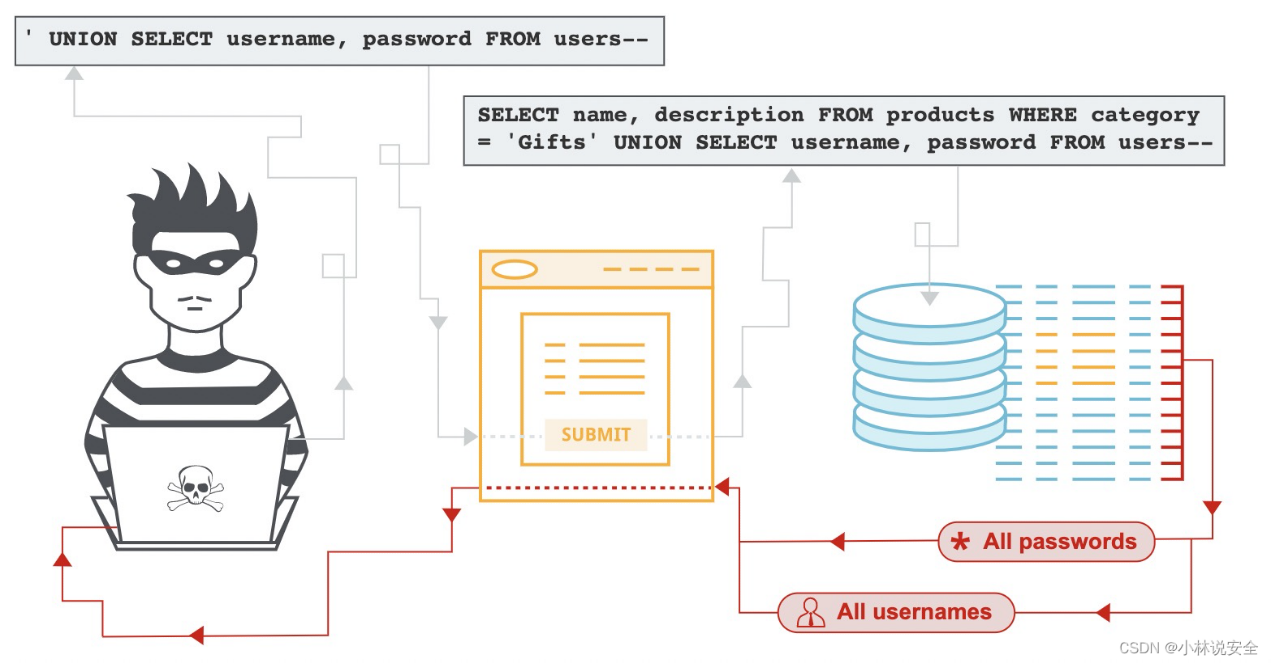
Web攻防-SQL注入&数据类型&参数格式&JSON&XML&编码加密&符号闭合&复盘报告



#知识点：  
1、Web攻防-SQL注入-产生原理&应用因素  
2、Web攻防-SQL注入-各类数据库类型利用  
----------------------------------------------  
1、Web攻防-SQL注入-参数类型\*参数格式  
2、Web攻防-SQL注入-XML&JSON&BASE64等  
3、Web攻防-SQL注入-数字字符搜索等符号绕过  
   
#章节点：(待补充)  
OWTOP10，SQL注入，文件安全（上传,下载,读取,删除,包含等），目录遍历，  
CSRF，SSRF，CRLF，SSTI注入，XML&XXE安全，RCE执行（代码或命令等）  
反序列化，业务逻辑（验证码，接口枚举，支付购买，机制验证，越权，并发等）  
未授权访问，失效访问控制，弱口令安全，第三方组件安全，CORS，JSONP等





演示案例：

* WEB攻防-SQL注入-数据库类型利用
* WEB攻防-SQL注入-参数类型&符号干扰
* WEB攻防-SQL注入-参数格式&参数编码

#数据库知识：  
1、数据库名，表名，列名，数据  
2、自带数据库，数据库用户及权限  
3、数据库敏感函数，默认端口及应用  
4、数据库查询方法（增加删除修改更新）  
   
#SQL注入产生原理：  
代码中执行的SQL语句存在可控变量导致  
   
#影响SQL注入的主要因素：  
1、数据库类型（权限操作）  
2、数据操作方法（增删改查）  
3、参数数据类型（符号干扰）  
4、参数数据格式（加密编码等）  
5、提交数据方式（数据包部分）  
6、有无数据处理（无回显逻辑等）  
   
#常见SQL注入的利用过程：  
1、判断数据库类型  
2、判断参数类型及格式  
3、判断数据格式及提交  
4、判断数据回显及防护  
5、获取数据库名，表名，列名  
5、获取对应数据及尝试其他利用  
   
#黑盒/白盒如何发现SQL注入  
1、盲对所有参数进行测试  
2、整合功能点脑补进行测试  
白盒参考后期代码审计课程  
   
利用过程：  
获取数据库名->表名->列名->数据（一般是关键数据，如管理员）  
   
案例说明：  
在应用中，存在参数值为数字，字符时，符号的介入，另外搜索功能通配符的再次介入，另外传输数据可由最基本的对应赋值传递改为更加智能的XML或JSON格式传递，部分保证更安全的情况还会采用编码或加密形式传递数据，给于安全测试过程中更大的挑战和难度。  
   
#数字，字符，搜索  
例：  
select \* from news where id=$id;  
select \* from news where name='$name';  
select \* from news where name like '%name%';  
符号干扰：有无单引号或双引号及通配符等  
order by 3  
union select 1,2,3  
知道数据库名：database() news\_db  
获取数据库名下的表名信息：  
借助自带的information\_schema.tables表（记录所有数据库名下的表名）  
1%' union select 1,2,table\_name from information\_schema.tables where table\_schema='news\_db'#  
admin下列名  
借助自带的information\_schema.columns表（记录所有数据库名下的表名对应的列名信息）  
1%' union select 1,2,column\_name from information\_schema.columns where table\_schema='news\_db' and table\_name='admin'#  
news\_db  
 admin  
 username,password  
1%' union select 1,2,password from admin#  
   
#XML，JSON，编码，混合  
XML:  
<?xml version="1.0" encoding="UTF-8"?>  
<news>  
 <article>  
 <id>1</id>  
 <title>xiaodi</title>  
 <content>i am xiaodi</content>  
 <created\_at>2025-03-07</created\_at>  
 </article>  
 <article>  
 <id>2</id>  
 <title>xiaodisec</title>  
 <content>i am xiaodisec</content>  
 <created\_at>2025-03-06</created\_at>  
 </article>  
</news>  
   
JSON:  
{  
 "news:"[  
 {  
 "id": 1,  
 "title": "xiaodi",  
 "content": "i am xiaodi",  
 "created\_at": "2025-03-07"  
 },  
 {  
 "id": 2,  
 "title": "xiaodisec",  
 "content": "i am xiaodisec",  
 "created\_at": "2025-03-06"  
 }  
 ]  
}  
   
Base64:  
{  
 "news": [  
 {  
 "id": "MQ==",  
 "title": "eGlhb2Rp",  
 "content": "aSBhbSB4aWFvZGk=",  
 "created\_at": "MjAyNS0wMy0wNw=="  
 },  
 {  
 "id": "Mg==",  
 "title": "eGlhb2Rpc2Vj",  
 "content": "aSBhbSB4aWFvZGlzZWM=",  
 "created\_at": "MjAyNS0wMy0wNg=="  
 }  
 ]  
}  
   
1、数据传输采用XML或JSON格式传递  
2、数据传输采用编码或加密形式传递  
3、数据传递采用JSON又采用编码传递  
   
#实例应用：  
1、JSON注入案例：  
SRC报告-众测下的SQL注入挖掘  
SRC报告-edu-SQL注入案例分享  
SRC报告-河南省xxxx某站存在SQL注入漏洞  
2、编码注入案例：  
互联网搜下对应说明  
<https://mp.weixin.qq.com/s/Xf08xaV-YcZsQopE19pPEQ>

涉及资源：[资源下载地址](https://docs.qq.com/doc/DQ3Z6RkNpaUtMcEFr)