Quali sono le differenze tra il paradigma “Client-Server” e il “peer to peer”?

La differenza sostanziale con il paradigma client-server e che non è chiarito a priori chi è il richiedente e chi è il servente, chi è il client e chi è il server, quindi in una rete p2p tutti gli host sono client e server nello stesso momento.
Grazie al modello P2P, i computer possono comunicare e condividere i file e altre risorse, invece di passare attraverso un server centralizzato.
In un sistema C/S le risorse necessarie per fornire il servizio sono concentrate nei server, che è l'unica entità che può fornire il servizio. Ad esso vengono affidati molti client, per questo all'aumentare del numero di richieste, le prestazioni peggiorano e In caso di failure del server il servizio non è più disponibile, invece se un peer si disconnette, il servizio continua comunque ad essere fornito dagli altri peer poichè le entità partecipanti condividono le proprie risorse per contribuire attivamente alla fornitura del servizio, la cui qualità dipende dalle risorse che ogni peer autonomamente mette a disposizione quindi il servizio è distribuito e decentralizzato.Le connessioni nei P2P devono essere richieste da una delle parti in causa. A differenza dell'approccio client-server, chi richiede la connessione non è ad un livello gerarchico inferiore; infatti entrambi i partecipanti sono alla pari (peer).
Nei C/S la risorsa è localizzata facilmente perché è noto a priori l'identità del server mentre nei sistemi P2P gli utenti accedono alle risorse in seguito ad una fase di ricerca e non utilizzando il DNS sono caratterizzati da connettività tipicamente non permanente. La decentralizzazione del P2P lo rende Scalabile e permette di ridurre i costi grazie alla condivisione, migliora la disponibilità del servizio, l'autonomia (Ogni peer decide cosa e quanto condividere) e la privacy, che nei sistemi C/S è difficile perché il server deve essere individuabile (indirizzo IP).

Quali sono i vantaggi dell’architettura di Gnutella rispetto a quella di Napster?

L’architettura NAPSTER è a DIRETORY CENTRALIZZATA cioè ha una localizzazione centralizzata e un trasferimento tra peer distribuito e presenta diverse limitazioni tra cui:
un Unico punto di guasto, un Collo di bottiglia prestazionale e i Diritti d’autore.
L’architettura GNUTELLA è QUERY FLOODING cioè rappresenta un approccio completamente distribuito per la localizzazione dei contenuti, e quindi va oltre le limitazioni dell’architettura.
Quando un utente vuole localizzare un file, invia in query flooding a tutti i vicini con cui ha stabilito connessioni TCP la richiesta e questi, a loro volta, inoltreranno la stessa richiesta a tutti i loro vicini. I peer che possiedono il file generano una query hit e rispondono alla richiesta ripercorrendo all’indietro la overlay network fino all’host origine. In questo modo l’utente iniziale può contattare uno dei peer che possiedono il file per avviare il trasferimento. Il problema dell’individuazione di peer è detto BOOTSTRAP, devo mantenere l’elenco degli ip oppure posso scaricare questo elenco da un sito. Dobbiamo quindi stabilire una connessione TCP con almeno un peer della lista. Una soluzione comune è quella di mantenere (nel client o su un qualche server remoto) un elenco di peer (indirizzi IP) che sono spesso connessi alla rete di copertura. Una volta effettuato l'accesso alla lista, attraverso la tecnica PING/PONG ricevo informazioni (indirizzo IP etc. etc.) su molti host adiacenti a quell’host, con una parte di questi stabilisco una connessione TCP. Utilizzando il query flooding posso scambiare informazioni coi nodi prossimi a lui per ottenere informazioni riguardo al percorso che il messaggio dovrà percorrere per raggiungere il nodo destinatario. Il percorso si crea dunque dinamicamente, a discapito di uno scambio di dati oneroso. Si evita però cosi il collo di bottiglia, che invece in napster rappresenterebbe un problema facendo tutti riferimento alla directory centrale. Di conseguenza anche il problema dei guasti è bypassato dal fatto che le informazioni si recuperano dinamicamente e da varie fonti.

A cosa serve il TTL (time to live) nella fase di ricerca in una rete p2p (peer to peer)? Per quale motivo è importante?

L'architettura QUERY FLOODING rappresenta un approccio completamente distribuito per la localizzazione dei contenuti. Nelle reti p2p decentralizzate si presenta un problema di scalabilità, in quanto, quando un peer procede con una richiesta di ricerca si genera molto traffico nella rete Internet. Per risolvere il problema i progettisti di Gnutella ad esempio sono ricorsi al QUERY FLOODING A RAGGIO LIMITATO, si è pensato di limitare l'inondazione di richieste ad un raggio stabilito. Hanno inserito un campo contatore nel messaggio di richiesta ed ogni volta che il messaggio di richieste viene inoltrato ad un altro nodo, il valore del TTL (time-to-live) viene decrementato di uno. Se il TTL > 0 allora il messaggio è valido, altrimenti non viene preso in considerazione e quando raggiunge lo zero il query flooding si interrompe. Lo svantaggio di questa soluzione è che non vengono più contattati tutti gli utenti e dunque vi è minor probabilità di trovare i file.

Quali sono le principali differenze tra il protocollo HTTP e il protocollo SMTP?

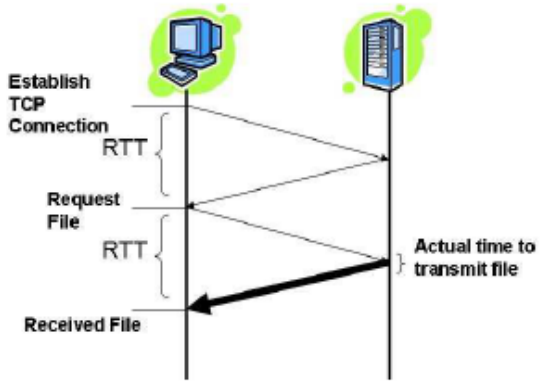
Entrambi utilizzano connessioni persistenti TCP, su porte diverse naturalmente. Il protocollo http riguarda il trasferimento file interet in generale, Smtp invece è specifico di messaggi mail (di tipo posta elettronica), comunicazione tra server mail. L http è un pull protoll, viene utilizzato per scaricare oggetti caricati su un server. l'HTTP e si riferisce alla comunicazione tra web server e client. Il client si preoccupa di fare la richiesta al server per le informazioni. Smtp è push protocol, serve a caricare oggetti su un server (dal server di posta di invio al server di posta di ricezione, dallo user agent al server di posta di invio), si preoccupa della comunicazione tra due server mail. Il mittente manda il messaggio sul suo server che si preoccupera di inviarlo al server del destinatario.
Smtp mette tutti gli oggetti in un unico messaggio mentre http incapsula ogni oggetto in un messaggio. SMTP converte tutto in ASCI a 7 bit e sfrutta il protocollo MIME (multipupose internet mail extension) per il file multimediale e poi incapsula in pacchetti TCP, mentre HTTP incapsula il tutto in pacchetti TCP senza effettuare nessuna conversione.

Quali sono le differenze tra HTTP e SMTP nella gestione di un documento che contiene sia testo che immagini?

Entrambi utilizzano connessioni persistenti TCP, su porte diverse naturalmente. Il protocollo http riguarda il trasferimento file interet in generale, Smtp invece è specifico di messaggi mail (di tipo posta elettronica), comunicazione tra server mail. L http è un pull protoll, viene utilizzato per scaricare oggetti caricati su un server. l'HTTP e si riferisce alla comunicazione tra web server e client. Il client si preoccupa di fare la richiesta al server per le informazioni. Smtp è push protocol, serve a caricare oggetti su un server (dal server di posta di invio al server di posta di ricezione, dallo user agent al server di posta di invio), si preoccupa della comunicazione tra due server mail. Il mittente manda il messaggio sul suo server che si preoccupera di inviarlo al server del destinatario.
Smtp mette tutti gli oggetti in un unico messaggio mentre http incapsula ogni oggetto in un messaggio. SMTP converte tutto in ASCI a 7 bit e sfrutta il protocollo MIME (multipupose internet mail extension) per il file multimediale e poi incapsula in pacchetti TCP, mentre HTTP incapsula il tutto in pacchetti TCP senza effettuare nessuna conversione.

Per quale motivo una connessione http permanente sfrutta meglio il protocollo TCP.

Il tcp è caratterizzato da un handshaking iniziale per instaurare connessione tra le entità coinvolte. Questo richiede del tempo iniziale. Nel caso di http permanente, ad esempio per spedire degli oggetti composti ciascuno di piu pacchetti, viene instaurata una sola connessione iniziale e vengono mandati gli oggetti, nel caso di quella non permanente invece la connessione cadrebbe dopo l'invio di ogni oggetto e sarebbe da instaurare ogni volta nuovamente.



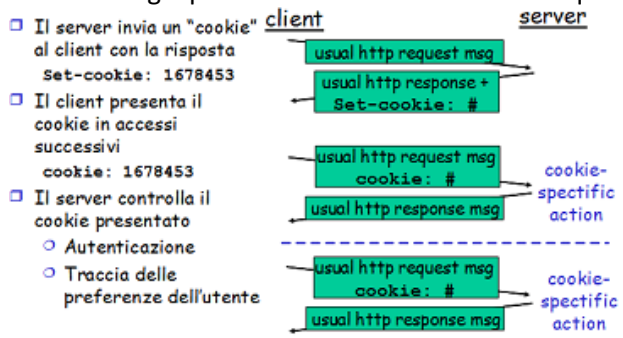
Si descrivano i passi e gli strumenti necessari per inviare una mail falsa a nome di un'altra persona semplicemente utilizzando il protocollo SMTP e si accenni alle possibili contromisure.

Con SMTP il client, una volta collegato al server, comunica tramite il comando MAILFROM< address@mail.it> il mittente.
Con il comando telnet ServerName 25 è possibile comunicare, da linea di comando, direttamente col server di posta senza un agent intermediario: quindi per mandare una mail falsa basta utilizzare il comando MAILFROM mettendo un indirizzo email falso. Per ovviare a questo problema, si fa uso del protocollo PGP (Pretty good privacy) che garantisce l'autenticità tramiteinserimento di un diggest nel messaggio.

Per quale motivo in http sono necessari i cookie? Descrivere cosa sono e come si utilizzano attraverso un semplice esempio.

Essendo il protocollo http stateless (ogni richiesta è indipendente da quelle precedenti e si conclude al momento della chiusura della connessione) i server non devono occuparsi dello stato di ciascuna connessione corrente, tuttavia potrebbe risultare utile per il server web, quindi sono necessari i cookie che vengono registrato in un database locale gestito dal proprio browser.
I cookie sono uno strumento che consente a HTTP di mantenere informazioni di stato fra una richiesta e l'altra ed hanno il vantaggio di essere memorizzati lato client e non lato server.
La tecnologia dei cookie presenta 4 componenti:
1. Una riga di intestazione nel messaggio di risposta HTTP (Set-cookie: <valore>)
2. Una riga di intestazione nel messaggio di richiesta HTTP (Cookie: <valore>)
3. Un file cookie mantenuto sul sistema terminale che esegue il lato client del protocollo
4. un database di back-end nel sito web

Possono essere utilizzati in diversi modi come ad esempio per riempire il carrello della spesa virtuale in siti commerciali o Per personalizzare la pagina web sulla base delle preferenze dell'utente (per esempio il motore di ricerca Google permette all'utente di decidere quanti risultati della ricerca voglia visualizzare per pagina).



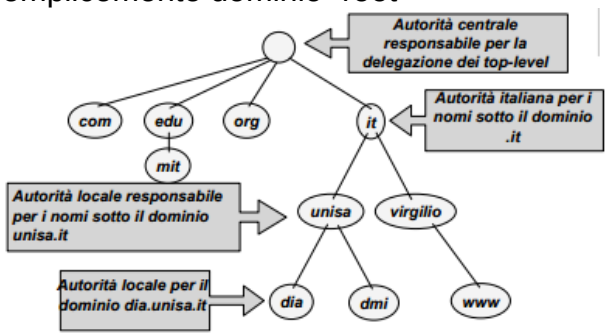
Per quale motivo i protocolli di tipo stop and wait non sono indicati in link ad alta latenza si chiarisca il concetto con un esempio caratterizzato da un link da 1Gbps e un RTT di 0.25s e dimensioni dei pacchetti da 3kb. Quale frazione della banda viene effettivamente usata?

Un protocollo stop and wait richiede un ACK per ogni pacchetto inviato; è il più banale ma implica un RTT (Round Trip Time) per ogni singolo pacchetto che inviamo. Se c'è un errore nella sequenza o una mancata consegna il protocollo dice di fermarsi e aspettare il reinvio, questo aggiunto al fatto che c'è alta latenza renderebbe il tutto inefficiente quando invece nel frattempo con un altro protocollo potrebbero essere inviati e accettati gli altri pacchetti. È quindi richiesto un RTT per pacchetto, se RTT è molto piu grande del tempo di consegna del pacchetto (RTT >> d) il protocollo diventa inefficiente; viene indicato con d il tempo richiesto a
inviare o ricevere un pacchetto.

Il link dell'esempio ha RTT = 0.25s e
d = 3x10^3 / 10^9 = 3x10^-6 s.
Per inviare ogni pacchetto si attende quindi 0.25s + 2*3x10^-6 s, riuscendo ad inviarne circa 4 al secondo, quindi
4*d= 4*3*10^3= 12 kbps.
La frazione della banda è
[12x10^3 / 10^9] * 10^2 = 0.0012%
che è pochissimo.

Descrivere la struttura gerarchica dei DNS e fare un esempio di schema ricorsivo.

I nomi di dominio sono organizzati secondo una struttura ad albero capovolto, che costituisce il "name space". La radice dell'albero è detta "radice non denominata" o più semplicemente dominio "root"

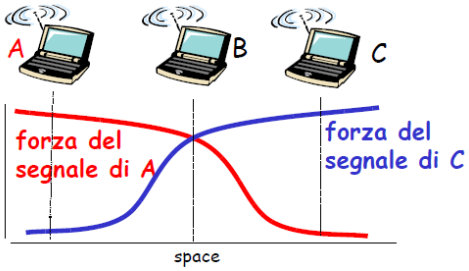


C'è il server radice al quale fanno rifermineto vari domini
A loro volta ci sono sottodomini che fanno rifermineto alle autorità garanti del dominio del livello superiore
Es
1 radice
1.1 com
1.1.1google.com
1.1.2msn.com
1.2 org
1.2.1altervista.org
1.3 net
1.3.1domino.net
Google desidera indirizzo ip di domnio.net risale a 1.1 quindi com risale a 1 radice che va in1.3 net
E trova 1.3.1 domino .net viene restituito a .net, che lo passa a radice, e lo comunica .com che lo porta google.com

Che tipo di meccanismo di instradamento mobile IP usereste nell'ambito della LAN del secondo
piano del dipartimento?
Instradamento diretto, poiché con instradamento indiretto si incontrerebbe il problema del routing triangolare.

Per quale motivo i pacchetti RTS hanno una minor probabilità di collisione rispetto ai pacchetti standard?
Per quale motivo nelle reti cablate non c'è il problema del terminale nascosto?
Come si risolve tale problema nelle reti wireless?

Il problema del terminale nascosto si verifica quando due stazioni A e C sufficientemente distanti da non potersi percepire tentano di comunicare con una terza stazione B posta a metà fra esse. Se una delle due sta già trasmettendo dei dati a B e l'altra inizia di sua spontanea volontà a fare la stessa cosa, si verifica in B una collisione che compromette entrambe le comunicazioni.
Questo fenomeno è noto col termine fading.



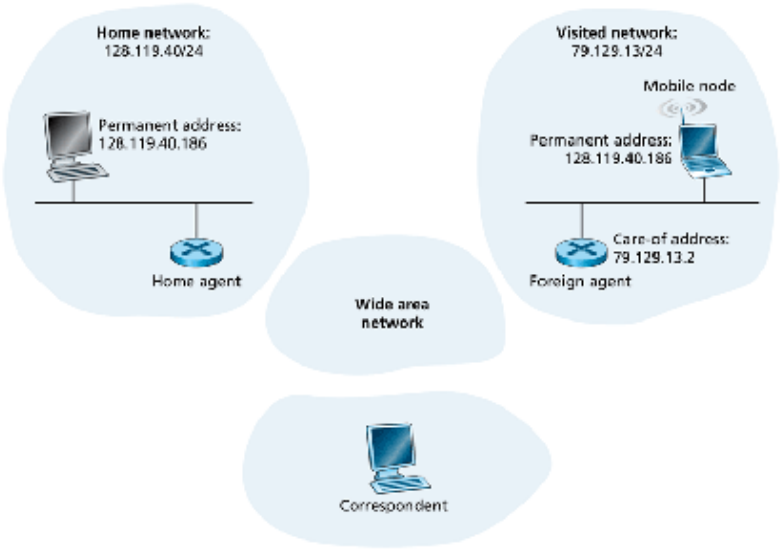
I pacchetti RTS (Request-To-Send) hanno una minor probabilità di collisione rispetto ai pacchetti standard perchè riguardano pacchetti di lunghezza ridotta. Per risolvere il problema del terminale nascosto, il canale può essere prenotato mediante l'introduzione di due nuovi pacchetti: Request-To-Send (RTS) e Clear-to-Send (CTS). Quando il trasmittente vuole inviare un frame dati, aspetta DIFS (Distributed Inter-Frame Space) e, se il backoff lo consente, invia un frame RTS all'Access Point indicando il tempo totale richiesto per la trasmissione (compreso il tempo di ricezione dell'ACK). Quando l'Access Point riceve un frame RTS, aspetta un tempo pari a SIFS (Short Inter-frame Space) e risponde in broadcast con un frame CTS contenente a sua volta la durata complessiva della prenotazione. In questo modo, pur non essendo tutti i terminali consapevoli dell'esistenza degli altri, l'arrivo di un frame CTS permette loro di capire che è in atto una trasmissione di cui essi non riescono ad essere in ascolto. si verificano ancora collisioni nel momento in cui due nodi tentano di prenotare il canale contemporaneamente o ad intervalli molto ravvicinati l'uno dall'altro. La probabilità di collisione e lo spreco di banda sono tuttavia minimi poiché la collisione riguarda pacchetti (RTS e/o CTS) di lunghezza ridotta.

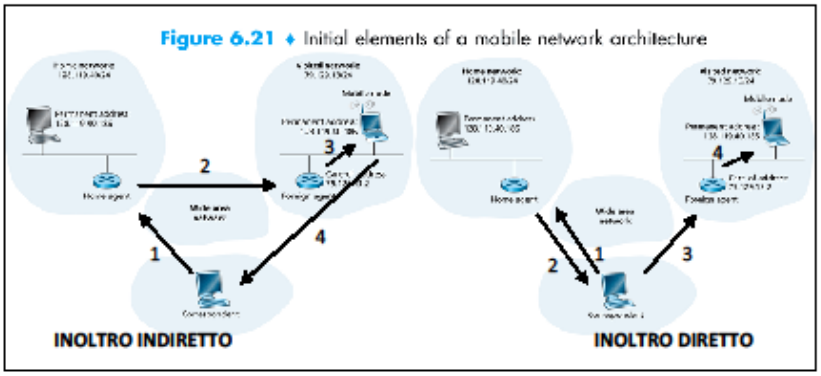
In cosa consiste il problema del "triangolo d'instradamento" nelle reti mobili?

Per quanto riguarda la gestione della mobilità, quando il nodo mobile si sposta da una rete ad un'altra durante una stessa sessione TCP è sufficiente aggiornare il database dell'agente domestico mediante la registrazione di un nuovo COA. Inoltre, l'istante in cui viene interrotta la vecchia connessione e instaurata una nuova è sufficientemente breve da comportare la perdita di un numero molto limitato di pacchetti: TCP provvederà a recuperare tale perdita chiedendo la ritrasmissione dei datagrammi persi.
Questo tipo di instradamento, sebbene completamente trasparente al corrispondente, comporta un'inefficienza nota come il problema del triangolo d'instradamento, è tipico dell'instradamento indiretto: i datagrammi indirizzati al nodo mobile devono prima essere instradati all'agente domestico e poi alla rete ospitante, anche in presenza di rotte più efficienti tra il corrispondente e il nodo mobile. Il caso estremo è quando i due nodi si trovano nella stessa sottorete, ma sono ignari della loro posizione reciproca, ad esempio un professore in visita presso un suo collega in un'altra università. I due sono seduti sulla stessa scrivania e si scambiano dati attraverso la rete. In tal caso i datagrammi dell'uno sono instradati all'agente domestico dell'altro e poi rimbalzati nella prima rete.

Si consideri la figura 6.21 e la si completi illustrando i principali flussi di informazione relativi all'instradamento diretto ed indiretto di un utente mobile.

Si consideri la figura 6.21 e la si completi illustrando i principali flussi di informazione relativi all'instradamento diretto ed indiretto di un utente mobile.





Si assuma che in coda ad un router siano presenti i pacchetti p1,p2,p3,p4,p5. I pacchetti p1 e p4
sono di una classe C1 di priorità inferiore rispetto alla classe C2 di p2,p3 e p5. Qual è la sequenza di output dal router per gli scheduling: FIFO, PRIORITA' e ROUND ROBIN? Cosa succede nel caso in cui lo scheduling sia di tipo WFQ se il peso della classe C1 è 1 e il peso di C2 è 2?

FIFO: p1 -> p2 -> p3 -> p4 -> p5
PRIORITA: p2 -> p3 -> p5 -> p1 -> p4
ROUND ROBIN:
Dato un certo numero di code ugualmente importanti, ogni coda viene interrogata a turno e, se ci sono pacchetti da trasmettere, ne viene immesso uno sul canale. È in altri termini previsto una sorta di avvicendamento circolare tra le diverse classi, in modo da offrire a tutti i flussi un servizio di trasmissione dei pacchetti totalmente equo e imparziale.
p2 -> p1 -> p3 -> p4 -> p5
Se lo scheduling è di tipo WFQ i pacchetti vengono trasmessi
contemporaneamente ma ad un RATE diverso (la classe con peso maggiore avrà un RATE di uscita più alto rispetto alla classe con peso minore, la banda totale viene suddivisa)
Sapendo che w1 e w2 sono i pesi delle rispettive classi C1 e C2
Ratec1= RateTOT*(w1/w1+w2)= (1/3) RateTOT
Ratec2=RateTOT*(w2/w1+w2)= (2/3) RateTOT

Si consideri un percorso fatto di 3 router che accettano pacchetti di dimensione 1Kb appartenenti a due classi di priorità P1 e P2. Tutti i router implementano la politica di scheduling WFQ e hanno la medesima banda in uscita di 10Mbps. La dimensione delle code assegnate a P1 e P2 è di 10 pacchetti.
1. Assegnare il peso W1 alla classe P1 affinché il ritardo massimo di un suo pacchetto nell'attraversare i 3 router sia di 150ms.
2. Qual è in questo caso il ritardo massimo di un pacchetto che appartiene a P2?

Nel Weighted Fair Queuing è possibile fornire un servizio differenziato ad ogni classe per un dato periodo di tempo. A ciascuna classe è assegnato un peso che è una frazione del peso complessivo di tutte le classi. In qualsiasi momento in cui nella coda di tale classe siano presenti pacchetti da spedire, WFQ garantisce che essi riceveranno una frazione del servizio pari al peso della classe a cui appartengono.
In generale, per un collegamento con frequenza trasmissiva pari a R, la classe i avrà sempre un rendimento almeno pari a:

$$R_i = R \times \frac{w_i}{\sum_j w_j}$$

il ritardo per ciascun router è coda/rate

quindi nel nostro caso abbiamo 3*coda/[rate*(w1/(w1+w2))]=ritMAX1
ritMAXp1=150mSec
(3*10*10^3)/[(10*10^6)(w1/(w1+w2))]=ritMAXp1 w2=4w1
ora abbiamo la proporzione tra i pesi quindi li possiamo determinare: w1=1, w2=4
Una volta che abbiamo i pesi si ricava immediatamente il ritardo massimo per i pacchetti P2, applicando la solita formula:
[3*bufferP2]/[Rate(w2/(w1+w2))]=ritMAXp2
=3,75mSec

Si consideri un leaky bucket di dimensione del secchio =10 pacchetti e token rate = 20 pacchetti al secondo. Nell'ipotesi che il traffico in entrata sia caratterizzato da 1 pacchetto ogni 2mS,
quanti pacchetti escono al più dal leaky bucket?

L'utilizzo del leaky bucket consente così di avere un meccanismo per:
● Individuare le applicazioni che violano i requisiti (analizzando i pacchetti che trasbordano dal buffer associato a quel flusso).
● Proteggere i flussi che viaggiano nel rispetto dei limiti (avviene in maniera trasparente contentendo l'effetto di quelli irregolari).
● Limitare il ritmo di trasmissione medio a quello imposto dal secchio.
In questo modo, l'host può anche produrre un traffico bursty senza creare problemi sulla rete: finché il data rate medio non supera quello tollerato dal secchio, tutto funziona regolarmente; se l'host genera più pacchetti di quelli che possono essere contenuti nei buffer del leaky bucket,
essi si perdono.

b (dimensione del bucket); r (token rate); p (traffico in entrata); N Pacchetti in uscita
Il massimo numero di pacchetti che escono dal leaky bucket nell'unità di tempo è dato da:
N=b+(r*t)
dove t è il tempo (che poniamo pari ad 1);
quindi N = 10+(20*1)=30 pk
Notiamo dunque che nel leaky bucket il numero dei pacchetti uscenti è indipendente dal tasso dei pacchetti in entrata.

Si consideri un percorso fatto di 2 router. Il primo implementa la politica weighted fair queuing (WFQ) con due classi di priorità P1 di peso 2 e P2 di peso 4, entrambe le classi di priorità hanno un buffer di 100 pacchetti. Il secondo implementa la semplice politica FIFO, ha un packet rate in uscita di 1000 pacchetti per secondo ed un buffer di 200 pacchetti. Qual è il packet rate che deve avere il primo router affinché il massimo ritardo di un pacchetto con priorità P1 che attraversa entrambi i router sia 500ms?

Nel WFQ è possibile fornire un servizio differenziato ad ogni classe per un dato periodo di tempo. A ciascuna classe è assegnato un peso che è una frazione del peso complessivo di tutte le classi. In qualsiasi momento in cui nella coda di tale classe siano presenti pacchetti da spedire, WFQ garantisce che essi riceveranno una frazione del servizio pari al peso della classe a cui appartengono.
In generale, per un collegamento con frequenza trasmissiva pari a R, la classe i avrà sempre un rendimento almeno pari a:

$$R_i = R \times \frac{w_i}{\sum_j w_j}$$

Il secondo router ha come parametri:
b_2=200 pk e r_2=1000 pk/s

$R_2 = \frac{200}{1000}$ = 0.2 s = 200 ms

Otteniamo quindi che
$R_1 = R_{TOT} - R_2$ = 300 ms

Sapendo inoltre che

$$R_1 = \frac{\frac{b1}{W1}}{r1.\frac{W1}{W1+W2}} = \frac{100}{r1.\frac{2}{2+4}} = 0.3 \text{ s}$$

Invertendo otteniamo che:

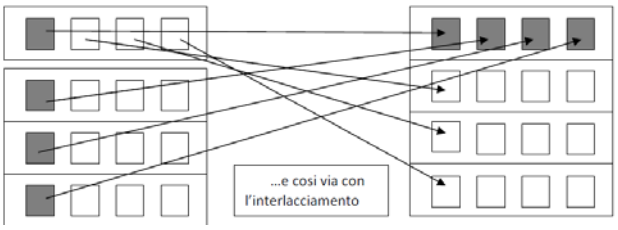
$$r1 = \frac{300}{0.3} = 1000 \text{ pk/s}$$

Per quale motivo l'header dei pacchetti RTP contiene un numero di sequenza ed una marcatura temporale? Discutere in che modo si potrebbe realizzare un protocollo affidabile usando solo datagrammi UDP

RTP consente di trasportare flussi multipli che dipendono dal numero di sorgenti e dalla natura del flusso (solo audio, audio e video). oltre al Payload Type e al Source Identifier ci sono altri 2 principali campi dell'intestazione (header):
Il Sequence Number (16 bit), numero di sequenza, è incrementato di un'unità per ogni pacchetto RTP inviato e può essere utilizzato dal ricevente per rilevare le perdite e ricostruire la sequenza dei pacchetti. Viene utilizzato perché UDP non assicura che i pacchetti arrivino con lo stesso ordine con cui l'abbiamo inviati. Il Timestamp (32 bit)che riporta l'istante del campionamento del primo gruppo di byte nel pacchetto dati. Tale valore può essere utile per combattere lo jitter nel caso si voglia implementare un meccanismo di ritardo di riproduzione fisso o adattivo per fornire la riproduzione sincronizzata. Nel caso della classe di streaming interattivo, il ritardo deve avere un valore tale da non pregiudicare l'interattività della comunicazione (impercettibile sotto ai 150ms, tollerabile entro i 400ms, intollerabile oltre tale valore). Un modo per calcolare tale ritardo consiste nel fornire ciascun pacchetto di una marcatura temporale. In questo modo, il ritardo end-to-end può essere calcolato come differenza fra l'istante in cui il pacchetto è ricevuto e l'istante in cui il pacchetto è stato generato. Qualora questo valore fosse maggiore di 400ms, è consigliabile considerare il pacchetto come perso e scartarlo dal flusso di riproduzione per evitare di distorcere la comunicazione. Inoltre il timestamp, in unione ai numeri di sequenza servono a distinguere i periodi di parlato da quelli di silenzio nel caso di ritardo di playout adattivo.
IP security fornisce garanzie a livello di rete cifrando tutti i dati inviati in un datagramma IP. Esso è composto di due protocolli: Authentication Header (AH) e Encapsulation Security Payload (ESP). In entrambi i protocolli l'host sorgente e l'host destinazione, prima di procedere all'invio di datagrammi cifrati, si scambiano l'handshake (determinando una chiave segreta tramite il metodo Diffie-Hellman) e creano un canale logico a livello di rete chiamato Security Association (SA). Se entrambi gli host vogliono scambiarsi datagrammi, è prevista la creazione di un'altra SA. Quindi IPsec trasforma il tradizionale livello di rete senza connessione in uno con connessioni logiche. Ogni SA è unicamente determinato dal protocollo di sicurezza impiegato (AH o ESP), dall'indirizzo IP sorgente e dall'identificativo di connessione.

Si consideri un codice FEC che agisce su blocchi fatti di 3 pacchetti +1 di recupero. Progettare e disegnare uno schema di interleaving affinché l'uso congiunto di codice FEC ed interleaving sia in grado di tollerare la perdita di un intero blocco di interleaving

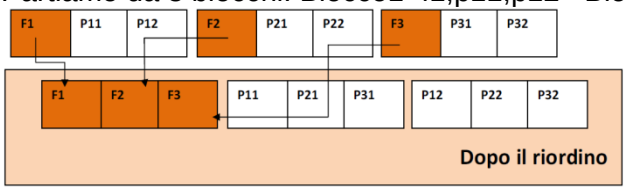
L'interlacciamento consiste nel suddividere in blocchi il flusso originale e in sequenza dei pacchetti, dopodiché "mischiare" opportunamente i pacchetti in modo tale che sia possibile supportare burst di perdite senza alcuna ridondanza e riuscendo a mascherare i pacchetti mancanti.
Ad esempio mettiamo i pacchetti di recupero tutti nello stesso blocco, così posso perdere un blocco intero a burst.



L'interlacciamento può migliorare significativamente la qualità con cui si percepisce un flusso audio e presenta anche bassa ridondanza. Il suo più grande vantaggio è che non richiede l'aumento di larghezza di banda dello stream. Lo svantaggio è che incrementa la latenza, limitando così il suo utilizzo in applicazioni interattive come la telefonia, sebbene possa dare buona prestazione nello streaming di audio memorizzato.

Si consideri un codice a correzione di errore che introduce un pacchetto di recupero ogni due pacchetti di dati. Si consideri inoltre una codifica interleaving che opera su blocchi formati ognuno da tre pacchetti, due di dati ed il corrispondente pacchetto di recupero. Si determini la massima lunghezza di un singolo burst di perdita di pacchetti a cui la codifica è tollerante, il minimo ritardo di riproduzione introdotto dalla codifica e lo spreco di banda introdotto.

La perdita dei dati può essere arginata mediante la correzione dell'errore in avanti (FEC, Forward Error Correction) e l'interlacciamento (Interleaving). L'interlacciamento consiste nel suddividere in blocchi il flusso originale e in sequenza dei pacchetti, dopodiché "mischiare" opportunamente i pacchetti in modo tale che sia possibile supportare burst di perdite senza alcuna ridondanza e riuscendo a mascherare i pacchetti mancanti.
Partiamo da 3 blocchi: Blocco1=f1,p11,p12 Blocco2=f2,p21,p22 Blocco3=f3,p31,p32



Questo schema può tollerare solo un errore nel primo blocco (che si corregge con F1), un errore nel
secondo blocco (che si corregge con F2) ed un errore nel terzo blocco (che si corregge con F3).
Applichiamo il seguente riordino (come in figura):
Blocco1=<F1, F2, F3>
Blocco2=<P11, P21, P31>
Blocco3=<P12, P22, P32>
Immaginiamo ora di perdere TUTTO il blocco 2. Adesso lo possiamo recuperare: con F1 recupero P11, con F2 recupero P21, con F3 recupero P31
Quindi in totale ora recupero 3 pacchetti, quindi possiamo tollerare la perdita di un burst di lunghezza 3. Lo spreco di banda è 1/3.
Il minimo ritardo riproduzione è determinato dal ritardo di trasmissione + ritardo per fare l'interleaving + ritardo del riordino + eventuale correzione.

In cosa consiste il paradosso del compleanno e per quale motivo è importante per la sicurezza delle funzioni hash? Spiegarlo attraverso un esempio.

Il paradosso del compleanno afferma che la probabilità che in un insieme di persone ce ne siano almeno due che festeggiano il compleanno nello stesso giorno è molto più alta di quella che potrebbe sembrare intuitivamente, tanto da sembrare paradossale. Supponendo n valori equiprobabili

$$p(k,n) = 1 - \frac{n!}{(n-k)!\, n^k} \approx 1 - e^{\frac{-k(k-1)}{2n}}$$

Se ora poniamo questa probabilità pari a 0.5, otteniamo che:

$$k = 1{,}1774\sqrt{n}$$

Se ipotizziamo che n sia uguale ai giorni dell'anno, otteniamo che in un gruppo di 23 persone scelto randomicamente e in maniera equiprobabile la probabilità che almeno due di esse siano nate lo stesso giorno è maggiore del 50%. Infatti:

$$k = 1{,}1774 \times \sqrt{365} = 22{,}49 \Rightarrow\ p(23,365) > 0.5$$

Il paradosso è importante nel calcolo dell'impronta di un documento tramite funzione hash perché, sapendo che se ho un documento e ho una funzione hash come ad esempio DES a 64 bit, è abbastanza difficile trovare un documento con la stessa impronta, tuttavia se considero un insieme grande di documenti la
probabilità che ve ne siano almeno due con la stessa impronta è troppo alta e rende la funzione inadeguata. Il paradosso è usato per quantificare la validità della funzione hash.

Su cosa si basa la sicurezza di un algoritmo a chiave pubblica? Descrivere brevemente gli aspetti principali di RSA.

L'algoritmo RSA è il più noto e diffuso algoritmo a chiave asimmetrica. La sicurezza si basa sul fatto che la decodifica si compie con la conoscenza di due chiavi, di cui una è di dominio pubblico, mentre l'altra la possiede solo il ricevente delle informazioni. Dunque i due individui si scambiano messaggi crittografati senza essersi mai scambiati la chiave di codifica (come avviene per la crittografia a chiave simmetrica). L'algoritmo RSA si basa sulla scelta di due numeri primi (p,q) molto grandi tali che il loro prodotto sia di 1024 bit. Il procedimento prevede poi di trovare altri due numeri (e,d) tali da soddisfare alcune proprietà algebriche elementari su p e q. Questi due numeri saranno rispettivamente una parte della chiave pubblica Kpub=(n,e) e di quella privata Kpri=(n,d). "e" viene scelto in modo tale che e<n, MCD(e,z)=1. z=(p-1)(q-1).
La sicurezza di RSA deriva dal fatto che non esistono (o meglio, non sono noti) algoritmi efficienti in grado di fattorizzare un numero nel prodotto di due numeri primi. A chi sono noti questi due numeri è facile effettuare codifica e decodifica (sebbene l'operazione sia onerosa
dal punto di vista computazionale perché prevede elevamenti a potenza), mentre per chi non li conosce scovarli è praticamente impossibile.

Assumete che un KDC server o un CA server si guasti. Chi può comunicare in modo sicuro e chi no nei due casi?

Nella crittografia a chiave pubblica c'è il problema di ottenere la chiave pubblica "vera" di qualcuno, problema che può essere risolto usando un intermediario di fiducia. Per la crittografia a chiave simmetrica, l'intermediario di fiducia è detto centro di distribuzione delle chiavi (Key Distribution Center) KDC, che è una singola entità di fiducia della rete con cui è stata stabilita una chiave segreta condivisa. Per la crittografia a chiave pubblica, l'intermediario di fiducia è detto autorità di certificazione (Certification Authorities) CA, che certifica che una chiave pubblica appartiene a una particolare entità. Una volta che la chiave pubblica è stata certificata, può essere distribuita da qualsiasi punto.

Se un KDC server si guasta continuano a comunicare in modo sicuro tutti gli utenti che usano chiavi asimmetriche+CA e tutti quelli che hanno chiavi simmetriche stabilite precedentemente al gusto (o in via privata). Però è chiaro che se un servizio web si basava sul KDC per far creare chiavi di sessione di volta in volta, ora non funziona più!

Se un CA server si guasta non c'è più modo di capire se una chiave pubblica è realmente associata ad un utente, è il caos! La CA mandava un certificato criptato con la chiave privata dell'utente, noi con la chiave pubblica (potenzialmente falsa causa man in the middle) che ci veniva fornita dal forse-utente potevamo decriptare e verificare se il certificato era effettivamente intestato a lui.. ma se la CA non va non possiamo comunicare in modo sicuro con i nuovi utenti (mi viene da pensare che se già sappiamo la chiave pubblica di un utente, non serve richiederla nuovamente alla CA, non so se esistono aggiornamenti di certificato).

Per quale motivo il protocollo rappresentato in figura 8.19 è da ritenersi insicuro? A che tipo di attacchi è soggetto? Come lo si può rendere sicuro? In cosa consiste l'attacco man in the middle, e come può essere evitato?

Per risolvere il problema dell'autenticazione si possono usare un nonce e la crittografia con chiave pubblica (invece di quella simmetrica): questo approccio, adoperato nell'ap5.0, evita il pericolo relativo a come le due parti si scambiano per la prima volta il valore della chiave segreta condivisa.

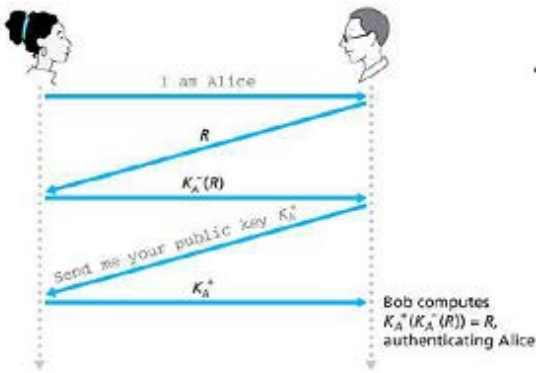


Figure 8.19 ♦ Protocol *ap5.0* working correctly

1. Alice invia il messaggio "Io sono Alice" a Bob.
2. Bob sceglie un nonce, R, e lo invia ad Alice. Ancora una volta, il nonce sarà usato per assicurarsi che Alice sia "viva".
3. Alice usa la sua chiave personale KA_pri sul nonce e invia il valore risultante KA_pri(R) a Bob. Poiché solo Alice conosce la sua chiave personale, nessuno tranne Alice può generare KA_pri(R).
4. Bob applica la chiave pubblica di Alice, KApub, al messaggio ricevuto. Quindi, Bob calcola R e autentica Alice.

Il protocollo ap5.0 è sicuro solo quanto la distribuzione delle chiavi pubbliche. L'intruso è in grado di interporsi in modo trasparente tra Alice e Bob, inoltrando i dati di Bob ad Alice, dopo averli decifrati e cifrati di nuovo, in modo tale che i due non si possano accorgere della sua presenza: questo è il caso dell'attacco detto con un uomo in mezzo (man-in-the-middle attack)..

Infatti Tommaso può fingere di essere Alice procedendo in questo modo:

1) Tommaso contatta Roberto
2) Roberto manda un nonce R ad Alice e Tommaso lo intercetta.
3) Tommaso manda KTpri(R) a Roberto
4) Roberto richiede la chiave pubblica di Alice Kpub, ma questa richiesta viene intercettata da Tommaso che gli manda la sua: KTpub
5) Roberto senza saperlo calcola KTpub(KTpri(R)) e verifica che KTpub(KTpri(R))=R

La falla del protocollo di autenticazione ap5.0 appena descritta può essere sfruttata da un malintenzionato per interporsi nella comunicazione in modo del tutto trasparente.

Per Alice e Roberto l'autenticazione è andata a buon fine, così credono che ciò che uno spedisce l'altro riceve ma in realtà non è così. Possiamo rendere sicuro il protocollo di autenticazione ap5.0 tramite la certificazione della chiave pubblica.

Per quale motivo per firmare un documento è necessario prima ottenerne l'impronta attraverso una funzione di hash? Quali caratteristiche deve avere una funzione di hash?

L'impronta digitale (o hash) è una sequenza di n bit ottenuta prendendo in ingresso il contenuto informativo di un intero messaggio. Le firme hash possono essere utilizzate per la creazione di firme digitali in quanto permettono la rapida creazione di firme digitali anche per file di grossa dimensioni. E' infatti più conveniente eseguire con rapidità un hashing del testo da firmare e poi autenticare solo quello piuttosto che eseguire algoritmi complessi di crittografia asimmetrica su moli di dati molto grandi.
Affinché essa sia effettivamente rappresentativa del messaggio originale, l'impronta deve soddisfare le seguenti proprietà:
1. Unidirezionalità. Deve essere facile (al massimo costo polinomiale) ottenere l'hash a partire dal messaggio originale e difficile (almeno costo esponenziale) ottenere il messaggio originale a partire dall'hash.
2. Resistenza forte alle collisioni. Dati due documenti D1 e D2, deve essere computazionalmente impossibile (ovvero complesso) ottenere due hash identici. Questo però non vieta l'esistenza, in linea teorica, di possibili collisioni.
3. Effetto a valanga. Se un documento D1 con impronta H1 viene modificato anche lievemente, allora la sua impronta deve cambiare notevolmente.
4. Equiprobabilità. Dati tutti i possibili valori dell'impronta a n bit, ciascuno di essi ha la stessa probabilità di essere restituito come hash di un qualsiasi messaggio. Quindi nella pratica documenti diversi hanno impronte diverse, dunque una qualsiasi modifica al documento genera una nuova impronta.

Un compratore si rivolge ad un sito di commercio elettronico. Quali strumenti garantiscono il compratore dell'identicità del sito di commercio elettronico e viceversa? Quale protocollo permette di mantenere il numero di carta di credito del compratore ignoto al sito di commercio elettronico?

Si potrebbe utilizzare la certificazione delle chiavi e i nonce, il compratore manda un nonce r1 con la chiave pubblica certificata del sito di commercio elettronico. Il sito cifra r1 e r2 con la chiave pubblica certificata del compratore. Quando il compratore riceve K(r1,r2) risalendo a r1 autentica il sito. Il sito, con le stesse modalità risale a r2 e autentica il compratore. Il protocollo per mantenere ignoto il numero della carta di credito al sito di commercio elettronico è SET, si basa su SSL (Secure Socket Layer).

Dovete realizzare un sistema per l'autenticazione in una rete aziendale non connessa ad Internet a cui accedono soli 5 utenti. Che soluzione proporreste? Motivare la risposta.

Poiché ci sono solo 5 utenti, io lascerei stare l'autenticazione intesa in senso classico ed utilizzerei un sistema con chiavi private. Infatti il solo fatto che due utenti conoscano la chiave privata da loro condivisa, ci assicura l'autenticazione. L'unico problema delle chiavi private è lo scambio delle chiavi e il numero delle chiavi: problema aggirabile perché, appunto, ci sono solo 5 utenti!

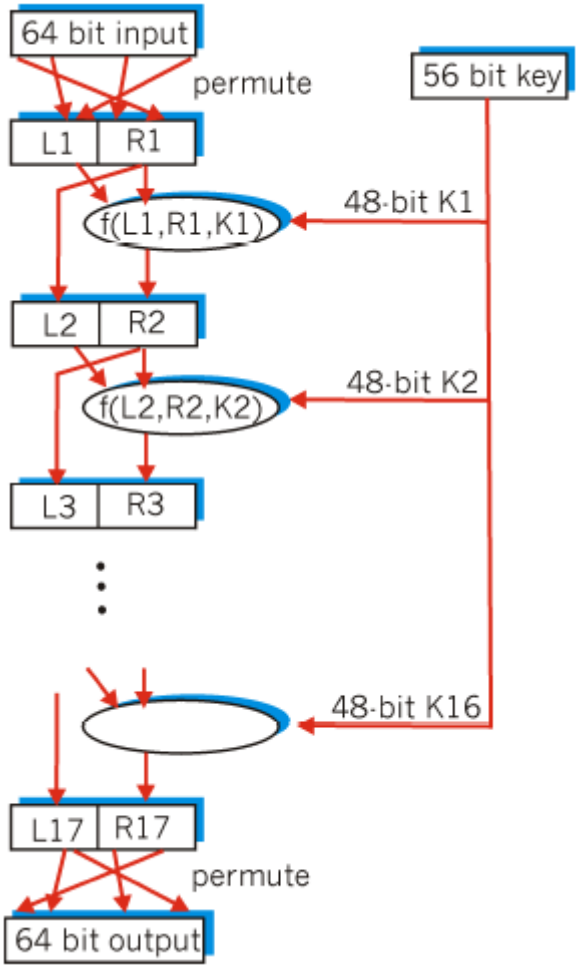
Descrivere graficamente un metodo per il calcolo dell'impronta di un documento che utilizza DES (e calcola impronte di 64 bit). Infine spiegare in dettaglio perché impronte di 64 bit non sono considerate sicure indipendentemente dal metodo utilizzato per il calcolo).

DES (Data Encryption Standard) è un importante algoritmo a chiave simmetrica. Grazie alla sua semplicità, esso è adatto a cifrare grandi quantità di dati e può essere implementato facilmente ed efficientemente anche a livello hardware.

2

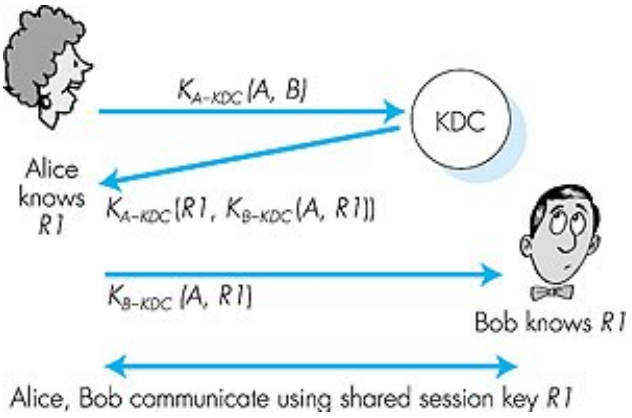
Struttura DES

- Permutazione iniziale
- 16 "rounds" identici in ciascuno si applica una funzione basata su 48 bit della chiave (ogni volta diversi)
- Permutazione finale



La modalità usata è quella del cifrario a blocchi e si chiama Electronic Codebook(ECB). Essa soffre di una sottile ma importante falla, dovuta al fatto che blocchi generati a partire da una stessa stringa in chiaro danno lo stesso risultato cifrato. Questo potrebbe consentire ai malintenzionati di ricavare la chiave qualora essi fossero a conoscenza della struttura del messaggio originale.
Per rendere DES più sicuro, si può effettuare una modifica nota come cifrario a blocchi incatenati (CBC, Cipher Block Chaining). Nella modalità CBC, ciascun blocco in chiaro di 64 bit viene prima messo in XOR col risultato cifrato del blocco precedente, in modo tale da creare una dipendenza fra i singoli blocchi. Quello iniziale è messo in XOR con una sequenza chiamata inizialization vector (IV) generata randomicamente.
Ormai una chiave di 56 bit non è più sufficiente a rendere l'algoritmo sicuro, motivo per cui le versioni più recenti di DES utilizzano chiavi a 112 bit.

Descrivere l’architettura di un KDC (Key distribution center). A cosa serve?



Il KDC è il centro di distribuzione delle chiavi e prevede che le entità si registrino in un sistema speciale in grado di fornire chiavi segrete per una conversazione cifrata simmetrica. La registrazione prevede che ogni entità condivida con KDC una chiave di sessione per far sì che la comunicazione col centro di distribuzione avvenga in maniera protetta (non è specificato come questa chiave venga generata).
Quando A vuole comunicare con B, invia a KDC una richiesta cifrata con la chiave di sessione. Il server, quindi, genera una chiave segreta S e risponde ad A con un messaggio cifrato che contiene:
1. La chiave segreta S da utilizzare per comunicare con B.
2. Un messaggio cifrato secondo la chiave di sessione fra B e KDC, destinato a B e contentente a sua volta:
   a. La chiave segreta S.
   b. L’identificativo dell’utente con cui B dovrà comunicare (in questo caso, A).

A invia questa seconda parte del messaggio principale, così com’è, a B che, una volta ricevuta, è in grado di estrarre le informazioni necessarie per instaurare la comunicazione con A decifrando il messaggio secondo la chiave di sessione che condivide con KDC. Ora A e B possono comunicare usando la chiave segreta S generata dal centro di distribuzione chiavi.
Serve quindi a far si che due terminali possano condividere in sicurezza delle chiavi segrete

Cos’è il pagerank intuitivamente e come può essere calcolato?
Cos'è il pagerank e quali vantaggi offre?

Attraverso il pagerank è possibile ordinare le pagine web in base alla loro importanza. La popolarità di una pagina è funzione della frequenza con cui essa si ipotizza venga acceduta da un utente che visita il Web in modalità random walk + teleporting. Il Random walk: Partendo da una pagina a caso, seleziona a ciascun passo un link di uscita in maniera equiprobabile.
Il Teleporting: Se una pagina non contiene link di uscita, salta a una pagina a caso nel grafo (cioè nel Web). Combinando assieme questi due procedimenti è possibile determinare la frequenza di visita di ciascuna pagina fino a raggiungere una condizione di equilibrio (steadystate) in cui la probabilità che un utente si trovi in una determinata pagina in un certo istante diventa costante (proporzionale alla frequenza di visita a lungo termine). La frequenza a lungo termine può essere calcolata attraverso un'astrazione del random walk, le catene di Markov. Una catena di Markov è una serie di n stati ed caratterizzato da matrice di probabilità di transizione; inoltre una catena di Markov è ergodica se scelti 2 stati è sempre possibile andare dall'uno all'altro. Il vantaggio del pagerank è che poiché il random walk con teleporting può essere rappresentato come una catena di Markov ergodica, PER CUI ESISTE ED è UNICA LA DISTRIBUZIONE DELLE FREQUENZE, possiamo sempre trovare questa distribuzione che è proprio uguale al pagerank.
il pagerank è "query indipendent" Perché dipende solo dalla struttura dei link: infatti è possibile calcolare il vettore di probabilità e quindi assegnare il pagerank (preprocessing) prima dell'effettiva richiesta da parte di un client (query processing).

Quali sono i motivi fondamentali per cui un motore di ricerca che utilizza informazioni topologiche ha una maggiore efficacia di un motore basato sulle solo informazioni testuali?

L'intuizione adottata da Google, Yahoo, MSN, etc. è stata quella di vedere il Web come un grafo in cui un collegamento (link) da una pagina A ad una pagina B definisce la qualità percepita dall'autore di A nei confronti di B (quality signal) e l'etichetta associata al collegamento (anchor text) descrive la pagina puntata (textual context). In questo modo è possibile indicizzare un documento sulla base di informazioni testuali e sulla base di informazioni topologiche, considerando rilevante non solo la frequenza di una parola nella pagina, ma anche le etichette adottate per descrivere la stessa dall'esterno, tenendo conto della popolarità (pagerank o (un)directed popularity).

Perché il pagerank è resistente allo spamming, inteso come la capacità di una entità di alzare indebitamente la rilevanza di una pagina, mentre il solo in-degree (numero di pagine che puntano alla pagina) non lo è?

Inizialmente si è pensato di attribuire a ciascuna pagina un peso che fosse funzione del numero di link entranti (in-degree). A questo punto, l'ordine con cui le pagine comparivano nei risultati di ricerca non era più strettamente dipendente dalla query ma era funzione della popolarità di ogni pagina. In altre parole: data una classifica di pagine web ordinata per popolarità (queryindependent) e mantenuta in un database, essa veniva riproprosta all'utente scartando quei risultati non inerenti alla parola cercata. Questo algoritmo, seppur interessante, era ancora sensibile allo spamming: un gruppo di autori poteva decidere di linkare fra loro le proprie pagine in modo tale da accrescere la popolarità di ciascuna di esse.
Successivamente con il pagerank le pagine sono diventate resistenti allo spamming poiché non conta più solamente il numero di pagine che hanno linkato quella data pagina, ma anche la frequenza con cui queste pagine sono state visitate. Una pagina ha un pagerank alto se è stata linkata da molte pagine aventi a loro volta pagerank alto;
quindi non è sufficiente chiedere ad un gruppo di amici, per quanto numeroso possa essere, di linkare la mia pagina nella loro.

Discutere quali sono le caratteristiche di una rete sociale e le loro implicazioni?

Una rete sociale consiste in un qualsiasi gruppo di entità connesse tra loro da diversi legami. Tutte le reti sociali hanno delle caratteristiche
comuni: sono grandi, complesse e hanno regolarità a livello macroscopico:

• Hanno una componente gigante connessa: la maggior parte dei nodi appartiene ad un'unica componente connessa, per questo motivo per raggiungere un nodo desiderato posso partire da uno qualsiasi dei nodi facenti parte la rete sociale.

• Hanno al loro interno un piccolo mondo cioè la distanza media tra i nodi è bassa, esiste un percorso di lunghezza limitata che connette una qualsiasi coppia di entità nel grafo.

• Il numero totale di archi è minore o uguale al numero totale di nodi, ovvero il grafo è debolmente connesso (non si può raggiungere un nodo qualsiasi a partire da un altro nodo con un unico passo): |E|<<|V|^2

• Clustering. Sono presenti delle piccole aggregazioni o grumi isolati interconnessi fra loro.

• È soggetto ad una legge di invarianza: Non vi è una scala privilegiata per osservare le proprietà macroscopiche della rete, ma esse si ripropongono ad ogni livello. Nelle reti sociali l'invarianza di scala si riferisce alla distribuzione del grado dei nodi (numero degli archi incidenti), che segue un andamento polinomiale (legge di potenza). In altri termini, la percentuale di nodi con grado pari a k è:

$$P(k) \sim k^{-\alpha}$$

Cosa garantisce il modello di reti sociali di Kleinberg rispetto a quello di Watts e Strogatz?
Quale caratteristica presenta il modello di Kleinberg per le reti sociali che non si riscontra nel modello di Watts e Strogatz?

Dal momento che il modello dei grafi casuali non era sufficientemente preciso, Watts e Strogatz proposero un altro modello. L'idea era quella di rispettare in un sol colpo le proprietà di clustering (grazie ai k collegamenti coi vicini), piccolo mondo (grazie all'abbassamento della distanza media dato dalla probabilità p) e componente connessa (grazie alle riconnessioni). Tuttavia, il modello non si presta ad essere navigato molto facilmente, rendendo difficile verificare su di esso le ipotesi di piccolo mondo. Kleinberg sosteneva che il tempo atteso di consegna di un messaggio nei modelli non navigabili è almeno polinomiale e ha dimostrato che qualunque algoritmo greedy su di esso non termina prima di un tempo pari a:

$$\Omega(n^{2/3})$$

Nel modello di Kleimberg un nodo u è direttamente collegato a v con una probabilità proporzionale a:

$$p \sim d^{-r}(u, v)$$

A differenza del modello di Watts-Strogatz, ha la caratteristica che la lunghezza attesa dei percorsi sorgente-destinazione, utilizzando il greedy routing, un algoritmo in cui ogni qual volta si deve scegliere un nuovo nodo di un percorso sorgente-destinazione si sceglie quello più vicino alla destinazione. Il comportamento del modello è determinato dal parametro r:
Se r è pari a 0 si ottiene una distribuzione uniforme di collegamenti fra due coppie qualsiasi.
Se r è pari a 1, si ottiene una distribuzione che è funzione dell'inverso della distanza fra i due nodi.
Se r = 2, la distribuzione è funzione dell'inverso del quadrato e il tempo atteso di consegna è

$$O(log^2 n)$$

Mentre se r ≠ 2, allora la lunghezza attesa è di $\Omega(n^{\alpha}), \alpha > 0$
Quindi paragonato con il modello Wattz-Strogatz dove la lunghezza attesa dei percorsi sorgente-destinazione, utilizzando qualunque algoritmo di navigazione, è

$$\Omega(n^{2/3})$$

si deduce che il modello di Kleimberg è migliore.

Per quale motivo i grafi random Erdos e Renyi non rappresentano in modo soddisfacente una rete sociale?

Il modello dei grafi casuali proposto da Erdos e Renyi è un modello di tipo probabilistico. Per ogni coppia di nodi, l'arco (u, v) che li connette esiste con probabilità p indipendente da tutti gli altri. Come risultato, per p sufficientemente grande la maggior parte dei nodi si trova in una componente gigante connessa, il cui diametro (massima distanza fra due nodi, intesa come numero minimo di archi fra essi) è logaritmico in funzione del numero di nodi e così anche la lunghezza media dei cammini. Il modello si rende quindi adatto a rappresentare piccoli mondi. Tuttavia, la natura probabilistica indipendente con cui vengono creati i collegamenti rende il modello quasi del tutto privo di agglomerati. Più precisamente, il coefficiente di clustering decresce come un'iperbole del numero di nodi:

$$O(1/n)$$

e quindi, essendo "basso", non rappresenta in modo soddisfacente una rete sociale.

In cosa consiste il greedy routing in reti sociali. In quali modelli funziona bene?
Cos'è il greedy routing nelle reti sociali e cosa dicono i risultati di Kleinberg in proposito?
Descrivere il modello di grafo su cui si ottengono tali risultati

Il greedy routing è un algoritmo utilizzato nelle reti sociali, in cui ogni qual volta si deve scegliere un nuovo nodo di un percorso sorgente-destinazione si sceglie quello più vicino alla destinazione. I risultati di Kleinberg riguardano la lunghezza attesa
Il comportamento del modello è determinato dal parametro r:
Se r è pari a 0 si ottiene una distribuzione uniforme di collegamenti fra due coppie qualsiasi.
Se r è pari a 1, si ottiene una distribuzione che è funzione dell'inverso della distanza fra i due nodi.
Se r = 2, la distribuzione è funzione dell'inverso del quadrato e il tempo atteso di consegna è

$O(log^2 n)$.

Mentre se r ≠ 2, allora la lunghezza attesa è di $\Omega(n^{\alpha}), \alpha > 0$.
Migliore del modello Wattz-Strogatz, dove la lunghezza attesa dei percorsi sorgente-destinazione, utilizzando qualunque algoritmo di navigazione, è : $\Omega(n^{2/3})$

Il modello a grafo è basato su una griglia bidimensionale n*n in cui ciascun nodo ha 4 vicini (eccetto angoli e bordi). Inoltre, con probabilità p vengono aggiunti per ciascun nodo dei collegamenti diretti casuali. Più precisamente, un nodo u è direttamente collegato a v con una probabilità proporzionale a:

$p \sim d^{-r}(u, v)$

Si consideri una pagina HTML fatta di due oggetti O1 e O2, dove O1 può essere spedito in un singolo pacchetto TCP/IP, mentre O2 ne richiede due. Sotto l'ipotesi che la rete non sia congestionata, quanto tempo impiegherei a scaricare la pagina nel caso di una connessione HTTP a) permanente, b) non permanente. Motivare la risposta.

•

Nel caso permanente il server mantiene la connessione attiva in seguito all'esaurimento della richiesta per un tempo pari al timeout, oltre il quale la chiude. Il vantaggio delle connessioni persistenti è che si risparmiano RTT, anche se la connessione rimane generalmente attiva per più tempo del necessario, mandando potenzialmente il server in sovraccarico se il timeout non è regolato correttamente. Esistono due modalità di richiesta degli oggetti, una incanalata (con pipelining) e una non-incanalata.

con pipelining usiamo 1 rtt per instaurare la connessione
dopodiché inizio calcolo istanti di tempo:
nel primo istante, oggetto da 1 pacchetto,
nel secondo istante, secondo oggetto da 2 pacchetti
totale 3

se è non incanalata c'è un altro rtt per l'attesa della risposta dopo il primo oggetto
totale 4

Nel caso Non permanente il server analizza la richiesta, risponde e chiude la connessione TCP, per cui sono necessari circa 2 RTT per ricevere ciascun oggetto: uno per 2/3 dell'handshake a tre vie TCP, uno per la parte rimanente di handshake (inclusa la richiesta dell'oggetto) e la risposta del server. Lo svantaggio di questo meccanismo è che ogni oggetto subisce lo slow start di TCP, rendendo i download più lenti di quanto si vorrebbe. Inoltre, vanno allocate risorse (buffer TCP) per ciascun oggetto richiesto e questo può creare problemi qualora il numero di connessioni contemporanee sia molto elevato.
quindi avrò: 1 rtt per apertura connessione
+1 rtt per richiesta invio +1 istante di trasmissione per il primo oggetto da un pacchetto.
cade connessione e quindi avrò bisogno di 1 nuovo rtt per invio nuova richiesta
+1 Primo istante primo pacchetto del secondo oggetto +1 secondo istante, secondo pacchetto secondo oggetto
Tot=6

Cos'è il KDC (Key Distribution Center) e come si differenzia da una Certification Authority

Il KDC é un centro di distribuzione di chiavi. Il kdc prevede che le entità si registrino in un sistema speciale in grado di fornire chiavi segrete per una conversazione cifrata simmetrica e NON usa chiave pubblica dell'utente. Quando A vuole comunicare con B, invia a KDC una richiesta cifrata con la chiave di sessione.
Il server, quindi, genera una chiave segreta S e risponde ad A con un messaggio cifrato che contiene:
1. La chiave segreta S da utilizzare per comunicare con B.
2. Un messaggio cifrato secondo la chiave di sessione fra B e KDC, destinato a B e contentente a sua volta:
   a. La chiave segreta S.
   b. L'identificativo dell'utente con cui B dovrà comunicare (in questo caso, A).

A invia questa seconda parte del messaggio principale, così com'è, a B che, una volta ricevuta, è in grado di estrarre le informazioni necessarie per instaurare la comunicazione con A decifrando il messaggio secondo la chiave di sessione che condivide con KD.
La CA garantisce la corrispondenza fra utente e chiave pubblica e risolve un grave problema di sicurezza introdotto dall'impiego di crittografia asimmetrica. Quando A vuole conoscere la chiave pubblica di B, piuttosto che chiederla a B stesso si rivolge alla CA per ottenere il certificato di B. Il certificato viene poi sottoposto a verifica della firma digitale sulla base della chiave pubblica della CA stessa, per assicurarsi che non sia stato
manipolato.

A cosa servono i pacchetti RTS/CTS e per quale motivo sono di piccole dimensioni?

I pacchetti RTS (Request-To-Send) e CTS (Clear-to-Send) sono di piccole dimensioni in modo da avere una minor probabilità di collisione rispetto ai pacchetti standard.Vengono usati per risolvere il problema del terminale nascosto. Quando il trasmittente vuole inviare un frame dati, aspetta DIFS (Distributed Inter-Frame Space) e, se il backoff lo consente, invia un frame RTS all'Access Point indicando il tempo totale richiesto per la trasmissione (compreso il tempo di ricezione dell'ACK). Quando l'Access Point riceve un frame RTS, aspetta un tempo pari a SIFS (Short Inter-frame Space) e risponde in broadcast con un frame CTS contenente a sua volta la durata complessiva della prenotazione. In questo modo, pur non essendo tutti i terminali consapevoli dell'esistenza degli altri, l'arrivo di un frame CTS permette loro di capire che è in atto una trasmissione di cui essi non riescono ad essere in ascolto. si verificano ancora collisioni nel momento in cui due nodi tentano di prenotare il canale contemporaneamente o ad intervalli molto ravvicinati l'uno dall'altro. La probabilità di collisione e lo spreco di banda sono tuttavia minimi poiché la collisione riguarda pacchetti (RTS e/o CTS) di lunghezza ridotta.

Quali sono i vantaggi nell'usare RTS/CTS nelle reti wireless?

I pacchetti RTS (Request-To-Send) e CTS (Clear-to-Send) sono di piccole dimensioni in modo da avere una minor probabilità di collisione rispetto ai pacchetti standard.Vengono usati per risolvere il problema del terminale nascosto. Quando il trasmittente vuole inviare un frame dati, aspetta DIFS (Distributed Inter-Frame Space) e, se il backoff lo consente, invia un frame RTS all'Access Point indicando il tempo totale richiesto per la trasmissione (compreso il tempo di ricezione dell'ACK). Quando l'Access Point riceve un frame RTS, aspetta un tempo pari a SIFS (Short Inter-frame Space) e risponde in broadcast con un frame CTS contenente a sua volta la durata complessiva della prenotazione. In questo modo, pur non essendo tutti i terminali consapevoli dell'esistenza degli altri, l'arrivo di un frame CTS permette loro di capire che è in atto una trasmissione di cui essi non riescono ad essere in ascolto. si verificano ancora collisioni nel momento in cui due nodi tentano di prenotare il canale contemporaneamente o ad intervalli molto ravvicinati l'uno dall'altro. La probabilità di collisione e lo spreco di banda sono tuttavia minimi poiché la collisione riguarda pacchetti (RTS e/o CTS) di lunghezza ridotta.

Quali sono le caratteristiche che deve avere una funzione di hash? Per quale motivo viene usata nell'ambito della firma digitale?

L'impronta digitale (o hash) è una sequenza di n bit ottenuta prendendo in ingresso il contenuto informativo di un intero messaggio.

$$Hash = f(Messaggio, n)$$

Affinché essa sia effettivamente rappresentativa del messaggio originale, l'impronta deve soddisfare le seguenti proprietà:
1. Unidirezionalità. Deve essere facile (al massimo costo polinomiale) ottenere l'hash a partire dal messaggio originale e difficile (almeno costo esponenziale) ottenere il messaggio originale a partire dall'hash.
2. Resistenza forte alle collisioni. Dati due documenti D1 e D2, deve essere computazionalmente impossibile (ovvero complesso) ottenere due hash identici. Questo non vieta l'esistenza, in linea teorica, di possibili collisioni.
3. Effetto a valanga. Se un documento D1 con impronta H1 viene modificato anche lievemente, allora la sua impronta deve cambiare notevolmente.
4. Equiprobabilità. Dati tutti i possibili valori dell'impronta a n bit, ciascuno di essi ha la stessa probabilità di essere restituito come hash di un qualsiasi messaggio.
Viene utilizzata nell'ambito della firma digitale perché se applicata a un documento fornisce un risultato che dovrebbe essere univoco. Su questo risultato si applica una firma che "personalizza" l'informazione da parte del creatore/mittente. Firmare un documento intero sarebbe più oneroso.

Dovete progettare un sistema sicuro per lo scambio di documenti di grandi dimensioni. Il sistema deve gestire un numero molto elevato di utenti, deve essere in grado di verificare l'identità di chi effettua il download e di garantire che il documento scaricato sia incomprensibile a terze parti. Che soluzione adottereste?

Userei un CA (Certification Authority). L'autorità di certificazione è un ente in grado di garantire la corrispondenza fra utente e chiave pubblica e risolve un grave problema di sicurezza introdotto dall'impiego di crittografia asimmetrica. Un utente (persona, router, etc...) registra la sua chiave pubblica con la CA fornendo una garanzia di identità (garantisce cioè di essere chi dice di essere). CA crea in tutta risposta un certificato che collegherà da questo momento in poi quella specifica chiave pubblica a quel determinato utente. Il certificato è firmato dalla CA secondo i meccanismi della firma digitale con una chiave privata propria dell'autorità. Quando A vuole conoscere la chiave pubblica di B, piuttosto che chiederla a B stesso si rivolge alla CA per ottenere il certificato di B. Il certificato viene poi sottoposto a verifica della firma digitale sulla base della chiave pubblica della CA stessa, per assicurarsi che non sia stato manipolato. Ciò denota una forma di fiducia implicita verso l'autorità stessa, poiché si assume che essa sia, come prima cosa, chi dice di essere. Il certificato è costituito almeno dal nome dell'autorità, la data di emissione, la data di scadenza, il nominativo del soggetto e la sua chiave pubblica. Esso può essere sospeso o revocato in ogni momento qualora avvenga un furto o uno smarrimento.

Per quale motivo il pagerank è “query independent”?

Il pagerank è "query indipendent" perché dipende solo dalla struttura dei link: infatti è possibile calcolare il vettore di probabilità e quindi assegnare il pagerank (preprocessing) prima dell'effettiva richiesta da parte di un client (query processing).

Attraverso il pagerank è possibile ordinare le pagine web in base alla loro importanza. La popolarità di una pagina è funzione della frequenza con cui essa si ipotizza venga acceduta da un utente che visita il Web in modalità random walk + teleporting. Combinando assieme questi due procedimenti è possibile determinare la frequenza di visita di ciascuna pagina fino a raggiungere una condizione di equilibrio (steadystate) in cui la probabilità che un utente si trovi in una determinata pagina in un certo istante diventa costante (proporzionale alla frequenza di visita a lungo termine).
il Random Walk con teleporting può essere modellizzato tramite una catena di markov ergodica (presi due stati c'è sempre un cammino che li congiunge) e sotto queste condizioni esiste ed è unica la distrubuzione delle frequenze che è proprio il pagerank.

Per quali motivi il protocollo UDP è di norma più indicato per le applicazioni real-time?

Il protocollo UDP viene spesso preferito al TCP per le applicazioni real-time in streaming e interattive. Il motivo di questa scelta sta nel fatto che per erogare servizi basati su queste due tipologie di applicazioni è fondamentale rispettare tempi di consegna molto brevi. UDP è efficace perché, pur non garantendo la consegna in ordine e senza perdita dei pacchetti, trasporta i datagrammi molto velocemente (sfruttando meglio la banda e riducendo il ritardo end-to-end), invia finchè non riceve una conferma di ricezione senza preoccuparsi del tempo richiesto per il trasporto affidabile). Inoltre l’udp è “senza connessione” quindi ci mette meno tempo rispetto a un tcp che richiede un handshaking iniziale

Come realizzereste in JAVA il meccanismo che consente ad un foreign agent di annunciare la propria presenza ad eventuali mobile node che si trovano nelle prossimità? Descrivere per grandi linee l'implementazione limitandosi ai soli aspetti di networking

```
// Codice del foreign agent
class UDPAgent
{
  public static void main(String args[]) throws Exception
  {
    while(true){
    DatagramSocket agentSocket = new DatagramSocket();
    InetAddress IPAddress = InetAddress.getByName("non so come ottenere indirizzo broadcast!");
    byte[] sendData = new byte[1024];
    byte[] receiveData = new byte[1024];
       String sentence="beacon";
    sendData = sentence.getBytes();
    DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, IPAddress, 9876);
    agentSocket.send(sendPacket);
    DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
    agentSocket.receive(receivePacket);
    //ora teoricamente dovrei estrarre indirizzo home agent e inviare il coa all'home agent
    clientSocket.close();
  }}
}

// codice del client
class UDPMobileNode
{
  public static void main(String args[]) throws Exception
    {
      DatagramSocket mobileNodeSocket = new DatagramSocket(9876);
        byte[] receiveData = new byte[1024];
        byte[] sendData = new byte[1024];

            DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
            mobileNodeSocket.receive(receivePacket);
            String data= new String( receivePacket.getData());
            if (data.equals("beacon")){
                        InetAddress IPAddress = receivePacket.getAddress();
                        int port = receivePacket.getPort();
                        String answer ="indirizzo del mio home agent";
                        sendData = answer.getBytes();
                        DatagramPacket sendPacket =
                        new DatagramPacket(sendData, sendData.length, IPAddress, port);
                        mobileNodeSocket.send(sendPacket);

                //il client termina e fa quello che gli pare.
    }
}
```

Un servizio real-time è caratterizzato da un R_spec con banda pari a 100Kbps e tempo di slack pari a 0.3s. Si considerino pacchetti di dimensione 1Kb e tre possibili percorsi alternativi costituiti da due router identici, che assegnano al servizio rispettivamente:
1. Banda in uscita 70Kbps, buffer 5 pacchetti
2. Banda in uscita 200Kbps, buffer 10 pacchetti
3. Banda in uscita 200Kbps, buffer 100 pacchetti
Quale dei percorsi garantisce il rispetto del R_spec? Giustificare la risposta.

Il Tempo di slack (S) permesso indica la differenza tra il ritardo end-to-end desiderato dalla sorgente e il ritardo effettivamente ottenuto tramite l'assegnazione di una banda R. Ogni elemento di rete crea un buffer in uscita ai token bucket (b, r) e (1, p) destinati al flusso e lo dimensiona sulla base di un valore tale per cui non si hanno perdite e i ritardi cadono nell'intervallo di tolleranza. In questo modo, se il router alloca una buffer di dimensione B al flusso e processa i pacchetti ad un rate R', allora si avrà che:

$$S_{OUT} = S - B \ / \ R'$$

Dato che il buffer potrebbe essere condiviso con altri flussi il massimo ritardo è dato dall'avere tutta la coda B piena, svuotata a ritmo R'.

Il flusso è quindi accettato soltanto se valgono le seguenti condizioni:

$$R' \geq r$$
$$B \geq b$$
$$S_{OUT} > 0$$

In caso contrario, le risorse non sono sufficienti e la prenotazione viene negata.

Nel nostro caso
il primo non va bene perché perché se la R_spec comunica una velocità di invio di pacchetti a 100Kbps, il router deve essere in grado di smaltire questi pacchetti. Se il rate di uscita del router è inferiore a quello specificato (come in questo caso 70<100) nella R_spec, il buffer si riempirebbe subito, e non riuscirebbe a smaltire i pacchetti in ingresso. Nell'ipotesi 3 abbiamo un ritardo rit3=2*100kb/(200kbps)=1 sec
Considerando che il tempo di slack è pari a 0,3 non va bene perché 1 s > 0.3 s .
Nell'ipotesi 2 abbiamo un ritardo rit2=2*10kb/(200kbps)=0,1sec
quindi la risposta è la 2 essendo rit2<slack

Descrivere per grandi linee l'implementazione in JAVA di un sever HTTP. Ci si concentri sui soli aspetti di networking indicando tra commenti gli altri aspetti che riguardano l'implementazione es: /* leggo il file richiesto */

Da un lato il client supponiamo richieda connessione sulla porta 1240

Il server

```
ServerSocket connetion=new socket(1240);
while (true)
socket client=connection.accept();
bufferedreader in=new buferedreader(new inputstreamreader(client.getinputstream());
printwirter out=new printwriter(clien.getoutputstream());
in.readline();
out.close();
in.close();
sock.close();
```

Si consideri la figura 6.21 e la si completi illustrando i principali flussi di informazione relativi all'instradamento diretto e indiretto di un utente mobile

Nel caso dell'instradamento indiretto, il corrispondente non fa altro che instradare il datagramma All'indirizzo permanente del nodo, inconsapevole dell'effettiva localizzazione dello stesso. Questo approccio è quello attualmente usato nello standard IP Mobile.

Nel caso dell'instradamento diretto, il corrispondente ha un proprio agente personale che si occupa di interrogare l'agente domestico per scoprire qual è l'attuale indirizzo fermoposta del nodo mobile. Una volta nota questa informazione, l'agente del corrispondente comunica via tunneling col nodo mobile usando il COA come destinazione del pacchetto IP esterno. Allo stesso modo, il nodo mobile risponde al corrispondente usando il COA come sorgente.