L'architettura SNMP si basa su due componenti fondamentali: il *manager SNMP* e gli *agenti SNMP*. Il manager, generalmente installato su un server centralizzato, invia richieste di informazioni ai dispositivi di rete, i quali ospitano un agente che raccoglie e organizza i dati operativi in una struttura standard chiamata *Management Information Base (MIB)*. La MIB è una sorta di database gerarchico che definisce quali informazioni possono essere lette o modificate, facilitando così la gestione centralizzata dei dispositivi.

1.7.3 Trap e Notifiche

Una caratteristica chiave di SNMP è la capacità di inviare notifiche in tempo reale, conosciute come *trap*. Queste notifiche sono messaggi inviati dagli agenti al manager in caso di eventi critici o variazioni rilevanti nelle prestazioni del dispositivo, come guasti hardware o errori di configurazione. Oltre ai trap, SNMP supporta anche i messaggi *inform*, che richiedono una conferma di ricezione, migliorando così l'affidabilità della comunicazione in contesti dove è necessario garantire che le notifiche siano state effettivamente ricevute.

1.7.4 Versioni e Sicurezza

Nel corso degli anni, SNMP ha subito evoluzioni significative. Le versioni iniziali, SNMPv1 e SNMPv2c, offrivano funzionalità di base ma presentavano importanti lacune in termini di sicurezza, poiché la trasmissione dei dati avveniva in chiaro senza autenticazione. SNMPv3 ha risolto questi problemi introducendo meccanismi di autenticazione e crittografia, come il modello di sicurezza basato su utenti (USM) e il controllo degli accessi (VACM). Questi miglioramenti hanno reso SNMP adatto anche per ambienti in cui la protezione delle informazioni è una priorità.

1.7.5 Applicazioni e Vantaggi

SNMP è ampiamente impiegato per monitorare le performance di reti eterogenee e garantire la continuità operativa. Permette di raccogliere statistiche in tempo reale, identificare colli di bottiglia e risolvere rapidamente eventuali problemi di rete. La sua capacità di integrare dispositivi di diversi produttori grazie a standard comuni e MIB condivise facilita la gestione centralizzata, riducendo i costi operativi e migliorando l'efficienza complessiva della rete. Inoltre, l'automazione offerta da SNMP aiuta a prevenire interruzioni di servizio, aumentando la qualità dell’esperienza utente e la resilienza delle infrastrutture di rete.

1.8 FTP – File Transfer Protocol

FTP (File Transfer Protocol) è un protocollo dell'applicazione usato per trasferire file tra due host attraverso una rete TCP/IP. Nato nei primi anni dello sviluppo di Internet, FTP consente a un client di inviare, ricevere, rinominare, cancellare e consultare file memorizzati su un server remoto. È ampiamente

impiegato per la pubblicazione e la gestione di contenuti web, la condivisione di file tra sistemi e l'aggiornamento remoto di software.

1.8.1 Funzionamento

FTP adotta un'**architettura client-server**, nella quale il **client FTP** si connette al **server FTP** per richiedere operazioni sui file. Il protocollo utilizza due connessioni TCP distinte:

- **Connessione di controllo (porta 21)**: usata per l'invio dei comandi (es. autenticazione, navigazione delle directory). Questa mantiene una connessione persistente.
- **Connessione dati (porta 20 o dinamica)**: usata per il trasferimento effettivo dei file. Questa mantiene, invece, una connessione non persistente.

Esistono due modalità di trasferimento:

- **Modalità attiva**: il server stabilisce la connessione dati verso il client.
- **Modalità passiva**: il server comunica al client su quale porta ascoltare, e il client stabilisce la connessione. Questa modalità è preferita in presenza di firewall o NAT.

Il client FTP invia comandi come `USER` (nome utente), `PASS` (password), `RETR` (scarica file), `STOR` (carica file), `LIST` (elenco directory), ricevendo dal server risposte codificate con numeri a tre cifre.

1.8.2 Utilità

FTP è utile in molti scenari, tra cui:

- **Distribuzione di software**: molti siti web offrono file scaricabili tramite FTP.
- **Gestione remota di siti web**: webmaster caricano e modificano i contenuti di un sito tramite client FTP.
- **Backup e sincronizzazione**: FTP può essere usato per automatizzare trasferimenti programmati di dati tra sistemi.

Tuttavia, FTP trasmette dati e credenziali in chiaro, rendendolo vulnerabile a intercettazioni. Per questo motivo, è stato affiancato da versioni sicure come **FTPS** (FTP su TLS/SSL) e **SFTP** (che, pur non essendo una variante di FTP, fornisce funzionalità analoghe tramite SSH).

In sintesi, FTP rappresenta uno strumento fondamentale per il trasferimento di file in rete, combinando semplicità d’uso con flessibilità, sebbene oggi sia spesso sostituito da alternative più sicure.