Descrivere il funzionamento di OSPF.

OSPF è un protocollo link-state che utilizza il flooding di informazioni riguardo lo stato di collegamenti di Dijkstra per la determinazione del percorso a costo minimo. Un router costruisce un grafo dell'intero sistema autonomo e manda in esecuzione l'algoritmo di Dijkstra per determinare un albero dei percorsi minimi verso tutte le sottoreti. I costi dei collegamenti vengono fissati dall'amministratore di rete. Ogni qualvolta si verifica un cambiamento dello stato di un collegamento il router manda informazioni di instradamento via broadcast a tutti gli altri router nel sistema autonomo. Inoltre invia periodicamente lo stato dei collegamenti anche se questo non è cambiato. OSPF è sicuro poiché i messaggi vengono scambiati solo tra router autenticati; supporta più percorsi con lo stesso costo verso la medesima destinazione usandoli entrambi, supporta sia l'instradamento unicast che multicast e infine permette di strutturare i sistemi autonomi in modo gerarchico. La gerarchia è posta su due livelli: aree locali e l'area di dorsale, il cui ruolo è quello di instradare il traffico tra le aree del sistema autonomo. Nelle aree locali vengono inviati gli stati dei collegamenti così facendo ogni nodo conosce solo il cammino più breve per le altre aree.

Descrivere le componenti del ritardo in una rete a pacchetto.

Le componenti del ritardo in una rete a pacchetto sono: il RITARDO DI ELABORAZIONE, il RITARDO DI ACCODAMENTO, il RITARDO DI TRASMISSIONE e il RITARDO DI PROPAGAZIONE.
Il RITARDO DI ELABORAZIONE include il tempo richiesto per esaminare l'intestazione del pacchetto e per determinare dove dirigerlo. Possono essere inclusi anche altri fattori, tra i quali il tempo per controllare errori a livello di bit eventualmente accorsi nel pacchetto durante la trasmissione dal nodo a monte al router A. Dopo l'elaborazione, il router A dirige il pacchetto verso la coda che precede il collegamento al router B. Una volta in coda il pacchetto subisce un RITARDO DI ACCODAMENTO mentre attende la trasmissione sul collegamento. La lunghezza di tale ritardo per uno specifico pacchetto dipenderà dal numero di pacchetti precedentemente arrivati, accodati e in attesa di trasmissione sullo stesso collegamento. Assumendo che i pacchetti siano trasmessi la politica FCFS, il pacchetto può essere trasmesso solo dopo la trasmissione di tutti quelli che lo hanno preceduto nell'arrivo. Sia L la lunghezza del pacchetto, in bit, e R bps la velocità di trasmissione del collegamento del router A al router B, il RITARDO DI TRASMISSIONE risulta essere L/R. Questo è il tempo richiesto per trasmettere tutti i bit del pacchetto sul collegamento. Una volta immesso sul collegamento, un bit deve propagarsi fino al router B. Il tempo impiegato è il RITARDO DI PROPAGAZIONE. Il bit viaggia alla velocità di propagazione del collegamento che dipende dal mezzo fisico. Il ritardo di propagazione è dato da d/v, dove d è la distanza tra i due router mentre v è la velocità di propagazione nel collegamento. Il ritardo di trasmissione è la quantità di tempo impiegata da parte del router per trasmettere il pacchetto in uscita, ed è funzione della lunghezza del collegamento, ma non ha niente a che fare con la distanza tra i router.

Descrivere come viene realizzato l'instradamento inter-AS nella rete Internet.

L'instradamento inter-AS in Internet viene realizzato attraverso il protocollo BGP. In BGP coppie di router si scambiano informazioni attraverso una connessione TCP semipermanente detta SESSIONE. Vi sono due tipologie di sessioni: quella tra due router all'interno di uno stesso sistema autonomo è detta iBGP, mentre quella tra sistemi autonomi distinti è chiamata eBGP. BGP basa il suo funzionamento sull'utilizzo di una tabella contenente prefissi, ossia informazioni sulla raggiungibilità delle diverse reti tra più sistemi autonomi. Quando un router gateway di un qualunque sistema autonomo riceve prefissi, appresi tramite sessioni eBGP, li distribuisce agli altri router del sistema autonomo, avvalendosi delle proprie sessioni iBGP, i quali li memorizzano nelle proprie tabelle di inoltro. L'utilizzo dei prefissi include anche degli attributi BGP che insieme ai prefissi stessi prendono il nome di rotta. Attributi BGP rilevanti sono AS-PATH e NEXT-HOP. Il primo elenca i sistemi autonomi attraversati dall'annuncio dei prefissi. Il secondo riporta l'interfaccia del router che inizia AS-PATH. Quando il gateway riceve la rotta applica le proprie politiche di importazione per decidere se

scartare o meno la rotta e se impostare determinati attributi quali parametri di scelta. La politica di importazione potrebbe far scartare la rotta al gateway poiché il sistema autonomo non vuole inviare traffico su uno dei sistemi autonomi presenti in AS-PATH oppure perché conosce una via migliore verso lo stesso prefisso. Qualora sia con iBGP che con eBGP un router riceva più rotte verso lo stesso prefisso, adotterà diverse regole di eliminazione per rimanere con una sola rotta possibile. Una prima regola sarebbe quella di assegnare un valore di preferenza locale ad una rotta per scegliere quella con valore più alto. Poi tra quelle con lo stesso valore di preferenza locale si sceglie quella con l'AS-PATH più corto. Se ciò non bastasse si sceglierà quella con il NEXT-HOP più vicino, avendo applicato l'algoritmo della patata bollente (HOT-POTATO ROUTING).

Descrivere le caratteristiche principali del protocollo HTTP 1.1.

HTTP 1.1 è un protocollo a livello di applicazione basato sul modello client server che definisce la struttura dei messaggi, le modalità di richiesta delle pagine dei web client e le modalità di invio delle medesime da parte dei server. HTTP 1.1 utilizza TCP come protocollo di trasporto e come tipologia di connessione quella persistente, la quale permette di spostare più oggetti su una singola connessione TCP ed evita di creare una nuova connessione TCP per ogni oggetto contenuto nella pagina web. Di default la connessione persistente utilizza il pipelining, ovvero la connessione TCP rimane aperta e il client richiede l'invio di tutti gli oggetti che trova nella pagina senza aspettare che il server termini di inviare l'oggetto richiesto in precedenza. Quando un'utente richiede una pagina web, il browser invia al server messaggi di richiesta HTTP per gli oggetti nella pagina e il server, dopo aver ricevuto le richieste, risponde con messaggi di risposta HTTP contenenti gli oggetti. Il client HTTP inizia una connessione TCP con il server e una volta stabilita i processi client e server accedono a TCP tramite le proprie socket; il server invia i file richiesti al client senza memorizzare alcuna informazione di stato a proposito del client, per questo HTTP è classificato come protocollo senza memoria di stato. Il messaggio di richiesta HTTP è costituito da un numero indefinito di righe dove la prima è la riga di richiesta e quelle successive le righe di intestazione. La riga di richiesta presenta tre campi: il campo metodo, il campo URL e il campo versione di HTTP. La seconda riga indica l'host su cui risiede l'oggetto richiesto; la riga successiva, ovvero user agent, specifica il browser utilizzato; si ha poi la riga dell'intestazione che indica la tIpologia di connessione al server, nel caso sia di tipo close, il browser sta comunicando al server che non si deve occupare di connessioni persistenti, ma vuole che il server chiuda la connessione dopo aver inviato l'oggetto richiesto; infine la riga di intestazione indicante la versione preferita dell'oggetto e se non disponibile il server invierà quello di default. I metodi presenti in HTTP 1.1 sono: il GET, ovvero il metodo indicante la richiesta di una determinata risorsa; poi c'è POST, simile a GET, con la differenza che i contenuti specifici della pagina dipendono da ciò che l'utente ha immesso nei campi del form; Il metodo HEAD è simile a GET ma quando un server riceve una richiesta con questo metodo, risponde con un messaggio di risposta tralasciando gli oggetti richiesti; il metodo PUT consente agli utenti di inviare un oggetto a un percorso specifico (directory) su uno specifico web server; il metodo DELETE consente invece la cancellazione di un oggetto su un server. Il messaggio di risposta di HTTP presenta tre sezioni: una riga di stato iniziale, sei righe di intestazione e il corpo. Quest'ultimo è il fulcro del messaggio perché contiene l'oggetto richiesto. La riga di stato presenta tre campi: la versione del protocollo, un codice di stato e un corrispettivo messaggio di stato. Tra i codici di stato e i relativi messaggi i più utilizzati sono: 200 OK (la richiesta ha avuto successo e si invia l'informazione), 301 Moved Permanently (l'oggetto richiesto è stato trasferito in modo permanente), 400 Bad Request (codice di errore generico che indica che la richiesta non è stata compresa dal server), 404 Not Found (il documento richiesto non esiste sul server), 505 HTTP Version Not Supported (il server non dispone della versione di protocollo HTTP richiesta). Nelle righe di intestazione invece abbiamo: la riga di intestazione Connection; la riga Date che indica l'ora e la data di creazione e invio, da parte del server, della risposta http; la riga Server indica che il messaggio è stato generato da un determinato web server; essa è analoga alla riga User-Agent nel messaggio di richiesta http; la riga Last Modified indica l'istante e la data il cui oggetto è stato creato o modificato per l'ultima volta; la riga di intestazione Content-Length contiene il numero di byte dell'oggetto inviato; la riga Content-Type indica il tipo dell'oggetto contenuto nel corpo. Le

differenze con la versione 1.0 sono che quest'ultima utilizza connessioni non persistenti e non implementa i metodi PUT e DELETE.

Illustrare le caratteristiche generali e i relativi vantaggi delle reti a commutazione di circuito e delle reti a pacchetto.

La rete a commutazione a pacchetto consiste nella divisione dei messaggi in pacchetti che attraverso collegamenti e commutatori a pacchetto verranno trasmessi al destinatario con una velocità pari alla velocità totale del collegamento stesso. Solitamente un commutatore a pacchetto utilizza un protocollo store and forward, ovvero prima di poter trasmettere deve ricevere l'intero pacchetto che immagazzinerà in un buffer di output. Nella rete a commutazione di circuito le risorse richieste lungo un percorso per consentire la comunicazione tra sistemi periferici sono riservate per l'intera durata della sessione di comunicazione. Le risorse possono essere allocate dividendole in base al tempo e alla frequenza. Nel primo caso si parla di TDM, nel secondo FDM. Nella commutazione a circuito la velocità di trasmissione è costante. Confrontando le due possiamo dire che la commutazione a pacchetto ha il vantaggio rispetto alla commutazione a circuito di dare la possibilità a più utenti di sfruttare le risorse avendo le stesse prestazioni, questo perché la commutazione a circuito alloca uno spazio (slot di tempo o frequenza) fisso all'utente che potrebbe anche non utilizzare. D'altro canto la commutazione a circuito ha dalla sua il fatto di poter garantire una velocità costante di trasmissione al contrario della commutazione a pacchetto che può incappare in fasi di congestione della rete qualora la velocità di arrivo dei pacchetti superi quella di inoltro sul collegamento d'uscita. Altra differenza è che la commutazione a pacchetto presenta perdita di pacchetti in caso di buffer pieno mentre la commutazione a circuito non ce l'ha. Ulteriore fattore da analizzare è che la commutazione a circuito è più adatta per servizi in tempo reale rispetto alla commutazione a pacchetto che presenta ritardi end-to-end variabili e non determinabili a priori.

Descrivere il funzionamento di uno switch.

Il ruolo dello switch è ricevere i frame in ingresso e inoltrarli sui collegamenti in uscita. Lo switch stesso è trasparente ai nodi; cioè, un nodo indirizza un frame a un altro nodo, piuttosto che indirizzarlo allo switch e invia il frame nella LAN, senza sapere che uno switch riceverà il frame e lo inoltrerà agli altri nodi. Il tasso al quale i frame giungono a una qualsiasi delle interfacce di uscita degli switch può temporaraneamente eccedere la capacità del collegamento di quell'interfaccia, poiché le interfacce di uscita dello switch hanno dei buffer, analogamente alle interfacce di uscita dei router per datagrammi. Più in dettaglio lo switch presenta le seguenti funzionalità: filtraggio, inoltro e autoapprendimento. Il filtraggio (o filtering) è la funzionalità dello switch che se un frame debba essere inoltrato a qualche interfaccia o scartato. L'inoltro consiste nell'individuazione dell'interfaccia verso cui il frame deve essere diretto e, quindi, nell'inviarlo a quell'interfaccia. Entrambe le operazioni sono eseguite mediante una tabella di commutazione contenente l'inirizzo MAC del nodo, l'interfaccia dello switch che conduce al nodo, l'instante di tempo in cui la voce per quel nodo viene inserita in tabella. Proprio le tabelle sono i mezzi su cui si basa l'ultima funzionalità offerta dagli switch: l'autoapprendimento. L'autoapprendimento infatti, permette di costruire automaticamente, in maniera dinamica e autonoma, le tabelle allo switch. Inizialmente la tabella è vuota; non appena riceve un frame, ne archivia il suo inidrizzo MAC, l'interfaccia da cui arriva e il momento di arrivo. La tabella sarà completa non appena tutti i nodi della LAN avranno inviato un frame da un determinato indirizzo sorgente, lo cancella dalla tabella. Per via dell'autoapprendimento, gli switch vengono definiti come dispositivi Plug-And_play. Tra i vantaggi offerti dallo switch ricordiamo: eliminazione delle collisioni che porta ad un significativo miglioramento delle prestazioni su LAN con collegamenti broadcast, collegamenti eterogenei ovvero grazie al fatto che uno switch isola un collegamento dall'altro; i diversi collegamenti nella LAN possono funzionare a velocità diverse e possono usare mezzi trasmissivi diversi; infine vi sono vantaggi legati alla gestione che diventa più facile e più sicura.

Descrivere gli algoritmi di routing distance vector.

Vediamo dapprima l'algoritmo DV generalizzato. Si base sulla formula di Bellman-Ford dove si definisce dx(y) come il cammino minimo tra x e y. Allora dx(y) = min{c(x,y)+dv(y)}, dove min è il minimo tra tutti i vicini di x. L'idea di base è ogni tanto, tutti i nodi, mandino i loro vettori delle distanze stimante ai propri vicini, in maniera asincrona. Quando uno dei nodi riceve la stima dei vettori delle distanze dei propri vicini, aggiornerà il proprio vettore delle distanze usando l'equazione di Bellman-Ford. Non appena vi è una modifica sui costi dei vettori delle distanze, tutti i vettori vengono prontamente informati e a loro volta modificheranno i propri vettori delle distanze. A causa di questo meccanismo si può incappare nel problema del conteggio all'infinito, ossia si è venuto a creare un ciclo tra i nodi che quindi blocca l'intero sistema. Più in dettaglio questo accade se un nodo sorgente instrada pacchetti ad un nodo destinatario tramite un nodo intermedio e il costo del cammino minimo tra la sorgente e il destinatario diventa più conveniente di quello tra intermedio e destinatario; allora il nodo destinatario instraderà traffico al nodo sorgente venendo quindi a creare un ciclo. Per ovviare al problema del conteggio all'infinito, l'algoritmo DV utilizza il meccanismo della "poisoned reverse" (inversione avvelenata) ovvero nella situazione descritta precedentemente il nodo sorgente comunicherà al nodo intermedio che il suo cammino minimo ha lunghezza infinita, ovvero non c'è collegamento anche se realmente ci sarebbe, facendo in modo che il nodo destinatario non possa in nessun caso instradare pacchetti al nodo sorgente. Da questo algoritmo ne sono stati originati diversi, tra cui il RIP usato nell'instradamento intra-AS. Il RIP funziona similmente all'algoritmo DV idealizzato. Tutti i collegamenti hanno costo unitario e il costo massimo di un percorso è limitato a 15. Ogni 30 secondi i router si scambiano aggiornamenti di instradamento (detti advertisement o ACK), che contengono all'interno fino a 25 sottoreti di destinazione all'interno del sistema autonomo, nonché la distanza del mittente da ciascuna di queste sottoreti. I router tengono quindi una tabella di instradamento che include il vettore delle distanze e la tabella di inoltro, ossia la sottorete di destinazione, il next-HOP o il router successivo lungo il percorso più breve verso la destinazione e il numero di HOP per aggiungere la sottorete di destinazione. Se un vicino non comunica nulla per 180 secondi, viene considerato morto e i suoi percorsi invalidi. Attraverso gli adv, tutti i router modificheranno in breve tempo le proprie tabelle di instradamento. Le tabelle sono gestite da un protocollo applicativo chiamato routed. Gli adv vengono inviati mediante pacchetti UDP ripetuti periodicamente. All'avvio il router invia una speciale richiesta RIP a tutte le sue interfacce, richiedendo ai router le proprie tabelle di routing. In questa maniera scoprirà i router adiacenti.

Descrivere come viene realizzata la comunicazione affidabile in TCP.

La comunicazione affidabile è un servizio al di sopra del servizio inaffidabile e best-effort di IP, il quale assicura che il flusso di byte che i processi leggono dal buffer di ricezione TCP non sia alternato, non abbia buchi, non presenti duplicazioni e rispetti la sequenza originaria. La comunicazione affidabile in TCP viene effettuata attraverso tre meccanismi diversi: il numero di sequenza del pacchetto, gli ACK cumulativi e il timeout. Attraverso l'utilizzo di un numero di sequenza all'interno del pacchetto dati si garantisce che, qualora un pacchetto si perda, il mittente, ricevendo l'ACK indicante il numero di sequenza precedente a quello perduto, capisca che la trasmissione non è andata a buon fine e ritrasmetta il pacchetto. Ovviamente in base al protocollo scelto, si sceglie se i pacchetti ricevuti fuori sequenza debbano essere immagazzinati o meno. Solitamente il meccanismo di ritrasmissione scatta dopo che il mittente riceve 3 ACK duplicati dell'ultimo pacchetto in sequenza riscontrato correttamente. Il meccanismo degli ACK cumulativi funziona nel seguente modo: il mittente riceve l'ACK che indica che fino a quel pacchetto riportato nell'ACK il destinatario ha ricevuto correttamente i pacchetti. Nel caso in cui il destinatario riceva in maniera corretta il pacchetto, ma l'ACK che lui invia si perde nel tragitto e allo stesso tempo il ricevente ha riscontrato in maniera corretta il pacchetto con il numero di sequenza successiva a quello il cui ACK è andato perduto, il mittente riscontrerà

quest'ultimo ACK e quindi, nonostante uno degli ACK sia andato perduto, non dovrà provvedere alla ritrasmissione del pacchetto il cui ACK è andato perso. L'ultimo meccanismo è quello del cosiddetto timeout. Il mittente imposta un timer di una lunghezza predefinita che parte non appena il pacchetto viene inviato. Se per caso il pacchetto non viene ricevuto o l'ACK inviato dal destinatario si perde, il mittente una volta scaduto il timer provvederà ad inviare nuovamente il pacchetto. Le controindicazioni riguardo all'utilizzo del timer sono: l'intervallo di tempo non deve essere troppo breve o troppo lungo in durata. Infatti, poniamo il caso di un timer che scatti in maniera intempestiva; si correrebbe il rischio che un pacchetto venga inviato nuovamente nonostante sia stato ricevuto correttamente in precedenza e che magari possa far perdere l'ACK che nel frattempo stava viaggiando dal destinatario al mittente rendendo vana la trasmissione e inefficiente il servizio. Se invece, il timer ha una durata troppo lunga vi è il rischio che in caso di perdita del pacchetto o dell'ACK, il mittente debba stare in attesa prima di poter inviare nuovamente il pacchetto. Tutto questo renderebbe il sistema lento e inefficiente. Per questo motivo il timer viene calcolato attraverso il Round Trip Time, ovvero il tempo che passa tra l'invio del pacchetto e la ricezione dell'ACK corrispondente. Legata all'impostazione del timer vi sono due varianti: la prima prevede il raddoppio del timeout dopo che scade per la prima volta; la seconda variante prevede il meccanismo della ritrasmissione rapida, ovvero se il mittente riceve 3 ACK duplicati prima della scadenza del timer, automaticamente decide di inviare nuovamente il pacchetto che è andato perduto. In questo modo il sistema è efficiente perché evita la congestione.

Descrivere il funzionamento del controllo di congestione TCP.

TCP adotta un approccio nel quale il mittente verifica la disponibilità di traffico sul canale: se è scarso incrementa il proprio tasso trasmissivo, altrimenti lo riduce. Questo vincolo viene imposto al mittente in maniera indiretta, ossia viene imposto solo quando si verificano eventi di perdita di pacchetti. La perdita viene segnalata al mittente in diversi modi, come la ricezione di 3 ACK duplicati o all'occorrenza di un timeout, ossia quando scade il timer imposto al pacchetto più vecchio non ancora riscontrato. Concretamente TCP lavora sul valore cwnd ossia la finestra di congestione, il vincolo visto precedentemente. L'algoritmo di controllo di congestione TCP (o algoritmo di Von Jacobson) si articola in tre componenti o fasi principali: slow start, congestion avoidance e fast recovery. Durante la fase di slow start il valore cwnd parte da 1 MSS e lo incrementa di 1 ad ogni riscontro di ACK. In altre parole ad ogni RTT raddoppia il tasso trasmissivo del mittente TCP, che cresce quindi in maniera esponenziale. In caso di perdita il mittente TCP rimette cwnd a 1 e si salva in SISTRESH il valore cwnd/2 ossia metà del valore di cwnd al momento del rilevamento della congestione, riprendendo la fase di slowstart. QUando cwnd è pari a SISTRESH, la fase di slowstart termina e si entra in quella di congestion avoidance. Vi è anche il caso di rilevamento di 3 ACK duplicati e in questa situazione TCP effettua una ritrasmissione rapida, cioè trasmette il pacchetto prima dell'effettivo scadere del timer ed entra nella modalità fast recovery. Durante la fase di congestion avoidance TCP incrementa di 1 MSS cwnd ad ogni RTT. Se si verifica un timeout CA si comporta come slowstart. Se invece vengono rilevati 3 ACK duplicati, cwnd viene dimezzata e a SISTRESH viene imposto il valore della metà di cwnd al momento del rilevamento degli ACK. Si entra quindi nella fase di fast recovery nella quale cwnd viene incrementato di 1 MSS ad ogni ACK duplicato ricevuto relativamente al segmento perso che ha causato l'entrata di TCP in fast recovey. Se si verifica un timeout, il segmento perso è ritrasmesso e si passa da fast recovery a slow start impostando a 1 MSS cwnd e a metà del valore di cwnd quando si è ritrovato l'evento di perdita per la variabile SISTRESH.

Descrivere come viene realizzato il controllo di flusso in TCP.

TCP offre un servizio di controllo di flusso alle proprie applicazioni per evitare che il mittente saturi il buffer del ricevente. Il controllo di flusso è pertanto un servizio di confronto sulla velocità, dato che paragona la frequenza di invio del mittente con quella di lettura dell'applicazione ricevente. TCP fa mantenere al mittente una variabile chiamata finestra di ricezione che gli fornisce un'indicazione dello spazio libero disponibile nel buffer del destinatario. Dato che TCP è full-duplex, i due mittenti mantengono finestre di ricezione distinte. Nel buffer del ricevente TCP ci sarà una spaceroom o finestra libera che coinciderà con la finestra libera del

ricevente rcwnd. Rcwnd si calcolerà come RcvBuffer - [LastByteRcvd - LastByteRead]. A sua volta il mittente terrà conto di due variabili LastByteSent e LastByteAcked, la cui differenza dovrà essere inferiore a rcwnd, altrimenti andrà in overflow. Il ricevente comunica al mittente la grandezza della sua rcwnd includendola nel pacchetto ACK che invia al mittente in modo che il mittente limiti l'invio di pacchetti.

Descrivere le caratteristiche principali di una rete Ethernet.

Ethernet venne concepita a metà degli anni '70 da Bob Metcafe e David Boggs. Originariamente prevedeva un cavo coassiale come bus per connettere i nodi. Ethernet con tipologia a bus è una LAN broadcast, in quanto tutti i frame trasmetti sono elaborati da tutte le schede di rete collegate al bus. Successivamente a metà degli anni '80, questa tipologia venne sostituita con una avente un collegamento a stella, basata su un hub, un dispositivo a livello fisico che agisce sui singoli bit, piuttosto che sui frame. Quando un bit arriva ad un'interfaccia, l'hub rigenera il bit amplificando la sua potenza e trasmettendolo a tutte le interfacce. All'inizio degli anni 2000 l'hub venne sostituito da un nuovo dispositivo, lo switch, il quale non solo è privo di collisioni ma è anche un vero e proprio commutatore sore e forward. Andiamo ad analizzare la struttura del pacchetto Ethernet; esso prevede sei campi: preambolo (8 byte), indirizzo di destinazione (6 byte), indirizzo sorgente (6 byte), tipo (2 byte), campo di dati (da 46 a 1500 byte) e controllo a ridondanza ciclica (4 byte). IL preambolo prevede i primi 7 byte i quali svolgono la funzione di "risvegliare le schede dei riceventi e sincronizzare i loro clock con quelli del trasmittente" e gli ultimi 2 bit dell'ottavo byte che hanno la funzione di avviare la scheda ricevente che i dati veri e propri stanno per arrivare. L'indirizzo di destinazione contiene l'indirizzo MAC della scheda di destinazione. L'indirizzo sorgente che include l'indirizzo MAC della scheda che trasmette il pacchetto. Il campo di dati contiene il datagramma, solitamente IP, ma può contenere anche altri tipi di pacchetto a livello di rete. Dato che l'unità massima di trasporto (MTU) è di 1500 byte per Ethernet, se il datagramma supera questo valore, l'host frammenta il datagramma. Se il datagramma è inferiore alla dimensione minima del campo dati, il campo dovrà essere riempito (stuffed) fino a raggiungere quel dato valore. Il tipo consente ad Ethernet di supportare vari protocolli di rete. Il campo tipo è assimilabile al campo protocollo nel datagramma del livello di rete e ai campi numero di porte nel segmento del livello di trasporto. Il CRC consente alla scheda di rete ricevente di rilevare la presenza di un errore nei bit del frame. A livello di rete, tutte le tecnologie Ethernet forniscono un servizio senza connessione, nel senso che quando una scheda di rete vuole inviare un datagramma a un host della rete non fa altro che incapsularlo in un frame Ethernet e immetterlo nella LAN. Tutte le tecnologie Ethernet forniscono un servizio non affidabile a livello di rete.

Descrivere il caching nell'applicazione WEB (WEB-caching).

Il WEB caching è la caching di documenti WEB sviluppatosi in Internet per due ragioni. La prima è che un proxy può ridurre in modo sostanziale i tempi di risposta alle richieste del client, in particolare se l'ampiezza di banda che costituisce il collo di bottiglia tra il client e il server di origine è molto inferiore rispetto all'ampiezza di banda minima tra client e proxy. Se esiste una connessione ad alta velocità tra il client e il proxy e se l'oggetto è nella cache allora questa sarà in grado di trasportare l'oggetto rapidamente al client. La seconda ragione è che l'utilizzo di proxy riduce sostanzialmente il traffico sul collegamento di accesso in Internet, con il vantaggio di non dover aumentare l'ampiezza di banda frequentemente e ottenere quindi una riduzione dei costi. Una WEB cache, nota anche come PROXY SERVER, è un'entità di rete che soddisfa richieste HTTP al posto del WEB server effettivo. Il proxy ha una propria memoria su disco (una cache) in cui conserva copie di oggetti recentemente richiesti. Una volta configurato opportunamente il browser dell'utente, in modo che tutte le richieste HTTP dell'utente vengano dirette innanzitutto al proxy server, ogni richiesta di oggetto perviene al proxy. Il proxy è contemporaneamente server e client: quando riceve richieste da un browser e gli invia risposte agisce da server, quando invia richieste e riceve risposte da un server di origine funziona da client.

Descrivere il funzionamento di un router NAT. Illustrare inoltre, con riferimento all'architettura dell'applicazione Skype, come sia possibile realizzare connessioni tra client situati entrambi dietro un router NAT.

I router NAT sono una delle implementazioni della tecnica NAT. Consiste nel modificare gli indirizzi IP dei pacchetti in transito su un sistema che instrada i pacchetti all'interno di una comunicazione tra due o pià host. Il router presenta un unico indirizzo IP omettendo i dettagli della rete domestica al mondo esterno. Esso acquisisce il suo indirizzo IP dal suo server DHCP la cui funzione è fornire gli indirizzi ai calcolatori presenti all'interno della rete domestica, verifica l'indirizzo del destinatario: se esso è in locale, instrada direttamente il pacchetto, se invece il destinatario si trova al di fuori della rete locale il router dapprima segna nella tabella di traduzione NAT gli indirizzi IP della sorgente e del destinatario e il numero di porta generato nell'occasione ponendo per buoni solo i numeri di porta liberi presenti nella sua NAT Translation Table. Dopodiché una richiesta HTTP che se presente invierà una risposta. Ricevuta la risposta con i dati richiesti il router NAT sostituirà l'indirizzo IP di destinazione con quello dell'host dietro NAT e la porta con quella del richiedente precedentemente memorizzata in tabella. Infine instraderà la risposta al richiedente. Nel caso dell'applicazione Skype è possibile attraverso un meccanismo relaging, ovvero il client esterno si connette a un relag server di proprietà Skype, che a sua volta invia tutto il traffico al client dentro il NAT.

Illustrare il funzionamento di BGP.

Il protocollo BGP (Border Gateway Protocol) si occupa di determinare percorsi tra sorgente e destinazione che interessano più sistemi autonomi. BGP, per ciascun sistema autonomo, mette a disposizione un modo per: ottenere informazioni sulla raggiungibilità delle sotto-reti da parte dei sistemi confinanti; propagare le informazioni di raggiungibilità a tutti i router interni a un sistema autonomo; determinare percorsi buoni verso le sotto-reti sulla base delle informazioni di raggiungibilità e delle politiche del sistema autonomo. I router ai capi di una connessione TCP sono chiamati BGP peer, e la connessione TCP con tutti i messaggi BGP è chiamata sessione BGP. Questa divide in sessione BGP esterna (eBGP) se coinvolge due sistemi autonomi, mentre quella tra router dello stesso sistema autonomo è chiamata sessione BGP interna (iBGP). BGP consente di conoscere quali sono le destinazioni raggiungibili attraverso sistemi autonomi vicini, in BGP le destinazioni non sono host ma prefissi (CIDR) che rappresentano una sotto-rete o una collezione di sotto-reti. Quando un gateway di un qualsiasi sistema autonomo riceve prefissi appresi tramite eBGP, utilizza le proprie sessioni iBGP per distribuire i prefissi agli altri router del sistema autonomo. In BGP un sistema autonomo viene identificato dal suo numero di sistema autonomo globalmente univoco. Quando un router annuncia un prefisso per una sessione BGP, include anche un certo numero di attributi BGP. In gergo un prefisso, insieme ai suoi attributi è detto rotta (route). I peer BGP scambiano annunci di rotte. Due attributi più importanti sono AS-PATH e NEXT-HOP. AS-PATH elenca i sistemi autonomi attraverso i quali è passato l'annuncio del prefisso. NETX-HOP indica l'indirizzo IP del router di bordo del sistema autonomo che deve essere usato come next hop verso la destinazione specificata.

Descrivere il funzionamento del protocollo RIP.

RIP è un algoritmo distance vector e di tipo distance metric, ovvero tutti i collegamenti hanno costo unitario. Il costo massimo di un percorso è limitato a 15. In RIP i router adiacenti si scambiano gli aggiornamenti di instradamento approssimativamente ogni 30 secondi. Questi aggiornamenti contengono degli elenchi comprendenti fino a 25 sottoreti di destinazione all'interno del sistema autonomo, nonché la distanza del mittente rispetto a ciascuna di tali sottoreti. Ciascun router mantiene una tabella di instradamento che include vettore delle distanze e tabella di inoltro, ossia la sottorete di destinazione, il router successivo lungo il percorso più breve verso la destinazione e la terza indica il numero di HOP per giungere alla sottorete di destinazione lungo il percorso più breve. Se un vicino non comunica per 180 secondi viene considerato morto, ossia i percorsi che passano da quel router sono considerati invalidi, viene inviato un nuovo advertisement ai vicini, e questi a turno inviano a loro volta nuovi advertisement se le loro tabelle vanno a modificarsi. La

notizia si sparge in tutta la rete per far in modo che ci siano cicli di informazione. Vengono compiuti al massimo 15 passi. Le tabelle vengono gestite da un processo a livello applicativo chiamato routed. Gli ads vengono mandati in pacchetti UDP e ripetuti periodicamente. Il messaggio inviato presenta le seguenti dimensioni: 8 byte di headder UDP, 4 byte di header RIP, 20 byte per 25 router. Per un pacchetto di massimo 512 byte, 25 router sono pochi per trasferire una intera tabella di routing, perciò verranno inviati più pacchetti. Al momento dell'invio, il router manda una speciale richiesta RIP a tutte le sue interfacce chiedendo a tutti i router le loro tabelle di routing, sapendo in questo modo i suoi router adiacenti.

Descrivere come viene effettuato il demultiplexing di livello 4 in Internet.

Per demultiplexing si intende il compito di trasportare i dati dei segmenti dei livelli di trasporto verso la giusta socket, decodificata dell'header contenente la provenienza e la destinazione del pacchetto aggiunto nella fase di multiplexing. In UDP la socket viene identificata da indirizzo IP e numero di porta: quando riceve un pacchetto, l'host ne osserva la porta di destinazione e instrada il pacchetto alla socket corrispondente alla porta di destinazione indicata. Di conseguenza se due segmenti UDP presentano diversi indirizzi IP e/o diversi numeri di porta di origine, ma hanno lo stesso indirizzo IP e lo stesso numero di porta di destinazione, saranno diretti allo stesso processo di destinazione tramite la medesima socket. In TCP la socket è identificata da 4 campi: indirizzo IP di origine, numero di porta di origine, indirizzo IP di destinazione e numero di porta di destinazione. L'host utilizza questi quattro valori per dirigere il segmento verso la socket appropriata ma, al contrario di UDP, due segmenti TCP in arrivo, aventi indirizzi IP di origine o numeri di porta di origine diversi, saranno sempre diretti a due socket differenti, anche a fronte di indirizzo IP o porta di destinazione uguali, con l'eccezione dei segmenti TCP che trasportano la richiesta per stabilire la connessione.

Descrivere le diverse modalità per la ricerca sulle reti P2P.

La ricerca di informazioni in P2P prevede diverse modalità; innanzitutto vi è un indice di peer con il proprio indirizzo IP e il proprio numero di porta. L'indice dei peer traccia dinamicamente l'indirizzo dei file condivisi dei peer con di conseguenza ogni peer che deve dare all'indice quali file è intenzionato a mettere in condivisione. I peer quindi per scaricare un file dovranno cercarlo all'interno dell'indice. Tra le modalità di ricerca vi è quella dell'indice centralizzato, ovvero un indice su un server che contiene indirizzi IP e file condivisi dal peer che funge da tramite ai peer in altre parole per scaricare un file del peer prima bisogna connettersi ai peer e poi chiedere la connessione con il peer selezionato. Questa modalità presenta diversi problemi tra cui colli di bottiglia nelle prestazioni e violazione di copyright. Vi è poi il query flooding. Totalmente decentralizzato, ogni peer indicizza i file che rende disponibili. I peer che effettuano la richiesta la inoltrano a tutti i peer per cui sono connessi e continuano così fino a quando qualcuno non risponde. Dal query flooding passiamo al DHT, tabella hash distribuita, che ogni peer potrà interrogare attraverso l'utilizzo di una chiave specifica; il database distribuito localizzerà i peer aventi chiave e valore corrispondenti e le resituirà al peer che ha fatto l'interrogazione. DHT è decentralizzato, sensibile e tollerante ai guasti. Infine vi è l'overlog gerarchico o DHT circolare, a metà tra indice centralizzato e query floding, e funziona in modo che ogni peer può essere un supernodo o connesso a un supernodo. Vi è connessione tra un supernodo e i suoi peer e tra diversi supernodi. Ogni supernodo conosce i file dei peer "figli" connessi a lui.

Descrivere il funzionamento del protocollo di accesso al mezzo (MAC) nelle reti Ethernet.

I protocolli MAC si dividono in 3 grandi categorie: quelli di suddivisione del canale, quelli ad accesso casuale e i protocolli a rotazione. I primi hanno la caratteristica di suddividere il canale in porzioni, che possono basarsi su tempo, frequenza, ecc., per un uso esclusivo vi è il caso di TDMA e FDMA. Quelli ad accesso casuale non presentano parametri specifici, ovvero il nodo se ha pacchetti da mandare li trasmette; se due nodi trasmettono insieme, si ha una collisione. Questi diversi protocolli hanno la funzione di scoprire le collisioni e di riprendere la trasmissione dopo la collisione tra questi ricordiamo CSMA/CD, CSMA/CA e SLOTTED ALOHA. Infine vi sono i protocolli di rotazione i quali cercano di mediare e unire i benefici delle altre 2 categorie. Tra questi ricordiamo il protocollo polling nel quale uno dei nodi designato come principale,

interpella a turno gli altri indicano il massimo numero di frame da poter inviare e se nota mancanza di segnale sul canale capisce che il nodo ha terminato. Il polling evita le collisioni avendo un'efficienza elevata ma al tempo stesso si hanno degli svantaggi quali ritardo di polling ovvero il tempo richiesto per notificare ai nodi il permesso di trasmettere e la debolezza ai guasti poiché se il nodo principale va giù, l'intero canale diventa inattivo. Vi è poi il protocollo token ring nel quale non vi è un nodo principale ma un frame di controllo detto token che circola tra i nodi seguendo un ordine prefissato. Se il nodo che ricevee il token non deve trasmettere, gira il token al successivo altrimenti trasmette il massimo numero di frame consentitogli. Il token ring è decentralizzato e altamente efficiente, ma al tempo stesso debole ai guasti; basta infatti che un nodo salti per far andare giù l'intero canale e nel caso che un nodo dimentichi di girare il token al nodo successivo bisogna implementare un valido sistema di recupero che rimetta in circolazione il token.

Descrivere l'architettura ed il funzionamento dei DNS.

Il DNS è un servizio dello strato applicativo, di tipo client-server e offerto agli altri protocolli di pari lv, il cui compito è la traduzione di host-name in indirizzi IP; con DNS si fa riferimento al protocollo che interroga un database distribuito e strutturato gerarchicamente (l'albero dei vari DNS-server), ma anche al database stesso. Esistono sostanzialmente 3 tipi di DNS-server:

1) Root server; i server radice sono in realtà un cluster ("grappolo") di server collegati, e sono 13 in tutta la rete;

2) Top-level Domain server (TLD) che si occupano si domini di primo livello come .org, .com e domini nazionali;

3) Server autorevoli associati ad organizzazioni di vario genere; devono fornire i record DNS di tutti gli host dell'organizzazione.

Ogni ISP ha inoltre un server locale (default name server). Generalmente affinché un host A sia in grado di inviare informazioni ad un host B nella rete, deve prima conoscere l'indirizzo IP di quest'ultimo: il lato client dell'applicazione DNS di A invia una query ad un DNS-server locale e attende la risposta; la sequenza di query successive può seguire un approccio iterativo oppure ricorsivo. Se l'approccio è ricorsivo, ogni server inoltra la query al server al livello superiore nell'albero fino al server radice, che rimanda la richiesta verso il basso nella direzione del server che detiene il record relativo a B; questo server invia una risposta, che segue il percorso inverso della query fino all'host A. Se invece l'approccio è iterativo, ogni server informa il server locale di A su quale sarà il successivo da dover interrogare per avvicinarsi all'host B, senza tenersi ulteriormente impegnato al passaggio d'informazioni. Il DNS sfrutta il caching per migliorare le prestazioni e ridurre l'impatto sulla rete, ma tiene in memoria i record per 2 giorni al massimo.

Un RR (resource record) contiene 4 campi: (Name, Type, Value, TTL). Il Time To Live, definisce quanto altro tempo il RR viene tenuto nella cache.

- Se Type=A allora Name è il nome dell'host e Value il suo IP;
- Se type=NS, Name è un dominio e Value è l'host-name del server autoritativo di quel dominio:
- Se Type=CNAME allora Value è il nome canonico dell'host il cui sinonimo è Name;
- Type=MX allora Value è il nome canonico di un mail-server il cui sinonimo è NAME.

Il messaggio DNS è provvisto di:

- un'intestazione, che contiene un identificatore di richiesta, un campo flag per il tipo di messaggio (domanda/risposta, autorevolezza del server, richiesta di ricorsione), e altri 4 campi inerenti al numero di occorrenze per i tipi di dato (A, NS, CNAME, MX);

- Sezione delle domande, per informazioni sulle richieste effettuate;
- Sezione delle risposte, che contiene il/i RR;
- Sezione autorevole contiene record di server autorevoli;
- Sezione aggiuntiva, per altri record utili.

Descrivere il problema del controllo della congestione e le diverse soluzioni nelle reti a pacchetto.

Il problema della congestione sopraggiunge quando la velocità di invio dei pacchetti è inferiore alla velocità di ricezione dei pacchetti nel buffer di output. Se duratura potrebbe portare alla perdita di pacchetti. Le concause della congestione nelle reti a commutazione di pacchetto sono l'utilizzo di un protocollo store-and-forward e la dimensione finita del pacchetto. Il protocollo store-and-forward prevede che l'invio del pacchetto a destinazione possa avvenire solo dopo l'avvenuta ricezione e salvataggio dell'intero pacchetto in entrata. Per questo motivo con l'arrivo di molti pacchetti il buffer tende a riempirsi rallentando di molto la trasmissione. Per ovviare a questi problemi si potrebbe innanzitutto operare sul protocollo da usare. Infatti a discapito del protocollo SF si può usare un protocollo CT ovvero non appena il pacchetto inizia ad essere letto, in particolare il suo indirizzo di destinazione, viene subito inviato al destinatario senza aspettare il termine della lettura e del salvataggio. Questo abbatte i ritardi di trasmissione, limitandoli alla sola latenza nella lettura e nell'invio effettivo del pacchetto. Altra soluzione usata nel protocollo TCP è la limitazione dell'invio dei pacchetti ovvero la sorgente si autoregola sulla quantità dei dati da inviare, evitando quindi congestione o perdita di pacchetti.

Descrivere il funzionamento delle reti peer to peer.

La rete P2P è un'architettura con dipendenza minima dal server questo poiché i peer, ossia end-systems, possono comunicare tra loro attraverso una connessione intermittente e con la possibilità di cambiare il proprio indirizzo. La distribuzione in P2P prevede che ciascun peer possa redistribuire qualsiasi parte del file ricevuto condividendo ed aiutando il server nel processo di distribuzione. All'inizio solo il server dispone del file, perciò deve inviare ciascun bit almeno una volta alla comunità di peer. Successivamente entreranno in gioco i peer che redistribuiranno ciò che hanno ricevuto dal server, incrementando così la capacità del sistema di distribuire il file in breve tempo. In uno dei client P2P più famosi, BitTorrent, la distribuzione avviene nel seguente modo: i peer che partecipano alla distribuzione del file vengono catalogati e inseriti in un file speciale speciale chiamato torrent. Quando un peer entra a far parte della distribuzione del file viene aggiunto al tracker, ossia un nodo di infrastuttura del torrent, che traccierà i peer che via via partecipano al torrent. All'interno del torrent i peer si scambiano chunks (256 kB). All'inizio i peer non hanno chunks ma li accumuleranno col passare del tempo. Si registrano al tracker per avere la lista dei peer e si connettono a questi. Mentre scaricano, ognuno effettua un upload dei chunks che fa entrare nuovi peer. Non appena un peer finisce il download può rimanere (in gergo: in "seeding") e contribuire all'uploading oppure lasciare il torrent cancellandosi dal tracker. I peer non seguono un ordine preciso nel distribuire i chunks ma, collegandosi con gli altri peer, acquistano la loro listadei chunks che possiedono e richiedono quelli che non hanno, basandosi sulla regola del "rarest first" ovvero il chunk con meno copie nei vicini va acquisito per primo. BitTorrent, grazie ad un algoritmo di "trading", attribuisce maggiore priorità ai peers vicini che inviano chunks a velocità più alta (detti "unchocked peer"). Ogni 30 secondi viene scelto un peer casualmente, detto "optimistically unchocked", che potrà diventare unchocked peer oppure non diventarvi.

Descrivere il funzionamento e le caratteristiche del protocollo ALOHA.

L'ALOHA è un protocollo di rete a livello MAC ovvero livello 2. Consiste in due versioni: la pura e la slotted. Innanzittto bisogna fare delle dovute assunzioni: i frame possiedono le medesime dimensioni e il tempo è

equicondiviso in parti uguali detti slot; i nodi cominciano a trasmettere solo all'inizio di uno slot temporale e sono sincronizzati, di conseguenza se due o più nodi trasmettono nello stesso slot di sicuro collideranno. Quando un nodo ottiene un frame lo trasmetterà nel prossimo slot; se non ci sono collisioni nel prossimo slot, il nodo può trasmettere il frame successivo; se invece ci sono collisioni, il nodo ritrasmette lo stesso frame su ogni slot successivo con probabilità p, finché la trasmissione non avviene con successo. L'efficienza dello slotted-ALOHA è nel caso migliore del 37%. ALOHA-puro non ha invece slot di tempo, è più semplice, e non ha sincronizzazione tra i nodi. Quando il frame viene trasmesso immediatamente, la probabilità di collisione aumenta. Nel caso migliore ALOHA-puro ha un'efficienza del 18%, circa la metà della versione slotted.

Descrivere il funzionamento del protocollo di accesso CSMA/CD nelle reti Ethernet.

È un protocollo di accesso multiplo con rilevamento della portante con rilevamento delle collisioni. Un protocollo CSMA/CD esegue una serie di operazioni effettuate attraverso una scheda di rete collegata a un canale broadcast. Ha il compito di ottenere un datagramma a livello di rete e preparare i frame a livello di collegamento che vengono bufferizzati. Come prima cosa deve riscontrare che un canale è libero, effettuando il carrier sensing, e nel caso in cui non risulti occupato deve iniziare la trasmissione dei frame precedentemente bufferizzati; se il canale risulta occupato rimane in attesa fino a che non rileva più segnali e solo allora invia il frame. Per fare ciò deve verificare la presenza di eventuali segnali provenienti da altre schede di rete sul canale broadcast. Quando la sceda di rete riesce a trasmettere l'intero frame senza rilevare segnali provenienti da altre schede ha finito il suo lavoro mentre se riscontra energia di segnale durante la trasmissione rilevando quindi una collisione (collision detection), interrompe la propria trasmissione e attende un tempo casuale, alla fine del quale effettuerà il carrier sensing e proverà di nuovo ad inviare il frame. Questo tempo casuale di attesa prima di una ritrasmissione è chiamato tempo di backoff e viene deciso da un algoritmo chiamato algoritmo di attesa binaria esponenziale (binary exponential backof). Questo algoritmo viene utilizzato nelle reti ethernet e funziono in questo modo: quando viene riscontrata l'n-esima collisione durante una trasmissione di un frame, si stabilisce casualmente un valore K nell'insieme {0, 1, 2, …,2^n-1}. Più il numero delle collisioni è alto, più grande sarà l'intervallo da cui K viene estratto. In ethernet, l'intervallo di tempo che un nodo deve aspettare è pari a quello necessario per inviare K volte 512 bit e il massimo valore che può assumere è 10. Questo tipo di algoritmo viene detto algoritmo di attesa binaria esponenziale poiché la cardinalità da cui viene estratto K cresce esponenzialmente con il crescere del numero delle collisioni raddoppiando il suo valore massimo a ognuna di esse.

Descrivere il funzionamento del protocollo GBN.

In GBN il mittente può trasmettere più pacchetti senza dover attendere alcun ACK, ma non può avere più di un dato massimo numero consentito N di pacchetti in attesa di ACK nella pipeline. GBN prevede quindi una finestra di dimensione N sull'intervallo dei numeri di sequenza ammissibili per i pacchetti trasmessi. La finestra scorre non appena viene riscontrato un ACK dal pacchetto più vecchio nella sequenza. Se l'ACK non viene inviato significa che il pacchetto è andato perduto e quindi ci sarà un buco nella sequenza. Il ricevente invierà sempre ACK cumulativi segnalando quindi al mittente la presenza di un buco nella sua finestra di ricezione. GBN implementa un timer di default che il mittente utilizza per il pacchetto più vecchio non riscontrato e che si riazzera al riscontro dell'ACK corrispondente a quel pacchetto. Qualora il timer scade, GBN fa in modo che il mittente invii tutti i pacchetti della finestra di invio che va dal buco alla fine della stessa. In questo modo potrà proseguire la trasmissione e i pacchetti che arrivano fuori sequenza vengono scartati.

Descrivere il funzionamento dell'applicazione di posta elettronica.

L'applicazione di posta elettronica presenta un'infrastruttura a tre livelli: a livello più alto vi sono gli user agent, a livello intermedio i server di posta e al livello più basso il protocollo SMTP. Gli user agent consentono agli utenti di leggere, rispondere, inoltrare, salvare e comporre i messaggi. I mail server contengono al loro interno le caselle di posa. Un messaggio parte dallo user agent per giungere al mail server del mittente per

proseguire fino al mail server del destinatario dove viene depositato nella sua casella. Per accedere ai messaggi della propria casella, l'utente dovrà autenticarsi con il serve ospitante mediante username e password. SMTP invece fa uso del servizio di trasferimento TCP per trasferire mail dal server del mittente a quello del destinatario del mittente a quello del destinatario. SMTP presenta un lato client, in esecuzione sul server del mittente e un lato server, in esecuzione sul server del destinatario. Entrambi i lati possono essere eseguiti su tutti i server di posta. Quando un server invia posta agisce come client SMTP, quando riceve agisce come server SMTP. In sintesi per inviare un messaggio da A a B: A invoca il proprio user agent passandogli tutte le istruzioni, user agent di A invia il messaggio al mail server di A ed è collocato in una coda di messaggi; il lato client SMTP eseguito sul server di A vede il messaggio nella coda e apre una connessione TCP verso un server SMTP in esecuzione sul server di B; dopo un handshaking SMTP, il client SMTP invia il messaggio sulla connessione TCP; presso il mail server di B, il lato server di SMTP riceve il messaggio, che finisce nella casella di B; B invocando il proprio user agent potrà leggere il messaggio. E vissero tutti felici e contenti.

Descrivere l'algoritmo di back-off esponenziale utilizzato nelle reti Ethernet.

L'algoritmo di back-off esponenziale entra in gioco nel protocollo ad accesso casuale CSMA/CD per risolvere la collisione quando due o più nodi collidono. Quando si riscontra l'n-esima collisione durante la trasmissione di un dato frame, viene stabilito in maniera casuale un valore k nell'insieme {0, 1, ...,(2^n)-1}. Più alto è il numero di collisioni, più grande sarà l'intervallo da cui k viene estratto. Questo numero k determina il tempo di attesa, dopo il quale il frame può essere trasmesso. In caso di nuove collisioni la cardinalità dell'insieme da cui è scelto k cresce esponenzialmente, raddoppiando il suo valore massimo. Per questo motivo, si parla di attesa binaria esponenziale.

Illustrare le differenze tra la versione TAHOE e la versione RENO di TCP.

La differenza principale tra TCP TAHOE e TCP RENO è che il primo non distingue tra la situazione in cui gli ACK non riscontrati fanno scadere il timer oppure gli ack duplicati tre volte, cosa che invece fa il RENO. Infatti sia in caso di congestione o di perdita, TAHOE utilizza il medesimo approccio, ossia si salva il valore della finestra dimezzato in una variabile ssthresh e imposta a 1 il valore iniziale della finestra di congestione corrente. In questo modo decongestiona la rete ma limita temporaneamente la velocità di trasmissione di dati TCP RENO invece distingue i due casi: se la perdita viene indicata dallo scadere di timer, viene applicato l'algoritmo di TCP TAHOE, facendo ripartire la trasmissione impostando la finestra al valore minimo di 1 MSS e ricominciando con slow start con crescita esponenziale; se la perdita è generata dalla ricezione di 3 ack duplicati, TCP RENO assume che la rete sia ancora in grado di trasferire qualcosa. In tal caso si entra nella fase di fast recovery, ovvero in una riduzione meno drastica della finestra di congestione. Il valore soglia viene impostato a metà del lavoro della finestra di congestione al momento della ricezione di 3 ack duplicati e la trasmissione riparte impostando il valore di finestra corrente pari al valore di soglia e proseguendo nell'invio con un incremento lineare di 1 MSS ad ogni RTT della finestra di congestione.

Descrivere l'algoritmo di inoltro del frame impiegato dai bridge.

Il bridge esegue l'inoltro di un frame nel seguente modo: cerca di capire dall'indirizzo del destinatario se questo si trova nello stesso segmento oppure no. Nel primo caso evita di inoltrare il frame in quanto presumibilmente il destinatario l'ha già ricevuto per condivisione del bus di comunicazione. Nel secondo caso il bridge inoltra la trama verso il segmento in cui si trova effettivamente il destinatario. Se non sa su quale segmento si trova il destinatario, il bridge inoltra il frame su tutte le porte tranne quella da cui l'ha ricevuta. Per inoltrare il frame verso i giusti domini, il bridge mantiene una tabella di forwarding contenente gli indirizzi MAC per ciascuna porta, e in base al suo contenuto è in grado di capire verso quale porta, e quindi verso quale dominio, inoltrare il frame. La tabella può essere impostata manualmente dall'amministratore o tramite un meccanismo di self-learning, ovvero ogni frame che entra permette al bridge di memorizzare sorgente e destinatario MAC e le interfacce.

Descrivere come vengono organizzate le DHT (tabelle hash distribuite)

Le tabelle hash distribuite vengono utilizzate per costruire una comunicazione P2P in modo che ci sia un database che memorizzi le coppie chiave valore di milioni di peer. In un sistema P2P con DHT ogni peer gestisce un piccolo sottoinsieme della totalità delle coppie e ogni peer può interrogare il database distribuito con una specifica chiave; il database distribuito ha il compito quindi di localizzare i peer che hanno la coppia chiave valore e restituire tali coppie al peer che ha fatto l'interrogazione. In oltre ogni peer può aggiungere nuove coppie chiave valore al database. Per costruire una DHT, come prima cosa viene associato un identificatore a ogni peer, un intero in un intervallo che va da [0, 2^n-1] per n fissato, rappresentabile con n bit. Anche ogni chiave deve essere un intero nello stesso intervallo. Per creare una corrispondenza tra chiavi e intervallo si usa una chiave hash. Una regola per assegnare le chiavi ai peer è quella di associare ogni coppia chiave valore al peer il cui identificatore è più vicino (il successore) alla chiave. Se un peer vuole inserire una nuova coppia di chiave, valore, determina il peer il cui identificatore è più vicino alla chiave e poi gli manda il messaggio per memorizzare la coppia. Per affrontare il problema della scalabilità di una DHT organizziamo i peer in un cerchio. Così facendo ogni peer tiene traccia del successore e del predecessore immediato. Questo schema circolare di peer forma una rete overlay. In una rete overlay ogni volta che un peer riceve un messaggio di richiesta per trovare una determinata chiave, conoscendo il suo successore e il suo predecessore può identificare se sia il più vicino alla chiave in questione; se il peer non è il più vicino si limita a mandare il messaggio al suo successore. Un approccio del genere non è efficiente perché nel caso peggiore vengono inviati N/2 messaggi. Un metodo per rendere più efficiente una rete overlay è quello di introdurre delle scorciatoie, in modo che ogni peer non tiene conto solo del predecessore e del successore ma anche di un numero piccolo di peer disseminati nel cerchio. In questo modo quando un peer riceve un messaggio lo inoltra direttamente al peer più vicino alla chiave. Reti organizzate in questo modo possono essere realizzate in tempo O(log N) dove N è il numero di peer .Un altro problema è quello di verificare periodicamente i peer attivi poiché in una rete P2P i peer possono andare via e tornare quando vogliono. A fronte di questo problema ogni peer non memorizzano solamente il predecessore e il successore ma anche il secondo successore e il secondo predecessore. In questo modo se un peer esce dalla rete, il suo predecessore e il suo successore sostituiscono i loro primo successore e predecessore con il secondo e richiedono al peer quello successivo e precedente per aggiornare la propria lista di peer conosciuti.

Descrivere come viene calcolato il timer di ritrasmissione nel protocollo TCP

Per calcolare il timer di ritrasferimento in un protocollo TCP, il primo accorgimento che dobbiamo fare e che il timer dovrebbe essere maggiore almeno del tempo del RTT. Indichiamo con simpleRTT il tempo che intercorre tra l'istante dell'invio del segmento e l'istante della ricezione dell'acknowledgment del segmento. In TCP anziché calcolare il simpleRTT per ogni segmento, si effettua una sola misurazione del simpleRTT alla volta; in ogni istante di tempo, il simpleRTT viene valutato per uno solo dei segmenti trasmessi per cui non si è ancora ricevuto l'acknowledgement, che comporta una misurazione di un nuovo simpleRTT per ogni RTT. Il valore di simpleRTT può variare a causa della congestione sui router e in base al diverso carico sui sistemi periferici, per questo si vuole effettuare una media dei valori di simpleRTT che chiamiamo EstimateRTT. Quando si ottiene un nuovo simpleRTT la media viene aggiornata secondo la formula:

EstimateRTT = (1 – a) * EstimateRTT + a*simpleRTT.

Quindi il nuovo valore della media dipende da una combinazione ponderata del suo precedente valore e del nuovo valore di simpleRTT (a=0,125). Oltre ad avere una stima per la media, è importante conoscere anche la varianza che chiamiamo devRTT.

devRTT = (1 - b) * devRTT + b * |simpleRTT-EstimateRTT|.

Se i valori di devRTT sono piccoli allora le fluttuazioni sono limitate mentre in caso di gradi fluttuazioni il valore di devRTT sarà grande. (b = 0,25) Ora ottenuti i valori di EstimateRTT e di devRTT il time-out viene calcolato in questo modo:

TimeoutIntervall = EstimateRTT + 4 * devRTT.

Descrivere il meccanismo dell'inversione avvelenata ("poisoned reverse") del protocollo RIP.

Il problema dei cicli attraverso un percorso può essere evitato utilizzando una tecnica nota come inversione avvelenata (poisoned reverse). L'idea di base è semplice, se z instrada tramite y per arrivare a x allora z avvertirà y che la sua distanza verso x è infinita anche se non è una cosa vera. Continuerà a dire questa bugia fintanto che instraderà attraverso y per arrivare a x. Dato che y crede che z non abbia un percorso verso x non tenterà mai di instradare verso x passando per z.

Descrivere il protocollo ARP.

Il protocollo ARP (Address Resolution Protocol) ha il compito di convertire indirizzi IP in indirizzi MAC, soltanto per i nodi nella stessa sottorete. Il protocollo ARP è composto da una tabella contenuta dalle corrispondenze IP, MAC e TTL (Time To Live, indica quando bisognerà eliminare una voce dalla tabella). Un nodo, con un certo indirizzo IP, per trasmettere un datagramma ad un altro nodo, costituisce prima uno speciale pacchetto, chiamato pacchetto ARP, composto da diversi campi tra cui indirizzi MAC e IP di chi invia e di chi riceve.

Descrivere il meccanismo degli ack ritardati in TCP.

Il protocollo TCP ha bisogno, alla base del sistema di controllo di ricezione, di acknowledgment. Spesso in TCP si fa uso degli ack cumulativi; una sua variante è quella degli ack ritardati, in gergo "piggyback". Quando il mittente invia al destinario un pacchetto di dati, con il meccanismo degli ack ritardati si può posticipare l'invio dell'ack e inglobarlo nel pacchetto successivo da inviare al mittente. In generale tale meccanismo non è molto efficiente, in quanto produce una crescita elevata dell'RTT. Per questo motivo, implementando il protocollo TCP si preferisce l'uso degli ack cumulativi e non del "piggyback", anche se il protocollo Telnet (basato su TCP) ne fa uso.