

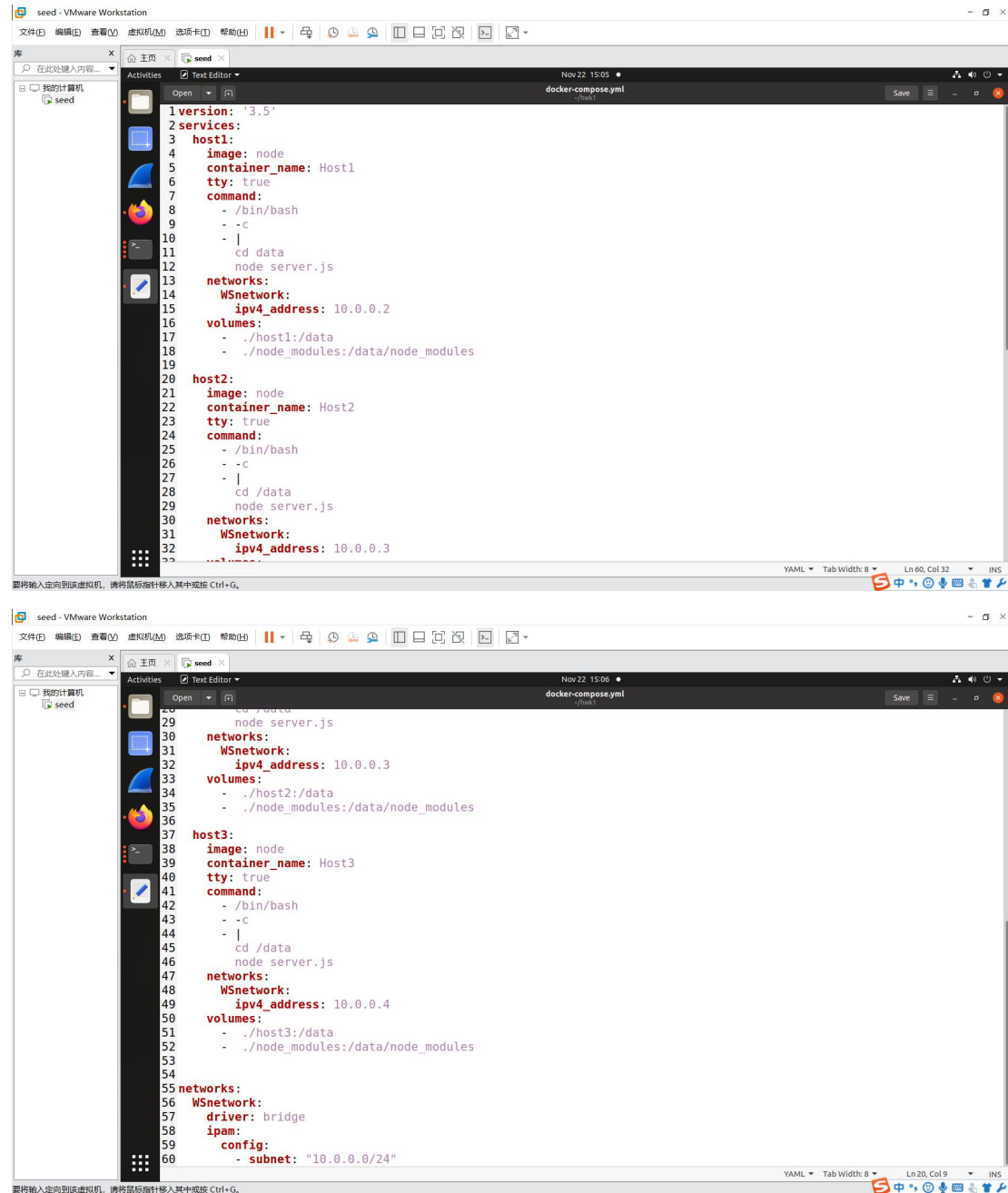
Web 安全实验报告 1

GitHub 账号: seu_syf

姓名: 宋雨帆

学号: 57118225

1、Docker 配置实现三个主机



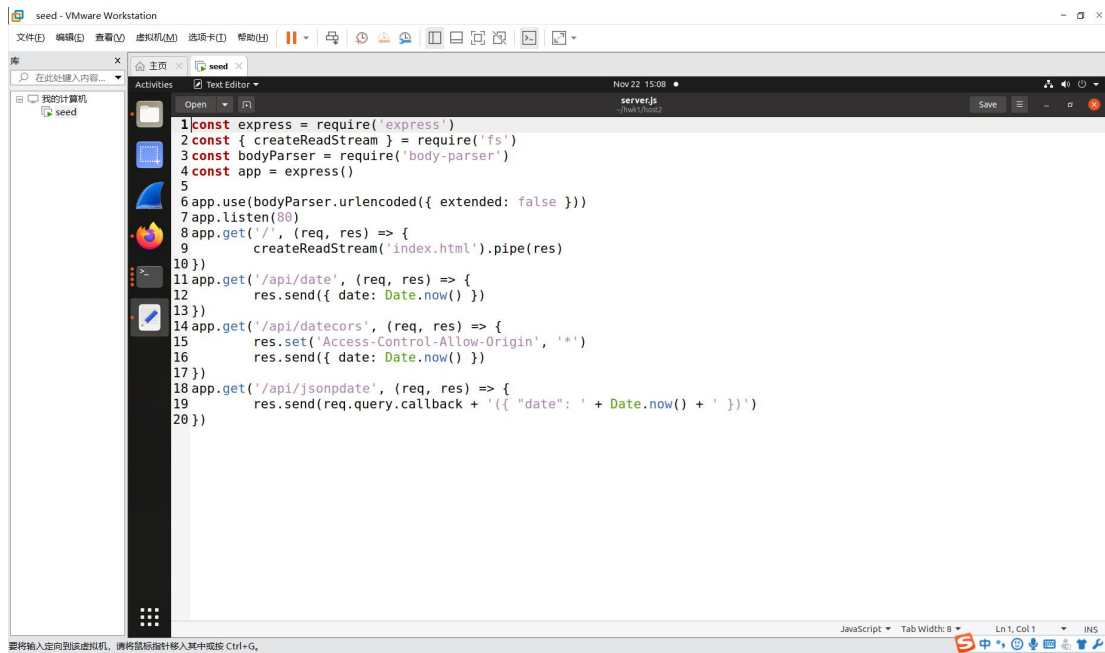
The first screenshot shows the configuration for Host1 and Host2 in a docker-compose.yml file. Host1 is configured with image: node, container_name: Host1, tty: true, and command: /bin/bash, -c, |, cd data, node server.js. It is connected to the WSNetwork and has an ipv4_address of 10.0.0.2. Host2 is configured with image: node, container_name: Host2, tty: true, and command: /bin/bash, -c, |, cd /data, node server.js. It is also connected to the WSNetwork and has an ipv4_address of 10.0.0.3. Both hosts share a volume named data, which is mapped to ./host1:/data and ./node_modules:/data/node_modules for Host1, and ./host2:/data and ./node_modules:/data/node_modules for Host2.

```
1 version: '3.5'
2 services:
3   host1:
4     image: node
5     container_name: Host1
6     tty: true
7     command:
8       - /bin/bash
9       - -c
10      - |
11        cd data
12        node server.js
13   networks:
14     WSNetwork:
15       ipv4_address: 10.0.0.2
16   volumes:
17     - ./host1:/data
18     - ./node_modules:/data/node_modules
19
20   host2:
21     image: node
22     container_name: Host2
23     tty: true
24     command:
25       - /bin/bash
26       - -c
27       - |
28         cd /data
29         node server.js
30   networks:
31     WSNetwork:
32       ipv4_address: 10.0.0.3
```

The second screenshot shows the configuration for Host3 and the network settings in the docker-compose.yml file. Host3 is configured with image: node, container_name: Host3, tty: true, and command: /bin/bash, -c, |, cd /data, node server.js. It is connected to the WSNetwork and has an ipv4_address of 10.0.0.4. It shares a volume named data, which is mapped to ./host3:/data and ./node_modules:/data/node_modules. The network settings are defined as WSNetwork with driver: bridge, ipam: config, and subnet: 10.0.0.0/24.

```
29   node server.js
30   networks:
31     WSNetwork:
32       ipv4_address: 10.0.0.3
33   volumes:
34     - ./host2:/data
35     - ./node_modules:/data/node_modules
36
37   host3:
38     image: node
39     container_name: Host3
40     tty: true
41     command:
42       - /bin/bash
43       - -c
44       - |
45         cd /data
46         node server.js
47   networks:
48     WSNetwork:
49       ipv4_address: 10.0.0.4
50   volumes:
51     - ./host3:/data
52     - ./node_modules:/data/node_modules
53
54 networks:
55   WSNetwork:
56     driver: bridge
57     ipam:
58       config:
59         - subnet: "10.0.0.0/24"
```

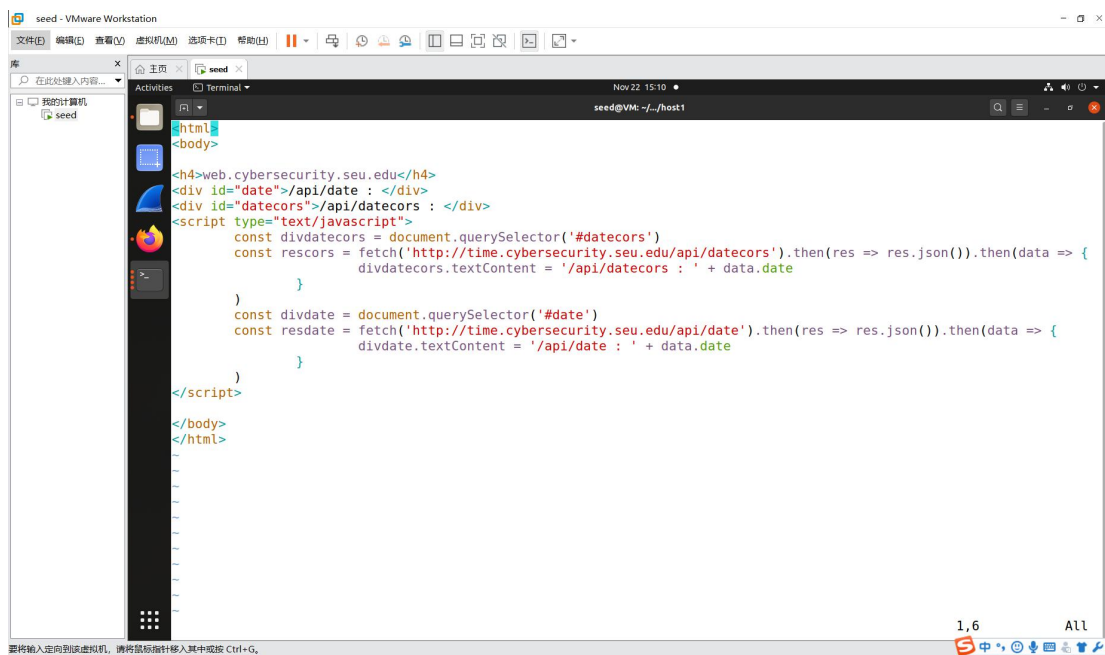
2、在 time.cybersecurity.seu.edu 主机上实现三个接口



The screenshot shows a VMware Workstation window with a text editor open. The text editor contains a JavaScript file named `server.js` with the following code:

```
1const express = require('express')
2const { createReadStream } = require('fs')
3const bodyParser = require('body-parser')
4const app = express()
5
6app.use(bodyParser.urlencoded({ extended: false }))
7app.listen(80)
8app.get('/', (req, res) => {
9    createReadStream('index.html').pipe(res)
10})
11app.get('/api/date', (req, res) => {
12    res.send({ date: Date.now() })
13})
14app.get('/api/datecors', (req, res) => {
15    res.set('Access-Control-Allow-Origin', '*')
16    res.send({ date: Date.now() })
17})
18app.get('/api/jsonpdate', (req, res) => {
19    res.send(req.query.callback + '({ "date": ' + Date.now() + ' })')
20})
```

3、在 `web.cybersecurity.seu.edu` 下实现页面通过 js 代码读取 `time.cybersecurity.seu.edu` 的接口数据



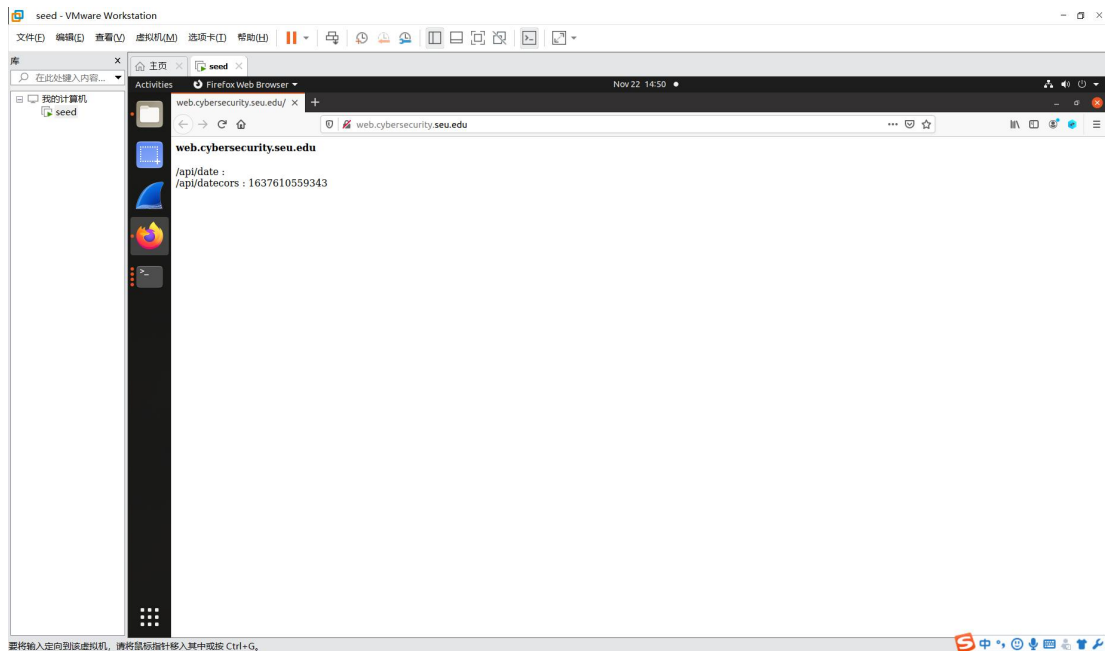
The screenshot shows a VMware Workstation window with a text editor open. The text editor contains an HTML file with the following code:

```
<html>
<body>

<h4>web.cybersecurity.seu.edu</h4>
<div id="date">/api/date : </div>
<div id="datecors">/api/datecors : </div>
<script type="text/javascript">
    const divdatecors = document.querySelector('#datecors')
    const rescors = fetch('http://time.cybersecurity.seu.edu/api/datecors').then(res => res.json()).then(data => {
        divdatecors.textContent = '/api/datecors : ' + data.date
    })
    const divdate = document.querySelector('#date')
    const resdate = fetch('http://time.cybersecurity.seu.edu/api/date').then(res => res.json()).then(data => {
        divdate.textContent = '/api/date : ' + data.date
    })
</script>
</body>
</html>
```

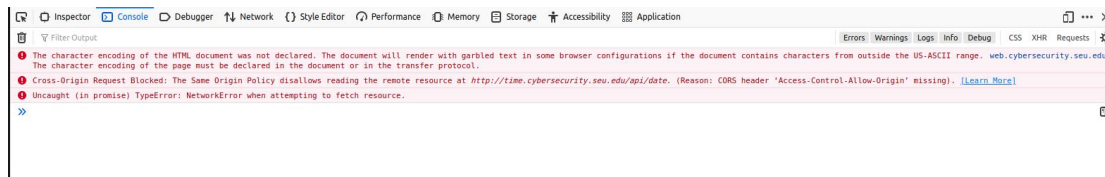
在 `time.cybersecurity.seu.edu` 设置 CORS 头的情况

`web.cybersecurity.seu.edu` 读取 `time.cybersecurity.seu.edu` 的接口数据成功

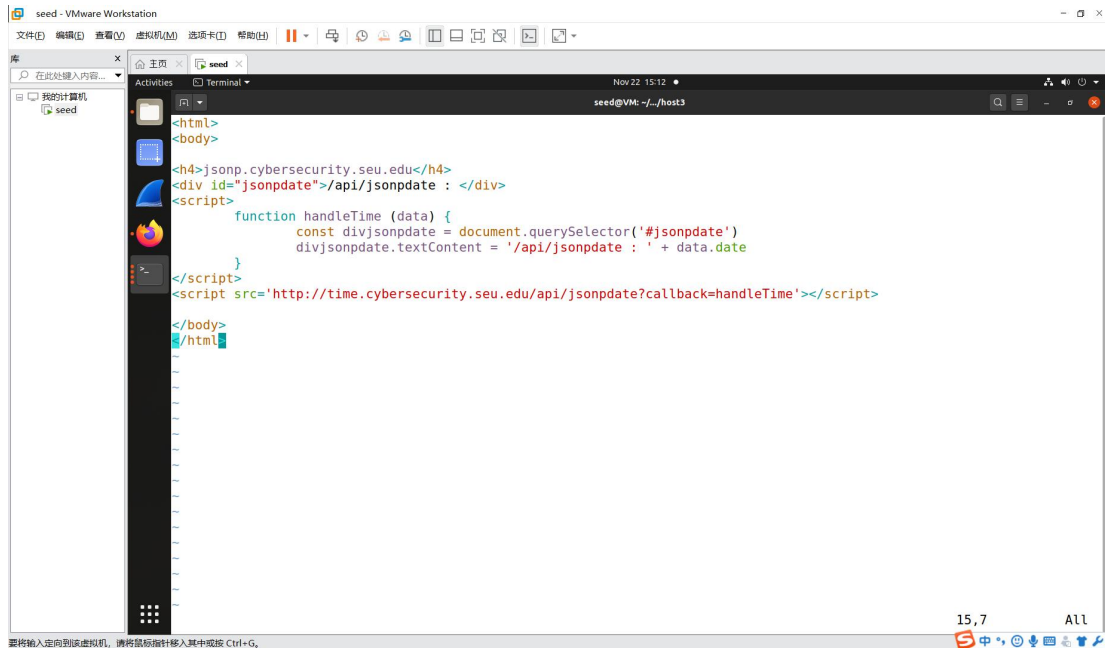


在 time.cybersecurity.seu.edu 未设置 CORS 头的情况

web.cybersecurity.seu.edu 读取 time.cybersecurity.seu.edu 的接口数据失败



4、在 jsonp.cybersecurity.seu.edu 下实现页面通过回调 js 代码读取 time.cybersecurity.seu.edu 的接口数据



在 time.cybersecurity.seu.edu 未设置 CORS 头的情况下

jsonp.cybersecurity.seu.edu 读取 time.cybersecurity.seu.edu 的接口数据成功

