Compilando Conocimiento

Redes Computacionales

Ciencias de la Computación

Oscar Andrés Rosas Hernandez
Laura Andrea Morales López

Febrero 2018

Índice general

Parte I

Las Redes en General

Capítulo 1

Introducción

1.1. Definición

1.1.1. Redes Computacionales

Decimos que una red de computadoras es un conjunto de nodos (sean computadoras personales, servidores, telefonos, etc...) interconectados por un medio físico y que se implementa una pila de protocolos para poder comunicarse entre si y compartir recursos.

Decimos que una red tiene que desempeñar las siguientes funciones:

- Encapsulamiento

  Es decir
- Control de Errores
- Direccionamiento
- Multiplexión
- Segmentación y Ensamblado
- Servicios de Transmisión
- Control de Flujo
- Entrega en Ordén
- Control de Conexión

1.2. Clasíficacion de Redes

1.2.1. Por su Alcance

- **PAN** (*Personal Area Network*) Redes de Área Personal
  A una distancia de aprox 1m. Con una extensión de $1m^2$
- **LAN** (*Local Area Network*) Redes de Área Local
  A una distancia de 10m-1km. Con una extensión de un cuarto.
- **CAN** Redes de Área Campus
- **MAN** (*Metropolitan Area Network*) Redes de Área Metropolitana
  A una distancia de 10km. Con una extensión de una ciudad.
- **WAN** (*Wide Area Network*) Redes de Área Amplia
  A una distancia de 1000km. Con una extensión de un País o Continente.
- **GAN** (*Global Area Network*) Redes de Área Global
  Con una extensión de un mundo.

1.2.2. Por su Relación Funcional

- **Cliente Servidor**
- **Igual a Igual (P2P)**

1.2.3. Por su Medio de Transmisión

- **Alambricas ó Guiado**:
  Cosas como: Cable Coaxial, Fibra Óptica, Cable Trenzado o UTP
- **Inalámbricas ó No Guiado**:
  Cosas como: Infrarrojo, LI-FI, WI-FI, Bluetooth

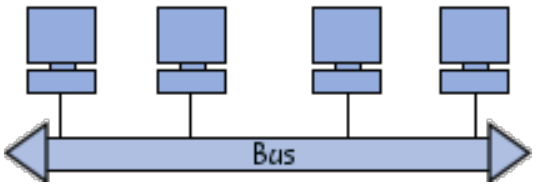
1.2.4. Por el Tipo de Transferencia

- **Simplex:** Si es que solo se puede enviar información en un sentido
- **Half-Duplex:** Si es que se puede enviar información en ambos sentidos, pero solo una a la vez.
- **Full-Duplex:** Si es que se puede enviar información en ambos sentidos, incluso ambos a la vez.

1.3. Topología de Redes

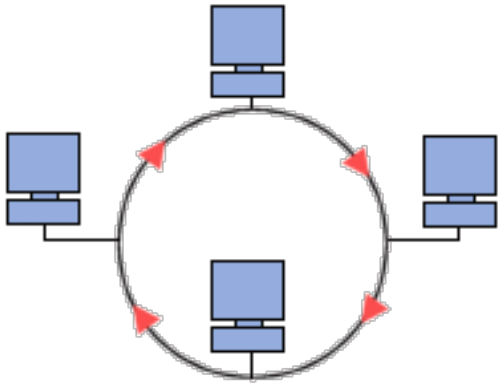
- **Bus**

  Todos el mismo medio de transmisión. Tampoco se pueden intentar comunicar más de 2 equipos al mismo tiempo, sino colecciones. No se necesita más que entrar al medio para poder permanecer a la Red. Son muy comunes. Si se daña el medio toda la red se viene abajo.



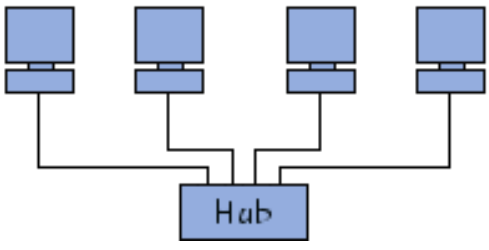
- **Anillo**

  Es muy seguro, no se puede añadir otro equipo fácilmente y no es bidireccional.



- **Estrella**

  Todos los equipos que están unidos a un mismo switch tendrán que enviar toda su información hacia el switch.



- **Árbol**

  Es una extensión natural de la topología de árbol, es decir, es lo mismo de que topología de estrella de una topología de estrella, es decir, donde cada elemento de la topología de forma de estrella aquí esta dado por una topología de estrella en si misma.

- **Malla**

  Es bueno... bueno una malla, donde no hay un solo camino ni forma de contactar a cada equipo. Es la mas robusta de todas.

Capítulo 2

Manejo de Errores

2.1. Chequeo de Redundancia Cíclica: CRC

2.1.1. Definición

Se usa en las tramas de acceso a la red. Un CRC es checksum que puede detectar la corrupción de datos que se almacenan y / o transmiten entre computadoras.

En términos generales, los CRC se calculan más eficientemente en hardware dedicado.

Dado un mensaje **M(X)** un mensaje de tamaño **m** bits a ser transmitido, se le adiccionan **n** bits de redundancia para formar una trama de **T(X)** de tamaño **m** + **n** la cual esta formada de la siguiente forma:

$$T = 2^n M + FCS$$

2.1.2. Componentes

- **FCS:** Es la secuencia comprobación de la trama.

  Se genera a partir del residuo de dividir $\frac{2^n M}{p}$ con n siendo el grado del polinomio.

  Nota que no llevaremos acarreo.

- **P:** Es un polinomo generador de la redundancia, de orden **n** y de tamaño **n+1** bits

Entonces enviamos T y si es que al llegar a la tarjeta de red $\frac{T}{P}$ nos deja un residuo de cero quiere decir que la trama es valida.

Nota que el polinomio tiene que cumplir con que su primer y ultimo bit sea 1.

2.1.3. Ejemplos

Ejemplo 1:

Dado $M(x) =' 1010001101'$ y $P(x) = 110101$

Entonces $P(x)$ simplemente es la simplificación de $P(x) = 1x^5 + 1^4 + 0x^3 + 1x^2 + 0x^1 + 1x^0$

Por lo tanto tenemos que:

$$
\begin{aligned}
T &= 2^n M + FCS && \text{Por definición} \\
&= 2^5(1010001101) + FCS && \text{Por Ahora multiplicamos por } 2^5 \\
&= (101000110100000) + FCS && \text{Manejo de bits } M << 5 \\
&= (101000110100000) + res\left(\frac{2^n M}{p}\right) && \text{Por definición} \\
&= (101000110100000) + rep\left(\frac{101000110100000}{110101}\right) && \text{Por definición} \\
&= (101000110100000) + res((110101)(1101010110) + 01110) && \text{Nota que el residuo tiene que tener } n \text{ bits} \\
&= 101000110101110 && \text{Magia}
\end{aligned}
$$

2.2. Hamming

Nada habla mas de un código :3

```python
import json, math

def CreateHammingCode(RawData, Show = False):

    Data = []
    DataIterator = 0
    ActualPow = 0
    RawDataIterator = 0

    while True:
        if (DataIterator + 1) == (2**ActualPow):
            Data += "X"
            ActualPow += 1
            DataIterator += 1
        else:
            Data += RawData[RawDataIterator]
            RawDataIterator += 1
            DataIterator += 1

        if (RawDataIterator == len(RawData)):
            break

    if Show: print(f"Parity Bits: {''.join(Data)}")
    if Show: print(f"Parity Bits: {ActualPow}")

    for ParityBit in range(0, ActualPow):

        CounterOf1 = 0
        for Bit, BitValue in enumerate(Data):
            if (Bit + 1) >> ParityBit & 1 == 1:
                if (Bit != (2**ParityBit - 1)):
                    if (Data[Bit] == "1"): CounterOf1 += 1

        if (CounterOf1 %2 == 0):
            Data[(2**ParityBit - 1)] = "0"
        else:
            Data[(2**ParityBit - 1)] = "1"

    return ''.join(Data)

def CheckErrorHammingCode(Data, Show = False):

    ErrorAdress = 0
    NumberOfParityBits = math.ceil(math.log2(len(Data)))

    for ParityBit in range(0, NumberOfParityBits):
        CounterOf1 = 0

        for Bit, BitValue in enumerate(Data):
            if (Bit + 1) >> ParityBit & 1 == 1:
                if (Data[Bit] == "1"): CounterOf1 += 1

        if (CounterOf1 %2 != 0):
            ErrorAdress += 2**(ParityBit)

    if Show: print(f"Error Adress: {ErrorAdress}")

    if ErrorAdress == 0:
        return Data

    Data = list(Data)

    if Data[ErrorAdress - 1] == "1":
        Data[ErrorAdress - 1] = "0"
    else:
        Data[ErrorAdress - 1] = "1"

    return ''.join(Data)

with open('HammingData.json', encoding='utf-8') as DataFile:
    HammingData = json.loads(DataFile.read())

    RawData = HammingData["ToEncodeByHamming"]
    DataToCheck = HammingData["ToCheckByHamming"]

    print(CreateHammingCode(RawData, True))
    print(CheckErrorHammingCode(DataToCheck, True))
```

2.3. Suma de Comprobación: CheckSum

Este algoritmo permite verificar la integridad de la PDU y su calculo es de la siguiente manera:

- Ordena los datos en palabras de 16 bits
- Poner ceros en la posición del checksum y sumar con acarreos
- Suma cualquier acarreo fuera de los 16 bits
- Complementar a uno

2.4. Bit de paridad

Es un bit extra a agregar, el total de los bits debe ser par o impar, se rellena a necesidad para completar la paridad o desacompletarla. Se indica el tipo de paridad.

Capítulo 3

Control de flujo

3.1. Parar y Esperar

3.2. Ventana Deslizante

3.2.1. Retroceder N

3.2.2. Rechazo selectivo

Capítulo 4

Enrutamiento

4.1. Estático

4.2. Dinámico

A traves de unos protocolos.

4.2.1. Dijsktra

4.2.2. Bellman Ford

Parte II

Protocolos

Capítulo 5

En General

5.1. Definiciones

Decimos que un Protocolo es un conjunto de reglas que regulan el intercambio de información entre entes.

5.1.1. Características

- **Sintaxis:**

  Es el formato de los datos, codificación y niveles de señal.

- **Semántica:**

  Información de control para manejo de errores.

- **Temporización:**

  Sincronización de velocidades y secuencias.

Capítulo 6

Protocolo OSI

6.1. Definición

Antes que nada, es un modelo de referencia. Pretende que los sistemas que son diseñados con base en el, se pueden comunicar sin problemas.

6.2. Partes

- **Capa Física:**

  Se encarga de la transmición de datos, de cadenas de bits no estrucurados sobre el medio físico y esta relacionado con:

  - Voltaje necasario para representar cada bit
  - Cuanto dura cada símbolo, es decir el tiempo de trama
  - Si se realiza simultáneamente en ambos sentidos o no
  - Como se establece una transmición y como interrumpirla
  - Especificar como serán los pines del conector de red, para que sirve cada pin pues

- **Capa de Enlace de Red:**

  Trabaja con direcciónes físicas.

  Proporciona el servicio de transferencia de datos (tramas) llevando a cabo la sincronización y correción de datos, así como el control de flujo.

- **Capa de Red:**

  Trabaja con IP (es decir, direcciones lógicas) para poder conectar dos redes. Es responsable del establecimiento, mantenimiento y cierre de conexión.

  También brinda las funciones de direccionamiento lógico y enrutamiento.

  Se direccionan de manera lógica y no física para evitar problemas con el hardware.

- **Capa de Transporte:**

  Hablaremos de si será un archivo orientado a conexión (TCP), o si no esta orientado a conexión (UDP), es decir la importancia que le damos a si queremos los datos integros o si requerimos gran velocidad.

  Proporciona seguridad, transferencia y transporte de datos entre los puntos finales también proporcionan mecanismos de control de flujo y de errores en el origen y destino.

  Esta proporciona el control de la comunicación entre diferentes aplicaciones, establece, gestiona y cierra la comunicación entre aplicaciones

- **Capa de Sesión:**

  Hablaremos los números de puerto, un identificador de programa que nos permite ejecutar varias aplicaciones al mismo tiempo, estas son permite ejecutar unos 65,536 aplicaciones en red TCP y otros 65,536 en UDP. Si, un montón.

- **Presentación:**

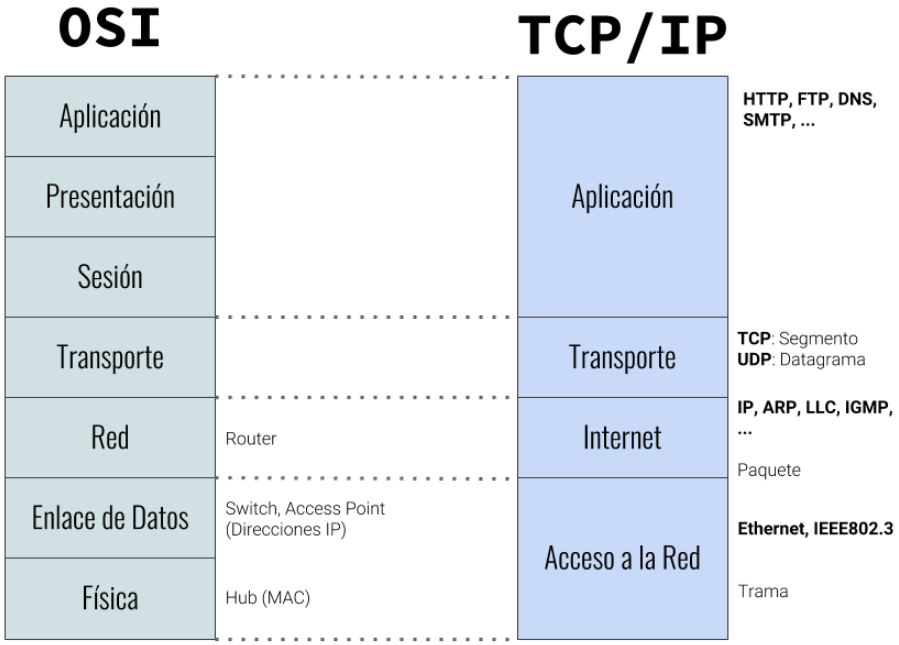
  Poder transmitir los datos de manera transparte y sin importar la arquitectura de las computadoras de origen y destino.

- **Aplicación:**

  Es donde trabajamos a nivel usario y ... poquito más.

  Proporciona un medio a los programas de aplicación para acceder a los servicios de red, contiene funciones de administración de aplicaciones distribuidas.

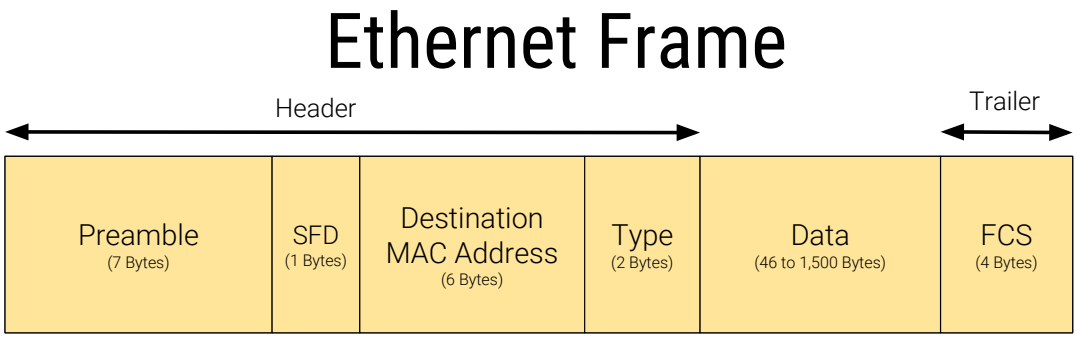
6.3. Diagrama OSI (Y su comparación vs IP)



Capítulo 7

Protocolo Ethernet

7.1. Frame



7.1.1. Explicación

Los campos que componen una trama ethernet son los siguientes:

- **Preámbulo (Preamble)**

  Campo con una secuencia de bits utilizada para sincronizar y estabilizar el medio físico antes de iniciar la transmisión. Es una secuencia de unos y ceros. conocida que permite a los nodos saber que esta llegando un nuevo frame.

  El patrón es el siguiente:

  10101010101010101010101010101010101010101010101010101010101010

  Tamaño: 7 Bytes

- **SFD (Start Frame Delimiter)**

  Delimitador de inicio de trama. Campo que contiene la secuencia 10101011.

  Indica el inicio de una trama de datos.

  Tamaño: 1 Byte

- **Dirección de Destino (Destination Address)**

  Campo que contiene la dirección MAC a la que se envía la trama.

  El bit más a la izquierda del campo indica cuando la dirección es individual (indicado por un 0) o un grupo de direcciones (indicado por un 1).

  El segundo bit desde la izquierda indica cuando la dirección destino es globalmente administrada (indicado por un 0).

La capa de enlace de datos del remitente añade la dirección de destino a la trama. La capa de enlace de datos del destinatario examina la dirección de destino para identificar los mensajes a recibir.

Tamaño: 6 Bytes

- **Dirección de Origen (Source Address)**

  Campo que contiene la dirección MAC del dispositivo que envía la trama.

  La dirección de origen es siempre una dirección individual y el bit más a la izquierda es siempre 0.

  Con ella el receptor conoce a quien debe dirigir las respuestas del mensaje.

  Tamaño: 6 Bytes

- **Tipo de Protocolo o Longitud**

  Este campo es el que distingue a las tramas IEEE 802.3 de las tramas Ethernet.

  Valores para este campo iguales o menores de x05DC (1500 en decimal) indican que es una trama IEEE 802.3 y el valor representa la longitud del campo de datos.

  Valores para este campo iguales o mayores de x0600 indican que es una trama Ethernet y el valor representa el tipo de protocolo.

  Tamaño: 2 Bytes

- **Datos and Pad(Payload)**

  Contiene los datos a transferir entre origen y destino. Si este campo fuera menor de 46 bytes se añade un campo de "relleno", es decir pad para mantener el tamaño mínimo de paquete.

  Tamaño: 46 a 1,500 Bytes

- **FCS (Frame Check Sequence)**

  Secuencia de verificación de trama.

  Campo que contiene un valor de para control de errores, CRC (Cyclical Redundancy Check). La verificación de redundancia cíclica (CRC), consiste en un valor calculado por el emisor que resume todos los datos de la trama. El receptor calcula nuevamente el valor y, si coincide con el de la trama, entiende que la trama se ha

  El campo FCS es generado ó calculado sobre los campos dirección de destino, la dirección de origen, el tipo/longitud y datos.

  Tamaño: 4 Bytes

Capítulo 8

Protocolo IP

8.1. Definiciones

Debido a la cantidad de cables necesarios para conectar cada red con cada otra red del mundo no todas las redes tienen una conexión directa, es decir, no existe un cable entre tu red lócal y los servidores de Facebook por ejemplo.

Por eso existe el Protocolo IP que nos permite comunicarnos entre redes.

En resumén lo que permite es que tu red local solo este conectada a unas pocas redes y a varios routers, estos tienen algo llamado una tabla de direcciones, que les permite navegar entre redes hasta encontrar su destino.

El enrutamiento es parecido a la recursión, en el sentido en que no soluciona tu problema sino que solo te lleva un paso más cerca.

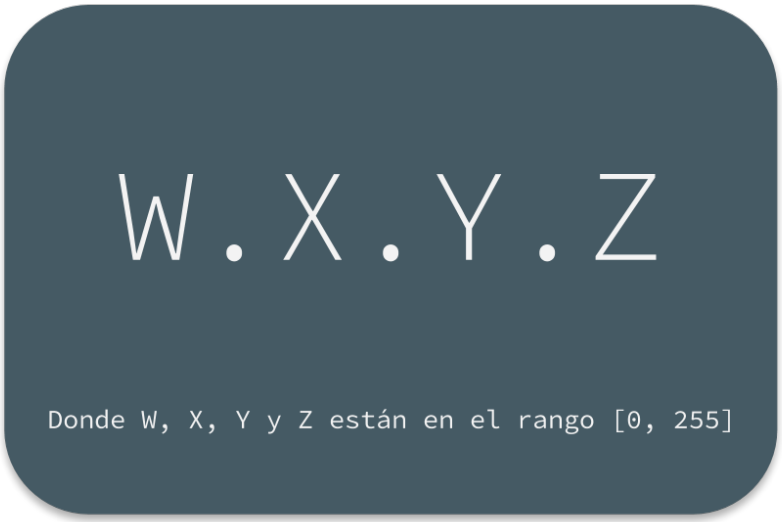
Direcciones IP

Es un identificador único (o casi, ya verás después porque). Necesitamos un identificador único porque es lo que nos permite enviar información y que la información que esperamos de regreso sepa a donde llegar.

8.2. Dirección IPv4

Como fue originalmente desarrollado este esquema podría alocar un identificador de **32 bits** a cada dispositivo que se quisiera conectar a internet. Esto nos daría algo así como 4 mill millones de posibles direcciones IP.

La convención es que estos serían representados como 4 conjuntos de 8 bits representados en decimal (una forma un poquito más amigable al público general), es decir:



Por ejemplo una IP v4 válida podría ser 140.247.220.12.

8.2.1. Problemas con IPv4

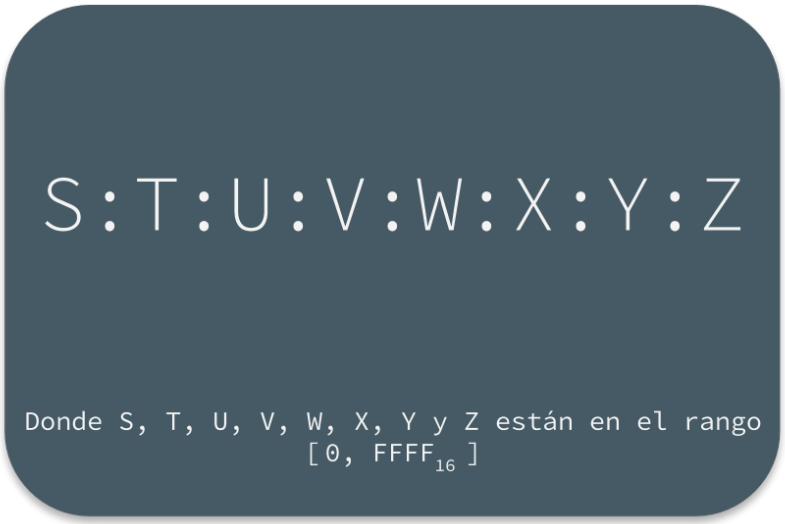
Ahora, recuerda que te dige que IP v4 acepta unos 4 mil millones de direcciones válidas, ahora el problema es que ahora mismo hay vivos mas de 7 mil millones de personas (A principios del siglo XXI) cada una con seguramente más de un dispositivo que quieran conectar a internet.

Por lo tanto tenemos que encontrar una forma de solucionar esto.

8.3. Dirección IPv6

Como vimos antes, ahora que parece que la cantidad de direcciones IPv4 se nos esta quedando corta, poco a poco estamos pasando de IPv4 a IPv6 que contará con nada menos y nada mas que **128 bits** para una dirección, es decir nos permitirá tener unas más o menos: $340,282,366,920,938,463,463,374,607,431,768,211,456$ posibles direcciones IP. Un chingo.

La convención es que estos serían representados como 8 conjuntos de 65535 bits representados en hexadecimal (porque de otra manera sale un númerote), es decir:



Por ejemplo una IPv6 podría ser $2001:0DB8:0000:0042:0000:8A2E:0370:7334$

8.3.1. Haciendo un poco más faciles las Direcciones IPv6

Ahora, todo esta mucho mejor que con IPv4, pero tenemos un pequeño problema, sus direcciones son moustrosamente enormes, por lo que tuvimos que hacer algunas simplificaciones para los humanos:

- Ignora los ceros dentro de cada grupo de 4 dígitos hexadecimales:

  Ejemplo:

  De $2001:0DB8:0000:0042:0000:8A2E:0370:7334$
  a $2001:0DB8:0:42:0:8A2E:370:7334$

- Si tienes un montón de ceros pon :: y da por sentado que quien lee esta dirección tiene cerebro y puede entender que ahí van ceros:

  Ejemplo:

  De $2001:0DB8:0000:0042:0000:0000:0000:0000$
  a $2001:0DB8:0000:0042::$

8.4. Clases IP

En general, el sistema de direccionamiento IPv4 se divide en cinco clases de direcciones IP. Todas las cinco clases están identificadas por el primer octeto de dirección IP.

8.4.1. Clases A

La Clase A son todas cuya representación en binario empieza por 0, es decir las direcciónes en el rango:

$0.X.X.X$ a $126.X.X.X$

El primer octeto es de la forma $N.H.H.H$, donde $N = Red$ y $H = Host$

La máscara de red entonces por defecto es $255.X.X.X$

La Clase A puede tener:

- 126 Redes $(2^7 - 2)$
- 16777214 Hosts $(2^{24} - 2)$

Los rangos $127.X.X.X$ se usan para loopback y para diagnóstico

8.4.2. Clases B

La Clase B son todas cuya representación en binario empieza por 10, es decir las direcciónes en el rango:

$128.0.X.X$ a $191.255.X.X$

El primer octeto es de la forma $N.N.H.H$, donde $N = Red$ y $H = Host$

La máscara de red entonces por defecto es $255.255.X.X$

La Clase B puede tener:

- 16384 Redes $(2^{14} - 2)$
- 65534 Hosts $(2^{16} - 2)$

8.4.3. Clases C

La Clase C son todas cuya representación en binario empieza por 10, es decir las direcciónes en el rango:

$192.0.0.X$ a $223.255.255.X$

El primer octeto es de la forma $N.N.N.H$, donde $N = Red$ y $H = Host$

La máscara de red entonces por defecto es $255.255.255.X$

La Clase C puede tener:

- 2097152 Redes $(2^{21} - 2)$
- 254 Hosts $(2^8 - 2)$

8.5. Subredes

Las clasificación original de las dirección IPv4 provocan un gran desperdicio.

Ejemplo:

Sea la empresa ABC que se conforma de los sig departamentos.

- Ventas 25 Host.
- Mercadotécnia 4 Host.
- Producción 50 Host.
- Recursos Humanos 10 Host.

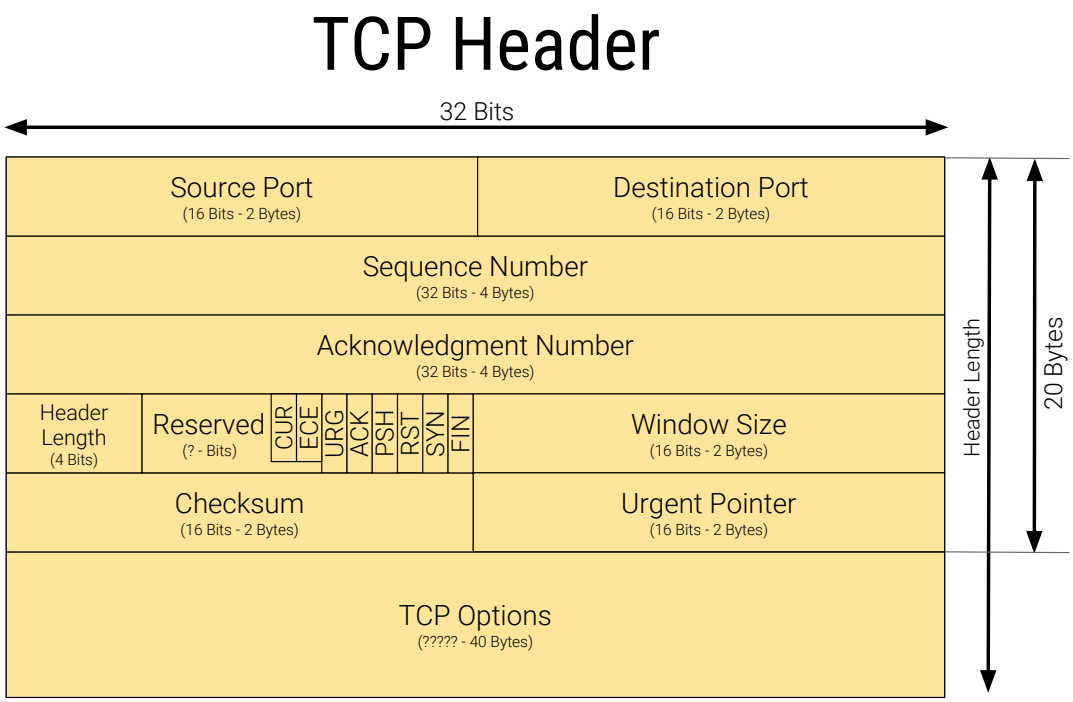
El esquema de direcciones podría ser :

- Opción A: 200.0.0.0/24
  Total Host: 89.
  Esto implica que todos los host estan en una misma clase C y ademas de cualquier área pueden entrar a los demás host.
  Sin mencionar el enorme desperdicio de host.
- Opción B: 1 Red de clase C para cada departamento.
  Ventas 200.0.0.0/24
  Mercadotécnia 200.0.1.0/24
  Producción 200.0.2.0/24
  Recursos Humanos 200.0.4.0/24
- Opción C: Dividir una red de la clase C en varias subredes.
  - 200.0.0.0/26

Capítulo 9

Protocolo TCP

9.1. Header - Encabezado



9.1.1. Explicación

- **Source Port - Puerto de Origen**

  Es solo el número de puerto de origen.

  Tamaño: 16 bits - 2 bytes

Capítulo 10

Protocolo UDP

Capítulo 11

DHCP

Espera, espera ... ¿Cómo obtengo mi dirección IP?

11.1. Introducción

Este es un protocolo que nos permite dar una dirección IP a cada dispositivo. Esta dado por *Dynamic Host Configuration Protocol*

Después de todo, si no existiera este protocolo, ¿Cómo obtengoo una dirección IP? No puedo tomar la que quiera, porque sino puede que haya más de una persona que piense como yo y eliga mi misma dirección. Y si sabes lo que es que alguien en tu calle tenga el mismo número que tu, sabes que ahí tienes un problema.

Capítulo 12

DNS

Espera, espera ... ¿y cuál es la IP de Cats.com?

12.1. Introducción

Este es una aplicación que nos permite hace una traducción de direcciones IP a nombres un poco más amigables para los seres humanos. Esta dado por *Domain Name System*. Es básicamente como las páginas amarillas del internet

Capítulo 13

Protocolo HDLC

Parte III

Aparatos Físicos

Capítulo 14

Hub

Capítulo 15

Switch

Capítulo 16

Routers

Capítulo 17

Access Points: Puntos de Acceso

Bibliografía

[1] Axel Ernesto Moreno Cervantes *Redes de Computación*. ESCOM, 2018.

[2] Nidia Cortez. *Redes de Computadoras* ESCOM, 2018.