Lezione n.7
IL PROTOCOLLO EMULE: LA RETE e2dk
LAURA RICCI
17/3/2008

eMule: UN PO' DI STORIA....

- Origine: il protocollo eDonkey, il client edonkey2000 (ed2k)
- E-donkey: rete peer-to-peer per la condivisione di file di qualsiasi tipo
- I client eDonkey si connettono alla rete per condividere file.
- I server eDonkey hanno una funzione simile al server NAPSTER: permettono agli utenti di localizzare i file all'interno della rete.
- Aggiunta dinamica di servers alla rete.
- A causa del costante cambiamento della rete dei server, i client devono aggiornarne costantemente la lista.
- Ultime versioni: utilizzanola DHT Kademlia

eMule: UN PO' DI STORIA.....

- eMule: il nome del progetto sottolinea sia le somiglianze che le diversità con il programma eDonkey2000
- eMule (mulo) simile ad un donkey (asino). Emule "emula" le funzionalità di eDonkey.
- sviluppato nel maggio 2002 dal programmatore tedesco Merkur, come un programma eDonkey-compatibile ma con molte più funzioni.
- sfrutta due diversi tipi di overlay networks
  - La rete eDonkey (ed2k), basata sul modello client/server.
  - La rete KAD (presente a partire dalla versione 0.42)
    - implementa KADEMLIA, una DHT molto efficiente
    - è una rete serverless in cui il carico di lavoro per la gestione dell'indice è distribuito a tutti i client connessi invece che ad un unico server
- Utilizzato ogni giorno in media da 3 milioni di utenti che condividono circa 500 milioni difiles

eMule: LA RETE ED2K

- Basato su un sistema P2P orientato al file sharing
  - basata su una estensione del protocollo eDonkey
  - architettura Client/Server
  - utilizza sia il protocollo TCP che il protocollo UDP
- Client open source: miglioramenti continue delle versioni del client eMule
- Il protocollo definisce le interazioni tra
  - Client e servers
  - Client e clients
- Strategie originali implementate nel client
  - Gestione delle code
  - Sistema dei crediti
  - Ordinamento delle parti di file scaricate
  - Gestione (recovery) di parti di file corrotte

eMule: IL CLIENT

Ogni client eMule

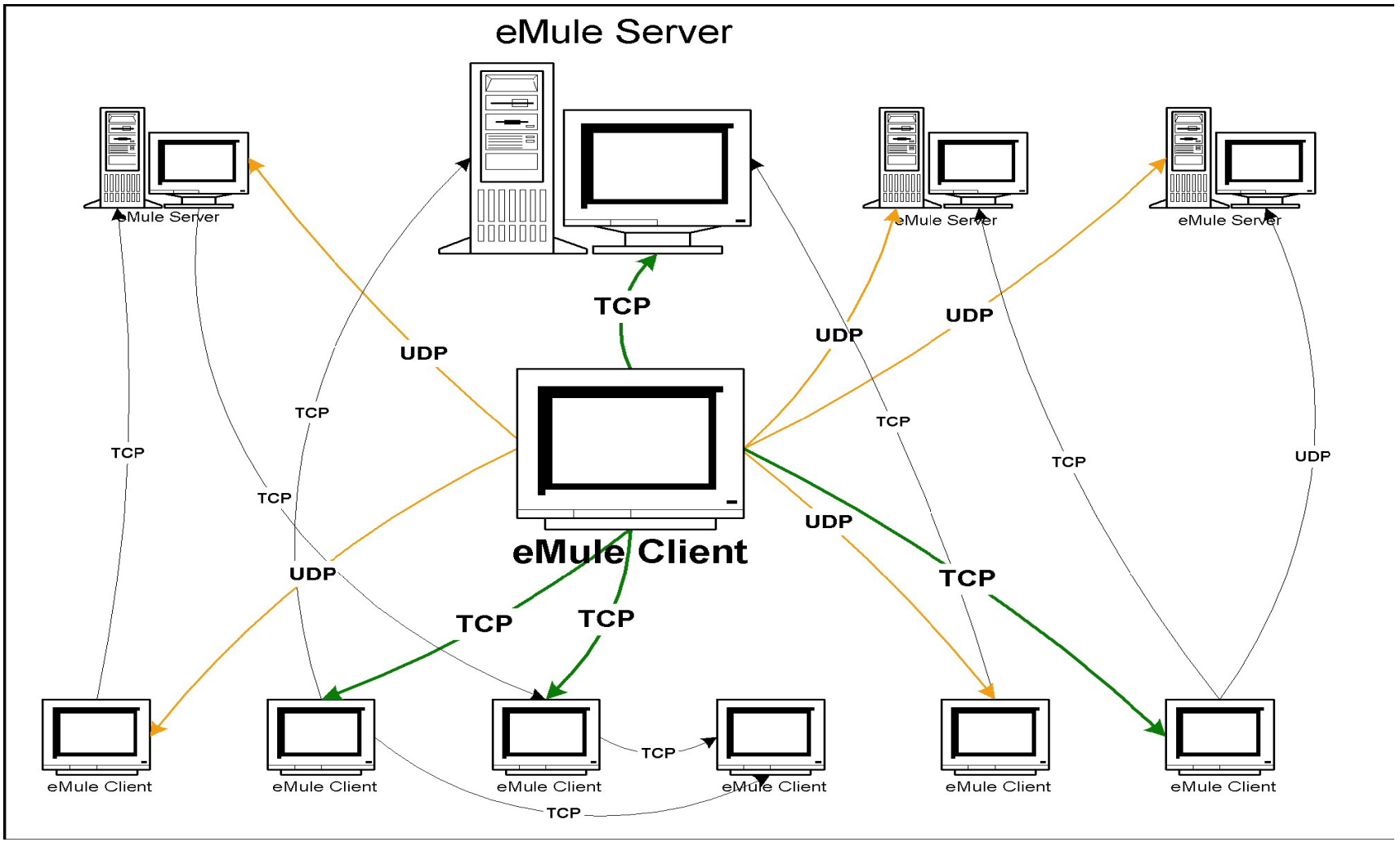
- possiede una lista di servers eMule precaricata
- si connette ad un unico server eMule mediante una connessione TCP. Il server può essere modificato dinamicamente con l'intervento dell'utente
- Apre centinaia di connessioni TCP con altri client eMule per effettuare il download/l'upload dei files
- può scaricare frammenti diversi dello stesso file da clients eMule diversi
- può effettuare l'upload di frammenti di un file F anche se il download di F non è ancora stato completato
- invia pacchetti UDP ai servers presenti nella sua lista per verificare la loro presenza sulla rete, per migliorare la ricerca di files,….
- invia pacchetti UDP agli altri clients per controllare il proprio stato nelle code degli altri clients

eMule: IL SERVER

Ogni server eMule

- Possiede un database in cui memorizza i descrittori dei files messi in condivisione dagli utenti ad esso connessi
- Invia ad ogni client eMule, al momento della sua connessione, informazioni circa il numero di utenti connessi ad esso e sul numero di files condivisi
- Non memorizza i files condivisi
- Non interagisce con gli altri server eMule
- Può essere utilizzato come 'punto di appoggio' per utenti a monte di NATs/firewalls

LA RETE ED2k:ARCHITETTURA



IDENTIFICAZIONE DELLE ENTITA' NELLA RETE eMule

USER ID (User Hash)

- stringa di 128 bits (16 bytes) ottenuta concatenando numeri random
- utilizzato per implementare il sistema dei crediti
- assegnato la prima volta che eMule viene lanciato, non viene modificato nelle sessioni successive
- memorizzato nel file preferences.dat nella directory emule/config, identifica il client/host

CLIENT ID

- Stringa di 32 bits (4 bytes)
- Valida solamente per la durata di una sessione TCP
- Può essere assegnato un ID ALTO oppure un ID BASSO a seconda del tipo di connessione tra l'utente e la rete

FILE HASH

- Stringa hash di 128 bits calolata mediante l'uso di MD4
- Assegnata dal sistema non appena il file viene messo in condivisione
- Identifica univocamente il file all'interno della rete
- Può essere individuato nella finestra file condivisi, colonna FILE ID

IL PROTOCOLLO CLIENT SERVER

- Ogni client si connette al massimo ad un server per volta
- L'utente può scegliere il server da una lista oppure lasciare la scelta automaticamente al sistema. Successivamente, il server può essere cambiato solo su intervento dell'utente
- Si stabiliscono due connessioni TCP: una dal client al server e viceversa
- Nel momento in cui la connessione viene stabilita il server
  - identifica il client
  - decide se assegnare al client un ID BASSO oppure un ID ALTO, valido per l'intera sessione e determinato in base alla raggiungibilità del client dall'esterno
  - può decidere di rifiutare la connessione con il client (ad esempio quando il numero di connessioni gestite dal server supera una soglia predefinita)

ASSEGNAZIONE DEL CLIENT ID

- Il Server assegna ad un client C un ID basso se C non può accettare connessioni in ingresso (ad esempio si trova amonte di un NAT o di un firewall)
- ID ALTO il client C consente agli altri client dalla rete eMule di connettersi liberamente alla porta TCP eMule 4662
- ID ALTO: viene calcolato a partire dall'indirizzo IP dell'host
  - Se l'indirizzo IP del client è X.Y.Z.W
    - ID = X+ 256*Y + 256*256*Z +256*256*256*W
- L'ID (alto o basso) viene visualizzato al momento della connessione col server
  - **06/05/2007 0.15.35: WARNING DonkeyServer No2 (62.241.53.16:4242) – Your**
  - **20603 port is not reachable. Please review your network config.**
  - **06/05/2007 0.15.35: Connessione stabilita con: DonkeyServer No2**
  - **06/05/2007 0.15.35: Caricamento fonti dal nuovo server in corso per tutti i file con**
  - **meno di 400 fonti**
  - **06/05/2007 0.15.35: Il nuovo ID è 4241605**
  - **06/05/2007 0.15.35: Ricevuti 1 nuovi servers**

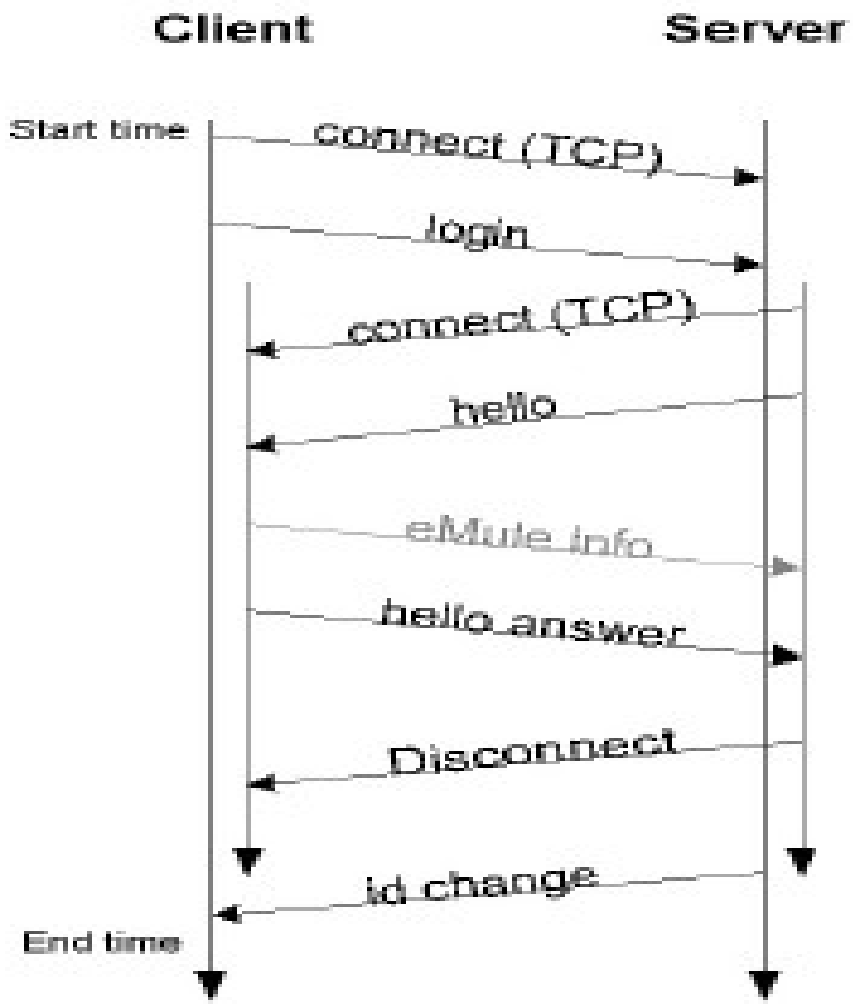
SVANTAGGI DI UN ID BASSO

- Un client con ID basso in generale non possiede un indirizzo pubblico a cui gli altri client si possono collegare
- La comunicazione richiede l'intervento del server eMule ed aumenta il carico computazionale sul server.
- Per questa ragione un server accetta poche connessioni da ID bassi e quindi è più facile che la connessione venga rifiutata da un server
- La connessione tra due utenti con ID basso non può avvenire (era possibile nelle versioni precedenti di eMule utilizzando il server come appoggio, poi questa funzionalità è stata eliminata per non sovraccaricare il server)
- un utente con ID basso può effettuare upload di file solo con utenti collegati allo stesso server.

COME FARE PER OTTENERE UN ID ALTO

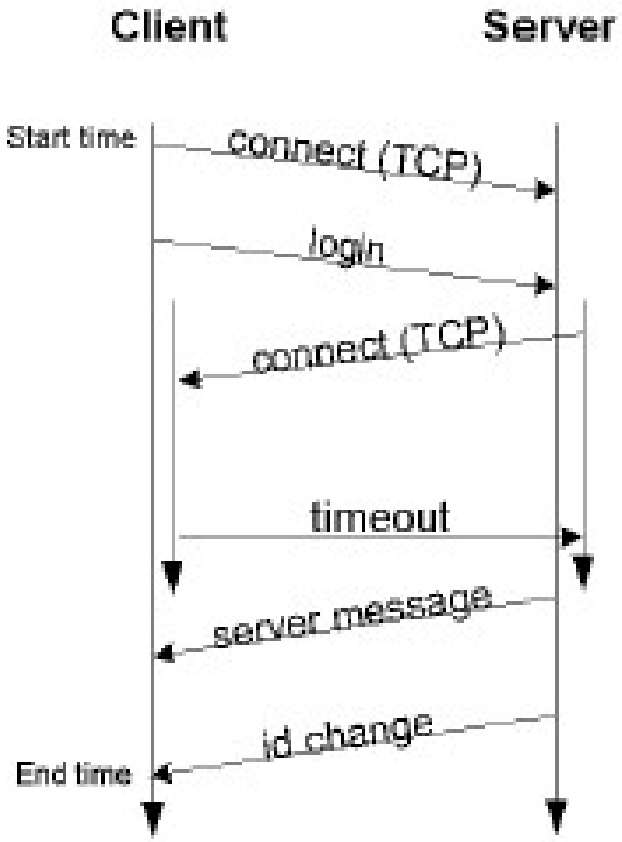
- Configurare opportunamente il firewall
- Aprire le porte sul router/NAT
  - Configurare il router/NAT in modo che tutte le comunicazioni provenienti dall'esterno e dirette alle porte TCP 4662 ed UDP 4672 siano instradate verso l'IP corrispondente al computer su cui è in esecuzione eMule
  - Se esistono più clients collegati al router mediante una rete locale, occorre associare alle diverse istanze di eMule porte diverse e "aprire" tutte queste porte sul router
  - E' inoltre necessario 'spegnere' il DHCP, perchè questo assegnerebbe un indirizzo IP diverso ogni volta che il sistema viene reinizializzato
  - L'apertura delle porte spesso può essere effettuata solo dall'amministratore della rete

ASSEGNAZIONE CLIENT ID ALTO



ID-CHANGE = assegnazione dell'id al client

ASSEGNAZIONE CLIENT ID BASSO

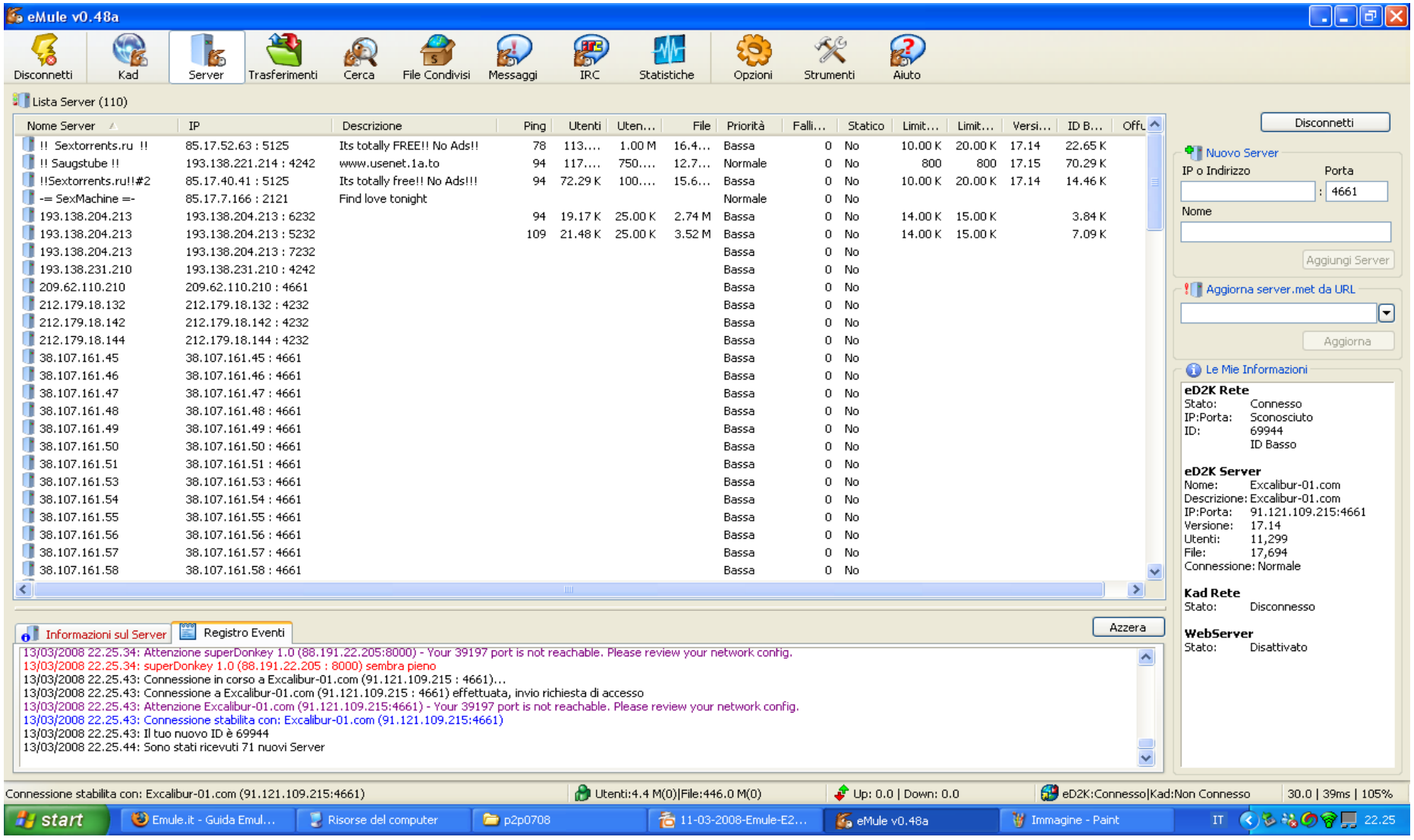


- Il server non riesce a connettersi al client
- Il server invia sulla connessione aperta dal client un messaggio del tipo:

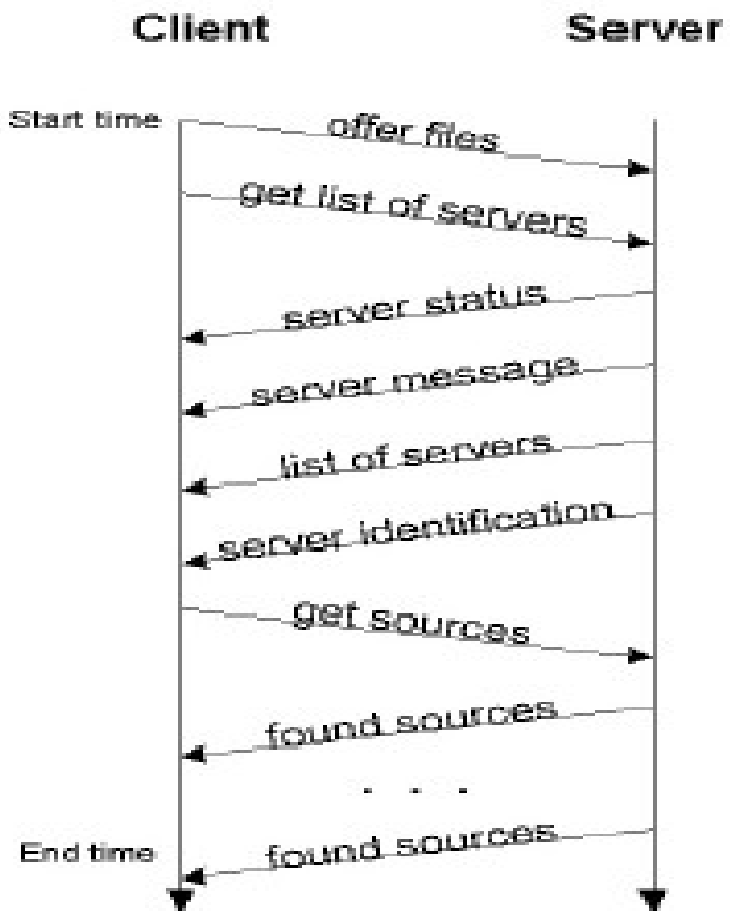
Warning: You have a low ID. Please review your network configuration and/or your settings

- la fase di connessione si conclude con un messaggio di ID CHANGE che assegna al client un ID basso

CONNESSIONE CON IL SERVER



CLIENT-SERVER START UP



MESSAGGI DI START UP

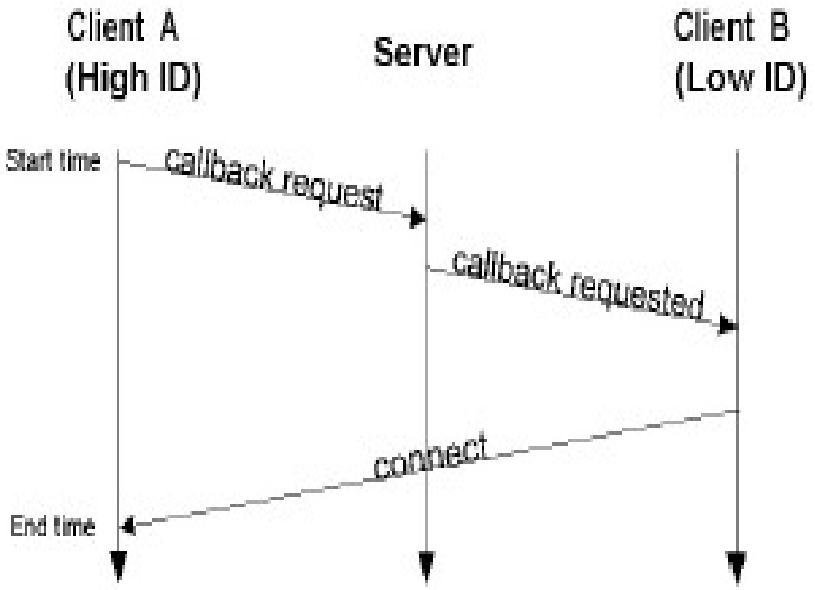
- Il client invia la lista dei files che intende condividere

- Il server invia una lista di servers da lui conosciuti

- La figura assume che il client C possieda una lista L di files richiesti al momento della connessione con il server.

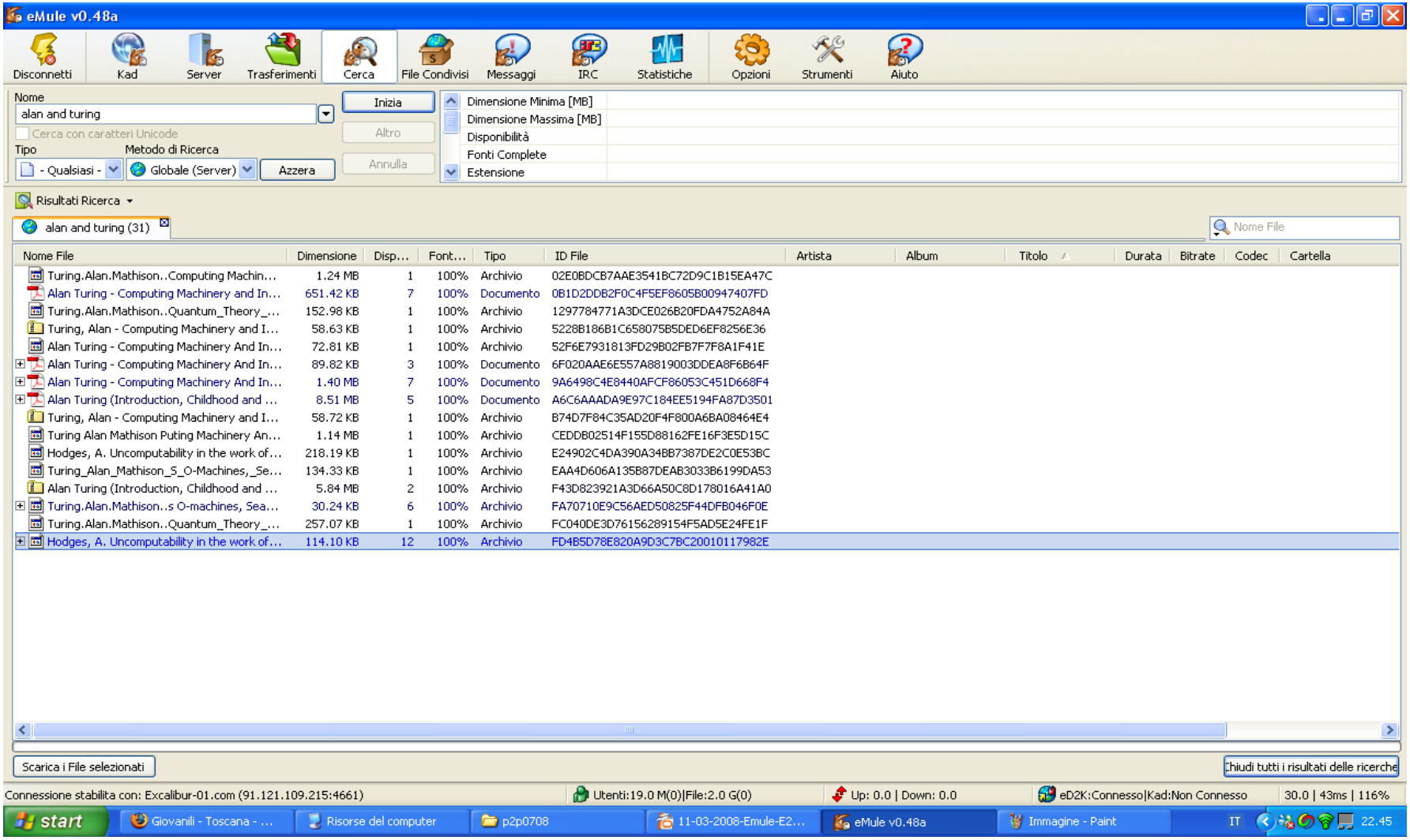
- Il server invia automaticamente nuove fonti per tutti i files in L per cui C possiede un numero limitato di fonti (valore soglia prestabilito)

ID BASSI: IL MECCANISMO DELLE CALLBACKS

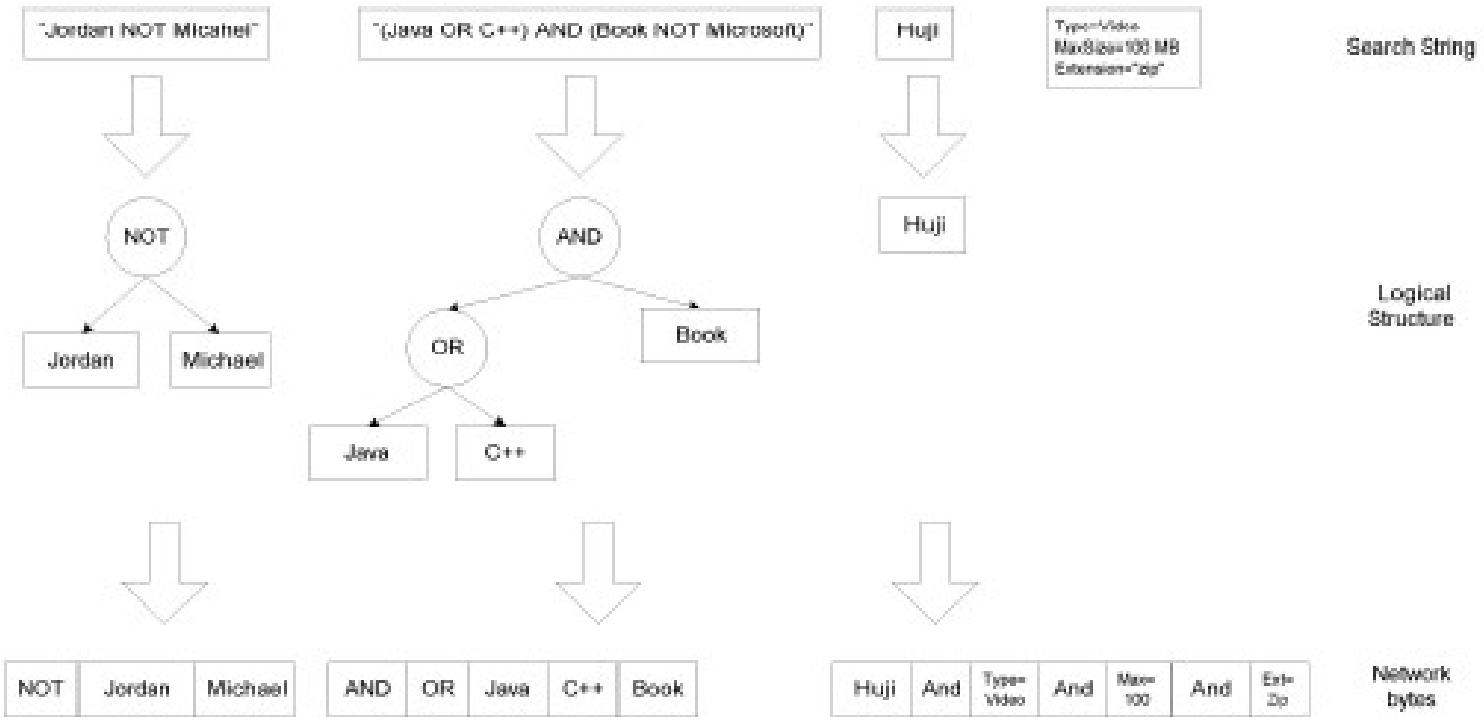


- Consentono ad un utente A con ID alto A di connettersi a B di ID basso
- A richiede un file memorizzato su B che ha un ID basso
- A chiede al server una 'callback'ed il server, che ha già aperto una connessione TCP con B, invia la richiesta di callback a B. B apre una connessione con A
- A e B devono essere connessi allo stesso server

Emule: RICERCA DI FILES

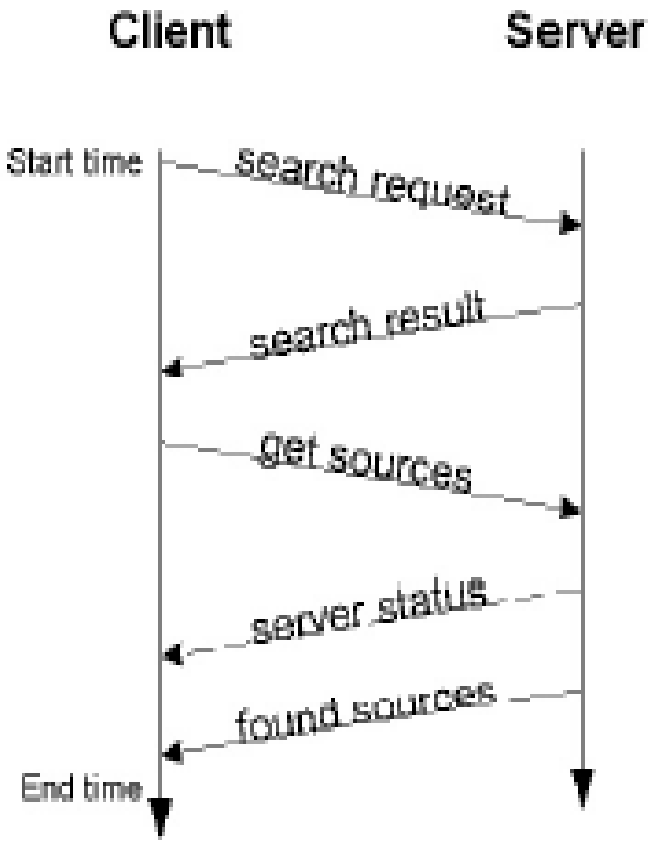


INTERAZIONE CLIENT/SERVER: RICERCA DI FILES



- ogni query booleana viene rappresentata mediante un albero
- l'albero viene linearizzato mediante una visita anticipata

INTERAZIONE CLIENT-SERVER: RICERCA DI FILES



- Il client invia una query che può contenere operatori booleani AND, OR, NOT

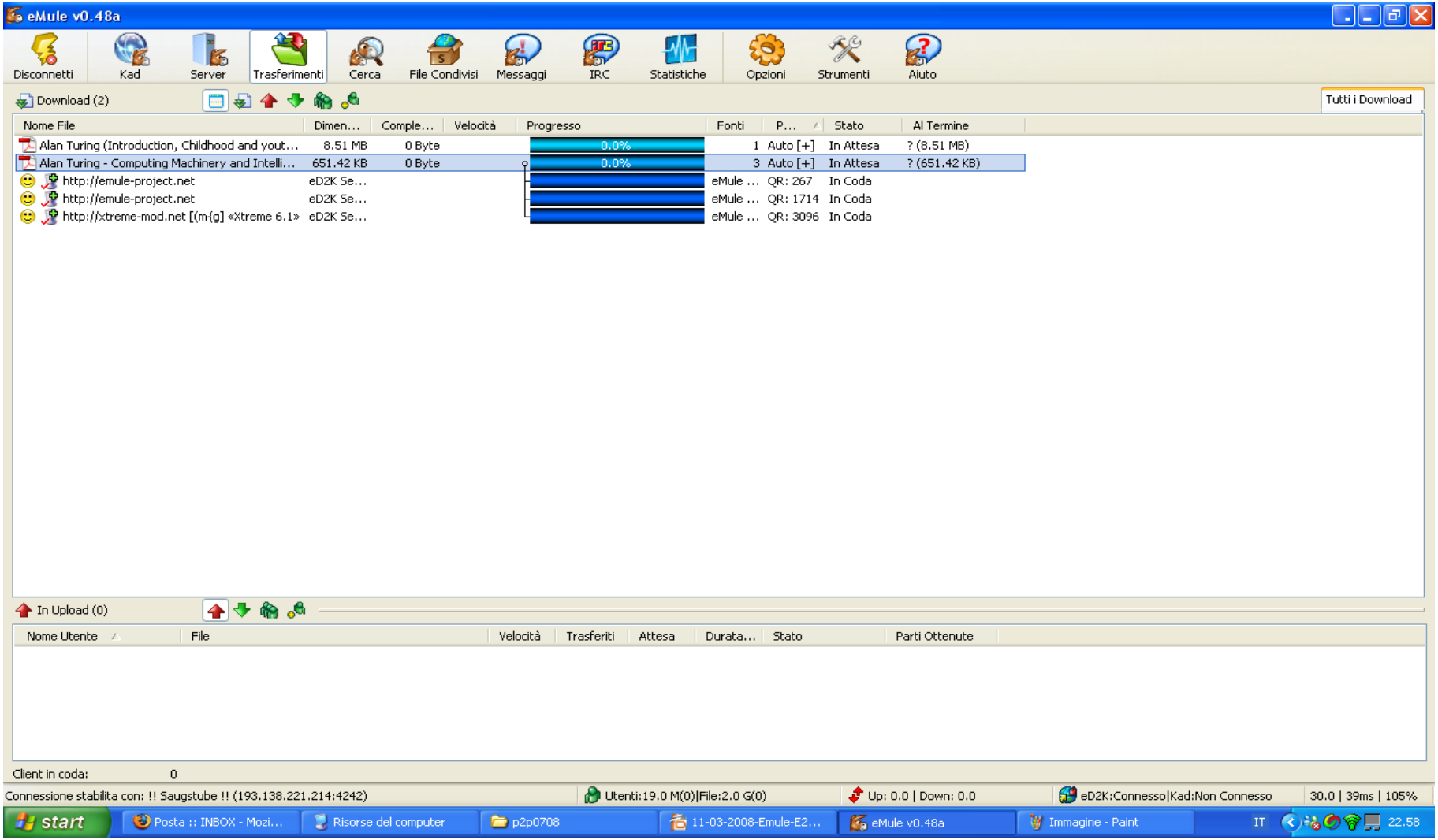
- il server risponde con una lista di files che soddisfano la query

- il client sceglie uno o più files da scaricare

- il server comunica una lista di fonti per i files prescelti (ogni fonte può possedere tutto il file o una parte di esso)

- status message contiene informazioni sul numero di utenti e di files gestiti dal server

INTERAZIONE CLIENT SERVER: LISTA DI FONTI



INTERAZIONI UDP TRA CLIENT E SERVERS

La comunicazione UDP tra client e server viene utilizzata per:

Keep alive:

- il client verifica periodicamente lo stato dei servers nella sua lista,
- la richiesta del client include un numero random su cui viene fatto l'eco da parte del server
- ogni volta che il client invia un keep alive al server, incrementa un contatore *C*
- ogni volta che il client riceve un messaggio di qualsiasi tipo dal server, *C* viene decrementato
- se *C* raggiunge un valore soglia, il server viene considerato offline e viene escluso dalla lista dei servers

Aumentare il numero di match per una query

Aumentare il numero di fonti

- se il numero di fonti per un certo files ricevute dal server di riferimento risulta sotto una certa soglia, il client invia pacchetti UDP agli altri servers nella sua server list per ottenere ulteriori fonti

COMUNICAZIONE CLIENT/CLIENT

- La maggior parte dei messaggi viene spedita su connessioni TCP
- Viene stabilita una diversa connessione TCP per ogni coppia [file, client]
- Connection startup: prevede un handshake simmetrico in cui i due clients si identificano mediante una procedura di Secure User Identification
- Ogni client
  - mantiene una singola coda in cui inserisce tutte le richieste di upload
  - coda di upload = coda a priorità
  - in ogni istante serve un numero limitato di download, in modo da non esaurire tutta la banda a sua disposizione
  - quando un client A richiede il download di un file da B, la sua richiesta
    - viene servita immediatamente se la coda di upload di B è vuota ed il numero di upload attivi in B non supera una certa soglia
    - altrimenti la richiesta viene inserita nella coda di upload di B
    - B invia ad A un messaggio indicando la posizione di A nella propria coda

INTERAZIONE TRA CLIENTS: GESTIONE DELLE CODE

- Priorità = (rating * waiting time) / 100
- La priorità di un client nella coda di upload dipende da
  - il tempo di attesa in coda
  - un rating che dipende
    - dal credito acquisito (vedi lucidi successivi) dal client CD che effettua il download presso il client CU che effettua l'upload (credito acquisito da CD per precedenti upload effettuati da CD nei confronti di CU) (1-10)
    - dalla priorità del file che si sta scaricando (0.2-1.8)
    - In genere, rating = credito acquisito * priorità del file
- Il download inizia quando il client raggiunge la prima posizione della coda
- Il downlaod continua finchè
  - Il client interrompe il downlaod
  - Preemption: un client con una priorità più alta richiede il downlaod

INTERAZIONE TRA CLIENTS: GESTIONE DELLE CODE

- Nel client eMule, nella finestra downloads è possibile individuare, accanto al nome del file in download il suo QR (Queue Rank)
- QR= 100, esistono 100 utenti in coda prima che devono essere serviti prima che la mia richiesta venga presa in considerazione
- QR da un'approssimazione del tempo di attesa, infatti è possibile che altri utenti mi superino sulla coda perchè possiedono un numero maggiore di crediti

INTERAZIONI TRA CLIENTS: IL SISTEMA DEI CREDITI

- Sistema di crediti: introdotto per incentivare la condivisione dei files da parte degli utenti
- Quando un client *C1* effettua l'upload di un file verso un client *C2*, *C1* viene 'ricompensato' da *C2* mediante l'attribuzione di un credito
- I crediti non sono globali,
- Un credito è un valore intero nell'intervallo 1-10 che viene assegnato ad ogni coppia ordinata (client, client)
- Un client memorizza in un proprio file i crediti degli altri clients da cui ha scaricato in precedenza qualche file
- I crediti di proprietà di un client *C* non vengono quindi memorizzati da *C*, ma dai clients che gli hanno concesso il credito, perchè hanno scaricato qualche file da *C*
- Non è quindi possibile visualizzare i propri crediti
- Viene introdotto un meccanismo di identificazione sicura dei clients per prevenire la richiesta illecita di crediti da parte di clients che non li posseggono

INTERAZIONE TRA CLIENTS: IL SISTEMA DEI CREDITI

- eMule per ogni utente tiene traccia, per ogni utente, di tutti i Megabyte forniti, per un periodo di 5 mesi .
- crediti mantenuti in un file nella cartella di configurazione di e-muale
- All'avvio eMule cancella i crediti scaduti

CALCOLO DEL VALORE DEI CREDITI

- Il valore dei crediti varia da 1 a 10 e viene calcolato mediante 2 formule
- Analizziamo il punteggio che un peer P_1 assegna al peer P_2 per influenzare la posizione di P_2 nella sua coda di upload
- P_1 calcola i seguenti valori
  - TotaleBytesRicevuti(da P_2) * 2 / TotaleBytesInviati (a P_2)
  - SQRT (TotaleBytesRicevuti(da P_2) +2)

  e poi sceglie il valore più basso tra i due
- Condizioni particolari
  - Se TotaleBytesRicevuti < 1MB il valore del credito è 1
  - Se TotaleBytesInviati = 0, il valore del credito è 10

CALCOLO DEL VALORE DEI CREDITI

Esempio: analizziamo il punteggio che il mio client eMule attribuisce ad un client C.

Se ho ricevuto in passato 10MB da C e ne ho inviati 1:

FORMULA 1

- crediti = (Megabytes ricevuti da C * 2) / Mega bytes inviati a C

  Crediti di C = (10 * 2) / 1 = 20 (cioè 10 dato che il massimo è 10)

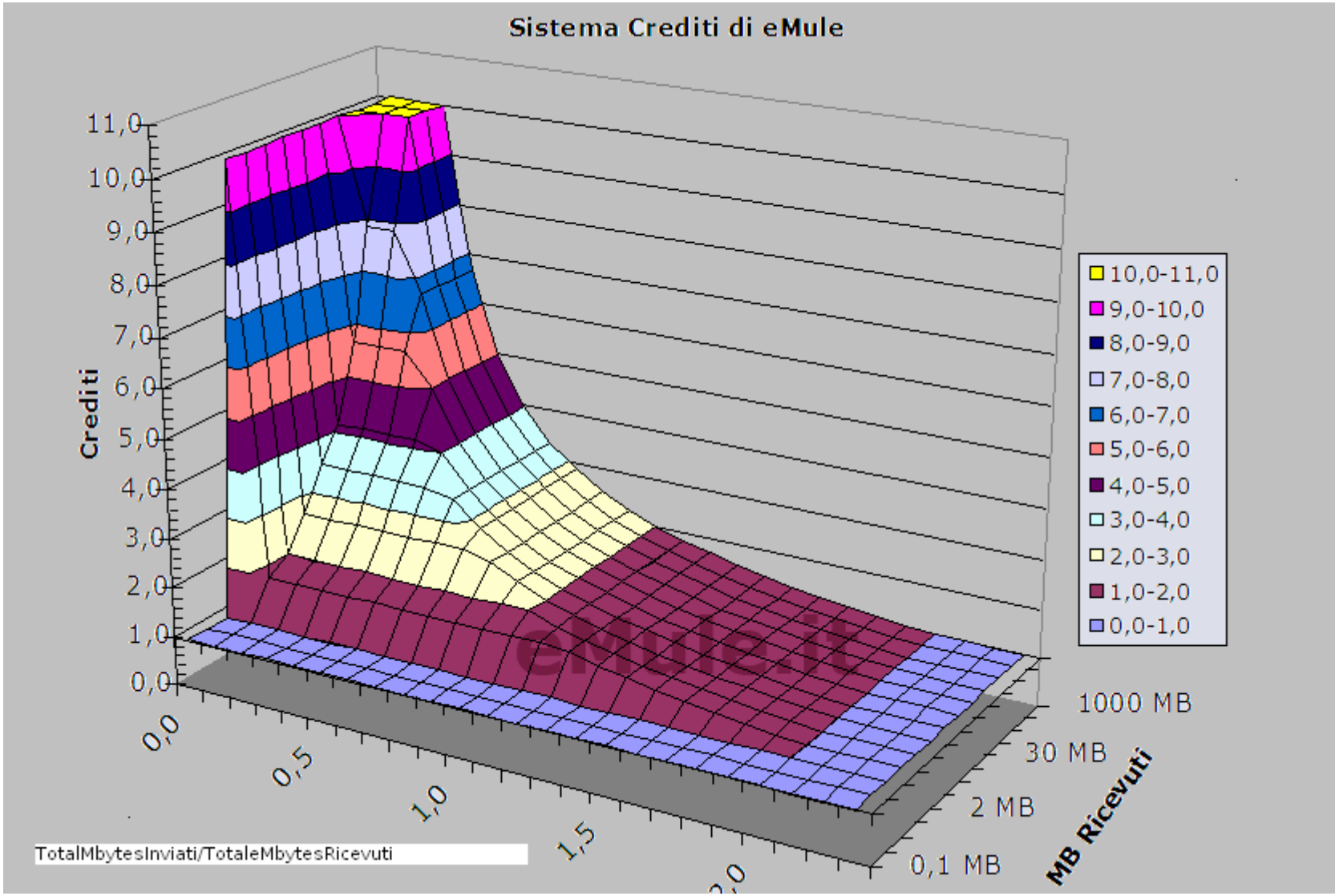
FORMULA2

- crediti = SQRT (Megabyte ricevuti + 2)

  Crediti di C = SQRT (10 + 2) = circa 3.5

infine eMule sceglie il valore più basso fra i due calcolati quindi 3.5.

CALCOLO DEL VALORE DEI CREDITI



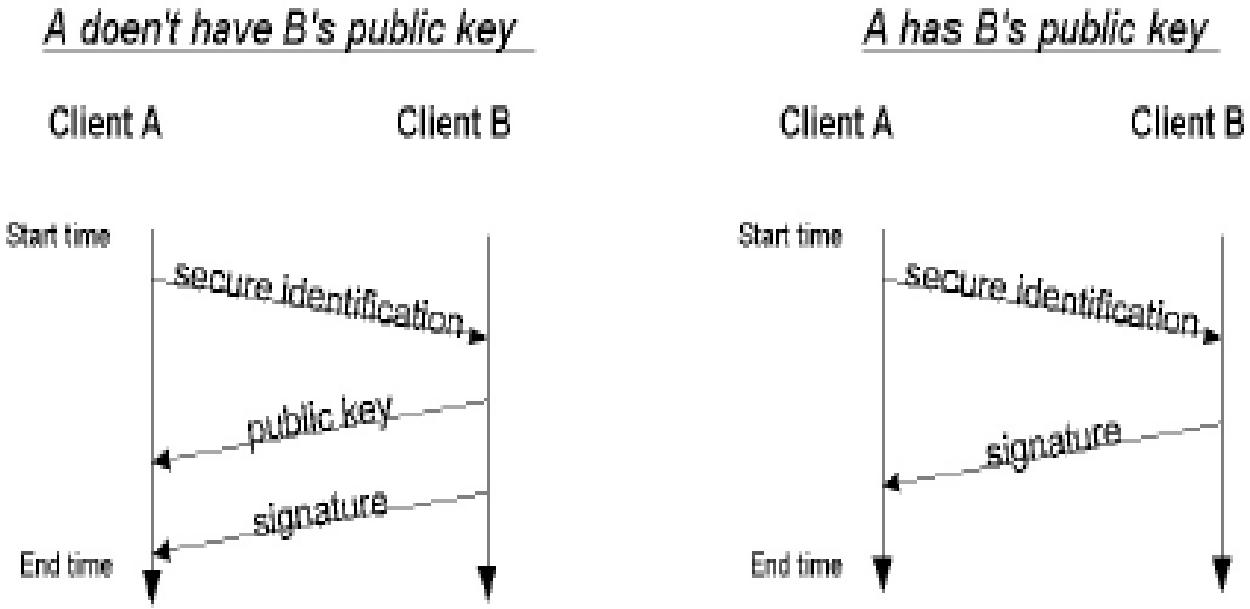
SECURE CLIENT IDENTIFICATION

- E' necessario utilizzare un sistema di identificazione sicura dei clients per evitare che un client si impossessi dei crediti destinati ad un altro client

- Basato su un sistema a crittografia asimmetrica (RSA)

- Supponiamo che un client B si deva autenticare presso il client A
  - B invia la sua identità e la firma generando la firma mediante la sua chiave privata,
  - A reperisce la chiave pubblica di B
  - A utilizza la chiave pubblica di B per decodificare la firma di B e verificare l'identità di B
  - eMule crea in automatico alla prima connessione la coppia chiave pubblica-chiave privata, che viene memorizzata nella cartella config di eMule.

SECURE CLIENT IDENTIFICATION



INTERAZIONE TRA CLIENTS: SCAMBIO FONTI

- Una connessione diversa per ogni coppia [file, client]
- A invia l'identificatore del file che intende scaricare
- B risponde con informazioni relative al file richiesto (status)
- Inoltre B passa ad A le fonti di A di sua conoscenza.

INTERAZIONE TRA CLIENTS: SCAMBIO DELLE FONTI

- Ogni volta che richiedo il download di un file, eMule ricerca le fonti per quel file tramite i servers contenuti nella lista del client
- La ricerca avviene subito sul server di riferimento e, ad intervalli più lunghi, sugli altri servers della lista (mediante comunicazione UDP)
- Inoltre, quando si contatta una fonte per il download, il client richiede alla fonte contattata la lista delle fonti in suo possesso per il medesimo file
- Ciò avviene sia se la fonte contattata è completa sia se solo alcune parti di quel file sono disponibili su quella fonte..
- Gli stessi client forniscono fonti per i files condivisi

VANTAGGI

- Ricerca alternativa di fonti: percentuale di fonti contattate su quelle disponibili nella rete eMule molto prossima al 100%.
- Maggiore velocità di contatto, non dovendo passare per i server.
- Minor carico di lavoro dei server, con evidente miglioramento delle prestazioni del sistema.

TRASFERIMENTO DI FILES TRA CLIENTS

- Un client può
  - scaricare solo una parte del file da un altro client
  - scaricare parti diverse dello stesso file da utenti diversi
- Le richieste di download si riferiscono a parti di files
- Tutti i blocchi scaricati da uno stesso client appartengono alla stessa parte
- Quando il downloader D richiede ad un uploader U il download di un file F, U può possedere solo alcune parti del file, perchè ne sta effettuando il download
- U invia a D un messaggio in cui indica se possiede tutto il file ed, in caso negativo quali parti possiede
- D può richiedere a U il download di alcune parti del file (massimo 3 parti per ogni messaggio)

TRASFERIMENTO DI FILES TRA CLIENTS

- Il client che effettua il download determina
  - quali parti scaricare del file selezionato
  - l'ordine con cui le parti vengono scaricate
  - da quale client vengono scaricate

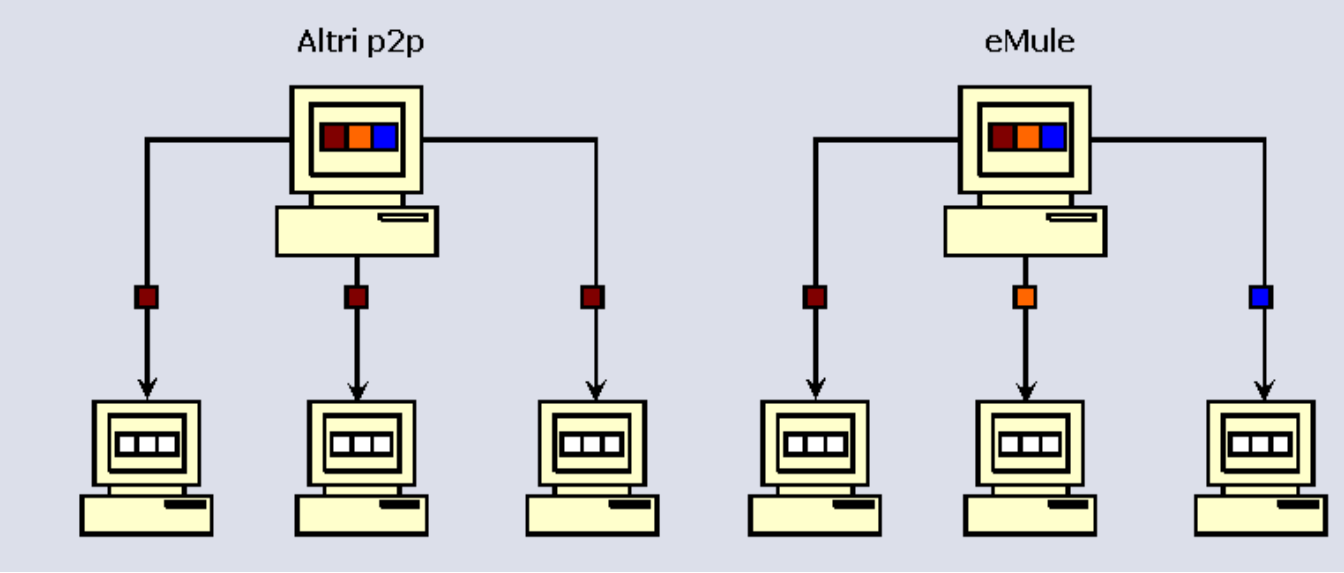
Selezione delle parti da scaricare

- Distribuire le richieste di parti tra le sorgenti conosciute
- Scaricare per prime le parti più rare
- Prima la prima e l'ultima parte

Part rating: viene attribuito un punteggio diverso ad ogni parte che dipende dalla

- Disponibiltà delle parti
- Parti importanti (prima e ultima parte)

eMule: UPLOADING DEI FILES

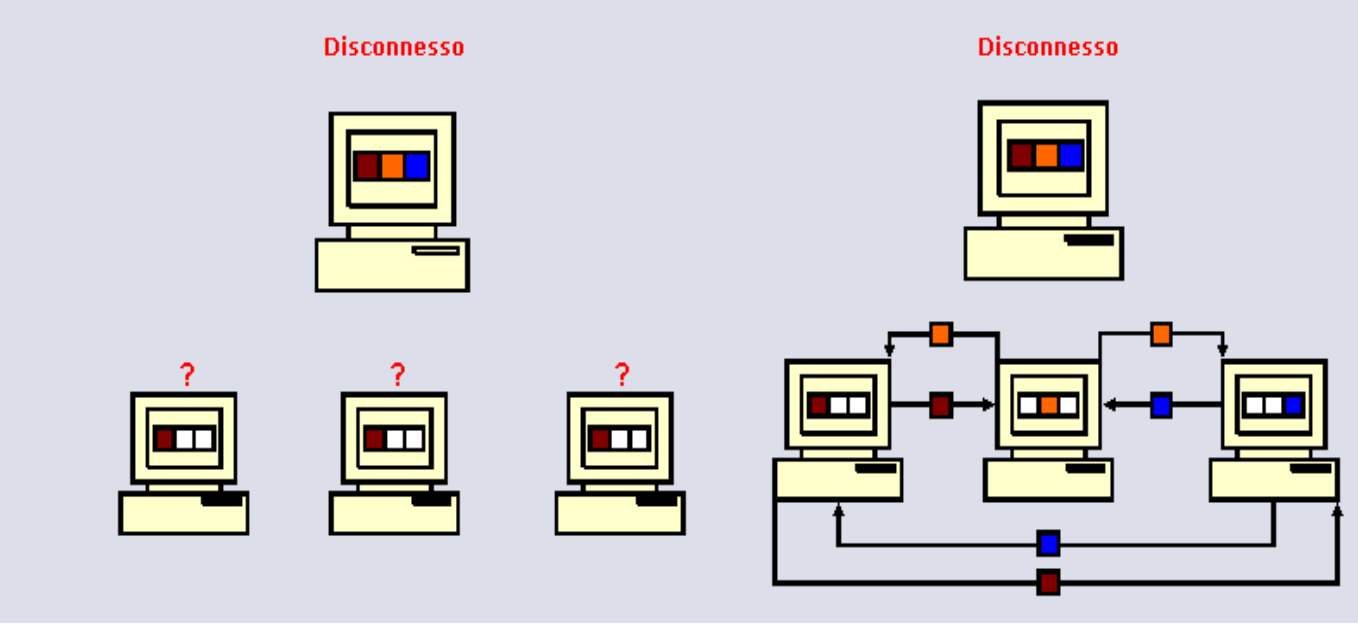


- strategia di uploading:

se un file viene richiesto da più di un downloader, eMule cerca di inviare a ciascun downloader una parte diversa del file

- in questo modo si tende a distribuire più rapidamente il file nella Rete e-Mule

eMule: UPLOADING DEI FILES



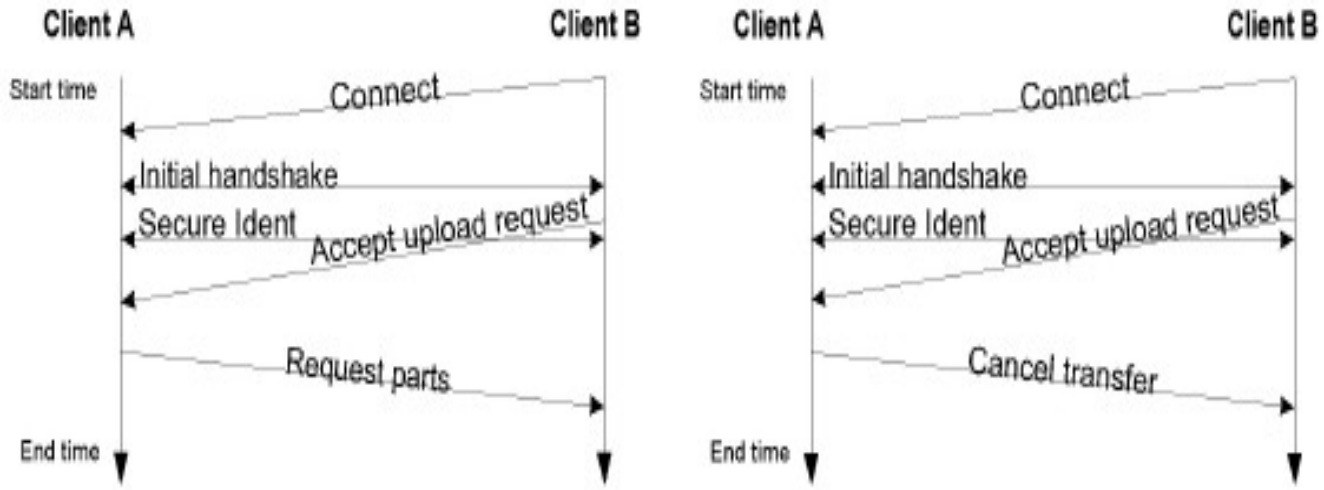
Supponiamo che l'uploader U si disconnetta dopo aver inviato una parte del file ad ogni client eMule

- se U ha inviato la stessa parte P del file F a tutti i clients, tutti si possiedono solo P e devono attendere la connessione successiva di U per poter scaricare F (figura di sinistra)
- se U ha distribuito parti diverse a client diversi, il file è presente nella rete ed i clients possono scambiarsi le parti in modo da ricostruire il file

TRASFERIMENTO DI FILES TRA CLIENTS

- Quando inizio il download di un file, vengono contattate più fonti e la richiesta viene inserita nelle code di upload di tutte le fonti
- Quando il client *C* ha completato il download dell'intero file, non lo notifica agli altri clients
- Quando un client invia il messaggio di upload a *C*, la proposta di uplaod viene rifiutata da *C*

TRASFERIMENTO DI FILES TRA CLIENTS



- Quando A raggiunge la prima posizione nella coda di upload di B

B apre una connessione con A

- B chiede una connessione ad A

- A può accettare la connessione , oppure rifiutarla, se il file è già stato scaricato da altre fonti

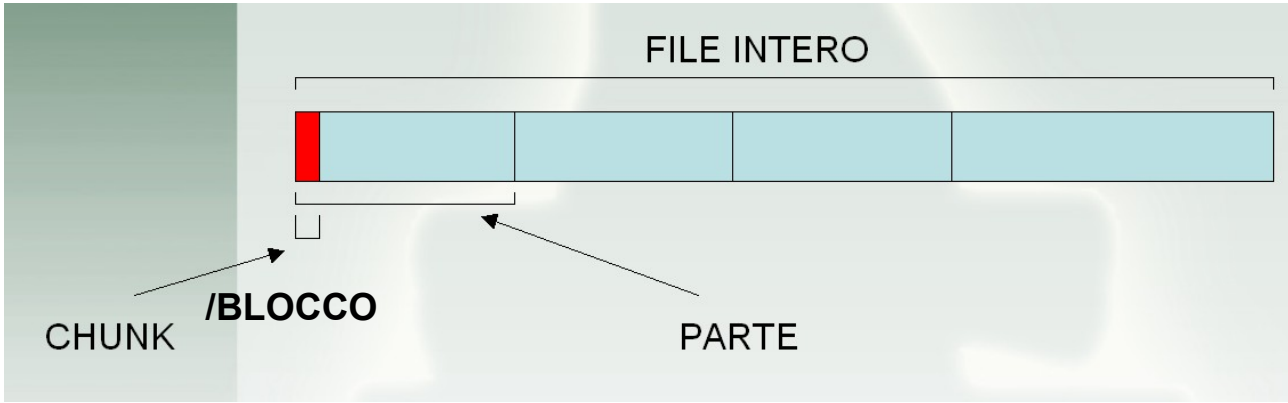
IDENTIFICAZIONE DEI FILES NELLA RETE E-MULE

- per identificare un file in una rete P2P, non è sufficiente usare il nome del file, perché file diversi potrebbero avere lo stesso nome e file con medesimo contenuto potrebbero avere nomi diversi.

Link e2k :

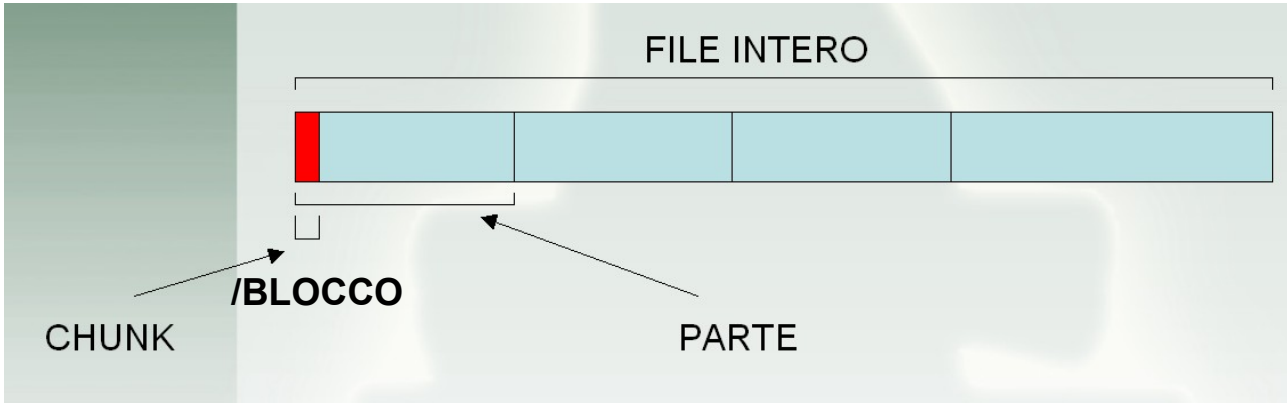
- Ed2k= abbreviazione di eDonkey2000, il nome del primo client sviluppato per reti eDonkey.
- I link e2k
  - identificano in modo univoco i file condivisi della rete eDonkey.
  - contengono tutte le informazioni necessarie per poter eseguire una ricerca del file nella rete condivisa e poter così individuare i client che hanno a disposizione quel file (la dimensione del file, l'MD4,.....)
  - non sempre indicano in modo esplicito l' indirizzo dal quale prelevare il file
  - permettono di trattare i file della rete eDonkey come i comuni file disponibili in Internet tramite protocolli più tradizionali come FTP o HTTP.
  - possono ad esempio essere inseriti in una pagina web, e permettere all' utente di interagire con il client eDonkey per il download automatico del file.

TRAS>FERIMENTO DI DATI TRA I CLIENTS



- ogni file presente nella rete eMule viene decomposto in parti e quindi ogni parte viene frammentata
  - parti da 9.28 MB
  - ogni parte è a sua volta divisa in chunks(blocchi) da 180K bytes
- Per ogni parte viene calcolato un hash mediante MD4
- HashSet:contiene gli hash di ogni parte del file
- I part hash sono utilizzati per calcolare l'hash finale delfile
- esempio: ed2k:// | file | ubuntu-5.10-install-i386.iso | 647129088 | 901E6AA2A6ACDC43A83AE3FC211120B0|/

TRAS>FERIMENTO DI DATI TRA I CLIENTS



Esempio di link e2k riferito ad un file composto da una sola parte

- ed2k://|file|ubuntu-5.10-install-i386.iso|lungh.file|901E6AA2A6ACD.....

Esempio di link e2k riferito ad un file composto da più parti

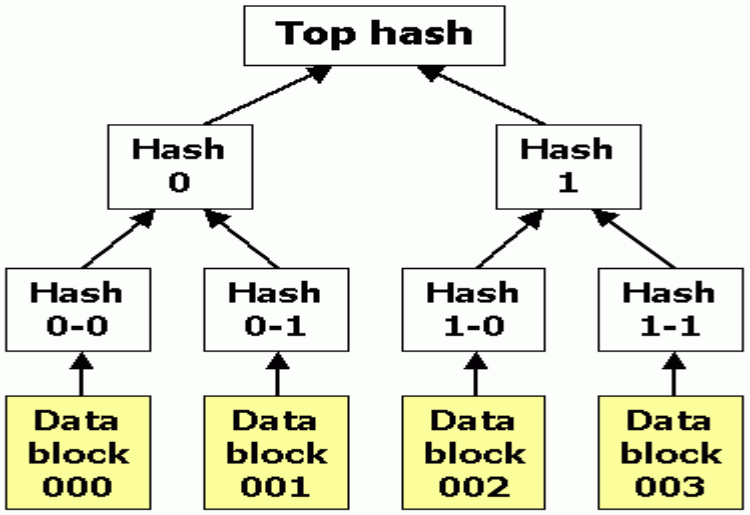
- ed2k://|file|ubuntu-5.10-install-i386.iso|lungh.file|901E6AA2A6ACD.....|
  p=264E6F6B.....:17B9A4D1DCE0E4C.....|/

riporta l'hash dell'intero file e quellodi ogni sua parte

INTELLIGENT CORRUPTION HANDLING

- Il client scarica l'hashset del file prima di una qualsiasi sua parte
- Ogni volta che un client scarica una parte del file, calcola l'hash H di quella parte e confronta H con il valore corrispondente contenuto nell'hash set
- Se i valori coincidono, la parte è integra e può essere messa in condivisione
- Se i valori non coincidono, si ripetere il download della parte, un chunk alla volta
- Per ogni chunk/blocco scaricato viene ricalcolato l'hash su tutta la parte, se il valore calcolato coincide con quello nell'hash set, la parte viene considerata 'riparata' e si evita di trasferire il resto dei chunks di quella parte
- Questo consente di evitare di ripetere in media il 50% del download dei chunks del file

HASH TREES

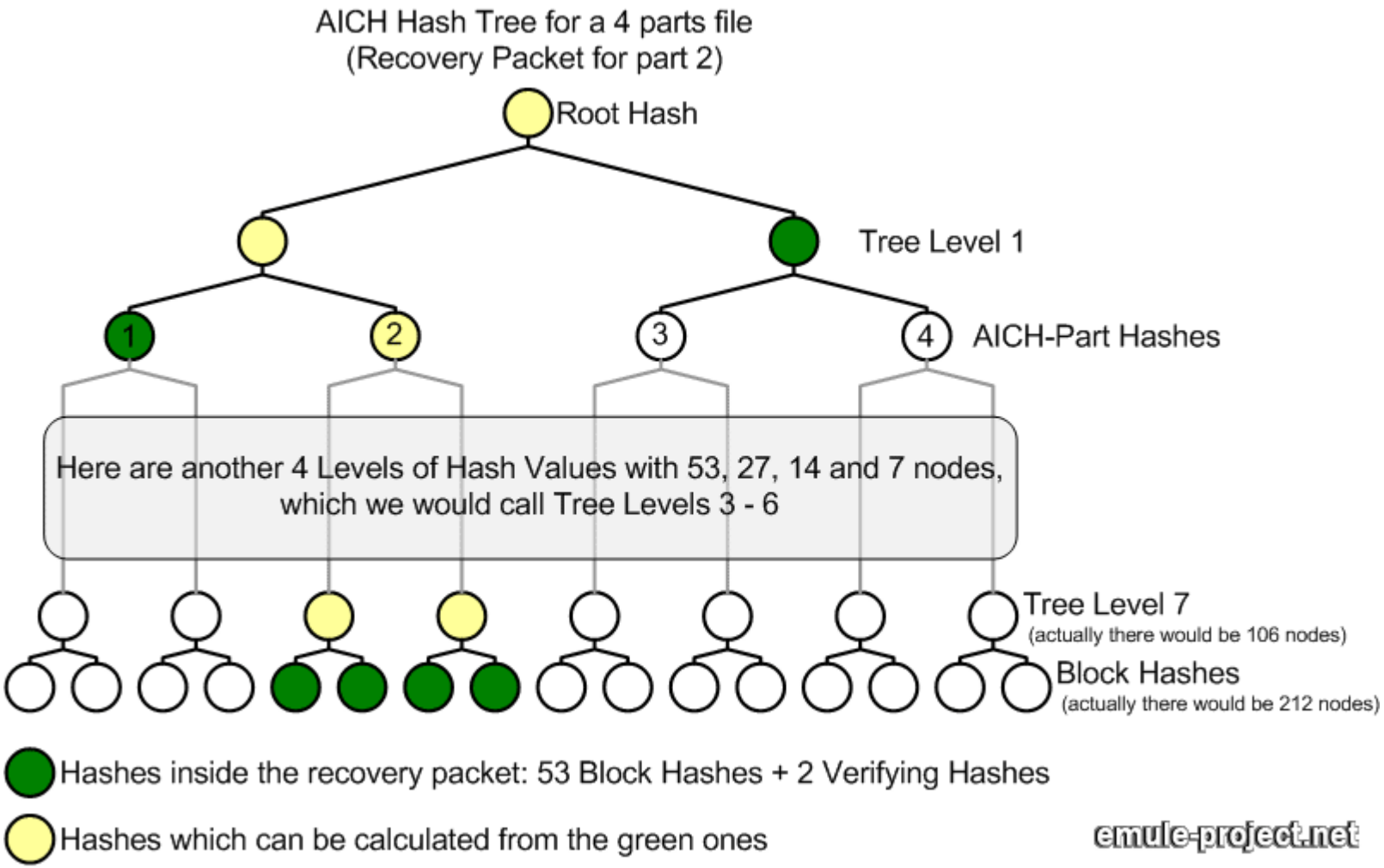


- Consideriamo un documento suddiviso in parti
- Le foglie contengono gli hash delle singole parti
- I nodi interni contengono l'hash dei rispettivi figli
- Ad esempio : hash 0 = hash (hash 00 || hash 01)
- in genere vengono utilizzati alberi binari, ma è possibile utilizzare anche alberi generali
- Vantaggio: E' possibile verificare l'integrità di ogni cammino dell'albero in modo indipendente, senza conoscere l'intero albero

AICH:ADVANCED INTELLIGENT CORRUPTION HANDLING

- Supponiamo che un client eMule scarichi una parte di un file e si accorga, tramite l'hash della parte (contenuto nell'hash set) che la parte è corrotta
- Nel caso dell' ICH, si scaricano di nuovo tutti i chunks della parte,fino a che la parte non risulta riparata
- AICH: approccio alternativo che permette di scaricare di nuovo solo i chunks corrotti
- E' necessario calcolare un hash per ogni blocco. L'hash set risultante può assumere dimensioni molto grandi (es: 24.000 hash per un file di 4 GB)
- Il client calcola l'hash set di un file che intende condividere e
  - lo organizza mediante un hash tree binario, come quello mostrato nella pagina precedente
  - lo memorizza su disco
  - lo invia ai client che ne fanno richiesta
  - l'hash tree è utilizzato per verificare l'integrità del file

UN ESEMPIO DI EMULE HASH TREE



AICH:ADVANCED INTELLIGENT CORRUPTION HANDLING

- Supponiamo che un client abbia scaricato una parte e rilevi, mediante l'hash set, che la parte è corrotta
- Il client deve ricercare l'hash set relativo a tutti i blocchi appartenenti alla parte corrotta
- Pacchetto di verifica: contiene questi hash più alcuni altri valori hash necessari per verificare se l'hash set dei blocchi è a sua volta integro
- Si calcola l'hash di ogni chunk della parte corrotta e si confronta con gli hash dei chunks di quella parte che sono contenuti nel pacchetto di verifica
- Il client conserva tutti i chunks il cui hash corrisponde all'hash contenuto nel pacchetto di verifica, mentre richiede di nuovo i chunks per cui non c'è corrispondenza

AICH:ADVANCED INTELLIGENT CORRUPTION HANDLING

- ogni volta che eMule rileva una corruzione in una parte del file chiede un pacchetto di recupero da un client casuale con un AICH completo
- Root Hash = valore contenuto nella root dell'hash tree
- si suppone che il client possieda un root hash sicuro
- il pacchetto di recupero contiene
  - tutti gli hash dei chunks della parte corrotta (53 block hashes)
  - alcuni hash di verifica dell'intero albero hash (numero hash di verifica x tale che 2^x >= numero di parti del file)
- il numero di hash di verifica dipende dal numero di parti del file
- quando riceve un pacchetto di recupero, il client verifica che il pacchetto di recupero non sia a sua volta corrotto utilizzando gli hash di verifica e confrontandoli con il proprio root hash
- combina gli hash dei blocchi, risale l'albero combianando i risultati con gli hash di verifica,fino ad ottenere il root hash
- Confronta il root hash con quello in suo possesso, se il root hash coincide,il pacchetto è integro

ADVANCED INTELLIGENT CORRUPTION HANDLING

- Problema dell'ICH:
  - quando l'errore è localizzato alla fine di una parte del file, ad esempio a 8,9 MB, l'ICH riscarica tutti i chunks delle parte
  - questo problema è dovuto al fatto che la porzione minima su cui viene calcolato l'hash è la parte di file e non il chunk
- AICH= Advanced Intelligent Corruption Handling:sistema introdotto dalla versione 0.44
- AICH Utilizza un hashtree
- Hashtree = set di hash "costruiti" a partire da blocchi di 180KB (chunk) e raggruppati in una struttura "ad albero".
- Usando gli hash contenuti in esso controlla tutti i blocchi da 180KB e identifica quelli corrotti, chiedendo lo scaricamento solo di essi.

ADVANCED INTELLIGENT CORRUPTION HANDLING

- L' albero di hash di tutti i file condivisi viene memorizzato nella cartella emule\config.
- Se eMule scarica un file e rileva una parte corrotta, chiede un "pacchetto di recupero" da un client che sia in possesso di un set hash A.I.C.H. completo di quel file
- Se in un chunk un solo byte è corrotto, eMule conserva tutti i blocchi da 180KB non corrotti, ad eccezione dell'unico blocco da 180KB che contiene il byte corrotto.
- Ad esempio, nel caso in cui il chunk corrotto si trovi in fondo al file, viene riscaricato solo l'ultimo chunk da 9.1 a 9.28 MB invece di tutti e 9.2 MB.
- Vantaggio: tramite l'A.I.C.H. è possibile recuperare parti di file senza dover essere costretti a riscaricare blocchi non corrotte.
- Se se non è possibile recuperare un file tramite AICH si utilizza ICH

AICH:ADVANCED INTELLIGENT CORRUPTION HANDLING

- Problema:diffusione di root hash corretti
- Reperiti mediante i link e2k
- Ogni client richiede agli altri client il root hash di un file
  - Se almeno altri 10 client mi inviano lo stesso Root Hash e questo Root Hash è uguale ad almeno il 92% or più dei Root Hash ricevuti,allora quelroot hash viene considerato affidabile

COMUNICAZIONE UDP CLIENT/CLIENT

- Un client *C* invia periodicamente pacchetti UDP ai clients da cui sta tentando di scaricare i files
- Il pacchetto UDP viene inviato per conoscere lo stato della richiesta di download effettuata da *C* presso *C1*
- Possibili risposte
  - La posizione di *C* nella coda di upload di *C1*
  - La coda di upload di *C1* è piena
  - *C1* non possiede il file richiesto da *C*