



## Contenido

Ethernet .....	2
TECNOLOGÍA ETHERNET .....	2
ESTÁNDARES DE ETHERNET .....	2
LLC.....	2
MAC .....	2
LISTADO DE FABRICANTES según IEEE .....	4
ENCAPSULACIÓN DE DATOS .....	5
PROPORCIONA TRES FUNCIONES PRINCIPALES .....	5
Control de acceso al medio .....	6
Proceso de acceso múltiple por detección de portadora .....	6
(CSMA) .....	6
CSMA/Detección de colisión .....	7
Método de acceso al medio CSMA/Prevención de .....	7
colisiones (CSMA/CA).....	7
PROCESAMIENTO DE TRAMAS .....	7
Dirección MAC .....	8
Dirección IP.....	8
ENCAPSULAMIENTO ETHERNET .....	8
UNICAST .....	10
BROADCAST .....	11
MULTICAST .....	12
IMPLEMENTACION ETHERNET CON uC ATMEGA328+ SHIELD ETH CAYENNE.....	13
IMPLEMENTACION ETHERNET CON uC ATMEGA328+ SHIELD ETH UBIDOTS STEM .....	18
IMPLEMENTACION ETHERNET CON uC ATMEGA328+ SHIELD ETH CLOUDMQTT .....	24
REFERENCIAS.....	28



## Ethernet

La migración hacia redes de 1 Gbps es una necesidad del mercado, debido al incesante requerimiento de mayor ancho de banda en las redes informáticas, como consecuencia del incremento de aplicaciones multimedia y de transferencia de volúmenes de datos cada vez mayores.

Por otro lado, existe en el mercado una necesidad permanente de incrementar el ancho de banda en las redes, debido al requerimiento de mayor ancho de banda que las aplicaciones presentan y a la transmisión de señales isócronas (voz y video).

Es por ello que resulta de gran interés para la industria, la posibilidad de migrar hacia redes de 1 Gbps, las cuales son 10 veces más rápidas que las de 100 Mbps, incrementando la tasa de información y, en consecuencia, la performance de las aplicaciones institucionales que operan en dichas redes.

El empleo de un código banda base apropiado en las placas de 1 Gbps permite la implementación de redes LAN con fibra óptica mono modo y multi modo denominadas 1.000 Base LX y 1.000 Base SX.

### TECNOLOGÍA ETHERNET

- ✓ Tecnología LAN más utilizada.
- ✓ Opera en la capa de enlace de datos y en la capa física.
- ✓ Familia de tecnologías de redes que se define en los estándares IEEE 802.2 y 802.3.
- ✓ Admite anchos de banda de datos de 10, 100, 1000, 10 000, 40 000 y 100 000 Mbps (100 Gbps).

### ESTÁNDARES DE ETHERNET

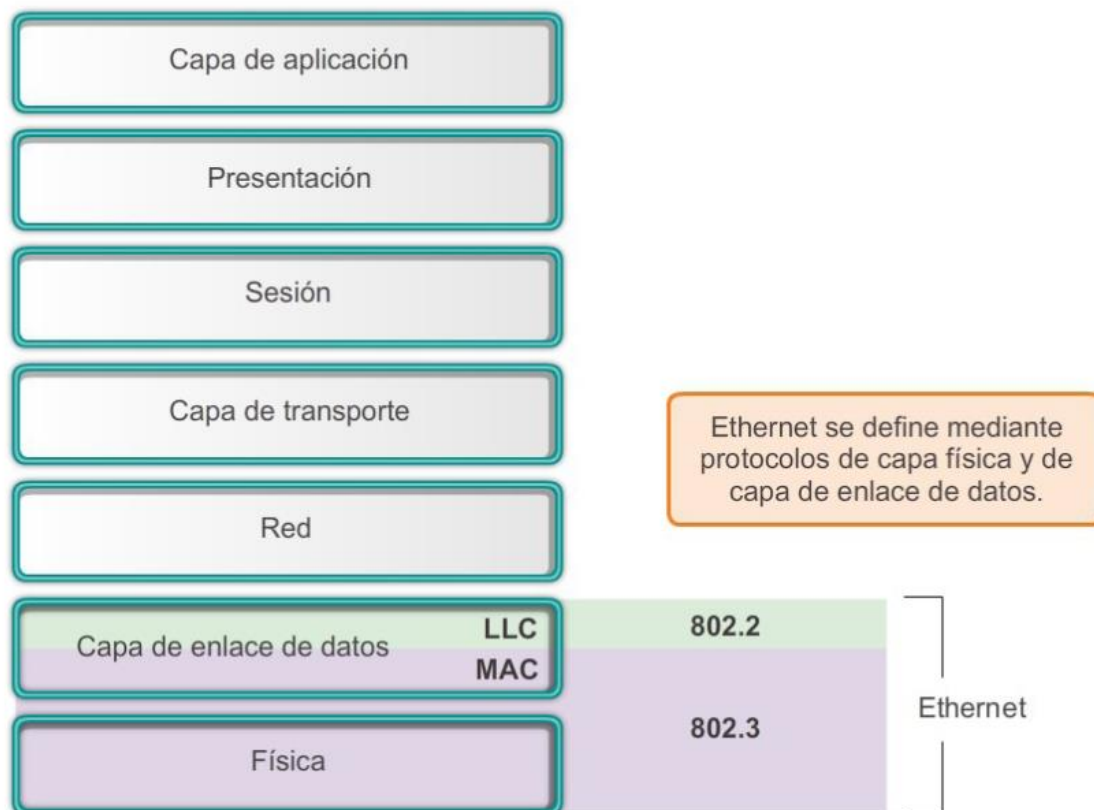
- ✓ Definen los protocolos de capa 2 y las tecnologías de capa 1.
- ✓ Operan en dos subcapas separadas de la capa de enlace de datos: La de control de enlace lógico (LLC) y la
- ✓ MAC.

### LLC

- ✓ Maneja la comunicación entre las capas superiores e inferiores.
- ✓ Toma los datos del protocolo de red y agrega información de
- ✓ control para ayudar a entregar el paquete al destino.

### MAC

- ✓ Constituye la subcapa inferior de la capa de enlace de datos.
- ✓ Se implementa mediante hardware, por lo general en la NIC de la PC.
- ✓ Tiene dos responsabilidades principales:
- ✓ Encapsulación de datos
- ✓ Control de acceso al medio



Fuente [2]

### MAC

Las tarjetas de red tipo Ethernet tienen una pequeña memoria en la que alojan un dato único para cada tarjeta de este tipo. Se trata de la dirección MAC, y está formada por 48 bits que se suelen representar mediante dígitos hexadecimales que se agrupan en seis parejas (cada pareja se separa de otra mediante dos puntos ":" o mediante guiones "-"). Por ejemplo, una dirección MAC podría ser F0:E1:D2:C3:B4:A5.

La mitad de los bits de la dirección MAC son usados para identificar al fabricante de la tarjeta, y los otros 24 bits son utilizados para diferenciar cada una de las tarjetas producidas por ese fabricante.



### LISTADO DE FABRICANTES según IEEE

- 02608C 3Com IBM PC; Imagen; Valid; Cisco
- 800010 AT&T
- 080069 Silicon Graphics
- 08002B DEC
- 080067 Comdesign
- 080046 Sony
- 00AA00 Intel
- 0080C2 IEEE 802.1 Committee
- 00000C Cisco
- 080009 Hewlett-Packard
- 08005A IBM
- 080020 Sun.
- 0000C0 Western Digital
- 00A03E ATM Forum
- 080003 ACC (Advanced Computer Communications).



Capa de Enlace de datos	Subcapa de control de enlace lógico								
	802.3: Control de acceso al medio								
Capa física	Subcapa de señalización física	10BASE-5 (500 m) 50 Ohm Coaxial tipo N	10BASE-2 (185 m) 50 Ohm Coaxial BNC	10BASE-T (100 m) 100 Ohm UTP RJ-45	100BASE-TX (100 m) 100 Ohm UTP RJ-45	1000BASE-CX (25 m) 150 Ohm STP mini-DB-9	1000BASE-T (100 m) 100 Ohm UTP RJ-45	1000BASE-ST (220-550m) MM Fiber SC	1000BASE-LX (550-5000m) MM or SM Fiber SC
	Medio físico								

Fuente [2]

## ENCAPSULACIÓN DE DATOS

- ✓ Armado de la trama antes de la transmisión y desarmado de la trama en el momento en que se la recibe.
- ✓ La capa MAC agrega un encabezado y un tráiler a la PDU de la capa de red.

## PROPORCIONA TRES FUNCIONES PRINCIPALES

- ✓ Delimitación de tramas: identifica un grupo de bits que componen una trama; sincronización entre los nodos emisor y receptor.
- ✓ Direccionamiento: cada encabezado Ethernet que se agrega a la trama contiene la dirección física (dirección MAC) que permite que la trama se entregue a un nodo de destino.
- ✓ Detección de errores: cada trama de Ethernet contiene un tráiler con una comprobación de redundancia cíclica (CRC) del contenido de la trama.



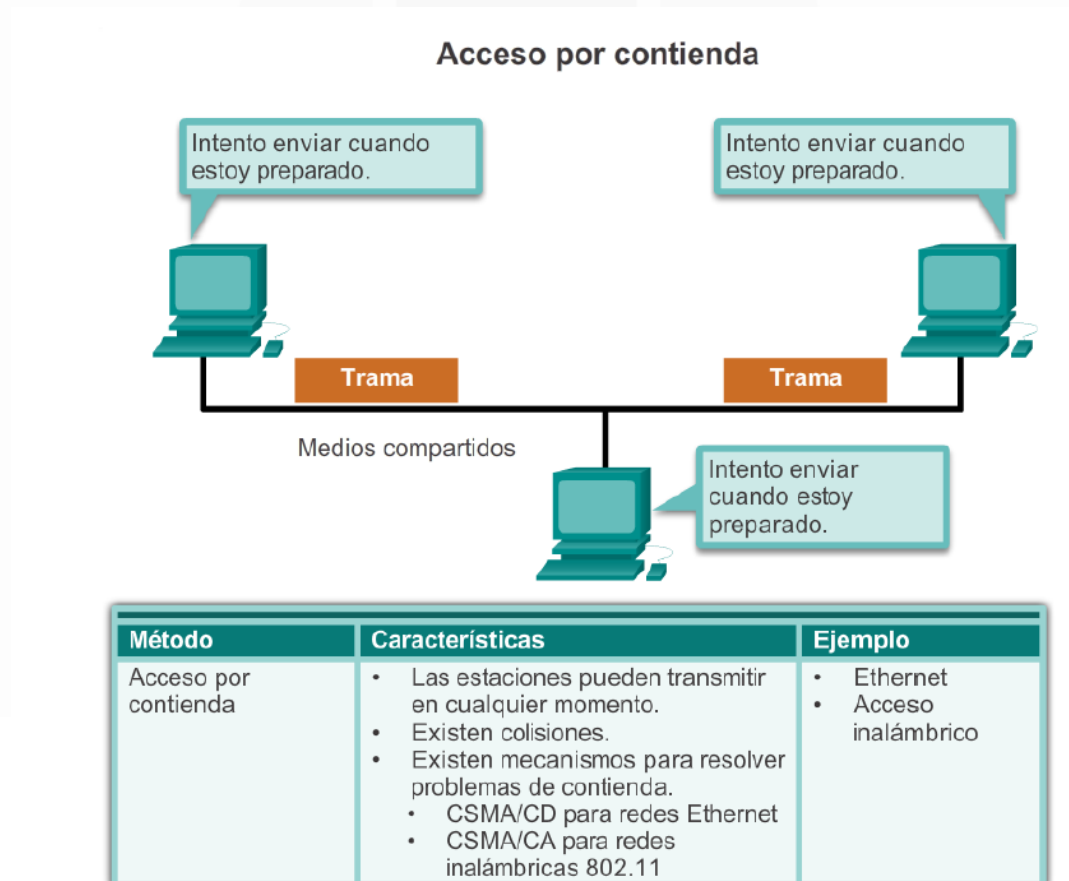
## Control de acceso al medio

- ✓ Responsable de la ubicación y la remoción de tramas en los medios.
- ✓ Se comunica directamente con la capa física.
- ✓ Si hay varios dispositivos en un único medio que intentan reenviar datos simultáneamente, los datos colisionan, lo que provoca que estos se dañen y no se puedan utilizar.
- ✓ Ethernet proporciona un método para controlar la forma en que los nodos comparten el acceso mediante el uso de una tecnología de acceso múltiple por detección de portadora (CSMA).

### Proceso de acceso múltiple por detección de portadora

#### (CSMA)

- ✓ En primera instancia, se utiliza para detectar si los medios transportan una señal.
- ✓ Si no se detecta una señal portadora, el dispositivo transmite sus datos.
- ✓ Si dos dispositivos transmiten al mismo tiempo, se produce una colisión de datos.



Fuente [2]



### CSMA/Detección de colisión

- ✓ El dispositivo controla los medios para detectar la presencia de una señal de datos.
- ✓ Si no hay una señal de datos, lo que indica que el medio está libre, el dispositivo transmite los datos.
- ✓ Si luego se detectan señales que muestran que otro dispositivo estaba transmitiendo al mismo tiempo, todos los dispositivos dejan de enviar y vuelven a intentarlo más tarde.
- ✓ Si bien las redes Ethernet se diseñan con tecnología CSMA/CD, con los dispositivos intermediarios actuales no se producen colisiones y los procesos utilizados por CSMA/CD son realmente innecesarios.
- ✓ Todavía se deben tener en cuenta las colisiones en conexiones inalámbricas en entornos LAN.

### Método de acceso al medio CSMA/Prevención de colisiones (CSMA/CA)

- ✓ El dispositivo examina los medios para detectar la presencia de una señal de datos. Si los medios están libres, el dispositivo envía una
- ✓ notificación a través de los medios sobre su intención de utilizarlos.
- ✓ El dispositivo luego envía los datos.
- ✓ Utilizado por las tecnologías de red inalámbricas 802.11.

Método	Características	Ejemplo
Acceso por contienda	<ul style="list-style-type: none"><li>• Las estaciones pueden transmitir en cualquier momento.</li><li>• Existen colisiones.</li><li>• Existen mecanismos para resolver problemas de contienda.<ul style="list-style-type: none"><li>• CSMA/CD para redes Ethernet</li><li>• CSMA/CA para redes inalámbricas 802.11</li></ul></li></ul>	<ul style="list-style-type: none"><li>• Ethernet</li><li>• Acceso inalámbrico</li></ul>

Fuente [2]

### PROCESAMIENTO DE TRAMAS

- ✓ Se asignan direcciones MAC a estaciones de trabajo, servidores, impresoras, switches y routers.
- ✓ Ejemplos de direcciones MAC: 00-05-9A-3C-78-00, 00:05:9A:3C:78:00 y 0005.9A3C.7800.
- ✓ Se reenvía el mensaje a una red Ethernet, se adjunta la información del encabezado al paquete que contiene la dirección MAC de origen y destino.
- ✓ Cada NIC revisa la información para ver si la dirección MAC de destino que está en la trama coincide con la dirección MAC física del dispositivo almacenada en la RAM.
- ✓ Si no hay coincidencia, el dispositivo descarta la trama.



- ✓ Si coincide con la dirección MAC de destino de la trama, la NIC pasa la trama a las capas OSI, donde tiene lugar el proceso de des encapsulación.

### Dirección MAC

- ✓ Esta dirección no cambia.
- ✓ Es similar al nombre de una persona.
- ✓ Se conoce como “dirección física” porque se asigna físicamente a la NIC del host.

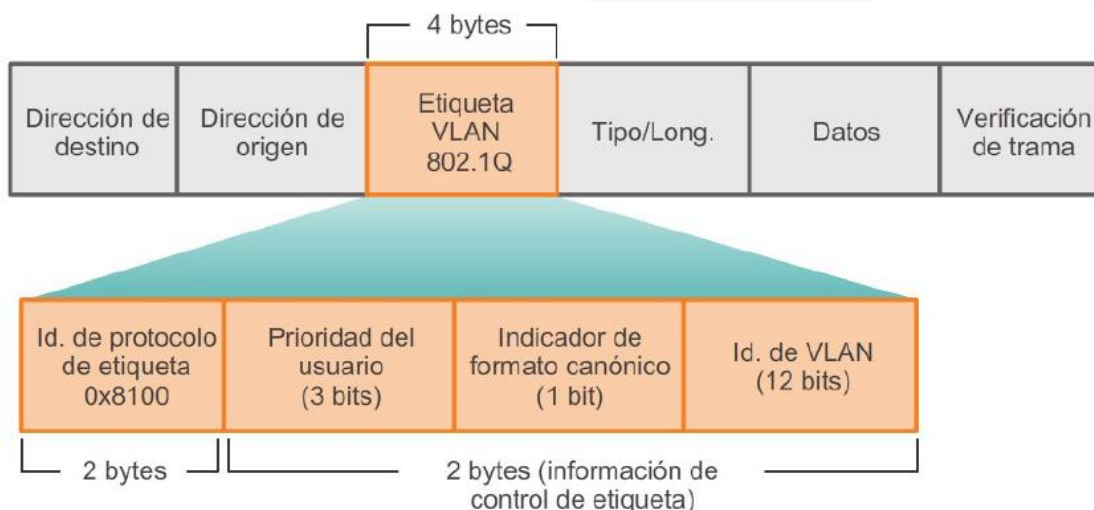
### Dirección IP

- ✓ Es similar a la dirección de una persona.
- ✓ Se basa en la ubicación real del host.
- ✓ Se conoce como “dirección lógica” porque se asigna lógicamente.
- ✓ Un administrador de red la asigna a cada host.

Para que una PC pueda comunicarse, se necesitan tanto la dirección MAC física como la dirección IP lógica, de la misma manera en que se necesitan el nombre y la dirección de una persona para poder enviarle una carta.

### ENCAPSULAMIENTO ETHERNET

- ✓ Los estándares Ethernet II e IEEE 802.3 definen la trama mínima en 64 bytes y la trama máxima en 1518 bytes.
- ✓ Una longitud menor que 64 bytes se considera un “fragmento de colisión” o “runt frame”.
- ✓ Si el tamaño de una trama transmitida es menor que el mínimo o mayor que el máximo, el dispositivo receptor descarta la trama.
- ✓ En la capa física, las diferentes versiones de Ethernet varían en cuanto al método para detectar y colocar datos en los medios.



Fuente [2]



**IEEE 802.3**

7	1	6	6	2	46 a 1500	4
Preámbulo	Delimitador de inicio de trama	Dirección de destino	Dirección de origen	Longitud	Encabezado y datos de 802.2	Secuencia de verificación de trama

**Campos Preámbulo y Delimitador de inicio de trama**  
Se utiliza para la sincronización entre los dispositivos emisor y receptor.

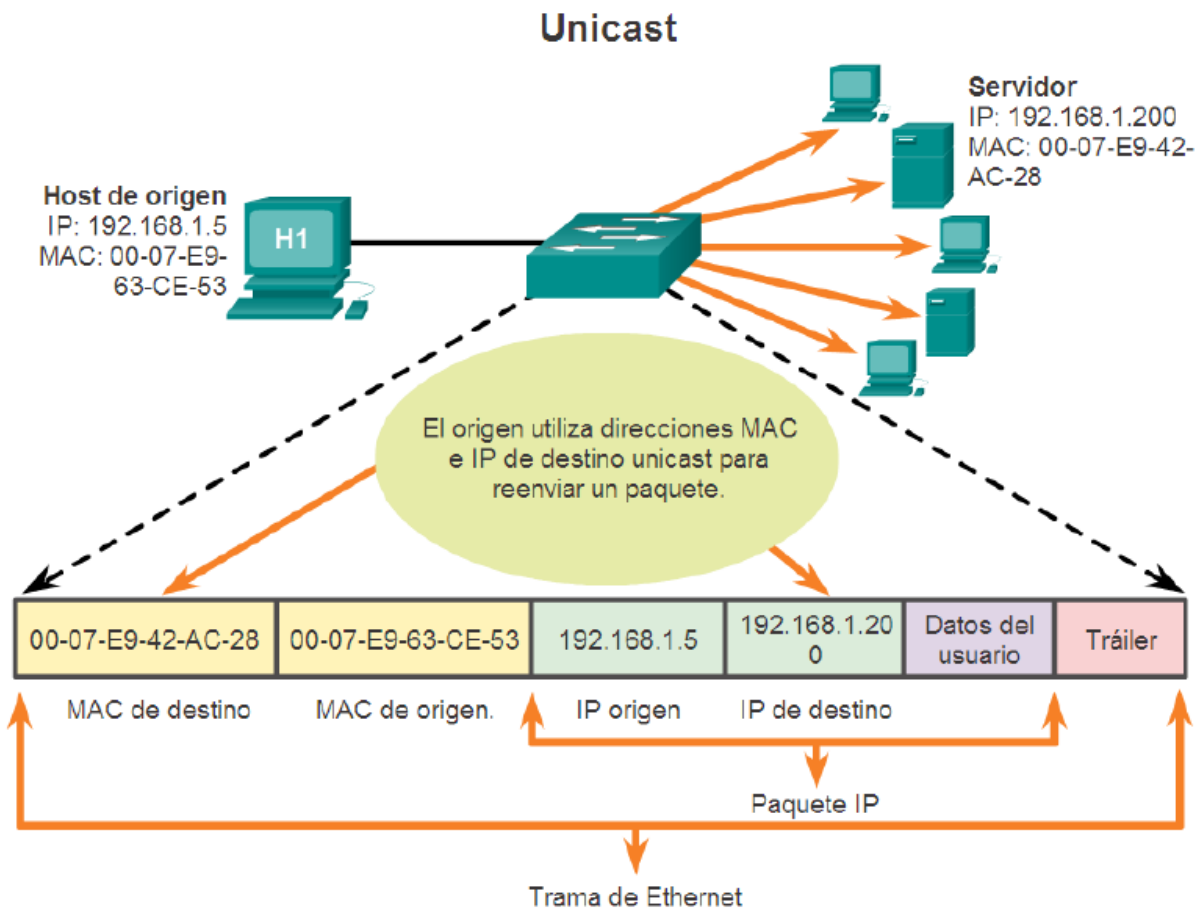
**Campo Longitud/tipo**  
Define la longitud exacta del campo de datos de la trama y describe qué protocolo se implementa.

**Campos Datos y Pad**  
Contienen los datos encapsulados de una capa superior, un paquete IPV4.

**Campo Secuencia de verificación de trama**  
Se utiliza para detectar errores en una trama con comprobación de redundancia cíclica (4 bytes); si los cálculos coinciden en el origen y el receptor, no se produjo ningún error.



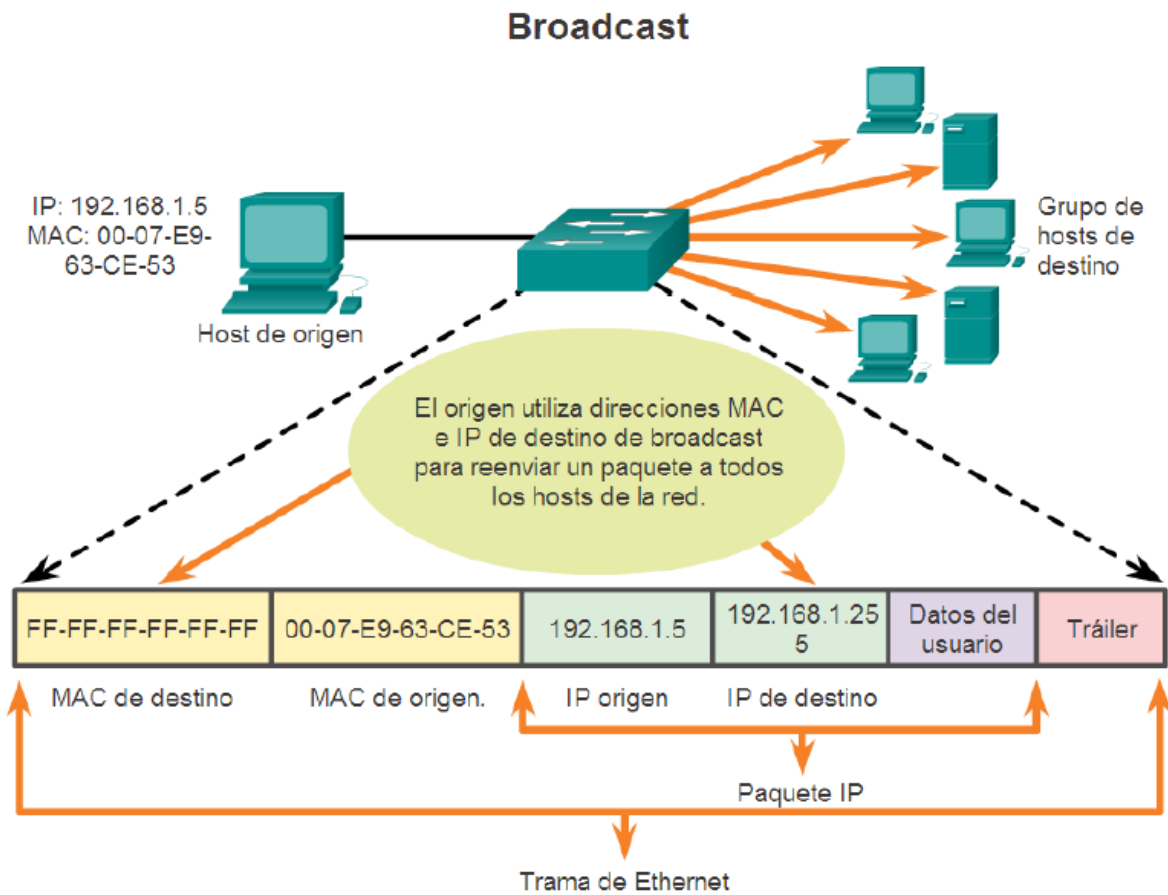
## UNICAST



Fuente [2]



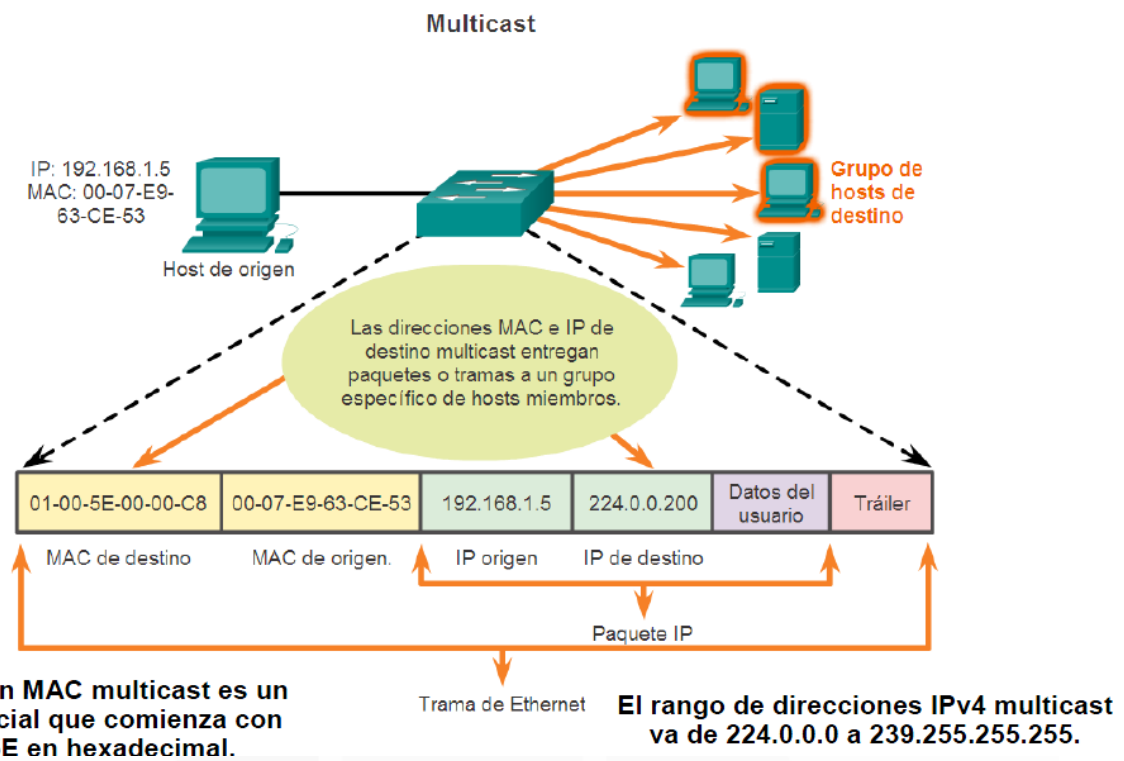
## BROADCAST



Fuente [2]



## MULTICAST



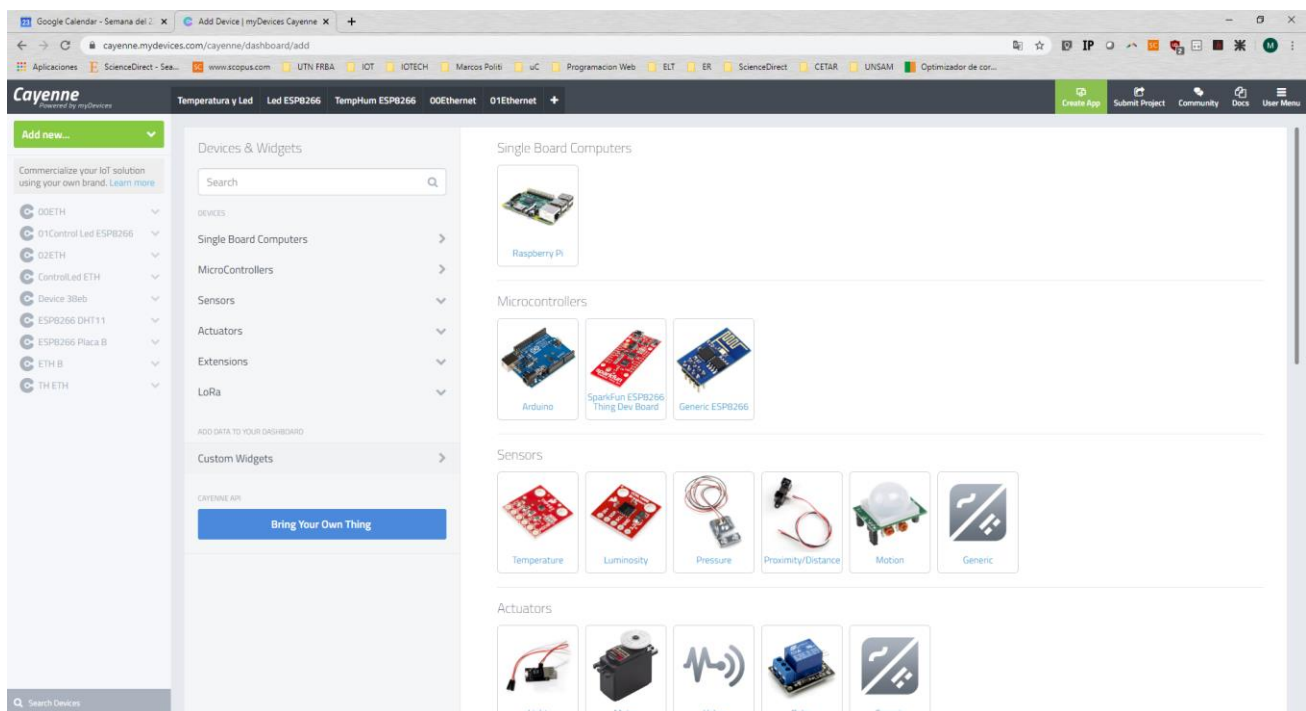
Fuente [2]



## IMPLEMENTACION ETHERNET CON uC ATMEGA328+ SHIELD ETH CAYENNE

Para realizar esta implementación vamos a usar el Arduino Uno+Shield Eth y el servicio Cayenne

Para tal fin ingresamos a la plataforma y elegimos ARDUINO



Seleccionamos luego la opción marcada en el caso que deseemos implementar el código directamente desde cayenne.



SELECT YOUR ARDUINO BOARD CONNECTION:

- Arduino Due
- Arduino Leonardo
- Arduino Mega
- Arduino Mini
- Arduino Nano
- Arduino Pro
- Arduino Pro Micro
- Arduino Pro Mini
- Arduino Uno
  - ☒ Ethernet Shield W5100
  - ☐ Ethernet Shield W5200
  - ☐ Ethernet Shield W5500
  - ☐ Manual Connection
  - ☐ Serial USB Connection
  - ☐ WiFi Shield
  - ☐ WiFi 101 Shield
  - ☐ Arduino MKR1000
  - ☐ Arduino ESP8266 WiFi
- Arduino Yun

MQTT USERNAME: 554cdf50-f0a9-11e7-abe9-1721c4c13600

MQTT PASSWORD: 598bec3b273b7116ce1f6db5387a53dd3b688f8

CLIENT ID: 5691fe60-6d6e-11ea-a38a-d57172a4b4d4

MQTT SERVER: mqtt.mydevices.com

MQTT PORT: 1883

NAME YOUR DEVICE (optional): Arduino

Waiting for board to connect...

NEED HELP?  
[Troubleshooting](#)  
[Download Cayenne library](#)  
[Installing Cayenne libraries](#)  
[Ask our community](#)

En ese caso la plataforma nos brindará el siguiente código para ingresarlo dentro de un sketch nuevo en el IDE de Arduino

```
/*
 * This example shows how to connect to Cayenne using an Ethernet W5100 shield and send/receive sample data.
 *
 * The CayenneMQTT Library is required to run this sketch. If you have not already done so you can install it from the Arduino IDE Library Manager.
 *
 * Steps:
 * 1. Set the Cayenne authentication info to match the authentication info from the Dashboard.
 * 2. Compile and upload the sketch.
 * 3. A temporary widget will be automatically generated in the Cayenne Dashboard. To make the widget permanent click the 'Add' button.
 */

// Define Cayenne debug messages
#define CAYENNE_DEBUG // Uncomment to show debug messages
#define CAYENNE_PRINT Serial // Comment this out to disable prints and save space
#include <CayenneMQTTEthernet.h>

// Cayenne authentication info. This should be obtained from the Cayenne Dashboard.
char username[] = "554cdf50-f0a9-11e7-abe9-1721c4c13600";
char password[] = "598bec3b273b7116ce1f6db5387a53dd3b688f8";
char clientId[] = "5691fe60-6d6e-11ea-a38a-d57172a4b4d4";

void setup() {
  Serial.begin(9600);
  Cayenne.begin(username, password, clientId);
}

void loop() {
  Cayenne.loop();
}

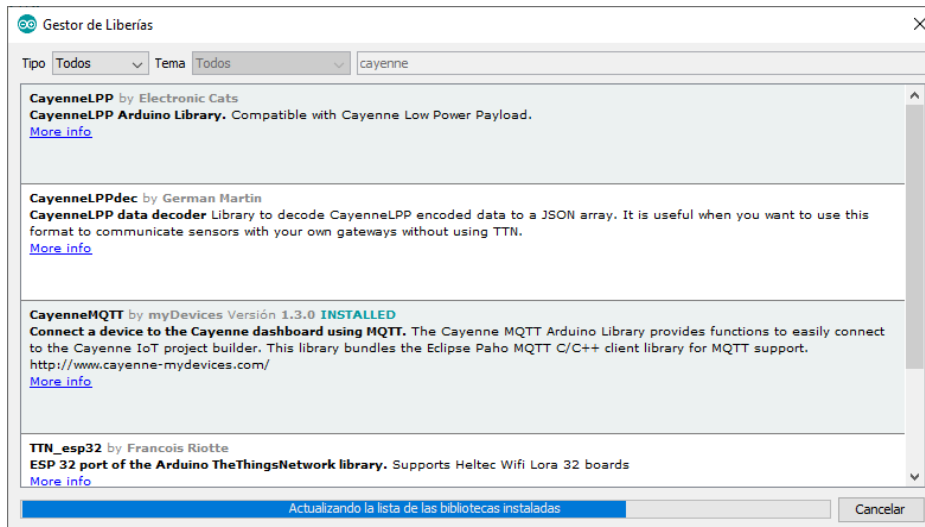
// Default function for sending sensor data at intervals to Cayenne.
// You can also use functions for specific channels, e.g. CAYENNE_OUT(1) for sending channel 1 data.
CAYENNE_OUT_DEFAULT() {
  // Write data to Cayenne here. This example just sends the current uptime in milliseconds on virtual channel 0.
  Cayenne.virtualWrite(0, millis());
  // Some examples of other functions you can use to send data.
  // Cayenne.celsiusWrite(1, 22.0);
  // Cayenne.luxWrite(2, 700);
  // Cayenne.virtualWrite(3, 50, TYPE_PROXIMITY, UNIT_CENTIMETERS);
}

// Default function for processing actuator commands from the Cayenne Dashboard.
// You can also use functions for specific channels, e.g. CAYENNE_IN(1) for channel 1 commands.
CAYENNE_IN_DEFAULT() {
  CAYENNE_LOG("channel %u, value %s", request.channel, getValue.asString());
  // Process message here. If there is an error set an error message using getValue.setError(), e.g. getValue.setError("error message");
}
```

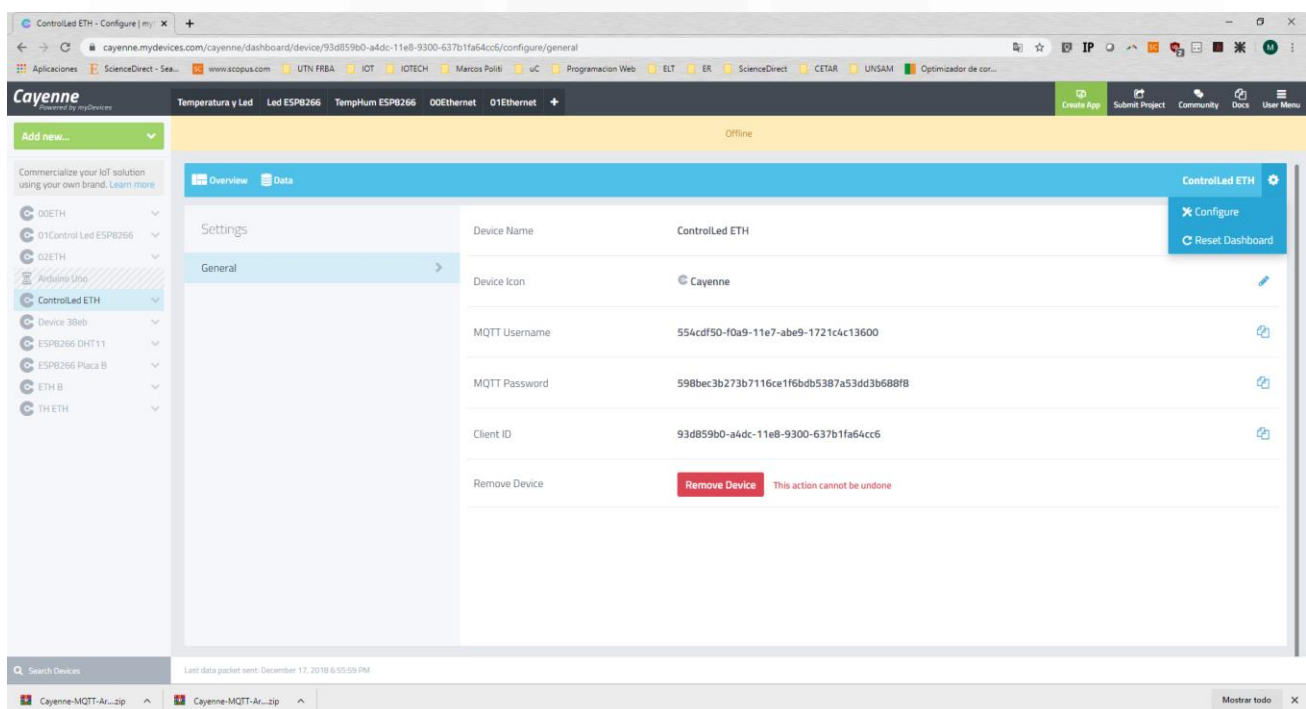
En IDE de Arduino debemos ir a Programas>Incluir librerías>Administrar bibliotecas

Ahí buscamos cayenne tal como se indica.

Contacto: [consultas@elearning-total.com](mailto:consultas@elearning-total.com)  
Web: [www.elearning-total.com](http://www.elearning-total.com)



Copiamos y pegamos los Ids



Cargando este programa

```
#define CAYENNE_PRINT Serial // Comment this out to disable prints and save space
#include <CayenneMQTTEthernet.h>
#define led 13
```

// Cayenne authentication info. This should be obtained from the Cayenne Dashboard.

Contacto: [consultas@elearning-total.com](mailto:consultas@elearning-total.com)  
Web: [www.elearning-total.com](http://www.elearning-total.com)



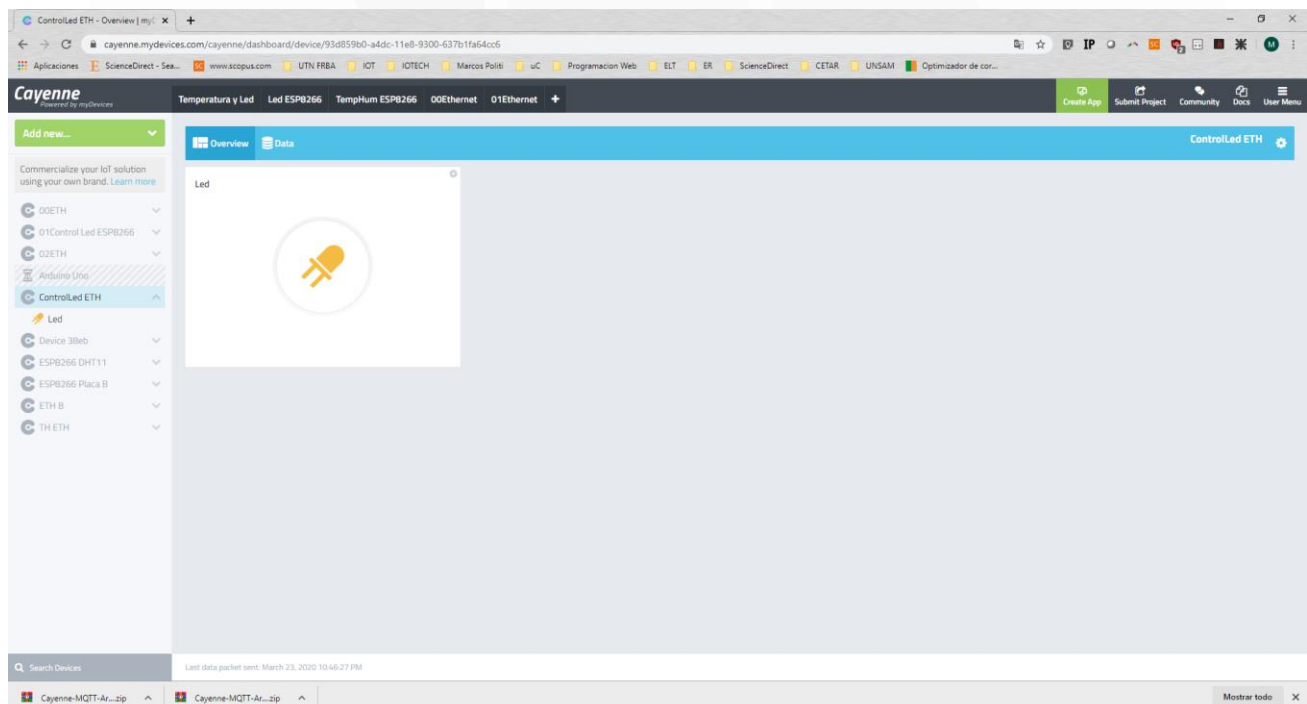
```
char username[] = "554cdf50-f0a9-11e7-abe9-1721c4c13600";
char password[] = "598bec3b273b7116ce1f6bdb5387a53dd3b688f8";
char clientID[] = "93d859b0-a4dc-11e8-9300-637b1fa64cc6";

void setup() {
  Serial.begin(9600);
  pinMode(led,OUTPUT);
  digitalWrite(led,LOW);
  Cayenne.begin(username, password, clientID);
}

void loop() {
  Cayenne.loop();
}

CAYENNE_IN(0)
{
  int entrada=getValue.asInt();
  Serial.println(entrada);
  digitalWrite(led,entrada);
}
```

Y procediendo a prender y apagar



Se observa el “ping” del dispositivo con la plataforma.





```
COM15
[0] MAC: FE-39-E0-17-D7-CC
[1195] IP: 192.168.1.103
[1196] Connecting to mqtt.mydevices.com:1883
[1713] Connected
0
1
```

☒ Autoscroll ☐ Mostrar marca temporal Nueva línea 9600 baudio Limpiar salida



## IMPLEMENTACION ETHERNET CON uC ATMEGA328+ SHIELD ETH UBIDOTS STEM

Para realizar esta implementación vamos a usar el Arduino Uno+Shield Eth y el servicio Ubidots Stem

Implementamos el código

```
/*
 * Lee por serial, y enviar el dato por puntero, debo tener cuidado con el
 * TAMANIOBUFFER que es el maximo que puedo recibir por Serial
 */
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>
#include <stdio.h>

#define pinDigital 8

#define TOKEN "BBFF-ulGcKpzEdA3HBOboNLg7KDEdj3xFoT"
#define DEVICE_LABEL "monitoreo"
#define VARIABLE_LABEL "temperatura"
#define MQTT_CLIENT_NAME "B" // MQTT client Name, put a random ASCII

char mqttBroker[] = "industrial.api.ubidots.com";
char payload[100];
char topicPublish[100];
char topicToSubscribe[100];

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
//IPAddress ip(10,12,1,97);

EthernetClient clientUbi;
PubSubClient client(clientUbi);
char str_temp[6];

void setup() {
  Serial.begin(115200);
  pinMode(pinDigital, OUTPUT);
  digitalWrite(pinDigital, LOW);

  // start the Ethernet connection:
  while (Ethernet.begin(mac) == 0) {
    Serial.println("Failed to configure Ethernet using DHCP");
    // try to configure using IP address instead of DHCP:
    //Ethernet.begin(mac, ip);
    delay(1000);
  }
  Serial.println(Ethernet.localIP());

  // give the Ethernet shield a second to initialize:
  delay(1000);
}
```



```
Serial.println("connected");
client.setServer(mqttBroker, 1883);
client.setCallback(callback);

sprintf(topicToSubscribe, "%s", ""); // Cleans the content of the char
sprintf(topicToSubscribe, "%s%s%s/lv", "/v1.6/devices/", DEVICE_LABEL,
"/luminarias");
Serial.println("subscribing to:");
Serial.println(topicToSubscribe);
client.subscribe(topicToSubscribe);

}

#define SI 1
#define NO 0
#define TAMANIOBUFFER 2

void loop() {

  uint8_t entrada[4];
  uint8_t *ptr_entrada;
  int publicar, leeSerial;

  int i;

  if (!client.connected()) {
    reconnect();
    client.subscribe(topicToSubscribe);
  }

  if (Serial.available())
  {
    for(i=0;i<TAMANIOBUFFER;i++)
    {
      entrada[i]=(uint8_t)Serial.read();
    }
    ptr_entrada=&entrada[0];
    publicar =SI;
    leeSerial=NO;
  }

  if(publicar==SI)
  {
    sprintf(topicPublish, "%s", "");
    sprintf(topicPublish, "%s%s%s%s", "/v1.6/devices/", DEVICE_LABEL,
"/", VARIABLE_LABEL);
    //Serial.println(topicPublish);

    client.publish(topicPublish, ptr_entrada);

    client.loop();
    delay(1000);
    leeSerial=SI;
    publicar=NO;
  }
}
```



```
}

}

/*****
 * Auxiliar Functions
 *****/

void callback(char* topic, byte* payload, unsigned int length)
{
    char PAYLOAD[5] = "    ";

    Serial.print("Mensaje Recibido: [");
    Serial.print(topic);
    Serial.print("] ");
    for (int i = 0; i < length; i++) {
        PAYLOAD[i] = (char)payload[i];
    }
    Serial.println(PAYLOAD);

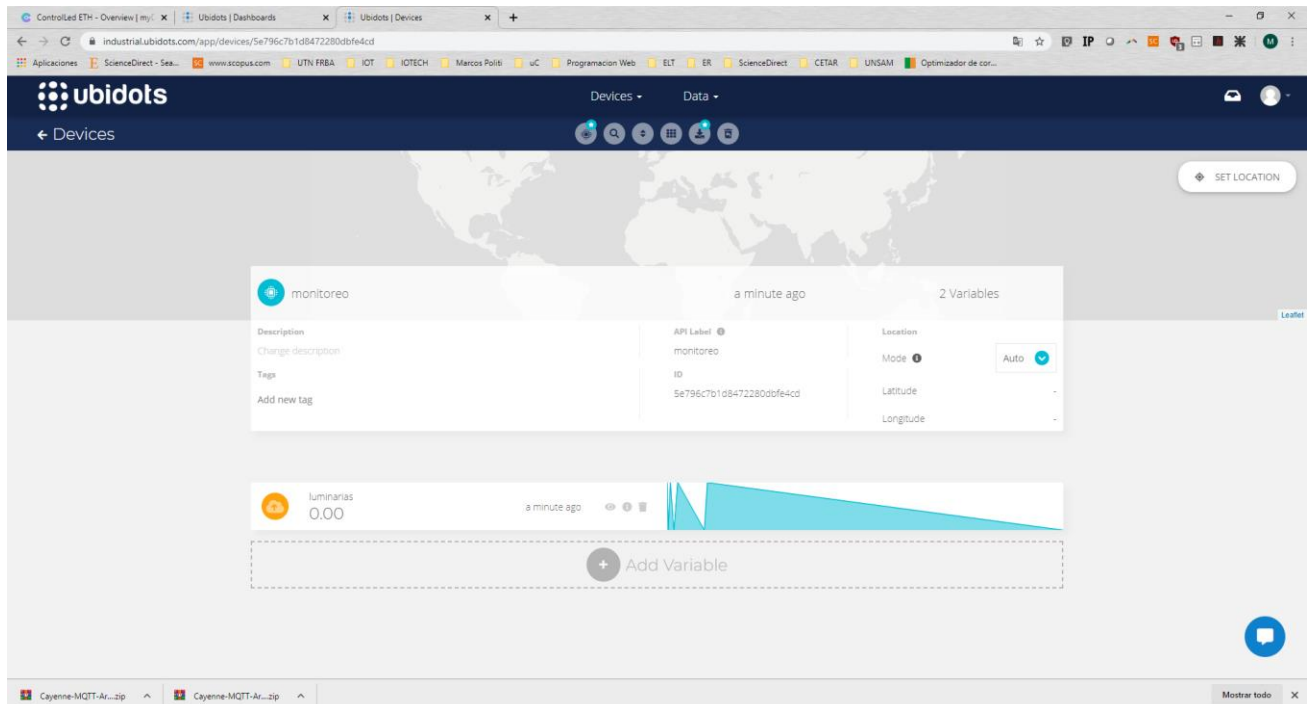
    if (payload[0] == '1'){
        digitalWrite(pinDigital, HIGH);
    }
    if (payload[0] == '0'){
        digitalWrite(pinDigital, LOW);
    }
}

void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        Serial.println("Attempting MQTT connection...");

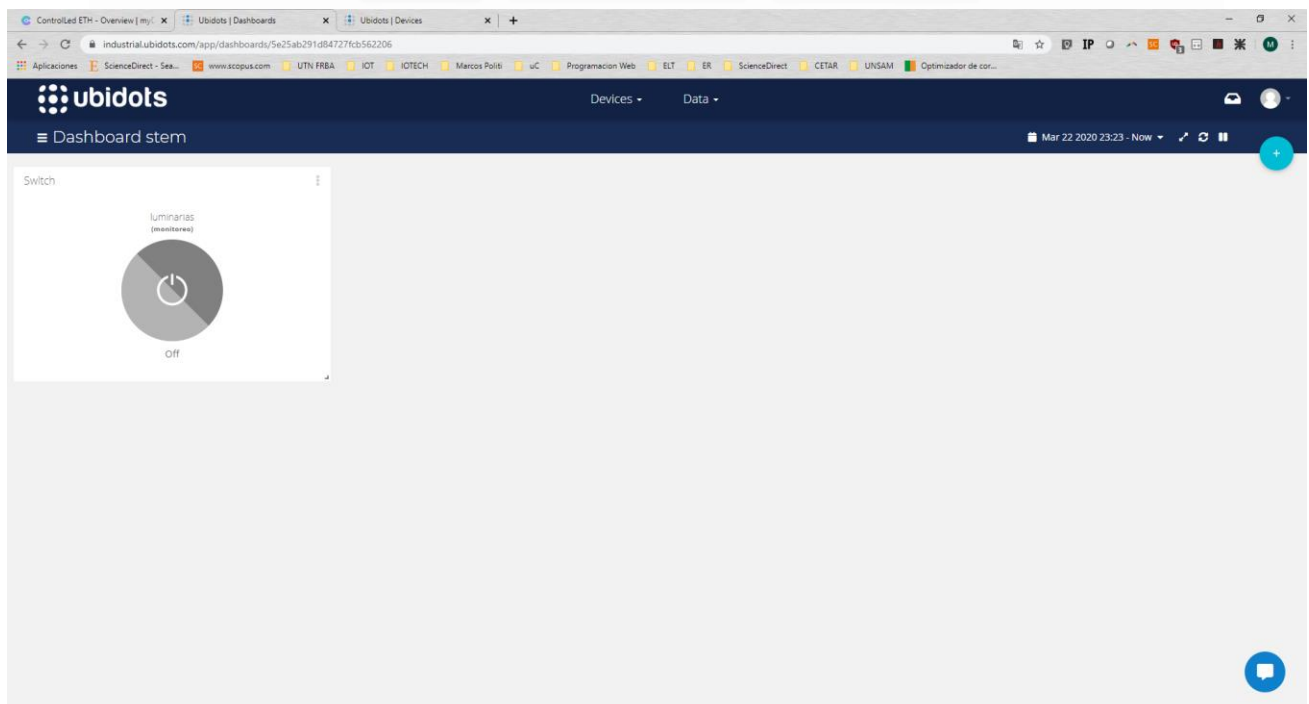
        // Attempt to connect
        if (client.connect(MQTT_CLIENT_NAME, TOKEN, "")) {
            Serial.println("connected");
        } else {
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 2 seconds");
            // Wait 2 seconds before retrying
            delay(2000);
        }
    }
}
}
```



Creamos el device, MONITOREO y una variable LUMINARIAS



Creamos el Dashboard



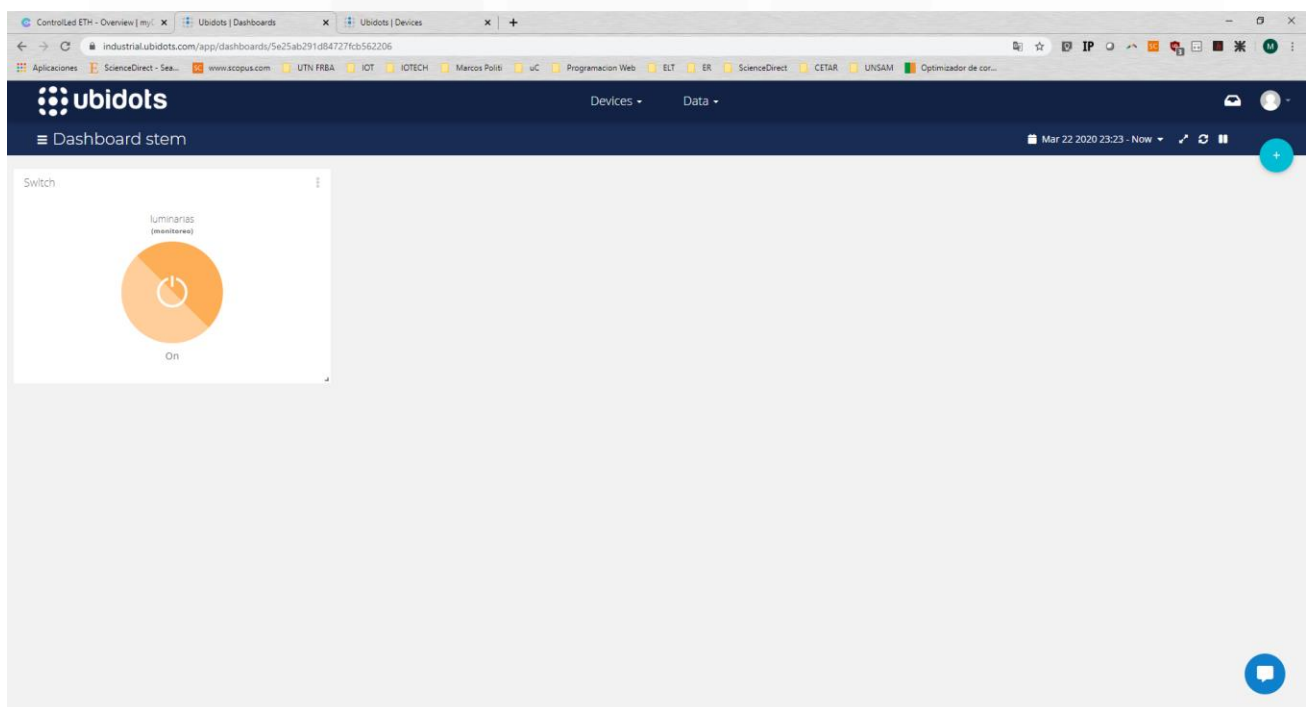
Al conectarnos:



```
COM17
192.168.1.101
connected
subscribing to:
/v1.6/devices/monitoreo/luminarias/lv
Attempting MQTT connection...
connected
Mensaje Recibido: [/v1.6/devices/monitoreo/luminarias/lv] 0
```

☒ Autoscroll ☐ Mostrar marca temporal Nueva línea 115200 baudio Limpiar salida

Luego interactuando con el dashboard





```
COM17
192.168.1.101
connected
subscribing to:
/v1.6/devices/monitoreo/luminarias/lv
Attempting MQTT connection...
connected
Mensaje Recibido: [/v1.6/devices/monitoreo/luminarias/lv] 0
```

☒ Autoscroll ☐ Mostrar marca temporal Nueva línea 115200 baudio Limpiar salida



## IMPLEMENTACION ETHERNET CON uC ATMEGA328+ SHIELD ETH CLOUDMQTT

Para realizar esta implementación vamos a usar el Arduino Uno+Shield Eth y el servicio cloudmqtt

Implementamos el siguiente código

```
#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>

#define pinDigital 8

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

EthernetClient ethClient;
PubSubClient client(ethClient);

char mqttBroker[] = "api.cloudmqtt.com";
char payload[200];
char topicPublish[150];
char topicToSubscribe[200];
char str_temp[6];

void callback(char* topic, byte* payload, unsigned int length)
{
    char PAYLOAD[5] = "    ";

    Serial.print("Mensaje Recibido: [");
    Serial.print(topic);
    Serial.print("] ");
    for (int i = 0; i < length; i++) {
        PAYLOAD[i] = (char)payload[i];
    }
    Serial.println(PAYLOAD);

    if (payload[0] == '1'){
        digitalWrite(pinDigital, HIGH);
    }
    if (payload[0] == '0'){
        digitalWrite(pinDigital, LOW);
    }
}

void reconnect() {
    // Loop until we're reconnected
    while (!client.connected()) {
        Serial.println("Attempting MQTT connection...");

        // Attempt to connect
```





```
if (client.connect ("A", "ETH001", "12345")) {
    Serial.println("connected");
} else {
    Serial.print("failed, rc=");
    Serial.print(client.state());
    Serial.println(" try again in 2 seconds");
    // Wait 2 seconds before retrying
    delay(2000);
}
}
}

void setup() {

    Serial.begin(115200);
    pinMode(pinDigital,OUTPUT);
    digitalWrite(pinDigital,LOW);

    while (Ethernet.begin(mac) == 0) {
        Serial.println("Failed to configure Ethernet using DHCP");
        // try to congifure using IP address instead of DHCP:
        delay(5000);
    }

    client.setServer("m12.cloudmqtt.com",11321);    // client is now ready for
use
    int estado=client.connect ("A", "ETH001", "12345");
    Serial.println(estado);

    client.setCallback(callback);

    sprintf(topicToSubscribe, "%s", ""); // Cleans the content of the char
    sprintf(topicToSubscribe, "%s", "/ETH001/led");
    Serial.println("subscribing to:");
    Serial.println(topicToSubscribe);
    client.subscribe(topicToSubscribe);

}

void loop() {

    if (!client.connected()) {
        reconnect();
        client.subscribe(topicToSubscribe);
    }

    // Values to send
    float temperatura = 5*random(0, 9);

    // 4 is minimum width, 2 is precision; float value is copied onto str_temp
    dtostrf(temperatura, 4, 2, str_temp);
```



```
sprintf(topicPublish, "%s", ""); // Cleans the topicPublish content
sprintf(topicPublish, "%s", "/ETH001/temperatura");
Serial.println(topicPublish);

sprintf(payload, "%s", ""); // Borro payload
sprintf(payload, "%s%s", payload, str_temp);

Serial.println(payload);
client.publish(topicPublish, payload);
Serial.println(" ");
client.loop();
delay(1000);
}
```

Se observa el ping de temperatura

The screenshot shows the CloudMQTT Websocket UI. On the left is a sidebar menu with options: DETAILS, SETTINGS, CERTIFICATES, USERS & ACL, BRIDGES, AMAZON KINESIS STREAM, WEBSOCKET UI (selected), STATISTICS, CONNECTIONS, and LOG. The main area is titled 'Websocket' and contains a 'Send message' form with 'Topic' and 'Message' fields and a 'Send' button. Below it is a 'Clear session' section with a 'Client ID' field and a 'Clear' button. On the right, a 'Received messages' table displays real-time data. A status bar at the top right shows 'Connected!' with a green checkmark and a message 'Connecting to m12.cloudmqtt.com...'. The browser's address bar shows the URL 'api.cloudmqtt.com/console/9849069/websocket'.

Topic	Message
/ETH001/temperatura	0.00
/ETH001/temperatura	0.00
/ETH001/temperatura	5.00
/ETH001/temperatura	10.00
/ETH001/temperatura	35.00
/ETH001/temperatura	0.00
/ETH001/temperatura	10.00

Y el ping de LED



CloudMQTT Console CASA

api.cloudmqtt.com/console/9849069/websocket

CloudMQTT CURSOS Giovanni

DETAILS  
SETTINGS  
CERTIFICATES  
USERS & ACL  
BRIDGES  
AMAZON KINESIS STREAM  
WEBSOCKET UI  
STATISTICS  
CONNECTIONS  
LOG

### Websocket

Messages are displayed in real-time as they are received by the broker. It's not possible to view historical data.

#### Send message

Topic:

Message:

#### Clear session

Clear any data for a client id

Client ID:

#### Received messages

Topic	Message
/ETH001/temperatura	30.00
/ETH001/temperatura	40.00
/ETH001/temperatura	0.00
/ETH001/temperatura	10.00
/ETH001/temperatura	35.00
/ETH001/temperatura	35.00
/ETH001/led	1
/ETH001/temperatura	35.00
/ETH001/temperatura	5.00
/ETH001/temperatura	30.00
/ETH001/temperatura	5.00
/ETH001/temperatura	0.00
/ETH001/temperatura	10.00
/ETH001/temperatura	20.00
/ETH001/temperatura	30.00
/ETH001/temperatura	30.00

COM17

Enviar

```
/ETH001/temperatura
10.00

/ETH001/temperatura
35.00

/ETH001/temperatura
35.00

/ETH001/temperatura
35.00

/ETH001/temperatura
5.00

Mensaje Recibido: [/ETH001/led] 1
/ETH001/temperatura
```

☐ Autoscroll ☐ Mostrar marca temporal

Nueva línea 115200 baudio Limpiar salida

Mg. Ing. Marcos Politi



## REFERENCIAS

[1] Documentos Maestría en Internet de las Cosas Universidad de Salamanca.

[2] CISCO, Introducción a Redes Ing. Aníbal Coto Cortes