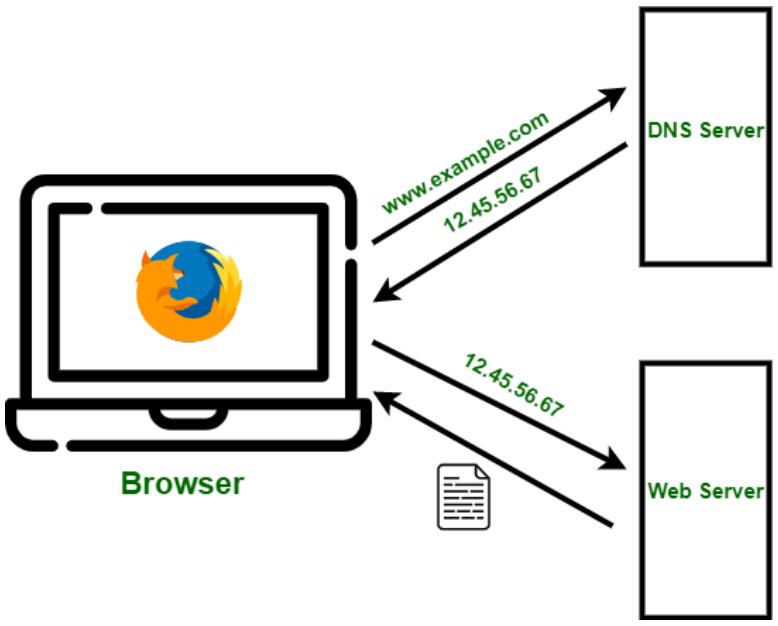
Protocolo DNS

El protocolo DNS (Domain Name System) es un sistema esencial en la infraestructura de internet que se encarga de traducir los nombres de dominio legibles por los humanos, en direcciones IP numéricas que son utilizadas por las computadoras para localizar y comunicarse con otros dispositivos en la red. El DNS actúa como una guía telefónica de internet, permitiendo que los usuarios accedan a sitios web y otros servicios sin necesidad de recordar complejas secuencias numéricas.

Una de las características principales del DNS es su estructura jerárquica y distribuida. Esta jerarquía comienza en la raíz, seguida de los dominios de nivel superior (como .com, .org, .net), y se extiende hacia los dominios específicos que incluyen subdominios. Esta organización permite una gestión eficiente y una resolución de nombres escalable en una red global. Además, el DNS es un sistema altamente tolerante a fallos, ya que está distribuido en múltiples servidores en todo el mundo, lo que asegura que, incluso si un servidor falla, otros puedan manejar las solicitudes.

El DNS también ofrece varias funcionalidades, como el almacenamiento en caché, que acelera la resolución de nombres al guardar temporalmente las respuestas de consultas anteriores. Asimismo, el protocolo permite la implementación de servicios como la distribución de cargas a través de balanceadores DNS y la asignación dinámica de direcciones IP mediante DNS dinámico (DDNS). El uso de DNS no se limita a la navegación web; también es fundamental en otros servicios de red, como el correo electrónico, donde se utiliza para localizar servidores de correo a través de registros MX (Mail Exchange).

En términos de aplicaciones, el DNS es indispensable para la operatividad diaria de internet y redes privadas. Es utilizado por navegadores web, aplicaciones móviles, y cualquier dispositivo que necesite conectarse a una red. También se usa en configuraciones avanzadas como redes de entrega de contenido (CDN), donde ayuda a dirigir a los usuarios al servidor más cercano, optimizando el rendimiento. Además, las organizaciones utilizan el DNS para gestionar el tráfico interno de sus redes y para implementar medidas de seguridad, como la filtración de contenido y la protección contra amenazas cibernéticas a través de servicios DNS seguros.



Desarrollo:

Para la presente práctica se va a realizar un servidor proxy DNS que con los ajustes correctos que se mencionarán más adelante se va a poder tener el rol como intermediario entre una consulta realizada por un cliente para poder bloquear las consultas que se encuentren en una lista negra o blacklist.

La blacklist va a tener la función de tener almacenadas las direcciones web que se desean restringir para los usuarios.

Si es el caso que se desea consultar una página que se encuentre en la lista negra se tendrá una respuesta de la forma No Exist Domain (No existe el dominio) ya que al estar en esa lista gracias al servidor lo interpreta como un dominio que no existe.

Para ello se requieren tres herramientas:

- Tener instalado MySQL Workbench (Opcional puedes utilizar cualquier otra base de datos)
- Tener instalado Node js instalado.
- Tener instalado un navegador de preferencia Firefox ya que por defecto tiene deshabilitada la función de DNS sobre HTTPS que te va a permitir comprobar el funcionamiento de nuestro servidor, también se puede desactivar en otros navegadores, pero se sugiere para omitir el paso de desactivarlo.

Inicialmente se va a crear un archivo de tipo JavaScript, se puede nombrar de forma libre, en este caso se va a nombrar como servidor_proxy.js.

La funcionalidad del código es un servidor DNS Proxy en Node.js que actúa como intermediario entre el cliente y un servidor DNS real. Utiliza las bibliotecas dgram para manejar consultas DNS sobre UDP y mysql2 para interactuar con una base de datos MySQL. Primero, se configura la conexión a MySQL especificando el host, el usuario, la contraseña y la base de datos. El servidor UDP se crea y escucha en el puerto N de la IP X.X.X.X Al recibir una consulta DNS, el servidor extrae el dominio de la consulta y verifica si está en una lista negra consultando la base de datos MySQL. Si el dominio está bloqueado, el servidor envía una respuesta falsa con el código NXDOMAIN, indicando que el dominio no existe. NXDOMAIN (Non-Existent Domain) es un código de error DNS que indica que el dominio no se encuentra. Si el dominio no está bloqueado, la consulta se reenvía a un servidor DNS real, en este caso, el servidor DNS de Google en la IP 8.8.8.8. El servidor DNS real procesa la consulta y devuelve la respuesta, que el proxy transmite de vuelta al cliente. De esta manera, el proxy filtra las consultas DNS según la lista negra y permite la resolución normal de dominios que no están bloqueados.

```
const dgram = require('dgram');
const mysql = require('mysql2');

// Configuración de MySQL
const db = mysql.createConnection({
    host: 'ip_de_tu_base_de_Datos',
    user: 'tu_usuario',
    password: 'tu_contraseña',
    database: 'nombre_de_tu_base_de_Datos'
});

db.connect((err) => {
```

```
    if (err) {
        console.error('Error al conectar a MySQL:', err);
        return;
    }
    console.log('Conectado a MySQL');
});

// Crear un servidor UDP para escuchar las consultas DNS
const server = dgram.createSocket('udp4');

// Función para verificar si un dominio está en la lista negra
function isBlacklisted(domain, callback) {
    const query = 'SELECT * FROM blacklist WHERE domain = ?';
    console.log(`Consultando la lista negra para el dominio: ${domain}`);
    db.query(query, [domain], (err, results) => {
        if (err) {
            console.error('Error al consultar la base de datos:', err);
            return callback(false);
        }
        console.log('Resultados de la consulta:', results);
        callback(results.length > 0);
    });
}

// Función para manejar las consultas DNS
server.on('message', (msg, rinfo) => {
    console.log('Consulta recibida:', msg);
    const domain = parseDomainFromQuery(msg);
    console.log('Dominio extraído:', domain);

    if (!domain) {
        console.log('Consulta DNS no válida');
        return;
    }

    isBlacklisted(domain, (blacklisted) => {
        if (blacklisted) {
            console.log(`Dominio bloqueado: ${domain}`);
            sendFakeResponse(msg, rinfo, server);
        } else {
            console.log(`Dominio no bloqueado: ${domain}`);
            forwardQueryToDNS(msg, rinfo, server);
        }
    });
});

// Función para analizar el dominio de la consulta DNS
function parseDomainFromQuery(msg) {
    const question = msg.slice(12);
    let domain = '';
    let i = 0;

    while (i < question.length) {
        const len = question[i];
        if (len === 0) break;
        domain += question.slice(i + 1, i + 1 + len).toString('utf8') + '.';
```

```
        i += len + 1;
    }

    return domain.slice(0, -1);
}

// Función para enviar una respuesta falsa (IP 0.0.0.0)
function sendFakeResponse(msg, rinfo, server) {
    const response = Buffer.alloc(msg.length);
    msg.copy(response);
    response[2] = 0x81; // Respuesta con error (NXDOMAIN)
    response[3] = 0x83;
    server.send(response, 0, response.length, rinfo.port, rinfo.address, (err) =>
{
        if (err) console.error('Error al enviar la respuesta falsa:', err);
    });
}

// Función para reenviar la consulta DNS al resolver real
function forwardQueryToDNS(msg, rinfo, server) {
    const client = dgram.createSocket('udp4');
    client.on('message', (response) => {
        server.send(response, 0, response.length, rinfo.port, rinfo.address,
(err) => {
            if (err) console.error('Error al enviar la respuesta del DNS real:',
err);
        });
        client.close();
    });

    client.send(msg, 0, msg.length, 53, '8.8.8.8', (err) => {
        if (err) console.error('Error al enviar la consulta al DNS real:', err);
    });
}

// Iniciar el servidor DNS Proxy
server.bind(5354, 'ip_del_servidor', () => {
    console.log('Servidor DNS Proxy escuchando en el puerto 3306');
});
```

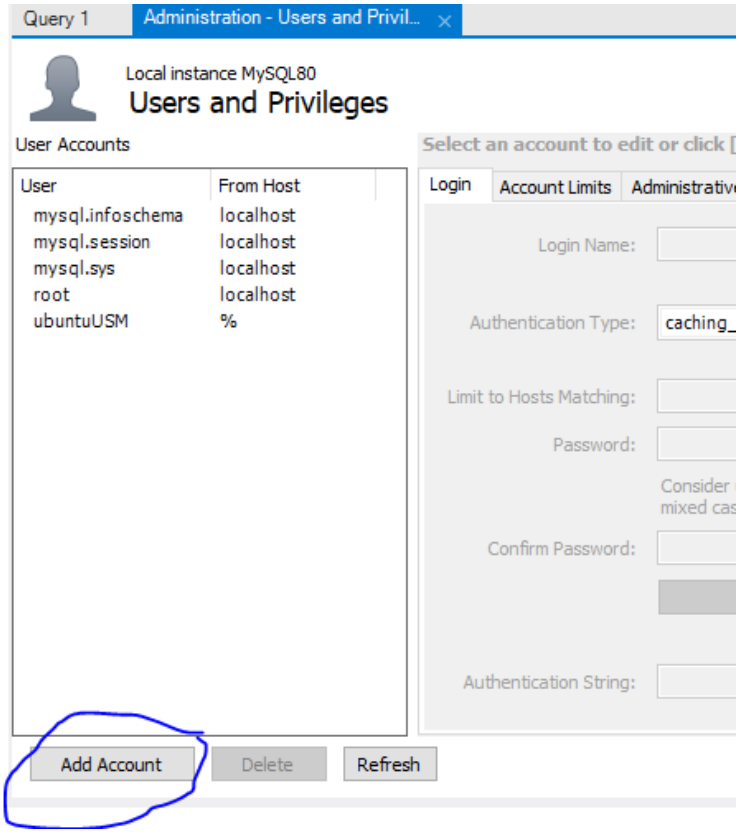
Una vez que se tenga la lógica se debe de crear la base de datos, en este caso utilice MySQL Workbench pero se puede utilizar Postgre, Oracle, etc.

```
CREATE DATABASE dns_proxy;
USE dns_proxy;
CREATE TABLE blacklist (
    id INT AUTO_INCREMENT PRIMARY KEY,
    domain VARCHAR(255) NOT NULL
);
```

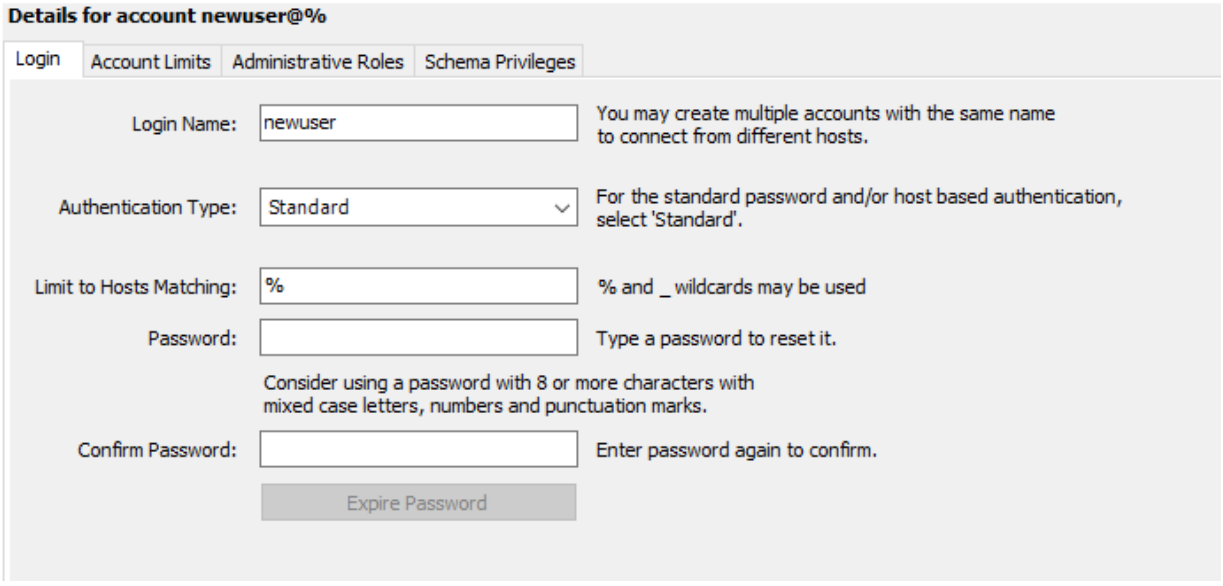
Se puede agregar un dominio para fines prácticos en este caso se va a intentar que se bloquee el sitio escom.ipn.mx (la página de la escuela).

```
INSERT INTO blacklist (domain) VALUES ('escom.ipn.mx');
```

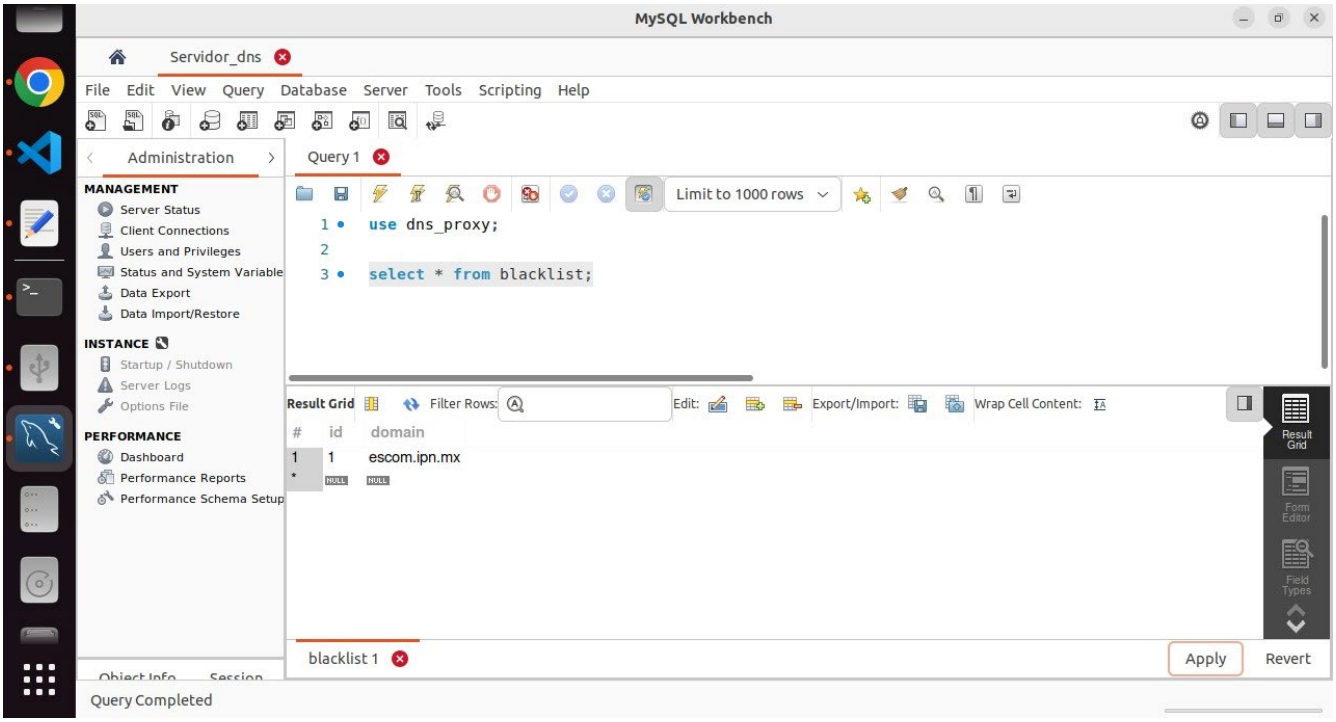
Se debe de configurar para que se puede acceder de forma remota desde cualquier IP concediendo privilegios.



Se presiona Add Acount y se llenan los datos del formulario:

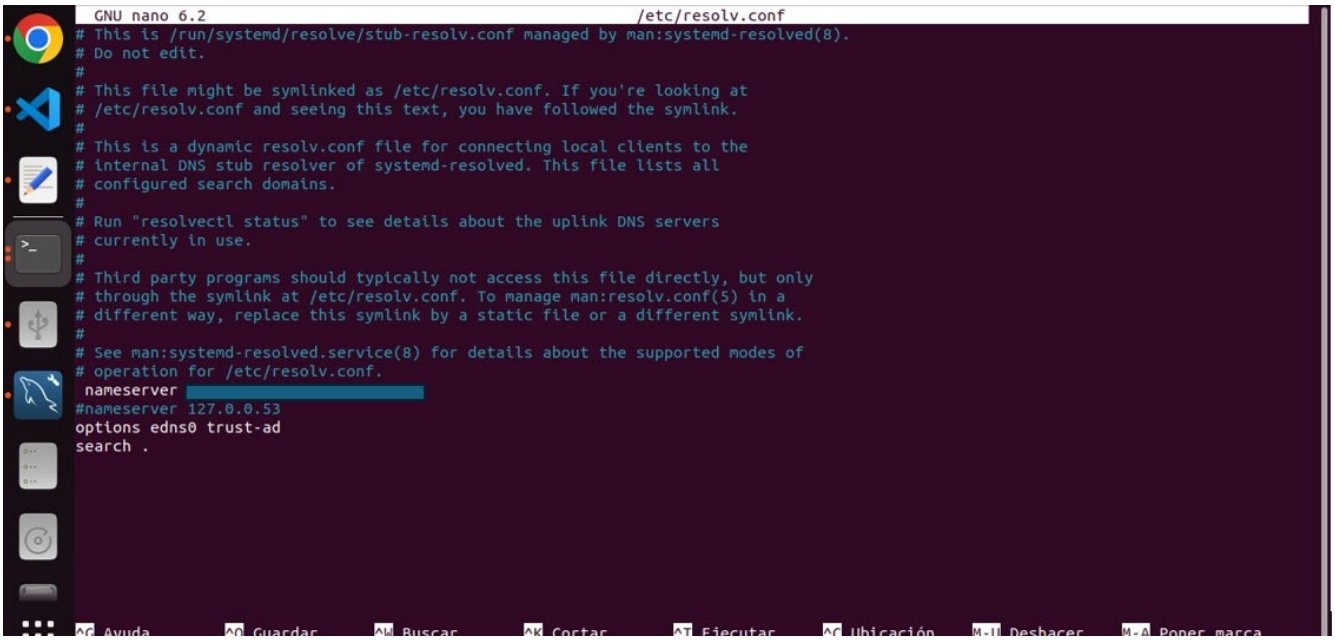


Una vez llenando el formulario se puede hacer la prueba en otro equipo, en este caso hare la prueba desde Ubuntu donde voy a ejecutar el servidor proxy DNS:

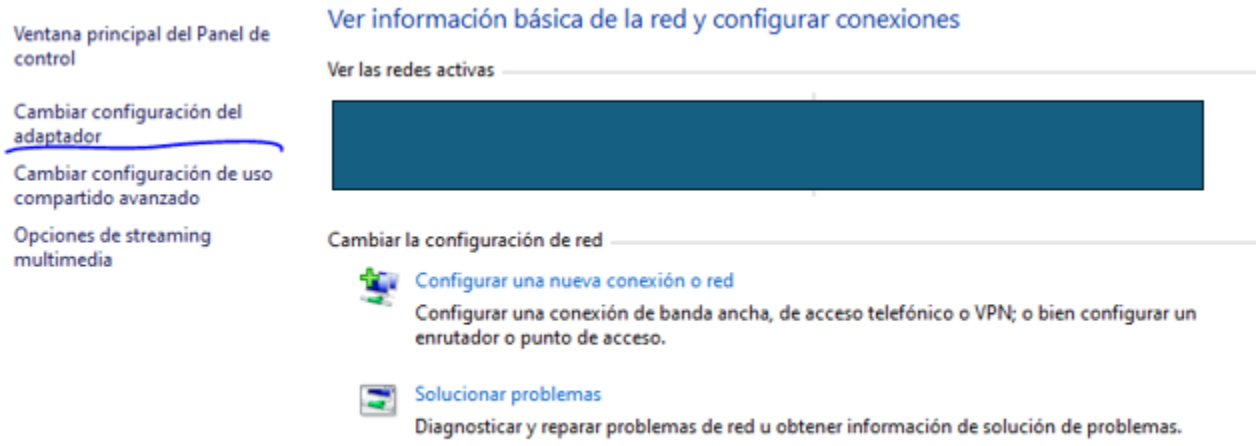


Ahora se debe de configurar algunos archivos en Ubuntu en este caso se debe de establecer la IP a la que va a apuntar el servidor DNS ya que se encuentra uno por defecto:

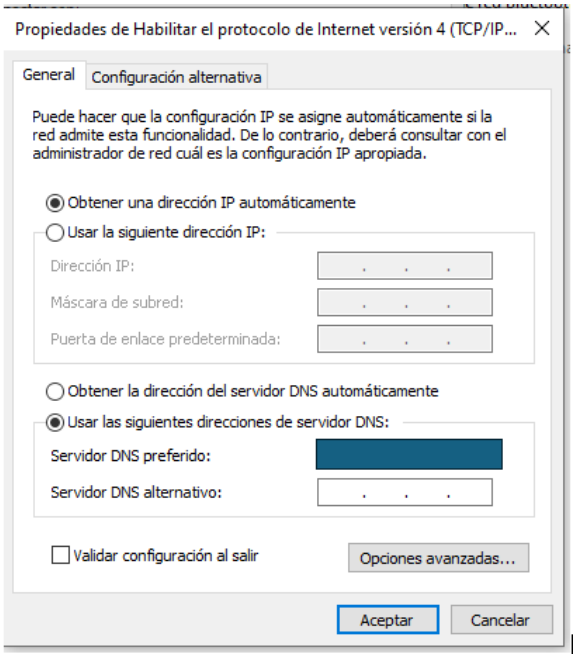
sudo nano /etc/resolv.conf



Después de eso se debe de configurar de lado de Windows el servidor DNS por preferencia y esto es desde el apartado de redes en el panel de control y se selecciona la opción de cambiar configuración del adaptador:

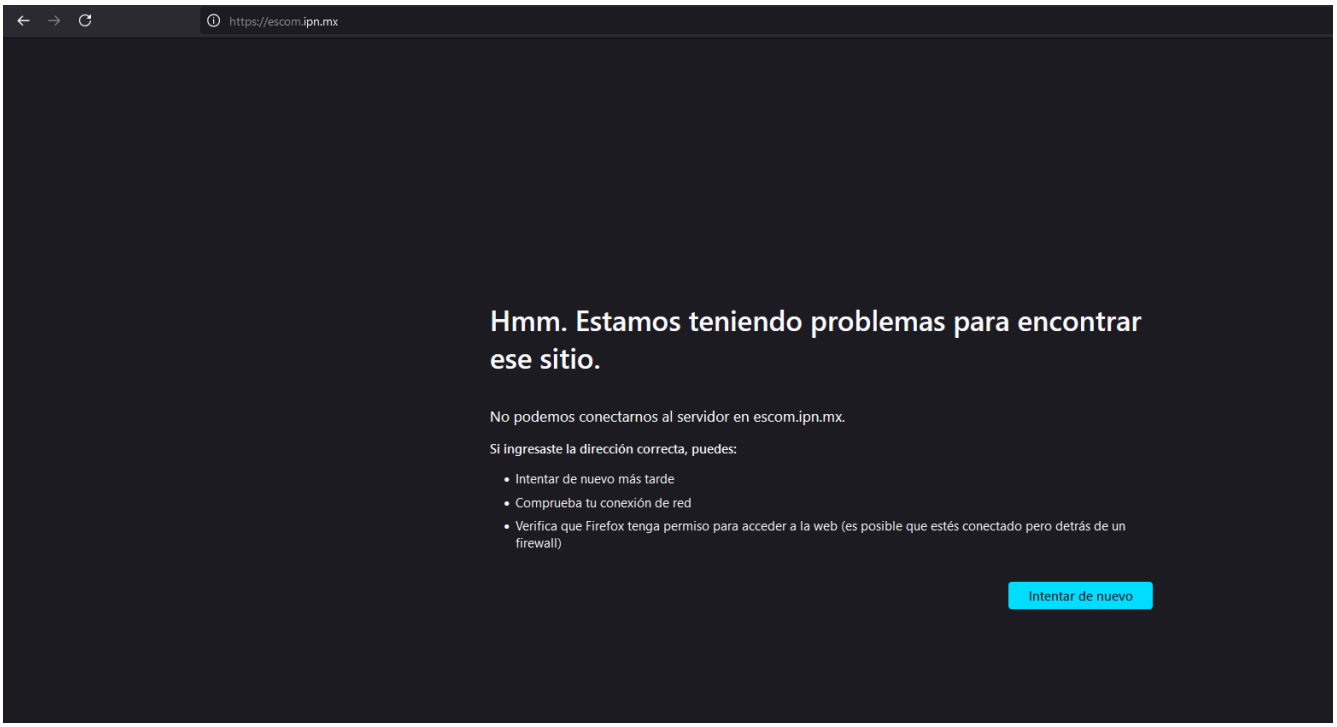


Se selecciona la opción de IPV4 y se presiona Propiedades:



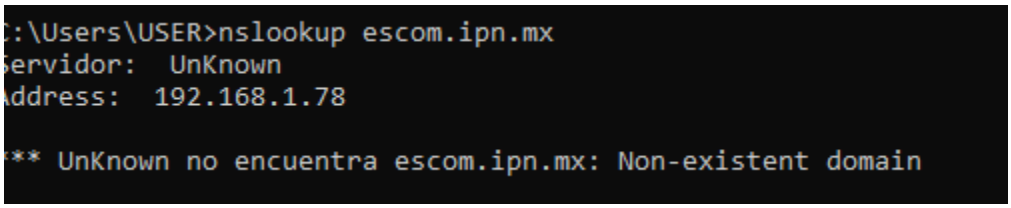
Se debe de poner la IP donde se esta ejecutando el servidor DNS, debe ser la misma que se puso como servername en Ubuntu.

Ahora se ejecuta el servidor en Ubuntu con node servidor_proxy.js



También si en CMD o Poweshell se puede comprobar con el siguiente comando:

```
nslookup escom.ipn.mx
```



La salida en Ubuntu que se ve es la siguiente:

De esta manera queda comprobado el funcionamiento del servidor proxy DNS.

En caso de error muchas veces se tiene que redirigir el tráfico que cae sobre el puerto 53 preferido de DNS al que se está utilizando esto se puede hacer con el siguiente comando:

```
sudo iptables -t nat -A PREROUTING -p udp –dport 53 -j REDIRECT –to-port 5354
sudo ufw allow 5354/udp
```