REDES DE COMUNICACIONES I

Práctica 2: Ethernet y ARP

Volver a: Prácticas

Objetivos de la práctica

- Familiarizarse con el protocolo Ethernet y con el protocolo ARP
- Entender las cabeceras y aprender a implementar un protocolo de comunicaciones

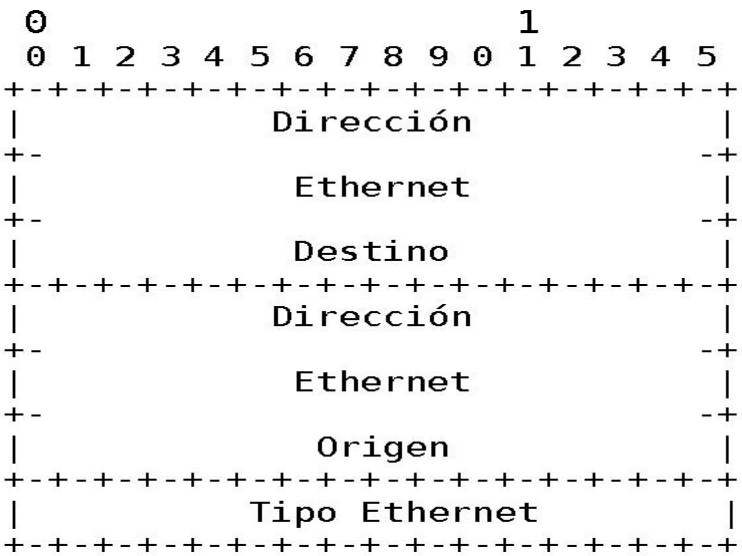
Introducción

Esta práctica tiene por objeto implementar los niveles inferiores de la pila de protocolos TCP/IP. Específicamente se construirá el nivel de enlace que en este caso se implementará usando el protocolo Ethernet. Sobre el nivel Ethernet se implementará el protocolo ARP que nos permitirá realizar traducciones entre direcciones de nivel de red y nivel de enlace. Ambos protocolos serán utilizados en la siguiente práctica para implementar el nivel de red (IP en nuestro caso).

A continuación analizaremos las cabeceras de Ethernet y ARP. Podemos obtener la descripción de cabeceras pedidas en sus respectivas RFCs (de aquí en castellano: http://www.rfc-es.org/) o estándares (http://www.ieee802.org/3/)

Ethernet (IEEE 802.3 Ethernet):

Ethernet es un protocolo usado en transmisiones de datos en redes de área local (LAN). La unidad básica de transmisión en este protocolo es la trama o *frame*. A continuación analizaremos el formato que tiene la cabecera de una trama Ethernet:



Campos:

- **Dirección Destino (6 bytes):** dirección Ethernet o MAC correspondiente al receptor de la trama. Si su valor es FF:FF:FF:FF:FF:FF se trata de una trama en difusión (***broadcast***) que está dirigida a todos los equipos de la red local.
- **Dirección Origen (6 bytes):** dirección Ethernet o MAC correspondiente al emisor.
- **Tipo Ethernet o *Ethertype*(2 Bytes):** identificador numérico que indica qué protocolo de nivel superior está encapsulado en la trama Ethernet actual. Los valores típicos que vamos a manejar para esta práctica son 0x0800 (IP) y 0x0806 (ARP). Otros tipos se reflejan en el siguiente documento: http://standards.ieee.org/develop/regauth/ethertype/eth.txt.

La cabecera Ethernet ocupa 14 Bytes y a continuación encontramos la carga útil o *payload* de la trama. Esta carga útil consiste en la cabecera y contenidos del protocolo de nivel superior que se haya especificado en el tipo de Ethernet.

Tamaños de las tramas:

- El tamaño mínimo de una trama Ethernet son 60 Bytes (excluyendo el CRC, que se genera en la tarjeta de red). Si el tamaño de la cabecera Ethernet más el contenido que se quiere enviar es menor a dicho límite se añade relleno (*padding*) hasta completar los 60 Bytes. Típicamente el relleno consiste en bytes con el valor 0.
- El tamaño máximo de una trama estándar Ethernet es 1514 Bytes (excluyendo nuevamente el CRC, que se genera en la tarjeta de red). Existen tramas de mayor tamaño llamadas *Jumbo Frames* que no se usarán en estas prácticas. A efectos de estas prácticas si el tamaño de la cabecera Ethernet más el contenido que se quiere enviar es mayor a 1514 supondremos que el envío no es posible.

ARP(RFC 826):

ARP es un protocolo que nos permite realizar traducciones entre direcciones de nivel de red y direcciones de nivel de enlace. En nuestro caso vamos a traducir direcciones IP (por ejemplo 192.168.1.1) a direcciones Ethernet (por ejemplo 00:AB:CD:EF:11:FF) dentro de una red local. Este protocolo es una pieza fundamental en las redes de comunicaciones ya que a la hora de enviar datos desde un equipo a otro utilizamos direcciones IP mientras que a nivel de enlace necesitaremos utilizar (típicamente) direcciones Ethernet.

Funcionamiento general:

Cuando un equipo quiere saber cuál es la dirección Ethernet asociada a una determinada dirección IP:

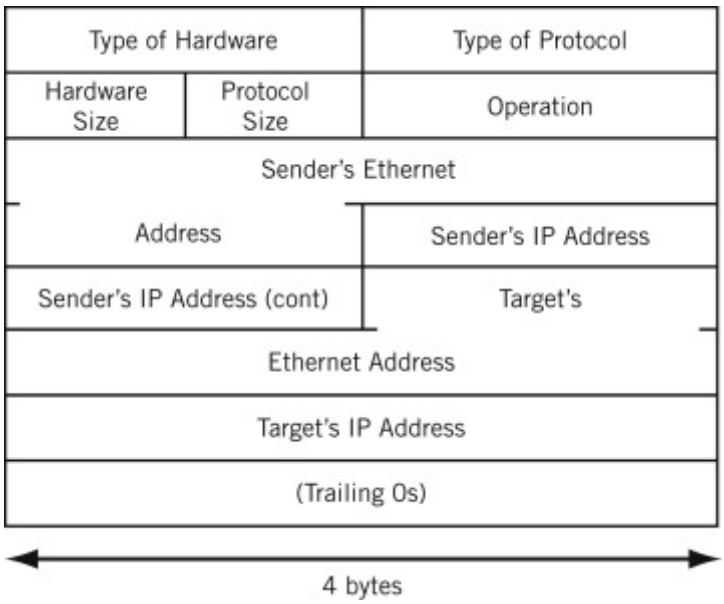
1. Envía una petición (*Request*) ARP en difusión (con dirección Ethernet destino = FF:FF:FF:FF:FF:FF) en la que especifica cual es su dirección Ethernet e IP y además especifica qué dirección IP quiere resolver. Por ejemplo si el equipo A (cuya MAC es 00:00:00:00:00:01 e IP 192.168.1.1) quiere saber cuál es la dirección MAC del equipo B (que tiene IP 192.168.1.2), mandará un paquete a todos los equipos (en difusión) preguntando "¿quién tiene la IP 192.168.1.2? Contestar a 192.168.1.1 con MAC 00:00:00:00:00:01".
2. Como este paquete ARP llega a todos los equipos de la red local, cuando llegue al equipo B (con IP 192.168.1.2 y MAC 00:00:00:00:00:02) dicho equipo generará una respuesta (*Reply*) ARP. Esta respuesta **NO** será en difusión ya que al conocer la dirección IP y MAC del equipo que pregunta puede responder de manera individual (*Unicast*). El paquete ARP de respuesta contestará "La dirección 192.168.1.2 está en 00:00:00:00:00:02".
3. El paquete de respuesta llegará al equipo A que extraerá la información y determinará que la MAC asociada a la IP 192.168.1.2 es 00:00:00:00:00:02.

Generalmente se espera que para una petición realizada se reciba una respuesta. Existe un caso especial de petición ARP llamada **ARP Gratuito** que realizan los equipos al inicializar su pila de red. Dicha petición consiste en preguntar por la propia dirección IP. Típicamente se espera que nadie conteste a dicha petición y si alguien contesta se determina que la propia IP está en uso por otro equipo.

Para evitar la repetición de peticiones ARP en un periodo corto de tiempo, típicamente se hace uso de una caché ARP con un temporizador. De este modo, si queremos hacer una resolución ARP y encontramos una entrada en la caché obtenemos los datos de ahí. En caso contrario hacemos la petición y si recibimos respuesta añadimos la respuesta a la caché para futuras consultas.

Cabecera:

El protocolo ARP tiene una cabecera fija que no dependerá del nivel de enlace ni del nivel de red y otra específica. Como en las prácticas únicamente traduciremos direcciones IP a direcciones Ethernet , a continuación se muestra la cabecera ARP particularizada para la combinación de direcciones de enlace Ethernet y direcciones de red IP:



Campos:

- **Hardware Type (2 Bytes):** Indica el tipo de direcciones de nivel de enlace. En el caso de direcciones Ethernet este valor es 0x0001.
- **Protocol Type (2 Bytes):** Indica el tipo de direcciones de nivel de red. En el caso de direcciones IP este valor es 0x0800.
- **Hardware Size (1 Byte):** Tamaño (en bytes) de las direcciones de nivel de enlace. En el caso de Ethernet este campo debe valer 6.
- **Protocol Size (1 Byte):** Tamaño (en bytes) de las direcciones de nivel de red. En el caso de IP este campo debe valer 4.
- **Opcode (2 Bytes):** Indicador de operación ARP. Si el paquete ARP es una petición (*Request*) este campo vale 0x0001 mientras que si es una respuesta (*Reply*) este campo vale 0x0002.
- **Sender Eth (6 Bytes):** Dirección Ethernet del emisor del paquete ARP.
- **Sender IP (4 Bytes):** Dirección IP del emisor del paquete ARP.
- **Target Eth (6 bytes):** Dirección Ethernet del receptor del paquete ARP. Si el Opcode del paquete es 1 este valor será la dirección nula (6 bytes a 0) ya que no se conoce cuál es la dirección Ethernet.
- **Target IP (4 Bytes):** Dirección IP del receptor del paquete ARP.

Ejercicios

El objetivo final de esta práctica es implementar un programa que haga resoluciones ARP. Para ello leerá por teclado una dirección IP, hará un petición ARP y si recibe respuesta mostrará la dirección Ethernet asociada. Para poder realizar estas acciones será necesario implementar Ethernet y ARP. En este sentido se propone partir del código que se proporciona, completar y ampliar el código entregado en los ficheros ethernet.py y arp.py. A continuación se describen las funciones que deben ser implementadas en ambos ficheros:

ethertnet.py:

startEthernetLevel(interface):

Descripción:

- Esta función recibe el nombre de una interfaz de red e inicializa el nivel Ethernet. Esta función debe realizar , al menos, las siguientes tareas:
  - Comprobar si el nivel Ethernet ya estaba inicializado (mediante una variable global). Si ya estaba inicializado devolver -1.
  - Obtener y almacenar en una variable global la dirección MAC asociada a la interfaz que se especifica
  - Abrir la interfaz especificada en modo promiscuo usando la librería rc1-pcap
  - Arrancar un hilo de recepción (rxThread) que llame a la función pcap_loop.
  - Si todo es correcto marcar la variable global de nivel incializado a True

Argumentos:

- **interface:** cadena de texto con el nombre de interfaz sobre la que inicializar el nivel Ethernet

Retorno:

- 0 si todo es correcto
- -1 en otro caso

stopEthernetLevel():

Descripción:

- Esta función parará y liberará todos los recursos necesarios asociados al nivel Ethernet. Esta función debe realizar, al menos, las siguientes tareas:
  - Parar el hilo de recepción de paquetes
  - Cerrar la interfaz (handle de pcap)
  - Marcar la variable global de nivel incializado a False

Argumentos:

- Ninguno

Retorno:

- 0 si todo es correcto
- -1 en otro caso

registerCallback(callback_func, ethertype):

Descripción:

- Esta función recibirá el nombre de una función y su valor de ethertype asociado y añadirá en la tabla (diccionario) de protocolos de nivel superior el dicha asociación. Este mecanismo nos permite saber a qué función de nivel superior debemos llamar al recibir una trama de determinado tipo. Por ejemplo, podemos registrar una función llamada process_IP_datagram asociada al Ethertype 0x0800 y otra llamada process_arp_packet asocaida al Ethertype 0x0806.

Argumentos:

- **callback_fun:** función de callback a ejecutar cuando se reciba el Ethertype especificado. La función que se pase como argumento debe tener el siguiente prototipo:
  - funcion(us,header,data,srcMac)
  - Dónde:
    - us: son los datos de usuarios pasados por pcap_loop (en nuestro caso este valor será siempre None)
    - header: estructura pcap_pkthdr que contiene los campos len, caplen y ts.
    - data: payload de la trama Ethernet. Es decir, la cabecera Ethernet NUNCA se pasa hacia arriba.

        - srcMac: dirección MAC que ha enviado la trama actual.
    - La función no retornará nada. Si una trama se quiere descartar basta con hacer un return sin valor y dejará de procesarse.
- **ethertype:** valor de Ethernetype para el cuál se quiere registrar una función de callback.

Retorno:

- Ninguno

sendEthernetFrame(data,len,etherType,dstMac):

Descripción:

- Esta función construirá una trama Ethernet con lo datos recibidos y la enviará por la interfaz de red. Esta función debe realizar, al menos, las siguientes tareas:
    - Construir la trama Ethernet a enviar (incluyendo cabecera + payload). Los campos propios (por ejemplo la dirección Ethernet origen) deben obtenerse de las variables que han sido inicializadas en startEthernetLevel
    - Comprobar los límites de Ethernet. Si la trama es muy pequeña se debe rellenar con 0s mientras que si es muy grande se debe devolver error.
    - Llamar a pcap_inject para enviar la trama y comprobar el retorno de dicha llamada. En caso de que haya error notificarlo

Argumentos:

- data: datos útiles o payload a encapsular dentro de la trama Ethernet
- len: longitud de los datos útiles expresada en bytes
- etherType: valor de tipo Ethernet a incluir en la trama
- dstMac: Dirección MAC destino a incluir en la trama que se enviará

Retorno:

- 0 si todo es correcto
- -1 en otro caso

process_Ethernet_frame(us,header,data):

Descripción:

- Esta función se ejecutará cada vez que llegue una trama Ethernet. Esta función debe realizar, al menos, las siguientes tareas:
    - Extraer los campos de dirección Ethernet destino, origen y ethertype
    - Comprobar si la dirección destino es la propia o la de broadcast. En caso de que la trama no vaya en difusión o no sea para nuestra interfaz la descartaremos (haciendo un return).
    - Comprobar si existe una función de callback de nivel superior asociada al Ethertype de la trama:
        - En caso de que exista, llamar a la función de nivel superior con los parámetros que corresponde:
            - us (datos de usuario)
            - header (cabecera pcap_pktheader)
            - payload (datos de la trama excluyendo la cabecera Ethernet)
            - dirección Ethernet origen
        - En caso de que no exista retornar

Argumentos:

- **us:** Datos de usuario pasados desde pcap_loop (en nuestro caso esto será None)
- **header:** estructura pcap_pkthdr que contiene los campos len, caplen y ts.
- **data:** bytearray con el contenido de la trama Ethernet

Retorno:

- Ninguno

arp.py:

initARP(interface):

Descripción:

- Esta función construirá inicializará el nivel ARP. Esta función debe realizar, al menos, las siguientes tareas:
  - Registrar la función del callback process_arp_frame (descrita más adelante) con el Ethertype 0x0806
  - Obtener y almacenar la dirección MAC e IP asociadas a la interfaz especificada
  - Realizar una petición ARP gratuita y comprobar si la IP propia ya está asignada. En caso positivo se debe devolver error.
  - Marcar la variable de nivel ARP inicializado a True

Argumentos:

- **interface:** cadena con el nombre de la interfaz dónde se debe inicializar el nivel ARP (por ejemplo h1-eth0)

Retorno:

- 0 si todo es correcto
- -1 en otro caso

ARPResolution(ip):

Descripción:

- Esta función intenta realizar una resolución ARP para una IP dada y devuelve la dirección MAC asociada a dicha IP o None en caso de que no haya recibido respuesta. Esta función debe realizar, al menos, las siguientes tareas:
  - Comprobar si la IP solicitada existe en la caché:
    - Si está en caché devolver la información de la caché
    - Si no está en la caché:
      - Construir una petición ARP llamando a la función createARPRequest (descripción más adelante)
      - Enviar dicha petición
      - Comprobar si se ha recibido respuesta o no:
        - Si no se ha recibido respuesta reenviar la petición hasta un máximo de 3 veces. Si no se recibe respuesta devolver None
        - Si se ha recibido respuesta devolver la dirección MAC
- Esta función necesitará comunicarse con el la función de recepción (para comprobar si hay respuesta y la respuesta en sí) mediante 3 variables globales:
  - awaitingResponse: indica si está True que se espera respuesta. Si está a False quiere decir que se ha recibido respuesta
  - requestedIP: contiene la IP por la que se está preguntando
  - resolvedMAC: contiene la dirección MAC resuelta (en caso de que awaitingResponse) sea False.
- Como estas variables globales se leen y escriben concurrentemente deben ser protegidas con un Lock

Argumentos:

- **interface:** ip a resolver

Retorno:

- **None** si no hay respuesta
- **MAC** asociada a la IP en caso de que haya respuesta

process_arp_frame(us,header,data,srcMac):

Descripción:

- Esta función procesa las tramas ARP. Se ejecutará por cada trama Ethenet que se reciba con Ethertype 0x0806 (si ha sido registrada en initARP). Esta función debe realizar, al menos, las siguientes tareas:
  - Extraer la cabecera común de ARP (6 primeros bytes) y comprobar que es correcta
  - Extraer el campo opcode
    - Si opcode es 0x0001 (Request) llamar a processARPRequest (ver descripción más adelante)
    - Si opcode es 0x0002 (Reply) llamar a processARPReply (ver descripción más adelante)
    - Si es otro opcode retornar de la función
    - En caso de que no exista retornar

Argumentos:

- us: Datos de usuario pasados desde la llamada de pcap_loop. En nuestro caso será None
- header: cabecera pcap_pktheader
- data: array de bytes con el contenido de la trama ARP
- srcMac: MAC origen de la trama Ethernet que se ha recibido

Retorno:

- Ninguno

processARPRequest(data,MAC):

Descripción:

- Esta función procesa una petición ARP. Esta función debe realizar, al menos, las siguientes tareas:
  - Extraer la MAC origen contenida en la petición ARP
  - Si la MAC origen de la trama ARP no es la misma que la recibida del nivel Ethernet retornar
  - Extraer la IP origen contenida en la petición ARP
  - Extraer la IP destino contenida en la petición ARP
  - Comprobar si la IP destino de la petición ARP es la propia IP:
    - Si no es la propia IP retornar
    - Si es la propia IP:
      - Construir una respuesta ARP llamando a createARPReply (descripción más adelante)
      - Enviar la respuesta ARP usando el nivel Ethernet (sendEthernetFrame)

Argumentos:

- **data:** bytearray con el contenido de la trama ARP (después de la cabecera común)
- **MAC:** dirección MAC origen extraída por el nivel Ethernet

Retorno:

- Ninguno

processARPReply(data,MAC):

Descripción:

- Esta función procesa una respuesta ARP. Esta función debe realizar, al menos, las siguientes tareas:
  - Extraer la MAC origen contenida en la petición ARP
  - Si la MAC origen de la trama ARP no es la misma que la recibida del nivel Ethernet retornar
  - Extraer la IP origen contenida en la petición ARP
  - Extraer la MAC destino contenida en la petición ARP
  - Extraer la IP destino contenida en la petición ARP
  - Comprobar si la IP destino de la petición ARP es la propia IP:
    - Si no es la propia IP retornar
    - Si es la propia IP:
      - Comprobar si la IP origen se corresponde con la solicitada (requestedIP). Si no se corresponde retornar

    - Copiar la MAC origen a la variable global resolvedMAC
    - Añadir a la caché ARP la asociación MAC/IP.
    - Cambiar el valor de la variable awaitingResponse a False
    - Cambiar el valor de la variable requestedIP a None
- Las variables globales (requestedIP, awaitingResponse y resolvedMAC) son accedidas concurrentemente por la función ARPResolution y deben ser protegidas mediante un Lock.

Argumentos:

- **data:** bytearray con el contenido de la trama ARP (después de la cabecera común)
- **MAC:** dirección MAC origen extraída por el nivel Ethernet

Retorno:

- Ninguno

createARPRequest(ip):

Descripción:

- Esta función construye una petición ARP y devuelve la trama con el contenido.

Argumentos:

- **ip:** dirección a resolver

Retorno:

- Bytearray con el contenido de la trama de petición ARP

createARPReply(ip,MAC):

Descripción:

- Esta función construye una respuesta ARP y devuelve la trama con el contenido.

Argumentos:

- **ip:** dirección IP a la que contestar
- **MAC:** dirección MAC a la que contestar

Retorno:

- Bytearray con el contenido de la trama de respuesta ARP

Adicionalmente se proporciona ya implementada la función (printCache) para imprimir la caché ARP.

Ejecución de pruebas:

Para realizar las pruebas durante el desarrollo y la validación final se hará uso de la herramienta Mininet. Se puede revisar el funcionamiento de dicha herramienta en el documento de entorno de trabajo al inicio de la sección de prácticas. Ejecutaremos al menos 2 tipos de pruebas:

1. Pruebas de resolución de direcciones:

1. Partiremos de la configuración básica en la que hay 2 hosts (h1 y h2).
2. Obtendremos las IPs y MACs de las interfaces h1-eth0 y h2-eth0 (por ejemplo, haciendo uso de la utilidad ifconfig). Para lo que sigue supondremos que h1-eth0 tiene IP 10.0.0.1 y MAC 00:00:00:00:00:01 y h2-eth tiene

IP 10.0.0.2 y MAC 00:00:00:00:00:02.
3. Ejecutaremos en las terminales de h1 y h2 el script practica2.py con el argumento --itf h1-eth0 y --itf h2-eth0, respectivamente.
4. Desde h1 introduciremos la IP de h2 (en nuestro ejemplo, 10.0.0.2) y se nos debe mostrar como respuesta la MAC de h2-eth0 (en nuestro ejemplo 00:00:00:00:00:02).
5. Acto seguido, escribiremos la letra p para mostrar por pantalla el contenido de la caché y comprobaremos que se ha añadido la MAC resuelta a la caché.
6. Volver a solicitar la dirección de IP de h2 y comprobar que no se realiza una petición ARP.

2. Pruebas de inicialización:

1. Partiremos de la configuración básica en la que hay dos hosts (h1 y h2).
2. Obtendremos las IPs y MACs de las interfaces h1-eth0 y h2-eth0 (por ejemplo, haciendo uso de la utilidad ifconfig) y cambiaremos la IP de uno de los hosts (por ejemplo h2).
3. Para lo que sigue supondremos que h1-eth0 tiene IP 10.0.0.1 y MAC 00:00:00:00:00:01 y h2-eth tiene IP 10.0.0.1 y MAC 00:00:00:00:00:02.
4. Ejecutaremos en las terminales de h1 y h2 el script practica2.py con el argumento --itf h1-eth0 y --itf h2-eth0, respectivamente.
5. Al ejecutar el script en la terminal de h2 debemos ver que el nivel ARP no inicializa ya que el ARP gratuito falla al tener las 2 interfaces la misma IP.

Para todas las pruebas se recomienda el uso de Wireshark para validar la correcta construcción y envio de tramas Ethernet

Criterios de evaluación

Ejercicios: Entrega especificada en la actividad de Moodle

- Normativa de entrega cumplida en su totalidad: 5%
- Fichero leeme.txt bien explicado: 5%
- Recibir tramas Ethernet, realizar comprobaciones y llamar correctamente a la función de callback de nivel superior 10%
- Enviar tramas Ethernet correctamente 10%
- Enviar correctamente peticiones ARP 10%
- Procesar correctamente peticiones ARP recibidas 15%
- Enviar correctamente respuestas ARP 10%
- Procesar correctamente respuestas ARP 15%
- Manejo correcto de la caché ARP 5%
- Uso correcto de Locks 5%
- Realiza correctamente el ARP gratuito 10%

Control individual: Cuestionario final sobre la práctica el día especificado en Moodle. No olvide ser puntual, el control empezará a "y 5".

Entrega

Archivos fuente completos **practica2.py, ethernet.py y arp.py**.

- Añada un archivo **leeme.txt** que incluya los nombres de los autores, comentarios que se quieran transmitir al profesor y, en caso de entregar algún archivo más, la descripción y/o explicación del mismo. **Además este fichero debe contener una sección donde se determine si se ha dado respuesta (Realizado/Parcialmente-Realizado/No-Realizado, y en caso afirmativo la explicación de cómo se ha validado) a cada criterio de evaluación solicitado.**

Comprima en un zip **TODO** lo que vaya a entregar y llámelo practica2_YYYY_PXX.zip, donde YYYY es el grupo al que pertenece (1301,1302,etc), y XX (y solo XX) es el número de pareja (**con dos dígitos**).

Por ejemplo, para la pareja 5 del grupo 1301: **$ zip practica2_1301_P05.zip ***

Solo es necesario que suba la entrega un miembro de la pareja.

Última modificación: viernes, 11 de octubre de 2019, 10:36

◄ Entrega P1

Material p2 ►

Volver a: Prácticas