# TY B.Tech. (CSE) – II [ 2022-23 ]
# 5CS372 : Advanced Database System Lab
# ISE - 1

**PRN: 2020BTECS00033**
**Name: Prathamesh Raje**

## Problem:

1. **How to find the machine ID or IP address of users accessing the web applications developed in Assignment No 3 & 4? Provide detailed design/solution.**

   **Solution:**

   **Using Node.js**

   In Node.js, we can obtain the IP address of a client making a request to a web application by accessing the request object's socket property. The IP address is available in the remoteAddress property of the socket object.

   **Code:**

```
const http = require('http');

const server = http.createServer((req, res) => {

  const ip = req.socket.remoteAddress;

  console.log(`Client IP address: ${ip}`);

  res.writeHead(200, {'Content-Type': 'text/plain'});

  res.end('Hello World\n');

});

server.listen(3000, () => {

  console.log('Server listening on port 3000');

});
```

   Here, the server creates an HTTP server that listens on port 3000. When a client makes a request to the server, the server logs the IP address of the

client to the console using the req.socket.remoteAddress property. We can use this IP address for logging or other purposes as needed.

To find the machine ID of users accessing the web applications using Node.js, We can use the os module, which provides a number of operating system-related utility methods and properties. Specifically, we can use the os.hostname() method to retrieve the host name of the machine running the Node.js application.

**Code:**

```
const os = require('os');

const server = http.createServer((req, res) => {

  const hostname = os.hostname();

  console.log(`Machine ID: ${hostname}`);

  res.writeHead(200, {'Content-Type': 'text/plain'});

  res.end('Hello World\n');

});

server.listen(3000, () => {

  console.log('Server listening on port 3000');

});
```

Here, when a client makes a request to the server, the server logs the host name of the machine running the Node.js application to the console using os.hostname(). We can use this host name for logging or other purposes as needed. Note that this will only give us the host name of the server running the Node.js application, and not the host name of the client machine making the request.

**Using PHP**

Following are the steps to find IP address using PHP scripting:

● Use echo $_SERVER["REMOTE_ADDR"];

● If client is behind either proxy server or a router then use echo $_SERVER['HTTP_X_FORWARDED_FOR'];

● Here $_SERVER is an array that contains server variables created by the web server.

**Code:**

```
function get_client_ip() {
  $ipaddress = '';
  if (isset($_SERVER['HTTP_CLIENT_IP']))
     $ipaddress = $_SERVER['HTTP_CLIENT_IP'];
  else if(isset($_SERVER['HTTP_X_FORWARDED_FOR']))
     $ipaddress = $_SERVER['HTTP_X_FORWARDED_FOR'];
  else if(isset($_SERVER['HTTP_X_FORWARDED']))
     $ipaddress = $_SERVER['HTTP_X_FORWARDED'];
  else if(isset($_SERVER['HTTP_FORWARDED_FOR']))
     $ipaddress = $_SERVER['HTTP_FORWARDED_FOR'];
  else if(isset($_SERVER['HTTP_FORWARDED']))
     $ipaddress = $_SERVER['HTTP_FORWARDED'];
  else if(isset($_SERVER['REMOTE_ADDR']))
     $ipaddress = $_SERVER['REMOTE_ADDR'];
  else
     $ipaddress = 'UNKNOWN';
  return $ipaddress;
```

Another way is to use third-party applications to fetch the IP addresses. Below are the some tools which can be used to fetch IP address: **1. Ipify 2. Ipinfo**

**Code:**

```
<script>
    $.get("https://ipinfo.io", function(response) {
          alert(response.ip)
       }, "json")
</script>
```

```
<script>
    $.getJSON("https://api.ipify.org?format=json", function(data) {
        $("#p").html(data.ip)
    })
</script>
```

2. **How to restrict the machine ID or IP address or domain of users from accessing the web applications developed in Assignment No 3 & 4? Provide detailed design/solution.**

**Solution:**

**Using Node.js**

To restrict the machine ID or IP address or domain of users from accessing the web applications using Node.js, we can use middleware in our Node.js application that checks the IP address or host name of the incoming request against a list of allowed addresses or domains. If the address or domain is not allowed, we can return a 403 Forbidden response to deny access.

**Code:**

```
const express = require('express');
const app = express();
const allowedIPs = ['192.168.1.100', '192.168.1.101'];
app.use((req, res, next) => {
  const clientIP = req.ip;
  if (!allowedIPs.includes(clientIP)) {
    return res.status(403).send('Access Denied');
  }
  next();
});
```

```
app.get('/', (req, res) => {

  res.send('Hello World');

});

app.listen(3000, () => {

  console.log('Server listening on port 3000');

});
```

Here, we define an array of allowed IP addresses (allowedIPs) and use middleware to check if the IP address of the incoming request is in the allowed list. If it's not in the list, we return a 403 Forbidden response to deny access. If it's in the list, we call next() to allow the request to continue to the next middleware or the route handler.

We can modify this example to check the host name or domain of the incoming request instead of the IP address by using req.hostname or req.get('host') instead of req.ip.

**Using PHP:**

We can use the $_SERVER['REMOTE_ADDR'] variable to get the IP address of the user accessing the web application.

**Code:**

```
$allowed_ips = array('192.168.0.1', '10.0.0.1'); // list of allowed IP addresses

if (!in_array($_SERVER['REMOTE_ADDR'], $allowed_ips)) {

   die("Access denied"); // terminate the script and display an error message

}
```

In this example, the **$allowed_ips array** contains a list of IP addresses that are allowed to access the web application. If the user's IP address is not in the $allowed_ips array, the script will terminate and display an error message.

To restrict access based on machine ID or domain, we can use similar logic, but we will need to modify the code to get the machine ID or domain name of the user accessing the web application.

**Code:**

```
$allowed_machine_ids = array('my-machine-id', 'another-machine-id'); // list of allowed machine IDs

$machine_id = gethostname(); // get the machine ID of the server hosting the web application

if (!in_array($machine_id, $allowed_machine_ids)) {

    die("Access denied"); // terminate the script and display an error message

}
```

Here, the **$allowed_machine_ids** array contains a list of machine IDs that are allowed to access the web application. The gethostname() function is used to get the machine ID of the server hosting the web application. If the machine ID is not in the $allowed_machine_ids array, the script will terminate and display an error message.