# Mysql字符编码利用技巧

<https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html>

## [0x00 Mysql字符集&字符编码介绍：](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html" \l "0x01-ctf)

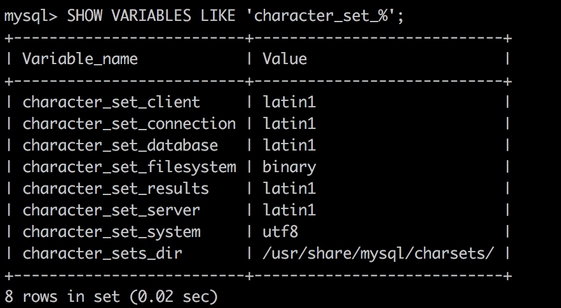
字符集是多个字符的集合，字符集种类较多，这里的字符可以是英文字符，汉字字符，或者其他国家语言字符。  
 常见字符集包括：ASCII字符集、LATIN1字符集、GB2312字符集、GBK字符集、GB18030字符集、Unicode字符集等。

对于mysql数据库系统而言，每个环节都涉及到字符存储。涉及到字符存储的地方，就涉及到字符集编码。

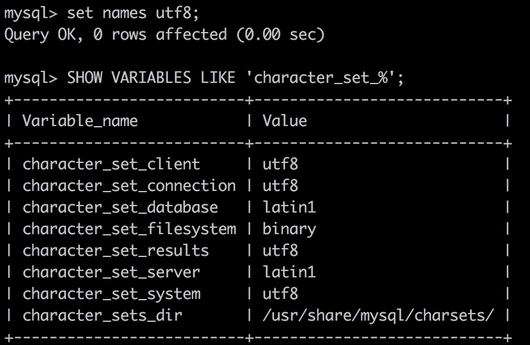
Mysql有客户端字符集和内部字符集：

客户端字符集：

打开mysql控制台，默认客户端字符集是latin1:



执行set names utf8，SHOW VARIABLES LIKE 'character\_set\_%'，即可得到如下结果：



Mysql内部字符集：

数据库中字段的字符集，比如某个表中username、password的字符集。

2008年鸟哥曾在[博客](http://www.laruence.com/2008/01/05/12.html" \t "_blank)中讲解了Mysql字符集：

1.MySQL Server收到请求时将请求数据从character\_set\_client转换为character\_set\_connection；

2.进行内部操作前将请求数据从character\_set\_connection转换为内部操作字符集。

当character\_set\_connection转换成内部操作字符集，二者字符集编码不同时，就会出现异常。

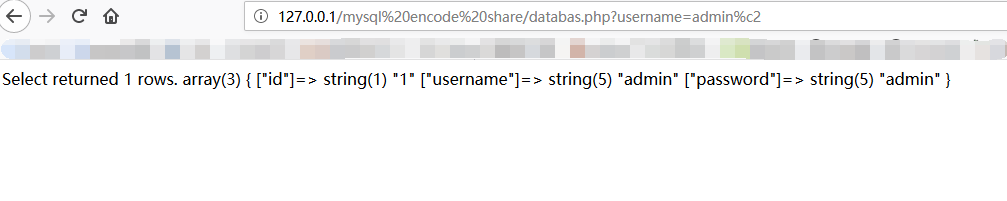
## [0x01 由某CTF题解说起](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x01-ctf)

大概代码如下：



$\_SERVER['REMOTE\_ADDR']:获取用户IP

这个if语句嫌疑很大，大概是考我们怎么登陆admin的账号，直接admin查询失败；admin%c2查询成功：



可见，传入的username=admin%c2，可以绕过php的检测if ($username === 'admin')，在mysql中可以正常查询。

## [0x02 Trick复现](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x02-trick)

那么，为什么执行SELECT \* FROM user WHERE username='admin%C2' and password='admin'却可以查出用户名是admin的记录？搭建环境测试下：

在数据库cat中创建表table1：

|  |
| --- |
| CREATE TABLE `table1` (  `id` int(10) unsigned NOT NULL AUTO\_INCREMENT,  `username` varchar(255) COLLATE **latin1**\_general\_ci NOT NULL,  `password` varchar(255) COLLATE **latin1**\_general\_ci NOT NULL,  PRIMARY KEY (`id`)  ) ENGINE=MyISAM AUTO\_INCREMENT=1 DEFAULT CHARSET=latin1 COLLATE=latin1\_general\_ci; |

**我特地将内部字符集设置为latin1，其实默认情况下，Mysql的字符集就是latin1。**

添加一个管理员账户：

|  |
| --- |
| INSERT `table1` VALUES (1, 'admin', 'admin'); |

编写如下代码databas.php：

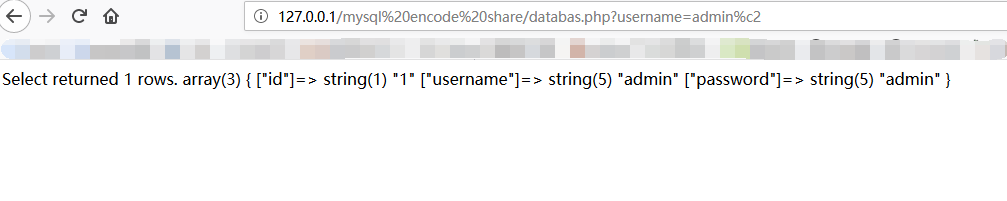
|  |
| --- |
| <?php  $mysqli = new mysqli("localhost", "root", "123456", "cat");  /\* check connection \*/  if ($mysqli->connect\_errno) {  printf("Connect failed: %s\n", $mysqli->connect\_error);  exit();  }  $mysqli->query("**set names utf8**");  $username = addslashes(**$\_GET[]**);  /\* Select queries return a resultset \*/**'username'**  $sql = **"SELECT \* FROM `table1` WHERE username='{$username}'";**  if ($result = $mysqli->query( $sql )) {  printf("Select returned %d rows.\n", $result->num\_rows);  while ($row = $result->fetch\_array(MYSQLI\_ASSOC))  {  var\_dump($row);  }  /\* free result set \*/  $result->close();  } else {  var\_dump($mysqli->error);  }  $mysqli->close(); |

**同时创建表table2，与table1设置相同的字段，添加相同的数据，唯一不同的是将username、password字符集设置为utf8：**



然后，访问databas.php:

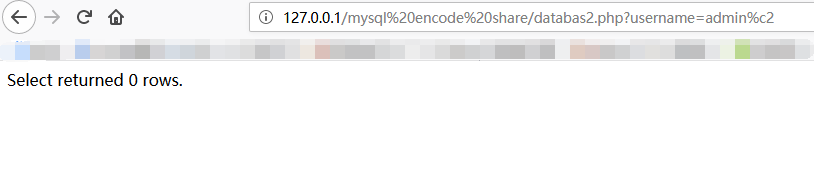
http://127.0.0.1/mysql encode share/databas.php?username=admin**%c2**，即可发现%c2被忽略，Mysql查出了username=admin的结果：



将databas.php中的table1改为table2，另存文件为：databas2.php，同样的方法访问：

127.0.0.1/mysql encode share/databas2.php?username=admin**%c2**

发现未查出结果：



## [0x03 Mysql字符集转换](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x03-mysql)

**造成这个结果的原因是，Mysql字段的字符集（Latin1）和php mysqli客户端设置的字符集（utf8）不相同**。

在我们这个案例中，character\_set\_client和character\_set\_connection被设置成了utf8，而内部操作字符集其实也就是username字段的字符集是latin1。于是，整个操作就有如下字符串转换过程：

utf8 --> utf8 --> latin1

最后执行比较username='admin'的时候，'admin'是一个latin1字符串。

## [0x04 漏洞成因](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x04)

那么，字符集转换为什么会导致%c2被忽略呢？

说一下我的想法，虽然我没有深入研究，但我觉得原因应该是，**Mysql在转换字符集的时候，将不完整的字符给忽略了**。

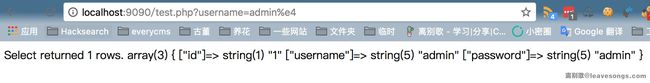
举个简单的例子，**佬**这个汉字的UTF-8编码是\xE4\xBD\xAC，我们可以依次尝试访问下面三个URL：

http://localhost:9090/test.php?username=admin**%e4**

http://localhost:9090/test.php?username=admin**%e4%bd**

http://localhost:9090/test.php?username=admin**%e4%bd%ac**

可以发现，前两者都能成功获取到username=admin的结果，而最后一个URL，也就是当我输入佬字完整的编码时，将会被抛出一个错误：

[](https://www.leavesongs.com/media/attachment/2017/04/09/137a7289-ea01-4d32-b11e-638f25c8ee8c.jpg)

[](https://www.leavesongs.com/media/attachment/2017/04/09/5d50006b-1fb5-435d-928b-9e8652d84025.jpg)

[](https://www.leavesongs.com/media/attachment/2017/04/09/4015771c-f706-4425-ae44-cc19a0c6dbc0.jpg)

为什么会抛出错误？原因很简单，因为**latin1并不支持汉字**，所以utf8汉字转换成latin1时就抛出了错误。

Insert into table1 values(2,’张三’,123); 张三成为乱码：

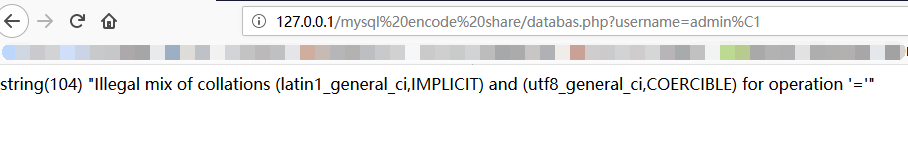


那前两次为什么没有抛出错误？因为前两次输入的编码并不完整，Mysql在进行编码转换时，就将其忽略了。

这个特点也导致，我们查询username=admin%e4时，%e4被省略，最后查出了username=admin的结果。

## [0x05 为什么只有部分字符可以使用](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x05)

试下别的字符可以不，%c1，但username=admin%c1就不行，这是为什么？



我简单fuzz了一下，如果在admin后面加上一个字符，有如下结果：

1. \x00~\x7F： 返回空白结果
2. \x80~\xC1： 返回错误Illegal mix of collations
3. \xC2~\xEF： 返回admin的结果
4. \xF0~\xFF： 返回错误Illegal mix of collations

关于所有的UTF-8字符，你可以在这个表中一一看到：[http://utf8-chartable.de/unicode-utf8-table.pl](http://utf8-chartable.de/unicode-utf8-table.pl" \t "_blank)

UTF-8编码是变长编码，可能有1~4个字节表示：

1. 一字节时范围是[00-7F]
2. 两字节时范围是[C0-DF][80-BF]
3. 三字节时范围是[E0-EF][80-BF][80-BF]
4. 四字节时范围是[F0-F7][80-BF][80-BF][80-BF]

然后根据RFC 3629规范，又有一些字节值是不允许出现在UTF-8编码中的：

[](https://www.leavesongs.com/media/attachment/2017/04/09/db28c7b4-4dc9-4592-9fc7-23f0290c3892.jpg)

所以，UTF-8第一字节的取值范围是：00-7F、C2-F4，

|  |  |
| --- | --- |
| UTF-8第一字节 | 规范后UTF-8第一字节 |
| 00-7F | 00-7F |
| C0-DF | C2-DF |
| E0-EF | E0-EF |
| F0-F7 | F0-F4 |

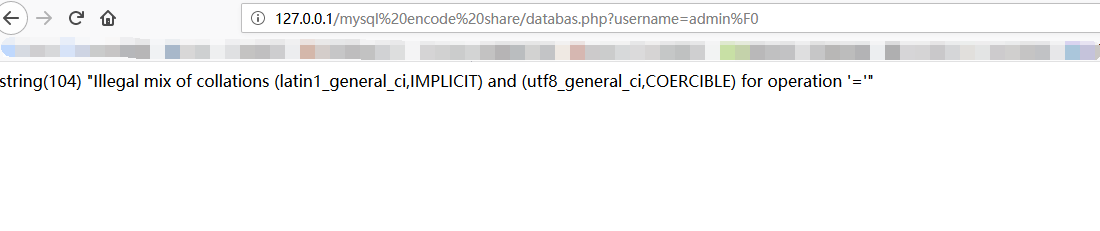
DF衔接E0 ，EF衔接F0，所以C2-DF，E0-EF，F0-F4合起来是C2-F4.

这也是我在admin后面加上80-C1、F5-FF等字符时会抛出错误的原因。

此处得到的适用范围是： 00-7F、C2-F4

## [0x06 Mysql UTF8 特性](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x06-mysql-utf8)

再试下admin%F0，查询失败，为什么username=admin%F0失败了？F0明明在C2-F4的范围中呀？



这又涉及到Mysql中另一个特性：**Mysql的utf8其实是阉割版utf-8编码，Mysql中的utf8字符集最长只支持三个字节**，

所以，我们回看前文列出的UTF-8编码第一字节的范围，

三字节时范围是**[E0-EF]**[80-BF][80-BF]  
四字节时范围是**[F0-F7]**[80-BF][80-BF][80-BF]

F0-F4是四字节才有的，所以我传入username=admin%F0也将抛出错误。

所以，最终，MySQL中UTF-8第一字节的取值范围是：00-7F、C2-EF。

## [0x07 总结](https://www.leavesongs.com/PENETRATION/mysql-charset-trick.html#0x07)

本文深入研究了Mysql编码的数个特性，相信看完本文，对于第一章中的CTF题目也没有疑问了。

通过这次研究，学习到一些知识点：

1. $\_SERVER['REMOTE\_ADDR']:获取用户IP

2.默认情况下，Mysql的字符集是latin1

3. Mysql字符集：

A.MySQL Server收到请求时将请求数据从character\_set\_client转换为character\_set\_connection；

B.进行内部操作前将请求数据从character\_set\_connection转换为内部操作字符集。

4. Mysql在转换字符集的时候，将不完整的字符忽略。

5. Mysql的utf8其实是阉割版utf-8编码，Mysql中的utf8字符集最长只支持三个字节。