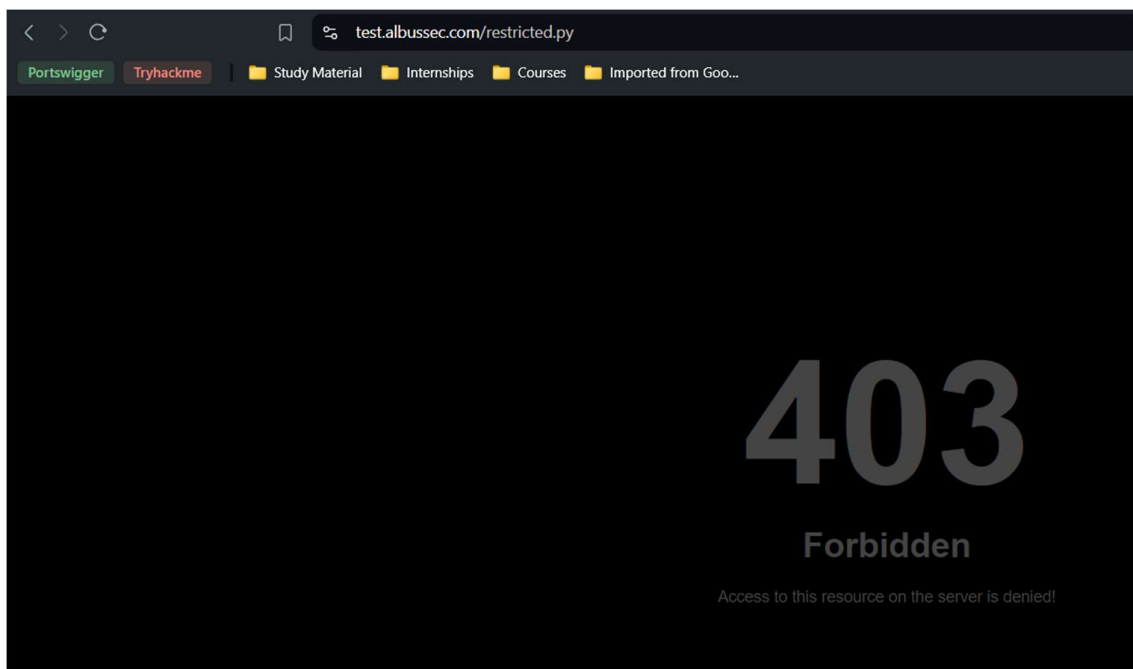


Albus Security Exam Challenge 6

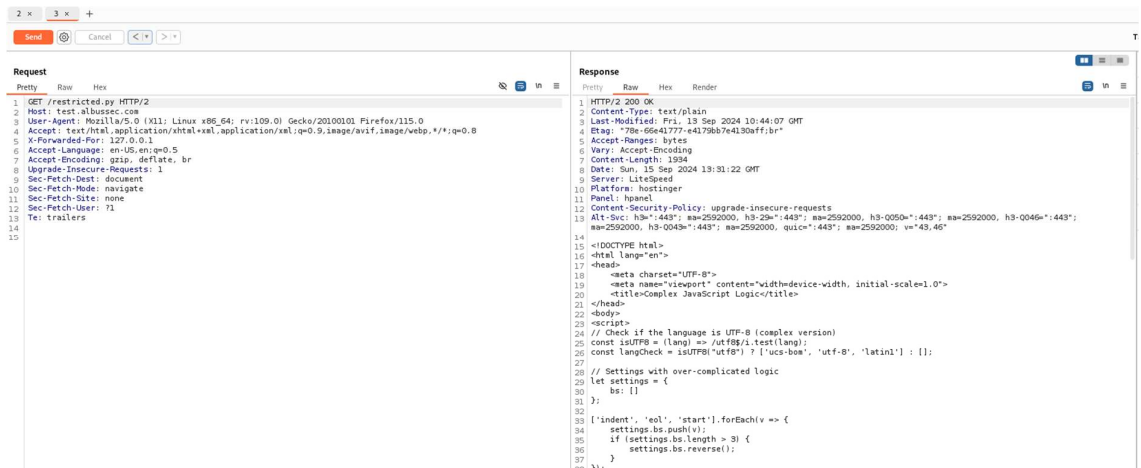
What steps are needed to bypass the 403 Forbidden error and access the **/restricted.py** file on the domain **test.albussec.com**

Steps:

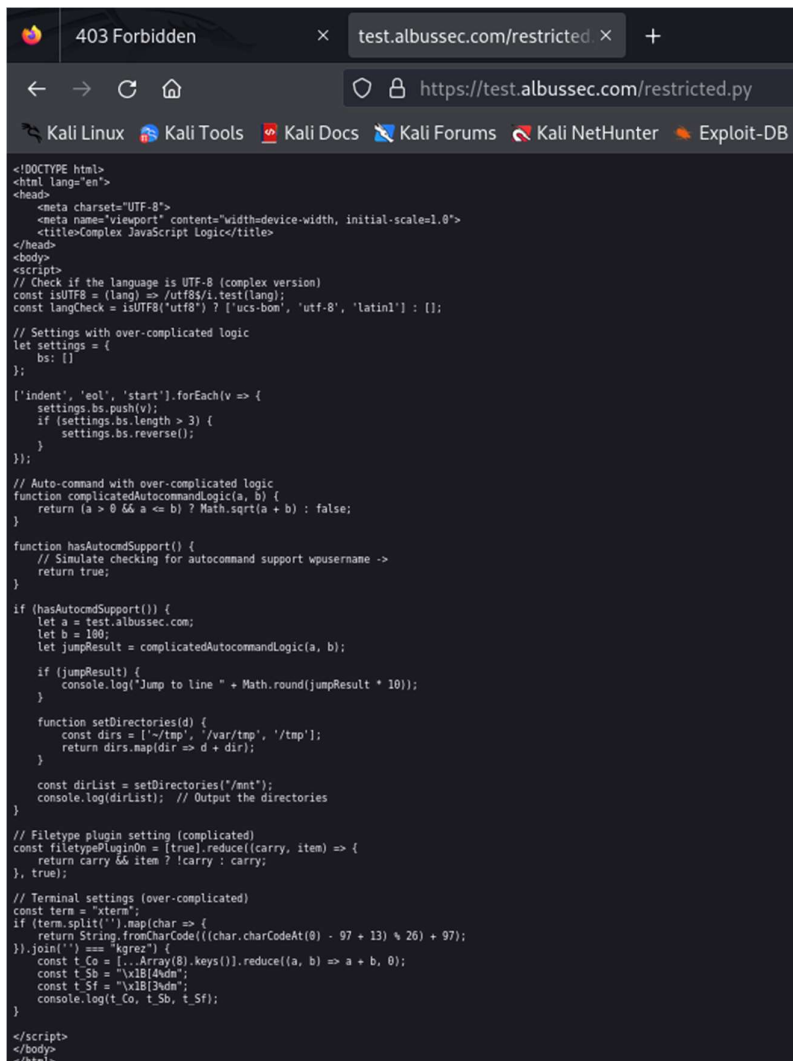
1. I used Burp Suite tool for this task.
2. I intercepted the **restricted.py** request from Browser to BurpSuite.
3. I analysed the request which gives me 403 Unauthorized Code.

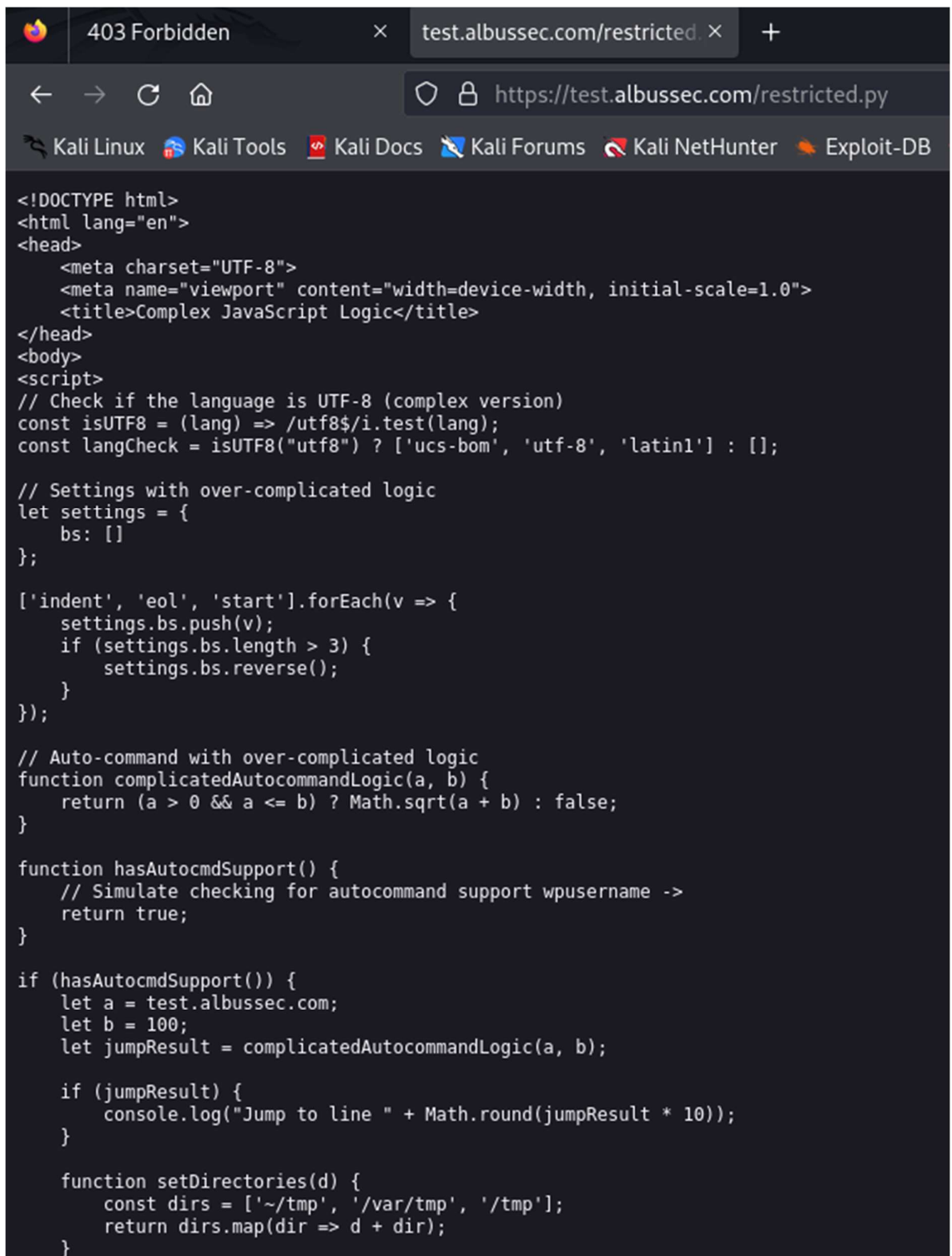


4. I tried many headers to bypass 403 Error but I didn't get the desired Result.
5. At Last, I tried a Header: - **X-Forwarded-For: 127.0.0.1** and I got a 200 OK Response.



Result:





The screenshot shows a web browser with two tabs. The first tab is titled "403 Forbidden" and the second tab is titled "test.albussec.com/restricted.py". The address bar shows the URL "https://test.albussec.com/restricted.py". Below the address bar, there are several navigation icons and a search bar. The search bar contains the text "Kali Linux", "Kali Tools", "Kali Docs", "Kali Forums", "Kali NetHunter", and "Exploit-DB". The main content area of the browser displays the source code of the file "restricted.py". The code is written in JavaScript and includes comments and function definitions. The code starts with a DOCTYPE declaration and a meta charset declaration. It then defines a function "isUTF8" and a variable "langCheck". It also defines a "settings" object and a "complicatedAutocommandLogic" function. The code ends with a "hasAutocmdSupport" function and a conditional block that calls "complicatedAutocommandLogic" and "setDirectories".

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Complex JavaScript Logic</title>
</head>
<body>
<script>
// Check if the language is UTF-8 (complex version)
const isUTF8 = (lang) => /utf8$/i.test(lang);
const langCheck = isUTF8("utf8") ? ['ucs-bom', 'utf-8', 'latin1'] : [];

// Settings with over-complicated logic
let settings = {
  bs: []
};

['indent', 'eol', 'start'].forEach(v => {
  settings.bs.push(v);
  if (settings.bs.length > 3) {
    settings.bs.reverse();
  }
});

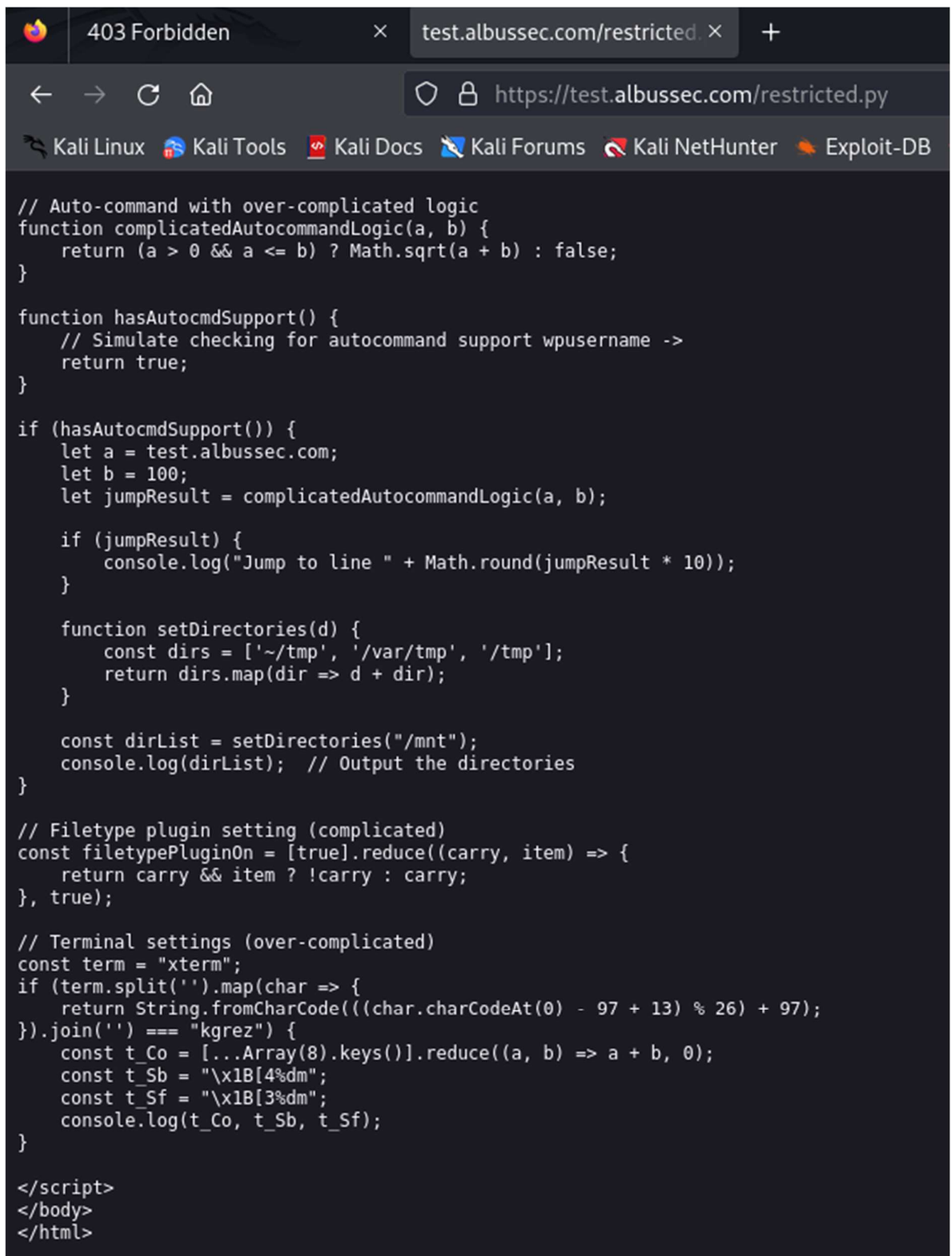
// Auto-command with over-complicated logic
function complicatedAutocommandLogic(a, b) {
  return (a > 0 && a <= b) ? Math.sqrt(a + b) : false;
}

function hasAutocmdSupport() {
  // Simulate checking for autocommand support w/username ->
  return true;
}

if (hasAutocmdSupport()) {
  let a = test.albussec.com;
  let b = 100;
  let jumpResult = complicatedAutocommandLogic(a, b);

  if (jumpResult) {
    console.log("Jump to line " + Math.round(jumpResult * 10));
  }

  function setDirectories(d) {
    const dirs = ['~/tmp', '/var/tmp', '/tmp'];
    return dirs.map(dir => d + dir);
  }
}
```



The image shows a web browser window with two tabs. The first tab is titled "403 Forbidden" and the second tab is titled "test.albussec.com/restricted.py". The address bar shows the URL "https://test.albussec.com/restricted.py". Below the address bar, there is a navigation bar with links to "Kali Linux", "Kali Tools", "Kali Docs", "Kali Forums", "Kali NetHunter", and "Exploit-DB". The main content area of the browser displays a JavaScript payload. The payload is a script that simulates checking for autocommand support, sets directories, and outputs the directory list. It also includes file type plugin settings and terminal settings. The script is enclosed in a <script> tag, and the body and html tags are closed at the bottom.

```
// Auto-command with over-complicated logic
function complicatedAutocommandLogic(a, b) {
    return (a > 0 && a <= b) ? Math.sqrt(a + b) : false;
}

function hasAutocmdSupport() {
    // Simulate checking for autocommand support wpusername ->
    return true;
}

if (hasAutocmdSupport()) {
    let a = test.albussec.com;
    let b = 100;
    let jumpResult = complicatedAutocommandLogic(a, b);

    if (jumpResult) {
        console.log("Jump to line " + Math.round(jumpResult * 10));
    }

    function setDirectories(d) {
        const dirs = ['~/tmp', '/var/tmp', '/tmp'];
        return dirs.map(dir => d + dir);
    }

    const dirList = setDirectories("/mnt");
    console.log(dirList); // Output the directories
}

// Filetype plugin setting (complicated)
const filetypePluginOn = [true].reduce((carry, item) => {
    return carry && item ? !carry : carry;
}, true);

// Terminal settings (over-complicated)
const term = "xterm";
if (term.split('').map(char => {
    return String.fromCharCode(((char.charCodeAt(0) - 97 + 13) % 26) + 97);
}).join('') === "kgrez") {
    const t_Co = [...Array(8).keys()].reduce((a, b) => a + b, 0);
    const t_Sb = "\x1B[4%dm";
    const t_Sf = "\x1B[3%dm";
    console.log(t_Co, t_Sb, t_Sf);
}

</script>
</body>
</html>
```