Esame di **Progettazione di servizi web e reti di calcolatori (01NBE)**
Corsi di Laurea in Ing. Gestionale e dell'Organizzazione d'Impresa
Prova scritta di teoria (7/7/2022)

> NOTA
> Le tracce delle soluzioni fornite in questo testo sono da considerarsi solo come un aiuto per comprendere i principali punti da toccare nel risolvere gli esercizi proposti ma non sono né esaustive né presentate in forma adeguata per l'elaborato da consegnarsi in sede d'esame.
> In particolare per molti esercizi le soluzione è volutamente schematica e ci si attende che il candidato spieghi adeguatamente i singoli punti, per dimostrare reale comprensione dell'argomento invece che semplice capacità mnemonica di ricordare i punti elencati nelle slide (o in queste tracce di soluzione).

Esercizio 1 ***(punti: 6)***

Nell'ambito della posta elettronica Internet, spiegare cosa sono rispettivamente un MUA, MSA e MS. Per ciascuno di essi indicare anche quali protocolli applicativi usano per il traffico in ingresso (ricezione) ed in uscita (spedizione).

Traccia di una possibile risposta

MUA (Mail User Agent) è l'applicazione con cui un utente può inviare (tramite un MSA), ricevere (tramite un MS) e più in generale gestire la propria posta elettronica.
spedizione: SMTP
ricezione: POP o IMAP

MSA (Mail Submissione Agent) è il server che accetta di ricevere posta (solo dai MUA di uno specifico gruppo di utenti) per poi trasmetterla a destinazione una catena di MTA (ossia un MHS).
spedizione: SMTP
ricezione:SMTP

MS (Mail Store) è il server che riceve ed immagazzina la posta destinata ad uno specifico gruppo di utenti.
spedizione: POP o IMAP
ricezione: SMTP

Esercizio 2 ***(punti: 6)***

Nelle architettura web tra Origin Server e User Agent possono essere presenti Proxy e Gateway. Spiegare cosa sono questi due componenti aggiuntivi, quali funzionalità svolgono, dove vengono solitamente posizionati (disegnare il relativo schema) e quali sono i benefici che apportano al sistema.

Traccia di una possibile risposta

Proxy = elemento bifronte, agisce come un server rispetto a UA e come UA rispetto a un server; riceve richieste HTTP dall'UA e se la risorsa richiesta è già presente nella sua cache la fornisce direttamente all'UA altrimenti effettua la richiesta all'OS, salva la risposta nella propria cache e la fornisce anche all'UA.
Proxy è solitamente posizionato vicino a UA (es. presso ISP degli utenti casalinghi, nella stessa Intranet degli utenti aziendali).
Proxy velocizza la risposta alle richieste HTTP (se la risposta è già presente nella sua cache) diminuendo così il traffico verso Internet (utile per aziende o ISP con poca banda verso Internet).

Gateway = elemento bifronte, agisce come un server rispetto a UA e come UA rispetto ad un OS; riceve richieste HTTP da un client (un UA o un Proxy) per uno specifico OS e se la risorsa richiesta è già presente nella sua cache (es. pagine statiche o immagini presenti sullo specifico OS) la fornisce direttamente al client, altrimenti effettua la richiesta all'OS, salva la risposta nella propria cache (solo se ha senso, ad esempio non per pagine dinamiche) e la fornisce anche al client.

Gateway è solitamente posizionato davanti a OS (es. nella stessa rete locale), potrebbe anche fungere da Reverse Proxy e/o Load Balancer.
Gateway velocizza la risposta alle richieste HTTP (se la risposta è già presente nella sua cache) diminuendo così il traffico verso l'OS ed il suo lavoro (utile per diminuire il carico sul server e velocizzare le risposte ai client).

Schema = slide 11 del set ProgWeb.

Esercizio 3 ***(punti: 6)***

Con riferimento a HTTP/1.1, spiegare che cosa è il *pipelining*, quali benefici apporta e quali problemi può generare.

Traccia di una possibile risposta

Normalmente un client che invia una richiesta HTTP ad un server deve aspettare la relativa risposta prima di effettuare una nuova richiesta. Col pipelining, un client può inviare più richieste senza aspettare la risposta per ciascuna di esse. Le risposte verranno inviate dal server nell'ordine corrispondente a quello delle richieste.

Il maggior vantaggio consiste nella velocità di risposta perché le richieste sono accorpate tutte insieme in un numero di segmenti TCP inferiore e non occorre attendere la risposta per ciascun richiesta.

Il principale svantaggio è che in caso di errore (di rete o applicativo) il client potrebbe dover rimandare tutte le richieste, anche quelle che hanno già ricevuto risposta.

Esercizio 4 ***(punti: 4)***

Supponendo di operare su un PC Windows correttamente configurato, scrivere i comandi da eseguire in una finestra di comando per:

- identificare il server da contattare per inviare posta all'utente `mario.rossi@gmail.com`
- identificare il nameserver primario del dominio a cui appartiene il server web `www.yahoo.com`
- trovare l'indirizzo IPv4 del server `www.polito.it`
- trovare il FQDN del nodo di rete avente indirizzo IPv4 `92.122.34.235`

Traccia di una possibile risposta

```
nslookup -q=MX gmail.com.

nslookup -q=SOA yahoo.com.

nslookup -q=A www.polito.it.

nslookup -q=PTR 235.34.122.92.in-addr.arpa.
```

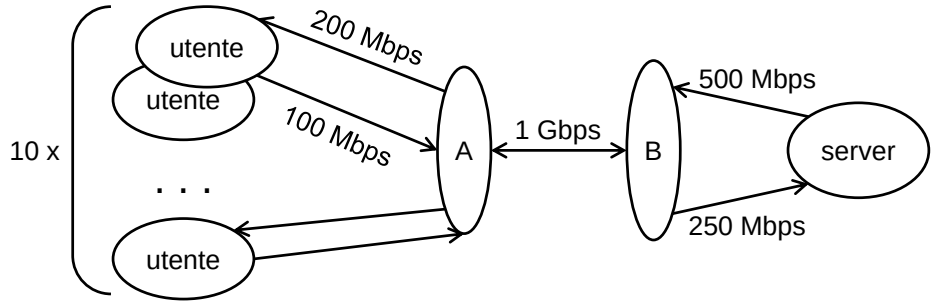
Esercizio 5 ***(punti: 5)***

Dieci utenti sono connessi alla stessa centrale telefonica (A), ciascuno tramite una linea ADSL da 200 Mbps in download e 100 Mbps in upload e devono comunicare con un server attestato presso una diversa centrale (B) con una linea ADSL da 500 Mbps in upload e 250 Mbps in download. Le due centrali sono collegate tra loro da una linea a 1 Gbps.

Tutti gli utenti devono prima scaricare dal server un file da 500 MB e dopo inviare al server un file da 100 MB. Sapendo tutti gli utenti operano simultaneamente, calcolare il tempo minimo dopo il quale tutte le operazioni sono concluse.

Traccia di una possibile risposta

Schema del collegamento:



I dieci utenti quando effettuano il download simultaneamente generano un potenziale traffico di $200\,\text{Mbps}$ *ciascuno, quindi un totale di* $200\,\text{Mbps} \cdot 10 = 2\,\text{Gbps}$. *Calcoliamo il collo di bottiglia per il download (ossia per il traffico nella direzione dal server verso i client):*

$$\min(2,1,0.5)\,\text{Gbps} = 500\,\text{Mbps}$$

Ogni utente deve scaricare un file da $500\,\text{MB}$ *quindi il tempo totale affinché tutti gli utenti abbiano terminato è dato da:*

$$10 \cdot 500\,\text{MB} \cdot 8/500\,\text{Mbps} = 80\,\text{s}$$

Dopo aver terminato il download, gli utenti effettuano simultaneamente gli upload, generando un traffico totale di $100\,\text{Mbps} \cdot 10 = 1\,\text{Gbps}$. *Calcoliamo il collo di bottiglia per la fase di upload (ossia per il traffico nella direzione dai client verso il server):*

$$\min(1,1,0.25)\,\text{Gbps} = 250\,\text{Mbps}$$

Ogni utente deve caricare un file da $100\,\text{MB}$ *quindi il tempo totale affinché tutti gli utenti abbiano terminato è dato da:*

$$10 \cdot 100\,\text{MB} \cdot 8/250\,\text{Mbps} = 32\,\text{s}$$

Quindi il tempo tale per il completamento di entrambe le fasi è pari a:

$$80\,\text{s} + 32\,\text{s} = 132\,\text{s}$$

Esercizio 6 *(punti: 5)*

Spiegare che cosa è in generale un *firewall*, poi illustrare le tipologie *packet-filter* e *application-gateway* indicandone vantaggi e svantaggi.

Traccia di una possibile risposta

firewall = punto di controllo del traffico tra una rete "esterna" (a bassa sicurezza, pericolosa) ed una rete "interna" (ad alta sicurezza, da proteggere); decide quale traffico lasciar passare e quale bloccare in base alla politica di sicurezza decisa dal responsabile della sicurezza

packet-filter = ispeziona i pacchetti IP singolarmente, decide se lasciarli passare o bloccarli in base al contenuto degli header L3 e L4 (es. indirizzo IP mittente o destinatario, protocollo L4 e porta)
vantaggi = molto veloce
svantaggi = non ispeziona il contenuto dei pacchetti e quindi non offre una grande sicurezza

application-gateway = agisce come un proxy, esaminando le richieste e le risposte a livello applicativo (es. HTTP, SMTP) per idnetificare e bloccare contenuto pericoloso (es. comandi di cancellazione, pagine web contenenti malware)
vantaggi = controllo del traffico in base al contenuto, quindi molto accurato, grande sicurezza
svantaggi = molto lento perché - fungendo da proxy - deve ri-assemblare il traffico a livello applicativo prima di poterlo esaminare