UDP

USER DATAGRAM PROTOCOL

UDP: USER DATAGRAM PROTOCOL [RFC 768]

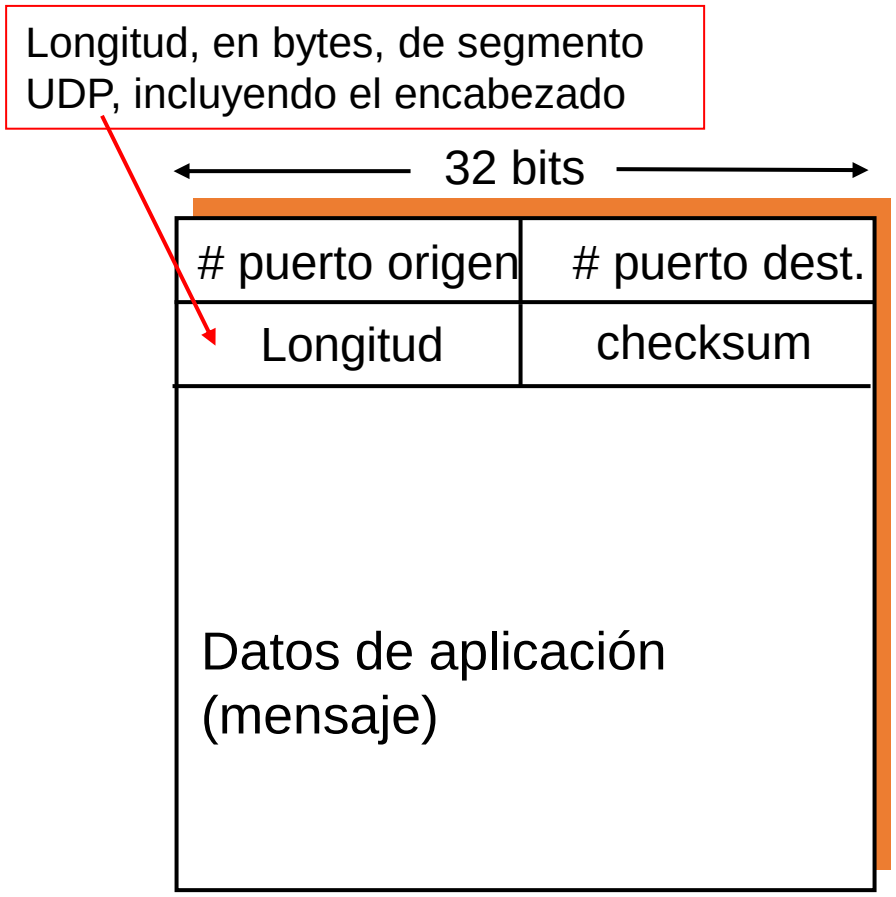
- Protocolo de transporte de Internet “sin adornos”
- Servicio de “mejor esfuerzo”, los segmentos UDP pueden ser:
  - Perdidos
  - Entregados fuera de orden a las aplicaciones
- *No orientado a la conexión:*
  - Sin handshaking entre el emisor y receptor UDP
  - Cada segmento UDP se maneja independientemente de los demás

¿Porqué existe UDP?

- No hay establecimiento de conexión (lo que puede agregar retardo)
- Simple: sin estado de conexión en emisor o en receptor
- Encabezado de segmento pequeño
- Sin control de congestionamiento

UDP

- Comúnmente utilizado para aplicaciones de difusión multimedia
  - Tolerante a pérdidas
  - Sensibles a la velocidad
- Otros usos para UDP
  - DNS
  - SNMP
- Transferencia fiable sobre UDP: agregar fiabilidad en la capa de aplicación
  - Recuperación de errores específica de cada aplicación



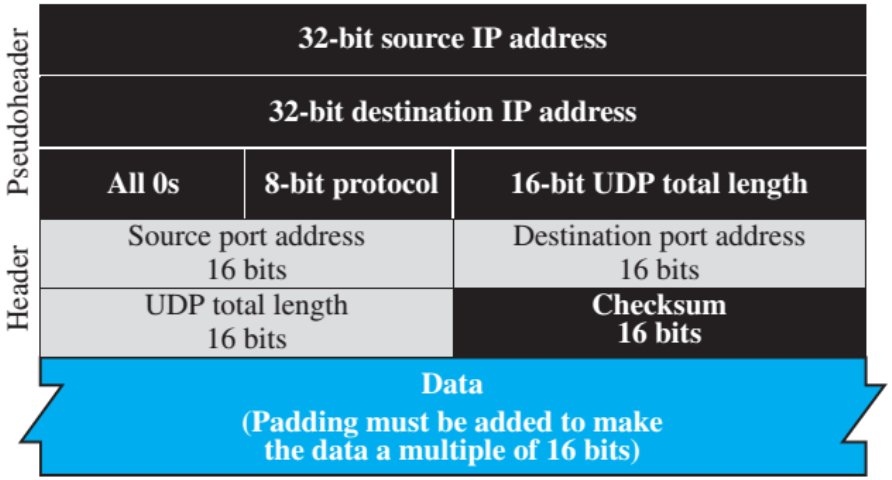
Formato de segmento UDP

CHECKSUM UDP

Objetivo: Detectar errores en segmentos transmitidos

Emisor:

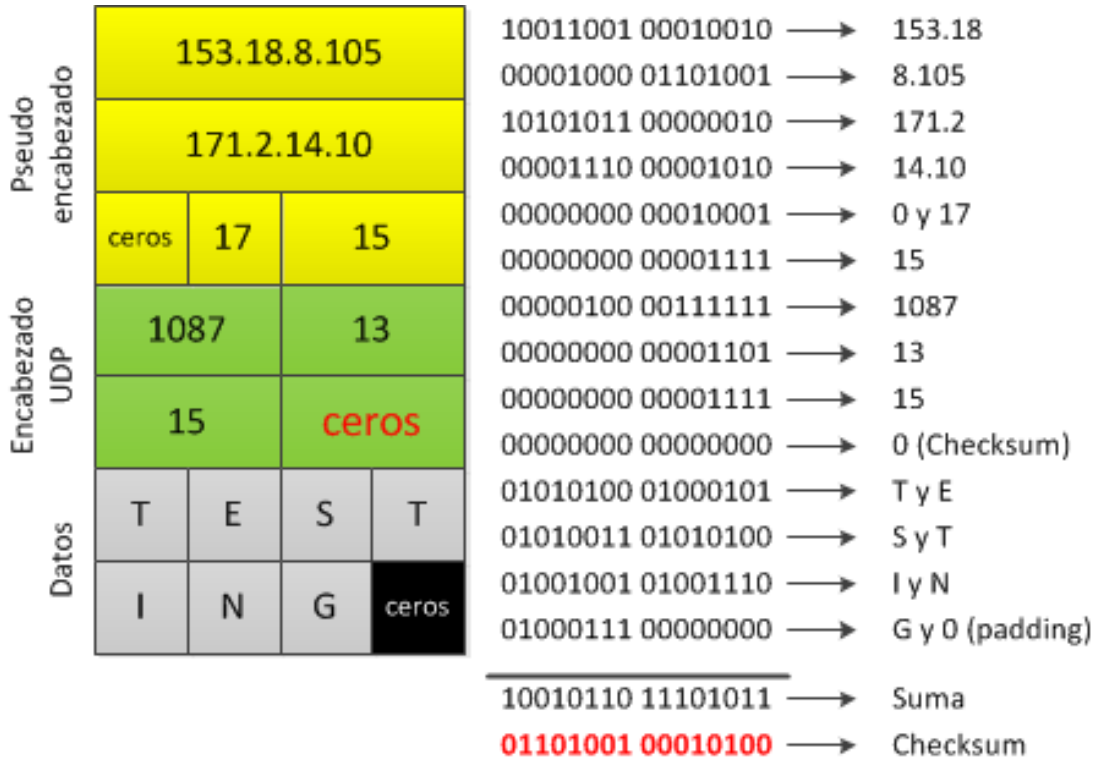
- Procesa un segmento como una secuencia de enteros de 16 bits
- **Checksum**: suma (complemento a 1) de los contenidos del segmento:
  - Seudoencabezado + encabezado UDP + Datos
- El emisor coloca el valor de la suma de verificación en el campo UDP checksum y descarta el seudoencabezado



CHECKSUM UDP

Ejemplo

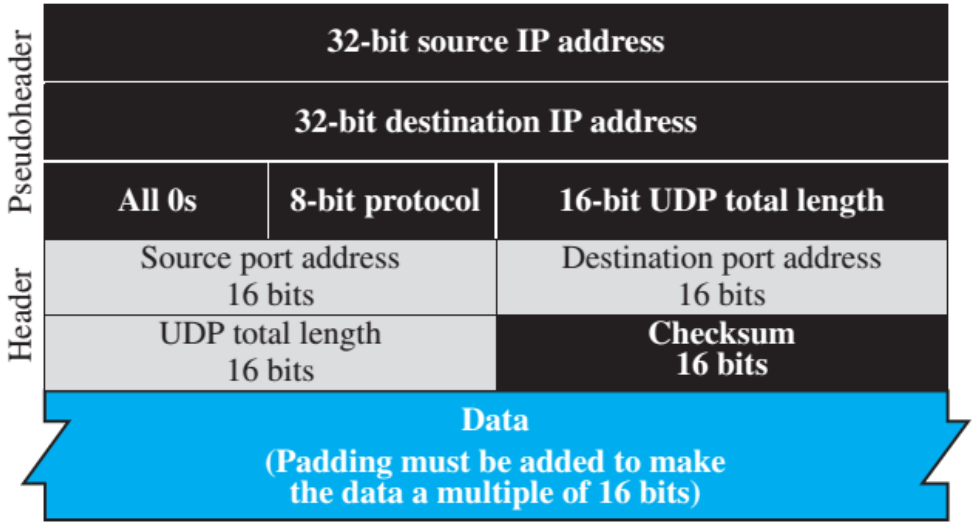
- El campo checksum tiene un valor inicial de cero.
- Después de calcular la suma de verificación, el pseudoencabezado se descarta (no se transmite).
- Si el checksum obtenido es 0x0000, este se reemplaza por 0xFFFF, para no confundirlo con un checksum no calculado.



CHECKSUM UDP

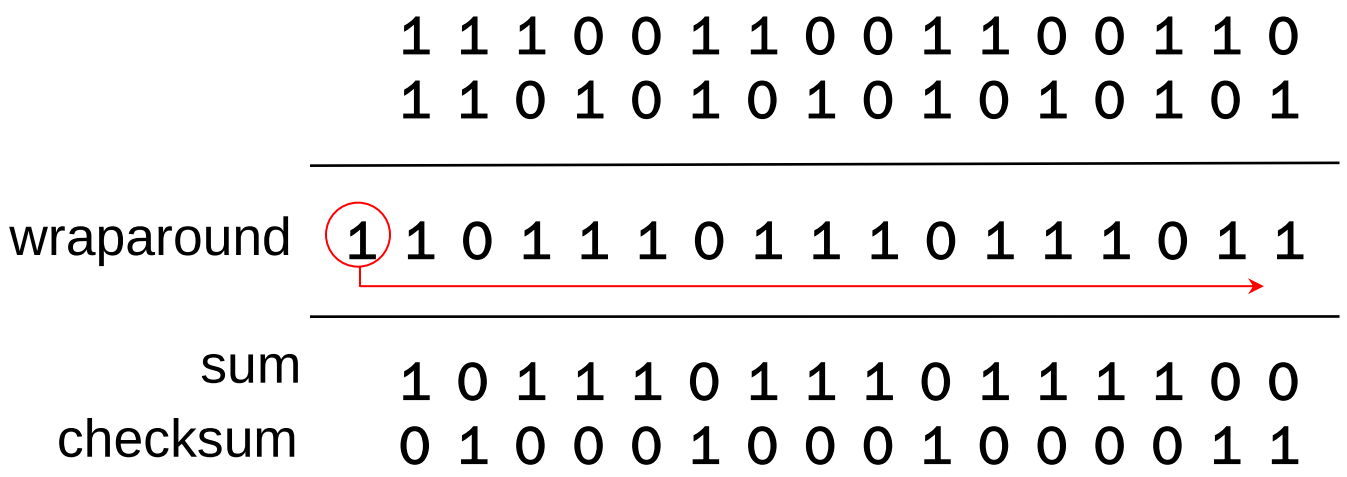
Receptor:

- Calcula la suma de verificación del segmento recibido + pseudoencabezado
- Verifica si la suma calculada es igual al valor del campo checksum:
  - NO – error detectado
  - SI – no se detectaron errores.



EJEMPLO DE CHECKSUM INTERNET

- Nota
  - En aritmética complemento a 1, cuando se suma números, un acarreo del bit más significativo debe sumarse al resultado



EJEMPLO DE CALCULO DE CHECKSUM

```c
/* set ip checksum of a given ip header*/
void compute_ip_checksum(struct iphdr* iphdrp){
  iphdrp->check = 0;
  iphdrp->check = compute_checksum((unsigned short*)iphdrp, iphdrp->ihl<<2);
}

/* Compute checksum for count bytes starting at addr, using one's complement of one's complement sum*/
static unsigned short compute_checksum(unsigned short *addr, unsigned int count) {
  register unsigned long sum = 0;
  while (count > 1) {
      sum += * addr++;
      count -= 2;
  }
  //if any bytes left, pad the bytes and add
  if(count > 0) {
      sum += ((*addr)&htons(0xFF00));
  }
  //Fold sum to 16 bits: add carrier to result
  while (sum>>16) {
      sum = (sum & 0xffff) + (sum >> 16);
  }
  //one's complement
  sum = ~sum;
  return ((unsigned short)sum);
}
```

EJEMPLO DE CALCULO DE CHECKSUM

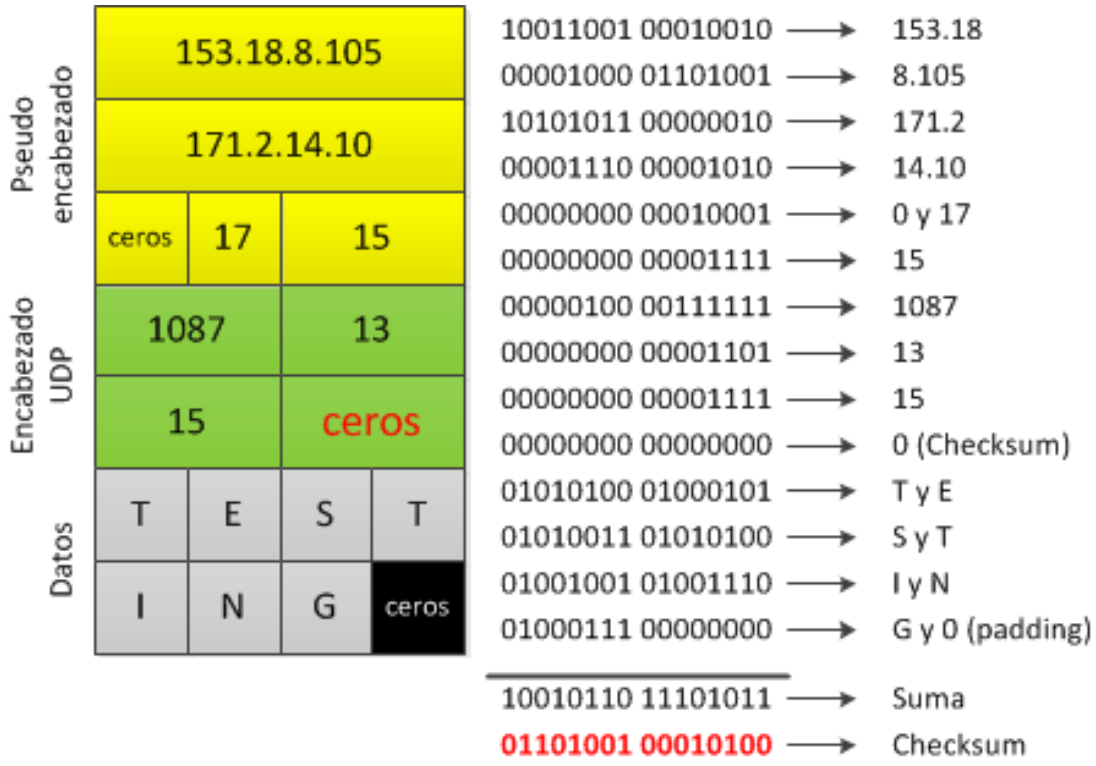
```c
/* set tcp checksum: given IP header and tcp segment */

void compute_tcp_checksum(struct iphdr *pIph, unsigned short *ipPayload)
{
    register unsigned long sum = 0;
    unsigned short tcpLen = ntohs(pIph->tot_len) - (pIph->ihl<< 2);
    struct tcphdr *tcphdrp = (struct tcphdr*)(ipPayload);

    //add the pseudo header
    //the source ip
    sum += (pIph->saddr>>16)&0xFFFF;
    sum += (pIph->saddr)&0xFFFF;

    //the dest ip
    sum += (pIph->daddr>>16)&0xFFFF;
    sum += (pIph->daddr)&0xFFFF;

    //protocol and reserved: 6
    sum += htons(IPPROTO_TCP);

    //the length
    sum += htons(tcpLen);

    //add the IP payload
    //initialize checksum to 0
    tcphdrp->check = 0;

    while (tcpLen > 1) {
        sum += * ipPayload++;
        tcpLen -= 2;
    }

    //if any bytes left, pad the bytes and add
    if(tcpLen > 0) {
        //printf("++++++++++padding, %dn", tcpLen);
        sum += ((*ipPayload)&htons(0xFF00));
    }

    //Fold 32-bit sum to 16 bits: add carrier to result
    while (sum>>16) {
          sum = (sum & 0xffff) + (sum >> 16);
    }
    sum = ~sum;

    //set computation result
    tcphdrp->check = (unsigned short)sum;
}
```

EJEMPLO DE CALCULO DE CHECKSUM

```
/* set tcp checksum: given IP header and UDP datagram */
void  compute_udp_checksum(struct  iphdr  *pIph,  unsigned
short *ipPayload) {
    register unsigned long sum = 0;
    struct udphdr *udphdrp = (struct udphdr*)(ipPayload);
    unsigned short udpLen = htons(udphdrp->len);

    //printf("~~~~~~~~~~~~~~~~~~~~~~udp len=%d\n", udpLen);
    //add the pseudo header
    //printf("add pseudo header\n");
    //the source ip
    sum += (pIph->saddr>>16)&0xFFFF;
    sum += (pIph->saddr)&0xFFFF;

    //the dest ip
    sum += (pIph->daddr>>16)&0xFFFF;
    sum += (pIph->daddr)&0xFFFF;

    //protocol and reserved: 17
    sum += htons(IPPROTO_UDP);

    //the length
    sum += udphdrp->len;

    //add the IP payload
    //printf("add ip payload\n");
    udphdrp->check = 0; //initialize checksum to 0
    while (udpLen > 1) {
        sum += * ipPayload++;
        udpLen -= 2;
    }

    //if any bytes left, pad the bytes and add
    if(udpLen > 0) {
        //printf("+++++++++++++++padding: %dn", udpLen);
        sum += ((*ipPayload)&htons(0xFF00));
    }

    //Fold sum to 16 bits: add carrier to result
    //printf("add carrier\n");
    while (sum>>16) {
          sum = (sum & 0xffff) + (sum >> 16);
    }

    //printf("one's complement\n");
    sum = ~sum;

    //set computation result
    udphdrp->check=((unsigned short)sum==0x0000)?0xFFFF:(unsigned short)sum;
}
```

EJERCICIO

1. En el segmento UDP de ejemplo, cambie la palabra TESTING por AVISO y calcule el campo Checksum correspondiente.



REFERENCIAS

1. https://gist.github.com/david-hoze/0c7021434796997a4ca42d7731a7073a
2. https://stackoverflow.com/questions/8845178/c-programming-tcp-checksum
3. https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers
4. https://en.wikipedia.org/wiki/American_Registry_for_Internet_Numbers
5. https://en.wikipedia.org/wiki/Internet_Assigned_Numbers_Authority