Web 安全作业一 JSONP

GitHub 账号：hhhgll
学号：57118303　姓名：陈丽如

一.实验内容：

1. 实现三个主机

容器配置如下：

```
Host1：ip = 10.0.0.2, domin = web.cybersecurity.seu.edu
Host2：ip = 10.0.0.3, domin = time.cybersecurity.seu.edu
Host3：ip = 10.0.0.4, domin = jsonp.cybersecurity.seu.edu
```

向主机的 hosts 文件中添加以下条目

```
10.0.0.2 web.cybersecurity.seu.edu
10.0.0.3 time.cybersecurity.seu.edu
10.0.0.4 jsonp.cybersecurity.seu.edu
```

2. 在 time.cybersecurity.seu.edu 上实现三个接口如下

host2:server.js 如下

```
const express = require('express')
const { createReadStream } = require('fs')
const bodyParser = require('body-parser')
const app = express()
app.use(bodyParser.urlencoded({ extended: false }))
app.listen(80)

app.get('/', (req, res) => {
    createReadStream('index.html').pipe(res)
})

app.get('/api/date', (req, res) => {
    res.send({ date: Date.now() })
})

app.get('/api/datecors', (req, res) => {
    res.set('Access-Control-Allow-Origin', 'http://web.cybersecurity.
seu.edu')
    res.send({ datecors: Date.now() })
})
```

```
    app.get('/api/jsondate', (req, res) => {
        let callback = req.query.callback;
        let Str = `${callback}(${JSON.stringify({ datejson: Date.now() })
})`;
        res.send(Str);
    })
```

3. 在 web.cybersecurity.seu.edu 上实现一个页面，在页面中通过 js 代码读取 time.cybersecurity.seu.edu 的接口数据

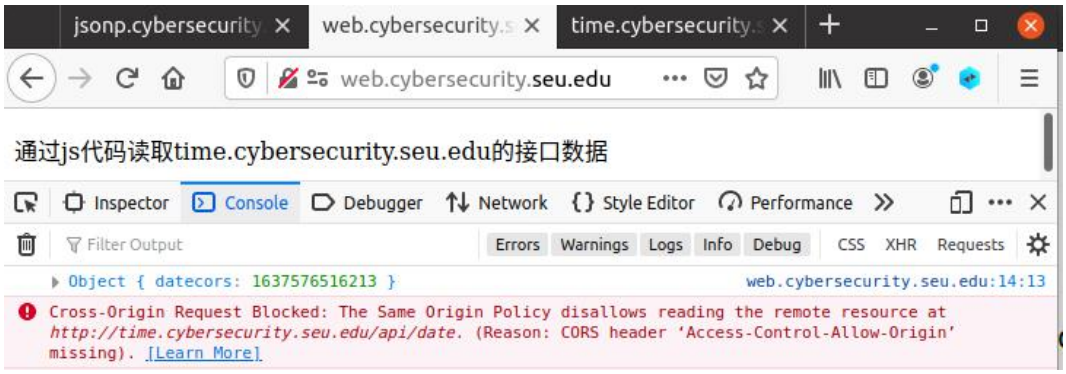
host1: index.html 如下

```
    <!DOCTYPE html>
    <html>
    <head>
    <meta charset="utf-8">
    <title>web.cybersecurity.seu.edu</title>
    </head>

    <body>
    <p>通过 js 代码读取 time.cybersecurity.seu.edu 的接口数据</p>
    <script type="text/javascript">
    async function get_corsdate() {
        const res = await fetch('http://time.cybersecurity.seu.edu/api/d
atecors')
        const data = await res.json()
        console.log(data)
        }
    get_corsdate();

    async function get_date() {
        const res = await fetch('http://time.cybersecurity.seu.edu/api/d
ate')
        const data = await res.json()
        console.log(data)
        }
    get_date();
    </script>
    </body>
    </html>
```

实验结果如图：



由于/api/datecors 接口设置了 CROS 头（如下图 1），所以 web.cybersecurity.seu.edu 读取到了/api/datecors 接口数据，并又控制台输出；

```
app.get('/api/datecors', (req, res) => {
    res.set('Access-Control-Allow-Origin', 'http://web.cybersecurity.seu.edu')
    res.send({ datecors: Date.now() })
})
```

图 1 api/datecors 设置 CROS 头部字段

而/api/date 接口未设置 CROS 头（如下图 2），web.cybersecurity.seu.edu 没有读取到/api/date 接口数据，控制台显示“跨域请求被阻止：同源策略不允许读取位于 http://time.cybersecurity.seu.edu/api/date 的远程资源。（原因：缺少 CORS 头‘Access-Control-Allow-Origin’）。”

```
app.get('/api/date', (req, res) => {
    res.send({ date: Date.now() })
})
```

图 2 api/date 未设置 CROS 头部字段

4. 在 jsonp.cybersecurity.seu.edu 下实现一个页面，在页面中通过回调 js 代码读取 time.cybersecurity.seu.edu 的接口数据

host3: index.htaml 如下：

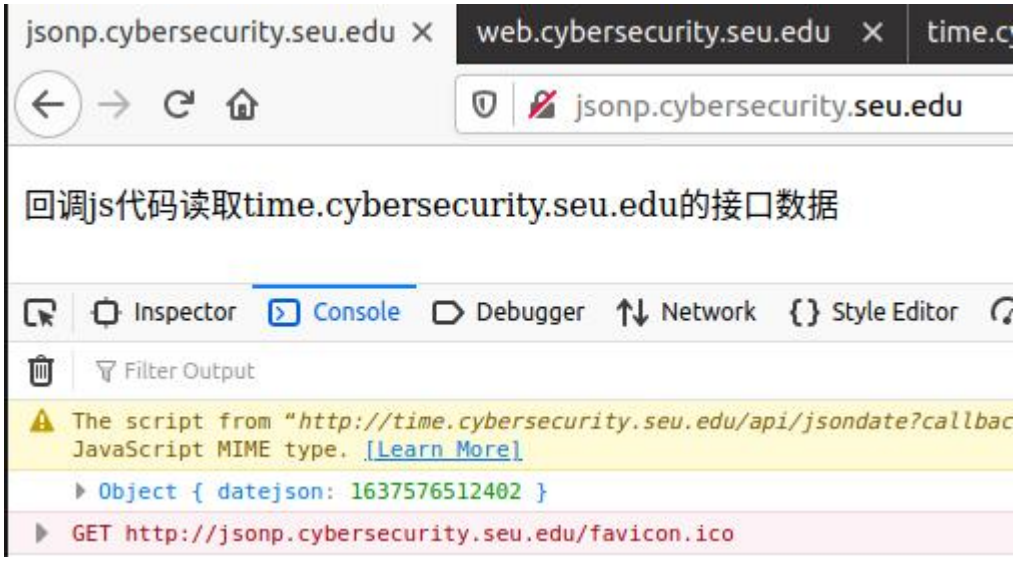
```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>jsonp.cybersecurity.seu.edu</title>
</head>

<body>
<p>回调 js 代码读取 time.cybersecurity.seu.edu 的接口数据</p>
<script>
```

```
    function handleTime(data) {
        console.log(data)
    }
    </script>

    <script src='http://time.cybersecurity.seu.edu/api/jsondate?callback
=handleTime'></script>
    </body>
    </html>
```

实验结果如图：



jsonp.cybersecurity.seu.edu 成功读取到 web.cybersecurity.seu.edu 的 /api/jsonpdate 接口数据，并由控制台输出。

二. 实验小结

CORS（Cross-origin resource sharing 跨域资源共享）是通过设置一个响应头来告诉浏览器，该请求允许跨域，浏览器收到该响应以后就会对响应放行。

JSONP（json with padding）利用了使用 src 引用静态资源时不受跨域限制的机制。主要在客户端搞一个回调做一些数据接收与操作的处理，并把这个回调函数名告知服务端，而服务端需要做的是按照 javascript 的语法把数据放到约定好的回调函数之中即可。