# Esame Reti 15 Gennaio 2019
# Parte A e B

- Idle RQ (tasso utilizzo e valore ottimale fin.)
- Slow start e incremento addizionale
- Socket server con Fork
- HTTP GET con if-modified-since
- NVT
- Funzionamento SMTP
- Formato messaggi posta + es. MIME

**Esercizio 2 (5 punti) Supponendo che la PDU di un protocollo IDLE RQ sia 1000 bit, qual è il tasso di utilizzo di un collegamento punto-punto a 200 Mbps sulla distanza di 8000 km? Qual è il valore ottimale della finestra?**

$T_{ix} = 1000 / 200000000 = 5 * 10^{-6}$
$2*T_p = 2 * 8000000 / 2*10^8 = 0.08$
$T_t = T_{ix} + 2T_p = 0.080005$ (trascurando i tempi di processazione, attualmente molto rapidi)

$U = 1 / ( 1 + 2T_p / T_{ix} ) = 1 / ( 1 + 0.08 / 5 * 10^{-6}) = 0.0000624$ %

La dimensione della finestra ottimale sarebbe 2000125 Bytes

**Esercizio 3 (9 punti)**
**Supponiamo di avere una connessione TCP tra A e B. Supponiamo che A abbia i seguenti valori iniziali:**

- **Finestra di congestione: cwnd = 504 byte (MSS = 504 byte)**
- **Soglia di slow start: ssthresh = 2520 byte.**
- **La advertised window rimane a 5000 byte in tutto lo scambio di dati (in ogni ack, è indicata la dimensione della finestra di 5000). Supponiamo che i seguenti eventi si verifichino in A:**
    - **Tempo t = 0: A invia 1 segmento a B**
    - **Tempo t = 1: A riceve un ACK per 1 segmento**
    - **Tempo t = 2: A invia 1 segmento a B**
    - **Tempo t = 3: A riceve un ACK per 1 segmento**
    - **Tempo t = 4: A invia 2 segmenti a B**
    - **Tempo t = 5: A riceve un ACK per entrambi i segmenti**

**Specificate la dimensione di cwnd e ssthresh in ciascun momento**

Partendo da un valore di cwnd che sta sotto la soglia ssthresh, si opera con un incremento moltiplicativo fino al raggiungimento della soglia di threshold:

t = 0
cwnd: 504 byte
ssthresh: 2520 byte

t = 1
cwnd: 1008 byte
ssthresh: 2520 byte

t = 2
cwnd: 1008 byte
ssthresh: 2520 byte

t = 3
cwnd: 2016 byte
ssthresh: 2520 byte

Alla ricezione dei due ack, la finestra di congestione supera la soglia di threshold, e quindi inizia l'incremento additivo.

t = 5
Alla ricezione del primo ACK

cwnd: 4032 byte
ssthresh: 2520 byte

Alla ricezione del secondo ACK

cwnd: 4033 byte
ssthresh: 2520 byte

Inoltre, in questo frangente di tempo l'AW non subendo modifiche e rimanendo 5000 byte, rimane sempre maggiore della finestra di congestione, e quindi sarà sempre quest'ultima ad essere la finestra utilizzata.
In più, non subendo perdite (o l'arrivo di più ACK per lo stesso segmento), la soglia di threshold non cambia.

## Parte B

Codice server

```
#include <socket.h>
#include <string.h>

int main(int argc, char* argv) {

	//Init della struttura
	struct sockaddr_in local, client;
	local.sin_family = AF_INET;
	local.sin_port = htons(8080);
	local.sin_address.s_addr = INADDR_ANY

	int sb = socket(AF_INET, SOCK_STREAM, 0);

	bind(sb, (struct sockaddr*) &local, sizeof(local));
	listen(sb, 5);

	while(1 == 1) {
		int ss, pid;
		ss = accept(sb, (struct sockaddr*) &client, sizeof(struct sockaddr));
		pid = fork();

		if(pid != 0) {
			//Processo padre: chiudo il socket di servizio
			close(ss);
		}

		else {
			//Processo figlio: chiudo il socket di benvenuto
			close(sb);

			char scelta[10];
			//Attendo un messaggio dal client
			recv(ss, &scelta, sizeof(scelta), 0)

			if(strcmp(scelta, “ricprod”) = 0) {
				ricerca_prod(ss);
```

```
                }

                else if(strcmp(scelta, “acqprod”) = 0) {
                      acq_prod(ss);
                }

                else {
                      char* msg = “Metodo non implementato”;
                      send(ss, &msg, sizeof(msg), 0);
                }

                //Chiudo il socket di servizio ed esco
                close(ss);
                return 0;
           }
      }
}

void ricerca_prod(int ss) {

      char nomeProd[500];
      int qta;
      char* msg = “Inserire nome prodotto”;
      send(ss, &msg, sizeof(msg), 0);

      recv(ss, &nomeProd, sizeof(nomeProd), 0);

      msg = “Inserire quantita’”;
      send(ss, &msg, sizeof(msg), 0);

      recv(ss, &qta, sizeof(qta), 0);
      qta = ntohs(qta);

      char* prodotti = ricerca_prodotto(&nomeProd, qta);
      send(ss, &prodotti, sizeof(prodotti), 0);
}

void acq_prod(int ss) {
      char codiceProd[10];
      char cc[16];
      int qta;

      char* msg = “Inserire codice prodotto”;
      send(ss, &msg, sizeof(msg), 0);
      recv(ss, &codiceProd, sizeof(codiceProd), 0);

      msg = “Inserire carta di credito”;
      send(ss, &msg, sizeof(msg), 0);
      recv(ss, &cc, sizeof(cc), 0);

      msg = “Inserire quantità”;
```

```
        send(ss, &msg, sizeof(msg), 0);
        recv(ss, &qta, sizeof(qta), 0);
        qta = ntohs(qta);

        long codiceAcq = acquista_prodotto(&codicePord, &cc, qta);

        codiceAcq = htonl(codiceAcq);
        send(ss, &codiceAcq , sizeof(codiceAcq), 0);
}
```

**(Per studenti online) Nell'ambito del protocollo HTTP si mostri una richiesta GET condizionale tramite l'header if-modied-since. Si discutano inoltre le possibili risposte a tale richiesta e per ognuna delle risposte mostrare il contenuto del messaggio HTTP comprensivo di header.**

Richiesta GET

```
GET /prova.html HTTP/1.1
User-Agent: Mozilla/5.0 (Firefox/3.5.5)
Accept: text/html
If-modified-since: Sun, 19 Apr 2020 07:28:00 GMT
```

Quando viene specificato questo campo in una richiesta HTTP, il server, se dispone di una versione più recente della risorsa richiesta la manda, altrimenti trasmette un header che dice che la risorsa è ancora quella e che quindi si può prelevare dalla cache del browser (rispondendo con un codice 304)

Se il server ha una risorsa più recente

```
HTTP/1.1 200 OK
Date: Mon, 20 Apr 2020 09:00:00 GMT
Server: Apache
Expires: Mon, 27 Apr 2020 09:00:00 GMT
Content-Type: text/html
Content-Length: {Lunghezza contenuto}

<!DOCTYPE html>
<html>
        <head><title>prova</title>
        <body>prova</body>
</html>
```

Se il server non ha una risposta più recente

```
HTTP/1.1 304 Not Modified
Date: Mon, 20 Apr 2020 09:00:00 GMT
Server: Apache
```

**Domanda 1) (3 punti)**
**Dopo aver introdotto il funzionamento del protocollo Telnet, si discuta in dettaglio il Network Virtual Terminal (NVT).**

Telnet è un protocollo che consente di inviare comandi da una macchina all'altra. Previa autenticazione (le cui credenziali che vengono inviate in chiaro, rendendo il protocollo poco sicuro= telnet consente di connettersi ad un sistema remoto in cui sia in esecuzione un server telnet (telnetd, ossia un processo che passerà i comandi digitati da client ad una shell locale di sistema operativo).

Telnet consetirà di inoltrare sia comandi destinati alla shell della macchina remota, sia di gestire le opzioni di comunicazione, permettendo di configurare le opzioni relative a telnet.

Telnet si basa su NVT, ossia un ambiente (come anche al protocollo FTP) che consente di fornire un terminale di rete basato su un'interfaccia standard ai sistemi remoti. Esso presenta un insieme minimo di funzioni comuni, che poi saranno ampliate nello specifico ambito di applicazione.
Grazie a NVT è così possibile non utilizzare software specifici lato client o server, piuttosto il software nelle macchine client / server si occuperà di tradurre le sequenze di comandi nel formato NVT e viceversa.
NVT si basa su UASCII a 7 bit.

**Domanda 2) (5 punti)**
**Nell'ambito di un sistema per l'invio/ricezione di mail spiegare brevemente il funzionamento del protocollo SMTP, presentare il formato dei messaggi di posta e presentare un esempio di messaggio MIME multipart.**

SMTP è un protocollo basato su TCP il cui scopo è permettere lo scambio di messaggi usando tecnologie che ne permettano la raccolta e l'accodamento (protocollo non interattivo).
SMTP prevede degli attori nel suo funzionamento: lo user agent, il mail server e la mail box.
Lo user agent, ossia il programma che contiene l'editor per la scrittura del messaggio, utilizzerà il protocollo SMTP per trasmettere il messaggio al mail server (se questo non risiede nella stessa macchina dello user agent). Il mail server analizza quindi chi è il destinatario, e tramite SMTP si mette in contatto con la macchina destinataria, e quindi le invia il messaggio, che sarà conservato nella relativa mailbox.
Se lo user agent del destinatario non risiede sulla stessa macchina della mailbox, esso vi farà accesso tramite protocolli POP o IMAP.
Il protocollo SMTP, per la comunicazione, segue tre fasi:

- handshaking (HELO, scambio di identità)
- trasferimento dei messaggi
- chiusura della connessione

I messaggi di posta hanno un formato diviso in intestazione e corpo.
L'intestastazone si compone di un minimo di tre campi:

- From:
- To:
- Subject:

A questi possono essere aggiunti altri campi utili, come Cc, Ccn e così via…
Il corpo è composto da caratteri ASCII, e tra intestazione e corpo c'è una riga vuota (CR LF).

Per poter includere contenuti multimediali, è possibile utilizzare MIME.
Esempio di messaggio MIME multipart:

```
From: gianmarco.barbato@studenti.unimi.it
To: segreteria.crema@unimi.it
Subject: Errore in calcolo tasse
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary=1234567

--1234567
Content-Transfer-Encoding: quoted-printable
Content-Type: text/plain

Buon giorno
```

```
Vorrei segnalare che il calcolo delle tasse è errato
Allego screen.
Grazie
--1234567
Content-Transfer-Encoding: base 64
Content-Type: image/jpeg

{base 64 del file da inviare}
--1234567
```