

# ESERCIZI RETI DI CALCOLATORI

- SITO PER ULTERIORI ESERCIZI: [\*Reti di Calcolatori - University of Venice \(24/25\) | INGInious\*](#)
- ESECIZI VISTI A LEZIONE/APPARSI NEGLI ESAMI DIVISI PER ARGOMENTI:

## Sommario

<b>PHYSICAL LAYER (NYQUIST, SHANON)</b>	2
Domande Teoria a Crocette .....	2
Esercizi .....	5
<b>DATALINK LAYER (FRAMING/BIT STUFFING, ACK, BIT PARITA', CHECKSUM)</b>	8
Domande Teoria a Crocette .....	8
Esercizi .....	10
<b>RELIABLE TRANSPORT (ABP, GO-BACK-N, SELECTIVE REPEAT)</b>	12
Esercizi .....	12
<b>DATALINK LAYER – SHARING RESOURCES (TDMA, S-ALOHA)</b>	17
Domande Teoria a Crocette .....	17
Esercizi .....	18
Domande Aperte .....	20
<b>NETWORK LAYER (FRAGMENTATION, HOT POTATO, ROUTING TABLE, DISTANCE VECTOR)</b>	21
Domande Teoria a Crocette .....	21
Esercizi .....	22
Domande Aperte .....	31
<b>TRANSPORT LAYER</b>	33
Esercizi .....	33
<b>SECURITY (OTP, DIEFFE-HELLMAN, RSA)</b>	35
Domande Teoria a Crocette .....	35
Esercizi .....	39
<b>DNS</b>	46
Domande Teoria a Crocette .....	46
Domande Aperte .....	46
<b>EMAIL</b>	49
Domande Aperte .....	49
<b>WWW (HTTP)</b>	49
Domande Aperte .....	49
<b>TRANSPORT LAYER (UDP, TCP)</b>	50
Domande Teoria a Crocette .....	50

Esercizi .....	52
Domande Aperte .....	54
<b>NETWORK LAYER (IPV4, IPV6, ICMP, OSPF).....</b>	<b>55</b>
Domande Teoria a Crocette .....	55
Esercizi .....	55
Domande Aperte .....	60
<b>BGP E INTERDOMAIN ROUTING.....</b>	<b>61</b>
Esercizi .....	61
Domande Aperte .....	61
<b>DATALINK LAYER – ETHERNET E WIFI (SPANNING TREE PROTOCOL, ARP, DHCP) .....</b>	<b>62</b>
Esercizi .....	63
Domande Aperte .....	68
<b>TODO.....</b>	<b>69</b>

## PHYSICAL LAYER (NYQUIST, SHANON)

### Domande Teoria a Crocette

Il bit-rate rappresenta la velocità di trasmissione dati di una comunicazione e può essere influenzato da vari fattori. Descrivi alcuni metodi per aumentare il bit-rate di una comunicazione basata su modulazione FM, partendo dall'equazione di Nyquist e spiegando per ciascuno di essi i potenziali vantaggi e svantaggi.

Come incrementare il bit-rate? 1) incrementare la frequenza 2) ridurre il Symbol Time 3) codificare più di un bit per ogni simbolo.

Tutte queste tecniche riducono il SNR (Signal Noise Ratio).

Idealmente possiamo codificare un numero di bit arbitrario, sempre se il ricevitore è abbastanza abile a riconoscere la differenza tra le sequenze che mandiamo.

Per incrementare il bit-rate dobbiamo incrementare la larghezza di banda. Larghezza di banda: l'intervallo tra più bassa e la più alta frequenza che usi per le comunicazioni.

- Aumentare la larghezza di banda (bandwidth): Questo è uno dei metodi principali derivati dai teoremi di Nyquist, secondo il quale la capacità di un canale è direttamente proporzionale alla larghezza di banda disponibile. Tuttavia, l'aumento della larghezza di banda richiede porzioni di spettro più ampie, che potrebbero non essere sempre disponibili
- Utilizzare più livelli di modulazione: Vengono trasmessi più bit per simbolo, usando un numero maggiore di simboli. Per ottenerlo si possono percorrere sue strade: si mantiene la distanza in frequenza tra un simbolo e l'altro, e quindi, si aumenta la banda, oppure si mantiene la banda ma i simboli saranno più vicini in frequenza, e si avrà bisogno di un ricevitore in grado di distinguere accuratamente tra diversi livelli di segnale

Il Signal to Noise Ratio (SNR) è un parametro fondamentale nelle comunicazioni. Come viene definito e perché è così importante per la qualità di una trasmissione?

Scegli un'alternativa:

- a. Il rapporto tra la potenza del segnale ricevuto e la potenza del rumore: un SNR più alto indica una migliore distinzione tra segnale e rumore, migliorando la capacità di decodifica del segnale
- b. La differenza in frequenza tra il segnale inviato e il rumore di fondo; più grande è il valore del rapporto, più distanti sono le frequenze del segnale e del rumore
- c. Il rapporto tra larghezza di banda del canale e la potenza del segnale; un SNR più alto permette l'uso di una maggiore larghezza di banda
- d. La quantità di rumore generato dal ricevente; un SNR più alto indica che vengono generate meno interferenze durante la trasmissione dal ricevitore

**SNR (signal-noise ratio) :** Se la potenza ricevuta è superiore alla sensibilità del ricevitore, la capacità di decodificare correttamente i dati dipende dal rapporto segnale-rumore (SNR): il rapporto tra l'energia del segnale e l'energia del rumore. Più alto è il SNR, meglio è. Risposta a.

Il SNR è il rapporto tra la potenza del segnale ricevuto e quella del rumore. Un SNR elevato è cruciale perché garantisce che il segnale possa essere facilmente distinto dalle interferenze, migliorando la qualità della trasmissione e riducendo il rischio di errori nella codifica dei dati.

La risposta corretta è: Il rapporto tra la potenza del segnale ricevuto e la potenza del rumore: un SNR più alto indica una migliore distinzione tra segnale e rumore, migliorando la capacità di decodifica del segnale

Quale dei seguenti fattori incide maggiormente sulla capacità di un canale di comunicazione di trasmettere dati ad alta velocità secondo il teorema di Shannon?

Scegli un'alternativa:

- a. Il tipo di modulazione del segnale utilizzata
- b. La velocità di propagazione del segnale nel mezzo fisico
- c. La temperatura dei componenti elettronici
- d. Il rapporto segnale-rumore (SNR) e la larghezza di banda disponibile.

Il teorema di Shannon afferma che la capacità massima di un canale dipende dalla larghezza di banda e dal rapporto segnale-rumore (SNR), che determinano la quantità massima di dati trasmissibili senza errori.

La risposta corretta è: Il rapporto segnale-rumore (SNR) e la larghezza di banda disponibile.

Come si definisce la sensitivity di un ricevitore

Scegli un'alternativa:

- a. La quantità di energia minima che deve arrivare al ricevitore perché possa accorgersi che un segnale sta arrivando
- b. Il rapporto segnale/rumore minimo che deve essere misurato al ricevitore per decodificare correttamente un segnale
- c. La quantità di energia massima che si può attenuare durante la propagazione del segnale, affinché il ricevitore possa ricevere il segnale
- d. La quantità di energia minima per definire la differenza tra un bit impostato a zero e uno impostato a uno

a. La sensibilità di una radio è la quantità di energia che si deve ricevere perché la radio faccia la detection di un segnale in arrivo.

Cosa è la modulazione FM digitale

Scegli un'alternativa:

- a. Considerato un segnale analogico detto portante (carrier), ed un segnale digitale detto modulante, la FM varia la frequenza della modulante aumentandola o riducendola sulla base del valore binario della portante
- b. Considerato un segnale analogico detto portante (carrier), ed un segnale digitale detto modulante, la FM varia l'ampiezza della modulante aumentandola o riducendola sulla base del valore binario della portante.
- c. Considerato un segnale analogico detto portante (carrier), ed un segnale digitale detto modulante, la FM varia la frequenza della portante aumentandola o riducendola sulla base del valore binario della modulante
- d. Considerato un segnale digitale detto portante (carrier), ed un segnale digitale detto modulante, la FM varia l'ampiezza e la frequenza della modulante aumentandola o riducendola sulla base del valore binario della portante.

c. Una modulazione di frequenza (FM) fa variare la frequenza di un segnale portante, sulla base del valore di un segnale modulante, che può essere un segnale digitale.

È possibile che un trasmettitore invii dati ad una velocità più alta di quella stabilita dall'equazione di Shannon per quel canale?

Scegli un'alternativa:

- a. No, l'equazione di Shannon identifica il limite massimo che può essere usato da un trasmettitore, dato un certo SNR in ricezione.
  - b. Si, il trasmettitore può usare una modulazione che invia più dati della capacità consentita da Shannon. La comunicazione non conterrà errori se la velocità è comunque minore del limite posto dall'equazione di Nyquist.
  - c. È possibile solo nei casi in cui il ricevitore usa una modulazione migliore di quella del trasmettitore.
  - d. Si, il trasmettitore può usare una modulazione che invia più dati della capacità consentita da Shannon, ma la ricezione sarà affetta da errori.
- d. L'equazione di Shannon limita il bit-rate massimo dato un certo SNR sul ricevitore, ma non impedisce al trasmettitore di inviare più dati, che però il ricevitore non riceverà tutti correttamente.

## Esercizi

### Exercise

Un router Wireless utilizza frequenze che vanno da 2401 MHz a 2441 MHz.

Supponi che codifichi 2 bit per ogni simbolo.

#### Qual'è il massimo bit-rate?

1 - **trovare la larghezza di banda**, così da trovare il numero massimo di simboli trasmisibili contemporaneamente su frequenze ortogonali.

Determinare la differenza tra le due frequenze :

$$B = 2.441.000.000 - 2.401.000.000 = 40.000.000 \text{ Hz (ossia } 40 \text{ MHz)}.$$

#### 2 - determinare il numero di livelli :

Il router codifica 2 bit per ogni simbolo : per codificare 2 bit per simbolo ci sono :

$$M = 2^2 = 4 \text{ livelli.}$$

#### 3 - infine applicare la formula di Nyquist :

$$C = 2B\log_2(M).$$

$$> \log_2(4) = 2.$$

$$C = 2 * 40.000.000 * 2 = 160\text{Mb/s (Mega bit per secondo).}$$

#### 4 - risultato :

Il massimo bit-rate è **160Mb/s**.

## Exercise

Una comunicazione ADSL (Asymmetric Digital Subscriber Line) cablata (wired) può essere descritta dai seguenti dati/numeri:

- Potenza di trasmissione: 100 mW ( $mW = 10^{-3}$  Watt).
- Rumore: 0,0001 mW.
- Attenuazione: la potenza si divide per 25 ogni km.
- Larghezza di banda: supponi 2,2 MHz.

**1 - Qual è il bit-rate massimo raggiungibile a 2 km?**

**2 - Qual è la migliore modulazione che possiamo usare (il valore più alto di M)?**

**3 - Qual è il bit-rate massimo raggiungibile?**

**1 -** Per calcolare il massimo bit-rate raggiungibile a 2km, dobbiamo partire dalla potenza di trasmissione attuale (100 mW). Dobbiamo però tenere conto che durante il percorso il segnale si riduce per attenuazione. La potenza si divide per 25 ad ogni km.

**1.1 - Calcolo della potenza di trasmissione con attenuazione dopo 2 km :**

Al km 0 : potenza di trasmissione = 100 mW

Km 1 : potenza =  $100/25 = 4$  mW

Km 2 : potenza =  $100/25^2 = 0,16$  mW

**1.2 -** Calcoliamo il rapporto segnale-rumore (SNR) a 2km. Basta dividere la potenza di trasmissione (dopo aver tenuto conto della attenuazione) per il valore del rumore:

Km 0 :  $SNR = 100/0,0001 = 1.000.000$  mW = 1.000 Watt = 1 kW

Km 1 :  $4/0,0001 = 40.000$  mW = 40 W

Km 2 :  $0,16/0,0001 = 1600$  mW = 1,6 W

**1.3 -** Applicare la formula/legge di Shannon (s sta per Shannon) :

$> 2,2$  MHz = 2.200.000 Hz.

$> \log_2(1 + 1600) \approx 10,6448$

$C_s = 2.200.000$  Hz \* 10,6448 mW = 23.418.560 bps (bit per secondo) = **23 Mb/s**

**1.4 - risultato :**

Il massimo bit-rate raggiungibile a 2km è **23 Mb/s**.

**2 -** Consideriamo solo le modulazioni che sono potenze di 2.

Dobbiamo rispondere alla domanda, qual'è la più alta modulazione che possiamo usare?

Cioè :  $2B\log_2(M) \leq C_s$ ?

**2.1 -** Allora, dobbiamo invertire la formula di Nyquist :

$C_s = 2B\log_2(M) \rightarrow C_s / 2B = \log_2(M) \rightarrow 2^{C_s/2B} = M$ .

$C_s = 23$  Mb/s = 23.000.000 bps (arrotondato per difetto).

B = 2.200.000 Hz.

$C_s/2B = 23.000.000 / (2 * 2.200.000) = 23.000.000 / 4.400.000 = 5,22727273$ .

**2.2 -** Calcolo del numero di livelli con la potenza :

$M = 2^{5,22727273} \approx 32$ .

Abbiamo trovato la potenza che si avvicinava di più a  $2^{5,22727273}$ .

**2.2 - tuttavia questo riduce il bit-rate :**

$C = 2B\log_2(M) = 2 * 2.200.000 * 5 = 22$  Mbit/s <  $C_s$

**Attenzione!** Se aumentassimo M (esempio = 6), con la formula di Nyquist verrebbe fuori un bit-rate maggiore di quello calcolato con la formula di Shannon (26,4 > 23).

Conclusione : Non ha senso trasmettere a una velocità superiore al limite di Shannon.

**3 - Calcolare il massimo bit-rate raggiungibile :**

Per calcolare il massimo bit-rate, dobbiamo calcolarlo al km 0 anziché più avanti perché così facendo non avremo attenuazione che lo diminuisce.

**3.1 - potenza** di trasmissione al km 0 : **100 mW**.

**3.2 - calcolo** il rapporto segnale-rumore a 0 km : potenza segnale / rumore =  
 $100\text{mW} / 0,0001\text{mW} = 1.000.000$  mW = **SNR**.

**3.3 - bit-rate massimo** (al km 0 perché senza attenuazione) :  $C = B * \log_2(1 + SNR)$ .  
 $> \log_2(1 + 1.000.000) \approx 19,9316$ .

$C = 2.200.000$  Hz \* 19,9316 mW = **43.849.520 bps ==> 43,85 Mbps**.

**3.3 - risultato :**

Il bit-rate più alto che si possa raggiungere è al km 0 con **43,85 Mbps**.

Considera un canale di comunicazione in una rete wireless che abbia le seguenti caratteristiche:

- banda totale: 400 MHz
- potenza di trasmissione:  $P_t = 400 \text{ mW}$
- rumore al ricevitore: 0.002 mW
- la distanza tra trasmittitore e ricevitore è 100 m

Considera che la potenza trasmessa si riduce con il quadrato della distanza. A distanza  $x$  la potenza ricevuta è data da  $P(x) = P_t/x^2$

Quale è il più alto valore di livelli per simbolo ( $M$ ) utilizzabile in questa comunicazione? Nota, si chiede il valore di livelli, non il valore di bit per simbolo.

Scegli un'alternativa:

- a.  $M = 3$
- b.  $M = 2$
- c.  $M = 4$
- d.  $M = 16$

1. Capacità di Shannon:

$$C_S = B \log_2 \left( 1 + \frac{S}{N} \right) = B \log_2 \left( 1 + \frac{0.04}{0.002} \right) = B \log_2(1 + 20) = B \times 4.39$$

2. Capacità di Nyquist:

$$C_N = 2B \log_2(M)$$

3. Confronto: Impongo  $C_S \geq C_N$ :

$$B \times 4.39 \geq 2B \log_2(M)$$

Dividendo per  $2B$ :

$$\log_2(M) \leq \frac{4.39}{2} = 2.195$$

4. Calcolo di  $M$ :

$$M \leq 2^{2.195} \approx 4.57$$

$M$  deve essere una potenza di 2. L'intero minore è  $M = 4$ .

Risultato:

La risposta corretta è: a.  $M = 4$

## Domande Aperte

Riportare le equazioni di Shannon e di Nyquist, spiegare il loro utilizzo e la loro differenza. Seguire la seguente traccia:

- Introdurre l'equazione di Shannon, spiegare i suoi termini ed il suo significato. Specificare le unità di misura dei termini e del risultato.
- Introdurre l'equazione di Nyquist, spiegare i suoi termini ed il suo significato. Specificare le unità di misura dei termini e del risultato.
- Fare un esempio numerico in cui si usano le due equazioni per stimare il throughput massimo di un collegamento.

# DATALINK LAYER (FRAMING/BIT STUFFING, ACK, BIT PARITA', CHECKSUM)

## Domande Teoria a Crocette

Si consideri un messaggio di lunghezza  $M$  bit, su cui si usa una tecnica di bit-stuffing con un marker di lunghezza  $K$ . Dopo aver applicato bit-stuffing il messaggio diventa lungo  $M+H$ . Chiamiamo overhead relativo il valore  $(M+H)/M$ . Se si aumenta la lunghezza del marker  $K$ , quali sono le conseguenze?

Scegli un'alternativa:

- a. Per messaggi lunghi ( $M \gg K$ ) si riduce l'overhead relativo.
- b. Aumenta la probabilità di ridurre l'overhead relativo in generale.
- c. Aumenta la probabilità di aggiungere degli 'zero' *stuffed* nel corpo del messaggio.
- d. Per messaggi brevi ( $M$  simile a  $K$ ) l'overhead relativo diminuisce.

- a. Il marker introduce un overhead fisso all'inizio ed alla fine del frame, quindi aggiunge un overhead assoluto costante  $2K$ . Aumentare  $K$  spreca quindi più bit. D'altra parte aumentare  $K$  riduce la probabilità che il marker si trovi all'interno dei dati, e quindi che si debba aggiungere degli zeri nel corpo del frame. Probabilisticamente, per messaggi lunghi conta di più l'overhead variabile che non quello fisso, quindi l'overhead relativo diminuisce con  $K$ , mentre per quelli brevi aumenta.

Descrivi brevemente il concetto di checksum e la sua funzione all'interno di un pacchetto, poi fornisci la definizione di Internet Checksum e applicala alla stringa RETI (traducendo le lettere in numeri dalla tabella ASCII) e verificane la sua correttezza.

Il checksum è una tecnica utilizzata per verificare l'integrità dei dati trasmessi all'interno di un pacchetto.

DEFINIZIONE Internet Checksum (vedi slide lezione datalink-1)

Esercizio:

[+] Converto la stringa RETI in valori esadecimales:

R = 0x52, E = 0x45, T = 0x54, I = 0x49

[+] Converto i valori da esadecimale a bit e poi li divido in blocchi da 16 bit

R = 0x52 = 0101 0010

E = 0x45 = 0100 0101

T = 0x54 = 0101 0100

I = 0x49 = 0100 1001

Ottengo quindi due blocchi: [0101 0010 0100 0101] e [0101 0100 0100 1001]

[+] Sommiamo i blocchi

[0101 0010 0100 0101] +'

[0101 0100 0100 1001] =

1010 0110 1000 1110

[+] Applichiamo il complemento

(1010 0110 1000 1110)' = 0101 1001 0111 0001 = 0x5971

Il pacchetto sarà composto come 0x59 0x71 0x52 0x45 0x54 0x49

[+] Verifichiamo che sia tutto corretto facendo la somma di tutte le parole di 16 bit contenute nel pacchetto, compreso il checksum

[0101 1001 0111 0001] +'

[0101 0010 0100 0101] +'

[0101 0100 0100 1001] =

1111 1111 1111 1111

Avendo ottenuto un parola da 16 bit contenente solo 1, il messaggio è stato ricevuto correttamente senza errori.

Qual è il principale limite del Parity Bit quando viene utilizzato come meccanismo per il checksum?

Scegli un'alternativa:

- a. Non può rilevare errori se vi è un numero pari di bit corrotti nel pacchetto
- b. Può correggere solo errori di un bit, ma non quelli che coinvolgono più bit
- c. Non può rilevare errori in cui viene modificato un solo bit del pacchetto
- d. Richiede che ogni pacchetto sia composto da un numero fisso di bit

Il Parity Bit è efficace solo nel rilevare errori in cui un numero dispari di bit è stato modificato. Se un numero pari di bit è corrotto, questo meccanismo non riesce a rilevare l'errore.

La risposta corretta è la a: Non può rilevare errori se vi è un numero pari di bit corrotti nel pacchetto

Come il meccanismo di ACK contribuisce a prevenire il sovraccarico del buffer del ricevente? (vedi State Machine in datalink-1)

Scegli un'alternativa:

- a. Gli ACK non influenzano la velocità di trasmissione, ma solo la verifica degli errori
- b. Il mittente invia pacchetti successivi senza attendere l'ACK, aumentando la velocità di trasmissione
- c. Gli ACK permettono al mittente di adattare la velocità di trasmissione in base alla capacità del ricevitore di elaborare i dati
- d. Gli ACK vengono inviati immediatamente dopo la ricezione, senza alcun ritardo

Il meccanismo di ACK consente al ricevitore di confermare la corretta ricezione ed elaborazione dei pacchetti. Il ricevitore aspetta che lo strato superiore abbia terminato l'elaborazione del pacchetto e poi invia l'ACK. Se il ricevitore non riesce a elaborare i dati alla stessa velocità con cui vengono ricevuti, il mittente riceve ACK ad un rate minore di quello con cui genera nuovi pacchetti e

deve rallentare la trasmissione. In questo modo, si previene il sovraccarico del buffer del ricevitore.

La risposta corretta è la c: Gli ACK permettono al mittente di adattare la velocità di trasmissione in base alla capacità del ricevitore di elaborare i dati

## Esercizi

### Exercise

Assumiamo di trasmettere sempre frame di 64 bit (quindi inviamo sempre frame della stessa grandezza).

Assumiamo che il nostro marcatore di frame sia 011110.

Qual è l'overhead medio del nostro livello di collegamento?

Come prima cosa, **ogni 64 bit, dobbiamo aggiungere 6 \* 2 = 12 bit per i marcatori** (6 bit per l'inizio e 6 bit per la fine —> 011110).

Dobbiamo evitare di avere 4 uni di seguito, quindi **ogni terzo uno, abbiamo bisogno di aggiungere (to stuff) lo zero**, questo accade con una probabilità di 1/8 data una sequenza di 3 bit.

La probabilità di avere tre 1 consecutivi è 1/8, **dato che ci sono  $2^3 = 8$  possibili combinazioni di 3 bit**.

Dato che potremmo avere 1 bit (lo zero) ogni 8 da aggiungere, il numero medio di zeri da poter aggiungere è : **bit totali / possibili bit aggiunti, quindi  $62/8 = 7,75$  bit aggiunti in media ~ 8.**

**NOTA!** 62 perché se il frame ha 64 bit, una sotto-sequenza di 3 bit può iniziare in una delle prime 62 posizioni (da 0 a 61), perché l'ultima sotto-sequenza di 3 bit inizia alla posizione 61 e termina alla posizione 63.

Quindi : overhead marcatori = 12, overhead per bit di stuffing = 7,75.

La percentuale di overhead rispetto ai dati utili/effettivi è :

$$C = (12 + 7,75) / (64 + 12 + 7,75) * 100\% \rightarrow 19,75 / 83,75 * 100\% \approx 23,5820896\%$$

**Risultato** : in conclusione, il 23% circa della capacità viene persa per colpa dell'overhead.

**Nota: più grande è il marcitore, maggiore è l'overhead fisso, ma minore è la probabilità di dover fare bit stuffing.** Quindi, una buona regola è: utilizza marker grandi, ma cerca di avere frame di grandi dimensioni per compensare l'overhead fisso dovuto ai marker.

## Exercises

Assumi di aver ricevuto 0100 1010 1111 0110 0011 0101. Questo contiene il checksum.

- **1 -** Puoi dire se ci sono stati errori?
- **2 -** Calcola il checksum a 4 bit C del frame  $F = 1100 \ 1011 \ 1110 \ 0110 \ 0011$ . Poi aggiungi il checksum al frame e verifica se è corretto.
- **3 -** Supponi che il mittente invii  $[F, C]$ , ma il ricevitore riceve  $[F', C]$ , dove  $F'$  è una versione di  $F$  con errori. Puoi trovare  $F'$  in modo che il checksum sia ancora valido, anche se  $F' \neq F$ ?

**1 -** Per verificare se ci sono stati errori, devo fare la somma di tutti i bit compreso il **checksum** e controllare che il risultato dia tutti bit = 1.

$$\begin{aligned}> 0100 + 1010 &= 1110 \\> 1110 + 1111 &= 11101 \\> 11101 + 0110 &= 100011 \\> 100011 + 0011 &= 100110 \\> 100110 + 0101 &= \mathbf{101011}\end{aligned}$$

Calcolo del carry :

101011 → 1011 + 0010 = 1101 → **sbagliato**! Dovrebbero essere solo uni. La risposta è : **sì, ci sono stati errori.**

**2 -** Calcolo il checksum :

$$\begin{aligned}> 1100 + 1011 &= 10111 \\> 10111 + 1110 &= 100101 \\> 100101 + 0110 &= 101011 \\> 101011 + 0011 &= 101110\end{aligned}$$

Calcolo del carry :

$$101110 \text{ — scomponi} > \mathbf{1110} + 0010 = 10000 \text{ — scomponi} > \mathbf{0000} + 0001 = \mathbf{0001}$$

Complemento a uno :

0001 → **1110 → il nostro nuovo checksum!**

**2.2 -** Verifica sia corretto il **checksum** per il frame ricevuto : nel risultato devono esserci solo uni :

$$\begin{aligned}> \mathbf{1110} + 1100 &= 11010 \\> 11010 + 1011 &= 100101 \\> 100101 + 1110 &= 110011 \\> 110011 + 0110 &= 111001 \\> 111001 + 0011 &= \mathbf{111100}\end{aligned}$$

Calcolo del carry :

111100 → 1100 + 0011 = 1111 → **il risultato è corretto! Il checksum calcolato è corretto.**

**3 -** Per quale altro frame differente dal precedente potrebbe essere corretto il checksum ?

Per esempio, dato che l'addizione è una operazione che supporta la proprietà commutativa potremmo cambiare l'ordine dei dati e comunque il checksum andrebbe bene (ovviamente i dati che verranno ricevuti avranno un significato differente, quindi  $F \neq F'$ ).

$F = 1100 \ 1011 \ 1110 \ 0110 \ 0011$ .

Trovare  $F'$  in modo che il **checksum** sia ancora valido con i dati sbagliati :

$F' = 1011 \ 1100 \ 0110 \ 1110 \ 0011$

$$\begin{aligned}> \mathbf{1110} + 1011 &= 11001 \\> 11001 + 1100 &= 100101 \\> 100101 + 0110 &= 101011 \\> 101011 + 1110 &= 111001 \\> 111001 + 0011 &= \mathbf{111100}\end{aligned}$$

Calcolo del carry :

111100 → 1100 + 0011 = 1111 → **il risultato è corretto!** Quindi possiamo concludere che è stato possibile trovare un frame differente con lo stesso checksum, allora questo vuol dire che **se ci fosse stata una interferenza che ha fatto passare il frame da F a F', non ce ne saremo accorti.**

Considera il seguente dato binario:

1010110010011111

Calcola il valore dell'IP checksum, considerando parole di lunghezza 4 bit.

Scegli un'alternativa:

- a. 1000
- b. 1110
- c. 01110
- d. 0101

Speziamo il dato in parole di 4 bit e facciamo la somma binaria

$1010 + 1100 + 1001 + 1111 = 101110$

Facciamo la somma in modulo a 1 su 4 bit, ovvero sommiamo:

$10 + 1110 = 10000$

Rifacciamo la somma in modulo a 1 su 4 bit:

$1 + 0000 = 0001$

Facciamo il modulo a 1 e otteniamo la soluzione

Si consideri la stringa binaria (divisa in gruppi di 8 bit per facilitare la lettura).

**10010110.1111010.1111111.11001001.10100101**

e si applichi bit-stuffing con il marker

**01111110**

Quanti bit vengono aggiunti alla stringa originale?

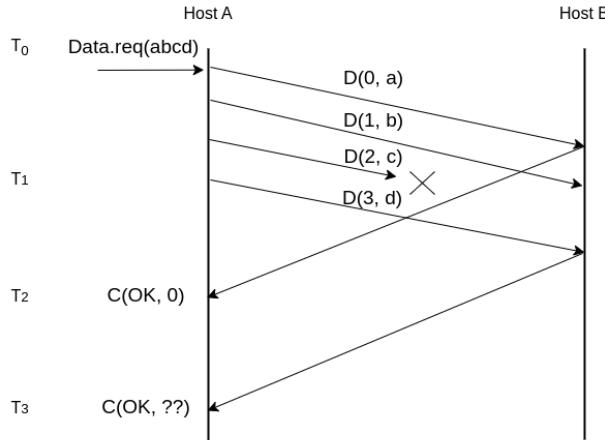
Scegli un'alternativa:

- a. 18
- b. 16
- c. 19
- d. 17

Con bit-stuffing si aggiunge il marker all'inizio ed alla fine del frame (totale di 8+8 bit) e si aggiunge uno zero ogni volta che la sequenza di bit è '11111'. La sequenza appare tre volte nella stringa quindi si aggiunge un totale di 19 bit: \_\_\_\_\_100101101111\_01011111\_11111\_00100110100101\_\_\_\_\_

## RELIABLE TRANSPORT (ABP, GO-BACK-N, SELECTIVE REPEAT)

### Esercizi



Immagina di aver implementato un datalink layer che utilizza Go-back-n e osserva in figura i frame del traffico che è stato generato. Assumi di avere una finestra di 4 frames. Quale delle seguenti affermazioni è **CORRETTA**? Quale Sequence Number conterrà l'ACK al momento  $T_3$ ?

Scegli un'alternativa:

- a. I primi due frame sono stati ricevuti correttamente. L'ultimo ACK conterrà come Sequence Number il valore 1.
- b. Tre frame sono considerati ricevuti correttamente. L'host A reinvia solo il frame 2.
- c. I primi due frame e l'ultimo sono stati ricevuti correttamente. L'ultimo ACK conterrà il numero 2 dell'unico frame che non è stato ricevuto.
- d. Solo il primo frame è considerato come ricevuto correttamente. L'ACK al tempo  $T_3$  conterrà il Sequence Number uguale a 0.

Nel protocollo Go-Back-N, i pacchetti devono essere ricevuti in ordine per poter essere processati correttamente. In questo scenario, i primi due pacchetti vengono ricevuti e processati senza problemi. Tuttavia, dato che il penultimo pacchetto non arriva a destinazione, il protocollo non permetterà la ricezione e l'elaborazione dei pacchetti successivi. Anche se l'ultimo pacchetto viene ricevuto, questo verrà scartato. Quindi invierà un ACK che riporta il Sequence Number dell'ultimo pacchetto ricevuto correttamente, ossia 1.

La risposta corretta è:

I primi due frame sono stati ricevuti correttamente. L'ultimo ACK conterrà come Sequence Number il valore 1.

In un Datalink che usa selective repeat, si dà un numero di sequenza per ciascun frame, ma il ricevitore manda comunque solo ACK cumulativi. L'ACK contiene l'ultimo numero ricevuto correttamente, non il prossimo atteso. La finestra del trasmettitore è lunga 5 ed il primo numero all'interno della finestra è 20. Non sappiamo cosa sia successo precedentemente, e quindi cosa contenga il resto della finestra. All'istante  $t_0$  in cui cominciamo ad osservare, il trasmettitore invia il 20, 21, 22, 23: il 21 non arriva a destinazione. Quale delle seguenti frasi è vera, riferita agli eventi al tempo  $t_1 > t_0$ .

Scegli un'alternativa:

- a. Il ricevente manda ACK separati per 20, 22, 23, ed il trasmettitore, una volta ricevuti, reinvia il 21.
- b. Se il trasmettitore riceve un ACK per il numero 23, farà scorrere la finestra ma solo fino al 20, il prossimo numero di sequenza inviato sarà il 21 che viene ritrasmesso.
- c. Se prima di ricevere un ACK, dal livello superiore arrivassero anche 24, 25, 26, verrebbero tutti inviati immediatamente.
- d. Se il trasmettitore riceve un ACK per il numero 23, farà scorrere la finestra ed il prossimo numero di sequenza inviato sarà il 24.

d) In questo esempio gli acknowledge non sono separati ma cumulativi (il che elimina una risposta), quindi se il trasmettitore riceve un ACK per il numero 23, significa che tutti i numeri fino al 23 sono stati ricevuti correttamente, e non c'è motivo di reinviare il 21 (che elimina un'altra risposta). Evidentemente il ricevitore aveva già ricevuto in invii precedenti il 21, ma non aveva ricevuto il 20 e quindi non poteva fare un ACK cumulativo. La finestra di invio è lunga 5 quindi prima di ricevere un ACK, il

trasmettitore non potrebbe inviare 24,25,26, ma solo 24, avendo già inviato 4 numeri (il che elimina un risposta).

Considera un sistema Go-Back-N in cui il trasmettitore invia frame utilizzando una finestra di dimensione  $W = 10$ . I numeri di sequenza indicano i frame. La capacità del canale è molto alta, pari a 1 Gb/s, e il RTT è 20 ms. Ogni frame ha una dimensione di 1500 B, mentre l'ACK corrispondente è grande 60B. Calcola il throughput medio del sistema (arrotondandolo al Mb/s) spiegando i passaggi compiuti e descrivendo alla fine la sequenza temporale dei passaggi compiuti nella comunicazione (al momento  $T_0 = ...$ ).

Il throughput in un sistema di trasmissione dati rappresenta la quantità di dati che può essere trasmessa correttamente attraverso una connessione in un dato intervallo di tempo. Nei protocolli di controllo del flusso come Go-Back-N, il throughput dipende strettamente dalla dimensione della finestra  $W$  e dal RTT quando il tempo di trasmissione è molto piccolo.

#### Dati

Capacità del canale = 1Gb/s = 10 bit/s<sup>9</sup>

Dimensione del frame di dati = 1500B = 12000 bit

Dimensione dell'ACK = 60B = 480 bit

Dimensione finestra ( $W$ ) = 10

RTT = 20 ms

#### Soluzione

Prima di tutto calcoliamo il tempo di trasmissione di un singolo pacchetto e di un ACK

$$\text{Tempo di trasmissione di un pacchetto} = t_{\text{trasm}} = 12000 \text{ bit} / 10^9 \text{ bit/s} = 0.012 \text{ ms}$$

$$\text{Tempo di trasmissione di un ACK} = t_{\text{trasm,ACK}} = 480 \text{ bit} / 10^9 \text{ bit/s} = 0.00048 \text{ ms}$$

Il tempo di andata e ritorno (RTT) deve includere anche il tempo di trasmissione degli ACK. Poiché il RTT è molto più grande rispetto al tempo di trasmissione di un singolo pacchetto e di un ACK, possiamo assumere che l'impatto del tempo di trasmissione degli ACK sia trascurabile rispetto all'RTT.

Quindi il throughput medio del sistema può essere approssimato come:

$$\text{Throughput medio} = (W * \text{Dimensione pacchetto}) / \text{RTT} = (10 * 12000 \text{ bit}) / 20 * 10^{-3} \text{ s} = 6 \text{ Mb/s}$$

#### Spiegazione

Nel protocollo Go-Back-N, un trasmettitore può inviare fino a  $W$  frame consecutivamente, senza attendere l'ACK per ciascuno. Questo permette al trasmettitore di "riempire" il canale fino al limite della finestra. Una volta che la finestra è stata "riempita", il trasmettitore deve aspettare di ricevere gli ACK per alcuni dei frame inviati prima di poter continuare a trasmettere. Poiché il sistema deve attendere almeno un ciclo RTT per ricevere gli ACK, il throughput effettivo è limitato dal numero di frame che possono essere trasmessi durante quell'RTT. In altre parole, il sistema può inviare  $W$  frame per ogni ciclo di RTT.

#### Sequenza Temporale

[Tempo  $T_0 = 0$ ] Il trasmettitore inizia a inviare il primo frame

[Tempo  $T_1 = T_0 + t_{\text{trasm}} = 0.012 \text{ ms}$ ] Il trasmettitore termina l'invio del primo frame e inizia a inviare il secondo

[Tempo  $T_2 = T_0 + (10 * t_{\text{trasm}}) = 0.12 \text{ ms}$ ] Il trasmettitore completa l'invio di tutti i frame della finestra

[Tempo  $T_3 = T_1 + \text{RTT} / 2 = 10.012 \text{ ms}$ ] Il ricevitore riceve il primo frame

[Tempo  $T_4 = T_3 + t_{\text{trasm,ACK}} = 10.01248 \text{ ms}$ ] Il ricevitore invia l'ACK per il primo frame

[Tempo  $T_5 = T_1 + \text{RTT} + t_{\text{trasm,ACK}} = 20.01248 \text{ ms}$ ] Il trasmettitore riceve il primo ACK e può trasmettere un nuovo frame

Nota che se volessimo fare un calcolo esatto avremmo avuto:

$$\text{Tempo di trasmissione di } 15000 \text{ B: } 20.01248 \text{ ms, Throughput medio} = 1500 * 8 / 0.02001248 = 599625.8334 \approx 6 \text{ Mb/s}$$

Un trasmettitore utilizza il meccanismo di sliding window con una finestra di dimensione 7. La latenza di un frame per raggiungere la destinazione è di 5 ms, i pacchetti hanno una grandezza di 4500 B e il link fisico che collega i due terminali ha una capacità di 10 Mb/s. Quanti frame si riescono a inviare completamente prima di ricevere il primo ACK? (Si ignori pure il tempo di trasmissione ed elaborazione dell'ACK per il calcolo)

Scegli un'alternativa:

- a. 5
- b. 1
- c. 3
- d. 7

### Dati

Dimensione finestra (W) = 7

Latenza = 5 ms

Dimensione dei pacchetti = 4500 B =  $36 * 10^3$  bit

Capacità del link = 10 Mb/s =  $10^7$  bit/s

### Soluzione

Per prima cosa calcoliamo quanto impiega un frame ad essere trasmesso,

$$T_{trasm} = \text{Dimensione dei pacchetti} / \text{Capacità del link} = (36 * 10^3 \text{ bit}) / (10 * 10^6 \text{ bit/s}) = 3.6 \text{ ms}$$

Dato che approssimativamente il RTT è il doppio della Latenza (quindi 10 ms), possiamo calcolare il tempo totale prima di ricevere un ACK dopo aver inviato un pacchetto,

$$T_{tot} = RTT + T_{trasm} = 13.6 \text{ ms}$$

Il tempo di elaborazione dell'ACK può essere ignorato. Sapendo che il  $T_{trasm}$  e il  $T_{tot}$  possiamo calcolare quanti frame possono essere inviati prima di ricevere il primo ACK facendo

$$\text{Numero frame inviati} = T_{tot} / T_{trasm} = 3.78$$

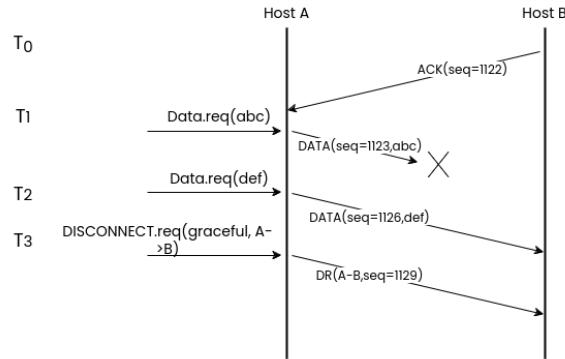
Quindi la risposta è che si riescono a inviare 3 frame prima di ricevere il primo ACK.

La risposta corretta è:

3

Nella figura seguente è rappresentato il traffico generato da una connessione go-back-n tra due terminali, Host A e Host B. Ogni segmento contiene un campo Sequence Number, che viene incrementato per ogni byte inviato all'interno della connessione. Dopo l'istante  $T_3$ :

- quali ACK verranno inviati dall'Host B e ricevuti dall'Host A?
- Come reagirà l'Host A a tali ACK?
- Quando Host A può chiudere la connessione?



### Soluzione prima domanda

Nel caso migliore, il terminale riesce a trasmettere senza ritardi dovuti a tentativi di ritrasmissione. Quindi la velocità di trasmissione dei dati sarà uguale alla capacità della rete, cioè 50 Mb/s. Il tempo di trasmissione quindi verrà calcolato semplicemente come

Tempo = Dati / Velocità di Trasmissione

Che sarà quindi

$$\text{Tempo} = 150 \text{ Mb} / 50 \text{ MB/s} = (150 * 10^6 \text{ bit}) / (50 * 10^6 \text{ bit/s}) = 3s$$

### Soluzione seconda domanda

In una situazione di carico massimo, dobbiamo calcolare il throughput del canale utilizzando la seguente formula

$$P = G * e^{-G}$$

Dove P è la probabilità di trasmettere con successo i dati, mentre G è il carico, ovvero il numero medio di frame inviati per slot. In questo caso, ogni terminale ha sempre qualcosa da inviare ad ogni slot, quindi il valore del carico G è pari al numero di terminali connessi al canale. Avendo 7 terminali si ha che,

$$P = 7 * e^{-7} = 0.0064$$

Di conseguenza, l'efficienza del canale crolla al 0.64%. La capacità effettiva del canale può essere calcolata come,  $50\text{Mb/s} * 0.0064 = 0.32 \text{ Mb/s}$ . Per ottenere la velocità di trasmissione per ogni terminale bisognerà dividere la capacità effettiva appena calcolata per il numero di terminali ottenendo,

$$\text{Velocità di Trasmissione} = \text{Capacità effettiva} / \# \text{Terminali} = 0.32 \text{ Mb/s} / 7 = 0.046 \text{ Mb/s}$$

Infine per ottenere il tempo impiegato da un terminale per inviare 150 Mb di dati in queste condizioni si avrà che,

$$\text{Tempo} = (150 * 10^6 \text{ bit}) / (0.046 * 10^6 \text{ bit/s}) = 3261s = 54 \text{ min}$$

### Soluzione terza domanda

Nel caso medio dove tutti i terminali connessi al canale di trasmissione hanno probabilità di stramettere un frame pari a 1/7, il carico del sistema G = 1. Sapendo questo, possiamo andare a calcolarci P,

$$P = 1 * e^{-1} = 0.37$$

La capacità effettiva del canale è uguale a  $50\text{Mb/s} * 0.37 = 18.5 \text{ Mb/s}$ . La velocità di trasmissione per ogni terminale si otterrà dividendo la capacità effettiva per il numero di terminali ottenendo,

$$\text{Velocità di Trasmissione} = \text{Capacità effettiva} / \# \text{Terminali} = 18.5 \text{ Mb/s} / 7 = 2.64 \text{ Mb/s}$$

Infine il tempo impiegato da un terminale per inviare 150 Mb di dati in queste condizioni sarà,

$$\text{Tempo} = (150 * 10^6 \text{ bit}) / (2.64 * 10^6 \text{ bit/s}) = 56.8 \text{ sec}$$

In un Datalink che usa Go-Back-N, il campo Sequence Number del pacchetto è di 3 bit. La finestra del trasmettitore è lunga 5 e il primo numero di sequenza nella finestra è 5. Nel momento in cui iniziamo ad osservare, il trasmettitore invia i frame 5, 6, 7, 0, 1. Dopo un breve istante, il trasmettitore riceve l'ultimo ACK contenente Sequence Number 6. A quel punto l'applicazione del trasmettitore invia nel buffer di invio anche i frame 2, 3. Quale delle seguenti affermazioni è vera?

Scegli un'alternativa:

- a. L'ACK con Sequence Number 6 indica che il ricevitore ha ricevuto correttamente i frame con Sequence Number fino a 5. Il trasmettitore aggiornerà la finestra e invierà i frame con Sequence Number 2 e 3.
- b. Il trasmettitore aggiornerà il primo numero di sequenza della finestra a 7, considerando ricevuti i frame 5 e 6, e invierà i frame con Sequence Number 2 e 3. Dopo un timeout, se il trasmettitore non riceve ACK, ritrasmetterà 7, 0, 1, 2, 3.
- c. Il trasmettitore non aggiornerà il primo numero di sequenza della finestra a 1, poiché l'ACK con Sequence Number 6 non conferma i frame 7 e 0. Il trasmettitore aggiornerà la finestra a 7 e aspetterà la ricezione degli ACK per i frame 7 e 0 prima di inviare i successivi frame.

Nei frame scambiati tra due terminali che utilizzano il meccanismo Selective Repeat, il campo Sequence Number è di 4 bit. La finestra del trasmettitore è lunga 6 e il primo numero di sequenza è 13. Il trasmettitore invia correttamente 6 frame e si ferma in attesa di un ACK. Successivamente il ricevitore riceve un ACK con il campo Sequence Number cumulativo contenente il valore 15, e dei selective ACK. Quale delle seguenti affermazioni è **VERA**?

Scegli un'alternativa:

- a. Se i Selective ACK includono solo 0 e 1, il trasmettitore ritrasmetterà 15 e 2 poiché non sono stati ancora confermati.
- b. Il trasmettitore aggiorna il primo numero di sequenza della finestra a 0 e, se riceve un Selective ACK per 1, considera questo frame confermato anche se ricevuto fuori sequenza.
- c. L'ACK con Sequence Number 15 indica che il ricevitore ha ricevuto i frame fino al 14 e sta aspettando il frame con Sequence Number 15, includendo eventualmente i Selective ACK per 0 e 1.
- d. Il trasmettitore considera ricevuti correttamente i frame fino a 15, e se c'è un selective ACK per 1, fa scorrere la finestra fino al valore 1.

# DATALINK LAYER – SHARING RESOURCES (TDMA, S-ALOHA)

## Domande Teoria a Crocette

Quale delle seguenti affermazioni descrive correttamente i vantaggi e gli svantaggi di Slotted-ALOHA rispetto a TDMA?

Scegli un'alternativa:

- a. TDMA è più efficiente di Slotted-ALOHA in condizioni di saturazione, ma Slotted-ALOHA è più flessibile e performante quando il carico di rete è basso.
- b. Slotted-ALOHA elimina completamente il rischio di collisioni, mentre TDMA è soggetto a collisioni.
- c. Slotted-ALOHA garantisce sempre un utilizzo efficiente del canale, mentre TDMA soffre di inefficienza quando la rete è congestionata.
- d. TDMA funziona meglio quando ci sono pochi terminali attivi, mentre Slotted-ALOHA è ottimale in reti con molti terminali che inviano traffico ad alta frequenza.

Slotted-ALOHA permette una maggiore flessibilità e prestazioni migliori a basso carico, ma soffre di collisioni ad alto carico. TDMA, d'altra parte, è più efficiente in condizioni di saturazione poiché non ci sono collisioni, ma può essere inefficiente quando il carico di rete è basso, poiché i time slot riservati potrebbero rimanere inutilizzati.

La risposta corretta è la a:

TDMA è più efficiente di Slotted-ALOHA in condizioni di saturazione, ma Slotted-ALOHA è più flessibile e performante quando il carico di rete è basso.

Per lo schema di accesso TDMA, quale delle seguenti affermazioni è **VERA**?

Scegli un'alternativa:

- a. Se un terminale deve tramettere urgentemente, può "prendere in prestito" lo slot di un altro terminale.
- b. Ogni terminale può trasmettere in qualsiasi momento, purché non ci siano collisioni
- c. Ogni terminale ha uno slot temporale fisso in cui può trasmettere, anche se gli altri terminali non stanno trasmettendo
- d. La rete funziona meglio a basso carico, riducendo i ritardi di trasmissione

In un sistema TDMA (Time Division Multiple Access), il tempo viene suddiviso in slot temporali fissi, e ogni terminale ha un proprio slot assegnato in cui può trasmettere. Questo significa che i terminali non possono trasmettere in qualsiasi momento, ma devono aspettare il proprio turno per trasmettere, anche se gli altri terminali non stanno utilizzando i loro slot. L'allocazione degli

slot è predefinita, e ogni terminale ha la garanzia di poter trasmettere in quel preciso intervallo di tempo, senza rischio di collisioni con altri terminali.

La risposta corretta è la c:

Ogni terminale ha uno slot temporale fisso in cui può trasmettere, anche se gli altri terminali non stanno trasmettendo.

Ti quanti link fisici c'è bisogno per creare una topologia full mesh con n nodi?

Scegli un'alternativa:

- a.  $n*(n-1)$
- b.  $n*(n-1)/2$
- c.  $n/2$
- d.  $n-1$

Per collegare n nodi uno con l'altro ci vogliono  $(n-1)$  collegamenti per ciascun nodo, quindi  $n*(n-1)$ , ma usando cavi full-duplex non c'è bisogno di due cavi per coppia ma di un solo cavo:  $n*(n-1)/2$ .

## Esercizi

### Exercise on TDMA

Considera un sistema TDMA in cui la **velocità di trasmissione è  $b = 100 \text{ Mb/s}$** , ci sono **n=5 terminali**, ciascuno dei quali è assegnato a uno slot di **lunghezza  $s = 1 \text{ ms}$** .

- 1 - Qual è la quantità massima di dati che un terminale può inviare in uno slot?
- 2 - Qual è la quantità massima di dati che il terminale può inviare in un secondo?
- 3 - Supponiamo che il terminale 1 debba inviare 250 kb, qual è il tempo necessario t? (suggerimento: non c'è una sola risposta, ma una migliore, una peggiore e una media).

Come prima cosa, dobbiamo convertire la velocità di trasmissione da Mbps a bps :

$$> b = 100 \text{ Mbps} \implies 100 * 1.000.000 = 100.000.000 = 10^8 \text{ bps.}$$

Poi convertiamo il tempo della lunghezza dello slot da millisecondi a secondi :

$$> s = 1\text{ms} \implies 1 * 10^{-3} = 10^{-3} \text{ s.}$$

1 -

Il **massimo ammontare di dati che un terminale può inviare ad uno slot** è dato dalla formula :

R = lunghezza slot \* bit-rate del sistema.

$$R = 10^{-3} \text{ s} * 10^8 \text{ bps} = 10^5 \text{ bit.}$$

Considera una rete che usa un MAC basato su un TDMA, con l'accesso condiviso tra 22 stazioni e slot di 1 ms (milli-secondo)

Quale deve essere la capacità minima C del canale per garantire che un terminale riesca a inviare 100MB in meno di 2 secondi?

Scegli un'alternativa:

- a. Almeno 5.2 Gb/s
- b. Almeno 9 Gb/s
- c. Almeno 8.1 Gb/s
- d. Almeno 13.3 Gb/s

Handwritten notes for TDMA capacity calculation:

$b = ?$   
 $n = 22$   
 $s = 1 \text{ ms}$

$\text{dati} = 100 \text{ MB} = 800 \text{ Mb}$   
 $\text{limite} = 2 \text{ secondi}$

$\frac{1000 \text{ ms}}{22} = 45 \text{ volte in cui un terminale può trasmettere al secondo}$

$\frac{400}{45} = 8,8 \text{ Mb/ms} = 8800 \text{ Mb/s} = 8,8 \text{ Gb/s}$

Una rete ha una capacità massima di 180 Mb/s, viene gestita con un TDMA, ed ha 6 terminali. Una seconda rete ha una capacità di 250 Mb/s e viene gestita con S-ALOHA, ha 5 terminali. In entrambi i casi il numero di slot è uguale al numero di terminali e si assume che un terminale che invia un frame ha dati per occupare uno slot intero. Scegliere la risposta **FALSA**, le risposte si riferiscono al comportamento medio e si può usare la formula che vale per il comportamento asintotico.

Scegli un'alternativa:

- a. Se nella rete S-ALOHA c'è un solo terminale che trasmette e tutti gli altri sono spenti, il suo throughput massimo è maggiore di quello dei terminali nella rete TDMA.
- b. Se nella rete S-ALOHA, la somma dei frame inviati da tutti i terminali è mediamente 1 per slot, mentre nella rete TDMA ci sono solo 3 terminali che trasmettono sempre, il throughput totale nella rete S-ALOHA è più alto di quello di TDMA
- c. Se in entrambe le reti, ciascun terminale ha sempre un frame da inviare in ogni slot, il throughput medio per terminale della rete TDMA è maggiore di quello della rete S-ALOHA
- d. Se nella rete S-ALOHA, la somma dei frame inviati da tutti i terminali è mediamente 1 per slot, il throughput massimo di S-ALOHA (per utente) è più alto di quello (per utente) di TDMA

Il throughput si una rete S-ALOHA è dato dalla formula  $Ge^G$  dove G è il carico, ovvero il numero medio di frame inviati per slot (complessivo per tutti i terminali). Il massimo di efficienza si ha quando G=1, che nel caso descritto produce un'utilizzo della rete di 1/e. Avendo 250Mb/s di capacità totale il throughput massimo è  $250/e = 92 \text{ Mb/s}$ , da dividere per i 5 terminali, quindi circa 18 Mb/s. Nella stessa situazione la rete TDMA invece ha efficienza massima, quindi  $180/6 = 30 \text{ Mb/s}$ . Quindi non è vero che S-ALOHA ha un throughput più alto per utente.

È invece vero che il throughput complessivo è maggiore della rete TDMA con 3 utenti (90Mb/s). È anche vero che se c'è un solo terminale che trasmette, in S-ALOHA non ci sono collisioni, quindi il singolo terminale può usare 250Mb/s mentre in TDMA avrà sempre comunque solo 30 Mb/s. Se ciascun terminale ha sempre qualcosa da trasmettere in tutti gli slot, allora G=5 e l'efficienza di S-ALOHA crolla al 3.3% ( $250 * 0.033 = 8 \text{ Mb/s}$ , divisa tra 5 terminali) mentre in TDMA rimane 30 Mb/s ed è quindi maggiore.

Dato un canale di comunicazione condiviso che utilizza Slotted-ALOHA, con capacità massima di 50 Mb/s e con 7 terminali ad esso connessi, calcola quanto ci metterebbe un terminale ad inviare 150 Mb di dati:

1. Nel caso migliore (nessun altro terminale ha pacchetti da inviare)
2. Nel caso di carico massimo (tutti i terminali connessi hanno sempre qualcosa da inviare)
3. Nel caso medio (ogni terminale spedisce un frame ogni 7 slots)

descrivendo il ragionamento effettuato per ciascuna ipotesi.

#### Soluzione prima domanda

Nel caso migliore, il terminale riesce a trasmettere senza ritardi dovuti a tentativi di ritrasmissione. Quindi la velocità di trasmissione dei dati sarà uguale alla capacità della rete, cioè 50 Mb/s. Il tempo di trasmissione quindi verrà calcolato semplicemente come

Tempo = Dati / Velocità di Trasmissione

Che sarà quindi

$$\text{Tempo} = 150 \text{ Mb} / 50 \text{ MB/s} = (150 * 10^6 \text{ bit}) / (50 * 10^6 \text{ bit/s}) = 3 \text{ s}$$

#### Soluzione seconda domanda

In una situazione di carico massimo, dobbiamo calcolare il throughput del canale utilizzando la seguente formula

$$P = G * e^{-G}$$

Dove P è la probabilità di trasmettere con successo i dati, mentre G è il carico, ovvero il numero medio di frame inviati per slot. In questo caso, ogni terminale ha sempre qualcosa da inviare ad ogni slot, quindi il valore del carico G è pari al numero di terminali connessi al canale. Avendo 7 terminali si ha che,

$$P = 7 * e^{-7} = 0.0064$$

Di conseguenza, l'efficienza del canale crolla al 0.64%. La capacità effettiva del canale può essere calcolata come,  $50\text{Mb/s} * 0.0064 = 0.32 \text{ Mb/s}$ . Per ottenere la velocità di trasmissione per ogni terminale bisognerà dividere la capacità effettiva appena calcolata per il numero di terminali ottenendo,

$$\text{Velocità di Trasmissione} = \text{Capacità effettiva} / \# \text{Terminali} = 0.32 \text{ Mb/s} / 7 = 0.046 \text{ Mb/s}$$

Infine per ottenere il tempo impiegato da un terminale per inviare 150 Mb di dati in queste condizioni si avrà che,

$$\text{Tempo} = (150 * 10^6 \text{ bit}) / (0.046 * 10^6 \text{ bit/s}) = 3261 \text{ s} = 54 \text{ min}$$

#### Soluzione terza domanda

Nel caso medio dove tutti i terminali connessi al canale di trasmissione hanno probabilità di stramettere un frame pari a 1/7, il carico del sistema G = 1. Sapendo questo, possiamo andare a calcolarci P,

$$P = 1 * e^{-1} = 0.37$$

La capacità effettiva del canale è uguale a  $50\text{Mb/s} * 0.37 = 18.5 \text{ Mb/s}$ . La velocità di trasmissione per ogni terminale si otterrà dividendo la capacità effettiva per il numero di terminali ottenendo,

$$\text{Velocità di Trasmissione} = \text{Capacità effettiva} / \# \text{Terminali} = 18.5 \text{ Mb/s} / 7 = 2.64 \text{ Mb/s}$$

Infine il tempo impiegato da un terminale per inviare 150 Mb di dati in queste condizioni sarà,

$$\text{Tempo} = (150 * 10^6 \text{ bit}) / (2.64 * 10^6 \text{ bit/s}) = 56.8 \text{ sec}$$

## Domande Aperte

- 1) TDMA e random access: tdma vs aloha.

Time-Division Multiple Access (TDMA) è il modo più intuitivo per dare il diritto di inviare messaggi a multipli host connessi ad un unico terminale. In un TDM, se la rete ha m terminali, il tempo è diviso in m slot di dimensione fissa, con uno slot assegnato a un solo terminale. Gli slot sono ripetuti in modo ciclico, quindi il terminale i può trasmettere solo allo slot i.

TDMA è efficiente quando la rete è satura. In questa condizione, il TDMA è alla massima efficienza: ogni slot è occupato e non ci sono collisioni. TDMA è poco efficiente quando la rete è libera. La limitazione del TDM è che ogni terminale avrà l'opportunità di trasmettere tutti gli 1/m slot, indipendentemente da ciò che fanno gli altri. Se la stazione i ha un frame da inviare al tempo i+1, dovrà attendere un ciclo completo di m slot prima di poter trasmettere, anche se gli altri nodi non hanno nulla da trasmettere. TDMA introduce quindi un ritardo medio di m/2 slot.

Quindi il TDMA è efficiente quando la rete è congestionata, ed è inefficiente quando la rete è scarica.

Random Access (il padre di tutti gli schemi di accesso casuale è Aloha) dove i terminali tentano di trasmettere e se si verifica una collisione viene rilevata e il frame viene inviato di nuovo.

Con ALOHA quando un nodo ha bisogno di inviare un frame, aspetta l'inizio dello slot successivo e lo trasmette. ALOHA ammette una probabilità di collisione, le collisioni fanno fallire la trasmissione. (La differenza con il TDMA è che ogni terminale può trasmettere in ogni slot).

Se due nodi trasmettono nello stesso slot, la trasmissione fallisce e i nodi entrano in stato **backlogged** (ritentano la trasmissione). Se un terminale backlogged riceve un altro (nuovo) frame da inviare, lo scarta.

G è il carico, ovvero il numero medio di frame che tentano di essere inviati nella rete (tasso di frame che viene inviato per slot nella rete).

- G=1 è l'ottimo, dove mediamente per ogni slot viene inviato un frame.
- G<1 la rete sarebbe scarica e si comporta in modo poco efficiente (perché ho pochi dati da inviare)
- G>1 le prestazioni collassano in quanto si verificano collisioni (Congestion Collapse).

Probabilità che in ciascuno slot voi mandiate correttamente un frame :  $P_s = G e^{-G}$

Quindi  $P_s = 1/e$  è l'efficienza massima ma questo significa che scarta 2/3 dei pacchetti per collisione.

Da un lato hai sistemi deterministici come TDMA, che garantiscono prestazioni e sono facili da implementare, ma sono inflessibili e sprecano risorse a basso carico. Dall'altro hai schemi di accesso casuali, che sono più veloci a basso carico, ma possono crollare totalmente ad alto carico. Sono entrambi utilizzati, in applicazioni diverse.

## NETWORK LAYER (FRAGMENTATION, HOT POTATO, ROUTING TABLE, DISTANCE VECTOR)

### Domande Teoria a Crocette

Considera una rete che utilizza l'algoritmo di routing Hot Potato. Quale tra le seguenti affermazioni è **VERA**?

Scegli un'alternativa:

- a. Hot Potato richiede che ogni nodo mantenga uno stato complesso della rete, compresi i costi di collegamento
- b. L'algoritmo Hot Potato si basa sulla conoscenza globale della rete per determinare il percorso ottimale per i pacchetti
- c. Hot Potato instrada i pacchetti verso il prossimo nodo disponibile senza tener conto del percorso completo

L'algoritmo Hot Potato si caratterizza per prendere decisioni di instradamento basate esclusivamente sulle informazioni locali disponibili al nodo (e dei

pacchetti che transitano attraverso), senza una conoscenza completa del percorso finale. Questo approccio può ridurre il tempo di transito dei pacchetti attraverso decisioni rapide e locali, ma funziona solo con topologie con strutture ad albero.

La risposta corretta è la c:

Hot Potato instrada i pacchetti verso il prossimo nodo disponibile senza tener conto del percorso completo

## Esercizi

### Exercise

Considera il caso in cui l'ordine di generazione dei vettori di distanza (DV) è strettamente alfabetico (da A a E).

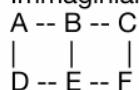
Crea un'altra sequenza di eventi e aggiornamenti delle tabelle di routing (RT), e verifica che raggiunga la convergenza.

Per eseguire l'esempio, possiamo considerare una rete con i router A, B, C, D, ed E e assumere che la generazione dei vettori di distanza (DV) avvenga in ordine alfabetico (da A a E).

Supponiamo che **tutti i collegamenti abbiano un costo unitario di 1**.

Scenario della Rete :

Immaginiamo la seguente topologia iniziale:



**Costi dei collegamenti:** tutti i collegamenti hanno un costo di 1.

Sequenza di Eventi e Aggiornamenti delle Tabelle di Routing

Passo 1: Inizializzazione

Ogni router inizializza la propria tabella di routing con il costo noto dei suoi collegamenti diretti.

**A:** {B: 1, D: 1}

**B:** {A: 1, C: 1, E: 1}

**C:** {B: 1, F: 1}

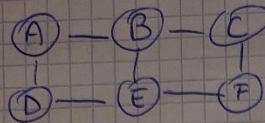
**D:** {A: 1, E: 1}

**E:** {D: 1, B: 1}

Passo 2: Generazione dei DV in ordine alfabetico

1. **Router A** genera un DV:  
**DV(A)**: {A: 0, B: 1, D: 1}
2. **Router B** riceve **DV(A)** e aggiorna la propria tabella:  
**RT(B)** dopo aver ricevuto **DV(A)**:  
Per **A**: il costo attraverso **A** è 2 ( $B \rightarrow A$ ) ma il costo diretto è 1, quindi non cambia.
3. **Router B** genera il suo DV:  
**DV(B)**: {A: 1, B: 0, C: 1, E: 1}
4. **Router C** riceve **DV(B)** e aggiorna la propria tabella:  
**RT(C)** dopo aver ricevuto **DV(B)**:  
Per **B**: il costo attraverso **B** è 2 ( $C \rightarrow B$ ) ma il costo diretto è 1, quindi non cambia.  
Per **A**: il costo attraverso **B** è 2 ( $C \rightarrow B \rightarrow A$ ) che è maggiore di 1 (direttamente).
5. **Router C** genera il suo DV:  
**DV(C)**: {B: 1, C: 0, F: 1}
6. **Router D** riceve **DV(C)** e aggiorna la propria tabella:  
**RT(D)** dopo aver ricevuto **DV(C)**:  
Per **C**: costo attraverso **B** è 2 ( $D \rightarrow B \rightarrow C$ ) ma il costo diretto è 2, quindi non cambia.
7. **Router D** genera il suo DV:  
**DV(D)**: {A: 1, E: 1, B: 1}
8. **Router E** riceve **DV(D)** e aggiorna la propria tabella:  
**RT(E)** dopo aver ricevuto **DV(D)**:  
Per **D**: costo attraverso **D** è 2 ( $E \rightarrow D$ ) che è uguale al costo diretto.
9. **Router E** genera il suo DV:  
**DV(E)**: {D: 1, B: 1}
10. **Router A** riceve **DV(B)**, **DV(D)**, e **DV(E)** e aggiorna la propria tabella:  
**RT(A)** dopo aver ricevuto gli aggiornamenti:  
Per **B**: costo attraverso **A** e **B** è 1.  
Per **D**: costo attraverso **A** e **D** è 1.

III) Distance Vector, costo 1 per i collegamenti  
generazione in ordine alfabetico



1. Tabella di Routing all'inizio:

A	B	C	D	E	F
A:0	F:2	C:0	D:0	E:0	F:0
B:1	A:1	B:1	A:1	D:1	C:1
D:1	C:1	E:1	F:1	B:1	E:1
C:2	E:1	A:1	D:2	F:1	A:2
E:2	D:2	D:2	C:3	A:2	B:2

2. Generazione DV:

A → B A:0 B:1 D:1  
A → D A:0 B:1 D:1

B → A A:1 B:0 C:1 D:2 E:1

B → C

B → E

C → B C:0 A:2 B:1 D:3 E:2 F:1

C → F

D → A D:0 A:1 B:2 E:1

D → E

E → B A:2 B:1 C:2 D:1 E:0 F:1

E → D

E → F

F → C A:3 B:2 C:1 D:2 E:1 F:0

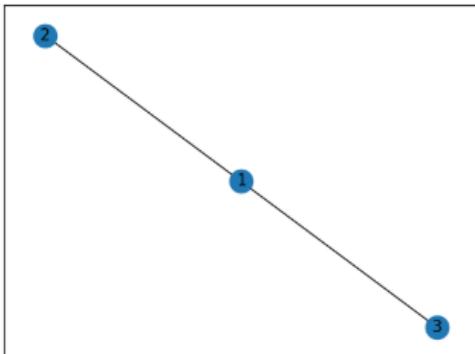
F → E

A → B A:0 B:1 D:1 C:2 E:2

A → D

B → A A:1 C:1 E:1 B:0 D:2 F:2

Considera la seguente rete, che supporta un routing distance vector, con **poison reverse** abilitato.



Le regole del DV routing sono le seguenti:

1. L'ordine in cui ciascun router genera un DV message, è esattamente l'ordine numerico del router ID
2. L'ordine in cui ciascun router riceve un DV message, è esattamente l'ordine numerico del router ID
3. Nel caso in cui si usi poison reverse, il valore di costo impostato è 10.000
4. Non vengono riportati i messaggi che non alterano le routing tables dei vicini.

Scegliete la traccia corretta dei messaggi distance vector necessari per arrivare a convergenza, usando la sintassi conosciuta: From -> To: Dest:Cost; Dest:Cost ecc...

Scegli un'alternativa:

- a. 1 -> 2:1:0  
1 -> 3:1:0  
2 -> 11:10000; 2:0  
2 -> 3:1:1; 2:0  
3 -> 11:10000; 2:1; 3:0  
3 -> 2:1:1; 2:10000; 3:0
- b. 1 -> 2:1:0  
1 -> 3:1:0  
2 -> 11:10000; 2:0  
3 -> 11:10000; 3:0  
1 -> 2:1:0; 2:10000; 3:1  
1 -> 3:1:0; 2:1; 3:10000
- c. 1 -> 2:1:0  
1 -> 3:1:0  
2 -> 11:1:1; 2:0  
3 -> 11:1:1; 3:0  
1 -> 2:1:0; 2:1; 3:1  
1 -> 3:1:0; 2:1; 3:1
- d. 1 -> 3:1:0  
2 -> 3:2:0  
3 -> 11:10000; 2:1; 3:0  
3 -> 2:1:1; 2:10000; 3:0

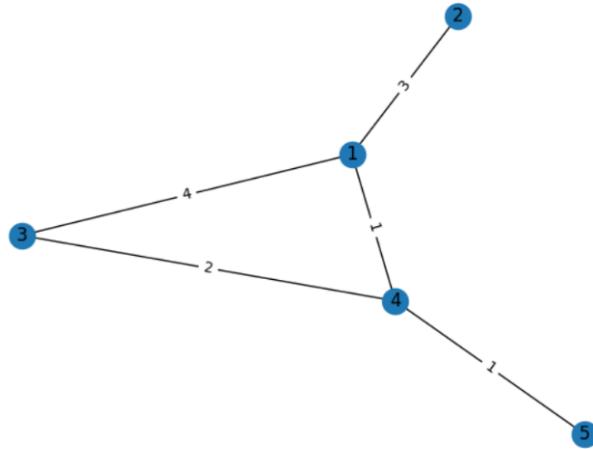
Risposta corretta.

Il router 2 non può inviare DV messages al router 3, perché non sono vicini, questo esclude due risposte.  
In una delle altre risposte non si usa poison reverse (ad esempio il router 2 non usa il valore 10000 quando invia al router 1 la rotta verso il router 1 stesso). Rimane solo una soluzione corretta.

b)

Considera la rete nella seguente figura, che supporta routing Distance Vector, e assumi che,

1. L'ordine in cui ciascun router genera un DV message, è esattamente l'ordine numerico del router ID
2. L'ordine in cui ciascun router riceve un DV message, è esattamente l'ordine numerico del router ID



Scrivi la lista dei messaggi generati nel corretto ordine e le routing table finali di ogni nodo. Puoi omettere i messaggi che non vanno ad alterare la routing table del ricevente.

La sintassi da utilizzare per scrivere i messaggi è la seguente:

**From -> To Dest:Cost; Dest:Cost; ...** (esempio del primo messaggio: 1 -> 2) 1:0)

Mentre quella utilizzata per le routing tables è:

**destinazione: next-hop, cost**

**La sequenza dei messaggi sarà:**

1 -> 2) 1:0  
1 -> 3) 1:0  
1 -> 4) 1:0  
2 -> 1) 1:3; 2:0  
3 -> 1) 1:4; 3:0  
3 -> 4) 1:4; 3:0  
4 -> 1) 1:1; 3:2; 4:0  
4 -> 3) 1:1; 3:2; 4:0  
4 -> 5) 1:1; 3:2; 4:0  
5 -> 4) 1:2; 3:3; 4:1; 5:0  
1 -> 2) 1:0; 2:3; 3:3; 4:1  
1 -> 3) 1:0; 2:3; 3:3; 4:1  
1 -> 4) 1:0; 2:3; 3:3; 4:1  
2 -> 1) 1:3; 2:0; 3:6; 4:4  
3 -> 4) 1:3; 2:7; 3:0; 4:2  
4 -> 1) 1:1; 2:4; 3:2; 4:0; 5:1  
4 -> 3) 1:1; 2:4; 3:2; 4:0; 5:1  
4 -> 5) 1:1; 2:4; 3:2; 4:0; 5:1  
5 -> 4) 1:2; 2:5; 3:3; 4:1; 5:0  
1 -> 2) 1:0; 2:3; 3:3; 4:1; 5:2  
1 -> 4) 1:0; 2:3; 3:3; 4:1; 5:2

**Tabelle di routing dopo la rottura dei link:**

*Nodo 1*

1: nh=1, cost=0  
2: nh=2, cost=3  
3: nh=2, cost=12  
4: nh=2, cost=13  
5: nh=2, cost=11

*Nodo 2*

2: nh=2, cost=0  
1: nh=1, cost=3  
3: nh=5, cost=9  
4: nh=5, cost=10  
5: nh=5, cost=8

*Nodo 3*

3: nh=3, cost=0  
1: nh=5, cost=12  
2: nh=5, cost=9  
4: nh=5, cost=3  
5: nh=5, cost=1

*Nodo 4*

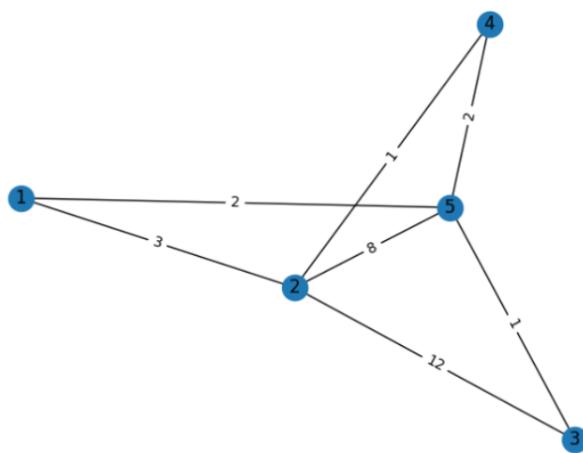
4: nh=4, cost=0  
1: nh=5, cost=13  
2: nh=5, cost=10  
3: nh=5, cost=3  
5: nh=5, cost=2

*Nodo 5*

5: nh=5, cost=0  
1: nh=2, cost=11  
2: nh=2, cost=8  
3: nh=3, cost=1  
4: nh=4, cost=2

Considera la seguente rete, che supporta un routing distance vector, con poison reverse abilitato. Le regole del DV routing sono le seguenti:

1. L'ordine in cui ciascun router genera un DV message, è esattamente l'ordine numerico del router ID
2. L'ordine in cui ciascun router riceve un DV message, è esattamente l'ordine numerico del router ID
3. Nel caso in cui si usi poison reverse, il valore di costo impostato è 10.000



Assumi che la rete sia arrivata a convergenza, a quel punto i link (1,5) e (2,4) smettono di funzionare. Descrivi le tabelle di routing per ciascun nodo, sia prima che dopo il guasto.

Le tabelle di routing sono nel formato:

destinazione: next-hop, cost

**Tabelle di routing prima la rottura dei link:**

*Nodo 1*  
1: nh=1, cost=0  
2: nh=2, cost=3  
3: nh=5, cost=3  
4: nh=5, cost=4  
5: nh=5, cost=2

*Nodo 2*  
2: nh=2, cost=0  
1: nh=1, cost=3  
3: nh=4, cost=4  
4: nh=4, cost=1  
5: nh=4, cost=3

*Nodo 3*  
3: nh=3, cost=0  
1: nh=5, cost=3  
2: nh=5, cost=4  
4: nh=5, cost=3  
5: nh=5, cost=1

*Nodo 4*  
4: nh=4, cost=0  
1: nh=2, cost=4  
2: nh=2, cost=1  
3: nh=5, cost=3  
5: nh=5, cost=2

*Nodo 5*  
5: nh=5, cost=0  
1: nh=1, cost=2  
2: nh=4, cost=3  
3: nh=3, cost=1  
4: nh=4, cost=2

**Tabelle di routing dopo la rottura dei link:**

*Nodo 1*  
1: nh=1, cost=0  
2: nh=2, cost=3  
3: nh=2, cost=12  
4: nh=2, cost=13  
5: nh=2, cost=11

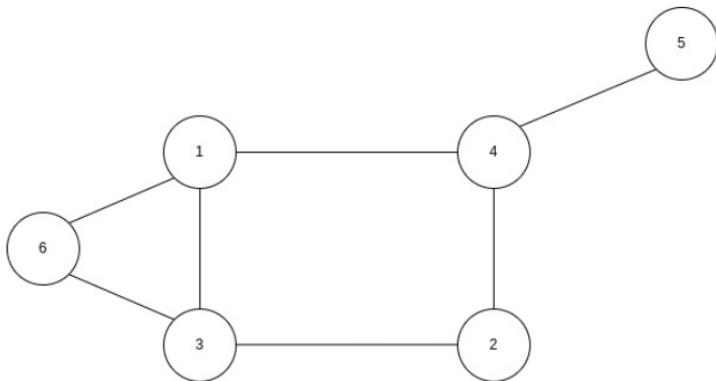
*Nodo 2*  
2: nh=2, cost=0  
1: nh=1, cost=3  
3: nh=5, cost=9  
4: nh=5, cost=10  
5: nh=5, cost=8

*Nodo 3*  
3: nh=3, cost=0  
1: nh=5, cost=12  
2: nh=5, cost=9  
4: nh=5, cost=3  
5: nh=5, cost=1

*Nodo 4*  
4: nh=4, cost=0  
1: nh=5, cost=13  
2: nh=5, cost=10  
3: nh=5, cost=3  
5: nh=5, cost=2

*Nodo 5*  
5: nh=5, cost=0  
1: nh=2, cost=11  
2: nh=2, cost=8  
3: nh=3, cost=1  
4: nh=4, cost=2

Considera la rete mostrata nelle figure sottostante:



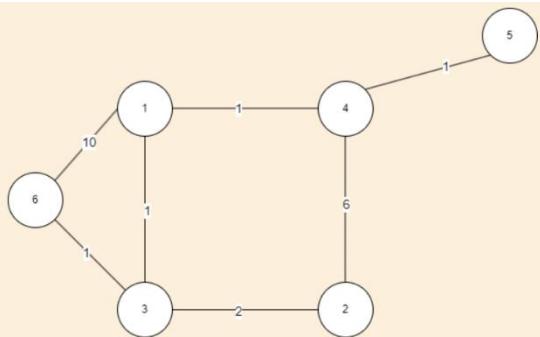
L'operatore che la gestisce vorrebbe che venissero usati i seguenti percorsi:

- $5 \rightarrow 4 \rightarrow 1 \rightarrow 3 \rightarrow 2$
- $6 \rightarrow 3 \rightarrow 1 \rightarrow 4$

Quale delle seguenti configurazioni dei pesi sugli archi garantisce che i percorsi proposti vengano rispettati sia che si usi Distance Vector o Link State?

Scegli un'alternativa:

- a.  $(6,1): 5, (6,3): 1, (1,4): 1, (4,5): 1, (2,4): 3, (3,2): 4, (3,1): 1$
- b.  $(6,1): 8, (6,3): 3, (1,4): 1, (4,5): 3, (2,4): 1, (3,2): 4, (3,1): 4$
- c.  $(6,1): 10, (6,3): 1, (1,4): 1, (4,5): 1, (2,4): 6, (3,2): 2, (3,1): 1$

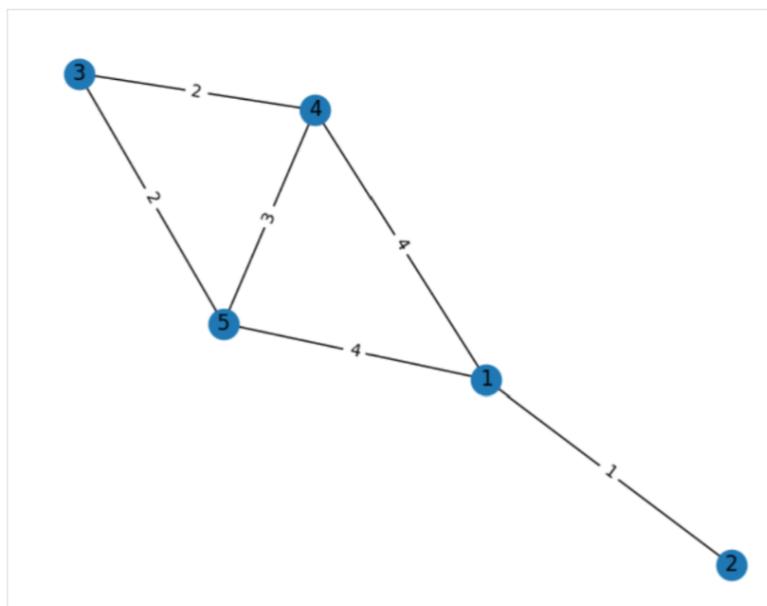


La risposta corretta è:

$(6,1): 10, (6,3): 1, (1,4): 1, (4,5): 1, (2,4): 6, (3,2): 2, (3,1): 1$

Considera la rete nella seguente figura, che supporta routing Distance Vector con Poison Reverse, e assumi che,

1. L'ordine in cui ciascun router genera un DV message, è esattamente l'ordine numerico del router ID
2. L'ordine in cui ciascun router riceve un DV message, è esattamente l'ordine numerico del router ID



Dopo aver raggiunto la convergenza, il link tra i nodi 4 e 5 subisce un guasto. Il protocollo Distance Vector in questa rete utilizza aggiornamenti periodici, ossia ogni terminale invia il proprio Distance Vector regolarmente, a intervalli di tempo prefissati. In caso di guasto su un collegamento della rete, i nodi direttamente connessi rilevano subito l'anomalia e aggiornano le proprie tabelle, ma informeranno gli altri nodi solo al momento del successivo invio del proprio Distance Vector. Al momento dell'invio dei DV, l'ordine seguito sarà sempre lo stesso (ovvero l'ordine numerico dei router ID). Quanti periodi servono perché le routing table del nodo 4 e 5 tornino a convergenza?

Scegli un'alternativa:

- a. 4
- b. 2
- c. 3
- d. 1

La rete converge dopo un round poiché solo i nodi 4 e 5 utilizzano il link (4,5), quindi nessun altro nodo è influenzato dal guasto di questo collegamento. All'inizio di un nuovo periodo, i nodi 3 e 1 inviano i loro DV, che includono i percorsi verso i nodi 4 e 5. Di conseguenza, i nodi 4 e 5 scelgono di comunicare tramite il nodo 3.

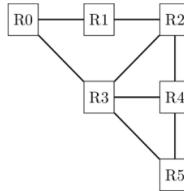
La risposta corretta è:

1

## Link State Routing

Collapse context

Consider the network shown in the figure below. Assuming that all link weights are set to 1, what is the link state packet that the routing protocol running on these routers will send once the network has converged? Remember that a link state packet contains a sequence number, an age and the cost to reach each direct neighbor.



Express your answer as a list fields separated by commas. For each field, use the <field>:<value> format, where <field> is the name of the field (use seq for sequence number and age for the age) and <value> the value of this specific field. If you need to specify a distance to a router, specify it as a pair that contains the router name as a field and the distance as the value.

Question 1: Link state packet for router R0

Provide the link state packet sent by router R0

Question 2: Link state packet for router R4

Provide the link state packet sent by router R4

### Router R0

Supponiamo che i vicini di R0 siano R1 e R3. Il LSP sarà quindi:

php Copia Modifica  
seq:<numero>, age:<valore>, R1:1, R3:1

### Router R4

Se i vicini di R4 sono R2, R3, e R5, il suo LSP sarà:

php Copia Modifica  
seq:<numero>, age:<valore>, R2:1, R3:1, R5:1

## Domande Aperte

- 1) pseudo codice di DV, spiegare count to infinity e spiegare BGP/differenze con BGP

### Sending DV ( pseudo code )

Periodicamente un router fa questo cosa qua'

```
1 Every N seconds:  
2     v = Vector()  
3     for d in R[]: Per ogni destinazione nella mia routing table  
4         # add destination d to vector  
5         v.add(Pair(d, R[d].cost)) Aggiungo una coppia che è destinazione e costo  
6     for i in interfaces  
7         # send vector v on this interface  
8         send(v, i) E questo lo invio a tutte le interfacce  
9
```

*(Note handwritten in the margin: Per ogni destinazione nella mia routing table, Aggiungo una coppia che è destinazione e costo (a quella destinazione pren data routing table))*

## Receiving DV (pseudo code)

*Ci sono due casi:*

```

1 # V : received Vector
2 # l : link over which vector is received
3 def received(V, l):
4     # received vector from link l
5     for d in V[]: Per ogni elemento del DV che ho ricevuto
6         if not (d in R[]): Se questa destinazione non sta nella mia Routing Table (è una destinazione non conosciuta)
7             # new route
8             R[d].link = l
9             R[d].cost = V[d].cost + l.cost
10            R[d].time = now()
11
12        else: Se trovo una destinazione nel mio DV che non fa parte della mia Routing Table
13            # existing route, should I update ?
14            if ((V[d].cost + l.cost) < R[d].cost) or (R[d].link == l): Controllo due cose:
15                R[d].link = l
16                R[d].cost = V[d].cost + l.cost
17                R[d].time = now()
    
```

*Ecco la spiegazione:*

- 1 - Se la destinazione ha un costo + il costo del link che è migliore di quello che conosco (e troviamo strada migliore aggiorniamo)
- 2 - Se mi manda questa informazione nuova è quella che già usavo come next-hop per quella destinazione.

line 13:

Se la destinazione era già conosciuta dal router, aggiorna la voce

## Generazione dei DV (variante della precedente)

```

1 Every N seconds:
2     for l in interfaces: Si cicla su tutte le mie interfacce (su tutti i link possibili)
3         # one vector for each interface
4         v = Vector() → per ognuna di queste io genero un vettore diverso
5         for d in R[]: Per tutte le destinazioni che sono nella mia R.T.
6             if (R[d].link != l): se la destinazione è diretta da l (dove l è l'interfaccia dove sto per
7                 v = v + Pair(d, R[d].cost) mandare quel DV)
8             else:
9                 v = v + Pair(d, infinity) costo infinito ma "che vuol dire: Non ti PASSARE
10                DAME."
11                send(v, l)
12            # end for d in R[]
13        # end for l in interfaces
    
```

Quando un collegamento si rompe in modo tale da partizionare la rete, i router di una partizione si inviano a vicenda i DV all'infinito, questo perché un router aggiorna la sua tabella sempre quando riceve un DV da un router adiacente per capire se ci sono cambiamenti nella rete.

Split horizon: dice che per un router l'insieme dei router con cui comunica non è tutto uguale, è diviso tra quei router che usa come next-hop, e tutti gli altri. Poison Reverse: il router sta avvelenando la tabella di routing del suo vicino, dicendogli che non ha un percorso verso una destinazione, se il percorso passa attraverso il vicino stesso.

È ancora possibile che avvenga un count-to-infinity, se il ricevimento di un DV avvelenato fallisce in un router allora può formarsi un count-to-infinity analogo a quello di prima.

Per risolvere il count-to-infinity i router non dovrebbero esportare solo la distanza, ma l'intero percorso verso la destinazione. Questo viene fatto nei protocolli di routing path-vector, come BGP.

# TRANSPORT LAYER

## Esercizi

### Exercises

Supponiamo che **MSL = 120 s** e che i numeri di sequenza siano a 32 bit.

- 1 - Qual è il throughput massimo **S** per garantire che non ci siano errori?
- 2 - È possibile avere due (o più) comunicazioni in corso tra la stessa coppia di host?
- 3 - Cosa avrebbe un impatto maggiore su Internet, cambiare il livello di trasporto o il livello di rete? Con impatto intendo, quante modifiche dovremmo fare per renderlo possibile.

#### 1. Throughput Massimo per Evitare Errori

Per garantire che non ci siano errori dovuti alla sovrapposizione dei numeri di sequenza, calcoliamo il throughput massimo **S** considerando:

- Un numero di sequenza a 32 bit si riavvolge ogni 4 GB trasmessi.
- Maximum Segment Lifetime (MSL) è di 120 secondi.

**Con un numero di sequenza a 32 bit,  $2^{32} \approx 4 \times 10^9$  byte  $\approx 4$  GB. Quindi, il riavvolgimento del numero di sequenza avviene ogni 4 GB trasmessi.**

Per evitare errori, il throughput massimo **S** deve essere tale che la quantità di dati trasmessi durante l'intervallo di tempo in cui il numero di sequenza può riavvolgersi non superi la capacità del numero di sequenza.

**Il throughput massimo S è:**

> Riavvolgimento ogni **4GB** trasmessi.

>  **$4 \times 10^9$  Byte** = 4 GB.

> 1 Byte = **8 bit**.

> MSL = 120 s.

$$S < (4 \times 10^9 \times 8) / 120 \approx (32 \times 10^9) / 120 \approx 266,666,666 \text{ bit/s} \approx 266 \text{ Mbps.}$$

Questo throughput è molto elevato per una connessione end-to-end, ma non impossibile. In pratica, l'MSL di 120 secondi è una stima conservativa; nella realtà, l'MSL è generalmente molto più basso. Pertanto, anche con velocità di trasmissione più alte, gli errori sono rari.

## 2. Comunicazioni Multiple tra la Stessa Coppia di Host

Sì, è possibile avere due (o più) comunicazioni simultanee tra la stessa coppia di host.

Questo viene realizzato utilizzando numeri di porta differenti per ciascuna connessione.

Ogni connessione TCP è identificata da una combinazione unica di indirizzo IP e numeri di porta per entrambi i lati della connessione, consentendo così la gestione di più connessioni parallele tra gli stessi host.

Sì, è possibile. Ogni connessione è identificata dagli indirizzi numerici degli endpoint e da due numeri di porta.

Ogni volta che una connessione viene avviata da un client, viene scelto un nuovo numero di porta di origine, quindi possono esserci più connessioni in corso perché gli endpoint possono distinguere i pacchetti in base a 4 parametri (2 indirizzi e 2 porte).

## 3. Impatto delle Modifiche sul Livello di Trasporto vs. Livello di Rete

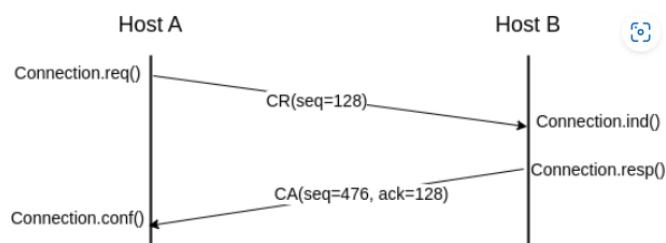
Cambiare il livello di trasporto di solito ha un impatto maggiore rispetto al livello di rete per i seguenti motivi:

- **Livello di Trasporto:** Modifiche al livello di trasporto (come TCP) richiedono aggiornamenti ai protocolli di comunicazione end-to-end e potrebbero richiedere modifiche alle applicazioni e ai sistemi software che utilizzano questi protocolli. Le modifiche devono essere propagate a tutte le applicazioni che dipendono dal protocollo di trasporto, il che può richiedere un ampio sforzo di aggiornamento.

- **Livello di Rete:** Modifiche al livello di rete (come IP) generalmente influenzano l'infrastruttura di rete, come i router e le tabelle di instradamento. Anche se le modifiche al livello di rete possono essere significative, spesso sono retrocompatibili con le implementazioni esistenti, il che può ridurre il numero di cambiamenti necessari a livello di applicazione.

In sintesi, **modificare il livello di trasporto impatta direttamente le comunicazioni applicative e richiede un aggiornamento esteso, mentre modificare il livello di rete tende a influenzare l'infrastruttura della rete e può essere gestito con modifiche meno invasive** se le nuove specifiche sono compatibili con quelle esistenti.

Il Three-Way Handshake consente di negoziare con successo l'instaurazione di una connessione al livello trasporto. Considera una connessione che inizia con lo scambio di due segmenti come mostrato nella figura sottostante.

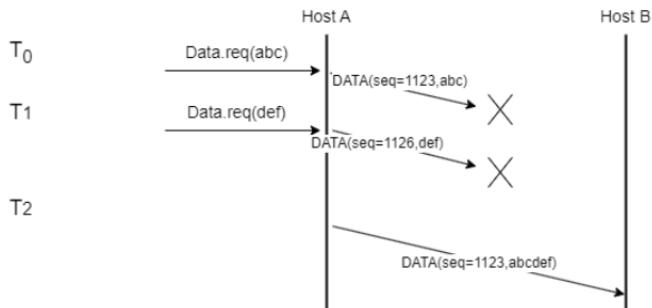


Gli eventi descritti in seguito avvengono in seguito a quelli descritti nella figura. Quale delle seguenti affermazioni è quella CORRETTA?

Scegli un'alternativa:

- a. Host B riceve `CR(seq=1432)`, risponderà con `CA(seq=234, ack=1432)`.
- b. Host A invia `CA(ACK=476)`, poi riceve `CA(seq=476, ack=128)`, lo scarterà come segmento duplicato.
- c. La connessione è stabilita e tutti i segmenti di tipo `CA` che arrivano devono essere considerati duplicati e scartati.
- d. Host B riceve un `CA(seq=476, ack=128)`, lo accetterà e l'handshake è concluso.

L'immagine sottostante rappresenta un possibile scambio di segmenti avvenuto durante una connessione a livello trasporto tra due terminali. Ciò che viene rappresentato riflette una corretta implementazione di un servizio affidabile orientato alla connessione di livello trasporto, basato su finestre? Scegli la risposta **CORRETTA**.



Scegli un'alternativa:

- a. No, perché il protocollo non permette di combinare segmenti ritrasmessi con dati nuovi, il sending buffer deve ritrasmettere esattamente ciò che è stato perso, evitando potenziali sovraccarichi nel receiving buffer.
- b. No, perché ciascun segmento perso deve essere ritrasmesso separatamente per rispettare il corretto ordine dei numeri di sequenza, altrimenti il receiving buffer potrebbe non gestire correttamente i dati.
- c. Sì, poiché in un protocollo orientato alla connessione affidabile, i dati persi possono essere ritrasmessi in un unico segmento più grande, combinando quelli già inviati e quelli mancanti, a condizione che il sending buffer li contenga ancora e che il receiving buffer li possa contenere.
- d. Sì, poiché la ritrasmissione dei dati persi può avvenire in un segmento di dimensioni maggiori per accelerare il processo, indipendentemente dalla capacità del receiving buffer di gestire l'intero blocco in una volta sola (i dati precedenti potrebbero già essere arrivati).

## SECURITY (OTP, DIEFFE-HELLMAN, RSA)

### Domande Teoria a Crocette

- ***“No repudiation” quando si verifica?***

#### 1. Crittografia a chiave pubblica e messaggi cifrati

Se Alice vuole inviare un messaggio cifrato a Bob:

Alice utilizza la chiave pubblica di Bob per cifrare il messaggio X.

Solo Bob, che possiede la chiave privata corrispondente, può decifrare il messaggio.

Questo processo garantisce confidenzialità, perché solo Bob può leggere il messaggio, ma non garantisce il non ripudio. Infatti, chiunque potrebbe utilizzare la chiave pubblica di Bob per cifrare un messaggio, e non c'è modo di provare che quel messaggio sia stato inviato da Alice.

#### 2. Come si garantisce il non ripudio

Per garantire il non ripudio, Alice deve firmare digitalmente il messaggio usando la sua chiave privata:

Alice firma il messaggio X con la sua chiave privata. Questo produce una "firma digitale" unica legata al messaggio.

Bob riceve il messaggio e la firma digitale.

Bob verifica la firma digitale utilizzando la chiave pubblica di Alice.

Se la verifica è valida:

Bob sa che il messaggio è autentico e che proviene da Alice, perché solo Alice possiede la chiave privata necessaria per generare quella firma.

Alice non può negare di aver inviato il messaggio, perché la firma digitale è legata unicamente alla sua chiave privata.

Quale delle seguenti affermazioni è **CORRETTA** riguardo alla cifratura a chiave pubblica e privata?

Scegli un'alternativa:

- a. Anche se una delle due chiavi viene rivelata, non è possibile derivare l'altra chiave, garantendo così la sicurezza del sistema
- b. Si usa solo per garantire l'autenticazione dei dati trasmessi, assicurando che il destinatario possa verificare l'identità del mittente
- c. L'operazione di cifratura deve essere più veloce rispetto a quella di decifratura per garantire un'efficace comunicazione
- d. Avendo a disposizione una delle due chiavi, è teoricamente impossibile generare la sua corrispettiva, poiché matematicamente scollegate
- e. La chiave pubblica deve essere mantenuta segreta, poiché la sua divulgazione potrebbe compromettere la sicurezza del sistema

La risposta corretta è la a:

Anche se una delle due chiavi viene rivelata, non è possibile derivare l'altra chiave, garantendo così la sicurezza del sistema

Perché viene utilizzato il **salt** durante l'hashing delle password?

Scegli un'alternativa:

- a. Il salt è utilizzato per rendere le Rainbow Table inefficaci, evitando che un attaccante possa utilizzare hash computati precedentemente
- b. Il salt garantisce che due password identiche inserite da utenti diversi generino lo stesso hash, semplificando la gestione del database delle password
- c. Il salt accelera il processo di hashing, riducendo il tempo necessario per controllare la validità delle password
- d. L'uso del salt garantisce che, in caso di attacco al database, le password rimangano completamente inaccessibili, indipendentemente dalla forza della password originale

Il **salt** è una stringa casuale aggiunta a una password prima di applicare una funzione di hashing.

### Come funziona il salt?

1. Un utente sceglie una password, ad esempio: "password123".
2. Si genera un salt casuale, es. "x4y7z!@".

3. Si concatena il salt alla password: "password123x4y7z!@".
4. Si calcola l'hash della stringa risultante, es. SHA-256("password123x4y7z!@").
5. L'hash viene salvato nel database **insieme al salt**.

Ogni utente avrà un salt diverso, quindi anche se due persone usano la stessa password, gli hash risultanti saranno diversi, rendendo inutili le Rainbow Table precomputate.

L'obiettivo principale del salt è di prevenire attacchi tramite Rainbow Table. Aggiungendo un salt unico per ogni password, gli hash risultanti diventano unici, anche se le stringhe di partenza sono identiche, rendendo inutile l'uso di tabelle computate precedentemente per decodificare gli hash.

La risposta corretta è:

Il salt è utilizzato per rendere le Rainbow Table inefficaci, evitando che un attaccante possa utilizzare hash computati precedentemente

Per quale motivo la cifratura a chiave pubblica è generalmente più lenta della cifratura simmetrica?

Scegli un'alternativa:

- a. Richiede più passaggi per cifrare i dati rispetto alla cifratura simmetrica
- b. Utilizza un numero minore di operazioni rispetto alla cifratura simmetrica, ma necessita di un canale sicuro per le chiavi
- c. Utilizza operazioni matematiche complesse, come l'esponenziazione modulare con numeri molto grandi
- d. Si basa su operazioni di XOR e permutazioni che sono più lente su grandi quantità di dati

La risposta corretta è:

Utilizza operazioni matematiche complesse, come l'esponenziazione modulare con numeri molto grandi

Alice desidera stabilire una comunicazione sicura con Bob, assicurando l'identità di Bob e che i messaggi scambiati tra di loro non possano essere letti da terze parti. Entrambi possiedono una coppia di chiavi, una pubblica ( $\text{Pub}_A$ ) e una privata ( $\text{Priv}_A$ ), firmate da due autorità di certificazione differenti,  $\text{CA}_Y$  e  $\text{CA}_X$ , che fanno però riferimento a una radice comune,  $\text{CA}_{\text{ROOT}}$ . Descrivi i passaggi che Alice e Bob devono compiere per arrivare ad implementare ciò che desiderano.

Il primo passo consiste nell'ottenere e verificare i certificati di Alice e Bob. Dopo aver ricevuto il certificato di Bob, Alice risale la cosiddetta *Chain of Trust* (catena di fiducia, la catena di certificati che va dalla chiave di Bob a  $\text{CA}_{\text{ROOT}}$ ) in modo ricorsivo, fino a trovare un certificato C firmato da  $\text{CA}_{\text{ROOT}}$  di cui si fida. Tutta la chain of trust deve essere fornita da Bob ad Alice. Alice utilizza quindi la chiave pubblica di  $\text{CA}_{\text{ROOT}}$  (che deve possedere) per verificare la validità di C e passa al

successivo nella catena. Alice compie questa operazione per tutti i certificati nella catena fino a raggiungere quello di Bob per poterlo verificare. Se tutto questo avviene correttamente, Alice, alla fine, ottiene  $\text{Pub}_B$  con la sicurezza che appartenga a Bob. Bob svolge le stesse operazioni per verificare la chiave pubblica di Alice. Una volta fatto tutto questo, Alice e Bob possono instaurare una chiave di sessione condivisa, che utilizzeranno per cifrare i dati in maniera efficiente e sicura. Possono utilizzare due metodi:

- *Diffie Hellman*: Alice e Bob possono generare una chiave di sessione condivisa tramite il protocollo Diffie-Hellman, firmando ciascun messaggio con la propria chiave privata. Questo metodo consente loro di stabilire una chiave condivisa senza trasmetterla direttamente sul canale, eliminando il rischio di un attacco MiTM
- *Scambio diretto della chiave di sessione*: In alternativa, uno dei due può generare una chiave di sessione e inviarla cifrata utilizzando la chiave pubblica dell'altro (ad esempio, Alice può cifrare la chiave di sessione con la chiave pubblica di Bob). Questo metodo richiede una fase premininare in cui vengono decisi i ruoli, e quindi un protocollo.

Dopo aver stabilito la chiave di sessione condivisa, Alice e Bob possono iniziare a cifrare i messaggi con questa chiave usando algoritmi a chiave simmetrica, garantendo così la riservatezza e l'integrità della loro comunicazione.

Quale è la differenza tra un security service e un security mechanism?

Scegli un'alternativa:

- a. I servizi di sicurezza necessitano di primitive di crittografia per essere realizzati, mentre un meccanismo di sicurezza (come la *availability*) dipende dalla configurazione e dal dimensionamento del sistema.
- b. Un servizio di sicurezza fornisce agli utenti del sistema alcune funzionalità di sicurezza di alto livello, ad esempio, la crittografia a chiave pubblica è un servizio di sicurezza. Un meccanismo di sicurezza è una funzione specifica che può essere utilizzata per ottenere un servizio, ad esempio l'integrità dei dati è un meccanismo di sicurezza.
- c. Un servizio di sicurezza è una proprietà di alto livello di un sistema, che viene implementata componendo vari meccanismi di sicurezza
- d. Un servizio di sicurezza richiede più risorse di un meccanismo di sicurezza, che invece, si basa su primitive crittografiche più semplici da implementare.

Un servizio di sicurezza è una proprietà di alto livello del vostro sistema, che viene ottenuta attraverso la composizione di meccanismi di sicurezza specifici. Un meccanismo si basa normalmente su primitive crittografiche che vengono utilizzate come building blocks per implementare il servizio stesso. Risposta c.

M è un messaggio, ENC() è una funzione di cifratura, MAC() è una funzione di Message Authentication Code, "+" è un operatore di concatenazione. Voendo inviare un messaggio cifrato ed autenticato con uno schema Encrypt then MAC, quale delle seguenti risposte descrive correttamente i dati inviati?

Scegli un'alternativa:

- a.  $\text{ENC}(M) + \text{MAC}(\text{ENC}(M))$
- b.  $\text{MAC}(\text{ENC}(M))$
- c.  $\text{ENC}(M) + \text{MAC}(M)$
- d.  $\text{ENC}(M + \text{MAC}(M))$

Lo schema Encrypt-then-MAC deve prima cifrare il messaggio M ottenendo  $C = \text{ENC}(M)$ , a quel punto calcolare il MAC sul messaggio cifrato  $H = \text{MAC}(C)$ , e poi concatenare entrambi e inviare sul canale. Uno schema MAC-then-encrypt invece invierà  $C + \text{MAC}(M)$ , oppure  $\text{ENC}(M + \text{MAC}(M))$  (nel secondo caso anche il MAC viene cifrato), e inviare solo il valore di ritorno della funzione MAC() invia solo un digest e non invia il dato cifrato. Risposta a.

## Esercizi

### Exercise

Considera la stringa :

"Caro Studente, il tuo voto finale è 18. Ci incontreremo in aula A432 alle 14:00 per registrare"

Il suo hash utilizzando la funzione di **hashing Md5 in formato esadecimale** è :

**hash = 3b3b28a9b694f5eed5f1221575a9cb4e.**

Supponiamo che il lettore sia pigro e controlli solo i primi e gli ultimi 3 caratteri: 3b3...b4e

Vuoi ingannarlo, modificando la stringa in modo che inizi con: Caro Studente, il tuo voto finale è 30...

1 - Quanti tentativi in media dovrà fare per avere successo?

2 - Puoi implementare il codice per rendere possibile questo?

**Hints :**

La stringa:

"Caro Studente, il tuo voto finale è 30, ci incontreremo in aula j356 alle 4:27 per registrare"

ha questo digest:

**Digest = 3b36ab65a0c6d089628a0628af124b4e**

Calcolare gli hash in Python è davvero facile:

```
1 import hashlib
2 digest = hashlib.md5("string to hash".encode('utf-8')).hexdigest()
```

Qui il codice C.

**Soluzione :**

Per affrontare l'esercizio, dobbiamo modificare la stringa iniziale in modo che il suo hash MD5, controllato solo nei primi e ultimi tre caratteri, combaci con quello desiderato. In particolare, vogliamo che la stringa inizi con "Caro Studente, il tuo voto finale è 30...", e quindi dobbiamo trovare un suffisso che faccia sì che l'hash MD5 della nuova stringa soddisfi il controllo.

**1 - Calcolare il numero medio di tentativi**

Per calcolare il numero medio di tentativi necessari, possiamo considerare la probabilità di successo in un tentativo e applicare il concetto di valore atteso.

**> Probabilità di successo in un tentativo:**

L'hash MD5 produce un digest di 32 caratteri esadecimali. Ogni carattere esadecimale ha 16 possibili valori (0-9 e a-f), quindi **ogni carattere ha 1/16 probabilità di essere corretto**. Poiché controlliamo solo i primi e gli ultimi 3 caratteri dell'hash, **stiamo cercando una corrispondenza per 6 caratteri in totale**.

**La probabilità di successo in un singolo tentativo per una stringa che genera un hash corretto in queste 6 posizioni è:**

$$P(\text{successo}) = (1 / 16)^6$$

Calcoliamo:

$$P(\text{successo}) = 1 / 16777216$$

Questo significa che in media **dovrai fare circa 16.777.216 tentativi per avere successo**.

Questo valore è il valore atteso, cioè il **numero medio di tentativi necessari**.

**2 - Implementare il codice in Python**

Per implementare un codice che trova una stringa che soddisfi il criterio, puoi utilizzare il seguente script Python. Il codice cerca di generare una stringa che, modificando l'ultima parte, produca un hash che corrisponde ai primi 3 e ultimi 3 caratteri specificati.

[Codice Python QUI](#).

## Esempio di utilizzo : Creazione delle chiavi

### 1. Scelta dei parametri:

Supponiamo che Alice e Bob abbiano concordato su:

> **n = 23** (un **numero primo**)

> **g = 5** (una radice primaria modulo 23)

### 2. Generazione delle chiavi private e pubbliche:

Alice sceglie una chiave privata segreta **a**:

> Supponiamo **a = 6**.

La chiave pubblica di Alice è:

**A = g^a mod n ==> 5^6 mod 23.**

Calcoliamo :

> **5^6 = 15625.**

> **15625 mod 23 = 8.**

Quindi, la **chiave pubblica di Alice è A = 8.**

---

Bob sceglie una chiave privata segreta **b**:

> Supponiamo **b = 15**

La chiave pubblica di Bob è:

**B = g^b mod n ==> 5^15 mod 23.**

Calcoliamo:

> **5^15= 30517578125**

> **30517578125 mod 23 = 15**

Quindi, la **chiave pubblica di Bob è B = 15.**

### 3. Scambio delle chiavi pubbliche:

Alice invia la sua chiave pubblica A a Bob.

Bob invia la sua chiave pubblica B ad Alice.

### 4. Calcolo della chiave segreta condivisa:

K' = Alice, K'' = Bob.

**Alice calcola la chiave segreta condivisa utilizzando la chiave pubblica di Bob B:**

**K' = B^a mod n = 15^6 mod 23**

Calcoliamo:

> **15^6 = 11390625**

> **11390625 mod 23 = 17**

Quindi, la chiave segreta calcolata da Alice è **K' = 17.**

---

**Bob calcola la chiave segreta condivisa utilizzando la chiave pubblica di Alice A:**

**K'' = A^b mod n ==> 8^15 mod 23**

Calcoliamo:

> **8^15 = 35184372088832**

> **35184372088832 mod 23 = 17**

Quindi, la chiave segreta calcolata da Bob è **K'' = 17.**

### 6. Sicurezza:

Se Eve intercetta i valori pubblici A = 8, B = 15, e conosce n = 23 e g = 5, **non può calcolare le chiavi private a e b a causa della difficoltà di calcolare i logaritmi discreti.** Quindi, non può ottenere la chiave segreta condivisa K.

**Conclusione :** Bob e Alice sono riusciti a generare una chiave segreta K = 17 che sia uguale per tutti e due e che non possa subire attacchi da Eve.

## Exercise

1 - Cosa è necessario per configurare una PKI e una relazione client-server?

Vai su [www.unive.it](http://www.unive.it), clicca sul lucchetto nel browser e rispondi alle seguenti domande:

2 - Qual è la lunghezza della catena fino alla root CA?

3 - Chi è la root CA?

4 - Qual è la lunghezza della chiave pubblica utilizzata da Unive?

5 - Qual è la funzione hash utilizzata per il certificato?

Ora, supponiamo che tu sia l'amministratore di sistema di Unive.

6 - Elenca tutte le chiavi necessarie per configurare il server web di [www.unive.it](http://www.unive.it).

7 - Per ogni voce nella domanda precedente, indica chi è il creatore di essa.

**1 - Per configurare una PKI e stabilire una relazione client-server sicura, sono necessari :**

**1. Certificati Digitali:**

> Certificato del server per autenticare e criptare le comunicazioni.

> (Opzionale) Certificato del client per l'autenticazione mutua.

**2. Chiavi Crittografiche:**

> Chiave privata e pubblica del server.

> (Opzionale) Chiave privata e pubblica del client.

**3. Autorità di Certificazione (CA):**

> Root CA e (eventualmente) CA intermedie per emettere e gestire i certificati.

**4. Infrastruttura di PKI:**

> Server di certificazione, database di certificati e sistema di revoca (CRL o OCSP).

**5. Politiche e Procedure:**

> Politiche di certificazione e procedure di sicurezza per la gestione dei certificati.

**6. Configurazione del Server e del Client:**

> Configurazione per utilizzare i certificati e le chiavi per stabilire connessioni sicure.

**7. Verifica e Testing:**

> Controllo della validità dei certificati e test della connessione sicura.

**2 - Lunghezza della catena fino alla root CA :**

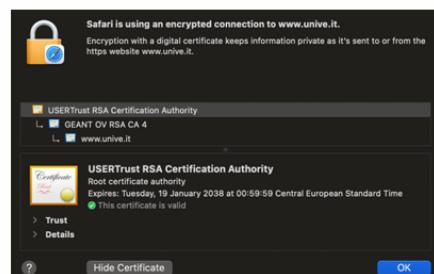
**La lunghezza della catena fino alla root CA è 3.**

La catena è composta da : certificato del server, certificato intermedio e certificato radice.

**Certificato del sito web ([www.unive.it](http://www.unive.it))** : Questo certificato appartiene al sito web specifico che stai visitando. Serve per dimostrare che il sito è autentico e sicuro per la navigazione.

**Certificato intermedio (GEANT OV RSA CA 4)**: Questo certificato collega il certificato del sito web al certificato radice. È un livello intermedio di fiducia che aiuta a verificare l'autenticità del certificato del sito.

**Certificato Radice (USERTrust RSA Certification Authority)** : Questo è il certificato principale emesso da un'autorità di certificazione riconosciuta. È il punto di fiducia finale su cui si basa l'intera catena di certificati.



### 3 - Chi è la root CA :

La root CA è **USERTrust RSA Certification Authority**,  
**Root certificate authority**.



### 4 - Lunghezza della chiave pubblica utilizzata da UniVe :

La chiave pubblica è la chiave che viene usata per la crittografia e la verifica delle firme (con la chiave pubblica non si decritta. La decrittografia viene effettuata esclusivamente con la chiave privata).

Key usages, usare la chiave pubblica per :

- > Encrypt (cifrare) : Cifrare i dati.
- > Verify (verificare) : Controllare firme digitali.
- > Wrap (avvolgere) : Cifrare altre chiavi.
- > Derive (derivare) : Creare altre chiavi.

**La lunghezza della chiave pubblica è di 2048 bit**, ossia 256 bytes ( $256 * 8 = 2048$ ).

### 5 - Funzione hash utilizzata per il certificato :

Ecco le funzioni hash utilizzate per ogni certificato :

#### 1. Certificato sito web :

Funzione Hash: **SHA-384** (usata per creare un riassunto dei dati)

Algoritmo di Firma: **RSA** (usato per firmare il riassunto creato con SHA-384)

- > **SHA-384 with RSA Encryption ( 1.2.840.113549.1.1.12 )**

#### 2. Certificato intermedio :

- > **SHA-384 with RSA Encryption ( 1.2.840.113549.1.1.12 )**

#### 3. Certificato root :

- > **SHA-384 with RSA Encryption ( 1.2.840.113549.1.1.12 )**

### 6 - Chiavi necessarie per configurare il server web di [www.unive.it](http://www.unive.it) :

Le chiavi necessarie per configurare il server web (HTTPS) di unive sono :

#### > Chiave privata

#### > Certificato del server (include chiave pubblica)

#### > Certificato intermedio

> **Certificato Root CA** (Generalmente non è necessario installarlo sul server, ma è utile per la verifica della catena di fiducia da parte dei client)

### 7 - Per ogni voce delle domande precedenti, indicare chi è il creatore :

**Certificato del sito web**: creazione di Unive (o dell'amministratore di sistema).

**Certificato intermedio**: creazione della Certification Authority intermedia.

**Certificato della root CA**: creazione della root CA stessa.

**Chiave pubblica Unive** : creata da UniVe stessa o dal suo amministratore di sistema.

**Funzione hash** : creata dal **National Institute of Standards and Technology (NIST)**. NIST ha sviluppato e standardizzato l'algoritmo SHA-384 come parte della famiglia di algoritmi Secure Hash Algorithm (SHA).

Si consideri un sistema crittografico RSA in cui la chiave pubblica (e,n) e quella privata (d,n) vengono creati a partire da: p=11, q=17, d=37, e=13. Quale delle seguenti frasi è FALSA?

Scegli un'alternativa:

- a. La cifratura di M=21 è C=21
- b. La firma di M=29 è S=7
- c. La cifratura di M=10 è C=164
- d. La firma di M=10 è S=142

Si calcola che  $n=p \cdot q=187$ . Svolgendo i calcoli si vede che  $29^{\wedge}37 \text{ mod } 187 = 105$  che ci dice la risposta sbagliata. È infatti vero che  $21^{\wedge}13 \text{ mod } 187 = 21$ , cioè che la cifratura di 21 è 21 stesso. Questo avviene perché la cifratura ritorna sempre un numero in modulo a n, quindi minore di n stesso e usando numeri piccoli, la probabilità di mappare un numero in se stesso è non trascurabile.

Utilizza l'algoritmo RSA per cifrare e decifrare il messaggio dato con i seguenti parametri:

- *Numeri primi*: p=19 e q=17
- *Esponente pubblico*: b=7
- *Messaggio da cifrare*: x=5

Calcola l'esponente privato "a" usando il [Teorema di Eulero](#) e mostra i passaggi per ottenere sia il messaggio cifrato sia quello decifrato.

Si consideri uno scambio di chiavi che usi il metodo di Diffie-Hellman con i seguenti parametri: a = 5, b = 3, n = 59, g = 3. Calcolare r, m, e la chiave k.

Scegli un'alternativa:

- a. r = 27, m = 7, k = 40
- b. r = 27, m = 6, k = 48
- c. r = 27, m = 7, k = 59
- d. r = 27, m = 7, k = 48

d)

Immagina di avere una rete di fiducia (Web of Trust) composta da sei utenti: A, B, C, D, E e F. Le relazioni di fiducia tra gli utenti sono le seguenti:

- A si fida di C, E
- B si fida di D, E
- C si fida di A, D
- D si fida di B, F
- E si fida di C, D
- F si fida di B, E

Ad un certo punto, A e F vogliono comunicare e per farlo devono scambiarsi le loro chiavi pubbliche, assicurando l'identità del loro proprietario. Tra le seguenti alternative, qual è una delle possibili catene che possono stabilire?

Scegli un'alternativa:

- a. A → D → F
- b. A → B → D → F
- c. A → C → D → F
- d. A → D → E → F

c) A->C->D->F X

RISPOSTA CORRETTA a. però probabilmente il prof ha sbagliato a scrivere e la soluzione è A->E->F

Un'azienda ha sviluppato una nuova funzione di hash che accetta in input solo messaggi ASCII di lunghezza arbitraria e restituisce un digest di 128 bit. Dopo la pubblicazione, Eve scopre una vulnerabilità nell'algoritmo e riesce a creare un oracolo. Questo, dato in input in testo ASCII e un digest target e una parola specifica, restituisce uno dei seguenti valori:

- **0** se e la parola non ha alcuna relazione con il digest target
- **1** se la parola è esattamente la stessa che ha generato il digest target
- **2** se la parola è una sottostringa del messaggio che ha generato il digest target

Quale dei seguenti metodi è il più efficiente per trovare il testo originale che genera un determinato digest, sfruttando l'oracolo di Eve? Immaginate uno pseudocodice (presente nella risposta).

Scegli un'alternativa:

- a. **Approccio ibrido:** Iniziare con un brute force per ogni singola lettera, trovare le lettere confermate come parte del messaggio, e poi usare l'oracolo per estendere una sottostringa. Complessità lineare con la lunghezza del testo.
- b. **Brute force sulle parole:** L'oracolo non è utile, il metodo migliore è provare ogni combinazione di parole possibili da un dizionario. Complessità esponenziale con il numero di parole nel testo.
- c. **Ricerca per sottostringhe:** Usare l'oracolo per rilevare le parole che sono sottostringhe del testo originale, e poi ricostruire il messaggio completo combinando tutte le sottostringhe trovate. Complessità cresce come il numero di combinazioni delle parole identificate.

L'approccio ibrido è il più efficiente perché combina il vantaggio di ridurre lo spazio di ricerca iniziale con un brute force sulle lettere e successivamente sfrutta l'oracolo per riconoscere le sottostringhe nel digest. Questo riduce il numero di tentativi rispetto al brute force diretto sulle parole o solo sulle lettere, permettendo a Eve di convergere più velocemente verso la parola originale.

Bob ha ideato un nuovo metodo di cifratura chiamato **BOB-CIPHER**, basato sull'operazione di XOR. Il metodo utilizza una chiave lunga 24 bit (generata precedentemente e scambiata prima di messaggio inviato su un canale sicuro) che viene XOR-ata con il testo da cifrare. Se il testo supera i 24 bit, viene prima fatto il padding (se necessario) per rendere il numero di byte contenuti, un multiplo di 3. Successivamente la chiave viene ripetuta su tutti i gruppi di 3 byte del testo in chiaro in modo da poter cifrare testi di lunghezza arbitraria.

Quale delle seguenti affermazioni è CORRETTA?

Scegli un'alternativa:

- a. La ripetizione della chiave per cifrare i gruppi di 3 byte di testo non introduce vulnerabilità, poiché l'operazione di XOR è sicura anche quando la chiave viene riutilizzata, garantendo la stessa sicurezza di una chiave lunga come il testo
- b. Il metodo di cifratura è sicuro anche per testi molto lunghi, poiché la ripetizione della chiave garantisce che la sicurezza non diminuisca con l'aumentare della lunghezza del messaggio
- c. Il cifrario è sicuro per testi fino a 24 bit.
- d. La chiave di 24 bit potrebbe risultare insufficiente per garantire la sicurezza su testi di qualsiasi lunghezza, poiché la ripetizione della stessa chiave su più blocchi di testo espone il cifrato a vulnerabilità

**PROTOCOLLI-----**

# DNS

## Domande Teoria a Crocette

Che cosa è un resource record di tipo PTR?

Scegli un'alternativa:

- a. È un resource record che viene aggiunto opzionalmente durante la registrazione del dominio per fare un reverse lookup, e permettere la risoluzione inversa da un IP a un dominio.
- b. È un resource record che viene utilizzato per specificare una chiave crittografica che verrà utilizzata per firmare le risposte del server DNS.
- c. È un resource record che viene aggiunto automaticamente durante la registrazione del dominio e permettere la risoluzione inversa da un dominio ad un IP.
- d. È un resource record che specifica quali indirizzi IP sono autorizzati ad inviare e-mail provenienti da quel dominio

Un RR PTR server a fare un reverse lookup, e quindi registra un dominio speciale, del tipo 1.2.3.4.in-addr.arpa associandolo al dominio in questione. È opzionale.

Il server DNS di amazon.com contiene un record SPF come il seguente:

```
spf2.0/prf ip4:207.171.160.0/19 ip4:87.238.80.0/21 ip4:72.21.192.0/19 ip4:52.119.213.144/28 -all
```

Come reagisce un server SMTP che riceve una connessione da un server con IP sorgente 52.119.213.212 e campo "MAIL From" con dominio amazon.com?

Scegli un'alternativa:

- a. SPF regola le connessioni in uscita da un server SMTP, non quelle in ingresso, quindi non si può dire
- b. L'email è ammessa
- c. L'email non viene ammessa
- d. La regola specifica che la decisione viene lasciata al server ricevente

La stringa indica quali range di indirizzi IP sono autorizzati ad inviare email usando il dominio amazon.com nel campo "MAIL From:". Tra questi c'è la rete 52.119.213.144/28 che contiene il range di indirizzi da 52.119.213.144 a 52.119.213.159 (144+16=160) che non contiene 52.119.213.212, quindi l'email non viene ammessa.

## Domande Aperte

- 2) Descrivere il funzionamento di DNS. Nella descrizione, seguire la traccia data dalle domande: come sono organizzati i nomi di dominio? Come avviene una risoluzione di un nome? Quali sono i vantaggi di usare un server DNS locale?
- 1) DNS per anti SPAM con 3 esempi, tra cui da menzionare DKIM (i tre esempi SPF FQDN DKIM e DMARC).

SPF SPF protegge EHLO eMAILFROM:Funzionaincombinazioneconilprotocollo DNS. Non protegge il campo From: Una entry SPF è una entry associata al nome di dominio di un'organizzazione che specifica quali sono gli indirizzi IP autorizzati a inviare e-mail. È una entry TXT. Ricorda che le entry TXT non hanno un formato, puoi scrivere qualsiasi cosa su di esse. Quando si ha una entry SPF la versione del protocollo è la versione 1-- un indirizzo IP è esplicitamente autorizzato a inviare e-mail tutti gli altri sono vietati

FQDNcheck È possibile che qualche MTA faccia la risoluzione inversa, quindi da dominio a ip (vedi dns). In questo modo viene controllato che l'ip con cui ti presenti sia lo stesso associato al dominio, se così non fosse è probabile che tu venga blacklisted e il messaggio droppato. È praticamente impossibile mettere su un MTA senza un IP pubblico associato ad un nome di dominio.

IP blacklisting Il server può fare la risoluzione di un nome di dominio e controllare se l'IP appartiene a una blacklist, e non accetterà mail da quell'IP. Se finisci in una blacklist è un macello uscirne, può capitare che se ti bucano il sito inizi a spammare mail a mano che i listatori blacklistano.

## DNS per Anti-SPAM: SPF, DKIM e DMARC

Il **Domain Name System (DNS)** viene utilizzato nei meccanismi anti-SPAM per verificare l'autenticità delle email e prevenire attacchi come il **phishing** e lo **spoofing**. Tre tecnologie chiave che sfruttano i record DNS sono **SPF**, **DKIM** e **DMARC**.

---

### 1) SPF (Sender Policy Framework)

- ◆ **Scopo:** Previene lo spoofing specificando quali server possono inviare email per un dominio.
- ◆ **Come funziona:**

- Il proprietario di un dominio pubblica un record **SPF** nel **DNS**, che elenca gli IP autorizzati a inviare email.
- Il server destinatario verifica se l'IP mittente è nella lista autorizzata.
- Se l'IP non è autorizzato, il messaggio può essere rifiutato o marcato come sospetto.

- ◆ **Esempio di record DNS SPF:**

```
v=spf1 ip4:192.168.1.1 ip4:192.168.2.2 include:altrodominio.com -all
```

 **Significato:**

- Solo gli IP **192.168.1.1** e **192.168.2.2** (più quelli autorizzati su `altrodominio.com`) possono inviare email per questo dominio.
- `-all` indica che tutte le altre email devono essere rifiutate.

---

### 2) DKIM (DomainKeys Identified Mail)

- ◆ **Scopo:** Garantisce che il contenuto dell'email non sia stato modificato durante il transito.

- ◆ **Come funziona:**

- Il server mittente firma l'email con una chiave privata.
- Il destinatario verifica la firma confrontandola con la chiave pubblica nel **DNS** del mittente.
- Se la firma è valida, l'email è considerata autentica.

- ◆ **Esempio di record DNS DKIM:**

`default._domainkey.esempio.com. TXT "v=DKIM1; k=rsa; p=MIGfMA0G..."`

 **Significato:**

- `v=DKIM1` → Specifica che il record è un DKIM di tipo 1.
  - `k=rsa` → Indica che viene usata una chiave RSA.
  - `p=MIGfMA0G...` → Contiene la chiave pubblica per la verifica.
- 

### 3) DMARC (Domain-based Message Authentication, Reporting & Conformance)

- ◆ **Scopo:** Definisce le policy per SPF e DKIM e fornisce report sul traffico email per un dominio.

- ◆ **Come funziona:**

- Il proprietario di un dominio pubblica un record **DMARC** nel **DNS**.
- I destinatari verificano se l'email passa SPF e DKIM.
- Se fallisce, seguono le istruzioni del record DMARC (ad es. rifiutare, segnalare o accettare l'email).

- ◆ **Esempio di record DNS DMARC:**

`_dmarc.esempio.com. TXT "v=DMARC1; p=reject; rua=mailto:report@esempio.com"`

 **Significato:**

- `v=DMARC1` → Specifica che è un record DMARC.
  - `p=reject` → Indica che le email non conformi devono essere rifiutate.
  - `rua=mailto:report@esempio.com` → Invia i report di analisi a `report@esempio.com`.
- 

#### **Riassunto delle Differenze**

Tecnologia	Scopo	Verifica	Azione
<b>SPF</b>	Previene spoofing	IP mittente	Blocca email da IP non autorizzati
<b>DKIM</b>	Garantisce integrità	Firma digitale	Verifica se il contenuto è stato alterato

Tecnologia	Scopo	Verifica	Azione
<b>DMARC</b>	Definisce policy e report	SPF + DKIM	Informa il destinatario su cosa fare con email sospette

Questi tre meccanismi, utilizzati insieme, migliorano la protezione del dominio e riducono la quantità di email di phishing e spam ricevute. 

## EMAIL

### Domande Aperte

1) Smtp, relay e esempi antispam

2) Spiegare il funzionamento congiunto di SPF e DKIM. Nella descrizione seguire la traccia data dalle seguenti domande:

- Quale problema risolve SPF? riportare un esempio di record TXT che usa la sintassi di SPF
- Quale problema risolve DKIM?
- Perché c'è bisogno di entrambi i protocolli?

## WWW (HTTP)

### Domande Aperte

HTML-----

1) spiegare HTML, i tag più usati, cosa sono codice client-side e server side e cos'è css

#### HTML (HyperText Markup Language)

HTML è il linguaggio di markup utilizzato per creare le pagine web. Non è un linguaggio di programmazione, ma serve a strutturare il contenuto delle pagine tramite tag. Ogni elemento HTML è racchiuso tra tag che ne definiscono il ruolo (titoli, paragrafi, immagini, link, ecc.).

#### Tag più usati in HTML

- <html>: indica l'inizio e la fine di una pagina web.
- <head>: contiene informazioni sulla pagina, come il titolo e i link ai fogli di stile.
- <title>: definisce il titolo della pagina visualizzato nella scheda del browser.
- <body>: contiene il contenuto principale della pagina.
- <h1> - <h6>: rappresentano i titoli (dal più grande <h1> al più piccolo <h6>).
- <p>: definisce un paragrafo di testo.
- <a href="url">: crea un collegamento ipertestuale.

- : inserisce un'immagine.
  - <ul>, <ol>, <li>: creano liste non ordinate e ordinate.
  - <table>: crea tabelle per visualizzare dati in formato tabellare.
  - <div> e <span>: usati per il raggruppamento e la formattazione di elementi.
- 

## Codice Client-Side e Server-Side

- **Client-Side:** si riferisce a codice eseguito dal browser dell'utente, come HTML, CSS e JavaScript. Ad esempio, JavaScript può essere usato per rendere dinamica una pagina, gestire eventi e validare moduli senza bisogno di interagire con il server.
  - **Server-Side:** riguarda codice eseguito sul server prima di inviare la risposta al client. Esempi di linguaggi server-side sono PHP, Python (Django, Flask), Node.js e Java (Spring). Viene usato per elaborare dati, accedere a database e generare pagine dinamiche.
- 

## CSS (Cascading Style Sheets)

CSS è un linguaggio di stile che definisce l'aspetto grafico delle pagine web. Permette di personalizzare colori, font, margini, layout e altri aspetti visivi degli elementi HTML. Si può applicare in tre modi:

1. **Inline CSS:** direttamente nell'attributo style di un elemento (<p style="color:red;"> Testo Rosso </p>).
2. **Internal CSS:** all'interno del tag <style> nella sezione <head> di un documento HTML.
3. **External CSS:** con un file esterno .css, collegato tramite <link rel="stylesheet" href="style.css">.

L'uso di CSS migliora la separazione tra contenuto e presentazione, rendendo il codice più pulito e manutenibile.

Questa è una risposta ben strutturata per la domanda d'esame. Se vuoi approfondimenti su qualche punto, chiedimi pure! 😊

# TRANSPORT LAYER (UDP, TCP)

## Domande Teoria a Crocette

UDP e TCP hanno checksum??

Quando TCP osserva un evento di *mild congestion*, come si comporta?

Scegli un'alternativa:

- a. La crescita esponenziale della cwnd è interrotta, la finestra viene riportata al valore minimo la crescita prosegue esponenzialmente
- b. La crescita esponenziale della cwnd è interrotta, la finestra viene ridotta di metà la crescita prosegue linearmente
- c. La finestra viene ridotta di metà e la crescita prosegue esponenzialmente
- d. La crescita esponenziale della cwnd è interrotta, la finestra viene riportata al valore minimo la crescita prosegue linearmente

Un evento di *mild congestion* è meno grave di uno di *severe congestion*. Accadono due cose, la crescita della finestra viene interrotta, e la finestra viene resettata a metà del suo valore. Per evitare che la finestra arrivi molto velocemente al valore che ha creato la congestione, la crescita da quel momento non è più esponenziale ma solo lineare.

Che differenza c'è tra una finestra di invio ed una finestra di congestione?

Scegli un'alternativa:

- a. La prima è dinamica e viene scalata sulla base del campo window dell'header TCP, la seconda è fissa.
- b. La prima serve a mantenere in memoria dati che non hanno ancora ricevuto un ACK, la seconda serve a non sovraccaricare i router sul cammino.
- c. La prima viene influenzata dall'opzione "scaling window", la seconda viene comunicata dal destinatario ad inizio connessione.
- d. La prima serve a non sovraccaricare il buffer di ricezione del destinatario, la seconda serve a al destinatario per mantenere i dati prima di inviarli all'applicazione.

La sending window serve salvare i dati che devono ancora essere confermati, nel caso debbano essere reinviati. la seconda viene usata per evitare di inviare dati con un bitrate che potrebbe portare alla congestione i router intermedi, e viene decisa dinamicamente da chi invia i dati sulla base dei segmenti persi.

## Esercizi

### Exercise

#### Consegna

##### 1. Memoria e Allocazione:

- > Il server dispone di 32 GB di memoria totale.
- > Di questi, 16 GB sono allocabili per il TCB (Transmission Control Block).

##### 2. Connessioni e Durata:

- > Il server riceve X connessioni al secondo.

> Ogni connessione dura z secondi, dove z è molto breve e trascurabile rispetto al Maximum Segment Lifetime (MSL).

- > Dopo z secondi, le connessioni terminano.

##### 3. Determinare il Numero di Connessioni:

> Calcolare quante connessioni al secondo il server può accettare senza superare la memoria allocata per il TCB.

##### 4. Dimensione del TCB:

- > Verificare la dimensione del TCB e il suo utilizzo della memoria.

Su Linux, è possibile ottenere le informazioni sul buffer di ricezione con il comando:

```
> cat /proc/sys/net/ipv4/tcp_rmem
```

I valori ottenuti sono:

- > Minimo: 4096 byte

- > Predefinito: 131072 byte

- > Massimo: 6291456 byte

### Soluzione

##### 1. Determinare la Dimensione di un TCB:

> La dimensione di un TCB è principalmente influenzata dalla dimensione del buffer di ricezione, che nel caso predefinito è di 131072 byte.

##### 2. Calcolare la Memoria Occupata per Connessione:

> Ogni connessione utilizza un buffer di ricezione di 131072 byte (o 128 KB) come valore predefinito.

##### 3. Calcolare la Memoria Totale Disponibile per il TCB:

> La memoria allocabile per il TCB è di 16 GB, che corrisponde a  $2^{34}$  byte.

##### 4. Calcolare il Numero Massimo di Connessioni:

> Per evitare di superare la memoria allocata, dobbiamo calcolare il numero massimo di connessioni che possono essere gestite contemporaneamente.

La formula per determinare il numero massimo di connessioni X è:

>  $16\text{ GB} = 2^{34}\text{ byte}$

Memoria per connessione = 131072 byte

$$X * 131072 \text{ byte} * 4 * 60 \text{ secondi} \leq 2^{34} \text{ byte}$$

Semplificando:

$$X \leq (2^{34}) / (131072 * 4 * 60)$$

$$X \approx 546$$

### Risultato

Il server può accettare fino a **546 connessioni al secondo**, assumendo che ogni connessione utilizzi il valore predefinito del buffer di ricezione e che ogni connessione duri solo z secondi.

## Esempio :

> Supponiamo di trasferire un file da 10 MB tra due computer utilizzando una connessione TCP. La dimensione massima di un segmento (MSS, Maximum Segment Size) è di 1 KB, la dimensione della finestra di invio (swnd) è di 1 MB e il tempo di andata e ritorno (RTT) è di 100 millisecondi (0,1 secondi).

### > Dati :

File da trasferire = 10 MB.

MSS = 1 KB.

swnd size = 1 MB.

RTT = 100 millisecondi ==> 0,1 secondi.

> Al tempo t0, il mittente, a causa dell'algoritmo di Nagle, invia tutti i segmenti di dimensione MSS che la finestra di invio (swnd) permette. Nel nostro esempio, la finestra di invio permette 1 MB, quindi il mittente può inviare 1 MB di dati. Poiché ogni segmento è di 1 KB, **il mittente invierà 1024 segmenti da 1 KB ciascuno** tutti insieme **al tempo t0**.

> **I segmenti vengono ricevuti al tempo t0 + RTT/2**. Supponendo che la capacità del collegamento sia molto alta e che non ci siano perdite di segmenti, i 1024 segmenti da 1 KB ciascuno arrivano al destinatario dopo metà del tempo di andata e ritorno (RTT/2), **quindi al tempo t0 + 0,05 secondi**.

> Dopo aver ricevuto tutti i segmenti, il destinatario invia un ACK cumulativo per confermare la ricezione di tutti i segmenti fino a quel punto. Questo ACK viaggia verso il mittente.

> L'ACK cumulativo viene ricevuto al tempo t0 + RTT. L'ACK cumulativo impiega il resto dell'RTT per tornare al mittente. Pertanto, il mittente **riceve l'ACK al tempo t0 + 0,1 secondi**.

> Quando il mittente riceve l'ACK al tempo t0 + RTT, il buffer di invio viene liberato, permettendo al **mittente di inviare un altro blocco di dati pari a 1 MB = 1024 KB**.

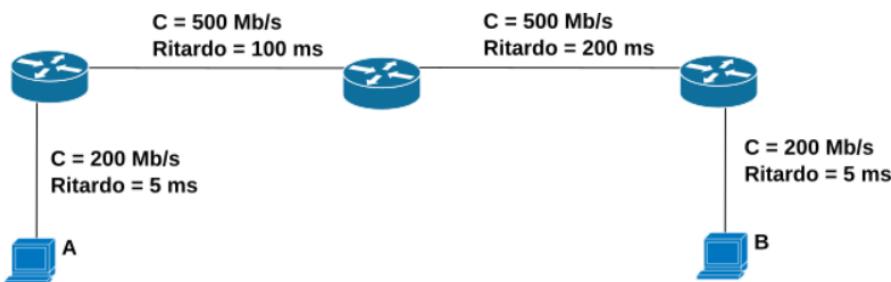
> **Per ogni RTT, il mittente invia un massimo di 1 MB di dati**. Quindi, il throughput massimo è dato dalla **dimensione della finestra divisa per RTT**:

Res = window size / RTT ==> 1 MB / 0,1 s = **10 MB/s**.

> Dato che il throughput massimo è di 10 MB/s e stiamo **trasferendo un file da 10 MB, il tempo totale per il trasferimento del file sarà la dimensione del file divisa per il throughput massimo**, quindi 10 MB diviso per 10 MB/s, che risulta in **1 secondo**.

> **Grazie all'algoritmo di Nagle, i dati vengono inviati in blocchi grandi quanto la finestra di invio**, e poiché l'ACK viene ricevuto ogni RTT, il throughput massimo (window/RTT) viene raggiunto fin dall'inizio della connessione TCP, rendendo il trasferimento del file efficiente.

Nella rete in figura dal terminale A al terminale B viene instaurata una connessione TCP. Chiamiamo S lo scaling factor della finestra di congestione impostato nella opzione TCP Scale Window e chiamiamo T il throughput massimo raggiungibile dalla connessione TCP. Quale delle seguenti risposte è vera?



Scegli un'alternativa:

- a. Se S=5, allora T=200 Mb/s
- b. Se S=14, T=13845 Mb/s
- c. Se S=0, allora T è minore di 1 Mb/s
- d. Se S=13 T è più grande che se S=9

Il throughput massimo in b/s raggiungibile da una connessione TCP è dato dalla larghezza della finestra di invio (in bit), diviso il RTT (in secondi). La finestra di invio è espressa in byte ed è larga al massimo  $2^{(16+S)}$  dove S è lo scaling factor. Una volta stimato il throughput massimo teorico della connessione, questo sarà comunque limitato dalla capacità del link più veloce, che in questo problema è 200 Mb/s. Il RTT è dato dalla somma dei ritardi, moltiplicato per due, quindi 0.620s. Si ottiene che con S=0, T=0.84 Mb/s, con S=13, S=14 e S=9, T=200Mb/s, con S=6, T=27 Mb/s.

b) è la risposta corretta.

All'inizio di una connessione TCP il terminale misura tre valori di RTT come segue: 64, 48, 128 (ms). Calcolare la stima di SRTT dopo la terza misura utilizzando l'algoritmo di Van Jacobson ( $\alpha = 1/8$ ). La prima stima di SRTT sarà data dal primo campione misurato. Si consideri RTTVar sia fissato a 48 (senza bisogno di calcolarlo), quale è il valore scelto di RTO? (nota che ci può essere un piccolo errore di arrotondamento, meno del 2%)

Scegli un'alternativa:

- a. SRTT = 70.25; RTO = 262.25
- b. SRTT = 61.90; RTO = 256.37
- c. SRTT = 64.37; RTO = 262.25
- d. SRTT = 94.34; RTO = 286.34

L'algoritmo di VJ calcola RTO come

$$RTO = SRTT + 4 * RTTVar$$

Ed SRTT è aggiornato nel seguente modo:

$$SRTT_i = (1-\alpha)SRTT_{i-1} + \alpha * RTT_i$$

Ovvero la stima di SRTT al campione i si calcola come  $(1-\alpha)$  per la stima al campione precedente, più  $\alpha$  per il valore di RTT misurato al campione i.

Abbiamo quindi:

$$SRTT_0 = 64$$

$$SRTT_1 = (7/8) * 64 + 48/8 = 62$$

$$SRTT_2 = (7/8) * 62 + 128/8 = 70.25$$

$$\text{Dato } RTTVar = 48,$$

$$RTO = 70.25 + 4 * 48 = 262.25$$

## Domande Aperte

Descrivere le chiamate più rilevanti che sono necessarie per aprire un TCP socket nello standard POSIX (BSD socket). Non è necessario enumerare in dettaglio tutti i parametri delle call, ma è necessario descriverne il funzionamento generale. Usare la seguente traccia:

- descrivere le chiamate più importanti per un server
- le differenze con un client

## Congestion window-----

### 1) Spiegare slow start e finestra di congestione:

- Come si relazione la finestra di congestione con le altre;
- Spiegare Mild e Severe;
- Come cambia la finestra con Mild e Severe.

introduciamo un'altra finestra, la cosiddetta finestra di congestione (cwnd). Cwnd è inizializzata a qualche valore fisso, inizialmente era cwnd0 =MSS ora è cwnd0=10\*MSS.

In ogni momento abbiamo window = min(cwnd, swnd, rwnd).

### 2) Algoritmo di Nagle

#### Algoritmo di Nagle

```
1 if rcv.wnd >= MSS and len(data) >= MSS: controllo se questo dato è maggiore del MSS (se effettivamente comprende tutto il MSS)
2   send one MSS-sized segment Allora posso inviare. Te manda tutto e appelli noti parole di parate tante cose velocemente
3 else: ABBIAVO DATI (nonabbastanza dati per comporre un MSS)
4   if there are unacknowledged data: ho dati da inviare ACK???
5     place data in buffer until acknowledgment has been received Allora appello che mi arrivi l'ACK
6   else: non ho nessun dato nel buffer che sto aspettando ACK?
7     send one TCP segment containing at most rcv.wnd data allora manda
```

L'idea 4-5)

## NETWORK LAYER (IPV4, IPV6, ICMP, OSPF)

### Domande Teoria a Crocette

Quali router sono contenuti nell'area zero di OSPF?

Scegli un'alternativa:

- a. I gateway che collegano la rete ad internet.
- b. I router di bordo di tutte le aree ed i router che non fanno parte di nessuna area
- c. I router che non sono "designated"
- d. I router che non usano BGP

I router di bordo delle varie aree e quelli che servono a collegarli, anche se non fanno parte di un'area. B.

## Esercizi

## Exercises

### Let's rewrite the table in binary

Prefix Match	Destination Network (red bits = netid)	Out NIC
?	00000000.00000000.00000000.00000000	0
?	01111100.10011100.00001100.00000000	1
?	01111100.10011100.00001101.00000000	0
?	01111100.10011100.00001101.10000000	2
?	11001000.11010100.00001100.01111111	3
?	11001000.11010100.00001100.01000000	4

Consideriamo :

124.156.12.12 == 01111100.10011100.00001100.00001100

Cerco la maggior corrispondenza di bit tra quelli della table.

### Example 1

Quindi, l'uscita è NIC = 1, perché ci sono due righe che corrispondono, ma sceglio quella con il prefisso di corrispondenza più lungo.

Prefix Match	Destination Network (red bits = netid)	Out NIC
✓	00000000.00000000.00000000.00000000	0
✗	01111100.10011100.00001100.00000000	1
✓	01111100.10011100.00001101.00000000	0
✗	01111100.10011100.00001101.10000000	2
✗	11001000.11010100.00001100.01111111	3
✗	11001000.11010100.00001100.01000000	4

### Example 2

Consideriamo :

124.156.13.2 == 01111100.10011100.00001101.00000010

Quindi l'uscita è NIC = 0.

Prefix Match	Destination Network (red bits = netid)	Out NIC
✓	00000000.00000000.00000000.00000000	0
✓	01111100.10011100.00001100.00000000	1
✗	01111100.10011100.00001101.00000000	0
✗	01111100.10011100.00001101.10000000	2
✗	11001000.11010100.00001100.01111111	3
✗	11001000.11010100.00001100.01000000	4

### Example 3

Consideriamo :

124.156.13.129 == 01111100.10011100.00001101.10000001

Quindi l'uscita è NIC = 2.

Prefix Match	Destination Network (red bits = netid)	Out NIC
✓	00000000.00000000.00000000.00000000	0
✗	01111100.10011100.00001100.00000000	1
✗	01111100.10011100.00001101.00000000	0
✗	01111100.10011100.00001101.10000000	2
✗	11001000.11010100.00001100.01111111	3
✓	11001000.11010100.00001100.01000000	4

### Example 4

Consideriamo :

200.212.12.79 == 11001000.11010100.00001100.01001111

Quindi l'uscita è NIC = 4.

### Example 5

Consideriamo :

124.156.12.35 == 11001000.11010100.00001100.00100011

Quindi l'uscita è NIC = 0.

Prefix Match	Destination Network (red bits = netid)	Out NIC
✓	00000000.00000000.00000000.00000000	0
✗	01111100.10011100.00001100.00000000	1
✗	01111100.10011100.00001101.00000000	0
✗	01111100.10011100.00001101.10000000	2
✗	11001000.11010100.00001100.01111111	3
✗	11001000.11010100.00001100.01000000	4

Prefix Match	Destination Network (red bits = netid)	Out NIC
✓	00000000.00000000.00000000.00000000	0
✗	01111100.10011100.00001100.00000000	1
✗	01111100.10011100.00001101.00000000	0
✗	01111100.10011100.00001101.10000000	2
✗	11001000.11010100.00001100.01111111	3
✗	11001000.11010100.00001100.01000000	4

### Example 6

Consideriamo :

200.156.12.1 == 11001000.10011100.00001100.00000001

Quindi l'uscita è NIC = 0.

## IP Addressing : Exercises 2

### Exercise n. 0

Possiedo una subnet 1.2.1.0/24. Posso assegnare a uno dei miei host l'indirizzo 1.2.1.255?

> NO! Il netid è 1.2.1, e l'indirizzo 1.2.1.255 è riservato per il broadcast della subnet.

Possiedo una subnet 1.2.0.0/16. Posso assegnare a uno dei miei host l'indirizzo 1.2.1.0?

> YES! Il netid è 1.2, e l'indirizzo 1.2.0.0 (con tutti gli zeri nell'hostid) è l'indirizzo di rete, quindi 1.2.1.0 è un indirizzo IP valido.

Possiedo una subnet 1.2.1.0/24. Posso assegnare a uno dei miei host l'indirizzo 1.2.1.0?

> NO! Il netid è 1.2.1 e l'indirizzo 1.2.1.0 (con tutti gli zeri nell'hostid) è riservato per l'indirizzo di rete della subnet.

### Exercise n. 1

#### Consegna :

Un'organizzazione ha bisogno di 15 indirizzi IP e deve affittarli da un fornitore di rete.

Supponiamo che il fornitore di rete possieda 1.2.0.0/16.

Qual è la lunghezza minima del prefisso di rete che l'organizzazione deve affittare?

Assegna gli indirizzi di rete, iniziando la numerazione da "tutti zeri", ad esempio, se hai bisogno di un /24, dovresti usare 1.2.0.0/24 (terzo byte impostato a 0).

Indica specificamente l'intervallo di indirizzi consentiti.

#### Soluzione :

Per contenere 15 indirizzi, sono necessari 4 bit, quindi 1.2.0.0/28 sarebbe sufficiente: dall'indirizzo 1.2.0.0 a 1.2.0.15 (16 IP, di cui 15 saranno effettivamente utilizzati).

Tuttavia, ogni subnet ha due indirizzi che non possono essere utilizzati, in questo caso: 1.2.0.0 (indirizzo di rete) e 1.2.0.15 (15 == 00001111) (indirizzo di broadcast).

Pertanto, l'unico modo per avere 15 indirizzi utilizzabili è affittare 1.2.0.0/27, che fornisce 30 indirizzi utilizzabili, di cui solo 15 saranno effettivamente usati.

Gli indirizzi disponibili andranno da 1.2.0.0 a 1.2.0.31, dei quali 30 sono utilizzabili (hostid da 1 a 30).

### Exercise n. 2

#### Consegna :

Un'organizzazione ha due sezioni, A e B. Nella rete A ci sono 14 host, nella rete B ci sono 16. Il fornitore possiede 1.2.0.0/16. Trova:

> La maschera di rete minima che l'organizzazione deve affittare dal fornitore.

> Una possibile allocazione e intervalli, assumendo "tutti 1" per il net id.

> Le tabelle di routing del router connesso a Internet.

#### Soluzione :

Il numero totale di indirizzi IP richiesti è 30, quindi una maschera di rete /27 è sufficiente.

Tuttavia, l'amministratore vuole suddividere la rete in due.

Ogni subnet spreca 2 indirizzi IP, quindi ha bisogno di  $30+2 \times 2=34$  indirizzi.

Questo richiede una rete /26, che offre 64 indirizzi di cui ne utilizzerà solo 30.

Affitta 1.2.255.192/26 (192 == 11000000) e la divide in due sottoreti:

> 1.2.255.192/27 (192 == 11000000) e 1.2.255.224/27 (224 == 11100000).



1.2.255.192/27 (192 == 11000000): Questo fornirà indirizzi da 1.2.255.192 a 1.2.255.223. Questi sono 32 indirizzi, dei quali il primo e l'ultimo non possono essere utilizzati.

1.2.255.224/27 (224 == 11100000): Questo fornirà indirizzi da 1.2.255.224 a 1.2.255.255. Questi sono 32 indirizzi possibili, dei quali il primo e l'ultimo non possono essere utilizzati.

The **FIB** sarà :

Destination Network	Out NIC	
0.0.0.0/0	eth3	==> vedi anche immagine a inizio esercizio
1.2.255.192/27	eth1	
1.2.255.224/27	eth0	

Tuttavia, gli indirizzi IP hanno un costo e l'amministratore non è soddisfatto di affittare 64 indirizzi e usarne solo 30. Esiste una soluzione migliore?

Una delle reti si adatta a una subnet /28 e l'altra a una /27.

Quindi potrebbe affittare due sottoreti diverse:

> 1.2.255.192/27 (240 == 11000000): Questo fornirà indirizzi da 1.2.255.192 a 1.2.255.223. Questi sono 30 indirizzi utilizzabili per la rete B.

> 1.2.255.224/28 (224 == 11100000): Questo fornirà indirizzi da 1.2.255.224 a 1.2.255.239. Questi sono 14 indirizzi utilizzabili per la rete A.

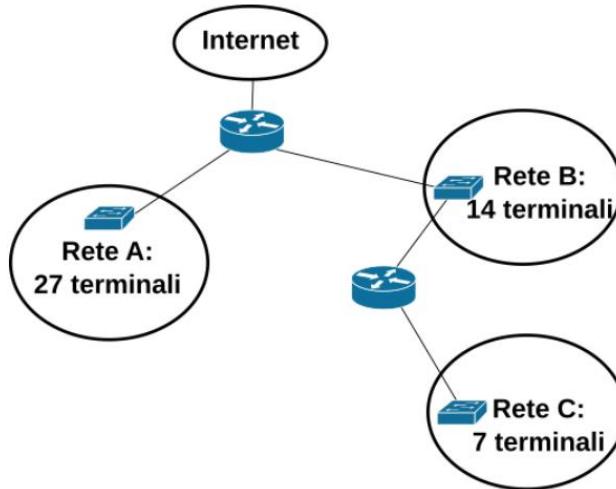
L'amministratore possiede ancora più indirizzi di quanti ne servano (32+16=48), ma pagherà meno rispetto a prima.

Tuttavia, possiede due sottoreti pubbliche, e non solo una, poiché non copre 1.2.255.240/28 (240 == 11110000).

La FIB sarà:	Destination Network	Out NIC
	0.0.0.0/0	eth3
	1.2.255.192/27	eth1
	1.2.255.224/28	eth0

Quindi, un pacchetto con IP di destinazione 1.2.255.250 sarà inviato al gateway predefinito. Questo è corretto perché l'intervallo 240-255 non fa più parte della rete locale.

Nella figura sono rappresentate tre LAN che fanno parte di un unico sistema autonomo, le icone rotonde sono router, le icone squadrate sono switch. L'amministratore della rete ha a disposizione tutta la subnet 11.0.0.0/24 e alle LAN devono essere assegnate subnet non sovrapposte. Decidere quale è l'assegnazione di indirizzi migliore (quella che usa il minor numero di IP della subnet) tenendo presente che anche le interfacce degli apparati che operano a livello di rete devono avere assegnato un'IP nella subnet di cui fanno parte.



Scegli un'alternativa:

- a. A) 11.0.0.0/27 B) 11.0.0.32/27 C) 11.0.0.64/29
- b. A) 11.0.0.0/27 B) 11.0.0.32/28 C) 11.0.0.48/28
- c. A) 11.0.0.0/27 B) 11.0.0.32/27 C) 11.0.0.64/28
- d. A) 11.0.0.0/27 B) 11.0.0.32/27 C) 11.0.0.32/27

Gli switch non sono apparati di livello di rete, quindi non hanno bisogno di indirizzi IP. Andando per esclusione: la rete più piccola ha bisogno di 7 indirizzi, a cui si sommano i due canonici non utilizzabili ed uno dell'interfaccia del router, quindi almeno 10, ovvero almeno una /28 ( $2^4 = 16$ ), che elimina una risposta. In una delle risposte le reti B e C usano lo stesso range di indirizzi, mentre si chiede di evitarlo. La rete B ha 14 terminali più due indirizzi dei due router, e 2 indirizzi sprecati, quindi ha bisogno almeno di una /27, il che elimina una terza risposta. Risposta corretta -> c.

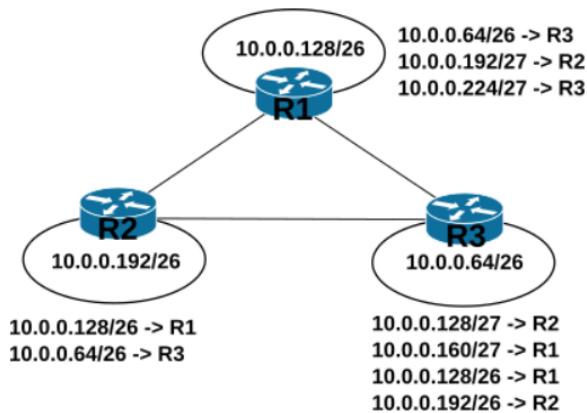
Per ogni subnet devi considerare alcuni indirizzi "persi" per esigenze tecniche:

1. **Indirizzo di rete**: il primo indirizzo della subnet non è assegnabile perché identifica la rete stessa.
2. **Indirizzo di broadcast**: l'ultimo indirizzo della subnet non è assegnabile perché viene usato per inviare messaggi a tutti i dispositivi della rete.

Quindi, per la **rete C**, che ha 7 terminali:

- Devi assegnare 7 indirizzi IP per i terminali.
- Devi assegnare 1 indirizzo per l'interfaccia del router che connette questa LAN.
- Devi considerare i 2 indirizzi "persi" (indirizzo di rete e indirizzo di broadcast).

Nella figura sono rappresentati tre router, con l'assegnazione delle subnet associate a ciascuna LAN collegata ad un router. Viene anche riportata la forwarding table di ciascun router (solo le entry verso altre reti e senza il default). Quale percorso farà un pacchetto con sorgente 10.0.0.119 e destinazione 10.0.0.144?



Scegli un'alternativa:

- a. R3→R1
- b. R1→R2→R3
- c. Nessuna delle precedenti, il pacchetto viene inserito in un loop.
- d. R3→R2→R1

d)

L'IP 10.0.0.119 fa parte della subnet 10.0.0.64/26 che parte contiene gli IP con ultimo intero [64-127]. La destinazione fa parte della subnet 10.0.0.128/26 che contiene gli IP [128-191]. Il pacchetto quindi parte dalla rete di R3 ed è destinato alla rete di R1. La forwarding table di R3 ha due righe che corrispondono, la prima (/26, [128-191]) e la seconda (/27, [128-159]). La seconda è più specifica, quindi il pacchetto viene mandato a R2. R2 ha una entry che invia il pacchetto a R1. Affinché ci fosse stato un loop, R2 avrebbe dovuto avere una regola del tipo 10.0.0.128/27 -> R3.

Un computer deve inviare un pacchetto IP con un payload di 1830 B di cui 20 B compongono l'header del livello di trasporto. L'MTU del link è MTU=1540. Il pacchetto verrà quindi frammentato. Quali sono i valori dei campi dell'header del secondo frammento generato? Si assuma il valore minimo della lunghezza dell'header IP.

Scegli un'alternativa:

- a. More Fragment = True; Total Length = 270; Fragment Offset = 185
- b. More Fragment = False; Total Length = 1540; Fragment Offset = 190
- c. More Fragment = False; Total Length = 330; Fragment Offset = 185
- d. More Fragment = False; Total Length = 330; Fragment Offset = 190

d)

## Domande Aperte

NAT-----

2) nat

OSPF-----

2) OSPF con link state routing, assegnazione IP e differenza rispetto al link state puro.

OSPF

2) Ospf:

1 struttura interna e autonomous system

2 esempio con ipv4 o ipv6 del routing

3 le 2 caratteristiche del ospf che lo differenziano dal LS

- OSPF, autonomous system, esempio distribuzione ip in una rete IPv4 o IPv6, link state protocol e differenze con OSPF

OSPF con esempio su ipv4 o iov6. spiegare Link State e dire quali sono le due differenze tra ospf e Link State puro

OSPF con link state routing, assegnazione IP (fare anche un esempio con IPv4 o IPv6) e differenza rispetto al link state puro.

# BGP E INTERDOMAIN ROUTING

## Esercizi

## **Step-by-Step Example : a small Internet**

AS 150

AS 150 possiede la rete 10.150.0.0/24, una rete di classe C. AS 150 è un AS stub a connessione multipla.

1. Router A in AS 150 annuncia 10.150.0.0/24 a Router D in AS11.
  2. Router A in AS 150 annuncia 10.150.0.0/24 a Router B in AS151.

Gli annunci BGP contengono il prefisso e il percorso per raggiungere il prefisso. AS150 aggiunge il percorso a se stesso: 150.

Entrambi AS151 e AS11 offrono transito ad AS150, quindi annunceranno il prefisso di AS150.

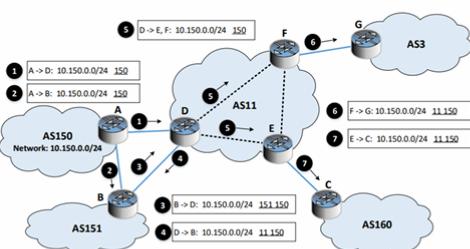
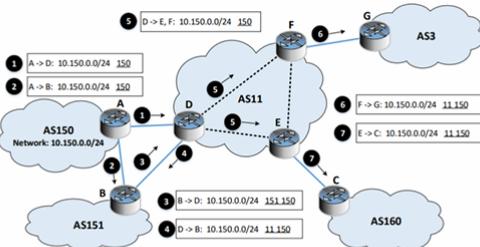
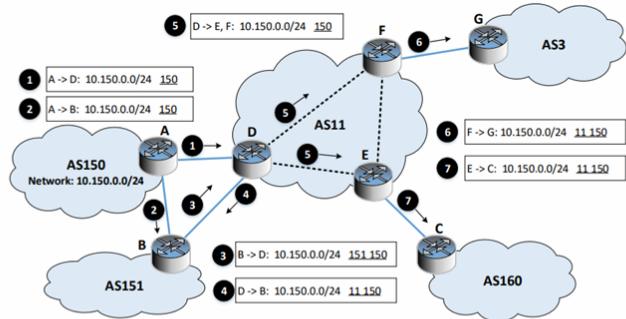
- Router B annuncia 10.150.0.0/24 a D, aggiungendo 151 al percorso.
  - Router D annuncia 10.150.0.0/24 a B, aggiungendo 11 al percorso.

B e D non annunciano 10.150.0.0/24 di nuovo ad A.

I-BGP

BGP può essere utilizzato anche all'interno di un singolo AS, e in questo caso si parla di IBGP (Internal-BGP).

- 5.** Router D, E e F fanno parte dello stesso AS. Pertanto, D annuncia 10.150.0.0/24 a E e F senza aggiungere nulla al percorso.



AS3 and AS160

AS3 e AS160 sono AS stub a connessione singola.

6. Router F annuncia 10.150.0.0/24 a G, aggiungendo 11 al percorso.
  7. Router E annuncia 10.150.0.0/24 a C, aggiungendo 11 al percorso.

Ora C ha appreso che per raggiungere 10.150.0.0/24 deve inviare il traffico a Router E, attraversare AS11 e poi raggiungere AS150. Lo stesso vale per G.

**Route is propagated**

Alla fine di questo processo, il prefisso 10.150.0.0/24 è stato propagato in tutto l'Internet (anche se piccolo). **Ogni router ora ha una rotta verso 10.150.0.0/24**. Ogni altro AS annuncerà i suoi prefissi e, seguendo la stessa procedura, **ogni router in Internet avrà una rotta corretta verso qualsiasi prefisso**.

## Domande Aperte

BGP -----

BGP e pathprepending

## **DATALINK LAYER – ETHERNET E WIFI (SPANNING TREE PROTOCOL, ARP, DHCP)**

## Esercizi

### Efficiency Calculation

Immagina che la **velocità di comunicazione sia di 100 Mb/s.**

T deve inviare un **frame di 1000 byte.**

Un **frame di ACK è lungo 14 byte** (l'intestazione del frame è più corta rispetto ai frame di dati).

Ignora il tempo di volo, supponi **802.11ac a 5 GHz.**

Qual è l'efficienza MAC dello schema mostrato prima?

### Simple MAC : Sigure Transmitter (Exercise)

Il tempo necessario per inviare 1000 byte è:

$$> 1000 \times 8 / 100.000.000 = 80.112 \mu s$$

Il tempo necessario per inviare 14 byte è:

$$> 14 \times 8 / 100.000.000 = 0.14 \mu s$$

Per inviare un frame abbiamo bisogno di:

$$> 34 + 80.112 + 16 + 0.14 = 130.252 \mu s$$

L'efficienza è:

$$> 80.112 / 130.252 \approx 0.61$$

Nota che l'efficienza non è completamente corretta perché per inviare 1000 byte è necessario inviare anche un'intestazione, quindi un frame più grande. Sta a te correggere il calcolo.

Per correggere il calcolo dell'efficienza, consideriamo anche l'intestazione del frame. Supponiamo che l'intestazione sia di 34 byte. Pertanto, il frame totale da trasmettere sarà composto da 1000 byte di dati più 34 byte di intestazione, per un totale di 1034 byte.

1. Tempo necessario per trasmettere 1034 byte:

$$> 1034 \times 8 / 100.000.000 = 827.2 \mu s$$

2. Tempo necessario per trasmettere 14 byte di ACK:

$$> 14 \times 8 / 100.000.000 = 0.14 \mu s$$

3. Tempo totale per inviare un frame e ricevere l'ACK:

$$> 827.2 \mu s + 16 \mu s + 0.14 \mu s = 843.34 \mu s$$

4. Efficienza:

$$> 827.2 / 843.34 \approx 0.980$$

Quindi, dopo aver corretto il calcolo per includere l'intestazione del frame, l'efficienza è di circa 98.0%.

Una rete Wi-Fi usa canali da 20 MHz, e ne può aggregare fino ad 8. Nella stessa stanza devono coesistere due reti usando fino ad un massimo di 8 canali complessivamente, senza farsi interferenza. La rete 1 deve supportare una capacità maggiore o uguale a 200 Mb/s e la rete 2 maggiore o uguale a 800 Mb/s. Si chiede di allocare staticamente il numero di canali (N) e la modulazione minima (X-QAM: modulazione con X livelli per simbolo) che garantiscono queste prestazioni. Se ci sono più soluzioni valide, si preferisca una soluzione che può funzionare con un SNR più basso.

Scegli un'alternativa:

- a. 1) 4-QAM, N: 3 2) 16-QAM, N: 5,
- b. 1) 512-QAM, N: 5 2) 4-QAM, N: 4,
- c. 1) 16-QAM, N: 2 2) 32-QAM, N: 5
- d. 1) 16-QAM, N: 1, 2) 8-QAM, N: 4,

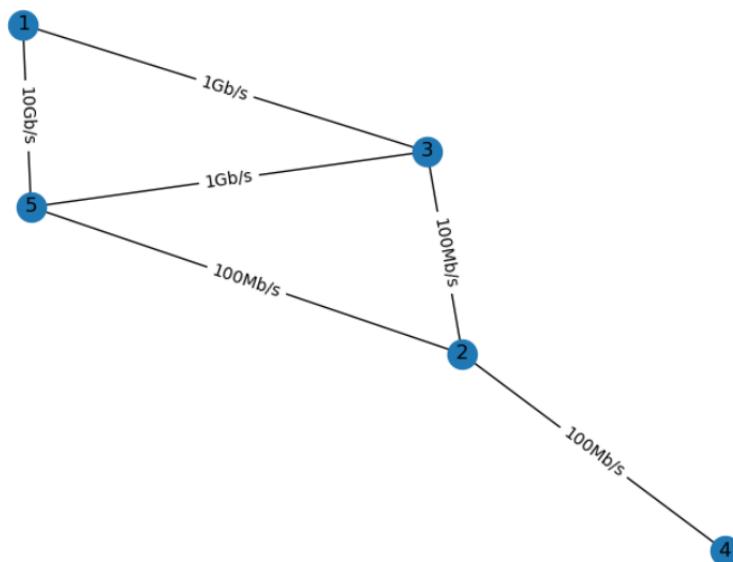
Una delle soluzioni prevede l'uso di 4 e 5 canali, ma a disposizione ce ne sono solo 8. Applicando l'equazione di Nyquist si possono valutare le performance delle altre soluzioni proposte:

$$C \text{ [Mb/s]} = 2 * N * 20 \text{ [MHz]} \log_2(X)$$

La soluzione (16-QAM, N: 1; 8-QAM, N: 4) garantisce solo 160 e 480 Mb/s quindi non è valida. Le altre due soluzioni sono entrambe valide, ma una usa delle codifiche più alte (16-QAM contro 4-QAM per la rete 1 e 32-QAM contro 16-QAM per la rete 2), e quindi avrà bisogno di un SNR più alto rispetto all'altra.

Risposta -> a.

Nella rete in figura, ogni nodo è uno switch, ed il numero è la parte più significativa dell'identificativo usato da Spanning Tree. Ciascun arco riporta la capacità del link. Per comodità il numero di porta di uno switch è lo stesso dell'identificativo a cui è collegato, quindi se lo switch 5 è collegato allo switch 8, la porta usata per il collegamento in uscita da 5, è la 8 e la chiamiamo 5.8. Quale è lo stato delle porte degli switch alla convergenza di STP?



Scegli un'alternativa:

a. 1.3: designated  
1.5: blocked

2.3: blocked  
2.4: designated  
2.5: root

3.1: root  
3.2: designated  
3.5: blocked

4.2: root

5.1: root  
5.2: designated  
5.3: designated

b. 1.3: designated  
1.5: designated

2.3: root  
2.4: designated  
2.5: root

3.1: root  
3.2: designated  
3.5: blocked

4.2: root

5.1: root  
5.2: designated  
5.3: designated

c. 1.3: designated  
1.5: designated

2.3: blocked  
2.4: designated  
2.5: root

3.1: root  
3.2: designated  
3.5: designated

4.2: root

5.1: root  
5.2: designated  
5.3: blocked

d. 1.3: designated  
1.5: designated

2.3: blocked  
2.4: designated  
2.5: root

3.1: root  
3.2: designated  
3.5: blocked

4.2: root

5.1: root  
5.2: designated  
5.3: designated

Capacity	Cost
10 Mbps	2000000
100 Mbps	200000
1 Gbps	20000
10 Gbps	2000
100 Gbps	200

Osservando le soluzioni proposte si nota che lo switch con identificativo 1 deve avere entrambe le porte allo stato designated, essendo la radice dell'albero (che elimina una soluzione) e nessuno switch può avere due porte nello stato root (che elimina un'altra soluzione). Delle due che rimangono, la differenza è nello stato di 3.5 e 5.3 che sono invertiti. È chiaro che a convergenza 5 ha un BPDU migliore si 3 (ha costo minore dal root) e quindi 5 imposta 5.3 a designated e di conseguenza 3.5 è blocked.

Volendo svolgere tutto il protocollo dall'inizio, bisogna ricordare che il valore numerico del costo di un link parte da 200 per un link a 100Gb/s ed è inversamente proporzionale alla velocità. Quindi non è importante ricordare il numero, basta ricordare che dividere per 10 la velocità significa moltiplicare per 10 il costo. In questo esercizio abbiamo link dal costo 2000 (10Gb/s), 20000 (1Gb/s) e 200000 (100Mb/s). Si tiene a mente poi che il root è lo switch con l'identificativo minore e le sue porte sono tutte designated. Se una porta è designated, la corrispondente porta dello switch collegato è blocked o root.

Per risolvere l'esercizio in modo efficiente si parte dal nodo con l'identificativo più basso e si impostano le sue porte allo stato 'designated'. La porta 5.1 diventa quindi 'root' visto che il percorso 1-5 ha costo minore di 1-3-5 (o qualsiasi altro cammino da 1 a 5).

Lo switch 3 riceve quindi BPDU [1,0,1] e [1,2000,5] dai vicini 1 e 5 rispettivamente. Calcola i priority vector [1,20000,1] e [1,220000,5], e visto che il suo BPDU è inizialmente [3,0,3], sceglie lo switch 1 come root e imposta 3.1 allo stato root. A quel punto il suo BPDU diventa [1,20000,3], lo confronta con [1,2000,5] ricevuto dal 5, e siccome il suo è peggiore, la porta 3.5 diventa blocked. Di converso la 5.3 diventa designated. Lo switch 2 ha un BPDU peggiore sia di 3 che di 5, e sceglie come padre lo switch 5 perché 1-5-2 ha costo minore di 1-3-2. Quindi 2.5 e 2.3 diventano root e blocked, mentre 3.2 e 5.2 diventano designated. Le porte 2.4 e 4.2 sono ovviamente designated e root.

## 1. Identificare lo switch root

Il root bridge è lo switch con l'identificativo più basso.

### 👉 Switch 1 è il root bridge

- **Tutte le sue porte sono in stato "designated".**

---

## 2. Calcolare i costi dei percorsi verso il root

Il costo del cammino dipende dalla velocità del link:

- **10 Gb/s → Costo 2000**
- **1 Gb/s → Costo 20000**
- **100 Mb/s → Costo 200000**

Ora, per ogni switch, calcoliamo il costo per raggiungere il root bridge:

- **Switch 5:** Collegato direttamente a 1 tramite un link da **10Gb/s (costo 2000)**
    - Percorso migliore: **1 → 5 (costo 2000)**
  - **Switch 3:** Collegato direttamente a 1 con **10Gb/s (costo 2000)**
    - Percorso migliore: **1 → 3 (costo 2000)**
  - **Switch 2:** Può raggiungere 1 tramite **3 o 5**
    - **1 → 3 → 2: costo 2000 + 20000 = 22000**
    - **1 → 5 → 2: costo 2000 + 2000 = 4000**
      - ✓ Sceglie il percorso **1 → 5 → 2 (costo 4000)**
  - **Switch 4:** Unico percorso attraverso 2 (**costo 4000 + 20000 = 24000**)
    - Percorso migliore: **1 → 5 → 2 → 4 (costo 24000)**
- 

### 3. Determinare lo stato delle porte

Regole principali:

- Il **root bridge** ha tutte le porte in stato "designated".
- Per ogni switch, la porta con il miglior percorso verso il root è "**root**".
- Le porte connesse a una root port sull'altro switch sono "**designated**".
- Se due switch hanno più collegamenti, quello con il percorso peggiore ha la porta "**blocked**".

#### Porta Stato

- 1.3 Designated
- 1.5 Designated
- 2.3 Blocked (perché 2 → 3 è un percorso peggiore di 2 → 5)
- 2.4 Designated
- 2.5 Root (miglior percorso verso il root bridge)
- 3.1 Root
- 3.2 Designated
- 3.5 Blocked (perché il percorso 3 → 5 è peggiore del percorso 5 → 3)
- 4.2 Root
- 5.1 Root
- 5.2 Designated
- 5.3 Designated
-

#### 4. Verifica finale

Dalle informazioni fornite, il nostro risultato coincide con la soluzione spiegata:

- **3.5 deve essere blocked** perché il miglior percorso al root è tramite 5, non 3.
  - **5.3 è designated**, poiché è connesso a una porta blocked su 3.
- 

#### Conclusione

- **Individua il root bridge** (switch con ID più basso).
- **Calcola i percorsi migliori** dal root bridge a ogni switch.
- **Assegna gli stati delle porte** seguendo le regole di STP

## Domande Aperte

2) Congestione nelle reti wifi, csma, evitarle senza dcf e backoff

802.11 -----

2) Spiegare MAC 802.11 nel problema di accessi multipli alla rete, come funziona DCF senza RTS/CTS.  
Cosa serve il tempo di backoff

802.11 parla di come il traffico di rete funziona, come i trasmettitori inviano pacchetti alla ricevente (AP), problema del nodo nascosto

Come funziona il MAC di 802.11, nel caso del problema del nodo nascosto ("hidden node"). Nella descrizione, seguire la traccia data dalle domande:

- quale problema introduce il nodo nascosto?
- cosa cambia nel DCF nel caso del nodo nascosto?
- quali sono i vantaggi e gli svantaggi di questi cambiamenti e come si mitigano gli svantaggi?

SPANNING TREE -----

Spiegare i principi di funzionamento dello Spanning Tree Protocol. Seguire la seguente traccia:

1) quale problema si risolve con STP?

- 2) descrivere nel dettaglio lo scambio dei frame necessari per la formazione dello STP
- 3) Riportare le condizioni di confronto dei BPDU che si usano per prendere decisioni in SPT

## TLS

Descrivere il protocollo TLS (nella versione 1.2). Nella descrizione, seguire la traccia data dalle domande:

- Quali sono vantaggi e svantaggi di applicare la crittografia ai vari layer dell'IP stack?
- Quali sono le proprietà dell'handshake protocol?
- A cosa servono i pacchetti ChangeCipherSpec e Finished?

## TODO

PSEUDO CODICE:

- NAGLE;
- DV e DV CON POISON REVERSE;
- BACKWARD LEARNING;
- LSP.

DOMANDE APERTE.