Node.js 目录穿越漏洞(CVE-2017-14849)

> 第一次见这个漏洞是在一次比赛中，但是只是利用了，没有分析，下面就是具体的分析原理。

漏洞环境p师傅在github上制作好了。CVE-2017-14849

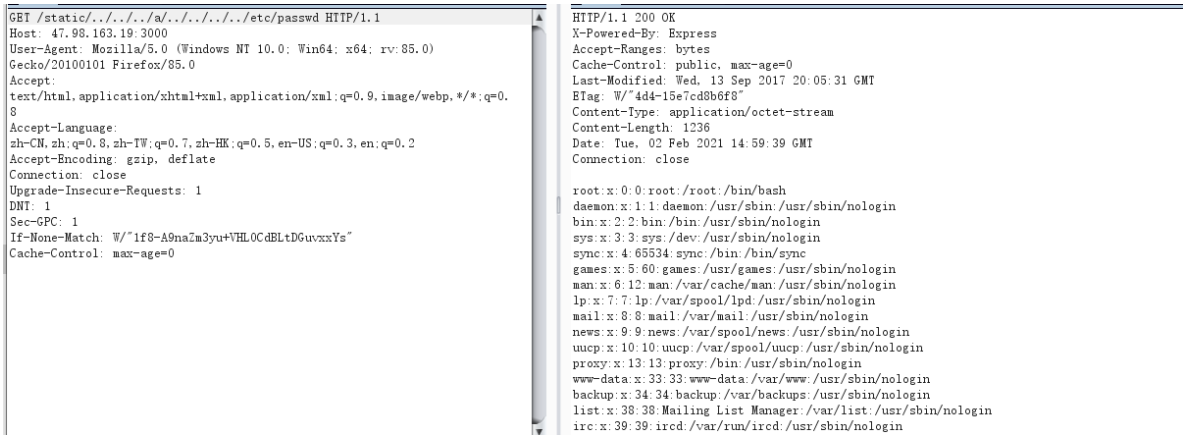
漏洞影响的版本：

- Node.js 8.5.0 & Express 3.19.0-3.21.2
- Node.js 8.5.0 & Express 4.11.0-4.15.5

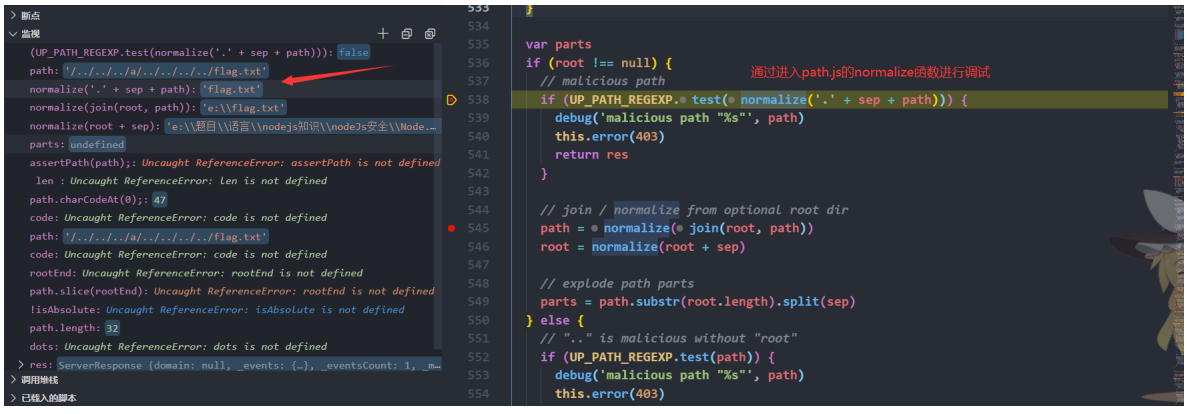
在linux系统中进入漏洞环境目录

```
cd vulhub/node/CVE-2017-14849/
docker-compose build
docker-compose up -d
```

提醒一下如果没有开放端口，可以进docker-compose.yml进行修改端口




```
normalizeStringWin32//去除. ..
```



注意：该漏洞是建立在文件夹通过express.static来托管的情况下，因为在这种情况下才会使用normalize函数进行path标准化。（发现来源于p神的分析）

比如代码这样写：

```
app.use(express.static(path.join(__dirname, 'static')));
```

那么payload应该是

```
/../../../a/../../../../etc/passwd
```

但代码如果是这样写的话：

```
app.use('/static',express.static(path.join(__dirname, 'static')));
```

那么payload应该为：

```
/static/../../../a/../../../../etc/passwd
```

Send模块通过 `normalize('.' + sep + path)` 标准化路径path后，并没有赋值给path，而是仅仅判断了下是否存在目录跳转字符。如果我们能绕过目录跳转字符的判断，就能把目录跳转字符带入545行的 `join(root, path)` 函数中，跳转到我们想要跳转到的目录中，这是Send模块的一个bug，目前已经修复。

再来看Node.js，Node.js 8.5.0对path.js文件中的 `normalizeStringPosix` 函数进行了修改，使其能够对路径做到如下的标准化：

```
1. assert.strictEqual(path.posix.normalize('bar/foo../..'), 'bar');
```

新的修改带来了问题，通过单步调试我们发现，可以通过 `foo../../` 和目录跳转字符一起注入到路径中， `foo../../` 可以把变量 `isAboveRoot` 设置为 `false` （代码161行），并且在代码135行把自己删掉；变量 `isAboveRoot` 为 `false` 的情况下，可以在 `foo../../` 两边设置同样数量的跳转字符，让他们同样在代码135行把自己删除，这样就可以构造出一个带有跳转字符，但是通过 `normalizeStringPosix` 函数标准化后又会全部自动移除的payload，这个payload配合上面提到的Send模块bug就能够成功的返回一个我们想要的物理路径，最后在Send模块中读取并返回文件。 `normalizeStringPosix` 函数如下图：

```
1  function normalizeStringPosix(path, allowAboveRoot) {
2    var res = '';
3    var lastSlash = -1;
4    var dots = 0;
5    var code;
6    var isAboveRoot = false;
7    for (var i = 0; i <= path.length; ++i) {
8      if (i < path.length)
9        code = path.charCodeAt(i);
10     else if (code === 47/*/*/)
11       break;
12     else
13       code = 47/*/*/;
14     if (code === 47/*/*/) {
15       if (lastSlash === i - 1 || dots === 1) {
16         // NOOP
17       } else if (lastSlash !== i - 1 && dots === 2) {
18         if (res.length < 2 || !isAboveRoot ||
19             res.charCodeAt(res.length - 1) !== 46/*.*/ ||
20             res.charCodeAt(res.length - 2) !== 46/*.*/) {
21           if (res.length > 2) {
22             const start = res.length - 1;
23             var j = start;
```

```
            for (; j >= 0; --j) {
              if (res.charCodeAt(j) === 47/*/*/)
                break;
            }
            if (j !== start) {
              if (j === -1)
                res = '';
              else
                res = res.slice(0, j);
              lastSlash = i;
              dots = 0;
              isAboveRoot = false;
              continue;
            }
          } else if (res.length === 2 || res.length === 1) {
            res = '';
            lastSlash = i;
            dots = 0;
            isAboveRoot = false;
            continue;
          }
        }
        if (allowAboveRoot) {
          if (res.length > 0)
            res += '/..';
          else
            res = '..';
          isAboveRoot = true;
        }
      } else {
        if (res.length > 0)
          res += '/' + path.slice(lastSlash + 1, i);
        else
          res = path.slice(lastSlash + 1, i);
        isAboveRoot = false;
      }
      lastSlash = i;
      dots = 0;
    } else if (code === 46/*.*/ && dots !== -1) {
      ++dots;
    } else {
      dots = -1;
    }
  }
  return res;
}
```

自己不知道说什么，反正提示了4个小时也是一知半解~ tcl

> Node.js CVE-2017-14849 漏洞分析