Client-side-again - picoGym Medium Web Exploitation

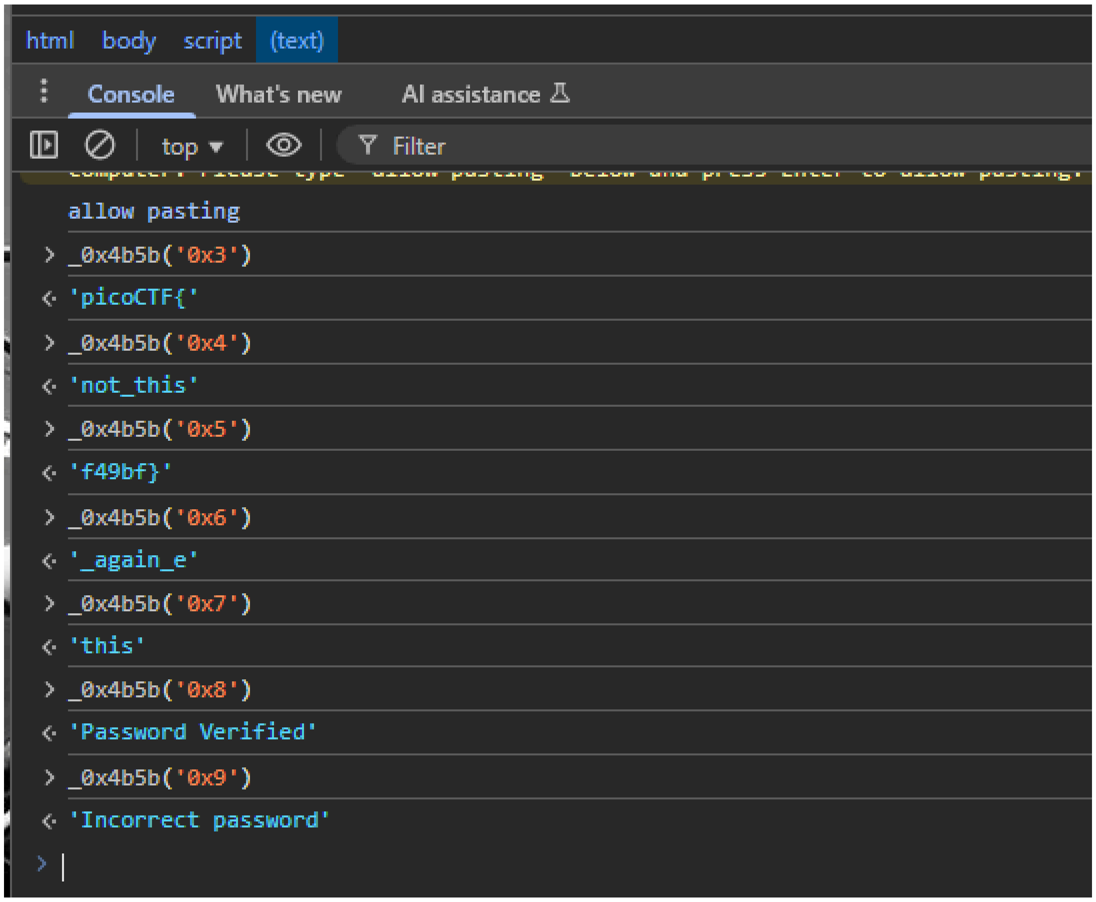
To start, the challenge presented me with a website to break into:
http://jupiter.challenges.picoctf.org:60786

After inspecting element on the website, I found a javascript block of code:

```
var _0x5a46 = ['f49bf}', '_again_e', 'this', 'Password\x20Verified', 'Incorrect\x20password',
'getElementById', 'value', 'substring', 'picoCTF{', 'not_this'];
(function(_0x4bd822, _0x2bd6f7) {
    var _0xb4bdb3 = function(_0x1d68f6) {
        while (--_0x1d68f6) {
            _0x4bd822['push'](_0x4bd822['shift']());
        }
    };
    _0xb4bdb3(++_0x2bd6f7);
}(_0x5a46, 0x1b3));
var _0x4b5b = function(_0x2d8f05, _0x4b81bb) {
    _0x2d8f05 = _0x2d8f05 - 0x0;
    var _0x4d74cb = _0x5a46[_0x2d8f05];
    return _0x4d74cb;
};

function verify() {
    checkpass = document[_0x4b5b('0x0')]('pass')[_0x4b5b('0x1')];
    split = 0x4;
    if (checkpass[_0x4b5b('0x2')](0x0, split * 0x2) == _0x4b5b('0x3')) {
        if (checkpass[_0x4b5b('0x2')](0x7, 0x9) == '{n') {
            if (checkpass[_0x4b5b('0x2')](split * 0x2, split * 0x2 * 0x2) == _0x4b5b('0x4')) {
                if (checkpass[_0x4b5b('0x2')](0x3, 0x6) == 'oCT') {
                    if (checkpass[_0x4b5b('0x2')](split * 0x3 * 0x2, split * 0x4 * 0x2) == _0x4b5b('0x5'))
{
                        if (checkpass['substring'](0x6, 0xb) == 'F{not') {
                            if (checkpass[_0x4b5b('0x2')](split * 0x2 * 0x2, split * 0x3 * 0x2) ==
_0x4b5b('0x6')) {
                                if (checkpass[_0x4b5b('0x2')](0xc, 0x10) == _0x4b5b('0x7')) {
                                    alert(_0x4b5b('0x8'));
                                }
                            }
                        }
                    }
                }
            }
        }
    } else {
        alert(_0x4b5b('0x9'));
    }
}
```

After analyzing the code, I found that it's an obfuscated password check and that I had to look up what the verify function was de-obfuscating:



```
allow pasting
> _0x4b5b('0x3')
<· 'picoCTF{'
> _0x4b5b('0x4')
<· 'not_this'
> _0x4b5b('0x5')
<· 'f49bf}'
> _0x4b5b('0x6')
<· '_again_e'
> _0x4b5b('0x7')
<· 'this'
> _0x4b5b('0x8')
<· 'Password Verified'
> _0x4b5b('0x9')
<· 'Incorrect password'
>
```

Looking at this output, I pieced the slices together that would makeup the flag and found this flag:

Flag: picoCTF{not_this_again_ef49bf}