LIVELLO 3: TRANSPORT LAYER

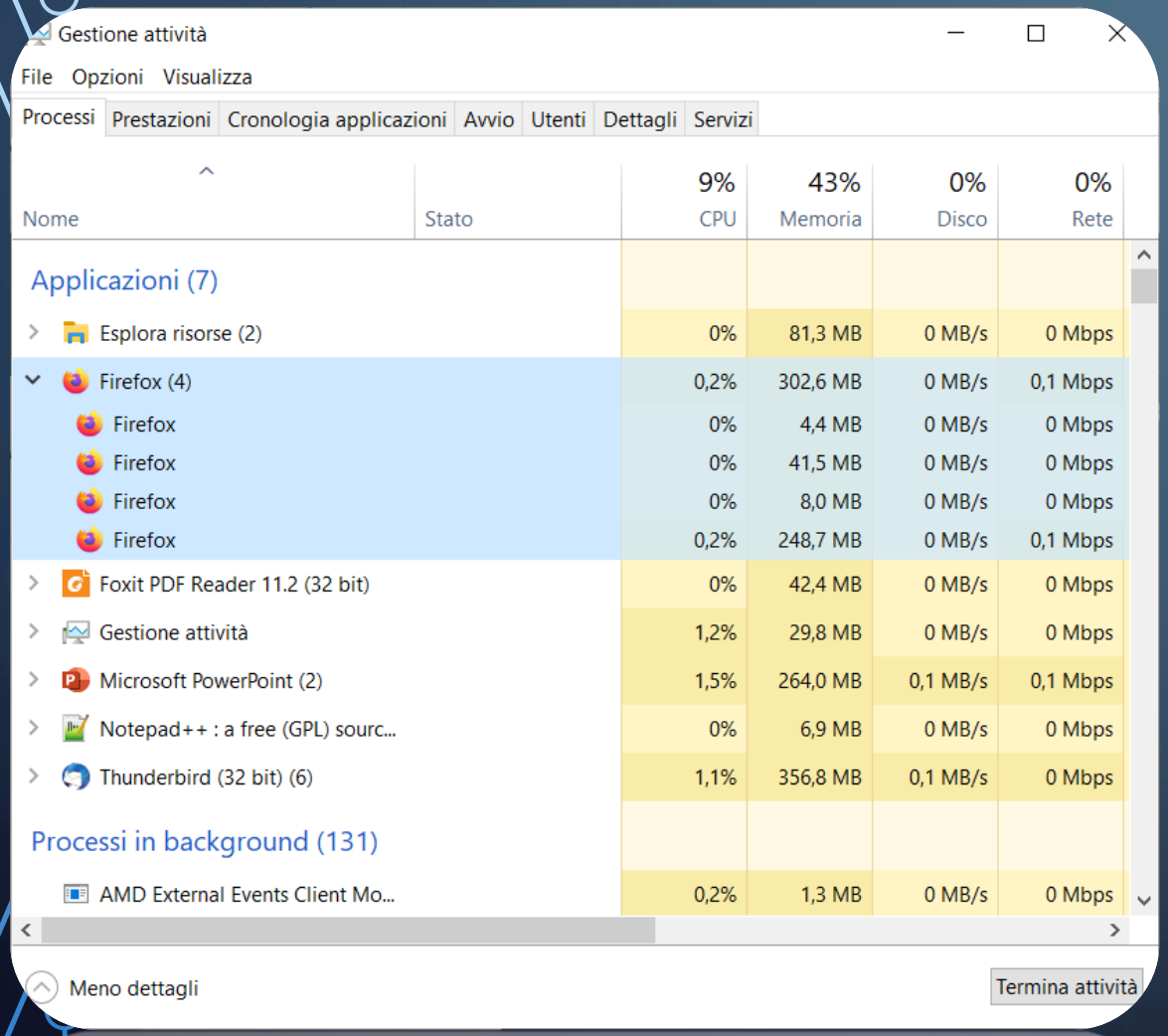
MODULO 11.3.4: BUS DI CAMPO

(VER. 2023)

FONTI:

- E. Baldino, R. Rondano, A. Spano, C. Iacobelli, «Internetworkong sistemi e reti», Juvenilia
- https://it.wikipedia.org/
- https://www.disi.unige.it/person/CostaG/smid_03/dispense_et_al/informatica_generale/reti_internet/2_internet.ppt

LIVELLO 3: TRANSPORT



I PROBLEMI DAL LATO DEL TRASMETTITORE:

Se diamo uno sguardo al gestore attività di Windows ci accorgiamo che possono esserci più istanze di uno stesso programma.

Per esempio Firefox ha 4 finestre aperte.

Ma se io seleziono una di quelle finestre e in quella finestra digito per esempio *www.google.com* e poi in un'altra digito *www.interno.gov.it* come fanno le due pagine ad aprirsi nella finestra giusta visto che i dati provengono tutti dalla stessa scheda di rete?

LIVELLO 3: TRANSPORT

I PROBLEMI DAL LATO DEL DESTINATARIO:

Nell'host di destinazione ci possono essere più utenti e più processi attivi contemporaneamente, alcuni in stato di run altri in sleep quindi quando un *segmento* (cioè un pacchetto del *transport layer*) viene ricevuto, il destinatario potrebbe non essere il processo o l'utente in esecuzione in quel momento.

L'header dal segmento inoltre non contiene alcuna informazione che permetta l'individuazione dell'applicazione o dell'utente destinatario del messaggio sull'host ricevente.

Per risolvere questo problema, ogni DTE nella rete è individuata da un indirizzo univoco (chiamato indirizzo IP) e poiché molteplici possono essere i servizi offerti dal sistema e molte le connessioni contemporanee è necessario un metodo per separare i singoli flussi di dati ed indirizzarli verso il corretto programma di gestione.

LE PORTE

Il problema della **ricezione** è stato risolto introducendo negli host dei **punti di accesso** ai quali consegnare i pacchetti che arrivano dalla rete chiamati anche **porte** o **socket** (una specie di casella postale). Saranno poi le applicazioni via via in esecuzione nel DTE ricevente a verificare se sulle porte ci sono segmenti a loro destinati.

Un destinatario è quindi identificato dalla coppia *IP:porta* .Quindi un host mittente non invierà una richiesta ad un programma, ma ad una coppia *IP:porta* (per esempio 200.198.0.1:80).

Per esempio un host mittente che deve contattare un web server di cui conosce l'indirizzo IP invierà i sui segmenti alla porta HTTP (80)

LE PORTE

- Ogni porta è identificata da un numero intero positivo codificato in 16 bit (quindi da 0 a 65535)
- I numeri di porta sono assegnati a livello internazionale da IANA
- Le porte sono le stesse sia per il TCP che per l'UDP, i due principali protocolli di tipo transport (che saranno visti qui in seguito), quindi ci possono essere teoricamente fino a 131.072 porte diverse.

LE PORTE

I numeri di porta sono classificabili in tre gruppi:

- **Porte conosciute:** sono assegnate dall'Internet Assigned Numbers Authority (IANA), sono quelle inferiori a 1024 (0-1023) e sono generalmente utilizzate a livello di sistema operativo o di processi di sistema. In genere rimangono in ascolto su queste porte applicazioni con funzioni di server
  - Alcuni esempi possono essere le applicazioni che utilizzino i protocolli FTP (21), SSH (22), TELNET (23), SMTP (25) e HTTP (80)

- **Porte registrate:** (1024-49151) sono utilizzate da applicazioni molto diffuse
  - Alcuni esempi sono Microsooft-SQL-Server (1433), MQTT (1883)

- **Porte private o dinamiche:** (49152-65535) sono liberamente utilizzabili da tutte le applicazioni utente, salvo l'occupazione contemporanea da parte di qualche altro processo

LE PORTE

Il problema in **trasmissione** è risolto allo stesso modo. L'application layer del trasmettitore quando fa una richiesta si identifica anch'esso tramite una porta "scelta casualmente" e i segmenti di risposta saranno indirizzati a questa

I pacchetti appartenenti ad una connessione del *transport layer* saranno quindi identificati dalla quadrupla

[<indirizzo IP sorgente>, <indirizzo IP destinazione>, <porta sorgente>, <porta destinazione>]

I pacchetti nella direzione opposta avranno ovviamente sorgente e destinazione scambiati (se non si sa esattamente cos'è un indirizzo IP, è sufficiente immaginarlo come l'indirizzo unico a livello mondiale del terminale).

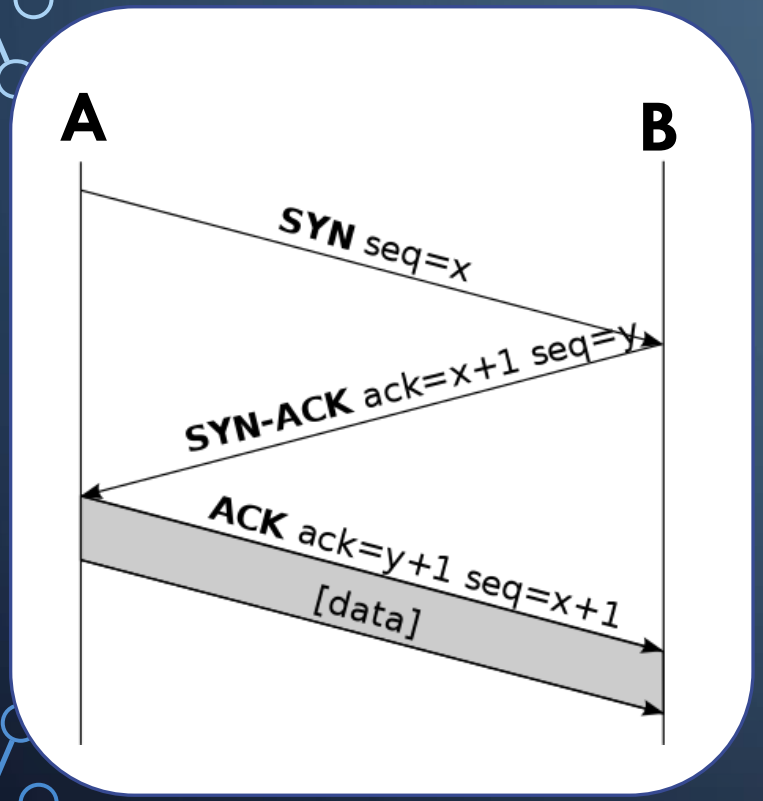
OSSERVAZIONE: Mentre la porta di destinazione è l'identificativo univoco del processo applicativo, la porta di sorgente è assegnata casualmente in maniera tale da identificare univocamente la connessione da parte del mittente col destinatario, questo perché ci possono essere più terminali all'interno di una rete locale che possono comunicare contemporaneamente con lo stesso server esterno

PROTOCOLLO TCP

- Il TCP è un protocollo di livello transport definito nella RFC 793 orientato alla connessione, ovvero prima di poter trasmettere dati deve stabilire la comunicazione, negoziando una connessione tra mittente e destinatario, che rimane attiva anche in assenza di scambio di dati e viene esplicitamente chiusa quando non più necessaria
- Garantisce la consegna a destinazione dei *segmenti* (cioè i pacchetti del *transport layer*) attraverso il meccanismo degli acknowledgements (su timeout)
  - il flusso di byte prodotto dall'applicazione sull'host mittente, viene frazionato in blocchi, detti segmenti, e consegnato al TCP sull'host destinatario che lo passerà all'applicativo indicato dal numero di porta del destinatario nell'header del segmento (es.: applicativo HTTP, porta 80), se il pacchetto viene ricevuto correttamente l'host mittente viene avvertito con un ACK (acknowledge). Nel caso l'ACK non arrivi entro un certo tempo (timeout) il pacchetto (che si chiama segment) viene ritrasmesso
- Garantisce che i dati trasmessi, se giungono a destinazione, lo facciano in ordine e una volta sola ("at most once")
  - il protocollo fornisce ai livelli superiori un servizio equivalente ad una connessione fisica diretta che trasporta un flusso di byte

TCP offre funzionalità di controllo di errore sui segment (pacchetti) pervenuti grazie al campo checksum contenuto nella sua PDU

NEGOZIAZIONE DI UNA CONNESSIONE TCP



1. Un client invia un pacchetto dati SYN su una rete IP a un server sulla stessa rete o a una rete esterna. L'obiettivo di questo pacchetto è di chiedere se il server è disponibile per nuove connessioni
2. il server di destinazione deve disporre di porte aperte in grado di accettare e avviare nuove connessioni. Quando il server riceve il pacchetto SYN dal nodo client, risponde e restituisce una ricevuta di conferma, il pacchetto ACK o SYN/ACK
3. il client riceve il SYN/ACK dal server e risponde con un pacchetto ACK
4. al termine di questo processo, viene creata la connessione e l'host e il server sono in grado di comunicare. Alla fine della trasmissione la connessione viene chiusa

PROTOCOLLO UDP

- UDP è un protocollo di tipo connectionless
  - non gestisce il riordinamento dei pacchetti né la ritrasmissione di quelli persi, ed è perciò generalmente considerato di minore affidabilità
- È molto rapido (non c'è latenza per riordino e ritrasmissione) ed efficiente per le applicazioni «leggere» o time-sensitive, non c'è ritardo per la fase di set-up della connessione
  - è usato spesso per la trasmissione di informazioni audio-video real-time come nel caso delle trasmissioni Voip
  - vista la snellezza, un server può supportare molti più client attivi

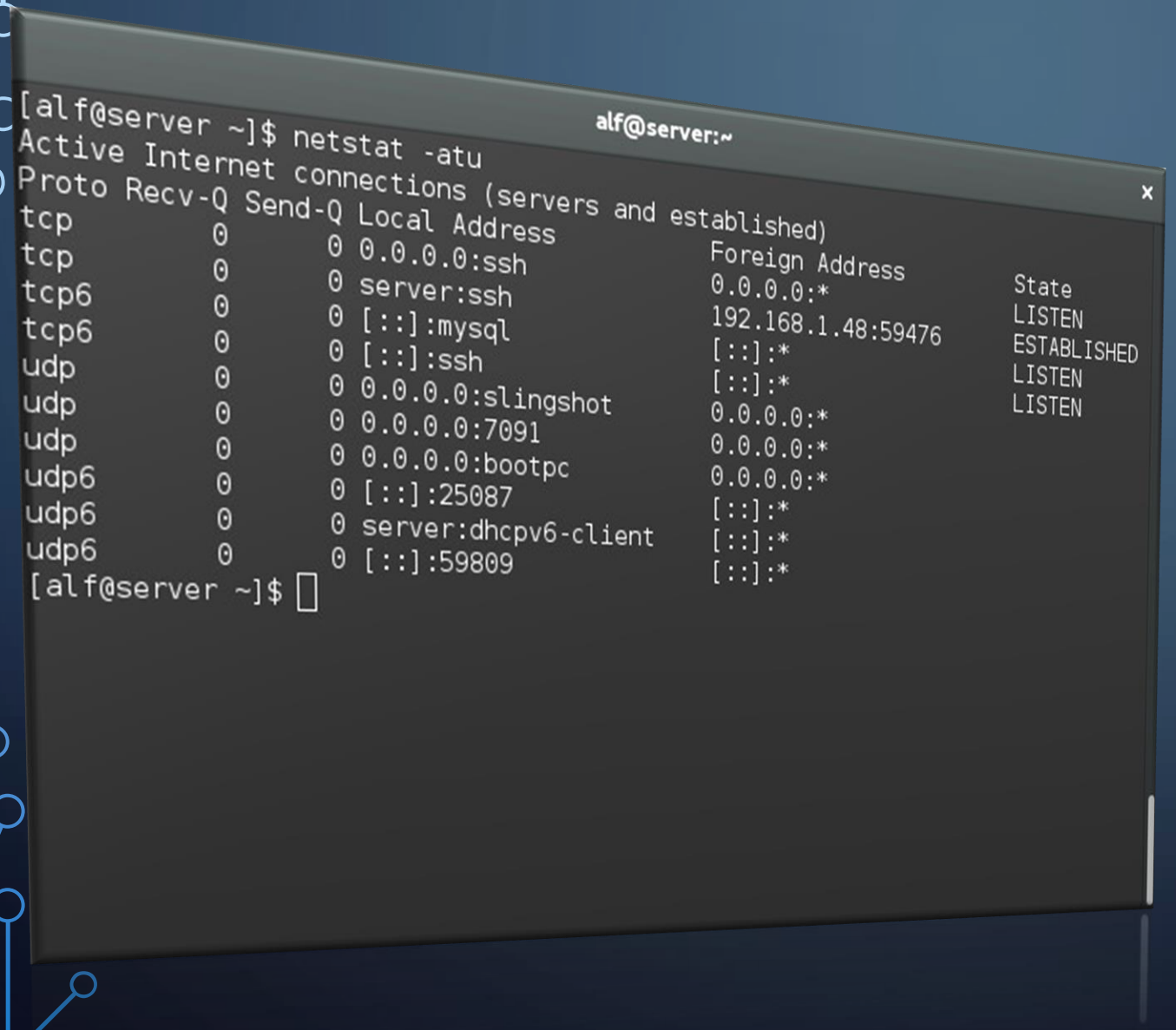
- L'UDP fornisce soltanto i servizi basilari del livello di trasporto, ovvero:
  - multiplazione delle connessioni, ottenuta attraverso il meccanismo di assegnazione delle porte
  - verifica degli errori (integrità dei dati) mediante una checksum, inserita in un campo dell'intestazione (header) del pacchetto

CONFRONTO TCP E UDP

Le principali differenze tra TCP e UDP (User Datagram Protocol) sono:

- TCP è un protocollo orientato alla connessione
  - pertanto, per stabilire, mantenere e chiudere una connessione, è necessario inviare segmenti di servizio i quali aumentano l'overhead di comunicazione. Al contrario, UDP è senza connessione ed invia solo i datagrammi richiesti dal livello applicativo
- UDP non offre nessuna garanzia sull'affidabilità della comunicazione
  - ovvero sull'effettivo arrivo dei segmenti, né sul loro ordine in sequenza in arrivo; al contrario di TCP
- **l'oggetto della comunicazione di TCP è un flusso di byte, mentre quello di UDP sono singoli i segmenti (pacchetti)**
  - L'utilizzo del protocollo TCP rispetto a UDP è, in generale, preferito quando è necessario avere garanzie sulla consegna dei dati o sull'ordine di arrivo dei vari segmenti
  - Al contrario UDP viene principalmente usato quando l'interazione tra i due host è idempotente o nel caso si abbiano forti vincoli sulla velocità e l'economia di risorse della rete (es. streaming in tempo reale, videogiochi multiplayer)

IDENTIFICARE LE CONNESSIONI TCP ATTIVE



Il comando **netstat**

- Visualizza le connessioni TCP attive, le porte su cui il computer è in ascolto, le statistiche Ethernet, la tabella di routing IP, le statistiche IPv4 (per i protocolli IP, ICMP, TCP e UDP) e le statistiche IPv6 (per i protocolli IPv6, ICMPv6, TCP su IPv6 e UDP su IPv6)
- Usato senza parametri visualizza le connessioni TCP attive